What is Performance Portability..?

"***Same code*** will ***run productively*** on a variety of architectures"[1]

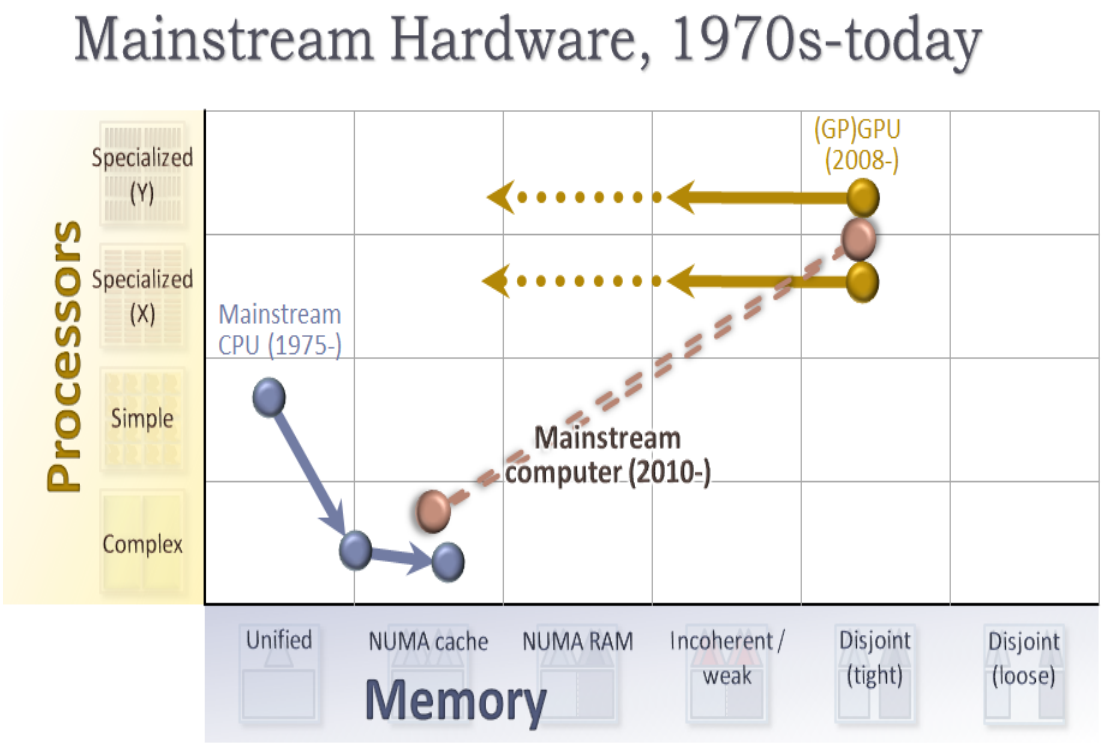
"ability to achieve a similar high ***fraction of peak performance*** across target architectures" [4]

"maximizing ***the amount of user code*** that can run on diverse devices and obtain (nearly the) ***same performance*** as a variant of the code that is ***written specifically for that device"*** [3]

"application's ***performance efficiency*** for a given platform that can be executed correctly on all platforms in given set" [5]

"achieving ***good performance*** on all the platforms ***without code modification***" [6]

What is performance portability and why should we care?



