LLM-ASSISTED CROSSOVER

IN GENETIC IMPROVEMENT OF SOFTWARE

PEKING UNIVERSITY

DIMITRIOS BOURAS
SERGEY MECHTEAV

UCL

JUSTYNA PETKE

MOTIVATION

End of Moore’s Law?

Hardware improvement has slowed down, we need to focus on software

MOTIVATION

End of Moore’s Law?

Hardware improvement has slowed down, we need to focus on software

Need for automation

In modern huge codebses manually optimising is unscalable, error-prone and expensive

MOTIVATION

End of Moore's Law?

Hardware improvement has slowed down, we need to focus on software

Need for automation

In modern huge codebses manually optimising is unscalable, error-prone and expensive

Traditional crossover in GI

Inefficient, lack of contextual awareness, too much wasted search effort

MOTIVATION

End of Moore's Law?

Hardware improvement has slowed down, we need to focus on software

Need for automation

In modern huge codebses manually optimising is unscalable, error-prone and expensive

Traditional crossover in GI

Inefficient, lack of contextual awareness, too much wasted search effort

Why LLMs?

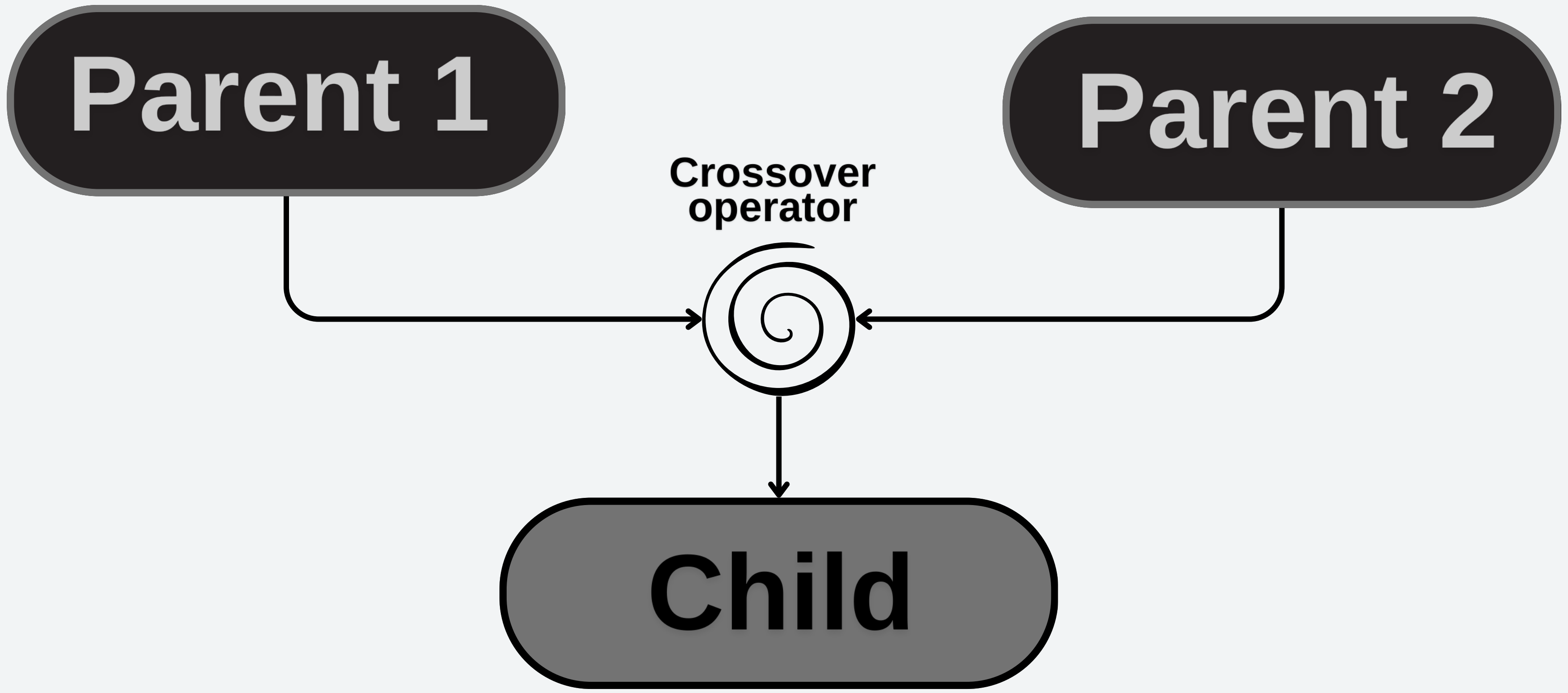
Need for an intelligent crossover operator with contextual awareness:

