Information Processing, Big Data, and the Cloud

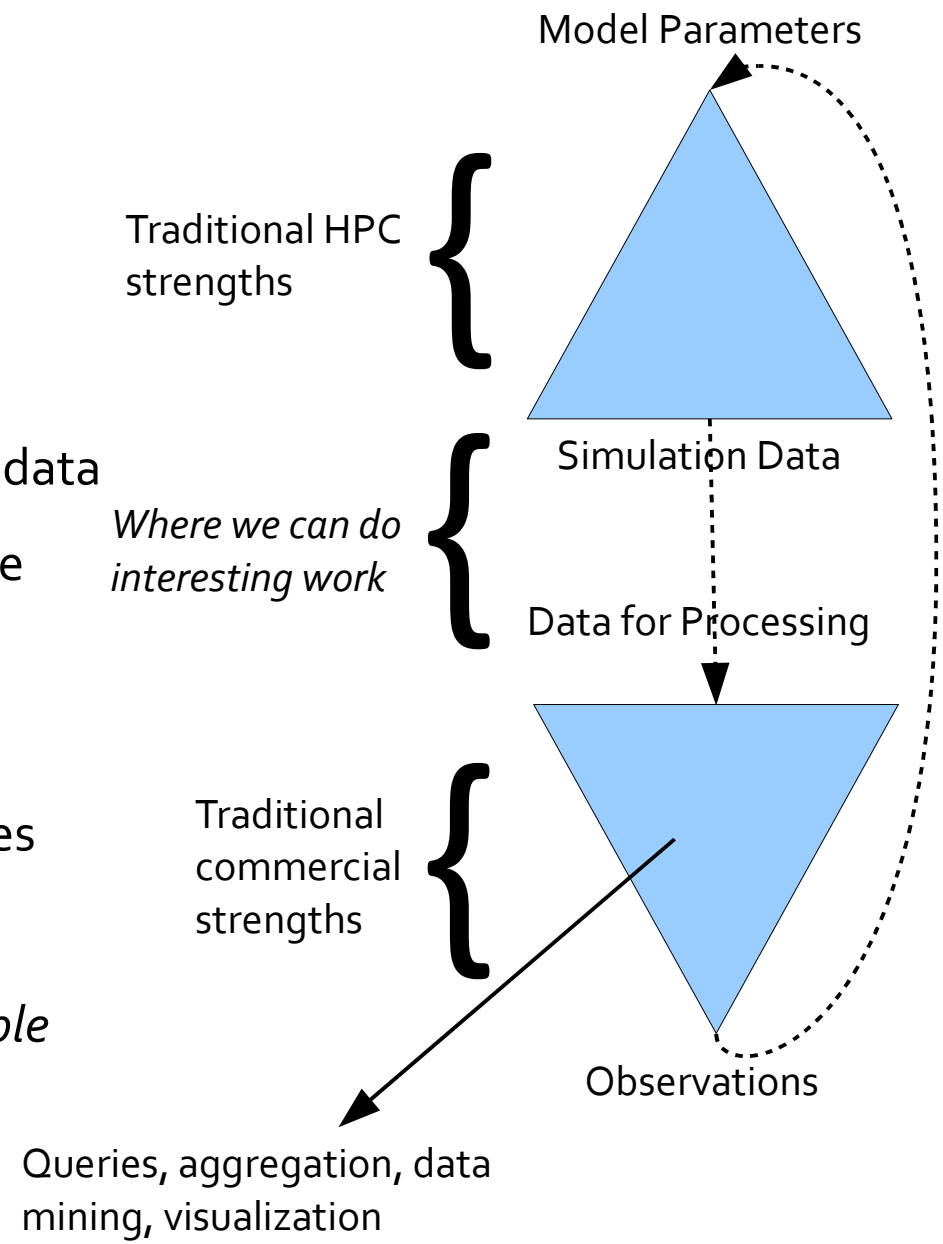
James Horey

Computational Sciences & Engineering
Oak Ridge National Laboratory

Fall Creek Falls 2010

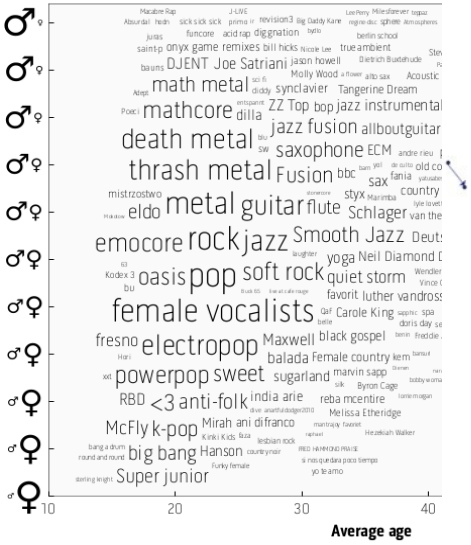
Information Processing Systems

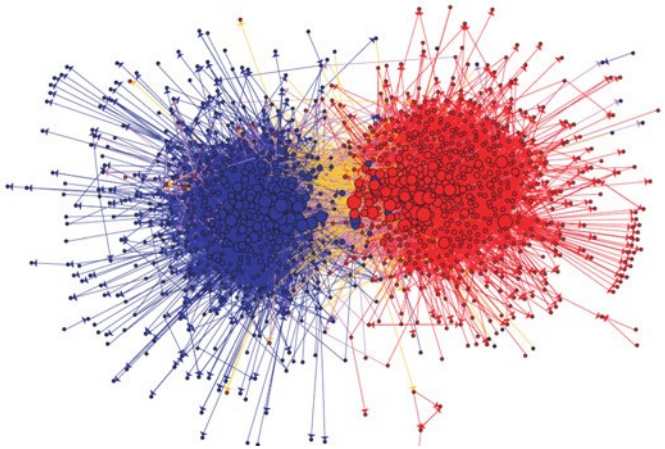
- Data-intensive computing
  - "Fourth" paradigm of science [J. Gray]
- Modeling & sim
  - Model parameters and generate lots of data
  - Assumes a-priori you know what you are doing
- Information processing
  - Take lots of data and produce summaries
  - You may not know what you want
- *These disciplines are not mutually incompatible*

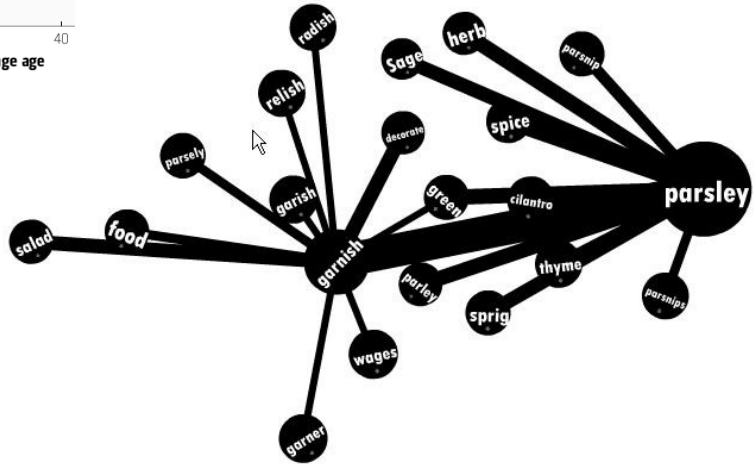


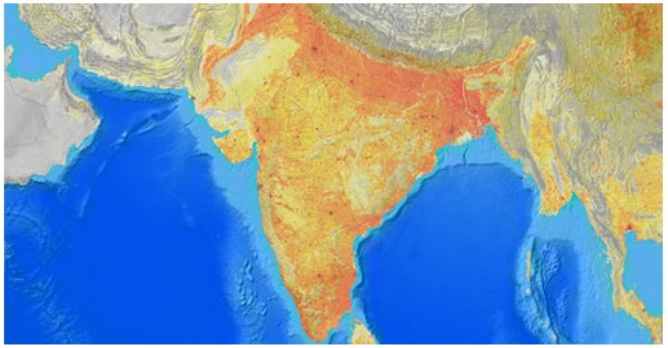
Sources of Data

- Text
  - Web, papers, emails
  - *Trends, NL-translation*
- Social media
  - Facebook, Twitter, Netflix
  - *Clique analysis, epidemic spread*
- Computer logs
  - Hardware failures, software bugs, security
  - *Predictive analysis, intrusion detection*
- Geospatial
  - GIS, satellite imagery, volunteered data
  - *Zonal statistics, population modeling*



