Toward space-time parallel simulations of phase-field models

December 4, 2017 | Robert Speck
Jülich Supercomputing Centre, Forschungszentrum Jülich GmbH

BMBF-Project ParaPhase

High-order space-time parallel phase-field simulations

- interface problems can be modeled with phase-field equations
- wide range of applications, e.g. fracture propagation in ceramics, drying soil etc.
- some key phenomena only emerge when the domain is large and simulation time long enough
- significant computing cost $\Rightarrow$ parallelism in space and time

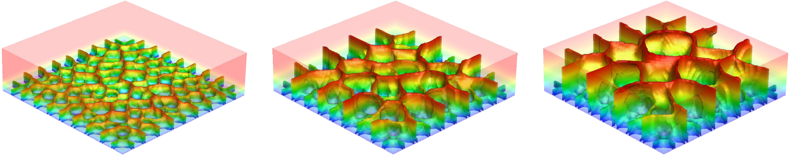


Figure: Modeling of fracture propagation in dry soil

BMBF-Project ParaPhase

High-order space-time parallel phase-field simulations

- interface problems can be modeled with phase-field equations
- wide range of applications, e.g. fracture propagation in ceramics, drying soil etc.
- some key phenomena only emerge when the domain is large and simulation time long enough
- significant computing cost $\Rightarrow$ parallelism in space and time

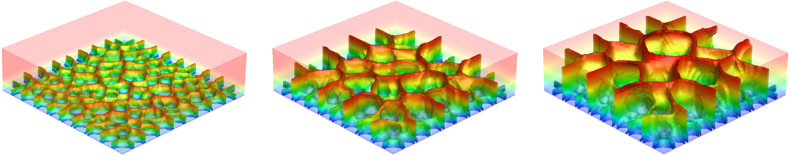


Figure: Modeling of fracture propagation in dry soil

The consortium

UNIVERSITÄT BAYREUTH

Uwe Glatzel
Applications

Freie Universität Berlin

Carsten Gräser
Numerics

TECHNISCHE UNIVERSITÄT DRESDEN

Oliver Sander
Numerics

JÜLICH FORSCHUNGSZENTRUM

Robert Speck
HPC

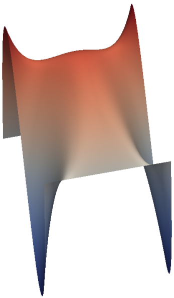
NVIDIA

Jiri Kraus
HPC

Universität Stuttgart

Marc-Andre Keip
Applications

A very simple toy problem (in space)



Model problem:

$$
\begin{aligned}
-\Delta u &= f \quad \text{on } \Omega := [0,1]^2 \\
u &= g \quad \text{on } \partial\Omega
\end{aligned}
$$

Standard approach: multigrid

- very efficient (mostly)
- supported by a lot of theory
- parallelizable

Parallel multigrid with DUNE

DUNE = Distributed and Unified Numerics Environment

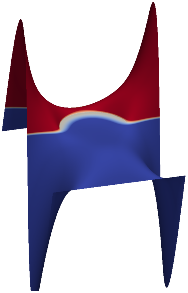
- modular C++ library for the solution of PDEs
- easy implementation of methods like Finite Elements
- slim interfaces, HPC-ready, easy to extend

Parallel multigrid with DUNE

- idea: let core modules handle the parallelization, algorithm should look (more or less) the same
- need: distributed data structures, reduction operations, communication
- allow overlapping and non-overlapping partitioning
- global index for each DoF

A less simple toy problem

Complexity $+=1$



Model problem:

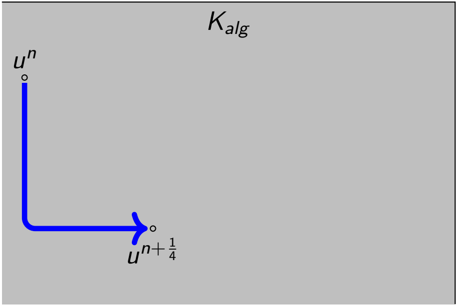
$$\begin{aligned} -\Delta u &= f && \text{on } \Omega := [0,1]^2 \\ u &\leq u_+ && \text{on } \Omega \\ u &= g && \text{on } \partial\Omega \end{aligned}$$

Left: toy problem with

$$u_+(x) = \begin{cases} 0.2 & ||x - x_0|| < r \\ \infty & \text{otherwise} \end{cases}$$

Multigrid for obstacle problems

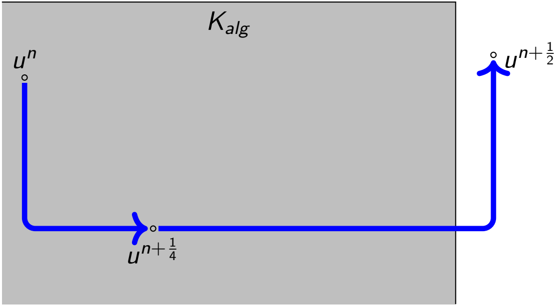
The TNNMG algorithm



- Pre-smoothing: projected Gauss-Seidel
- Linear multigrid step for reduced problem
- Project on K_{alg}
- Line search to ensure monotonicity

