A brief introduction to McStas and McXtrace

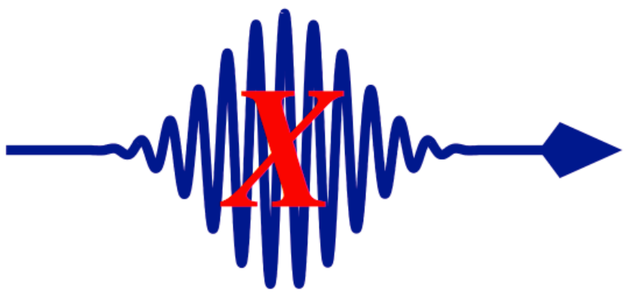
Peter Willendrup[1,2], Erik Knudsen[1], E Farhi[3]

[1]NEXMAP, Physics Department, Technical University of Denmark, Denmark

[2]ESS Data Management & Sofware Center, Denmark

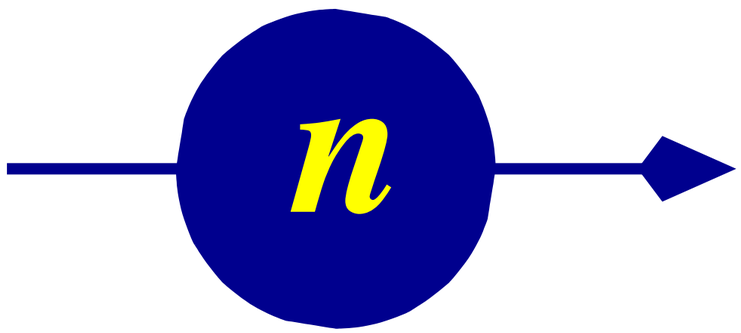
[3]Spectroscopy group, Institut Laue-Langevin (ILL), France

McXtrace



DTU ESRF

McStas



NEXMAP DTU PAUL SCHERRER INSTITUT PSI NEUTRONS FOR SCIENCE ess EUROPEAN SPALLATION SOURCE

McStas-McXtrace 2018 GPU Hackathon, Dresden

Agenda

- What are scattering techniques?
- McStas and McXtrace - what they do and how
- Under the hood!
- Performance / profiling
- Our plan

Scattering techniques in 1 minute! :-)

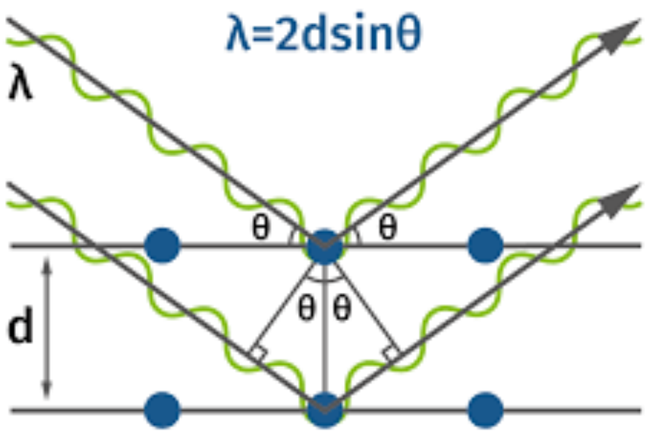
Probing matter
(to investigate properties of structure, dynamics, defects, ... – Be it on surface or in bulk)

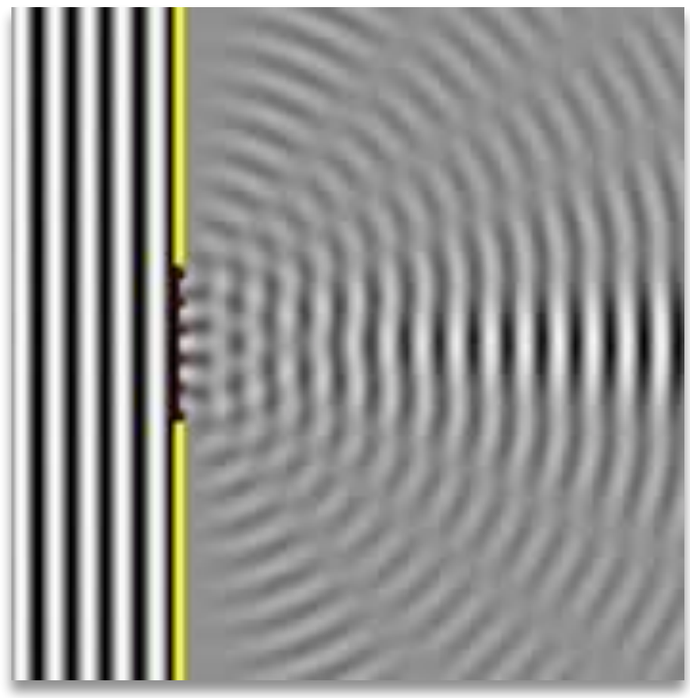
Beam of particles
(neutrons, X–rays, visible light, electrons, ...)

