High Performance Scientific computing
Lecture 2

S. Gopalakrishnan

Why Parallel?

Gordon Moore

Robert Dennard

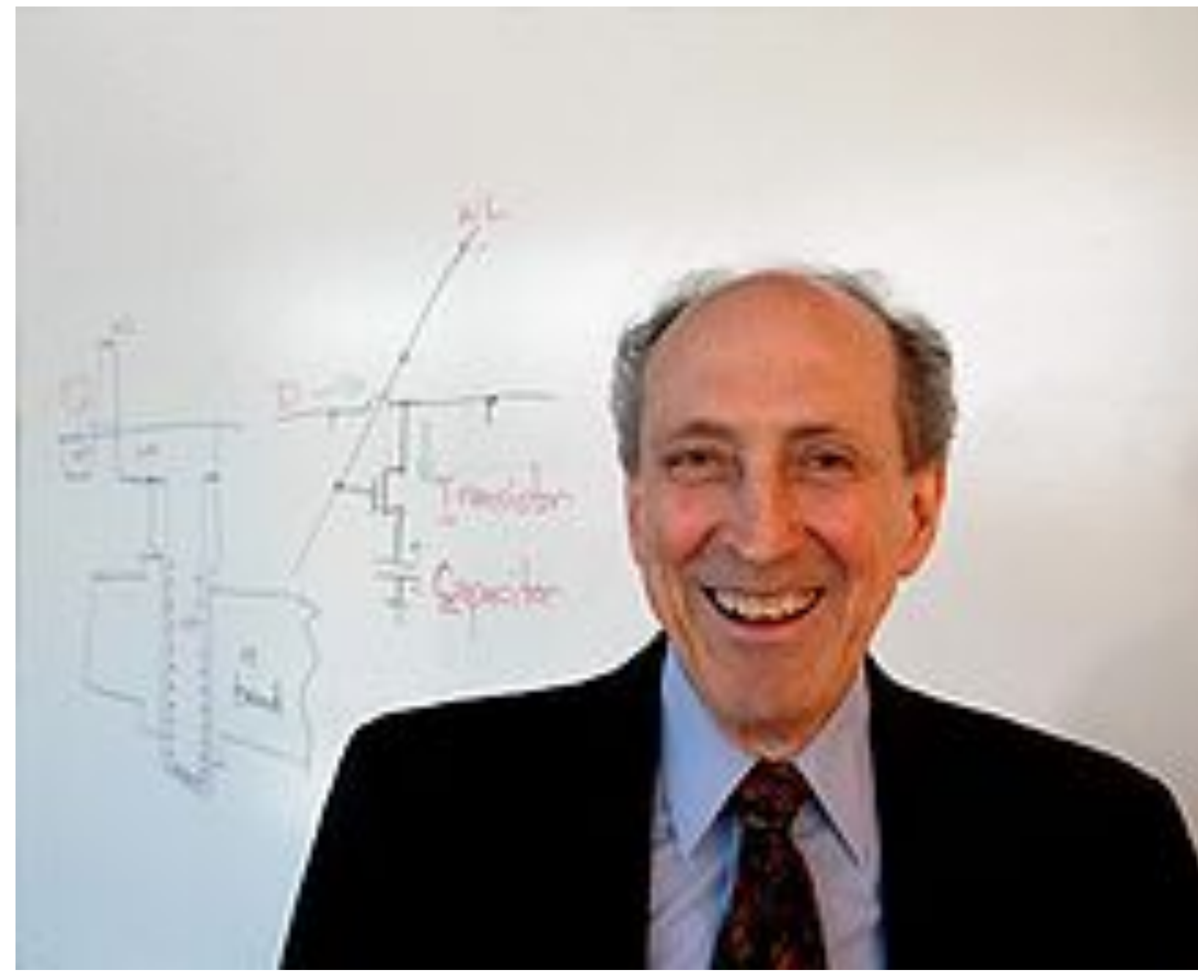
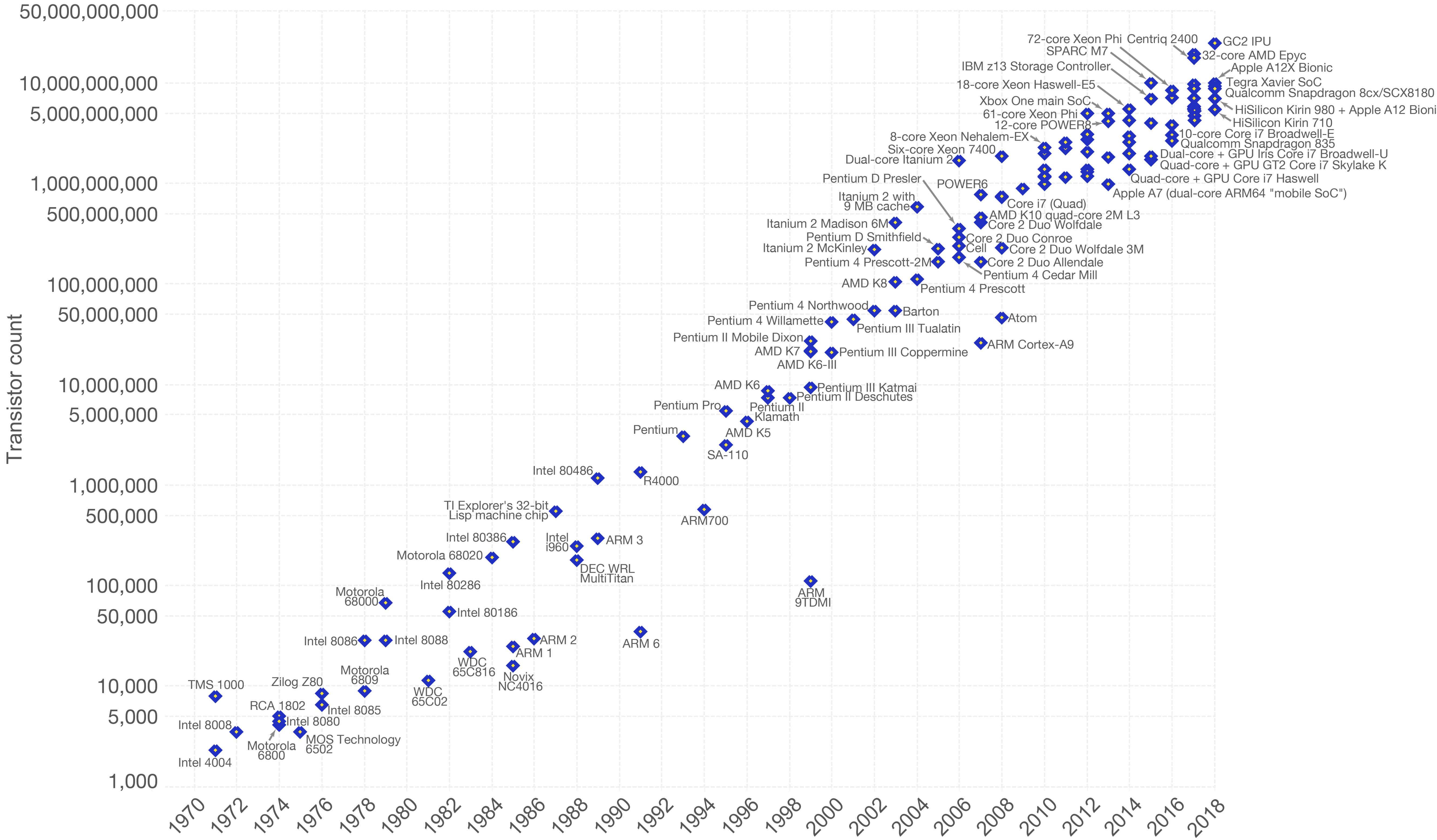


Image: ©Wikipedia

Moore's Law – The number of transistors on integrated circuit chips (1971-2018)

Moore's law describes the empirical regularity that the number of transistors on integrated circuits doubles approximately every two years. This advancement is important as other aspects of technological progress – such as processing speed or the price of electronic products – are linked to Moore's law.



