Scalability and Algorithm-Based Fault Tolerance for Plasma Physics Simulations with GENE

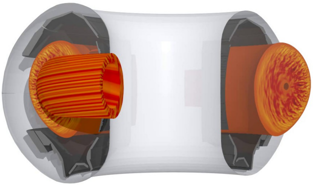
Big Data Meets Computation @ IPAM

Dirk Pflüger

Simulation of Large Systems, IPVS/SimTech, Universität Stuttgart

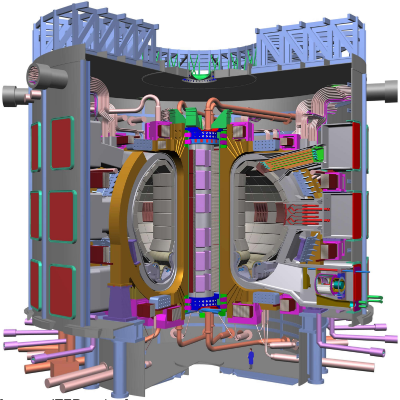
(joint work with M. Heene, A. Hinojosa)

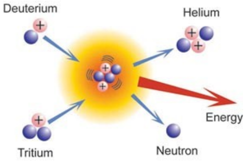
February 2, 2017



PDE: Turbulence simulations of hot fusion plasmas

- Idea: new, CO_2-free source of energy for the generations to come
- EXAHD with H.-J. Bungartz (TUM), M. Griebel (Bonn), T. Dannert (RZG), F. Jenko (UCLA)


[source: ITER project]



SPPEXA

Practically Unlimited Ressources

Contents:

- **Deuterium** in bath tub full of water and **Lithium** in used laptop battery suffice for family over 50 years

Behind the Scenes

- Dilute/hot plasmas are (almost) collisionless
- Not magneto-hydrodynamic, but kinect description (Vlasov):

$$\left[\frac{\partial}{\partial t} + \vec{v}\frac{\partial}{\partial \vec{x}} + \frac{q}{m}\left(E + \frac{\vec{v}}{c} \times B\right)\frac{\partial}{\partial \vec{v}}\right] f(\vec{x}, \vec{v}, t) = 0$$

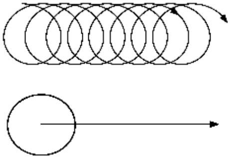
  - Distribution function $f(\vec{x}, \vec{v}, t)$
  - 6D in state space
  - Coupled to Maxwell equations

Behind the Scenes

- Dilute/hot plasmas are (almost) collisionless
- Not magneto-hydrodynamic, but kinect description (Vlasov):

$$\left[\frac{\partial}{\partial t} + \vec{v}\frac{\partial}{\partial \vec{x}} + \frac{q}{m}\left(E + \frac{\vec{v}}{c} \times B\right)\frac{\partial}{\partial \vec{v}}\right] f(\vec{x}, \vec{v}, t) = 0$$

  - Distribution function $f(\vec{x}, \vec{v}, t)$
  - 6D in state space
  - Coupled to Maxwell equations



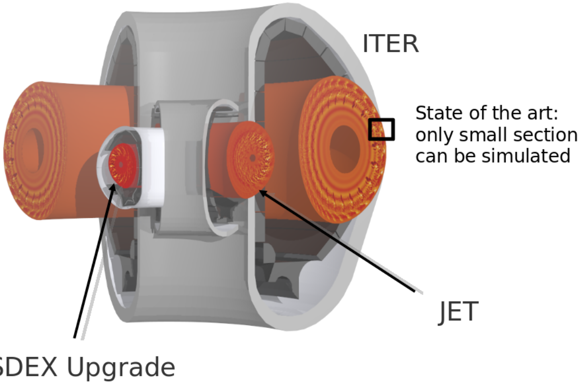
- Gyrokinetics: remove fast gyromotion (smallest scale)

$$\left[\frac{\partial}{\partial t} + \tilde{\vec{v}} \cdot \frac{\partial}{\partial \vec{x}} + \tilde{F}\frac{\partial}{\partial v_{||}}\right] f(\vec{x}, v_{||}, \mu, t) = \Delta(f)$$

  - 5D
  - $\tilde{\vec{v}}$ and $\tilde{F}$ are complex expressions, contain evaluation of E and B

Numerical Simulations for Actual Tokamaks with GENE

Aim: global simulations of ITER

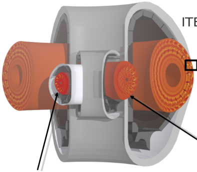


Gyrokinetic Electromagnetic Numerical Experiment

http://www.genecode.org

Numerical Simulations for Actual Tokamaks with GENE

Goal: global simulation with physical realism

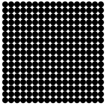


- Szenario for simulation of "numerical ITER"
  - Global, non-linear runs
  - At least 10^{11} grid points, 10^{6} time steps
  - $>$1 TB just to store single result in memory (complex)
- Possible at all?

