QUANTUM MINI-APPS: A FRAMEWORK FOR DEVELOPING AND BENCHMARKING QUANTUM-HPC APPLICATIONS.

NISHANT SAURABH[1], PRADEEP MANTHA[2], FLORIAN J. KIWIT[2,4], SHANTENU JHA[3] AND ANDRE LUCKOW[2,4].

FIRST INTERNATIONAL WORKSHOP ON HIGH PERFORMANCE AND QUANTUM COMPUTING INTEGRATION (HPQCI) @ HPDC 2024.

Utrecht University[1] (NL), Ludwig Maxmilian University[2] (Germany), Rutgers University[3] (USA), BMW Group[4] (Germany)

Image credit: Midjourney

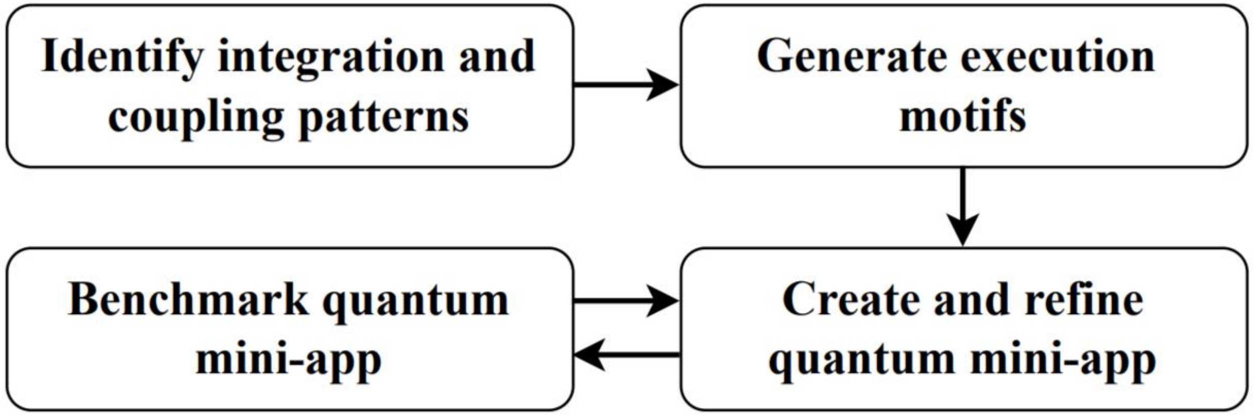
EVOLVING QUANTUM HARDWARE AND ITS INTEGRATION WITH HPC SYSTEMS: CHALLENGES AND CRITICAL QUESTIONS

Quantum-HPC software stack is fragmented, each specialized in specific hardware, algorithms and applications. Many critical questions need to be addressed when developing Quantum-HPC systems and applications:

- What are common **quantum-HPC application scenarios**?
- What are common **execution patterns** in quantum-HPC system and applications?
- How are **classical** and **quantum** component **coupled**?
- How can **the quantum software** and **hardware stack** be optimally integrated?
- How can we **map** classical and quantum components to appropriate **processing elements**?
- How can we **characterize** and **optimize** the **performance** of quantum applications?
- How can we **develop** and prototype **efficient quantum applications**?

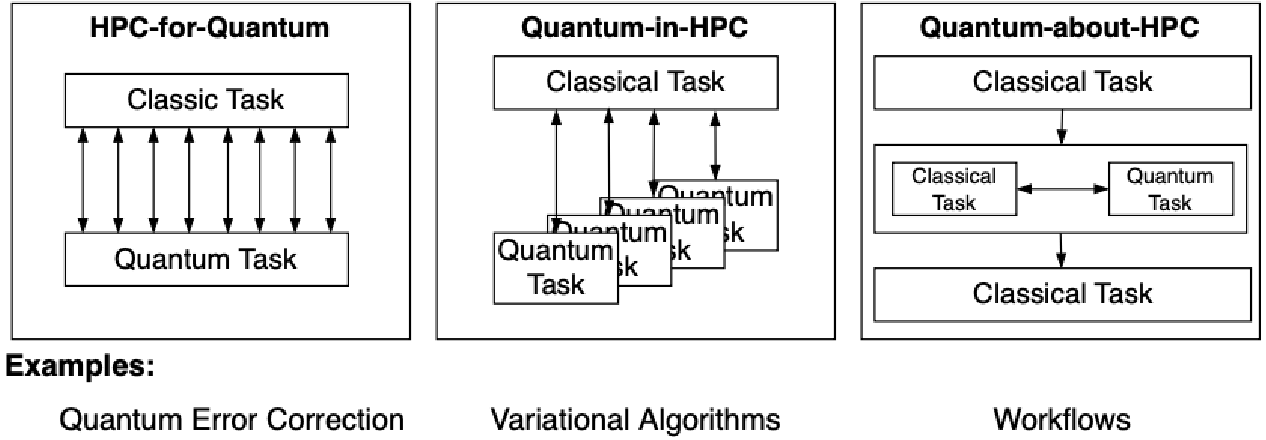
QUANTUM MINI-APPS: TOWARDS DESIGN, EVALUATION, AND REFINEMENT OF FRAGMENTED QUANTUM STACK.