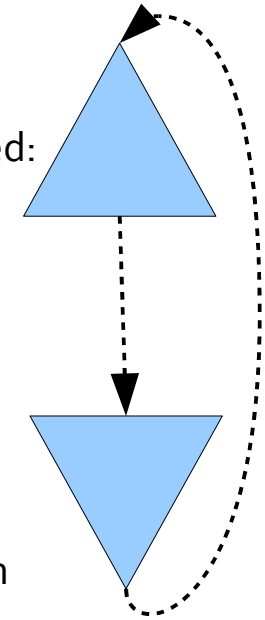






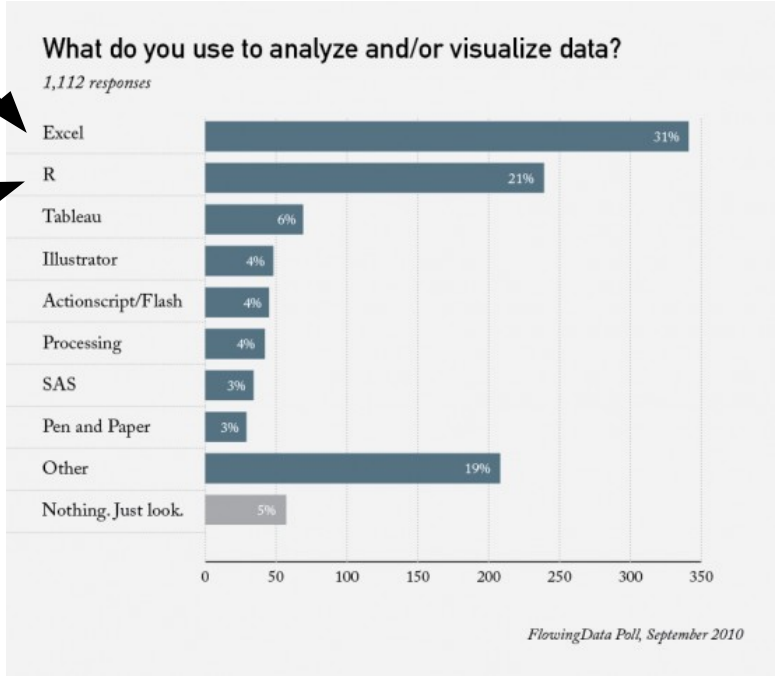
Significance

- Modeling & sim is heavily invested in parallel and distributed systems
  - More computers → better simulations
  - Modeling & sim is very heterogeneous, needs complicated tools (MPI)
- IPS mostly done on single machines
  - Data tools emphasize ease-of-use, visualization
  - IPS applications often homogeneous
- But things are changing
  - Google, FB, Twitter
  - More data, more complex, more tools



Most popular programming tool in the world

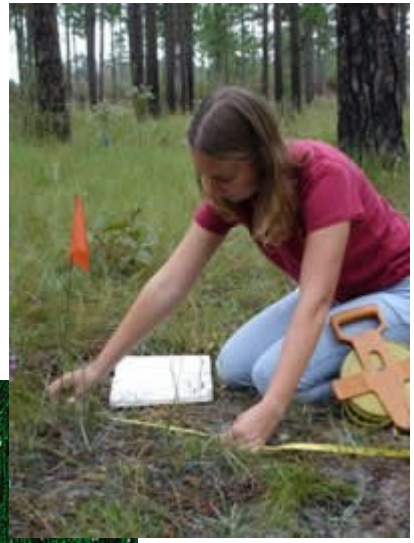
Only statisticians like R

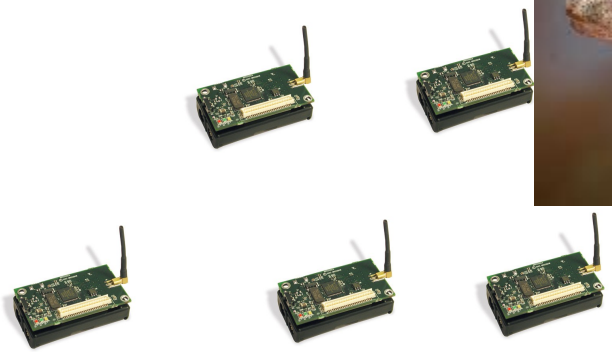


Data Singularity

- Once upon a time we were data starved
  - Collecting data was/is hard, slow, and analog
- Now we have more data than we know what to do with
  - Collecting data is become easy{ier}, fast{er}, and digital
  - Sensors, mobile devices, volunteered data
- Data generated at increasing rates
- We must analyze data at a rate faster than collection or we will drown

< 100 kb / day

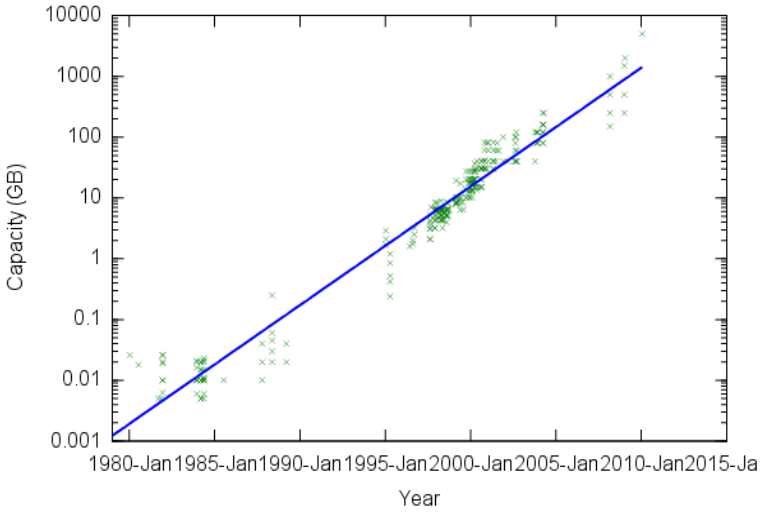


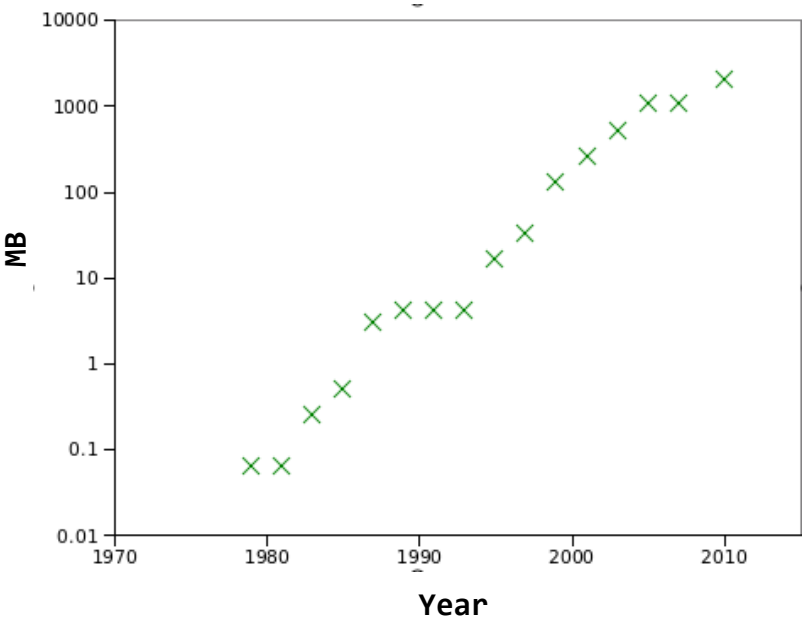


Application	Data generated per day
DNA Sequencing (Illumina HiSeq machine)	1 TB
Large Synoptic Survey Telescope	30 TB; 400 Mbps sustained data rate between Chile and NCSA
Large Hadron Collider	60 TB

What is Big Data?