Sparse Grids – Hierarchical Approach

- High-dimensional problems suffer "curse of dimensionality"
  - Effort $\mathcal{O}((2^n)^d)$ $\Rightarrow$ too Big Data

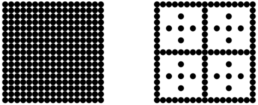
	full grid
5d, level 10	$> 10^{15}$



Sparse Grids – Hierarchical Approach

- High-dimensional problems suffer "curse of dimensionality"
  - Effort $\mathcal{O}((2^n)^d) \Rightarrow$ too Big Data
- Therefore: hierarchical discretization
  - Sparse grids: $\mathcal{O}(2^n \cdot n^{d-1})$ [Zenger 91]
  - Makes high-dimensional discretizations possible

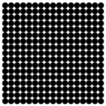
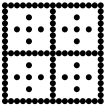
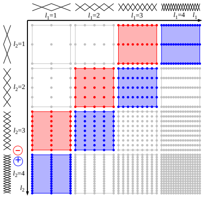
	full grid	sparse grid
5d, level 10	$> 10^{15}$	25,416,705



Sparse Grids – Hierarchical Approach

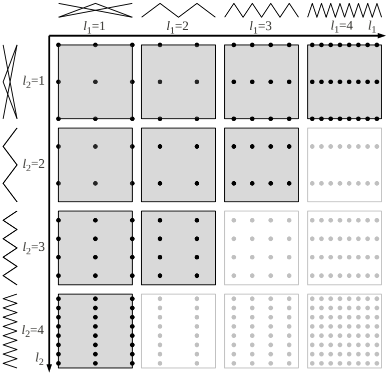
- High-dimensional problems suffer “curse of dimensionality”
  - Effort $\mathcal{O}((2^n)^d)$ $\Rightarrow$ too Big Data
- Therefore: hierarchical discretization
  - Sparse grids: $\mathcal{O}(2^n \cdot n^{d-1})$ [Zenger 91]
  - Makes high-dimensional discretizations possible

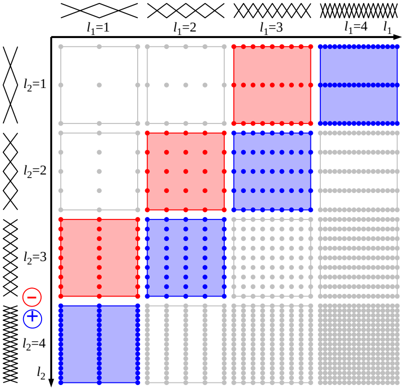
	full grid	sparse grid	sg combination technique
5d, level 10	$> 10^{15}$	25,416,705	1,876 $\times$ 82,000

- Combination technique (multivariate extrapolation-style scheme)
  - Multiple, but smaller grids: $\mathcal{O}(d \cdot n^{d-1})$ problems of size $\mathcal{O}(2^n)$

Sparse Grid vs. Combination Technique





Overview

Scalability

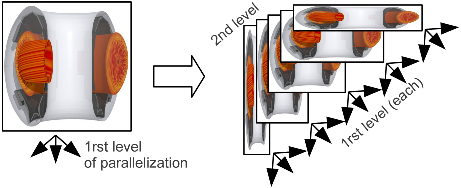
Problem of standard solver: global communication within each time-step

Scalability

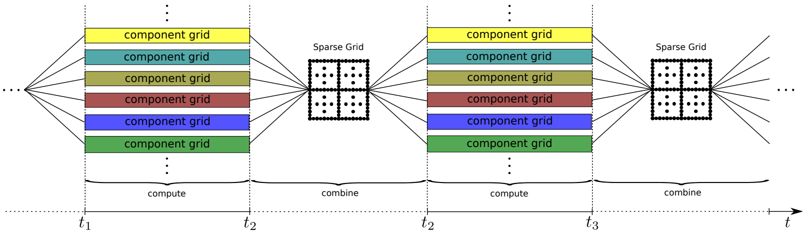
Problem of standard solver: global communication within each time-step

Use hierarchical ansatz

- Two-level approach
- Numerics: decoupling into locally coupled problems
- Algorithms: second level of parallelism
- First level: no need to scale to exascale



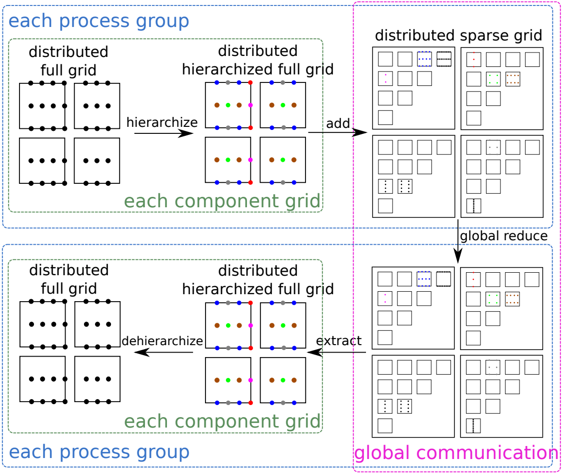
Time-Dependent PDEs



- Gather-scatter steps every time-interval
- Remaining reduced global communication

Global Communication

Optimal communication schemes



Global Communication

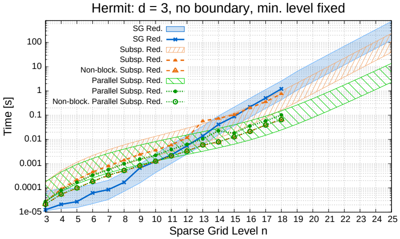
- Minimize number of communications (Range Query Trees):

$$\mathcal{O}(\log(dn^{d-1})) \times \mathcal{O}(2^n n^{d-1})$$

- Minimize package size

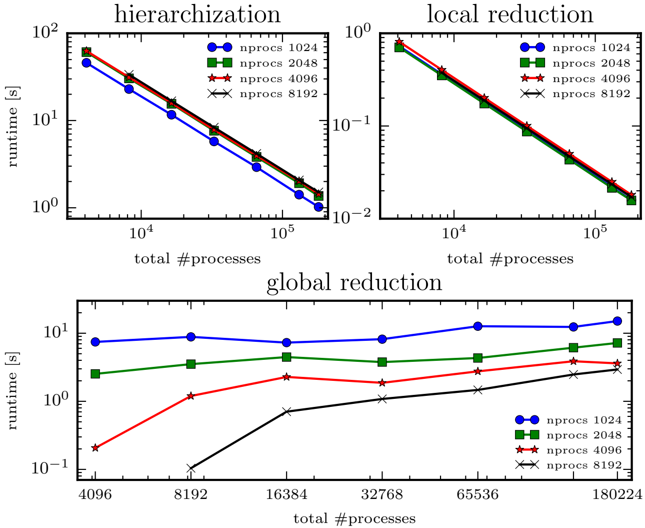
$$\mathcal{O}(2n \cdot n^{d-1}) \times \mathcal{O}(2^{n-1})$$

