Parallelles Programmieren

Motivation: Wir wollen grössere Aufgaben bewältigen
oder bekannte Probleme mit besserer
Auflösung simulieren.

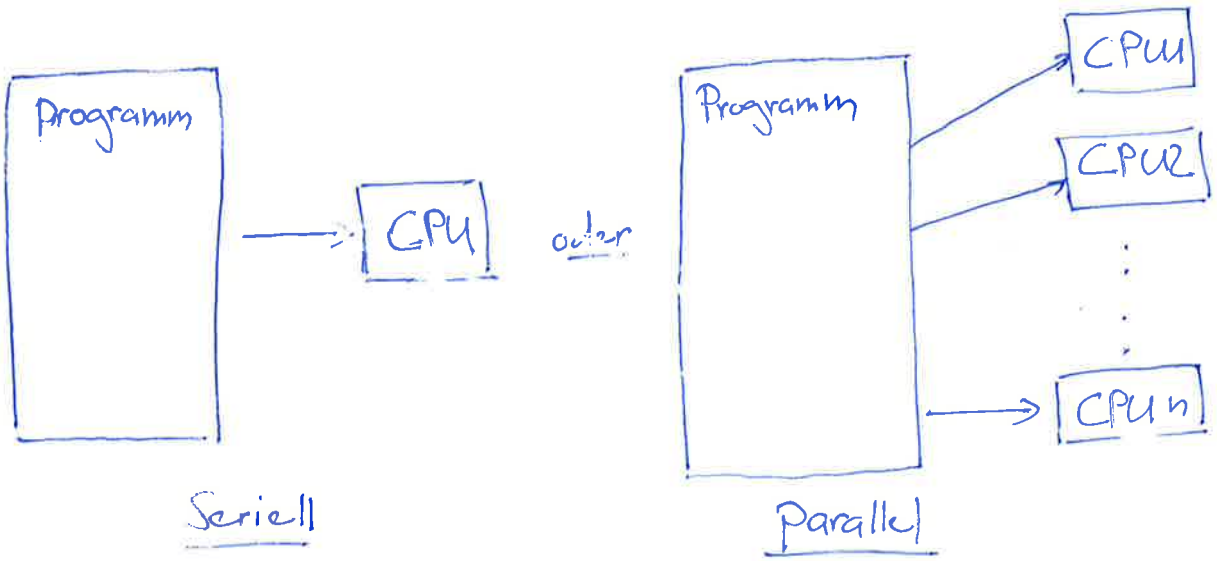
z.B.: Sonnensystem — grössere Systeme mit mehr Planeten oder deutlich längere Entwicklung
\ mehr Detail: Monde, Asteroiden

↳ je nach Fragestellung.

Problem: Wir benötigen deutlich mehr Rechenleistung!

Lösung: 1) Immer leistungsstärkere CPU (bis ~2004)

2) Wir teilen das Problem auf mehrere
"normale" CPUs auf.



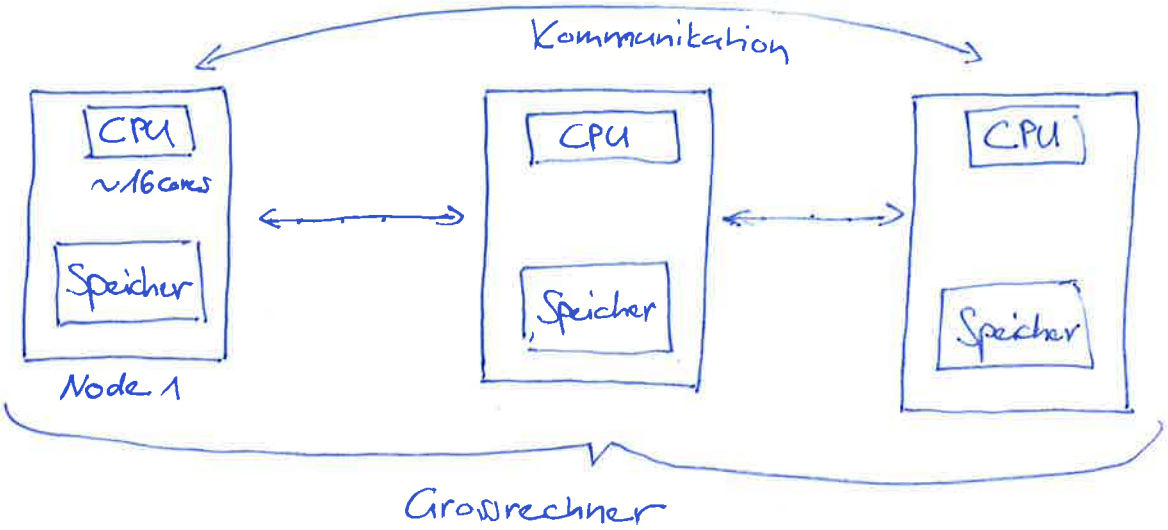
Seriell

Parallel

↓

Zukunft, das CPU Leistung ~2004 bei etwa 3 GHz stagniert

Wir benötigen Grossrechner (Engl. Supercomputer) mit tausenden CPU Kernen.



