Parallel Computing: The Good, the Bad and the Ugly

ICCS, Berkeley, Jan 28, 2011

Pradeep K. Dubey

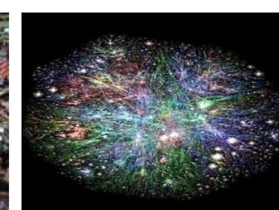
IEEE Fellow and Director of Parallel Computing Lab
Intel Corporation

Who We Are: Parallel Computing Lab

- Parallel Computing -- Research to Realization
  - Worldwide leadership in throughput/parallel computing, industry role-model for application-driven architecture research, ensuring Intel leadership for this application segment
  - Dual Charter:
    - Application-driven architecture research and multicore/manycore product-intercept opportunities
- Architectural focus:
  - “Feeding the beast’ (memory) challenge, domain-specific support, massively threaded machines, unstructured accesses, distributed decomposition
- Workload focus:
  - Multimodal real-time physical simulation, Behavioral simulation, Interventional medical imaging, Large-scale optimization (FSI), Massive data computing, non-numeric computing
- Industry and academic co-travelers
  - Mayo, HPI, CERN, Stanford (Prof. Fedkiw), UNC (Prof. Manocha), Columbia (Prof. Broadie)
- Recent accomplishments:
    - First TFlop SGEMM and highest performing SparseMVM on KNF silicon demo’ed at SC’09
    - Fastest LU/Linpack demo on KNF at ISC’10
    - Fastest search, sort, and relational join – Best Paper Award for Tree Search at SIGMOD 2010





Who Needs Compute

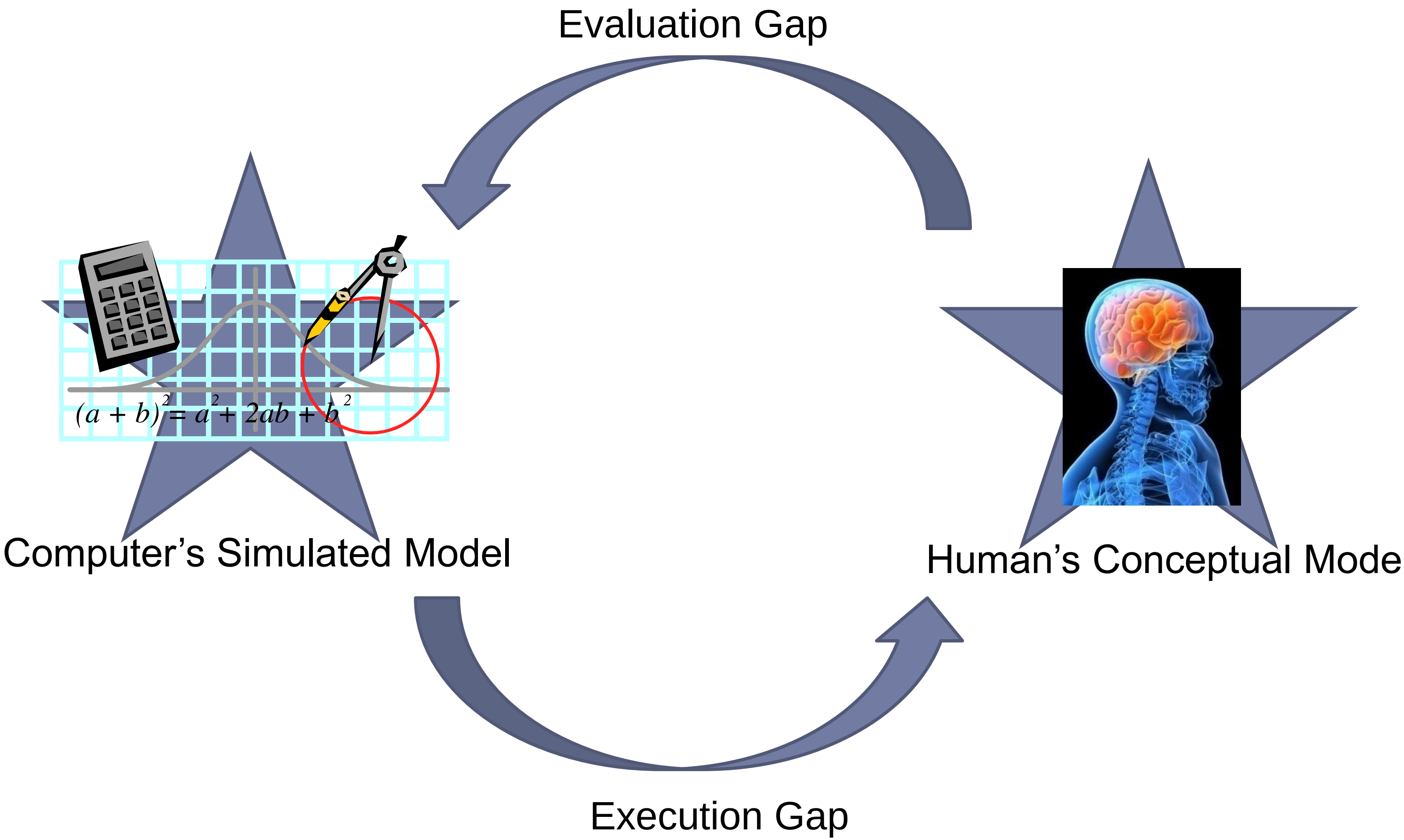
Traditional drivers of compute

- *Norman's Gulf*: Quest for natural human-machine interface
- Entertainment: Unending fascination with virtual and unreal
- The *data deluge*: The problem of drinking out of fire hydrant
- Real-time analytics: *Decision delayed is objective denied*
- *Curious minds want to know* (HPC): Science moves on!

