Job level parallel search in software auto-tuning

Yuga Yajima, Akihiro Fujii, Teruo Tanaka
Kogakuin University

Software auto-tuning

- Software auto-tuning (AT) is a technology to improve the performance by automatically controlling "performance parameters"
  - ► "performance parameters" are parameters that affect performance in a program
- In AT, to search for appropriate values for the performance parameters, the program is iteratively executed while fitting various values to the parameters
- In case there are multiple performance parameters, the appropriate value is to be searched among their combinations
- Since it would take an enormous amount of time to try all combinations, we have been researching for efficiency of searching

Parallel searching using "Jobs"

- To improve efficiency, we developed a mechanism to parallelize the program trials [1]
- Our searching uses jobs for parallelization (refer : Fig.1)
  - ► The "job" is an unit that user submits the supercomputer system to execute a program
  - ► The AT tool creates a list of candidate values to performance parameters and submits jobs for each candidate in the list
  - ► Each job applies the candidate which is sent from the AT tool to the parameters and executes (=> **parallelized**)
  - ► When execution is complete, the user program returns the performance at that candidate value, and the AT tool aggregates them
  - ► The AT tool continues to search until it determines that it has achieved sufficiently good performance
- When the emphasis is on search algorithms, the number of candidates is determined by the algorithm in spite of the system circumstances
  - ► If the number of submitted jobs is less than the number of **available** slots at that moment, only some of the available nodes are used and execution is inefficient (refer : The left part of Fig. 3)
  - ► The number of available slots is not constant because it varies depending on system availability

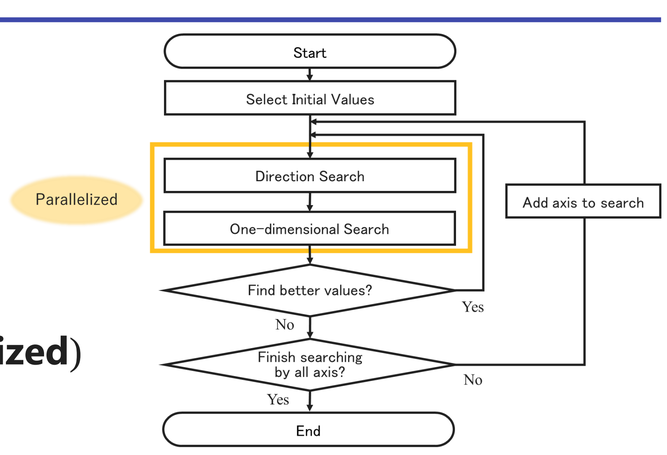

Fig. 1 Flow of our searching

Parallel job execution with the number of available slots

- **The AT algorithm can freely choose to try or not to try** for each parameter
  - ► The AT algorithm works even with only partial results of search candidates
- By using this characteristic, we changed the method of creating the list of candidates **from the algorithm based to the system based**
  - ► Decide dynamically on the number of submitted jobs according to the number of available slots at that moment
    - **If there is no room in the node, job execution is in pending**, and it takes time from submitting the job to the start of execution
    - By measuring the time from submitting to starting execution (refer : Fig. 2), the AT tool can recognize that each job is or is not pending
    - By cancelling pending jobs, execution can always be performed at the maximum number of parallelism

