OpenTuner

An Extensible Framework for Program Autotuning

Jason Ansel, Shoaib Kamil, Kalyan Veeramachaneni, Jonathan Ragan-Kelley,
Jeffrey Bosboom, Una-May O'Reilly, and Saman Amarasinghe

Carl Seifert
cs2331@cam.ac.uk

Goal

Goal

Modify configuration → tune software heuristic, e.g.,

Goal

Modify configuration → tune software heuristic, e.g.,

- Program size

Goal

Modify configuration → tune software heuristic, e.g.,

- Program size
- Time

Goal

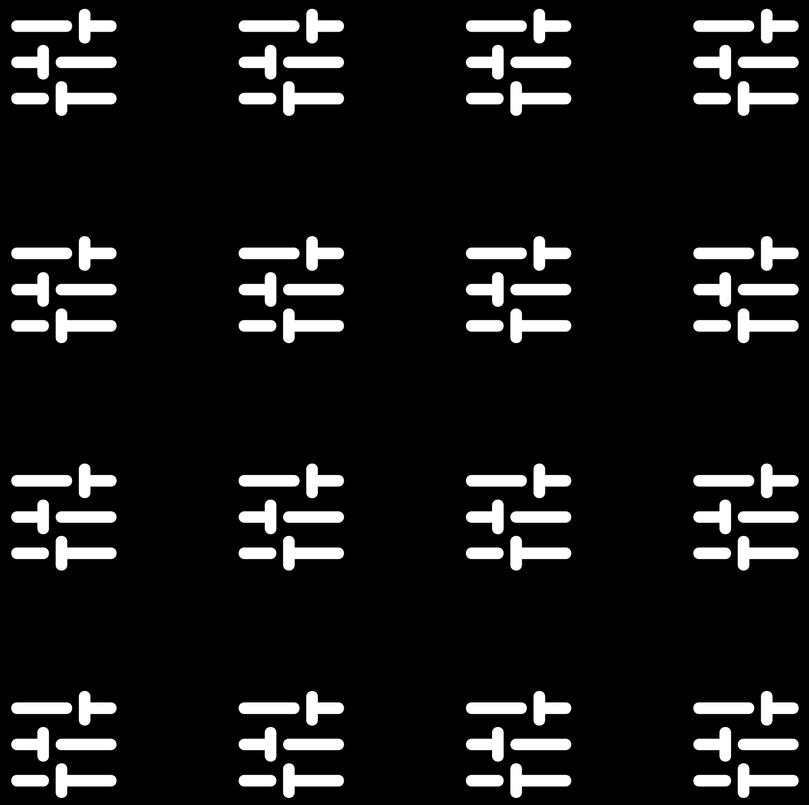
Modify configuration → tune software heuristic, e.g.,

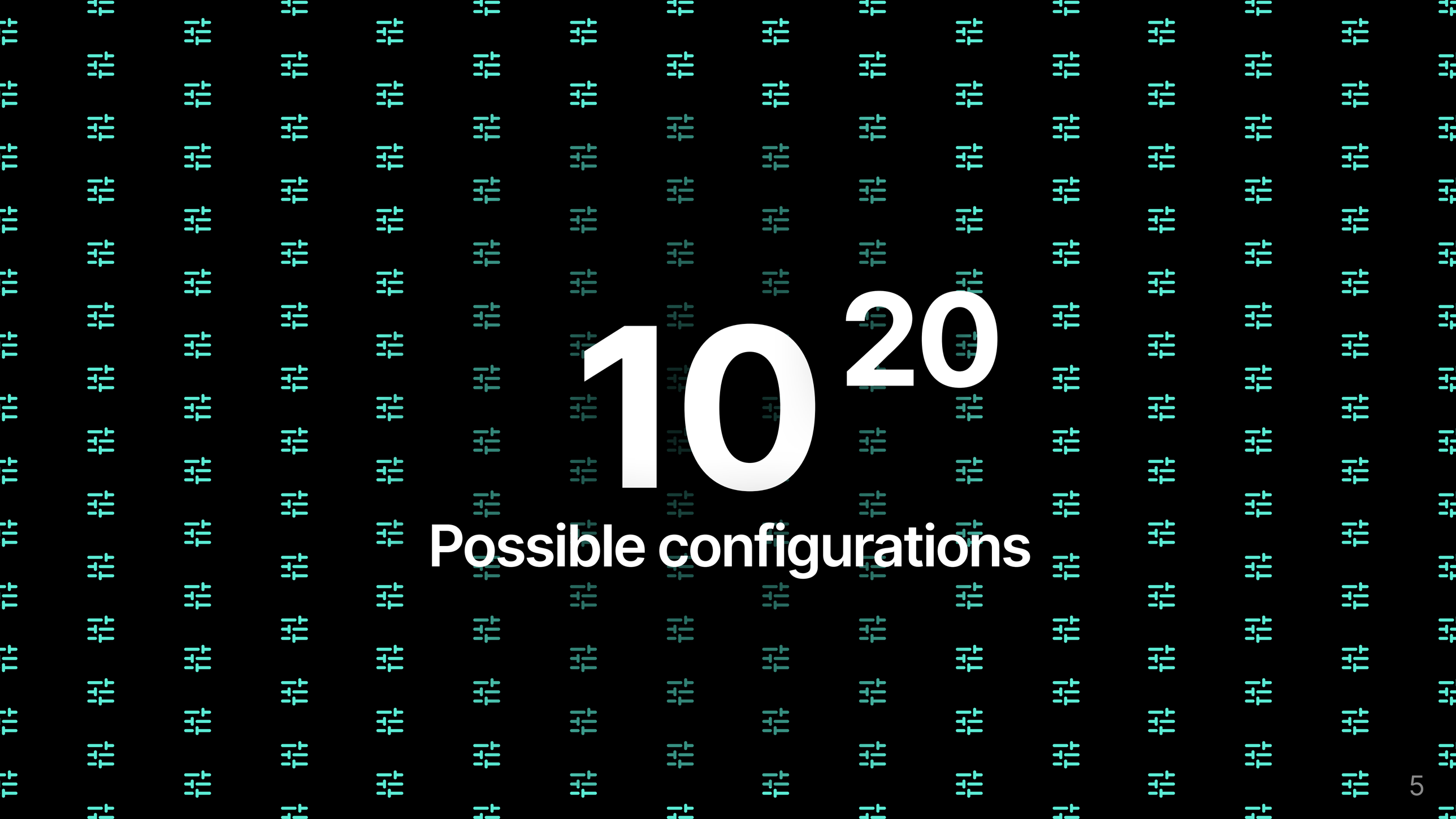
- Program size
- Time
- Resource usage (e.g., memory)

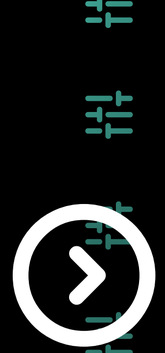
Goal

Modify configuration → tune software heuristic, e.g.,

- Program size
- Time
- Resource usage (e.g., memory)
- Energy consumption




10^{20}
Possible configurations



10 3600

Possible configurations

Autotuning

Autotuning

Challenges

Autotuning

Challenges

1 **Configuration Representation**
Need domain-informed configuration space representation

Autotuning

Challenges

1. **Configuration Representation**
   Need domain-informed configuration space representation

2. **Configuration Space Size**
   Need search technique

Autotuning

Challenges

1. **Configuration Representation**
   Need domain-informed configuration space representation

2. **Configuration Space Size**
   Need search technique

3. **Configuration Space Landscape**
   Continuity? Dimensionality? Coupled parameters?

Autotuning

Challenges

1. **Configuration Representation**
   Need domain-informed configuration space representation

2. **Configuration Space Size**
   Need search technique

3. **Configuration Space Landscape**
   Continuity? Dimensionality? Coupled parameters?

? **Effect on search techniques?**

Configuration Space Landscape

Technique – Hill Climber Optimisation

Configuration Space Landscape

Technique – Hill Climber Optimisation


Configuration Space Landscape
Discontinuities, Smaller Hills, Plateaus, Coupling

Configuration Space Landscape

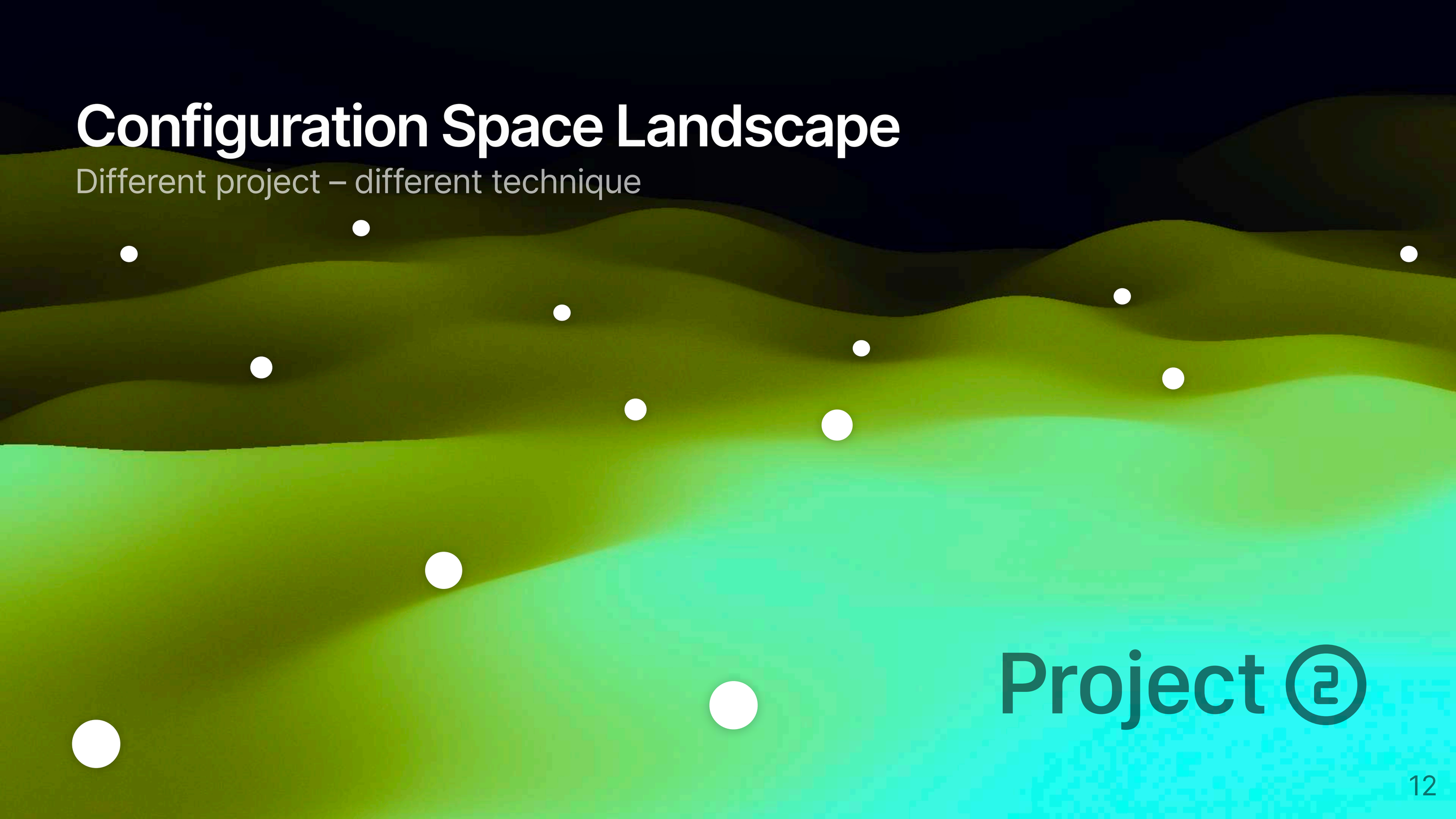
Discontinuities, Smaller Hills, Plateaus, Coupling

Configuration Space Landscape

Different project – different technique

Configuration Space Landscape

Different project – different technique



Project 2

Autotuning

Challenges

1. **Configuration Representation**
   Need domain-informed configuration space representation

2. **Configuration Space Size**
   Need search technique

3. **Configuration Space Landscape**
   Continuity? Dimensionality? Coupled parameters?

Autotuning

Challenges

1 **Configuration Representation**
Need domain-informed configuration space representation

2 **Configuration Space Size**
Need search technique

3 **Configuration Space Landscape**
Continuity? Dimensionality? Coupled parameters?

? **Search technique efficacy depends heavily on landscape (→ on project)**
Previous work: Custom representations & techniques for specific problem

OpenTuner

Search

OpenTuner

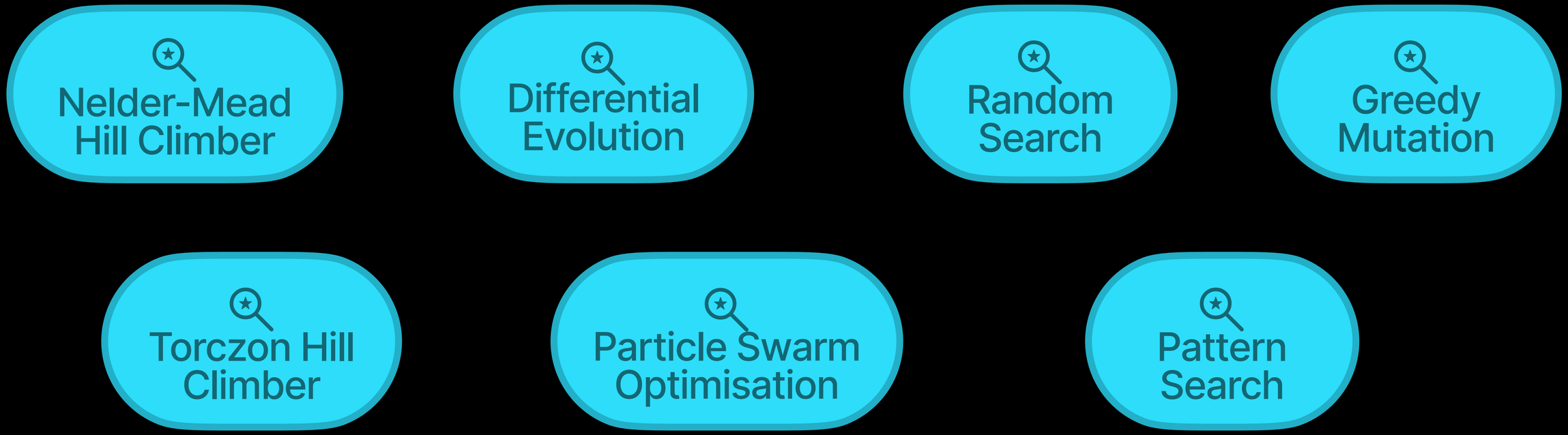
Search

Use multiple search techniques
(“Ensemble”) simultaneously

OpenTuner

Search

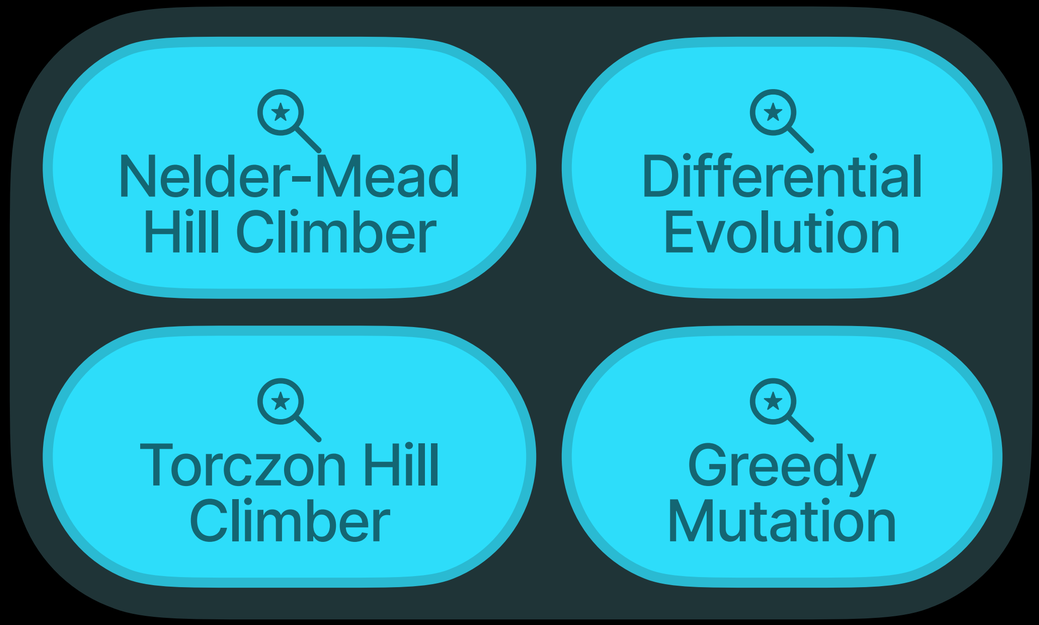
Use multiple search techniques
(“Ensemble”) simultaneously



