

Making Programming the Fourth 'R of Literacy

Arun Chauhan, Indiana University

I399

School of Informatics and Computing
Indiana University

Mar 29, 2010



“A New Kind of Science”

Stephen Wolfram

“Computing is as fundamental as the physical, life, and social sciences.”

Peter J. Denning and Paul S. Rosenbloom
Communications of the ACM, Sep 2009

“What our community should really aim for is the development of a curriculum that turns our subject into the fourth R—as in ‘rogramming—of our education systems.

...

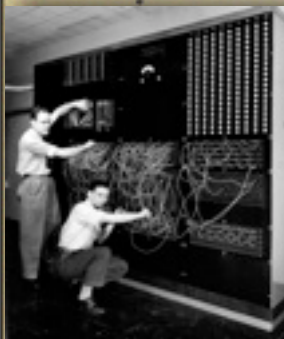
A form of mathematics can be used as a full-fledged programming language, just like Turing Machines.”

Matthias Felleisen and Shriram Krishnamurthy
Communications of the ACM, Jul 2009

Programming as the Fourth ‘R of Literacy, Arun Chauhan, I399, 2010-03-29



Programming



Programming



“Why can't you be like the Math Department, which only needs a blackboard and wastepaper basket? Better still, like the Department of Philosophy. That doesn't even need a wastepaper basket ...”

Arthur C. Clarke
3001: The Final Odyssey

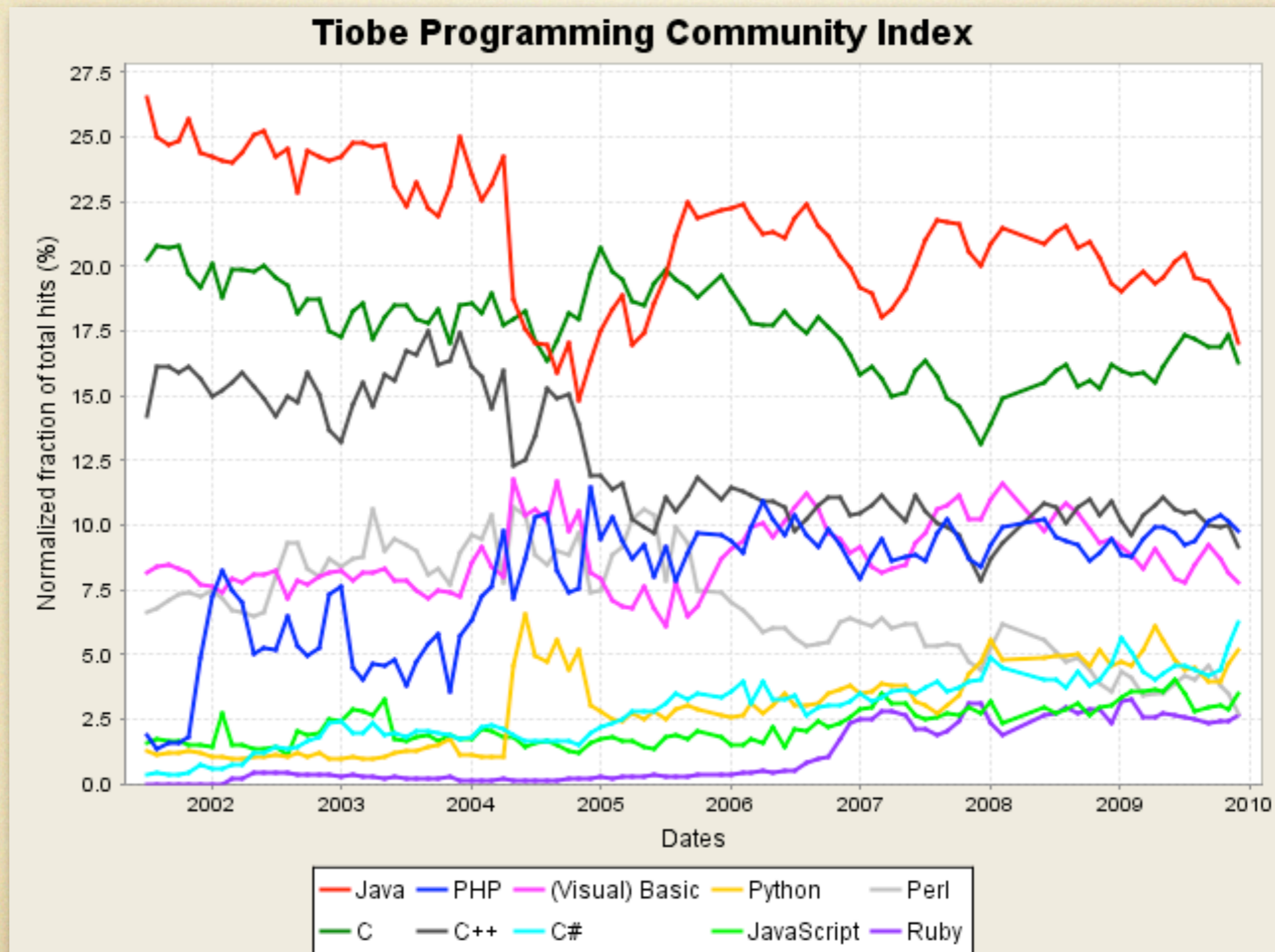


Computers are for Computing and ...