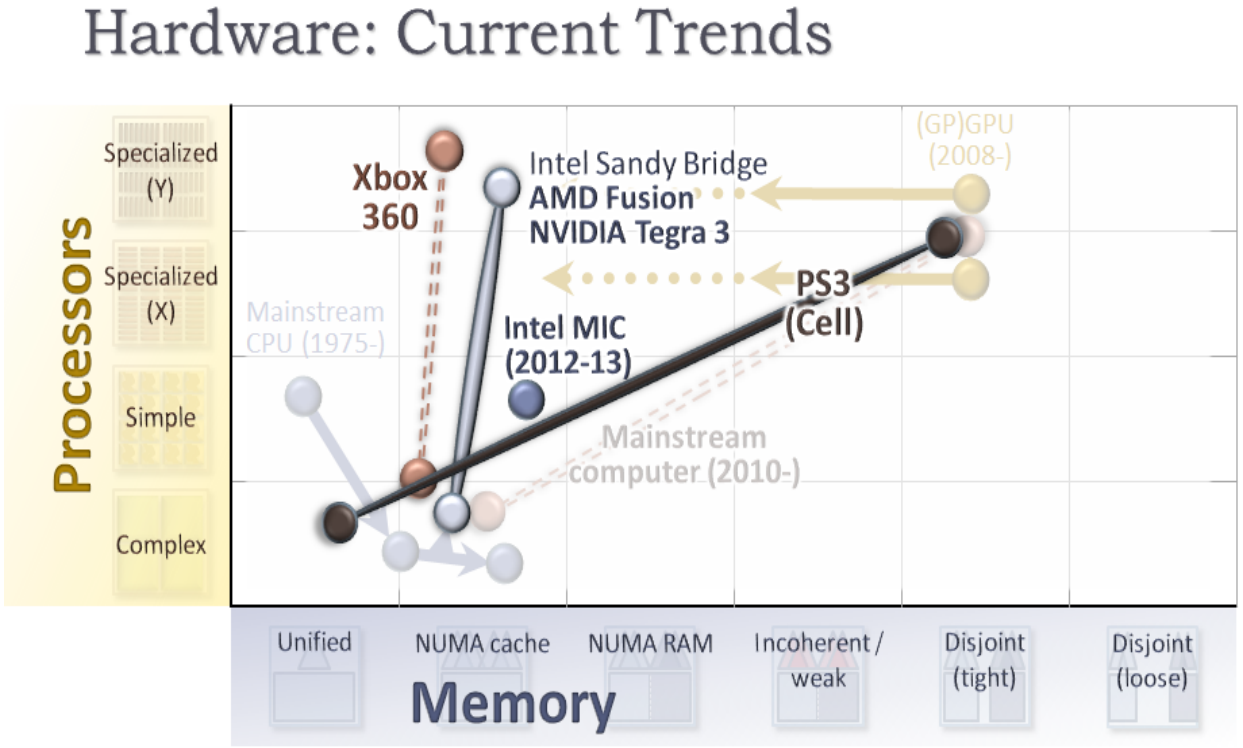
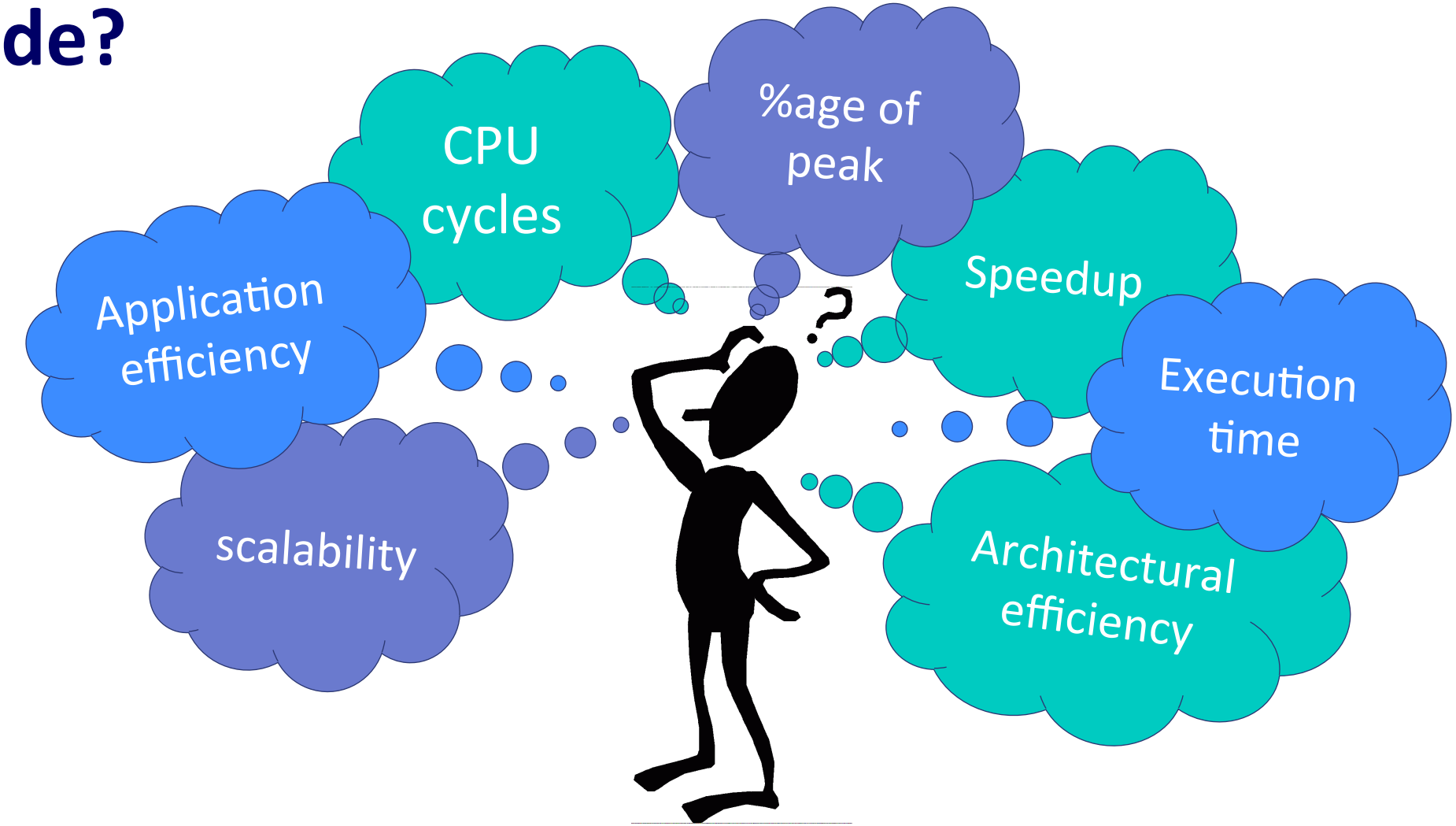


