Università
della
Svizzera
italiana

Faculty
of Informatics

Institute of
Computational
Science
ICS

Explicit Parallel-in-time integration of an Acoustic-Advection System

Daniel Ruprecht, Rolf Krause

Institute of Computational Science
University of Lugano (USI)

HPZC High Performance and High Productivity Computing

CONSORTIUM FOR SMALL SCALE MODELING
COSMO

Why bother?

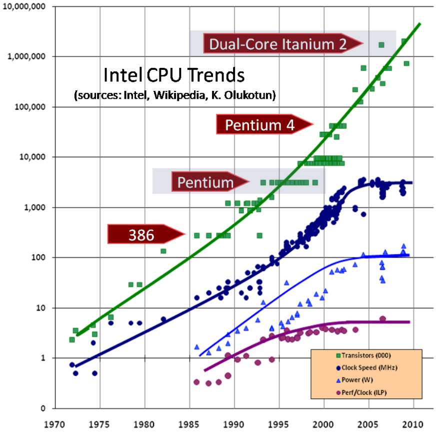
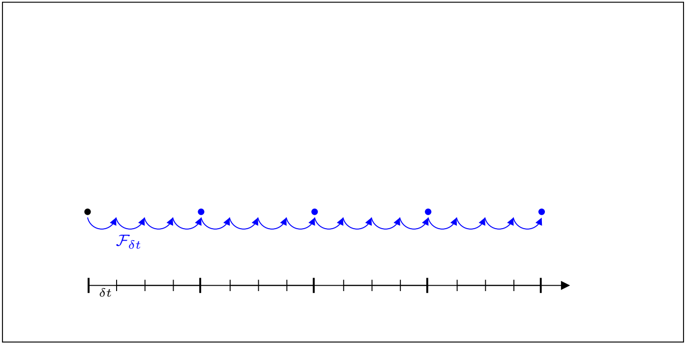


Figure: Courtesy of Herb Sutter, `www.gotw.ca`, from: *The free lunch is over - a fundamental turn toward concurrency in software*

Parareal: Parallel-in-Time Integration

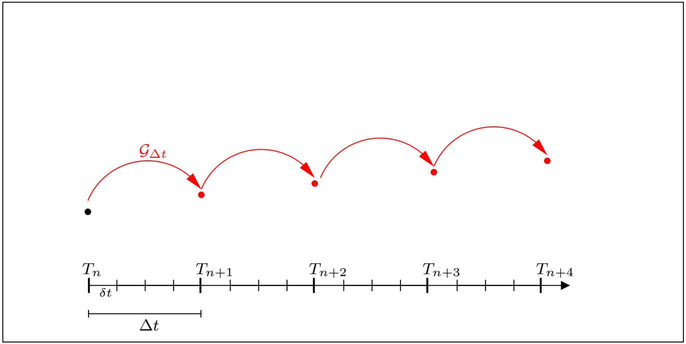


- $\mathcal{F}_{\delta t}$: accurate, computationally expensive

Reference

[Lions et al. (2001)], [Gander and Vandewalle (2007)]

Parareal: Parallel-in-Time Integration

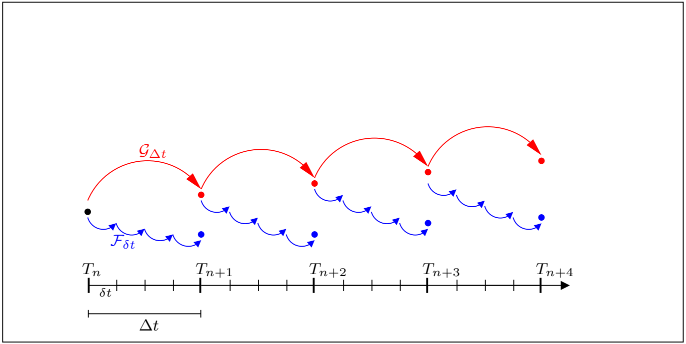


- $\mathcal{F}_{\delta t}$: accurate, computationally expensive
- $\mathcal{G}_{\Delta t}$: coarse, computationally cheap

Reference

[Lions et al. (2001)], [Gander and Vandewalle (2007)]

