Mapping the Monte Carlo dose calculation algorithm on the GPU

Sami Hissoiny
Elekta LTD.

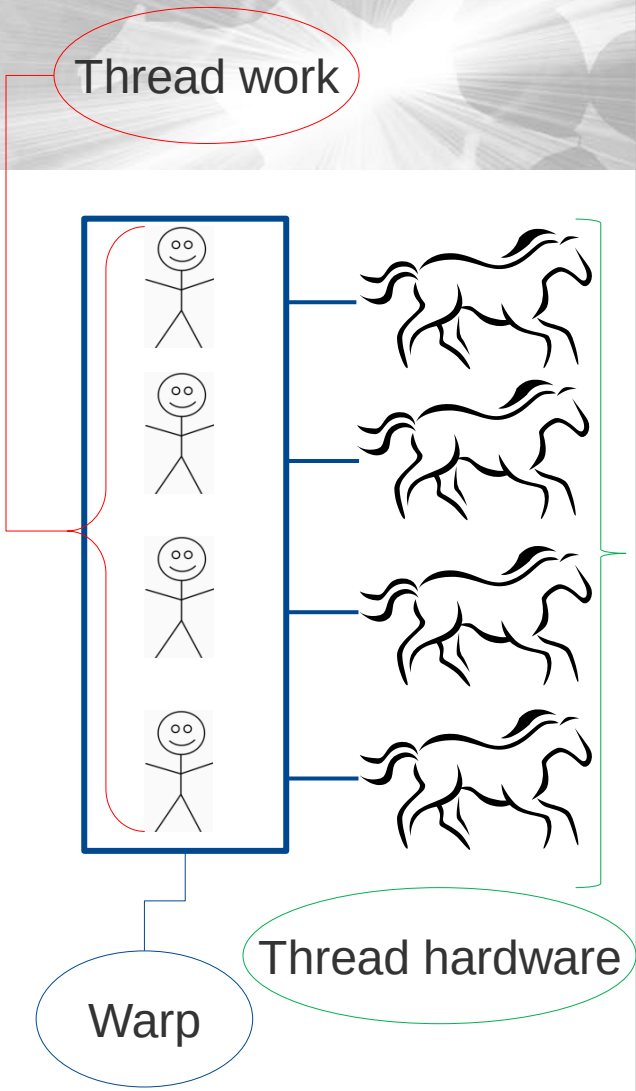
Human Care Makes the Future Possible

ELEKTA

Disclaimer

- Sami Hissoiny works for Elekta LTD
- Elekta has a license for GPUMCD. The benchmarks in this presentation are based on GPUMCD.

Situation



ELEKTA

Situation – Classic photon loop

```
while ( photon.energy > minEnergy)
{
  sample mfp/distance to next interaction
  advance to next interaction point
  if(out of bounds)
       terminate
  sample interaction type
  simulate interaction
  if(secondary particles created)
       simulate secondary particles
}
```

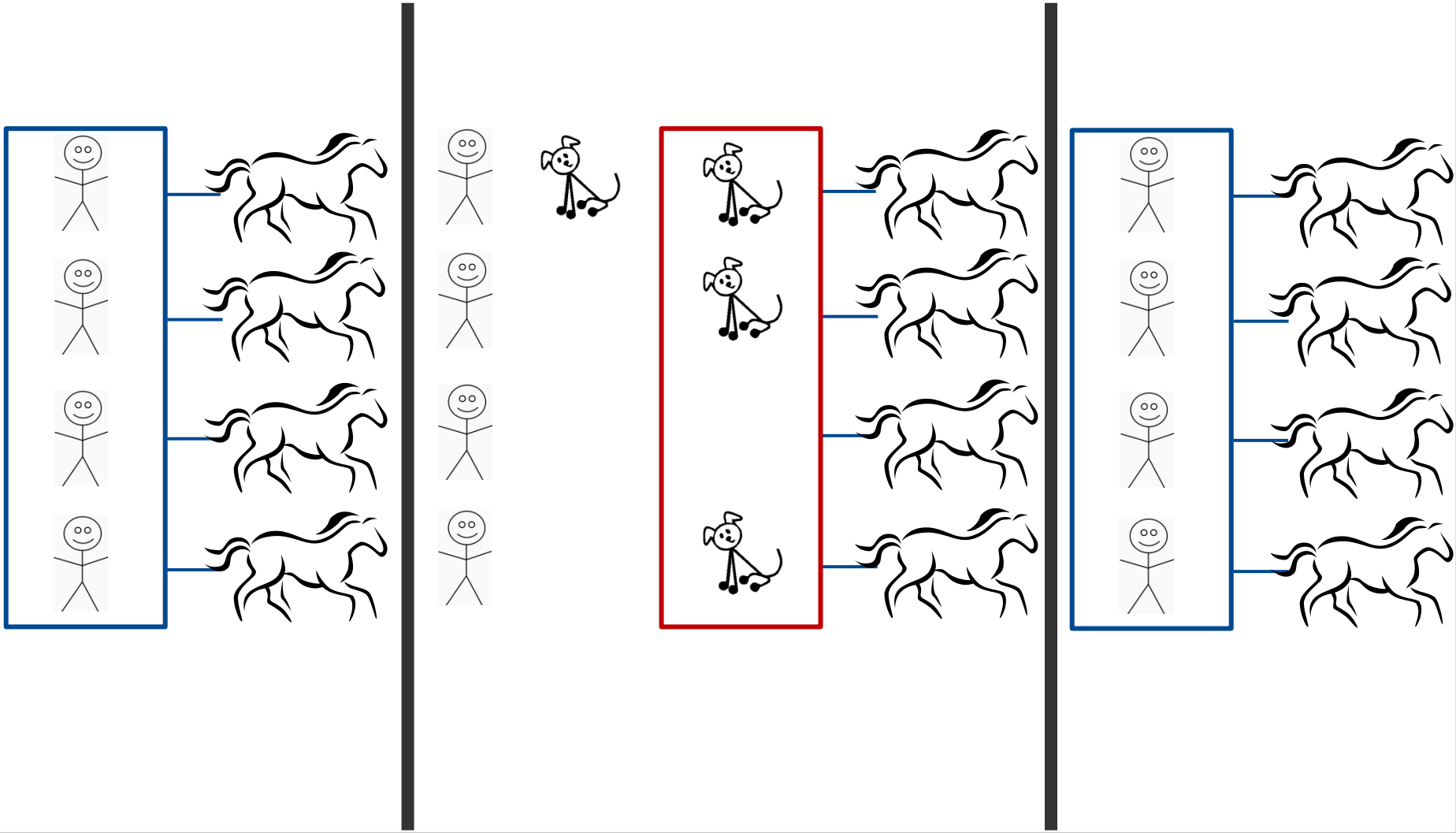
Classic mapping : one photon <-> one thread

