Growing Your Open-Source Projects

2019 Workshop on Open-Source EDA Technology (WOSET)

International Conference of Computer Aided Design (ICCAD)

Nov 7th | Westminster, CO

GitHub: https://github.com/tsung-wei-huang

Tsung-Wei Huang
Department of Electrical and Computer Engineering
University of Utah, Salt Lake City, UT

The EDA/CAD Research Landscape

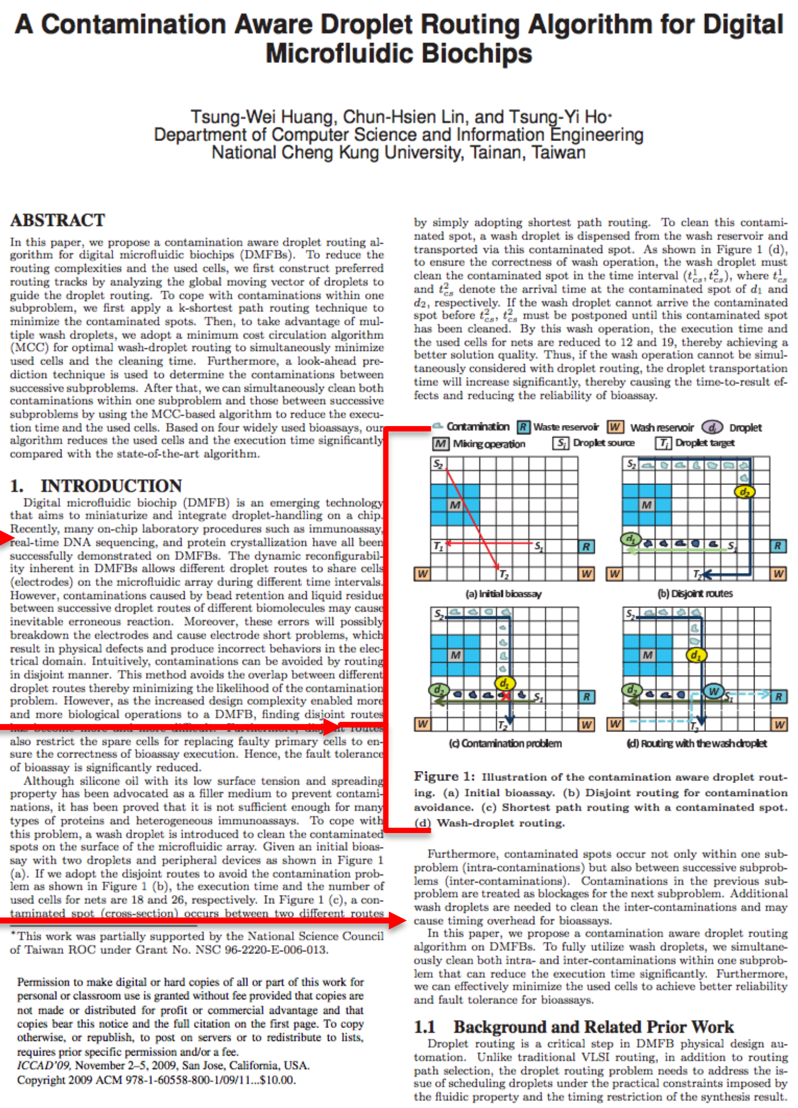
- ❑ **My first ICCAD paper …**
  - ❑ New algorithm
  - ❑ New better results
  - ❑ Benchmarks & testcases
  - ❑ Internal prototype code

New problem formulation in …

New algorithm and implementation to outperform existing solutions by …

Experimental results showed …

2009 IEEE/ACM ICCAD, San Jose, CA



A Contamination Aware Droplet Routing Algorithm for Digital Microfluidic Biochips

Tsung-Wei Huang, Chun-Hsien Lin, and Tsung-Yi Ho*
Department of Computer Science and Information Engineering
National Cheng Kung University, Tainan, Taiwan

ABSTRACT

In this paper, we propose a contamination aware droplet routing algorithm for digital microfluidic biochips (DMFBs). To reduce the routing complexities and the used cells, we first construct preferred routing tracks by analyzing the global moving vector of droplets to guide the droplet routing. To cope with contaminations within one subproblem, we first apply a k-shortest path routing technique to minimize the contaminated spots. Then, to take advantage of multiple wash droplets, we adopt a minimum cost circulation algorithm (MCC) for optimal wash-droplet routing to simultaneously minimize used cells and the cleaning time. Furthermore, a look-ahead prediction technique is used to determine the contaminations between successive subproblems. After that, we can simultaneously clean both contaminations within one subproblem and those between successive subproblems by using the MCC-based algorithm to reduce the execution time and the used cells. Based on four widely used bioassays, our algorithm reduces the used cells and the execution time significantly compared with the state-of-the-art algorithm.

