Théo Matricon

2025–current

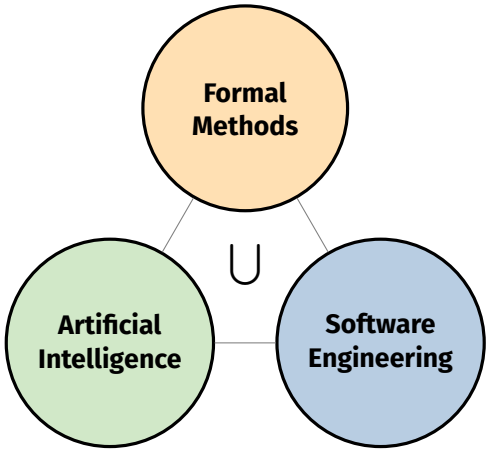
Postdoc with Mathieu Acher
Code Generation, Reproducibility
DiverSE, Software Engineering team
IRISA, Rennes

2021–2024

PhD supervised by Nathanaël Fijalkow
Scaling domain agnostic techniques for program synthesis
M2F, Formal Methods team
LaBRI, Bordeaux

Medical Condition (RQTH)

Research Contributions

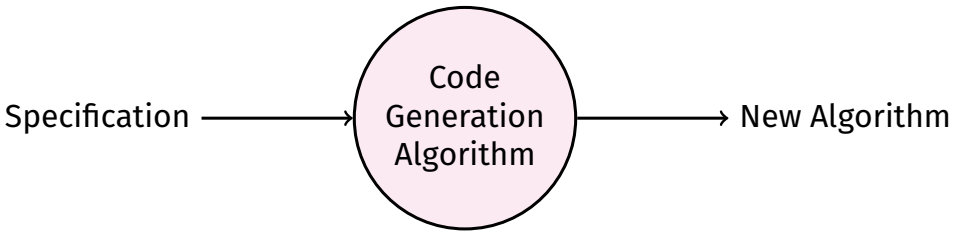


Selected Publications

- 2026, 2 TOSEM ($Q1$) accepts with major revisions
- 2026, MSR (A)
- 2025, TACAS (A)
- 2025, AAAI (top 20%) (A^*)
- 2025, IST ($Q1$)

  Postdoc
- 2022, AAAI (top 20%) (A^*)

PhD Motivation



PhD Motivation: In Practice

Specifications

Logic

$$\forall a, b$$
$$f(a, b) \geq a$$
$$f(a, b) \geq b$$
$$f(a, b) \in \{a, b\}$$

Examples

$$f(1, 5) = 5$$
$$f(2, 1) = 2$$
$$f(-3, -9) = -3$$

Natural Language

'Write a function that takes the maximum of its two arguments.'

Produced Algorithm

```
def max(a: int, b: int) ->int:
    if a <=b:
        return b
    else:
        return a
```

Program Synthesis: Problem

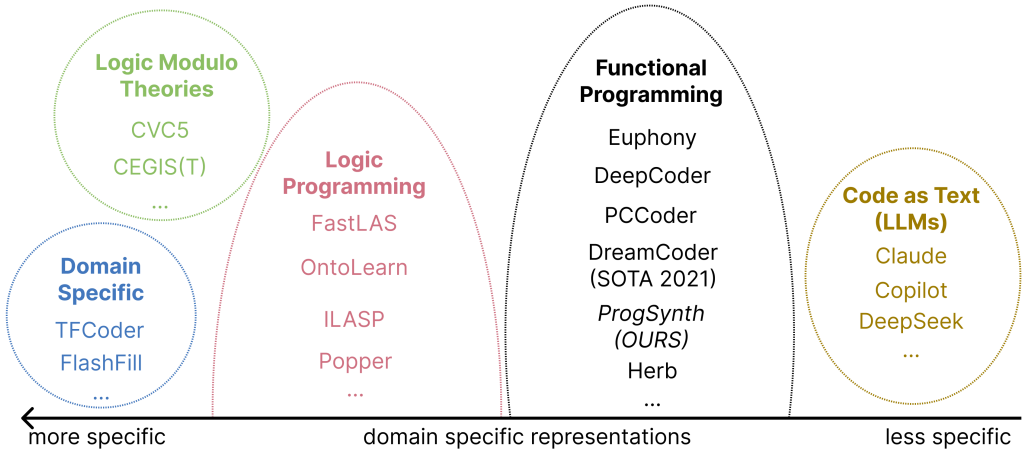
Input

- the search space G:
  a deterministic tree grammar
- a specification $\mathcal{C}$:
  checks if a program $p \in \mathcal{L}(G)$ matches the specification