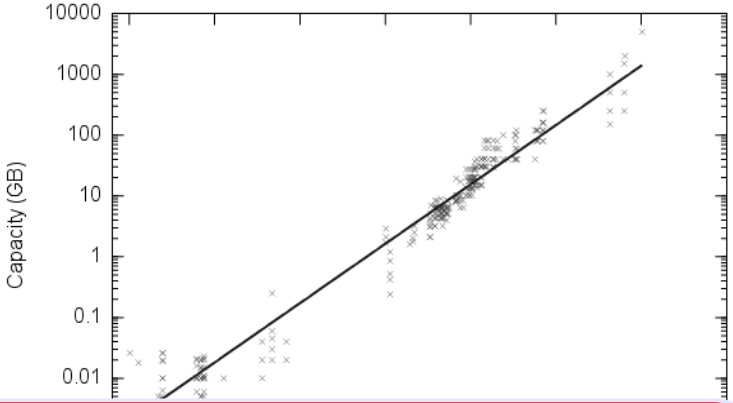
- Data-related problems that exceed local capacity
  - Data becoming available at collection/storage limits [Shankar]
  - Single-machine analysis approaching memory limits
    - "my graph won't fit in memory so I can't use R"
  - Gap is increasing
- Traditional data tools need to evolve
  - Employ distributed resources
  - Push computation to the edge for aggressive filtering
  - Enable wider range of data scientists



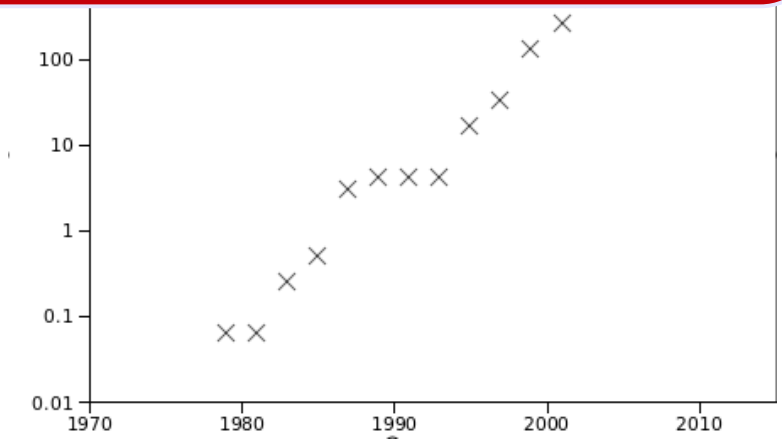


What is Big Data?

- Data-related problems that exceed local capacity
  - Data becoming available at collection/storage limits [Shankar]
  - Single-machine analysis approaching memory
  - Gap is increasing
- Traditional data tools need to evolve
  - Employ distributed resources
  - Push computation to the edge for aggressive filtering
  - Enable wider range of data scientists

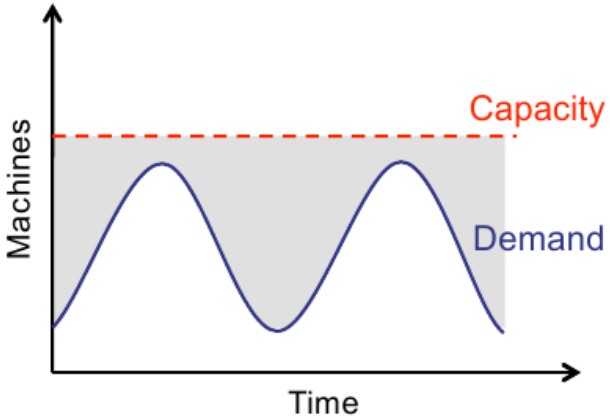


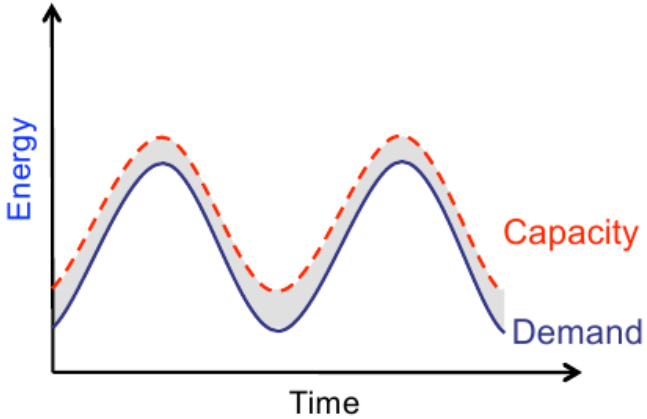
What will these tools look like?
Where can I get lots of machines?



Cloud Computing

- "Maybe I'm an idiot, but I have no idea what anyone is talking about" [L. Ellison]
- [Hardware, platform]-as-a-service
- No contracts, no grants, almost instantly available
- Enable scale-up and scale-down

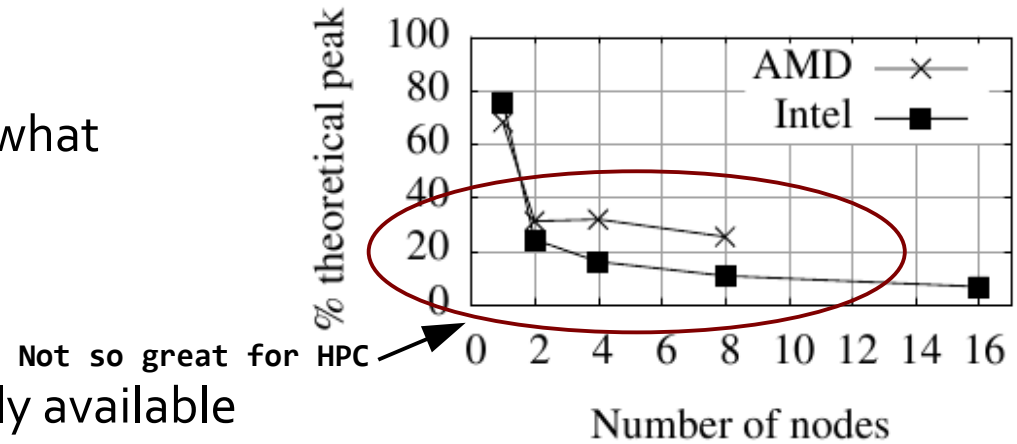


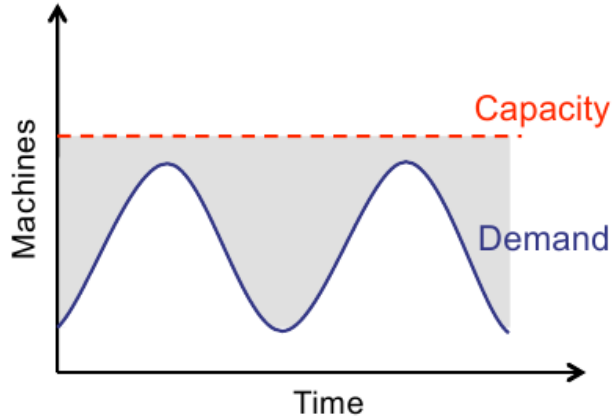


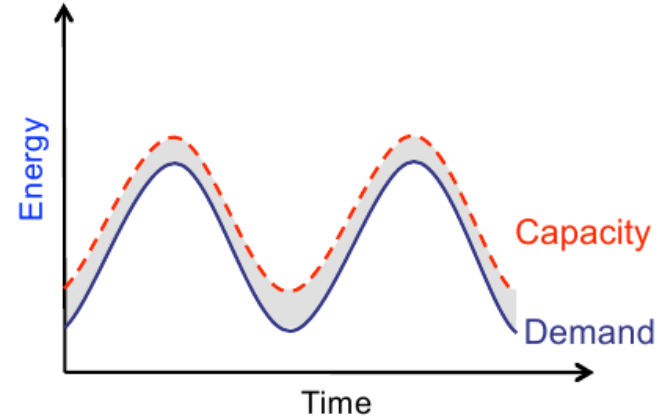
"Instances"	Platform	Cores	Memory	Disk
Small - $0.085 / hr	32-bit	1	1.7 GB	160 GB
Large - $0.34/ hr	64-bit	4	7.5 GB	850 GB – 2 spindles
XLarge - $0.68/ hr	64-bit	8	15.0 GB	1690 GB – 3 spindles
Options....extra memory, extra CPU, extra disk, ...				