$\vec{p}_{in}, E_{in}$

$\vec{p}_{out}, E_{out}$

- Wavelength and / or energy comparable to the effects one wants to study





Neutron simulation McStas - for reactors and spallation sources

- Flexible, general simulation utility for neutron scattering experiments.
- Original design for **M**onte **C**arlo **S**imulation of **t**riple **a**xis **s**pectrometers
- Developed at DTU Physics, Uni CPH, ILL, PSI
- V. 1.0 by K Nielsen & K Lefmann (**1998**) RISØ
- Main development site is DTU Physics

GNU GPL
Open Source

Project website at
http://www.mcstas.org

X-ray simulation McXtrace - for synchrotrons and FELs

- Flexible, general simulation utility for X-ray scattering experiments.
- X-ray "port" of the McStas framework
- Developed at DTU Physics, ESRF, Uni CPH, (Saxslab)
- V. 1.0 in (**2012**) RISØ DTU
- Main development site is DTU Physics

GNU GPL
Open Source

Project website at
http://www.mcxtrace.org

Neutron simulation McStas - for reactors and spallation sources

- Flexible, general simulation utility for neutron scattering experiments.
- Original design for **M**onte **C**arlo **S**imulation of **t**riple **a**xis **s**pectrometers
- Developed at DTU Physics, Uni CPH, ILL, PSI
- V. 1.0 by K Nielsen & K Lefmann (**1998**) RISØ
- M
- E
- x
- Developed at DTU Physics, ESRF, Uni CPH, (Saxslab)
- V. 1.0 in (**2012**) RISØ DTU
- Main development site is NEXMAP

GNU GPL
Open Source

ebsite at
w.mcstas.org

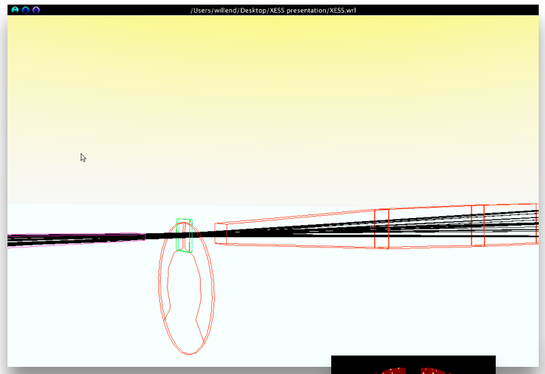
Shared repo https://github.com/McStasMcXtrace/McCode
Shared technology
Shared solutions
Different physics

