Multilevel Resiliency for PDE Simulations

Mark Adams[2] Jed Brown[1], **Peter Brune**[1], Barry Smith[1]

[1]Mathematics and Computer Science Division, Argonne National Laboratory
[2]Applied Physics and Applied Mathematics Dept., Columbia University

Presented at the The 8th Workshop of the INRIA-ANL-Illinois Joint Laboratory on Petascale Computing

November 21, 2012

Overview

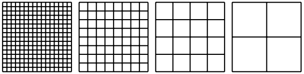
We'll discuss:

1. introduction to multilevel methods
2. multilevel methods on the extreme scale
3. multilevel checkpointing framework
4. multilevel error detection

With the goal of showing:

1. algorithmic resiliency advantages of multigrid
2. a basic scheme for multigrid-based checkpointing and error detection
3. a way forward for extreme-scale algorithmic resiliency

Multigrid Preliminaries

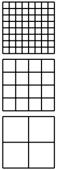


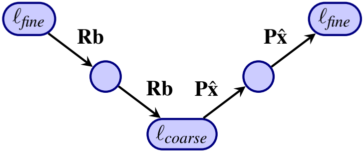
Multigrid is an $O(n)$ method for solving linear algebra problems by defining a hierarchy of scale. A Multigrid method is constructed from:

1. a series of discretizations
   - coarser approximations of the original problem
   - constructed algebraically or geometrically
2. intergrid transfer operators
   - restriction $\mathbf{R}$ and injection $\hat{\mathbf{R}}$ (fine to coarse)
   - prolongation $\mathbf{P}$ (coarse to fine)
3. Smoothers ($\mathbf{S}$)
   - correct the high frequency error components
   - Richardson, Jacobi, Gauss-Seidel, etc.
   - Gauss-Seidel-Newton or optimization methods

Multigrid

- **Multigrid** methods uses coarse correction for large-scale error

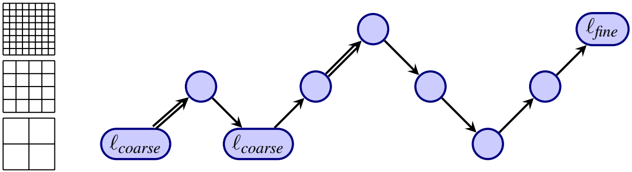




Algorithm $MG(\mathbf{A},\mathbf{b})$ for the solution of $\mathbf{Ax}=\mathbf{b}$:

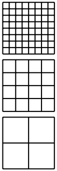
$\mathbf{x}=\mathbf{S}^m(\mathbf{x},\mathbf{b})$	pre-smooth
$\mathbf{b}^H=\mathbf{R}(\mathbf{r}-\mathbf{Ax})$	restrict residual
$\hat{\mathbf{x}}^H=MG(\mathbf{RAP},\mathbf{b}^H)$	recurse
$\mathbf{x}=\mathbf{x}+\mathbf{P}\hat{\mathbf{x}}^H$	prolong correction
$\mathbf{x}=\mathbf{x}+\mathbf{S}^n(\mathbf{x},\mathbf{b})$	post-smooth

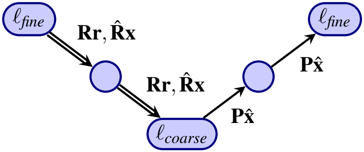
Full Multigrid(FMG)



- start by going directly to coarse
- do number of V-cycles with each going one finer
- $\mathbf{x}$ is injected to finer levels as visited
- truncation error within one cycle
- highly efficient solution method