Cloud Computing

- "Maybe I'm an idiot, but I have no idea what anyone is talking about" [L. Ellison]
- [Hardware, platform]-as-a-service
- No contracts, no grants, almost instantly available
- Enable scale-up and scale-down
- Opaque interconnect



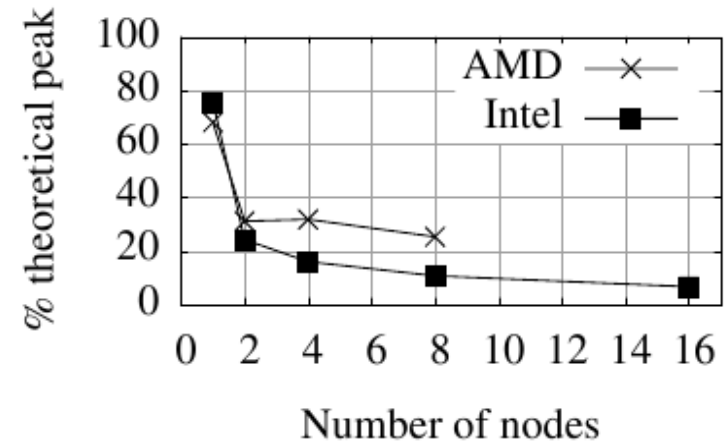




"Instances"	Platform	Cores	Memory	Disk
Small - $0.085 / hr	32-bit	1	1.7 GB	160 GB
Large - $0.34/ hr	64-bit	4	7.5 GB	850 GB – 2 spindles
XLarge - $0.68/ hr	64-bit	8	15.0 GB	1690 GB – 3 spindles
Options....extra memory, extra CPU, extra disk, ...				

Cloud Computing

- "Maybe I'm an idiot, but I have no idea what anyone is talking about" [L. Ellison]
- [Hardware, platform]-as-a-service
- No contracts, no grants, almost instantly available



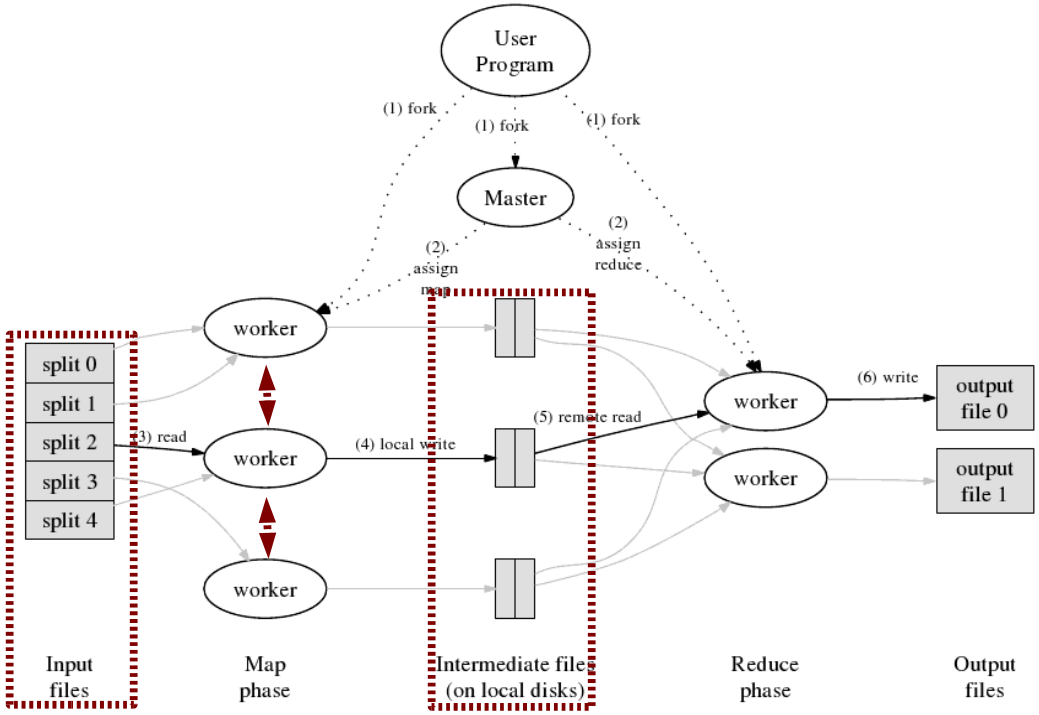
Can Clouds help us with the Big Data problem?

- Opaque interconnect

"Instances"	Platform	Cores	Memory	Disk
Small - $0.085 / hr	32-bit	1	1.7 GB	160 GB
Large - $0.34/ hr	64-bit	4	7.5 GB	850 GB – 2 spindles
XLarge - $0.68/ hr	64-bit	8	15.0 GB	1690 GB – 3 spindles
Options....extra memory, extra CPU, extra disk, ...				

MapReduce as a Starting Point

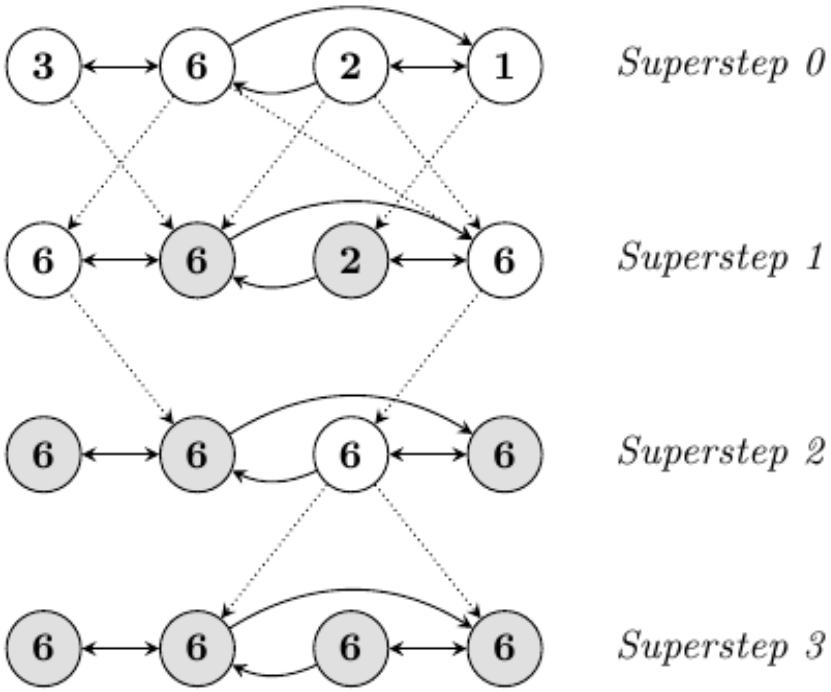
- MapReduce is a functional, data-parallel model
  - Maps compute over individual data elements, output <key, value> pairs
  - System sorts by key
  - Reduce operations compute over collections of data elements by key
- Reverse index, PageRank, word histograms
  - Map operations run in parallel
  - Reduce operations parallelized by keys
- Redundant map/reduce instances for fault tolerance
- Scales to > 10,000 cores



[Dean, Ghemawat '04]

MapReduce for Graph Processing

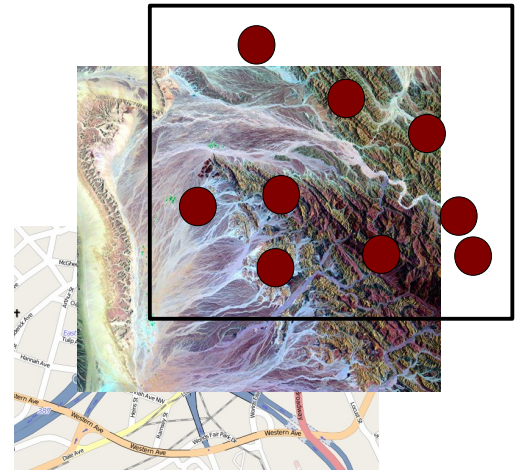
- Google Pregel
  - Local functions applied to every vertex
  - Vertices send/receive messages across edges at each timestep
  - Nodes retain state across iterations (no need to touch disk)
  - Global aggregation
- Consensus, clustering, bipartite matching
- Hashing + local lookup for communication