Parareal: Parallel-in-Time Integration

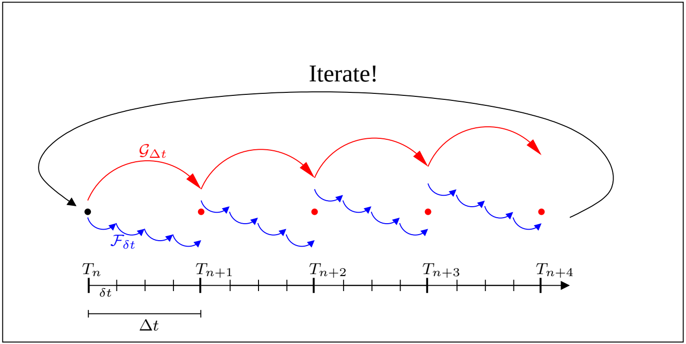


- $\mathcal{F}_{\delta t}$: accurate, computationally expensive
- $\mathcal{G}_{\Delta t}$: coarse, computationally cheap

Reference

[Lions et al. (2001)], [Gander and Vandewalle (2007)]

Parareal: Parallel-in-Time Integration

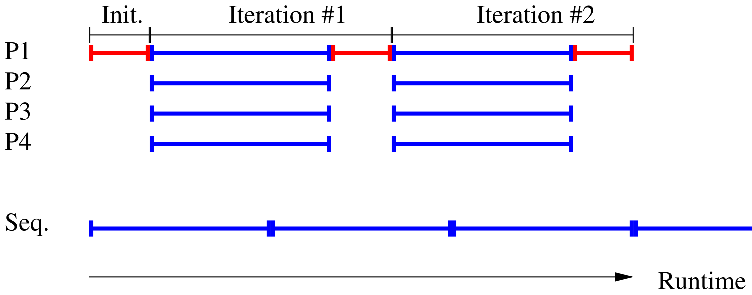


- $\mathcal{F}_{\delta t}$: accurate, computationally expensive
- $\mathcal{G}_{\Delta t}$: coarse, computationally cheap

Reference

[Lions et al. (2001)], [Gander and Vandewalle (2007)]

Speedup Estimate



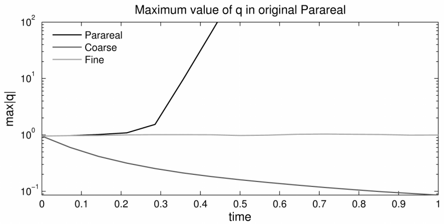
Estimated Speedup

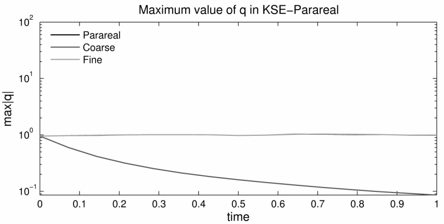
$$s \approx \frac{N_{\mathrm{t}} \tau_{\mathrm{f}}}{N_{\mathrm{c}} \tau_{\mathrm{c}} + N_{\mathrm{it}} \left(N_{\mathrm{c}} \tau_{\mathrm{c}} + \frac{N_{\mathrm{t}}}{N_{\mathrm{c}}} \tau_{\mathrm{f}} \right)} \leq \min \left\{ \frac{N_{\mathrm{c}}}{N_{\mathrm{it}}}, \frac{N_{\mathrm{t}} \tau_{\mathrm{f}}}{N_{\mathrm{c}} \tau_{\mathrm{c}}} \right\}$$

Algorithm

{ ## *Initialization (Sequential)* ## }
$Y_0^0 = y_0$
for $i = 1$ to N_c **do**
  $Y_i^0 = \mathcal{G}_{\Delta t}\left(Y_{i-1}^0, T_i, T_{i-1}\right)$
end for
{ ## *Iteration* ## }
$k := 0$
repeat
  { # *Predictor (Parallel)* # }
  for $i = 1$ to N_c **do**
    $D_i := \mathcal{F}_{\delta t}\left(Y_{i-1}^k, T_i, T_{i-1}\right) - \mathcal{G}_{\Delta t}\left(Y_{i-1}^k, T_i, T_{i-1}\right)$
  end for
  { # *Corrector (Sequential)* # }
  $Y_0^{k+1} = y_0$
  for $i = 1$ to N_c **do**
    $Y_i^{k+1} := \mathcal{G}_{\Delta t}\left(Y_{i-1}^{k+1}, T_i, T_{i-1}\right) + D_i$
  end for
  $k := k + 1$
until $k = N_{\mathrm{it}}$

Instability of Parareal for hyperbolic problems





Example: Two Dimensional Advection

$$q_t + \mathbf{U} \cdot \nabla q = 0, \quad \mathbf{U} = (1,1),\ \Omega = [0,1]^2,$$

with $q(x,y,0)$ Gauss peak.