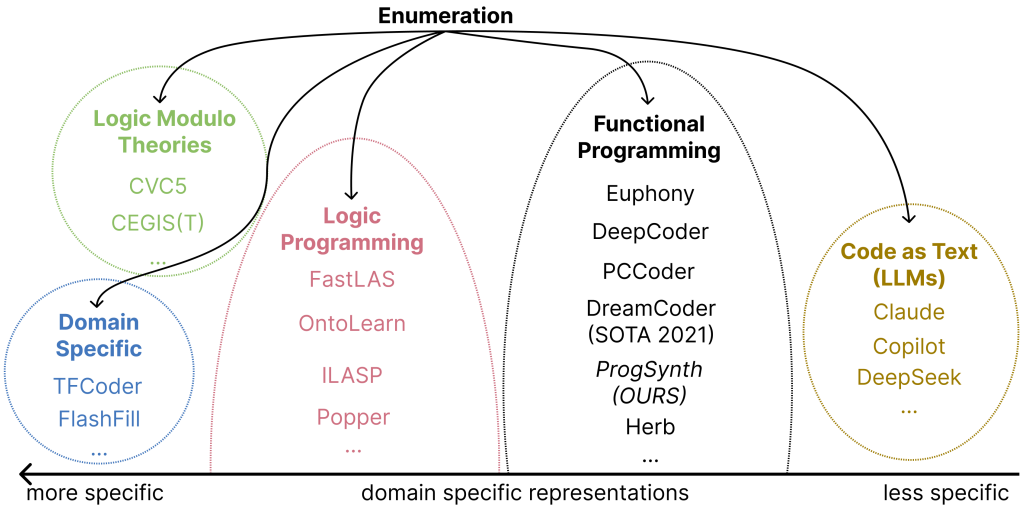
Output

- a program in the search space that matches the specification:
  $p \in \mathcal{L}(G)$ such that $\mathcal{C}(p) = \checkmark$

Program Synthesis: Frameworks



Program Synthesis: Frameworks



Program Synthesis: Enumeration Problem

Input

- the search space G:
  a deterministic tree grammar with a cost for each tree
- a specification $\mathcal{C}$:
  checks if a program $p \in \mathcal{L}(G)$ matches the specification

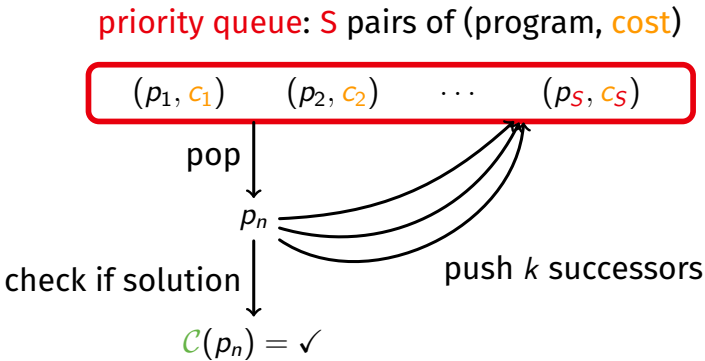
Goal

- enumeration of programs in order of non-decreasing costs

Delay

- time complexity between enumeration of two successive programs
  in terms of number of enumerated programs

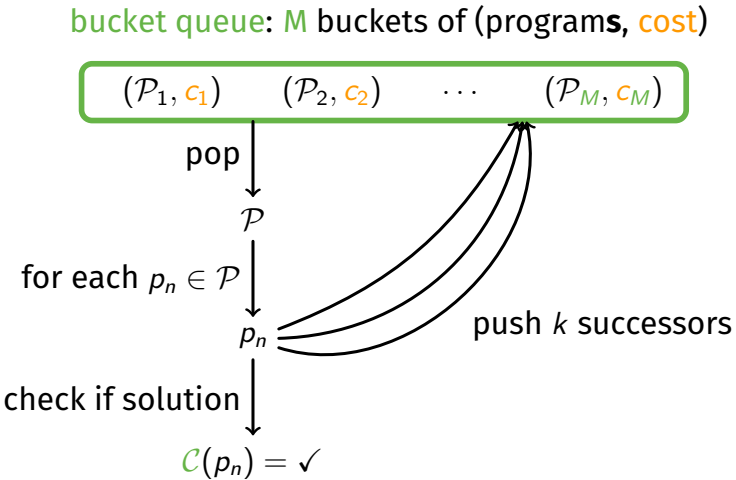
Enumeration Algorithm Before Our Contribution



