JÜLICH
FORSCHUNGSZENTRUM

PyPinT

Towards a framework for rapid prototyping of iterative parallel-in-time algorithms

May 28, 2014 | Dieter Moser, Torbjörn Klatt, Dr. Robert Speck <{d.moser,t.klatt,r.speck}@fz-juelich.de> | 3rd Workshop on Parallel-in-Time Integration Methods

Overview

1. Recap of existing Parallel-in-Time Algorithms
2. The *PyPinT* Framework Explained
3. Goals
4. Proof of Concept — Examples Analyzed

JÜLICH
FORSCHUNGSZENTRUM

PyPinT

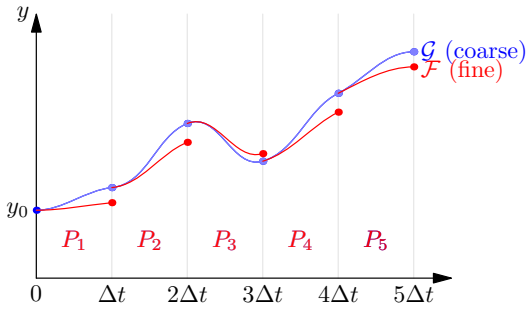
Part I: Existing Parallel-in-Time Approaches

May 28, 2014 | Dieter Moser, Torbjörn Klatt, Dr. Robert Speck <{d.moser,t.klatt,r.speck}@fz-juelich.de>

Parareal

Reference

- coarse $\mathcal{G}$ and fine $\mathcal{F}$ propagators make parareal flexible and modular
- an initial value is improved iteratively
- order is controllable through the fine propagator $\mathcal{F}$



[Courtesy of M. Emmett, LBNL]

$$y_{m+1}^{k+1} = \mathcal{F}(y_m^k) + \mathcal{G}(y_m^{k+1}) - \mathcal{G}(y_m^k)$$

IDC and DC

$$y'(t) = f(y(t), t), \quad y(0) = y_0$$

Using the residual

$$r(t) = f(t, \tilde{y}(t)) - \tilde{y}(t)$$

to compute the error

$$e'(t) = f(t, \tilde{u} + e) - f(t, \tilde{y}) + r'(t)$$
$$e(0) = 0$$

for the next update

$$\tilde{y}_{j+1} = \tilde{y}_j + e_j$$

$$y'(t) = f(y(t), t), \quad y(0) = y_0$$

Using the residual

$$r(t) = f(t, \tilde{y}(t)) - \tilde{y}(t)$$

to compute the error

$$e'(t) = f(t, \tilde{u} + e) - f(t, \tilde{y}) + r'(t)$$
$$e(0) = 0$$

for the next update

$$\tilde{y}_{j+1} = \tilde{y}_j + e_j$$

RIDC

RIDC

PFASST

JÜLICH
FORSCHUNGSZENTRUM

PyPinT

Part II: The PyPinT Framework Explained

May 28, 2014 | Dieter Moser, Torbjörn Klatt, Dr. Robert Speck <{d.moser,t.klatt,r.speck}@fz-juelich.de>

Basic Concept

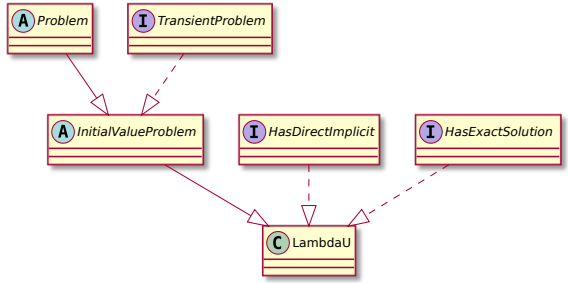
- *Python* ≥ 3.2 as language of choice
  for ease of use and extensibility (cf. *NumPy*, *SciPy*)
- well-conceived and intuitive abstract interfaces
  for reusable code ensuring DRY principle
- modular building blocks
  for fast exchange of algorithms' building blocks
- integrated analyzation tools
  for introspection and plotting (cf. *matplotlib*)
- usage of a sophisticated testing framework
  nobody writes bug-free code

Modules

Abstract Modeling of PinT Algorithms

```
pypint
  ~.problems
  ~.solvers
  ~.integrators
  ~.communicators
  ~.multi_level_providers
  ~.plugins
  ~.solutions
```

- interfaces for problem setups
- generic problem with specializations via mixins

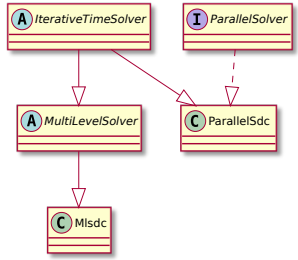


Modules

Abstract Modeling of PinT Algorithms

```
pypint
  ~.problems
  ~.solvers
  ~.integrators
  ~.communicators
  ~.multi_level_providers
  ~.plugins
  ~.solutions
```

- interfaces for iterative time solvers
- providing generic building blocks of solvers



JÜLICH
FORSCHUNGSZENTRUM

PyPinT

Part III: Goals for PyPinT

May 28, 2014 | Dieter Moser, Torbjörn Klatt, Dr. Robert Speck <{d.moser,t.klatt,r.speck}@fz-juelich.de>

JÜLICH
FORSCHUNGSZENTRUM

PyPinT

Part IV: Proof of Concept

May 28, 2014 | Dieter Moser, Torbjörn Klatt, Dr. Robert Speck <{d.moser,t.klatt,r.speck}@fz-juelich.de>

Thank you for your attention!

Questions?

(now or later)

PyPinT is on GitHub: `https://github.com/Parallel-in-Time/PyPinT`

Dieter Moser
Juelich Supercomputing Centre
Building 16.3, Room 022
Tel: +49 2461 61 96453
eMail: d.moser@fz-juelich.de

Torbjörn Klatt
Juelich Supercomputing Centre
Building 16.3, Room 022
Tel: +49 2461 61 96452
eMail: t.klatt@fz-juelich.de

Dr. Robert Speck *
Juelich Supercomputing Centre
Building 16.3, Room 131
Tel: +49 2461 61 1644
eMail: r.speck@fz-juelich.de

* corresponding author