Overview of the Parallel Coupling Problem

S.E. Kruger, D. Bernholdt, R.Bramley

Madison Slow MHD Meeting

2007 December 5

with many thanks to J. Larson

Goals of this talk: Help bridge the gap in communication

- Physicists: “Move my data from here to there, computer scientists”
- Computer scientists:
  - What’s the data layout?
  - What’s the data decomposition?
  - Who controls mpi_init?
  - How much data are you moving?
- Physicists: Uhh, can’t you move my data for me?

4

1

Parallel Code Coupling Problem

- Parallel Coupling Problem:
  - How does one integrate two or more separate codes into an integrated simulation on parallel computers?

- There are many sub-problems associated with this in general:
  - "MxN" problem:
    How does one efficiently move data from one set of processors to another?
  - Load balancing:
    How does one make efficient use of multiple processors?
  - Temporal integration:
    How does one advance the coupled system accurately?
  - Spatial integration:
    How does one couple the different spatial discretization accurately; I.e., how does one do the interpolation while preserving the most important conserved quantities?
  - Software engineering:
    How does enable swapability?
  - Many others (including psychological and sociological)

Computer Science

Applied Math

Parallel Coupling Problem Has Been Faced by Many Communities

- **Climate Modeling Community**
  - **CCSM, ESMF, GSMF, SCCM, FMS, ..**
- **Space Weather**
  - **CISM, SWMF**
- **Fluid Dynamics/Structural Mechanics**
  - **ROCCOM, …**

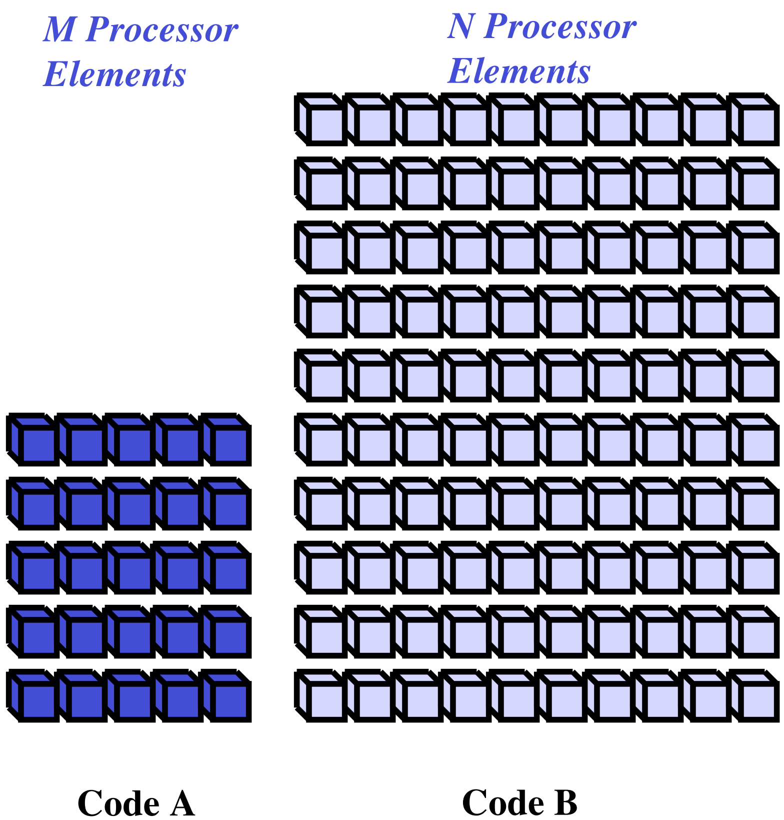
Despite many solutions to the previous problems, these solutions in general are not easily reused