1. INTRODUCTION

Digital microfluidic biochip (DMFB) is an emerging technology that aims to miniaturize and integrate droplet-handling on a chip. Recently, many on-chip laboratory procedures such as immunoassay, real-time DNA sequencing, and protein crystallization have all been successfully demonstrated on DMFBs. The dynamic reconfigurability inherent in DMFBs allows different droplet routes to share cells (electrodes) on the microfluidic array during different time intervals. However, contaminations caused by bead retention and liquid residue between successive droplet routes of different biomolecules may cause inevitable erroneous reaction. Moreover, these errors will possibly breakdown the electrodes and cause electrode short problems, which result in physical defects and produce incorrect behaviors in the electrical domain. Intuitively, contaminations can be avoided by routing in disjoint manner. This method avoids the overlap between different droplet routes thereby minimizing the likelihood of the contamination problem. However, as the increased design complexity enabled more and more biological operations to a DMFB, finding disjoint routes has become more and more difficult. Furthermore, disjoint routes also restrict the spare cells for replacing faulty primary cells to ensure the correctness of bioassay execution. Hence, the fault tolerance of bioassay is significantly reduced.

Although silicone oil with its low surface tension and spreading property has been advocated as a filler medium to prevent contaminations, it has been proved that it is not sufficient enough for many types of proteins and heterogeneous immunoassays. To cope with this problem, a wash droplet is introduced to clean the contaminated spots on the surface of the microfluidic array. Given an initial bioassay with two droplets and peripheral devices as shown in Figure 1 (a). If we adopt the disjoint routes to avoid the contamination problem as shown in Figure 1 (b), the execution time and the number of used cells for nets are 18 and 26, respectively. In Figure 1 (c), a contaminated spot (cross-section) occurs between two different routes by simply adopting shortest path routing. To clean this contaminated spot, a wash droplet is dispensed from the wash reservoir and transported via this contaminated spot. As shown in Figure 1 (d), to ensure the correctness of wash operation, the wash droplet must clean the contaminated spot in the time interval (t^1_{cs}, t^2_{cs}), where t^1_{cs} and t^2_{cs} denote the arrival time at the contaminated spot of d_1 and d_2, respectively. If the wash droplet cannot arrive the contaminated spot before t^2_{cs}, t^2_{cs} must be postponed until this contaminated spot has been cleaned. By this wash operation, the execution time and the used cells for nets are reduced to 12 and 19, thereby achieving a better solution quality. Thus, if the wash operation cannot be simultaneously considered with droplet routing, the droplet transportation time will increase significantly, thereby causing the time-to-result effects and reducing the reliability of bioassay.

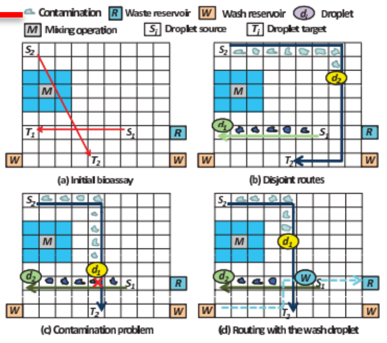


Figure 1: Illustration of the contamination aware droplet routing. (a) Initial bioassay. (b) Disjoint routing for contamination avoidance. (c) Shortest path routing with a contaminated spot. (d) Wash-droplet routing.

Furthermore, contaminated spots occur not only within one subproblem (intra-contaminations) but also between successive subproblems (inter-contaminations). Contaminations in the previous subproblem are treated as blockages for the next subproblem. Additional wash droplets are needed to clean the inter-contaminations and may cause timing overhead for bioassays.