- Computers as general-purpose tools
 - communication, navigation, data collection, entertainment, etc.
- Computers as computing tools
 - problem solving
 - data processing and analysis



TIOBE Index



TIOBE: Top 20

Position Dec 2009	Position Dec 2008	Delta in Position	Programming Language	Ratings Dec 2009	Delta Dec 2008	Status
1	1	=	Java	17.061%	-2.31%	A
2	2	=	C	16.285%	+0.12%	A
3	4	↑	PHP	9.770%	+0.29%	A
4	3	↓	C++	9.175%	-1.72%	A
5	5	=	(Visual) Basic	7.778%	-1.70%	A
6	6	=	C#	6.258%	+1.61%	A
7	7	=	Python	5.185%	+0.62%	A
8	9	↑	JavaScript	3.515%	+0.45%	A
9	8	↓	Perl	2.692%	-0.91%	A
10	11	↑	Ruby	2.653%	+0.34%	A
11	10	↓	Delphi	2.301%	-0.75%	A
12	13	↑	PL/SQL	1.494%	+0.35%	A
13	35	↑↑↑↑↑↑↑↑↑↑	Objective-C	1.159%	+1.00%	A
14	14	=	SAS	0.911%	+0.07%	A
15	19	↑↑↑↑	Lisp/Scheme	0.881%	+0.37%	A--
16	17	↑	ABAP	0.723%	+0.12%	A-
17	15	↓↓	Pascal	0.698%	+0.01%	B
18	21	↑↑↑	ActionScript	0.655%	+0.17%	B
19	12	↓↓↓↓↓	D	0.587%	-0.60%	B
20	20	=	Lua	0.585%	+0.09%	B



Teaching Programming

Table 1.


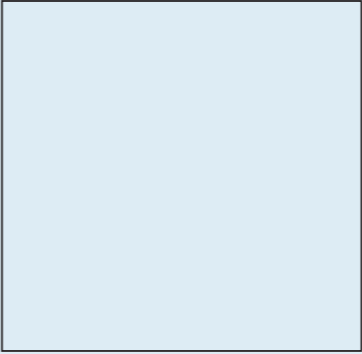
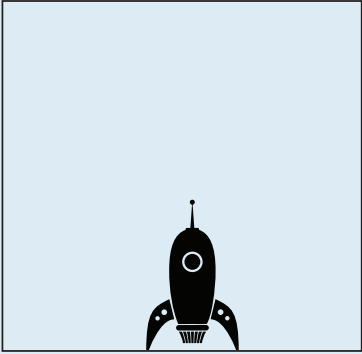
1	2	3	4	5	...	x
1	4	9	16	?	...	?

Teaching Programming

Figure 1.

`placeImage ( , 25, 0, )`

Figure 2.

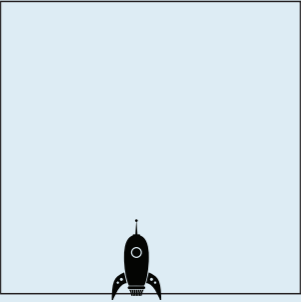
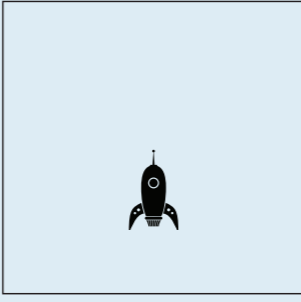
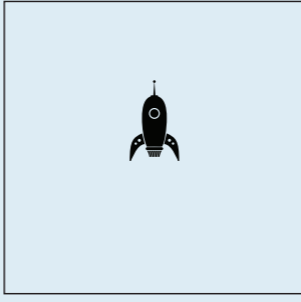
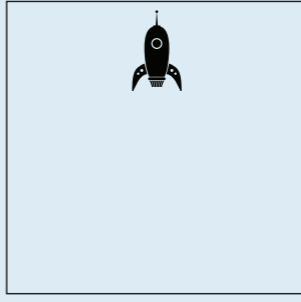
`placeImage ( , 25, 0, ) = `

Teaching Programming

Table 2.

0	1	2	3	...	t
0	10	20	30	...	$height(t) = ?$

Table 3.

0	1	2	3	...	t
				...	$rocket(t) = ?$

Teaching Programming

Figure 3.

