Scalable Quantum Cloud Scheduling

Optimizing Resource Allocation for Efficient NISQ Computing

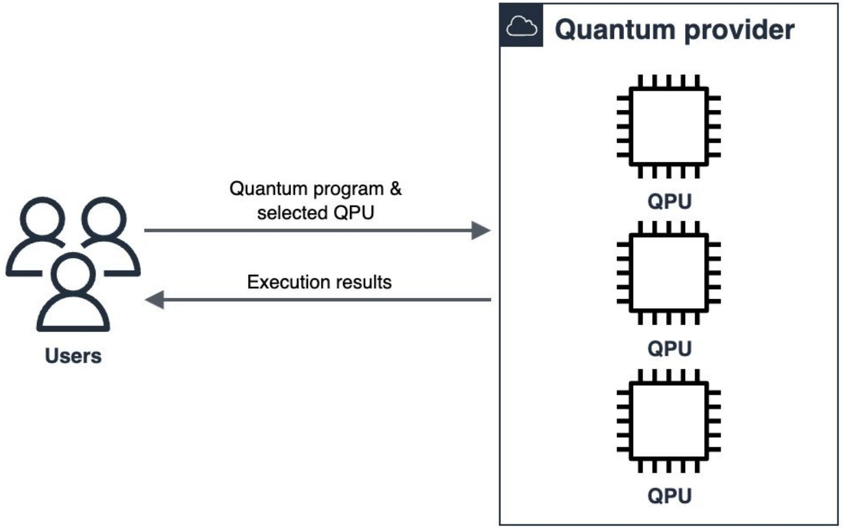
Dmitry Lugovoy
Advisors: Emmanouil Giortamis, Francisco Romão

Quantum computing

- Potential for tackling intractable problems that classical computers cannot
  - Application areas: cryptography, drug discovery, optimization
- Current state: Noisy Intermediate-Scale Quantum (NISQ)
  - Characterized by inherent operational noise
  - Do not have required scale to fully mitigate errors
- Publicly accessible only through quantum cloud providers

Quantum cloud

- Providers: IBM, GCP, AWS, Azure, etc.
- Available capacities: Up to 433 quantum bits
- Access model: Users schedule quantum programs for execution on quantum computer of their choice and pay for execution time



State-of-the-art

Current research:

- Quantum resource allocation schemes [1], [2]
- Automatic single-job scheduling, optimizing for balanced QoS [3]
- Platform for quantum applications, allowing for QPU selection based on user's QoS preferences for individual jobs [4]

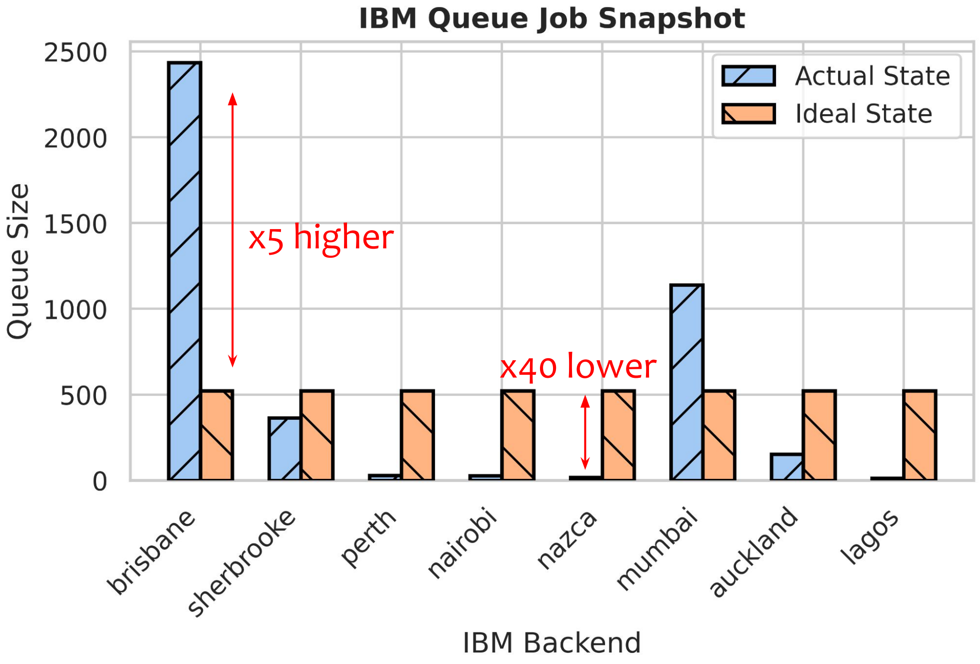
Quantum multi-job scheduling is severely understudied