Situation – Classic photon loop

```
while ( photon.energy > minEnergy)
{
  sample mfp/distance to next interaction
  advance to next interaction point
  if(out of bounds)
       terminate
  sample interaction type
  simulate interaction
  if(secondary particles created)
       simulate secondary particles
}
```

Problem

simulate secondary particles



ELEKTA

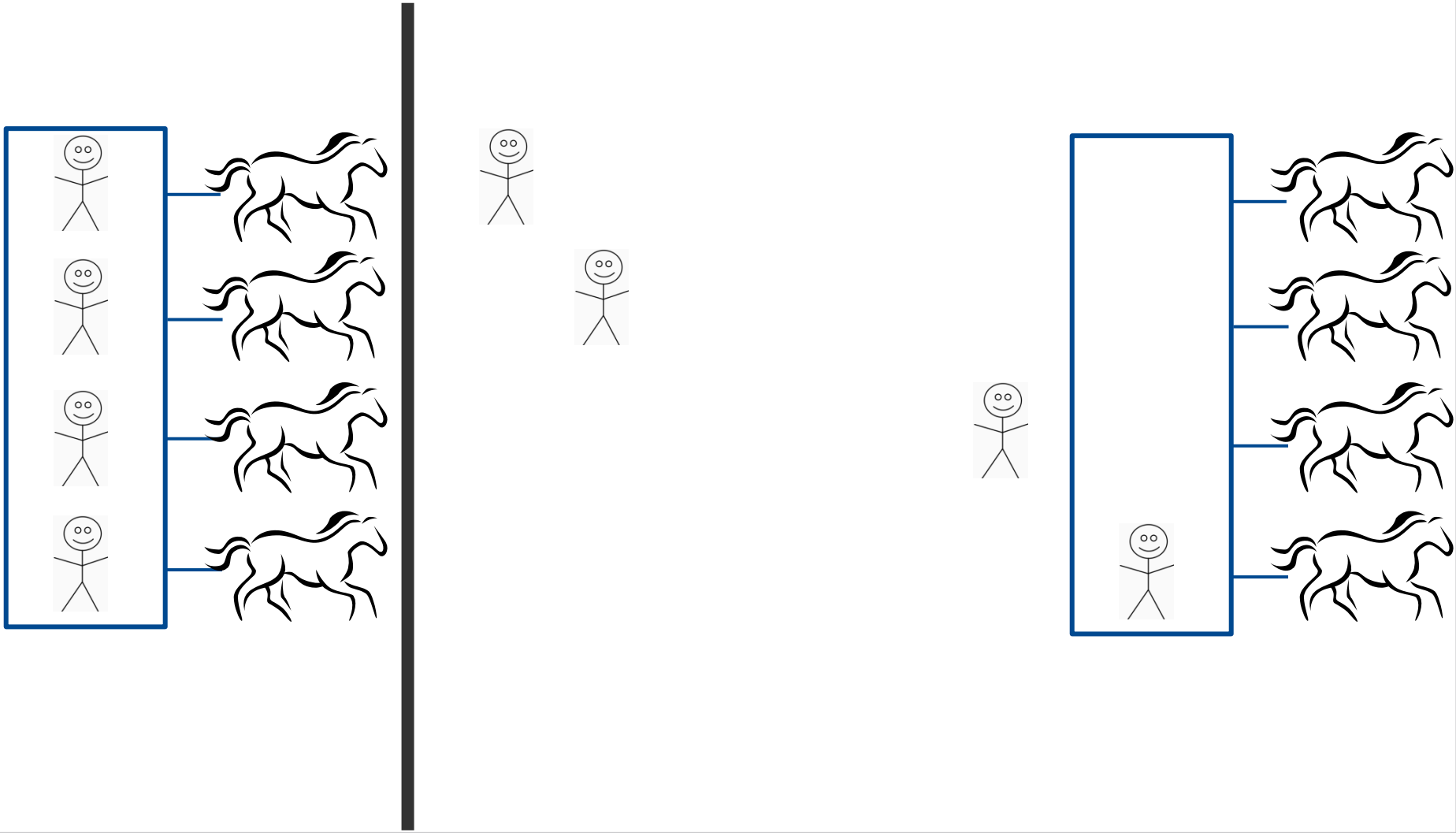
Solution

Seperate buffers and delayed secondary particle simulation

```
while ( photon.energy > minEnergy)
{
  sample mfp/distance to next interaction
  advance to next interaction point
  if(out of bounds)
       terminate
  sample interaction type
  simulate interaction
  if(secondary particles created)
       {store secondary photons, store secondary electrons}
}
```

Problem

```
while ( photon.energy > minEnergy)
 if(out of bounds)
      terminate
```



