Container technology for phenomenology tools: the `udocker` middleware suite

Emanuele A. Bagnaschi (DESY Hamburg)

INDIGO - DataCloud

28 November 2017
11th Annual Helmholtz Alliance Workshop on “Physics at the Terascale”
DESY, Hamburg

In collaboration with J. Gomes (LIP), I. Campos (IFCA), M. David (LIP), L. Alves (LIP), J. Martins (LIP), J. Pina (LIP), A. López-Garcia (IFCA), P. Orviz (IFCA)

Based on Gomes J. et al [1711.01758]

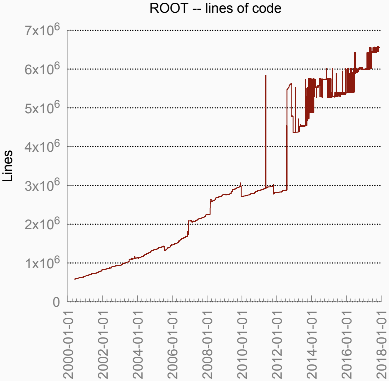
✉ emanuele.bagnaschi@desy.de

Outline

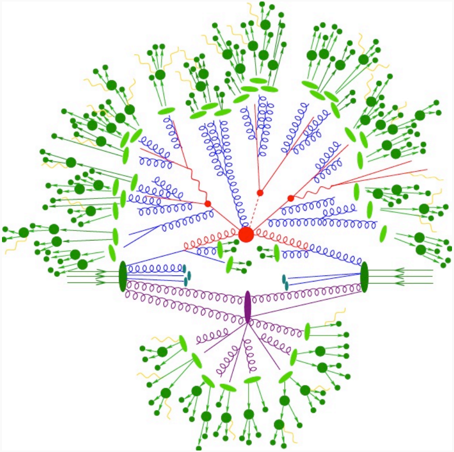
Outline

Motivations

- Complexity of phenomenology codes has risen consistently in the past years.
- Three drivers:
  1. (Precise) physics at the LHC requires sophisticated simulations.
  2. Large data set and complex analyses.
  3. Study of complex models of fundamental physics beyond the Standard Model.



Outline



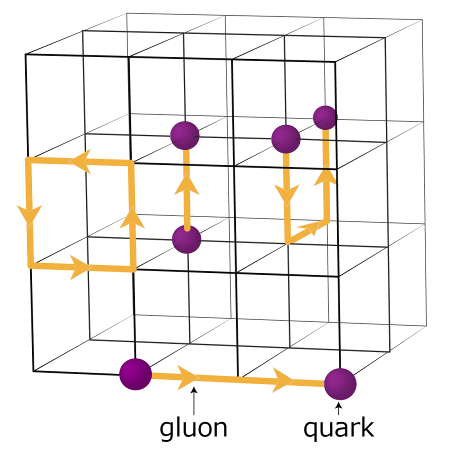
[SHERPA]

Examples

- Analysis (e.g. ROOT).
- Monte Carlo frameworks such as Madgraph_aMC@NLO, SHERPA, POWHEG-BOX, Whizard ...
- Lattice QCD computations.
- Global likelihood studies of BSM models (e.g. MasterCode, GAMBIT) ...
- Beyond HEP: molecular dynamics.

...

Outline



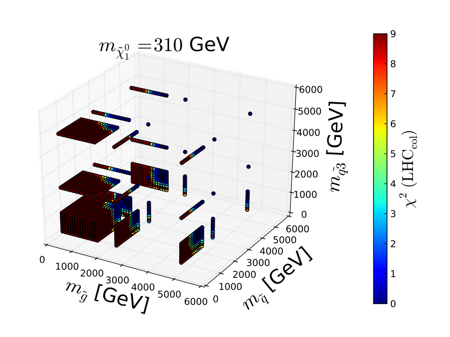
[Home page of G. Kossu]

Examples

- Analysis (e.g. `ROOT`).
- Monte Carlo frameworks such as `Madgraph_aMC@NLO`, `SHERPA`, `POWHEG-BOX`, `Whizard` . . .
- Lattice QCD computations.
- Global likelihood studies of BSM models (e.g. `MasterCode`, `GAMBIT`) . . .
- Beyond HEP: molecular dynamics.

. . .

