X-Informatics Parallel Computing Overview of Basic Principles

June 19 2013

Geoffrey Fox
gcf@indiana.edu
http://www.infomall.org/X-InformaticsSpring2013/index.html

Associate Dean for Research, School of Informatics and Computing
Indiana University Bloomington

2013

Big Data Ecosystem in One Sentence

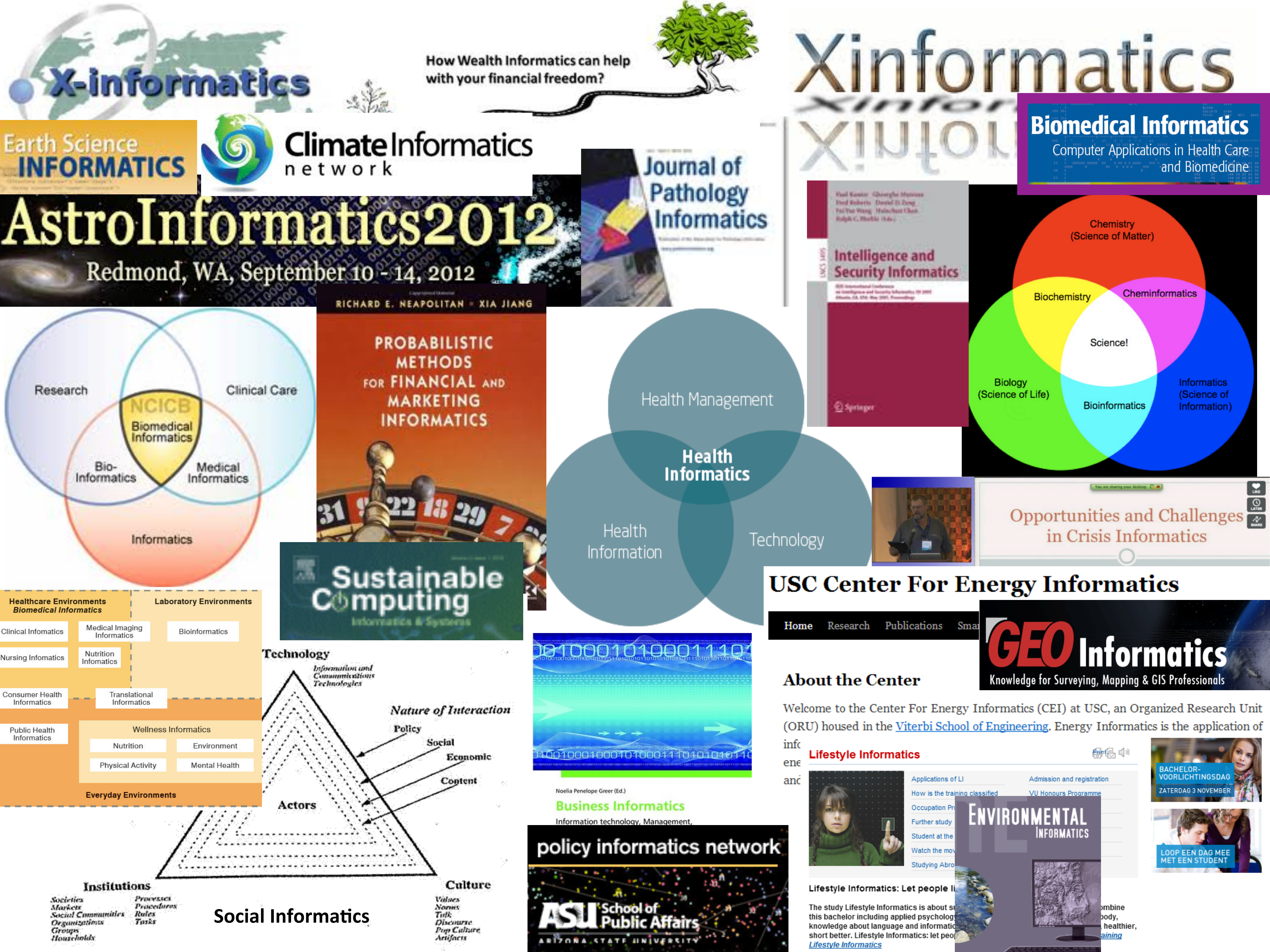
Use Clouds running Data Analytics Collaboratively processing Big Data to solve problems in X-Informatics (or e-X)

X = Astronomy, Biology, Biomedicine, Business, Chemistry, Climate, Crisis, Earth Science, Energy, Environment, Finance, Health, Intelligence, Lifestyle, Marketing, Medicine, Pathology, Policy, Radar, Security, Sensor, Social, Sustainability, Wealth and Wellness with more fields (physics) defined implicitly

Spans Industry and Science (research)

Education: Data Science see recent New York Times articles

http://datascience101.wordpress.com/2013/04/13/new-york-times-data-science-articles/


X-informatics
How Wealth Informatics can help with your financial freedom?
Xinformatics
Earth Science INFORMATICS
Climate Informatics network
Journal of Pathology Informatics
Biomedical Informatics
Computer Applications in Health Care and Biomedicine
AstroInformatics2012
Redmond, WA, September 10 - 14, 2012
RICHARD E. NEAPOLITAN • XIA JIANG
PROBABILISTIC METHODS FOR FINANCIAL AND MARKETING INFORMATICS
Intelligence and Security Informatics
Springer
Chemistry (Science of Matter)
Biochemistry
Cheminformatics
Science!
Biology (Science of Life)
Bioinformatics
Informatics (Science of Information)
Research
Clinical Care
NCICB
Biomedical Informatics
Bio-Informatics
Medical Informatics
Informatics
Health Management
Health Informatics
Health Information
Technology
Opportunities and Challenges in Crisis Informatics
Sustainable Computing
Informatics & Systems
USC Center For Energy Informatics
Home
Research
Publications
About the Center
Welcome to the Center For Energy Informatics (CEI) at USC, an Organized Research Unit (ORU) housed in the Viterbi School of Engineering. Energy Informatics is the application of
GEO Informatics
Knowledge for Surveying, Mapping & GIS Professionals
Healthcare Environments
Biomedical Informatics
Laboratory Environments
Clinical Infomatics
Medical Imaging Informatics
Bioinformatics
Nursing Infomatics
Nutrition Infomatics
Consumer Health Informatics
Translational Informatics
Public Health Informatics
Wellness Informatics
Nutrition
Environment
Physical Activity
Mental Health
Everyday Environments
Technology
Information and Communications Technologies
Nature of Interaction
Policy
Social
Economic
Content
Actors
Institutions
Societies
Markets
Social Communities
Organizations
Groups
Households
Processes
Procedures
Rules
Tasks
Culture
Values
Norms
Talk
Discourse
Pop Culture
Artifacts
Social Informatics
Noelia Penelope Greer (Ed.)
Business Informatics
Information technology, Management,
policy informatics network
ASU School of Public Affairs
ARIZONA STATE UNIVERSITY
Lifestyle Informatics
Applications of LI
How is the training classified
Admission and registration
VU Honours Programme
ENVIRONMENTAL INFORMATICS
BACHELOR-VOORLICHTINGSDAG
ZATERDAG 3 NOVEMBER
LOOP EEN DAG MEE MET EEN STUDENT
Lifestyle Informatics: Let people li
Lifestyle Informatics