OpenTuner

Search

Use multiple search techniques (“Ensemble”) simultaneously

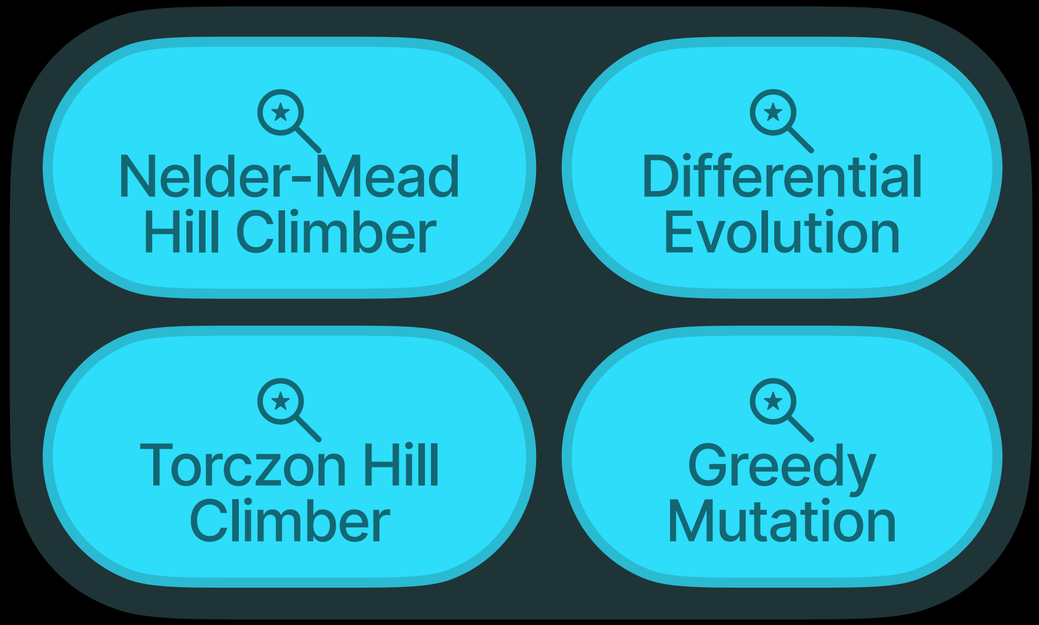


OpenTuner

Search

Use multiple search techniques (“Ensemble”) simultaneously

Meta techniques group techniques



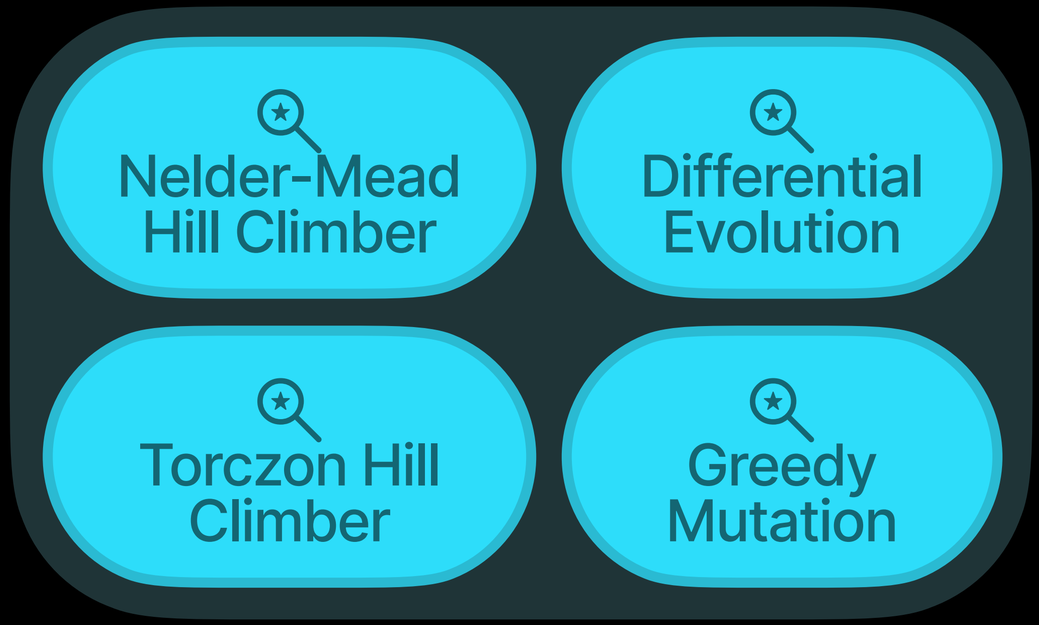
OpenTuner

Search

Use multiple search techniques (“Ensemble”) simultaneously

Meta techniques group techniques

Can nest (meta) techniques



OpenTuner

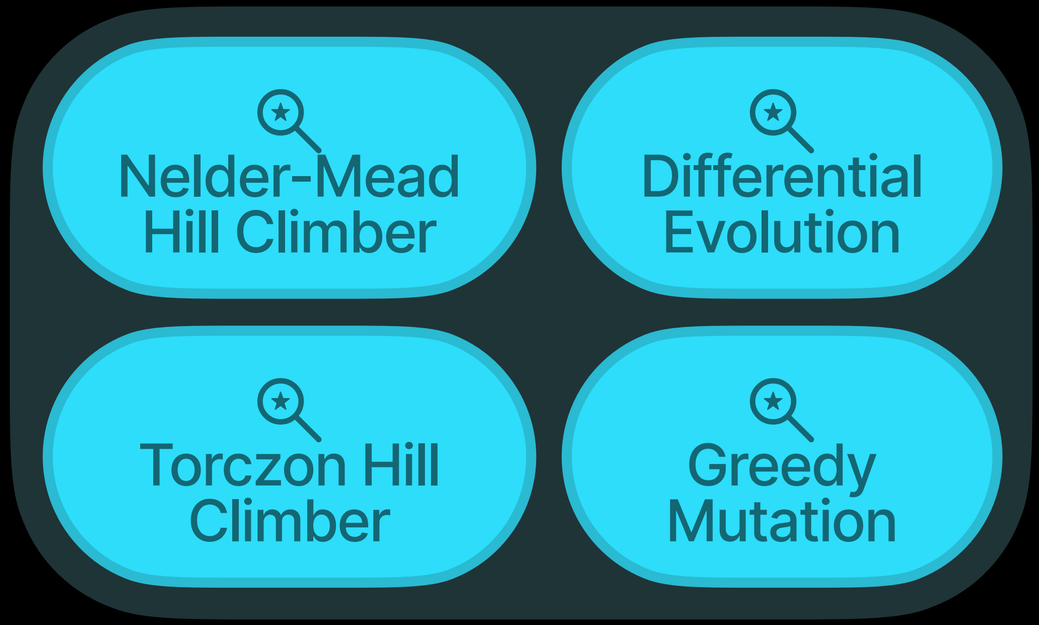
Search

Use multiple search techniques (“Ensemble”) simultaneously

Meta techniques group techniques

Can nest (meta) techniques

Dynamically allocate test runs to sub-techniques
(e.g., Round Robin, Multi-armed bandit)



OpenTuner

Search

Use multiple search techniques (“Ensemble”) simultaneously

Meta techniques group techniques

Can nest (meta) techniques

Dynamically allocate test runs to sub-techniques
(e.g., Round Robin, Multi-armed bandit)

EXPLOITATION + **EXPLORATION**

OpenTuner

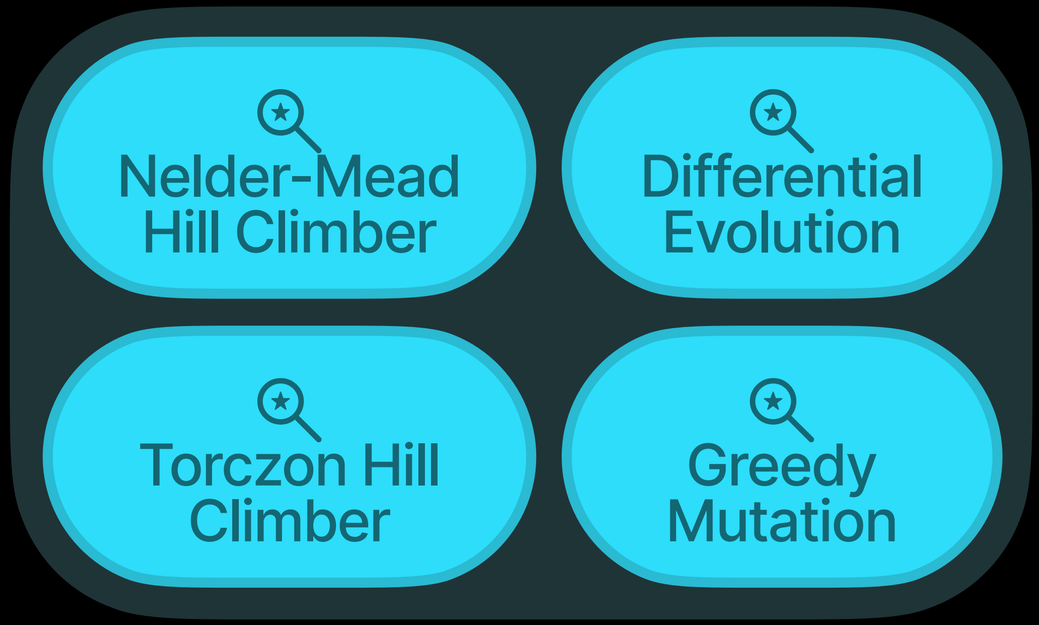
Search

Use multiple search techniques (“Ensemble”) simultaneously

Meta techniques group techniques

Can nest (meta) techniques

Dynamically allocate test runs to sub-techniques
(e.g., Round Robin, Multi-armed bandit)



$$\text{argmax}_T \left(\frac{2}{|V_T|(|V_T|+1)} \sum_{i=0}^{|V_T|} i\,V_{i,T} + C\sqrt{\frac{2\lg|H|}{|V_T|}} \right)$$

