Heterogeneous HPC Compute Continuum: A Roadmap for Workflow Mapping & Scheduling From Sensor to Supercomputer

Aasish Kumar Sharma, Julian Kunkel, University of Göttingen, Göttingen, Germany

Research Motivation: Optimization of workload mapping and scheduling in Heterogeneous HPC-Compute Continuum (H-HPC-CC).

Key Challenges:

- **Users**: Seek minimum overall completion time, full feature flexibility, with smooth data transfers, and scalability.
- **Hosts**: Aim for optimum resource utilization & less idle time to justify the associated infrastructure and operation costs.
- **Workload Managers**: Must adapt to heterogeneous tasks, dependencies, and delays to available heterogeneous nodes.

Current Status: Workload managers are domain-specific, lack task dependencies & data latency evaluation across nodes.

Research Methodology: Strategic Research The Hybrid Waterfall Model

1. **Phase I:** Literature Review – State-of-the-Art & Gap Analysis
   - 1.1 Reading papers and doing systematic literature review
   - 1.2 Understanding the issues, state-of-the-art and analyzing the gaps.
2. **Phase II:** Work on Gaps Found During Literature Review
   - 2.1 Answer the questions raised after literature review analysis.
   - 2.2 Workout to fill the outlined gaps from the literature and the analysis.
3. **Phase III:** Implement, Evaluate, & Revise The Workouts
   - 3.1 Apply the workouts, do the necessary performance & quality tests.
   - 3.2 Validate the outcomes and analyze the weakness/impedances.
   - 3.3 Revise/optimize the workouts and re-evaluate the work.
4. **Phase IV:** Thesis Write-up and Submission.

Convinced

Example Scenario

Imagine an MRI diagnosis use case where commands are fired from edge devices and tests are carried-out throughout the system linking IoT (a hospital MRI) to Cloud and HPC system (supercomputer with thousands of processors) - all working together like an integrated system having invisible highway of computing power.

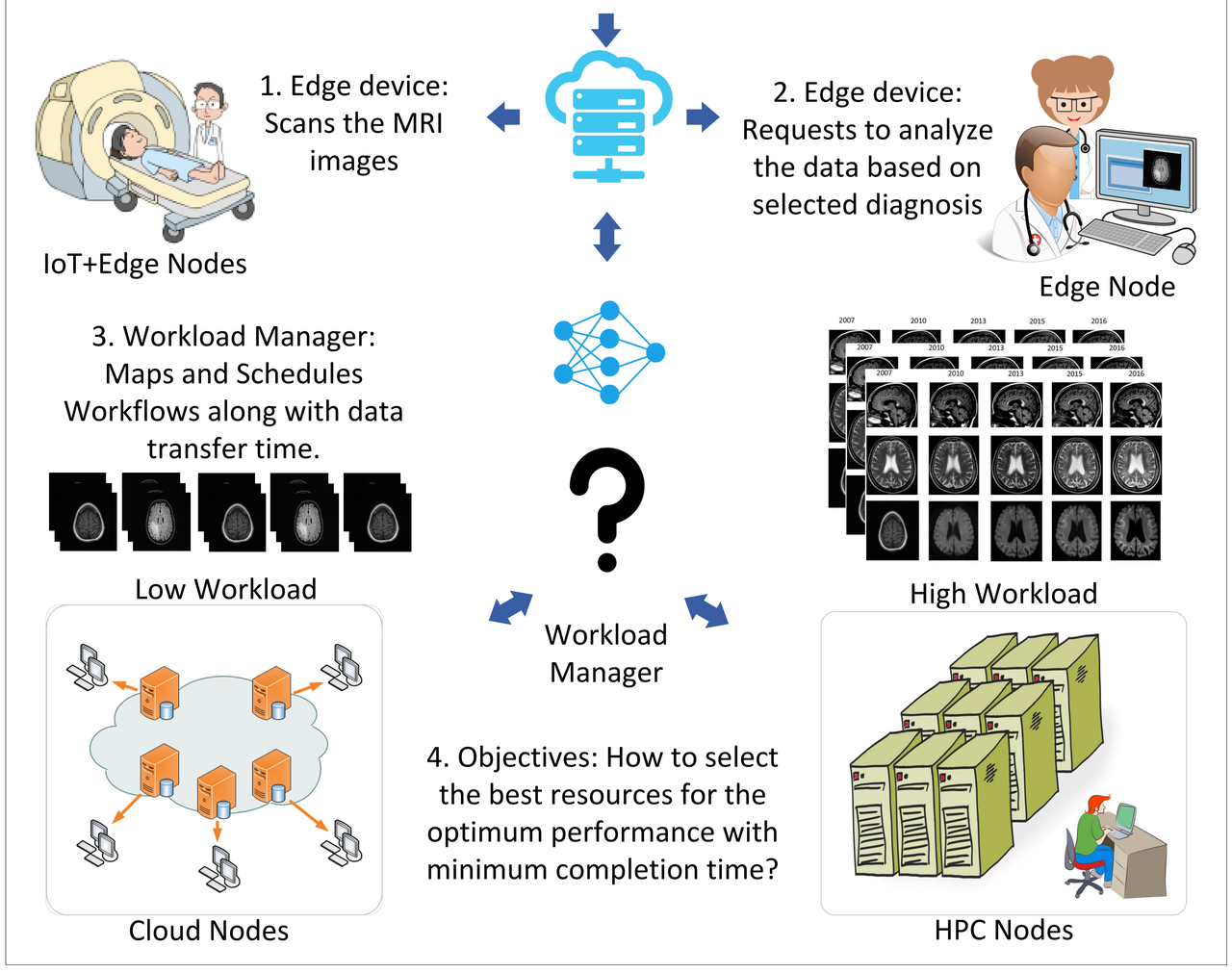
This is the **Compute Continuum**: tiny edge devices, city-scale servers, cloud data centers, even supercomputers, seamlessly linked so each task runs exactly where it is fastest and cheapest.

