Karlsruhe Institute of Technology

UNIVERSITAT JAUME I

Are we doing the right thing?
- A Critical Analysis of the Academic HPC Community

20th PDSEC Workshop
May 24th 2019 | Rio de Janeiro


Hartwig Anzt

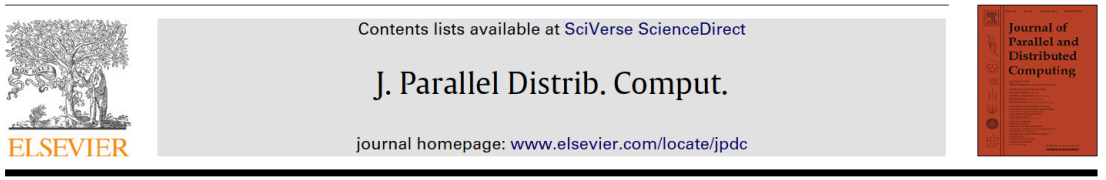

Goran Flegar

Hartwig Anzt, Goran Flegar

The HPC Algorithm Landscape

The Typical Publication in HPC Conferences / Journals

- An article describing a **new algorithm / implementation** outperforming existing solutions.
- **Performance benchmarks** on high-end HPC resources (not even archived)
- **Internal prototype code** (not publicly accessible)


Contents lists available at SciVerse ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

A block-asynchronous relaxation method for graphics processing units

Hartwig Anzt [a,*], Stanimire Tomov [b], Jack Dongarra [b,c,d], Vincent Heuveline [a]

[a] Karlsruhe Institute of Technology, Germany
[b] University of Tennessee Knoxville, USA
[c] Oak Ridge National Laboratory, USA
[d] University of Manchester, UK

HIGHLIGHTS

- Block-asynchronous relaxation on GPU-accelerated systems.
- Method's high iteration rate compensates a low convergence rate (competitive to CG).
- GPU thread-block scheduling reduces non-deterministic behavior and stochastic noise.
- Analysis of different communication strategies for multi-GPU usage.
- Block-asynchronous relaxation inherently tolerant to hardware error.

New Algorithm

GPU implementation

Performance results

The HPC Algorithm Landscape

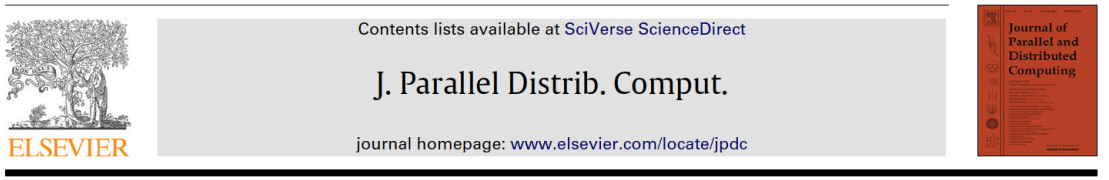
Karlsruhe Institute of Technology

The Typical Publication in HPC Conferences / Journals

- An article describing a **new algorithm / implementation** outperforming existing solutions.
- **Performance benchmarks** on high-end HPC resources (not even archived)
- **Internal prototype code** (not publicly accessible)

How does the community benefit from reading this?

\+ **New ideas** presented;
\+ **Performance evaluations** presented;
\- Performance evaluations are typically "**selective**";
\- Users need to **re-implement code**;
\- Difficult if **few details** are provided;



A block-asynchronous relaxation method for graphics processing units

Hartwig Anzt [a,*], Stanimire Tomov [b], Jack Dongarra [b,c,d], Vincent Heuveline [a]

[a] Karlsruhe Institute of Technology, Germany
[b] University of Tennessee Knoxville, USA
[c] Oak Ridge National Laboratory, USA
[d] University of Manchester, UK

New Algorithm

HIGHLIGHTS

- Block-asynchronous relaxation on GPU-accelerated systems.
- Method's high iteration rate compensates a low convergence rate (competitive to CG).
- GPU thread-block scheduling reduces non-deterministic behavior and stochastic noise.
- Analysis of different communication strategies for multi-GPU usage.
- Block-asynchronous relaxation inherently tolerant to hardware error.