At step n

$S = O(n)$
$k = O(1)$

Our Contribution: Constant Delay



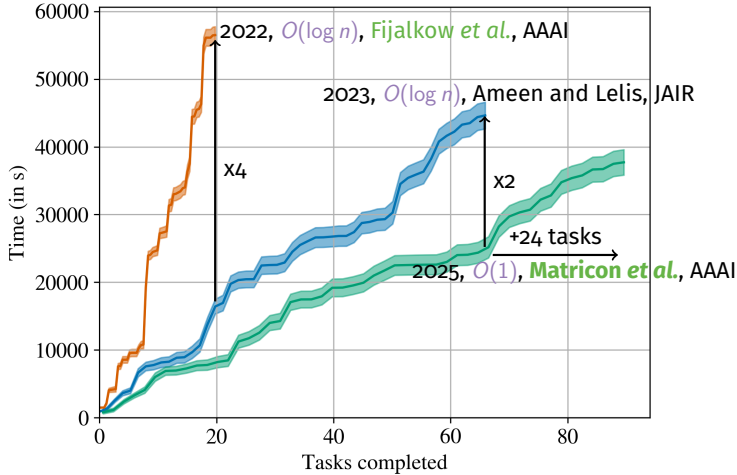
At step n

$M = O(1)$
$k = O(1)$

Our Contribution: Empirical Results

Major Papers

- 2017, ML + Enum., Balog *et al.*, ICLR
- 2018, $O(\log n)$, Lee *et al.*, PLDI



Our Contribution: Summary

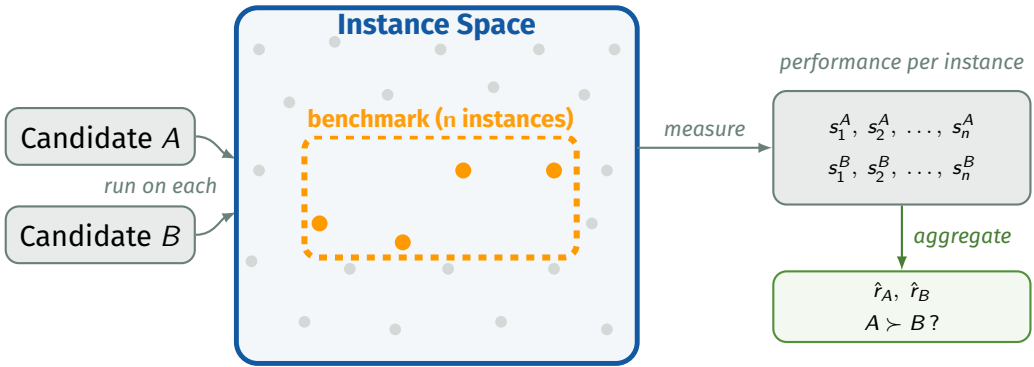
We prove bounded differences in cost:
$\exists M,\ \forall n,\ cost\_next(p_n) - cost(p_n) \leq M$