- Derivation in BSP/PEM model



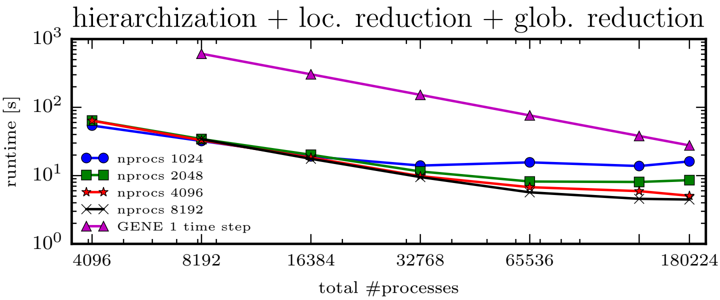
[joint work with R. Jacob (ITU, Algorithm Engineering)]

Runtimes on Hazel Hen



Runtimes on Hazel Hen

Total time



Overview

Resilience for the Exa-Age

Ever decreasing mean time between failure

- Massive replication of hardware
- Smaller scales (higher integration)
- Hardware possibly with less checks
- ...

Resilience for the Exa-Age

Ever decreasing mean time between failure

- Massive replication of hardware
- Smaller scales (higher integration)
- Hardware possibly with less checks
- ...

Two categories:

1. Hard faults
2. Soft/silent faults

Hard Faults

Errors that trigger signals to the user

- Node, OS, network or process failure
- Software crashes

$\Rightarrow$ Default MPI response: abort application

Hard Faults

Errors that trigger signals to the user

- Node, OS, network or process failure
- Software crashes

$\Rightarrow$ Default MPI response: abort application

Solutions

- Recompute (checkpoint-restart)
  - Checkpoint on HD / RAM
  - Lossless
  - Expensive storage/communication operations
  - Restart even more expensive

Hard Faults

Errors that trigger signals to the user

- Node, OS, network or process failure
- Software crashes

⇒ Default MPI response: abort application

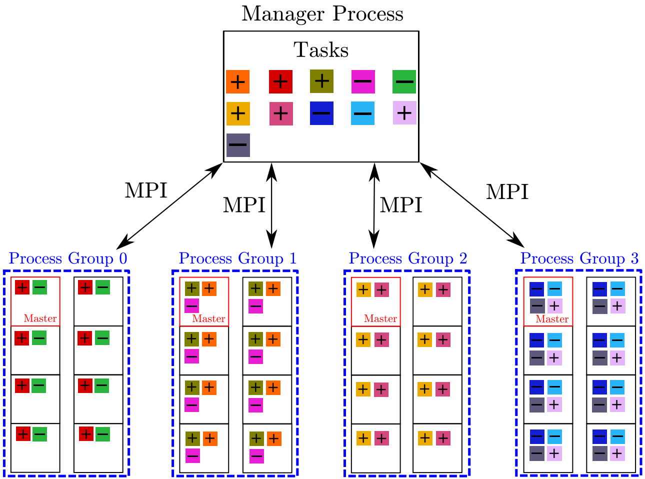
Solutions

- Recompute (checkpoint-restart)
  - Checkpoint on HD / RAM
  - Lossless
  - Expensive storage/communication operations
  - Restart even more expensive
- Continue w/o recomputation
  - Requires adapted numerical schemes
  - No/minor extra computational effort
  - Lossy

  ⇒ **algorithm-based fault-tolerance (ABFT)**

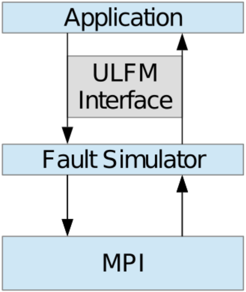
Communication Scheme

Master-worker model



Software Stack

- Fault simulation layer
- Implements interface of ULFM plus `kill_me()` functionality



Selective Reliability

- Focus on critical parts

Algorithm: The Combination Technique in Parallel

for all *combination grids* $\Omega_{\underline{i}}$ **do in parallel**
  $u_{\underline{i}} \leftarrow u(\underline{x}, t = 0)$; // Set initial conditions
while *not converged* **do**
  for all *combination grids* $\Omega_{\underline{i}}$ **do in parallel**
    $u_{\underline{i}} \leftarrow$ solver$(u_{\underline{i}}, N_t)$; // Solve the PDE on grid $\Omega_{\underline{i}}$ (N_t timesteps)
  $u_{\underline{n}}^{(c)} \leftarrow$ reduce$(c_{\underline{i}} u_{\underline{i}})$; // Combine solutions
  for all $\underline{i} \in \mathcal{I}_{\underline{n},q,\tau}$ **do**
    $u_{\underline{i}} \leftarrow$ scatter$(u_{\underline{n}}^{(c)})$; // Sample each $u_{\underline{i}}$ from new $u_{\underline{n}}^{(c)}$

Selective Reliability

- Focus on critical parts

Algorithm: The Combination Technique in Parallel

for all *combination grids* $\Omega_{\underline{i}}$ **do in parallel**
  $u_{\underline{i}} \leftarrow u(\underline{x}, t = 0)$; // Set initial conditions
while *not converged* **do**
  for all *combination grids* $\Omega_{\underline{i}}$ **do in parallel**
    $u_{\underline{i}} \leftarrow \texttt{solver}(u_{\underline{i}}, N_t)$; // Solve the PDE on grid $\Omega_{\underline{i}}$ (N_t timesteps)
  $u_{\underline{n}}^{(c)} \leftarrow \texttt{reduce}(c_{\underline{i}} u_{\underline{i}})$; // Combine solutions
  for all $\underline{i} \in \mathcal{I}_{\underline{n},q,\tau}$ **do**
    $u_{\underline{i}} \leftarrow \texttt{scatter}(u_{\underline{n}}^{(c)})$; // Sample each $u_{\underline{i}}$ from new $u_{\underline{n}}^{(c)}$