Recent catalysts of compute

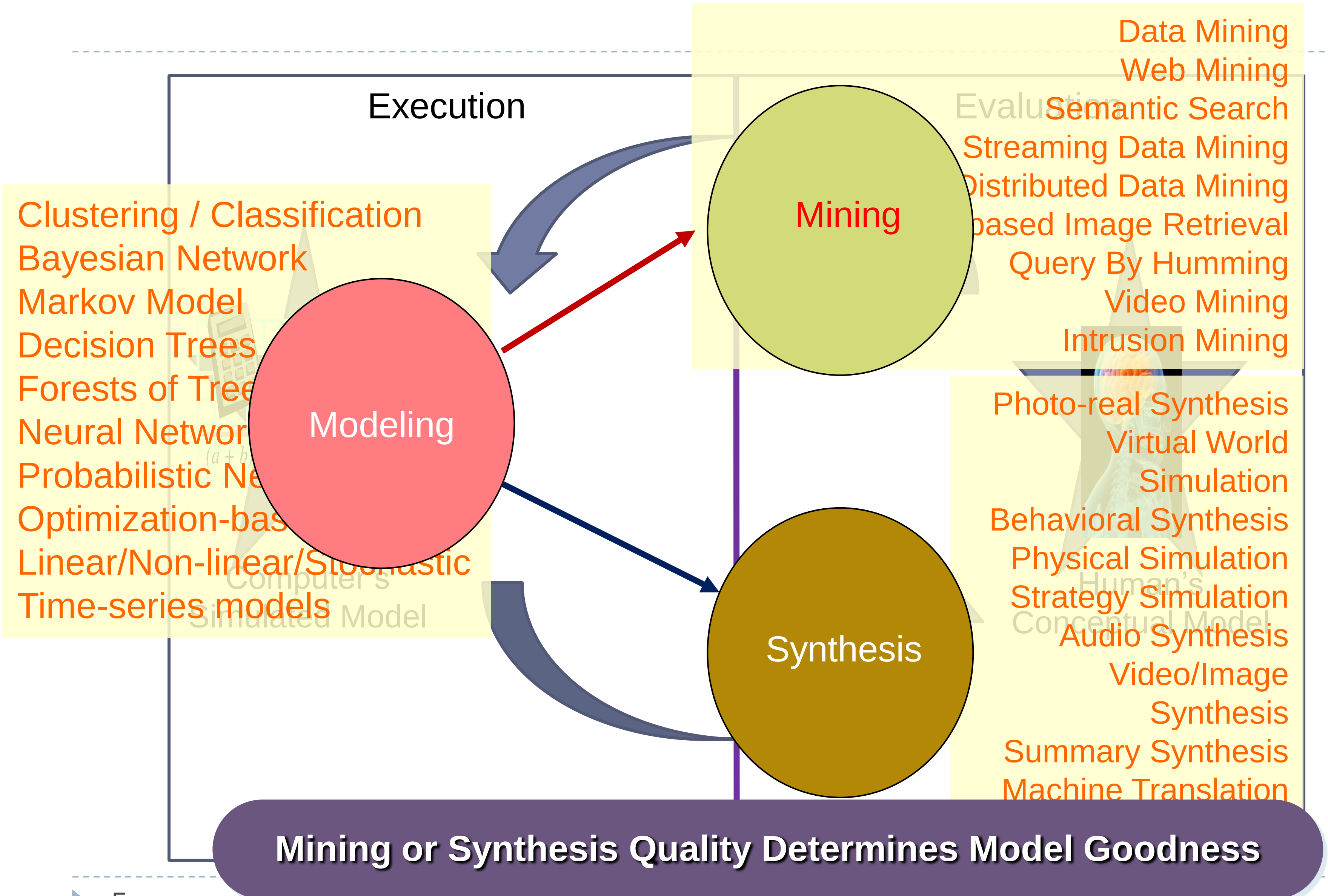
- Changing demographics of computer users
- Massive compute meets massive data
- Connected computing