z.B.: Zbox 4 @ ICS: 192 nodes / 2x 8 cores per node (3072 cores)

Dora @ CSCS: 1256 nodes / ~~12~~ 24 cores per node (30'144 cores)

↳ benötigt spezielle 1) Programmiertechniken und
2) Bibliotheken, um einen Grossrechner effizient zu nützen (Architektur!)

1) Das Problem muss parallelisierbar sein
↳ z.B.: Sonnensystem
Laplace-Gleichung } erst effizient ab gewisser Grösse!
↳ Verhältnis Rechenzeit / Kommunikation

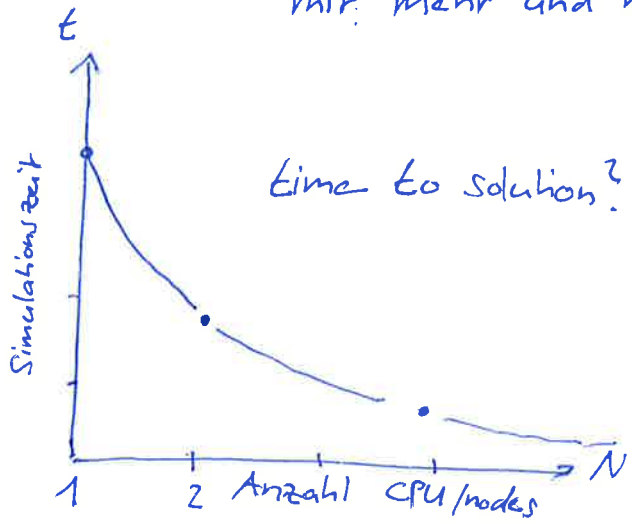
2) oft verwendet: 1) MPI (Message Passing Interface) off node
↳ Funktionen zum Austausch von Daten zwischen Prozessen.
↳ ineffizient on node, da Daten immer noch kopiert werden

2) OpenMP
↳ Kompileranweisungen, schnelles parallelisieren von bestehendem Code mit threads on node
↳ nur shared memory → on node only

↳ hybrid: MPI + OpenMP

Effizienz des Parallelcodes

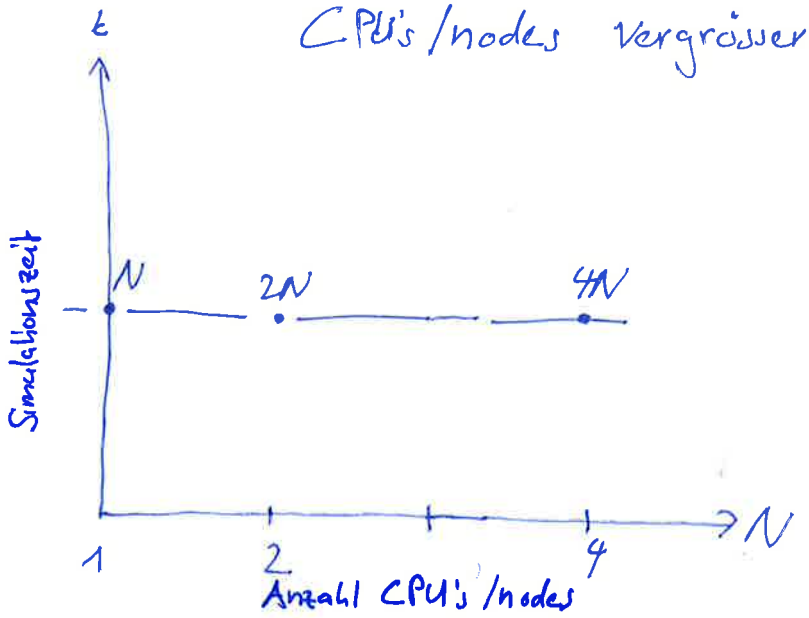
1) "Strong scaling": Problem wird mit fixer Auflösung mit mehr und mehr CPUs simuliert



Time to solution?

