

JEnt on Heterogeneous Computing Architectures

(A Look at JEnt v3.6.3 Performance on the Apple M4)

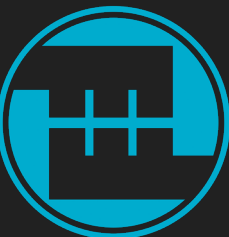
Joshua E. Hill, PhD



20260428
CMUF EWG

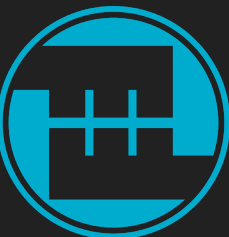
Introduction

- Heterogeneous Computing Architectures (e.g., big.LITTLE, P-cores/E-cores) are used in many settings where power efficiency is important.
- This type of architecture provides multiple architectures on a single chip.
 - This leads to complications in JEnt validation.
- One such architecture is the Apple M chip.
 - Thanks to Mark Smith (Halibut Electronics) for providing access to an Apple Mac Mini (2024) with an M4 chip.



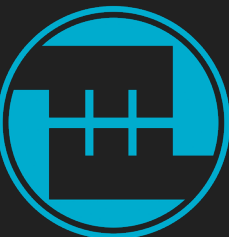
Background

- Heterogeneous Computing Architectures has a long history but its penetration into general purpose computers is (hand wave) somewhat recent.
 - It has been commonly integrated into cooling- or energy-constrained embedded devices.
 - Cell phones
 - Passively cooled embedded systems.
- More recently, it has become common on general purpose computers (e.g., Apple's M chips, some recent Intel Core chips).



M4 Summary Observations

- The M4 uses a big.LITTLE processor architecture (big: P-core, LITTLE: E-core)
 - These chips essentially include two distinct CPU architectures (i.e., operating environments) on a single chip.
 - This forces the vendor / tester to either:
 - Use technical controls to force a particular core to be selected, or
 - Perform two distinct entropy evaluations (one for P-cores, one for E-cores) and selection of the worst-case assessment.
- The size of the memory region has a substantial impact on the amount of variation observed in JEnt timing.

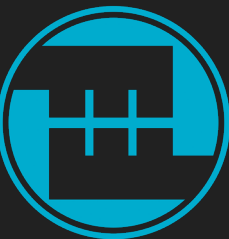


Initial Architectural Review



Hardware Notes

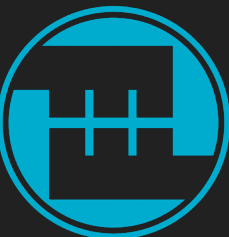
- The M4 is a modern AArch64 architecture.
 - The M4 (and the other M1-M5 chips) all use a heterogeneous architecture, with faster (and more power-intensive) P-cores and slower (and more power-efficient) E-cores.
 - The instruction set is version 9.2-A.
 - There are several variations of M4 chips.



Hardware Notes

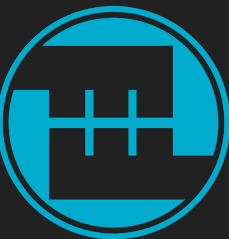
The system we tested is the baseline Apple Mac Mini (2024) configuration.

- 10-core (1 cluster of 4 P-cores, 1 cluster of 6 E-cores), advertised as operating at 1GHz to 4.5GHz.
- 16GB RAM, 16MB RAM cache:
 - A per-core 128kB (P-core) or 64kB (E-core) L1 cache.
 - A 16MB L2 cache is largely cluster-specific but can (at the cost of additional latency) be accessed across clusters.



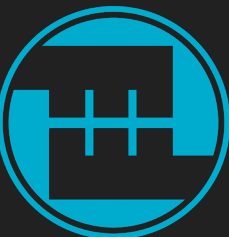
JEnt Architectural Notes

- In this environment the `clock_gettime_nsec_np(CLOCK_UPTIME_RAW)` call is used as the clock source.
- `CLOCK_UPTIME_RAW`:
 - Is a monotonic timer.
 - Reflects “wall time” while the system is not sleeping.
 - The core’s clock frequency alters the amount of wall time required for a fixed calculation.
 - For a fixed type of core, the expected raw delta value is inversely proportional to the clock frequency.



Hardware Notes

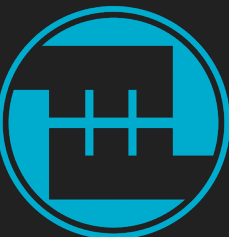
- The core and frequency selection is influenced by the process's QoS (Quality of Service).
- Foreground processes ($\text{QoS} > 9$):
 - Get assigned to a P-core and run at the maximum clock rate currently supported by the cluster unless there are no P-cores available.
 - If assigned to an E-core, that E-core is run at the highest clock rate supported by the cluster.
- Background processes ($\text{QoS} \leq 9$):
 - Are generally assigned to an E-core.
 - That E-core is run near the bottom of its clock rate range (1GHz).



Hardware Notes

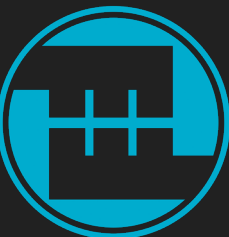
Thermal throttling and CPU loading impact the core's selected clock frequency.

- P-cores operate at one of 17 different frequencies (1.3GHz to 4.5GHz).
- E-cores operate at one of 6 different frequencies (1.0GHz to 2.5GHz).



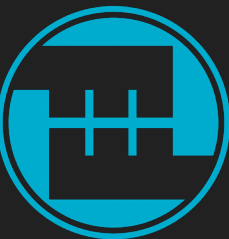
Hardware Notes

- I'm not sure how to reliably **force** the use of a specific type of core or clock frequency.
- A user can request a background QoS value using `taskpolicy -b`.
- In the long-run, performance on an otherwise idle system stabilized (due to thermal equilibrium being reached).



JEnt Architectural Notes

- There is a substantial performance difference between the P-cores and the E-cores.
- The P-core is very fast for single threads:
 - The maximum tolerable `osr` produced by `jitterentropy-osr` is unworkably large for P-cores.
- E-cores are significantly slower.
 - The maximum tolerable `osr` produced by `jitterentropy-osr` is more modest for E-cores.