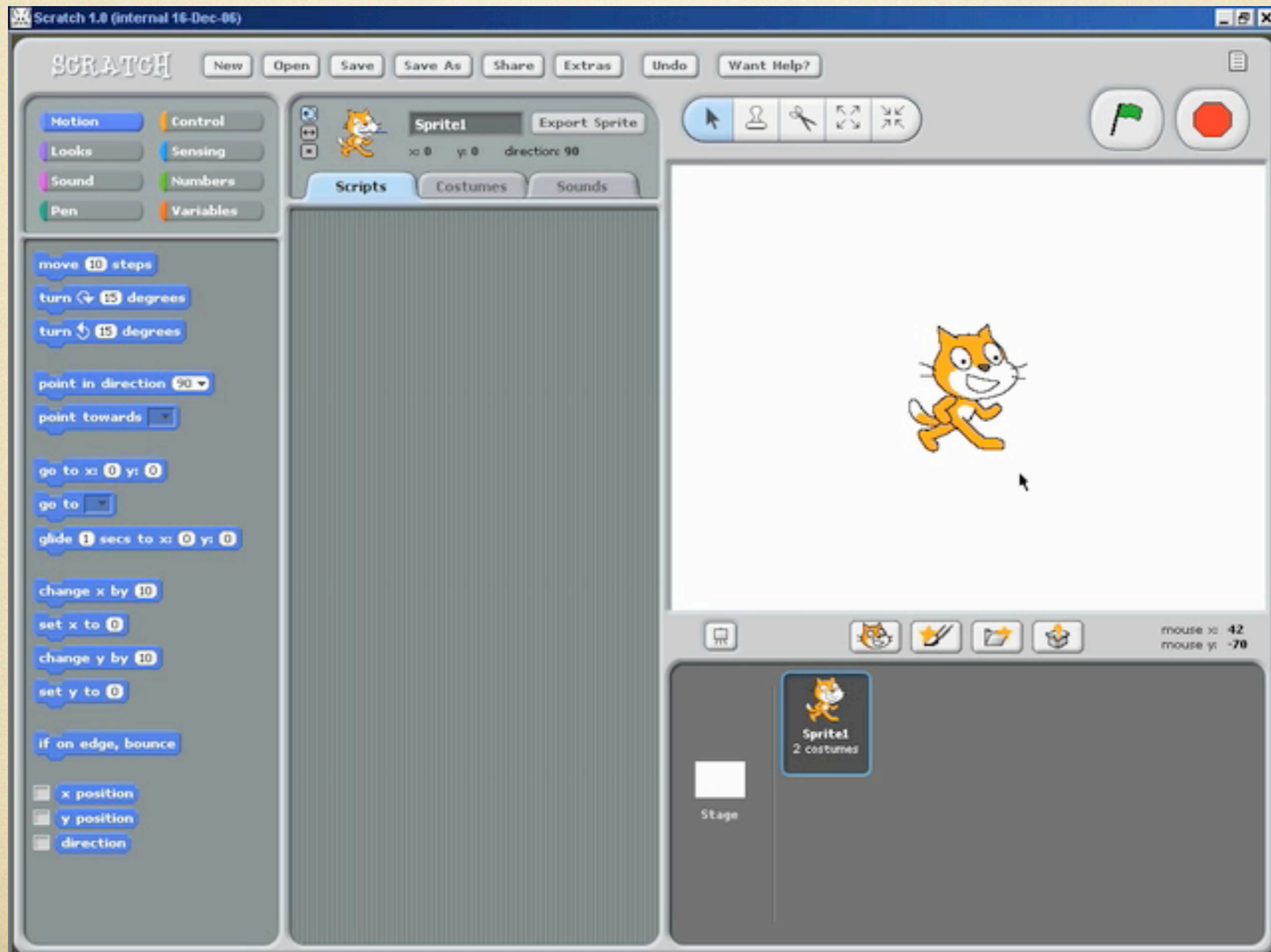
$rocket(t) = placeImage (\text{rocket} , 25, 10 \cdot t, \text{canvas})$

Figure 4.