[1]: Optimal Stochastic Resource Allocation for Distributed Quantum Computing, arXiv preprint '22
[2]: Stochastic Qubit Resource Allocation for Quantum Cloud Computing, NOMS '23
[3]: Adaptive job and resource management for the growing quantum cloud, QCE '21
[4]: PlanQK, https://planqk.de/

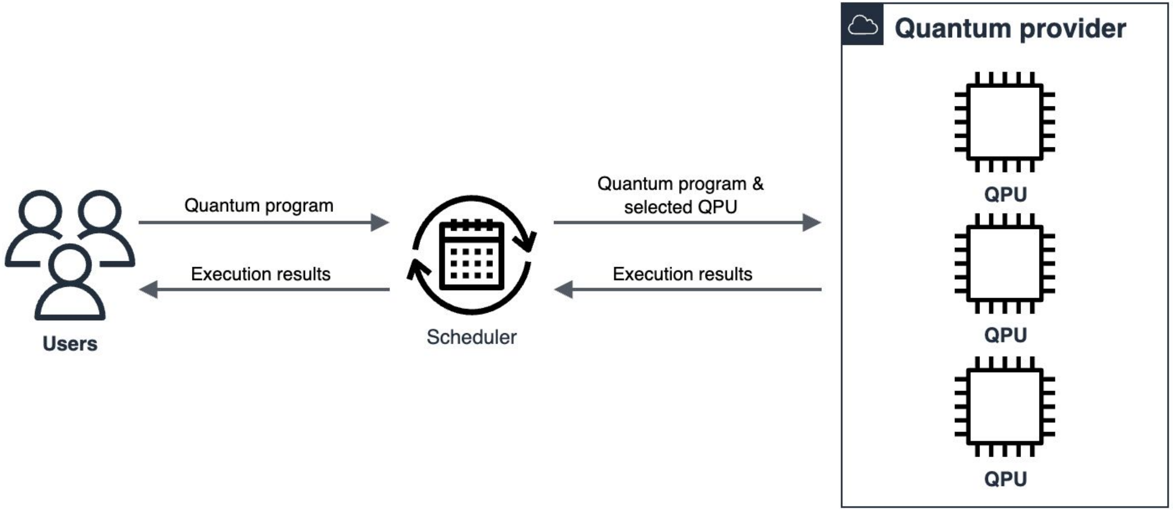
Manual QPU selection

Manual QPU selection leads to:

- Uneven workload distribution
- QPU underutilization
- Sub-optimal fidelity
- High waiting time



