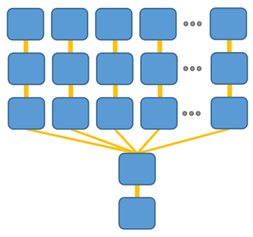
Adaptive Execution Planning in Biomedical Management Systems

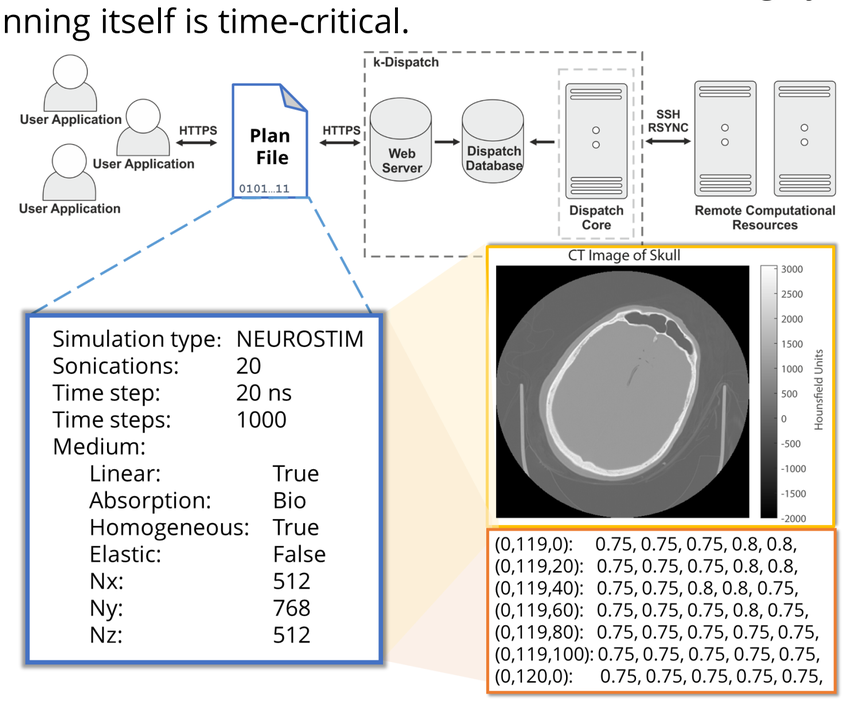
Marta Jaros[1], Bradley E. Treeby[2] and Jiri Jaros[1]
[1]Faculty of Information Technology, Brno University of Technology, Centre of Excellence IT4Innovations, CZ
[2]Department of Medical Physics and Biomedical Engineering, University College London, UK

1 k-Dispatch (Dispatch Server Module)

k-Dispatch is a service providing automated task scheduling, execution and monitoring in the process of ultrasound treatment planning. A treatment simulation comprises a workflow to be computed on a remote HPC cluster. Parsing, scheduling and the execution of this workflow is based on the provided plan file holding simulation setup and patient data.



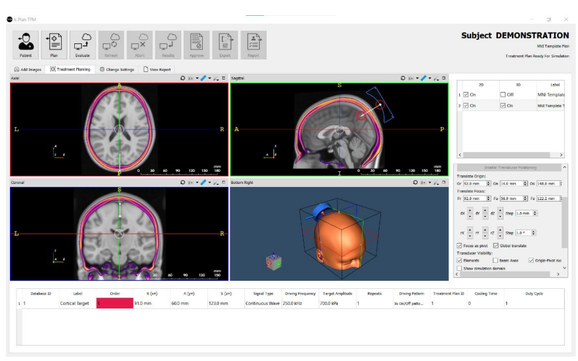
The workflow is defined as a weighted task graph where the nodes represent individual tasks possibly differing in their nature and computational demands. The design of a good workflow execution schedule is the key to minimize the computational cost and meet the time constraints. Since the HPC environments are highly dynamic, the planning itself is time-critical.

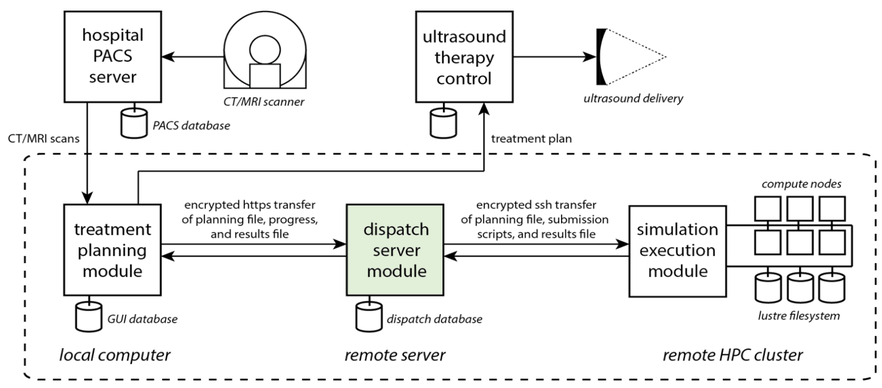


Manual execution of individual tasks is tedious and time-consuming even for expert users. Consequently, cluster utilization may be limited. Owing to the k-Dispatch's *'run and forget'* approach, the users are completely screened out from the complexity of HPC systems.