This implies: priority queues $O(\log n)$ $\rightarrow$ bucket queues $O(1)$

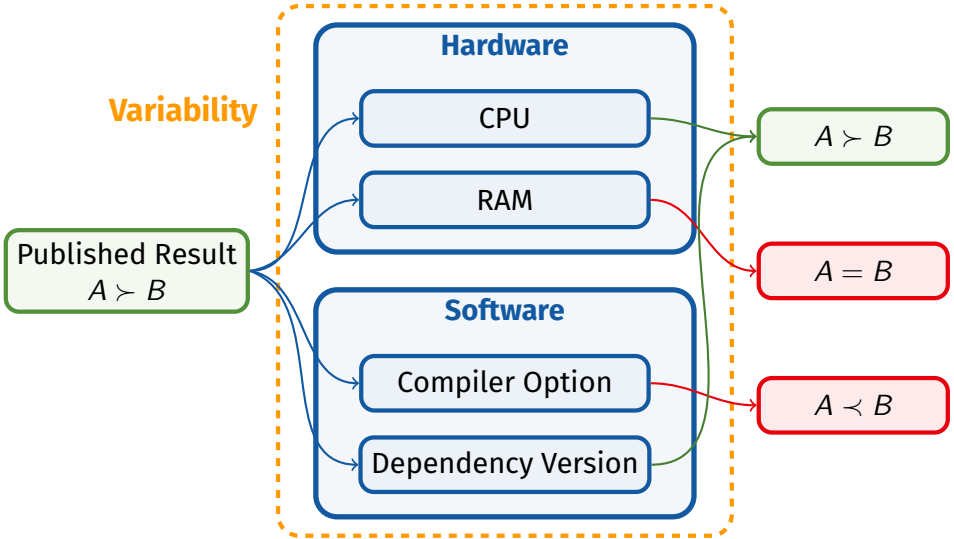
Impact

Published in **AAAI 2025** (+oral: 20% of accepted papers).
Fastest ranked enumeration for program synthesis **in practice**.
First algorithm with $O(1)$ delay $\rightarrow$ closes open question.

Evaluations



Evaluations: In Practice



My Research Project

Observations

- Exponential growth in scientific publications **[1]**
- Reproducibility crisis in computational science
- Evaluations are a key aspect of reproducibility

[1] Bornmann and Mutz, "Growth rates of modern science: A bibliometric analysis based on the number of publications and cited references", JAIST

My Research Project

Observations

- Exponential growth in scientific publications **[1]**
- Reproducibility crisis in computational science
- Evaluations are a key aspect of reproducibility

How to design reliable and efficient evaluations?

Reliable $\rightarrow$ formal guarantees

Efficient $\rightarrow$ reduce cost of evaluations

[1] Bornmann and Mutz, "Growth rates of modern science: A bibliometric analysis based on the number of publications and cited references", JAIST

My Research Project

Observations

- Exponential growth in scientific publications **[1]**
- Reproducibility crisis in computational science
- Evaluations are a key aspect of reproducibility

How to design reliable and efficient evaluations?

Reliable $\rightarrow$ formal guarantees

Efficient $\rightarrow$ reduce cost of evaluations

Main Challenges

SE Modelling the huge hardware and software variability

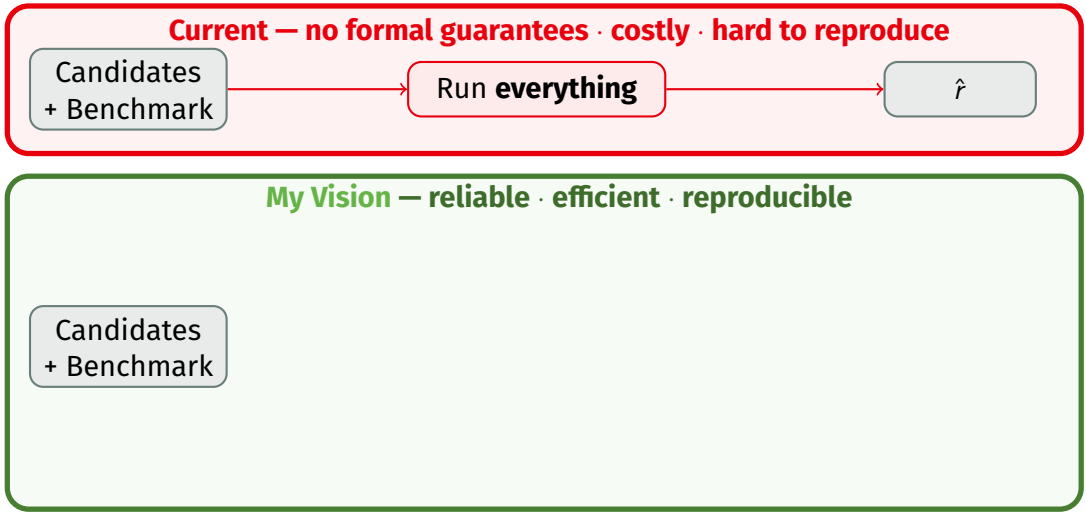
AI Predicting heterogeneous evaluation data