$rocket(t) = placeImage (\text{rocket} , 25, height(t), \text{canvas})$

Scratch

<http://scratch.mit.edu/>



Problem

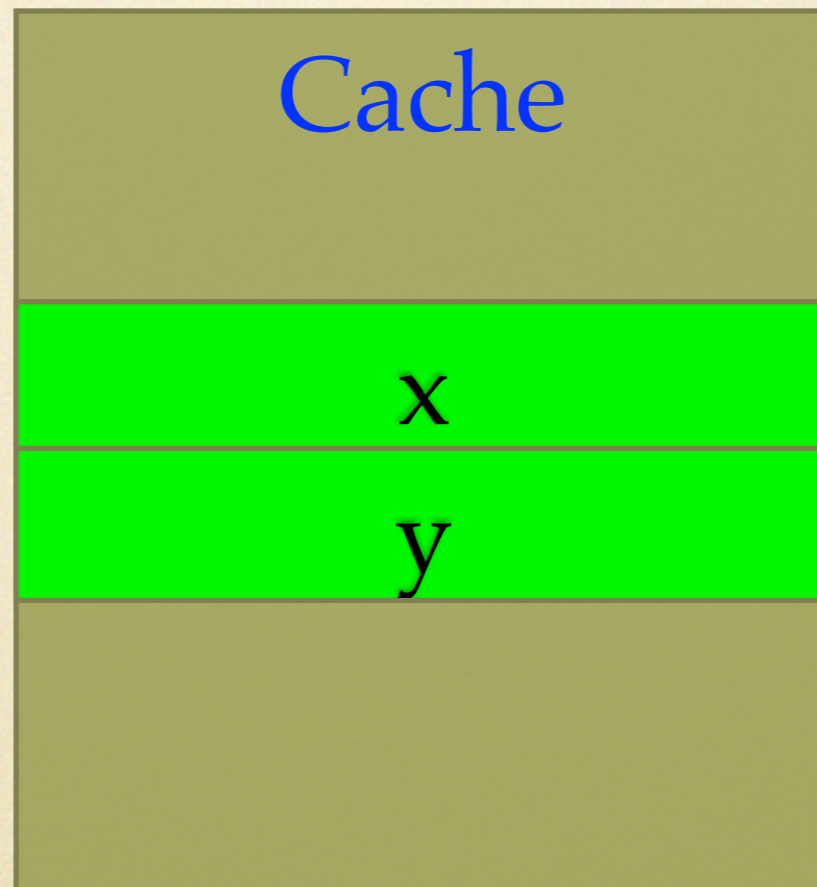
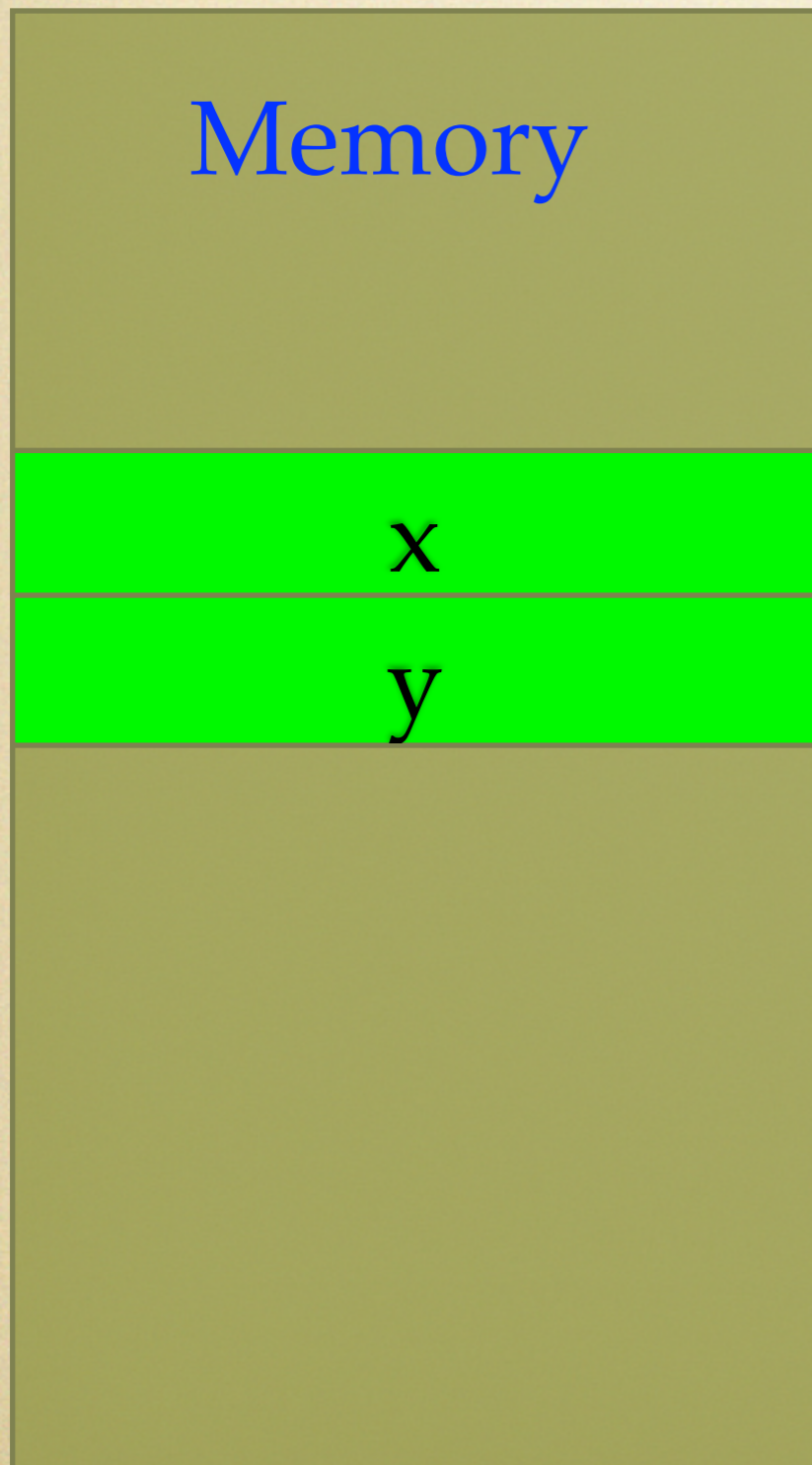
- Nice programming languages
 - domain-specific
 - often dynamically typed and interpreted
- Poor performance
 - inefficient use of computing resources
 - inefficient use of energy



