Waveform Iterative Methods for Solving Initial Value Problems

Andrew Lumsdaine
Department of Computer Science and Engineering
University of Notre Dame
Notre Dame, IN 46556

Mark Reichelt
Research Laboratory of Electronics
Massachusetts Institute of Technology
Cambridge, MA 02139

Jacob White
Research Laboratory of Electronics
Massachusetts Institute of Technology
Cambridge, MA 02139

Overview

1. Motivation
2. Classical waveform techniques for solving initial value problems
3. Accelerating classical waveform techniques
4. Application: semiconductor device transient simulation
5. Experimental results
6. Conclusion

Everything you always wanted to know about waveform methods . . . .

1. What are they?
2. What are they good for?
3. Don't they converge too slowly?
4. Are they *really* useful for anything?
5. Do you have any experimental results?
6. What are the open questions?

Cluster-Based Computing with PVM

Advantages:
- Availability
- Cost
- Ease of use

Disadvantages
- Communication latency — O(10ms)

Motivation

If solving

$$Ax = b$$

takes longer in parallel than serially, can

$$\dot{x} + Ax = f$$

be solved faster in parallel than serially?

Solving Initial Value Problems

Consider the nonlinear initial value problem

$$F(x, \dot{x}, t) = 0$$
$$x(0) = x_0$$

Standard approach for numerical solution
1. Discretize in time (with implicit formula)
2. Solve nonlinear problem with Newton-Raphson
3. Solve linear system with Gaussian elimination

Disadvantages
- Parallelization = parallelization of matrix solution
- Bad when matrix solution does not parallelize well
- Bad when synchronization costs are high (as in a cluster of workstations)

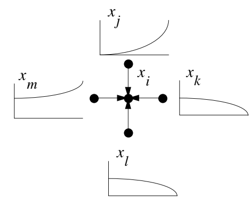
Waveform Methods

The equation $F(x, \dot{x}, t) = 0$ represents

$$F_1(x_1, \ldots, x_N, \dot{x}_1, \ldots, \dot{x}_N, t) = 0$$
$$F_2(x_1, \ldots, x_N, \dot{x}_1, \ldots, \dot{x}_N, t) = 0$$
$$\ldots$$
$$F_N(x_1, \ldots, x_N, \dot{x}_1, \ldots, \dot{x}_N, t) = 0$$

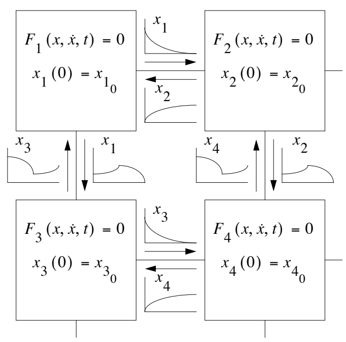
Basic idea:
- Before discretization
- Map one equation per processor
- Each processor solves scalar initial value problem

Waveform Iteration



- Compute $x_i(t)$ for $t \in [0, T]$
- Hold $x_j(t)$, $x_k(t)$, $x_l(t)$, $x_m(t)$ fixed
- Compute $x_j(t)$, etc.

Data to Processor Mapping



Waveform Relaxation

Gauss-Jacobi for $Ax = b$

Iterate: For $k = 0, 1, \ldots$ solve
For $i = 0, 1, \ldots$ solve

$$a_{ii} x_i^{k+1} = b_i - \sum_{j \neq i} a_{ij} x_j^k$$

Jacobi WR for $\dot{x} + Ax = f$

Iterate: For $k = 0, 1, \ldots$ solve
For $i = 0, 1, \ldots$ solve

$$\dot{x}_i^{k+1} + a_{ii} x_i^{k+1} = f_i - \sum_{j \neq i} a_{ij} x_j^k$$

Methods for Nonlinear Problems

Jacobi WR for solving $\dot{x} + F(x) = 0$

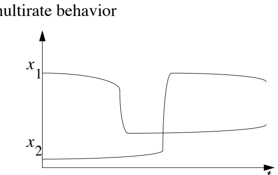
Iterate: For $k = 0, 1, \ldots$ solve
For $i = 0, 1, \ldots$ solve

$$\dot{x}_i^{k+1} + F_i(x_i^k, \ldots, x_i^{k+1}, \ldots, x_N^k) = 0$$
$$x_i^{k+1}(0) = x_{i_0}$$