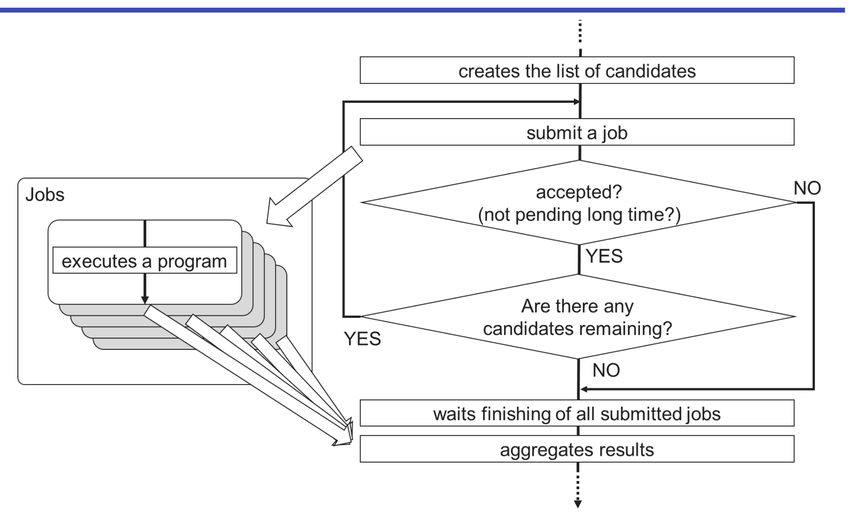

Fig. 2 Recognizing pending of jobs

Experiment and Result

- We tested this mechanism by applying it to hyperparameter tuning in machine learning for pedestrian route prediction
  - ► The performance is an error between the predicted arrival point and the actual arrival point. Lower value is better
- The experiment was conducted on Nagoya University's supercomputer, "Flow" Type II Subsystem
- The AT tool now **recognizes the computational resources available** at every step, and uses all available slots for parallel execution. As a result, the search time was reduced while maintaining the final performance (refer : Fig. 3)
  - ► At that moment, the number of available slots was 60. The searching time reduced from 4.1 hours to 2.6 hours
  - ► The performance that the AT tool extracted was worse, from 1.07 [m] to 1.12 [m], but improved from the program's original performance of 1.85 [m]

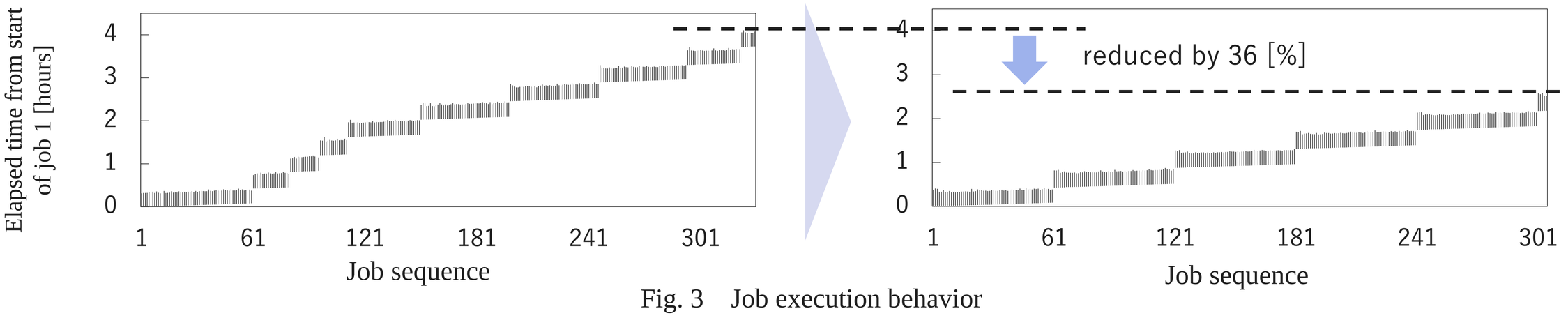

Fig. 3 Job execution behavior

Conclusion

- AT requires an enormous amount of time due to repeated trial, and we have made the search more efficient through estimation and parallelization
- The AT algorithm has the characteristic that can freely choose to try or not to try for each parameter
- The AT tool recognized the number of available slots in the system, and a time reduction was achieved

Acknowledgement

This work was partially supported by "Joint Usage/Research Center for Interdisciplinary Large-scale Information Infrastructures" in Japan (Project ID: jh220044 and jh230045) and JSPS KAKENHI Grand Number JP 18K11340 and 23K11126.

Reference

[1] Sorataro Fujika, Yuga Yajima, Teruo Tanaka, Akihiro Fujii, Yuka Kato, Satoshi Ohshima, and Takahiro Katagiri. Parallelization of Automatic Tuning for Hyperparameter Optimization of Pedestrian Route Prediction Applications using Machine Learning. HPC Asia 2023.