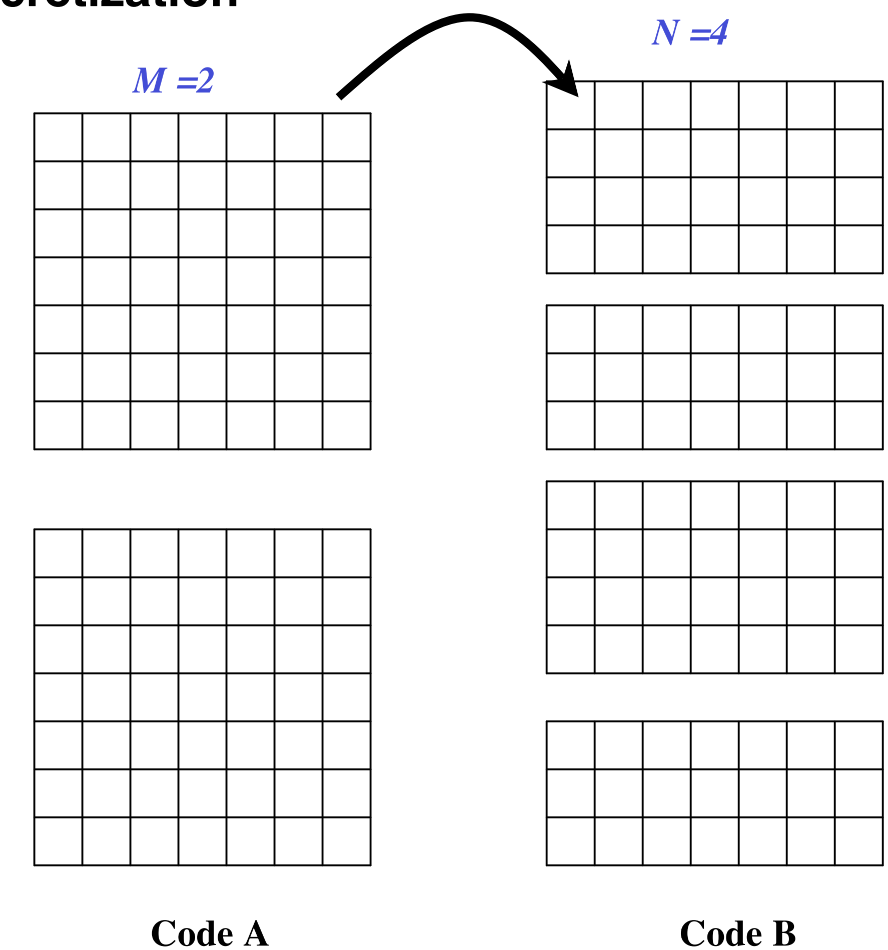
NEXT: Discuss MxN + interpolation

Will phrase things in terms used by CCSM parallel coupling framework

MxN issues is the data movement problem

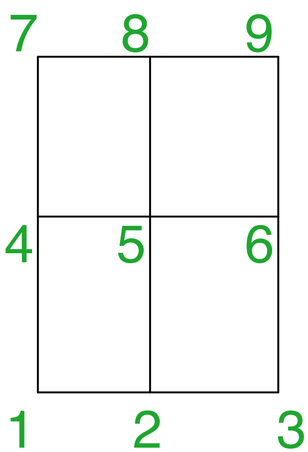
- **Problem:**
  Move field from code A (F_A) on M PEs to code B (F_B) on N PEs
- **Assume for now that we have exact same discretization (no interpolation).**
- **Consider just two-dimension discretization**





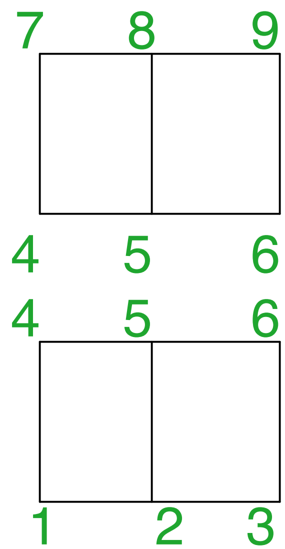
MxN issues can be cast as a matrix-vector multiplication

- **Consider very simple case:**



Code A

M* =*1



Code B

N* =*2

- **No interpolation and same ordering: Globally this is an identity matrix**
- **To handle domain decomposition: Two matrix-vector multiplies:**

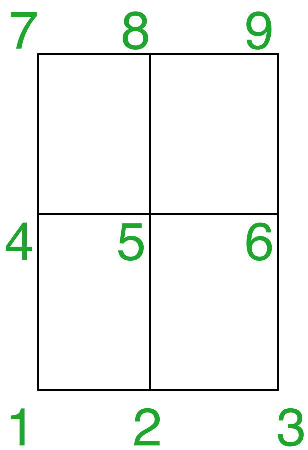
$$\begin{pmatrix} F_B^4 \\ F_B^5 \\ F_B^6 \\ F_B^7 \\ F_B^8 \\ F_B^9 \end{pmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} F_A^1 \\ F_A^2 \\ F_A^3 \\ F_A^4 \\ F_A^5 \\ F_A^6 \\ F_A^7 \\ F_A^8 \\ F_A^9 \end{pmatrix}$$

Diagonal matrix/sequential ordering implies contiguous in memory.

