ML-based Tuning of RNTuple I/O Parameters

Dante Niewenhuis
University of Amsterdam
dante.niewenhuis@cern.ch

Giovanna Lazzari Miotto
UFRGS (BR)
giovanna.lazzari.miotto@cern.ch

Javier Lopez-Gomez
CERN
javier.lopez.gomez@cern.ch

Jakob Blomer
CERN
jakob.blomer@cern.ch

Introduction

Reading and writing ROOT files is affected by I/O parameters. The influence of the different I/O parameters on the effectiveness of the machine is, however, not very well-known. In this work we explore effective I/O parameter configurations using AI methods.

Goals:

1. Find analysis-dependent configurations that work well for that particular analysis
2. Find out whether there exists a common set of I/O parameters that work well enough regardless of the analysis task
3. Gain insight into the roles of I/O parameters

Parameters

Four tunable parameters are used that can take values from a predefined list. When mutating a parameter, discrete variables can only take a neighboring value while categorical parameters can take any other value.

Parameter	Type	Values	Base
Compression	Categorical	[none, zlib, lz4, lzma, zstd]	lz4
Page Size	Discrete	[16 KB, 32 KB, 64 KB, 128 KB, 256 KB, 512 KB, 1 MB, 2 MB, 4 MB, 8 MB, 16 MB]	64 KB
Cluster Size	Discrete	[20 MB, 30 MB, 40 MB, 50 MB, 100 MB, 200 MB, 300 MB, 400 MB, 500 MB]	50 MB
Cluster Bunch	Discrete	[1, 2, 3, 4, 5]	1

Tab 1: Parameters used in this work with their base values.

Single Benchmark

- Evolve an optimal configuration for all benchmarks
- All benchmarks resulting configurations are different

ATLAS

Parameter	Value
Compression	lz4
Page Size	4 MB
Cluster Size	20 MB
Cluster Bunch	2

CMS

Parameter	Value
Compression	zstd
Page Size	1 MB
Cluster Size	500 MB
Cluster Bunch	2

H1

Parameter	Value
Compression	zstd
Page Size	8 MB
Cluster Size	400 MB
Cluster Bunch	2

LHCb

Parameter	Value
Compression	lz4
Page Size	4 MB
Cluster Size	20 MB
Cluster Bunch	4

Tab 2: Optimal configuration for the different benchmarks

- The configurations do not all perform well on the other benchmarks
- On the CMS benchmark, the differences between configurations is the biggest

Bench / Config	Atlas	CMS	H1	LHCB
Atlas	**45.39**	17.158	47.913	29.329
CMS	34.461	**67.675**	43.396	26.893
H1	34.98	58.522	**78.383**	25.886
LHCB	43.082	9.943	44.606	**30.619**

Tab 3: Performance of each Optimal configuration evaluated on the other benchmarks

Aggregating Benchmarks

- Optimize the configuration for the average performance on all four benchmarks.
- Good on all benchmarks, but not great on any.

Parameter	Value
Compression	lz4
Page Size	2 MB
Cluster Size	500 MB
Cluster Bunch	3

Tab 3: Optimal configuration for average benchmark

Atlas	CMS	H1	LHCB
44.786	60.184	58.322	29.280

Tab 4: Result of the optimal average configuration

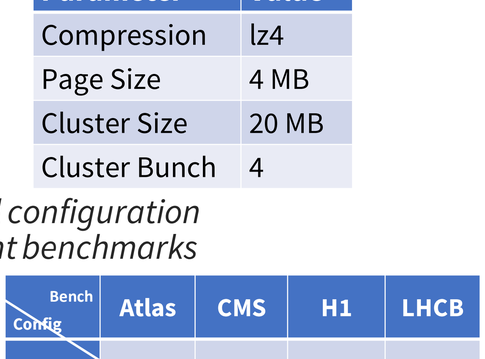

Fig 1: Performance in each benchmark and the average during evolving

Determining Performance

- Four different benchmarks are used (ATLAS, CMS, H1, LHCb) [2]
- All benchmarks consist of generating a file, and processing it
- Throughput and size of the generated file are used as metrics
- Both are normalized using a base configuration

$$TI_c = \frac{T_c - T_{base}}{T_{base}} * 100\% \qquad SD_c = \frac{S_{base} - S_c}{S_{base}} * 100\%$$

- Performance combines the two metrics
- The metrics are weighted using TR (in this work TR=.5)

$$P_c = TR * TI_c + (1 - TR) * SD_c$$

AI model

Hillclimber:

1. Mutate current configuration
2. Determine new performance
3. If higher, keep new configuration

Problem: susceptible to get stuck in local maxima.

Simulated Annealing:

- Add chance to accept mutations, even if they decrease performance.
- Probability to accept is based on a decreasing temperature (fig 2&3).