Decomposition

Parallel Computing I

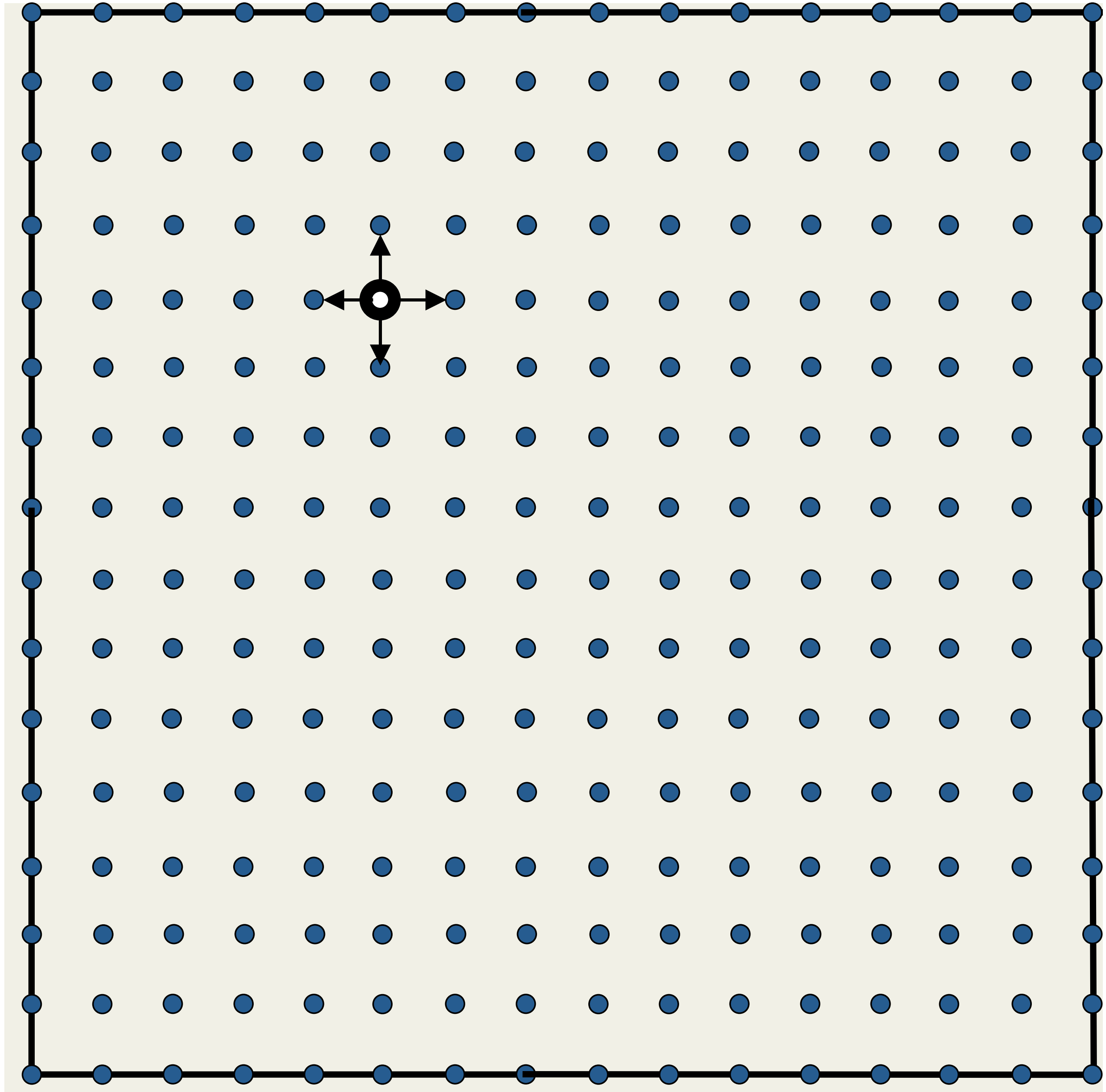
- Our applications are running in a cloud with up to 100K servers in it
  - A million or so cores each of which is independent
- When you or rather your phone/tablet/laptop accesses cloud, you take a core which handles your request
  - So we can easily handle a million requests at same time
- This is simplest form of parallel computing – parallelism over jobs or users
- But other forms are needed to
  - Access giant databases
  - Calculate model-based recommendations linking items together
  - Do a large clustering
  - Answer a search question (similar to accessing giant database)
- These use "data parallelism" also used on supercomputers to calculate complex model of universe, fusion, battery, deformation of car in crash, evolution of molecules in a protein dynamics