In this paper, we propose a contamination aware droplet routing algorithm on DMFBs. To fully utilize wash droplets, we simultaneously clean both intra- and inter-contaminations within one subproblem that can reduce the execution time significantly. Furthermore, we can effectively minimize the used cells to achieve better reliability and fault tolerance for bioassays.

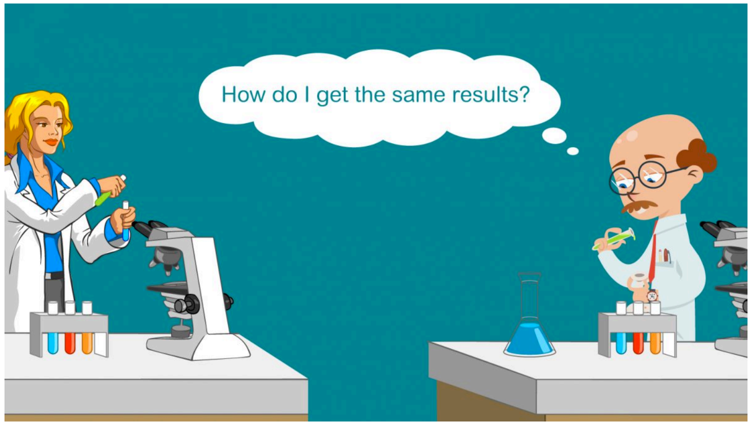
1.1 Background and Related Prior Work

Droplet routing is a critical step in DMFB physical design automation. Unlike traditional VLSI routing, in addition to routing path selection, the droplet routing problem needs to address the issue of scheduling droplets under the practical constraints imposed by the fluidic property and the timing restriction of the synthesis result.

*This work was partially supported by the National Science Council of Taiwan ROC under Grant No. NSC 96-2220-E-006-013.

A Critical Question …

- ❑ ***How does the community benefit from reading this?***
  - ☺ Presented a new problem formulation
  - ☺ Presented a new algorithm and implementation
  - ☺ Presented large improvement over existing solutions
  - ☹ Performance evaluation is “selective”
  - ☹ Difficult to “reproduce” the result
  - ☹ Wasted time on “re-implementing” the code



We want new algorithms & results:
- Open and accessible
- Fully reproducible
- Easy to integrate to my packages
- Ready to use/alter by other scientists

Why Are We Sluggishly Changing this?

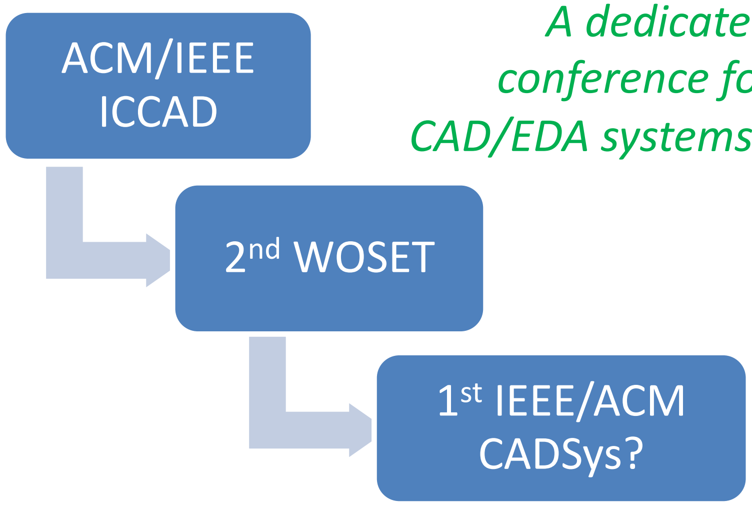
- **From the academic perspective …**
  - effort (prototype code) << effort (production code)
  - Does not reward software/system development
  - Promotion is largely based on scientific papers
  - Slow acceptance of the scientific software engineer
- **From the industrial perspective …** *
  - cost (software error) << cost (hardware error)
  - Wants to keep algorithms/IPs confidential
  - Tools are highly customer-driven, lacking API standards
  - The monopoly locks people to proprietary tools

Extremely inefficient and unsatisfying!