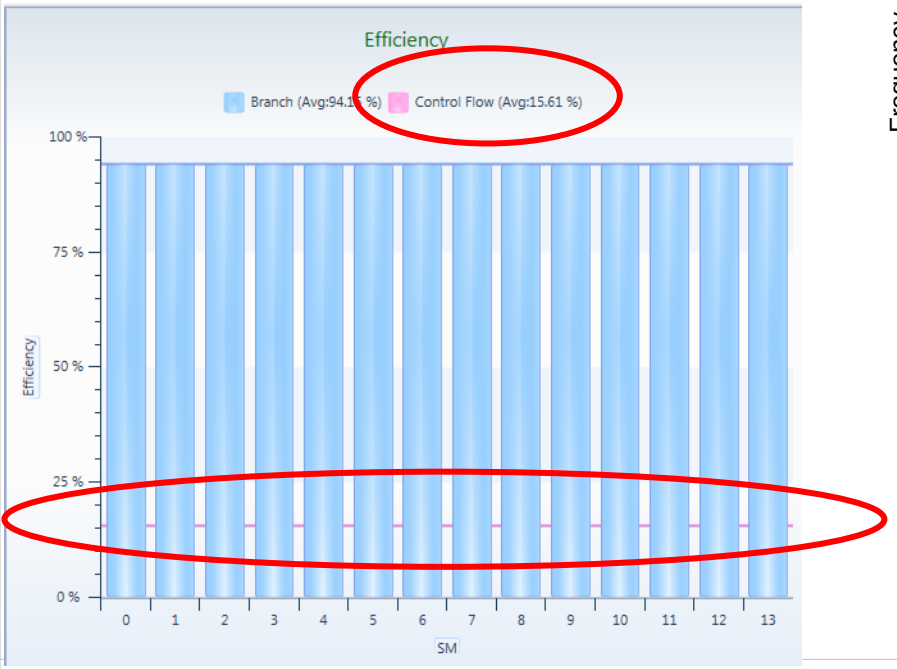
Problem

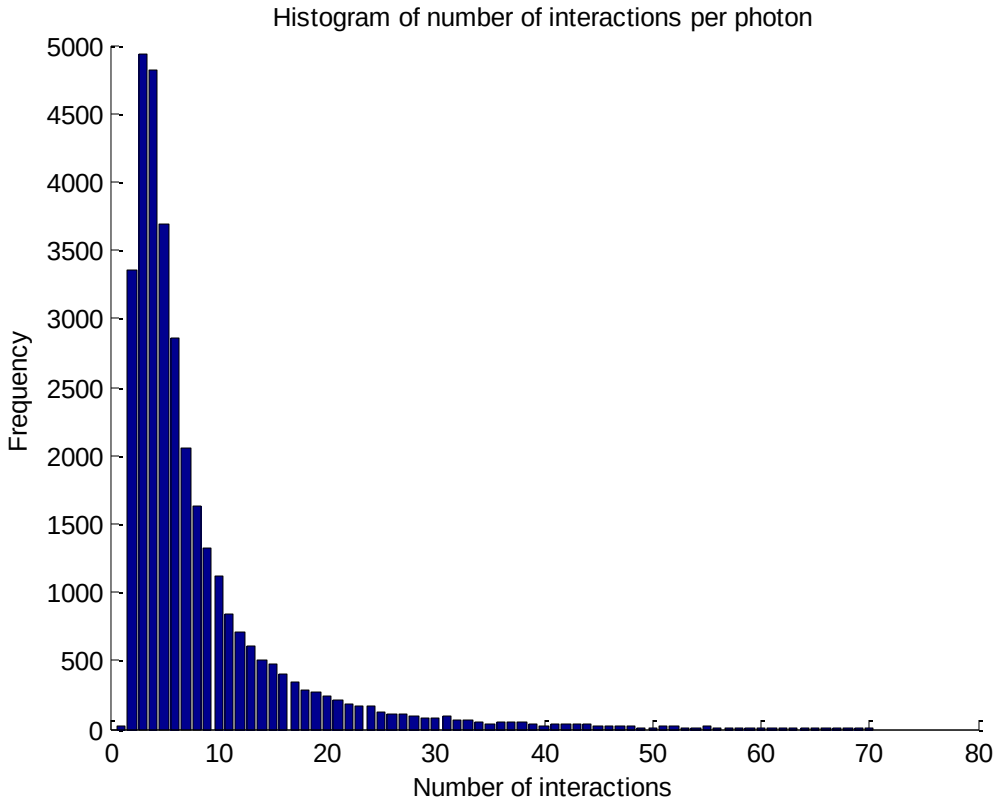
```
while ( photon.energy > minEnergy)
 if(out of bounds)
     terminate
```

On average, 4 interactions per photon.

17% have 4±1 interactions.

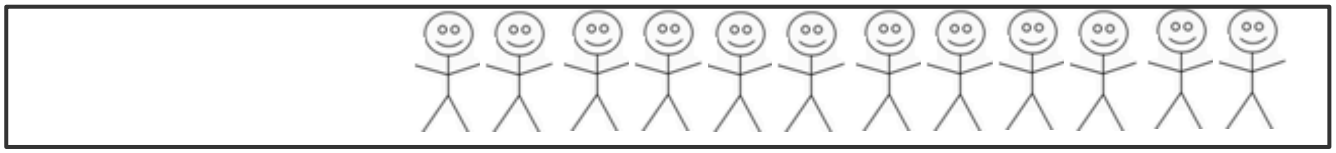
6.56% have 14+ interactions:
89% prob to have at least one particle with 14+ interactions in a warp.

