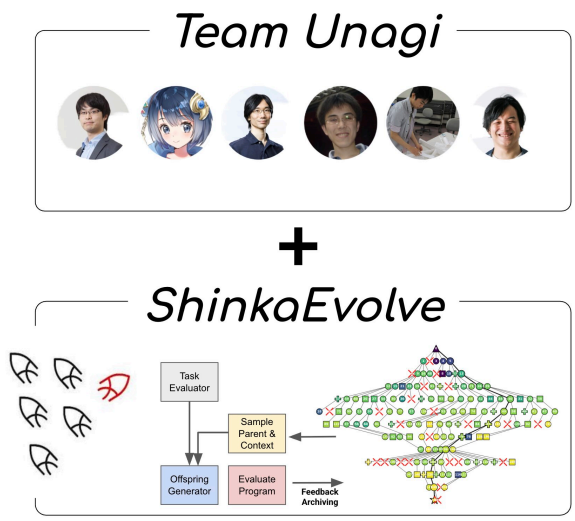
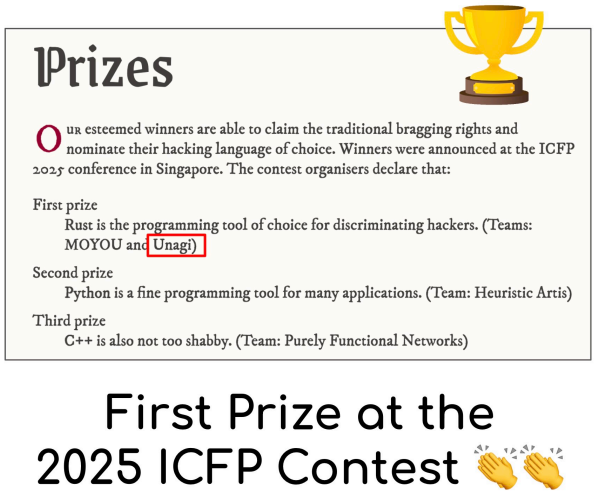
sakana.ai

ShinkaEvolve in Action: How a Human-AI Partnership Conquered a Coding Challenge

October 16, 2025





The author of this article, Takuya Akiba, is a research scientist at Sakana AI.

Summary

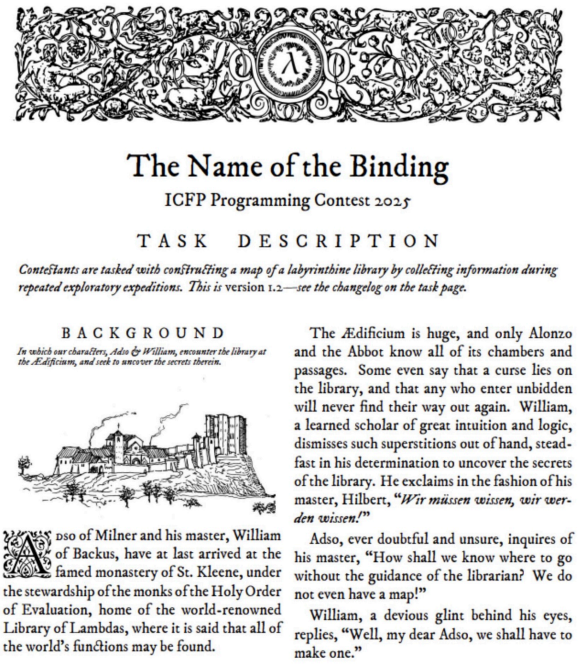
Long before I joined Sakana AI as a founding Research Scientist, I was captivated by the world of competitive programming, and have been active for years with my friends in **Team Unagi**. This long-standing passion and teamwork recently culminated in a first place finish at the **2025 ICFP Programming Contest**. This result was driven by a particularly special collaboration: one between our human team and the AI system I now help develop.

This post shares a case study on how we applied **ShinkaEvolve**, Sakana AI's evolutionary code optimization framework, to significantly improve our solution's performance on a complex task. Specifically, we provided ShinkaEvolve with our team's manually-written code, and it used large language models to iteratively evolve that code to minimize solver execution time. By automatically optimizing the SAT encoding at the heart of our approach, ShinkaEvolve accelerated our solver by up to 10x. This not only enabled us to tackle large-scale problems that were previously out of reach but also

created a virtuous cycle where insights discovered by the AI directly fed back into and enhanced our human development process.

