Parallel Computing

- Traditionally, computer software has been written for serial computation.
- To solve a problem, an algorithm is constructed and implemented as a serial stream of instructions.
- These instructions are executed on a central processing unit on one computer.
- Only one instruction may execute at a time—after that instruction is finished, the next is executed.

Parallel Computing

- Parallel computing, uses multiple processing elements simultaneously to solve a problem.
- This is accomplished by breaking the problem into independent parts so that each processing element can execute its part of the algorithm simultaneously with the others.
- The processing elements can be diverse and include resources such as a single computer with multiple processors, several networked computers, specialized hardware, or any combination of the above.

Parallel Computing

- Frequency scaling was the dominant reason for improvements in computer performance from the mid-1980s until 2004.
- The runtime of a program is equal to the number of instructions multiplied by the average time per instruction.
- Maintaining everything else constant, increasing the clock frequency decreases the average time it takes to execute an instruction.
- An increase in frequency thus decreases runtime for all computation-bounded programs.

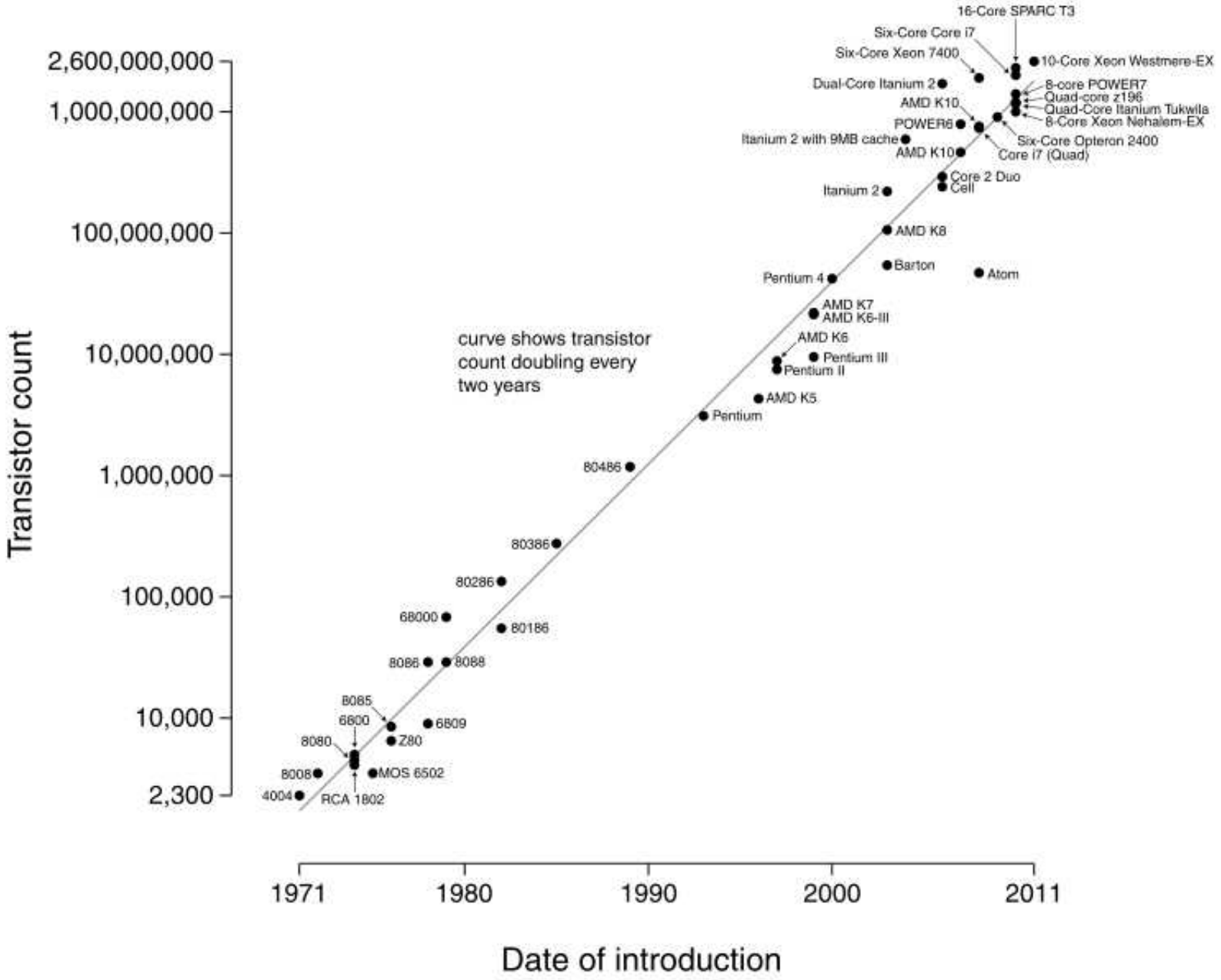
Parallel Computing

- Power consumption by a chip is given by the equation
  - $P = C \times V^2 \times F$
    - where P is power, C is the capacitance being switched per clock cycle (proportional to the number of transistors whose inputs change), V is voltage, and F is the processor frequency (cycles per second).
- Increases in frequency increase the amount of power used in a processor.
- The end of frequency scaling as the dominant computer architecture paradigm.

Parallel Computing

- Moore's law describes a long-term trend in the history of computing hardware:
  - the number of transistors that can be placed inexpensively on an integrated circuit doubles approximately every two years.
- Despite power consumption issues, and repeated predictions of its end, Moore's law is still in effect.
- With the end of frequency scaling, these additional transistors can be used to add extra hardware for parallel computing.

Microprocessor Transistor Counts 1971-2011 & Moore's Law



Parallel Computing

- Optimally, the speed-up from parallelization would be linear
  - doubling the number of processing elements should halve the runtime, and doubling it a second time should again halve the runtime.
- Very few parallel algorithms achieve optimal speed-up.
- Most of them have a near-linear speed-up for small numbers of processing elements, which flattens out into a constant value for large numbers of processing elements.

Parallel Computing

- The potential speed-up of an algorithm on a parallel computing platform is given by Amdahl's law, originally formulated by Gene Amdahl in the 1960s.
  - It states that a small portion of the program which cannot be parallelized will limit the overall speed-up available from parallelization.
- A program solving a large mathematical or engineering problem will typically consist of several parallelizable parts and several non-parallelizable (sequential) parts.