Real-World Impact

The general framework applicable across HPC-CC domains for workload (e.g., MLOps, Climate, Health) optimization.

For example: a mobile MRI scanner in a rural clinic. Our system lets the device process critical image slices on a nearby edge server for instant diagnosis—and then sends full-resolution data to the cloud for 3D reconstruction. That change alone cuts patient wait time from hours to minutes.

Scenario: Digital Twin for Brain Magnetic Resonance Image (MRI) Diagnosis Workload



Phase I: Literature Review

A Review of Tools and Techniques for Optimization of Workload Mapping and Scheduling in Heterogeneous HPC System [1]

Short Review Approach & Procedures: Strategic Literature Review

1. Outlined the strategic questions for review based on problem statement.
2. Outlined data collection strategy and selection approaches,

2.1 Open Search and Selection
- Used specific search terms to search the papers.
- Dataset -> Google Scholar (total 40 papers)

2.2 Manual Selection
- Selected specific papers
- Dataset -> Google Scholar (total 26 papers)

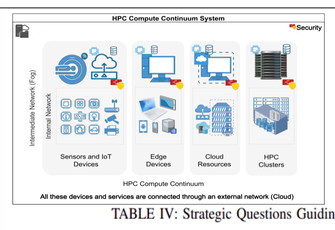
3. Drafted the review analysis and results into an article.

Findings
- Outlined multiple tools & techniques for workload optimization:
- They can be classified into four categories:
  1. **Mathematical** approaches, like Integer Linear Programming (LP),
  2. **Meta-Heuristics** approaches, mostly Nature-Inspired algorithms,
  3. **Heuristics** based approaches, mainly search and sorting methods, &
  4. **Emerging - Hybrid** approaches, like AI-ML & Quantum Computing.

Main Matrices
- They mostly look for Efficiency, Cost, Time, Scalability, and Robustness.
- Most of them have reported less then 30% of overall improvement.

Gaps Found
- No clear mathematical model formulating H-HPC-CC system & workload,
- Data delay time & tasks dependencies are neglected, to avoid complexity.

Next to find
- Among outlined approaches/tools and techniques
- **Q1.** How we can use tools & techniques to optimize in our use case?
- **Q2.** Which is the best tool or technique for our use cases and why?