FM Giving formal guarantees while still being efficient

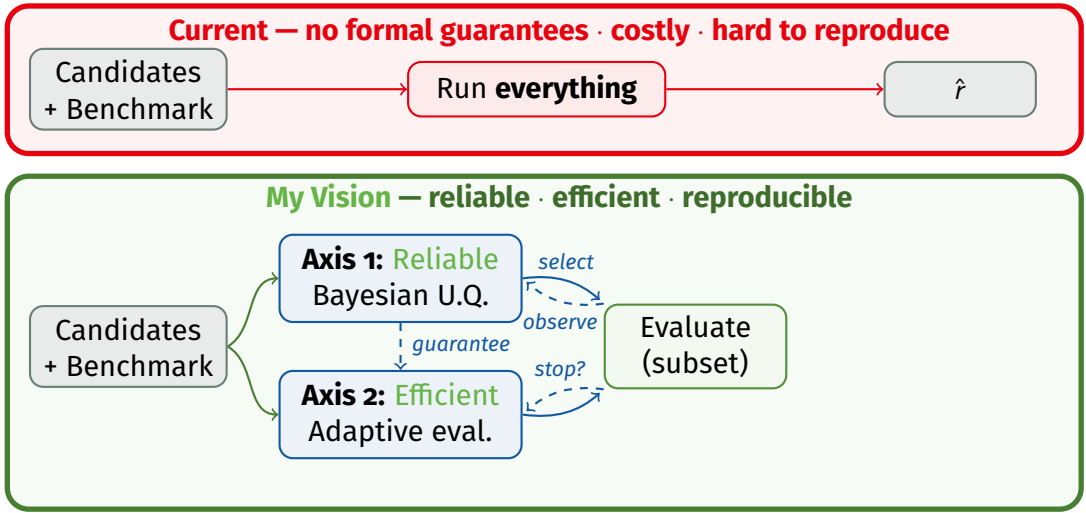
[1] Bornmann and Mutz, "Growth rates of modern science: A bibliometric analysis based on the number of publications and cited references", JAIST

My Research Project

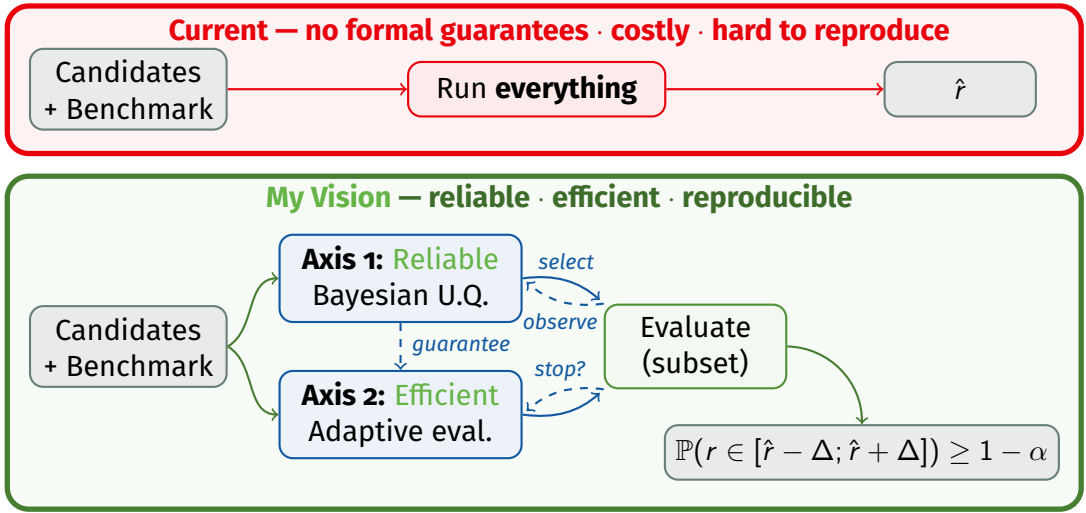
My Research Project



My Research Project



My Research Project



Classic Approach vs. Our Approach

Classic Approach

What is evaluated

Run all n instances — exhaustively

Our Approach

What is evaluated

Adaptive subset + early stopping
(∼50% fewer runs **[1]**)

[1] Matricon *et al.*, "Efficiently ranking software variants with tiny benchmarks" under revision at TOSEM

Classic Approach vs. Our Approach

Classic Approach

What is evaluated

Run all n instances — exhaustively

Statistical framework

assumes i.i.d. & normality

Our Approach

What is evaluated

Adaptive subset + early stopping
(∼50% fewer runs **[1]**)

