Special Topics in Symbolic and Quantum Computation

Marek Rychlik

Course Description

This graduate-level course explores contemporary problems in *symbolic computation* and *quantum algorithms*, emphasizing the mathematical structures that enable algorithmic advances and efficient implementations. The course is project-oriented: students will propose and complete a substantial semester-long research project grounded in mathematical theory and supported by computational experimentation.

Two interconnected themes anchor the course:

1. **Symbolic reasoning and program synthesis** using logic-based methods (e.g., logic program-ming, constraint solving, and type-guided search).
2. **Algebraic and number-theoretic foundations of quantum algorithms**, including struc-tured implementations and high-performance simulation (including Shor-type methods and GPU-oriented approaches where appropriate).

Projects may include (non-exhaustive) topics such as type-driven program synthesis, constraint-based reasoning systems, algebraic improvements or analyses of quantum algorithm components, or optimized simulation of structured quantum circuits. The course combines theoretical development with algorithmic implementation, correctness analysis, and performance evaluation, with emphasis on reproducibility and research-style communication.

Topics to Be Covered

Core Topics

- Logic programming fundamentals; unification; constraint systems
- Type systems and type-guided search for program synthesis
- Formal specification; correctness principles for symbolic systems

- Fourier transforms over finite groups; modular arithmetic foundations
- Algebraic structure in Shor-type algorithms and related subroutines
- Complexity considerations for structured algorithms
- Principles of high-performance simulation (including GPU-oriented computation where appro-priate)
- Evaluation methodology: baselines, metrics, ablations, failure analysis; reproducibility practices

Advanced / Project-Dependent Topics

- Search-space reduction strategies; constraint propagation; synthesis evaluation protocols
- Circuit resource estimation; decomposition strategies; numerical stability and precision tradeoffs
- Accelerated linear-algebra kernels and structured transformations for simulation
- Verification and testing strategies tailored to symbolic and quantum computational artifacts

Texts (if relevant)

There is no single required textbook. Readings will be drawn from research papers and selected references appropriate to the chosen project tracks. Representative references may include:

- M. Nielsen and I. Chuang, *Quantum Computation and Quantum Information*.
- J. W. Lloyd, *Foundations of Logic Programming* (or comparable Prolog/logic programming ref-erence).
- Selected research articles in program synthesis, symbolic reasoning, and quantum algorithms.
- Selected materials on high-performance computing and GPU architectures relevant to simulation projects.

Prerequisites

Graduate standing in Mathematics or a closely related field. Students should have preparation in:

- Linear algebra (including eigenvalues and Fourier transforms)
- Abstract algebra (groups and rings)
- Mathematical maturity at the graduate level
- Prior programming experience (e.g., Python, C/C++, Julia, or similar)

Background in quantum mechanics, logic programming, or machine learning is helpful but not required.

Learning Outcomes

By the end of the course, students will be able to:

1. Formulate mathematically precise problems in symbolic or quantum computation.
2. Design algorithms grounded in algebraic or logical structure.
3. Analyze correctness and computational complexity of structured methods.
4. Implement and test algorithms using appropriate computational tools and best practices.
5. Evaluate performance and limitations using principled experimental methodology.
6. Produce a research-style written report with clear mathematical exposition and reproducible results.
7. Present technical results in a seminar-style oral presentation.

Approximate Schedule (15 Weeks)

Weeks 1–2: Course overview; research workflow expectations; foundations of symbolic computa-tion (unification, constraints); overview of quantum algorithms and algebraic structure; project track selection.

Weeks 3–4: Type systems and type-guided search for synthesis; specification and correctness; Fourier transforms over finite groups; modular arithmetic review.

Weeks 5–6: Advanced symbolic reasoning (constraint propagation, synthesis strategies); algebraic structure in Shor-type algorithms and related subroutines; evaluation design (baselines, metrics).

Weeks 7–8: High-performance simulation principles (data layout, computational kernels, preci-sion/stability considerations); **project proposals** and feedback; milestone planning.

Weeks 9–11: Project development in track-specific seminars; individual consultations; **Mile-stone 1:** baseline implementation + theoretical justification.

Weeks 12–13: Performance analysis and refinement; ablations and failure analysis; **Milestone 2:** working system with preliminary results and evaluation.

Weeks 14–15: Final project presentations; submission of final research-style report and repro-ducible artifact (code, experiments, documentation).

Notes: This proposal is intended as a graduate special topics offering. Enrollment will be deter-mined by student interest; the course structure supports a range of project directions within the unifying theme of symbolic and quantum computation.