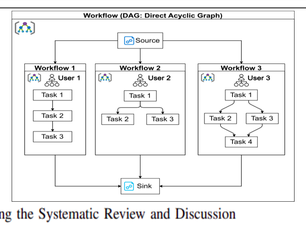


TABLE IV: Strategic Questions Guiding the Systematic Review and Discussion

Question ID	Strategic Question Grouped by Different Objectives
	Problem Understanding
Q1	What are the fundamental challenges in workload mapping and scheduling for heterogeneous computing environments?
Q2	How can we effectively model workload mapping and scheduling problems to reflect real-world scenarios and constraints?
	Optimization Formulation
Q3	What are the key performance metrics and objectives when formulating optimization problems?
Q4	What are the state-of-the-art algorithmic approaches and optimization techniques for workload mapping and scheduling?
Q5	Are there any methods that could provide near-optimal solutions with reduced computational complexity?
	Resource Management and Energy Efficiency
Q6	How can resources in heterogeneous computing environments be effectively managed and allocated for workload execution?
Q7	What strategies and tools are available for optimizing workload mapping and scheduling to reduce energy consumption?
Q8	How can we strike a balance between performance (execution time) and energy efficiency in scheduling decisions?
Q9	What technologies can be integrated into scheduling for performance/energy savings?
	System Implementation
Q10	What methods exist for parallelizing scheduling decisions to efficiently handle large-scale workloads?
	Tool Support and Integration
Q11	What are the available software toolkits, frameworks, and libraries for implementing workload mapping and scheduling strategies?
Q12	How can these tools be integrated into existing HPC and cloud computing environments?
	Evaluation and Benchmarking
Q13	How do we benchmark and evaluate the performance of different workload mapping and scheduling approaches?
Q14	What are the key considerations when conducting experiments and comparing the effectiveness of various tools and techniques?
	Applications and Best Practices
Q15	What are some real-world applications and case studies where effective workload mapping and scheduling have significantly improved system performance?
Q16	How do the lessons learned from these analyses inform best practices for solving similar problems?

Phase II: System & Workload Modeling Integrated With Snakemake Scheduler Plugin

Workflow-Driven Modeling for the Compute Continuum: An Optimization Approach to Automated System and Workload Scheduling [2]

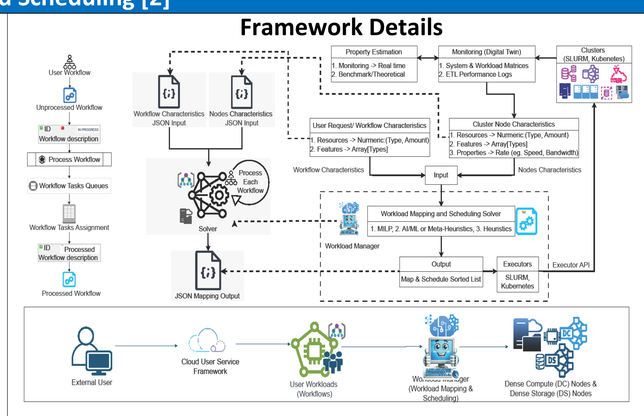
Motivation
- Optimum resources utilization across HPC-CC for Scientific Workflows.
- Overcome manual workload mapping across HPC-CC to reduce inefficiency.
- Use existing tools for unified & optimized cross-platform scheduling.

Research gap:
- No framework offers unified system & workload modeling with:
- Optimization-based, Automated mapping & scheduling, and
- Seamless integration with established workflow management tools..

Proposed Modeling Framework: Overview

- Unified modeling of:
  - Systems - Compute Continuum (resources, features, properties),
  - Workloads - Workflows (tasks utility, and dependencies).
- Optimization:
  - Multi-objective optimization for automated mapping & scheduling.
- Workflow integration:
  - Snakemake scheduler extension, practical deployment.
- Evaluation:
  - Performance & scalability tests with real-world workloads.