Multigrid for obstacle problems

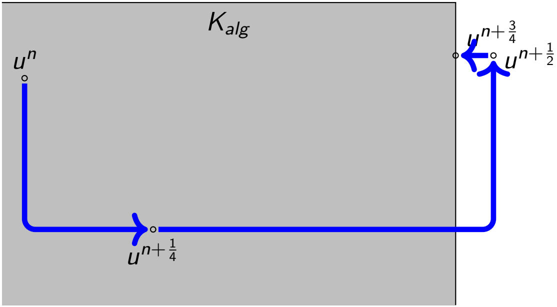
The TNNMG algorithm



- Pre-smoothing: projected Gauss-Seidel
- Linear multigrid step for reduced problem
- Project on K_{alg}
- Line search to ensure monotonicity

Multigrid for obstacle problems

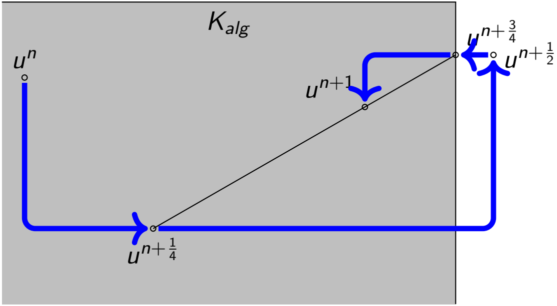
The TNNMG algorithm



- Pre-smoothing: projected Gauss-Seidel
- Linear multigrid step for reduced problem
- Project on K_{alg}
- Line search to ensure monotonicity

Multigrid for obstacle problems

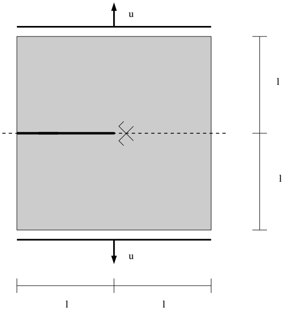
The TNNMG algorithm

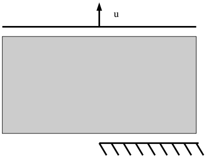


- Pre-smoothing: projected Gauss-Seidel
- Linear multigrid step for reduced problem
- Project on K_{alg}
- Line search to ensure monotonicity

Application: Notched domain under tension

A Boundary Value Problem

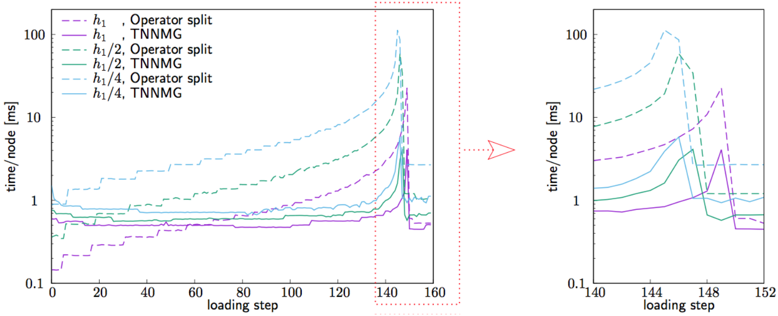




- Single-edge-notched domain under tensile loading
- Only upper half of the specimen is simulated (symmetry)
- Deformation by a linear increasing non-homogeneous Dirichlet boundary condition at the top

Application: Notched domain under tension

TNNMG vs. standard operator splitting



Adding time

Complexity $+= 1$

What if our problems are time-dependent?

- example: Allen-Cahn or Cahn-Hilliard equations: diffuse interface model for phase transition and separation phenomena
- TNNMG allows semi- or fully implicit time-stepping with provable convergence
- parallel TNNMG = parallelization in space for time-dependent phase-field problems

Adding time

Complexity $+=1$

What if our problems are time-dependent?

- example: Allen-Cahn or Cahn-Hilliard equations: diffuse interface model for phase transition and separation phenomena
- TNNMG allows semi- or fully implicit time-stepping with provable convergence
- parallel TNNMG = parallelization in space for time-dependent phase-field problems

Are we done, then?

Limits of purely spatial parallelization

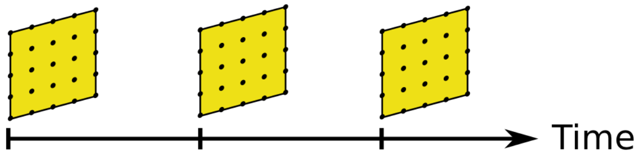


Figure: Time-stepping to solve time-dependent partial differential equations.

- Spatial parallelization reduces runtime **per time-step**
- Strong scaling saturates eventually because of communication
- Costs for **more time-steps** are not mitigated

Limits of purely spatial parallelization

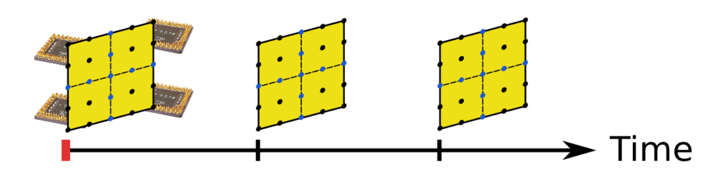


Figure: Time-stepping to solve time-dependent partial differential equations.

