Chengyuan Ma

Email: `MaCY404@mit.edu`
Github: `https://github.com/ma-chengyuan`
Cambridge, MA 02139-4306

Education

- **Massachusetts Institute of Technology** (expected graduation May 2025) 2021–Present
  - BSc and MEng in Computer Science and Engineering. GPA: 5.0
  - Courses taken in major: Distributed Systems, Compiler, Software Performance Engineering, Cryptography, Theory of Computation, Computer System Engineering, Advanced Algorithms, Large-scale Symbolic Systems, Software Construction, Artificial Intelligence, Dynamic System Modelling and Design, Design and Analysis of Algorithms, Computation Structures, Fundamentals of Programming, Battlecode Programming Competition.

Skills

- **Programming**: Rust, Python, C++, TypeScript & JavaScript, Go, KDB/q, C# (with Unity), Java, Scheme, Kotlin, Julia, RISC-V. (ordered by approx. decreasing proficiency and recent usage)
- **Hardware**: Raspberry Pi, Arduino.
- **Software**: Unity, TEX, Vim, JetBrains IDEs, Linux, Mathematica, MATLAB, Git, Inkscape.
- **Languages**: Mandarin, English.

Experiences

- Global Quantitative Strategies, **Citadel LLC**, Chicago Jun–Aug 2024
  Quantitative research engineer intern
- MIT Computation Structures Group, **Zero-knowledge Proof Generation with WebGPU**, Cambridge Jan 2024–Present
  Undergraduate researcher
  - Research GPU-accelerated zk-SNARK generation in browser with portable performance across GPU vendors.
  - Implemented CPU-GPU co-computed browser-side multi-scalar multiplication (Rank 4 of ZPrize 2023).
- Global Quantitative Strategies, **Citadel LLC**, New York Jun–Aug 2023
  Quantitative developer intern
- MIT d'Arbeloff Lab, **Rehabilitation Therapy with Collaborative Robot and Mixed Reality**, Cambridge Oct 2022–May 2024
  Undergraduate researcher
  - Developed Mixed Reality app with HoloLens to improve approachability of robotic therapy by projecting avatar overlay.
  - Programmed the ABB Yumi dual-arm collaborative robot to perform trajectory control in rehabilitation therapy.
  - Proposed an Optimal Non-Iterative Alignment solver aligning the virtual avatar with the robot and an end-to-end pipeline.
  - Authored two papers (accepted by IEEE IROS'23 and RSS'24).
- InfoLab, MIT Computer Science and AI Lab (CSAIL), **Social Interactions in Robotics**, Cambridge Feb–Oct 2022
  Undergraduate researcher
  - Developed GPU-accelerated recursive MDP solver with 1000x speedup with CUDA (paper accepted by AAAI'23).
  - Experimented with RL algorithms such as DQN, VPG, and PPO as alternative implementations of lab models.
  - Adapted a Unity environment (VirtualHome) for the lab to experiment multi-agent decisions.

Activities

- **Competitive Programmer** 2016–2020, Jun 2022
  - *3rd Place* at First-year Team Tournament of MIT Battlecode Competition in 2022. Led team to develop distributed and collaborative multi-agent AI to accomplish strategic goal real-time under tight computational resource constraints.
  - Developed proficiency in advanced data structures and algorithms, both in terms of analysis and implementation in C++.
  - *Bronze medalist* at National Olympiad in Informatics (NOI), the top-level competition in China, in 2020.
  - *First Prize and Top 3%* at NOI (Provincial) in Shanghai for four years in a row from 2016—2020.
  - *Platinum-level contestant* at USA Compouting Olympiad (USACO) from 2018—2020.
- **Coordinator and TA**, Program in Algorithmic and Combinatorial Thinking (PACT), Princeton / Virtual 2018, 2020–Present
  - Coordinate and organize marketing, admission and logistics of PACT China (2021–22) and PACT Asia (2023–24), which are intense summer programs teaching curious high school students the mathematical foundations of theoretical CS.

Personal Programming Projects

1. **DormSoup — A full-stack large-language-model-powered event platform for MIT** (TypeScript, 2023, `dormsoup.mit.edu`): Use language models to parse, summarize, and catalog MIT community events from diversely formatted "dorm spam" emails, improving event information coverage and engagement. 800 total users.
2. **Web-Scheme — A Scheme compiler with a WebAssembly based runtime** (Scheme + C++, 2022): Performs hygienic syntactic expansion, CPS transform, and tail-call-optimized code generation. Compiles itself. Runtime uses mark-and-sweep GC.
3. **Just-LATEX — Painless true LATEX-based equation rendering for Markdown $\to$ HTML** (Rust, 2022): Extracts and replaces LATEX from Markdown with SVGs rendered by the true LATEX engine. More powerful than MathJax and easy to integrate in various workflows. Has SVG optimizer based on B-Tree-in-Trie for efficient path deduplication, achieving 80% reduction in SVG size.
4. **NTTP — An obfuscated SOCKS5 client and server** (Go, early 2019): Uses mod-257 Number Theoretic Transform to obfuscate

SOCKS5 proxy traffic to circumvent Chinese national Internet censorship and proxy blockage (GFW). Peak throughput around ~30MiB/s. Ineffective in early 2020 as the adversary upgraded to more sophisticated probing strategies.

5. **Ekho — An ICMP-based SOCKS5 proxy** (Rust, early 2021): A sequel to NTTP. Uses KCP over ICMP to ensure reliable transmission and circumvent national firewall's probing and blockage on TCP and UDP. I re-implemented KCP from scratch, together with congestion controllers such as BBR and PCC. Never blocked.
6. **Particle — A lexer generator** (Rust, early 2019): Generates efficient lexers from regular expression by performing reduction to DFA. Flexible and declarative grammar to define lexers on user-defined token types. Location information is preserved in tokenization. Supports Unicode. Allows dumping DFA in Graphviz format for visualization and debugging.
7. **A web app dewarping book pages** (JavaScript & WASM, early 2020): Similar to CamScanner and Office Lens, but can handle curved edges of pages. Automatically calculates perspective parameters from user-defined control points, fits cubic splines to edges, and produces de-warped rectangular page. All computation takes place on the browser side using WebAssembly.
8. **An automatic differentiation framework** (C++, late 2019): A naïve version of TensorFlow. Constructs static computation graphs and uses reverse accumulation to calculate gradients. Supports matrix arithmetic, softmax, dropout, and common activation functions; Easy to define new operators and optimizers. Performs well on MNIST and simple networks.
9. **An interpreter for a homemade scripting language** (C++, mid 2018): A functional and minimal language, supporting basic control flows, lexical-scoped closures, and arbitrary-precision arithmetic. Hand-written recursive-descent parser with robust error recovery and decent error messaging. Also has a GUI to visualize AST for debugging purposes.

I enjoy learning through various personal programming projects. More on `https://github.com/ma-chengyuan`.