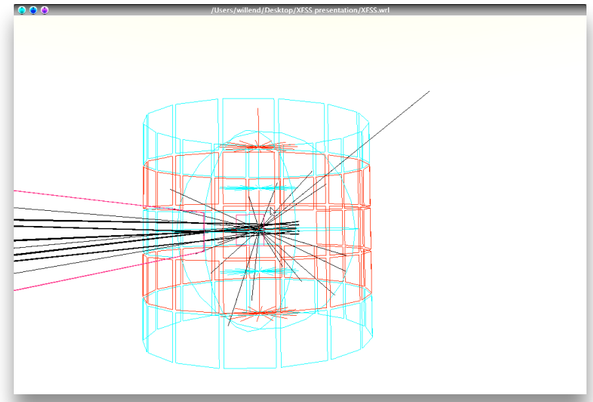
GNU GPL
Open Source

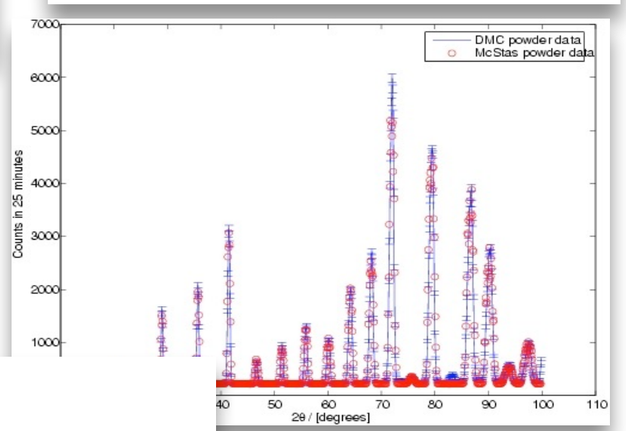
Project website at
http://www.mcxtrace.org

What are the codes used for?

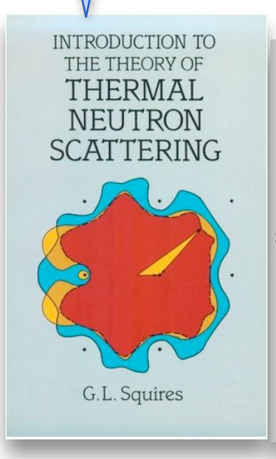
- Instrumentation
- Virtual experiments
- Data analysis
- Teaching

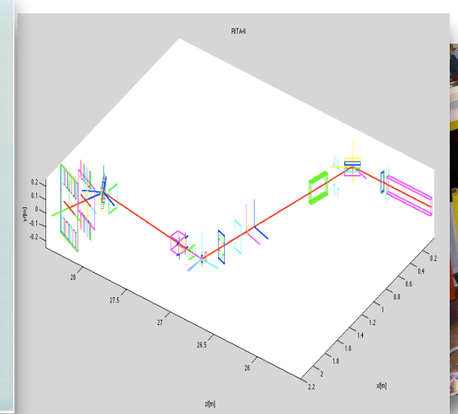


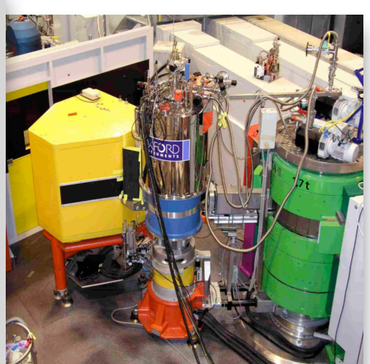


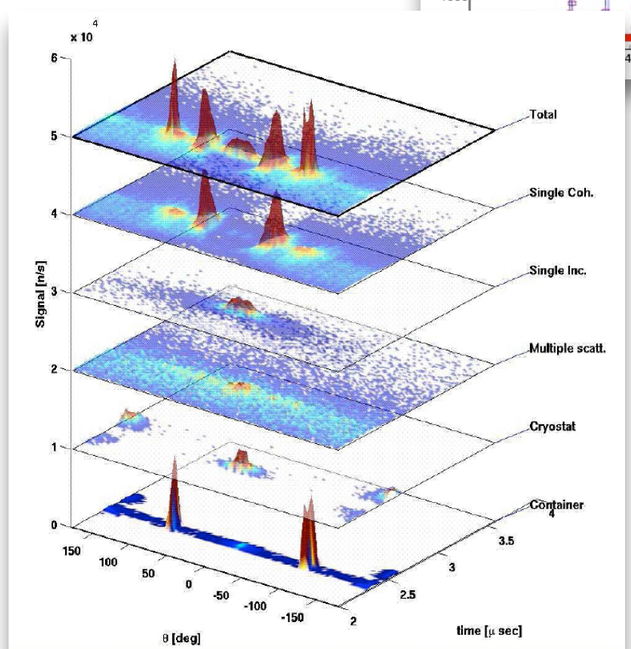


(DTU, KU schools & workshops)









