Université de Strasbourg

Parareal Algorithm: An Advanced Analysis

Internship Presentation

Oussama BOUHENNICHE

Supervisors: M. Christophe PRUD'HOMME

University of Strasbourg
Cemosis

August 21, 2024

Overview

Introduction

Context

- Solving time-dependent ordinary differential equations (ODEs) numerically is crucial in computational fields.
- Traditional methods can be highly time-consuming, especially for long simulations or intricate systems.

The Parareal algorithm[1] a parallel-in-time integration method designed to efficiently solve time-dependent differential equations

[1] J.-L. Lions, Y. Maday, and G. Turinici, "Résolution d'edp par un schéma en temps pararéel," *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics*, vol. 332, no. 7, pp. 661–668, 2001.

Objectives

1. Analyzing the convergence of the Parareal algorithm by evaluating the number of iterations required for different solver combinations
2. Investigate how the order of convergence of the Parareal algorithm varies with different solvers
3. Evaluating the performance of the Parareal algorithm in relation to the number of processes used

Background

Parareal Algorithm 1/3

Developed by Lions, Maday, and Turinici in 2001, It aims to solve problems of the form:

$$\frac{du}{dt} = f(t, u) \quad , \; u(t_0) = u_0$$

- Accelerates time-dependent ODE solving using parallel computing.
- Decomposes the time domain into smaller intervals that can be processed simultaneously.

Parareal Algorithm 2/3

Université de Strasbourg

- **How it Works**
    - Combines coarse and fine propagators.
    - Iteratively corrects the coarse solution with fine solution details.
- **Parallelization**
    - Distributes computation across multiple processors.
    - Significantly reduces computation time for large-scale problems.

Parareal Algorithm 3/3

Algorithm 1 Parareal Algorithm (parallel)

1: **Given:** Coarse function G, fine function F, number of time steps N, maximum number of iterations K, ϵ tolerance, number of process.

2: **Initialize: in sequential**

- $U_0^0 = u_0; \quad U_{j+1}^0 = G(t_j, t_{j+1}, U_j^0) \quad j = 0, 1, \ldots, N-1$

3: **while** $|U^k - U^{k-1}| > \epsilon$ **do**

- run the fine solver on each process and given a sub-interval of the time slices, in parallel: $F(t_j, t_{j+1}, U_j^{k-1}) \qquad j = 0, 1, ..., N-1.$
- communicate fine solution and update u in sequetial and share it to all process: $U_{j+1}^k = G(t_j, t_{j+1}, U_j^k) + F(t_j, t_{j+1}, U_j^{k-1}) - G(t_j, t_{j+1}, U_j^{k-1}).$

4: **end while**

5: **Return:** U^k

Methodology and Implementation

Previous Works

Université de Strasbourg

Python Package for Parareal Algorithm

- **Foundation:** Developed a Python package for solving time-dependent differential equations using the Parareal algorithm.
- **Core Features:**
  - **Solver Integration:** Supports multiple ODE solvers (explicit/implicit Euler, Runge-Kutta, SciPy solvers).
  - **Parallelization:** Utilizes Python's multiprocessing and MPI for parallel execution.
  - **Customization:** Allows user-defined solvers, adjustable time intervals, convergence criteria, and maximum iterations.

Experimental Setup 1/2

Université de Strasbourg

- **Parallel Environment**
  - Tests conducted on a high-performance computing system with multiple cores (Gaya)
- **Solver Configurations**
  The study involved a variety of solvers to cover a broad range of explicit and implicit methods:
  - **Explicit Solvers:**
    - Explicit Euler
    - Explicit RK2
    - Explicit RK4
  - **Implicit Solvers:**
    - Implicit Euler
    - Implicit RK2
    - Implicit RK4

Experimental Setup 2/2

Université de Strasbourg

- **Test Systems:**
  - **Lorenz system**[2]

$$\begin{cases} \frac{dx}{dt} = \sigma(y - x) \\ \frac{dy}{dt} = x(\rho - z) - y \\ \frac{dz}{dt} = xy - \beta z \end{cases}$$

  - **System 1:** $\frac{dy}{dt} = -ty^2$
  - **System 2:** $\frac{dy}{dt} = -y + cos(t)$
  - **System 3:** $\frac{dy}{dt} = -2y$

[2] E. N. Lorenz, "Deterministic nonperiodic flow," *Journal of atmospheric sciences*, vol. 20, no. 2, pp. 130–141, 1963.