Challenge #1:
There is no mathematical
model for **data locality**



One Slide Primer on Locality



`x = 10;`

`...`

`y = x + 2;`

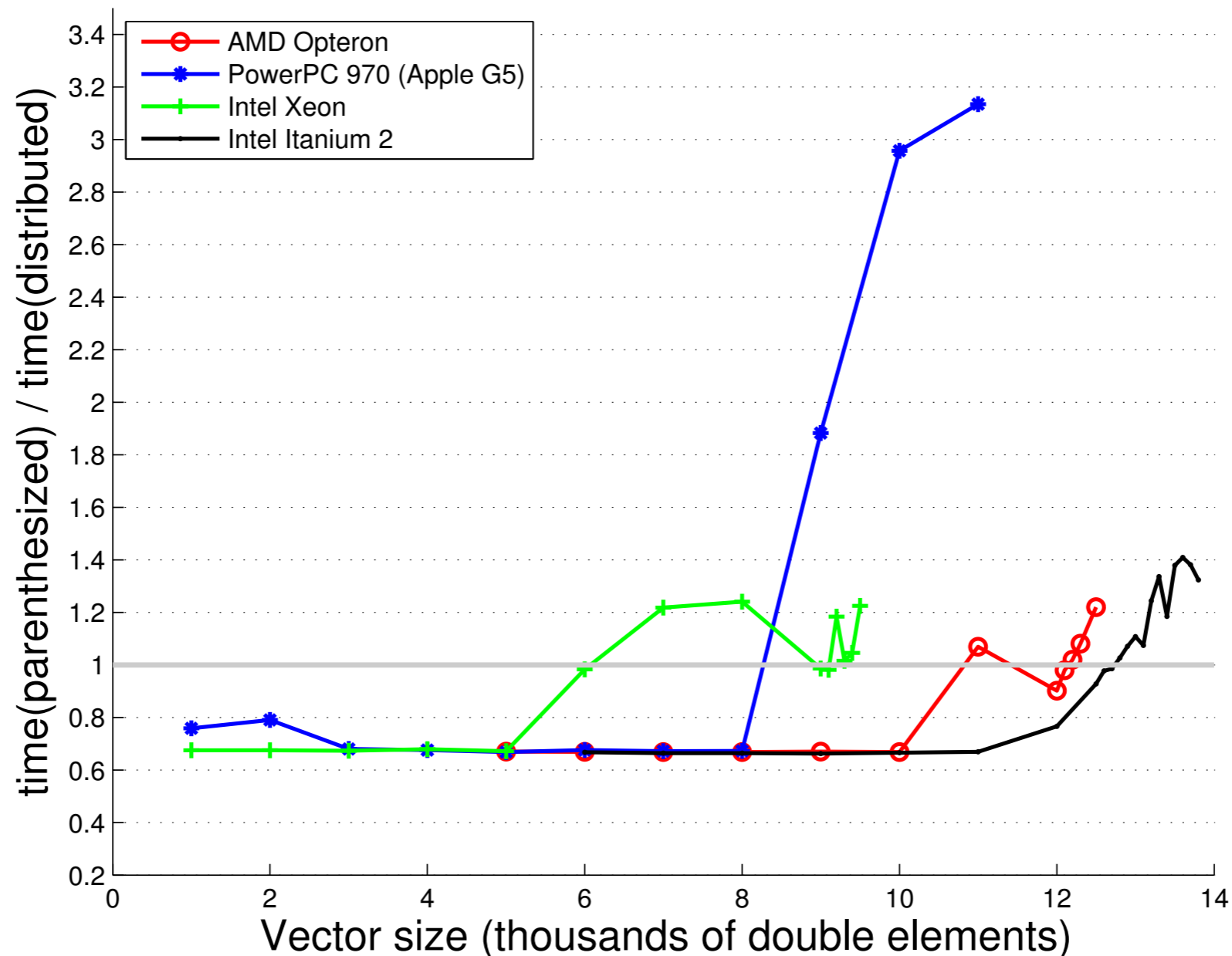
Temporal locality

Spatial locality



An Empirical Study

Implementing A Big Expression