The Challenge: An 'Anything Goes' Programming Contest

The ICFP Programming Contest (ICFP-PC) is an international competition known for its creative freedom. With no restrictions on team size, programming languages, or the use of external tools like AI, the contest is a true test of multifaceted problem-solving. This year's task was to navigate and map an unknown maze using a series of ambiguous hints. The objective was to determine the maze's complete structure with the fewest queries possible, demanding a strategy that could extract maximum information from limited clues.



The Name of the Binding

ICFP Programming Contest 2025

TASK DESCRIPTION

Contestants are tasked with constructing a map of a labyrinthine library by collecting information during repeated exploratory expeditions. This is version 1.2—*see the changelog on the task page.*

BACKGROUND

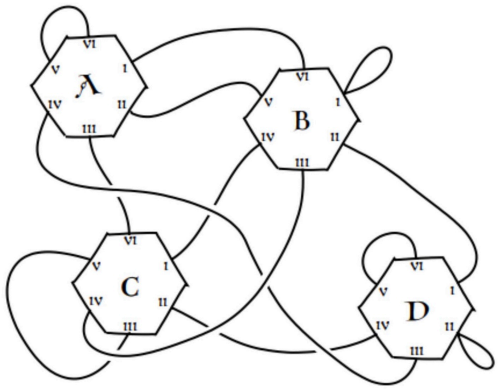
In which our characters, Adso & William, encounter the library at the Ædificium, and seek to uncover the secrets therein.

Adso of Milner and his master, William of Backus, have at last arrived at the famed monastery of St. Kleene, under the stewardship of the monks of the Holy Order of Evaluation, home of the world-renowned Library of Lambdas, where it is said that all of the world's functions may be found.

The Ædificium is huge, and only Alonzo and the Abbot know all of its chambers and passages. Some even say that a curse lies on the library, and that any who enter unbidden will never find their way out again. William, a learned scholar of great intuition and logic, dismisses such superstitions out of hand, steadfast in his determination to uncover the secrets of the library. He exclaims in the fashion of his master, Hilbert, "*Wir müssen wissen, wir werden wissen!*"

Adso, ever doubtful and unsure, inquires of his master, "How shall we know where to go without the guidance of the librarian? We do not even have a map!"

William, a devious glint behind his eyes, replies, "Well, my dear Adso, we shall have to make one."



An example of a 4-room Ædificium.

ICFP Programming Contest 2025 Task description (version 1.2): https://icfpcontest2025.github.io/specs/task.pdf

Our Strategy: Solving with SAT Encoding

From our initial analysis, we identified that the problem could be solved with very few queries if we could leverage all available information perfectly. This led us to an approach centered on a SAT (Boolean Satisfiability) solver. By encoding all constraints and observations into a single logical formula, we could use the solver to find a valid solution that satisfied all conditions simultaneously.

While powerful, a SAT solver's performance is critically dependent on the quality of the encoding. A good encoding makes the problem's structure clear to the solver, while a poor one can make even a simple problem computationally intractable. Crafting an efficient encoding is an art, requiring deep insight and significant trial and error.

$$(\bar{x_1} \vee x_2 \vee x_3) \wedge (x_1 \vee \bar{x_3}) \wedge (\bar{x_2} \vee x_3) \wedge (x_1 \vee x_2 \vee \bar{x_3})$$
$$(x_1 \vee x_2) \wedge (\bar{x_1} \vee x_2) \wedge (x_1 \vee \bar{x_2}) \wedge (\bar{x_1} \vee \bar{x_2})$$

Examples of logical formulae. The SAT problem is to find a variable assignment that satisfies these formulae.