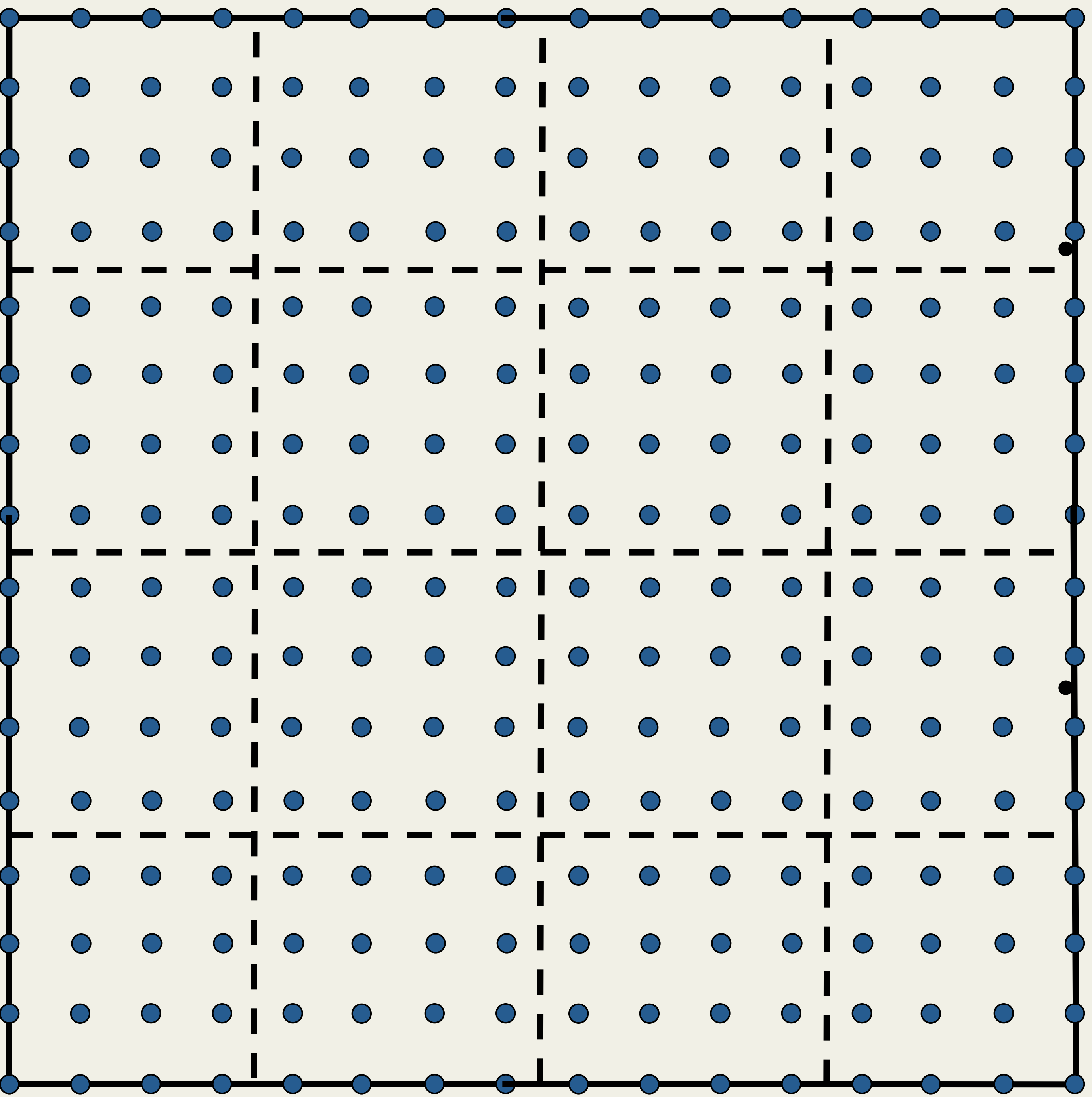
Parallel Computing II

- Instead of dividing cores by users we (also) divide (some of) cores so each has a fraction of data
- In study of air flow over a plane, you divide air into a mesh and determine pressure, density, velocity at each mesh point
  - Weather forecasting similar
- Here data is mesh points
- For search, data is web pages
- For eCommerce, data is items and/or people
- For Kmeans, data is points to be clustered

Update on the Mesh

14 by 14 Internal Mesh

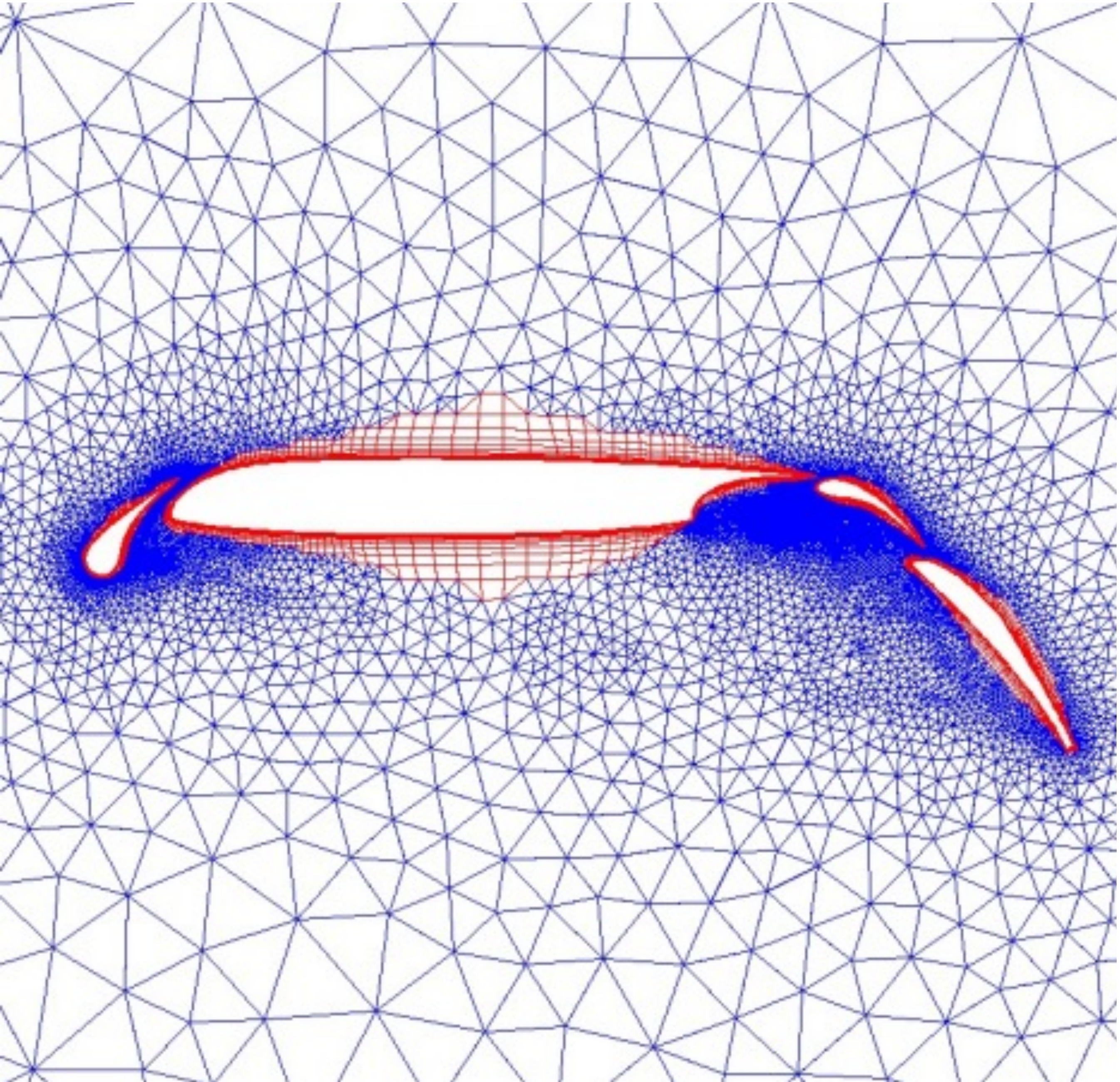




Parallelism is Straightforward

- If one has 16 processors, then decompose geometrical area into 16 equal parts
- Each Processor updates 9 12 or 16 grid points independently