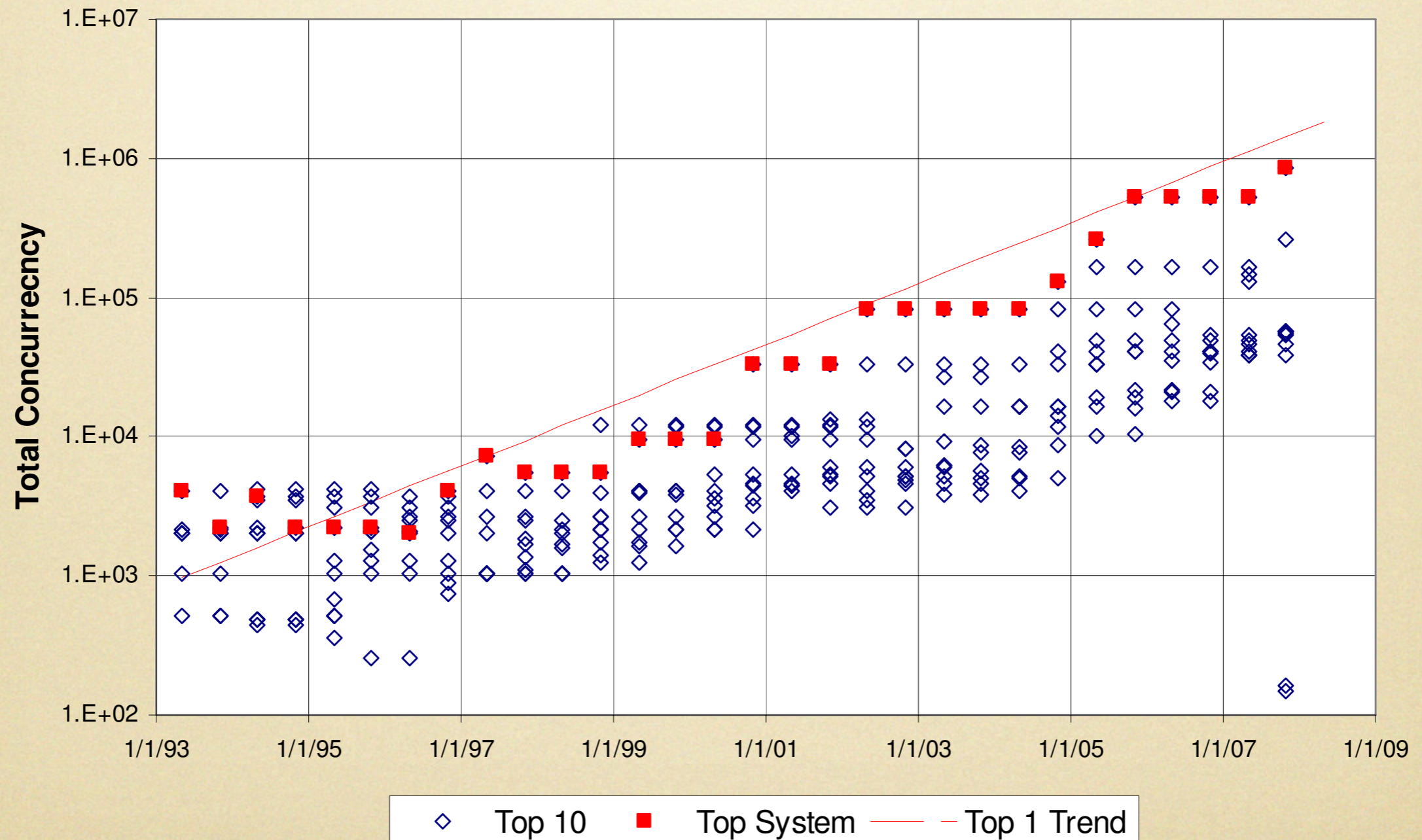
Reuse Distances

```
x = a + b;  
c = a + d[i]*100;  
y = x * 10;
```

Reuse Distance = 6 (a, b, c, d, i, 100)

Concurrency Trends

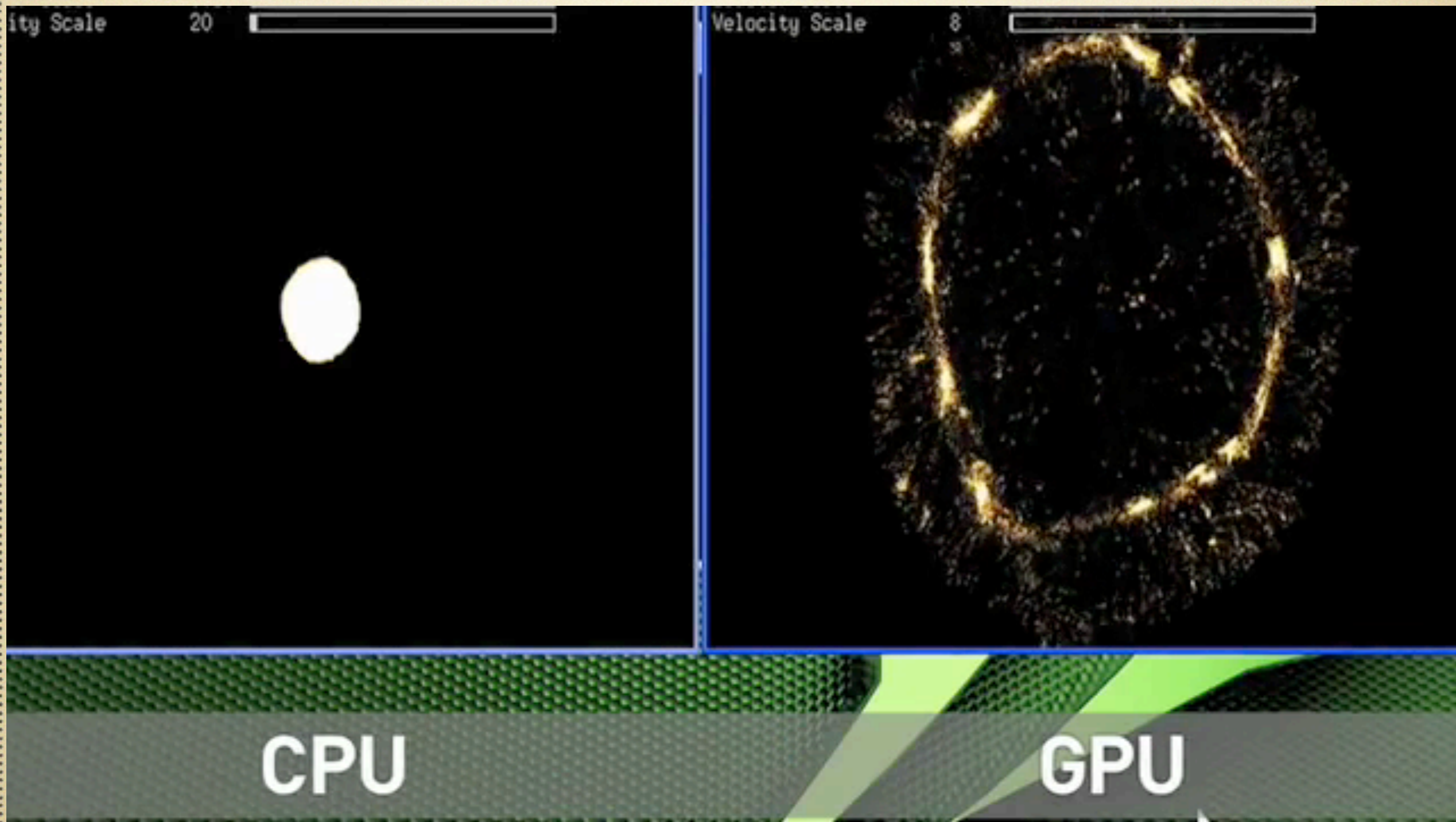
(ExaScale Computing Study, Peter Kogge et al.)