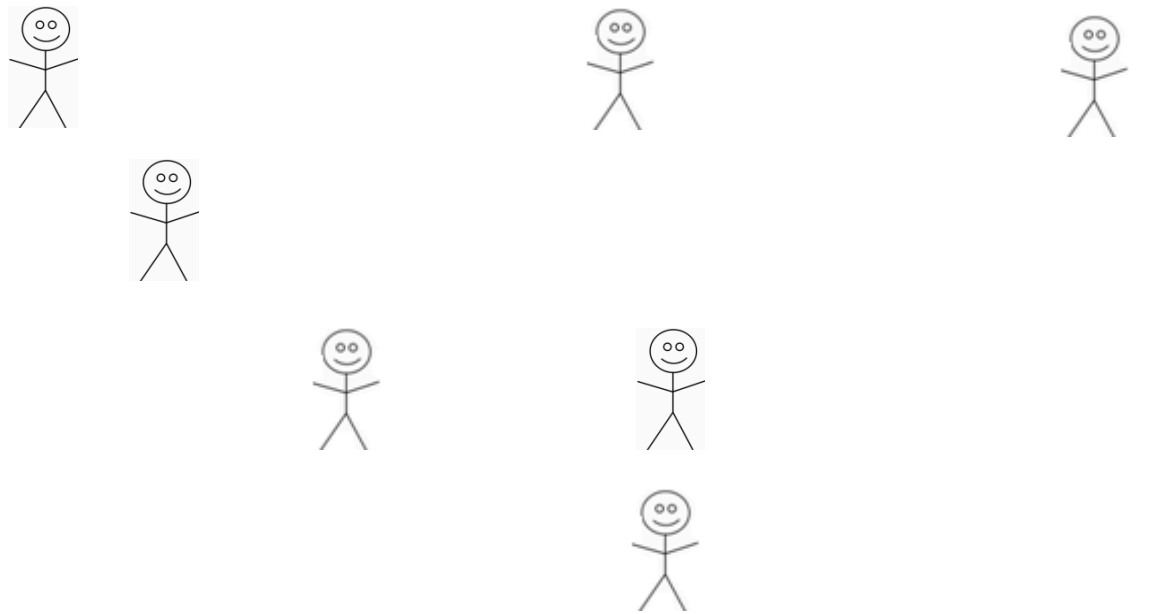


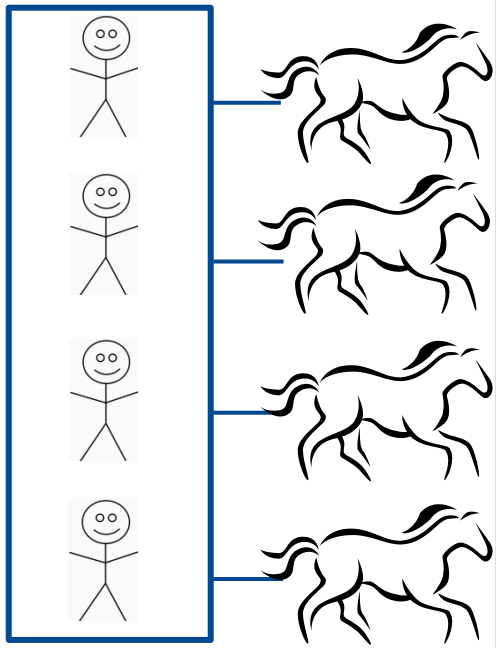


ELEKTA

Solution



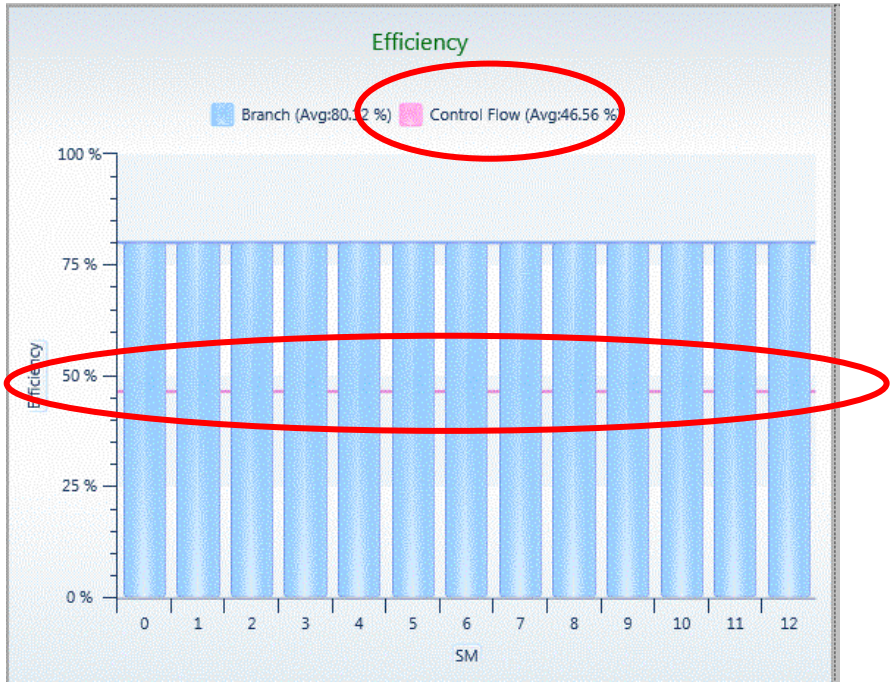


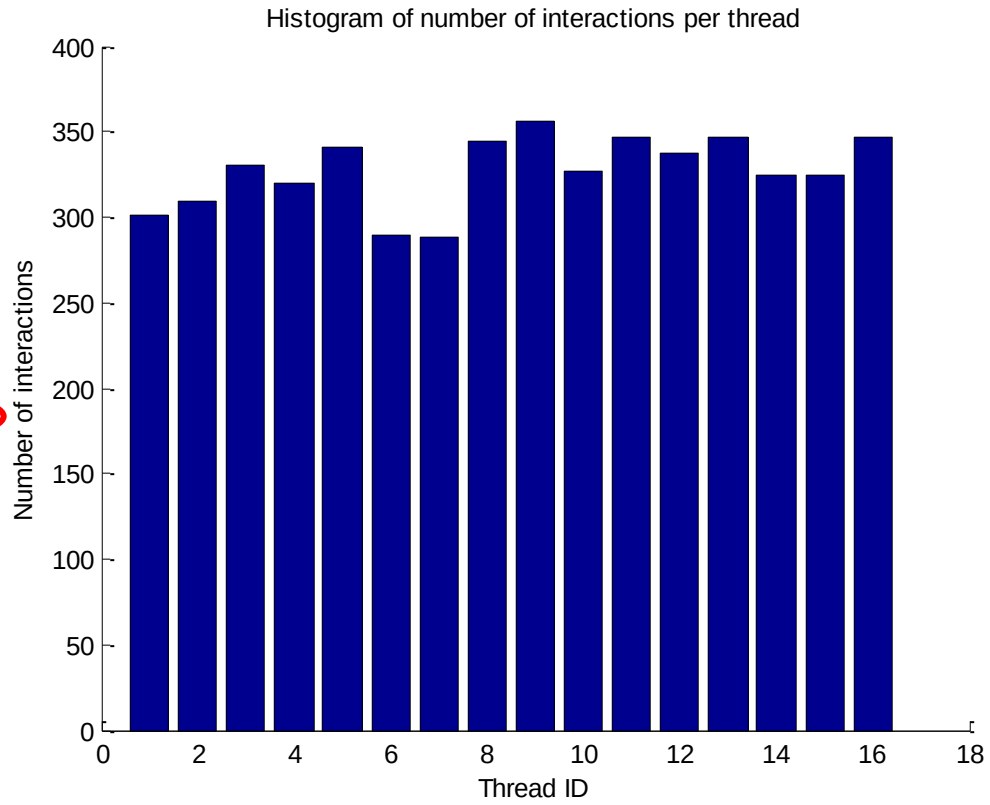


Solution – atomic pool of particles

```
while ( [my particle alive] or [particles in the pool])
{
  if (my particle is terminated)
       fetch new particle from the pool
  sample mfp/distance to next interaction
  advance to next interaction point
  if(out of bounds)
       terminate
