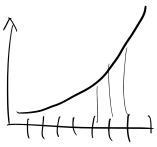
ESC401, ESC402

Why parallel computing?

Natur is parallel speed of light - c

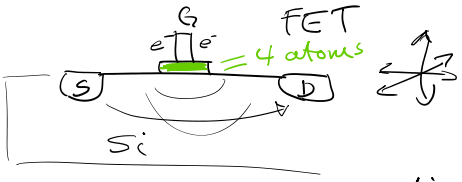
Parallel speed up (100x)

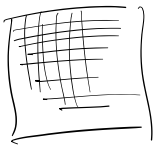
$(1.6)^{10}$ — gaining ~ 10 years on serial codes.



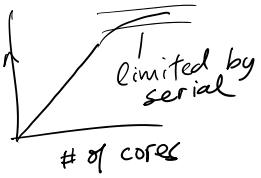
1 per year speed up MOORE's LAW
factor of CPUs

S: Source
D: Drain
G: Gate





Limits to parallel computing
AMDAHL's LAW (IBM)



How can parallelize?

① Vectors (single core)

256 bits

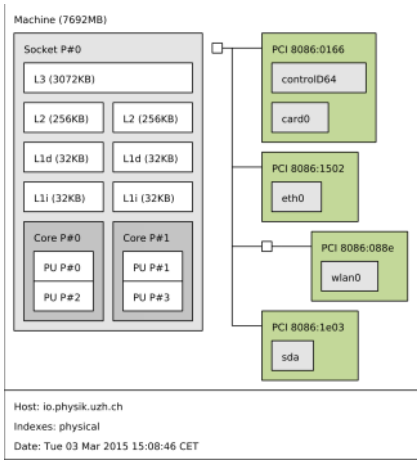
GPUs Tensor Ops
4x4 Matrix ⊗ ✓
and half-precision

⊕

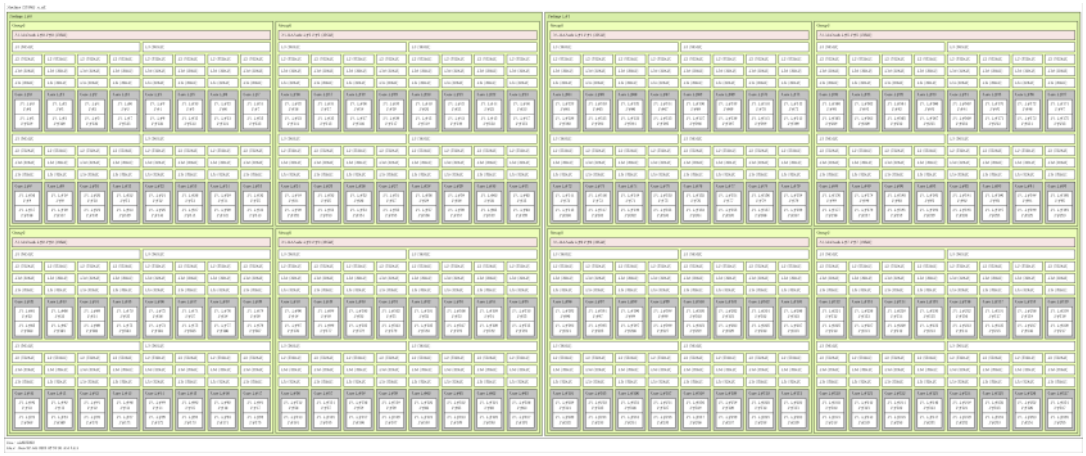
AVX-512 (bits)

② Thread level
Independent Threads | Symmetric Case SIMT
Intel, AMD > core — 2 threads Spanning cores
Shared Memory

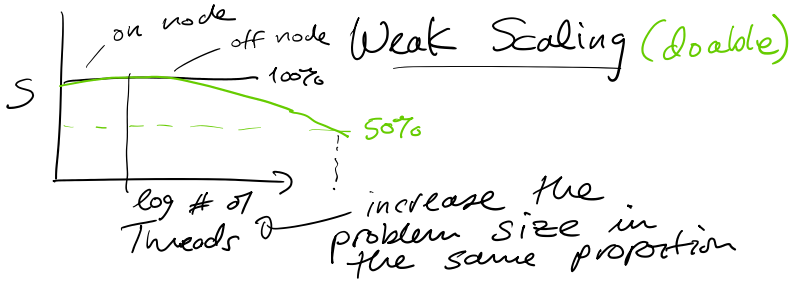
③ Message Passing Spanning Nodes/Sockets
Communication Network!