Data source: Wikipedia (https://en.wikipedia.org/wiki/Transistor_count)
The data visualization is available at OurWorldinData.org. There you find more visualizations and research on this topic.

Image: ©Wikipedia

MOSFET

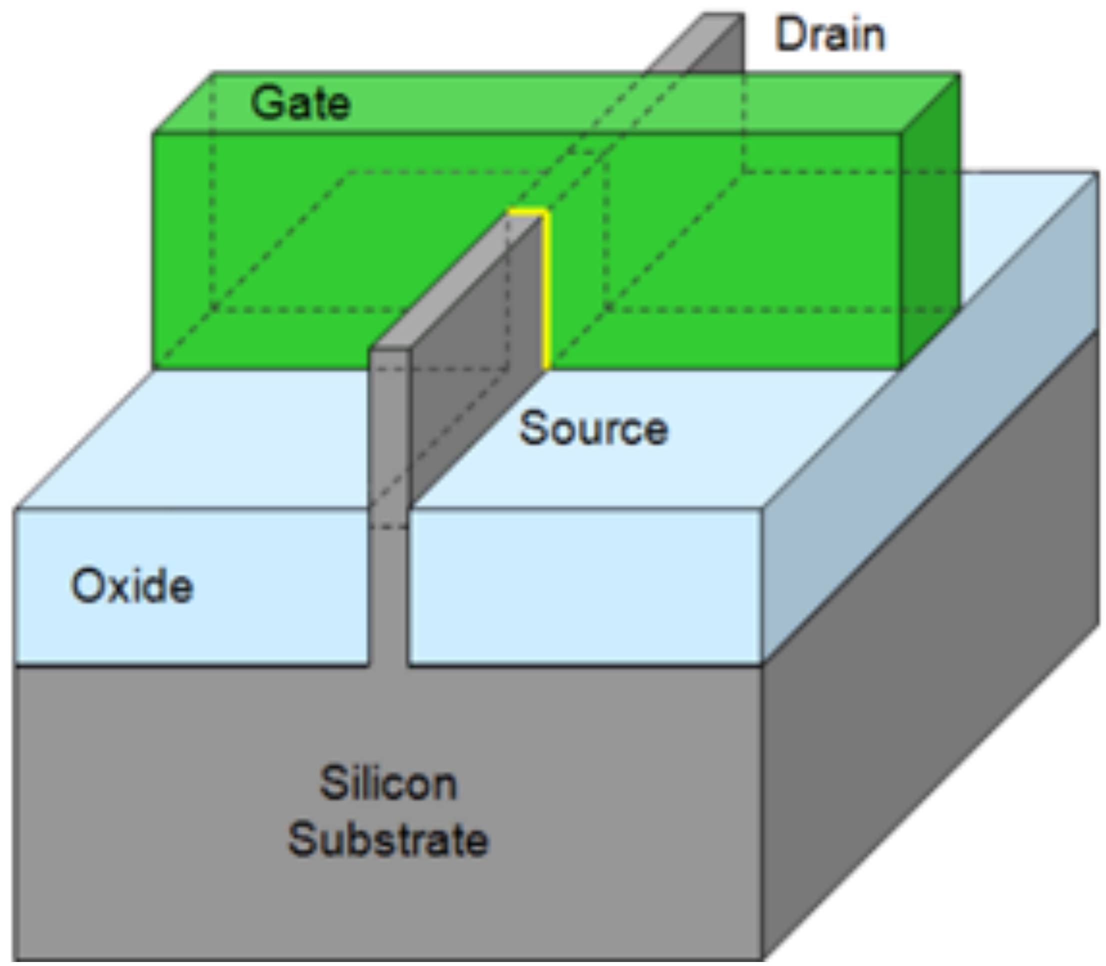