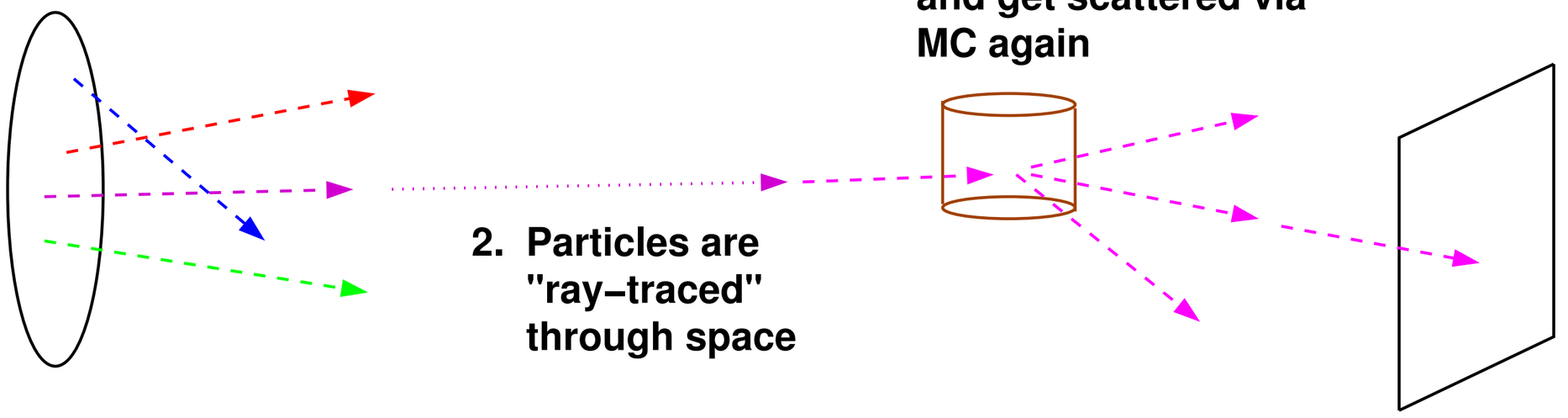
McStas

McXtrace

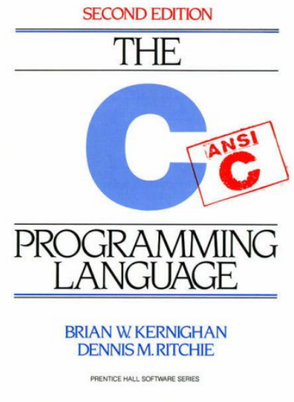
Main ingredients:

1.Monte Carlo
2.ray-tracing

1. Particles emitted with random starting conditions via MC



- Ray-tracing here typically happens in the “expensive” direction source -> detector
- Our neutrons are massive particles at limited speed - parabolas in gravitational field!



McStas overview

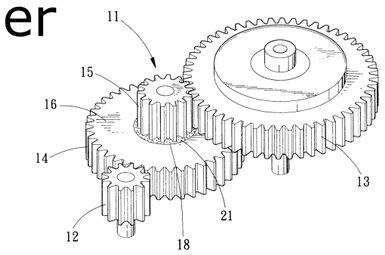
- Portable code (Unix/Linux/Mac/Windoze)