256 Threads

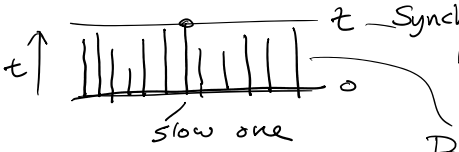


Strong Scaling (hard)
Same problem and make it faster



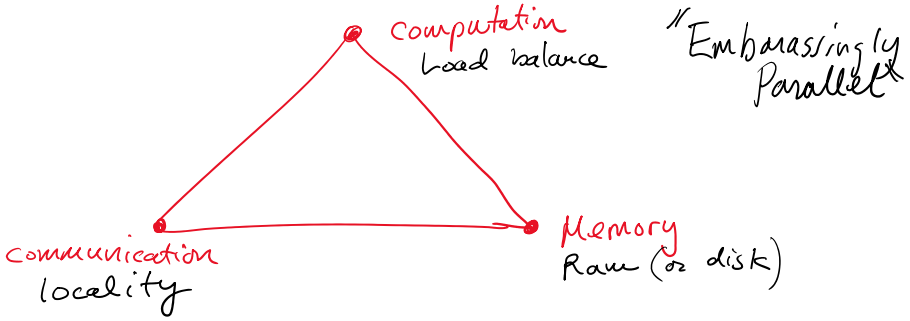
Weak Scaling (doable)
increase the problem size in the same proportion

MIMD - multiple instruction Multiple data
↳ Finish a task at different times

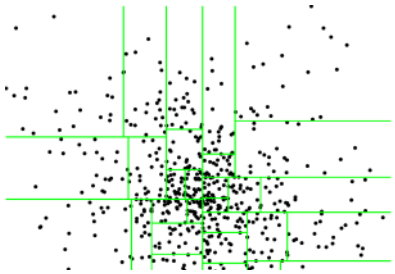


t — Synchronization point.

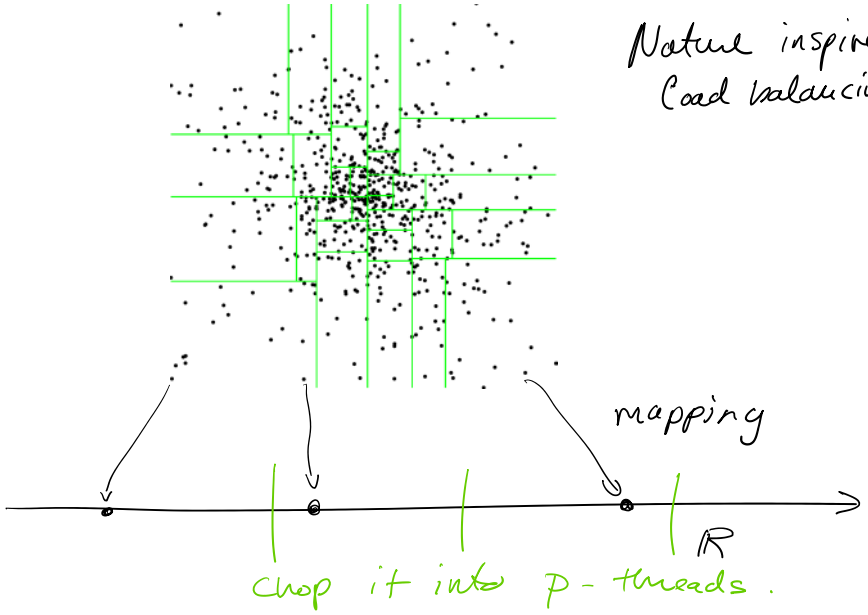
Balance the work!
Load balancing

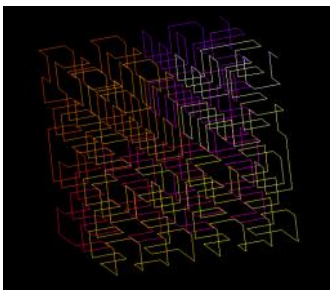


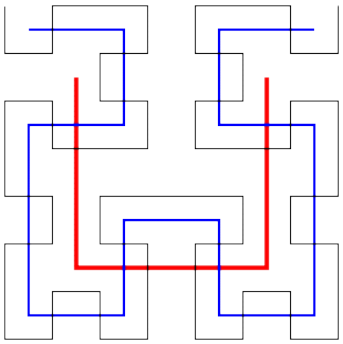
"Embarassingly Parallel"