- Spatial parallelization reduces runtime **per time-step**
- Strong scaling saturates eventually because of communication
- Costs for **more time-steps** are not mitigated

Limits of purely spatial parallelization

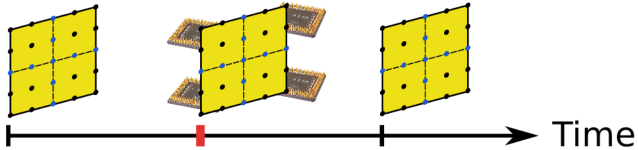


Figure: Time-stepping to solve time-dependent partial differential equations.

- Spatial parallelization reduces runtime **per time-step**
- Strong scaling saturates eventually because of communication
- Costs for **more time-steps** are not mitigated

Limits of purely spatial parallelization

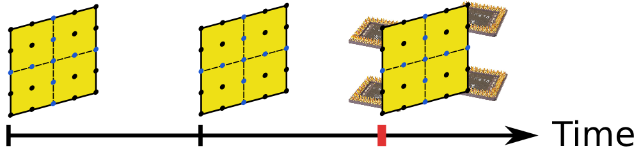


Figure: Time-stepping to solve time-dependent partial differential equations.

- Spatial parallelization reduces runtime **per time-step**
- Strong scaling saturates eventually because of communication
- Costs for **more time-steps** are not mitigated

Limits of purely spatial parallelization

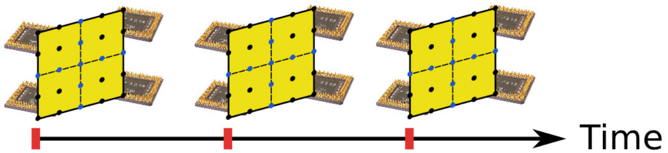


Figure: Time-stepping to solve time-dependent partial differential equations.

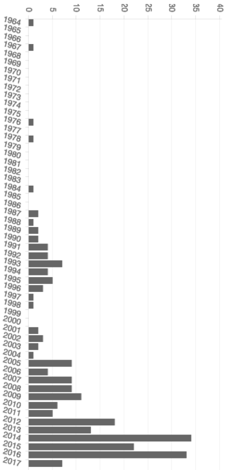
- Spatial parallelization reduces runtime **per time-step**
- Strong scaling saturates eventually because of communication
- Costs for **more time-steps** are not mitigated

→ Can we compute multiple time-steps simultaneously?

Parallel-in-Time (“PinT”) approaches

“50 years of parallel-in-time integration”, M. Gander (2015)

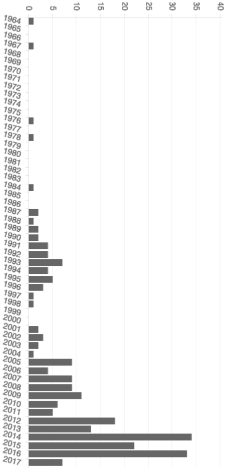
- Interpolation-based approach (Nievergelt 1964)
- Predictor-corrector approach (Miranker, Liniger 1967)
- Parabolic or time multi-grid (Hackbusch 1984) and (Horton 1992)
- Multiple shooting in time (Kiehl 1994)
- Parallel Runge-Kutta methods (e.g. Butcher 1997) and extrapolation (Richardson 1910)
- Parareal (Lions, Maday, Turinici 2001)
- PITA (Farhat, Chandesris 2003)
- Guided Simulations (Srinavasan, Chandra 2005)
- RIDC (Christlieb, Macdonald, Ong 2010)
- PFASST (Emmett, Minion 2012)
- MGRIT (Falgout et al 2014)



Parallel-in-Time (“PinT”) approaches

“50 years of parallel-in-time integration”, M. Gander (2015)

- Interpolation-based approach (Nievergelt 1964)
- Predictor-corrector approach (Miranker, Liniger 1967)
- Parabolic or time multi-grid (Hackbusch 1984) and (Horton 1992)
- Multiple shooting in time (Kiehl 1994)
- Parallel Runge-Kutta methods (e.g. Butcher 1997) and extrapolation (Richardson 1910)
- Parareal (Lions, Maday, Turinici 2001)
- PITA (Farhat, Chandesris 2003)
- Guided Simulations (Srinavasan, Chandra 2005)
- RIDC (Christlieb, Macdonald, Ong 2010)
- PFASST (Emmett, Minion 2012)
- MGRIT (Falgout et al 2014)



A quick algebraic introduction to PFASST

Basic building block: spectral deferred corrections (SDC)

Consider the Picard form of an initial value problem on $[T_n, T_{n+1}]$

$$u(t) = u_0 + \int_{T_n}^{t} f(u(s))ds,$$

discretized using spectral quadrature rules with nodes t_m:

$$u_m = u_0 + \Delta t QF(u) \approx u_0 + \int_{T_l}^{t_m} f(u(s))ds,$$

then SDC methods can be seen as (approximative) Gauss-Seidel iteration to solve this collocation problem for all u_m.