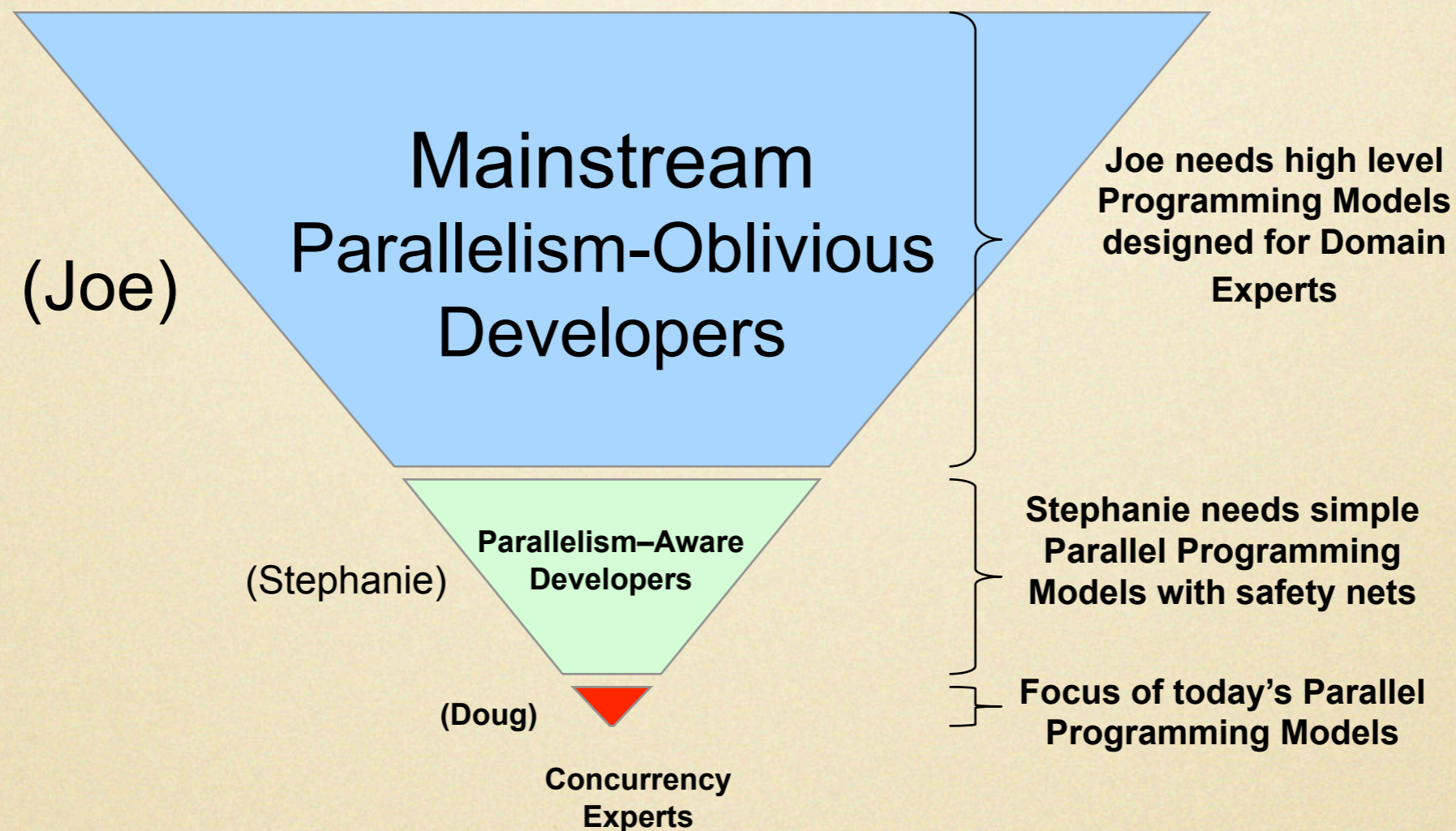
Challenge #2:
There is no easy way to write
parallel programs



Parallelism is Useful!