=> Simple mpi_send/recv

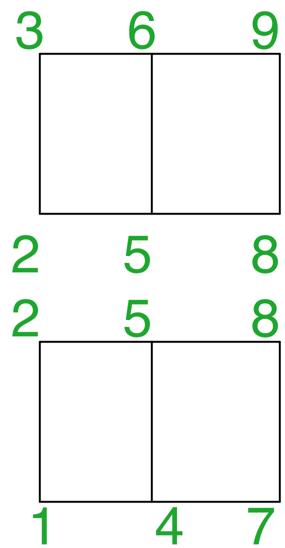
“Linearization” of communication can affect performance/difficulty

- **Consider very simple case:**



Code A

M =1



Code B

N =2

- **No interpolation and same ordering: Globally this is an identity matrix**
- **To handle domain decomposition: Two matrix-vector multiplies:**

$$\begin{pmatrix} F_B^2 \\ F_B^3 \\ F_B^5 \\ F_B^6 \\ F_B^8 \\ F_B^9 \end{pmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} F_A^1 \\ F_A^2 \\ F_A^3 \\ F_A^4 \\ F_A^5 \\ F_A^6 \\ F_A^7 \\ F_A^8 \\ F_A^9 \end{pmatrix}$$

Non-diagonal matrix/non-sequential ordering implies non-contiguous in memory.

$\Rightarrow$Data reordering needed

Many of the same issues in linear solver work

Interpolation represented easily in matrix-vector multiply framework

- Represent previous decomposition/interpolation operator as R_{BA}
- Represent interpolation as I_{BA}
- Then symbolically: $F_B = I_{BA} R_{BA} F_A$

- How should we calculate I_{BA}?
  - Define simple interpolants (linear, splines, …)
  - Define "conservative interpolation" to conserve various fluxes (used by CCSM in form of CPL6 code)
  - For fusion: Make sure DIV(B)=0

- Other issues:
  - Local interpolants => sparse matrices
  - Fourier interpolants => dense matrices
  - Finite-element representations may require mass matrix inversions
  - Generally, framework is the same

MPI Issues Affect Solution Methods

- Basic issue is who has the mpi_init and who allocates the processors
  - MPSD?
    One mpi_init
  - MPMD
    - Every component has its own mpi_init
    - Used by CCSM
    - Seems like it would easy, but in the end time synchronization makes things hard

4

1

Several solutions to the MxN problem using a strong coupling paradigm have been developed

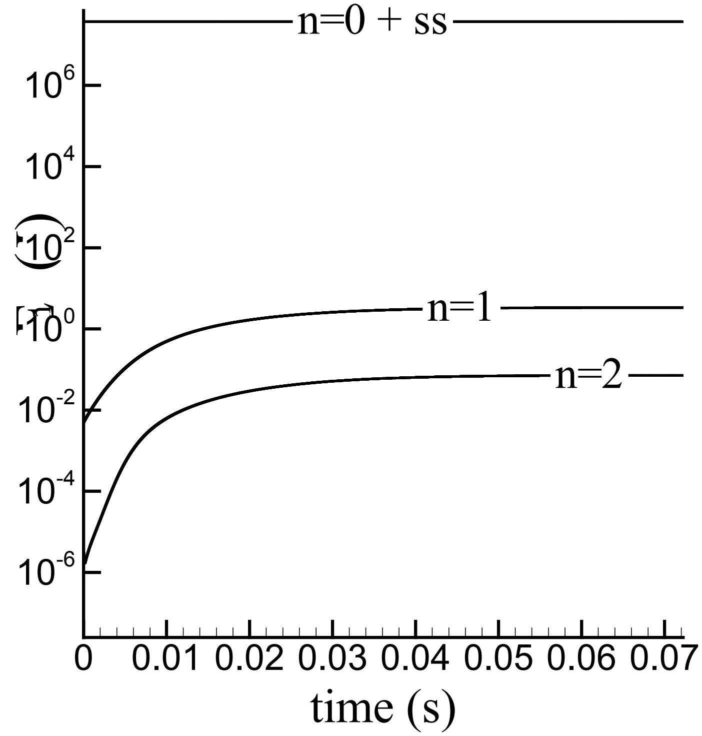
- InterCOMM
  - Developed by Sussman et.al.
  - Used by the CISM framework
  - “Venerable”
- IU MPIO-based solution
  - Felipe Bertrand and Randy Bramley
  - Showed scaling problems but Randy says he knows work around
  - Key advantage: automatic switching between file-based development and in-core memory.
- MCT
  - Larson, Jacob, et.al.
  - Used by the CCSM (truly production-level code)
  - 4 Creates separate group of processors for doing the multiplication
    - Consistent with MPMD approach of CCSM
    - Used mostly for explicit code coupling due to the
    - 1 Has additional bells and whistles especially for time discretization (extrapolation, interpolation, …)
- Various RMI approaches (Babel-RMI, …)