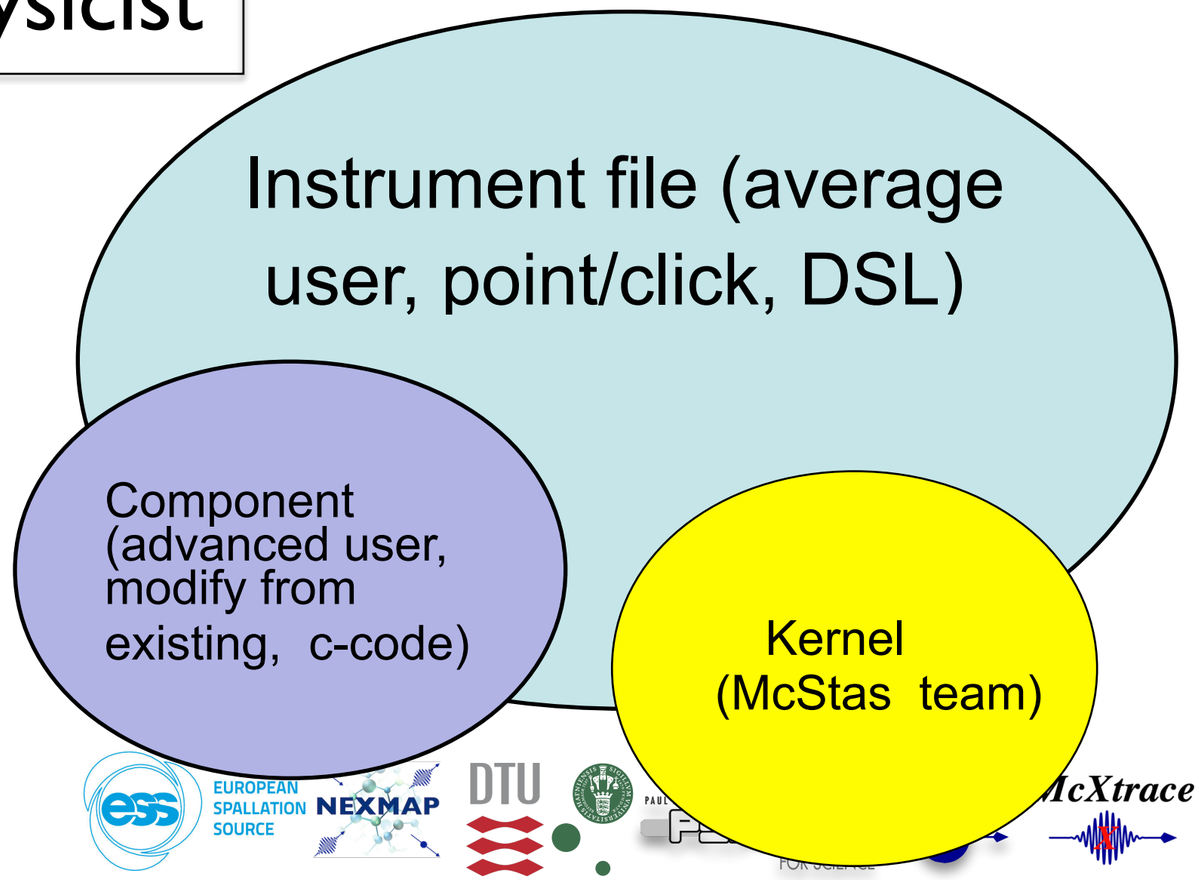
- Ran on everything from iPhone to 1000+ node cluster!

- 'Component' files (~150) inserted from library
  - Sources
  - Optics
  - Samples
  - Monitors
  - If needed, write your own comps