- Produce more offsprings that survive
- Produce fitter offsprings
- Guide the search more efficiently.

BACKGROUND

GI AND CROSSOVER

GI	Automated technique that applies evolutionary algorithms to improve software performance by modifying code or parameters, inspired by natural selection
Variant	A version of the software being improved, with some modification made on it



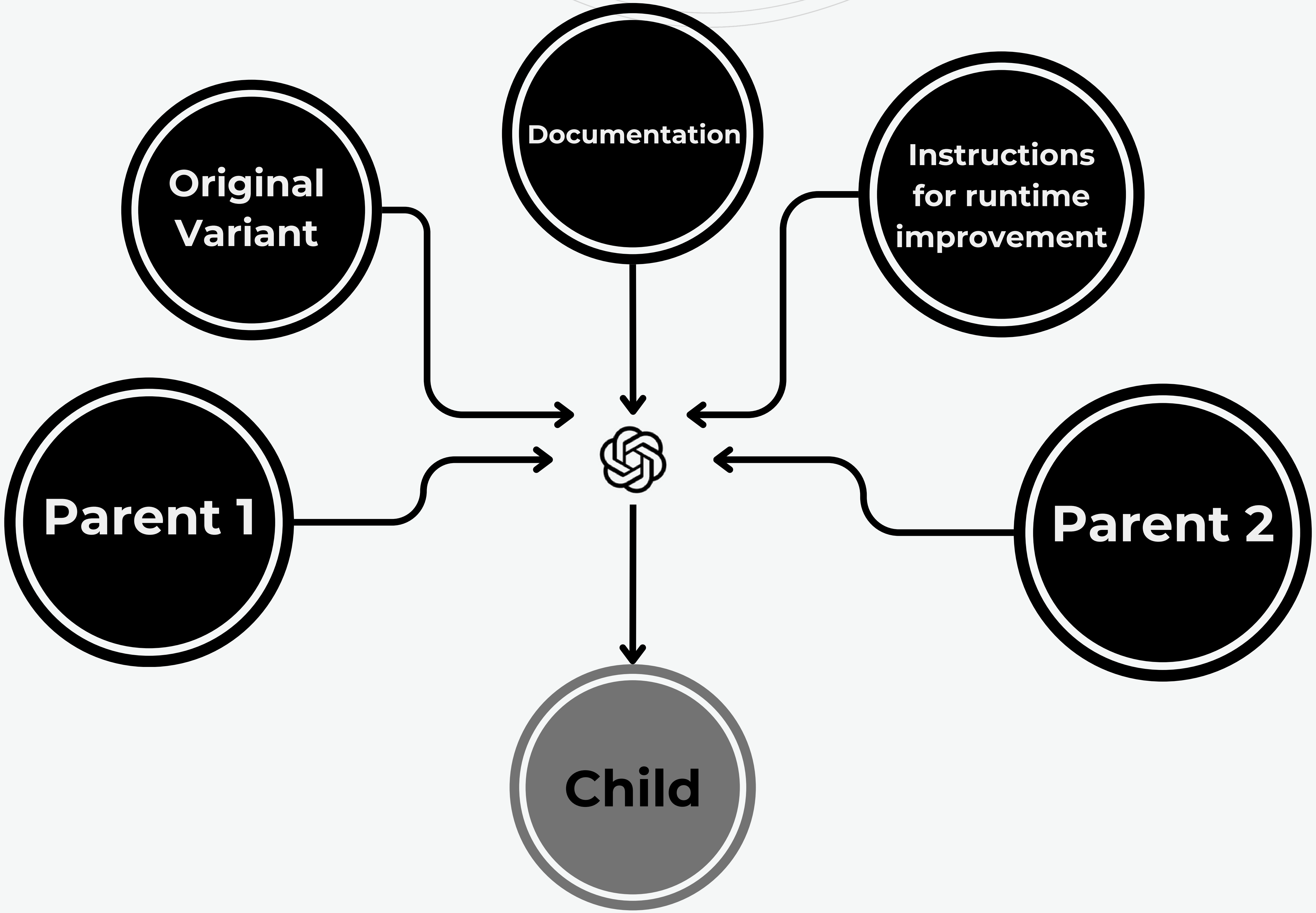
Mutation	Intoduces new changes to a variant to explore new areas of the search space
Crossover	Combines 2 parent variants together to create a new variant: the child