JEnt Architectural Notes

A large maximum tolerable `osr` yields some surprising outcomes for the heuristic assessment procedure:

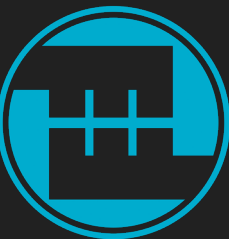
- Meeting the SP 800-90B Section 4.5 requirements isn't possible for `osr` settings larger than 186.
- The amount of data required to support such large `osr` values is substantial.
 - We gathered 600 million samples for the main testing configuration (P-core, large memory region).
- Given that we can't completely control if the process is on a P-core vs E-core, it is probably best to establish the maximum tolerable `osr` on the E-core (which is slower than the P-core, thus supports a smaller maximum tolerable `osr`).



Testing Configuration Notes

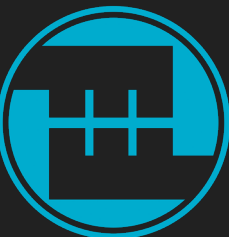
Most of the JEnt v3.6.3 configuration was left at the default settings, but:

- We did most of our data gathering with a 128MB JEnt memory region.
 - This prevents most memory updates from using only cache.
- We also did some testing using the default JEnt memory region size.
 - The default behavior results in a 128kB memory region.
 - Essentially all the memory updates will quickly be resolved in cache.



Testing Configuration Notes

- Test run characteristics:
 - The system was otherwise idle during testing.
 - CLI processes are foreground processes by default (thus have a QoS > 9).
 - The E-core testing flagged the test application as a background process (QoS ≤ 9).
- The timings for P-cores and E-cores are clearly distinct.
 - Roughly an order of magnitude difference.

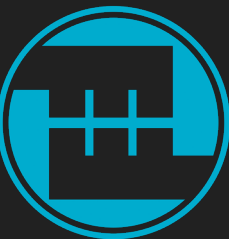


Large Memory Region Testing

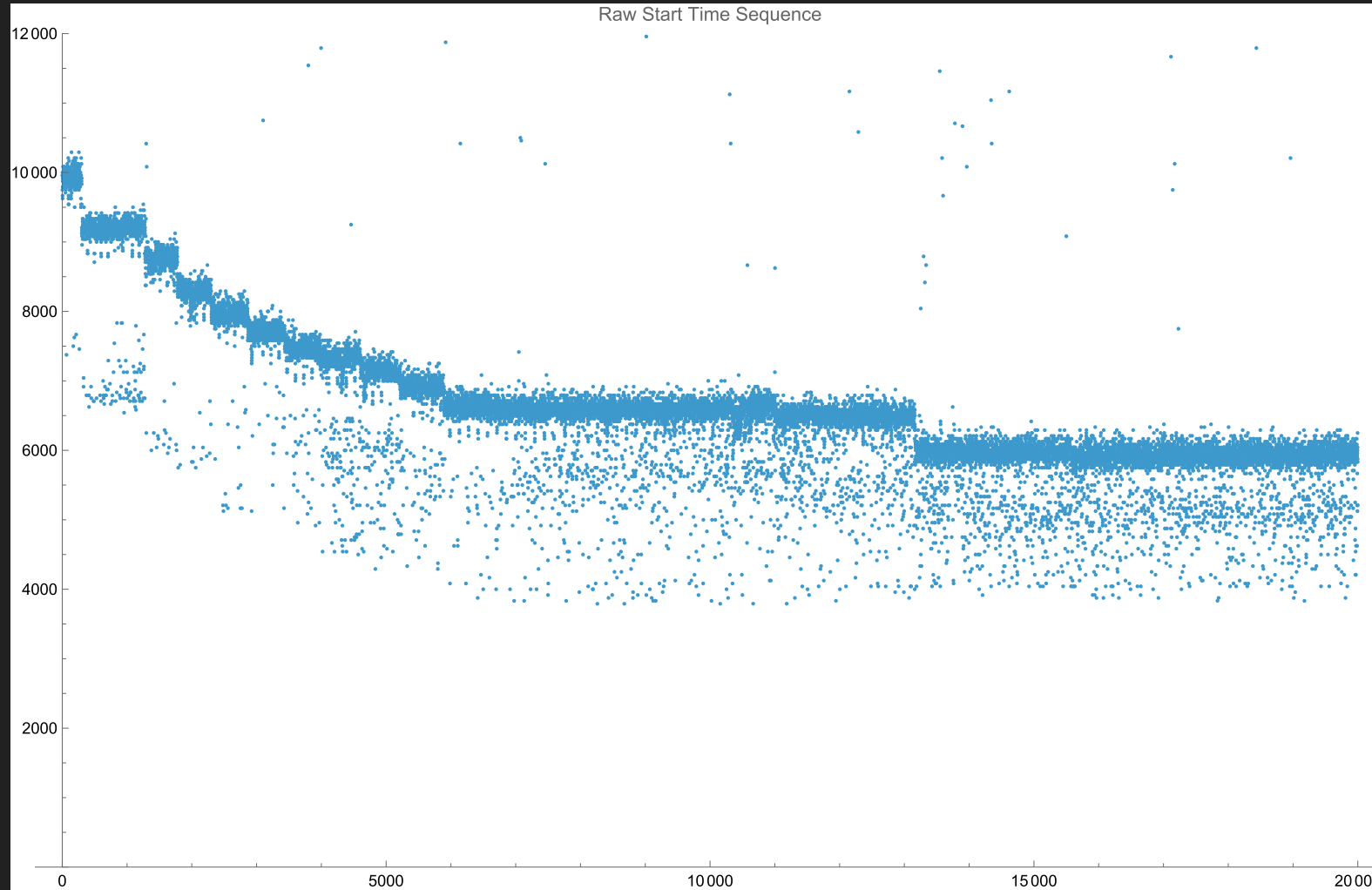


Data Summary: P-Core Large Memory Region

- 600 million samples.
- 1828 symbol values.
 - The full data range is [3708, 833917].
 - 95% of the data is in the range [4917, 6541].