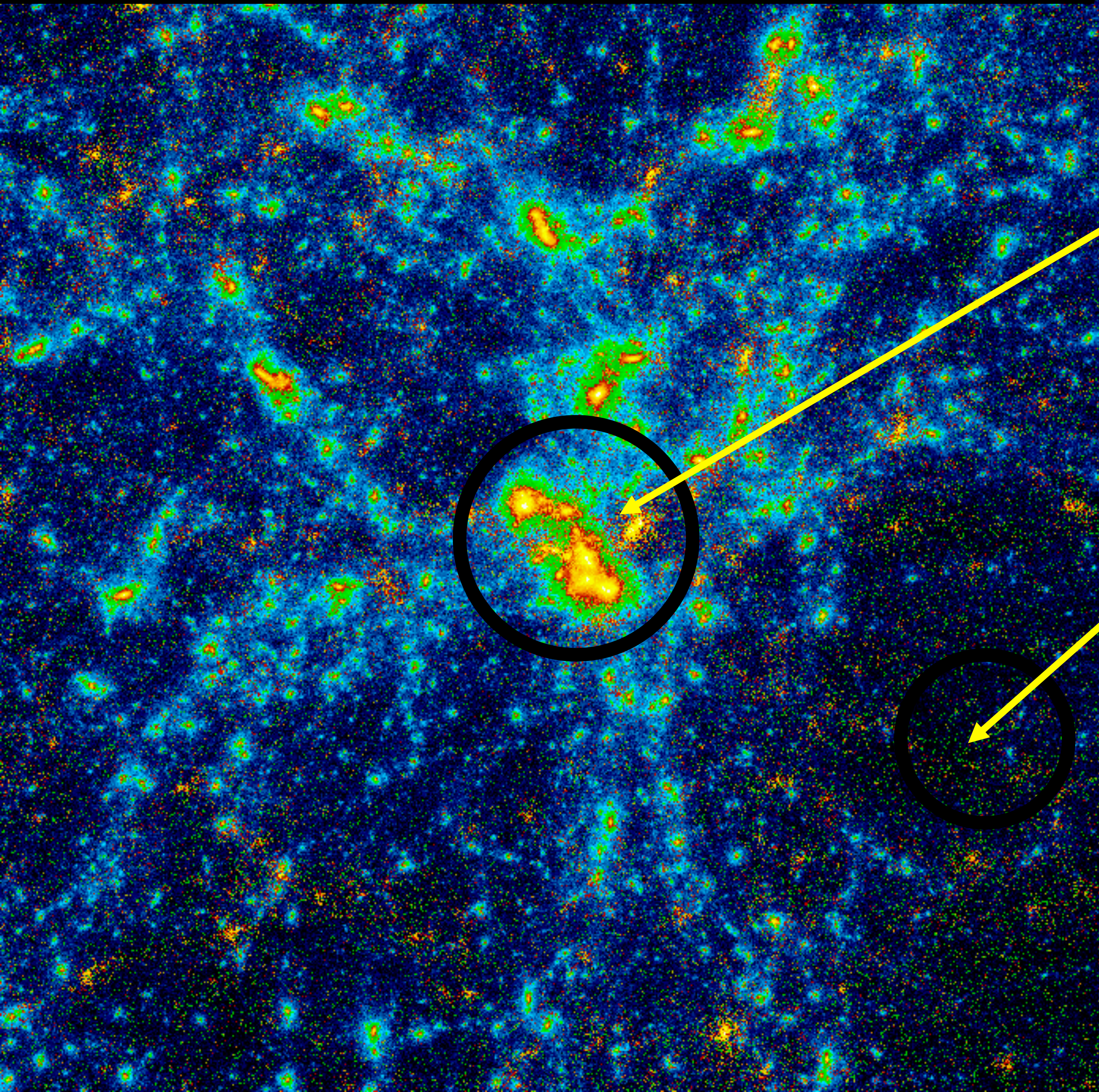
Irregular 2D Simulation -- Flow over an Airfoil



The Laplace grid points become finite element mesh nodal points arranged as triangles filling space

All the action (triangles) is near near wing boundary

Use domain decomposition but no longer equal area as equal triangle count



- Simulation of cosmological cluster (say 10 million stars)
- Lots of work per star as very close together (may need smaller time step)
- Little work per star as force changes slowly and can be well approximated by low order multipole expansion

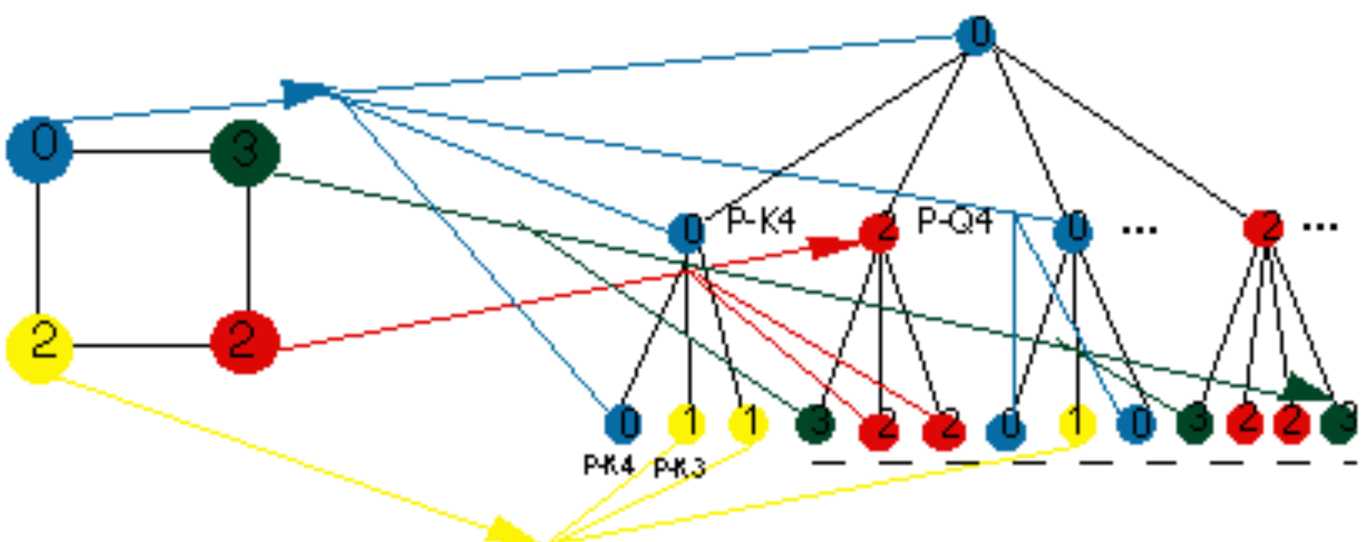
Further Decomposition Strategies

California gets its independence

Missile Tracking

- Not all decompositions are quite the same
- In defending against missile attacks, you track each missile on a separate node -- geometric again
- In playing chess, you decompose chess tree -- an abstract not geometric space

Computer **Chess Tree**



Current Position (node in Tree)

First Set Moves

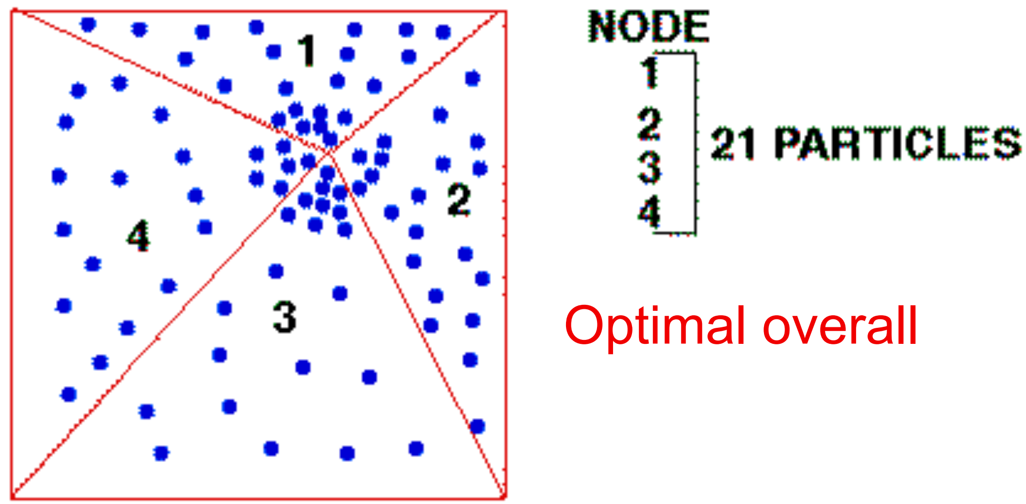
Opponents Counter Moves

Optimal v. stable scattered Decompositions

- Consider a set of locally interacting particles simulated on a 4 processor system

