CGT 581I - Parallel Graphics and Simulation

Knights Landing Programming Overview

Bedrich Benes, Ph.D.
Professor
Department of Computer Graphics
Purdue University

KNL

- KNL is a many-core processor
- It is a high-performance processor
- Can be optimized by standard techniques …but also some special

KNL - Levels of Parallelism

1. Manage domain parallelism (via MPI)
2. Increase Thread Parallelism (OpenMP/Thread Building Blocks)
3. Exploit Data Parallelism (Vectorization)
4. Improve Data Locality (Data Layout and still capture)

Levels of Parallelism

Category	Characteristics	Methods
Domain parallelism	Heavy computation in the domain, limited communication of data between domain	MPI (Chapter 15) TBB flow graph (Chapter 8) PGAS (Chapter 16) Frameworks (e.g., Hadoop, spark) - beyond the scope of this book
Thread parallelism	Data sharing between threads, high degree of (mostly) independent tasks. Potential for excessive synchronization going unnoticed	OpenMP (Chapter 8) TBB tasking (Chapter 8) PGAS (Chapter 16)
Data parallelism	Same algorithm/operation applied to multiple data items	Libraries (Chapters 13, 17) Auto vectorization (Chapter 9) OpenMP "simd" (Chapter 9) Code transformation (analysis tool to help: Chapter 10) AVX-512 intrinsics (Chapter 12)
Data locality	Arranging algorithms to minimize data movement Arranging data to reduce the need for data movement Arranging data for temporal reuse, to better utilize caches so as to reduce data movement all the way from memory	AOS to SOA, and related transforms; SDLT (Chapter 11) Code transformations (analysis tools to help: Chapter 14)

© Jeffers et al. *Intel Xeon Phi Processor High Performance Programming Knights Landing Edition*

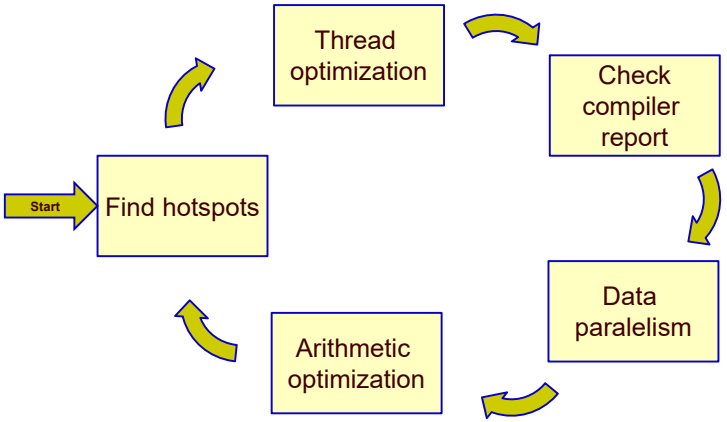
Code Modernization

- Term used for activities that update an existing code to be more efficient on modern architectures
- How much do I need to modify?

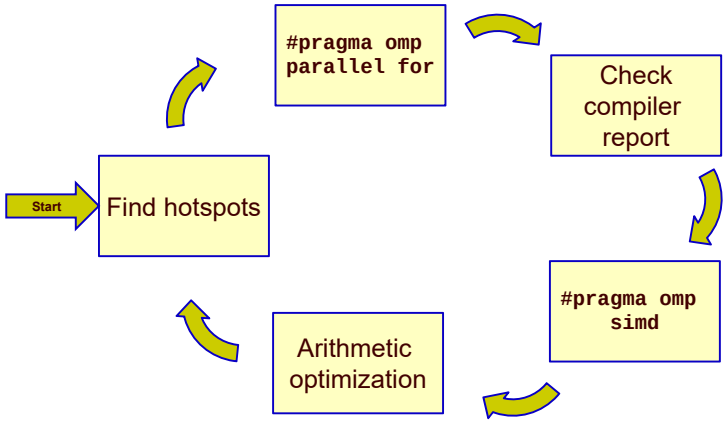
Code Modernization

- **Code refactoring:** restructuring existing computer code without changing its external behavior.
- Improves nonfunctional attributes. (code readability, reduces complexity)

Evolutionary Optimization



Evolutionary Optimization (OpenMP)



Roofline

- Roofline is *an estimate of the highest possible performance*
- Is the application communication or calculation limited?
- Is the application calculation or memory limited? (CPU-bound, memory-bound)

Roofline

- Fine-grained granularity:
  - a large number of small tasks
  - amount of work associated is low
  - good for SM
  - good load balancing
- Coarse-grained granularity:
  - small number of large tasks
  - good for MPI
  - potentially bad load balancing

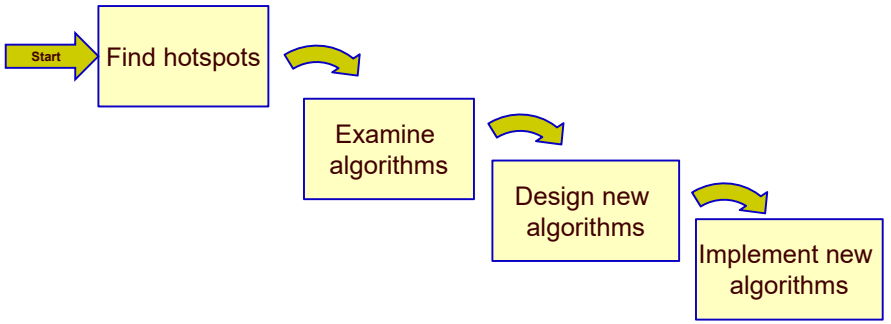
Roofline

- f_n the FLOPs needed to run the app
- f_a the FLOPs available on the system
- Roofline:

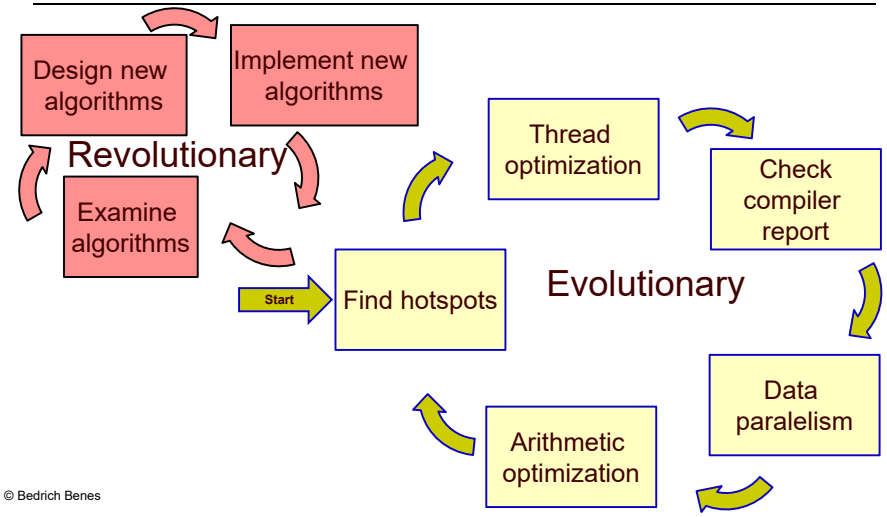
$$s = \frac{f_n}{f_a} \; [seconds]$$

idealized "roofline"

Revolutionary Optimization



Evolutionary + Revolutionary



Reading

- Intel Xeon Phi Processor High Performance Programming Knights Landing Edition
- James Jeffers, James Reinders, and Avinash Sodani
- ISBN: 9780128091944
- Morgan Kaufmann
- Chapter 7