OpenTuner

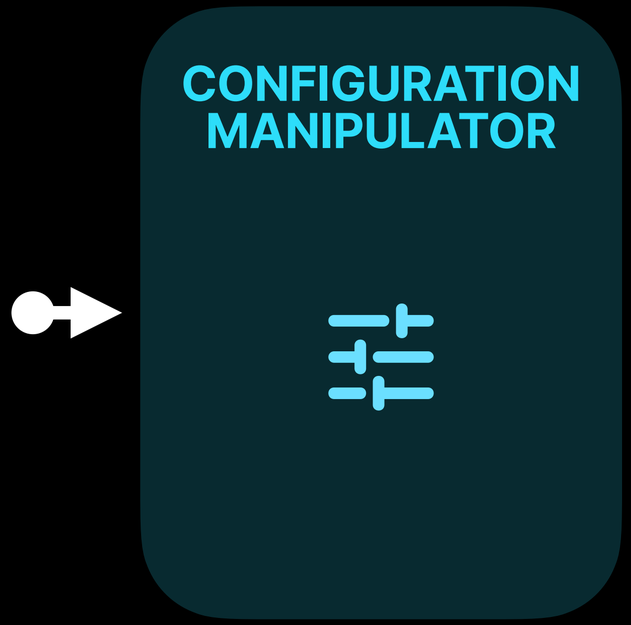
Components

OpenTuner

Components

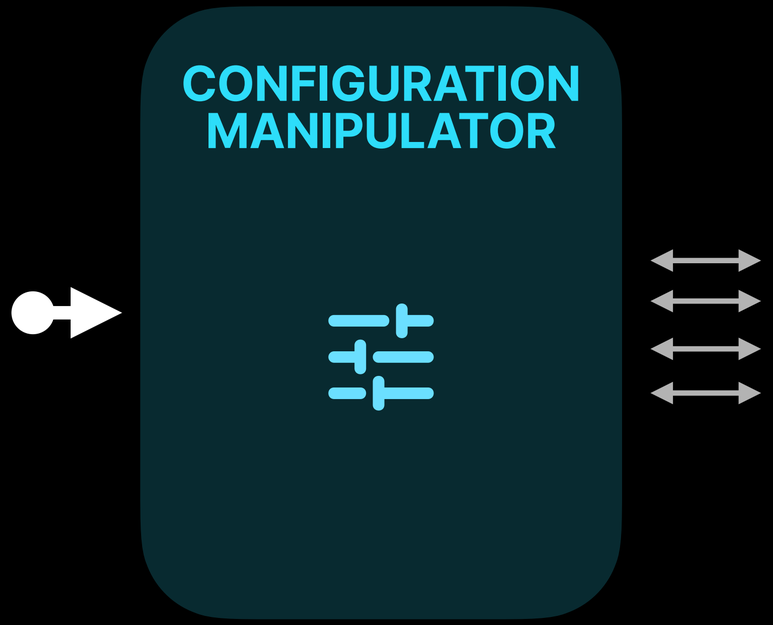
OpenTuner

Components



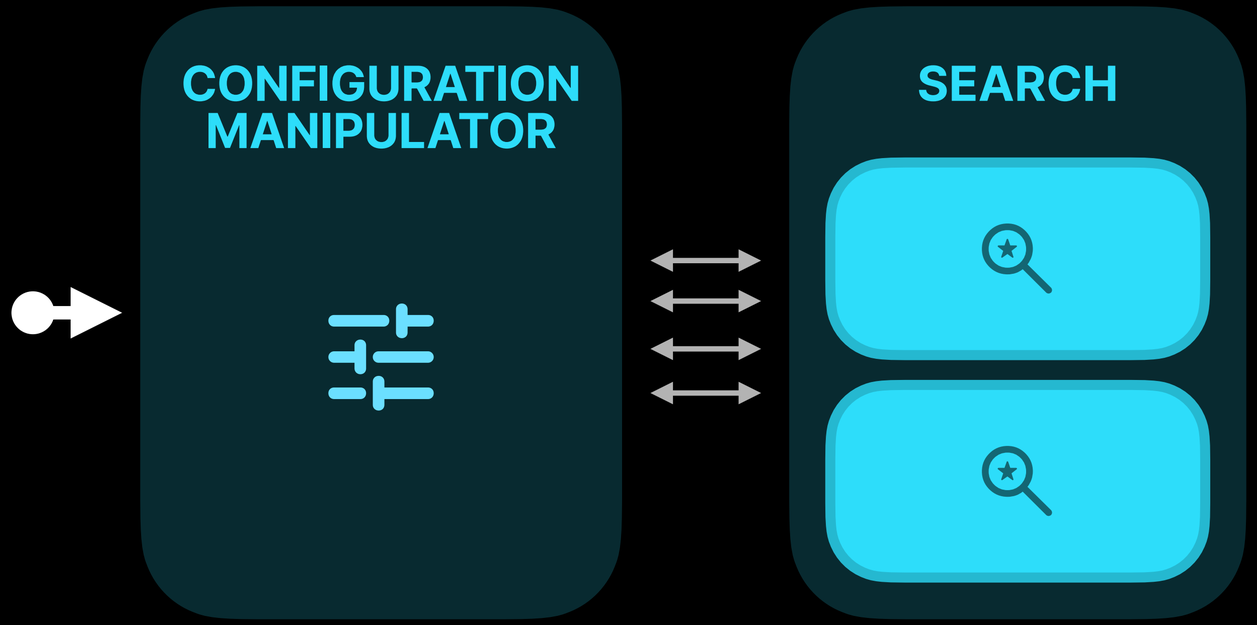
OpenTuner

Components



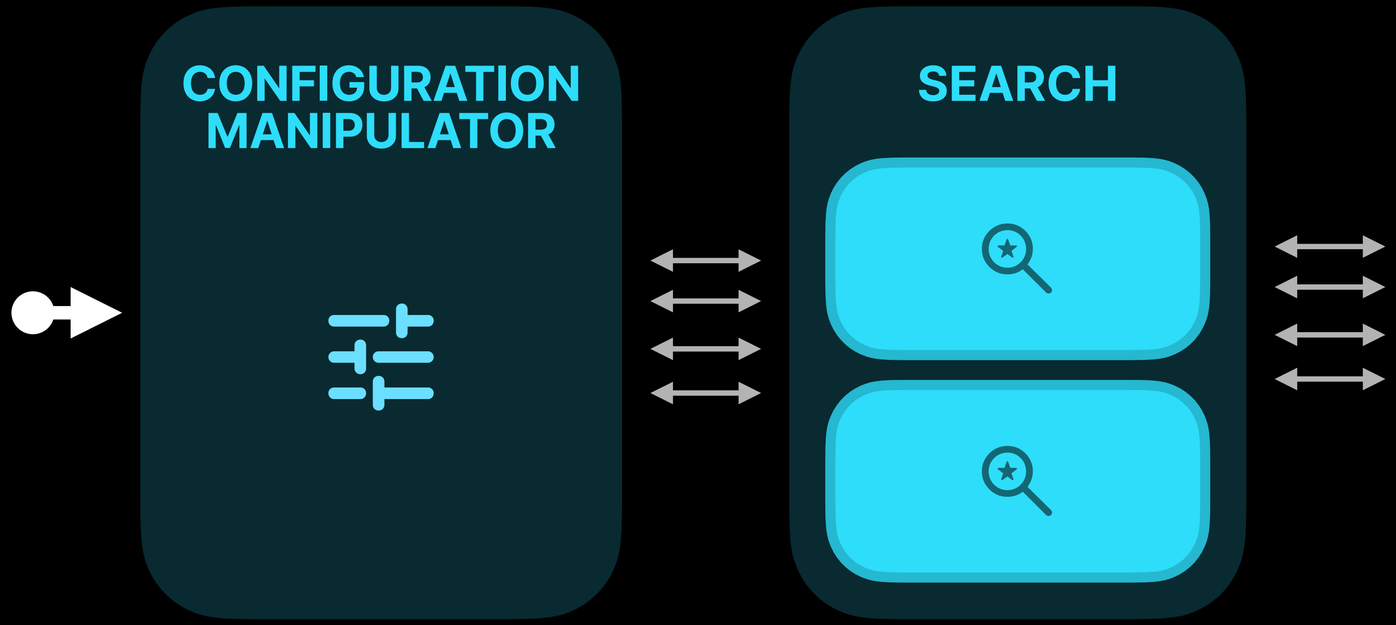
OpenTuner

Components



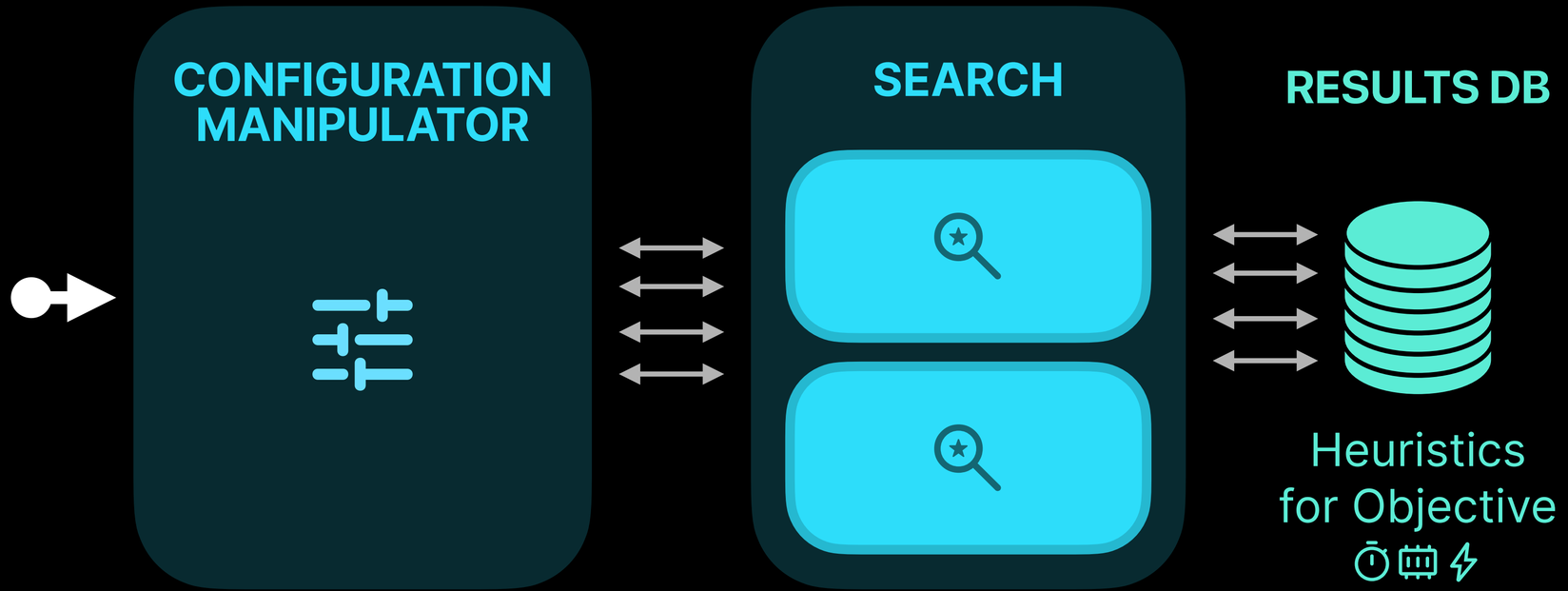
OpenTuner

Components



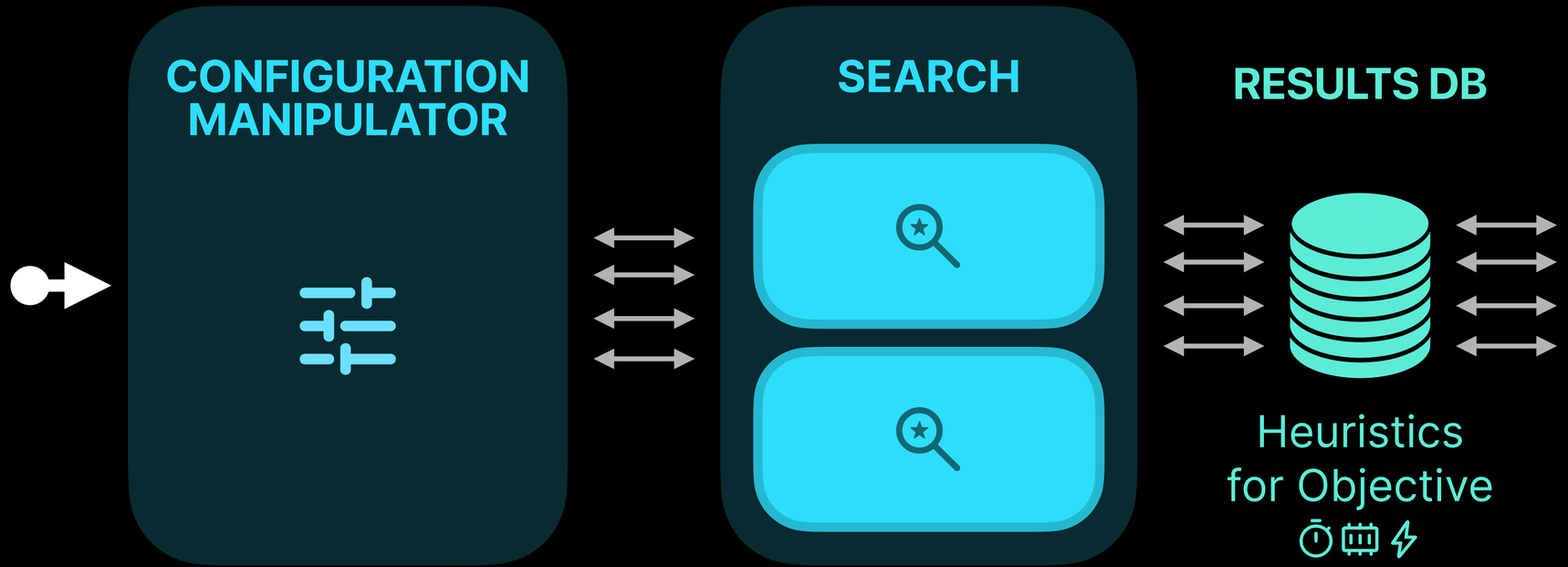
OpenTuner

Components



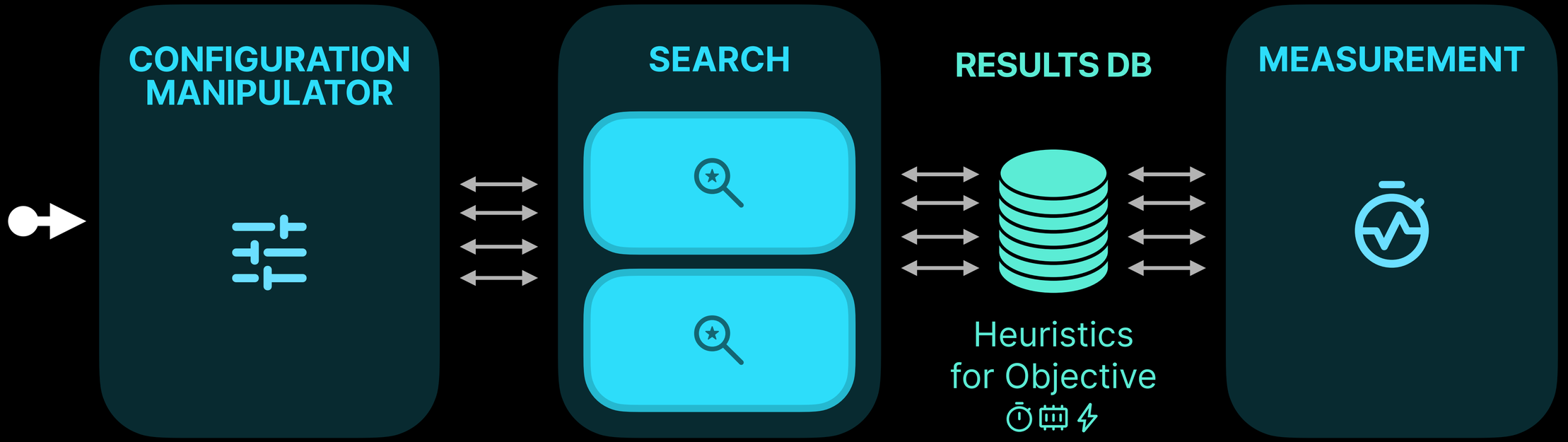
OpenTuner

Components



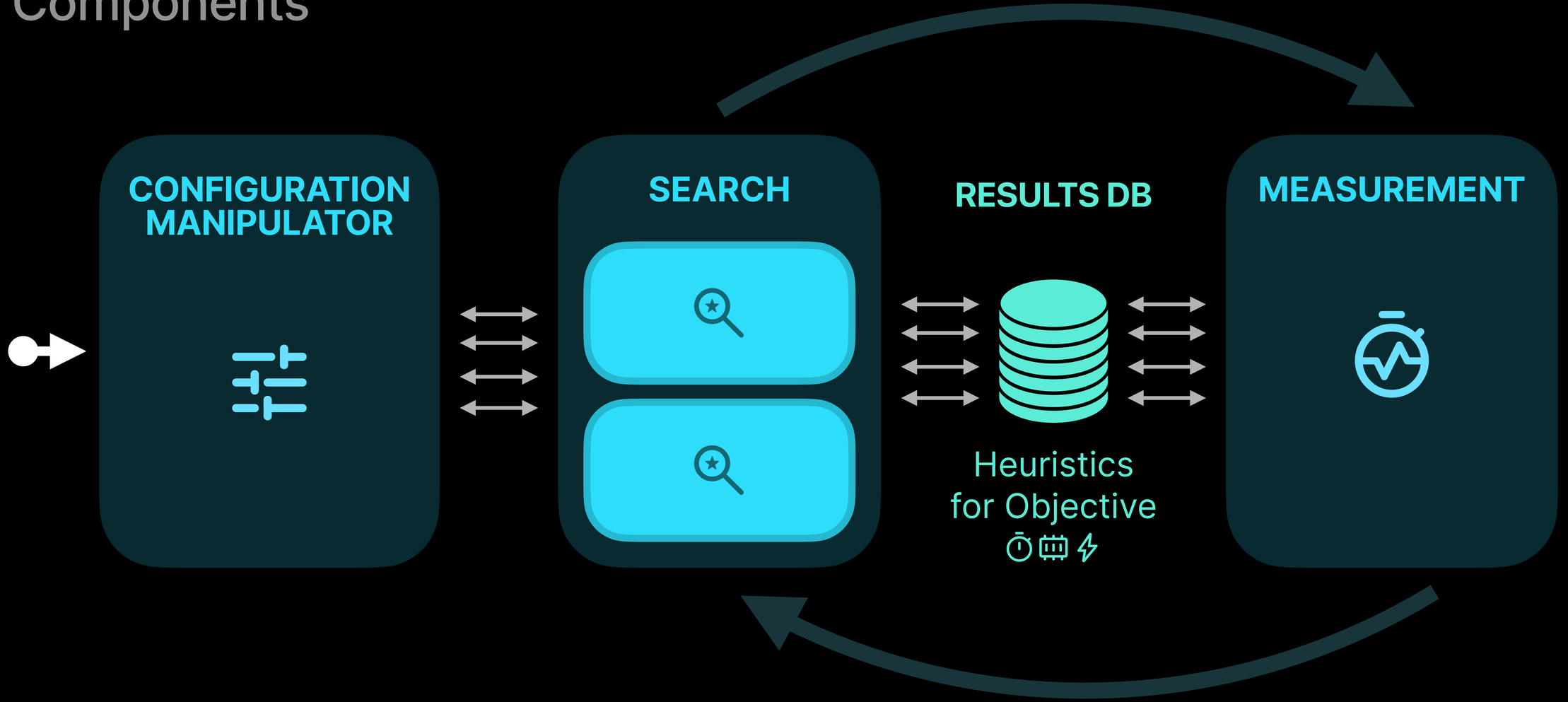
OpenTuner

Components



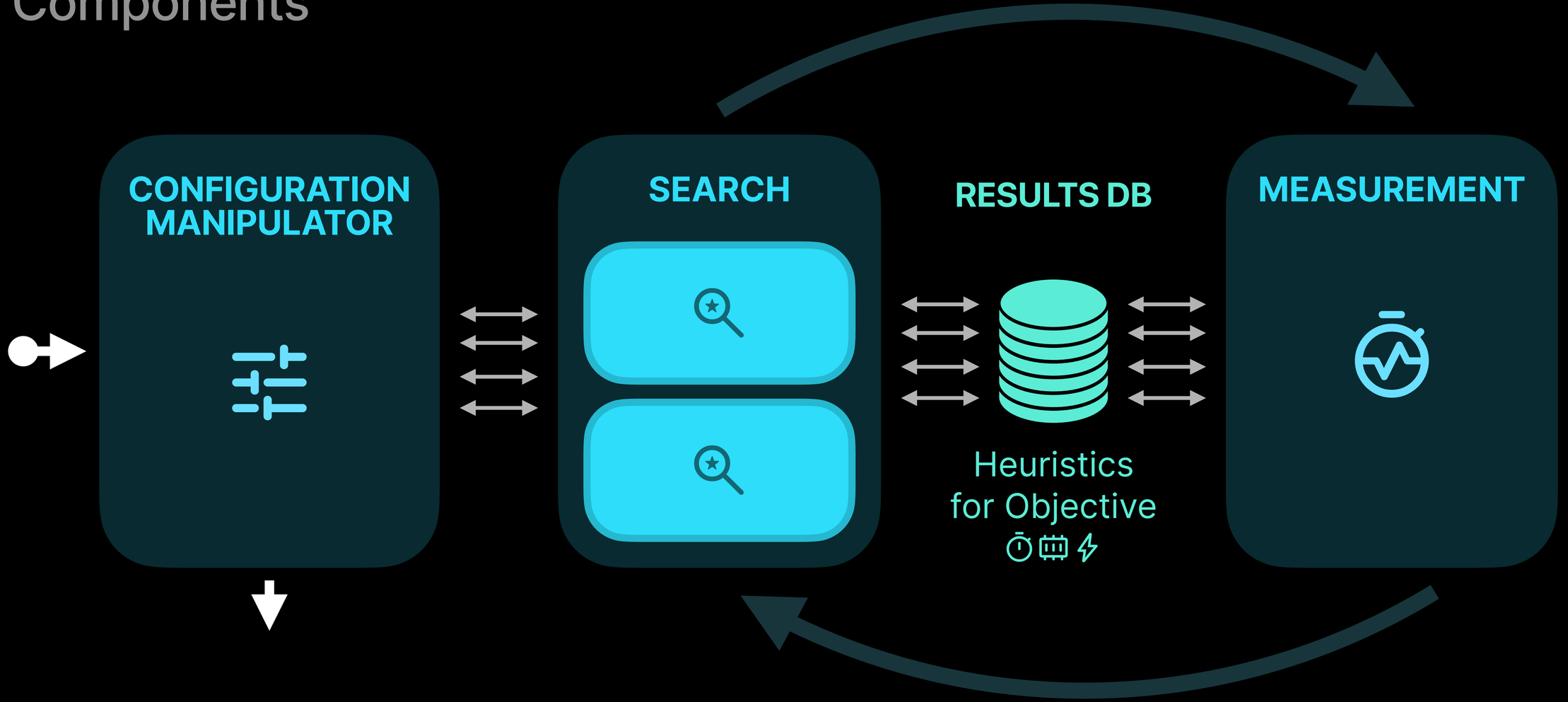
OpenTuner

Components



OpenTuner

Components



OpenTuner

Solutions

1. Configuration Representation

2. Configuration Space Size
3. Configuration Space Landscape

OpenTuner

Solutions

1. **Configuration Representation**
   Python abstraction (API): primitive, complex, custom data types
2. **Configuration Space Size**
3. **Configuration Space Landscape**

OpenTuner

Solutions

1. **Configuration Representation**
   Python abstraction (API): primitive, complex, custom data types

2. **Configuration Space Size**

3. **Configuration Space Landscape**
   Ensemble of search techniques:
   concurrent, dynamic evaluation budget allocation, cooperative