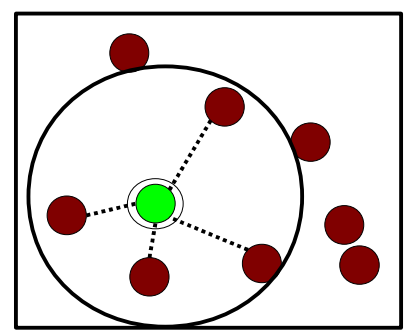


[Malewicz 2010]

MapReduce for Geospatial

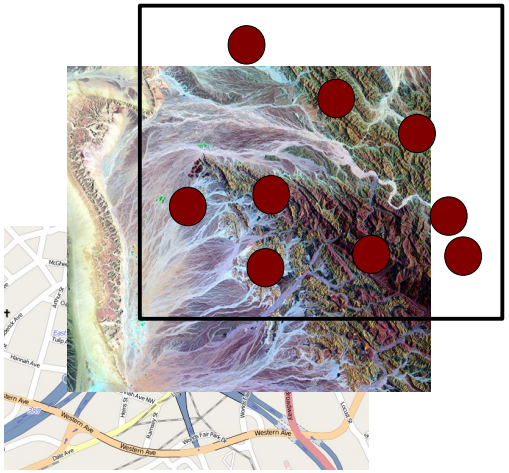
- SenseReduce
  - Designed for spatiotemporal data
  - Operate over temporal windows
- Ability to layer multiple data sources
  - "Join these vector files and perform zonal statistics with this raster file"
- Ability to read nearby geographic features
  - Bulk Synchronous, Abstract Regions
- Stack multiple analysis functions
  - Perform average, then min, etc.
  - Reduces redundant work

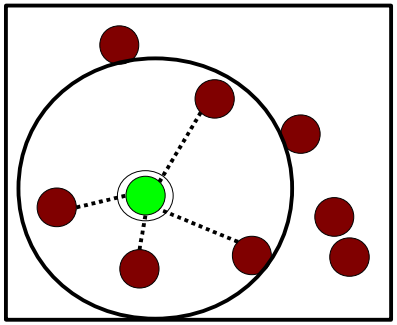




MapReduce for Geospatial

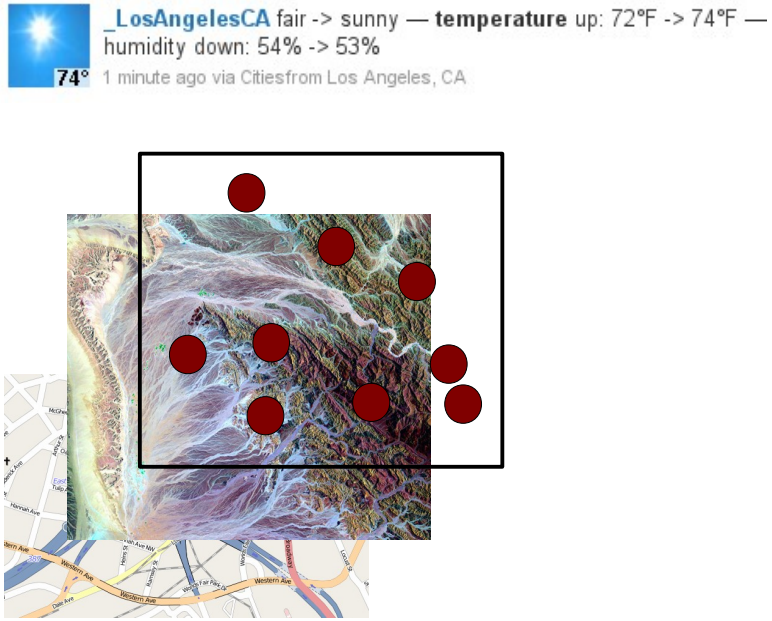
- SenseReduce
  - Designed for spatiotemporal data
  - Operate over temporal windows
- Ability to layer multiple data sources
  - "Join these vector files and perform zonal statistics with this raster file"
- Ability to read nearby geographic features
  - Bulk Synchronous, Abstract Regions
- Stack multiple analysis functions
  - Perform average, then min, etc.
  - Reduces redundant work

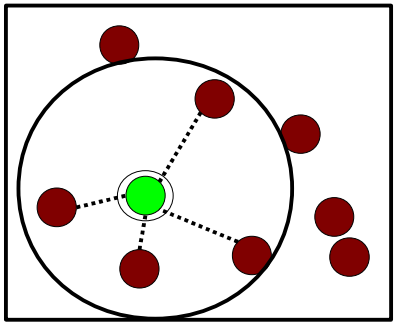




MapReduce for Geospatial

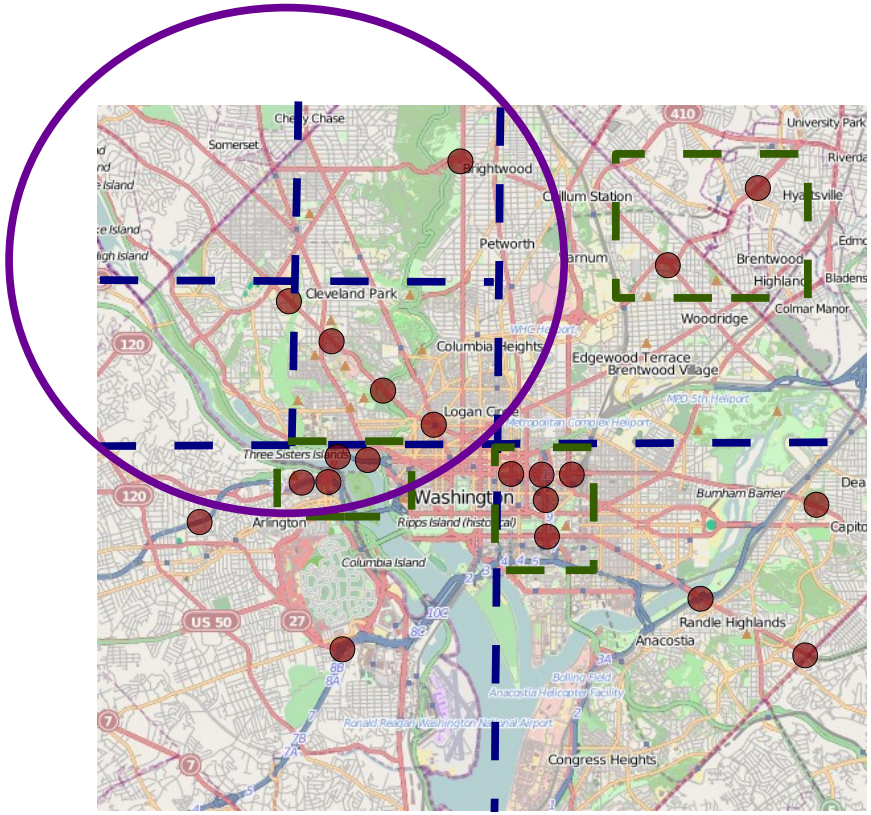
- SenseReduce
  - Designed for spatiotemporal data
  - Operate over temporal windows
- Ability to layer multiple data sources
  - "Join these vector files and perform zonal statistics with this raster file"
- Ability to read nearby geographic features
  - Bulk Synchronous, Abstract Regions
- Stack multiple analysis functions
  - Perform average, then min, etc.
  - Reduces redundant work