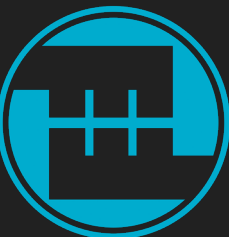


Data Summary: P-Core Large Memory Region

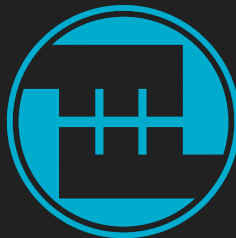
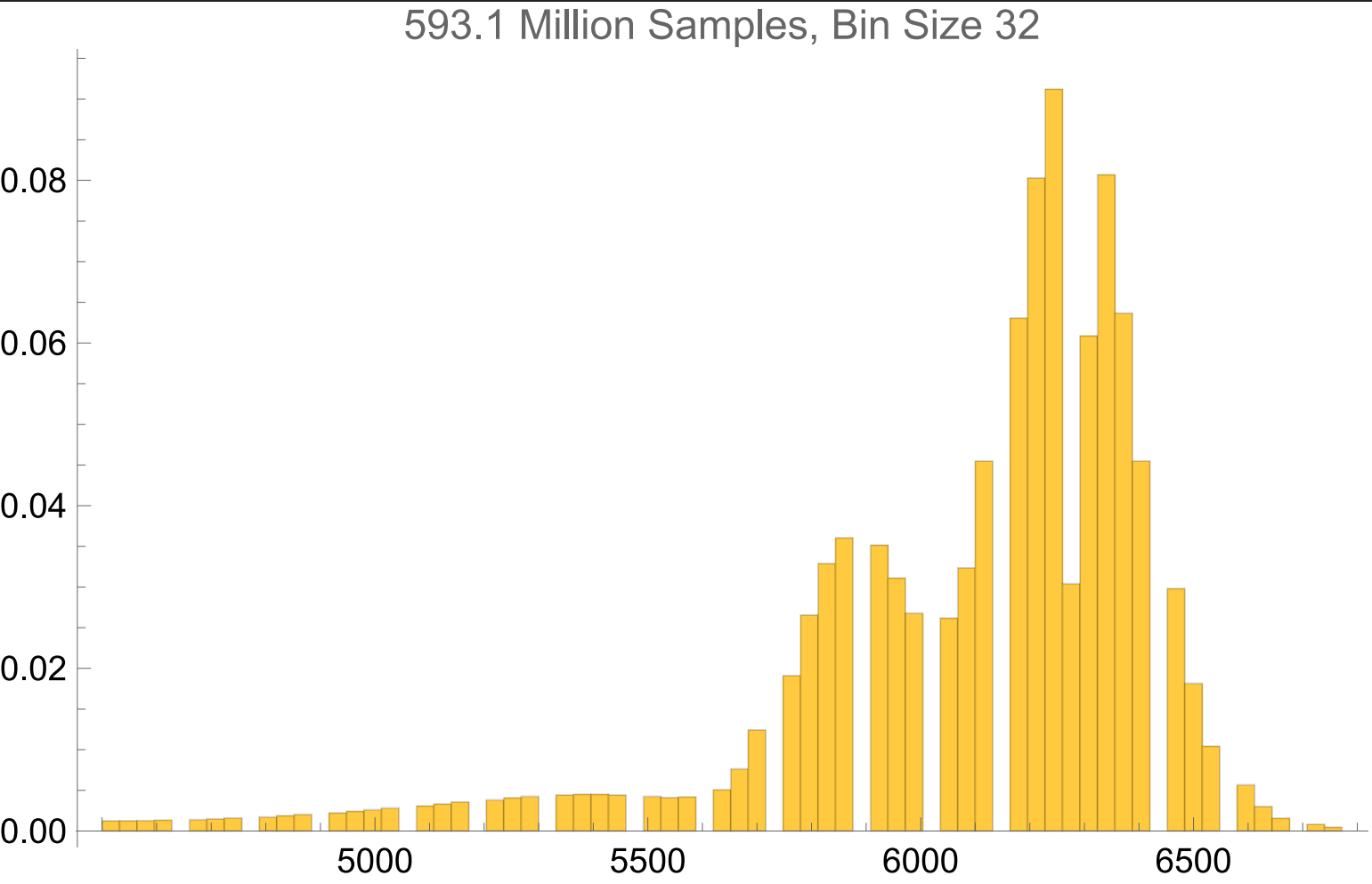


Data Summary: P-Core Large Memory Region

- This time series suggests standard microarchitectural speedups over time (e.g., improvements in code caching, speculative execution, and branch prediction).
- This time series also depicts the steady state.

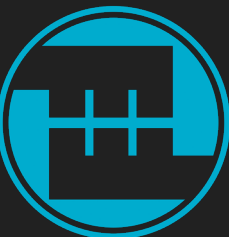


P-Core Large Memory Region Histogram



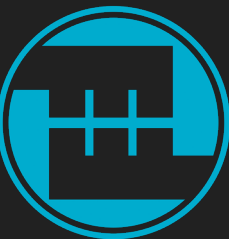
Results Summary: P-Core Large Memory Region

- If we allow 0.5s per output on P-cores, then the maximum tolerable osr is approximately 242.
- The data does not test as IID for any tested decimation level (up to 600).
 - We have not yet tried adjusting the translation to see how smaller translations might assess.
- The data assesses as having about 2.9 bits of min entropy per sample.