Outline



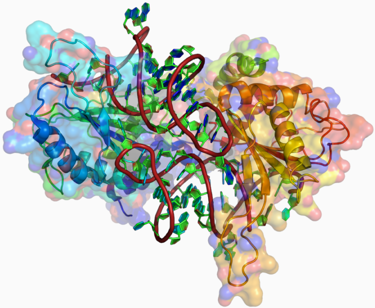
[MasterCode, 1504.03260]

Examples

- Analysis (e.g. `ROOT`).
- Monte Carlo frameworks such as `Madgraph_aMC@NLO`, `SHERPA`, `POWHEG-BOX`, `Whizard` . . .
- Lattice QCD computations.
- Global likelihood studies of BSM models (e.g. `MasterCode`, `GAMBIT`) . . .
- Beyond HEP: molecular dynamics.

. . .

Outline



[Gromacs, courtesy of R. Capelli]

Examples

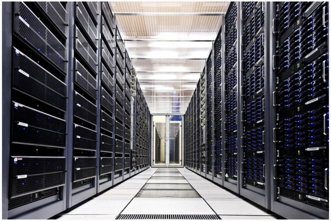
- Analysis (e.g. ROOT).
- Monte Carlo frameworks such as Madgraph_aMC@NLO, SHERPA, POWHEG-BOX, Whizard . . .
- Lattice QCD computations.
- Global likelihood studies of BSM models (e.g. MasterCode, GAMBIT) . . .
- Beyond HEP: molecular dynamics.

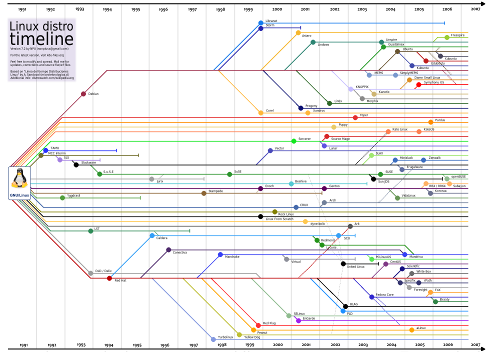
. . .

Deployment issues

Deployment issues

- Complexity means that these codes have a lot dependencies.
- Heterogeneous batch clusters.
- Large collaborations run at different sites.

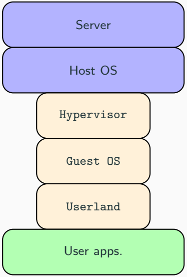




Possible solutions

Virtual machines

- *Emulation* of a full computer system.
- Many different hypervisors exist today: `KVM`, `VirtualBox`, `XEN` etc.



Operating-system level virtualization

- Old idea, e.g. `chroot`, `FreeBSD` ($>$ 4.0) `jails`.
- Linux containers: cgroups ($>$ 2.6.24), namespace support ($>$ 2.4.19).

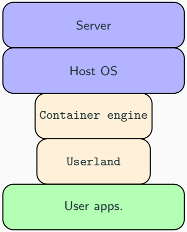
Advantages over VMs

- Lightweight approach to virtualization (less resource hungry, more running in parallel on a single host).
- `Linux container`: easy to deploy on recent `Linux` systems (kernel version $>$ 3.8).

Possible solutions

Virtual machines

- *Emulation* of a full computer system.
- Many different hypervisors exist today: `KVM`, `VirtualBox`, `XEN` etc.



Operating-system level virtualization

- Old idea, e.g. `chroot`, `FreeBSD` (> 4.0) `jails`.
- Linux containers: cgroups ($> 2.6.24$), namespace support ($> 2.4.19$).

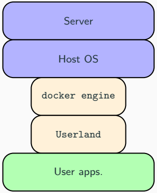
Advantages over VMs

- Lightweight approach to virtualization (less resource hungry, more running in parallel on a single host).
- `Linux container`: easy to deploy on recent `Linux` systems (kernel version > 3.8).

Docker containers and the udocker suite

Docker and udocker

- As suggested by the name, `udocker` uses `docker` containers.
- `Docker`: software framework to automatize the deployment of application inside Linux containers.
- Other options, such as `LXC`, are available to use the Linux container infrastructure.



- Middleware suite developed in the context of the `INDIGO data-cloud` project to run `docker` containers in userspace, without requiring root privileges (both for installation and execution).

Udocker inner workings

Features

- udocker pre-compiled code and the containers are download to ${UDOCKER_DIR}, by default ${HOME}/.udocker.
- Docker layered FS is UnionFS based. Images are pulled by downloading the corresponding layers and metadata (docker Hub REST API).
- udocker implements parsing of docker container and of a subset of metadata.
- Different execution engines: PTRACE, LD_PRELOAD, runC, Singularity.
- Tested with GPGPU and MPI aware applications.

PTRACE engine

- Implements through PRoot.
- PRoot uses PTRACE to change the pathnames dynamically and to execute the binary transparently inside the container (P2 mode).
- Patches have been written to make SECCOMP works with PTRACE (P1 mode).