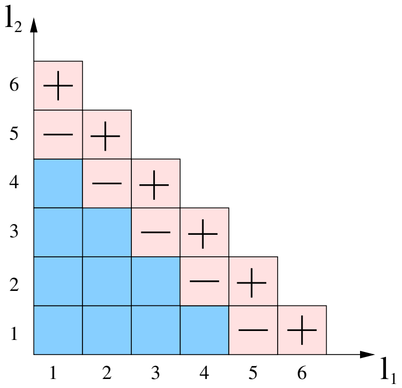
Selective Reliability

- Focus on critical parts

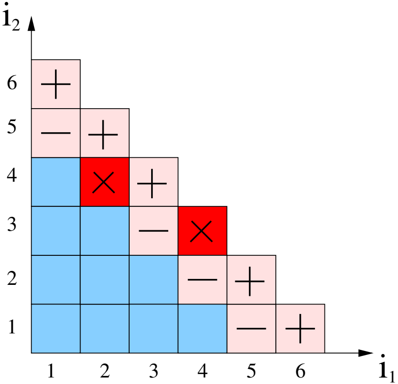
Algorithm: The Combination Technique in Parallel

for all *combination grids* $\Omega_{\underline{i}}$ **do in parallel**
    $u_{\underline{i}} \leftarrow u(\underline{x}, t = 0)$; // Set initial conditions
while *not converged* **do**
    for all *combination grids* $\Omega_{\underline{i}}$ **do in parallel**
        $u_{\underline{i}} \leftarrow$ `solver(`$u_{\underline{i}}$,N_t`);` // Solve the PDE on grid $\Omega_{\underline{i}}$ (N_t timesteps)
    `mitigateFaults();` // Mitigate faults
    $u_{\underline{n}}^{(c)} \leftarrow$ `reduce(`$c_{\underline{i}} u_{\underline{i}}$`);` // Combine solutions
    for all $\underline{i} \in \mathcal{I}_{\underline{n},q,\tau}$ **do**
        $u_{\underline{i}} \leftarrow$ `scatter(`$u_{\underline{n}}^{(c)}$`);` // Sample each $u_{\underline{i}}$ from new $u_{\underline{n}}^{(c)}$

2D Example



2D Example



ABFT: Fault-Tolerant Combination Technique

Find alternative combination, exclude missing solutions

- Starting point: standard CT coefficients

$$u_{\vec{n}}^{c}(\vec{x}) = \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{\vec{l} \in \mathcal{I}_{\vec{n},q}} u_{\vec{l}}(\vec{x})$$

ABFT: Fault-Tolerant Combination Technique

Find alternative combination, exclude missing solutions

- Starting point: standard CT coefficients

$$u_{\vec{n}}^{c}(\vec{x}) = \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{\vec{l} \in \mathcal{I}_{\vec{n},q}} u_{\vec{l}}(\vec{x})$$

In case of failure: use inclusion-exclusion principle to determine adapted combination

ABFT: Fault-Tolerant Combination Technique

Find alternative combination, exclude missing solutions

- Starting point: standard CT coefficients

$$u^c_{\vec{n}}(\vec{x}) = \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{\vec{l} \in \mathcal{I}_{\vec{n},q}} u_{\vec{l}}(\vec{x})$$

In case of failure: use inclusion-exclusion principle to determine adapted combination

1. Solve generalized coefficient problem (GCP):

$$\max_w Q'(w), \qquad Q'(w) := \sum_{l \in \mathrm{I}\downarrow} 4^{-\|i\|_1} w_l, \qquad \text{s.t. } w_l \in \{0,1\} \;\; \forall l \in \mathrm{I}\downarrow$$

ABFT: Fault-Tolerant Combination Technique

Find alternative combination, exclude missing solutions

- Starting point: standard CT coefficients

$$u^c_{\vec{n}}(\vec{x}) = \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{\vec{l} \in \mathcal{I}_{\vec{n},q}} u_{\vec{l}}(\vec{x})$$

In case of failure: use inclusion-exclusion principle to determine adapted combination

1. Solve generalized coefficient problem (GCP):

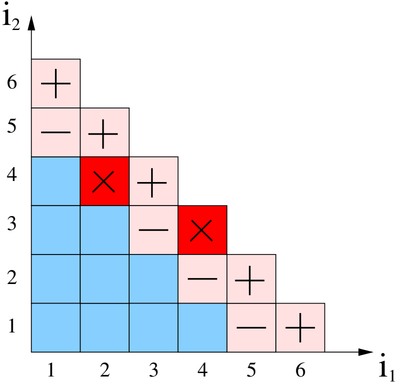
$$\max_{w} Q'(w), \qquad Q'(w) := \sum_{l \in \mathrm{I}\downarrow} 4^{-\|i\|_1} w_l, \qquad \text{s.t. } w_l \in \{0,1\} \;\; \forall l \in \mathrm{I}\downarrow$$