BACKGROUND

TRADITIONAL CROSSOVER OPERATORS

1point	Choose 1 crossover point in both parents and swap edits at this point
2point	Choose 2 crossover points at both parents for more flexibility
Concat	Append the edits of the 2 parents
Uniform Concat	Interleave edits from both parents uniformly before appending them
Uniform Inter	Interleave the edits of the 2 parents together

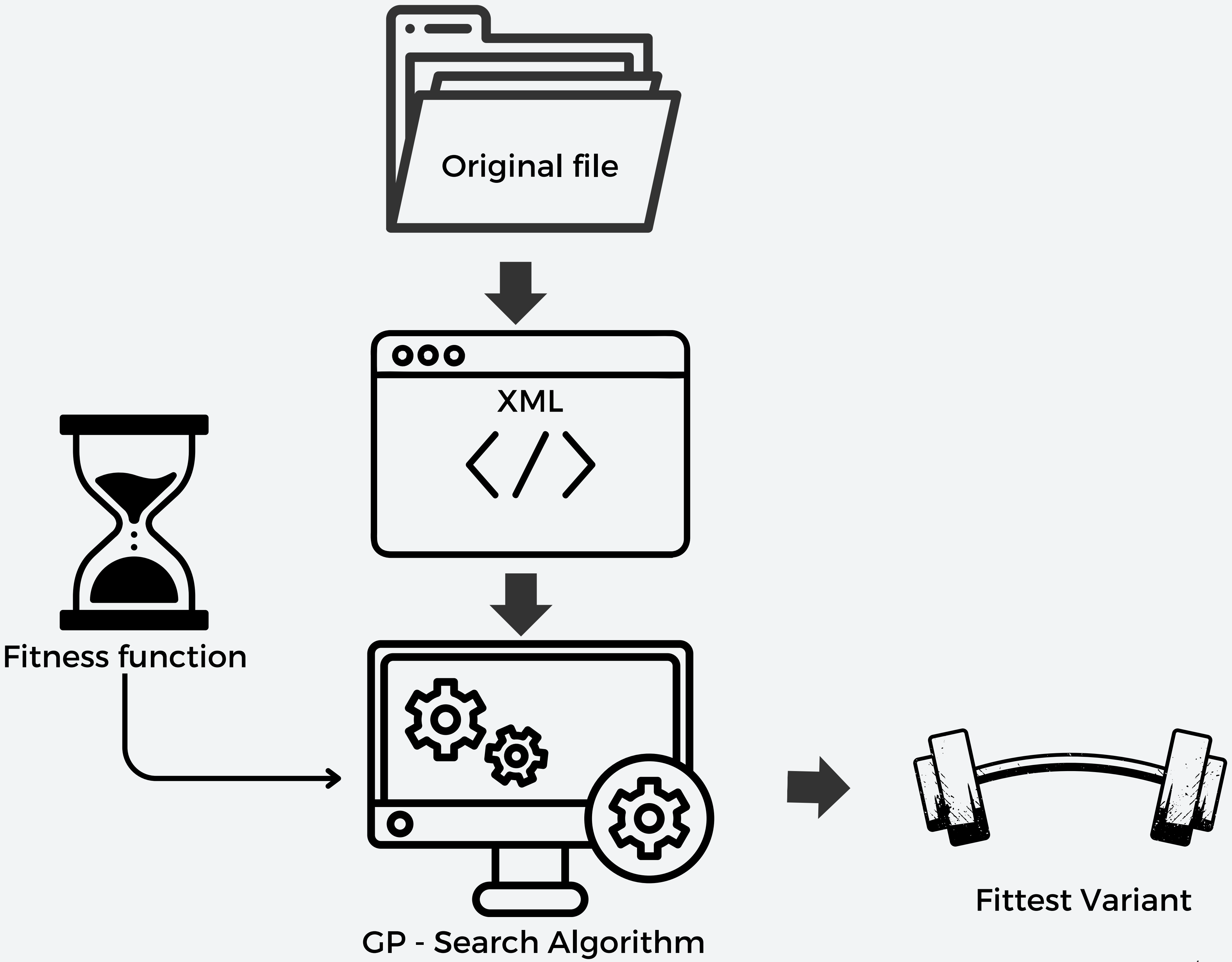
LLM ASSISTED CROSSOVER OPERATOR



MAGPIE FRAMEWORK

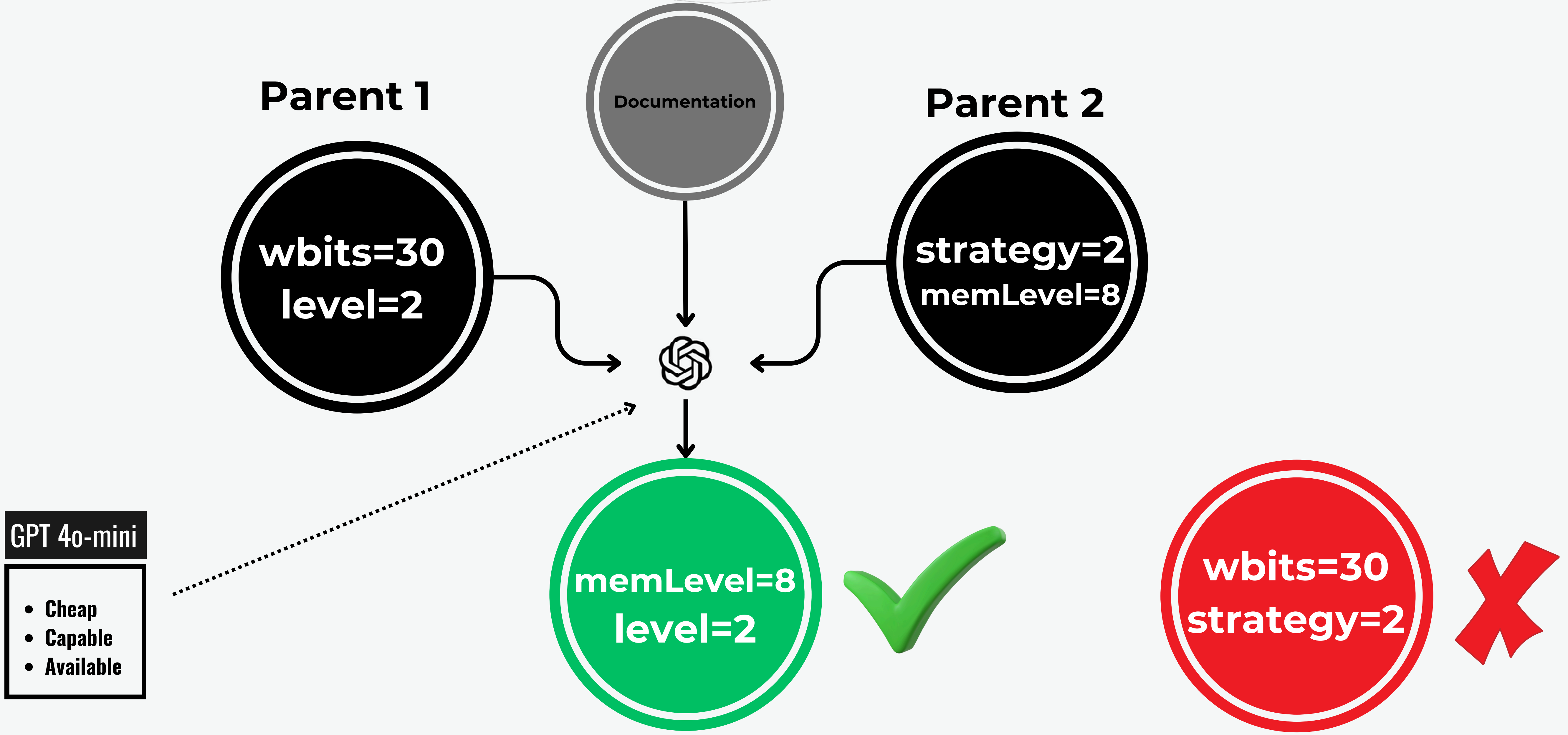
WHY MAGPIE?