Mode	Description	Changes container
P1	PRoot+SECCOMP	No
P2	PRoot	No

Udocker inner workings

Features

- udocker pre-compiled code and the containers are download to ${UDOCKER_DIR}, by default ${HOME}/.udocker.
- Docker layered FS is UnionFS based. Images are pulled by downloading the corresponding layers and metadata (docker Hub REST API).
- udocker implements parsing of docker container and of a subset of metadata.
- Different execution engines: PTRACE, LD_PRELOAD, runC, Singularity.
- Tested with GPGPU and MPI aware applications.

LD_PRELOAD engine

- Based on the Fakechroot library.
- Implemented several workarounds to address Fakechroot shortcomings and to avoid letting the containerized application load system libraries.
- Modified version of PatchELF to perform the modifications of the binaries.

Mode	Description	Changes container
F1	exec w/ direct loader	symlinks
F2	F1 + mod. loader	F1+ld.so
F3	ELF header mod.	F2+ELF headers
F4	F3 + new execs and libs	as F3

Udocker inner workings

Features

- udocker pre-compiled code and the containers are download to ${UDOCKER_DIR}, by default ${HOME}/.udocker.
- Docker layered FS is UnionFS based. Images are pulled by downloading the corresponding layers and metadata (docker Hub REST API).
- udocker implements parsing of docker container and of a subset of metadata.
- Different execution engines: PTRACE, LD_PRELOAD, runC, Singularity.
- Tested with GPGPU and MPI aware applications.

RunC engine

- Support for unprivileged User Namespace and rootless container using RunC.
- udocker performs the translation between docker metadata and cli args and the OCI specs to run the container in unprivileged mode.

Singularity

- Support running singularity containers.

Mode	Description	Changes container
R1	rootless usermod namesp.	resolv,passwd
S1	singularity	passwd

Udocker inner workings

Features

- udocker pre-compiled code and the containers are download to ${UDOCKER_DIR}, by default ${HOME}/.udocker.
- Docker layered FS is UnionFS based. Images are pulled by downloading the corresponding layers and metadata (docker Hub REST API).
- udocker implements parsing of docker container and of a subset of metadata.
- Different execution engines: PTRACE, LD_PRELOAD, runC, Singularity.
- Tested with GPGPU and MPI aware applications.

RunC engine

- Support for unprivileged User Namespace and rootless container using RunC.
- udocker performs the translation between docker metadata and cli args and the OCI specs to run the container in unprivileged mode.

Singularity

- Support running singularity containers.

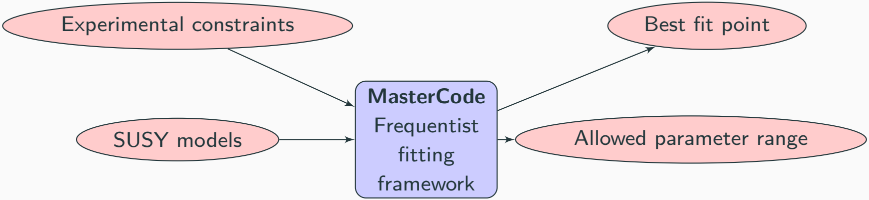
Mode	Description	Changes container
R1	rootless usermod namesp.	resolv,passwd
S1	singularity	passwd

Complex libraries dependencies: `MasterCode`

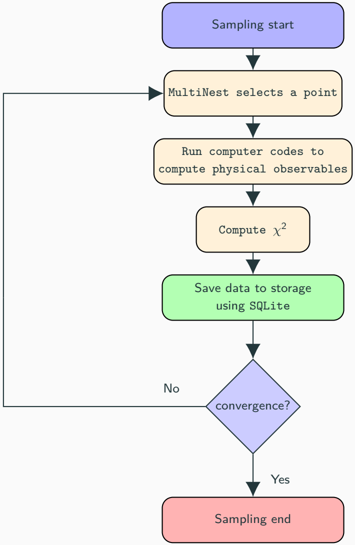
MasterCode

Global likelihood studies of BSM models

- Mixed collaboration of experimentalists and theorists to understand the status of BSM models in light of current constraints.
- Use of the available collider data, electro-weak precision observables and DM constraint to fit the best value and the likelihood profile of the model parameters.
- See my talk in the BSM-session this morning for our latest pMSSM study.



Structure of the framework


Sampling start
MultiNest selects a point
Run computer codes to compute physical observables
Compute χ^2
Save data to storage using SQLite
convergence?
No
Yes
Sampling end

Codes

Spectrum generation
SoftSUSY
Higgs sector and $(g-2)_\mu$
FeynHiggs, HiggsSignals, HiggsBounds
B-Physics
SuFla, SuperISO
EW precision observables
FeynWZ
Dark matter
MicrOMEGAs, SSARD

- During our last study, we sampled a total of 2×10^9 points.
- We thank DESY for the resources provided by the NAF2/BIRD cluster.