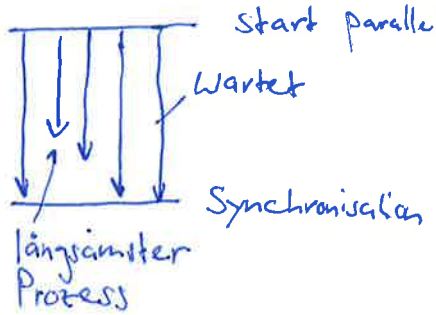
Ideal: bei der doppelten Anzahl CPUs/nodes halbiert sich die Simulationszeit

2) "Weak scaling": Die Auflösung wird mit der Anzahl CPUs/nodes vergrössert



Ideal: bei doppelter Anzahl CPUs/nodes kann das Problem mit der doppelten Auflösung zur gleichen Zeit simuliert werden.

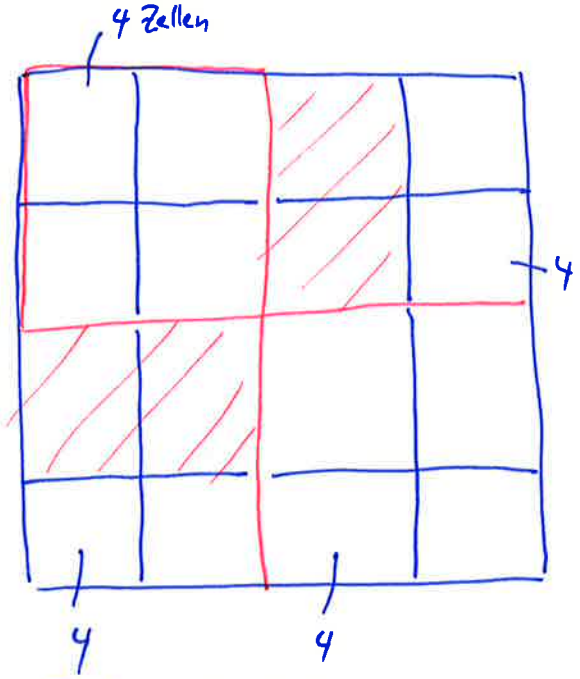
Problem: Kommunikation / Synchronisation



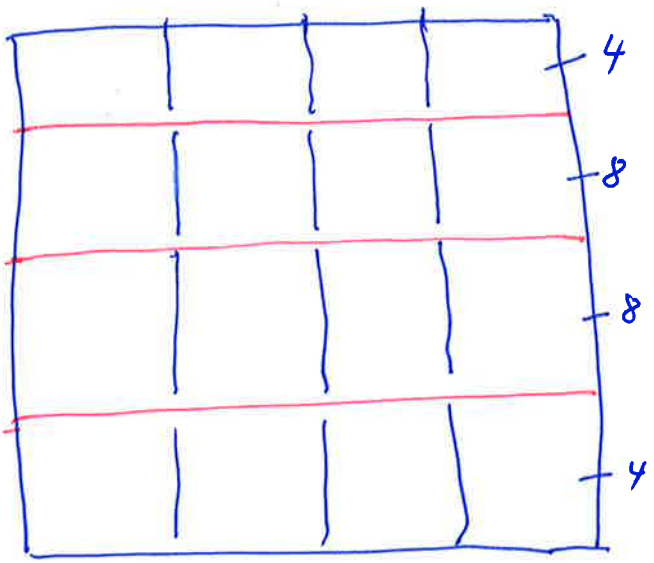
z.B: Sonnensystem (DRIFT, ACCEL, KICK, DRIFT)
Laplace (1x durch das Gitter)

Kommunikation

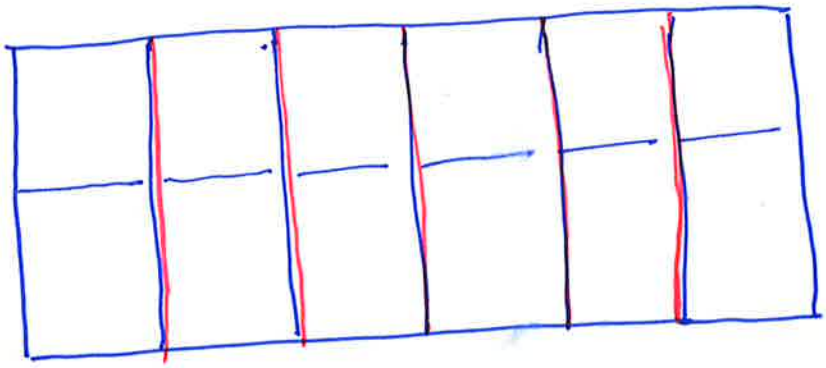
Bsp: Gitter



Total: 16 Zellen



24 Zellen



Hängt von Geometrie des Problems ab!

20 / 32