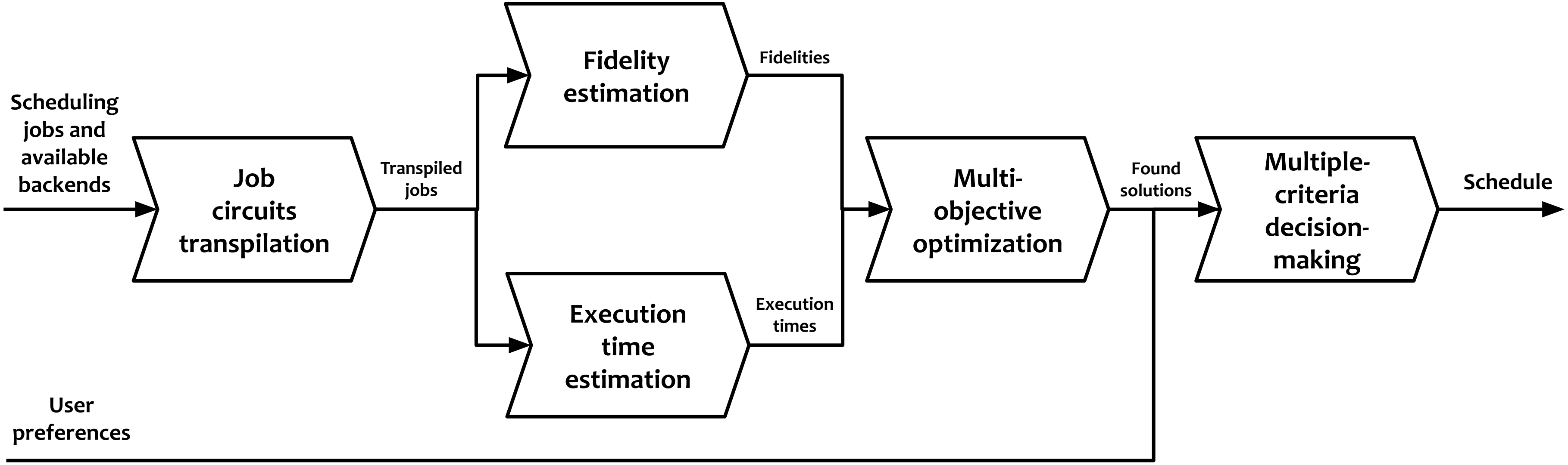
Scalable Quantum Cloud Scheduling



Design goals:

- Many-to-many scheduling
- Optimization for conflicting objectives (fidelity vs waiting time)
- Execution time estimation
- Customizable objective priorities
- Scalability with the growing quantum cloud

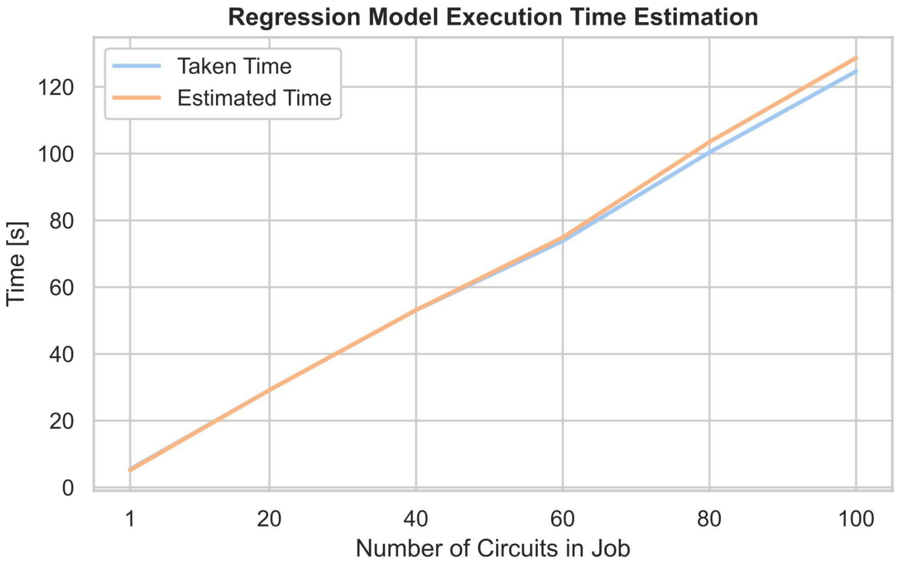
System overview



Input: scheduling jobs, available QPUs, and user priorities
Output: job-to-QPU assignments