Running MasterCode with (u)docker

- Due to its complex structure, MasterCode is a perfect test case to show the advantages of using containerization to ease the deployment.
- We have built a docker container to support MasterCode.

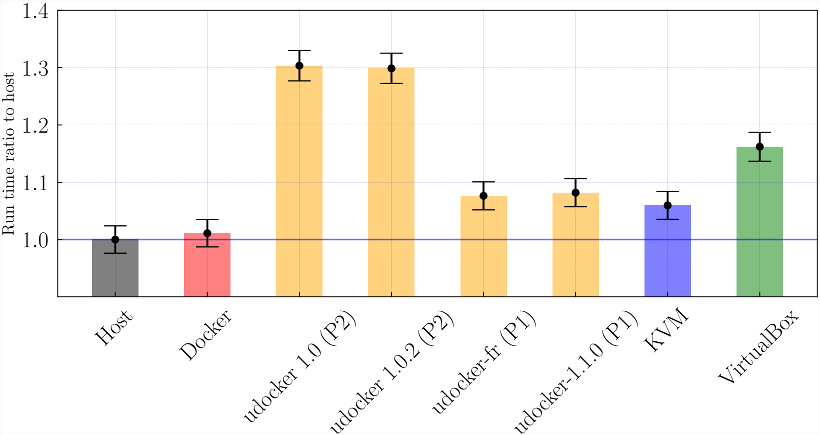
Docker

```
emanuele [0]> docker pull \
 indigodatacloud/docker-mastercode

emanuele [0]> docker run -t -i \
 -v ${HOME} \
 -w ${HOME}/my_mc_dir \
indigodatacloud/docker-mastercode \
 /bin/bash
```

udocker

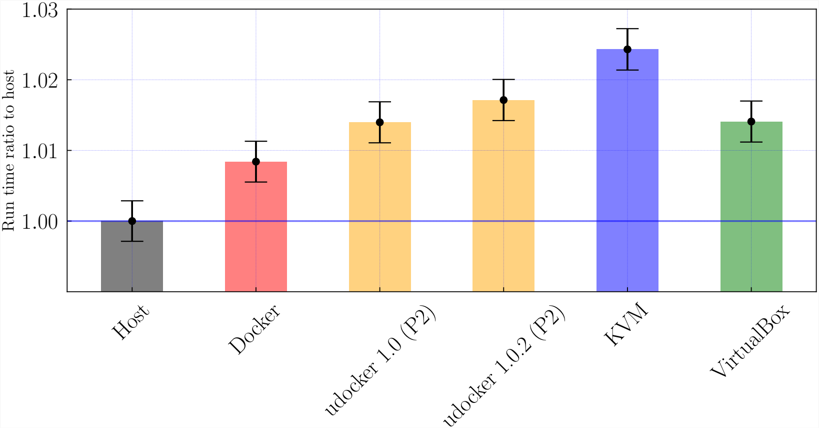
```
emanuele [0]> git clone \
 https://github.com/indigo-dc/udocker.git

emanuele [0]> cd udocker.git

emanuele [0]> ./udocker.py \
 pull indigodatacloud/docker-mastercode

emanuele [0]> ./udocker.py \
 create indigodatacloud/docker-mastercode

emanuele [0]> ./udocker.py \
-v /home/emanuele/mastercode \
-w /home/emanuele/mastercode \
'/bin/bash -c "run_mastercode_container.sh"'
```

Benchmarking: compilation



- P2 (PTRACE) mode slower by about 30% w.r.t. to host. Expected, since compilation implies a lot of syscalls.
- The PTRACE mode with SECCOMP filtering (P1 – the default) improves a lot the situation, making udocker as fast as the VMs.

Benchmarking: sampling



- The sampling phase is characterized by less I/O activity.
- `udocker` close to `docker` performances, only $\mathcal{O}(1.5\%)$ performance hit even in P2 mode.

MPI simulations: `OpenQCD`

MPI simulations: Open QCD

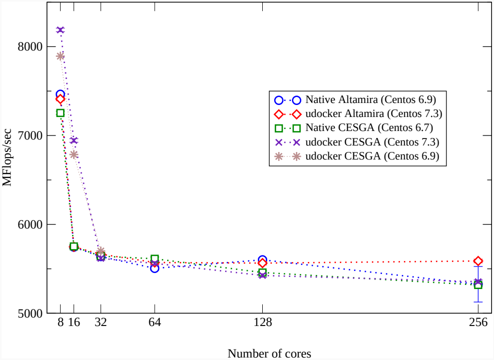
- Lattice QCD is a strongly computing-characterized discipline (hundreds of millions of CPU hours/year).
- Current simulations run spread over thousands of processor cores in parallel.
- OpenQCD is a very advanced GPL-licensed code to run lattice simulations.