A quick algebraic introduction to PFASST

Basic building block: spectral deferred corrections (SDC)

Consider the Picard form of an initial value problem on $[T_n, T_{n+1}]$

$$u(t) = u_0 + \int_{T_n}^{t} f(u(s))ds,$$

discretized using spectral quadrature rules with nodes t_m:

$$(I - \Delta t QF)(\vec{u}) = \vec{u}_0$$

then SDC methods can be seen as (approximative) Gauss-Seidel iteration to solve this collocation problem for all u_m.

A quick algebraic introduction to PFASST

Basic building block: spectral deferred corrections (SDC)

Consider the Picard form of an initial value problem on $[T_n, T_{n+1}]$

$$u(t) = u_0 + \int_{T_n}^{t} f(u(s))ds,$$

discretized using spectral quadrature rules with nodes t_m:

$$(I - \Delta t Q F)(\vec{u}) = \vec{u}_0$$

then SDC methods can be seen as (approximative) Gauss-Seidel iteration to solve this collocation problem for all u_m.

$\Rightarrow$ Use this for block smoothing in space-time multigrid = PFASST

A quick algebraic introduction to PFASST

Parallel Full Approximation Scheme in Space and Time

We now glue L time-steps together, using N to transfer information from step l to step $l+1$. We get:

$$\begin{pmatrix} I - \Delta t QF & & & \\ -N & I - \Delta t QF & & \\ & \ddots & \ddots & \\ & & -N & I - \Delta t QF \end{pmatrix} \begin{pmatrix} \vec{u}_1 \\ \vec{u}_2 \\ \vdots \\ \vec{u}_L \end{pmatrix} = \begin{pmatrix} \vec{u}_0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

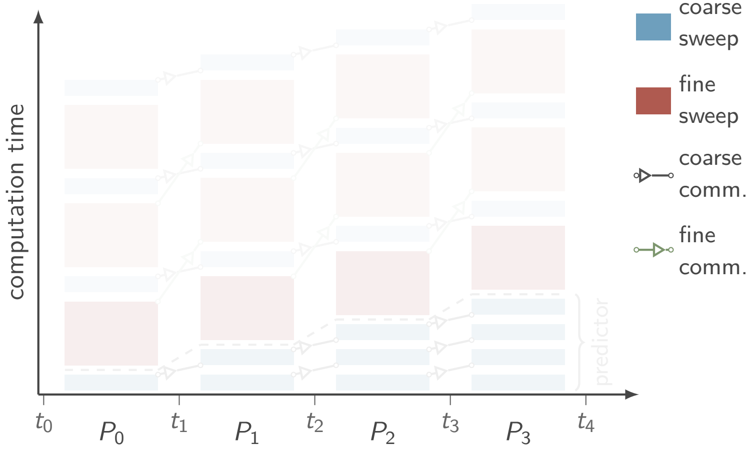
PFASST:

- use (linear/FAS) multigrid to solve this system iteratively
- fine-level smoother: parallel block Jacobi with SDC in the blocks
- coarse-level: serial block Gauß-Seidel with SDC in the blocks

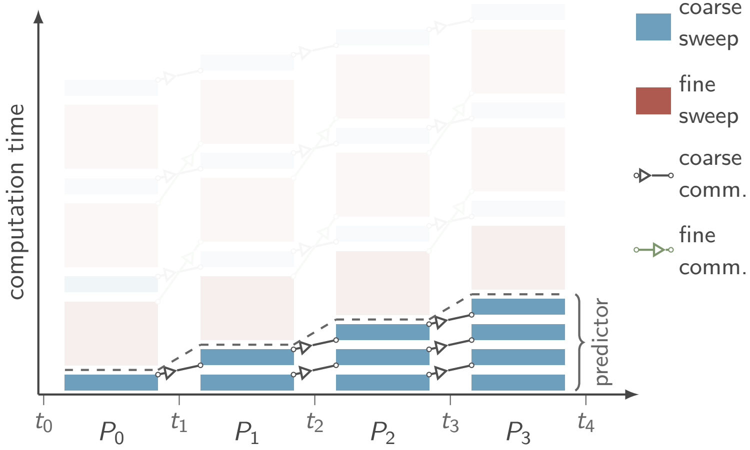
SPONSORED BY THE Federal Ministry of Education and Research