Image: ©Wikipedia

MOSFET scaling

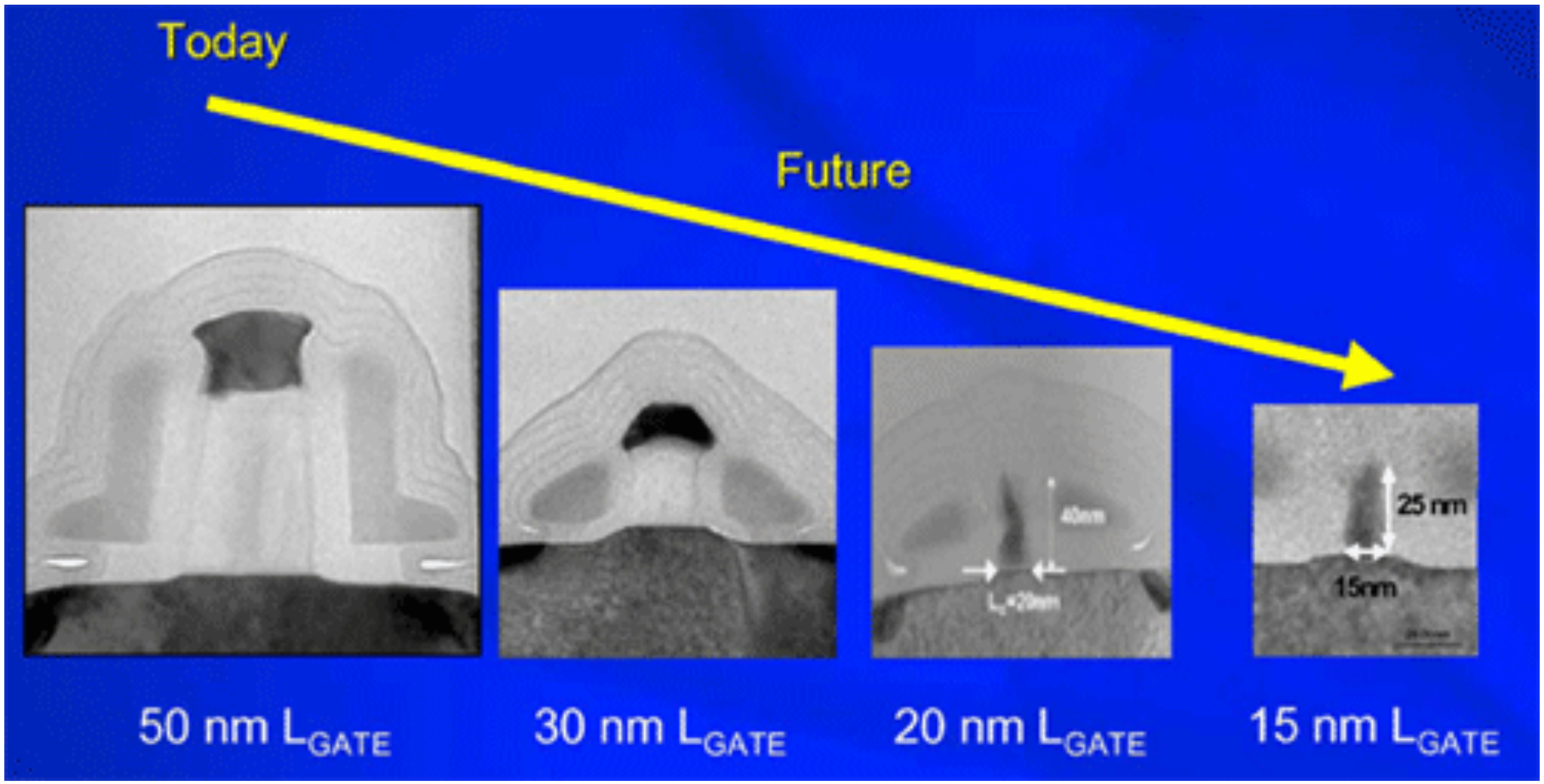
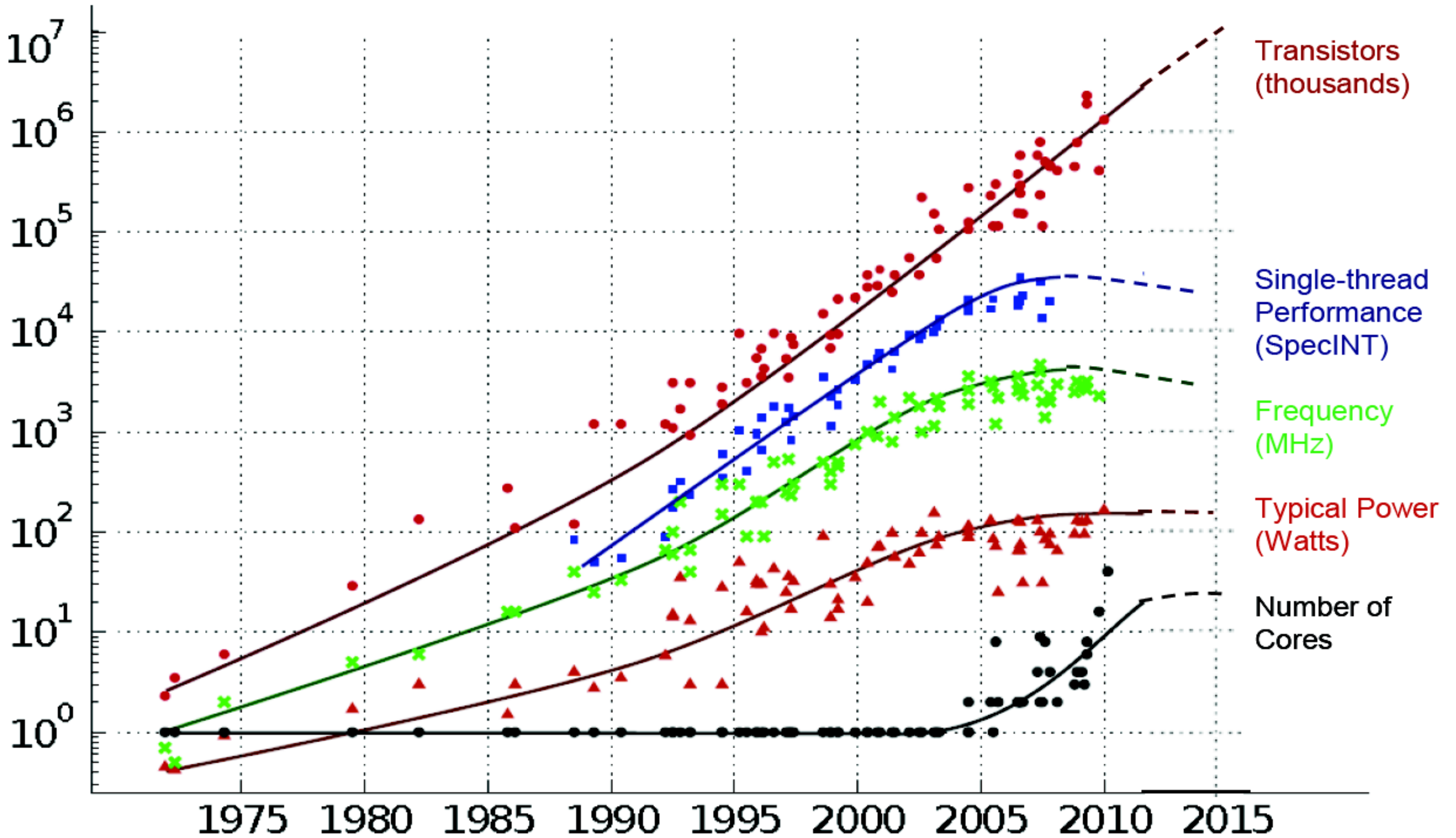


Image: ©Intel Corp.

