A New Look on Data Parallelism
Space vs. Time

Sven-Bodo Scholz

Kolloqium Programmiersprachen und Grundlagen der Programmierung
Maria Taferl, Österreich

14. Oktober 2009

Hardware Evolution — from a Customer Perspective

???

???

What we have

- academically studied for several decades
- affordable only by HPC labs with deep pockets
- programmed by experts
- the majority of applications use tailor-made MPI solutions!

$\Rightarrow$ Parallelisation is very hard!

Current (Industry) Approach:

It is expected that this will enable research to

- solve all unsolved parallelisation issues within a few years!
- cope with much more diverse hardware!
- cope with much more diverse applications!
- invent solutions applicable by general practitioners!

$\Rightarrow$ Opportunity / Obligation for programming language research to provide adequate tools!

The Dawn of a Software Revolution

- Many of the "old truths" do no longer hold!
  - Sequential Truth: redundant computations are evil!
  - Parallel Truth: redundant computation may reduce synchronisation!
  - Sequential Truth: excessive storage use is evil!
  - Parallel Truth: replication of data may eliminate communication!
  - ...

The Dawn of a Software Revolution

- Many of the "old truths" do no longer hold!
  - Sequential Truth: redundant computations are evil!
  - Parallel Truth: redundant computation may reduce synchronisation!
  - Sequential Truth: excessive storage use is evil!
  - Parallel Truth: replication of data may eliminate communication!
  - ...
- Depending on the target hardware we may need to shift between those!

The Dawn of a Software Revolution

- Many of the "old truths" do no longer hold!
  - Sequential Truth: redundant computations are evil!
  - Parallel Truth: redundant computation may reduce synchronisation!
  - Sequential Truth: excessive storage use is evil!
  - Parallel Truth: replication of data may eliminate communication!
  - ...
- Depending on the target hardware we may need to shift between those!

$\Rightarrow$ A declarative approach is needed!

Data-Parallelism

- Fundamental idea:

  Formulate Algorithms in terms of SPACE rather than TIME

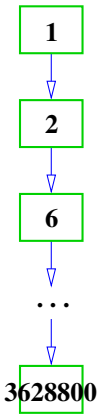
Data-Parallelism

- Fundamental idea:

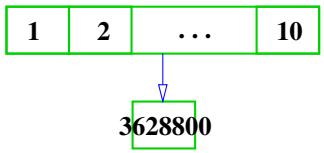
  Formulate Algorithms in terms of SPACE rather than TIME

- Example: factorial

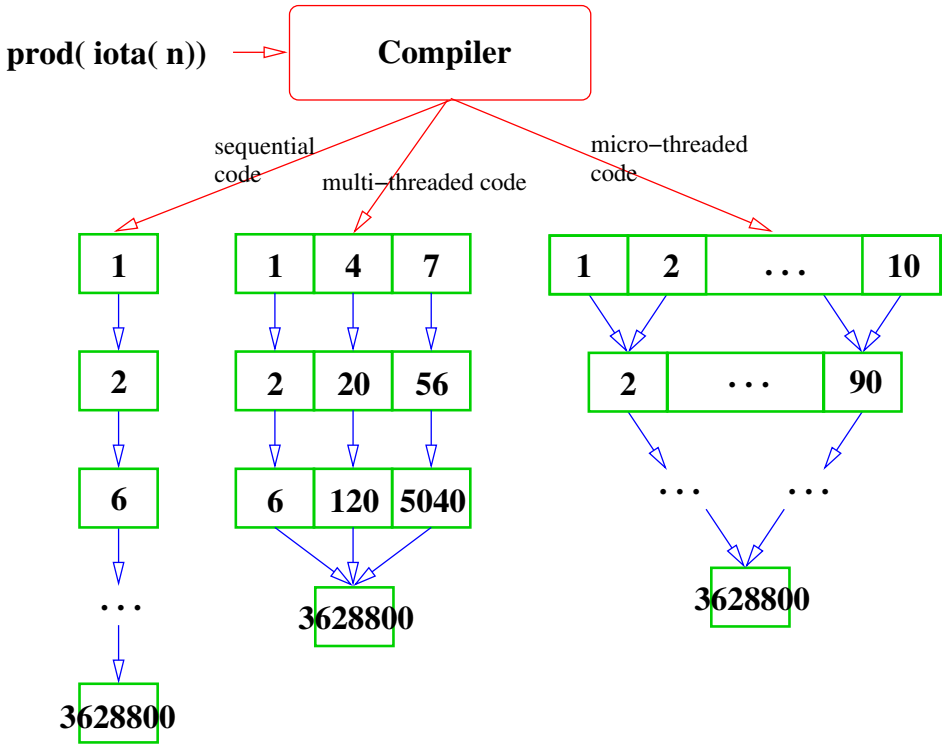
```
prod = 1;
for( i=1; i<=10; i++) {
  prod *= i;
}
```




```
prod( iota( 10))
```



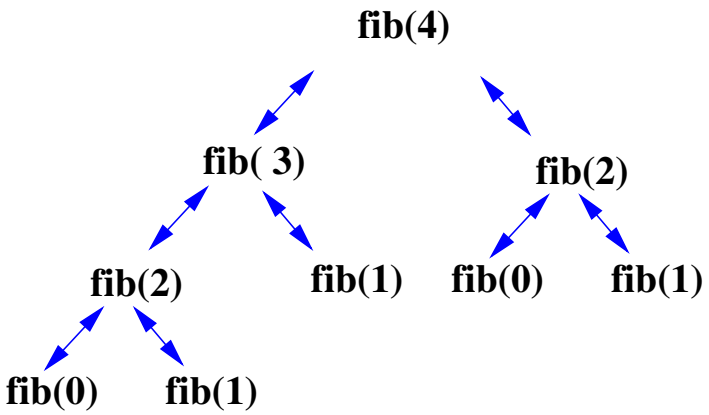
The Compilation Challenge — a first glimpse —



⇒ Different hardware architectures require different code generation strategies!