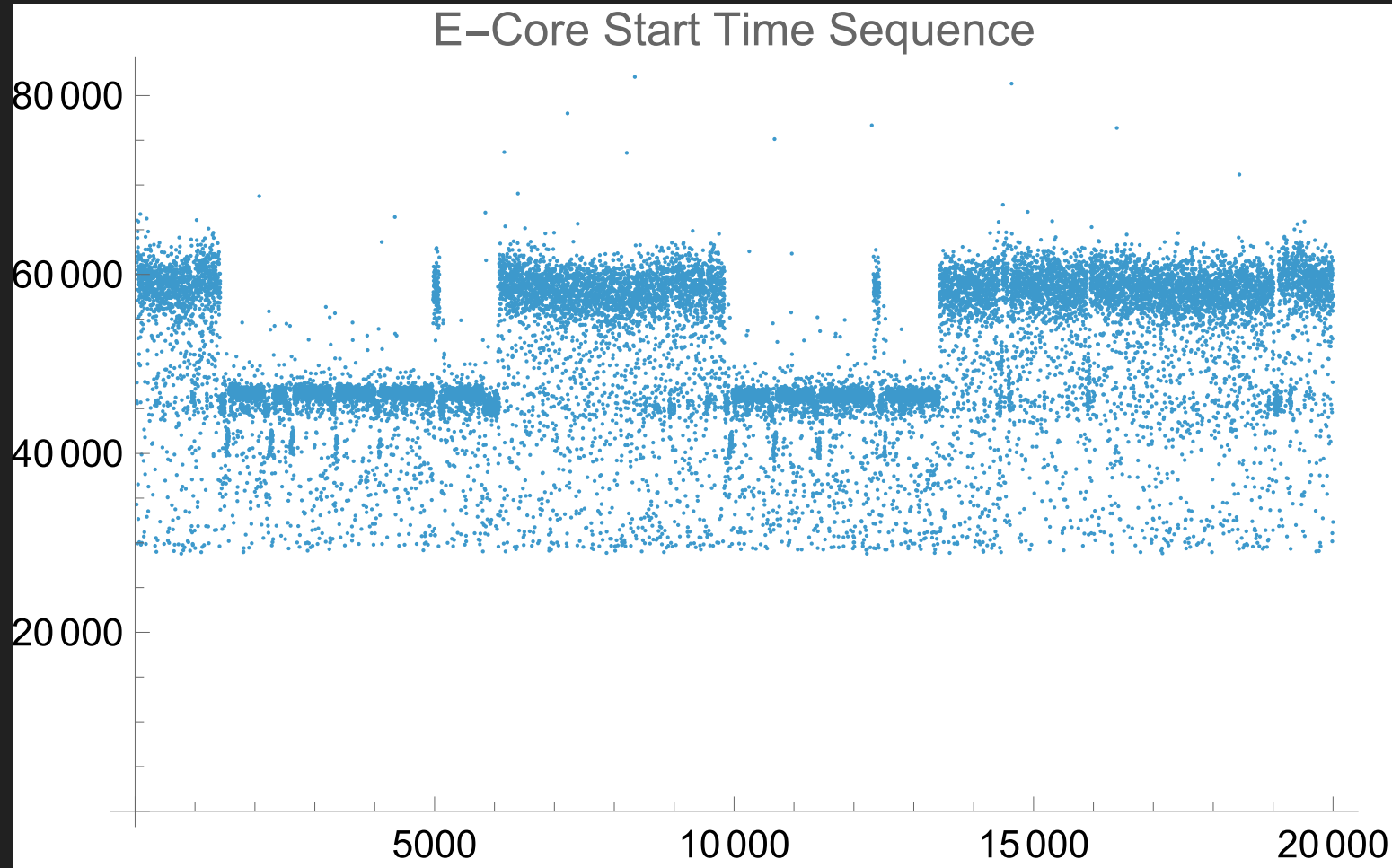


Data Summary: E-Core Large Memory Region

- 100 million samples.
- 15240 symbol values.
 - The full data range is [10708, 250405916].
 - 95% of the data is in the range [29250, 61709].

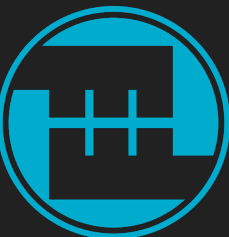


Data Summary: E-Core Large Memory Region

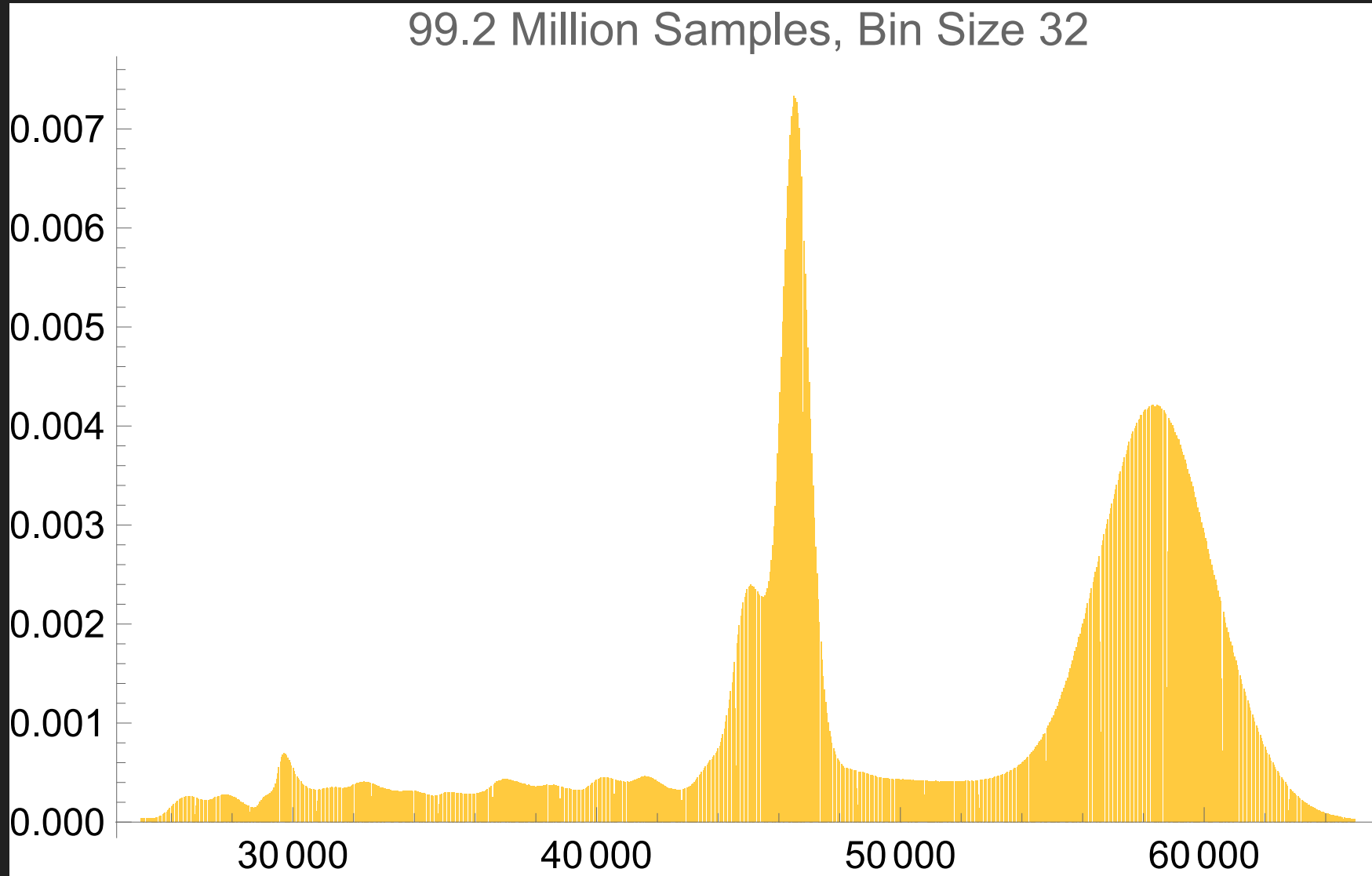


Data Summary: E-Core Large Memory Region

- There is no evident speedup as the code loops.
 - The steady-state behavior is essentially the same as the initial behavior.
- The time series suggests at least two different timing sub-distributions.