- **Language Agnostic :** everything is translated to XML formar
- **Multiple modification types:** source code and parameter configuration
- **Multiple search types:** GP, Local Search etc
- **Easy to modify :** Written in Python with documentation support



MOTIVATIONAL EXAMPLE

COMPRESSION LIBRARY



LLM Response Summary

To optimize runtime I will select the following edits: level =2, which lowers the compression level, resulting in faster compression and memLevel = 8 which increases the memory level, improving performance and lowering runtime

BENCHMARKS - EXPERIMENTS

Selected Benchmarks

Benchmark	Description	Parameter	Source Code	Language
MiniSAT Hack	SAT solver	✓	✓	C++
MiniSAT	SAT solver	✓	✓	C++
WEKA	Data mining tool	✓	✓	Java
zlib	Compression library	✓		Python
SciPy	Scientific computing	✓		Python
Sat4j	Boolean satisfiability	✓	✓	Java
LPG	AI planner	✓		C

Configs

- Model : GPT 4o-mini
- pop_size = 20
- offspring_elitism = 0.2
- offspring_crossover = 0.6
- offspring_mutation = 0.2
- epochs = 11

RESULTS
AVERAGE BEST RUNTIME

Comparison of Crossover methods based on average best fitness

Crossover Method	AOF(s)	AF(s)	Avg Ranking
UniformConcat	4.598	6.094	3.00
Concat	4.734	5.841	3.55
1Point	4.971	6.333	3.82
2Point	5.169	6.463	4.27
UniformInter	4.991	6.348	4.09
LLM-Assisted	**4.477**	**5.834**	**2.27**

AOF (s): Average Optimal Fitness in seconds.
AF (s): Average Fitness of all variants in seconds.

-8.5% AOF
-6.1% AF
} compared to the average of the 5 methods

RESULTS
REACHING PERFORMANCE MILESTONES

Comparison of Crossover methods based on time needed to reach performance milestones

Crossover Method	25% Improvement	50% Improvement	75% Improvement	100% Improvement
UniformConcat	56.45 variants	68.55 variants	112.27 variants	227.27 variants
Concat	72.91 variants	105.82 variants	140.73 variants	228.73 variants
1Point	70.18 variants	78.55 variants	179.45 variants	242.27 variants
2Point	97.73 variants	129.64 variants	170.18 variants	230.73 variants
UniformInter	77.36 variants	130.27 variants	135.18 variants	244.91 variants
LLM-Assisted Crossover	**39.82 variants**	**48.27 variants**	**119.00 variants**	**209.18 variants**

Reached performance milestones with 25.6% fewer generated variants on average.

RESULTS

NUMBER OF VIABLE VARIANTS

Comparison of Crossover methods based on the number of viable variants produced

Crossover Method	Average Number of Viable Variants
UniformConcat	187.73
Concat	186.27
1Point	182.18
2Point	185.73
UniformInter	185.55
LLM-Assisted Crossover	**194.45**

On average of 4.8% more viable variants

RELATED WORK

Traditional Crossover

- Easy , cheap
- No context-awareness
- Random selection of modifications

LLM-ASSISTED CROSSOVER

- LLM's contextual knowledge
- Documentation

RELATED WORK

Traditional Crossover

- Easy , cheap
- No context-awareness
- Random selection of modifications

- LLM's contextual knowledge
- Documentation

Semantic & Context-Aware Crossover

- Semantically Driven Crossover
- Locally Geometric Semantic Crossover
- Scalability Issues
- Computationally expensive
- Requires program structure analysis