GPU implementation

Performance results

The HPC Algorithm Landscape

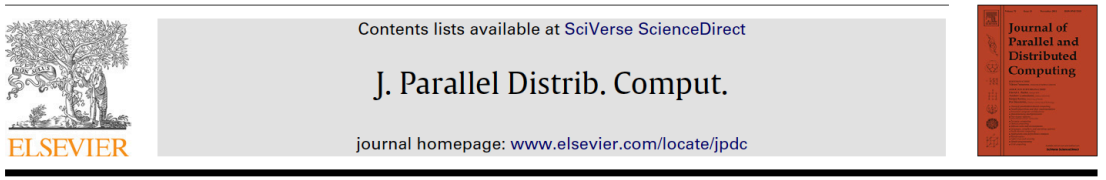
Karlsruhe Institute of Technology

The Typical Publication in HPC Conferences / Journals

- An article describing a **new algorithm / implementation** outperforming existing solutions.
- **Performance benchmarks** on high-end HPC resources (not even archived)
- **Internal prototype code** (not publicly accessible)

How does the community benefit from reading this?

+ **New ideas** presented;
+ **Performance evaluations** presented;
- Performance evaluations are typically "**selective**";
- Users need to **re-implement code**;
- Difficult if **few details** are provided;



A block-asynchronous relaxation method for graphics processing units

Hartwig Anzt [a,*], Stanimire Tomov [b], Jack Dongarra [b,c,d], Vincent Heuveline [a]

[a] Karlsruhe Institute of Technology, Germany
[b] University of Tennessee Knoxville, USA
[c] Oak Ridge National Laboratory, USA
[d] University of Manchester, UK

New Algorithm

HIGHLIGHTS

- Block-asynchronous relaxation on GPU-accelerated systems.
- Method's high iteration rate compensates a low convergence rate (competitive to CG).
- GPU thread-block scheduling reduces non-deterministic behavior and stochastic noise.
- Analysis of different communication strategies for multi-GPU usage.
- Block-asynchronous relaxation inherently tolerant to hardware error.

GPU implementation

Performance results

UNESCO science report: towards 2030

The HPC Algorithm Landscape

Karlsruhe Institute of Technology

The Typical Publication in HPC Conferences / Journals

- An article describing a **new algorithm / implementation** outperforming existing solutions.
- **Performance benchmarks** on high-end HPC resources (not even archived)
- **Internal prototype code** (not publicly accessible)

How does the community benefit from reading this?

\+ **New ideas** presented;
\+ **Performance evaluations** presented;
\- Performance evaluations are typically "**selective**";
\- Users need to **re-implement code**;
\- Difficult if **few details** are provided;

In a **perfect world**, new algorithms, implementations & performance results are

- fully reproducible;
- publicly accessible;
- ready to be used by the community / domain scientists;
- integrated into community packages;

Contents lists available at SciVerse ScienceDirect

J. Parallel Distrib. Comput.

journal homepage: www.elsevier.com/locate/jpdc

A block-asynchronous relaxation method for graphics processing units

Hartwig Anzt [a,*], Stanimire Tomov [b], Jack Dongarra [b,c,d], Vincent Heuveline [a]

[a] Karlsruhe Institute of Technology, Germany
[b] University of Tennessee Knoxville, USA
[c] Oak Ridge National Laboratory, USA
[d] University of Manchester, UK

New Algorithm

HIGHLIGHTS

- Block-asynchronous relaxation on GPU-accelerated systems.
- Method's high iteration rate compensates a low convergence rate (competitive to CG).
- GPU thread-block scheduling reduces non-deterministic behavior and stochastic noise.
- Analysis of different communication strategies for multi-GPU usage.
- Block-asynchronous relaxation inherently tolerant to hardware error.

GPU implementation

Performance results

Figure 1.5: **Trends in scientific publications worldwide, 2008 and 2014**

13.7% Growth in publications with authors from Europe between 2008 and 2014, the region with the greatest share of publications: 39.3%

60.1% Growth in publications with authors from Africa between 2008 and 2014

109.6% Growth in publications with authors from Arab states between 2008 and 2014

UNESCO science report: towards 2030

The HPC Algorithm Landscape

Karlsruhe Institute of Technology

The Typical Publication in HPC Conferences / Journals

- An article describing a **new algorithm / implementation** outperforming existing solutions.
- **Performance benchmarks** on high-end HPC resources (not even archived)
- **Internal prototype code** (not publicly accessible)

How does the community benefit from reading this?