Evaluation

Evaluation

- ☑ 7 different project with different configuration representations

Evaluation

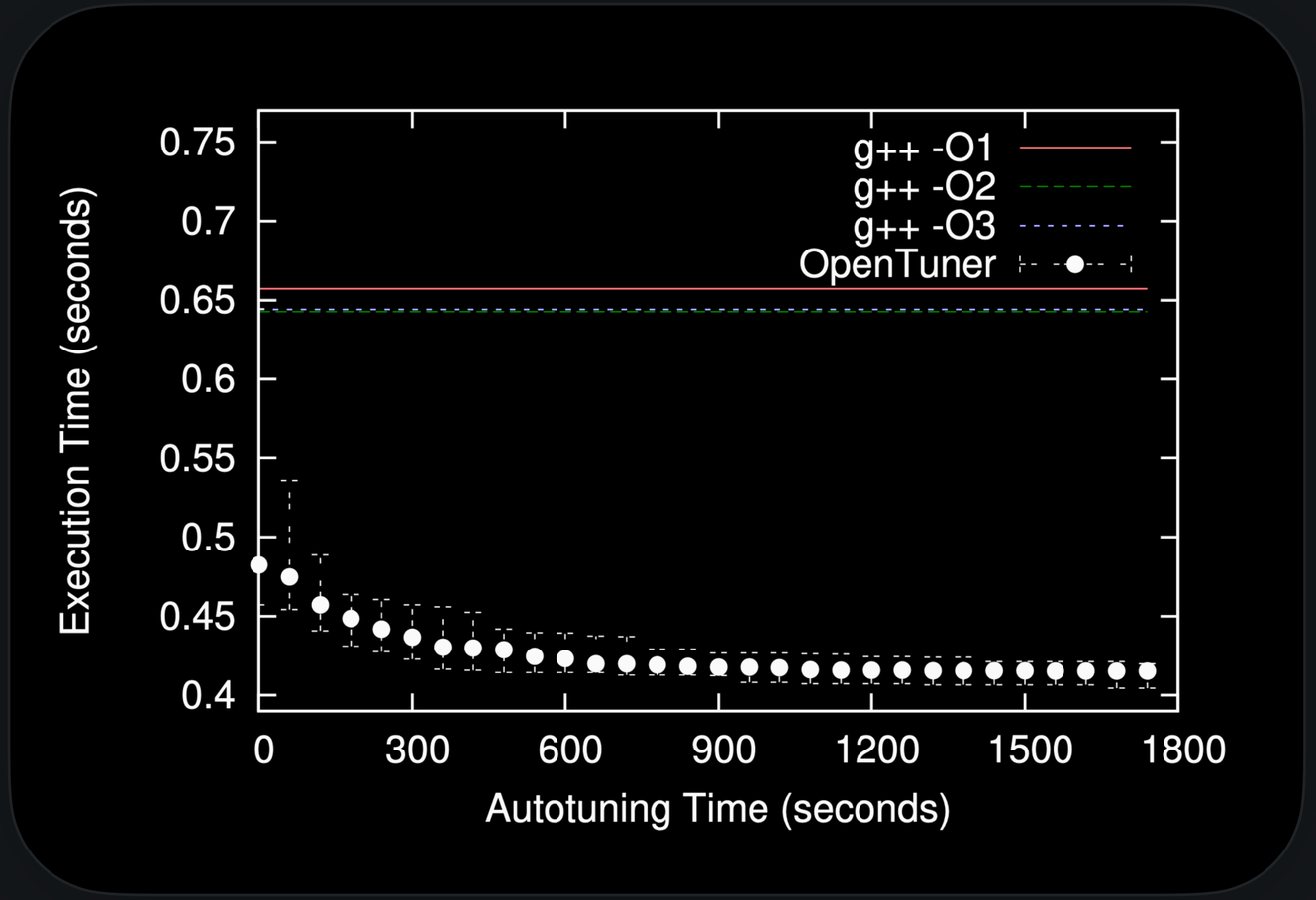
- [x] 7 different project with different configuration representations
- [x] Heuristic tuned to manual equivalent or optimum

Evaluation

- ☑ 7 different project with different configuration representations
- ☑ Heuristic tuned to manual equivalent or optimum
- ☑ Domain-specific success

