

Painting the Picture of Software Impact with the Research Software Directory

Jurriaan H. Spaaks

October 29, 2018

Tom Klaver

WSSSPE6.1

Stefan Verhoeven

Amsterdam, The Netherlands

Jason Maassen

Tom Bakker

Atze van der Ploeg

Ben van Werkhoven

Willem van Hage

Rob V. van Nieuwpoort

What is a Research Software Directory?

- A content management system for software
- Promotes **findability**, **reproducibility**, and **citability** of the software in it
- Enables qualitative assessment of software **impact**

You can get your own!

- Permissive license (Apache-2.0)
- Check out our instance here:
`https://www.research-software.nl`

Kernel Tuner

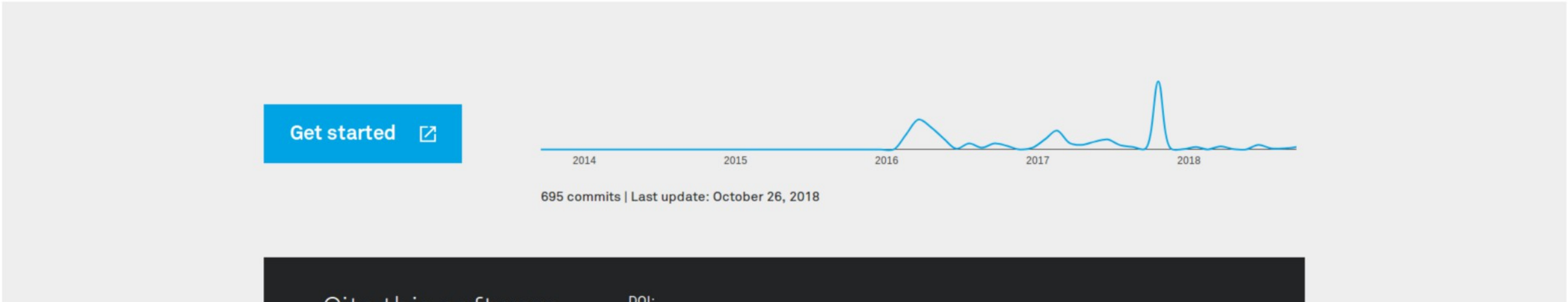
14

mentions

8

contributors

Kernel Tuner greatly simplifies the development of highly-optimized and auto-tuned CUDA, OpenCL, and C code, supporting many advanced use-cases and optimization strategies that speed up the auto-tuning process.



Cite this software

DOI:
10.5281/zenodo.1220114
Copy to clipboard

Choose a version:
0.1.9

Choose a citation style:
BibTeX
Download file

What Kernel Tuner can do for you

- Allows developers to easily unit test and auto-tune GPU code
- Generic auto-tuning of user-defined parameters for CUDA, OpenCL, and C kernels
- Supports more than 20 different search optimization methods to speedup tuning
- Successfully used in 5 different eScience projects, across various disciplines

Kernel Tuner simplifies the development of efficient GPU programs, or *kernels*. It does so by making kernels written in C/C++, OpenCL, or CUDA accessible from Python, while taking care of the required synchronization between data kept in host memory and data kept in device memory.

This has a number of advantages. First, it simplifies *auto-tuning* of the kernel parameters. In fact, Kernel Tuner comes standard with a variety of strategies for efficiently searching the parameter space, leading to greatly improved performance of tuned kernels. Second, it allows for unit testing of GPU code from within Python.

Kernel Tuner does not add any additional dependencies to the kernel code, and does not require extensive code changes. Furthermore, it is noteworthy that kernels tuned by Kernel Tuner do not require any changes after tuning to make them production ready--tuned kernels can be used as-is from any host programming language.