\+ **New ideas** presented;
\+ **Performance evaluations** presented;
\- Performance evaluations are typically "**selective**";
\- Users need to **re-implement code**;
\- Difficult if **few details** are provided;

In a **perfect world**, new algorithms, implementations & performance results are

- fully reproducible;
- publicly accessible;
- ready to be used by the community / domain scientists;
- integrated into community packages;

Established community software packages

- are the powertrain behind many scientific simulation codes;
- often fall short in providing production-ready implementations of novel algorithms;
- often accept merge requests that lack comprehensive documentation and rigorous performance assessment;

The HPC Algorithm Landscape

The Typical Publication in HPC Conferences / Journals

- An article describing a **new algorithm / implementation** outperforming existing solutions.
- **Performance benchmarks** on high-end HPC resources (not even archived)
- **Internal prototype code** (not publicly accessible)

How does the community benefit from reading this?

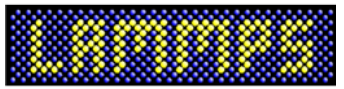
+ **New ideas** presented;
+ **Performance evaluations** presented;
- Performance evaluations are typically "**selective**";
- Users need to **re-implement code**;
- Difficult if **few details** are provided;

In a **perfect world**, new algorithms, implementations & performance results are

- fully reproducible;
- publicly accessible;
- ready to be used by the community / domain scientists;
- integrated into community packages;

Established community software packages

- are the powertrain behind many scientific simulation codes;
- often fall short in providing production-ready implementations of novel algorithms;
- often accept merge requests that lack comprehensive documentation and rigorous performance assessment;



The HPC Algorithm Landscape

The Typical Publication in HPC Conferences / Journals

- An article describing a **new algorithm / implementation** outperforming existing solutions.
- **Performance benchmarks** on high-end HPC resources (not even archived)
- **Internal prototype code** (not publicly accessible)

How does the community benefit from reading this?

+ **New ideas** presented;
+ **Performance evaluations** presented;
- Performance evaluations are typically "**selective**";
- Users need to **re-implement code**;
- Difficult if **few details** are provided;

In a **perfect world**, new algorithms, implementations & performance results are

- fully reproducible;
- publicly accessible;
- ready to be used by the community / domain scientists;
- integrated into community packages;

Established community software packages

- are the powertrain behind many scientific simulation codes;
- often fall short in providing production-ready implementations of novel algorithms;
- often accept merge requests that lack comprehensive documentation and rigorous performance assessment;

Why are we not changing the system?

- $\text{effort}(\text{ Prototype Code }) << \text{effort}(\text{ Production Code })$;
- The academic system does not reward software development;
- Promotion and appointability based on scientific papers;
- Sluggish acceptance of the "*scientific software engineer*";
- Authors want to keep algorithms confidential;

The HPC Algorithm Landscape

The Typical Publication in HPC Conferences / Journals

- An article describing a **new algorithm / implementation** outperforming existing solutions.
- **Performance benchmarks** on high-end HPC resources (not even archived)
- **Internal prototype code** (not publicly accessible)

How does the community benefit from reading this?

+ **New ideas** presented;
+ **Performance evaluations** presented;
- Performance evaluations are typically "**selective**";
- Users need to **re-implement code**;
- Difficult if **few details** are provided;

In a **perfect world**, new algorithms, implementations & performance results are

- fully reproducible;
- publicly accessible;
- ready to be used by the community / domain scientists;
- integrated into community packages;

Established community software packages

- are the powertrain behind many scientific simulation codes;
- often fall short in providing production-ready implementations of novel algorithms;
- often accept merge requests that lack comprehensive documentation and rigorous performance assessment;

Why are we not changing the system?

- $\text{effort}(\text{ Prototype Code }) << \text{effort}(\text{ Production Code })$;
- The academic system does not reward software development;
- Promotion and appointability based on scientific papers;
- Sluggish acceptance of the "*scientific software engineer*";
- Authors want to keep algorithms confidential;

Extremely inefficient and unsatisfying!

Let's try to change the System!

