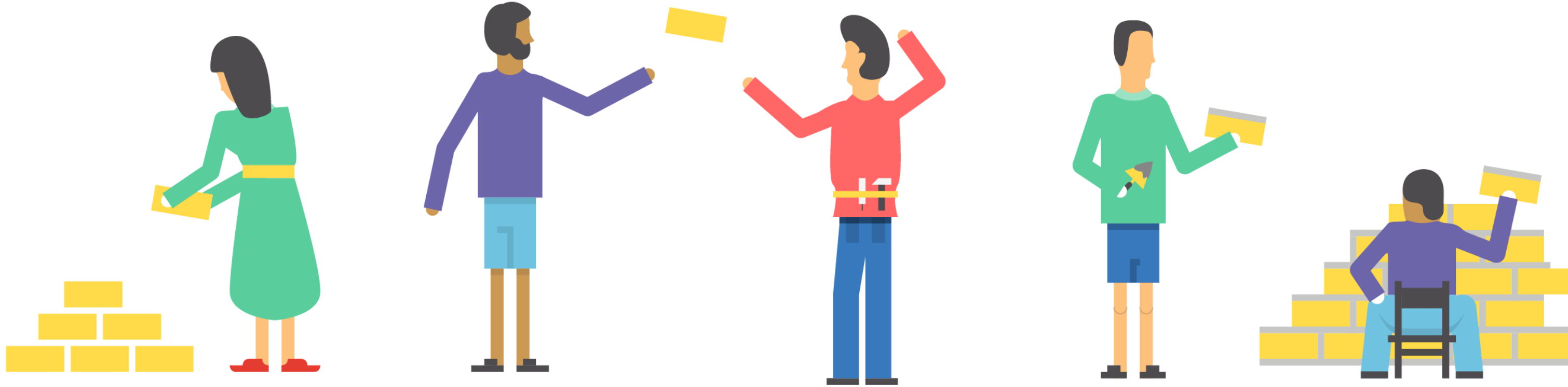


Towards generic volunteer computing platform

Konstantin Nikitin, Andrey Ustyuzhanin, Alexander Baranov



Preface

Skygrid^a (Docker^b + Apache YARN^c):

- 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 → C++, LAMP → AWS, etc.

Fortunately, code reuse it not a problem for DiBroCop:

- LLVM^d = Set of compilers + well defined intermediate representation (IR)
- Emscripten^e: 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^f,
- Try PNaCl^g, Silverlight^h as another (but platform-specific) task runners. Generalization of all approaches.
- Run several tasks in parallel, using WebWorkersⁱ.

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://github.com/anaderi/skygrid>

^b<https://www.docker.io>

^c<http://hadoop.apache.org/docs/r2.6.0/hadoop-yarn/hadoop-yarn-site/YARN.html>

^d<http://llvm.org/>

^e<http://kripken.github.io/emscripten-site/>

^f<http://bellard.org/jslinux/>

^g<http://nativeclient.googlecode.com/svn/data/site/pnacl.pdf>

^h<http://www.microsoft.com/SilverLight/>

ⁱhttps://developer.mozilla.org/en-US/docs/Web/API/Web_Workers_API/Using_web_workers

Pythia

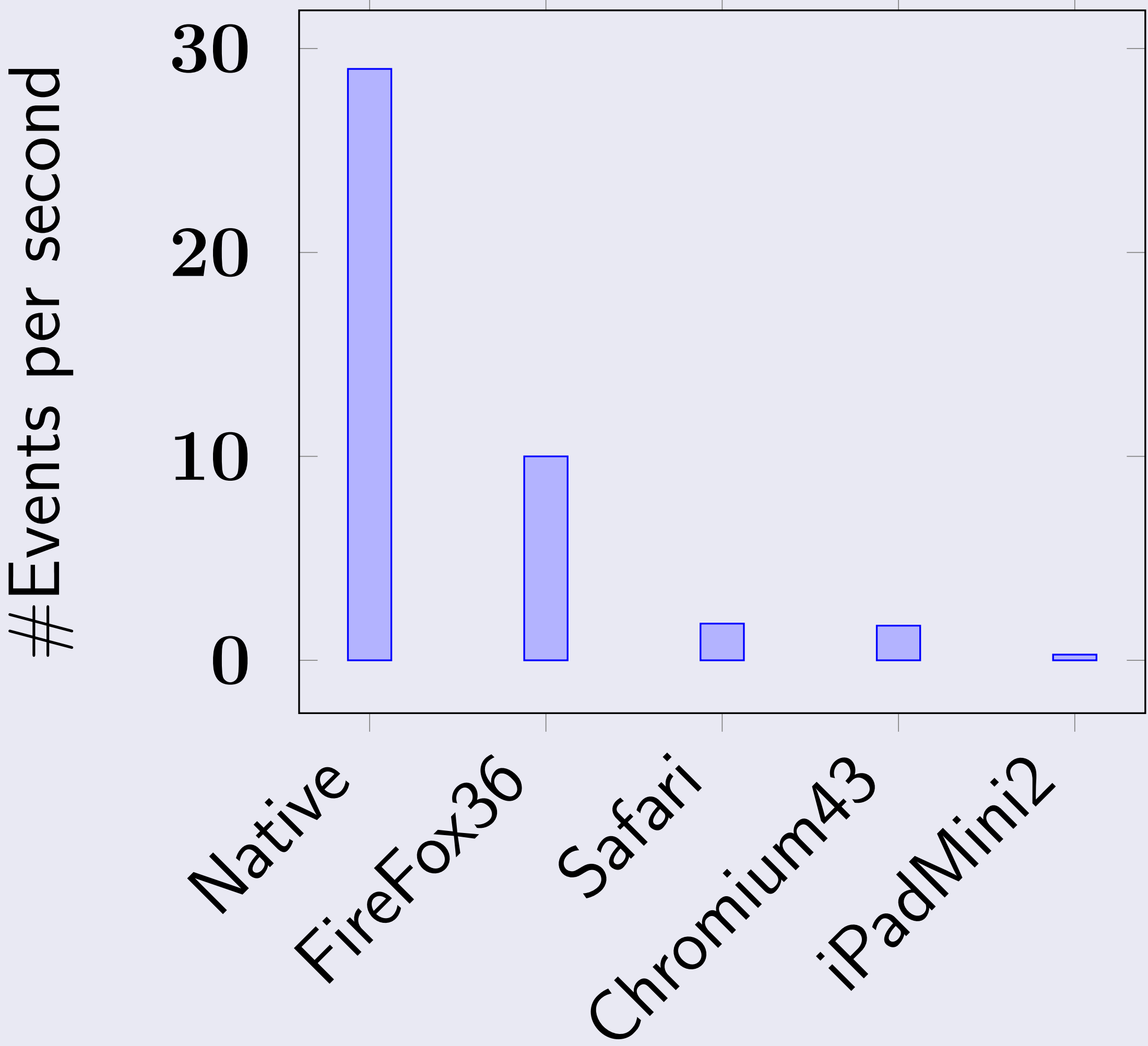
Library for Monte-Carlo event generation. Written on C++.

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



Platform	#Events per second
Native	~29
FireFox36	~10
Safari	~2
Chromium43	~2
iPadMini2	~0.5