Norman’s Gulf



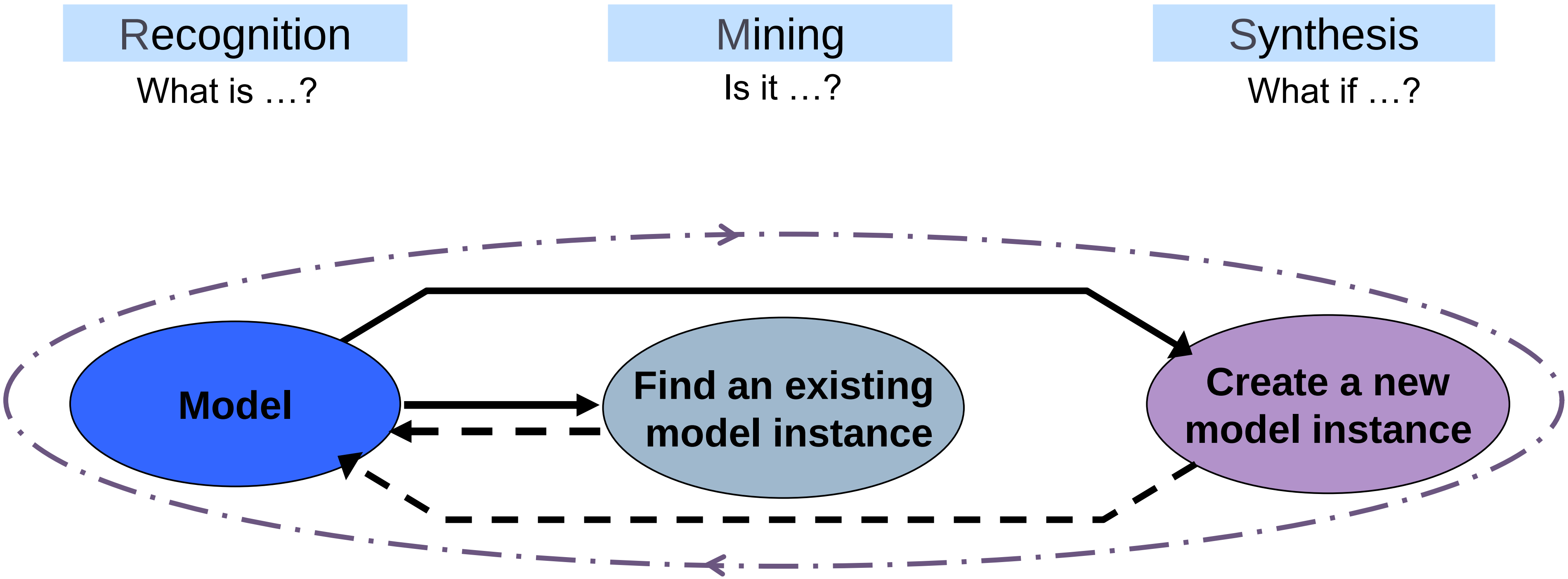
Pradeep.Dubey@intel.com

Decomposing Compute-Intensive Apps



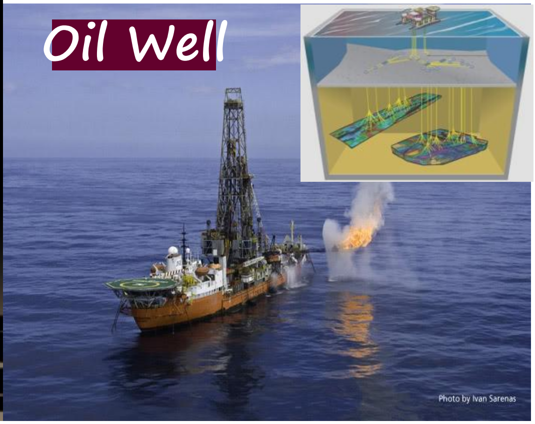
Pradeep.Dubey@intel.com Jan 2011

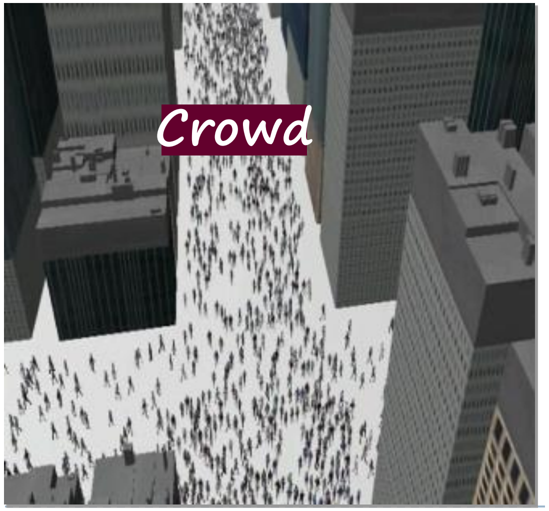
Interactive RMS Loop



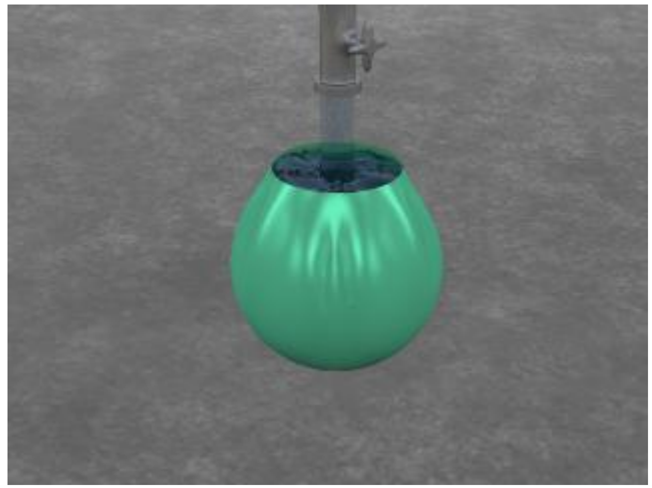
Most RMS apps are about enabling interactive (real-time) RMS Loop (iRMS)

Illustrative Parallel Computing Apps





Insatiable Appetite for Compute …

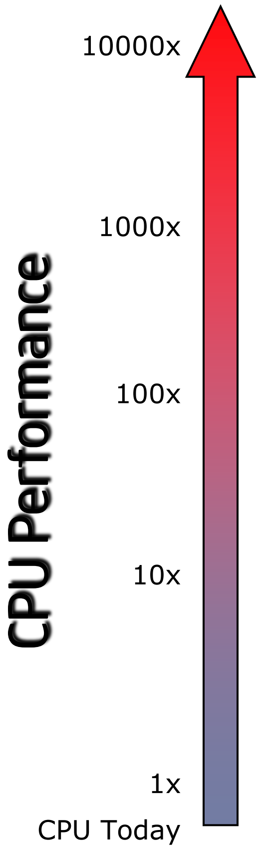


Prof. Ron Fedkiw, Computer Science/Stanford and Jon Su, PCL/Intel Labs

(Deformable and thin) Solid-Fluid Coupling

10s simulation takes 4 days on a Tflop compute node!

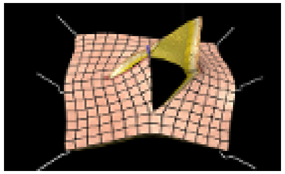
More the better …



- Haptic dynamics in haptic training apps
  - System fully usable in the operating room
- Real-time implicit simulation for 100K elements
  - For real-time training tools & very accurate prediction
- Interactive quasi-statics simulations of 100K elements
  - Good usability for planning and prediction
- Offline dynamic Simulations of 100K elements
  - Limited usability for prediction
- Offline quasi-statics simulations of 10K elements
  - Impractical for use in clinical environments

