Assignment 4 – Task 1
Multicore (dynamic) power envelope

- Assume: There is no static power consumption: $W(f=0) = 0$
- W_d Dynamic power consumption of the chip at $f = f_0$
- Δf clock frequency change ($\Delta f = f - f_0$)

→ **Power consumption of 1 core** $W = W_d \cdot \left(1 + \frac{\Delta f}{f_0}\right)^3$

(a) Overclocking by 30% → $\frac{\Delta f}{f_0} = 0.3$ → $W \approx W_d \cdot 2.2$

(b) Power dissipation with m cores $\left(\Delta v = {}^{\Delta f}/_{f_0}\right)$:

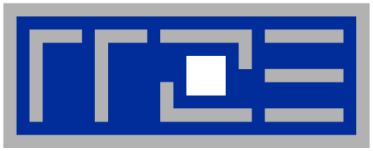
$$W(m) = m \cdot W_d \cdot (1 + \Delta v)^3 = W_d \implies \Delta v = m^{-1/3} - 1$$

If $\Delta v = -\frac{1}{2} \implies m = 8$

(c) Corner cases @ $m = 8$

- Memory-bound → no gain at all
- Compute-bound → speedup = $m \cdot (-\Delta v) = 4$.

Assignment 4 – Task 2
Parallel π by integration



- The code

```
#pragma omp parallel for private(x) reduction (+:sum)
 for (int i=0; i < n; i++) {
    x = (i+0.5)*delta_x;
    sum += (4.0 / (1.0 + x * x));
  }
```

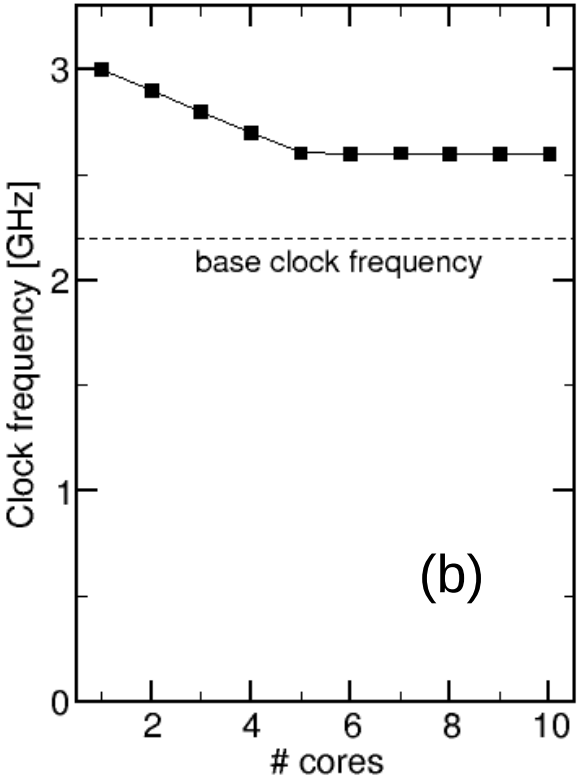
- Compile with `-qopenmp` (plus the other options)
- Run:

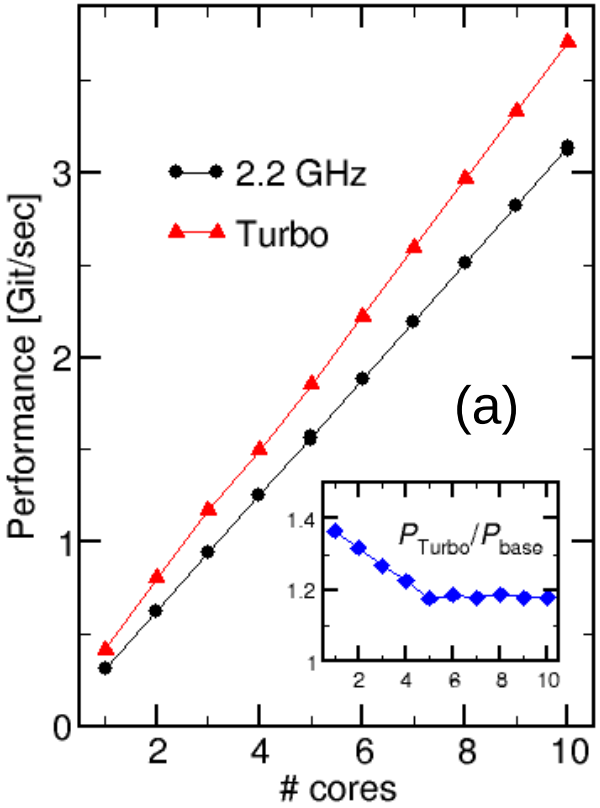
```
$ OMP_NUM_THREADS=X OMP_PLACES=cores \
                    OMP_PROC_BIND=close ./a.out
```