- Works out of the box
- Leverages LLM training

LLM-ASSISTED CROSSOVER

RELATED WORK

Traditional Crossover

- Easy , cheap
- No context-awareness
- Random selection of modifications

- LLM's contextual knowledge
- Documentation

Semantic & Context-Aware Crossover

- Semantically Driven Crossover
- Locally Geometric Semantic Crossover
- Scalability Issues
- Computationally expensive
- Requires program structure analysis

- Works out of the box
- Leverages LLM training

Machine Learning-Assisted Crossover

- Deep Neural Crossover
- Adaptive crossover
- Requires pre-training and fine-tuning

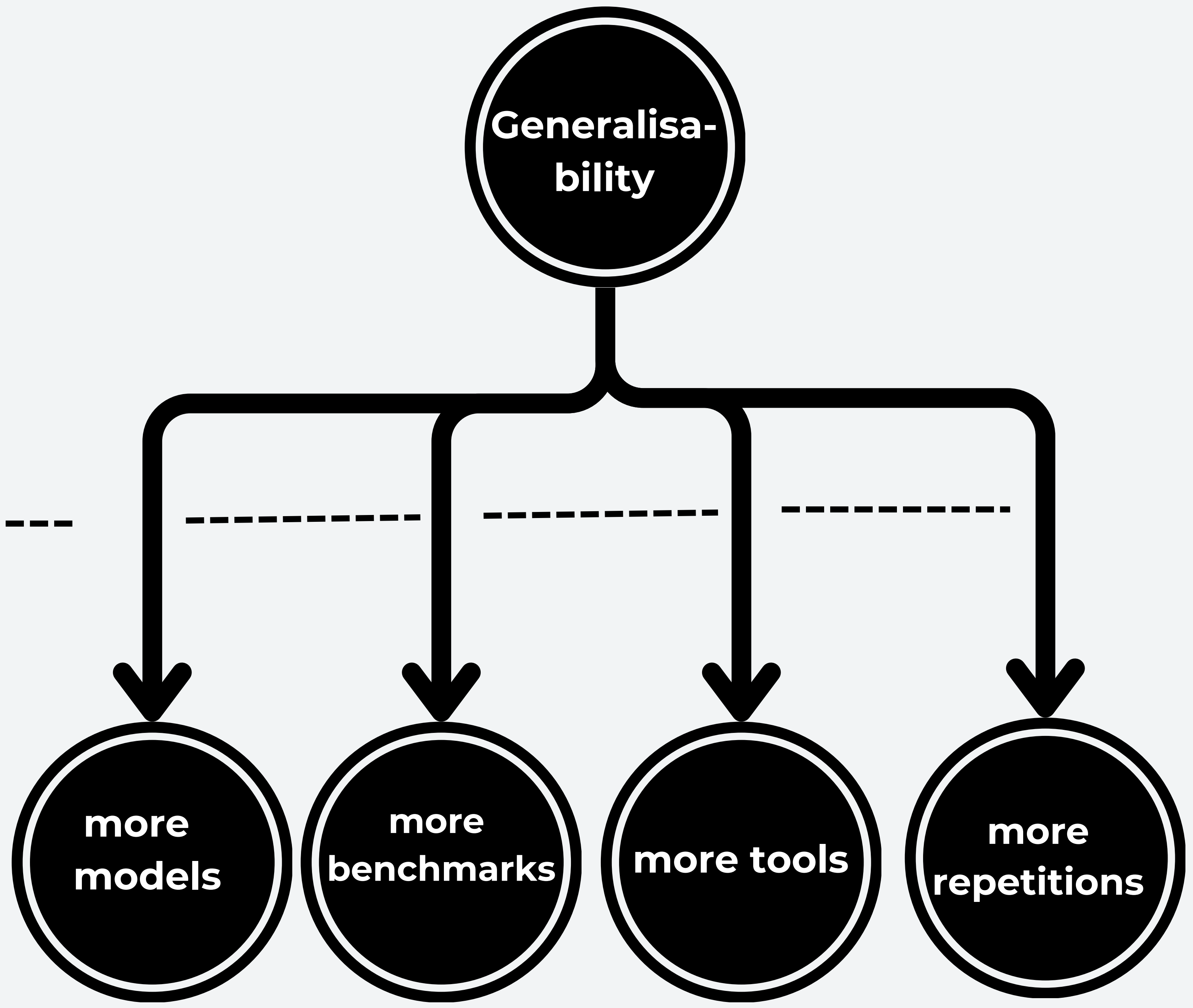
- No pre-training or fine-tuning needed
- No extra data required