Evaluation

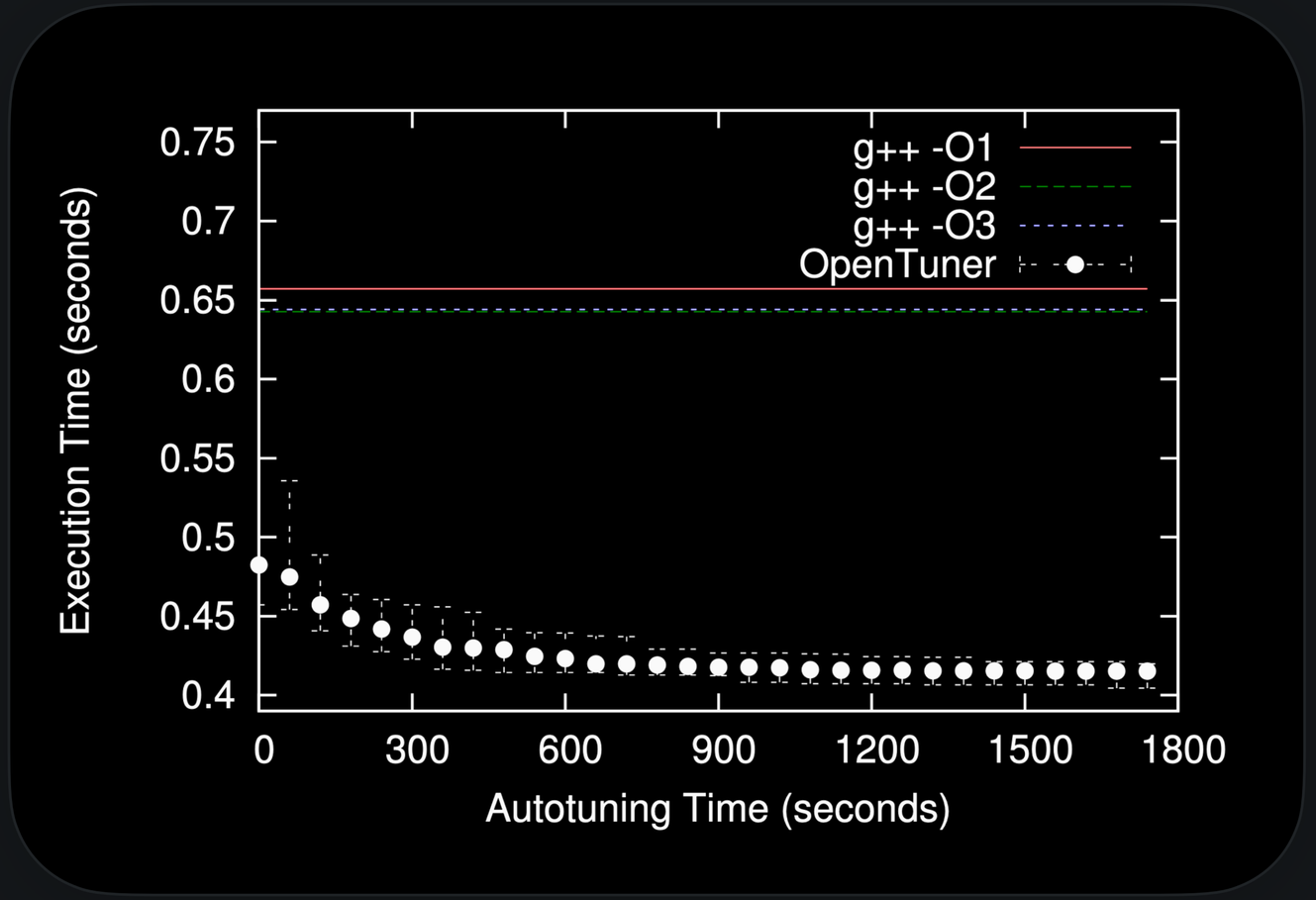
Experiment – GCC Flags



Evaluation

Experiment – GCC Flags

CONFIGURATION

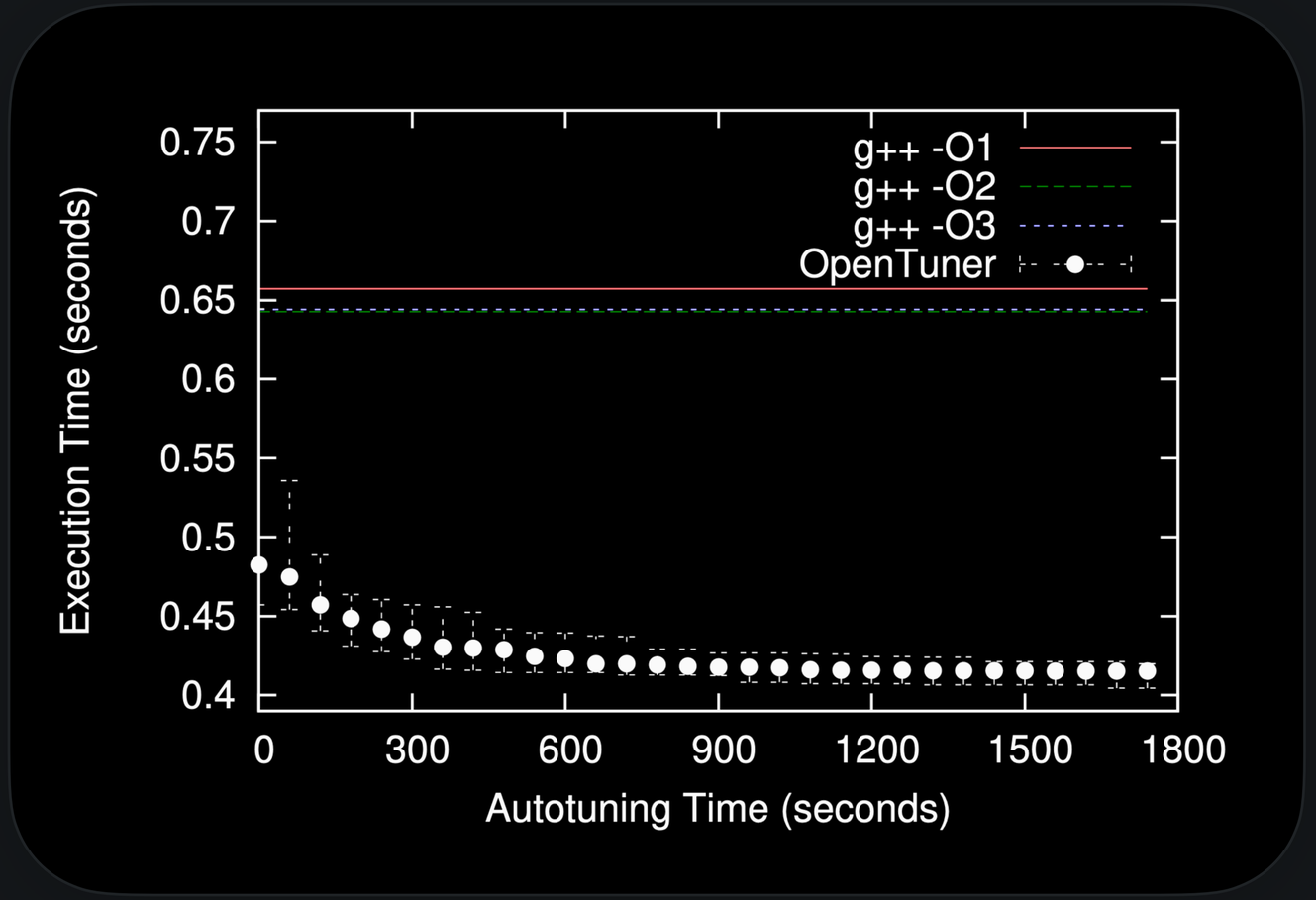


Evaluation

Experiment – GCC Flags

CONFIGURATION

Compiler flags



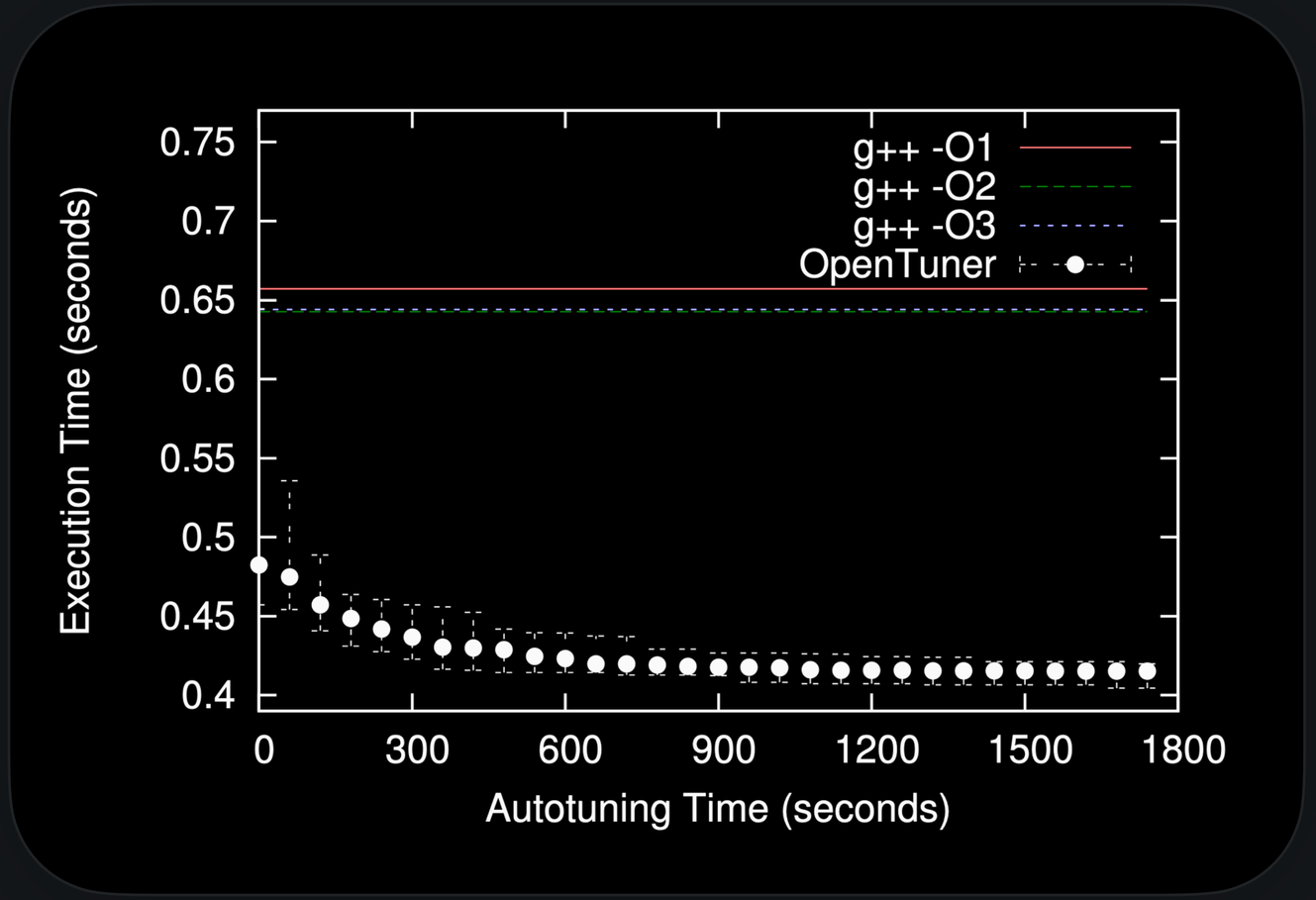
Evaluation

Experiment – GCC Flags

CONFIGURATION

Compiler flags

OBJECTIVE



Evaluation

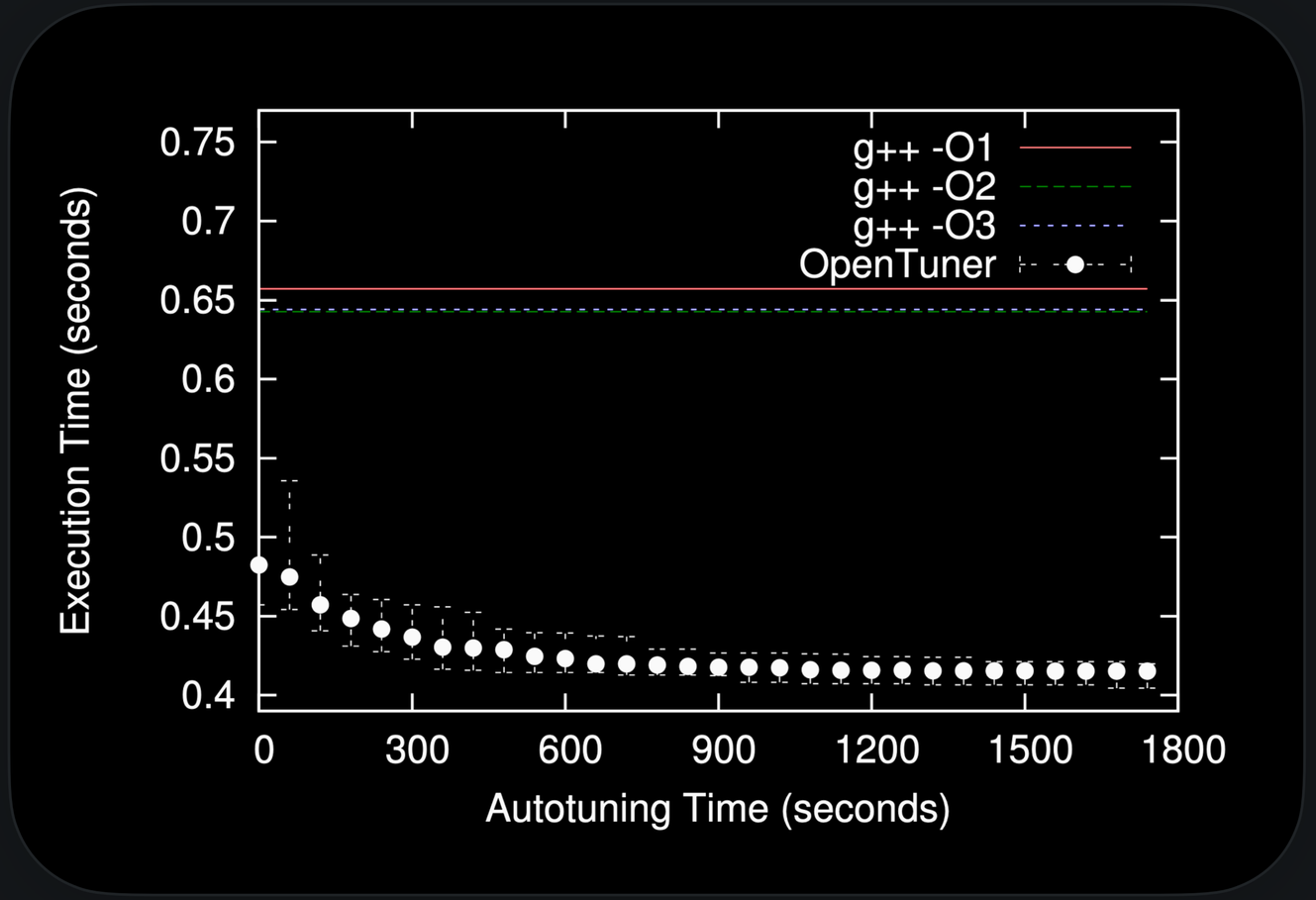
Experiment – GCC Flags

CONFIGURATION

Compiler flags

OBJECTIVE

⏱ Execution Time



Evaluation

Experiment – GCC Flags

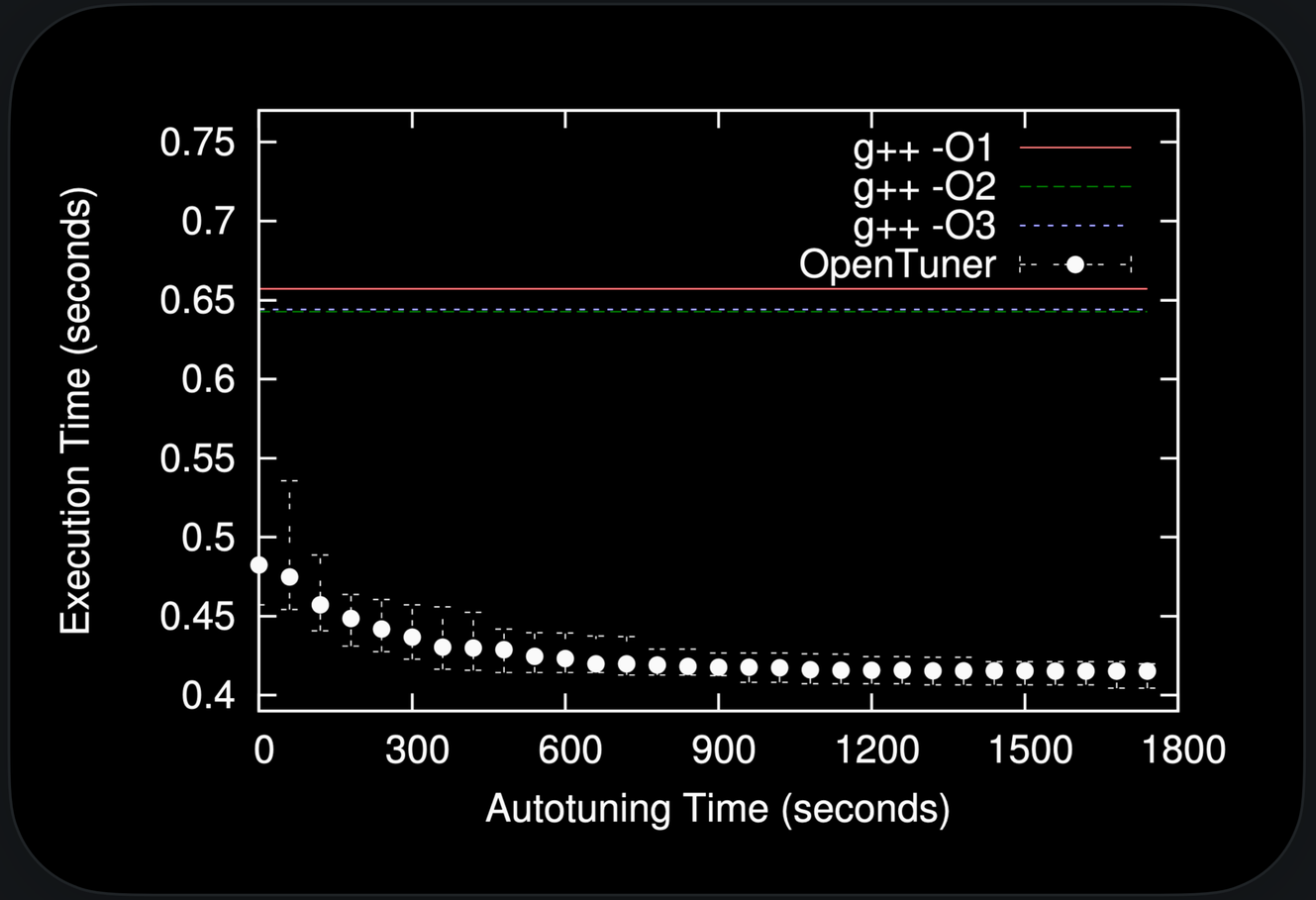
CONFIGURATION

Compiler flags

OBJECTIVE

⏱ Execution Time

RESULTS



Evaluation

Experiment – GCC Flags

CONFIGURATION

Compiler flags

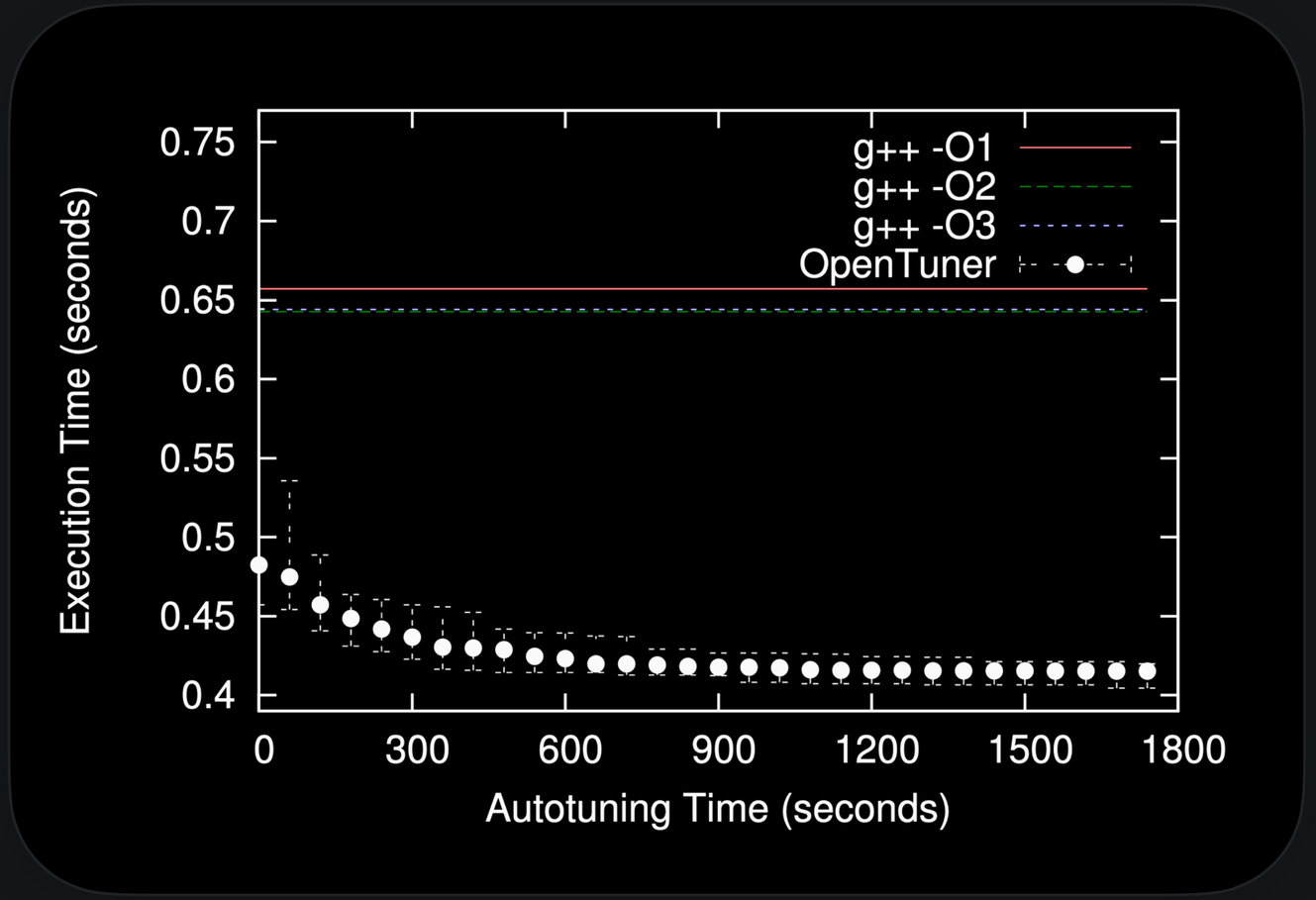
OBJECTIVE

⏱ Execution Time

RESULTS

Raytracing

≈ −30%



Evaluation

Experiment – GCC Flags

CONFIGURATION

Compiler flags

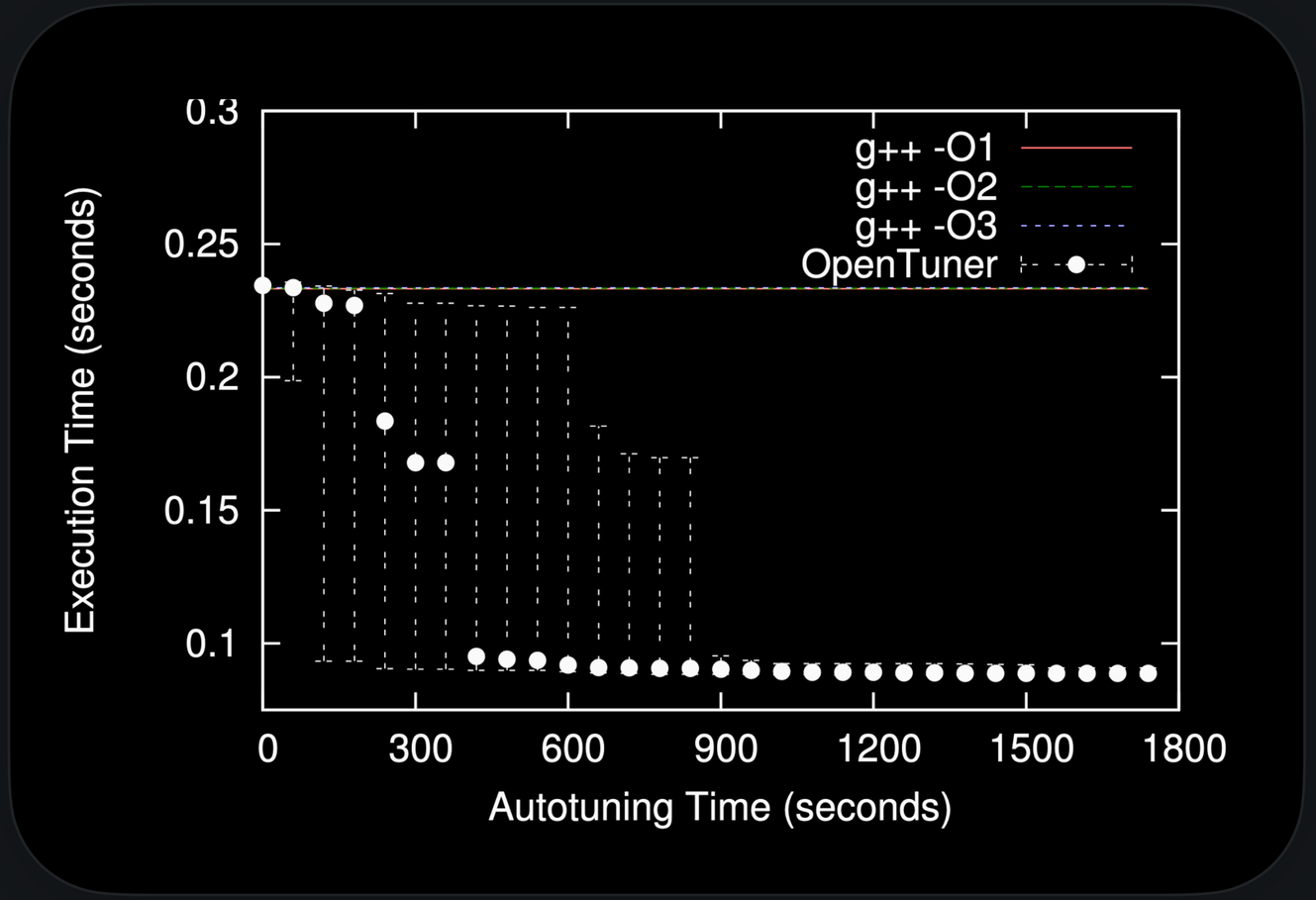