JÜLICH FORSCHUNGSZENTRUM

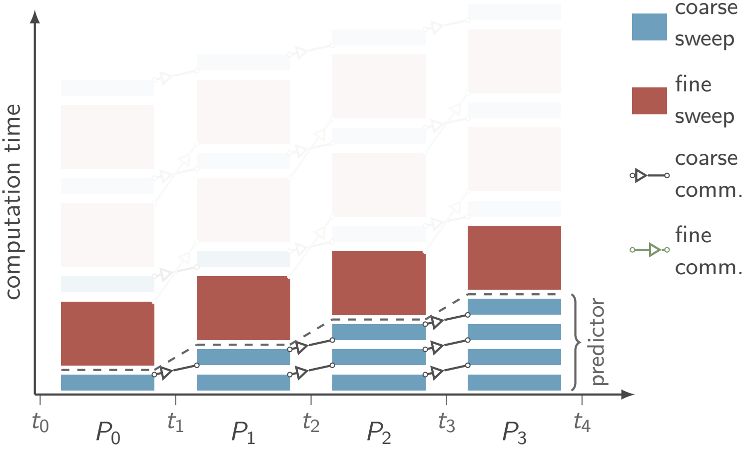
A quick visual introduction to PFASST



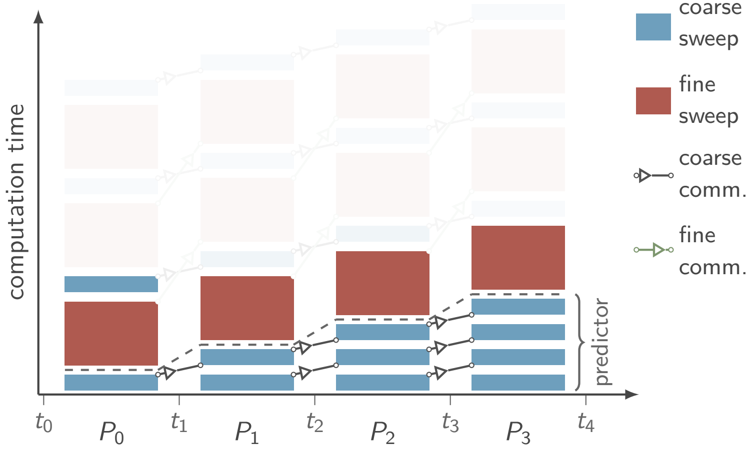
A quick visual introduction to PFASST



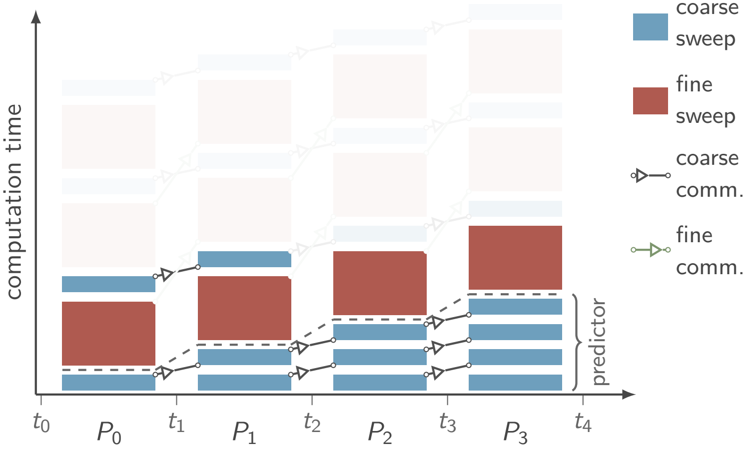
A quick visual introduction to PFASST



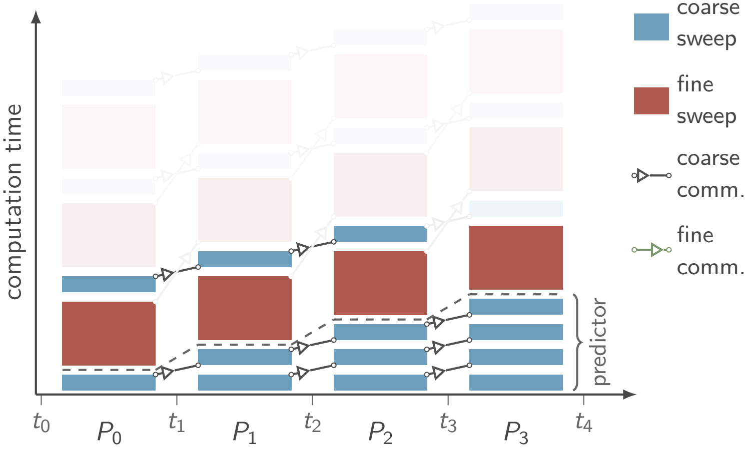
A quick visual introduction to PFASST



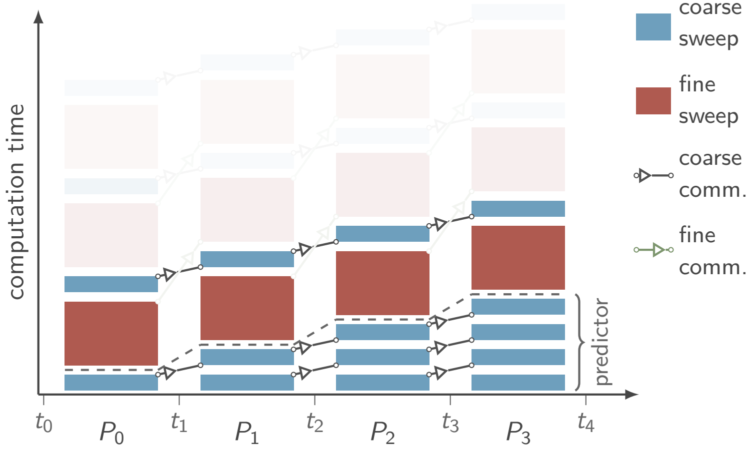
A quick visual introduction to PFASST



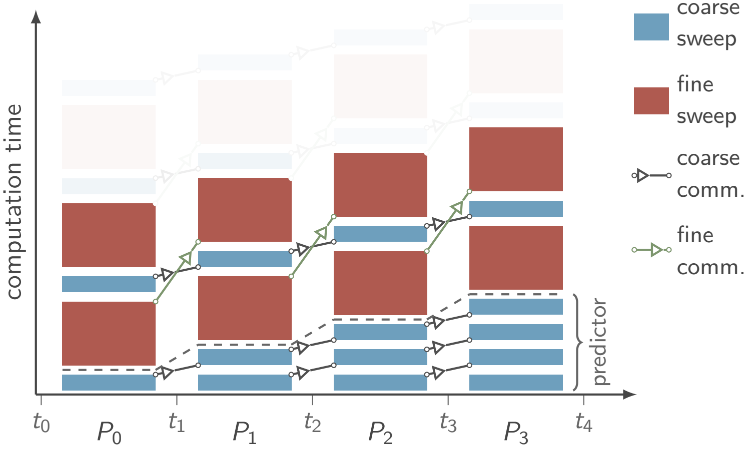
A quick visual introduction to PFASST



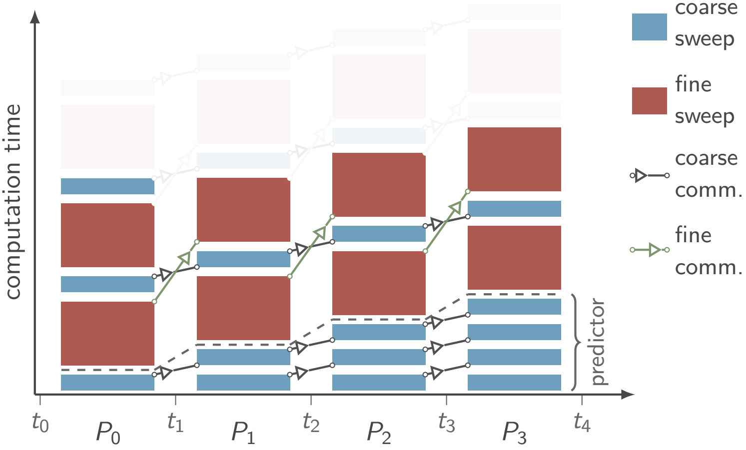
A quick visual introduction to PFASST



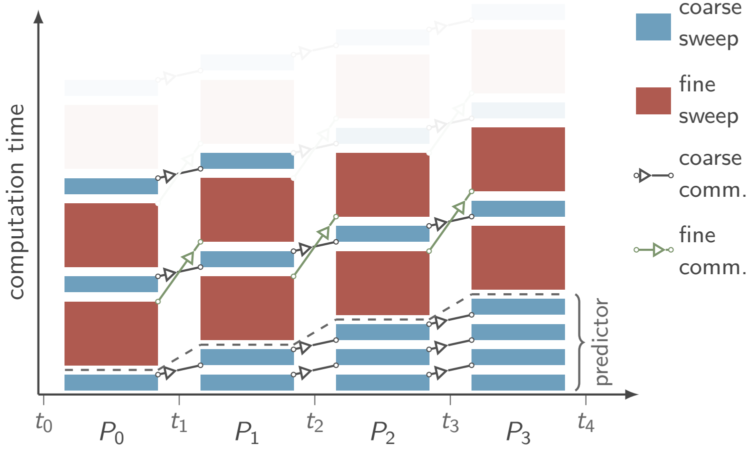
A quick visual introduction to PFASST



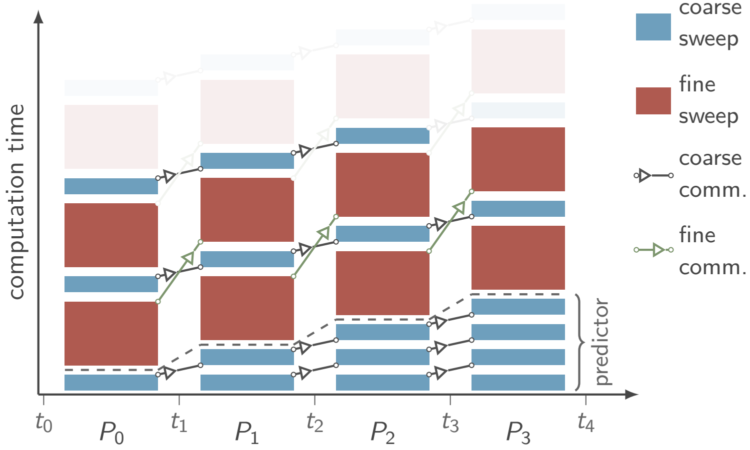
A quick visual introduction to PFASST