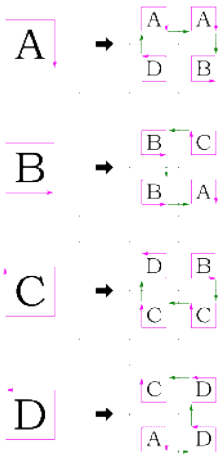


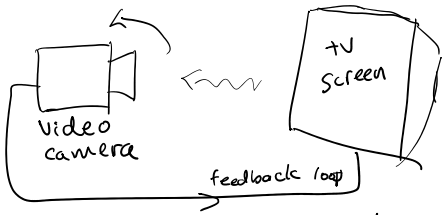
Nature inspired
Load balancing









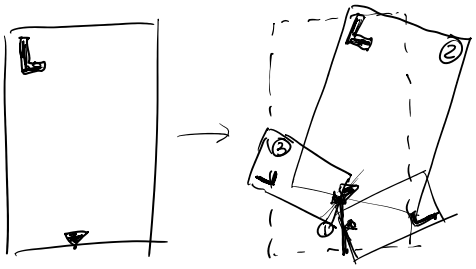


Fractal generator

MRCM
Multiple reduction copy machine

$$\begin{pmatrix} x_{neu} \\ y_{neu} \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix}$$

Affine Transformation



Barnsley's Fern

use 4 Affine Transf.

Fortune Wheel
Reduction Copy
Machine

lens	a	b	c	d	e	f	p	feature
1	0	0	0	0.16	0	0	0.01	"stem"
2	0.85	0.04	-0.04	0.85	0	1.6	0.85	whole thing (all the leaves)
3	0.20	-0.26	0.23	0.22	0	1.6	0.07	left leaf (lower)
4	-0.15	0.28	0.26	0.24	0	0.4	0.07	right leaf (lower)

Set white
$x = y = 0$
$n = 0$
while $(n < nMax)$ {
    r = random
    if $(r < 0.01)$
        $(x_n, y_n) = f_1(x, y)$
    else if $(r < 0.86)$
        $(x_n, y_n) = f_2(x, y)$
    else if $(r < 0.93)$
        $(x_n, y_n) = f_3(x, y)$
    else
        $(x_n, y_n) = f_4(x, y)$
    draw pixel (x_n, y_n) green
    $(x, y) = (x_n, y_n)$
    n++
}

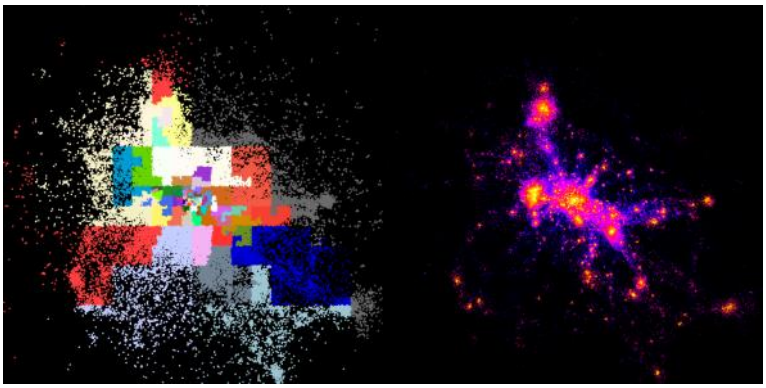
Check out this app on the web!

A Barnsley Fern Generator (chradams.co.uk)

lens	a	b	c	d	e	f	p
1	0.03	0	0	0.1	0	0	0.02
2	0.85	0	0	0.85	0	1.5	0.6
3	0.8	0	0	0.8	0	1.5	0.1
4	0.2	-0.08	0.15	0.22	0	0.85	0.07
5	-0.2	0.08	0.15	0.22	0	0.85	0.07
6	0.25	-0.1	0.12	0.25	0	0.3	0.07
7	-0.2	0.1	0.12	0.2	0	0.4	0.07

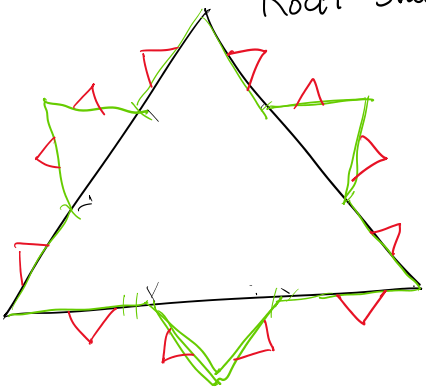
Enjoy and

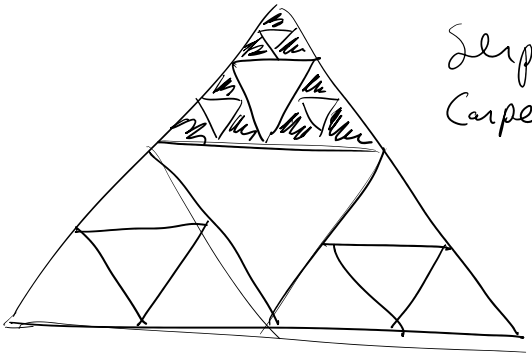
Enjoy and
Merry
Christmas!



Decomposition into threads using 3-D Hilbert curve (fractal).

Koch Snowflake





Sierpinski Carpet.