Issues of time dependence impact computer science coupling to be discussed next

- **What is the temporal dependence of the source?**

$$\vec{E} + \vec{v} \times \vec{B} = \eta \vec{J} + \vec{F}_e^{rf}(x,t)$$

- **For the plasma parameters that we are studying:**
  - **Slowing down time is ~100msec-1 sec?**
  - **Mod-B changes slow and small**



- **MHD time scales:**
  - **NIMROD time step ~ 1 τ_A ~1E-7 sec**
  - **Tearing mode growth rate ~ 1-100 msec**
  - **Total simulation time ~ 100 msec**

Plan of attack (note: different notation than Eric)

- **Phase 0:**
  Phenomenological model for F_{RF}
  a. Axisymmetric source, b. Non-axisymmetric source

- **Phase 1:**
  Assume F_{rf} is ~constant in shape over simulation time
  - **Coupling procedure:**
    - **Calculate $F_{rf}(x)$ from same equilibrium file that NIMROD uses**
    - **NIMROD reads $F_{rf}(x)$ and uses it like phenomenological sources (include such time dependence as phasing)**

- **Phase 2:**
  Assume F_{rf} is much slower than NIMROD time step
  - **Need to calculate source as NIMROD runs, but do not need to do it every time step**

- **Phase 3:**
  Calculate F_{rf} at every time step

- **Phase 4:**
  Incorporate more advanced closures

Phase 1: Time-independent source

- **NIMROD Input: EFIT equilibria**
- **GENRAY Input: EFIT equilibria**
- **CQL3D Output/NIMROD Input: J_{RF}(R,Z,phi)**
- **Coupling: No coupling while NIMROD is run**
- **Goal:**
  - **MHD people to understand how to run GENRAY/CQL3D**
  - **Machinery for interpolating fields from GENRAY to NIMROD**
  - **Understand resolution problems for realistic current deposition profiles**

- **Numerical details**
  - **Interpolation is always fraught with peril**
  - **NIMROD's C0 elements have particular problems**

Phase 2: Time-dependent source, file-based coupling

- NIMROD Input: EFIT equilibria
- GENRAY Input: NIMROD n=0 magnetic fields @ (R,Z,phi) locations
- GENRAY Output/NIMROD Input: Q_{RF}(R,Z,phi,velocity,time) @ (R,Z,phi) locations
- Coupling: File-based as the simulation progresses
- Goal:
  - Get moment machinery worked out
  - Revisit previous calculations
  - Spatial locations
  - Preliminary explorations of temporal dependence (if F_{RF} changes then we need to work harder)

- Questions:
  - Should we use the fast MHD framework here?
  - Seems like it would be easy (on NIMROD side, we DELETE file and INQUIRE about file to handle the signaling)

- Numerical details:
  - Genray needs to accept B(R,Z,phi) and not flux.
  - NIMROD needs to put out B in a form that Genray can make sense of
  - NIMROD needs to take moments of Q_{RF} accurately

Goals of Phase 3: Time-dependent source, in-core coupling

- **NIMROD Input: EFIT equilibria**
- **GENRAY Input: NIMROD n=0 magnetic fields @ (R,Z,phi) locations**
- **GENRAY Out/NIMROD In Q_{RF}(R,Z,phi,velocity,time) @ (R,Z,phi)**
- **Coupling: In core as the simulation progresses**
- **Goal:**
  - **Learn how to do MxN coupling**

- **Questions:**
  - **What MxN solution do we use?**
    - **IU MCIO solution seems like a natural step based on previous solution**

- **Numerical details:**
  - **GENRAY needs to be “librarified” to let NIMROD control mpi_init.**

Phase 4: Time-dependent source with realistic closures

- Add realistic closures
- Can be developed in parallel to other phases.
- Could enter at Phase 2.

Other issues

- How do we test the coupling?
  - Can we do an Ohm’s law like the RF calculation of deposited power?