100K - 1M elements

10 - 100K elements

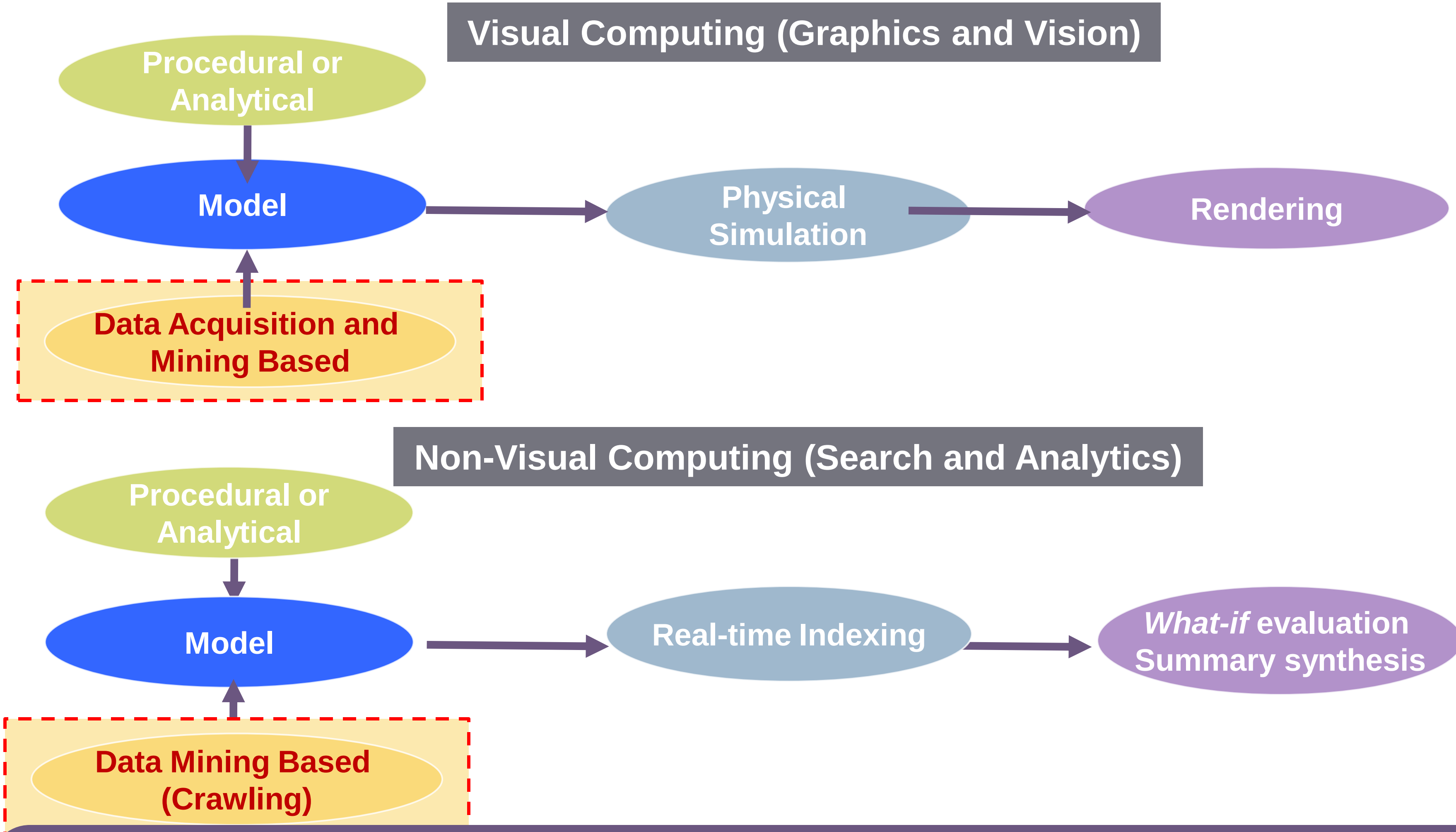


1 - 10K elements

Force simulations for visual rendering: 10s of Hz
Force simulations for haptic rendering: KHz or more

Entertainment to Interventional Medical Imaging: Physics plays a critical role and drives compute!