Often written by contributor / physicist

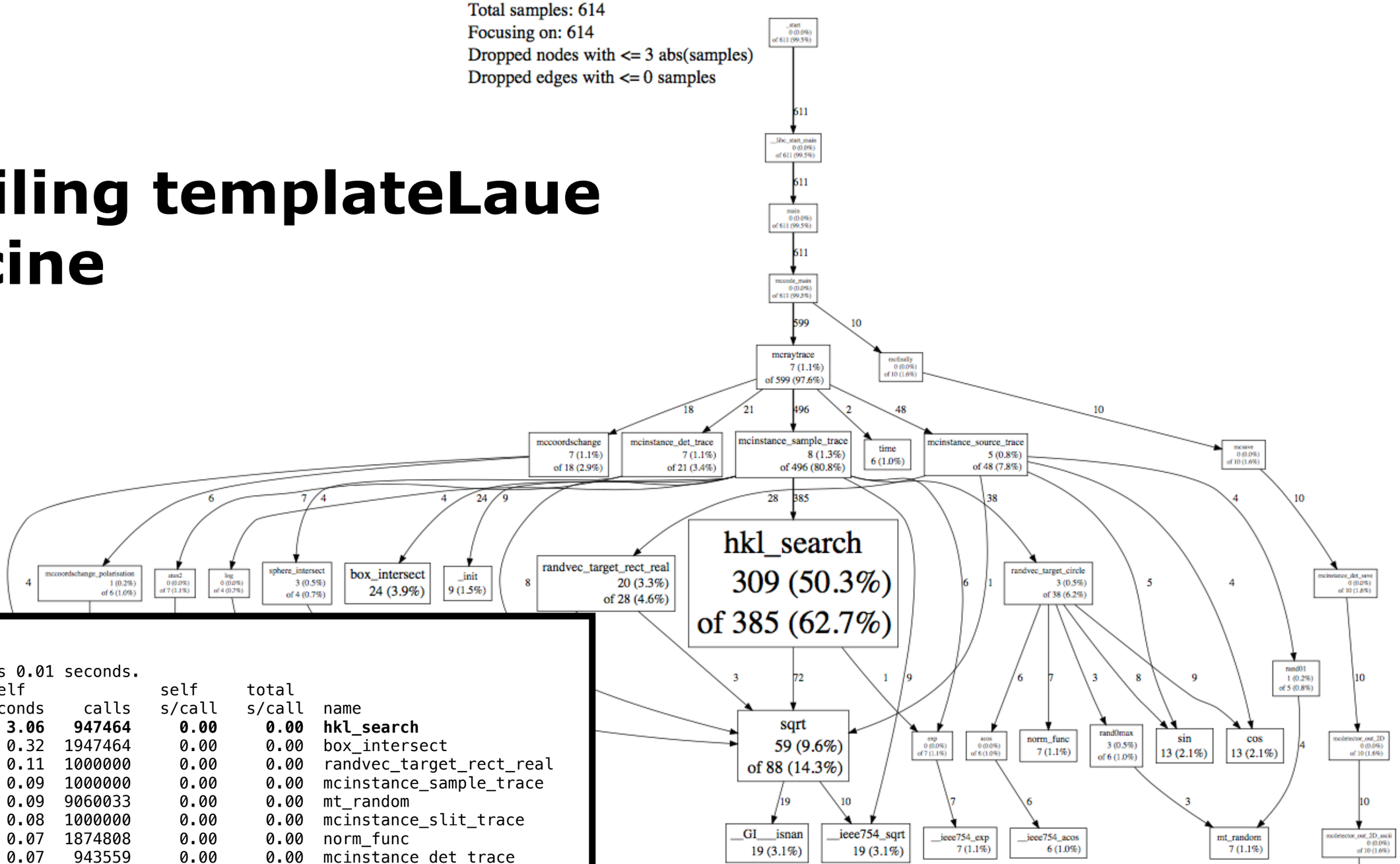
- DSL + ISO-C code gen.
  - LeX+Yacc based "parser-generator"
  - Code generation ⟹ ISO C (or pretty close)
  - Transparent use of compiler





templateLaue.out
Total samples: 614
Focusing on: 614
Dropped nodes with <= 3 abs(samples)
Dropped edges with <= 0 samples