FAS





Algorithm $FAS(\mathbf{F}, \mathbf{x}, \mathbf{b})$ for the solution of $\mathbf{F}(\mathbf{x}) = \mathbf{b}$:

$$\mathbf{x} = \mathbf{S}^m(\mathbf{x}, \mathbf{b}) \qquad \text{pre-smooth}$$

$$\mathbf{b}^H = \mathbf{R}[\mathbf{b}] + \overbrace{[\mathbf{F}^H(\hat{\mathbf{R}}\mathbf{x}) - \mathbf{R}\mathbf{F}(\mathbf{x})]}^{\tau \text{ correction}} \qquad \text{restrict residual}$$

$$\mathbf{x}^H = \hat{\mathbf{R}}\mathbf{x} \qquad \text{inject solution}$$

$$\hat{\mathbf{x}}^H = FAS(\mathbf{F}^H, \mathbf{x}^H, \mathbf{b}^H) \qquad \text{recurse}$$

$$\mathbf{x} = \mathbf{x} + \mathbf{P}[\hat{\mathbf{x}}^H - \hat{\mathbf{R}}\mathbf{x}] \qquad \text{prolong correction}$$

$$\mathbf{x} = \mathbf{x} + \mathbf{S}^n(\mathbf{x}, \mathbf{b}) \qquad \text{post-smooth}$$

τ Correction

- $\mathbf{F}^H(\mathbf{x}^H) = \mathbf{R}\mathbf{b} + [\mathbf{F}^H(\hat{\mathbf{R}}\mathbf{x}) - \mathbf{R}\mathbf{F}(\mathbf{x})]$ contains the term we call τ

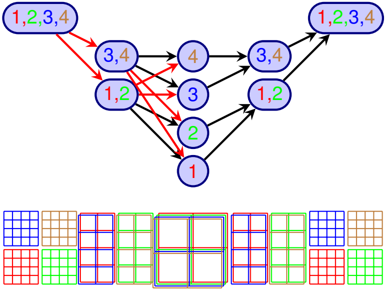
$$\tau = \mathbf{F}^H(\hat{\mathbf{R}}\mathbf{x}) - \mathbf{R}\mathbf{F}(\mathbf{x})$$

- encodes the "difference" between problems $\mathbf{F}(\mathbf{x})$ and $\mathbf{F}^H(\mathbf{x}^H)$
- exact fine solution is solution to τ-corrected coarse problem
- τ tells us how the fine problem can improve the coarse problem
- τ has same size as coarse solution

τ is the magic that makes this whole talk possible

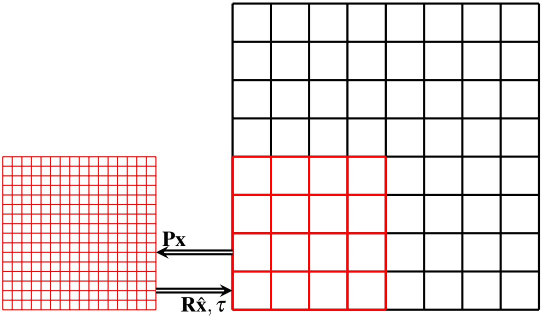
Extreme-Scale Multigrid: Redundant Coarse Problems

- simplest idea: local redundancy
- calculate coarser levels redundantly on subsets of processors
- requires more communication in fewer stages
- coarse problems must be duplicated; requires off-process restriction
- reduced synchronization



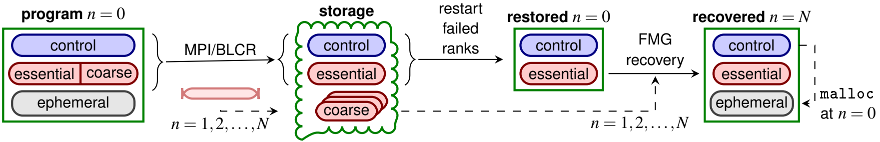
Extreme-Scale Multigrid: Segmental Refinement

- more complicated idea: ficticious fine grid, τ-corrected coarse
- originally for 70's very low memory; recently revived for extreme scale
- loop and "zoom" on subdomains
    - construct fine grid problem in cache
    - smooth locally
    - inject
- data dependencies *vertical* rather than *horizontal* between levels