Figure source: Herb Sutter, Welcome to the Jungle [2]

How to measure performance portability of a code?



Questions Questions..

Given a code:

- How to measure/predict its performance portability
  - ✓ across similar architectures (e.g. set of CPUs)
  - ✓ across dissimilar architectures (e.g. set of CPUs and GPUs)

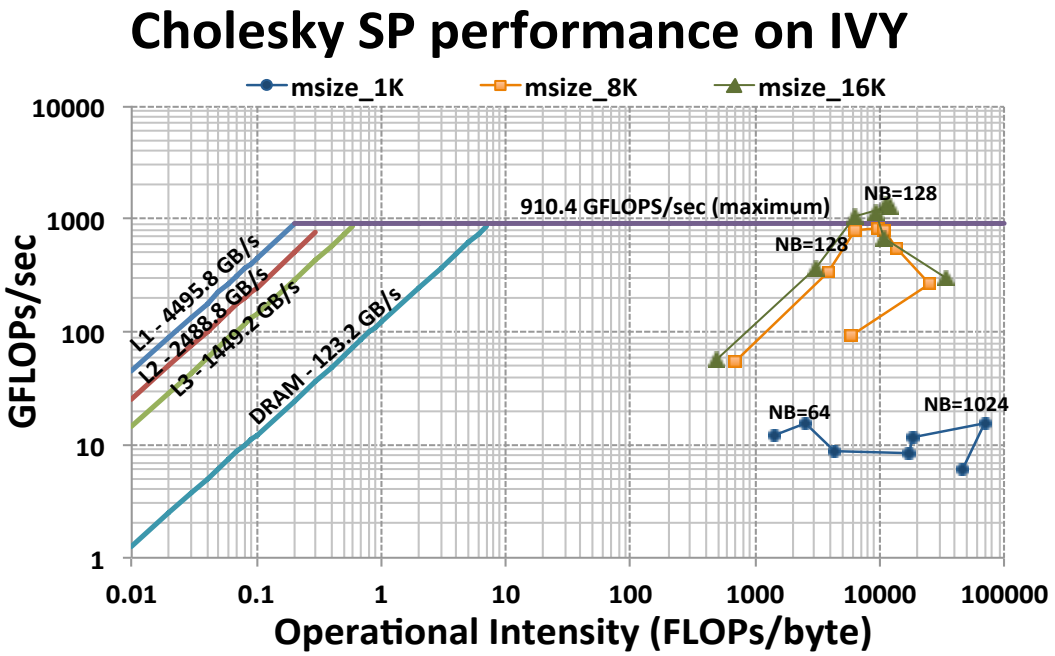
- What makes a code MORE or LESS performance portable than others?