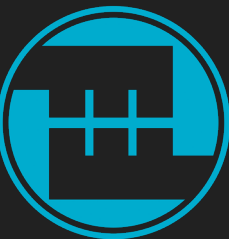


E-Core Large Memory Region Histogram



Results Summary: E-Core Large Memory Region

- If we allow 0.5s per output on E-cores, then the maximum tolerable osr is approximately 30.
- The data does not test as IID for any tested decimation level (up to 100).
- The full data set assesses as having about 5.3 bits of min entropy per sample.
- Sub-distribution A assesses at 4.9 bits of min entropy per sample.
- Sub-distribution B assesses at 6.2 bits of min entropy per sample.



Results Summary: Large Memory Region

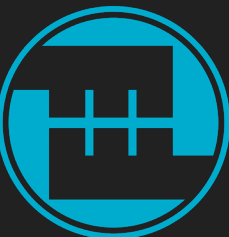
- JEnt raw data from the E-core seemed to have dramatically more symbols and assesses as having more entropy per symbol when compared to the JEnt raw data from the P-core.
- If we normalize to the same sample size, we get the following results:

Core Type	Symbol Count	95% Lower Bound	95% Upper Bound	Assessed Entropy
P-Core	1282	4708	6459	2.9
E-Core	15240	29250	61709	4.9



Results Summary: Large Memory Region

- Neither data set tested as essentially IID for any tested decimation level.
- We can't absolutely control the P-core vs E-core use, so we use the minimum of the maximum tolerable osr settings and $H_{\text{initial non-decimated}}$.
 - $\overline{\text{osr}} = 30$ (associated with the E-core).
 - $H_{\text{initial non-decimated}} = 2.9$ (associated with the P-core).
- The corresponding $H_{\text{submitter}} = \frac{1}{30}$.

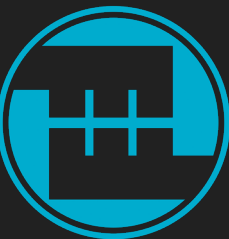


Default Memory Region Testing

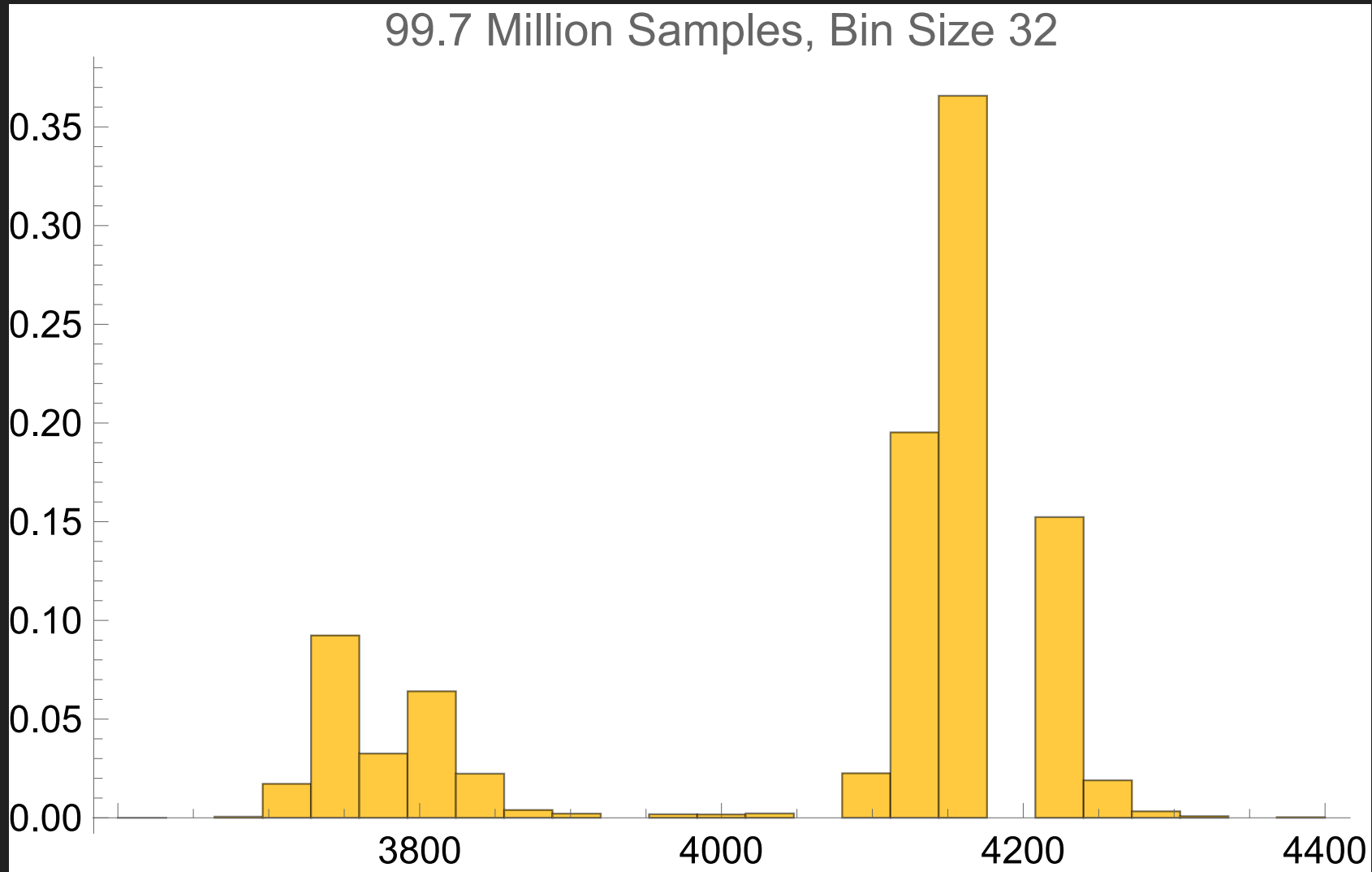


Data Summary: P-Core Default Memory Region

- 100 million samples.
- 922 symbol values.
 - The full data range is [3625, 47666].
 - 95% of the data is in the range [3750, 4250].

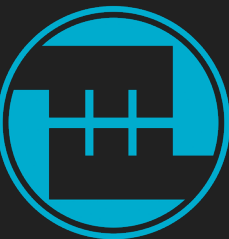


P-Core Default Memory Region Histogram



Results Summary: P-Core Default Memory Region

- If we allow 0.5s per output on P-cores, then the maximum tolerable osr is approximately 332.
- The data does not test as IID for any tested decimation level (up to 100).
- The data assesses as having about 1.0 bits of min entropy per sample.