2. Obtain new combination coefficients:

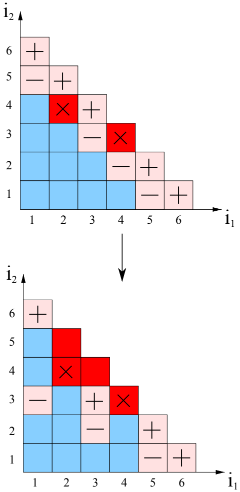
$$c_l = (M^{-1} w)_l$$

- Extra computations only on lower scales required

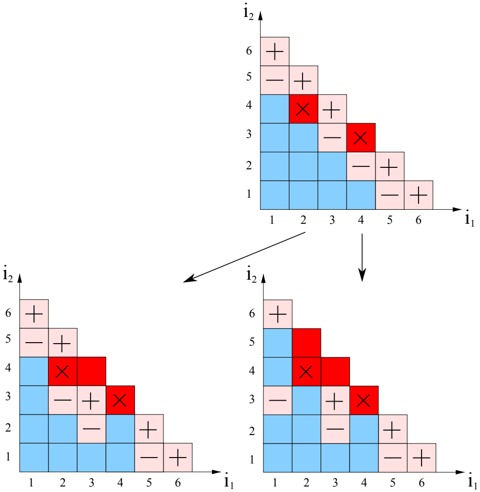
GCP: 2D Example



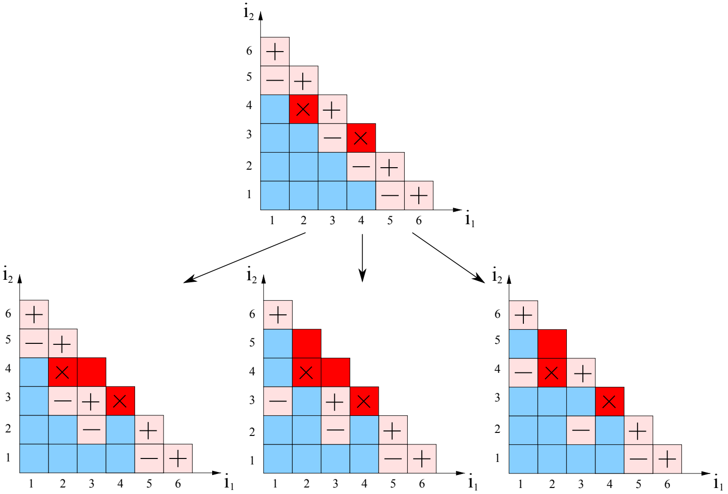
GCP: 2D Example



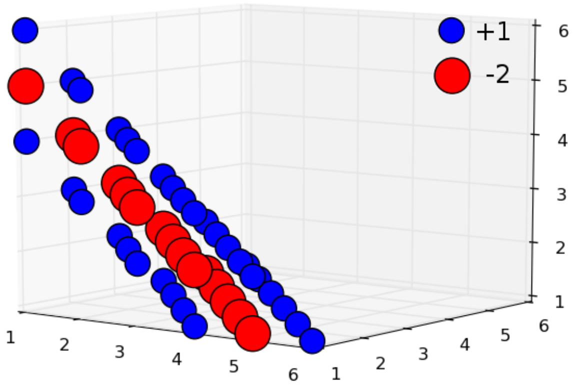
GCP: 2D Example



GCP: 2D Example

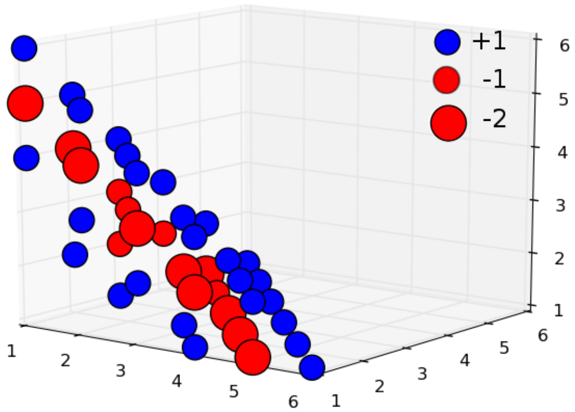


GCP: Example 3D



No faults

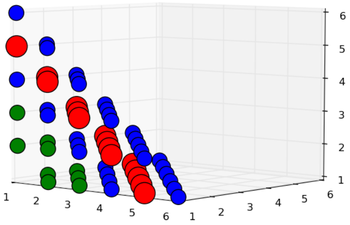
GCP: Example 3D

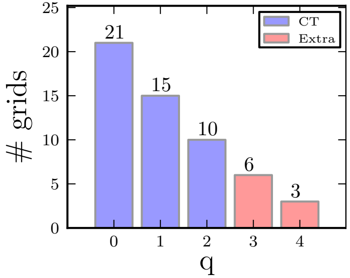


2 faults

GCP: Example 3D

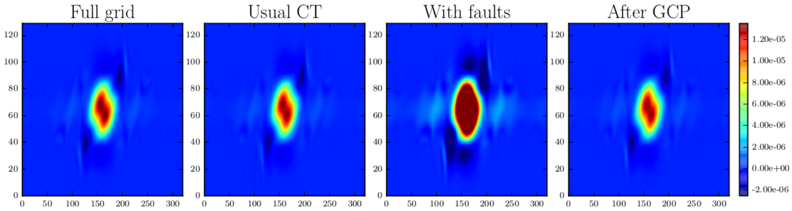
- For arbitrary faults: GCP prohibitively expensive
- Fast solution possible if enough extra grid layers available
- Only fraction of computational effort $\Rightarrow$ faults in lower layers unlikely
- Precompute some extra layers in advance





Results Using GENE

Example:

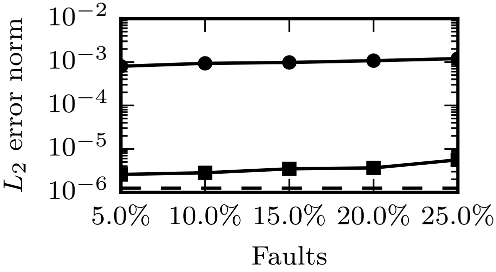


Good reconstruction (visual inspection)