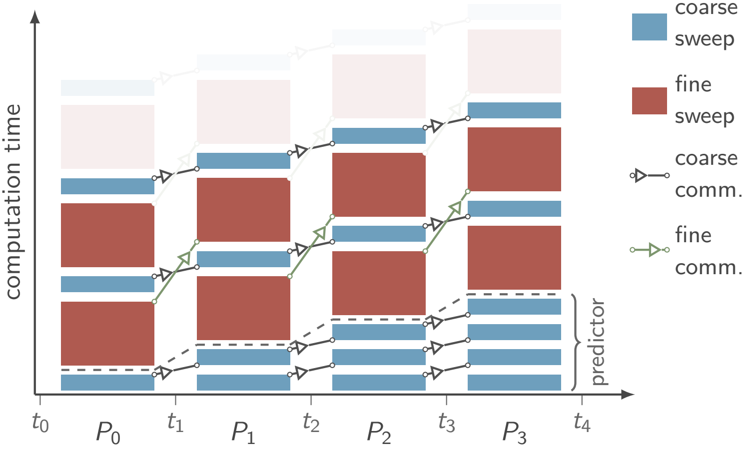
A quick visual introduction to PFASST



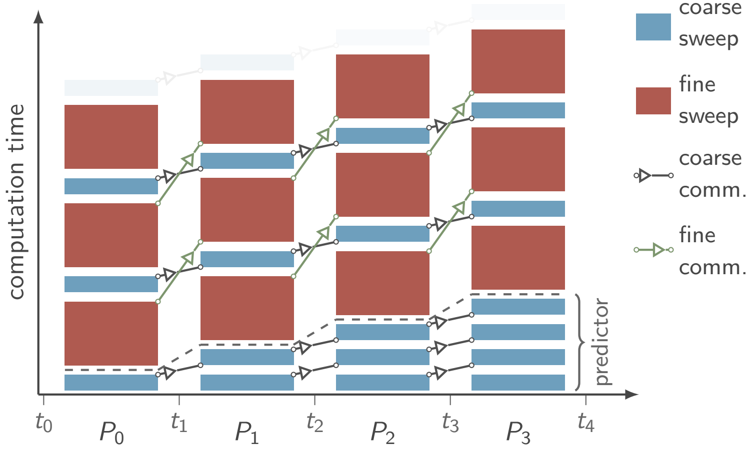
A quick visual introduction to PFASST



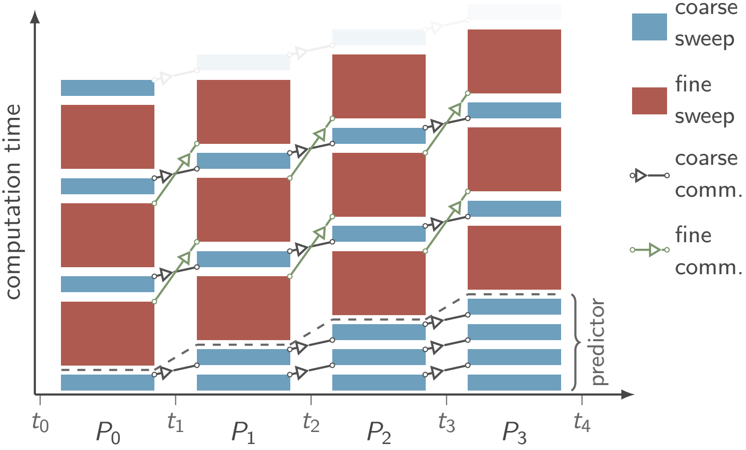
A quick visual introduction to PFASST



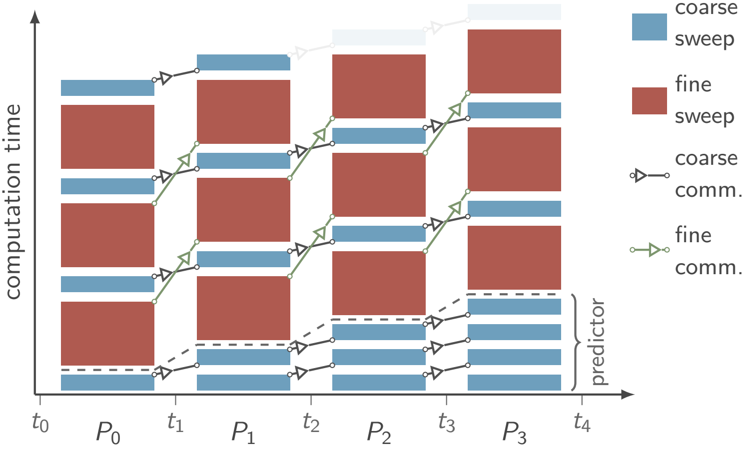
A quick visual introduction to PFASST



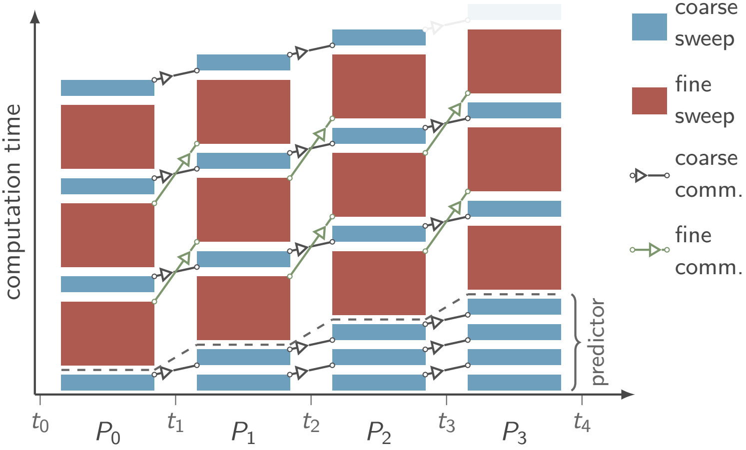
A quick visual introduction to PFASST



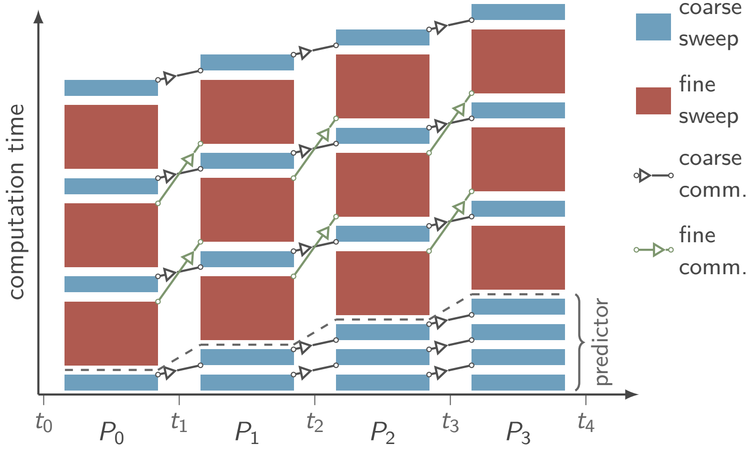
A quick visual introduction to PFASST



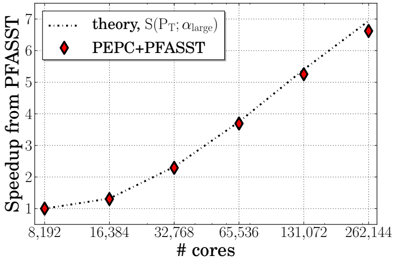
A quick visual introduction to PFASST



A quick visual introduction to PFASST

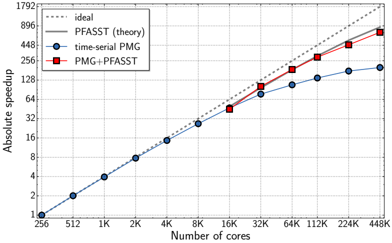


Examples: vortex particles and space-time multigrid



