Parallel Numerical Algorithms

Chapter 7 – Differential Equations
Section 7.1 – Ordinary Differential Equations

Michael T. Heath

Department of Computer Science
University of Illinois at Urbana-Champaign

CS 554 / CSE 512
Outline

1. Ordinary Differential Equations
   - Parallelism in Solving ODEs
   - Waveform Relaxation
   - Boundary Value Problems for ODEs
Minor potential sources of parallelism in solving initial value problem for system of ODEs $y' = f(t, y)$ include:

- For multi-stage methods (e.g., Runge-Kutta), computation of multiple stages in parallel.
- For multi-level methods (e.g., extrapolation), computation of multiple levels (e.g., with different step sizes) in parallel.
- For multi-rate methods, integration of slowly and rapidly varying components of solution in parallel.
Major potential sources of parallelism in solving initial value problem for system of ODEs $y' = f(t, y)$ include

- Evaluation of right-hand-side function $f$ in parallel (e.g., evaluation of forces for $n$-body problems)
- Parallel implementation of linear algebra computations (e.g., solving linear system in Newton’s method for stiff ODEs)
- Partitioning equations in system of ODEs into multiple tasks (e.g., waveform relaxation, discussed next)
Picard Iteration

- Consider initial value problem for system of $n$ ODEs
  \[ y' = f(t, y), \quad t \geq t_0, \text{ with IC } y(t_0) = y_0 \]

- Starting with $y_0(t) \equiv y_0$, Picard iteration is given by
  \[ y_{k+1}(t) = y_0 + \int_{t_0}^{t} f(s, y_k(s)) \, ds \]

- If $f$ satisfies Lipschitz condition, then Picard iteration converges to solution of IVP

- Convergence may be slow, but parallelism is excellent, as problem decouples into $n$ independent 1-D quadratures
Picard iteration is simple fixed-point iteration on function space

Picard iteration is often too slow to be useful, but other such iterations may be more rapidly convergent

Iterative methods of this type are commonly called *waveform relaxation*
For $n = 2$, consider iteration

$$\begin{bmatrix}
y_1^{(k+1)}(t) \\
y_2^{(k+1)}(t)
\end{bmatrix}' = \begin{bmatrix}
f_1(t, y_1^{(k+1)}(t), y_2^{(k)}(t)) \\
f_2(t, y_1^{(k)}(t), y_2^{(k+1)}(t))
\end{bmatrix}$$

System of two independent ODEs can be solved in parallel

Method generalizes in obvious way to arbitrary system of $n$ ODEs and decouples system into $n$ independent ODEs

Because of its analogy to Jacobi iteration for linear algebraic systems, method is called \textit{Jacobi waveform relaxation}
Gauss-Seidel Waveform Relaxation

- Convergence rate of Jacobi waveform relaxation is improved by *Gauss-Seidel waveform relaxation*, illustrated here for $n = 2$

\[
\begin{bmatrix}
  y_1^{(k+1)}(t) \\
  y_2^{(k+1)}(t)
\end{bmatrix}' = \begin{bmatrix}
  f_1(t, y_1^{(k+1)}(t), y_2^{(k)}(t)) \\
  f_2(t, y_1^{(k+1)}(t), y_2^{(k+1)}(t))
\end{bmatrix}
\]

- Unfortunately, system is no longer decoupled, so parallelism is lost unless components are reordered, analogous to red-black or multicolor ordering

- More generally, multi-splittings can further enhance parallelism in waveform relaxation methods
Potential sources of parallelism in solving boundary value problems for ODEs include

- For finite difference and finite element methods, parallel implementation of resulting linear algebra computations (e.g., cyclic reduction for tridiagonal systems)
- Multi-level methods
- Multiple shooting method


References – Parallel Solution of ODEs


References – Parallel Solution of ODEs


