Fault-Tolerant Parallel-in-Time Integration

July 07, 2019 | Alba Troci & Robert Speck | Jülich Supercomputing Centre

Federal Ministry
of Education
and Research

JÜLICH
Forschungszentrum

Institution involved

Structural organization of Jülich Forschungzentrum

- Jülich Forschungszentrum
- Institute of Advance Simulations
- Jülich Supercomputing Center
- Department of Mathematics
- Parallel-in-Time Integration group (PinT)
  - Mathematical analysis of spectral deferred correction (SDC) and PFASST
  - Large-scale applications with PFASST on HPC systems
  - Algorithm based fault tolerance.

Motivation

As computers feature more and more hardware components, the probability of one component failing during a simulation increases.

Approach for dealing with faults:
Check pointing combined with a backward recovery strategy:
◇ here, the state of the system is frequently saved
◇ when a failure occur, the simulation is rolled back to the last correct state and restarted from there.

Drawbacks
◇ Very costly
◇ Often times even infeasible strategy, in particular on extreme scale machines.

Our approach for dealing with failures

Parallel-in-Time Integration:
Many algorithms in this field :

- Deal with node failure and bitflips in an efficient and rather non-intrusive way.
- Extend strong scaling limits of spatial parallelization and/or improve utilization of very large machines.

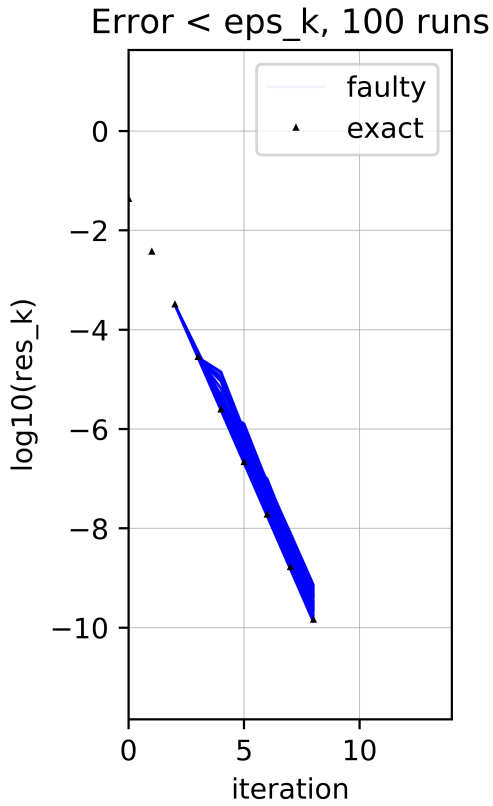
Existing methods:
"Parallel-across-the-steps" methods like Parareal or PFASST share features that make them natural candidates for algorithmic-based fault tolerance:

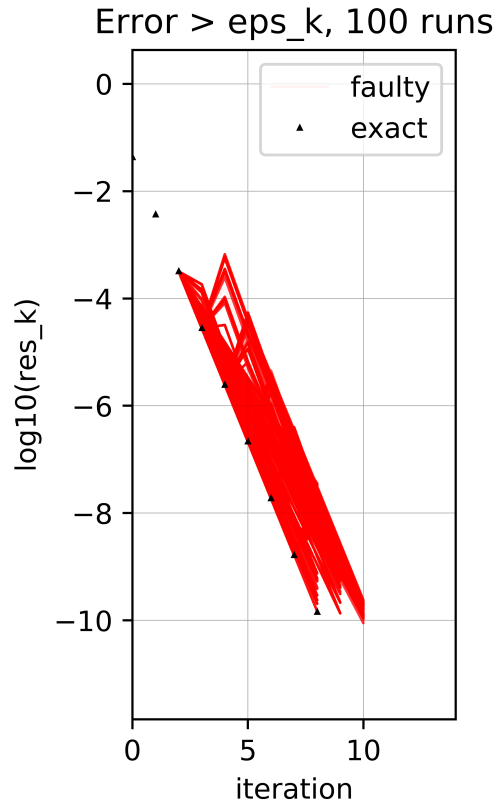
- they hold copies of the (approximate) solution at different times on different processes
- they are iterative as well as hierarchical.

What specifically am I doing?

Soft failures

- Find a bound on error for which the perturbed and non perturbed systems need the same number of iteration to converge.





What specifically am I doing?

Soft failures

- Detection strategies for errors larger than our bound, linear case
- Formula for the number of extra iterations after failure for convergence
  $\Rightarrow$ Knowing these two facts we can decide how many extra iterations to tolerate after failure.

What next?

- Detection strategies for nonlinear case
- Soft failures with PFASST algorithms.