- Scientific papers and the Hirsch-Index (H-Index) will remain the Gold Standard for appointability.

- **We can move the publication system towards crediting software contributions!**
  - Reproducibility initiatives (ACM Journals)
  - Artifact evaluation (Euromicro, Supercomputing,…)

Let's try to change the System!

- Scientific papers and the Hirsch-Index (H-Index) will remain the Gold Standard for appointability.

- **We can move the publication system towards crediting software contributions!**
  - Reproducibility initiatives (ACM Journals)
  - Artifact evaluation (Euromicro, Supercomputing,…)

- Let's go even further:

  Well-documented software patches as full conference contribution

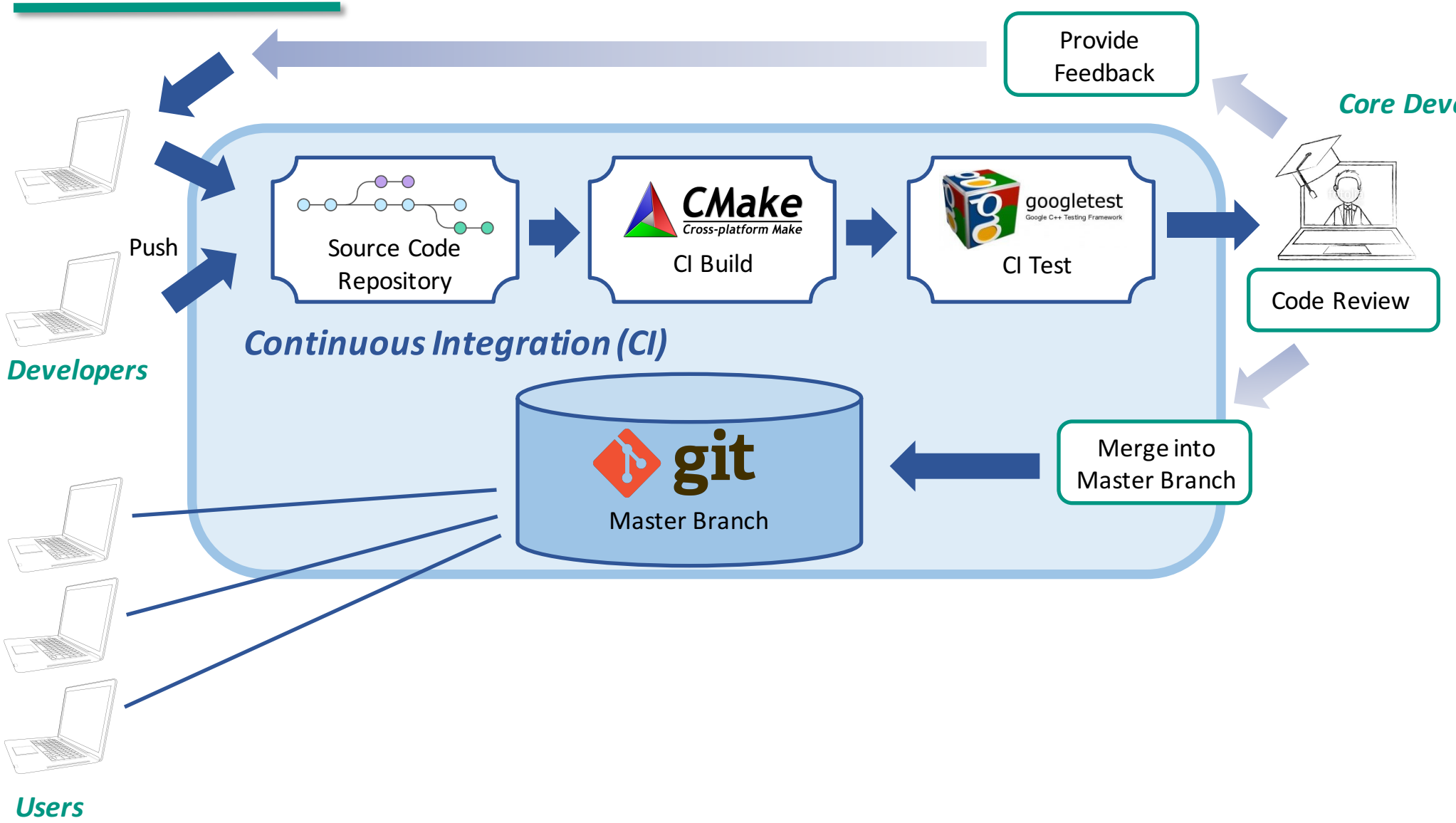
  - Well-documented software patches contributing new algorithms/ implementations;
  - Comprehensive code documentation and performance assessment;
  - Feedback from code reviewers;
  - Software patch description + performance evaluation included in conference proceedings;

