Towards generic volunteer computing platform

Konstantin Nikitin, Andrey Ustyuzhanin, Alexander Baranov

CERN Organisation
Europeenne pour la Recherche
Nucleaire
CH-1211 Geneve 23
Switzerland

Preface

Skygrid (Docker[a] + Apache YARN[b]):

- High level of hardware and software abstraction,
- Run tasks with conflicting requirements in parallel on the same host,
- Quite no requirements to host machine.

JavaScript — virtualisation without installation and side effects. Language interpreters evolved very much lately.
Resouce and scope management are performed by web-browser, what is good at it.

Project Description

"DiBroCop is a project for computation unit utilisation by tasks which are executed within a web-browser."

Code reuse

Backward compatibility is one of the most important features, *eg.* C $\rightarrow$ C++, LAMP $\rightarrow$ AWS, etc.

Fortunately, code reuse it not a problem for DiBroCop:

- LLVM[c] = Set of compilers + well defined intermediate representation (IR)
- Emscripten[d]: LLVM IR is interpreted by JavaScript.

So, quite every program or library, which is not heavy dependent from OS internals could be run on JavaScript.

Next steps

- Integration with Skygrid (as a worker),
- Packing more applications and libraries,
- Docker-in-browser via linux-in-js[e],
- Try PNaCl[f], Silverlight[g] as another (but platform-specific) task runners. Generalization of all approaches.
- Run several tasks in parallel, using WebWorkers[h].

As a result we are going to create in-browser competitor to seti@home, but

- Available for all platforms, for all kinds of architectures,
- Easy to participate, just by opening a web page.

[a] https://www.docker.io
[b] http://hadoop.apache.org/docs/r2.6.0/hadoop-yarn/hadoop-yarn-site/YARN.html
[c] http://llvm.org/
[d] http://kripken.github.io/emscripten-site/
[e] http://bellard.org/jslinux/
[f] http://nativeclient.googlecode.com/svn/data/site/pnacl.pdf
[g] http://www.microsoft.com/SilverLight/
[h] https://developer.mozilla.org/en-US/docs/Web/API/Web_Workers_API/Using_web_workers

Pythia

Library for Monte-Carlo event generation. Written on C++.
A few patches was done to compile it with Emscripten (working with files and native exceptions).

https://xni.github.io/pythia/main01.html

Speed tests