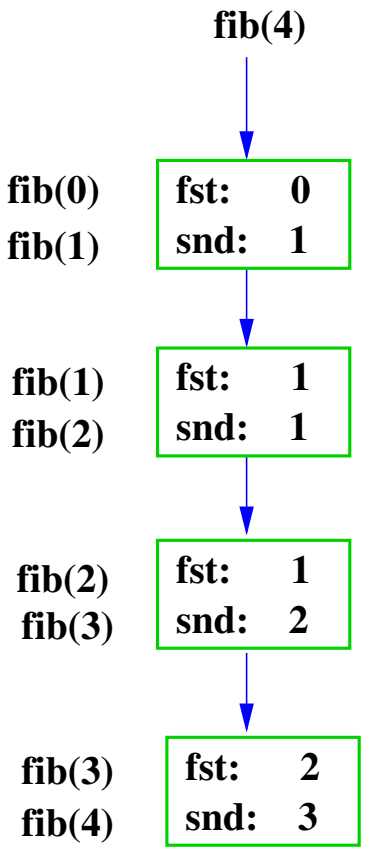
A second example: Fibonacci numbers

```
fib( 0) = 0
fib( 1) = 1
fib( n) = fib( n-1) + fib( n-2)
```



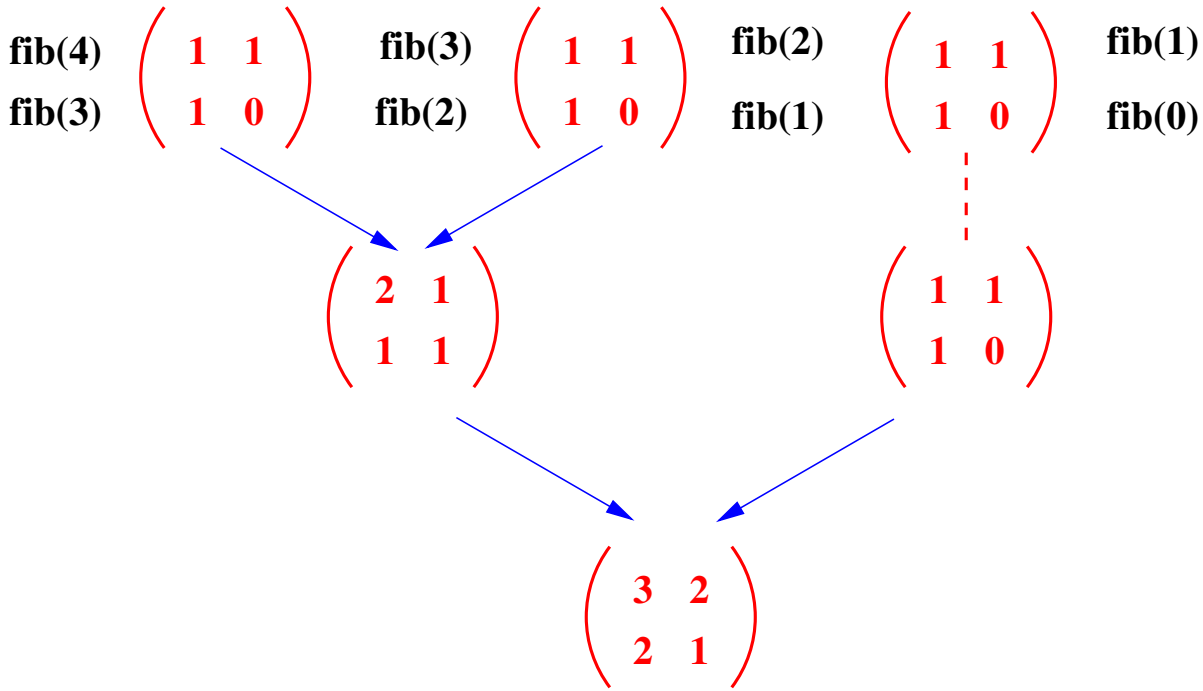
Fibonacci numbers: second attempt!

```
fib( n) = inner( 0, 1, n)
inner( fst, snd, 0) = fst
inner( fst, snd, n) = inner( snd, fst+snd, n-1)
```



Fibonacci numbers: now Data Parallel!

```
fib( n) = matprod( genarray( [n], [[1, 1], [1, 0]])) [0,0]
```



Credo

- Nobody wants to buy a new machine if he does not benefit in terms of performance!
- Hand-parallelising programs is just to hard!
- Multicores will enforce a software revolution!

Conclusions

- Data-Parallel Programmming is not just the ability to parallelise loops without dependencies!
- It encourages different program specifications where dependencies are expressed in space rather than time!
- Iterations are expressed as vectors / arrays!
- check it out!

`www.sac-home.org`