Software Patches as Conference Contribution

Karlsruhe Institute of Technology

- ✓ **Full reproducibility** and **traceability** is ensured;
- ✓ Not only reviewers but the complete **community can track the software patch**;
- ✓ The versioning systems helps to **identify the main contributors** of a software contribution, **ensuring full recognition**;
- ✓ The versioning systems also **links to the right person in case of technical questions**;
- ✓ **Novel algorithms** and hardware-optimized implementations are **quickly integrated into community packages**;
- ✓ The **code quality is increased** as the community can comment on the patches;
- ✓ Software patches as conference contributions naturally imply an **extremely high level of code documentation**;
- ✓ Presenting patches at a conference **makes** the whole **community aware of a new feature**;
- ✓ **Domain scientists** can directly **interact with software developers**;

A Healthy Software Development Cycle

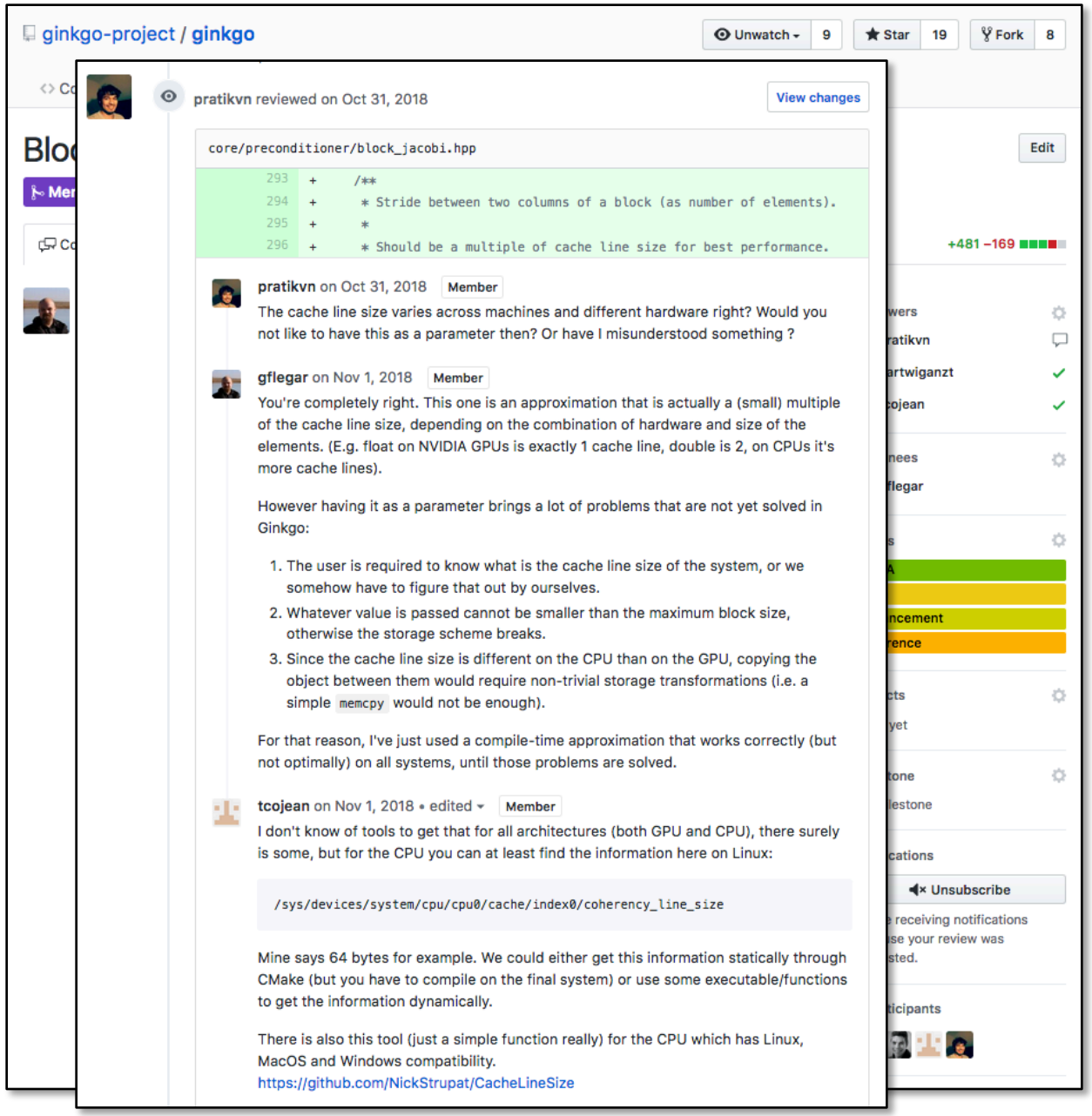


Software Patches

- Software patches usually submitted as *merge-/ push- request* in the *software versioning system* (e.g. Git).
- The patches are accompanied by detailed documentation explaining code functionality and feature usage.

Software Patches

- Software patches usually submitted as *merge-/ push- request* in the *software versioning system* (e.g. Git).
- The patches are accompanied by detailed documentation explaining code functionality and feature usage.
- The community can comment and review the code.