Statistical framework

Adjustable complexity

[1] Matricon *et al.*, "Efficiently ranking software variants with tiny benchmarks" under revision at TOSEM

Classic Approach vs. Our Approach

Classic Approach

What is evaluated

Run all n instances — exhaustively

Statistical framework

assumes i.i.d. & normality

Variability

Single environment assumed
results may not transfer

Our Approach

What is evaluated

Adaptive subset + early stopping
(∼50% fewer runs **[1]**)

Statistical framework

Adjustable complexity

Variability

Multi-environment hierarchical model
quantifies generalizability

[1] Matricon *et al.*, "Efficiently ranking software variants with tiny benchmarks" under revision at TOSEM

Classic Approach vs. Our Approach

Classic Approach

What is evaluated

Run all n instances — exhaustively

Statistical framework

assumes i.i.d. & normality

Variability

Single environment assumed
results may not transfer

Reproducibility

binary: reproduced or not

Our Approach

What is evaluated

Adaptive subset + early stopping
(∼50% fewer runs **[1]**)

Statistical framework

Adjustable complexity

Variability

Multi-environment hierarchical model
quantifies generalizability

Reproducibility

graded: exact → ordinal → trend

[1] Matricon *et al.*, "Efficiently ranking software variants with tiny benchmarks" under revision at TOSEM

Axis 1: Reliable Evaluations

Variability

CPU-A • GCC-12 • Linux

$\hat{r}_A = 0.82 \quad \hat{r}_B = 0.75 \quad A \succ B$

CPU-B • GCC-14 • Linux

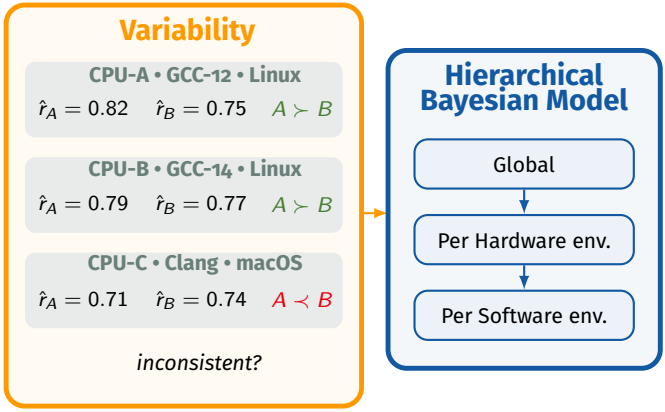
$\hat{r}_A = 0.79 \quad \hat{r}_B = 0.77 \quad A \succ B$

CPU-C • Clang • macOS

$\hat{r}_A = 0.71 \quad \hat{r}_B = 0.74 \quad A \prec B$

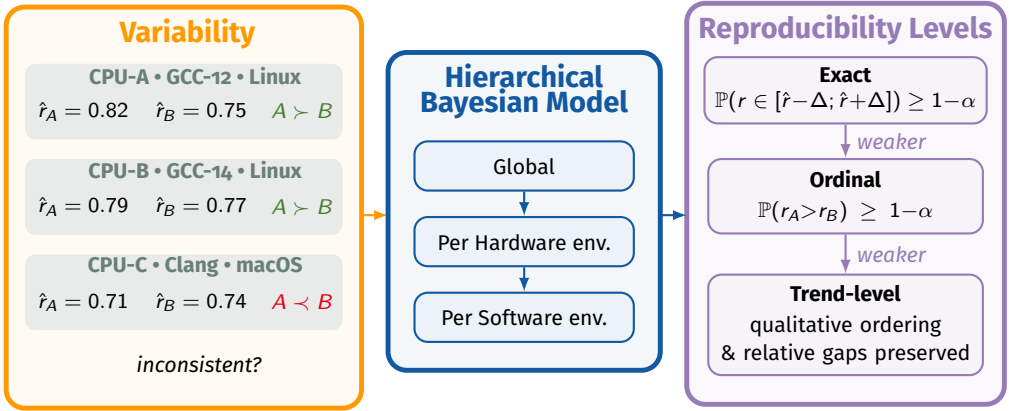
inconsistent?