- **Quantum Mini-Apps:** Simplified version of quantum-HPC applications encapsulating high-level execution patterns emulating unique workload patterns based on interaction between quantum-HPC components
- **Stages of Developing Quantum Mini-Apps:**
  - **First stage:** Identifying quantum-HPC integration strategies and coupling patterns by understanding interaction between classical HPC and quantum tasks and components
  - **Second stage:** Capturing fine-grained execution patterns/motifs, i.e. unique workflow configurations with combined coupling modes between classical and quantum components
  - **Third stage:** Creating quantum mini-apps with one or more high-level execution motifs



DEVELOPING EFFECTIVE INTEGRATION PATTERNS FOR QUANTUM-CLASSICAL APPLICATIONS THAT EXTRACT COMMON RECURRING INTERACTION MODES.

- **Integration Patterns:** Recurring interaction type for combining quantum and classical tasks in quantum computing applications
- **Characteristics:**
  - **Coupling:** Represents time sensitivity of interaction between quantum and classical tasks (tight, medium, or loose)
  - **Application Structure:** Refers to how an application utilizes different types of parallelism (ensemble, task, data, accelerators)

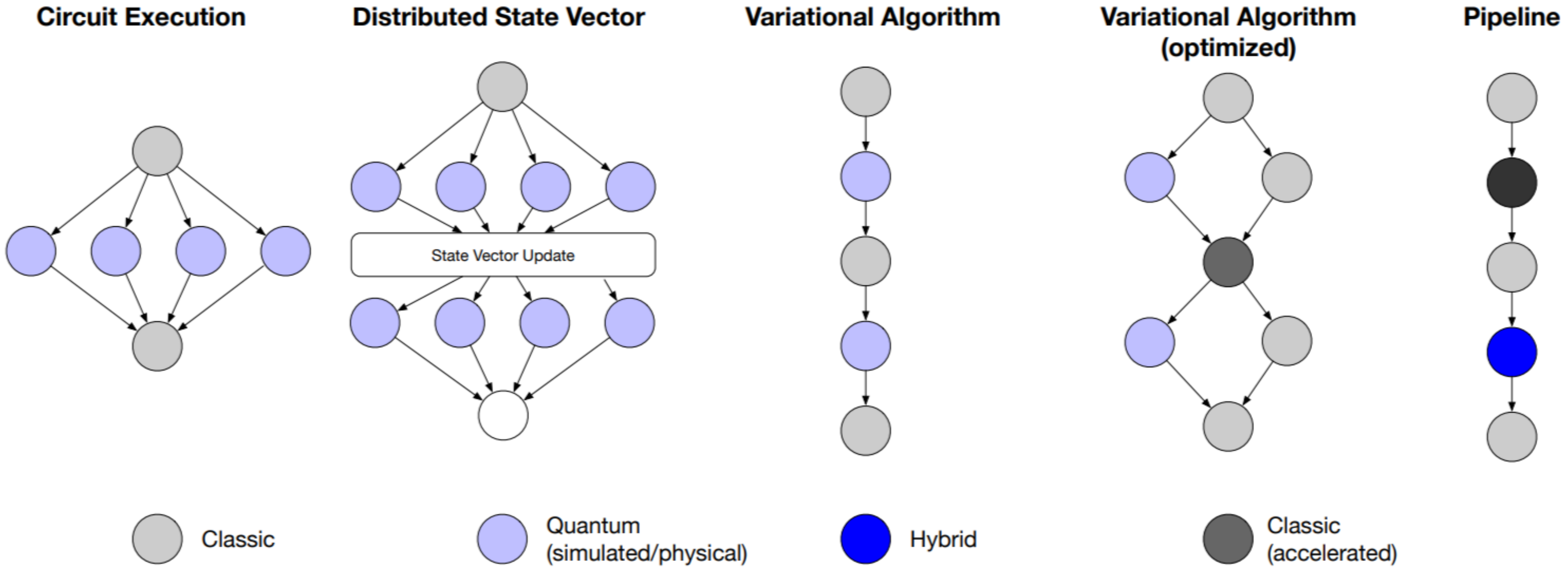


Saurabh, N., Jha, S., & Luckow, A. (2023, July). A conceptual architecture for a quantum-HPC middleware. In 2023 IEEE International Conference on Quantum Software (QSW) (pp. 116-127). IEEE.

Alexeev, Yuri, et al. "Quantum-centric Supercomputing for Materials Science: A Perspective on Challenges and Future Directions." arXiv preprint arXiv:2312.09733 (2023).

MOTIFS: CAPTURE EXECUTION PATTERNS THAT AID THE DESIGN OF THE MIDDLEWARE SYSTEM.