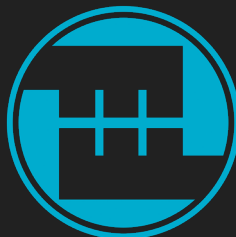
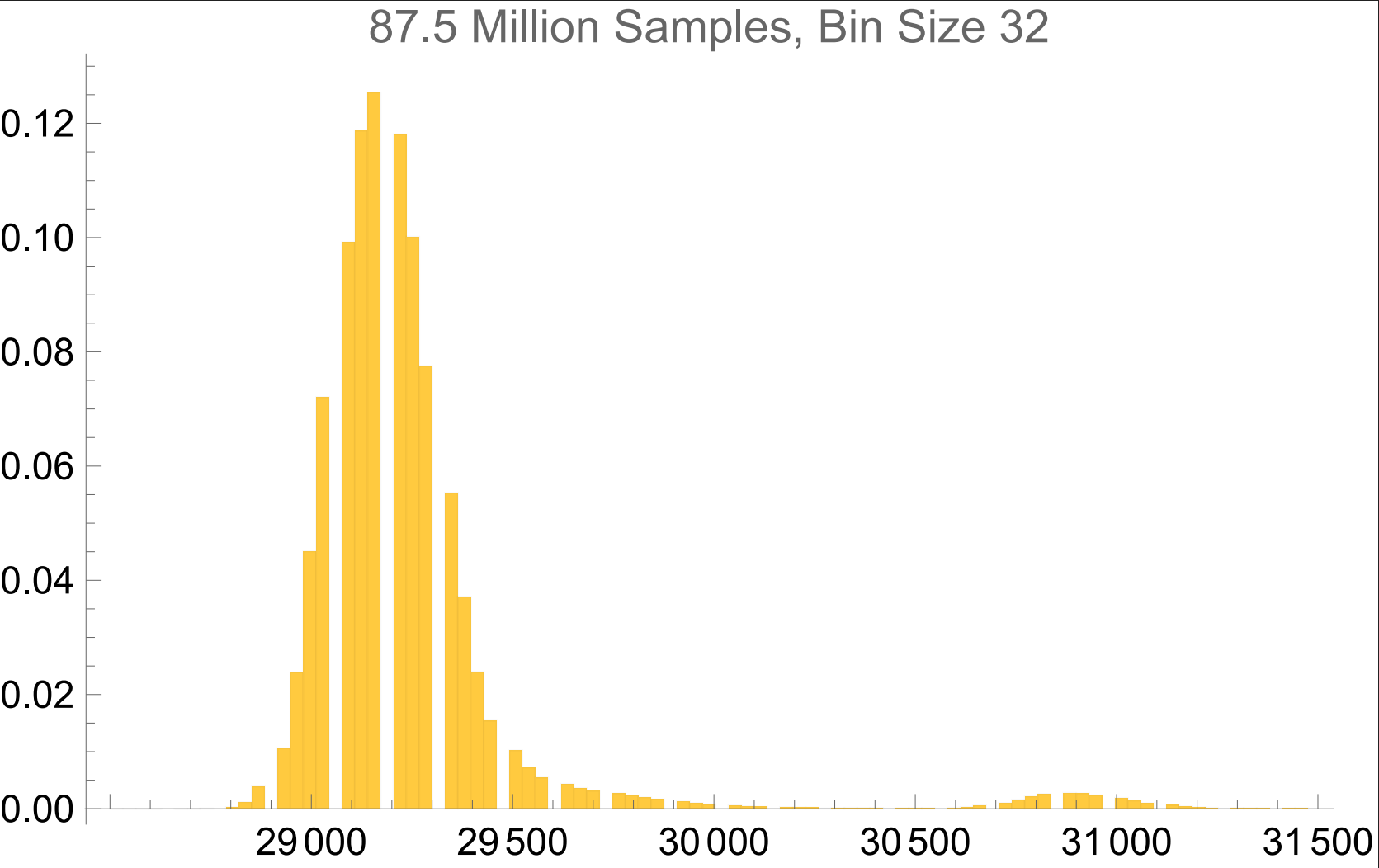


Data Summary: E-Core Default Memory Region

- 100 million samples.
- 15240 symbol values.
 - The full data range is [10667, 327691709].
 - 95% of the data is in the range [11167, 30834].

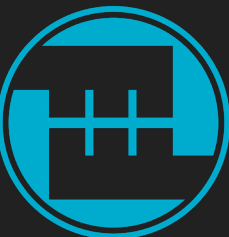


E-Core Default Memory Region Histogram



Results Summary: E-Core Default Memory Region

- If we allow 0.5s per output on E-cores, then the maximum tolerable osr is approximately 51.
- The data does not test as IID for any tested decimation level (up to 100).
- The full data set assesses as having about 2.6 bits of min entropy per sample.



Results Summary: Default Memory Region

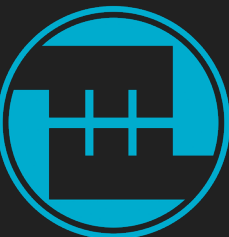
- JEnt raw data from the E-core seemed to have dramatically more symbols and assesses as having more entropy per symbol when compared to the JEnt raw data from the P-core.

Core Type	Symbol Count	95% Lower Bound	95% Upper Bound	Assessed Entropy
P-Core	922	3750	4250	1.0
E-Core	12465	11167	30834	2.6