× Read less

Tags

GPU

High performance computing

Multi-scale & multi model simulations

Real time data analysis

Optimized data handling

Big data

Programming Language

Python

CUDA

OpenCL

License

Apache-2.0

Purpose

- Search

-

-

-

- Adoption

Participating organizations

netherlands **Science** center

Mentions



Writing Testable GPU Code

By Ben van Werkhoven
April 12, 2018

Kernel Tuner

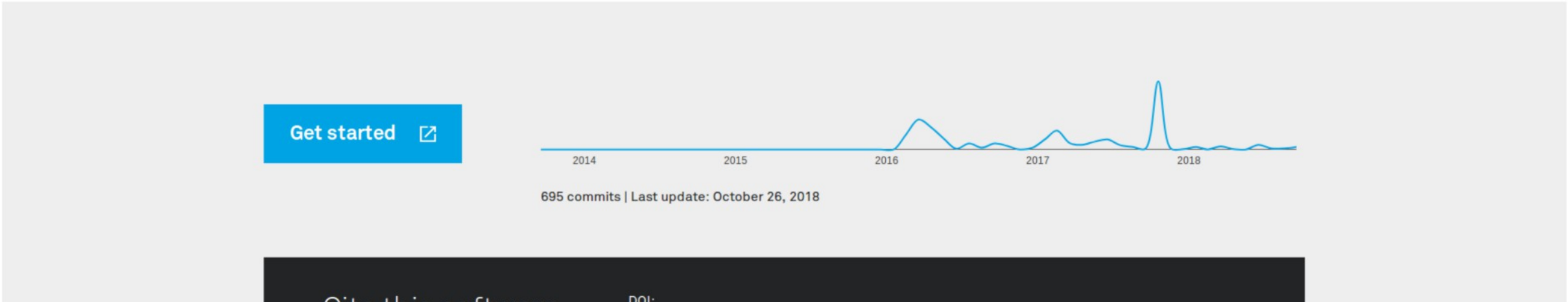
14

mentions

8

contributors

Kernel Tuner greatly simplifies the development of highly-optimized and auto-tuned CUDA, OpenCL, and C code, supporting many advanced use-cases and optimization strategies that speed up the auto-tuning process.



Cite this software

DOI:
10.5281/zenodo.1220114

Copy to clipboard

Choose a version:
0.1.9

Choose a citation style:
BibTeX

Download file

What Kernel Tuner can do for you

- Allows developers to easily unit test and auto-tune GPU code
- Generic auto-tuning of user-defined parameters for CUDA, OpenCL, and C kernels
- Supports more than 20 different search optimization methods to speedup tuning
- Successfully used in 5 different eScience projects, across various disciplines

Kernel Tuner simplifies the development of efficient GPU programs, or *kernels*. It does so by making kernels written in C/C++, OpenCL, or CUDA accessible from Python, while taking care of the required synchronization between data kept in host memory and data kept in device memory.

This has a number of advantages. First, it simplifies *auto-tuning* of the kernel parameters. In fact, Kernel Tuner comes standard with a variety of strategies for efficiently searching the parameter space, leading to greatly improved performance of tuned kernels. Second, it allows for unit testing of GPU code from within Python.

Kernel Tuner does not add any additional dependencies to the kernel code, and does not require extensive code changes. Furthermore, it is noteworthy that kernels tuned by Kernel Tuner do not require any changes after tuning to make them production ready--tuned kernels can be used as-is from any host programming language.

× Read less

Tags

GPU

High performance computing

Multi-scale & multi model simulations

Real time data analysis

Optimized data handling

Big data

Programming Language

Python

CUDA

OpenCL

License

Apache-2.0

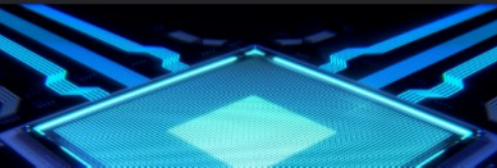
Purpose

- Search
- Find it
- Know that you found it
- Get started
- Adoption

Participating organizations

netherlands **Science** center

Mentions



Writing Testable GPU Code

By Ben van Werkhoven
April 12, 2018

Answer visitor's questions
using inclusive language

Kernel Tuner

14

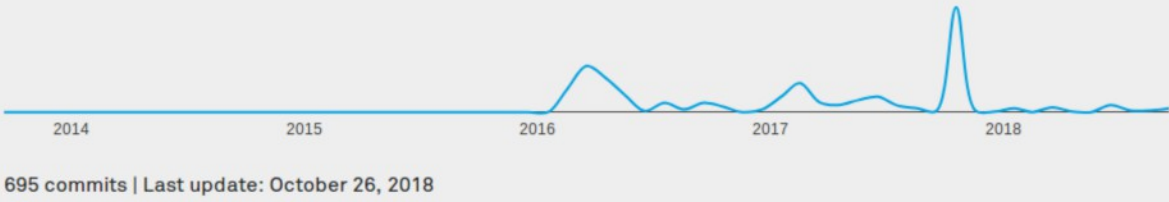
mentions

8

contributors

Kernel Tuner greatly simplifies the development of highly-optimized and auto-tuned CUDA, OpenCL, and C code, supporting many advanced use-cases and optimization strategies that speed up the auto-tuning process.

