The 44th IEEE Real-Time Systems Symposium

RTSS 2023

Author Response

Title: Hierarchical MARL: An Energy and Thermal-Aware Scheduler for OpenMP DAG Workloads
Authors: Mohammad Pivezhandi, Abusayeed Saifullah and Ali Jannesari

Instructions

The author response period has begun. The reviews for your submission are displayed on this page. If you want to respond to the points raised in the reviews, you may do so in the boxes provided below. The word limit of your response is 500.

Please note: *you are not obligated to respond to the reviews*.

For reference, you may see the review form that reviewers used to evaluate your submission. If you do not see some of the filled-in fields in the reviews below, it means that they were intended to be seen only by the committee. See the review form HERE.

Review #1

Overall score (1-6):	3
Reviewer's Confidence (1-4):	1
Contributions (1-4):	2
Applicability to real-time systems (1-4):	3
Correctness/Credibility (1-3):	2
Presentation (1-4):	3

Detailed Comments

- Short summary:

The paper proposes an online reinforcement learning approach for energy aware efficient task allocation on multi-core processors. The proposed method uses a cooperative hierarchical multi-agent reinforcement learning (MARL) and a non-policy double deep Q-network (DDDQN) algorithm. The objective is to minimize the makespan of a Directed Acyclic Graph (DAG) workload while optimizing energy consumption.

The authors present experiments conducted on Intel Xeon 12-core and Intel Core i7 4-core processors. They compare their approach against all Linux governors and multi-speed scheduler on the basis of energy consumption, execution time and average temperature. The results showed that the proposed approach showed promising results with a 26% reduction in energy consumption and a 16% improvement in execution time as compared to the best Linux governor. the average temperature was also observed to have been reduced by more than 3 degrees centigrade.

- Points in favor (bullet list):

-Well written paper -Highly relevant topic and greatly useful for industry applications -Experiments show significant improvement over state-of-art

- Points against (bullet list):

-Application to industry not discussed -Limitation of the approach not discussed

- Major comments supporting the "Overall score" rating:

-This paper addresses a highly relevant and significant issue for industry applications including automotive industry where problems related to overheating are becoming more and more pertinent. As such solutions for such issues are needed. However, there is not much discussion around how this approach can be applied for industrial use case or if there is any such future work is planned. It is recommended to provide details of application to industry. Additionally, the discussion on the limitations of the approach are not discussed, hence it is recommended to the authors to include limitations of the approach as well as any constraints. For example is this approach applicable to real-time operating systems and to embedded systems as well?

- Other comments for the author(s):

Review #2

Overall score (1-6):	3
Reviewer's Confidence (1-4):	3
Contributions (1-4):	3
Applicability to real-time systems (1-4):	2
Correctness/Credibility (1-3):	2
Presentation (1-4):	2

Detailed Comments

- Short summary:

This paper proposes a hierarchical multi-agent reinforcement learning (MARL) approach with double-deep dueling Q-networks (DDDQN) for energy and thermal-aware scheduling of OpenMP DAG workloads in homogeneous multi-core processors. The method uses cooperative hierarchical MARL, where one agent selects cores and frequency scaling as a function of profiler results, and other selects core combinations as a function of temperature sensor results. Their proposed algorithm incorporates profiling data and employs attention networks to assign weights to different factors such as makespan, energy consumption, and core temperatures, allowing for flexible adjustment of their relative importance during online training. The proposed online learning algorithm demonstrates low computational complexity when allocating resources to DAG benchmarks while ensuring thermal feasibility by accounting for per-core temperature constraints in the targeted processor. They have evaluated their approach on Intel Xeon 2680 v3 and Core I7 platform with OpenMP implemented workloads and show that the proposed approach outperforms existing heuristic-based schedulers and Linux governors in terms of energy consumption, makespan, and temperature metrics.

- Points in favor (bullet list):

```
+ The problem selected by the authors of the paper is relevant in the context of designing efficient ene
    in homogeneous multi-core processors for irregular workloads in terms of unpredictable task executio
+ The proposed work performs better than Linux built-in DVFS governors in terms of energy consumption, m
+ Results are not simulation based. They have used real platforms with processors such as Intel Xeon 268
   implemented DAG applications as workload in their experiments.
```

