CS 475 GRAD 510
Fall 2019

Sanjay Rajopadhye
Colorado State University

Name Tag Rules

Name in big bold letters on outside, other info on inside

- Name: Sanjay Rajopadhye
- What you want to be called: Sanjay
- Pronunciation* (optional): Sun-juy Raaj-Oh-path-yay (in "path" make the t sound like a d). Don't worry if you don't get it right, it's almost always mispronounced, even in India).
- Major dept: CS/ECE
- Status: Professor (e.g., 3rd year Ph.D.)
- Interesting fact: Last year, Ft Collins became the city that I've lived in for the longest time [before that it was Kharagpur, India]
- Motivation: Why are you taking this class and what you want to get out of it

* In Indian names, "a" is almost always pronounced as a short "u" sound as in gun, fun, etc., or a long "aa" sound as in calm, bard, etc.
These rules are used in many parts of Asia, e.g., pronounce "Bagdad?"

About me

- Education
  - Undergrad: B.Tech, IIT Kharagpur (India) 1980
  - Graduate: PhD University of Utah (running Utes) 1986
- Professional trajectory:
  - University of Oregon (fighting ducks) 86-91
  - Oregon State University (beavers) 91-92
  - IRISA/University of Rennes, France (???) 92-01
  - Colorado State University (rams) since 2001
- Research Interests/contributions:
  - HPC, Accelerators, FPGAs, compilers, programing languages
  - Polyhedral model: a mathematical framework for describing, transforming and "compiling" massively parallel computations

Colorado State University 3

GTA

- Name: Fatemeh Hashemi Chaleshtori
- What you want to be called: Fatemeh
- Major dept: CS
- Status: First year PhD
- Interesting fact: Traveled to Japan (Nagoya) for robotic competition in 2017

Colorado State University 4

GTA

- Name: Brandon Gildemaster
- What you want to be called: Brandon
- Major dept: CS
- Status: Second year MS
- Interesting fact: this summer I went on a road trip from Fort Collins to Kansas then up to South Dakota and finally out to Boston visiting some friends and family along the way

Outline for today's lecture

- Let's play a game
  - Pose (and answer) questions and pose more
- Why parallelism
- Overview of speedup and efficiency
- Plan for the class
- Background quiz
  - What do you already know (so I know what to review)
  - Make sure clickers work

Card Game Rules

- Basic step repeat ad nauseum (while true do):
  - take a card from the left
  - compare with card in your right hand
    - keep largest
    - in the beginning keep the one you get
  - give a card to the right
- Protocol
  - Only right hand used for communication
  - Keep left hand card private (close to the chest)
- One team leader/conductor
- Ace is highest

Colorado State University 7

Card Game Rules (2)

- Organize team into a chain
- Conductor feeds the first player (from their left)
  - From shuffled deck of cards
- Last player places output card face down and stacks them up
- Play for 5 minutes
- What is coming out from the last player?

Colorado State University 8

Questions (easy & hard)

- What does this "distributed" algorithm do?
- Is the algorithm "synchronous" or "asynchronous"
  - Yes / No /Maybe
- How can we count the number of "steps" in order to analyze it?
  - Sequential steps / parallel steps
- When does it start
  - When first processor starts or when all processors are working
- When does it finish?

Colorado State University 9

More Questions

- How many processors (p) needed to process a sequence of (n) inputs?
- How many steps does this take?
- How to get all the outputs out (only the last processor can do this)
- How to use it when $n > p$
- Can we discuss speedup/efficiency of this?

Colorado State University 10

Outline for today's lecture

- Let's play a game
  - Pose (and answer) questions and pose more
- Why parallelism
- Overview of speedup and efficiency
- Plan for the class
- Background quiz
  - What do you already know (so I know what to review)
  - Make sure clickers work

Why Parallel Programming

Need for **speed**

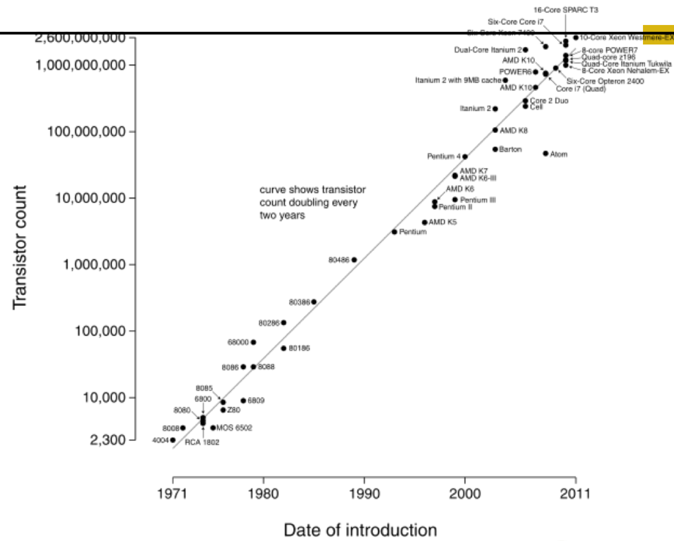
- Applications require orders of magnitude more compute power than we have right now. Speed came for a long time from technology improvements, mainly increased clock speed and chip density.
- Technology improvements have slowed down, and the only way to get more speed is to exploit **parallelism**. All computers are now parallel computers.