Framework Details



OriginalSnakefile	Modeling Integration With Snakemake	Extended Snakefile

```
rule all:
    input: "results/output.txt"
    resources:
        slurm_partition = cpuNode

rule demo:
    output: "results/output.txt"
    threads: 2
    resources:
        mem_mb=1024,
        runtime=5,
        slurm_partition = gpuNode
    shell:
        "echo Hello > {output}"
    ...
```

```
configfile: "scheduler_config.yaml"

rule all:
    input: "results/output.txt"

rule demo:
    output: "results/output.txt"
    threads: 2
    resources:
        mem_mb=1024,
        runtime=5
    params:
        job_specification={
            "features": ["cpu"],
            "resources": {"input_size_mb": 10, "
                output_size_mb": 5},
            "properties": {"cpu_flops": 1e9}
        }
    shell:
        "echo Hello > {output}"
```

System Model
Ni = {Ri, Fi, Pi}
- Ri: Resources (e.g., Cores, memory, storage)
- Fi: Features (e.g., ISA, accelerators)
- Pi: Properties (e.g., compute rates)

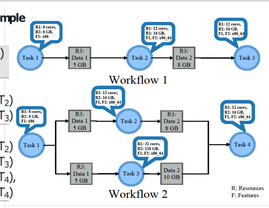
Workload Model
Tj = {Rrj , Frj , Uj, δj}
- Rrj ⊆ Ri: Resource compatibility
- Frj ⊆ Fi: Feature compatibility
- Uj: Resource usages, (data delays)
- δj: Task dependencies:
- DAG → precedence

Multi-Objective Function: Minimize { α(Uijxij) + β(Cmax) }, where xij = 1, node i is assigned to task j, 0 otherwise, and Cmax: Overall completion time.

The table on left shows a sample system and characteristics of its three nodes. Similarly, table on right shows a sample workload and characteristics of its tasks with two workf ows (shown in f gure 2).

Nodes	N_1	N_2	N_3...
Node Name	Node 1	Node 2	Node 3
Core (R^1)	8	48	2572
Storage (R^3 in TB)	0.5	20	210
Features	F^1	F^1, F^2	F^1, F^2
Data Transfer Rate (P^3 in GB/s)	100	100	100
Processing Speed (P^1 GFLOPS)	1	1	1

W Id	T Id	(R^1)	(F^1)	(R^3)	(d_{ij}) N 1,N 2,N 3	($d_{ij'j}$) (same)	$\delta(j,j')$
W_1	T_1	8	F_1	2	(3, 3, 3)	0.02	-
	T_2	12	F_1, F_2	5	(5, 5, 5)	0.05	$\delta(T_1, T_2)$
	T_3	12	F_1, F_2	8	(2, 2, 2)	0.08	$\delta(T_2, T_3)$
W_2	T_1	8	F_1	2	(3, 3, 3)	0.02	-
	T_2	12	F_1, F_2	5	(5, 5, 5)	0.05	$\delta(T_1, T_2)$
	T_3	32	F_1, F_2	5	(2, 2, 2)	0.05	$\delta(T_1, T_3)$
	T_4	12	F_1, F_2	10	(2, 2, 2)	0.10	$\delta(T_2, T_4)$, $\delta(T_3, T_4)$



An optimum solution for MRI sample workload: Mapping and Scheduling

Status	Workf ow Id	Task Id	Optimal Node	Start (sec.)	End (sec.)	Resource Usage	Makespan (sec.)
Optimal	W_1	T_1	N_2	0.0	3.0		
		T_2	N_2	3.0	8.0		
		T_3	N_2	8.0	10.0		
	Total					32.0	10.0
Optimal	W_2	T_1	N_1	0.0	3.0		
		T_2	N_1	3.0	8.0		
		T_3	N_2	3.02	5.02		
		T_4	N_1	8.0	10.0		
	Total					64.0	10.0

Phase III: Implementation I - Evaluations

GrapheonRL: A Graph Neural Network and Reinforcement Learning Framework for Constraint and Data-Aware Workflow Mapping and Scheduling in Heterogeneous HPC Systems [3]

Motivation
- Overcome the limitations found in Phase II for large scale workload.