Get started



Cite this software

Choose a version:

0.1.9

Choose a citation style:

BibTeX

Download file

DOI:

10.5281/zenodo.1220114

Copy to clipboard

What Kernel Tuner can do for you

- Allows developers to easily unit test and auto-tune GPU code
- Generic auto-tuning of user-defined parameters for CUDA, OpenCL, and C kernels
- Supports more than 20 different search optimization methods to speedup tuning
- Successfully used in 5 different eScience projects, across various disciplines

Kernel Tuner simplifies the development of efficient GPU programs, or *kernels*. It does so by making kernels written in C/C++, OpenCL, or CUDA accessible from Python, while taking care of the required synchronization between data kept in host memory and data kept in device memory.

This has a number of advantages. First, it simplifies *auto-tuning* of the kernel parameters. In fact, Kernel Tuner comes standard with a variety of strategies for efficiently searching the parameter space, leading to greatly improved performance of tuned kernels. Second, it allows for unit testing of GPU code from within Python.

Kernel Tuner does not add any additional dependencies to the kernel code, and does not require extensive code changes. Furthermore, it is noteworthy that kernels tuned by Kernel Tuner do not require any changes after tuning to make them production ready--tuned kernels can be used as-is from any host programming language.

Read less

Tags

- GPU
- High performance computing
- Multi-scale & multi model simulations
- Real time data analysis
- Optimized data handling
- Big data

Programming Language

- Python
- CUDA
- OpenCL

License

- Apache-2.0

Participating organizations

netherlands **Science** center

Mentions



Writing Testable GPU Code

By Ben van Werkhoven
April 12, 2018

in C/C++, OpenCL, or CUDA accessible from Python, while taking care of the required synchronization between data kept in host memory and data kept in device memory.

This has a number of advantages. First, it simplifies *auto-tuning* of the kernel parameters. In fact, Kernel Tuner comes standard with a variety of strategies for efficiently searching the parameter space, leading to greatly improved performance of tuned kernels. Second, it allows for unit testing of GPU code from within Python.

Kernel Tuner does not add any additional dependencies to the kernel code, and does not require extensive code changes. Furthermore, it is noteworthy that kernels tuned by Kernel Tuner do not require any changes after tuning to make them production ready--tuned kernels can be used as-is from any host programming language.

X Read less

[{} Programming Language](#)

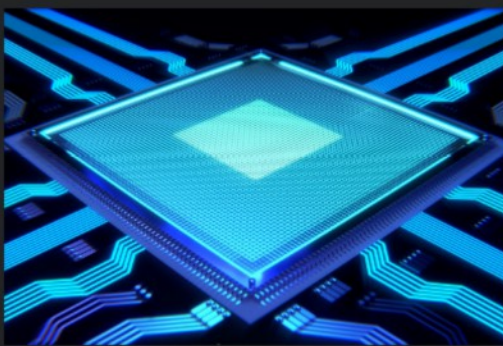
Python
CUDA
OpenCL

[License](#)

Apache-2.0

social & scientific context

Mentions



Writing Testable GPU Code

By Ben van Werkhoven
April 12, 2018

[Visit our blog](#)

- 4 Computer programs +
- 1 Conference paper +
- 2 Journal articles +
- 6 Presentations +

Projects with
Kernel Tuner



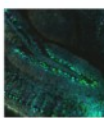
A Jungle Computing Approach to
Large-Scale Online Forensic
Analysis

Programming tools that simplify
application development and
deployment
Prof. Henri Bal
VU University Amsterdam



3D Geospatial Data Explor. for
Modern Risk Management Systems

The country below sea level



Parallelisation of multi point-cloud
registration

Studying subcellular structures and
functions
Dr. Bernd Rieger
Delft University of Technology



Real-time detection of neutrinos
from the distant Universe

Observing processes that are
inaccessible to optical telescopes
Dr. Dorothea Samtleben
Leiden University