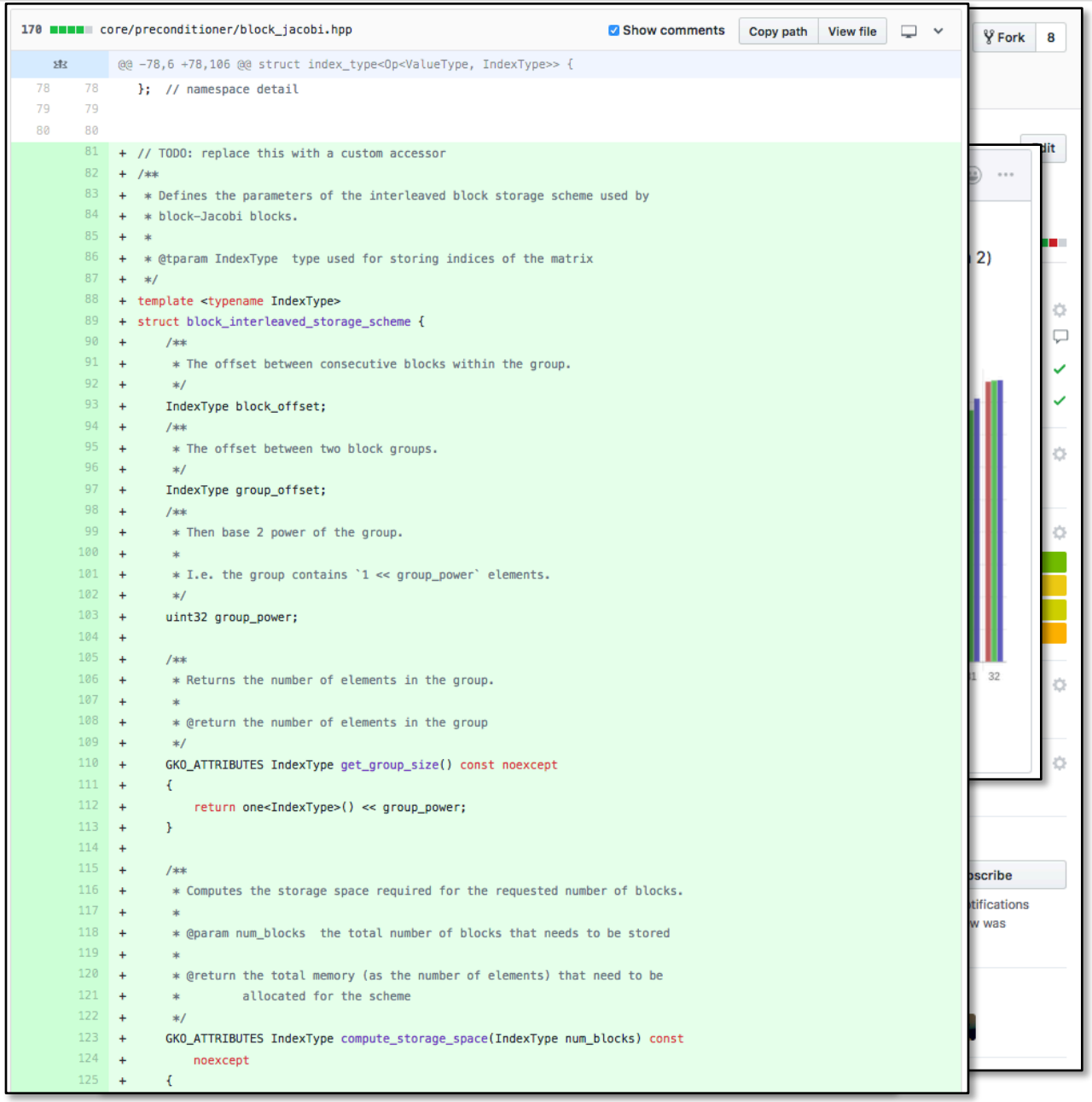


Software Patches

- Software patches usually submitted as *merge-/ push- request* in the *software versioning system* (e.g. Git).
- The patches are accompanied by detailed documentation explaining code functionality and feature usage.
- The community can comment and review the code.
- The submitter can attach a performance analysis to the software patch.