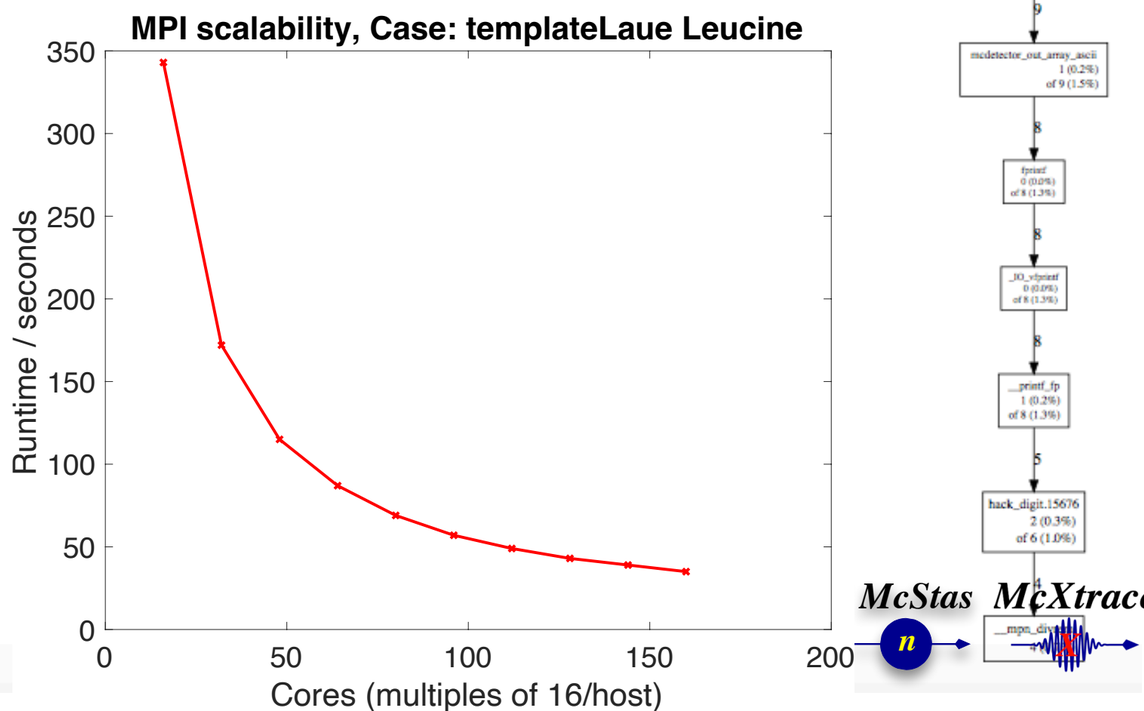
Profiling templateLaue Leucine




```
Flat profile:

Each sample counts as 0.01 seconds.
  %   cumulative   self              self     total
 time   seconds   seconds    calls   s/call   s/call  name
 69.39      3.06     3.06   947464     0.00     0.00  hkl_search
  7.26      3.38     0.32  1947464     0.00     0.00  box_intersect
  2.49      3.49     0.11  1000000     0.00     0.00  randvec_target_rect_real
  2.04      3.58     0.09  1000000     0.00     0.00  mcinstance_sample_trace
  2.04      3.67     0.09  9060033     0.00     0.00  mt_random
  1.81      3.75     0.08  1000000     0.00     0.00  mcinstance_slit_trace
  1.59      3.82     0.07  1874808     0.00     0.00  norm_func
  1.59      3.89     0.07   943559     0.00     0.00  mcinstance_det_trace
  1.36      3.95     0.06   937404     0.00     0.00  randvec_target_circle
  1.36      4.01     0.06 12887127     0.00     0.00  rot_apply
  1.13      4.06     0.05  3000000     0.00     0.00  randpm1
  0.91      4.10     0.04 12887134     0.00     0.00  rot_test_identity
  0.91      4.14     0.04  2963030     0.00     0.00  rand01
  0.91      4.18     0.04  1000000     0.00     0.00  mcraytrace
  0.68      4.21     0.03  3097003     0.00     0.00  rand0max
  0.68      4.24     0.03  1000007     0.00     0.00  rot_transpose
  0.68      4.27     0.03  1000000     0.00     0.00  mcinstance_source_trace
  0.68      4.30     0.03   943559     0.00     0.00  sphere_intersect
  0.45      4.32     0.02  4943562     0.00     0.00  coords_add
  0.45      4.34     0.02  4943559     0.00     0.00  mccoordschange
  0.45      4.36     0.02   669159     0.00     0.00  normal_vec
  0.34      4.38     0.02  5481374     0.00     0.00  vec_prod_func
  0.23      4.39     0.01  4943559     0.00     0.00  mccoordschange_polarisation
  0.23      4.40     0.01  1000007     0.00     0.00  mcget_run_num
  0.23      4.41     0.01        1     0.01     0.01  read_hkl_data
  0.11      4.41     0.01                             rot_print
  0.00      4.41     0.00  3000005     0.00     0.00  coords_set

…
```



Goals and challenges for hackathon

- **Historically** single-CPU simulations performed in **seconds -> hours**

- **Challenges**
  - Inclusion of “complicated sample models” pose new challenges wrt. needed statistics **-> days**
  - New particle sources like ESS (neutrons) and XFEL (X-rays) pose new challenges wrt. needed statistics **-> days**
  - McCode Simulations are “purpose-made” so **hard to say what** a “**generic**” simulation **is** here

- **Goals** for this week:
  - **Implement and investigate** prototype-options for doing “heavy code parts” on GPU
  - Look at **existing** (non-functional) **OpenCL** prototypes in the code tree

- **Prioritise**
  - **Non-invasive** code edits / additions
  - Sensible **co-existence** with current **MPI** solution