DIRAC

Distributed Radio Astronomical
Computing
Dr. Sarod Yatawatta
ASTRON

Contributors

[Ben van Werkhoven](#)
Netherlands eScience Center

[Alessio Sclocco](#)
Netherlands eScience Center

[Patrick Bos](#)
Netherlands eScience Center

[Inti Pelupessy](#)
Netherlands eScience Center

[Felipe Zapata](#)
Netherlands eScience Center

[Johannes Hidding](#)
Netherlands eScience Center

+ Show all contributors

CONTACT PERSON



[Ben van Werkhoven](#)
Netherlands eScience Center
✉ Mail Ben

Help people get started



Kernel Tuner

14

mentions

8

contributors

Kernel Tuner greatly simplifies the development of highly-optimized and auto-tuned CUDA, OpenCL, and C code, supporting many advanced use-cases and optimization strategies that speed up the auto-tuning process.

Get started

2014 2015 2016 2017 2018

695 commits | Last update: October 26, 2018

Cite this software

DOI:
10.5281/zenodo.1220114 Copy to clipboard

Choose a version:
0.1.9

Choose a citation style:
BibTeX Download file

What Kernel Tuner can do for you

- Allows developers to easily unit test and auto-tune GPU code
- Generic auto-tuning of user-defined parameters for CUDA, OpenCL, and C kernels
- Supports more than 20 different search optimization methods to speedup tuning
- Successfully used in 5 different eScience projects, across various disciplines

Kernel Tuner simplifies the development of efficient GPU programs, or *kernels*. It does so by making kernels written in C/C++, OpenCL, or CUDA accessible from Python, while taking care of the required synchronization between data kept in host memory and data kept in device memory.

This has a number of advantages. First, it simplifies *auto-tuning* of the kernel parameters. In fact, Kernel Tuner comes standard with a variety of strategies for efficiently searching the parameter space, leading to greatly improved performance of tuned kernels. Second, it allows for unit testing of GPU code from within Python.

Kernel Tuner does not add any additional dependencies to the kernel code, and does not require extensive code changes. Furthermore, it is noteworthy that kernels tuned by Kernel Tuner do not require any changes after tuning to make them production ready--tuned kernels can be used as-is from any host programming language.

Read less



Tags

- GPU High performance computing
- Multi-scale & multi model simulations
- Real time data analysis
- Optimized data handling Big data

Programming Language

Python
CUDA
OpenCL

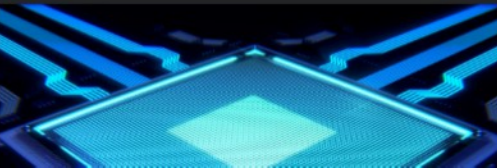
License

Apache-2.0

Participating organizations

netherlands **Science** center

Mentions



Writing Testable GPU Code

By Ben van Werkhoven
April 12, 2018

1. Findability ✓
2. Qualitative measurement of impact
3. Data sources
4. Reproducibility & Citability

Qualitative measurement of impact

netherlands **Science** center

research software directory

Find software

About the directory

GGIR

71

mentions

7

contributors

Converts raw data from wearables into insightful reports for researchers investigating human daily physical activity and sleep.

Get started

624 commits | Last update: October 01, 2018

Cite this software

DOI:

10.5281/zenodo.1429520

Copy to clipboard

Choose a version:

1.6.4

What GGIR can do for you

- GGIR is an R-package to process and analysis multi-day data collected with wearable raw data accelerometers for physical activity and sleep research.
- GGIR uses this information to describe the data per day of measurement or per measurement, including estimates of physical activity, inactivity, and sleep. As part of the pipeline GGIR performs automatic signal calibration, detection of sustained abnormally high values, detection of sensor non-wear and calculation of average magnitude acceleration based on a variety of metrics.
- GGIR is the only open source licensed software that provides a full pipeline for both physical activity and sleep analyses, with a high freedom for the user to configure the analyses to their needs.
- The package has been used for domain science in 50+ publications, and is supported by 8 methodological publications.