Axis 1: Reliable Evaluations



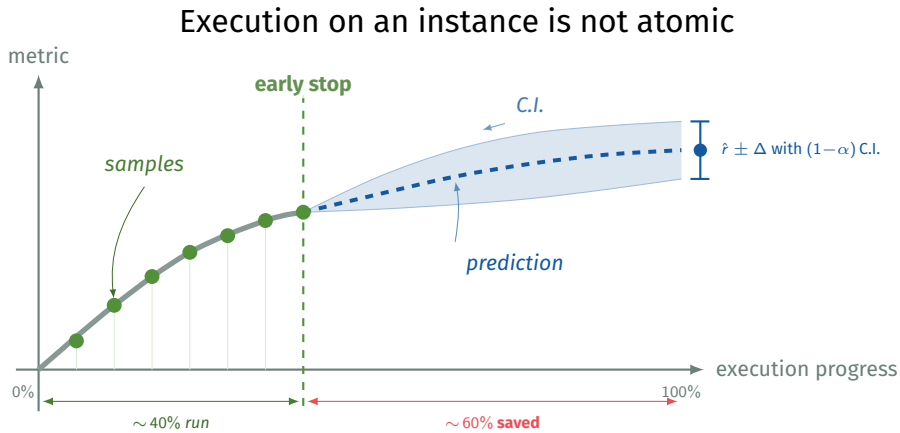
Ernst, "Bayesian hierarchical modelling for tailoring metric thresholds", MSR, 2018

Axis 1: Reliable Evaluations



Ernst, "Bayesian hierarchical modelling for tailoring metric thresholds", MSR, 2018

Axis 2: Efficient Evaluations



Integration

Transversal project with many applications in empirical fields

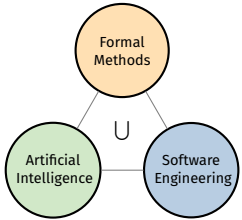
→ CRIStAL, UMR 9189, Lille, **Spirals** (SEEDS)

Frugality and Evaluations: Clément Quinton, Romain Rouvoy
Trust: Pierre Bourhis (Citadelle)
Statistics: Scool team

→ IRISA, UMR 6074, Rennes, **DiverSE** (rRaise)

Reproducibility: Mathieu Acher, Olivier Barais, Djamel E. Khelladi
Frugality: *Romain Lefeuvre*, Quentin Perez

Théo Matricon

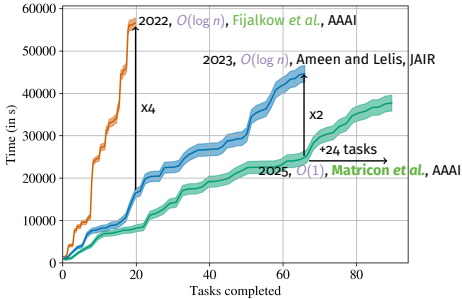


Selected Publications

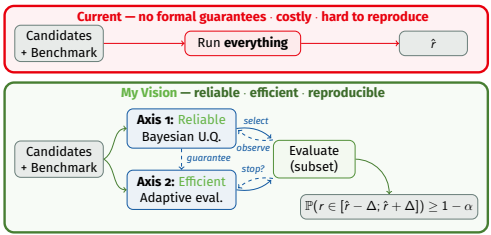
2 AAAI →

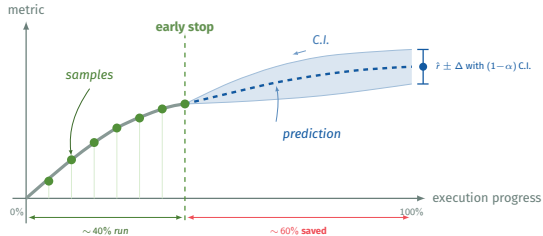
IST, MSR, 2 TOSEM

TACAS



Research Project





[1] G. Bathie, N. Fijalkow, T. Matricon, B. Mouillon, and P. Vandenhove, "LTL$_f$ learning meets boolean synthesis," in *Tools and Algorithms for the Construction and Analysis of Systems: 32nd International Conference, TACAS 2026, Held as Part of the International Joint Conferences on Theory and Practice of Software, ETAPS 2026, Proceedings*, 2026. [Online]. Available: https://theomat.github.io/files/ltl.pdf

[2] R. Robbes, T. Matricon, T. Degueule, A. Hora, and S. Zacchiroli, "Promises, perils, and (timely) heuristics for mining coding agent activity," in *2026 IEEE/ACM 23rd International Conference on Mining Software Repositories (MSR)*, IEEE, 2026.