LLM-ASSISTED CROSSOVER

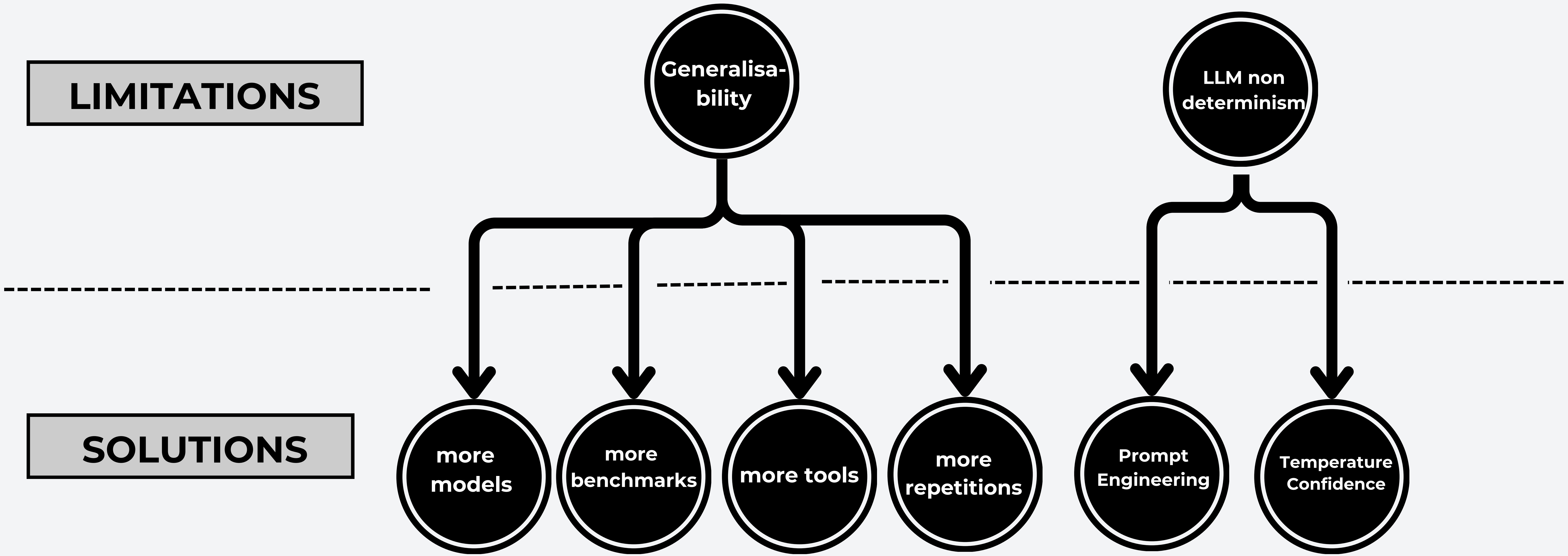
LIMITATIONS -FUTURE WORK

LIMITATIONS

SOLUTIONS



LIMITATIONS -FUTURE WORK



THANK YOU FOR YOUR ATTENTION!

https://github.com/SOLAR-group/LLM_Assisted_Crossover

KEY TAKEAWAYS

- Initial Exploration of LLMs for Crossover in GI of Software
- Provided Support for LLM Assisted Crossover in the MAGPIE Framework
- -8.5% runtime
- 25.6 % faster to reach milestones
- 4.8% more viable variants

LLM assisted crossover is promising.
Future work should focus on more experiments!!!