Parallel Programming and Heterogeneous Computing

A1 - Terminology

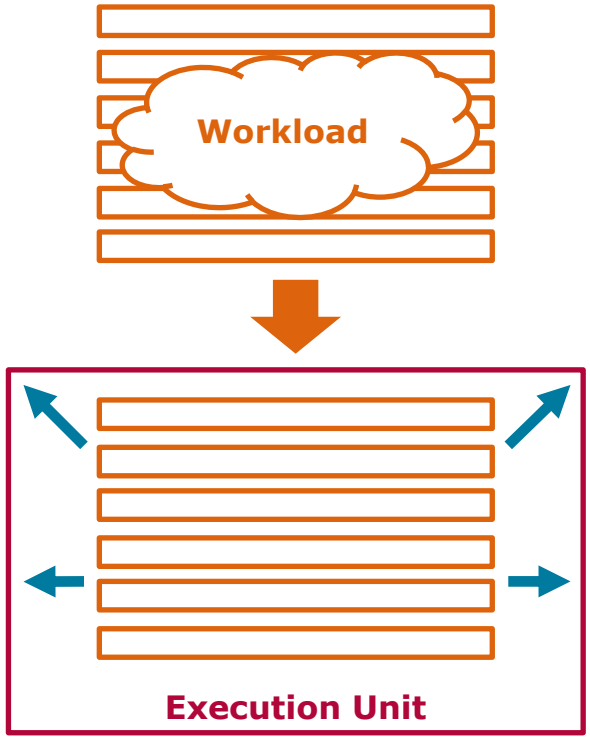
Max Plauth, Sven Köhler, Felix Eberhardt, Lukas Wenzel and Andreas Polze
Operating Systems and Middleware Group

So Many Words...

ParProg20 A1 Terminology

Lukas Wenzel

[Pfister1998]
Three Ways of Doing Things Faster



- Work Harder (execution capacity)

: Workload

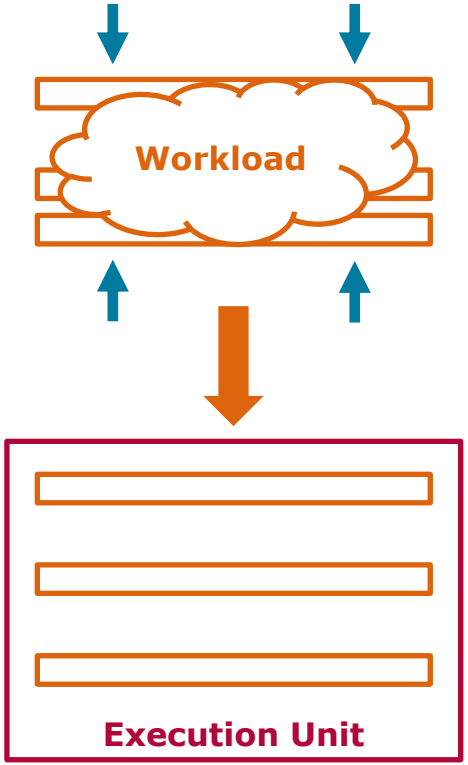
collection of operations that are executed to produce a desired result

~ Program, Application

: Execution Unit

facility that is capable of executing the operations of a workload

[Pfister1998]
Three Ways of Doing Things Faster



- Work Harder (execution capacity)
- Work Smarter (optimization)

: Workload

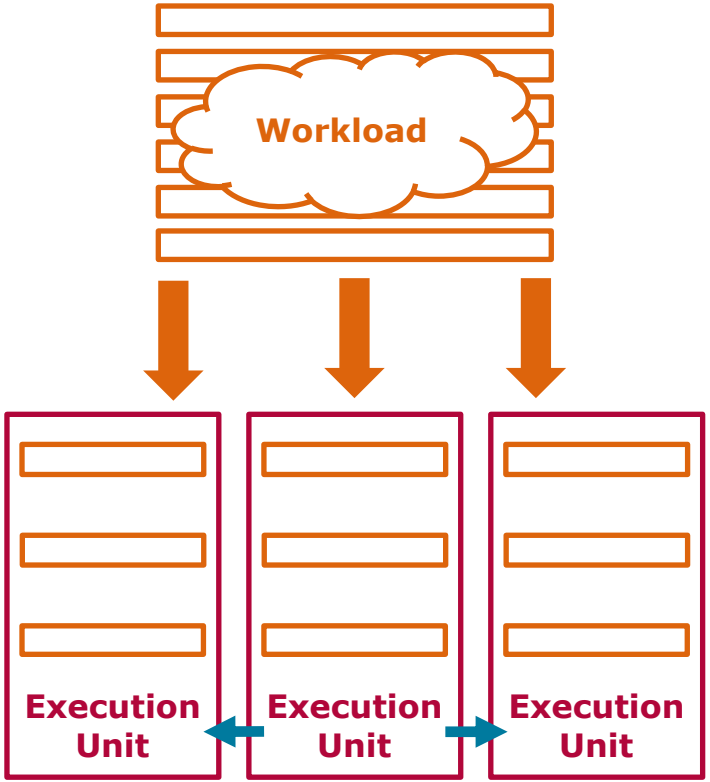
collection of operations that are executed to produce a desired result