  sample interaction type
  simulate interaction
  if(secondary particles created)
       {store secondary photons, store secondary electrons}
}
```

Solution – atomic pool of particles





	Relative Time
Base	1.00
Atomic pool	0.75

ELEKTA

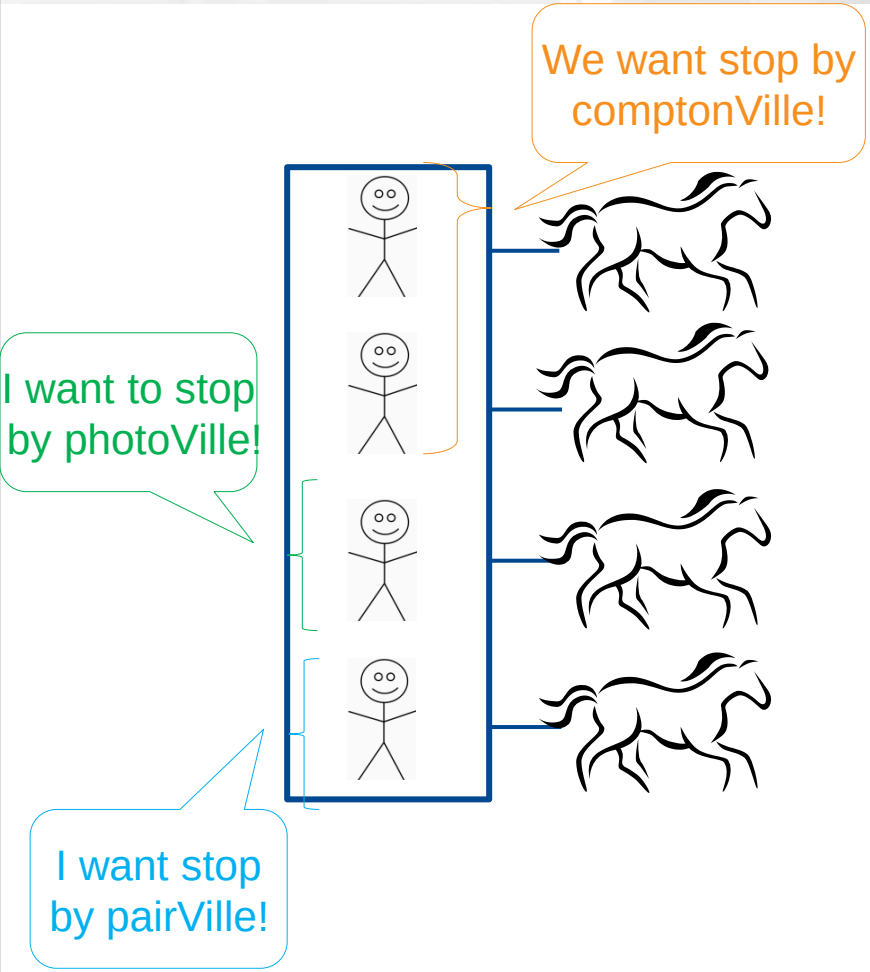
Problem\solution

```
advance to next interaction point
```

- Classic raytracing approach : accumulate « crossed » MFPs until point of interaction
  - Every particle (within a warp) as slow as the particle that has to cross the highest number of boundaries.
  - Lots of random memory accesses (material index, density, cross section values) for every thread in every voxel.
- Woodcock/delta interaction approach: homogenize the phantom w.r.t. total cross section in every voxel
  - No need for explicit raytracing. Advance the particle by x centimeters.
  - Risk of stopping in places where we did not have to -> fictitious interaction
- Woodcock tracking makes the photon kernel ~18 times faster.

