CellML Workshop 2008 – openCMISS and FieldML/CellML

Chris Bradley

Department of Physiology, Anatomy and Genetics, University of Oxford
Cellular processes

- Magnesium Buffering
  - MgATP
  - MgADP
  - MgPi
- Calcium Buffering
  - Calmodulin
  - Ca^{2+}
- Glycogenolysis
  - Contraction
- Intrinsic Buffers
  - Na^+
- MgADP + Pi + H^+
- H_2PO_4^-
- CO_2 transport
- 3Na^+
- 2K^+
- 3MgATP
- 3MgADP + 3Pi
Bidomain representation

Cell model = 

Extracellular Space
σ_e, Φ_e

Cell Membrane

Intracellular Space
σ_i, Φ_i

I_{ion}
Bidomain equations

\[ \nabla \cdot [(\sigma_i + \sigma_e) \nabla \Phi_e] = -\nabla \cdot (\sigma_i \nabla V_m) + I_{\text{stim}1} \]

\[ \nabla \cdot (\sigma_i \nabla V_m) + \nabla \cdot (\sigma_e \nabla \Phi_e) = A_m \left( C_m \frac{\partial V_m}{\partial t} + I_{\text{ion}} \right) - I_{\text{stim}2} \]

\[ V_m = \Phi_e - \Phi_i \]

Where:
- \( V_m \) = transmembrane potential
- \( C_m \) = membrane capacitance
- \( A_m \) = membrane surface/volume ratio
- \( I_{\text{ion}} \) = ionic source current from cellular model
- \( I_{\text{stim}} \) = stimulus current
- \( \Phi_e \) = extracellular potential
- \( \Phi_i \) = intracellular potential
- \( \sigma_e \) = intracellular conductivity tensor
- \( \sigma_i \) = intracellular conductivity tensor
Future electrical activation problem:

Average human heart is approx 130 mm x 90 mm x 70 mm = $8.19 \times 10^5$ mm$^3$. Assume 50% is ventricle.

For 100 $\mu$m spacing would need $4.23 \times 10^8$ computational points (cp).

For 30 ODEs/cp we would have $1.27 \times 10^{10}$ ODEs to solve at each time instance.

Assuming 100 flop/ODE we would have $1.27 \times 10^{12}$ flop per time instance.

A 1 ms time step will give $1.27 \times 10^{15}$ flop/s real simulation time or $5.26 \times 10^{16}$ flop/min real time.
Solution time calculation

• A 2.4 GHz Intel Core 2 can achieve ~ 1.7 Gflops with linpack (best case!)
• To simulate 1 min real time on 1 processor would therefore take $4.48 \times 10^7$ s or **518.5** days!
• Or, in other words, to get the solution time down to 1 day would require 519 processors assuming perfect speedup (not very likely!)

• We are dealing with very, very big problems that require large parallel computers!
openCMISS

• Re-engineering of CMISS computational engine.
• Want
  – a library based approach to enable use in multiple applications.
  – Modular, easily extendable and programmable.
  – MPI based distributed memory + OpenMP + serial code
  – Easy to understand and program by novices.
• Open source on sourceforge
  – http://sourceforge.net/projects/opencmmiss
• Fortran 95.
• Object “based”
• Still in the preliminary stage – about 73,000 lines.
Parallelisation?
Domain decomposition
Domain decomposition
Ghosting
Mesh Decomposition
Mesh Decomposition
Regions

Regions contain:
• Coordinate system
• Nodes
• Meshes
• Fields
• Problems
• Daughter and parent regions
Fields

- Fields are the central object for storing information and framing the problem.
- Fields have a number of field variables i.e., $u$, $\partial u/\partial n$, $\partial u/\partial t$, $\partial^2 u/\partial t^2$.
- Each field variable has a number of components.
- A field is defined on a decomposed (broken up) mesh.
Field parameter vector

Component, $u_1$

Field, $u$

Component, $u_2$

Parameter vector, $x$

Local

Ghost

Update

Global DOFs

Proc 0

Proc 1
How can each field variable component vary over the mesh?

- Element-based
- Node-based
- Point-based
Generic temporal-spatial problem

1. Loop over time
   - Update current field parameters from previous (or initial) time step

2. Loop over elements
   - Integrate Cell Models
   - Calculate element equations
   - Endloop

3. Assemble global equation system
4. Solve spatial system
5. Calculate field parameters at next time step
6. Endloop
Generic temporal-spatial problem (distributed)

1. Loop over time
2. Update current field parameters from previous (or initial) time step
3. Start ghost transfer of previous time step solution
4. Loop over internal elements
5. Integrate Cell Models
6. Calculate element equations
7. Endloop
8. Wait until transfer has finished
9. Loop over boundary+ghost elements
10. Integrate Cell Models
11. Calculate element equations
12. Endloop
13. Assemble global equation system
14. Solve spatial system
15. Calculate field parameters at next time step
16. Endloop
Problem/Field description

Fields:
- Geometric
- Fibre
- Material ($\sigma_i$, $\sigma_e$, $A_m$, $C_m$ etc.)
- Potential ($V_m$, $\Phi_e$, $\Phi_i$ etc.)
- Source (cell variables)
- Source material (cell parameters)
Source material field

• Want to be able to vary any cell parameter spatially

• Two options
  – Interpolate every parameter before evaluating a cell model.
    • Most general
    • Computationally expensive as most parameters are constant
  – Only allow certain parameters to vary and interpolate them
Fields Involved

- Geometric field
- Fibre field
- Potential fields ($V_m$, $\Phi_e$, $\Phi_i$ etc.)
- Material fields ($\sigma_i$, $\sigma_e$, $A_m$, $C_m$ etc.)
- Source field (cell state variables)
- Constant source material field (cell parameters)
- Varying source material field (cell parameters)
Distributed solvers
Cell Model Evaluation Scope

- Continuum Field Calculations
- Cell ODE Integrations
- Cell RHS Evaluation

(openCMISS) → CELL

Time:
- $t_0$
- $t_1$
- $t_2$
Acknowledgement