Assignment 4 – Task 2
Parallel π by integration

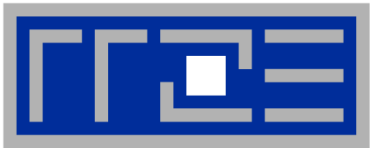
Performance and clock speed:

- Actual clock speed $f = P \times \frac{7\frac{cy}{it}}{\#\ cores}$





Assignment 4 – Task 3
Peak performance of Xeon Phi KNC/Nvidia Volta



- Xeon Phi “Knights Corner”
  - 60 cores
  - f=1.05 GHz
  - AVX-512 instruction set (512-bit registers, 1 full-width DP FMA instruction per cycle)

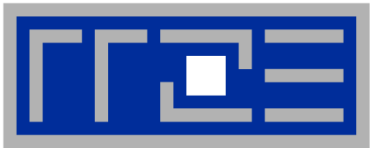
$$P_{peak} = 60 \times 2\,\frac{instr}{instr} \times 1\,\frac{instr}{cy} \times 16\,\frac{flop}{instr} \times 1.05\,\frac{Gcy}{s} = 1{,}008\ \text{GFlops/s}$$

cores | n_{FMA} | n_{super} | n_{SIMD} | f

- Nvidia Volta V100
  - 80 SM units (streaming multiprocessors), 64 SP “cores” w/ 1 FMA each
  - f=1.4 GHz

$$P_{peak} = 80 \times 2\,\frac{instr}{instr} \times 1\,\frac{instr}{cy} \times 64\,\frac{flop}{instr} \times 1.4\,\frac{Gcy}{s} = 14{,}336\ \text{GFlops/s}$$

Assignment 4 – Task 4
A simple power model for multicore chips $W(n, f)$



- W_d Dynamic power consumption of a running core (see Assignment 4 – T3)
- Δv relative clock frequency change (see Assignment 4 – T3
- n cores used (of n_{max} available) (see Assignment 4 – T3)
- W_0 Baseline power consumption of the chip (all cores idle): $W(n = 0) = W_0$

→ **Power** $W(n, f) = W_0 + nW_d(1 + \Delta v)^3$

Assignment 4 – T3

- P(1) Performance of serial program @ $\Delta v = 0$
- P_{limit} Maximum performance of parallel program
- P(n) Performance of parallel program on n cores
- T(n) Time to solution on n cores

→ **Time to solution** $T(n) = \frac{1}{\min(nP\,(1)(1+\Delta v), P_{limit})}$

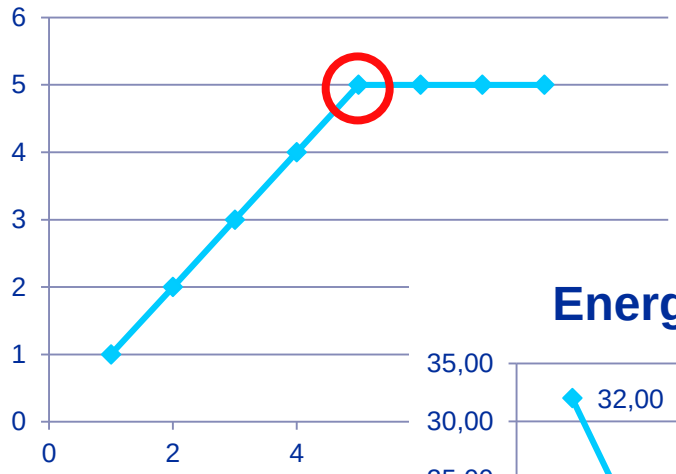
- E(n) Energy to solution

→ **Energy to solution** $E(n) = W(n, f)T(n) = \frac{W_0 + nW_d(1+\Delta v)^3}{\min(nP(1)(1+\Delta v), P_{limit})}$