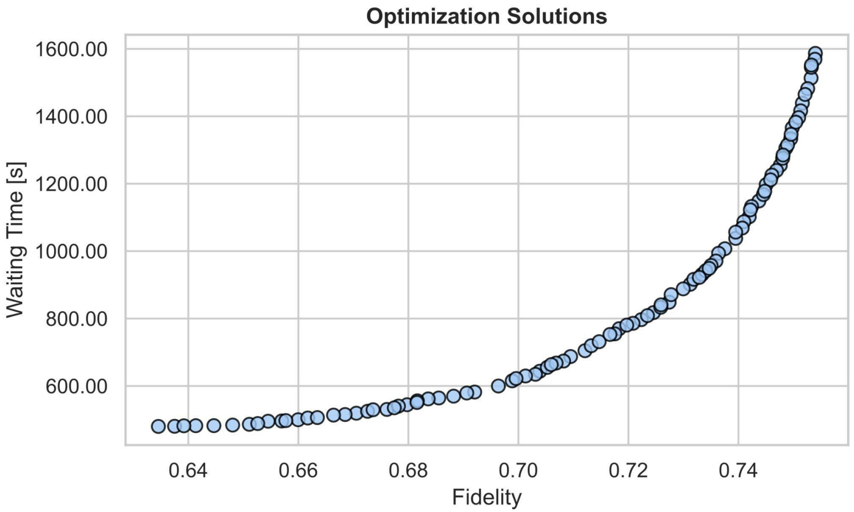
Execution time estimation

- Collected data set: >7000 jobs
- Approaches:
  - Circuit DAG traversal
    - Gate length based
    - Calibration based
  - **Regression analysis**
- Best model: Extra trees regressor
  - Ensemble supervised machine learning method that uses decision trees
- R^2 score: 0.988 (best possible 1.0)
  - Indicates how much of the variation of a dependent variable is explained by an independent variable



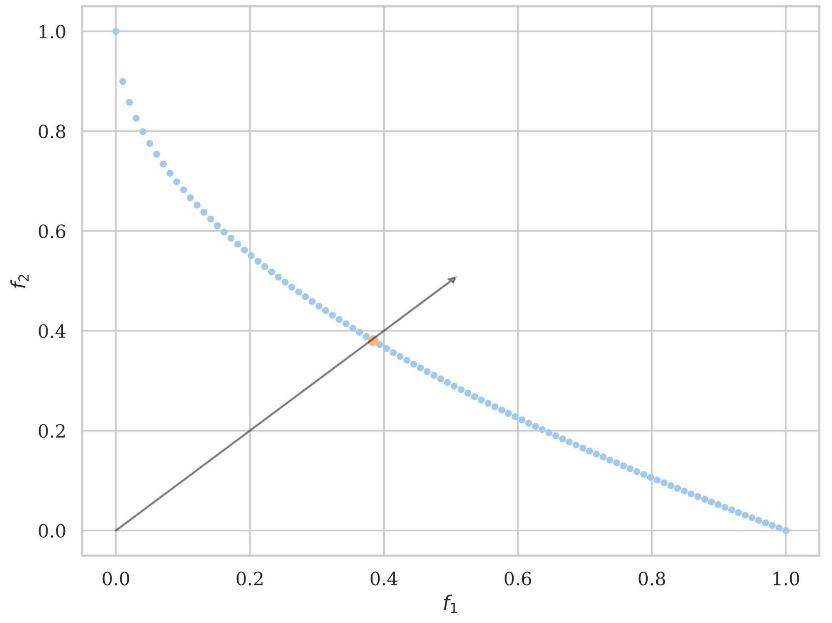
Multi-objective optimization

- Problem: Schedule *n* jobs (consisting of multiple circuits) on *m* QPUs
- Objectives:
  - Fidelity ↑
  - Waiting time ↓
- Problem formulation:
  - Binary variable (problem dimensionality - n×m)
  - Discrete variable (problem dimensionality - n)
- Solution: Genetic algorithm
  - Adaptive heuristic search algorithm inspired by the process of natural selection. Uses a population of candidate solutions, applies genetic operators, and iteratively evolves these solutions over generations

