Optimisation methods for Monte Carlo

Stuart Ansell

Rutherford Appleton Labs, Oxfordshire, U.K.

October 2, 2006

OutLine

Required Optimisations to MC codes

Simulation geometries are becoming much more complex

- Point tallies are energy/time bound AFTER track distance (Don't need to record the whole time/energy table)
- Point tallies with windows and geometry limits (All deterministic tallies should be non-model scoped)
- Avoid continuous of create/destroy dynamic casts
- Free initialization memory
- Page faults from long goto's account for 60% of runtime CPU [MCNPX].
- There is only one stack space: Don't Waste It
- Probability bias the simulation

Realization of Optimisations to MC codes

- Memory allocated dynamically once

Realization of Optimisations to MC codes

- Memory allocated dynamically once
  - Larger running footprint
  - 30% Reduction in run-time

- Memory allocated dynamically once
  - Larger running footprint
  - 30% Reduction in run-time
- Point tally improvement

Realization of Optimisations to MC codes

- Memory allocated dynamically once
  - Larger running footprint
  - 30% Reduction in run-time
- Point tally improvement
  - Window needs to be correctly set
  - Faster than non-focused point tallies
  - x1000000 faster than without Point tallies

Realization of Optimisations to MC codes

- pre-run Weight window generator

Realization of Optimisations to MC codes

- pre-run Weight window generator
  - Cannot Fail(unlike WWG)
  - Uses Prior simulation selection in available

Realization of Optimisations to MC codes

- pre-run Weight window generator
  - Cannot Fail(unlike WWG)
  - Uses Prior simulation selection in available
- Pipe line copies are minimised

Realization of Optimisations to MC codes

- pre-run Weight window generator
  - Cannot Fail(unlike WWG)
  - Uses Prior simulation selection in available
- Pipe line copies are minimised
  - Wrap data sets into object

Job Control

- We can run 300 MCNPX jobs an hour.

Job Control

- We can run 300 MCNPX jobs an hour.
  - Automate run/submission/analysis

Job Control

- We can run 300 MCNPX jobs an hour.
  - Automate run/submission/analysis
- Need active geometry handling. [MCNPX included geometry is extremely poor]

Job Control

- We can run 300 MCNPX jobs an hour.
  - Automate run/submission/analysis
- Need active geometry handling. [MCNPX included geometry is extremely poor]
- The easiest is a programming language
  - Compiler checking avoid stupid run-time problems
  - Parameters within a tightly defined environment
  - Pre-run verification

Example

122 5 0.11102 1 -2 3 -4 5 -6 (-11 : 12)

Example

In object with surfaces a,b,c,d,e,f

- Monte Carlos depends on boolean algebra
- The algebra density is proportional to the component[4]

Example

122 5 0.11102 1 -2 3 -4 5 -6 (-11 : 12)

Example

In object with surfaces a,b,c,d,e,f

- Monte Carlos depends on boolean algebra
- The algebra density is proportional to the component[4]
- It is mostly hidden

Proof.

$a := surface\ x = 5$ (px 5)

$b := surface\ x = 10$ (px 10)

$b => a$ and $a' => b'$

Optimisation Algebra (cont)

Example

122 5 0.11102 $ab'cd'ef'(g' + h)$

$a => b$

$b' => a'$

Substitution of $a => b$ by (a'+b)

Objective is to minimise literals **terms**

Silicon chip optimisation : Minimise number of links

Current strategy:

- Examine system and expand complements
- *AND* additional knowledge for parallel planes, cylinders etc.
- Quinie-McClusky method to produce minimum both SOP and POS *(DNF and CNF)*
- Factorize (FPD and *Good Factor*)
- Remerge the tree by top-base substitution
- Factor for a *humanly present* form

Optimisation Algebra (cont)

- Take a long time. Restricted to complementary object roll out
- The output often incomprehensible
- Faster QM method needed (or to be avoided)
- By far the most useful MCNPX code for non-MCNPX applications

Basic moderator Equations (*Wrong*)

$$D\nabla^2\phi(t) - \Sigma_a\phi(t) + s = -\frac{1}{v}\frac{\delta\phi(t)}{\delta t}$$

e.g. Sigma-Pile solution for a cube

$$\phi(t) = const\,\exp(-\frac{1 + B^2 L_T^2}{t_d})$$

$$B_{lmn}^2 = (\frac{l\pi}{a})^2 + (\frac{m\pi}{b})^2 + (\frac{n\pi}{c})^2$$

$\phi(t)$ = Neutron Flux(time)

D = Diffusion Length

t_d = Diffusion Time

L_T = Transport length (D^2/Σ_a)

TS2 ISIS second target station project

- Solutions by perturbation
  - *e.g. reflector / moderator*

Other Solutions of Moderator Equations

- Solutions by perturbation
  - *e.g. reflector / moderator*
- Solutions by overlap
  - *e.g. Vanes*

Other Solutions of Moderator Equations

- Solutions by perturbation
  - *e.g. reflector / moderator*
- Solutions by overlap
  - *e.g. Vanes*
- Solutions by boundary instability
  - *e.g. Castles*

They are all generic equations:

1. Solve the sum series for the optical boundary problem
2. Use convolution/subtraction methods
3. Repeat for all higher orders (set fundamental length by factor 2,3,4 etc.)
4. Create a set of solutions and index them.
5. *Find the initial source distribution in terms of each boundary solution*

6. Substitute

$$\phi(r,t) = \sum_{index} T_{index}(t) * FSol_{index} \qquad (1)$$

7. into

$$L_T^2 \Delta\phi(r,t) - \phi(r,t) = t_d \frac{\delta\phi(r,t)}{\delta t} - \frac{s(r)\delta(t)}{\Sigma_a} \qquad (2)$$

Conclusions

- Code your multi-parameter runs

Conclusions

- Code your multi-parameter runs
- Get a replacement for MCNPX (Geant ??) and integrate to the sample simulation

Conclusions

- Code your multi-parameter runs
- Get a replacement for MCNPX (Geant ??) and integrate to the sample simulation
- Mathematics should still be used with modern design

Conclusions

- Code your multi-parameter runs
- Get a replacement for MCNPX (Geant ??) and integrate to the sample simulation
- Mathematics should still be used with modern design
- We still don't know how to build the best moderator