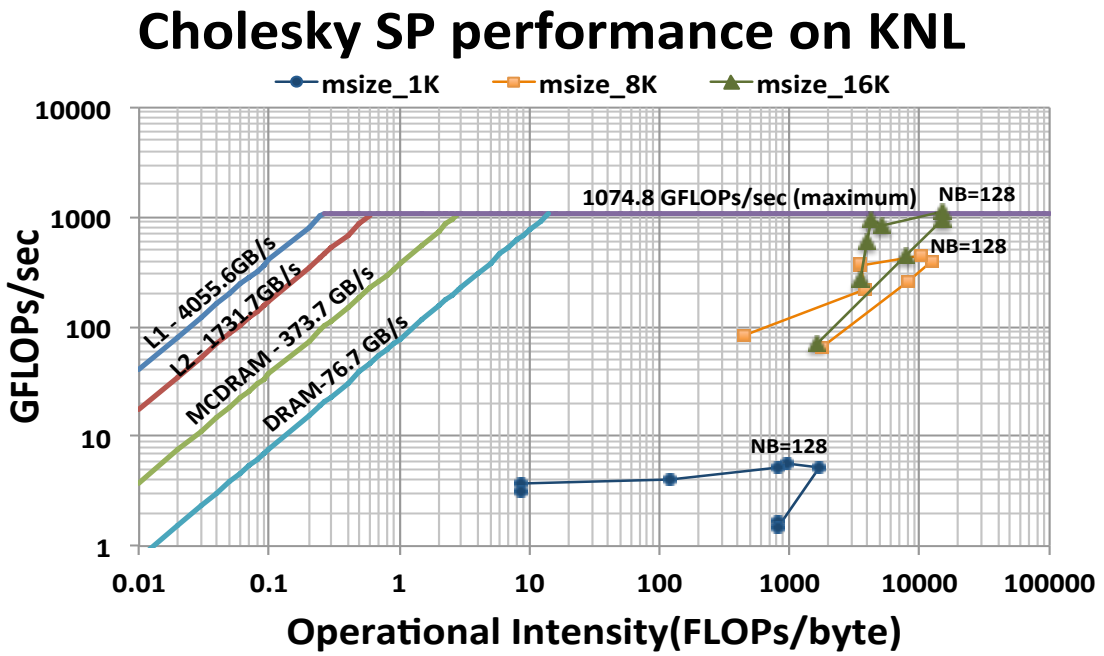
  Code structures, data layout, looping order, i/p dataset type or size…

- Optimal performance on one platform vs sub-optimal yet similar performance on multiple platforms?
- Single source code vs different versions of source code?

Exploring performance portability through Roofline Model

- Cholesky (linear algebra) performance by changing block size





Ivy Bridge (IVY)	Intel Xeon CPU E7-4860 v2 @ 2.60GHz, 96 core (12 x 2 x4), 32 kB L1, 256 KB L2, 30 MB L3
Knights Landing (KNL)	Intel Xeon Phi(TM) CPU 7210 @ 1.30GHz, 256 cores (64x4x1), 32kB L1, 1024 kB L2

Testing existing definitions and metrics

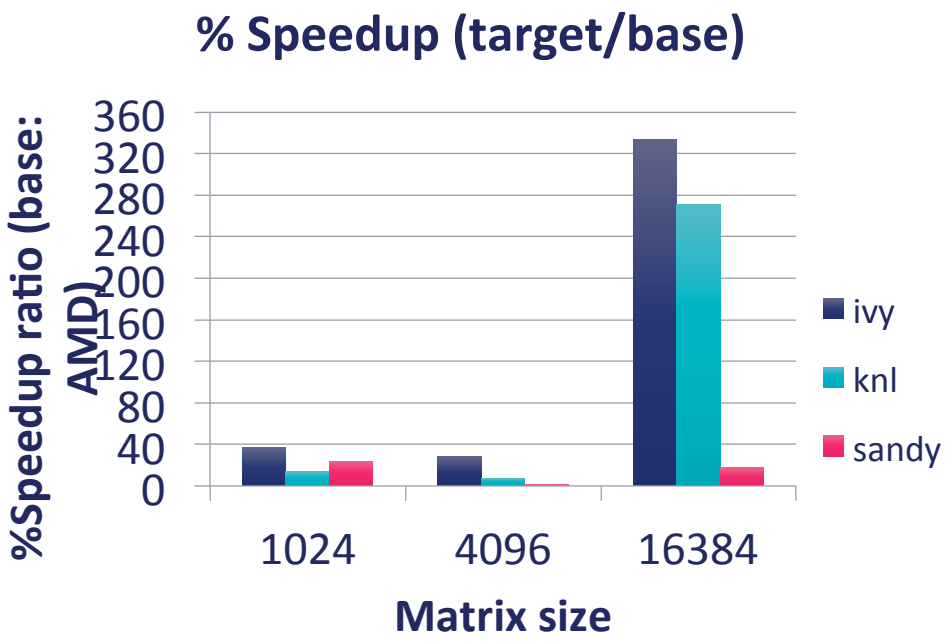
- Similar percentage of peak (**architectural efficiency**) across different architectures [4]
  - number of threads: max /optimal?
  - Architectures with higher peak, more Gflops performance but smaller %age of peak?