- Points against (bullet list):
- Lack of real-time analysis. The work does not consider real time task deadlines. There is no mechanism provided to take into account real-time constraints, no penalty on deadline violation during training.
- Evaluation is done on homogeneous desktop systems. No evaluation on realtime embedded platforms, neither on heterogeneous platforms. - The benchmarks used as workload are not described in detail in the paper.
- The overhead analysis of the profiling, training and inferencing phase is not provided in detail. It is not clear what is the total overhead of the framework compared to existing approaches.
- The work misses out on relevant discussions and descriptions in the context of experiment details, assumptions considered and explanation of observed results. This in turn reduced the correctness and credibility of the work. In general, the flow of the paper in some places is difficult to follow due to lack of sufficient information and explanation. I have listed a few in the major comments.
- Major comments supporting the "Overall score" rating:
- In section III A, in Figure 2, the size of the nqueens benchmark considered is not mentioned. The description of the target architecture is not given. Assuming it is tested on 12-core Intel Xeon 2680 v3, why the result of 11 cores is considered is not explained at the place. It is not clear why the temperature difference is comparatively higher for omp-tasks-manual. In section III B, the explored scenarios and configurations in multi-core processor execution, the experiment setup and workload are not described properly.
- In section IV on system model, the task model does not consider any deadline. It is not clarified, what is the type of tasks considered here. Are they periodic or aperiodic or sporadic in nature? Are there any other background workloads executing while evaluating the performance of a single workload? The power model and temperature models are described but not used anywhere in the rest of the paper. The initial sections motivate use of RL in heterogeneous systems with P and E cores etc. It is also not clear why details on heterogeneous architecture of Intel Core i7 12th generation processor are provided even though the proposed method is for homogeneous multi-core systems. None of the experiments are evaluated on Intel Core i7 12th generation.
- In section V, a lot of the details provided are standard knowledge on Reinforcement Learning. However, the overall training and inferencing processes used for their target problem are not properly mentioned. It is mentioned that their proposed online learning based scheduler adjusts resources by selecting suitable core combinations and frequency settings. However, no information is given on how exploration vs exploitation during training is handled. During initial training phase, while exploring different actions may lead to sub-optimal resource allocation and potential device damage due to high temperatures and deadline misses. It is not clear what are workloads used for training and testing, what is the description of the networks used in the DDDQN. On which core is the computation in RL agent happening and how it's interference with the executing workload is handled? In the reward calculation, what are the final values of trained weights used in the inference ? These are not provided. Are these values specific to each workload, or are they generic for any workload ? The values considered for the hyper-parameters used in their method are not mentioned as well. What is the predefined counter threshold considered for epochs and episode termination condition.
- In section VI, the evaluation section is not well written and lacks adequate description and explanation. No details of the 43 OpenMP tasks considered for evaluation are provided. It is not clear, out of these 43 tasks which of them are considered for training and testing. Figure 9 is difficult to understand due to lack of sufficient description of the plot. The number of episodes considered is quite low compared to standard RL training sessions. The important results in Figure 11 are not described and explained. It is not clear why makespan for learned policy is more for 250 episodes compared to 200 episodes, even though CPU utilization is more. There is no explanation why the average temperature is more for performance governor and learned policy at 200 episodes. The average temperature in the figure is mentioned in ^degree F which is incorrect. It will be interesting to compare the branch misses count and other state variable counts along with the makespan comparison. What is the considered workload whose execution time is given in Table 1. How many branches and threads are considered in that workload?
- Other comments for the author(s):

```
+ Some minor grammatical errors in the paper:
```

- "per-core speed adjustment remain unaddressed" -> "per-core speed adjustment remains unaddressed".
- "The total number of parameters required for this network can be found in 3." -> "The total number of parameters required for this network can be found in Lemma 3."

Review #3

Overall score (1-6):	1
Reviewer's Confidence (1-4):	3
Contributions (1-4):	2
Applicability to real-time systems (1-4):	3
Correctness/Credibility (1-3):	1
Presentation (1-4):	1

Detailed Comments

- Short summary:

This paper uses RL for Energy and Thermal-Aware Scheduling of multicore processors.

- Points in favor (bullet list):

- Points against (bullet list):

You stated "we introduce separate attention networks, namely the Profiler Reward Attention and Temperature Reward Attention" in "Fig. 7: Training attention-based network to weight input observations for each agent as reward values", You train attention networks to determine the weights in the TL reward function. Your approach is an elementary mistake that violates the basic principles of RL. The RL reward function must be pre-defined. You CANNOT train the reward function simultaneously with the RL agent. The references you cited [15][46] describe the attention mechanism in general, and do not support your approach. Therefore, I would like to ask that you share your source code to verify your experimental results.