Results Using GENE (2)

Small (reduced) problem

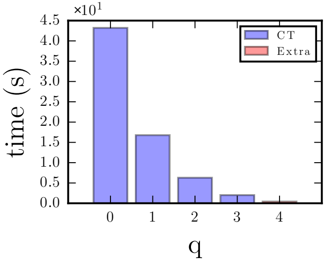
- 4D: $x, z, \mu, v_{\|}$
- $\vec{l}_{\min} = [2, 3, 2, 4]$, $\vec{l} = [6, 7, 6, 8] \Rightarrow$ 69 combination grids



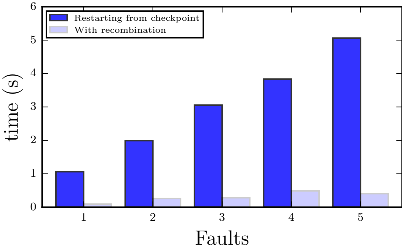
Excellent recovery properties!

Computational Effort

Accumulated time
to compute partial grids



Gain by ABFT



Significant savings in runtime

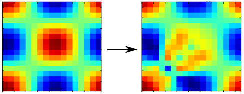
Silent/Soft Faults

No signal to user

- Faults unnoticed unless searched for
- Most common type: Silent Data Corruption (SDC)
  Errors in arithmetic operations, memory corruption, bit flips

1 0 1 0 1 1 1 0 1

1 0 1 0 0 1 1 0 1



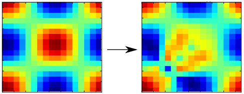
Silent/Soft Faults

No signal to user

- Faults unnoticed unless searched for
- Most common type: Silent Data Corruption (SDC)
  Errors in arithmetic operations, memory corruption, bit flips

1	0	1	0	1	1	1	0	1

1	0	1	0	0	1	1	0	1



Common solutions

- Checksums
- Replication (process/data)

⇒ Significant overhead (effort, resources)

Selective Reliability

- Focus on critical parts

Algorithm: The Combination Technique in Parallel

for all *combination grids* $\Omega_{\underline{i}}$ **do in parallel**
$\quad u_{\underline{i}} \leftarrow u(\underline{x}, t = 0)$; // Set initial conditions
while *not converged* **do**
$\quad$ **for all** *combination grids* $\Omega_{\underline{i}}$ **do in parallel**
$\quad\quad u_{\underline{i}} \leftarrow$ `solver`$(u_{\underline{i}}, N_t)$; // Solve the PDE on grid $\Omega_{\underline{i}}$ (N_t timesteps)
$\quad$ `mitigateFaults();` // Mitigate faults
$\quad u_{\underline{n}}^{(c)} \leftarrow$ `reduce`$(c_{\underline{i}} u_{\underline{i}})$; // Combine solutions
$\quad$ **for all** $\underline{i} \in \mathcal{I}_{\underline{n},q,\tau}$ **do**
$\quad\quad u_{\underline{i}} \leftarrow$ `scatter`$(u_{\underline{n}}^{(c)})$; // Sample each $u_{\underline{i}}$ from new $u_{\underline{n}}^{(c)}$

Selective Reliability

- Focus on critical parts

Algorithm: The Combination Technique in Parallel

for all *combination grids* $\Omega_{\underline{i}}$ **do in parallel**

  $u_{\underline{i}} \leftarrow u(\underline{x}, t = 0)$; // Set initial conditions

while *not converged* **do**

  for all *combination grids* $\Omega_{\underline{i}}$ **do in parallel**

    $u_{\underline{i}} \leftarrow$ `solver(`$u_{\underline{i}}$,N_t`);` // Solve the PDE on grid $\Omega_{\underline{i}}$ (N_t timesteps)

  `checkForSDC();` // Cheap sanity check

  `mitigateFaults();` // Mitigate faults

  $u_{\underline{n}}^{(c)} \leftarrow$ `reduce(`$c_{\underline{i}} u_{\underline{i}}$`);` // Combine solutions

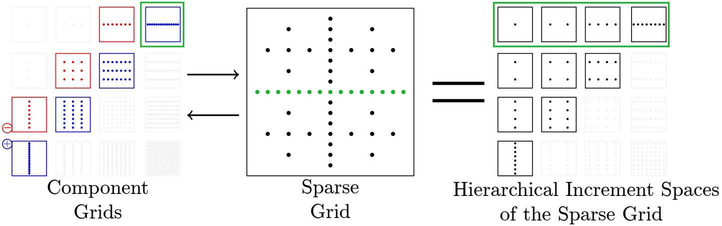
  for all $\underline{i} \in \mathcal{I}_{\underline{n},q,\tau}$ **do**

    $u_{\underline{i}} \leftarrow$ `scatter(`$u_{\underline{n}}^{(c)}$`);` // Sample each $u_{\underline{i}}$ from new $u_{\underline{n}}^{(c)}$

Silent/Soft Faults

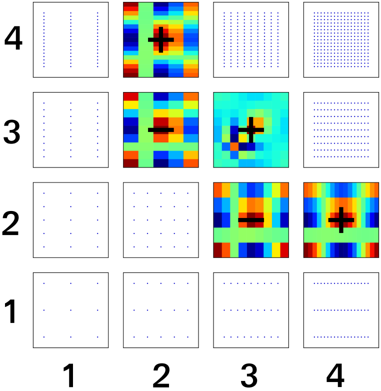
Exploit hierarchical approach

- Similar discretizations lead to similar results
- Exploit redundancy and hierarchical representation to check for faults
- Detection of outliers possible
- Direct integration into communication schemes possible (Subspace Reduce)