Parallel Computing: Visual or Analytics



Growing Importance of Data Driven Models

Pradeep.Dubey@intel.com

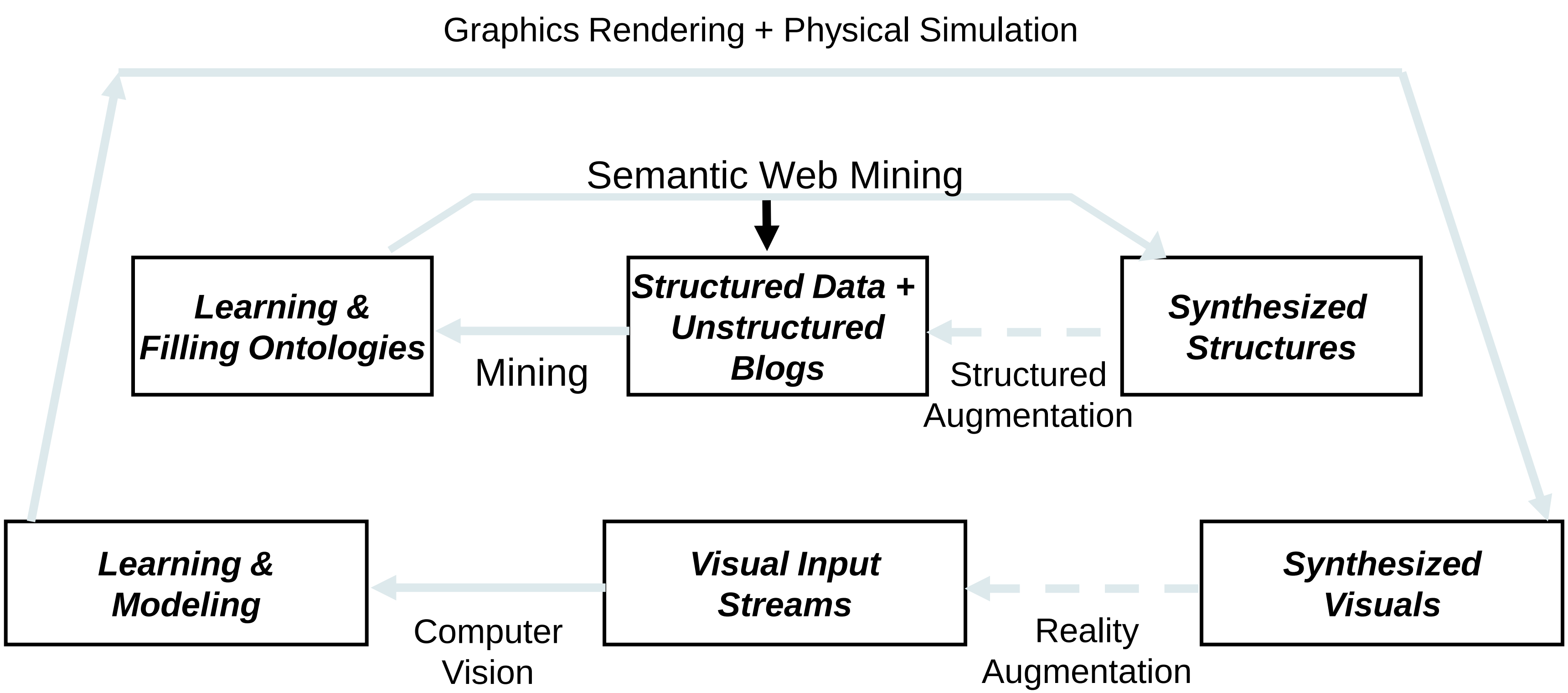
Massive Data & Ubiquitous Connectivity

- Data-driven models are now tractable and usable
  - We are not limited to analytical models any more
  - No need to rely on *heuristics* alone for unknown models
  - Massive data offers new algorithmic opportunities
    - Many traditional compute problems worth revisiting
- Web connectivity significantly speeds up model-training
- Real-time connectivity enables continuous model refinement
  - Poor model is an acceptable starting point
  - Classification accuracy improves over time

Pradeep.Dubey@intel.com

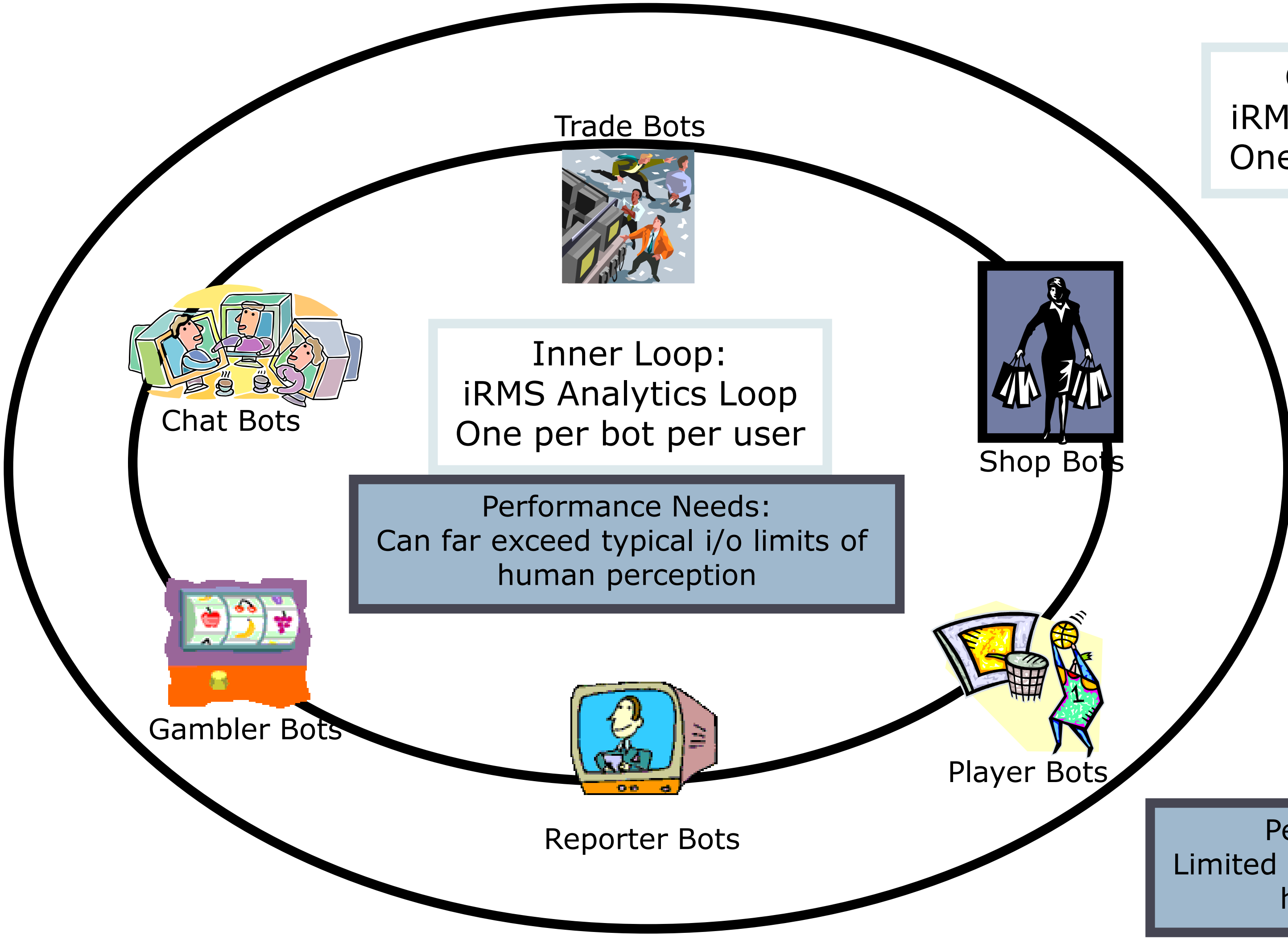
Nested RMS

Recognition	Mining	Synthesis
What is …?	Is it …?	What if …?



