Understanding Parallelism and the Limitations of Parallel Computing
Understanding Parallelism:

Sequential work

After 16 time steps: 4 cars

(c) RRZE 2013
Understanding Parallelism:
Parallel work

After 4 time steps: 4 cars

“perfect speedup”
Understanding parallelism:

Shared resources, imbalance

- Unused resources due to resource bottleneck and imbalance!
- Waiting for synchronization
- Waiting for shared resource
Limitations of Parallel Computing: Amdahl's Law

Ideal world: All work is perfectly parallelizable

Closer to reality: Purely serial parts limit maximum speedup

Reality is even worse: Communication and synchronization impede scalability even further
Limitations of Parallel Computing:
Calculating Speedup in a Simple Model ("strong scaling")

\[ T(1) = s + p \] = serial compute time (=1)

parallelizable part: \( p = 1 - s \)

purely serial part \( s \)

Parallel execution time:
\[ T(N) = s + \frac{p}{N} \]

General formula for speedup:
Amdahl's Law (1967)
"strong scaling"

\[ S^k_p = \frac{T(1)}{T(N)} = \frac{1}{s + \frac{1-s}{N}} \]
Limitations of Parallel Computing:

Amdahl's Law ("strong scaling")

- **Reality:** No task is perfectly parallelizable
  - Shared resources have to be used serially
  - Task interdependencies must be accounted for
  - Communication overhead (but that can be modeled separately)

- **Benefit of parallelization may be strongly limited**
  - "Side effect": limited scalability leads to inefficient use of resources
  - **Metric: Parallel Efficiency**
    *(what percentage of the workers/processors is efficiently used):*

\[ \varepsilon_p(N) = \frac{S_p(N)}{N} \]

- **Amdahl case:**

\[ \varepsilon_p = \frac{1}{s(N-1)+1} \]
Limitations of Parallel Computing:

Adding a *simple communication model for strong scaling*

- **Serial compute time:**
  \[ T(1) = s + p = \text{serial compute time} \]

- **Parallel part:**
  \[ p = 1 - s \]

- **Parallel time:**
  \[ T(N) = s + \frac{p}{N} + Nk \]

**General formula for speedup:**

\[
S^k_p = \frac{T(1)}{T(N)} = \frac{1}{s + \frac{1-s}{N} + Nk}
\]

(c) RRZE 2013
Limitations of Parallel Computing:

Amdahl's Law ("strong scaling")

- Large N limits
  - at $k=0$, Amdahl's Law predicts
    \[
    \lim_{N \to \infty} S_p^0(N) = \frac{1}{s}
    \]
    independent of $N$!

- at $k \neq 0$, our simple model of communication overhead yields a behaviour of
  \[
  S_p^k(N) \xrightarrow{N \gg 1} \frac{1}{Nk}
  \]

- Problems in real world programming
  - Load imbalance
  - Shared resources have to be used serially (e.g. IO)
  - Task interdependencies must be accounted for
  - Communication overhead

(c) RRZE 2013
Limitations of Parallel Computing:

Amdahl’s Law ("strong scaling") + comm. model

Scalability Laws

(c) RRZE 2013
Limitations of Parallel Computing:

Amdahl’s Law ("strong scaling")

Parallel efficiency:

- <10%
- ~50%

**Graph Details**

- **S(N)** vs. **# CPUs**
- **s=0.01**
- **s=0.1**
- **s=0.1, k=0.05**

(c) RRZE 2013
Limitations of Parallel Computing:  
How to mitigate overheads

- Communication is not necessarily purely serial
  - Non-blocking crossbar networks can transfer many messages concurrently – factor $Nk$ in denominator becomes $k$ (technical measure)
  - Sometimes, communication can be overlapped with useful work (implementation, algorithm):
    - Communication overhead may show a more fortunate behavior than $Nk$
    - "superlinear speedups": data size per CPU decreases with increasing CPU count $\rightarrow$ may fit into cache at large CPU counts

(c) RRZE 2013
Serial fraction $s$ may depend on

- **Program / algorithm**
  - Non-parallelizable part, e.g. recursive data setup
  - Non-parallelizable IO, e.g. reading input data
  - Communication structure
  - Load balancing (assumed so far: perfect balanced)
  - …

- **Computer hardware**
  - Processor: Cache effects & memory bandwidth effects
  - Parallel Library; Network capabilities; Parallel IO
  - …

**Determine $s$ "experimentally":**
Measure speedup and fit data to Amdahl’s law – but that could be more complicated than it seems…
Scalability data on modern multi-core systems

An example

1→2 cores on socket

1→2 sockets on node

Scaling across nodes

Chipset

Memory
Scalability data on modern multi-core systems

The scaling baseline

- Scalability presentations should be grouped according to the largest unit that the scaling is based on (the “scaling baseline”)

- Amdahl model with communication: Fit

  \[
  S(N) = \frac{1}{s + \frac{1-s}{N} + kN}
  \]

  to inter-node scalability numbers

  \(N = \# \text{ nodes, } >1\)
Application to “accelerated computing”

- SIMD, GPUs, Cell SPEs, FPGAs, just any optimization…
- Assume overall (serial, un-accelerated) runtime to be $T_s = s + p = 1$
- Assume $p$ can be accelerated and run $\alpha$ times faster. We neglect any additional cost (communication…)
- To get a speedup of $r\alpha$, how small must $s$ be? Solve for $s$:

$$r\alpha = \frac{1}{s + \frac{1-s}{\alpha}} \Rightarrow s = \frac{r^{-1} - 1}{\alpha - 1}$$

- At $\alpha=10$ and $r=0.9$ (for an overall speedup of 9), we get $s \approx 0.012$, i.e. you must accelerate 98.8% of serial runtime!
- Limited memory on accelerators may limit the achievable speedup