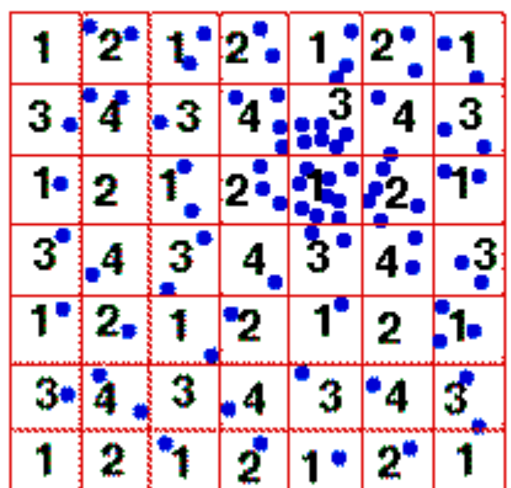
Block Scattered and Domain Decomposition Distributions Illustrated

DOMAIN DECOMPOSITION



Optimal overall

SCATTERED DECOMPOSITION

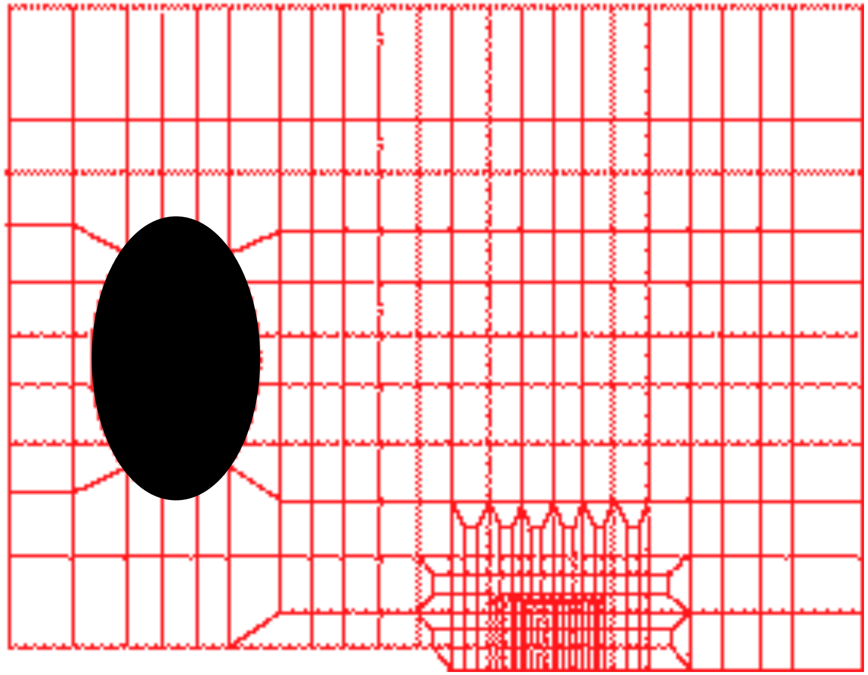


NODE	PARTILE COUNT
1	28
2	19
3	21
4	16

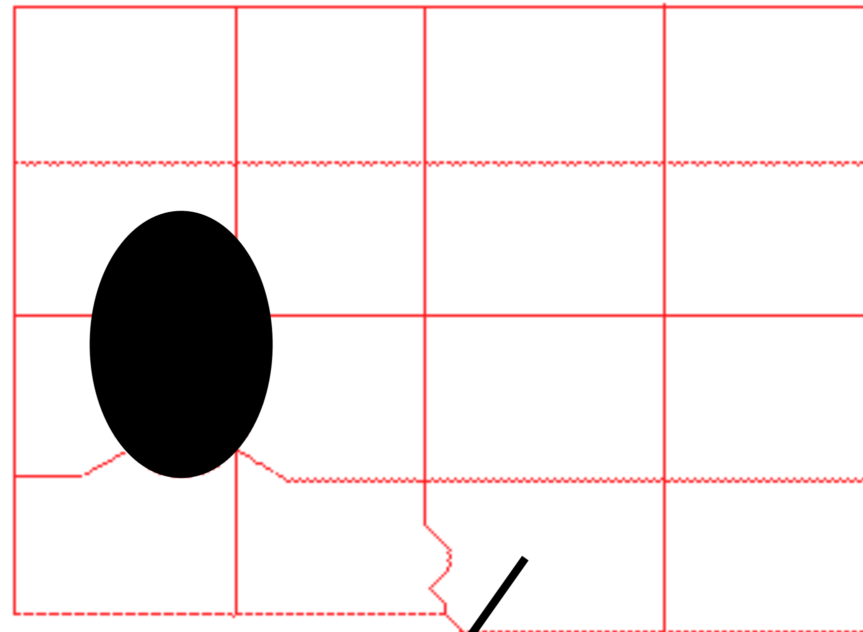
Parallel Computing III

- Note if I chop points up in Kmeans, then a given cluster is likely to need points in multiple cores
- As a user, I want search results from all web pages
- As an item, I want to be linked to all other items in eCommerce
- In parallel chess, I need to find best move whatever core discovered it and fancy optimal search algorithm (alpha-beta search) needs shared database and can have very different time complexity on each core
  - Need equal number of good moves on each core
- The things in red are decomposed across cores and communication between cores needed to either synchronize or send information
  - If cores on same CPU can use "shared memory"
- Fundamental problem in parallel computing is to efficiently coordinate cores and
  - Minimize synchronization and communication overhead
  - Make cores do roughly equal amount of work
- Famous cloud parallel computing approach called MapReduce invented by Google with sweetly named software called Hadoop from Yahoo
- Science used software called MPI – Message passing Interface

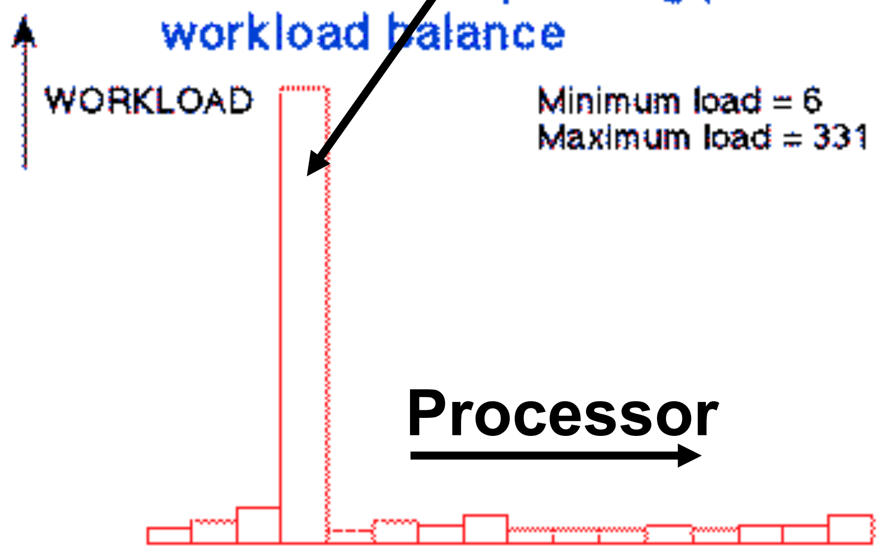
Parallel Irregular Finite Elements



- Here is a cracked plate and calculating stresses with an equal area decomposition leads to terrible results
  - All the work is near crack