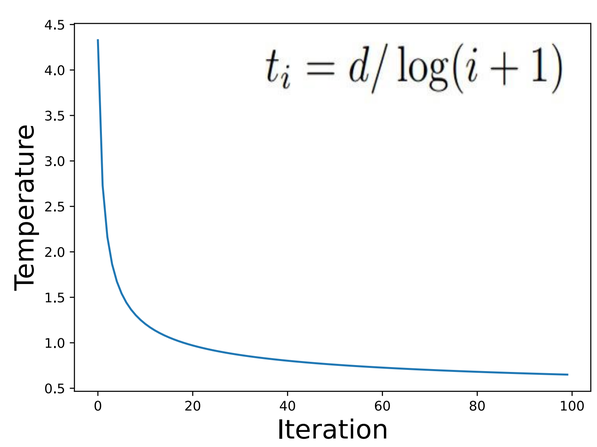

Fig 2: Temperature *t* at iteration *i*

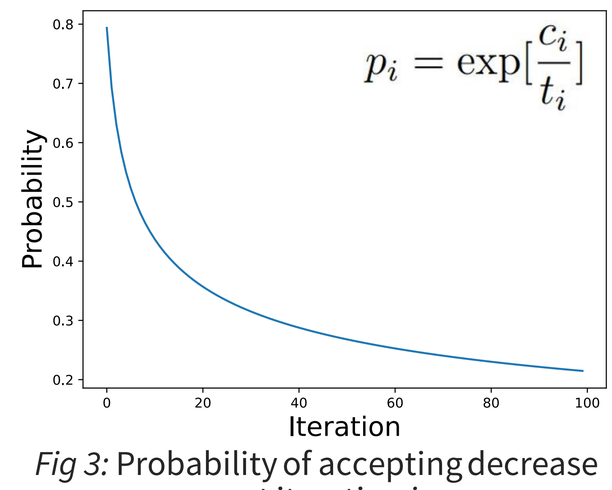

Fig 3: Probability of accepting decrease c at iteration *i*

Solution Space

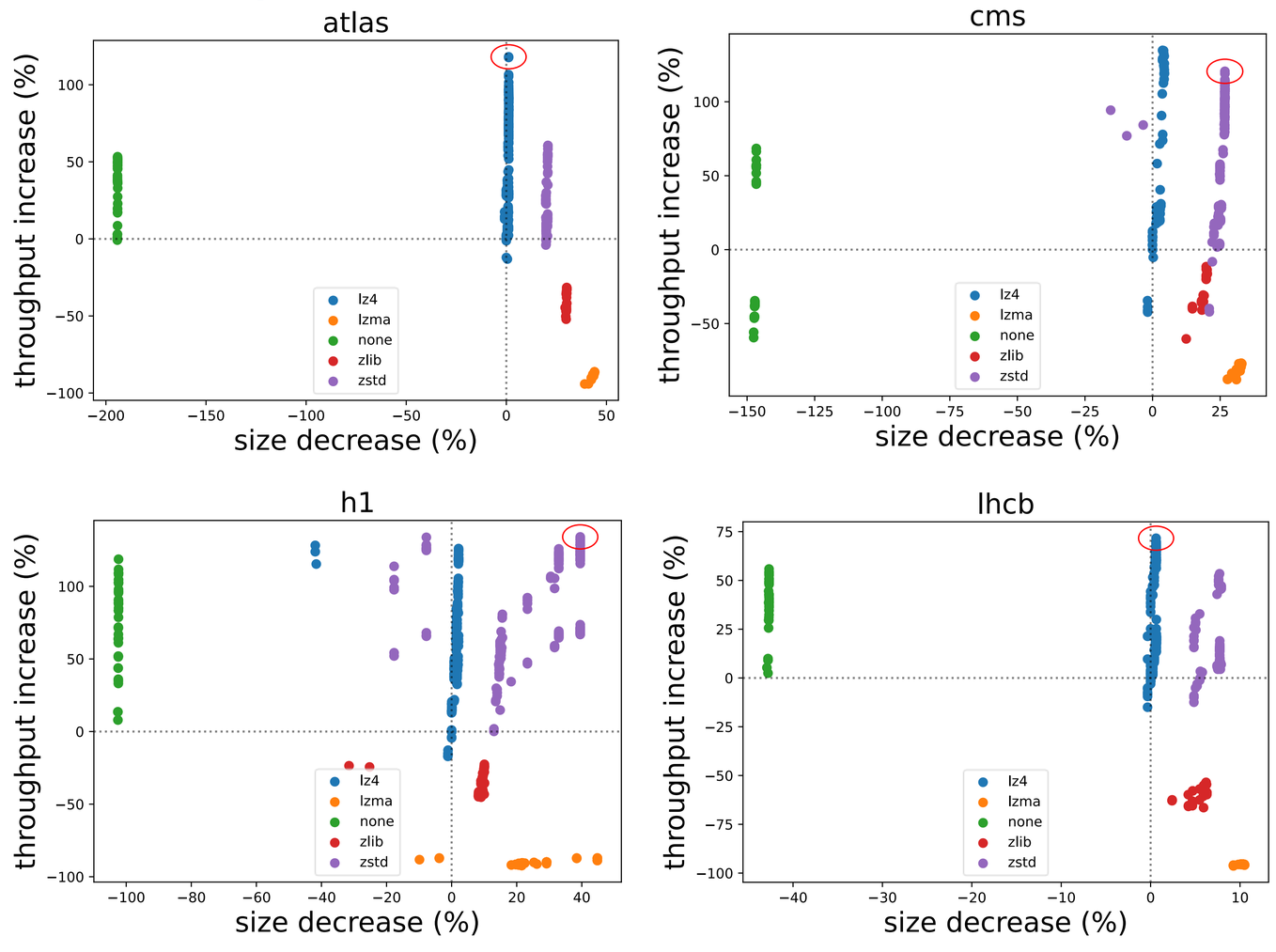

Fig 4: Scatter of all performance results on each of the four benchmarks

Conclusion

The parameters were successfully optimized. However, no configuration performed optimal on all benchmarks. The best performing configuration were too memory intensive to be viable.

Future Work

- Add memory as metric
- Tuning more parameters
- Genetic Algorithms

References

[1] https://github.com/DanteNiewenhuis/CERN-parameter-optimization
[2] https://github.com/jblomer/iotools