technische universität dortmund

computer science 12

Master/Bachelor Thesis

Applying Reinforcement Learning on Dependency Graph Generation

M.sc. Junjie Shi
Prof. Dr. Jian-Jia Chen
Otto-Hahn Str. 16
Technische Universität Dortmund
28.July.2020

To prevent race conditions or data corruptions, concurrently accessing the same resource is prohibited by exploiting mutual exclusion. Therefore, many semaphore based resource synchronization protocols have been widely used to realize mutual exclusion and prevent deadlocks and priority inversions. In modern multiprocessor platforms, many resource synchronization protocols have been proposed, e,g., Multiprocessor Priority Ceiling Protocol (MPCP) [4], Distributed Priority Ceiling Protocol (DPCP) [5], Flexible Multiprocessor Locking Protocol (FMLP) [1], and so on.

For several decades, the primary focus when considering multiprocessor synchronization and locking in real-time systems has been the design and analysis of resource sharing protocols, where the protocols decide the order in which the new incoming requests access the shared resources dynamically. Contrarily, the Dependency Graph Approaches (DGA), that was proposed by Chen et al. [2] in 2018, pre-computes the order in which tasks are allowed to access resources, and consists of two individual steps:

1. A dependency graph is constructed to determine the execution order of the critical sections guarded by *one binary semaphore or mutex lock*.
2. Multiprocessor scheduling algorithms are applied to schedule the tasks by respecting the constraints given by the constructed dependency graph(s).

However, generating a dependency graph for a given task set is expensive w.r.t. time and computation resources. Furthermore, an optimal dependency graph in first step may not result in an optimal schedulability in second step. Therefore, we treat the first step as an expensive black box optimization problem, and the objective is to improve the schedulability in second step.

Reinforcement learning method, i.e., Model-Based Optimization (MBO), also known as *Bayesian optimization* [3], is usually applied to solve the expensive black box optimization problem:

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x})$$

for a given function $f(\boldsymbol{x})\colon \mathcal{X} \to \mathbb{R}$ with $\mathcal{X} \subset \mathbb{R}^p$. We assume that the true expensive black box function can be approximated through a surrogate. This surrogate is a regression method that is comparably inexpensive to be evaluated. Typically a Gaussian process regression is chosen. To start the optimization, an initial design $\mathcal{D}$ of k points, laid out in a Latin hypercube design, is evaluated on the expensive function and yields the outcomes $\boldsymbol{y}$. In the following, the sequential model-based optimization iteratively repeats the following steps until a predefined budget is exhausted:

1. A Gaussian process is fitted to all past evaluations, serving as a surrogate to estimate f globally.
2. An acquisition function is optimized to determine the most promising point $\hat{\boldsymbol{x}}$: $\hat{\boldsymbol{x}} = \arg\max_{\boldsymbol{x}\in\mathcal{X}} acq(\boldsymbol{x})$.
3. $y = f(\hat{\boldsymbol{x}})$ is evaluated, $\hat{\boldsymbol{x}}$ and y are added to $\mathcal{D}$ and $\boldsymbol{y}$.

The acquisition function has to balance exploration (evaluate points where the surrogates prediction is uncertain) and exploitation (evaluate points that are predicted to be optimal by the surrogate). The final optimal result $\hat{\boldsymbol{x}}^*$ is the input that leads to the maximal observed objective value.

In this work, the student has to first study the mechanism of Dependency Graph Approach. Then, the student needs to design and implement an approach for generating the dependency graph by using reinforcement learning. In the end, the performance of different dependency graph generation methods are evaluated by applying the same task sets.

Required Skills:

- Knowledgeable of python programming
- Basic knowledge of tasks and schedulers
- R programming knowledge is beneficial

Acquired Skills after the work:

- Knowledge of Resource Synchronization
- Knowledge of Reinforcement Learning
- Knowledge of Open-Source Software
- Design, Analysis, Implementation of schedulability simulator for real-time system.

References

[1] A. Block, H. Leontyev, B. Brandenburg, and J. Anderson. A flexible real-time locking protocol for multiprocessors. In *RTCSA*, pages 47–56, 2007.

[2] J.-J. Chen, G. von der Brüggen, J. Shi, and N. Ueter. Dependency graph approach for multiprocessor real-time synchronization. In *IEEE Real-Time Systems Symposium, RTSS*, pages 434–446, 2018.

[3] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization*, 13(4):455–492, 1998.

[4] R. Rajkumar. Real-time synchronization protocols for shared memory multiprocessors. In *Proceedings.,10th International Conference on Distributed Computing Systems*, pages 116 – 123, 1990.

[5] R. Rajkumar, L. Sha, and J. P. Lehoczky. Real-time synchronization protocols for multiprocessors. In *Proceedings of the RTSS*, 1988.