(a) Minimal energy to solution

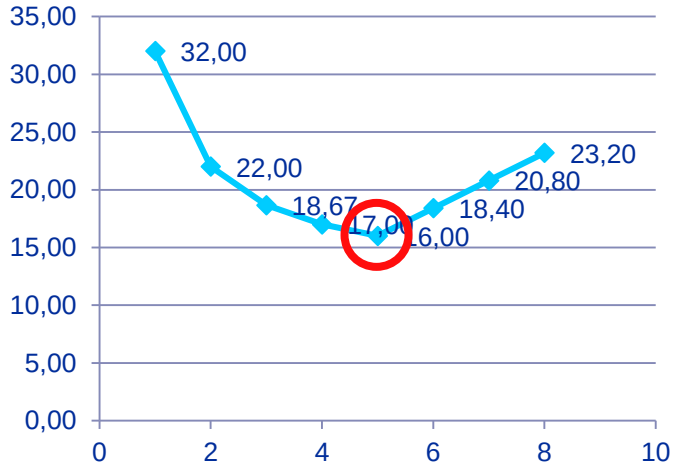
$$E(n) = \frac{W_0 + nW_d(1+\Delta\nu)^3}{\min(nP(1)(1+\Delta\nu), P_{limit})}$$

Performance

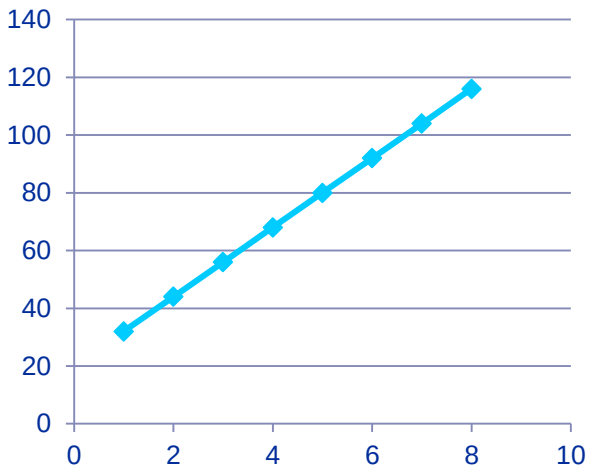


$$P(1) = 1\ \mathrm{s}^{-1}$$
$$P_{limit} = 5\ \mathrm{s}^{-1}$$
$$W_0 = 20\ \mathrm{W}$$
$$W_d = 12\ \mathrm{W}$$
$$\Delta\nu = 0$$

Energy to solution



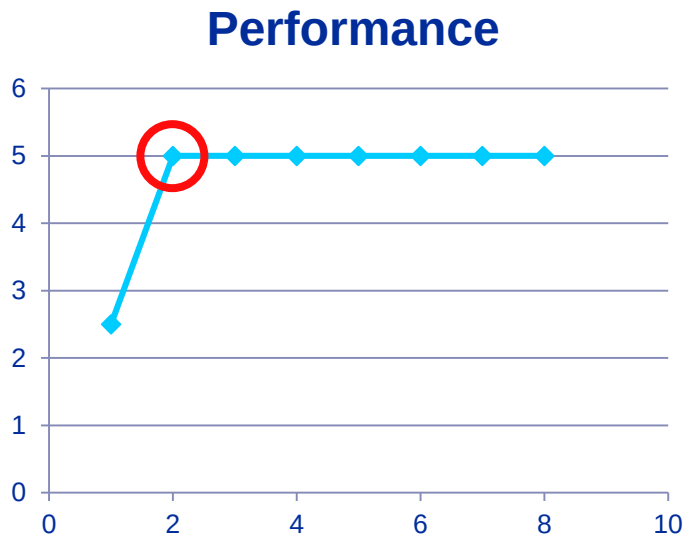
Power

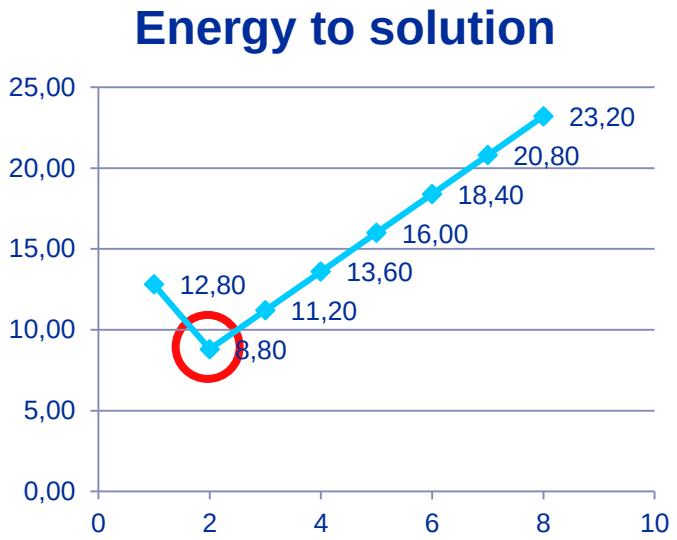


→ Minimal energy at the **saturation** point!