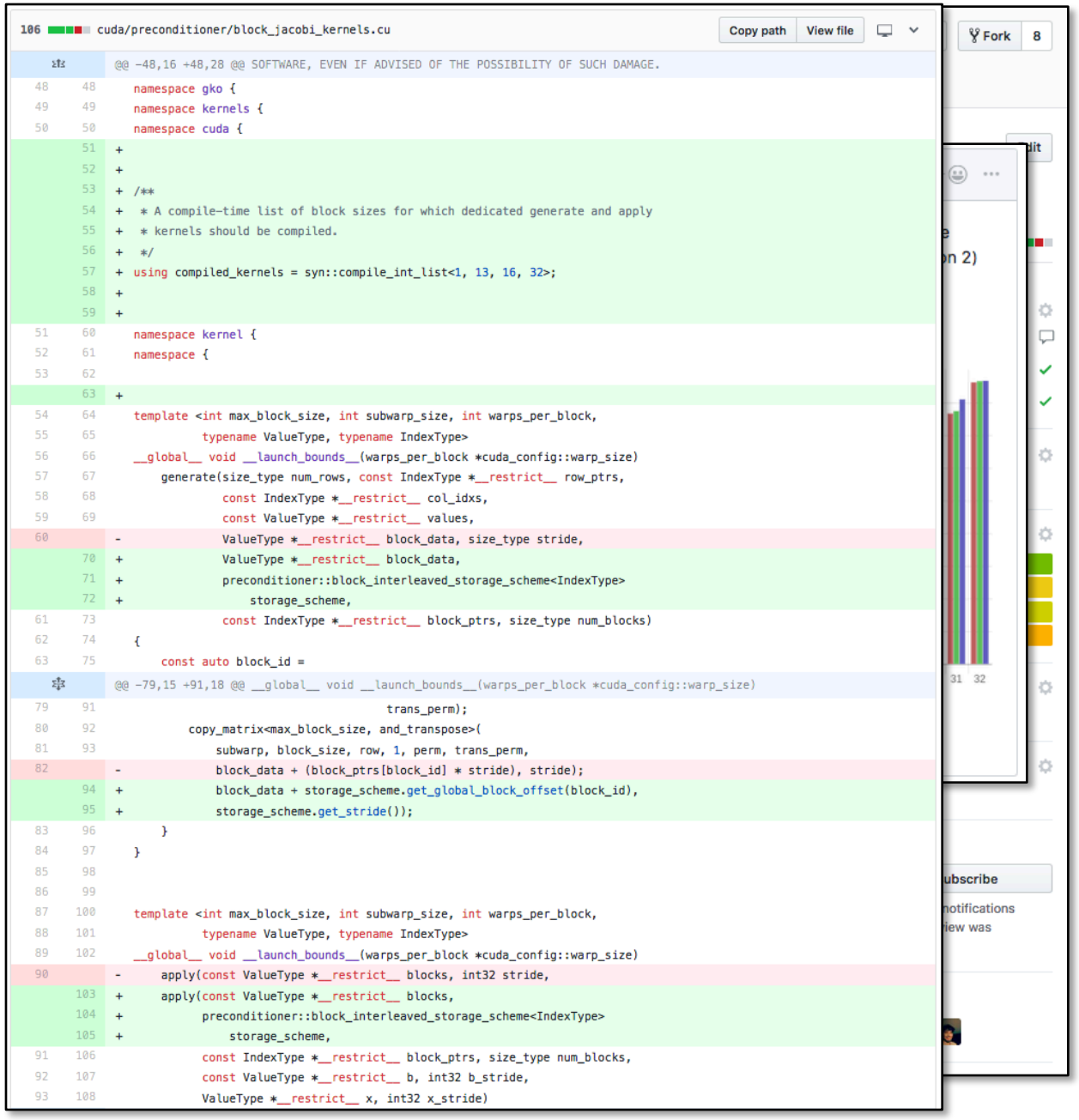
Software Patches

- Software patches usually submitted as *merge-/ push- request* in the *software versioning system* (e.g. Git).
- The patches are accompanied by detailed documentation explaining code functionality and feature usage.
- The community can comment and review the code.
- The submitter can attach a performance analysis to the software patch.
- Software patches can either add new functionality…



Software Patches

- Software patches usually submitted as *merge-/ push- request* in the *software versioning system* (e.g. Git).
- The patches are accompanied by detailed documentation explaining code functionality and feature usage.
- The community can comment and review the code.
- The submitter can attach a performance analysis to the software patch.
- Software patches can either add new functionality…
  … or change / enhance existing code.



Software Patches as Conference Contribution

Karlsruhe Institute of Technology

Envisioned Workflow:

1. The algorithm/implementation **developer submits a software patch to a community package** with
   - detailed description of the functionality and code documentation;
   - comprehensive performance assessment;
   - mark the patch for a conference contribution;
2. The **core development team and the community**
   - **comments** on the algorithm, the implementation, and the performance;
   - **reviews** and ultimately merges the patch;
3. The **developer submits the patch as a conference contribution**
   - linking to all documentation, performance results, and comments;
   - acknowledging significant comments from community;

Software Patches as Conference Contribution

Karlsruhe Institute of Technology

Envisioned Workflow:

1. The algorithm/implementation **developer submits a software patch to a community package** with
   - detailed description of the functionality and code documentation;
   - comprehensive performance assessment;
   - mark the patch for a conference contribution;