OBJECTIVE

⏱ Execution Time

RESULTS

Raytracing

≈ −30%



Evaluation

Experiment – GCC Flags

CONFIGURATION

Compiler flags

OBJECTIVE

⏱ Execution Time

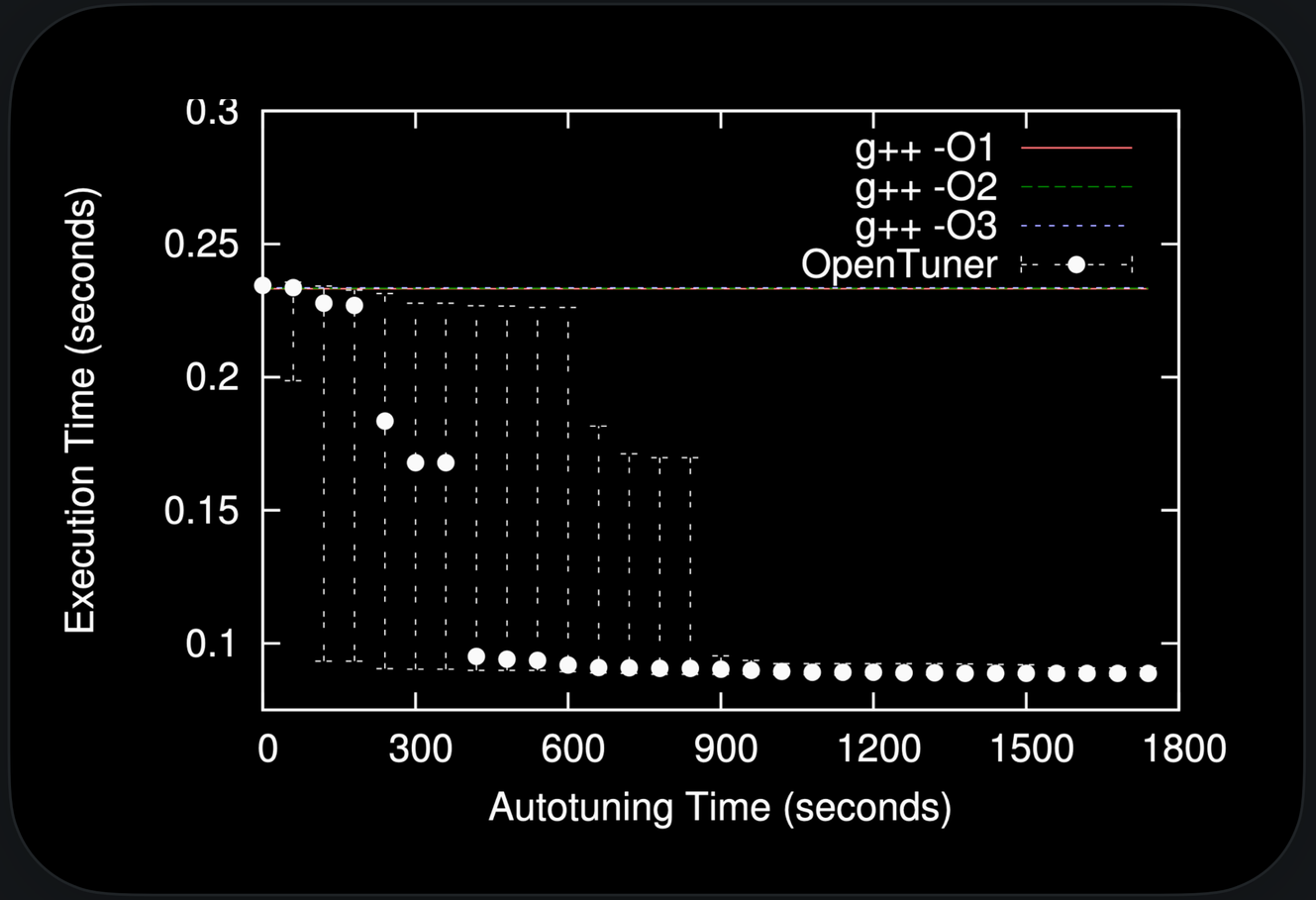
RESULTS

Raytracing

≈ −30%

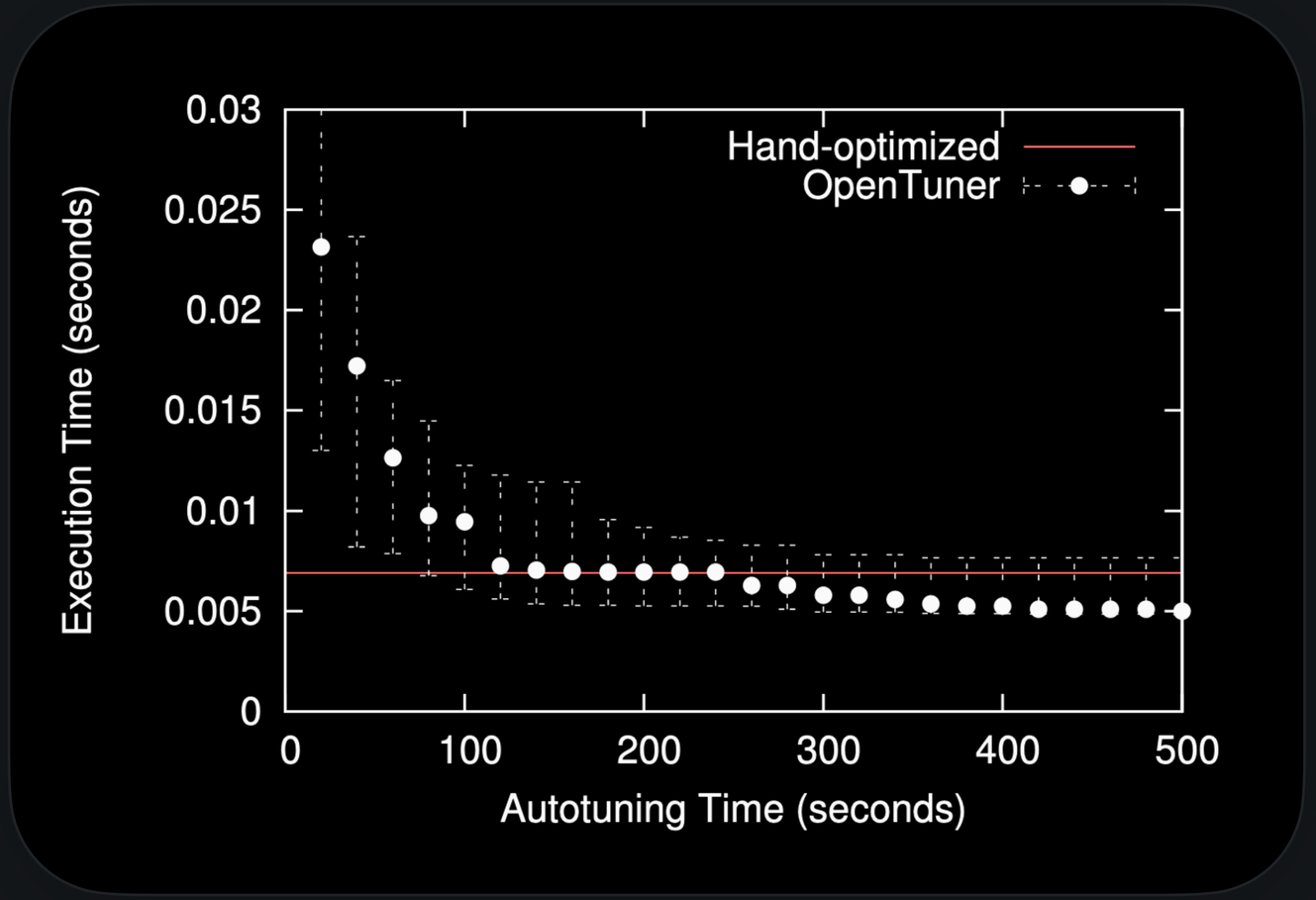
Matrix Multiplication

≈ −60%



Evaluation

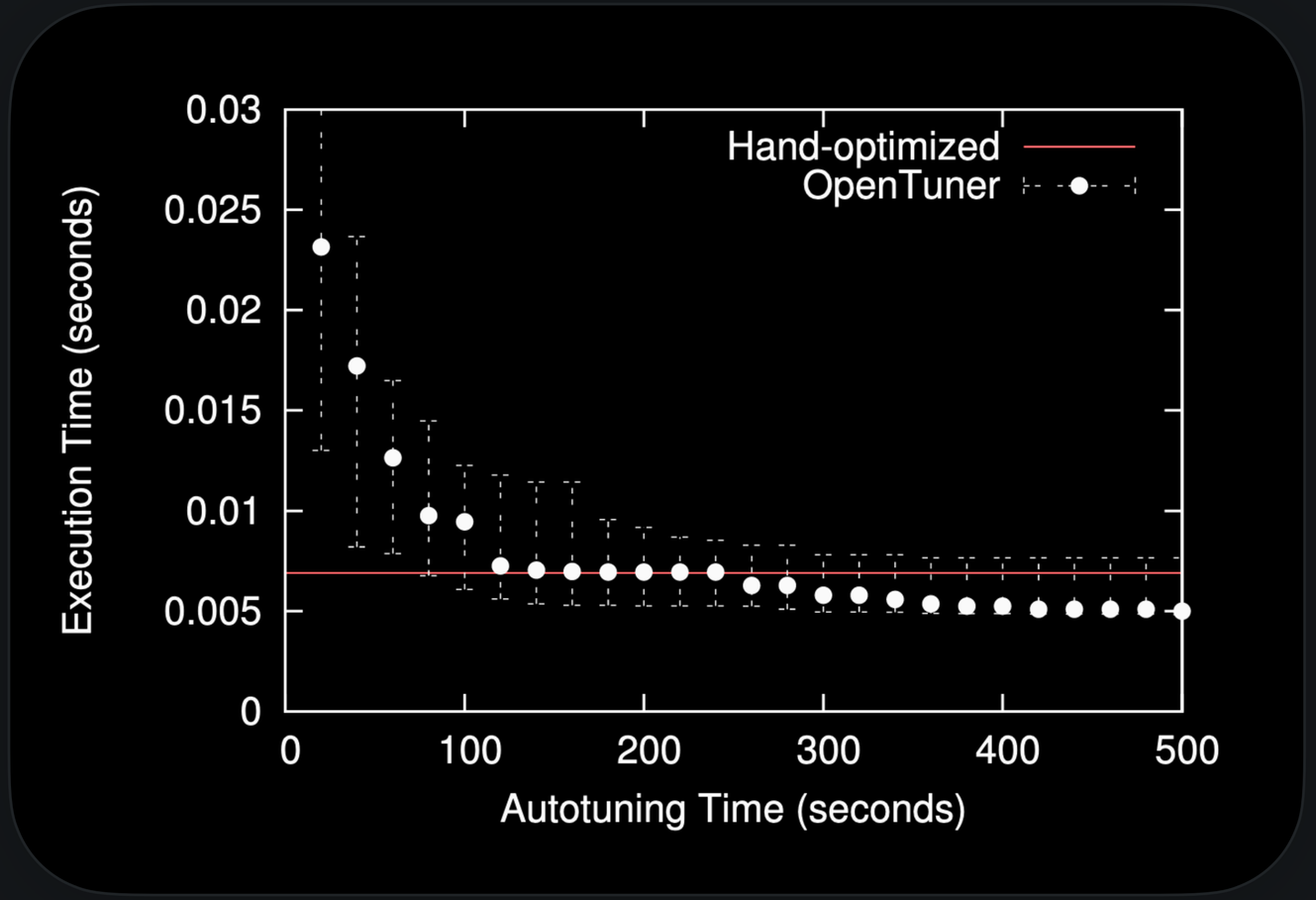
Experiment – Halide Graphics Processing Pipeline Schedule



Evaluation

Experiment – Halide Graphics Processing Pipeline Schedule

CONFIGURATION

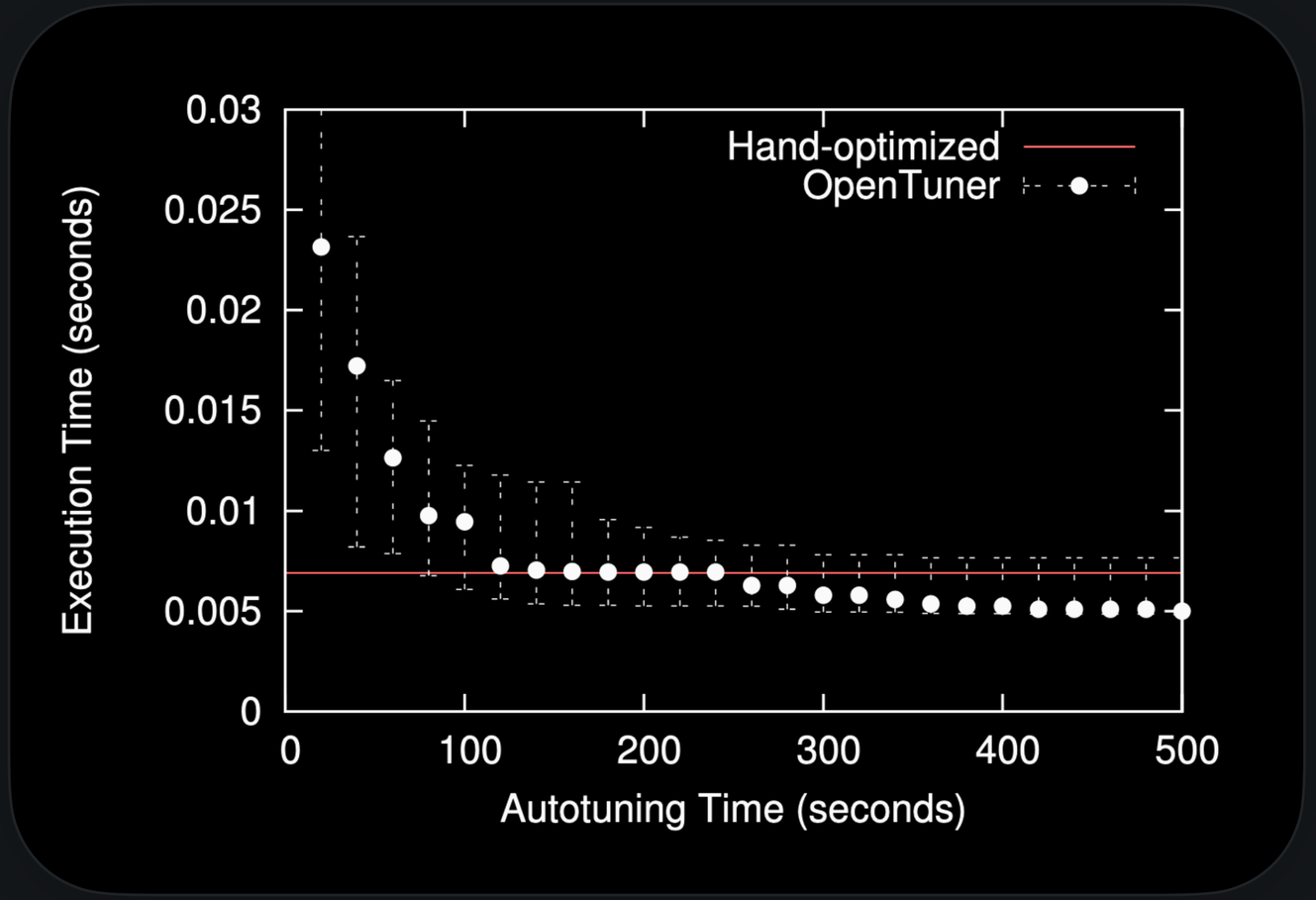


Evaluation

Experiment – Halide Graphics Processing Pipeline Schedule

CONFIGURATION

Pipeline schedule order



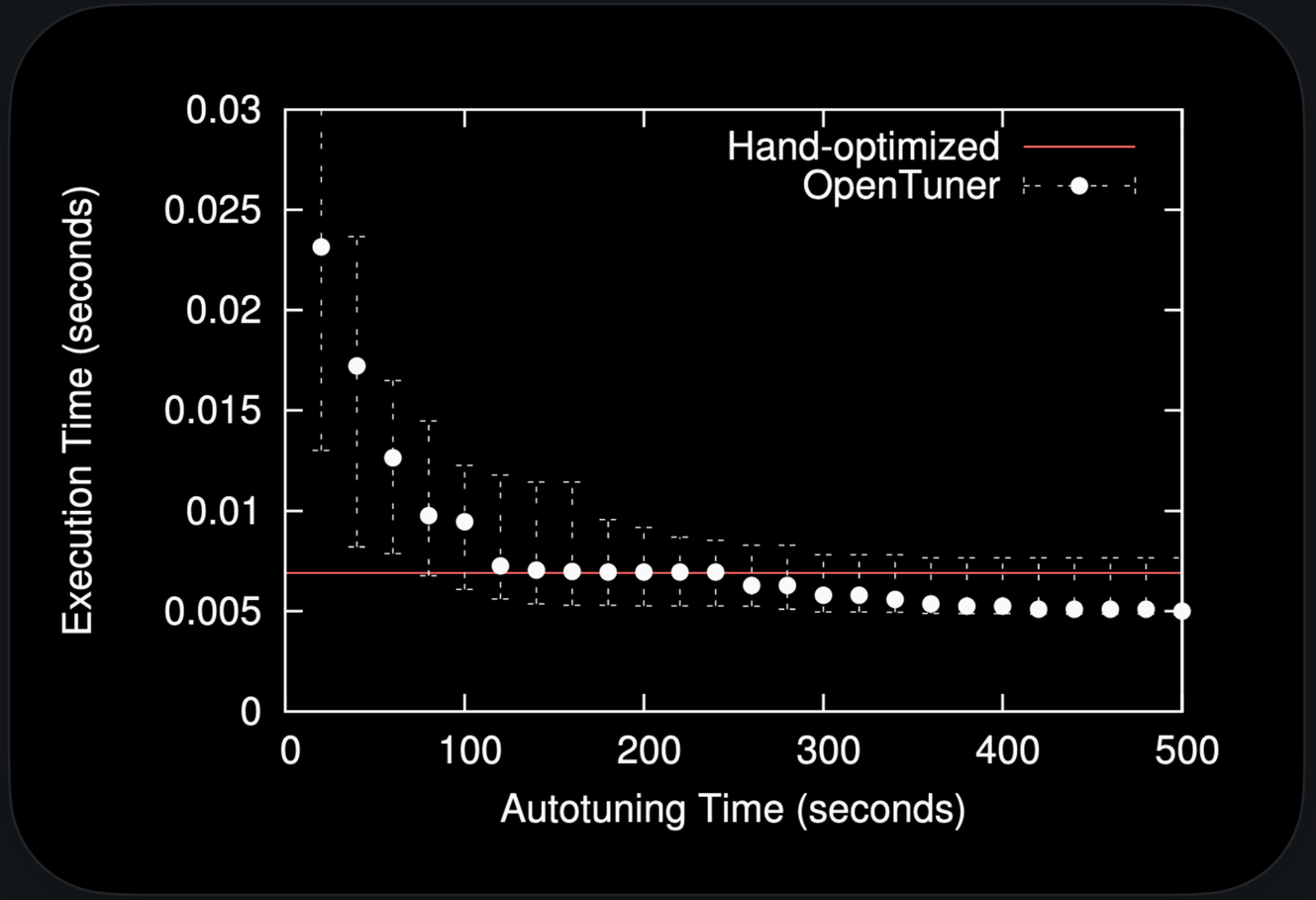
Evaluation

Experiment – Halide Graphics Processing Pipeline Schedule

CONFIGURATION

Pipeline schedule order

OBJECTIVE



Evaluation

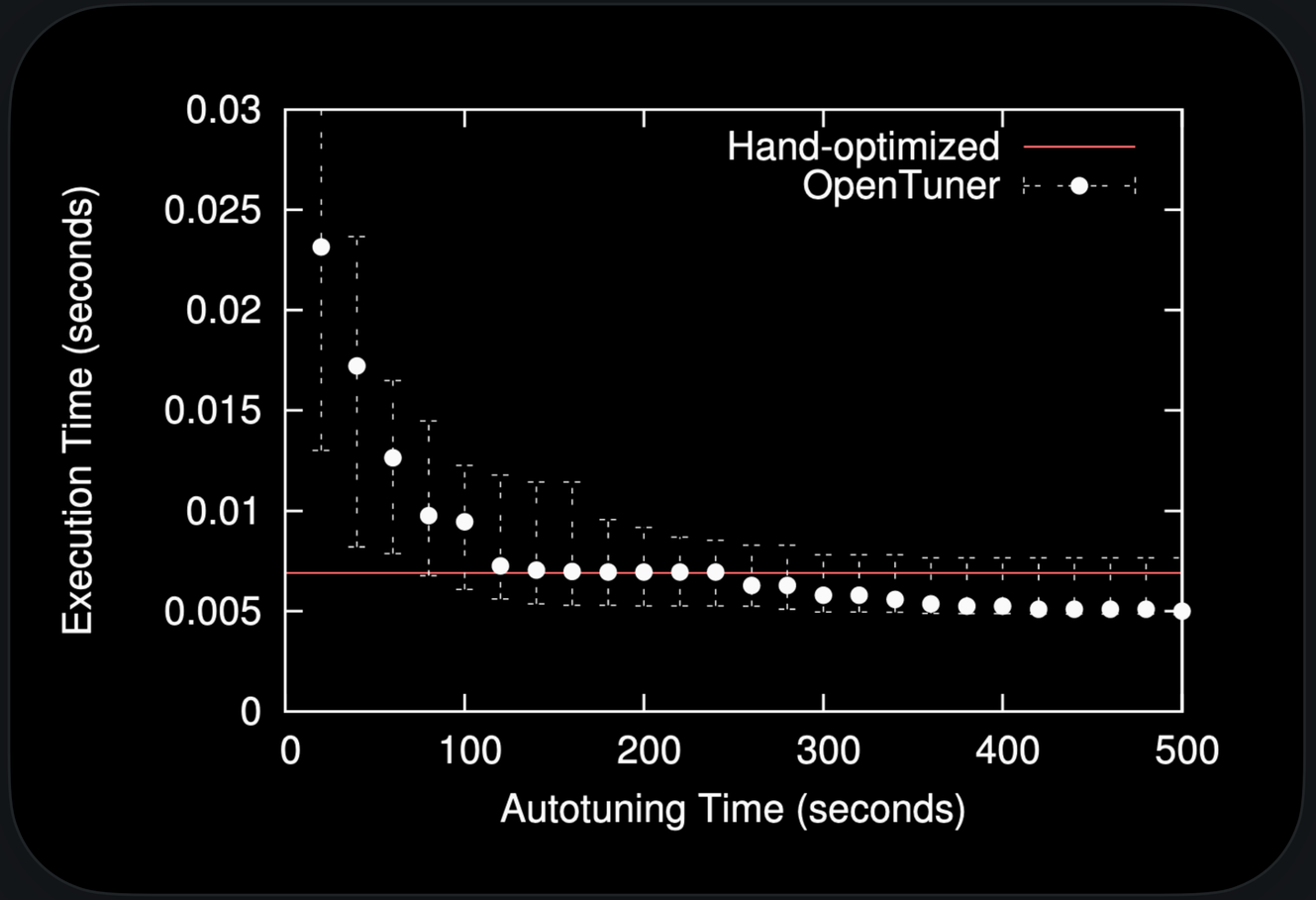
Experiment – Halide Graphics Processing Pipeline Schedule