"Algorithm 1 Proposed Hierarchical Q-learning RL" is described unclearly. Please continue to use Example 1 to illustrate it, esp. how is the action space of 285 divided among the multiple RL agents, how many RL agents? Algorithm 1 seems to be a regular RL algorithm. It does not fit the standard definition of hierarchical RL/MARL. Hierarchical RL is defined as follows, taken from "Hierarchical Reinforcement Learning: A Comprehensive Survey" https://dl.acm.org/doi/abs/10.1145/3453160. "Hierarchical Reinforcement Learning (HRL) decomposes a long-horizon reinforcement learning task into a hierarchy of subproblems or subtasks such that a higher-level policy learns to perform the task by choosing optimal subtasks as the higher-level actions. A subtask may itself be a reinforcement learning problem with a lower-level policy learning to solve it [39]. This hierarchy of policies collectively determines the behavior of the agent. Task decomposition effectively reduces the original task's long horizon into a shorter horizon in terms of the sequences of subtasks. This is because each subtask is a higher-level action that persists for a longer timescale compared to a lower-level action, a property that is often referred to as temporal abstraction." In Section "3.5 Multi-agent Hierarchical Reinforcement Learning (MAHRL)" of the survey paper: "Going beyond standard MARL, the benefits of task decomposition can also be exploited for multi-agent coordination by dividing a joint task into multiple subtasks distributed across different HRL agents. Each agent can perform more than one subtask and different agents learn to coordinate their higher-level policies (that choose the subtasks) rather than just the primitive action policies. In certain cases, the agents may only coordinate at the higher level while treating their primitive action policies as independent of each other [58]. Learning to coordinate among HRL agents is called Multi-agent HRL (MAHRL) [30]."

The term "double-deep dueling Q-networks (DDDQN)" needs a reference. I think the proper term is "dueling double-deep Q-networks (DDDQN)". The term "non-policy" appears several times, but the proper term is "off-policy". I suspect this mistake resulted from automatic translation from a foreign language. Section "A. Why Non-Policy DDDQN Reinforcement Learning?" describes generic background on Q learning in too much detail, and is not necessary in this paper on applications of Q-Learning.

This paper lacks adequate discussions of related work and comparisons to State-of-the-Art. RL-based multicore scheduling is a well-studied topic, c.f., this survey paper from 2018 and its many citations: Pagani S, Manoj P D S, Jantsch A, et al. Machine learning for power, energy, and thermal management on multicore processors: A survey[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2018, 39(1): 101-116. However you do not cite or compare with any related works based on RL. You stated "The proposed approach is compared with a multi-speed Federated clustered energy-aware scheduler similar to [6]." But what do you mean by "similar to"? Why do you choose this one reference among so many related works? Please explain [6] in detail in "II. RELATED WORK".
[6] S. Maity, R. Roy, A. Majumder, S. Dey, and A. R. Hota, "Future aware dynamic thermal management in cpu-gpu embedded platforms," in 2022 IEEE Real-Time Systems Symposium (RTSS). IEEE, 2022, pp. 396–408.

In Lemma 1: "#As a result, the total number of actions can be approximated by m^k." How do you obtain m^k from the sum formula of a_T? For Example 1, you obtain the total number of action combinations for three cores and ten frequency levels to be 285, but m^k=3^10=59049. In Lemma 2, "Lemma 2. The exponential n^k", what is n?

- Major comments supporting the "Overall score" rating:

The approach of training the reward function definition together with the RL agent is simply wrong. The term "Hierarchical MARL" is a misnomer, since the proposed RL algorithm does not fit the standard definition in the ML literature. It lacks proper comparisons to other works on RL-based multicore scheduling.

- Other comments for the author(s):

(Rebuttal) Questions for Authors

1. Please compare with RL-based SoTA, c.f., Pagani S, Manoj P D S, Jantsch A, et al. Machine learning for power, energy, and thermal management on multicore processors: A survey[J]. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2018, 39(1): 101-116.
2. Please cite references to support using attention networks to define reward functions. I think this is a mistake.
3. Please share your source code, if not to the public, at least to the reviewers.
4. In Fig. 6, the term "actor" should be "action"? Since "actor" is synonymous as "agent".
5. Please show the typical RL training performance plots, e.g., the increase of cumulative reward with training episodes.
6. Please explain the different terms in the rewards function, esp. "Profiler Reward". It seems you are trying to maximize the difference between the old state value and the new state value, but why are they defined this way, intuitively?

Review #4

Overall score (1-6):	3
Reviewer's Confidence (1-4):	2
Contributions (1-4):	2
Applicability to real-time systems (1-4):	3
Correctness/Credibility (1-3):	2
Presentation (1-4):	3

Detailed Comments

- Short summary:

This paper proposes an online reinforcement learning approach to efficiently allocate tasks based on their profiling data, including the makespan of the tasks, to the appropriate core combinations according to their temperature status to minimize energy consumption.