Jacobi WRN
- Take only one Newton step
- Use timepoint value from previous waveform iterate as initial guess for Newton

Advantages

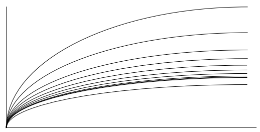
- Avoid directly solving large matrices
- Exploit multirate behavior



- Parallelizable
- Synchronize only once per WR iteration (independent of number of timesteps)
- Communicate in large packets

Disadvantages

- Slow convergence



Acceleration schemes

- Waveform SOR
- Waveform Krylov-subspace techniques

Operator Equation Formulation

Jacobi WR for $\dot{x} + Ax = f$ (compactly)

Iterate: For $k = 0, 1, \ldots$ solve

$$\dot{x}^{k+1} + Mx^{k+1} = Nx^{k+1} + f$$

where $A = M - N$

Solving for x^{k+1}

$$x^{k+1} = \int_0^t \Phi_M(t,s)\,Nx^k(s)\,ds$$

$$+ \Phi_M(t,0)\,x(0) + \int_0^t \Phi_M(t,s)\,f(s)\,ds$$

Operator Equation Formulation

Rewrite compactly as:

$$x^{k+1} = \boldsymbol{K}x^k + \psi$$

Solution x satisfies

$$(I - \boldsymbol{K})\,x = \psi$$

Properties of $\boldsymbol{K}$

- Compact
- Not self adjoint

Waveform GMRES

1. *Start:* Set $r^0 = \psi - (I - \boldsymbol{K})\,x^0$, $v^1 = r^0 / \|r^0\|$
2. *Iterate:* For $k = 1, 2, \ldots$, until satisfied do:

$h_{j,k} = \langle (I - \boldsymbol{K})\,v^k, v^j \rangle$, $j = 1, \ldots, k$

$\hat{v}^{k+1} = (I - \boldsymbol{K})\,v^k - \sum_{j=1}^{k} h_{j,k} v^j$

$h_{k+1,k} = \|\hat{v}^{k+1}\|$

$v^{k+1} = \hat{v}^{k+1} / h_{k+1,k}$

3. *Form approximate solution:*

$x^k = x^0 + V^k y^k$ where y^k minimizes $\|\beta e_1 - \bar{H}^k y^k\|$

Fundamental Operations

- Operator-Waveform product

  To calculate $w \equiv (I - \boldsymbol{K})\,p$

  1. Calculate $\boldsymbol{K}p$ with one step of normal WR, i.e., solve

$$\dot{y} + My = Np$$
$$y(0) = 0$$

  2. Set $w = p - y$

- Inner product

  Use L_2 inner product

$$\langle x, y \rangle = \sum_{i=1}^{N} \int_0^T x_i(t)\,y_i(t)\,dt$$

  Integration performed in parallel followed by global sum

Nonlinear Systems

Apply Newton to $\dot{x} + F(x) = 0$

Iterate: For $m = 0, 1, \ldots$, solve

$$\dot{x}^{m+1} + J_F(x^m)\,x^{m+1} = J_F(x^m)\,x^m - F(x^m)$$

$$x^{m+1} = x_0$$

with WGMRES

Device Transient Simulation

- Governing equations

$$\frac{kT}{q}\nabla\cdot(\varepsilon\nabla u) + q(p - n + N_D - N_A) = 0$$

$$\nabla\cdot J_n - q\left(\frac{\partial n}{\partial t} + R\right) = 0$$

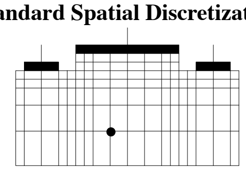
$$\nabla\cdot J_p + q\left(\frac{\partial p}{\partial t} + R\right) = 0$$