CONFIGURATION

Pipeline schedule order

OBJECTIVE

⏱ Execution Time



Evaluation

Experiment – Halide Graphics Processing Pipeline Schedule

CONFIGURATION

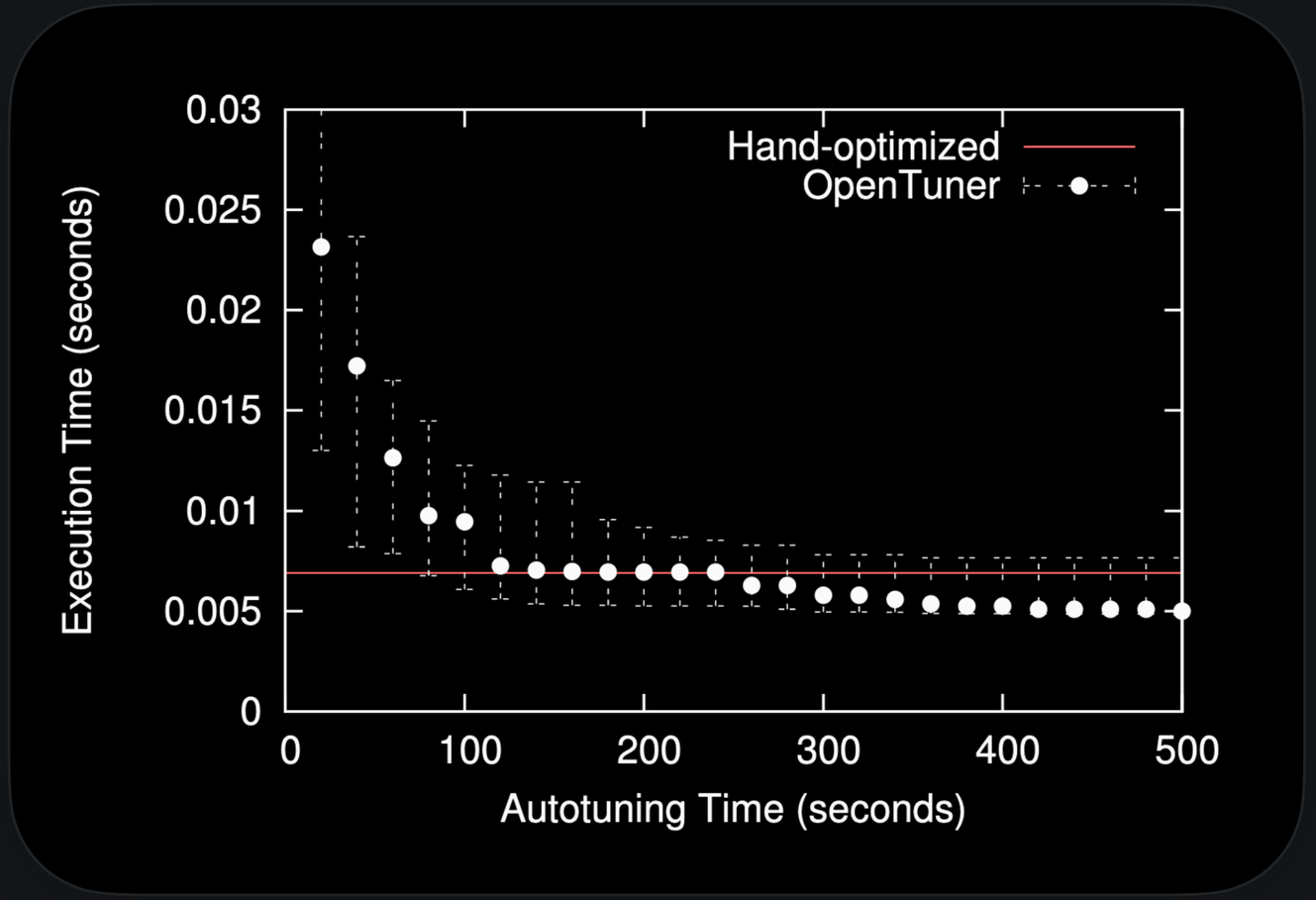
Pipeline schedule order

OBJECTIVE

⏱ Execution Time

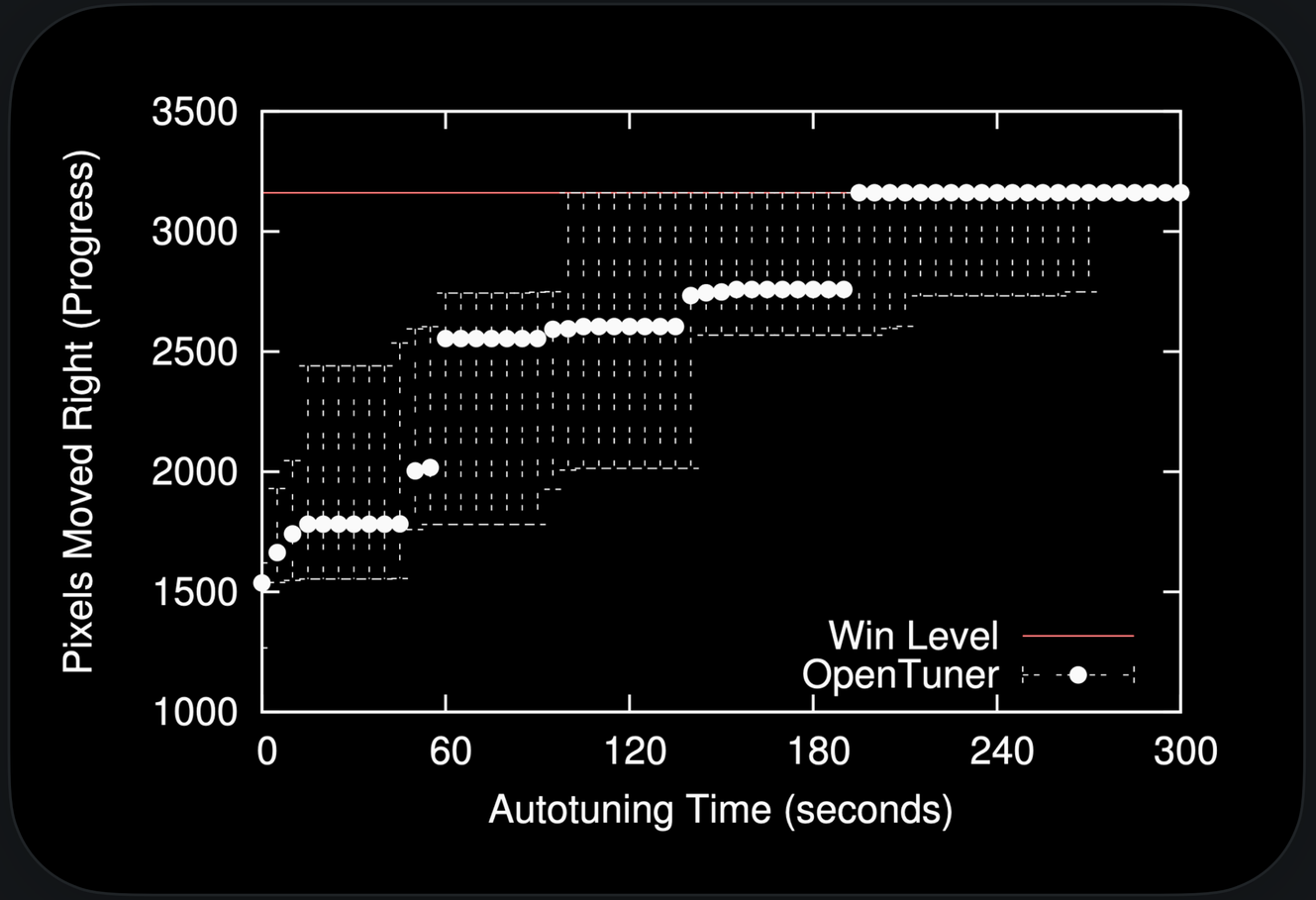
RESULTS

Blur Pipeline
reaches hand-optimized performance



Evaluation

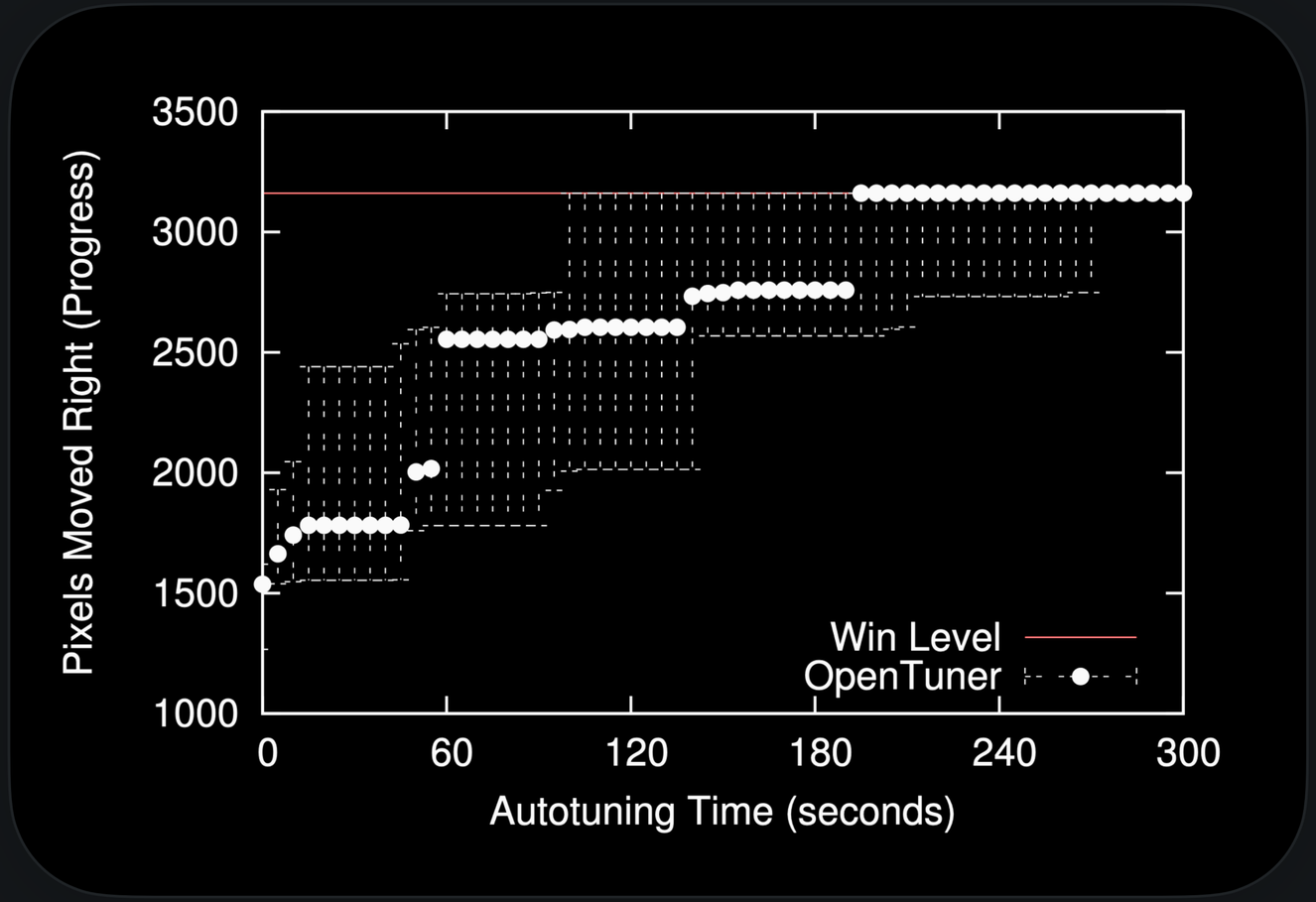
Experiment – Super Mario Bros.



Evaluation

Experiment – Super Mario Bros.

CONFIGURATION

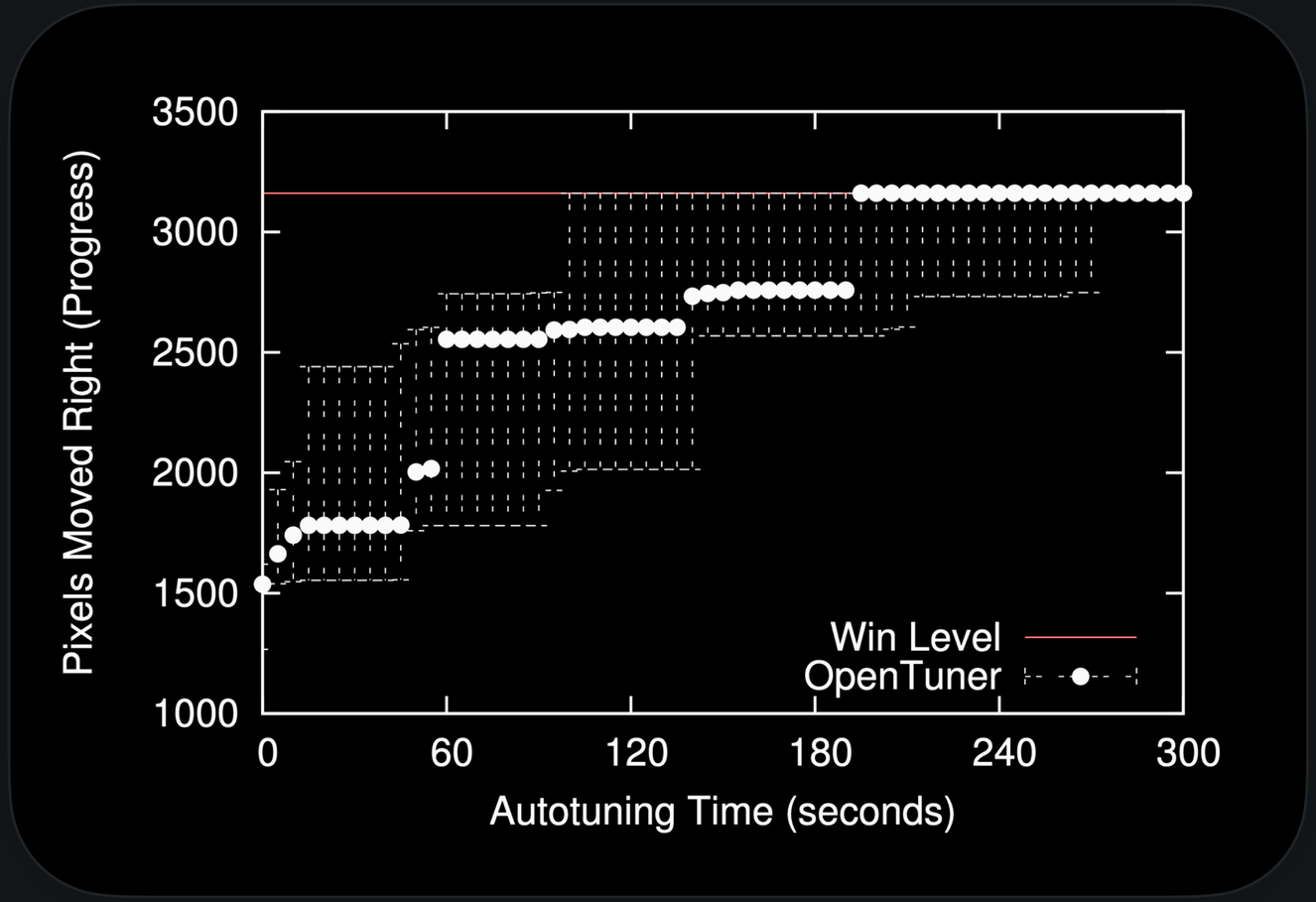


Evaluation

Experiment – Super Mario Bros.