Moore's Law

- Empirical observation (1965) by Gordon Moore:
  - Chip density: #transistors that can be inexpensively placed on an integrated circuit, is increasing exponentially, doubling approximately every two years.
- http://en.wikipedia.org/wiki/Moore's_law
- Held true until now and is expected to hold until at least 2025.

Colorado State University 13

Microprocessor Transistor Counts 1971-2011 & Moore's Law



Colorado State University 14

Exponential growth Consequences

- If two quantities grow exponentially at different rates, what happens to their ratio?

$$y_1 = a^x \text{, and } y_2 = b^x \text{ for } a \geq b \geq 1, \textit{therefore} \left(\frac{a}{b}\right)^x = r^x \text{ , r} \geq 1$$

- $1.1^n = O(2^n)$ But $2^n \neq O(1.1^n)$
- For computer systems: different growth rates lead to walls (e.g., memory wall, power wall)

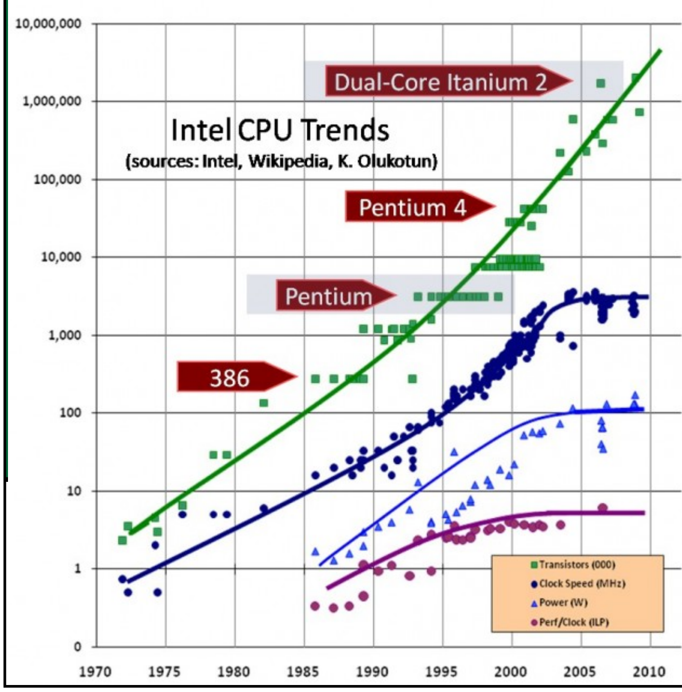
Moore's Law gaps

- Memory gap/wall:

  Memory bandwidth and latency improve much slower than processor speeds (since mid 80s, this was addressed by ever increasing on-chip caches)

- (Brick) Power Wall
  Higher frequency means more power, i.e., more heat. So chips are getting exceedingly hot

  Consequence: We can no longer increase the clock frequency of our processors.



source:
The Free Lunch Is Over Her
Dr. Dobbs, 2005

e University

Solution: Parallelism (2005)

- The goal is to deliver performance. The only solution to the power wall is to have multiple processor cores on a chip, because we can still increase the chip density.
- Nowadays, there are no more processors with one core.
- "The processor is the new transistor."
  - 2005 prediction: Number of cores will continue to grow at an exponential rate (at least until 2015)
  - Has this happened ?

Colorado State University 18

Outline for today's lecture

- Let's play a game
  - Pose (and answer) questions and pose more
- Why parallelism
- Overview of speedup and efficiency
- Plan for the class
- Background quiz
  - What do you already know (so I know what to review)
  - Make sure clickers work

Colorado State University 19

Recap this week's reading assignment

- Start on Ch 17 before tomorrow's lab
- Inside the front cover
- Chapter 3: Parallel Algorithm Design

Colorado State University 20

Speedup

- Again, why do we write parallel programs?
  - Go faster than sequential program
- What is speedup?

  T_1 = sequential time to execute a program
  sometimes called T, or S

  T_p = time to execute the same program

with

p processors (or cores, PEs)

$S_p = T_1 / T_p$ speedup for p processors

Colorado State University 21

Ideal/Linear Speedup

- Ideal speedup: p fold speedup: $S_p = p$
- Ideal not always possible. WHY?
  - Certain parts of the computation are inherently sequential
  - Tasks are data dependent, so not all processors are always busy, and need to synchronize
  - Remote data needs communication
    - Memory wall PLUS Communication wall
- Linear speedup: $S_p = \beta p$

  β is usually less than 1

Colorado State University 22

Efficiency

- Speedup is usually not ideal, nor linear
- We express this in terms of efficiency E_p:
  $E_p = S_p / p$
  - E_p defines the average utilization of p processors
  - Range?
- What does $E_p = 1$ signify?
- What does $E_p = \beta$ (for $0<\beta<1$) signify?