Tests Setup 1/2

Université de Strasbourg

Number of Parareal Iterations and Convergence Order Tests

- **Tested on 3 systems**
- **Solvers Used**: Explicit/Implicit (Euler, RK2, RK4).
- **Initial Setup**:
  - Initial conditions set as specified.
  - Time span $[0, 5]$ discretized.
- **Execution**: Recorded number of Parareal iterations to reach tolerance.
- **Convergence Order P Calculation**:

$$P = \frac{1}{n} \sum_{i=0}^{n-1} \frac{\log\left(\frac{e_{i+1}}{e_i}\right)}{\log\left(\frac{\Delta_{i+1}}{\Delta_i}\right)}$$

Tests Setup 2/2

Université de Strasbourg

Execution Time and Speedup Tests

- **System Tested**: Lorenz system.
- **Solvers Used**: Explicit/Implicit (Euler, RK2, RK4).
- **Initial Setup**:
  - Initial conditions set as specified.
  - Time span $[0, 10]$ discretized.
- **Parallel Execution**:
  - Used up to 128 processes with MPI.
  - Recorded execution times T_P.
- **Speedup S Calculation**:

$$S = \frac{T_1}{T_P}$$

Parallel Test Execution

Université de Strasbourg

- **Parallel Testing:** Shell scripts created to run tests in parallel, significantly reducing execution time.
- **Automation:**
  - Automates test execution using parallel processing.
  - Handles large test suites efficiently.
- **Test Configuration:** Defines test parameters, file names, and configurations.
- **Test Execution:** Executes tests by calling Python scripts with necessary arguments.
- **Result Handling:** Saves results and cleans up temporary files.

Results and Analysis

Number of Parareal Iterations for Convergence

Université de Strasbourg

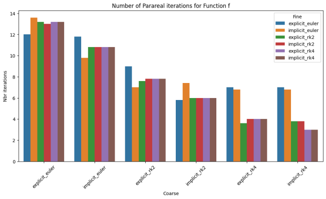

Figure: Number of Parareal iterations for system 1

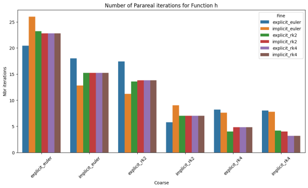

Figure: Number of Parareal iterations for system 2

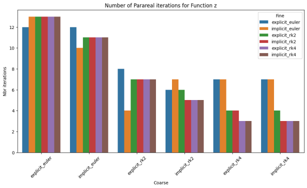

Figure: Number of Parareal iterations for system 3

- The number of iterations required for convergence is significantly influenced by the choice of coarse solver
- Higher-order coarse solvers lead to faster convergence

Université de Strasbourg

Order of Convergence

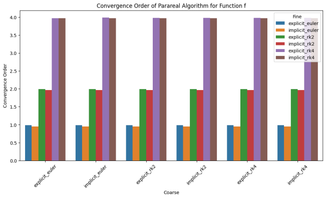

Figure: Convergence Order of Parareal Algorithm for system 1

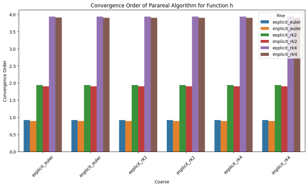

Figure: Convergence Order of Parareal Algorithm for system 2

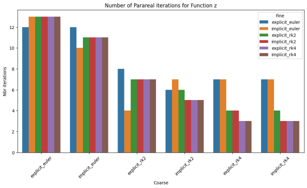

Figure: Convergence Order of Parareal Algorithm for system 3

- The fine solver dictates the overall convergence order of the Parareal algorithm
- Higher-order fine solvers enhance accuracy and improve the convergence rate

Execution Time and Speedup

Université de Strasbourg

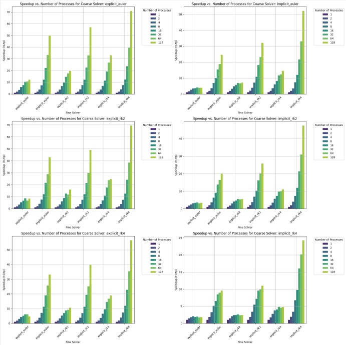

Figure: Speedup vs. Number of Processes for Different solvers combination

- Parallel execution greatly reduces time, showing near-linear speedup with fewer processes
- Speedup decreases with more processes, especially for explicit solvers, due to communication overhead

Conclusion

Conclusion

Université de Strasbourg

Key Findings

- **Convergence Analysis:**
  - Higher-order coarse solvers lead to faster convergence.
- **Order of Convergence:**
  - Fine solver dictates convergence order.
  - Higher-order fine solvers improve accuracy and rate.
- **Performance Evaluation:**
  - Parallel execution reduces time, with near-linear speedup initially.
  - Diminishing returns with more processes due to communication overhead.

Future Prospects:

- Further exploration with Parareal for PDEs using frameworks like feel++.

References I

[1] J.-L. Lions, Y. Maday, and G. Turinici, "Résolution d'edp par un schéma en temps pararéel," *Comptes Rendus de l'Académie des Sciences-Series I-Mathematics*, vol. 332, no. 7, pp. 661–668, 2001.

[2] E. N. Lorenz, "Deterministic nonperiodic flow," *Journal of atmospheric sciences*, vol. 20, no. 2, pp. 130–141, 1963.

Thank you for your attention

Oussama BOUHENNICHE

Supervisors: M. Christophe PRUD'HOMME

University of Strasbourg
Cemosis

August 21, 2024