+ Read more

Tag

Big data

{} Programming Language

R

License

LGPL-2.0

Participating organizations

Activinsights

Inserm

MRC Epidemiology Unit

netherlands **Science** center

UCL

UNIVERSIDAD DE GRANADA

University of Leicester

Mentions

1

Conference paper

+

68

Journal articles

+

2

Web pages

+

[illegible]

Qualitative measurement of impact

netherlands **eScience** center
research software directory

Find software
About the directory

Case Law App

5

mentions

1

contributor

Web application for exploring the citation network of Dutch court decisions.

Get started

92 commits | Last update: October 10, 2018

Cite this software

Choose a version:

v0.5.0

Choose a citation style:

BibTeX

DOI: 10.5281/zenodo.1454411

Copy to clipboard

Download file

What Case Law App can do for you

- Provides a graphical representation of Dutch case law for researchers
- Allows exploring the network of citations between court decisions
- Contains an example network about employer liability

+ Read more

Tags

Visualization

Text analysis & natural language processing

Programming Language

JavaScript

License

Apache-2.0

Participating organizations

Maastricht University

netherlands **eScience** center

Mentions

How can network analysis lead to a new way of studying court decisions?

By Netherlands eScience Center
April 03, 2017

[Visit our blog](#)

1 Book

+

1 Conference paper

+

1 Newspaper article

+

1 Video recording

+

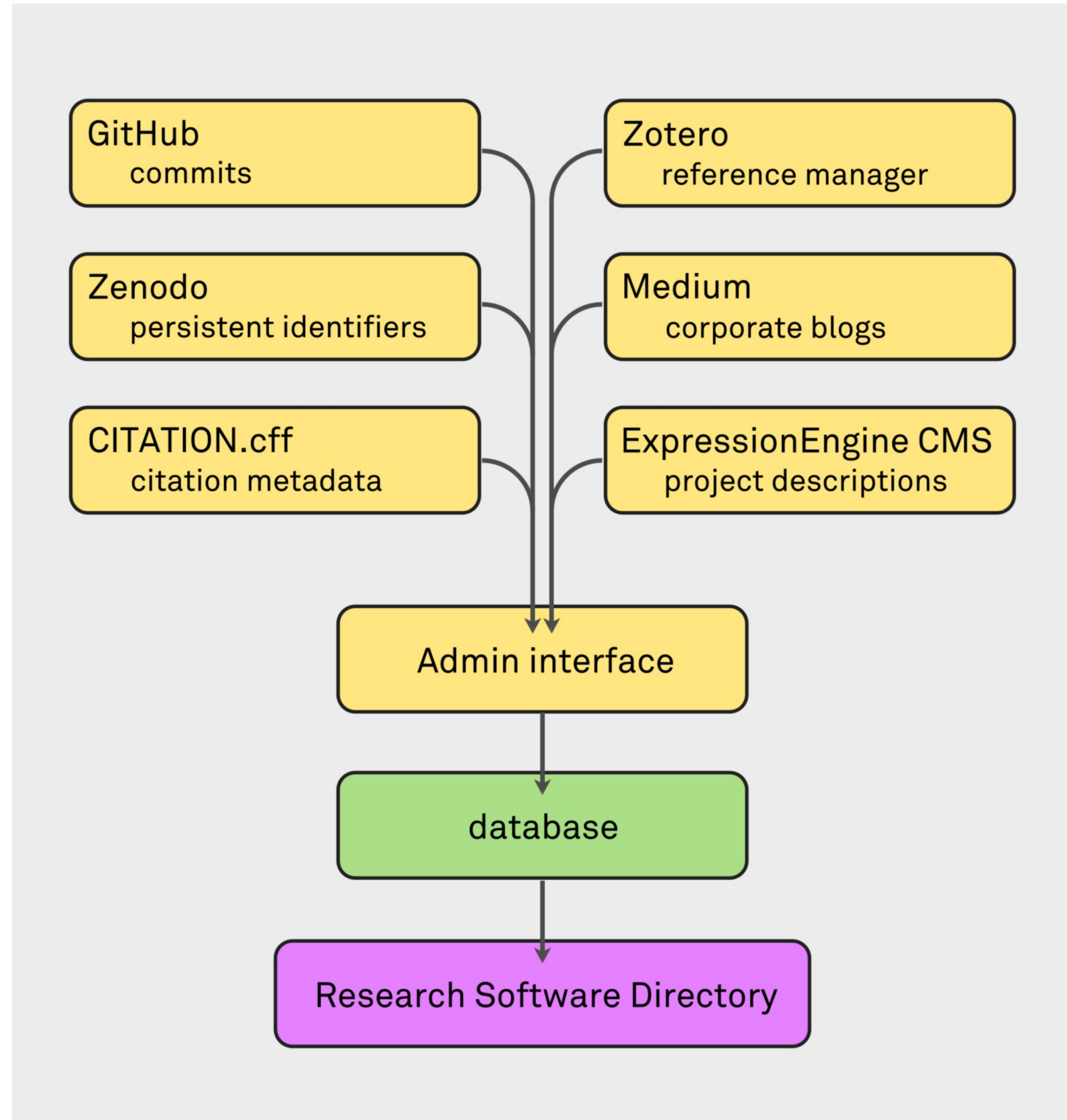
"It's quite amazing to see how, thanks to this tool, a student can, in some ways, outperform the expert."