Research gap: (Phase II)
- What tool/techniques could dynamically address the changes, & large scale workload

Mapping and Scheduling Objectives:
1. Minimum Makespan (overall completion time)
2. Optimum Resource Usage (compute & storage)

Constraints:
1. One task should be assigned to only one node.
2. Task should be assigned within the node capacity.
3. Task feature request should be respected.
4. Task dependency should be respected.
5. Data transfer time should be considered when tasks are assigned to nodes.

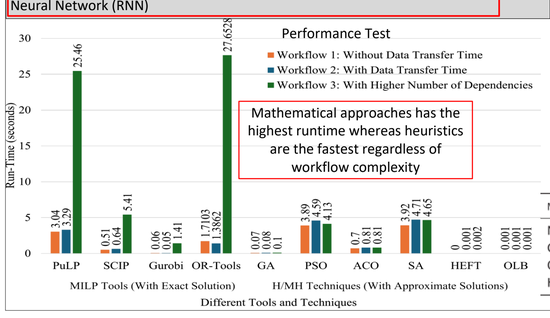
All Four Approaches Outlined In Phase I: Literature Review, Are Evaluated Applying Above Objectives and Constraints

1. Mathematical: Integer/Mixed Integer Linear Programming (ILP/MILP)
→ **Tools**: PuLP, SCIP, Gurobi, OR-Tools

2. Meta-Heuristics: Nature Inspired
→ **Techniques:** Genetic Algorithm (GA), Particle Swarm Optimization (PSO), Ant Colony Optimization (ACO), Simulated Annealing (SA)

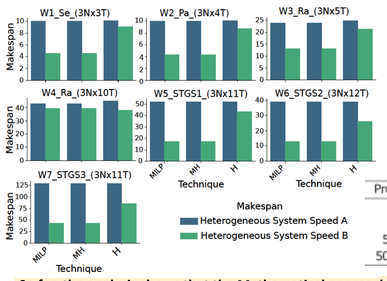
3. Heuristics: Search, Sort
→ **Techniques:** Heterogeneous Earliest Finish Time (HEFT), Opportunistic Load Balancing (OLB)

4. Emerging-Hybrids: AI/ML, Quantum Computing
→ **Techniques:** Graph Neural Network (GNN), Reinforcement Learning (RL), Recurrent Neural Network (RNN)



Mathematical approaches has the highest runtime whereas heuristics are the fastest regardless of workflow complexity

Quality Test: Faceted View for Minimum Makespan On Varying System & Workflows



MILP tools \& MH/H techniques makespan test. Tested over different workflows (W1 to W7) with varying nodes (N), tasks (T) \& edges (dependencies). Where Se: Sequential, Pa: Parallel, Ra: Random, STGS: Standard Task Graph

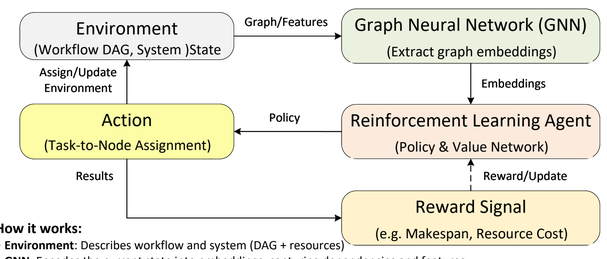
Test shows MILP provides optimum solution in all the cases.

Scalability Analysis: Time to Solution

Problem Size	MILP	Meta-heuristic	Heuristic
5 ×5	0.02s	0.03s	0.00s
50 ×50	-	77.78s	0.01s
500 ×500	-	6513.33s	0.24s
5000 ×5000	-	-	560.59s

So far, the analysis shows that the Mathematical approaches provides best solution but are not scalable, whereas Heuristics are scalable but does not provide optimal solution. To overcome the situation we apply GNNRL applying emerging approaches – AI/ML

GrapheonRL: GNN RL Scheduler Framework for Workflow Mapping & Scheduling



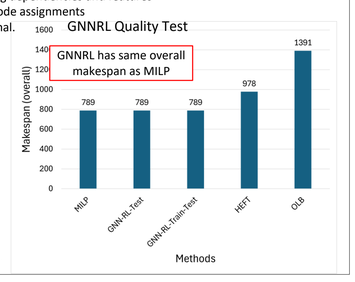
How it works:
- **Environment**: Describes workflow and system (DAG + resources)
- **GNN**: Encodes the current state into embeddings, capturing dependencies and features
- **RL Agent**: Uses embeddings to select (or sequence) task-node assignments
- **Reward**: Environment simulates result, returns reward signal.