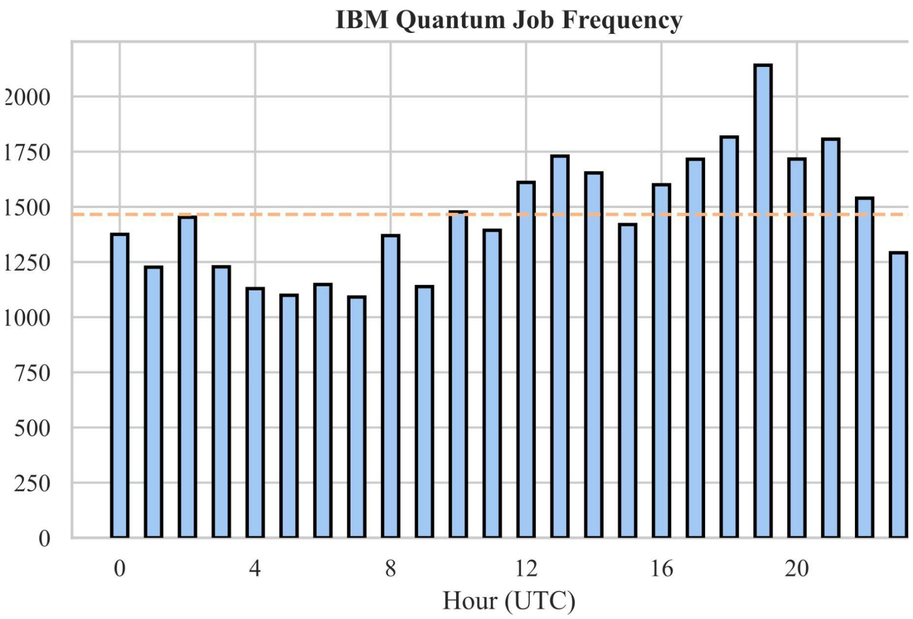


Multi-criteria decision-making

- Output of optimization - Pareto front of solutions
- MCDM methods help making a decision, which one to choose
- Choice criteria: User scores, indicating the importance of the objectives
  - $\alpha \times$ fidelity + $\beta \times$ waiting time, s.t. $\alpha + \beta = 1$
- Used method: Pseudo-Weights
  - Calculates a score for each objective, allowing to compare solutions by performance across multiple objectives

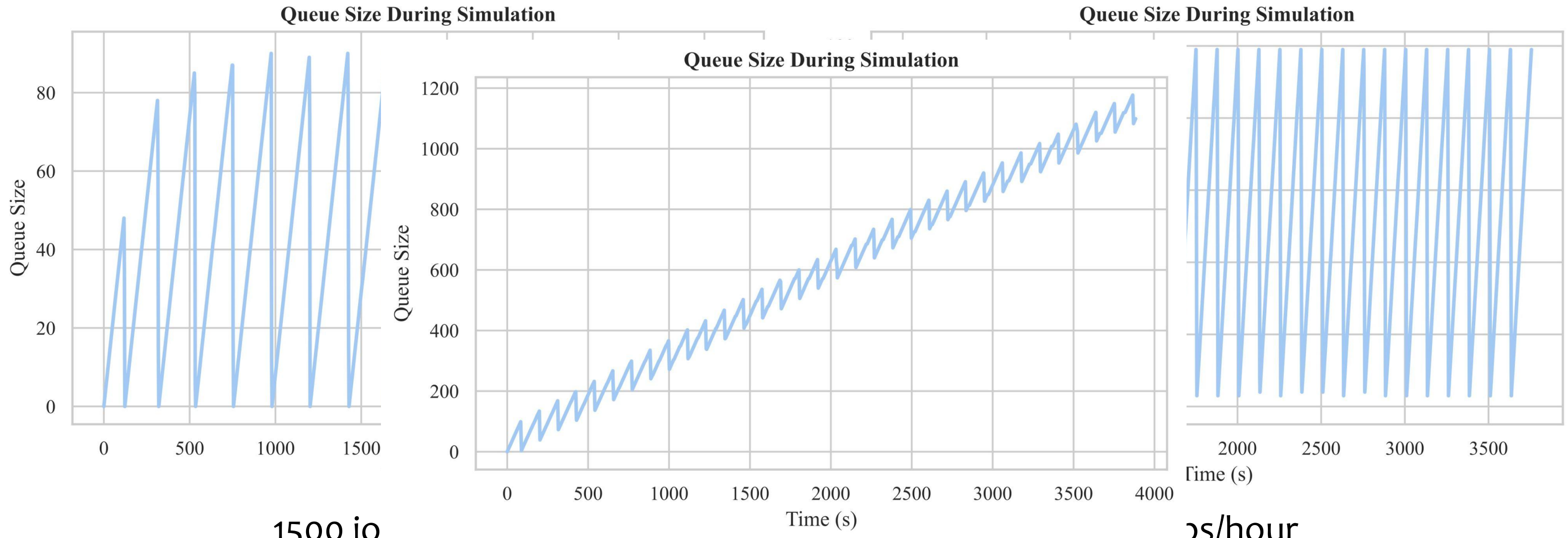


Evaluation - Quantum cloud workload



Number of incoming jobs greatly varies over the day, ranging from 1100 to 2050 jobs per hour, total average being ~1500 jobs per hour