- Drift-Diffusion approximation

$$J_n = -qD_n(n\nabla u - \nabla n)$$

$$J_p = -qD_p(p\nabla u + \nabla p)$$

Standard Spatial Discretization



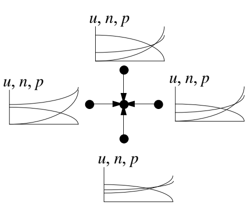
Each mesh node contributes three equations

$$\frac{\varepsilon kT}{q}\sum_j\left\{\frac{d_{ij}}{L_{ij}}(u_i - u_j)\right\} - qA_i(p_i - n_i + N_D - N_A) = 0$$

$$qD_n\sum_j\left\{\frac{d_{ij}}{L_{ij}}[n_jB(u_j - u_i) - n_iB(u_i - u_j)]\right\} - qA_i\left(\frac{dn_i}{dt} + R_i\right) = 0$$

$$qD_p\sum_j\left\{\frac{d_{ij}}{L_{ij}}[p_iB(u_j - u_i) - p_jB(u_i - u_j)]\right\} + qA_i\left(\frac{dp_i}{dt} + R_i\right) = 0$$

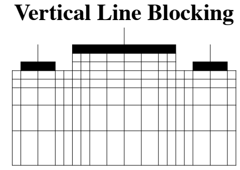
Waveform Relaxation Step



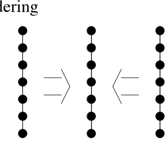
At each grid node i solve 3×3 system for

$$\begin{pmatrix} u_i^{k+1}(t) \\ n_i^{k+1}(t) \\ p_i^{k+1}(t) \end{pmatrix} \text{ using } \begin{pmatrix} u_j^{k}(t) \\ n_j^{k}(t) \\ p_j^{k}(t) \end{pmatrix}$$

Vertical Line Blocking

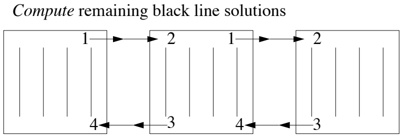


- For each vertical mesh line
  Solve for u, n, p for nodes in line
- Use red/black ordering

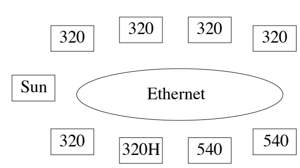


Parallel Implementation

- Manager divides mesh and sends lines to worker programs
- Worker WR iteration:
  1. *Send* black line solutions
     Compute internal red line solutions
  2. *Receive* black line solutions
     Compute remaining red line solutions
  3. *Send* red line solutions
     Compute internal black line solutions
  4. *Receive* red line solutions
     Compute remaining black line solutions

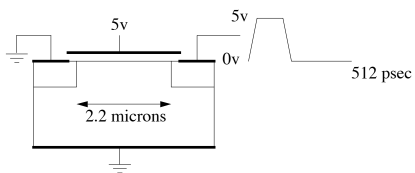


Experimental Setup



- Workstation cluster
  IBM RS/6000 as workers (8)
  Sun SparcStation as manager (1)
- Standard Ethernet (10Mbit/sec)
- PVM 2.4
- 1,2,4 machine results on model 320 machines

Experimental Device Model



- N-channel MOSFET
- 19×31 mesh
- First order BDF
- 256 fixed timesteps

Experimental Results

Method	#Procs	Wall clock sec
WRN	1	8230.23
	2	4469.91
	4	2712.58
	8	1571.92
WN/WGMRES	1	*
	2	*
	4	925.6
	8	504.5
Point (direct)	1	2462.48
Point (GMRES)	1	1221.98
	2	6931.86

Future Work

- Multirate implementation
- PVM 3.1
- Other parallel machines
- Parallelize in time (timepoint pipelining)
- Theoretical convergence estimates
- Other acceleration techniques
- Other applications

Conclusion

- Accelerated waveform methods are a very effective parallel approach for initial value problems
- Waveform methods are especially attractive for loosely coupled processors (e.g., workstation cluster with PVM)
- For semiconductor device transient simulation:
  - Fastest serial method did not parallelize well (it slowed down on two processors)
  - WGMRES on 8 workstations more than twice as fast as fastest serial method for device transient simulation