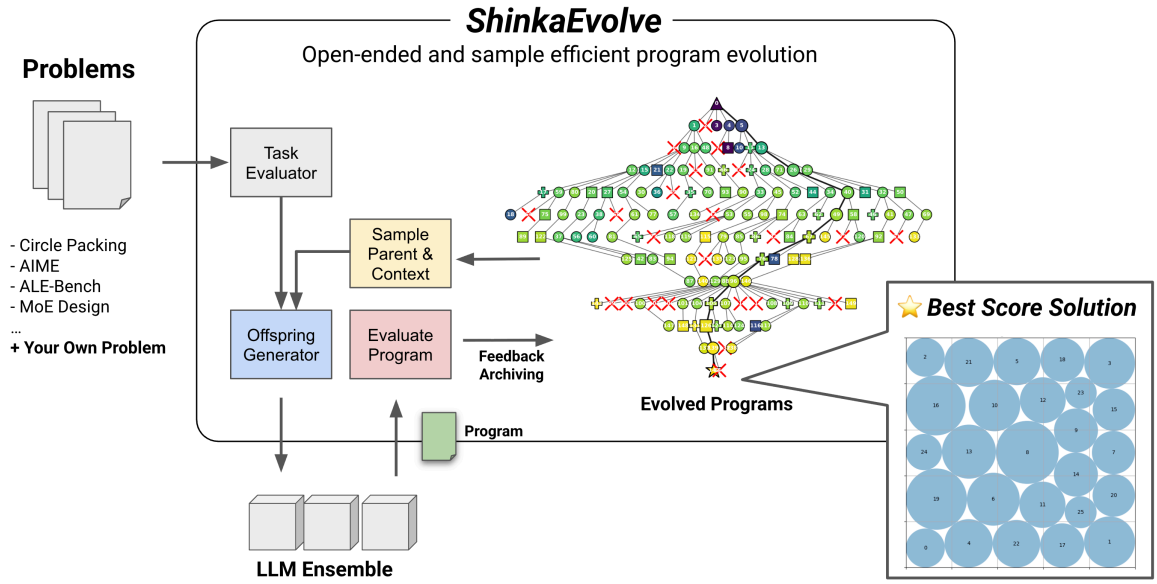
Simply making the formula smaller isn't always the answer; sometimes, adding well-designed "auxiliary variables" can guide the solver's search and dramatically speed up the process. A team member developed a highly effective initial encoding, but it struggled with the computation time on larger problem instances, creating a significant bottleneck.

Applying ShinkaEvolve for Optimization

To break through this performance barrier, we turned to ShinkaEvolve. We tasked it with optimizing the Rust code of our SAT encoding, using the solver's execution time as the fitness function to minimize.

ShinkaEvolve is an open-source framework we developed at Sakana AI to *evolve* and optimize the code of challenging algorithms, powered by an ensemble of large language models (LLMs). As detailed in our previous posts, it has proven effective on a variety of challenging problems.

- 📝 Blog: ShinkaEvolve: Evolving Large Language Models with Genetic Algorithms
- 📄 Paper: ShinkaEvolve: A Large Language Model-driven Framework for Algorithm Evolution
- 👨‍💻 Code: Open Source Github Repository



The ShinkaEvolve framework constructs an archive of evaluated programs, generates new programs, and evaluates their fitness.

Trial & Error by ShinkaEvolve

Using our team's manually-written code as a starting point, ShinkaEvolve began the optimization process.

- Experiment setting: https://github.com/icfpc-unagi/icfpc2025/blob/main/evo/exp2/main.py
- Prompt: https://github.com/icfpc-unagi/icfpc2025/blob/main/evo/exp2/prompt.md

It performed 320 trials, with the total computational cost for the experiment being around $60.

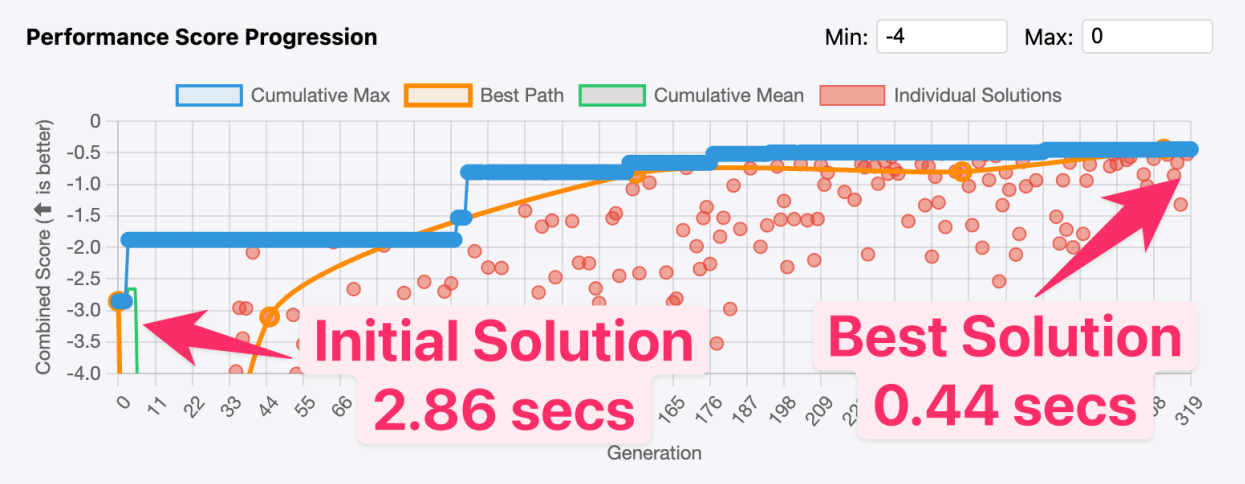
Scores improve with each generation, with several significant breakthroughs visible along the way.