Results Summary: Default Memory Region

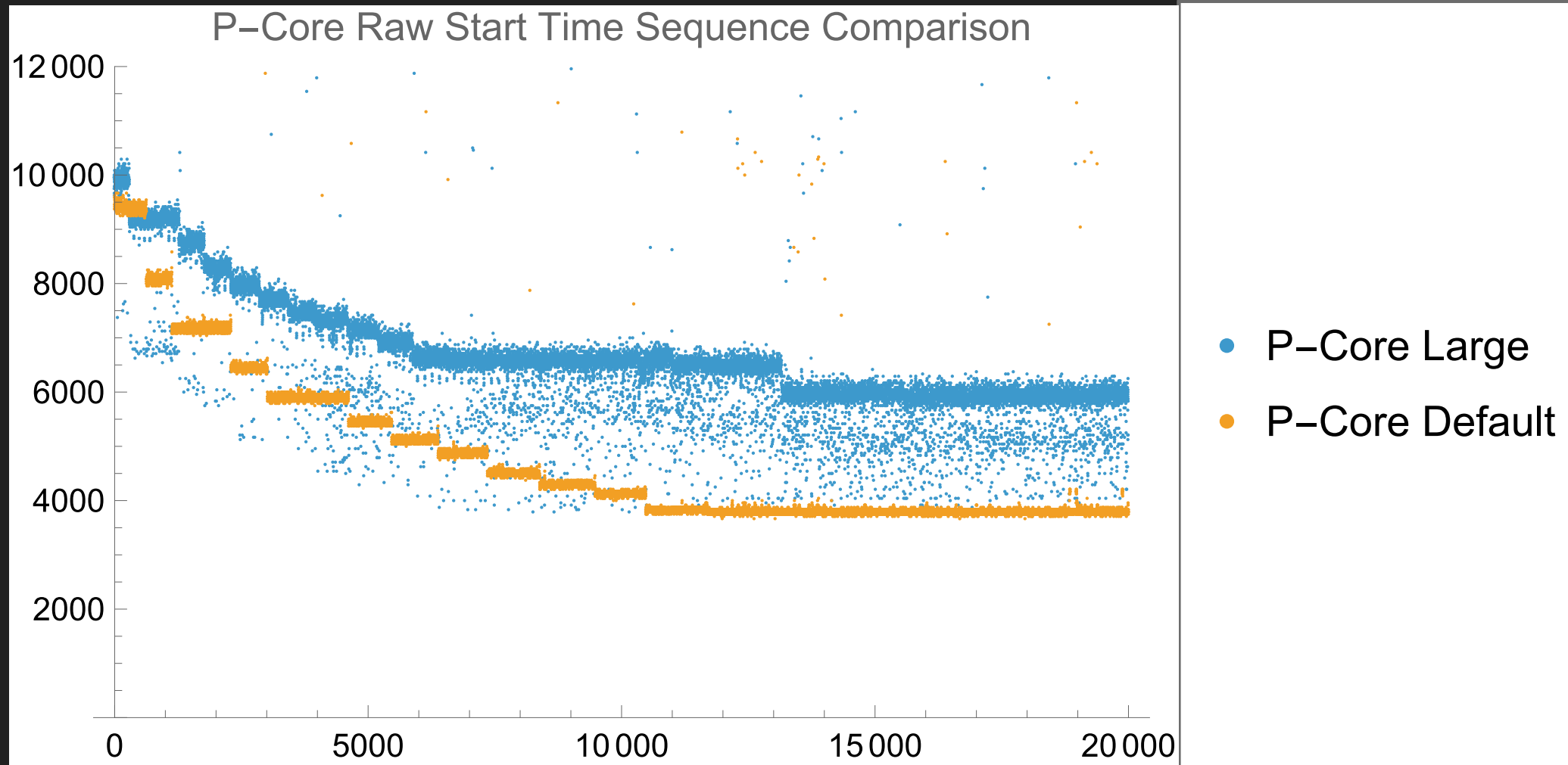
- Neither data set tested as essentially IID for any tested decimation level.
- We can't absolutely control the P-core vs E-core use, so we use the minimum of the maximum tolerable osr settings and $H_{\text{initial non-decimated}}$.
 - $\overline{\text{osr}} = 51$ (associated with the E-core).
 - $H_{\text{initial non-decimated}} = 1.0$ (associated with the P-core)
- The corresponding $H_{\text{submitter}} = \frac{1}{51}$.