(b) Energy to solution: Increase serial performance

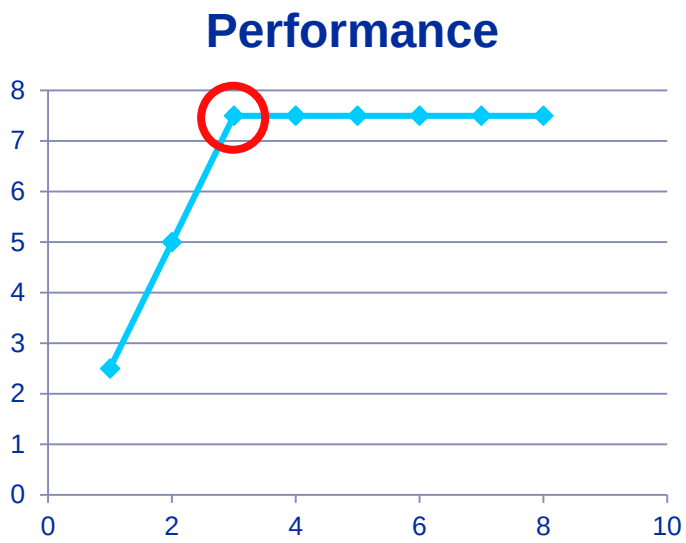
$$P(1) = 2.5\ \mathrm{s}^{-1}$$
$$P_{limit} = 5\ \mathrm{s}^{-1}$$
$$W_0 = 20\ \mathrm{W}$$
$$W_d = 12\ \mathrm{W}$$
$$\Delta\nu = 0$$





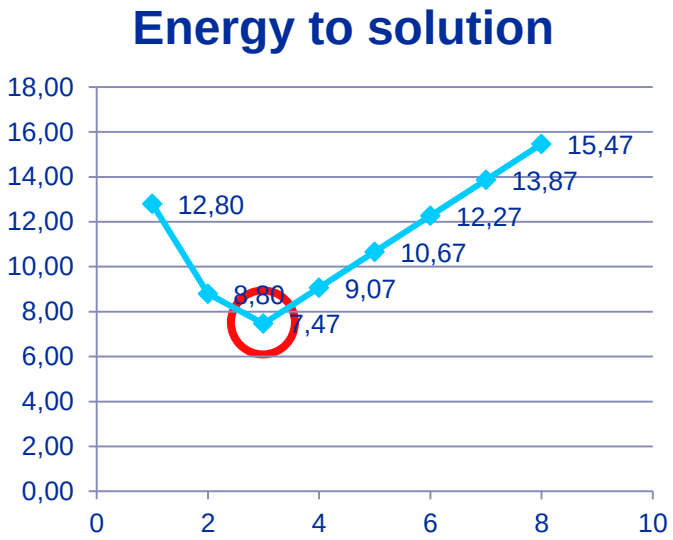
→ Smaller energy to solution at *smaller* core count with faster serial code!

(c) Energy to solution: Increase saturated performance



$$P(1) = 2.5\ \mathrm{s}^{-1}$$
$$P_{limit} = 7.5\ \mathrm{s}^{-1}$$
$$W_0 = 20\ \mathrm{W}$$
$$W_d = 12\ \mathrm{W}$$
$$\Delta\nu = 0$$

→ Smaller energy to solution at *larger* core count with larger saturated performance!



(d) Energy to solution: Dependence on clock speed

- Prerequisite: We do not want to sacrifice saturated performance → $nP(1) \geq P_{limit}$
- For small W_0, expect dominant impact of numerator in

$$E(t) = \frac{W_0 + nW_d(1+\Delta\nu)^3}{\min(nP(1)(1+\Delta\nu), P_{limit})}$$

- Consequence: Turn down clock speed so that P_{limit} is reached as late as possible