2. The **core development team and the community**
   - **comments** on the algorithm, the implementation, and the performance;
   - **reviews** and ultimately merges the patch;
3. The **developer submits the patch as a conference contribution**
   - linking to all documentation, performance results, and comments;
   - acknowledging significant comments from community;
4. The **conference committee / external reviewers** do a "light" review of functionality, documentation, performance.
5. If accepted, the conference contribution is presented along with a **user tutorial or application examples**;
6. The submission is as a **regular paper** included in the conference proceedings
   - potentially featuring a shorter general introduction;
   - including the algorithm description and performance assessment;
     potentially including code segments, digital artifacts, or a link to the merge request;
   - listing all (significant) code reviewers / commenters;

Summary and Limitations

Summary:

- We argue for accepting **well-documented and performance-analyzed software patches** as **regular conference contribution** included in the conference proceedings.
- **Versioning Systems** are an excellent tool to **track scientific software development** preserving **intellectual property**.

Limitations:

- Not applicable to Position Papers (This paper being an example ☺).
- Not applicable to purely theoretical papers or papers presenting an idea, only.
- Algorithms / implementations can fall under export control.

In a larger picture, **accepting software patches as conference contribution** is another step **in the direction of entrenching scientific software development** as an **academic field**, and moving the academic evaluation system from traditional metrics (like the H-Index) towards **community-advancing software contributions**.

better scientific software

https://bssw.io/

HELMHOLTZ RESEARCH FOR GRAND CHALLENGES

Helmholtz Impuls und Vernetzungsfond
VH-NG-1241