Evaluation - System throughput



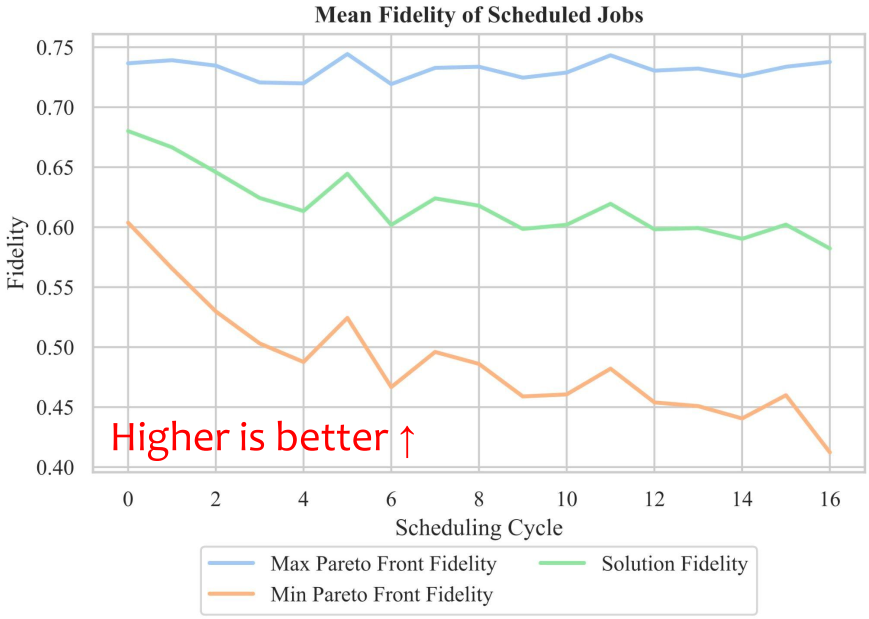
1500 jo

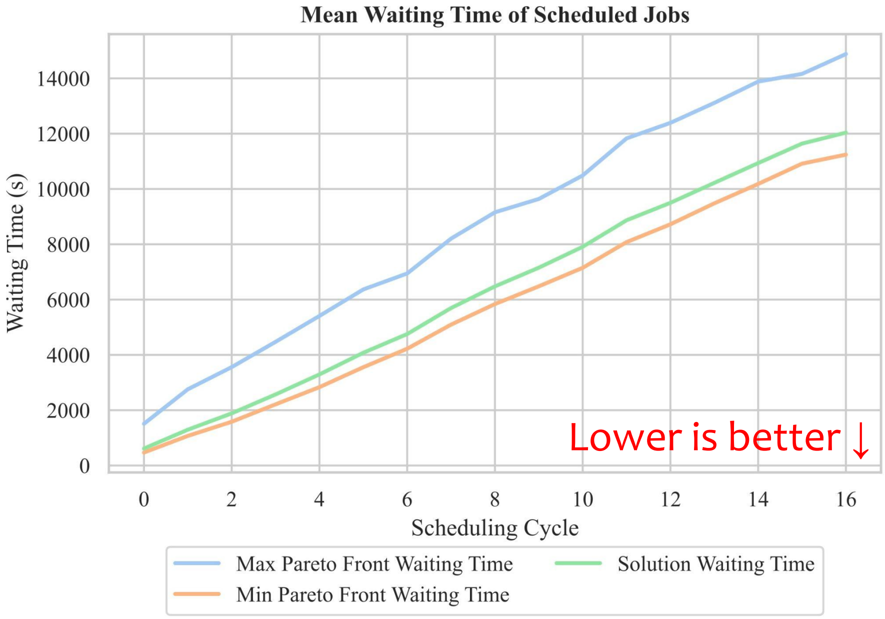
os/hour

4500 jobs/hour

System successfully handles the current workload and doubled workload. However, with a workload tripled, the system begins to lag behind.