Problem

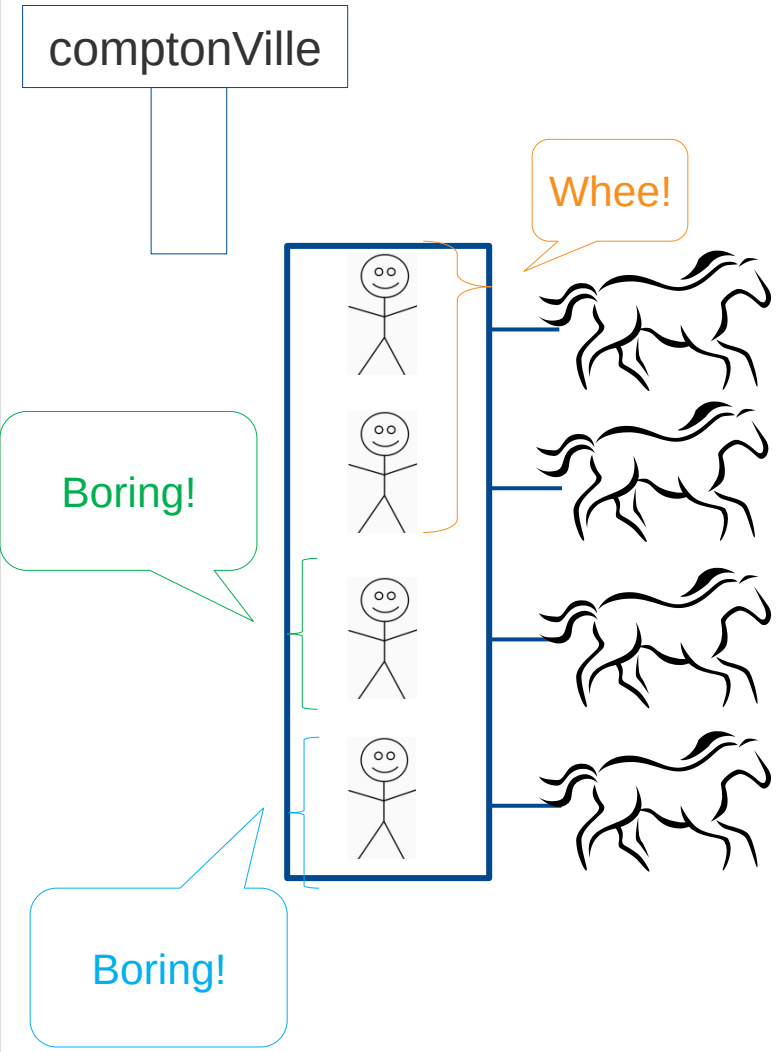
`simulate interaction`



ELEKTA

Problem

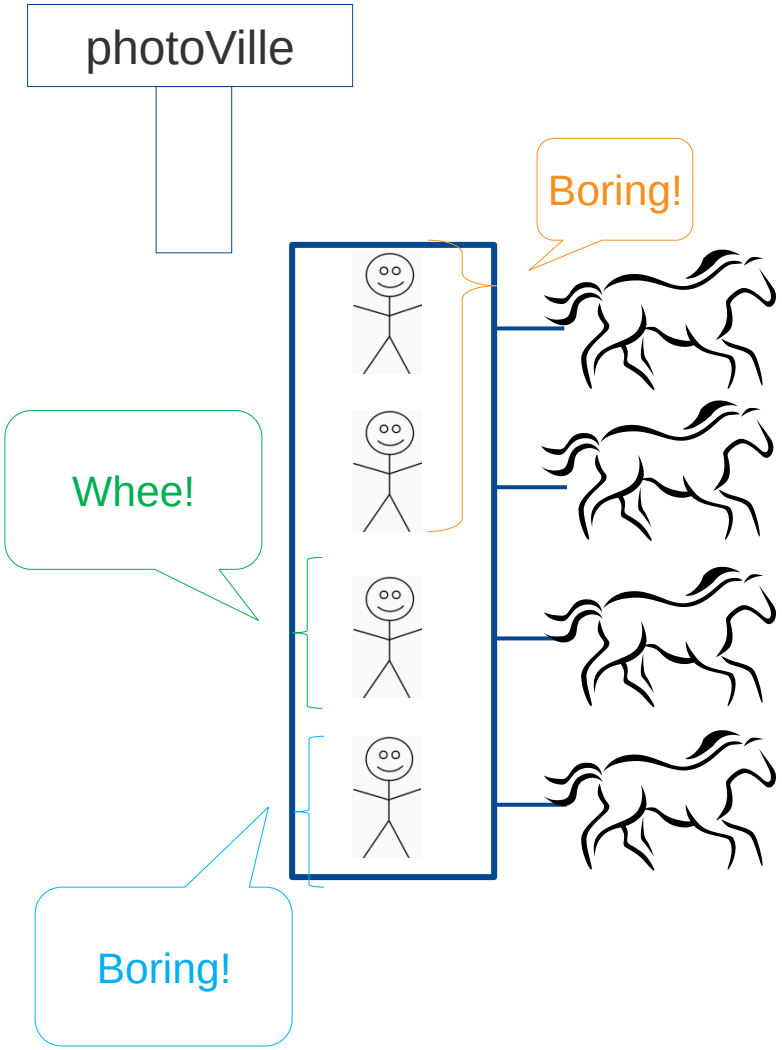
`simulate interaction`



ELEKTA

Problem

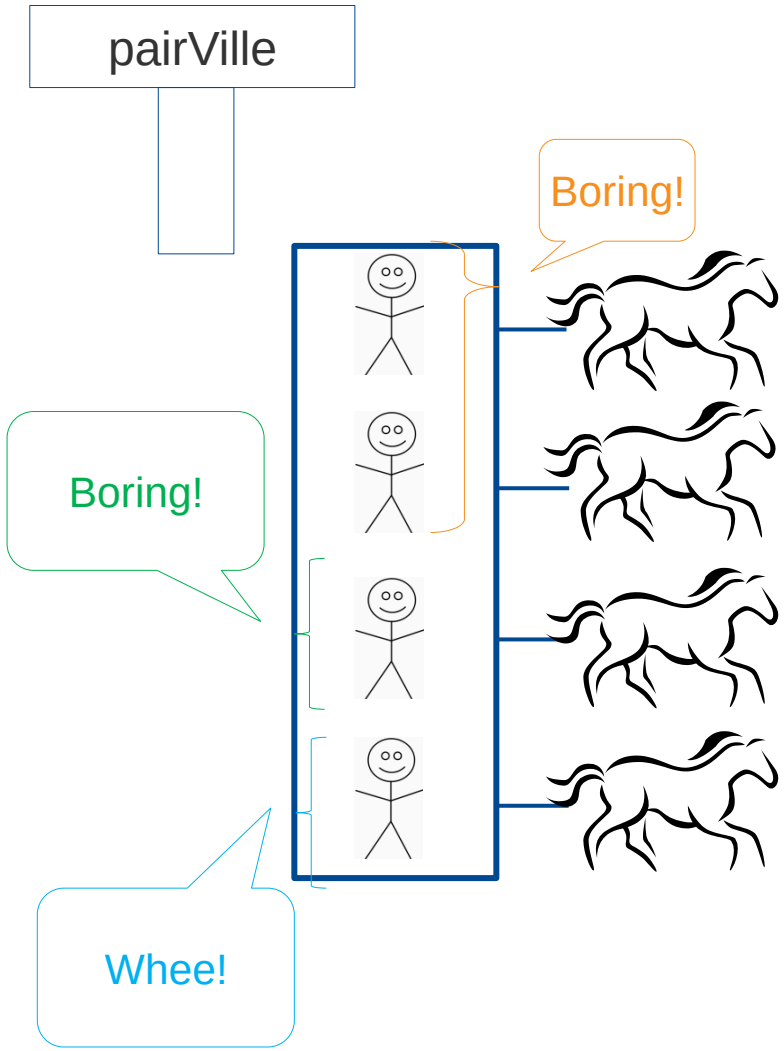
`simulate interaction`



ELEKTA

Problem

`simulate interaction`



ELEKTA

Problem `simulate interaction`

But is it really?

	Relative time
Base	1.00
Base + one interaction	1.09
Base + one compton	1.06

```
while ( photon.energy > minEnergy)
{
   sample mfp/distance to next interaction
   advance to next interaction point
   if(out of bounds)
      terminate
   sample interaction type
   simulate interaction
   if(secondary particles created)
      store secondary particles
}
```

Problem `simulate interaction`

But is it really?

	Relative time
Base	1.00
Base + one interaction	1.09
Base + one compton	1.06

```
while ( photon.energy > minEnergy)
{
   sample mfp/distance to next interaction
   advance to next interaction point
   if(out of bounds)
      terminate
   sample interaction type
   simulate interaction
   simulate interaction
   if(secondary particles created)
      store secondary particles
}
```

Problem `simulate interaction`

But is it really?

	Relative time
Base	1.00
Base + one interaction	1.09
Base + one compton	1.06

```
while ( photon.energy > minEnergy)
{
   sample mfp/distance to next interaction
   advance to next interaction point
   if(out of bounds)
      terminate
   sample interaction type
   simulate interaction
   simulate compton
   if(secondary particles created)
      store secondary particles
}
```

Not much to gain by enforcing all interactions to be the same within a warp.

Problem

- Scoring
  - A graphics card has loads of GPU<->memory bandwidth, if the correct access patern is used.
  - ~Never used in Monte Carlo

	K20c	8800GT
No scoring	1.00	1.00
Non-atomic scoring	1.10	1.08
Atomic scoring	1.12	1.16

Conclusions

- Monte Carlo can run on the GPU, but it is definitely not ideal.
- Several modifications to the « CPU way » of doing Monte Carlo should be applied to better map the problem on the GPU architecture.