– Gijs van Dijk , Professor of Private Law, Maastricht University

– Gijs van Dijck , Professor of Private Law, Maastricht University

1. Findability ✓
2. Qualitative measurement of impact ✓
3. Data sources
4. Reproducibility & Citability

Data sources



https://research-software.nl

Xenon | Research Software

https://research-software.nl

Search

netherlands science center research software directory

Find software About the directory

Xenon

6 mentions 9 contributors

If you are using remote machines to do your computations, and don't feel like learning and implementing many different APIs, Xenon is the tool for you.

Get started

2273 commits | Last update: July 09, 2018

Cite this software

DOI: 10.5281/zenodo.1287235 Copy to clipboard

Choose a version: 2.6.2 Choose a citation style: BibTeX Download file

What Xenon can do for you

- Provides an easy-to-use interface for distributed computing developers
- Enables the use of different file transfer protocols and scheduling systems on remote machines
- No need to learn and implement many different APIs
- Successfully used in many eScience tools and projects

Read more

Tags

Big data Optimized data handling High performance computing

Programming Language

Java

License

Apache-2.0

Participating organizations

netherlands science center VU UNIVERSITY AMSTERDAM

admin interface

Research Software Directory

https://www.research-software.nl/admin

Search

python-pcl
PyXenon
QMflows
QTLTableMiner++
recipy
ReGIS
Research Software Directory
Rig
ROOT-conda-recipes
Sagecal
SalientDescriptor-matlab
SalientDetector-matlab
SalientDetector-python
scriptcwl
ShiCo
SIGA.py
SPOT
StoryTeller
Structure from Motion
sv-callers
SyGMA
Texcavator
The CrowdTruth Framework
Twiqs
Via Appia Visualization
Xenon
Xenon command line interface
Xenon gRPC server
xenon-flow
xtas

Brand name

Xenon

Is published

☒

Is featured

☒

Use with caution, not everything can be featured

Short statement

Short **software statement**: in max. 30 words explain when your software might be useful and what it solves.
Example for Xenon:
If you are using remote machines to do your computations, and don't feel like learning and implementing many different APIs, Xenon is the tool for you
If you are using remote machines to do your computations, and don't feel like learning and implementing many different APIs, Xenon is the tool for you.

Bullet list

Answer the following questions (in Markdown with bullet points (*)):

What does your software provide for what user?
What does your software do?
What makes your software unique?
List some highlights/awards:
Example for Xenon:

- * Provides an easy-to-use interface for distributed computing developers
- * Enables the use of different file transfer protocols and scheduling systems on remote machines
- * No need to learn and implement many different APIs
- * Successfully used in many eScience tools and projects
- * Provides an easy-to-use interface for distributed computing developers
- * Enables the use of different file transfer protocols and scheduling systems on remote machines
- * No need to learn and implement many different APIs
- * Successfully used in many eScience tools and projects

Read more

Text shown when Read more button is pressed

Many applications use remote storage and compute resources. To do so, they need to include code to interact with the scheduling systems and file transfer protocols used on those remote machines.

Unfortunately, many different scheduler systems and file transfer protocols exist, often with completely different programming interfaces. This makes it difficult for applications to switch to a different system or support multiple remote systems simultaneously.