2 Application

k-Dispatch is a k-Plan's module for work offloading to remote HPC re-sources. The k-Plan system performs a model based treatment planning for ultrasound (US) therapy. Target position and US transducer para-meters are defined in the treatment planning module via a medical GUI using patient-specific CT/MR images.





The predicted acoustic and thermal output is calculated using remote HPC resources managed by the dispatch server module. A successful treatment plan can be exported to an ultrasound therapy device for patient delivery.

3 Monitoring and Fault Tolerance

Submitted jobs are periodically monitored and their statuses updated. In the case of failure, faulty and cancelled jobs are detected and restarted. Suspicious jobs may be detected using timestamps and log files.

4 Adaptation on Cluster Utilization

The run configuration of a few tasks is slightly perturbed to explore the local vicinity in the search space. Collected performance data allows to detect performance anomalies, and adapt on actual cluster utilization as well as changes in HW and SW configuration.

5 Workflow Execution Planning

The task graph exploits concurrency and dependencies in the workflow. The run configuration for each task is optimized in one pass over all available allocations. The optimization process uses collected historical performance data. Due to incomplete datasets, interpolation and machine learning methods are to be used. Candidate workflows are evaluated to find the one that meets given time and cost constraints, and minimizes queueing times.

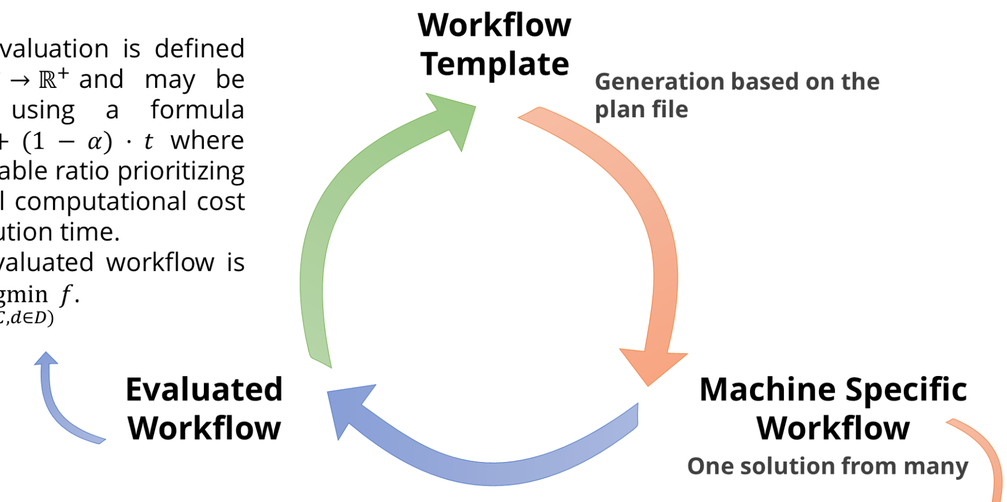
Presumptions:

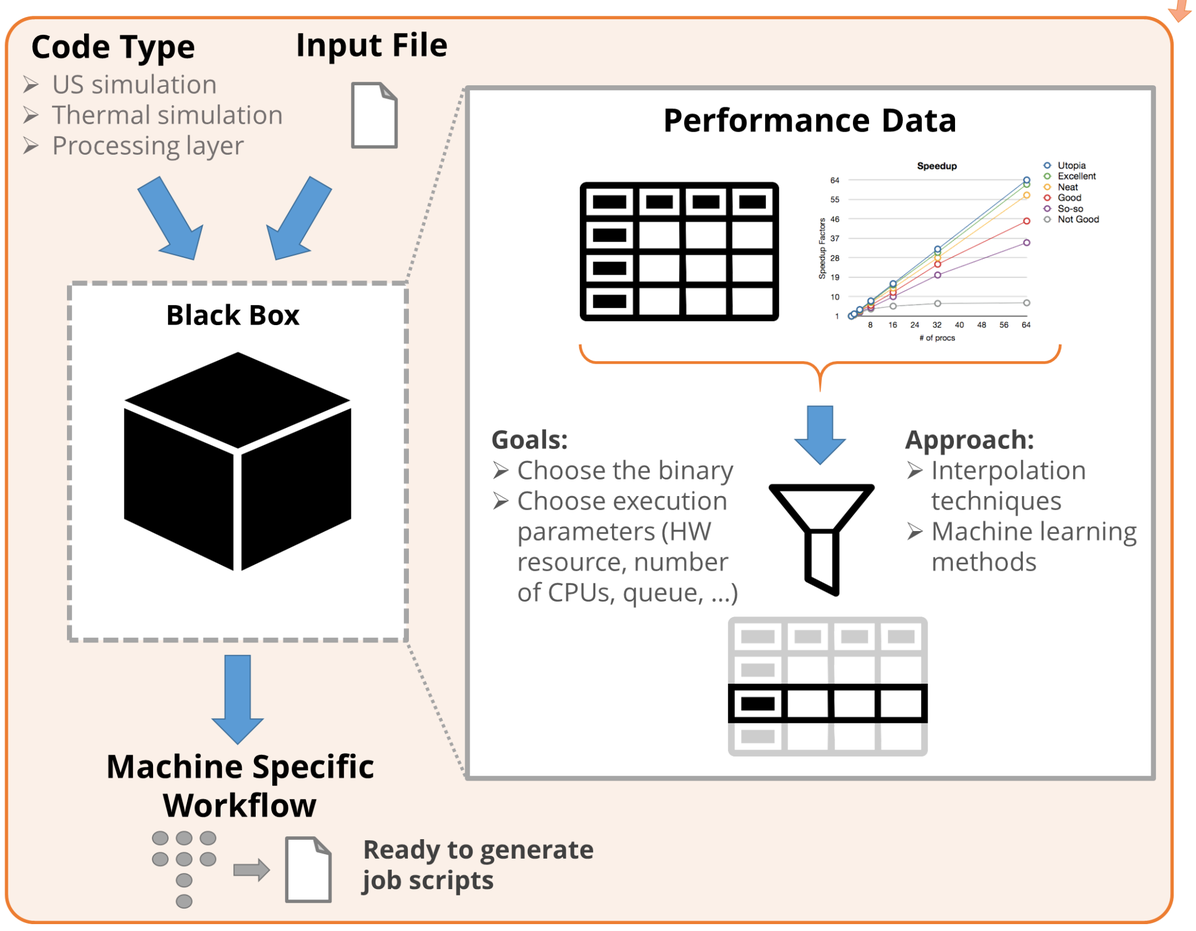
1. Consider a set of allocations $A^+ \subseteq A$ the user can operate with.
2. All possible binary executables for $a \in A^+$ are defined as $D \in (B_1, B_2, \ldots, B_n)$ where n is the number of code types within the workflow. $B_i = \{b_1, b_2, \ldots, b_m\}$ is a set of available binaries for a given code type. B may be an empty set.
4. p is a price function returning the computational cost of the workflow.
$$p : G \times C \times D \rightarrow \mathbb{R}^+$$
5. t a time function returning the execution time of the workflow.
$$t : G \times C \times D \rightarrow \mathbb{R}^+$$

Algorithm:

1. Create a workflow $G = (V, E)$ from the workflow template and input data.
2. Select candidate allocations $C = \{c \in A^+ | c.status == active \wedge c.hour\_left > 0.0\}$.
3. Generate and evaluate workflows for all combinations of candidate allocations C and binary executables D.

- Workflow evaluation is defined as $\mathbb{R}^+ \times \mathbb{R}^+ \rightarrow \mathbb{R}^+$ and may be calculated using a formula $f = \alpha \cdot p + (1 - \alpha) \cdot t$ where α is a selectable ratio prioritizing the minimal computational cost or the execution time.
- The best evaluated workflow is given by $\underset{(c \in C, d \in D)}{\operatorname{argmin}} f$.





6 Conclusions

k-Dispatch is a service for managing medical applications. Since the execution configuration strongly affects the final tasks mapping, the execution planning is of the highest priority. Currently, k-Dispatch enables users to easily execute predefined workflows on various HPC systems by only providing medical input data. The execution is planned statically using default configurations with a negligible time complexity.

7 Current and Future Work

Next steps in the development are to
(1) collect performance data for various code types,
(2) improve the logic to select run configurations,
(3) study jobs scheduling simulators,
(4) further evaluate the implemented logic and selected run configuration on both, simple and real-world, workflows,
(5) execute tested workflows in a real HPC environment.

This work was supported by the Ministry of Education, Youth and Sports of the Czech Republic through the e-INFRA CZ (ID:90140). This work was supported by the UCL EPSRC Impact Acceleration Account (EPSRC IAA 2017-20 Discovery-To-Use) and the Engineering and Physical Sciences Research Council (EPSRC) grant numbers EP/L020262/1, EP/M011119/1, EP/P008860/1, and EP/S026371/1. This work was supported by FIT/FSI-J-21-7435 Acceleration of Selected Evolutionary Communication Techniques for Solving combinatoric NP-complete tasks project and FIT-S-20-6309 Design, Optimization and Evaluation of Application Specific Computer Systems.