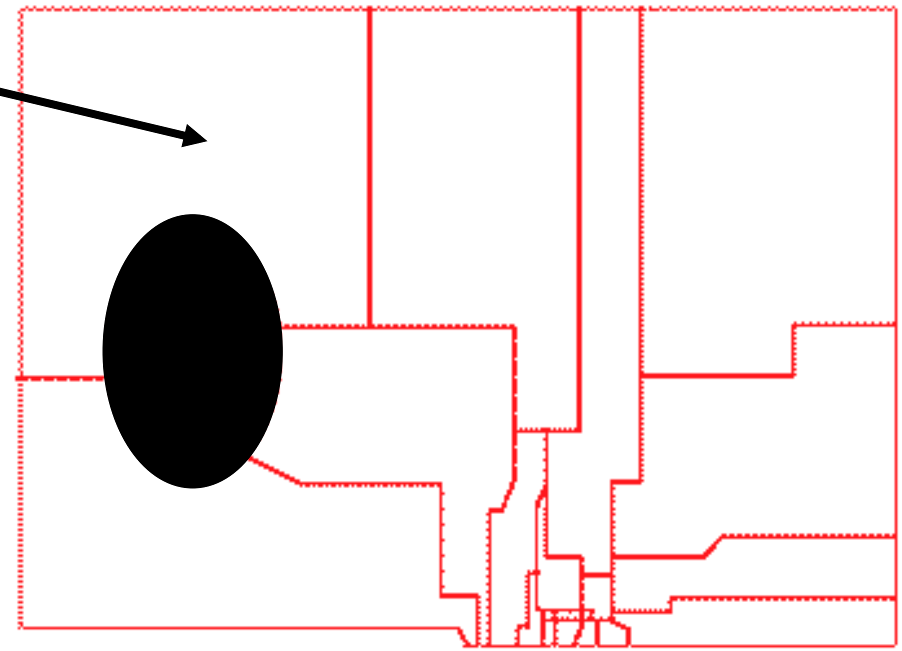


- And the corresponding poor workload balance



Irregular Decomposition for Crack

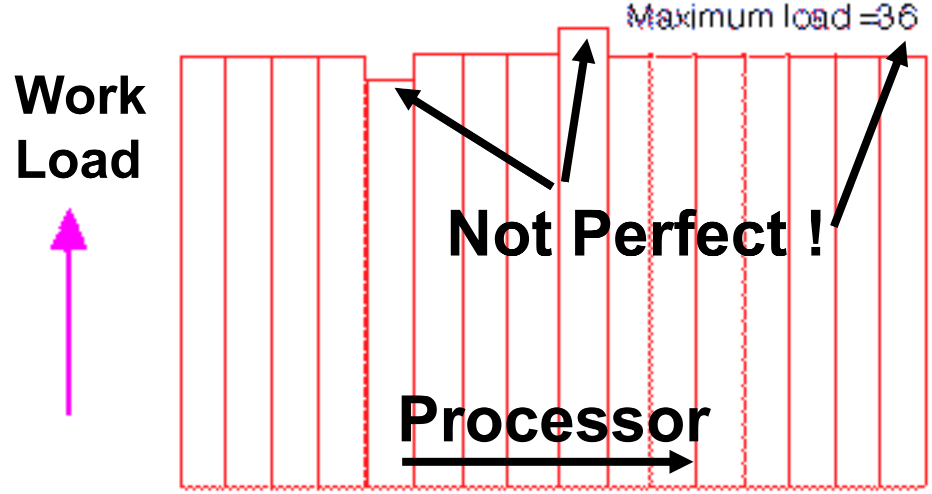
Region assigned to 1 processor



- Concentrating processors near crack leads to good workload balance
- equal nodal point -- not equal area -- but to minimize communication nodal points assigned to a particular processor are contiguous
- This is NP complete (exponenially hard) optimization problem but in practice many ways of getting good but not exact good decompositions

- And excellent workload balance

Minimum load = 32
Maximum load =36



Parallel Processing in Society

Parallel Processing in Society

It's all well known ……

Parallel Processing and Society

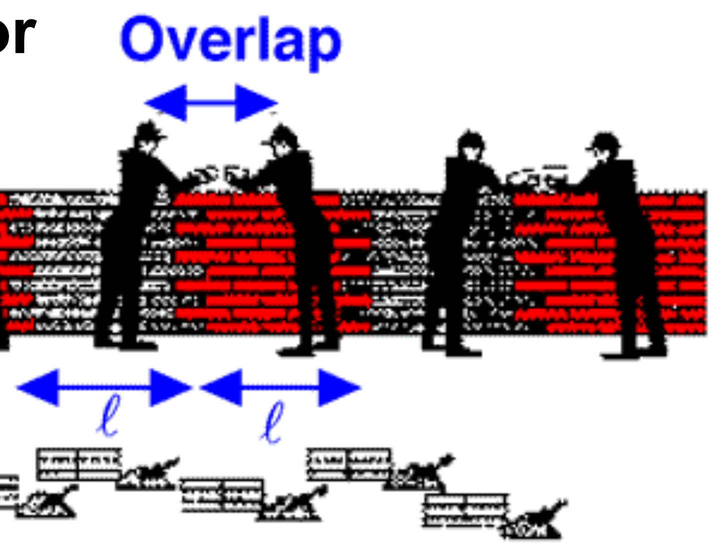
- The fundamental principles behind the use of concurrent computers are identical to those used in society - in fact they are partly why society exists.

- If a problem is too large for one person, one does *not* hire a *SUPERman*, but rather puts together a team of ordinary people...

- cf. Construction of Hadrians Wall

Concurrent Construction of a Wall Using N = 8 Bricklayers
Decomposition by Vertical Sections