~ Program, Application

: Execution Unit

facility that is capable of executing the operations of a workload

[Pfister1998]
Three Ways of Doing Things Faster



- Work Harder (execution capacity)
- Work Smarter (optimization)
- **Get Help (parallelization)**

: Workload

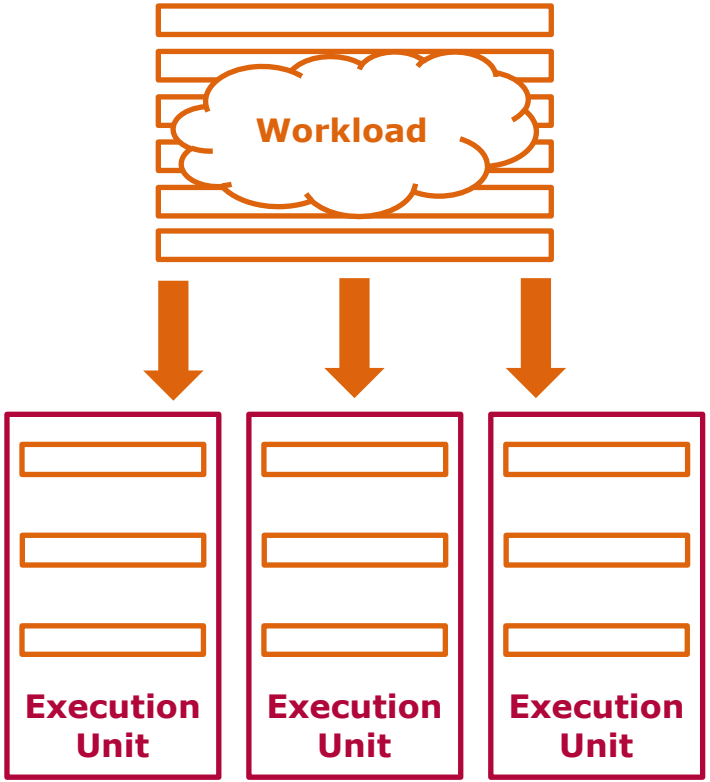
collection of operations that are executed to produce a desired result

~ Program, Application

: Execution Unit

facility that is capable of executing the operations of a workload

Parallelization



- Concept is simple to grasp
- Realization is more complex:
  - different languages
  - different execution environments
  - different patterns

- Parallelism is a **hardware property** that must be **exploited by software**

An Important Distinction

Concurrency

Capability of a machine to have multiple tasks in progress at any point in time

- Can be realized without parallel hardware

Parallelism

Capability of a machine to perform multiple tasks simultaneously

- Requires parallel hardware

: Parallelism

: Concurrency

: Distribution

Any parallel program is a concurrent program,

some concurrent programs cannot be executed correctly in parallel.

Distribution

Form of Parallelism, where tasks are performed by multiple communicating machines

Concurrency ⊃ Parallelism ⊃ Distribution

sometimes Concurrency \ Parallelism called "Concurrency"

Execution Model [Breshears2009]

A Workload is divided into tasks,
which represent independent operation sequences.