CONFIGURATION

Sequence of button presses



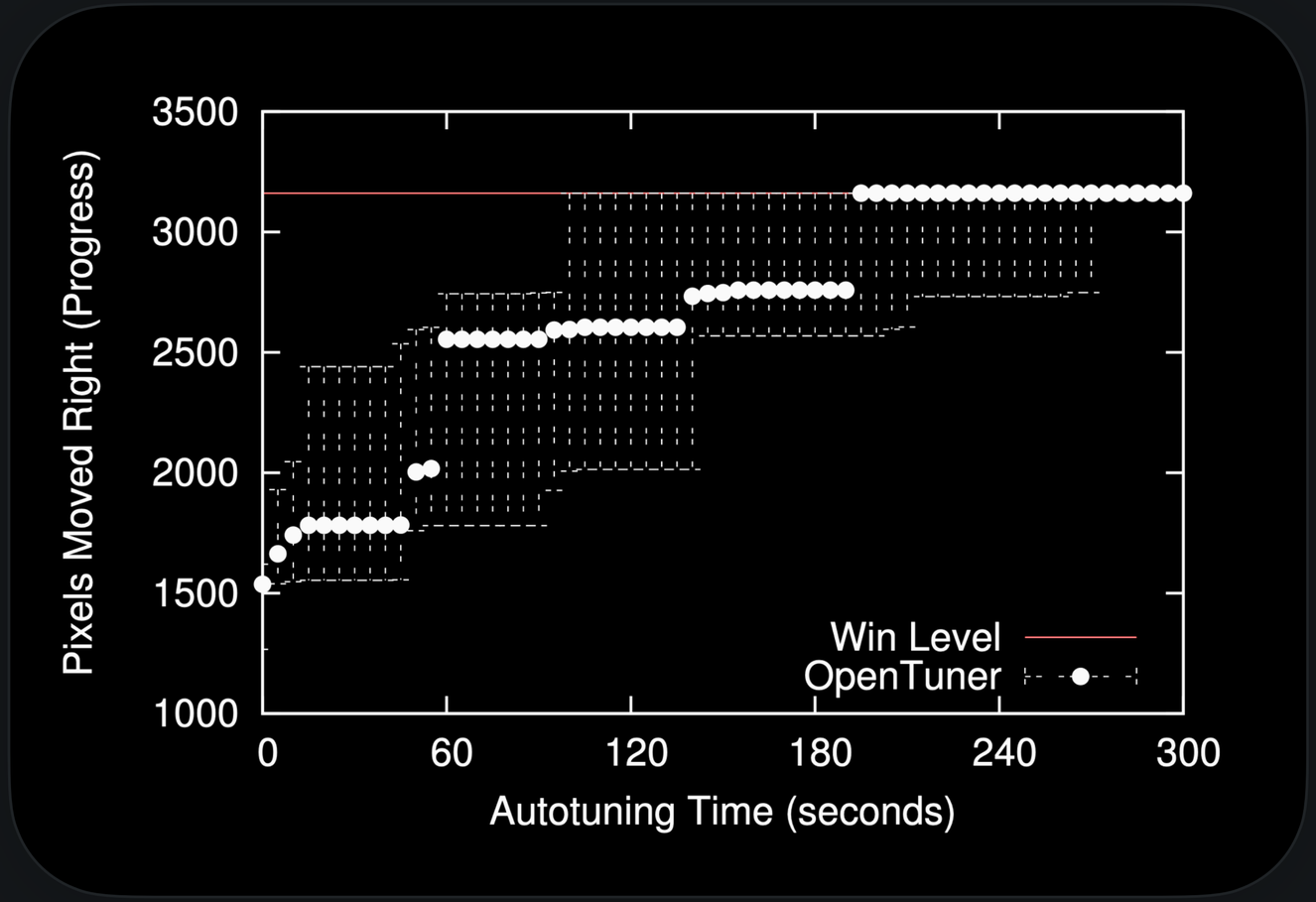
Evaluation

Experiment – Super Mario Bros.

CONFIGURATION

Sequence of button presses

OBJECTIVE



Evaluation

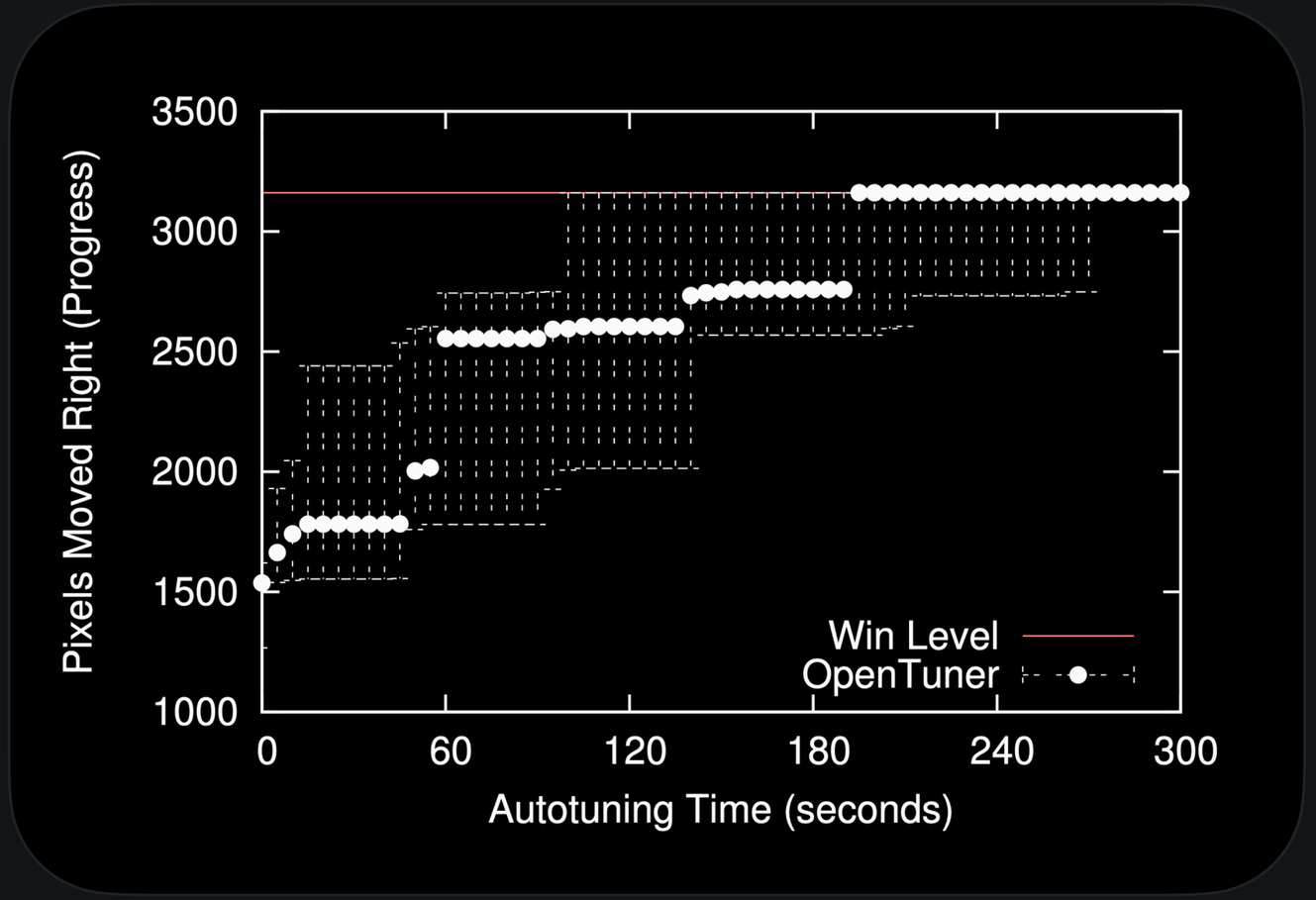
Experiment – Super Mario Bros.

CONFIGURATION

Sequence of button presses

OBJECTIVE

🏅 Execution Time



Evaluation

Experiment – Super Mario Bros.

CONFIGURATION

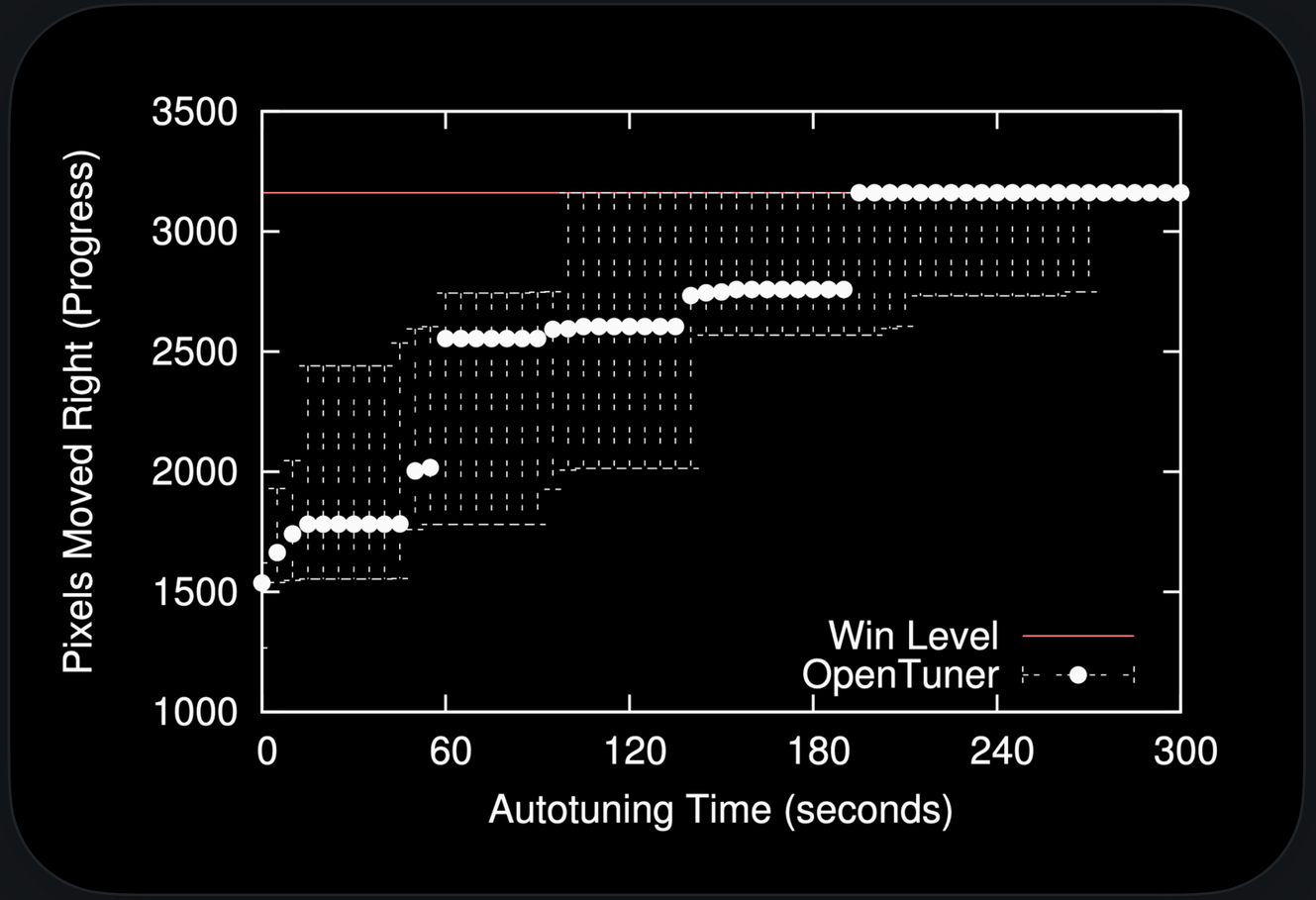
Sequence of button presses

OBJECTIVE

🏅 Execution Time

RESULTS

Successful coordination of jump: right moment, duration, speed, return on pit



Conclusions

[28] J. Ansel et al.: Petabricks: A language and compiler for algorithmic choice. In Proceedings of the 2009 ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI, 2009.

Conclusions

Motivated by *PetaBricks*[28]

[28] J. Ansel et al.: Petabricks: A language and compiler for algorithmic choice. In Proceedings of the 2009 ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI, 2009.

Conclusions

Motivated by *PetaBricks*[28]

Configuration representation + search techniques
must be adapted to domain!

[28] J. Ansel et al.: Petabricks: A language and compiler for algorithmic choice. In Proceedings of the 2009 ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI, 2009.

Conclusions

Motivated by *PetaBricks*[28]

Configuration representation + search techniques must be adapted to domain!

Multiple concurrent search techniques

[28] J. Ansel et al.: Petabricks: A language and compiler for algorithmic choice. In Proceedings of the 2009 ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI, 2009.

Conclusions

Motivated by *PetaBricks*[28]

Configuration representation + search techniques must be adapted to domain!

- Multiple concurrent search techniques
- Extensible configuration abstraction

[28] J. Ansel et al.: Petabricks: A language and compiler for algorithmic choice. In Proceedings of the 2009 ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI, 2009.

Conclusions

Motivated by *PetaBricks*[28]

Configuration representation + search techniques must be adapted to domain!

- Multiple concurrent search techniques
- Extensible configuration abstraction

FOSS software

[28] J. Ansel et al.: Petabricks: A language and compiler for algorithmic choice. In Proceedings of the 2009 ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI, 2009.

Conclusions

Motivated by *PetaBricks*[28]

Configuration representation + search techniques must be adapted to domain!

- Multiple concurrent search techniques
- Extensible configuration abstraction

FOSS software

→ **Portable autotuner**

Thoughts...

[7] V. Dalibard, M. Schaarschmidt, and E. Yoneki: BOAT: Building Auto-Tuners with Structured Bayesian Optimization, WWW, 2017.
[32] S. Cereda, S. Valladares, P. Cremonesi and S. Doni: CGPTuner: a Contextual Gaussian Process Bandit Approach for the Automatic Tuning IT Configurations Under Varying Workload Conditions, VLDB, 2021.

Thoughts...

- Extensibility + Abstractions

[7] V. Dalibard, M. Schaarschmidt, and E. Yoneki: BOAT: Building Auto-Tuners with Structured Bayesian Optimization, WWW, 2017.
[32] S. Cereda, S. Valladares, P. Cremonesi and S. Doni: CGPTuner: a Contextual Gaussian Process Bandit Approach for the Automatic Tuning IT Configurations Under Varying Workload Conditions, VLDB, 2021.

Thoughts...

- Extensibility + Abstractions
- Portability from domain-agnostic design

[7] V. Dalibard, M. Schaarschmidt, and E. Yoneki: BOAT: Building Auto-Tuners with Structured Bayesian Optimization, WWW, 2017.
[32] S. Cereda, S. Valladares, P. Cremonesi and S. Doni: CGPTuner: a Contextual Gaussian Process Bandit Approach for the Automatic Tuning IT Configurations Under Varying Workload Conditions, VLDB, 2021.

Thoughts…

- ➕ Extensibility + Abstractions
- ➕ Portability from domain-agnostic design
- ➖ No feature to express a configurations' direct effect on objective, only as a whole through measurement (but: trivial fix)

[7] V. Dalibard, M. Schaarschmidt, and E. Yoneki: BOAT: Building Auto-Tuners with Structured Bayesian Optimization, WWW, 2017.
[32] S. Cereda, S. Valladares, P. Cremonesi and S. Doni: CGPTuner: a Contextual Gaussian Process Bandit Approach for the Automatic Tuning IT Configurations Under Varying Workload Conditions, VLDB, 2021.

Thoughts…

- \+ Extensibility + Abstractions
- \+ Portability from domain-agnostic design
- − No feature to express a configurations' direct effect on objective,
  only as a whole through measurement (but: trivial fix)
- − GCC flags example
  desirable behavior, sometimes trivial flags

[7] V. Dalibard, M. Schaarschmidt, and E. Yoneki: BOAT: Building Auto-Tuners with Structured Bayesian Optimization, WWW, 2017.

[32] S. Cereda, S. Valladares, P. Cremonesi and S. Doni: CGPTuner: a Contextual Gaussian Process Bandit Approach for the Automatic Tuning IT Configurations Under Varying Workload Conditions, VLDB, 2021.

Thoughts…

- ➕ Extensibility + Abstractions
- ➕ Portability from domain-agnostic design
- ➖ No feature to express a configurations' direct effect on objective,
  only as a whole through measurement (but: trivial fix)
- ➖ GCC flags example
  desirable behavior, sometimes trivial flags
- ❓ Maintainability of autotuned configs + practicality if multiple target platforms

[7] V. Dalibard, M. Schaarschmidt, and E. Yoneki: BOAT: Building Auto-Tuners with Structured Bayesian Optimization, WWW, 2017.
[32] S. Cereda, S. Valladares, P. Cremonesi and S. Doni: CGPTuner: a Contextual Gaussian Process Bandit Approach for the Automatic Tuning IT Configurations Under Varying Workload Conditions, VLDB, 2021.

Thoughts…

- ➕ Extensibility + Abstractions
- ➕ Portability from domain-agnostic design
- ➖ No feature to express a configurations' direct effect on objective,
  only as a whole through measurement (but: trivial fix)
- ➖ GCC flags example
  desirable behavior, sometimes trivial flags
- ❓ Maintainability of autotuned configs + practicality if multiple target platforms
- ❓ OpenTuner not being developed further

[7] V. Dalibard, M. Schaarschmidt, and E. Yoneki: BOAT: Building Auto-Tuners with Structured Bayesian Optimization, WWW, 2017.

[32] S. Cereda, S. Valladares, P. Cremonesi and S. Doni: CGPTuner: a Contextual Gaussian Process Bandit Approach for the Automatic Tuning IT Configurations Under Varying Workload Conditions, VLDB, 2021.

Thoughts…

- \+ Extensibility + Abstractions
- \+ Portability from domain-agnostic design
- − No feature to express a configurations' direct effect on objective,
  only as a whole through measurement (but: trivial fix)
- − GCC flags example
  desirable behavior, sometimes trivial flags
- ? Maintainability of autotuned configs + practicality if multiple target platforms
- ? OpenTuner not being developed further
- ? Today: Bayesian Optimisation (BOAT[7], CPGTuner[32])

[7] V. Dalibard, M. Schaarschmidt, and E. Yoneki: BOAT: Building Auto-Tuners with Structured Bayesian Optimization, WWW, 2017.

[32] S. Cereda, S. Valladares, P. Cremonesi and S. Doni: CGPTuner: a Contextual Gaussian Process Bandit Approach for the Automatic Tuning IT Configurations Under Varying Workload Conditions, VLDB, 2021.

OpenTuner

An Extensible Framework for Program Autotuning

Jason Ansel, Shoaib Kamil, Kalyan Veeramachaneni, Jonathan Ragan-Kelley,
Jeffrey Bosboom, Una-May O'Reilly, and Saman Amarasinghe

Carl Seifert
cs2331@cam.ac.uk

Font: rsms.me/inter. Icons: fonts.google.com/icons