GNNRL Shows it can provide MILP level best solution as well as can handle load equal to heuristics.

Scale Test: Runtime by Method and Problem Size

Method (Behavior)	10×10	100×100	1K×1K	10K×10K
MILP (Optimal)	0.44	—	—	—
GNN-RL-Test (Optimal)	0.65	1.58	27.78	290.40
OLB (Greedy)	0.01	0.09	0.72	75.51
HEFT (Greedy)	0.01	0.11	0.93	107.86

Runtime is in seconds, and K = 1000



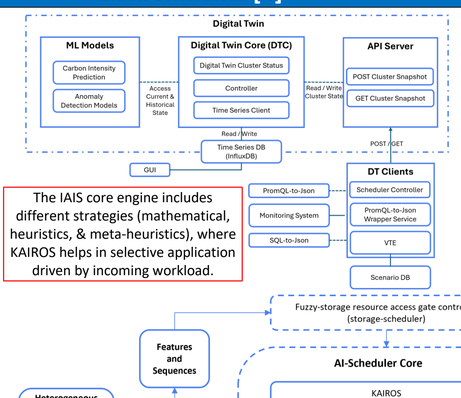
Phase III: Implementation II – DECICE Integrated AI Scheduler

Design and Implementation of Integrated AI Scheduler for Dynamic Cloud Workloads Allocation in Kubernetes Environments [4]

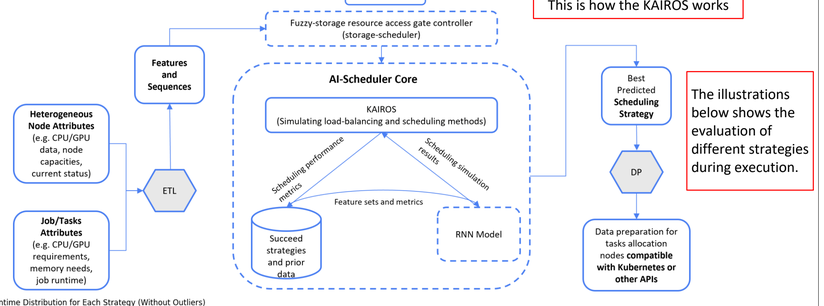
Motivation
- **Need for AI Integration**: There's a growing need to integrate AI-driven techniques to improve scheduling decisions, especially with evolving workloads.

Objectives
- Develop a modular, scalable AI-based scheduler integrated with Kubernetes and KubeEdge.
- Enhance workload orchestration using: Temporal learning (RNN, TCN), Reinforcement Learning (PPO), Strategy evaluation (KAIROS)
- Minimize makespan and maximize throughput with real-time system feedback.
- Ensure energy efficiency and carbon-aware scheduling.

Research Methodology
- Quantitative benchmarking and architectural modeling using DECICE framework. Digital Twin (DT) acts as a real-time system mirror.
- Scheduling strategy driven by: KAIROS multi-strategy engine, Hash-based caching for job-node combinations, PPO to select optimal strategy dynamically
- Integration of Fuzzy Access Controller (FAC) for handling uncertainty in storage access.