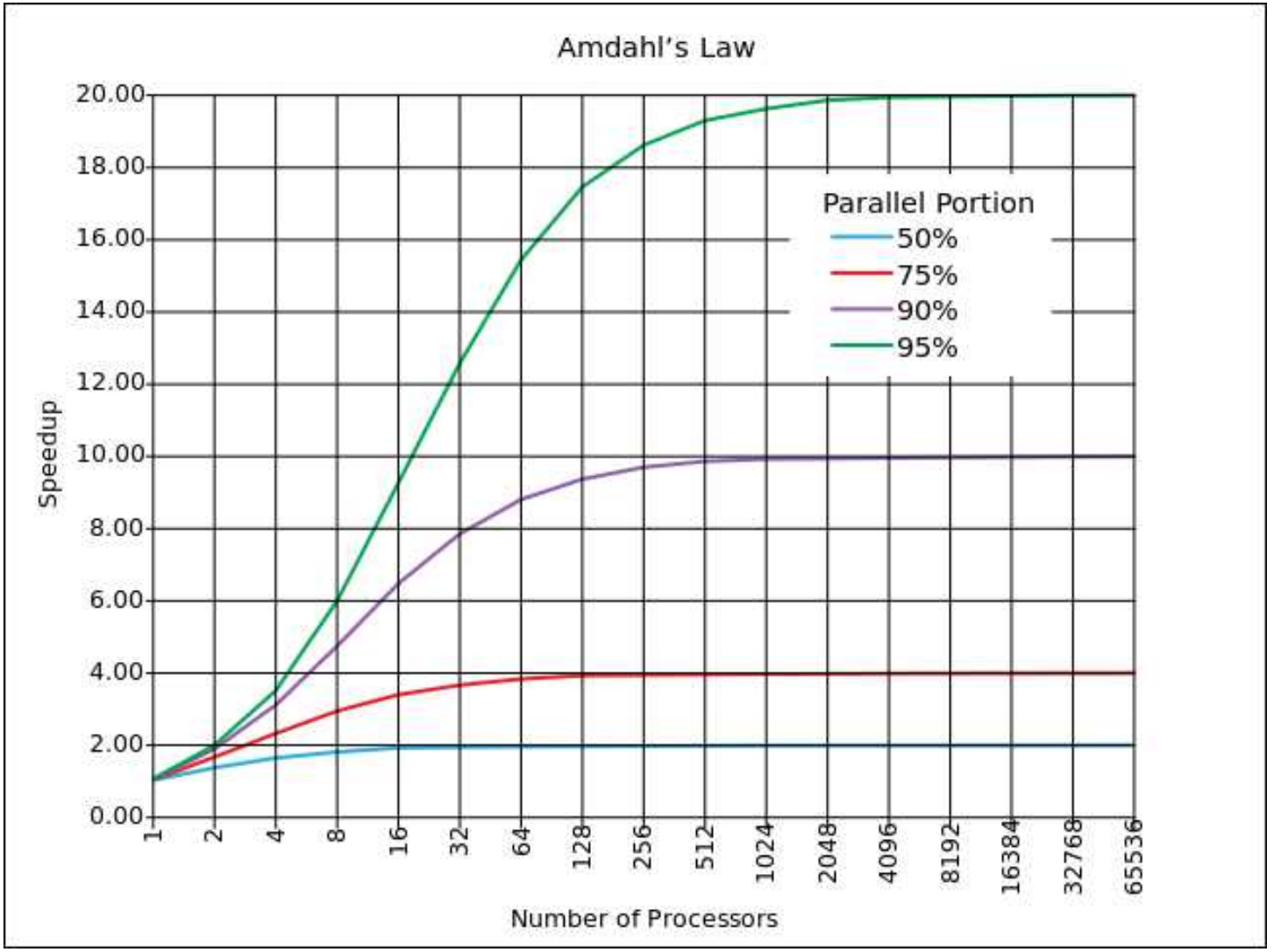
Parallel Computing

- If α is the fraction of running time a sequential program spends on non-parallelizable parts, then

$$S = \frac{1}{\alpha}$$

- is the maximum speed-up with parallelization of the program.

Parallel Computing



http://en.wikipedia.org/wiki/File:AmdahlsLaw.svg

Parallel Computing

- Types of Parallelism
  - Bit-level parallelism
  - Instruction level parallelism
  - Data parallelism
  - Task parallelism
- Classes of Parallel Computers
  - Multicore Computing
  - Symmetric Multiprocessing
  - Distributed Computing

Bit level parallelism

- Parallel computing based on increasing the processor word size.
- Earliest form of increasing parallelism.
- Higher word size reduces number of instructions to be executed to operate on variables which are larger than the processor word size.
- A 16 bit processor can add two 16 bit integers faster than a 8 bit processor.

Instruction level parallelism

- Parallelism using the potential overlap among instructions.
- Is a measure of how many operations in a computer program can be done simultaneously.
  - E = A + B
  - F = C + D
  - G = E + F
- Can be completed in 2 units of time giving an ILP of 3/2.

Data Parallelism

- Also knows as loop-parallelism.
- Distributes data across different processing nodes.
- SIMD
  - each processor performs the same task on different pieces of distributed data.

Task Parallelism

- Also known as function parallelism or control parallelism.
- Focuses on distributing execution processes (threads) across different parallel computing nodes.
- Each processor executes a different thread (or process) on the same or different data.

Multicore computing

- A multi-core processor is a single computing component with two or more independent actual processors (called "cores"), which are the units that read and execute program instructions.
- The multiple cores can run multiple instructions at the same time.

Symmetric Multiprocessing

- Computer hardware architecture where two or more identical processors are connected to a single shared main memory and are controlled by a single OS instance.
- SMP systems allow any processor to perform tasks which operate or any data independent of the data location provided the tasks are not duplicated.
- If multicore hardware is involved, each core is treated as a separate processor.

Distributed Computing

- A distributed computing system consists of multiple autonomous computers that communicate through a computer network.
- Distributed computing refers to the use of distributed systems to solve computational problems.
- A problem is divided into many tasks, each of which is solved by one or more computers.
- They communicate with each other by message passing.

Thank You.