Results: A 10x Speedup and New Capabilities

The optimization from ShinkaEvolve yielded substantial performance gains:

- **Mid-scale problem (18 rooms):** Execution time improved from 2.86s to 0.44s (a **~6.5x** speedup).
- **Large-scale problem (24 rooms):** Execution time dropped from 127s to 13s (a **~10x** speedup).

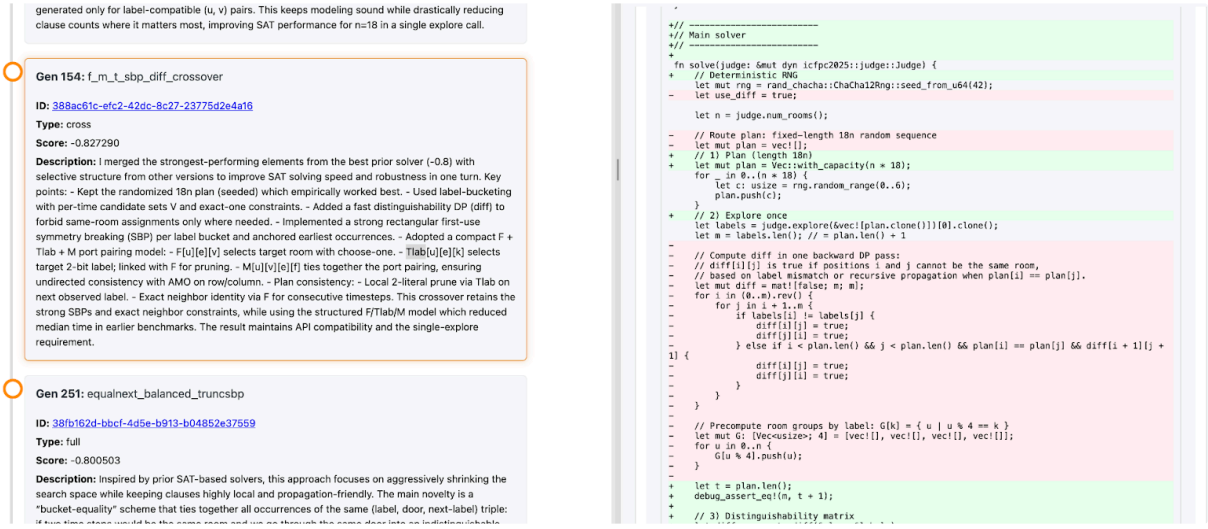
This speedup was a breakthrough. It effectively transferred to even larger cases, allowing us to solve 30-room instances within a realistic timeframe—something that was previously impossible. This new capability was immediately integrated into our submissions, significantly contributing to our team's score.



The solution's evolution is visualized via the ShinkaEvolve interactive web interface.

AI Discovery Gives Hint to Human Insight

Beyond the performance metrics, one of the most compelling outcomes was how ShinkaEvolve's findings provided valuable insights that guided our own development. The AI's improvements were not an opaque black box.



An example of the code changes proposed by ShinkaEvolve. The box on the left contains some of the ideas that led to the breakthrough in our approach.

ShinkaEvolve generated several key improvements, with the most impactful being a change in how the maze's topology was represented. Our original encoding used a direct representation: "Door 1 of Vertex A connects to Door 2 of Vertex B." ShinkaEvolve discovered a more abstract, intermediate representation by adding an auxiliary variable that meant: "Door 1 of Vertex A first connects to Vertex B."

This change allowed the solver to focus on the higher-level decision of "which vertices are connected" before getting into the details of "which doors are connected," making the search far more efficient. This core idea was a generalizable principle. Our team later successfully applied this very concept manually when designing a solver for different problems.

Conclusion: A Collaborative Model for Problem-Solving

While our success was the result of the entire team's hard work, from designing the overall approach to implementing various heuristics across multiple areas and building efficient testing infrastructure, this case study highlights the powerful role that AI can play in complex software optimization.

It demonstrates a highly effective workflow where humans define the overall strategy and create a strong baseline, while AI performs targeted, intensive searches for improvements within that framework. The insights generated by the AI are then interpreted by humans and reapplied to new challenges. This collaborative and complementary relationship between human and AI expertise points toward a powerful and productive future for research and problem-solving.

Acknowledgements. I would like to thank the organizers of the 2025 ICFP Programming Contest for a fantastic and challenging event. This achievement was a genuine team effort, and I want to acknowledge the incredible work of all my teammates in Team Unagi. I would also like to thank my colleagues at Sakana AI who developed this powerful ShinkaEvolve framework.