> Parareal can become unstable for hyperbolic problems ☹

Algorithm

{ ## *Initialization (Sequential)* ## }
$Y_0^0 = y_0$
for $i = 1$ to N_c **do**
    $Y_i^0 = \mathcal{G}_{\Delta t}\left(Y_{i-1}^0, T_i, T_{i-1}\right)$
end for
{ ## *Iteration* ## }
$k := 0$
repeat
    { # *Predictor (Parallel)* # }
    for $i = 1$ to N_c **do**
        $D_i := \mathcal{F}_{\delta t}\left(Y_{i-1}^k, T_i, T_{i-1}\right) - \mathcal{G}_{\Delta t}\left(Y_{i-1}^k, T_i, T_{i-1}\right)$
    end for
    { # *Corrector (Sequential)* # }
    $Y_0^{k+1} = y_0$
    for $i = 1$ to N_c **do**
        $Y_i^{k+1} := \mathcal{G}_{\Delta t}\left(Y_{i-1}^{k+1}, T_i, T_{i-1}\right) + D_i$
    end for
    $k := k + 1$
until $k = N_{\text{it}}$

Krylov-Subspace-Enhanced Parareal

- Evolution of $\mathcal{F}$ is known on

$$S^k := \operatorname{span}\left\{ Y_i^{k'} : i = 1, \dots N_{\mathrm{c}}, k' \leq k \right\}$$

- Projection

$$\mathbf{P}^k : \mathbb{R}^d \mapsto S^k$$

- Enhance $\mathcal{G}$ to

$$\mathcal{K}_{\Delta t}(Y) := \mathcal{G}_{\Delta t}\left((\mathbf{I} - \mathbf{P}^k) Y \right) + \mathcal{F}_{\delta t}\left(\mathbf{P}^k Y \right)$$

- Requires QR decomposition in every iteration.
- Demonstrated to work for hyperbolic problems in structural dynamics.

  $>$ **Fluid problems?**

References

- introduced by [Farhat et al. (2006), Cortial and Farhat (2008)]
- recast in Parareal framework in [Gander and Petcu (2008)]

Explicit coarse integrator

- Split equation:

$$\mathbf{q}_t = \mathbf{F}(\mathbf{u}) = \mathbf{F}_{\text{fast}}(\mathbf{q}) + \mathbf{F}_{\text{slow}}(\mathbf{q})$$

- Runge-Kutta scheme (EE, RK-2, RK-3) for $\mathbf{F}_{\text{slow}}$
- For every stage, perform N_{sound} forward-backward steps for $\mathbf{F}_{\text{fast}}$ with fixed $\mathbf{F}_{\text{slow}}$
- *Here: Use split forward Euler with diffusive 1st order upwind flux and divergence damping!*

References

[Skamarock and Klemp (1992), Wicker and Skamarock (1998, 2002)]

Numerical example

- **Model:**

$$\begin{aligned} u_t + \mathbf{U}\cdot\nabla u + c_{\mathrm{s}}\pi_x &= 0 \\ v_t + \mathbf{U}\cdot\nabla v + c_{\mathrm{s}}\pi_y &= 0 \\ \pi_t + \mathbf{U}\cdot\nabla \pi + c_{\mathrm{s}}\nabla\cdot\mathbf{u} &= 0 \end{aligned}$$

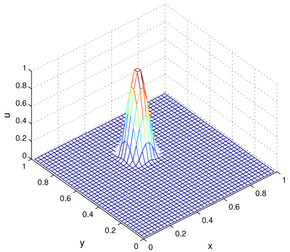
  with $c_{\mathrm{s}} = 30$.