Dennard Scaling of MOSFET's (1974)

Parameter	Factor
Dimension	1/k
Voltage	1/k
Current	1/k
Capacitance	1/k
Delay time	1/k
Power dissipation/ circuit	$1/k^2$

35 YEARS OF MICROPROCESSOR TREND DATA



Original data collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond and C. Batten
Dotted line extrapolations by C. Moore

Parallel computing

- High performance computing involves parallel programming
- You cannot run away from parallel programming in the future :).
- Problem is people do not think in parallel.
- Automatic parallelization has been a disaster so far.
- There are other limits / roadblocks to parallelism. Messages need to be communicated.
- Amdahl’s Law

Amdahl’s law

Amdahl's law states that if *P* is the proportion of a program that can be made parallel (i.e., benefit from parallelization), and (1 – *P*) is the proportion that cannot be parallelized (remains serial), then the maximum speedup that can be achieved by using *N* processors is given as

$$S(N) = \frac{1}{(1-P) + \frac{P}{N}}$$

Image: ©Wikipedia

Amdahl's law

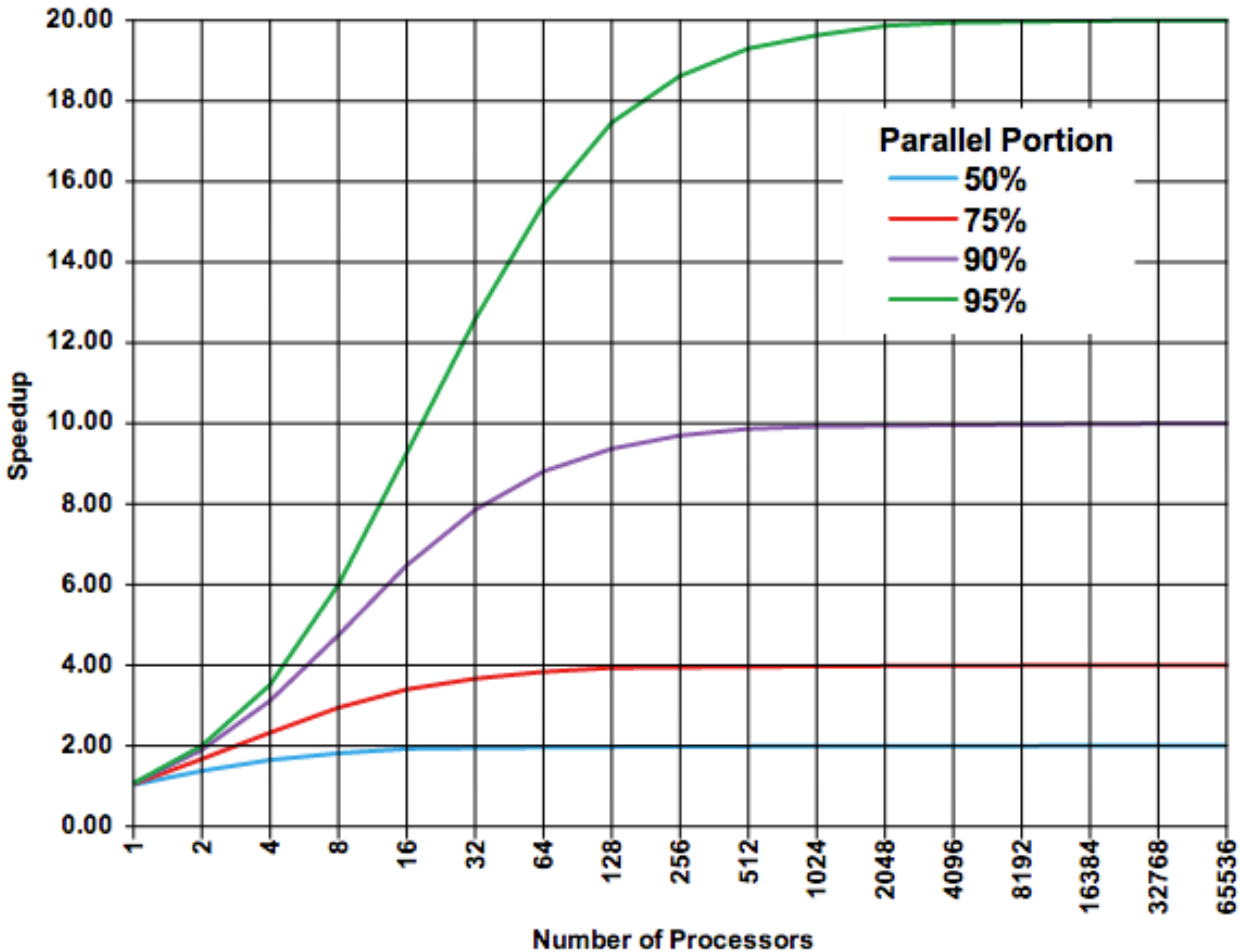


Image: ©Wikipedia

Numerical Modeling ?

- Why do we need need numerical modelling in the first place?

Numerical Modeling ?

- Why do we need need numerical modelling in the first place?
- Usually some complex or not so complex phenomena needs to studied.

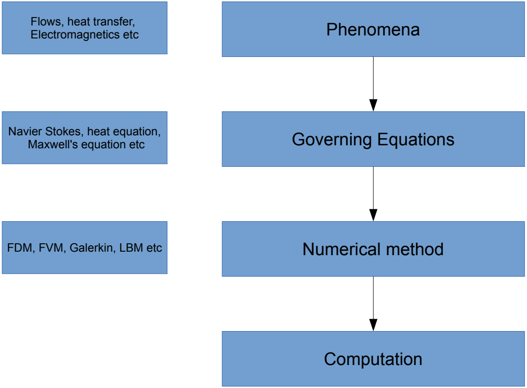
Numerical Modeling ?

- Why do we need need numerical modelling in the first place?
- Usually some complex or not so complex phenomena needs to studied.
- Analytical solutions are not easily available or in most cases not achievable at all.

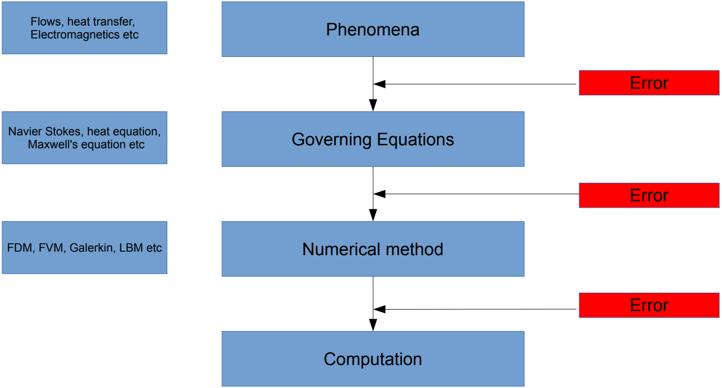
Numerical Modeling ?

- Why do we need need numerical modelling in the first place?
- Usually some complex or not so complex phenomena needs to studied.
- Analytical solutions are not easily available or in most cases not achievable at all.
- What are the steps involved in numerical modelling?