SDC Check: Compare Pairs of Solutions

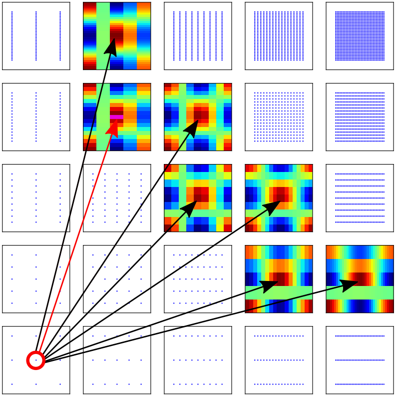
- Similar discretizations should lead to similar results



Pair	$\hat{\beta}^{(\underline{s},\underline{t})}$
(2, 4) (2, 3)	3.98e-01
(3, 2) (2, 4)	1.11e+00
(4, 2) (2, 3)	1.11e+00
(3, 2) (2, 3)	6.32e-01
(3, 3) (4, 2)	**9.85e+05**
(3, 3) (2, 3)	**1.07e+06**
(3, 2) (4, 2)	3.98e-01
(2, 4) (4, 2)	1.27e+00
(3, 2) (3, 3)	**1.07e+06**
(2, 4) (3, 3)	**9.85e+05**

$$\hat{\beta}^{(\underline{s},\underline{t})} := \max_{\underline{l}\leq\underline{s}\wedge\underline{t}} \; \max_{\underline{j}\in\mathcal{I}_{\underline{l}}} \frac{\left|\alpha^{(\underline{t})}_{\underline{l},\underline{j}} - \alpha^{(\underline{s})}_{\underline{l},\underline{j}}\right|}{\min\left\{\left|\alpha^{(\underline{t})}_{\underline{l},\underline{j}}\right|, \left|\alpha^{(\underline{s})}_{\underline{l},\underline{j}}\right|\right\}}$$

SDC Check: Outlier detection



$$\bar{u}(0,0) = [1.002, \mathbf{5.356}, 0.998, 1.002, 1.001, 1.001, .999]$$

2D Example

Advection equation

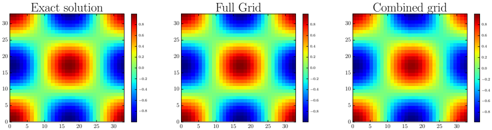
$$\frac{\partial u}{\partial t} + c_x \frac{\partial u}{\partial x} + c_y \frac{\partial u}{\partial x} = 0 \quad \Omega = [0,1]^2$$

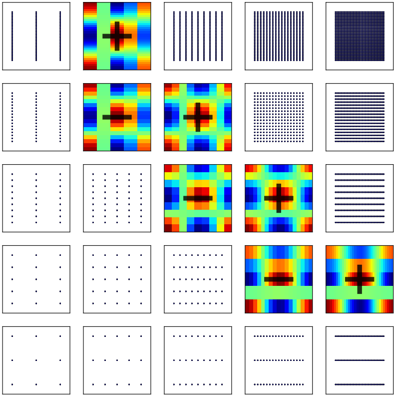
- Periodic boundary conditions
- Constant advection velocities c_x, c_y
- Initial condition $u(x, y, t = 0) = \sin(2\pi x)\sin(2\pi y)$
- Lax-Wendroff scheme (2nd order space + time)
- Error/solution at $t = 0.5$ compared to analytical solution

$$u(x, y, t) = \sin(2\pi(x - c_x t))\sin(2\pi(y - c_Y t))$$

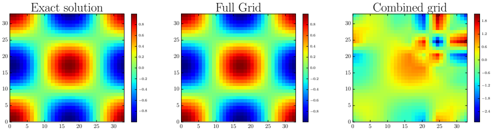
- Corruption of one single data point in initial condition