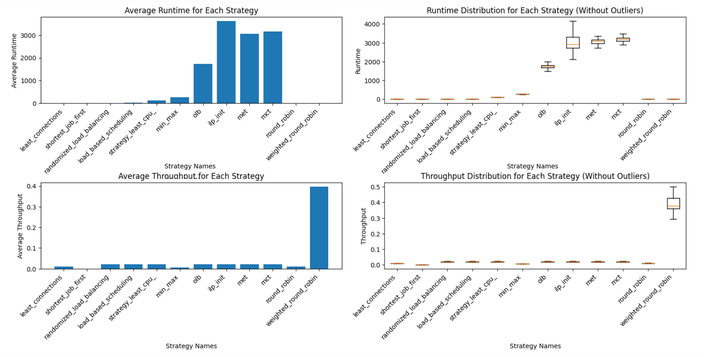


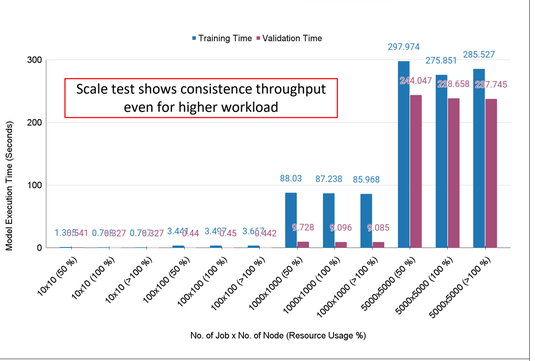
The IAIS core engine includes different strategies (mathematical, heuristics, & meta-heuristics), where KAIROS helps in selective application driven by incoming workload.



This is how the KAIROS works

The illustrations below shows the evaluation of different strategies during execution.





Evaluation & Discussion

Evaluation
- GrapheonRL achieves MILP-level optimality at much larger scale.
- IAIS is flexible, constraint-aware, easy to adapt for new objectives.
- Plugs transparently into existing Snakemake workflows.
- Fully dynamic algorithmic approach.

Discussion
- Compared to mathematical and heuristics,
- The hybrid approaches provide better results for higher scale when up-front training time are considered.

Conclusion & Future Work

Conclusion
1. Unified mathematical modeling of heterogeneous systems & workloads,
2. Multi-objective optimization framework with scalable solution methods,
3. Seamless integration with established workflow management tools.
4. GrapheonRL and IAIS bridges the gap between optimality and scalability
   - Mainly for heterogeneous HPC-CC mapping and scheduling.

Future Work
- Real-time monitoring,
- Extend to energy optimization, and
- ML-driven resource estimation.

Preprints & Publications

[1] Preprint: Authors: Aasish Kumar Sharma, Julian Kunkel (2024) "*A Review of Tools and Techniques for Optimization of Workload Mapping and Scheduling in Heterogeneous HPC System*" **In Peer review of journal** SAGE, International Journal of High Performance Computing Applications; **Available at:** ArXiv: DOI: https://doi.org/10.48550/arXiv.2505.11244

[2] Publication: Authors: Aasish Kumar Sharma, Christian Boehme, Patrick Gelß, Ramin Yahyapour, Julian Kunkel (2025). "*Workflow-Driven Modeling for the Compute Continuum: An Optimization Approach to Automated System and Workload Scheduling*". **Presented at** workshop: 13th IEEE International Workshop on Modeling and Verifying Distributed-Embedded Applications (MVDA 2025); **Status:** Accepted & Presented; **Available at:** ArXiv DOI/Link: https://doi.org/10.48550/arXiv.2505.12184

[3] Publication: Authors: Aasish Kumar Sharma, Julian Kunkel (2025). "*GrapheonRL: A Graph Neural Network and Reinforcement Learning Framework for Constraint and Data-Aware Workflow Mapping and Scheduling in Heterogeneous HPC Systems*"; **Presented at** Emerging Advances in Technologies & Applications(EATA), IEEE Annual International Computer Software and Applications Conference (COMPSAC 2025), **Status:** Accepted & Presented; **Available at:** ArXiv: DOI/Link: https://doi.org/10.48550/arXiv.2506.00260

[4] Publication: Authors: Bidollahkhani, M., Sharma, A. K., Nanavati, S. P., Seyedkazemi Ardebili, M., Mamulashvili, G., Aydin, M., Stein, F., Akbari, M., & Kunkel, J. M. (2025). "*Design and Implementation of Integrated AI Scheduler for Dynamic Cloud Workloads Allocation in Kubernetes Environments*". **In Proceedings** of Future Technologies of Computing Conference (FTC 2025). **Status:** Accepted; **Available at:** https://www.decice.eu/about/