Basic resilience strategy

We assume the following simple model of checkpointing and recovery:



control
- program stack
- configuration

essential
- time-dependent solution
- current optimization iterate

ephemeral
- assembled matrices
- preconditioners
- residuals

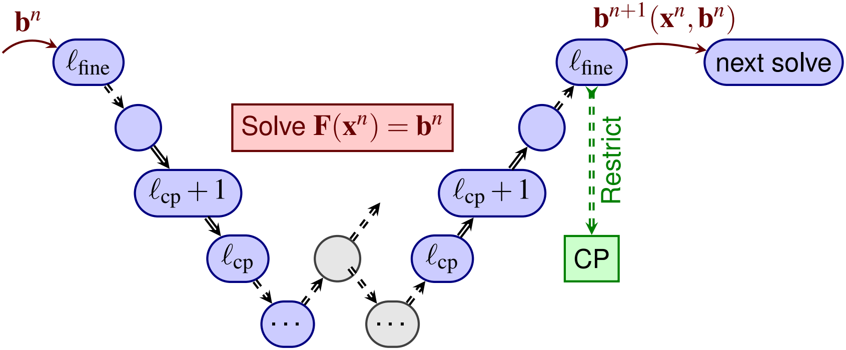
Essential State Recovery

- coarse level checkpoints are orders of magnitude smaller
- can be stored at greater frequency
- quick recovery of local essential state from coarse history
- FMG recovery needs only nearest neighbor processors

We introduce **FAS Checkpointing** for rapid recovery of essential state

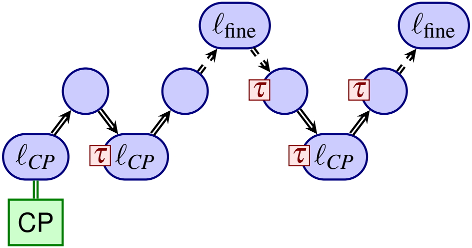
- minimal and lossy essential state storage
- whole state may be quickly recovered in total failure
- rapid local catch-up for failed processes

FAS Checkpointing



- essential state: converged solutions at end of timesteps
- checkpoint converged state at level ℓ_{CP}
- ℓ_{CP} several levels down
- CP several orders of magnitude smaller than converged state

FAS Recovery



- recover using FMG anchored at $\ell_{\text{cp}} + 1$
- needs only ℓ_{cp} neighbor points
- τ correction is local
- FMG recovery only accesses levels finer than ℓ_{CP}
- Only failed processes and neighbors participate in recovery

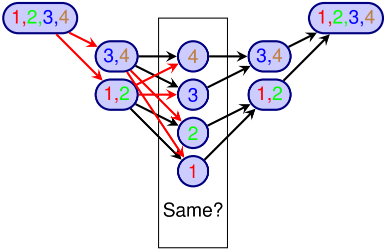
Other uses for coarse checkpoints

potential advantages to having coarse solutions around:

- lightweight high-time-resolution snapshots
- transient adjoint computation
- postprocessing
- coarse in-situ visualization

Redundant Coarse-Grid Error Detection

The redundant coarse problem may be used to trivially check for errors:



However, this is uninteresting and doesn't exploit the algorithm; can we do anything better?

τ-Correction Error Detection

ℓ_{check} $||\mathbf{F}^H(\hat{\mathbf{R}}\mathbf{x}) - \mathbf{Rb} - \tau||$ small

$\Uparrow$ ✓

ℓ_{fine} $||\mathbf{F}(\mathbf{x}) - \mathbf{b}||$ small

ℓ_{check} $||\mathbf{F}^H(\hat{\mathbf{R}}\mathbf{x}) - \mathbf{Rb} - \tau||$ large

$\Uparrow$ X

ℓ_{fine} $||\mathbf{F}(\mathbf{x}) - \mathbf{b}||$ small

- $\mathbf{x}$ solving fine problem $\mathbf{F}(\mathbf{x}) = \mathbf{b}$
- check residual of $\hat{\mathbf{R}}\mathbf{x}$ on a τ-corrected coarse grid

As $\hat{\mathbf{R}}\mathbf{x}$ solves the τ-corrected coarse grid problem, residual should be small

- incorrect result indicates error in fine grid residual evaluation
- identifies the location of the error

Conclusions

- multilevel methods allow efficient simulation
- can be leveraged to increase resiliency
- FAS Checkpointing allows for minimal overhead state reconstruction
- FAS Error Detection and Correction can be built into the solves
- other possibilities (Ensemble MG, etc.) loom

We've made progress towards having this working

- segmental refinement experiments and experience (Adams, 2012)
- FAS framework in place in PETSc