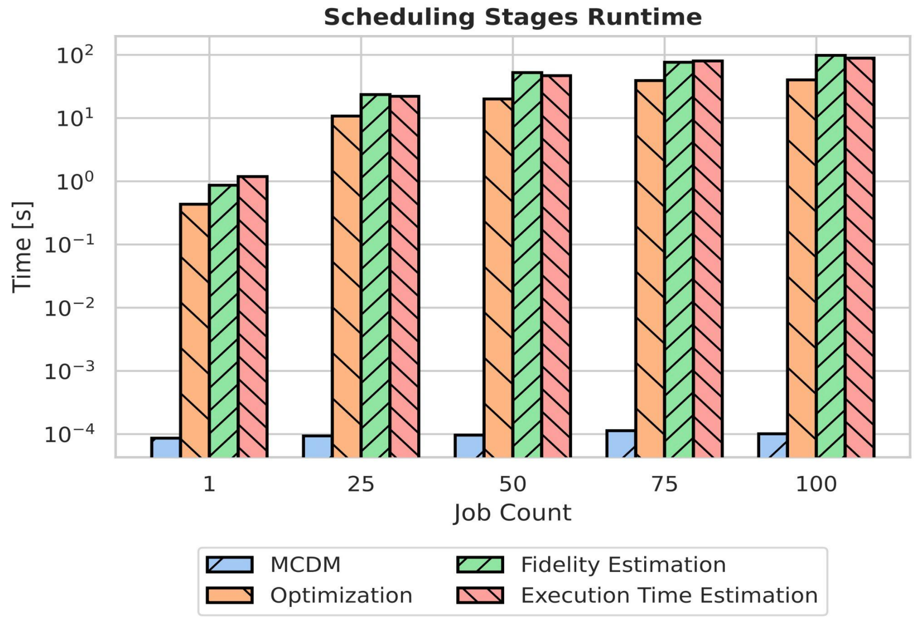
Evaluation - System stability





System shows stable performance throughout the simulation, consistently identifies diverse solutions and effectively avoids falling into local optima

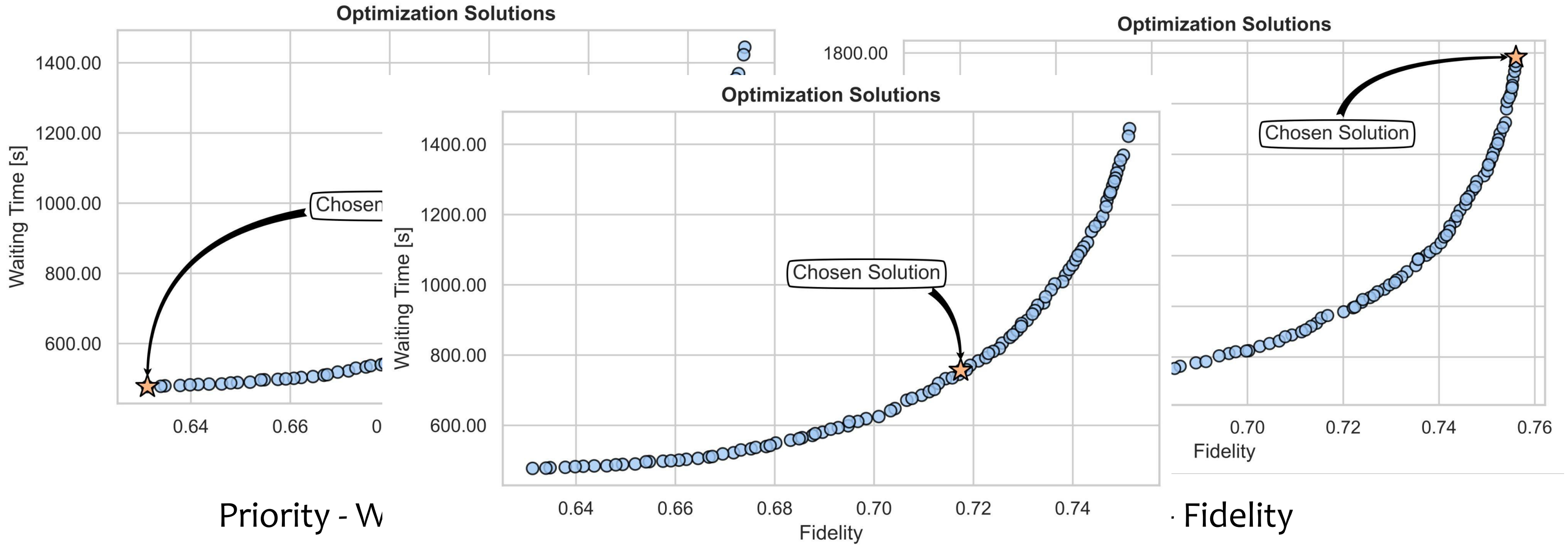
Evaluation - Scheduling stages performance