Pradeep.Dubey@intel.com

Nested RMS Instance: Virtual World

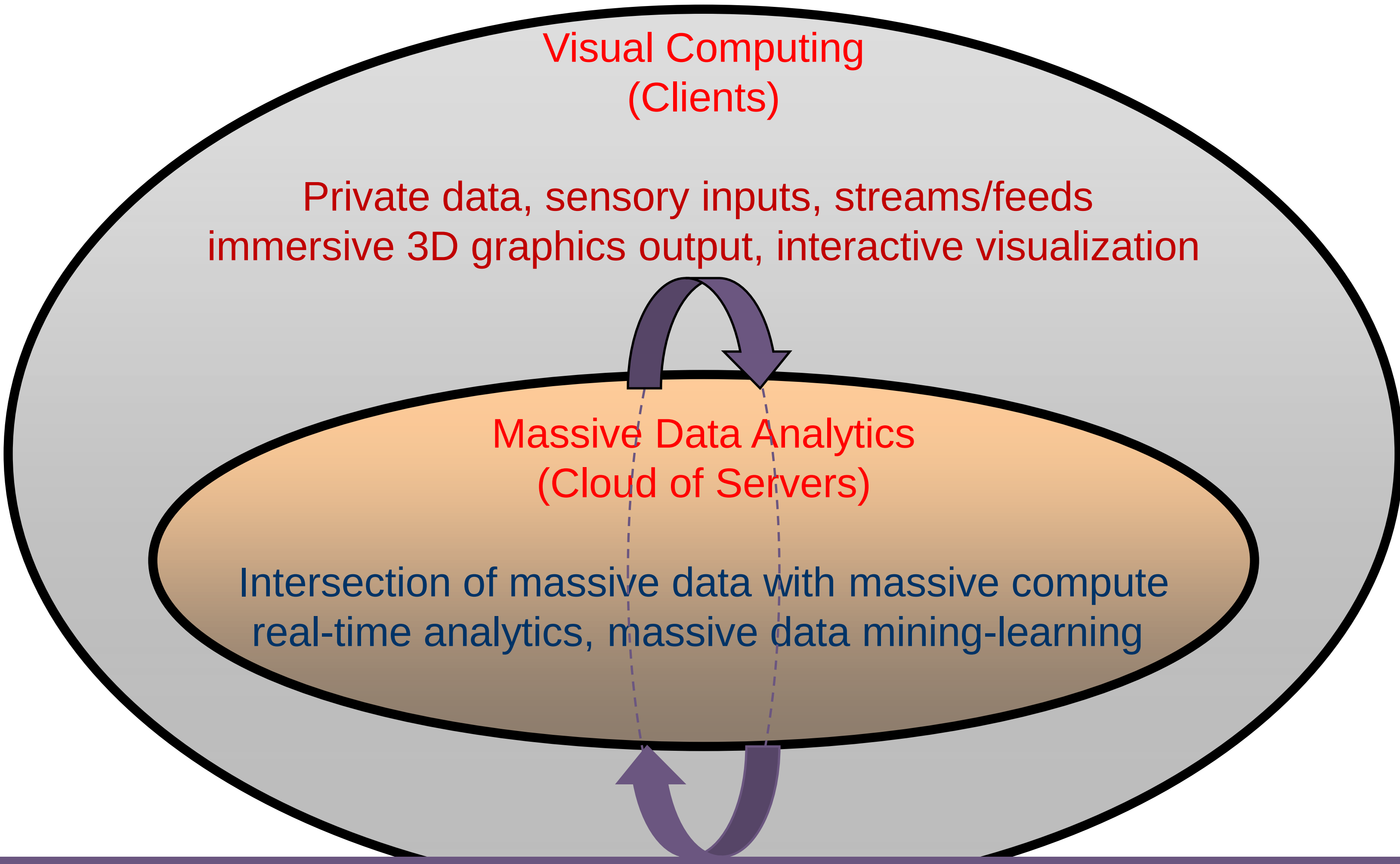


Outer loop:
iRMS Visual Loop:
One Per Real User

Performance Needs:
Limited by input/output limits of human perception

Pradeep.Dubey@intel.com Jan 2011

Where is my computer ☺



Architectural Implications Are Radical!

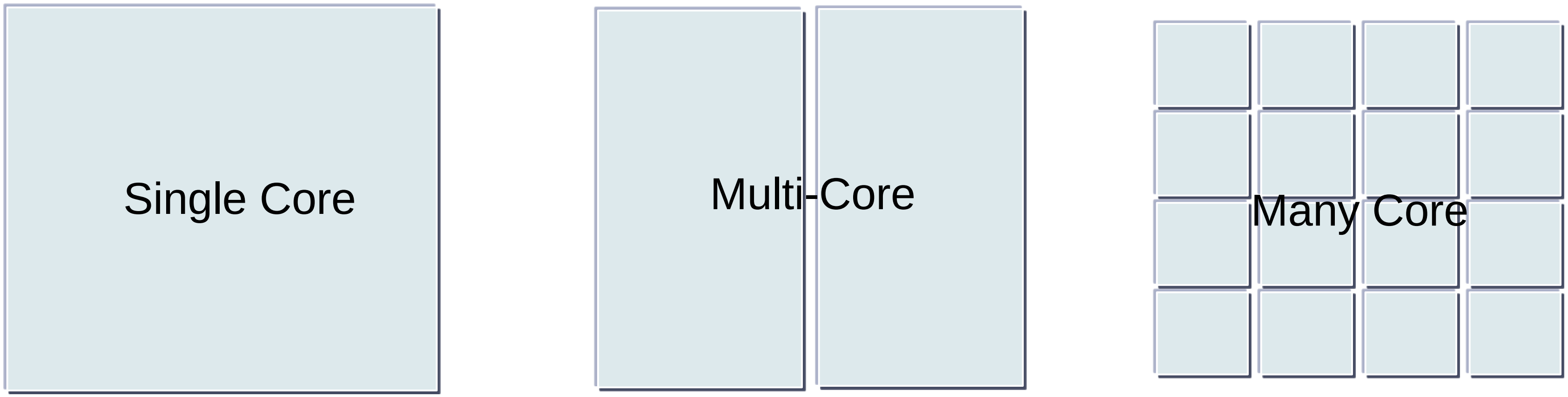
Pradeep.Dubey@intel.com

Architectural Challenges

- Compute density
- Data management: *Feeding the Beast*
- Distributed decomposition
- Non-ninja parallel programming