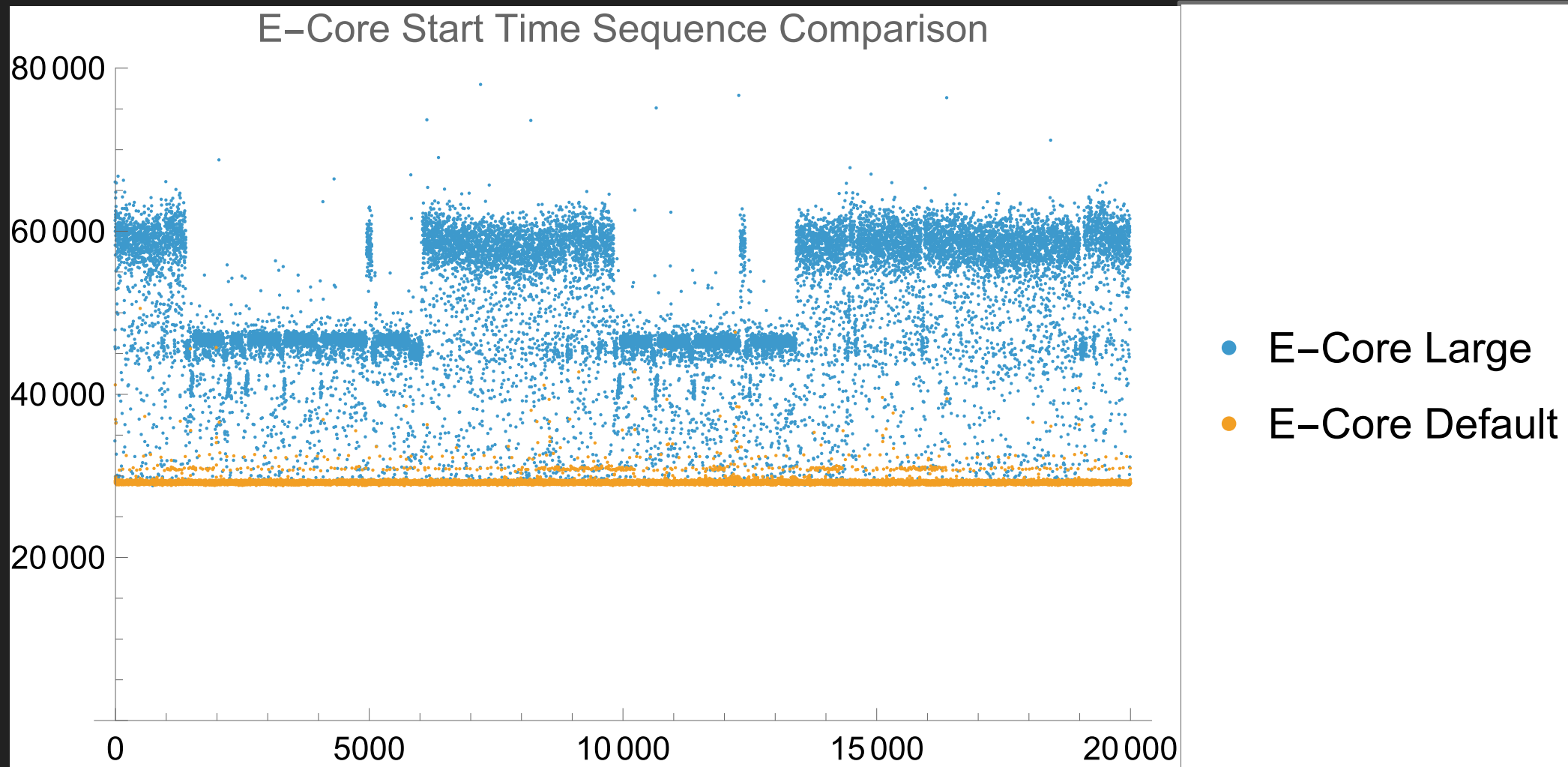
Testing Retrospective



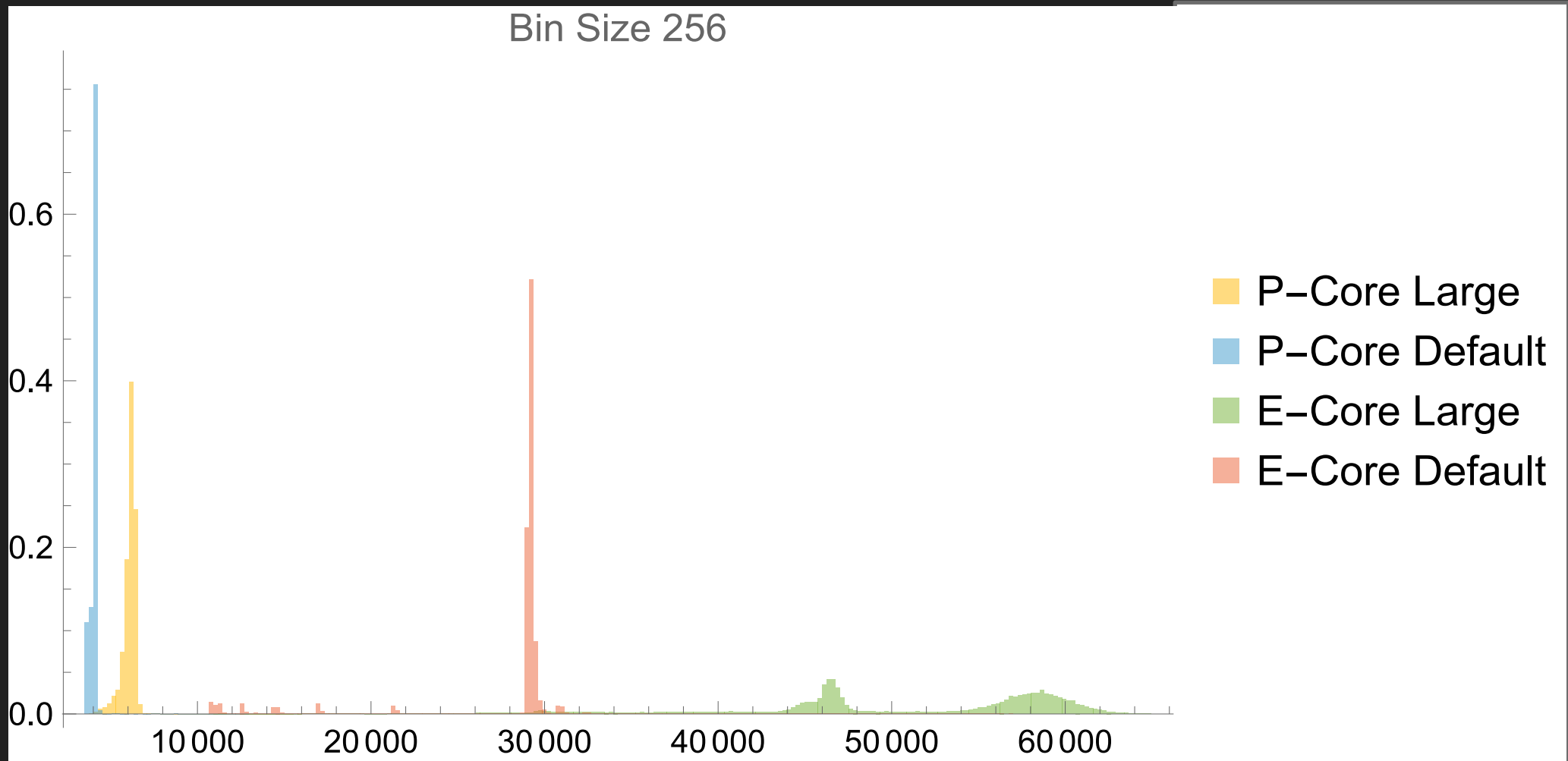
Summary Observations: Initial Time Series (P-Core)



Summary Observations: Initial Time Series (E-Core)



Summary Observations: Histogram Comparison



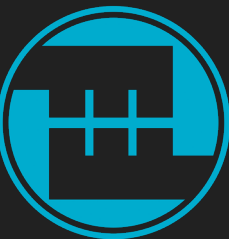
Summary Observations

Memory Region	Core Type	Max osr (0.5s)	Symbol Count	95% Lower Bound	95% Upper Bound	Ess. IID?	Assessed Entropy
Large	P-Core	242	1282	4708	6459	No	2.9
Default	P-Core	332	922	3750	4250	No	1.0
Large	E-Core	30	15240	29250	61709	No	4.9
Default	E-Core	51	12465	11167	30834	No	2.6



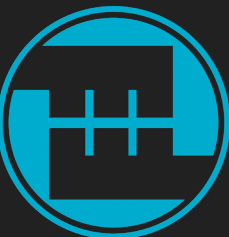
Summary Observations

- The level of variation of JEnt raw data (and probably available entropy) is different between P-cores and E-cores.
 - The P-cores were faster, and the timing was more consistent.
 - The E-cores were slower, and the timing displayed more variation.
 - In most configurations, it isn't possible to characterize this architecture without assessing JEnt on both P-cores and E-cores.
- Size of the JEnt memory region has a major impact on the performance
 - We encountered more variation (and probably available entropy) with larger memory regions.



Conclusion

- On heterogeneous architectures, JEnt behavior may vary significantly by core.
 - There can be substantial entropy rate differences between the cores.
- There are options:
 - Use OS interfaces to force the use of one specific type of core (if such APIs are available and are effective).
 - If not available, evaluate the performance on all the types of cores and take the worst case.



References

- <https://eclecticlight.co/2024/11/18/inside-m4-chips-e-and-p-cores/>
- <https://eclecticlight.co/2024/12/05/inside-m4-chips-cpu-core-management/>
- <https://eclecticlight.co/2024/12/02/inside-m4-chips-controlling-frequency/>
- <https://eclecticlight.co/2024/11/11/inside-m4-chips-p-cores/>