- **Motifs**: High-level execution patterns which enable capturing complex task structure of quantum-HPC applications and help identify integration opportunities with optimized HPC capabilities (e.g. GPU acceleration)
- **Six quantum-HPC core Motifs:** Circuit execution, distributed state vector simulation, circuit cutting, error mitigation, variational quantum algorithms (sequential and optimized), and multi-stage pipelines

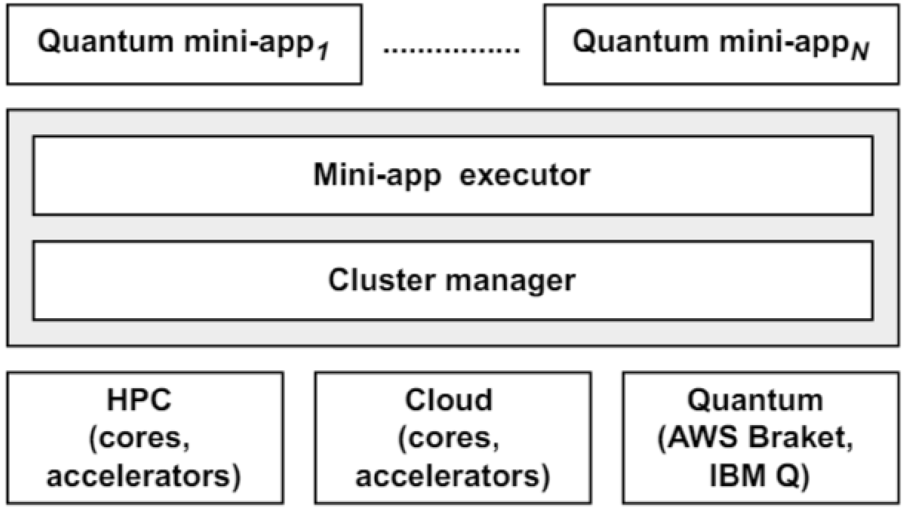


MOTIFS: CAPTURE EXECUTION PATTERNS THAT AID THE DESIGN OF THE MIDDLEWARE SYSTEM.

Motif	Interaction	Coupling	Example	Middleware Capabilities
Circuit Execution	Concurrent computation for different parameters	Loosely coupled, homogeneous tasks	Qiskit Estimator/ Sampler, Qiskit Aer with Dask [53]	Workload management across heterogeneous resources
Distributed State Vector Simulation	Concurrent computation of state vector updates & its synchronization across all tasks	Tightly coupled static and homogeneous tasks	Large-scale quantum simulations, e. g., cuQuantum [26]	Integration with HPC technologies, e. g., MPI, cuQuantum, ROCm
Circuit Cutting	Concurrent execution & reconstruction of tasks across QPUs.	Medium-to-tight coupling within/outside coherence time, task heterogeneity depends on circuit cutting algorithm	Qiskit Circuit Knitting Toolbox [6], Pennylane Circuit Cutting [66]	Optimal partitioning & placements of tasks/cuts on available (simulated) QPUs for balanced workload.
Error Mitigation	Loosely coupled execution of multiple circuit variants on PEs & results aggregation	Loosely coupled	Qiskit Runtime Error Mitigation [54]	Dynamic allocation & adaptation to right mix of classical & quantum resources
Variational Quantum Algorithms (VQA)	Concurrent execution of classical and quantum resources & Quantum components	Coupling outside coherence window for heterogeneous tasks, e. g., interleaved ML & QC parts sharing GPU	QAOA [27], VQE [51], VQE IS [59] QGAN [57], QuGen [55]	Collocate & balance quantum and classical resources
Multistage Pipelines	Encapsulated stages & its contained transitions with control & data flow	Heterogeneous & varying resource demands between stages	QML workflow MD workflows [16]	Optimized resource estimation for pipeline/individual stages, e. g., for dynamic resource pooling

QUANTUM MINI-APP FRAMEWORK: DESIGN OBJECTIVES, ARCHITECTURE AND EXAMPLE MINI-APPS.

- **Design Objectives:**
  - **Bridging Quantum hardware and software**: Characterizing quantum hardware constraints (i.e. coherence time, gate fidelity) for qubit control, error mitigation etc.
  - **Managing heterogeneity and complexity**: Estimating workload and resource patterns with evolving integration of QPUs with classical accelerators (e.g. FPGA)
  - **Scalability**: Providing a holistic approach to design and evaluation of quantum software with evolving quantum hardware.
- **Mini-App Examples:**
  - Quantum simulation Mini-App with circuit execution and distributed state vector motifs.
  - VQA Mini-App with interplay between parameterized quantum circuit, classical optimizer and cost function.
  - QML Pipeline Mini-App with multistage pipeline motifs



Source code: Quantum Mini-App framework implementation. https: //github.com/radical-cybertools/quantum-mini-apps.

CONCLUSION AND FUTURE WORK.

- **Quantum-HPC motifs** can provide a conceptual understanding of quantum application and workloads and serve as an underpinning for Mini-Apps . This work identified six of them.
- **Quantum Mini-App framework** offers a systematic approach to implement application kernels represented by motifs and benchmark performance and scalability of quantum applications.
- **Future work**
  - Implementing additional Mini-Apps representing six identified motifs.
  - Integrating application-oriented benchmarks with Quantum Mini-App framework
  - Developing a middleware system, i.e. pilot-quantum to simplify workload and resource management, investigate strategies to abstract allocation, management and optimization of heterogeneity in quantum stack using Quantum Mini-App framework .