Pradeep.Dubey@intel.com

Multicore Versus Manycore



$$S = \frac{1}{(1-P) + \frac{P}{N}}$$

$$S = \frac{1}{(1-P)K_N + \frac{P}{N}}$$

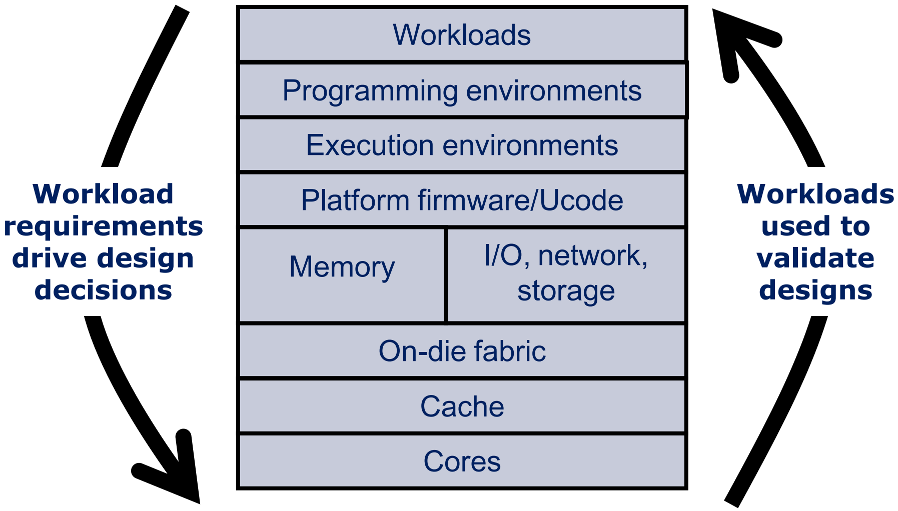
Many Core makes sense for workloads with high enough "P "- parallel component - for simplicity, we call these Highly Parallel

$$\text{For } S \geq 1,\ P \geq \frac{N(K_N - 1)}{NK_N - 1}$$

S = speedup, P = parallel fraction, # of Cores = N, Kn = single thread performance (single core/multicore)

Our Approach: Start at the Top

Architecture-aware analysis of computational needs of parallel applications (arch-app co-design)



Focus on specific co-travelers and domains: HPC/Imaging/Finance/Physical Simulations/Medical/…

Step 1: Algorithm/parallelization
Step 2: Architecture-specific
  Intel Xeon, Intel MIC, Nvidia GTX, …
Step 3: Platform-specific: CPU+GPU, multi-card, multi-node, cluster …
Step 4: Productivity or "Bridging the *Ninja* Gap"
  Languages: C/C++, OpenCL, Cuda, Ct (ArBB), …
  Libraries: MKL, domain-specific …

Architecture Specs

	Intel Westmere	Intel KNF
Sockets	2	1
Cores/socket	6	32
Core Frequency (GHz)	3.3	1.2
SIMD Width	4	16
Peak Compute	316 GFLOPS	1,228 GFLOPS

Ratio of peak compute = 4X

Pradeep.Dubey@intel.com

Case-Study-I (3-D Stencil Operations)[1]

Algorithm/Optimization	Incremental Speedup
SIMDfication	1.8X
Multi-threading (Non-blocked version is bandwidth bound)	2.1X
Perform Cache-blocking (2.5D Spatial + 1D Temporal)[2]	
Blocking Optimization	1.7X
Multi-threading (Blocked version is compute-bound and scales further)	1.8X
SIMD Further scaling of compute-bound code	1.9X
ILP Optimization	1.1X
Overall Speedup	**24.1X**

1. Performance data on Intel Core i7 975, 4c at 3.33 GHz
2. Details in SC'10 paper (3.5-D Blocking Optimization for Stencil Computations on Modern CPUs and GPUs by Nguyen et al.)

Case-Study-II (FFT)[1]

Algorithm/Optimization	Incremental Speedup
Algorithm (Radix-4 Vs/ Radix-2)	1.72X
Multi-threading (Naïve Partitioning)	3.05X
Multi-threading (Intelligent Partitioning: load balanced)	1.23X
SIMDfication (Full V/s Partial SIMD)	1.18X
Memory Management (Double Buffering)	1.32X

Overall Speedup	**10.1X**

1. Performance data on Intel Core i7 975, 4c at 3.33 GHz

Case-Study-III (Sparse Matrix Vector Multiplication)[1]

Algorithm/Optimization	Incremental Speedup
Multi-threading (Naïve Partitioning)	1.72X
Multi-threading (Intelligent Partitioning: load balanced)	2.2X
SIMDfication	1.13X
Cache Blocking	1.15X
Register Tiling	1.2X

Overall Speedup	**6.0X**

1. Performance data on Intel Core i7 975, 4c at 3.33 GHz

Case-Study-IV (Graph Traversal)[1]

Algorithm/Optimization	Incremental Speedup
Efficient Layout (Cache-Line Friendly)	10.1X
Hierarchical Blocking (Cache/TLB Friendly)	3.1X
SIMD	1.29X
ILP	1.35X
Multi-threading (Linear Scaling for compute-bound code)	3.9X
Overall Speedup	**212.6X**

1. Performance data on Intel Core i7 975, 4c at 3.33 GHz

Case-Study-V (Tree Search)[1,2]

Algorithm/Optimization	Incremental speedup
Efficient Layout (Memory Page-Blocking)	1.53X
Cache-Line Blocking	1.4X
SIMD	1.8X
ILP	2X
Multi-threading	3.9X

Overall Speedup	**30.1X**

1. Performance data on Intel Core i7 975, 4c at 3.33 GHz
2. Details in SIGMOD'10 paper (FAST: Fast Architecture Sensitive Tree Search on Modern CPUs and GPUs by Kim et al.)