Xenon solves this problem by providing a single programming interface to many different types of remote resources. As a result, changing from one scheduler to

Concept DOI

The Zenodo concept DOI. Not an URL
10.5281/zenodo.597993

Getting started URL

https://github.com/NLeSC/Xenon

Human readable identifier in url for this item

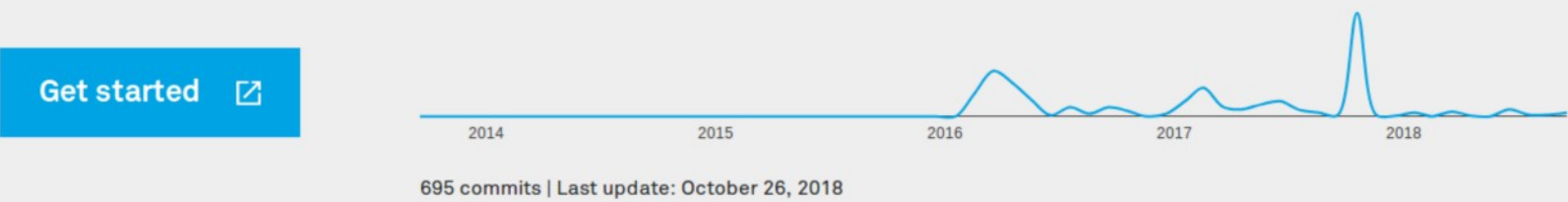
1. Findability ✓
2. Qualitative measurement of impact ✓
3. Data sources ✓
4. Reproducibility & Citability

Kernel Tuner

148

mentions contributors

Kernel Tuner greatly simplifies the development of highly-optimized and auto-tuned CUDA, OpenCL, and C code, supporting many advanced use-cases and optimization strategies that speed up the auto-tuning process.



Cite this software

DOI: 10.5281/zenodo.1220114

Copy to clipboard

Choose a version: 0.1.9

Choose a citation style: BibTeX

Download file

What Kernel Tuner can do for you

- Allows developers to easily unit test and auto-tune GPU code
- Generic auto-tuning of user-defined parameters for CUDA, OpenCL, and C kernels
- Supports more than 20 different search optimization methods to speedup tuning
- Successfully used in 5 different eScience projects, across various disciplines

Kernel Tuner simplifies the development of efficient GPU programs, or *kernels*. It does so by making kernels written in C/C++, OpenCL, or CUDA accessible from Python, while taking care of the required synchronization between data kept in host memory and data kept in device memory.

This has a number of advantages. First, it simplifies *auto-tuning* of the kernel parameters. In fact, Kernel Tuner comes standard with a variety of strategies for efficiently searching the parameter space, leading to greatly improved performance of tuned kernels. Second, it allows for unit testing of GPU code from within Python.

Kernel Tuner does not add any additional dependencies to the kernel code, and does not require extensive code changes. Furthermore, it is noteworthy that kernels tuned by Kernel Tuner do not require any changes after tuning to make them production ready--tuned kernels can be used as-is from any host programming language.

X Read less

Tags

GPU High performance computing

Multi-scale & multi model simulations

Real time data analysis

Optimized data handling Big data

Programming Language

Python CUDA OpenCL

License

Apache-2.0

Participating organizations

netherlands **Science** center

Mentions



Writing Testable GPU Code

By Ben van Werkhoven
April 12, 2018

Cite this software

DOI: 10.5281/zenodo.1220114

Copy to clipboard

Choose a version: 0.1.9

Choose a citation style: BibTeX

Download file


- Emphasize software can be cited
- Make it easy to do so

Software only has DOI:

Cite this software

DOI:

10.5281/zenodo.1220114


 Copy to clipboard

Software has DOI resulting from
GitHub-Zenodo integration:


Cite this software

DOI:

10.5281/zenodo.1220114

 Copy to clipboard

Choose a version:

0.1.9 

Same as previous,
plus release has CITATION.cff

Cite this software

DOI:

10.5281/zenodo.1220114


 Copy to clipboard

Choose a version:

0.1.9 

Choose a citation style:

BibTeX 

 Download file

Let's stay in touch

☎ +31 (0)20 460 4770

✉ info@esciencecenter.nl

🌐 www.esciencecenter.nl

📖 blog.esciencecenter.nl

🐦 eScienceCenter

♥ eScienceCenter

in [linkd.in/1j2uS8S](https://www.linkedin.com/company/esciencecenter/)

<https://www.research-software.nl>