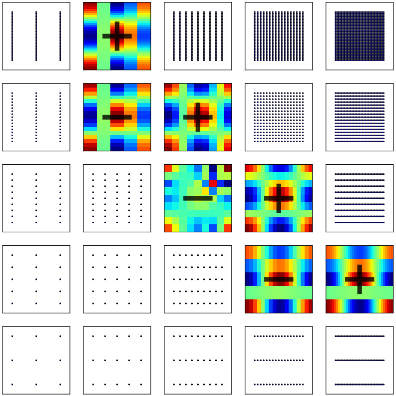
2D Example



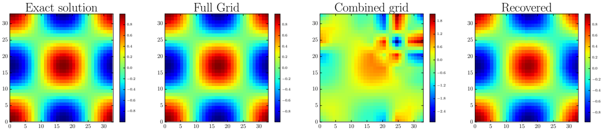


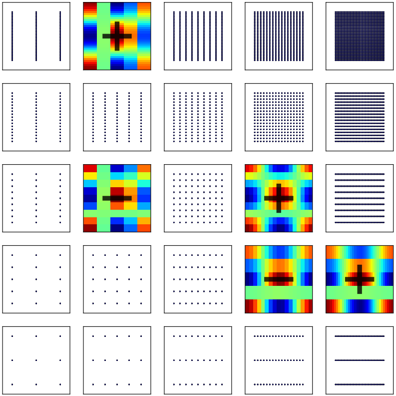
2D Example





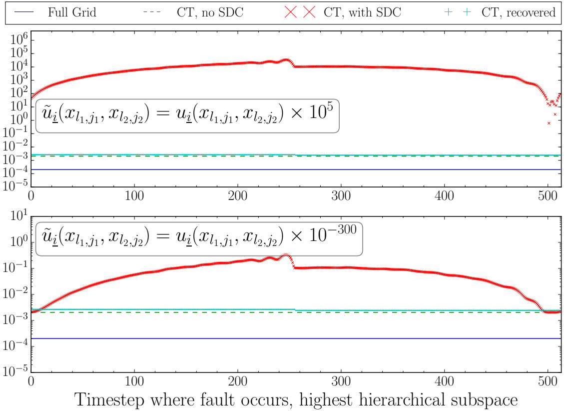
2D Example





2D Example: Simulated Soft Faults

- Inserting one soft fault
- Measuring L2-error at the end



Higher-D: Advection-Diffusion Equation

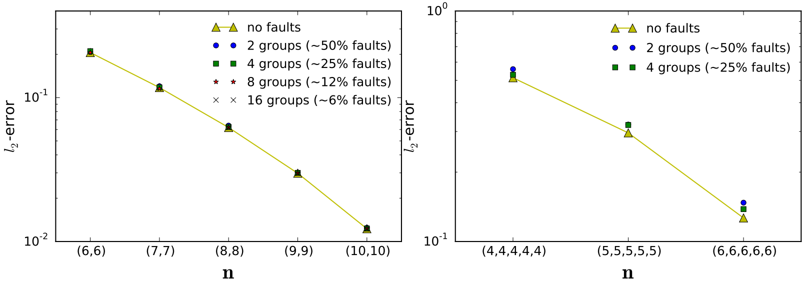
$$\begin{aligned} \partial_t u - \Delta u + \vec{a} \cdot \nabla u &= f && \text{in } \Omega \times [0, T) \\ u(\cdot, t) &= 0 && \text{in } \partial\Omega \\ u(\cdot, 0) &= u_0 && \text{in } \Omega \end{aligned}$$

$$\Omega = [0,1]^d, \vec{a} = (1, \ldots, 1)^T, u_0 = e^{-100 \sum_{i=1}^d (x_i - 0.5)^2}$$

- Implemented in DUNE-pdelab
- FVM, explicit time integration

Results

- Fault in second time step
- Relative error w.r.t. full-grid solution (**n** = **11** in 2D, **n** = **7** in 5D)
- Computations on Hazel Hen (HLRS)
- 2D, 5D:



Again: excellent recovery properties!

Overview

Summary

Gyrokinetics

- High-dimensional problem with urgent need for compute resources

Summary

Gyrokinetics

- High-dimensional problem with urgent need for compute resources

Hierarchical multilevel splitting provides novel handles on exa-challenges

- **Scalability**
  - 2nd level of parallelism
  - Numerical decoupling, extrapolation
  - Exploit hierarchical splitting for optimal communication
- **ABFT at low cost**
  - Exploit hierarchical scheme
  - Recombination rather than recomputation
- **Silent faults**
  - Exploit underlying hierarchical basis
  - Detection and treatment of silent faults possible

Summary

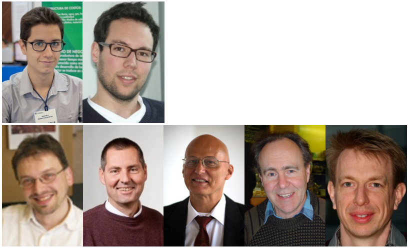
Gyrokinetics

- High-dimensional problem with urgent need for compute resources
- Sparse grids: ”Too Big Data” ⇒ Big Data

Hierarchical multilevel splitting provides novel handles on exa-challenges

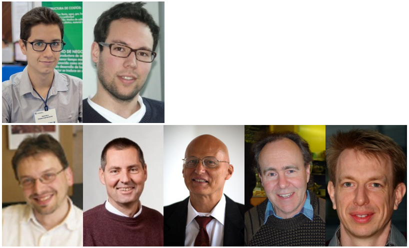
- **Scalability** reduce data in communication
  - 2nd level of parallelism
  - Numerical decoupling, extrapolation
  - Exploit hierarchical splitting for optimal communication
- **ABFT at low cost** avoid data storage and I/O
  - Exploit hierarchical scheme
  - Recombination rather than recomputation
- **Silent faults** limit communicated data
  - Exploit underlying hierarchical basis
  - Detection and treatment of silent faults possible

Thanks to:



. . . and all others!

Thanks to:



. . . and all others!

Thank you for your interest!

Mario Heene, Alfredo Parra Hinojosa, Hans-Joachim Bungartz, and Dirk Pflüger.
A massively-parallel, fault-tolerant solver for time-dependent pdes in high dimensions.
In *Euro-Par 2016*, Grenoble, June 2016.
Accepted.

Philipp Hupp, Mario Heene, Riko Jacob, and Dirk Pflüger.
Global communication schemes for the numerical solution of high-dimensional PDEs.
Parallel Computing, 52:78 – 105, 2016.

Mario Heene and Dirk Pflüger.
Scalable algorithms for the solution of higher-dimensional PDEs.
In *Software for Exascale Computing-SPPEXA 2013-2015*, pages 165–186. Springer International Publishing, 2016.

Alfredo Parra Hinojosa, Christoph Kowitz, Mario Heene, Dirk Pflüger, and Hans-Joachim Bungartz.
Towards a fault-tolerant, scalable implementation of GENE.
In *Recent Trends in Computational Engineering-CE2014*, pages 47–65. Springer International Publishing, 2015.

Alfredo Parra Hinojosa, Brendan Harding, Hegland Markus, and Hans-Joachim Bungartz.
Handling silent data corruption with the sparse grid combination technique.
In *Proceedings of the SPPEXA Symposium*, Lecture Notes in Computational Science and Engineering. Springer-Verlag, February 2016.

Dirk Pflüger, Hans-Joachim Bungartz, Michael Griebel, Frank Jenko, Tilman Dannert, Mario Heene, Alfredo Parra Hinojosa, Christoph Kowitz, and Peter Zaspel.
EXAHD: An exa-scalable two-level sparse grid approach for higher-dimensional problems in plasma physics and beyond.
In *Euro-Par 2014 Workshop, Part II*, volume 8806 of *Lecture Notes in Computer Science*, pages 566–577. Springer-Verlag, December 2014.