Lower is better ↓

Fidelity and execution time estimation account for the most system runtime, the core optimization step consumes only a quarter of it

Evaluation - MCDM



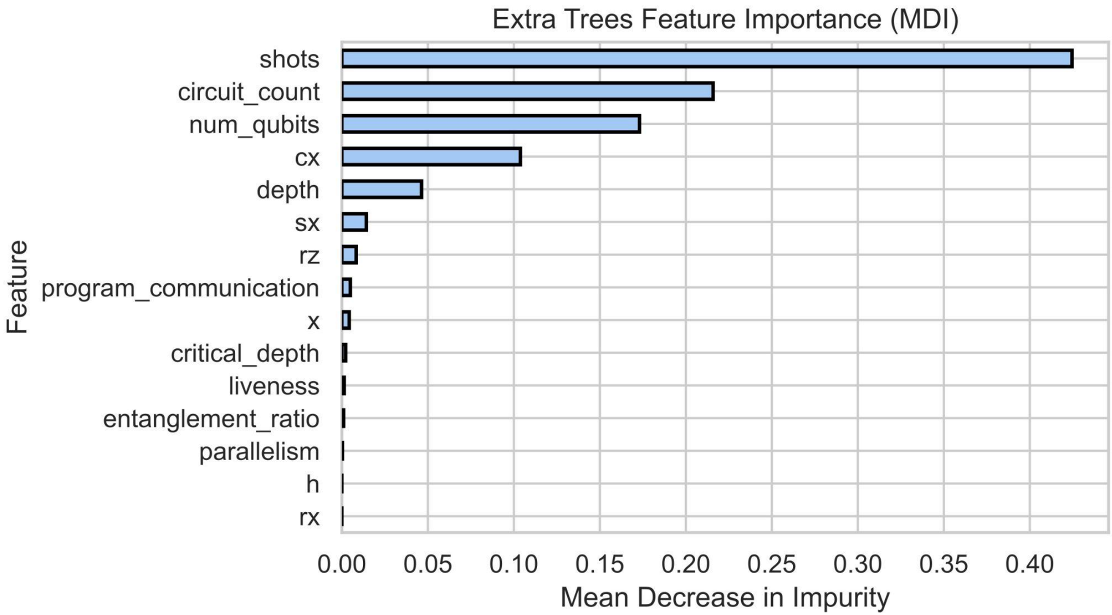
Priority - W

Fidelity

Priority - Tradeoff

MCDM module consistently identifies solutions that best correspond to the specified priorities (lowest waiting time, highest fidelity, tradeoff)

Evaluation - Execution time estimation



Higher is better ↑

Only five features seem to be important for our model: #shots, #circuits, total # used qubits, total #SWAP gates, total depth

Conclusion

- Quantum resource scheduling is challenging
  - Poor circuit execution time estimation by quantum cloud providers
  - Trade-off between fidelity and waiting time
  - Understudied multi-circuit scheduling

- Our proposal: Scalable Quantum Cloud Scheduler
  - Accurate execution time estimator, based on regression model
  - Multi-objective scheduling of multiple circuits to multiple QPUs
  - Selection of the optimal solution based on the objective priorities
  - Great scalability of scheduling time with the number of jobs