Pradeep.Dubey@intel.com Jan 2011

Case-Study-VI (Matrix Multiply)[1, 2]

Algorithm/Optimization	Incremental Speedup
Loop Inversion	9X
Cache-Tiling	1.33X
Multithreading	2.4X
SIMD	2.2X

Overall Speedup	**64X**

1. Performance data on Intel Core i7 975, 4c at 3.33 GHz
2. HiPC'2010 (Goa, India) Tutorial "Architecture Specific Optimizations for Modern Processors" by Dhiraj Kalamkar et.al.

Learning

- Parallel algorithms offer best speedup-effort RoI
  - Algorithmic core needs to evolve from pre-multicore era
- Technology-aware algorithmic improvements offer the next best speedup-effort RoI
  - Increasing compute density and data-parallelism
- Special attention to the least-scaling part of modern architectures: BW/op will be increasingly more critical to performance
  - Locality aware transformations
- Architecture-specific speedup is orders of magnitude less than commonly believed
  - *100-1000x CPU-GPU speedup myth*

Pradeep.Dubey@intel.com

Summary

Massive Data Computing

- Insatiable appetite for compute
- It’s all about three C’s:
  - Content – Connect -- Compute

Algorithmic Opportunity

- Algorithmic core needs to evolve from serial to parallel
- Massive data approach to traditional compute problems
  - *Data … data everywhere, … not a bit of sense … ☺*

Performance Challenge

- Performance variability on the rise with parallel architectures
- *Feeding the Beast: increasingly a performance bottleneck*
- Programmer productivity key to market success

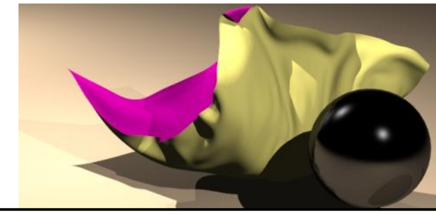
Pradeep.Dubey@intel.com

Thank You!

Questions?

Putting it all together

Sensory Immersion



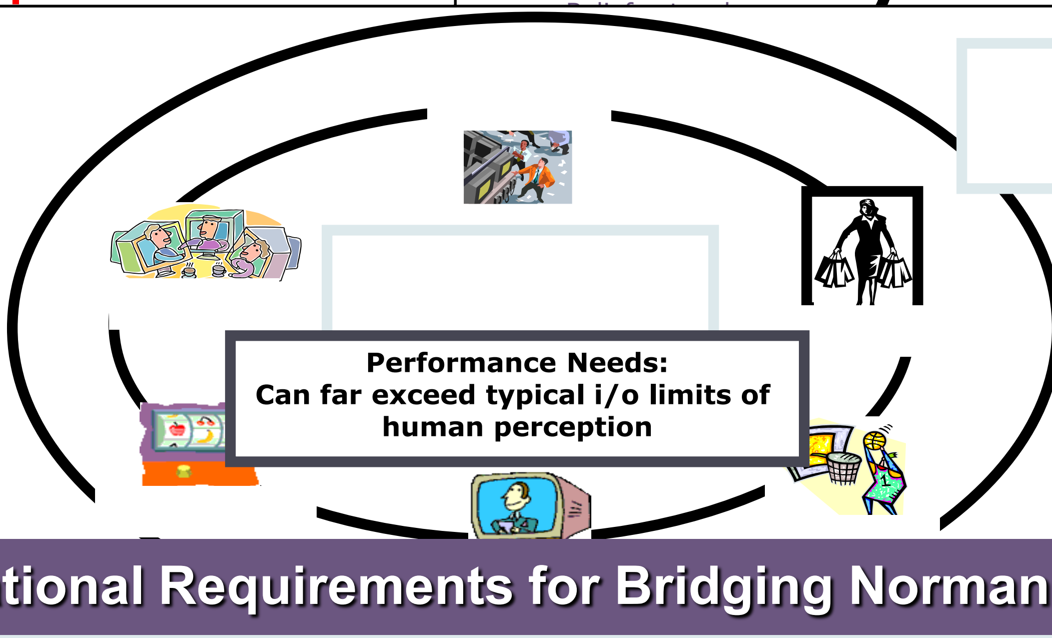
Behavioral Immersion

Rendering
Simulation

Machine learning
Neural networks
Probabilistic reasoning

Fuzzy logic

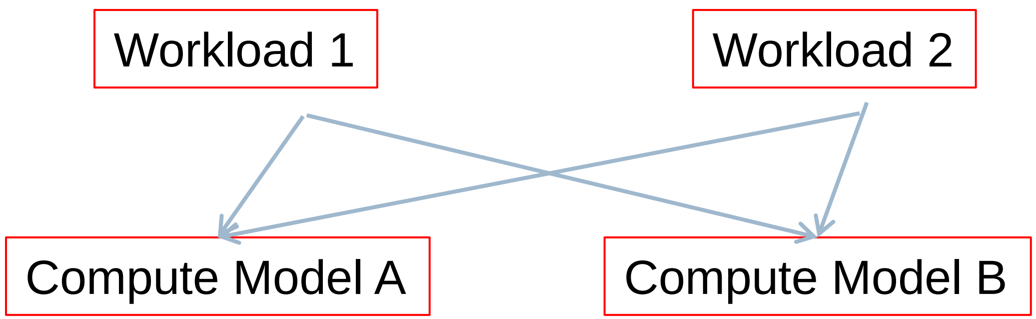
Super Immersion



Computational Requirements for Bridging Norman's Gulf Are Huge!

Heterogeneous Computing – What it is

- Platform-driven
- Workload-driven ← Our focus
- Power/Form-factor driven



Degree of Heterogeneity

Memory Mgmt: Explicit-Implicit
Data Parallelism: Explicit-Implicit
Threads: Latency-Throughput