- Points in favor (bullet list):

1. An effective approach for efficiently allocating tasks based on their profiling data and reducing energy consumption
2. Fine-grained control over individual core frequencies

- Points against (bullet list):

1. The motivation should be improved.
2. The authors merely studied the effect of the unpredictability of the DAG scheduling. However, many factors of DAG scheduling should be discussed.
3. The preciseness of the statements could be improved.
4. The evaluation of the platforms and proposed method can be improved which are listed the detailed comments.

- Major comments supporting the "Overall score" rating:

The paper proposes learning-based fine-grained energy- and thermal-aware scheduling. It is an interesting research problem within the scope of RTSS. However, the motivation should be improved. And the DAG scheduling should be discussed in more detail. Besides, the preciseness of the statements could be improved.

- Other comments for the author(s):

1. The paper applies Q-learning for OpenMP DAG workloads. The Q-learning based DVFS for power/energy-efficient computing is widely studied in the past 5-8 years. It would be appreciated if the author could provide a more comprehensive related work. And the authors should clearly state the improvements from previous works. For example, "Machine Learning for Power, Energy, and Thermal Management on Multicore Processors: A Survey", "Profit: Priority and power/performance optimization for many-core systems", and other related works.
2. This work is based on the motivation that leakage power, which correlates exponentially with chip temperature, becomes more than dynamic power (roughly 22% to over 63%). This motivation is from reference [1], which was published in 2003. But in reference [1], this motivation is cited from another work published in 2002. I would suggest the authors provide a more accurate source of these numbers. In addition, what is the current status?
3. The study of this paper is based on the real-time system. However, there is no introduction or discussion of the real-time concept, such as schedulability. Besides, the authors mainly studied the effect of the unpredictability of the DAG on its execution results (core allocation, scaling voltage, and frequency). However, the authors do not discuss the impact caused by the topology structure of the DAG, which cannot be ignored.
4. Since tied tasks can only be executed on their specified cores, the execution of DAG becomes much more complicated. Hence, it is necessary to discuss the effect caused by the precedence constraint.
5. The word "approximately" should not be used in Lemma 1. As a lemma, its conclusion should be an exact number, an upper bound, or a lower bound.
6. The System Model Section introduces that there is only one DAG in the system. However, in Fig. 4, J0 and J1 seem like two jobs from different DAGs. Please explain this contradiction.
7. If it is possible, I would suggest the authors to measure the static and dynamic power on the evaluated platforms (Intel Xeon 2680 v3 and 4-core Core I7).
8. The power and thermal models in Section III and Section V.A are textbook-level knowledge. It seems that most models in Section III are not utilized in the rest of sections.
9. The authors state the time used for changing processor voltage is a maximum of 20µs. Could the author explain how it is implemented and measured? Will these intervals be included in the model and evaluation?
10. In evaluation, the proposed method should be compared with state-of-the-art DVFS strategies in recent years.

Review #5

Overall score (1-6):	4
Reviewer's Confidence (1-4):	3
Contributions (1-4):	3
Applicability to real-time systems (1-4):	4
Correctness/Credibility (1-3):	2
Presentation (1-4):	3

Detailed Comments

- Short summary: The authors propose a power management strategy that is based on profiling data and takes into account different optimization targets: make span, energy comsumption, and core temperature. The approach (a RL-based energy scheduler) is evaluated against relevant state-of-the-art schedulers. The approach is tailored to OpenMP DAG workloads.
- Points in favor (bullet list):

The approach takes into account critical metrics that have huge impact on energy consumption and temperature that are ignored in previous work: cache pressure, memory traffic, etc.

The paper contains a good comparison that shows the effectiveness of the approach

- Points against (bullet list):

The approach is tailored to a single openMP task which limits the applicability and relevance of the approach.

- Major comments supporting the "Overall score" rating:

I like the overall approach very much since it takes into account very important effects that have been ignored in previous work. The main weak point is that the approach is tailored to openMP tasks. The approach would be much stronger if it was applicable to generic work loads also considering a bunch of independently executed applications.

Also, only one single DAG is considered. This limits the applicability of the approach drastically. Powerful execution platform are rarely used for executing a single openMP application. There must be quite some cores idling in the process of executing this single application.

Overall I tend to accept the paper. However for the final version I would like the experimental section to be strengthened.

- Other comments for the author(s):

(Rebuttal) Questions for Authors

1. How can you extend your approach to more generic work loads
2. What is the utilization of the whole system during execution? more metrics are needed.
3.

Submit Response to Reviewers

Use the following boxes to enter your response to the reviews. Please limit the total amount of words in your comments to 500 words (longer responses will not be accepted by the system).

Response to Review #1:

Response to Review #2:

Response to Review #3:

Response to Review #4:

Response to Review #5:

General Response to Reviewers:

Submit