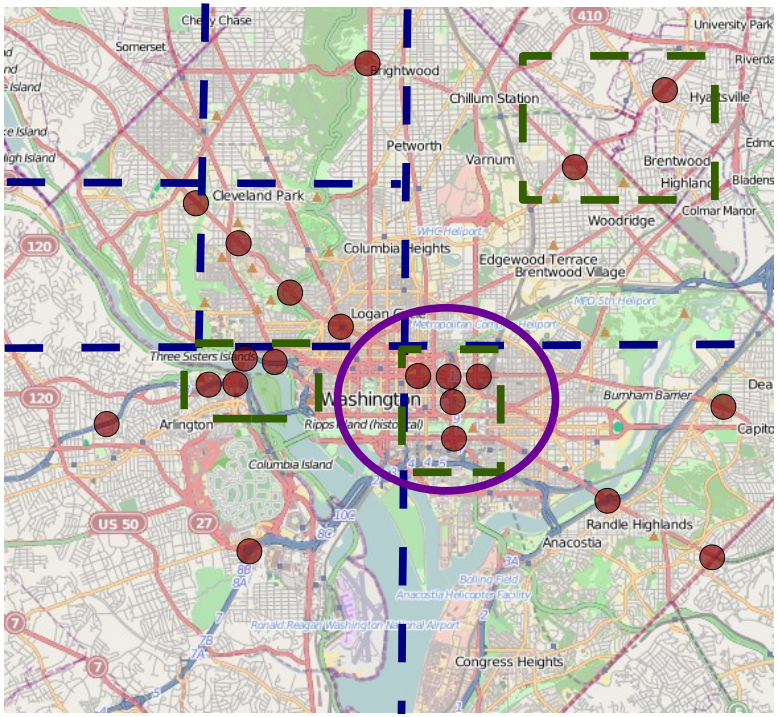
Partitioning Algorithms

- Data must be split across multiple machines
- Divide features into equal areas
  - May artificially split features
  - *Metadata update consistency a problem*
- Divide by logical features
  - May lead to unbalance if features have different complexity
- Customized partitioning schemes, including ability to "stack"
  - i.e. first split by feature, then by area



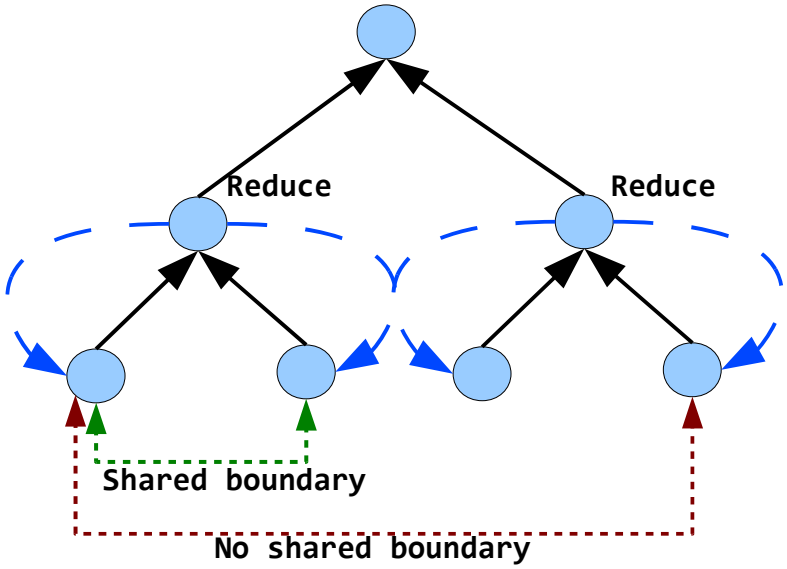
Partitioning Algorithms

- Data must be split across multiple machines
- Divide features into equal areas
  - May artificially split features
  - *Metadata update consistency a problem*
- Divide by logical features
  - May lead to unbalance if features have different complexity
- Customized partitioning schemes, including ability to "stack"
  - i.e. first split by feature, then by area



Scheduling and Load-balancing

- Use geography-constrained aggregation tree
  - Parents perform partial reduce
  - Parents also point to geographically close nodes → minimizes communication across nodes, simplifies load-balancing
- Streaming data changes computation load over time
  - Mobile sensors
  - Weather remote sensing
- Rebalance computation during map and reduce stages
  - Number of features as indicators of load

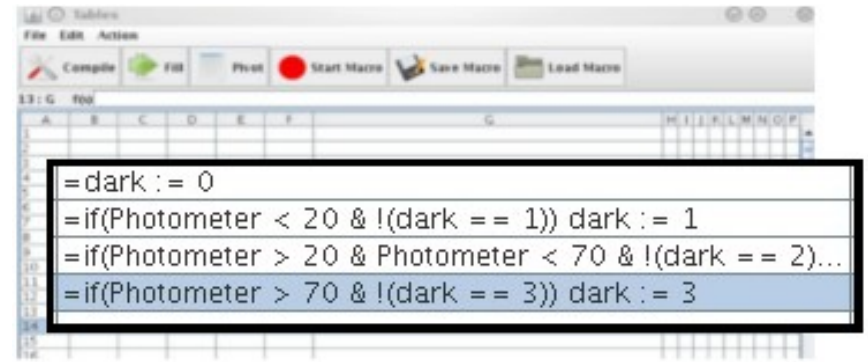


Programming at the Edge

- Data from "smart" network of sensors
  - CPU + storage + communication
- Edge processing
  - Reduce cost
  - Reduce latency
- *What is the right model for edge processing?*
  - MapReduce ↔ data logger
  - Functional data stream [Newton], spreadsheets [Horey, DCOSS],
- *How to schedule aggregation on these devices?*
  - Assuming: information processing has to take place somewhere [Horey, KDCloud]

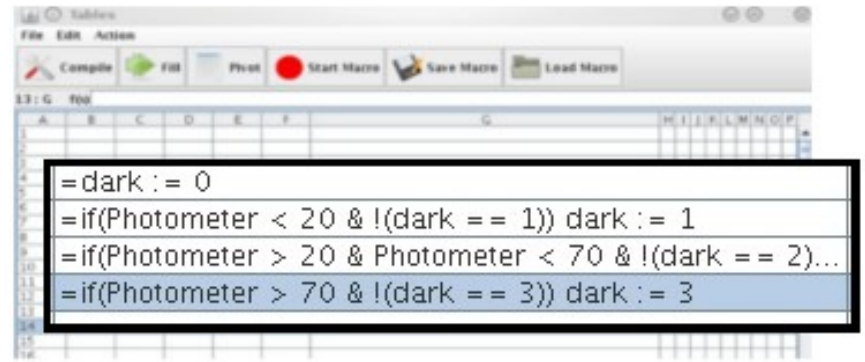
Programming at the Edge

- Data from "smart" network of sensors
  - CPU + storage + communication
- Edge processing
  - Reduce cost
  - Reduce latency
- *What is the right model for edge processing?*
  - MapReduce ↔ data logger
  - Functional data stream [Newton], spreadsheets [Horey, DCOSS],
- *How to schedule aggregation on these devices?*
  - Assuming: information processing has to take place somewhere [Horey, KDCloud]

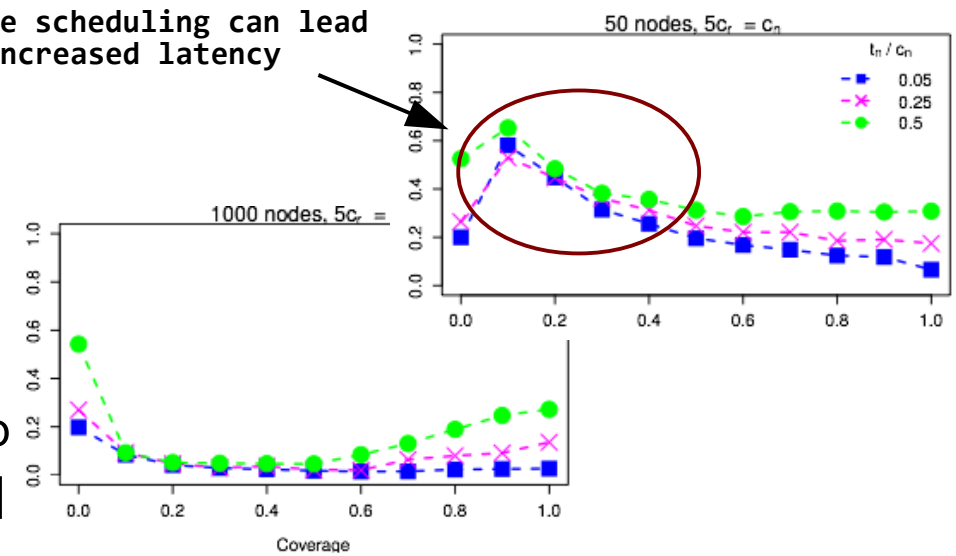


Programming at the Edge

- Data from "smart" network of sensors
  - CPU + storage + communication
- Edge processing
  - Reduce cost
  - Reduce latency
- *What is the right model for edge processing?*
  - MapReduce ↔ data logger
  - Functional data stream [Newton], spreadsheets [Horey, DCOSS],
- *How to schedule aggregation on these devices?*
  - Assuming: information processing has to take place somewhere [Horey, KDCloud]