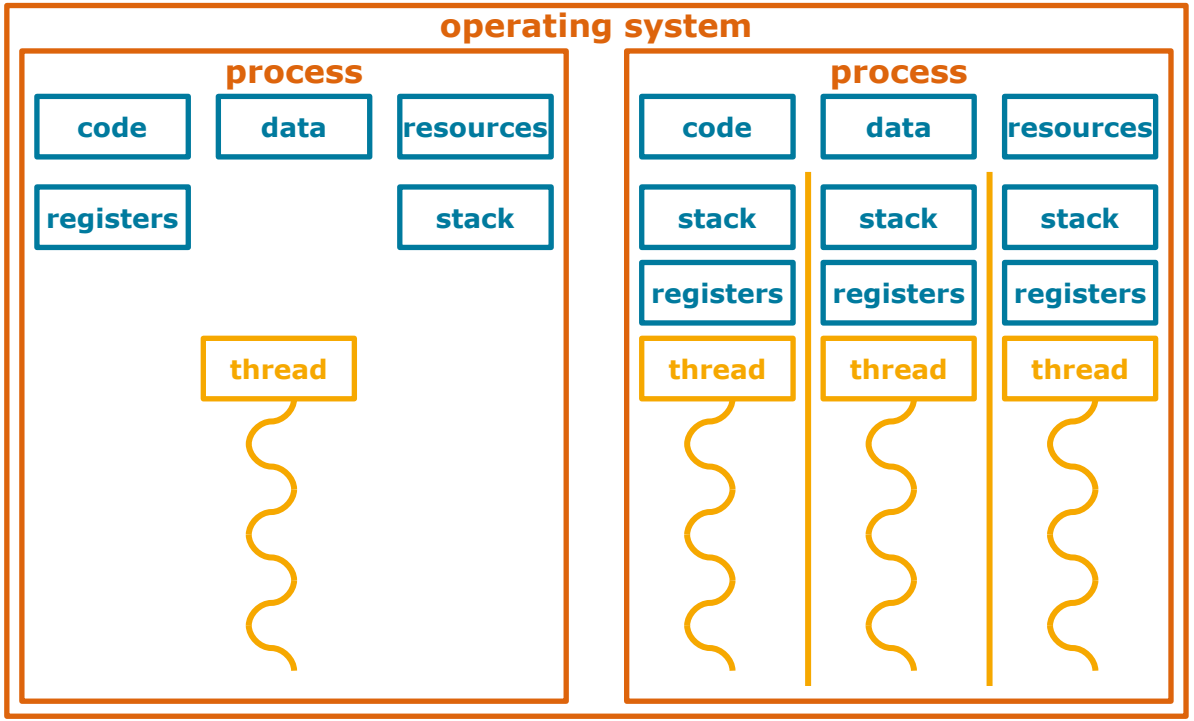
: Task

- Operations affect the *execution state* (private or shared)
- Operations are considered *atomic*:
  Effect of operation is applied either completely or not at all
- Operation granularity depends on execution environment:
  - usually machine instruction (real hardware)
  - sometimes source code line (some interpreters, theoretical models)

Excursion
Processes and Threads

Tasks can be executed by processes or threads,

which are operating system concepts.



Execution Model [Mattson2004]

- **Execution unit**
  - Tasks are executed by execution units
  - Mapping during development or at runtime

- **Processing element**
  - Hardware element running one execution unit
  - Depends on hardware scenario: logical processor, core, machine, ...
  - Multiple execution units may run on a single processing element, using a scheduling mechanism

: Execution Unit

: Processing Element

Execution Model

- **Concurrent execution**
  - = **Interleaving** of operations from multiple tasks (= **execution order**)
  - **Non-deterministic result** in general, because of unpredictable execution order
  - **Concurrent algorithms** produce the desired result for all execution orders

- **Parallel execution**
  - = **Overlapping** of operations from multiple tasks
  - Execution order is a *partial order*, as some operations happen neither before nor after others
  - Perceived execution order might change from different perspectives

Concurrent execution strictly encompasses parallel execution, but is more often used in contrast to signify interleaving without overlapping and the existence of a total execution order.

: Execution Order

Levels of Concurrency

In most systems, concurrency is present on multiple levels

- *Multiple processor cores execute different instruction streams in parallel*
- *Hardware Interrupts trigger service routines concurrently to regular code*
- *Operating system performs internal tasks (like page swapping, I/O handling) concurrently to applications*
- *Multiple applications run concurrently*
- *Application may use multiple concurrent threads*

Concurrency and Correctness

Tasks may access shared resources

(primarily memory locations,
also operating system facilities like communication channels,
files, consoles, network links, ...)

- Concurrent access to shared resources makes result sensitive to execution order

Race condition

= Only a subset of all possible execution orders leads to the desired result

- Well-known issue since the 1960's, identified by E. Dijkstra
- Difficult to detect, as incorrect executions though possible might happen rarely enough to escape notice.