- vortex particles with Barnes-Hut tree code in space
- reduced force calculation on coarse level

→ 7x additional speedup



- 3D heat equation with multigrid solver in space
- coarsening via reduction of grid points and discretization order

→ 4x improved speedup

And phase-field problems?

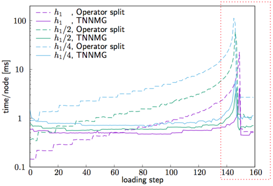
Done

- PFASST in time and finite elements in space ☑
- dune-PFASST: a DUNE module for parallel-in-time integration ☑
- PFASST for reaction diffusion problems, e.g. Allen-Cahn ☑
- coupling of PFASST in time with (serial) TNNMG in space ☑

To-do

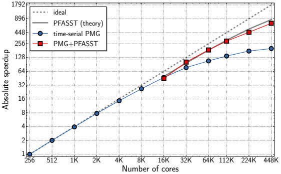
- coupling of PFASST with space-parallel solver ☐
- non-smooth problems ☐
- adaptivity in space and time ☐

Three takeaways



Parallel TNNMG method provides reliable and efficient solver for phase-field problems

Parallel-in-Time integration (PinT) can help to extend prevailing scaling limits



images/lego-pile.jpg

Space-time parallelization within **DUNE** leaves us with a ton of things to play with

The PinT Community

To learn more about PinT check out the website

`www.parallelintime.org`

and/or join one of the PinT Workshops, e.g.

7th Workshop on Parallel-in-Time Integration

- May 2-4, 2018
- Roscoff, France
- by Yvon Maday et al.

Also, there is a mailing list, join by writing to

`parallelintime+subscribe@googlegroups.com`

Toward space-time parallel simulations of phase-field models

December 4, 2017 | Robert Speck

Jülich Supercomputing Centre, Forschungszentrum Jülich GmbH