Running MPI codes w/ udocker

- Download and install the container as with MasterCode.
- caveat: exactly the same version of MPI on the host and in the container.
- With udocker, the mpiexec of the host system is used to submit the MPI processes.

```
emanuele [0]> ${HOST_OPENMPI_PATH_BIN}/mpiexec \
 -np 128 udocker run \
 --hostenv --hostauth --user=${USERID} \
 --workdir=${OPENQCD_CONTAINER_DIR} \
 openqcd \
 ${OPENQCD_CONTAINER_DIR}/ym1 -i ym1.in
```

Scaling test



- Scaling performance as a function of the cores for the computation of application of the Dirac operator to a spinor field (Practically, it is a sparse-matrix-matrix × vector multiplication).
- `udocker` at *least* as fast as the host.
- At CESGA `udocker` *faster* than host because of newest libraries for 8 and 16 cores.

GPU-accelerated simulations: DisVis and Gromacs

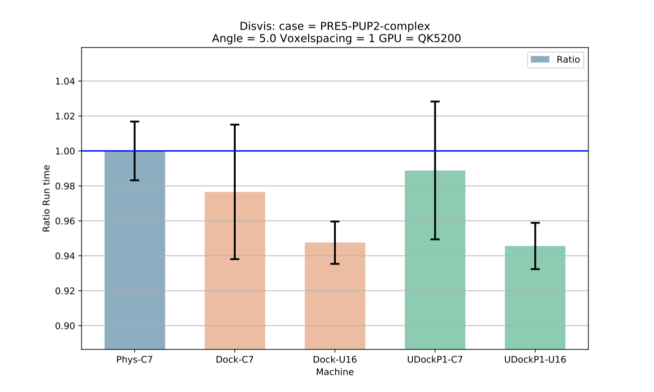
Biomolecular complexes: DisVis, Powerfit and Gromacs

- `DisVis` and `Powerfit` are MIT-licensed codes available on `GitHub` to model biomolecular complexes.
- They leverage GPUs through `OpenCL`, via `PyOpenCL`
- `Gromacs`is a molecular dynamics package for both biochemical and non-biochemical systems.

Running w/ `udocker`

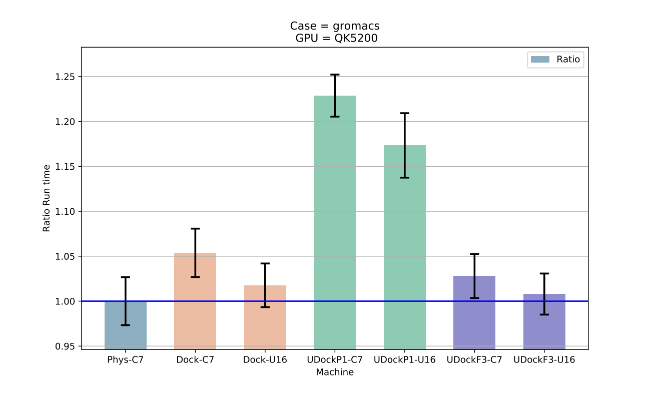
- Download and install the container as with `MasterCode`.
- caveat: exactly the same version of the NVIDIA drivers and libraries needs to be installed on the host and the container.

DisVis



- `udocker` and `docker` have same performance as the host when using `CentOS7`.
- Improved performance due to newer userland libraries when using `Ubuntu16`.

Gromacs



- `udocker` and `docker` are worse than the host when using `CentOS7` of $\mathcal{O}$(3-5%).
- Same performances when using `Ubuntu16`.
- Use of P1 mode results in $\mathcal{O}$(22%) performance hit (due to communication between the GPGPU and the CPU threads – Gromacs spawns 8 OpenMP threads / GPU).

Conclusions

Conclusions

- We have presented the udocker middleware suite, which allow to run seamless docker container in user-space without root access.
- The goal of udocker is to ease the deployment of complex frameworks on (heterogeneous) clusters.
- Requires no intervention from a system administrator (no root access required!).
- For CPU intensive application, there is basically no performance hit.
- I/O bounded applications require more care (use of Fn modes vs Pn modes to reduce the performance hit).
- Available on GitHub at https://github.com/indigo-dc/udocker/.

INDIGO - DataCloud