: Shared Resources

∋ *execution state*

: Race Condition

Concurrency and Correctness

Tasks may synchronize

(i.e. change their behavior depending on the state of other tasks)

- Can be used to control the set of possible execution orders

Deadlock

= Multiple tasks wait for each other in a cyclic pattern

- No operations can be executed

Livelock

= Multiple tasks change their state in response to each other in a cyclic pattern

- Operations are executed, but no new execution states can be reached

: Synchronization

: Deadlock

: Livelock

Optimization Goals

- Decrease **Latency** – process a single workload faster (= **speedup**)
- Increase **Throughput** – process more workloads in the same time
- ➢ Both are **Performance** metrics

- **Scalability**: make best use of additional resources
  - **Scale Up**: Utilize additional resources on a machine
  - **Scale Out**: Utilize resources on additional machines
- **Cost/Energy Efficiency**:
  - minimize cost/energy requirements for given performance objectives
  - *alternatively: maximize performance for given cost/energy budget*
- **Utilization**: minimize idle time (=waste) of available resources
- **Precision-Tradeoffs**: trade performance for precision of results

: Latency

~ Response Time

: Throughput

: Scalability

: Efficiency

Parallel Programming

Parallel programming is a programming paradigm

- Can be realized by various **programming models**

Programming model

- = Application view of the execution environment
- May use different levels of abstraction
- May be influenced by underlying hardware and software layers or present independent features
- Implemented by **programming languages** and/or libraries and frameworks

Programming languages

- Consist of syntax/semantics and standard library, e.g. C & libc, C++ & STL
- Often determine only parts of the programming model, while other parts are chosen by libraries, frameworks or programmers

: Programming Paradigm

Abstract specification of features and structures common to multiple programming models.

(other definitions exist)

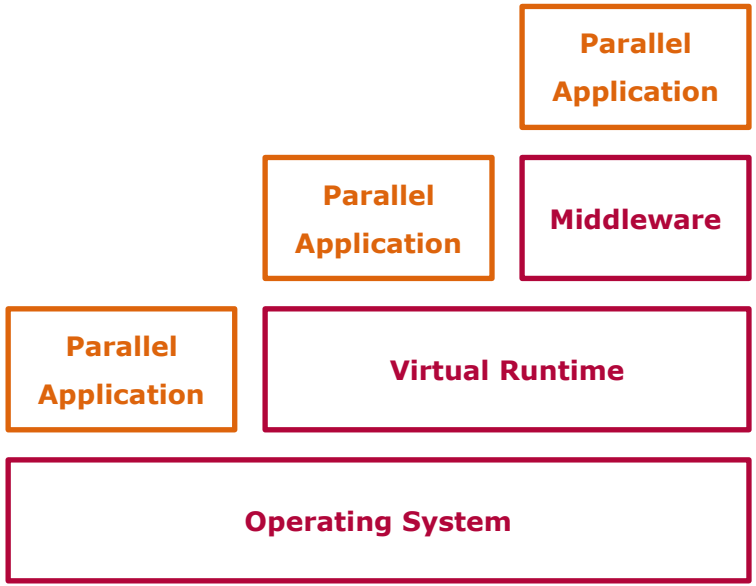
: Programming Model

: Programming Language

Parallel Programming

Parallel programming environments exist on different levels

- Middleware
  - *Apache Hadoop, ...*
- (Virtual) Runtime environments
  - *Java, .NET or C++ thread support*
- Programming languages
  - *Asynchronous and event-based programming*
- Operating systems
  - *Native processes, threads*
  - *Synchronization support*
- Bare-metal hardware
  - *Context switch support*
  - *Hardware thread IDs*



Classifications used in this course

Data-Parallel • Multi-Tasking • Message Passing • Implicit Parallelism

	Examples
Data-Parallel	OpenCL, CUDA
Multi-Tasking	Threads (PThreads, std::thread, ...), OpenMP
Message Passing	MPI (OpenMPI), Actors (Erlang, Scala), CSP (Go, occam)
Implicit Parallelism	Map/Reduce, Functional (Lisp, ...), HPF

Literature

[Pfister1998]

"In Search of Clusters" Pfister, Gregory F. 2nd edition. Prentice-Hall Inc. 1998

[Breshears2009]

"The Art of Concurrency" Breshears, Clay. O'Reilly Media Inc. 2009

[Mattson2004]

"Patterns for Parallel Programming" Matson, Timothy G and Sanders, Beverly and Massingill, Berna. Pearson Education. 2004

HPI Hasso Plattner Institut
Digital Engineering · Universität Potsdam

And now for a break and a cup of Earl Grey.

*or beverage of your choice