Naïve scheduling can lead to increased latency

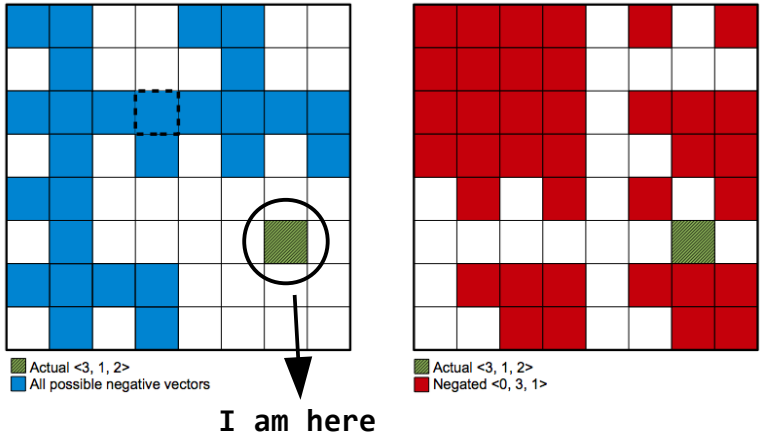


Dangers and Pitfalls

- Lots of data → lots of opportunities for security and privacy violations
- People will *voluntarily* give you sensitive information
  - And sue you later for using it
- *Can we implement information processing applications in an anonymous fashion?*
  - In general: No (Netflix prize)
  - For specific apps: Yes
- Negative Quad Tree
  - Don't report your location
  - Still build spatial distributions
  - ~100,000 samples in 32x32 area (1 km^2)

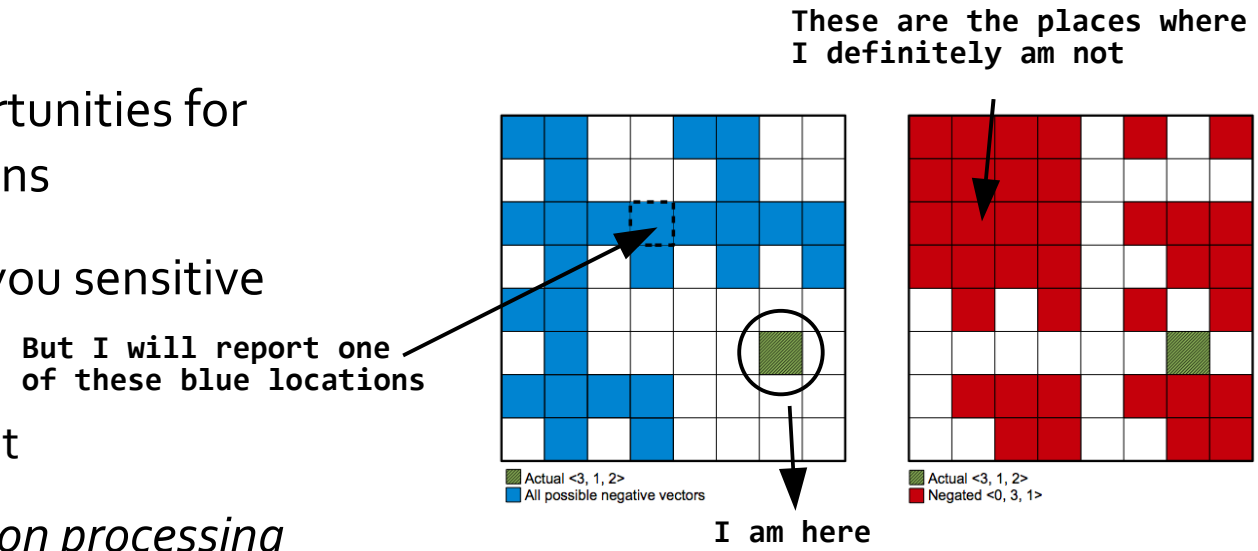
Dangers and Pitfalls

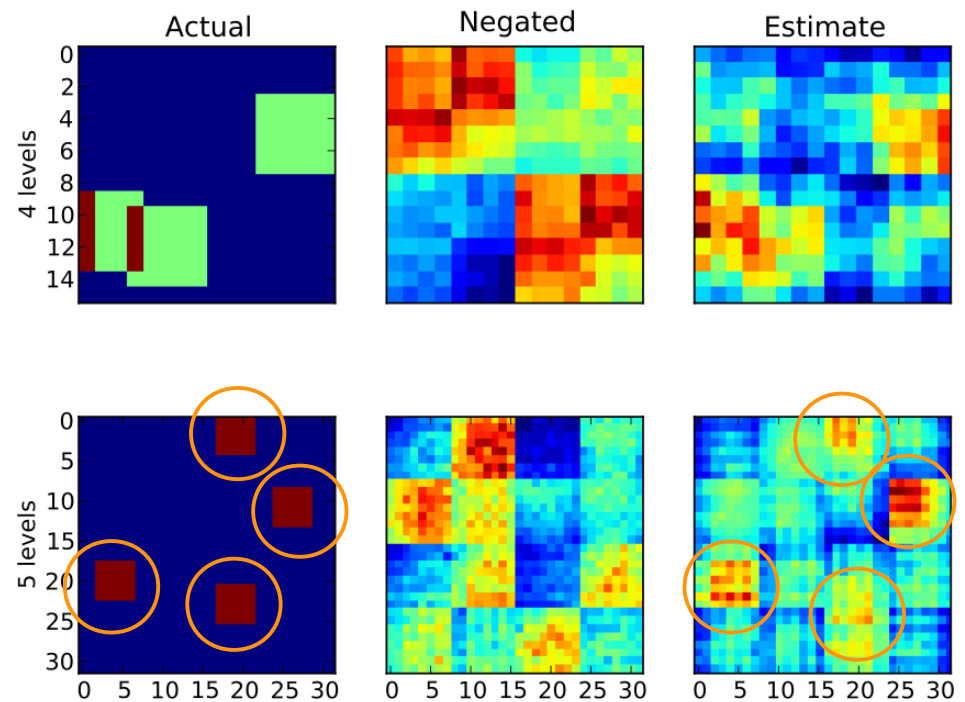
- Lots of data → lots of opportunities for security and privacy violations
- People will *voluntarily* give you sensitive information
  - And sue you later for using it
- *Can we implement information processing applications in an anonymous fashion?*
  - In general: No (Netflix prize)
  - For specific apps: Yes
- Negative Quad Tree
  - Don't report your location
  - Still build spatial distributions
  - ~100,000 samples in 32x32 area (1 km^2)



Dangers and Pitfalls

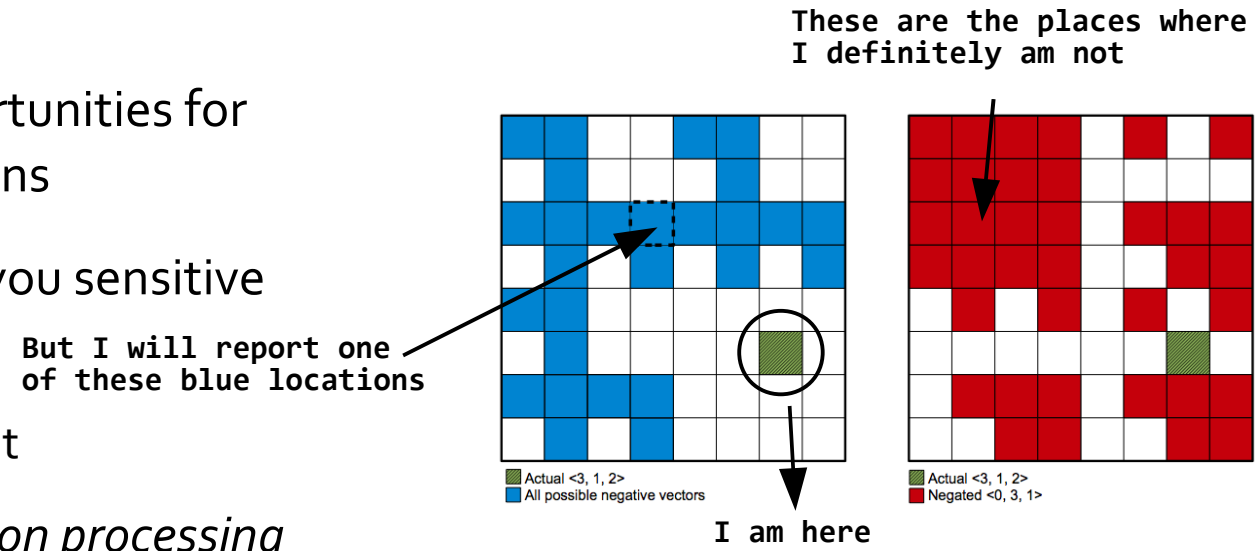
- Lots of data → lots of opportunities for security and privacy violations
- People will *voluntarily* give you sensitive information
  - And sue you later for using it
- *Can we implement information processing applications in an anonymous fashion?*
  - In general: No (Netflix prize)
  - For specific apps: Yes
- Negative Quad Tree
  - Don't report your location
  - Still build spatial distributions
  - ~100,000 samples in 32x32 area (1 km^2)