- **Advection:** (solid body rotation)

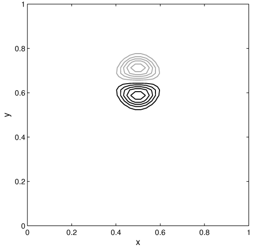
$$\mathbf{U} = \Omega\left[(-y,x) + \left(\frac{1}{2}, -\frac{1}{2}\right)\right]$$

  $\max(\mathbf{U}) \approx 1.5$, and $\Omega = \pi$.

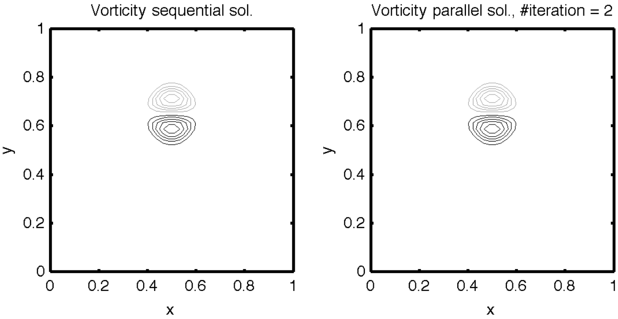
- Divergence damping: $C_{\mathrm{div,c}} = 0.1,\ C_{\mathrm{div,f}} = 0.005$



Vorticity at T = 0



Comparison of vorticity



Vorticity

$$\omega := u_y - v_x, \quad \frac{\partial \omega}{\partial t} + \mathbf{U} \cdot \nabla \omega = \nabla \cdot \mathbf{u}$$

Speedup

$N_p = 4,\ C_c = 4.0$				
$\#it.$	Error	Runtime	Speed up	Efficiency
1	3.8×10^{-2}	4.7 s	3.1	78 %
2	2.4×10^{-2}	9.5 s	1.5	38 %

$N_p = 6,\ C_c = 4.0$				
$\#it.$	Error	Runtime	Speed up	Efficiency
1	5.0×10^{-2}	3.6 s	4.1	68 %
2	1.3×10^{-2}	7.4 s	2.1	35 %
3	7.0×10^{-3}	11.3 s	1.3	22 %

$N_p = 8,\ C_c = 4.0$				
$\#it.$	Error	Runtime	Speed up	Efficiency
1	6.6×10^{-2}	3.0 s	4.9	61 %
2	1.5×10^{-2}	6.5 s	2.2	28 %
3	4.8×10^{-3}	10.4 s	1.4	18 %
4	4.0×10^{-3}	13.4 s	1.1	14 %

Conclusions and Questions

Conclusions

1. KSE - Parareal is applicable for 2D linear acoustic-advection.
2. Use of purely explicit propagators is possible.
3. Speed up is achieved but parallel efficiency is inherently moderate.
4. "Easy" to implement if $\mathcal{F}$, $\mathcal{G}$ are available.

Questions

1. What about more complex models, e.g. fully nonlinear compressible Euler eq.?
2. Implementation in a hybrid MPI (spatial) / Open MP (temporal) approach ?
3. Other choices for $\mathcal{G}$ and $\mathcal{F}$?
4. Reduced models (e.g. sound-proof) in $\mathcal{G}$ for larger coarse time steps?

References

J.-L. Lions and Y. Maday and G. Turinci,
A "parareal" in time discretization of PDE's
C. R. Acad. Sci. Sér. I Math., 332 (2001), pp. 661-668.

C. Farhat and J. Cortial and C. Dastillung and H. Bavestrello,
Time-parallel implicit integrators for the near-real-time prediction of linear structural dynamic responses,
Int. J. Numer. Meth. Engng., 67 (2006), pp. 697–724.

M. J. Gander and S. Vandewalle,
Analysis of the parareal time-parallel time-integration method,
SIAM J. Sci. Comp., 29 (2007), pp. 556-578.

J. Cortial and C. Farhat,
A time-parallel implicit method for accelerating the solution of non-linear structural dynamics,
Int. J. Numer. Meth. Engng., 77 (2008), pp. 451–470.

M. J. Gander and M. Petcu
Analysis of a Krylov subspace enhanced parareal algorithm for linear problems
ESAIM Proc., 25 (2008), pp. 114–129.

The Future ?

International Exascale Software Project Roadmap

`www.exascale.org`

- 100 million – 1 billion cores
- Clock rates 1 – 2 GHz
- Hundreds of cores per die
- Massive multithreaded fine grain concurrency per node
- Power consumption of maximum 25 MWatts