Colorado State University 23

More careful analysis

- $T_1 = 1, \quad T_p = \sigma + (o+\pi)/p$
- $S_p = 1 / (\sigma + (o+\pi)/p)$

  σ is sequential fraction of the program

  π is parallel fraction of the program

  $\sigma+\pi = 1$

  o is parallel overhead (does not occur in sequential execution)

Draw speedup curves for

$p=1, 2, 4, 8, 16, 32 \quad \sigma= ¼ , ½ \quad o= ¼ , ½$

When p goes to ∞, S_p goes to?

Colorado State University 24

More careful analysis

- $T_1 = 1$, $T_p = \sigma + (o+\pi)/p$
- $S_p = 1 / (\sigma + (o+\pi)/p)$

σ is sequential fraction of the program

π is parallel fraction of the program

$\sigma+\pi = 1$

o is parallel overhead (does not occur in sequential execution)

Draw speedup curves for

p=1, 2, 4, 8, 16, 32 σ= ¼ , ½ o= ¼ , ½

When p goes to ∞, S_p goes to?

Colorado State University 25

Amdahl's law (from reading)

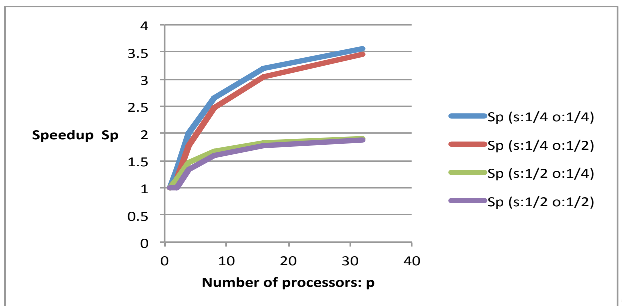
Colorado State University 26

Plotting speedups

$T_1 = 1, \quad T_p = \sigma + (o + \pi)/p$

$S_p = 1/(\sigma + (o + \pi)/p)$

p	Sp (s:1/4 o:1/4)	Sp (s:1/4 o:1/2)	Sp (s:1/2 o:1/4)	Sp (s:1/2 o:1/2)
1	1	1	1	1
2	1.33	1.14	1.14	1.00
4	2.00	1.78	1.45	1.33
8	2.67	2.46	1.68	1.60
16	3.2	3.05	1.83	1.78
32	3.56	3.46	1.91	1.88



Outline for today's lecture

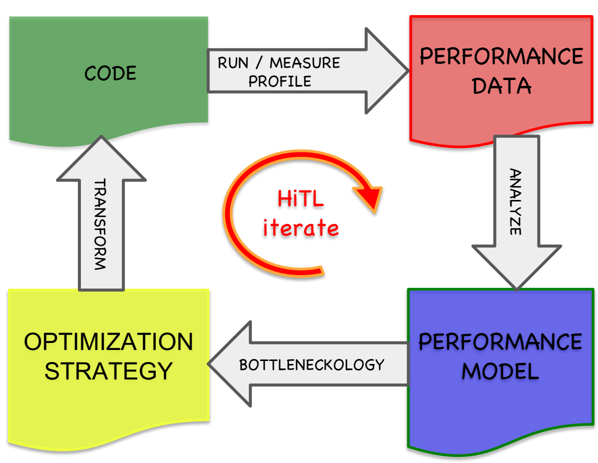
- Let's play a game
  - Pose (and answer) questions and pose more
- Why parallelism
- Overview of speedup and efficiency
- Plan for the class
- Background quiz
  - What do you already know (so I know what to review)
  - Make sure clickers work

Course outline

- Basic parallel programming principles
- Explicitly parallel program (in C and dialects of C)
  - Shared memory (in OpenMP)
  - GPU/Accelerators (in CUDA)
  - Clusters/distr. memory machines (in MPI)
- Write-execute-measure-analyze-report

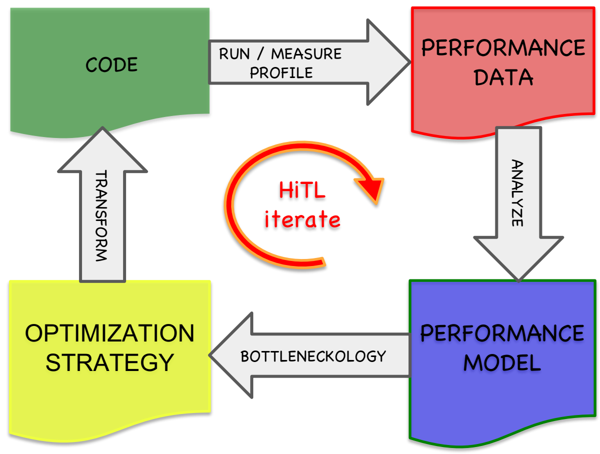
Colorado State University 29

HPC Tuning



Colorado State University 30

Users = boxes; Tools = arrows



Colorado State University 31