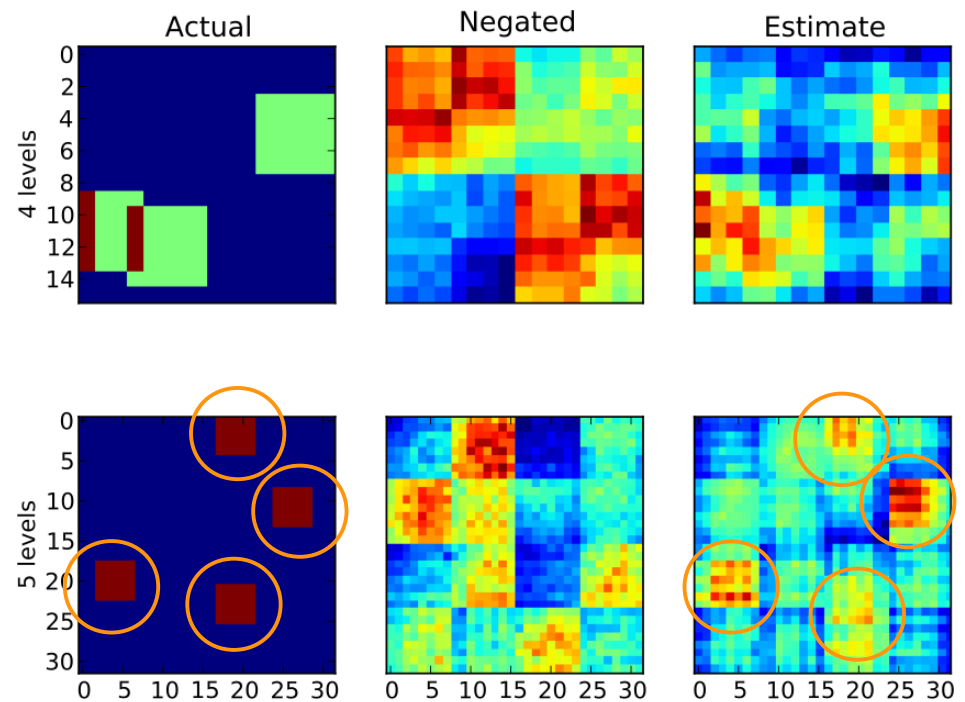




Dangers and Pitfalls

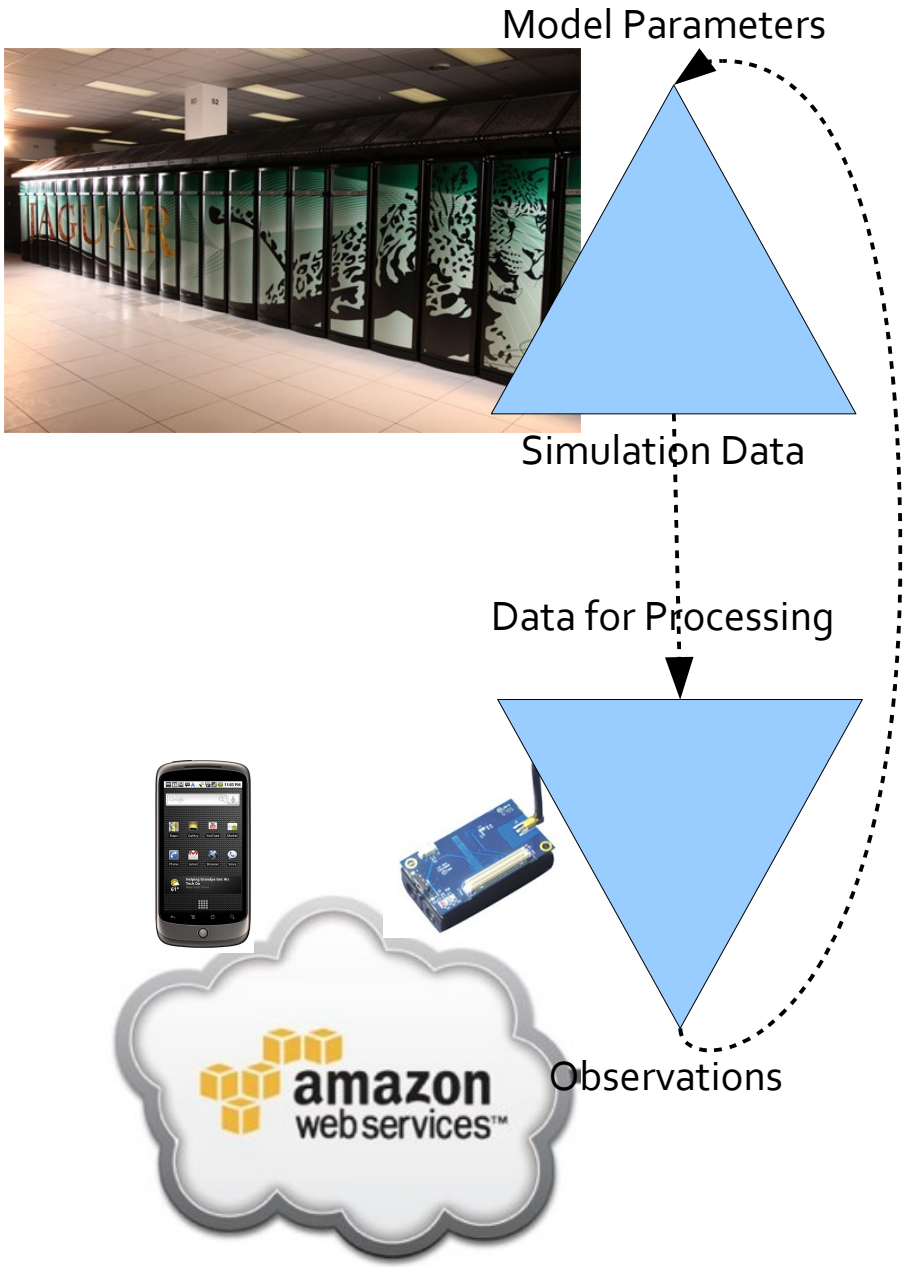
- Lots of data → lots of opportunities for security and privacy violations
- People will *voluntarily* give you sensitive information
  - And sue you later for using it
- *Can we implement information processing applications in an anonymous fashion?*
  - In general: No (Netflix prize)
  - For specific apps: Yes
- Negative Quad Tree
  - Don't report your location
  - Still build spatial distributions
  - ~100,000 samples in 32x32 area (1 km^2)





Open Questions

- *Do we have a single programming model?*
  - Cloud → Mobile → Sensors, Text → Graphs → Geospatial
- *Will computation actually run at the edge?*
  - Need compelling cyberphysical examples (latency, cost, etc.)
- *Are there lessons from HPC that can be applied to IPS (and vice-versa)?*
  - Non-blocking collectives [Hoefler]
- *What is the right storage paradigm?*
  - Posix, SQL, BigTable, VDB
- *What programming language?*
  - IPS users are more varied
- *What application domains can benefit from both approaches?*
  - Computational transportation

Conclusions

- HPC applications and platforms will directly affect and benefit from IPS
  - Bigger simulations → more data → more IPS
  - Simulations + sensor data → better simulations
- This will require new methods and tools
  - Better programming tools
  - Bigger computational platforms
- Tools and methods benefit each other
  - MPI ↔ MapReduce