Types of (Parallel) Programmers



Courtesy: Vivek Sarkar, Rice University



One Slide Primer on Parallelism

Shared
Memory

x

Synchronize

```
x = 10;
```

```
...
```

```
y = x + 2;
```

```
x = 20;
```

```
...
```

```
y = x + 2;
```

Distributed Memory

x

Pass messages

x

Parallelism Oblivious Users

- Programming languages-driven
 - implicit parallelism, compiler support
- Operating System-driven
 - innovative solutions to leverage extra cores
- Architecture-driven
 - Instruction-level parallelism, hyper-threading



Observations for Parallelism-Aware and Expert Users

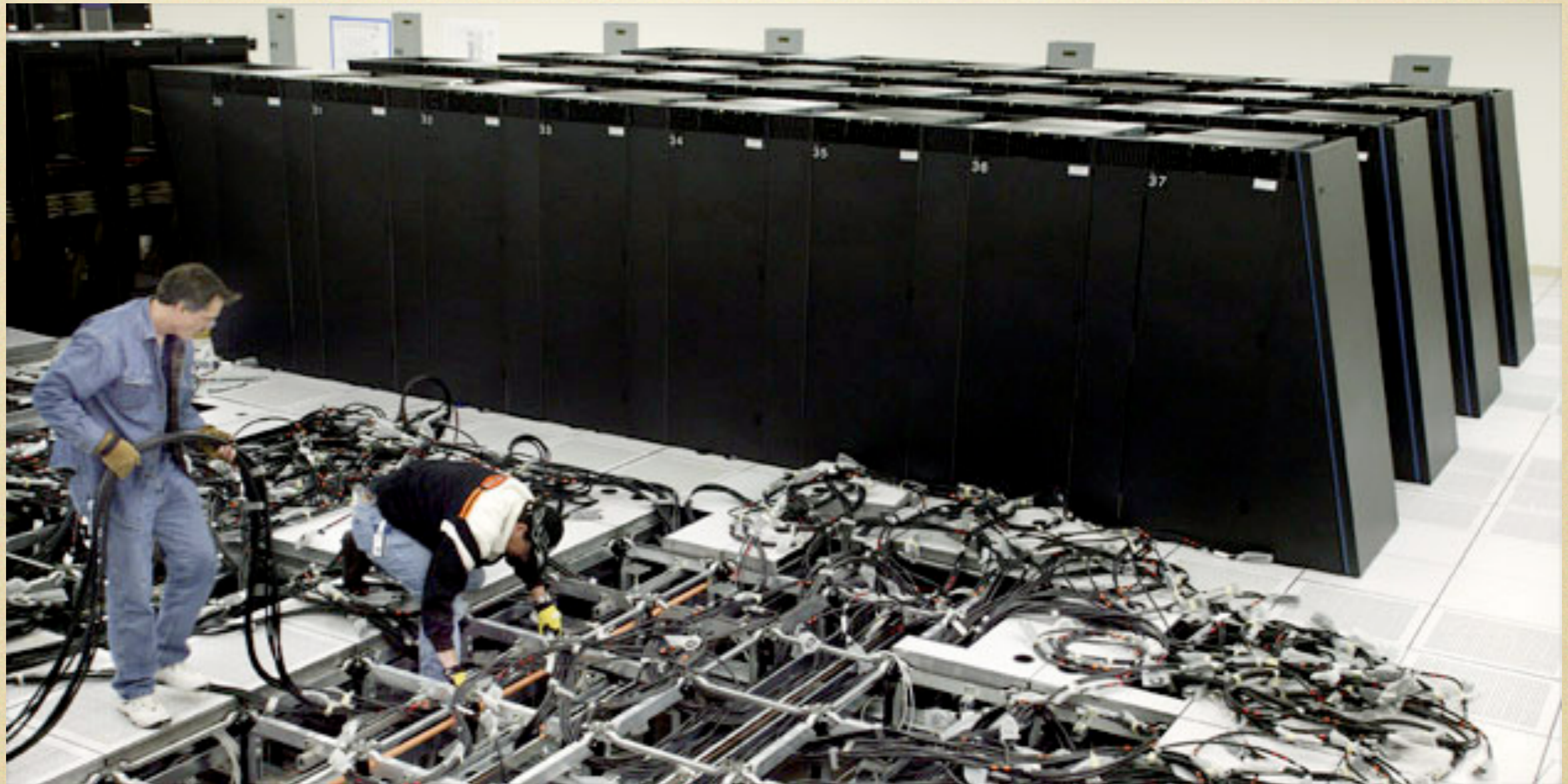
- Completely automatic parallelization has had limited success
- Writing parallel programs is hard; optimizing and maintaining them is harder!
- Compilation technology has worked well in communication optimization



Concluding Remarks

- Educating the next generation for the fourth 'R'
 - Computing is a core technique in an increasing number of fields
 - programming is no longer restricted to scientists and engineers
- Taking care of non-expert programmers
 - an exponentially growing class
 - locality and parallelism problems
- Solving problems for expert programmers
 - tools to address computational bottlenecks

Toward Exascale (10^{18})



What Should You Do?

- Educate yourself in the basics
 - computer architecture
 - programming languages
 - compilers
- Learn parallel programming!



Research Interests

- High-level Languages
 - Ruby, MATLAB, R, etc.
- Heterogeneous parallel computing
- Large memory-footprint applications
- Automatic parallelization

<http://www.cs.indiana.edu/~achauhan>

<http://phi.cs.indiana.edu/>