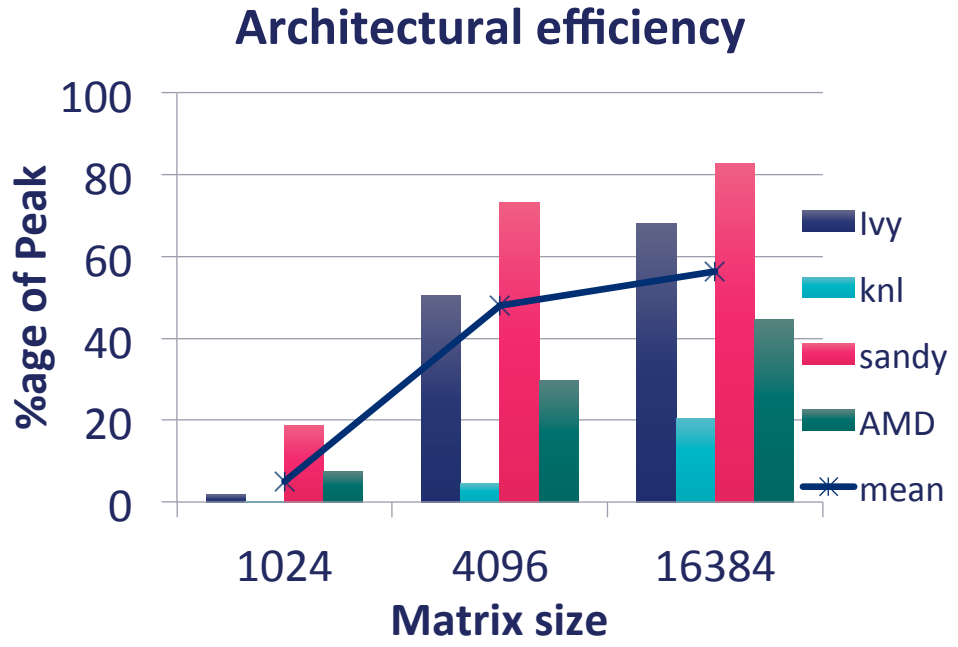
- Performance portability (from base to target) on **n** nodes the **percentage of speedup** of an application on a target system, $\mathbf{S_{tn}}$, w.r.t to the speedup on the base system, $\mathbf{S_{bn}}$ [6]
  - Number of threads: Same/Max/Optimal?
  - Scalability reflective of absolute performance?

- **Performance efficiency** (mean of architectural efficiency/application efficiency) across a set of architectures [5]

Testing existing definitions and metrics

Cholesky Factorization (linear algebra) code from PLASMA library





SandyBridge (sandy)	Intel Xeon E5-2650 , 16 cores (16x1x1), L1: 32kB, L2: 256KB, L3: 20MB
AMD	Opteron 6272, 32 cores (8x2x2), L1: 16kB, L2: 2048kB, L3, 6MB

References

[1] Jeff Larkin, ***Performance Portability through Descriptive Parallelism*** , DoE Meeting on Performance portability, April 2016

[2] Herb Sutter, **Welcome to the Jungle**, 2012

[3] H. Carter Edwards, Christian R. Trott, and Daniel Sunderland. ***Kokkos: Enabling manycoreperformance portability through polymorphic memory access patterns***. Journal of Parallel and Distributed Computing, 74(12), 2014.

[4] S. Mcintosh-Smith, M. Boulton, D. Curran, and J. Price. ***On the performance portability of structured grid codes on many-core computer architectures***. ISC'14, pages 53–75, 2014.

[5] SJ Pennycook, JD Sewall, and VW Lee. **A metric for performance portability**. arXiv preprint arXiv:1611.07409, 2016.