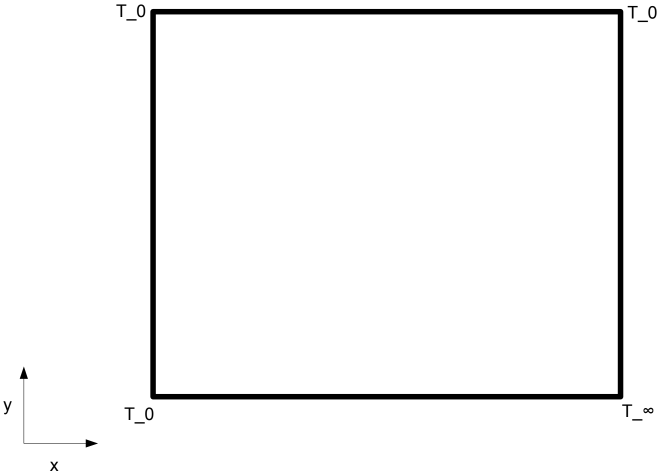
Numerical Modeling



Numerical Modeling



Steady State Heat Conduction

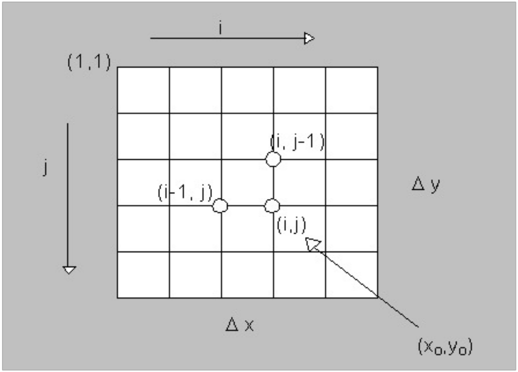


Steady State Heat Conduction

Phenomenon is modelled using Laplace's equation

$$\frac{\partial^2 T}{\partial x^2} + \frac{\partial^2 T}{\partial y^2} = \nabla^2 T = 0$$

which can be discretised on a grid as,



Steady State Heat Conduction

The discretised equation at a single point (i,j) is

$$\frac{T_{i-1,j} - T_{i,j} + T_{i+1,j}}{(\Delta x)^2} + \frac{T_{i,j-1} - T_{i,j} + T_{i,j+1}}{(\Delta y)^2} = 0$$

Assemble all the equations for all unknown points in the Matrix form and then solve

$$Ax = B$$

You can choose any Linear Algebra Solver (iterative or Direct). Iterative is more efficient.

So what are the steps involved ?

One writes a computer code,

```
#define NC       1000                  /* Number of Cols        */
#define NR       1000                   /* Number of Rows        */
#define NITER    1000                   /* Max num of Iterations */
#define MAX(x,y) ( ((x) > (y)) ? x : y )

#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <mpi.h> /* only for timing */
#include <sys/time.h>

void initialize( float t[NR+2][NC+2] );
void set_bcs    ( float t[NR+2][NC+2]);

int main( int argc, char **argv ){

  int        niter;                      /* iter counter  */

  float      t[NR+2][NC+2];             /*temperature */
  float      told[NR+2][NC+2];          /* previous temperature*/
  float      dt;                         /* Delta t       */

................................
```

Compile and Execute!!!

Steady State Heat Conduction

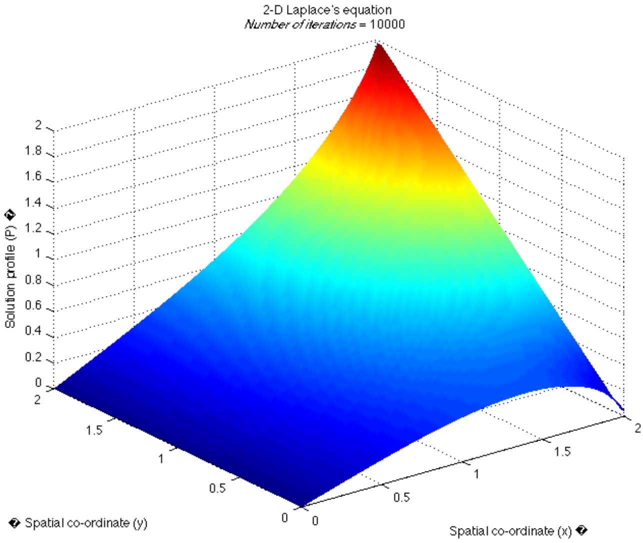


Image: Mathworks