8-person parallel processor



- Domain Decomposition is Key to Parallelism

  Need "Large" Subdomains $\ell \gg \ell_{overlap}$

Divide problem into parts; one part for each processor

Quantitative Speed-Up Analysis for Construction of Hadrian's Wall

- Quantitatively

S= Speed-up = $N\,\varepsilon$

efficiency

Number of Bricklayers

$\varepsilon \sim 1 - \text{constant}\ \dfrac{\ell_{overlap}}{\ell}$

ℓ = size (in metres) of wall assigned to each bricklayer

$\ell_{overlap}$ = overlap region

~ 6 metres in this case

Amdahl's Law of Parallel Processing

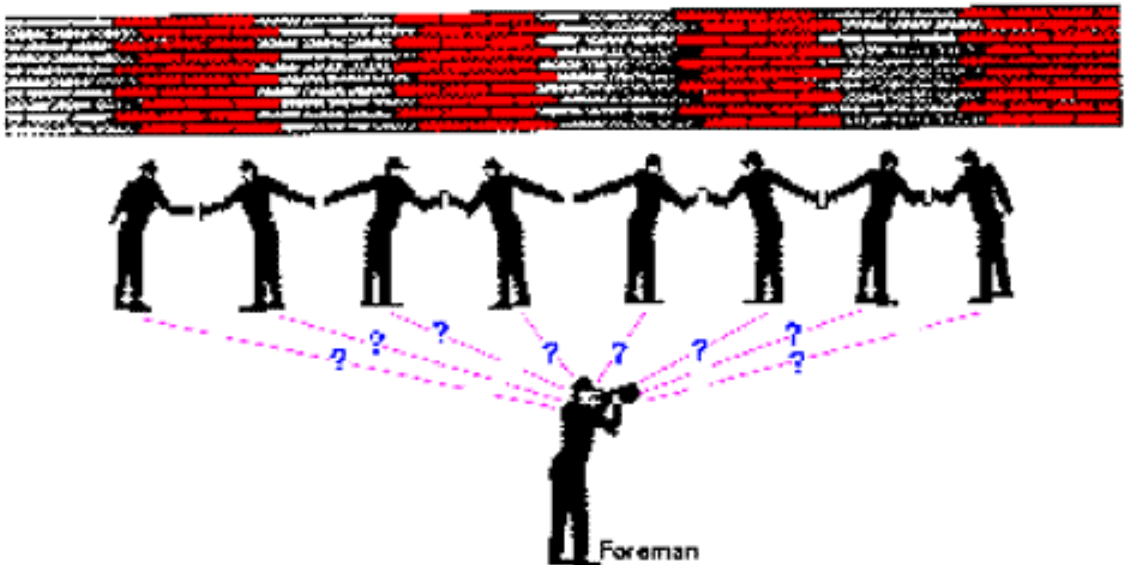
- Speedup S(N) is ratio Time(1 Processor)/Time(N Processors); we want $S(N) \geq 0.8\ N$
- Amdahl's law said no problem could get a speedup greater than about 10
- It is misleading as it was gotten by looking at small or non-parallelizable problems (such as existing software)
- For Hadrian's wall S(N) satisfies our goal as long as $l >$ about 60 meters *if* $l_{overlap}$ = about 6 meters
- If l is roughly same size as $l_{overlap}$ then we have "too many cooks spoil the broth syndrome"
  - One needs large problems to get good parallelism but only large problems need large scale parallelism

Pipelining --Another Parallel Processing Strategy for Hadrian's Wall

- ***"Pipelining" or decomposition by horizontal section is:***
  - In general less effective
  - and leads to less parallelism
  - (N = Number of bricklayers must be < number of layers of bricks)

Exploit Aspect of problem which gives largest parallelism

Hadrian's Wall Illustrates that the Topology of Processor Must Include Topology of Problem



- Hadrian's Wall is one dimensional
- Humans represent a flexible processor node that can be arranged in different ways for different problems
- The lesson for computing is:
  Original MIMD machines used a hypercube topology. The hypercube includes several topologies including all meshes. It is a flexible concurrent computer that can tackle a broad range of problems. Current machines use different interconnect structure from hypercube but preserve this capability.

General Speed Up Analysis

- Comparing Computer and Hadrian's Wall Cases

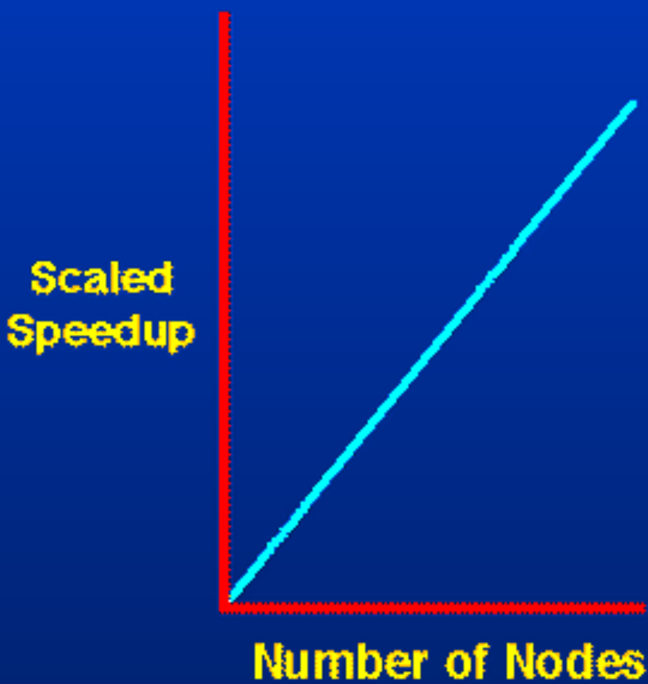
$$\text{Speedup } S = \varepsilon N$$

$$\varepsilon = 1 - \frac{\text{constant}}{n^{1/d}} \cdot \frac{t_{comm}}{t_{calc}}$$

General	*Hadrian's Wall*
n = Grain Size	n = number of bricks laid by each mason $n \propto l$
d = Problem Dimension	d = 1 for a one dimensional wall (d = 2 for laying tiles on floor of Hadrian's Palace)
t_{calc} = Time to do each calculation	Time to lay one brick
t_{comm} = Time to communicate unit of information between nodes	Time to discuss/adjust brick laid at join between domains assigned adjacent masons

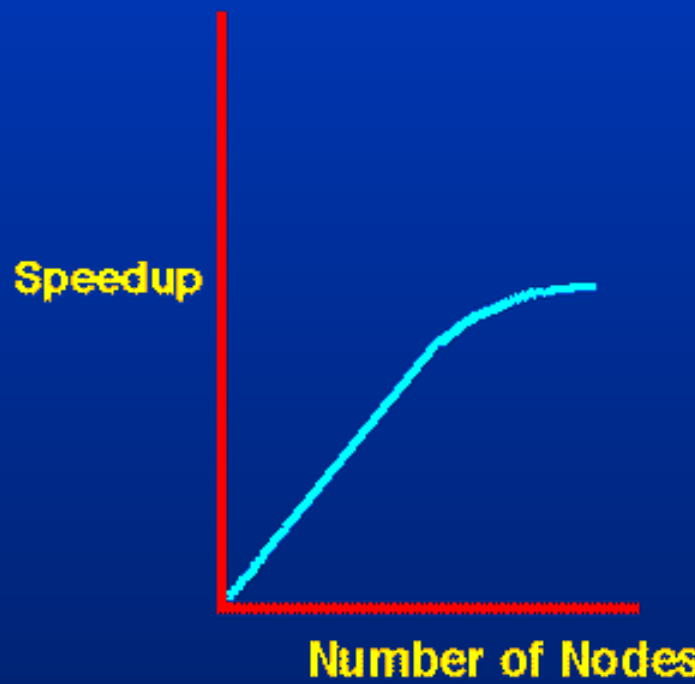
Speed Up as a Function of Grain Size

- (a) Fixed Grain Size n — (b) Fixed Problem Size



Amount of problem in each node fixed.

Problem size α N



n gets small; communication and control dominate.