[3] A. Martin, D. E. Khelladi, T. Matricon, and M. Acher, "Re-evaluating metamorphic testing of chess engines: A replication study," *Information and Software Technology*, vol. 181, p. 107 679, 2025, ISSN: 0950-5849. DOI: https://doi.org/10.1016/j.infsof.2025.107679 [Online].

Available: https://www.sciencedirect.com/science/article/pii/S0950584925000187

[4] T. Matricon, M. Acher, H. Spieker, and A. Gotlieb, "Efficiently ranking software variants with minimal benchmarks," *arXiv preprint arXiv:2509.06716, under revision at TOSEM*, 2025.

[5] T. Matricon, N. Fijalkow, and G. Lagarde, "Ecosearch: A constant-delay best-first search algorithm for program synthesis," in *Proceedings (AAAI Artificial Intelligence and Interactive Digital Entertainment Conference)*, 2025. [Online]. Available: https://arxiv.org/abs/2412.17330

[6] R. Robbes, T. Matricon, T. Degueule, A. Hora, and S. Zacchiroli, "Empirical study of coding agent adoption on github," *Under submission*, 2025.

[7] G. Shabadi, N. Fijalkow, and T. Matricon, "Theoretical foundations for programmatic reinforcement learning," in *GenPlanAI workshop AAAI (preprint)*, 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2402.11650

[8] H. Spieker, T. Matricon, N. Belmecheri, J. E. Betten, G. L. B. Lyan, H. Borges, Q. Mazouni, D. Gross, A. Gotlieb, and M. Acher, "Prompting for performance: Exploring llms for configuring software," in *The 37th IEEE International Conference on Tools with Artificial Intelligence (ICTAI)*, 2025. [Online]. Available: https://arxiv.org/pdf/2507.09790

[9] T. Matricon, N. Fijalkow, and G. Margueritte, "Wikicoder: Learning to write knowledge-powered code," in *Model Checking Software*, G. Caltais and C. Schilling, Eds., Springer Nature Switzerland, 2023, pp. 123–140, ISBN: 978-3-031-32157-3. [Online]. Available: https://rdcu.be/dITGA

[10] M. Anastacio, T. Matricon, and H. Hoos, "Challenges of acquiring compositional inductive biases via meta-learning," in *ECMLPKDD Workshop on Meta-Knowledge Transfer*, P. Brazdil, J. N. van Rijn, H. Gouk, and F. Mohr, Eds., ser. Proceedings of Machine Learning Research, vol. 191, PMLR, Sep. 2022, pp. 11–23. [Online]. Available: https://proceedings.mlr.press/v191/anastacio22a.html

[11] N. Fijalkow, G. Lagarde, T. Matricon, K. Ellis, P. Ohlmann, and A. Potta, "Scaling neural program synthesis with distribution-based search," in *International Conference on Artificial Intelligence, AAAI*, 2022. [Online]. Available: https://arxiv.org/abs/2110.12485

[12] T. Matricon, N. Fijalkow, G. Lagarde, and K. Ellis, "Deepsynth: Scaling neural program synthesis with distribution-based search," *Journal of Open Source Software*, vol. 7, no. 78, p. 4151, 2022. DOI: 10.21105/joss.04151

[13] T. Matricon, M. Anastacio, N. Fijalkow, L. Simon, and H. H. Hoos, "Statistical Comparison of Algorithm Performance Through Instance Selection," in *27th International Conference on Principles and Practice of Constraint Programming (CP 2021)*, L. D. Michel, Ed., ser. Leibniz International Proceedings in Informatics (LIPIcs), vol. 210, Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2021, 43:1–43:21, ISBN: 978-3-95977-211-2. DOI:

10.4230/LIPIcs.CP.2021.43 [Online]. Available: https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.CP.2021.43

[14] D. Duplyakin, A. Uta, A. Maricq, and R. Ricci, "In datacenter performance, the only constant is change," in *2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*, IEEE, 2020, pp. 370–379.

[15] N. A. Ernst, "Bayesian hierarchical modelling for tailoring metric thresholds," in *Proceedings of the 15th international conference on mining software repositories*, 2018, pp. 587–591.

[16] L. Bornmann and R. Mutz, "Growth rates of modern science: A bibliometric analysis based on the number of publications and cited references," *Journal of the association for information science and technology*, vol. 66, no. 11, pp. 2215–2222, 2015.