Grain size $n \; \alpha$ 1/N

Nature's Concurrent Computers

- At the finest resolution, collection of neurons sending and receiving messages by axons and dendrites
- At a coarser resolution
  Society is a collection of brains sending and receiving messages by sight and sound
- Ant Hill is a collection of ants (smaller brains) sending and receiving messages by chemical signals
- Lesson: All Nature's Computers Use Message Passing
- With several different Architectures

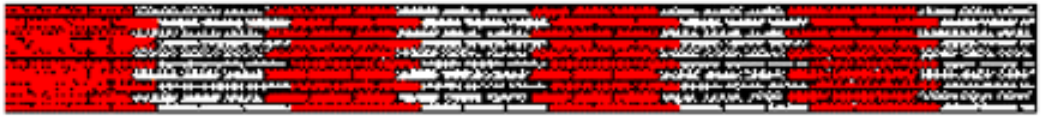
Neural Network

The Web is also just message passing

Parallel Processing for Hadrian's Wall

Comparison of The Complete Problem to the subproblems formed in domain decomposition

For Hadrian's Wall. the complete problem:



is similar to the subtask performed by an individual bricklayer

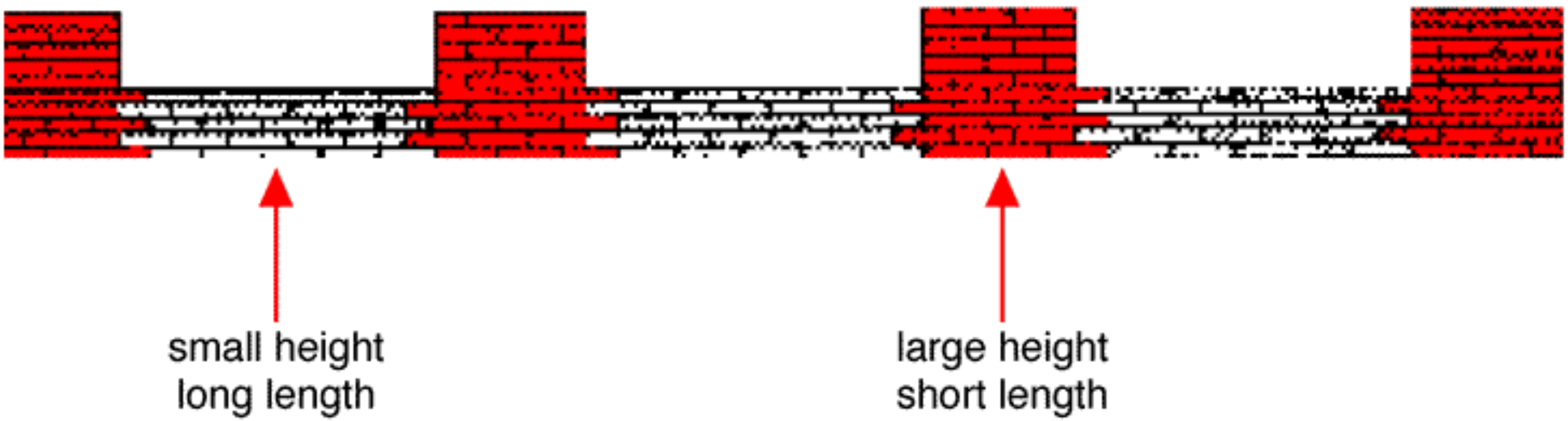
Changed is
- Geometry
- Boundary Conditions

- *The case of Programming a Hypercube*
  - Each node runs software that is similar to sequential code
  - e.g., FORTRAN with geometry and boundary value sections changed

1984 Slide – today replace hypercube by cluster

Hadrian's Wall Illustrating an Irregular but Homogeneous Problem

- Geometry irregular but each brick takes about the same amount of time to lay.
- Decomposition of wall for an irregular geometry involves equalizing number of bricks per mason, not length of wall per mason.



Equal work not
Equal area of underlying domain
is load balancing requirement

Some Problems are Inhomogeneous Illustrated by:

An Inhomogeneous Hadrian Wall with Decoration

- Fundamental entities (bricks, gargoyles) are of different complexity
- Best decomposition dynamic

- Inhomogeneous problems run on concurrent computers but require dynamic assignment of work to nodes and strategies to optimize this
- (we use neural networks, simulated annealing, spectral bisection etc.)

Global and Local Parallelism Illustrated by Hadrian's Wall

- **Global Parallelism**
  - Break up domain
  - Amount of Parallelism proportional to size of problem (and is usually large)
  - Unit is Bricklayer or Computer node

Between CPU's
Called Outer Parallelism

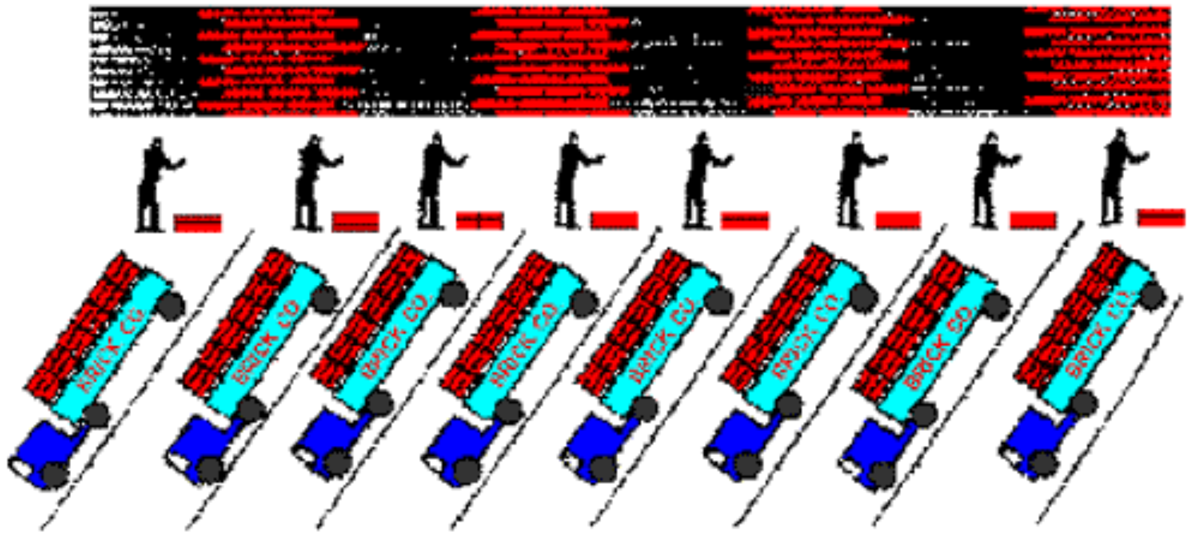
- **Local Parallelism**

Inside CPU or Inner Parallelism

  - Do in parallel local operations in the processing of basic entities
    - e.g. for Hadrian's problem, use two hands, one for brick and one for mortar while ...
    - for computer case, do addition at same time as multiplication
  - Local Parallelism is limited but useful
- **Local and Global Parallelism Should both be Exploited**

Parallel I/O Illustrated by Concurrent Brick Delivery for Hadrian's Wall

Bandwidth of Trucks and Roads Matches that of Masons



- Disk (input/output) Technology is better matched to several modest power processors than to a single sequential supercomputer

Key to MapReduce

- Concurrent Computers natural in databases, transaction analysis

Comparison of Concurrent Processing in Society and Computing

- Problems are large - use domain decomposition
  Overheads are edge effects
- Topology of processor matches that of domain - processor with rich flexible node/topology matches most domains
- Regular homogeneous problems easiest but
  irregular or
  Inhomogeneous
  } work with proper decomposition/planning
- Can use local and global parallelism
- Can handle concurrent calculation and I/O
- Nature always uses message passing as in parallel computers (at lowest level)