** Conversation with industrial partners in EDA/CAD companies*

The Most Essential Building Block: Mindset

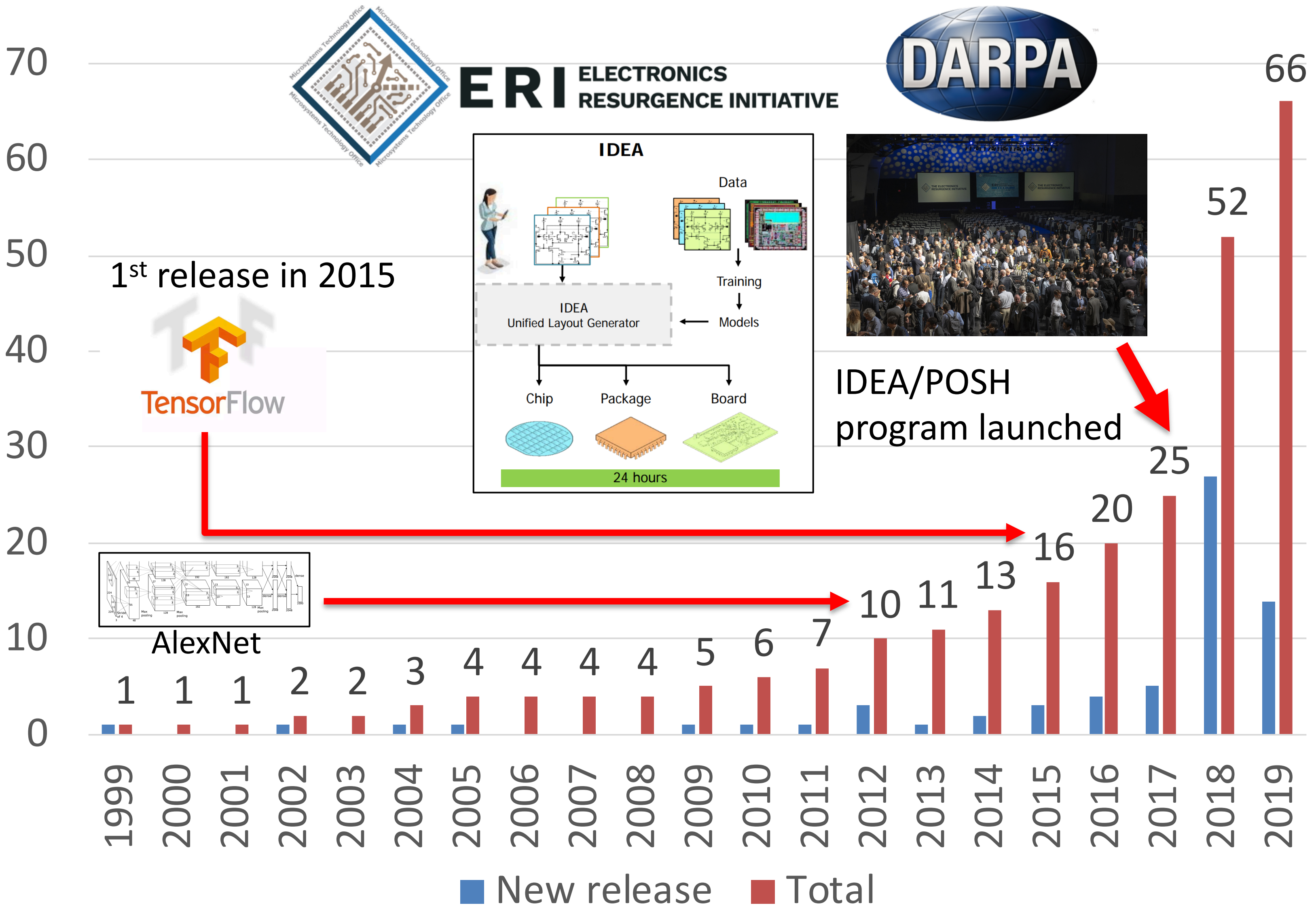
- ❑ **Let's work together to change the system**
  - ❑ Open source to enable quick sharing of new ideas
- ❑ **Publication systems should credit software dev**
  - ❑ Innovation should include system implementation
    - API, software architecture, documentation, design strategies
  - ❑ Artifact reproducibility evaluation using ACM badges



A dedicated conference for CAD/EDA systems?

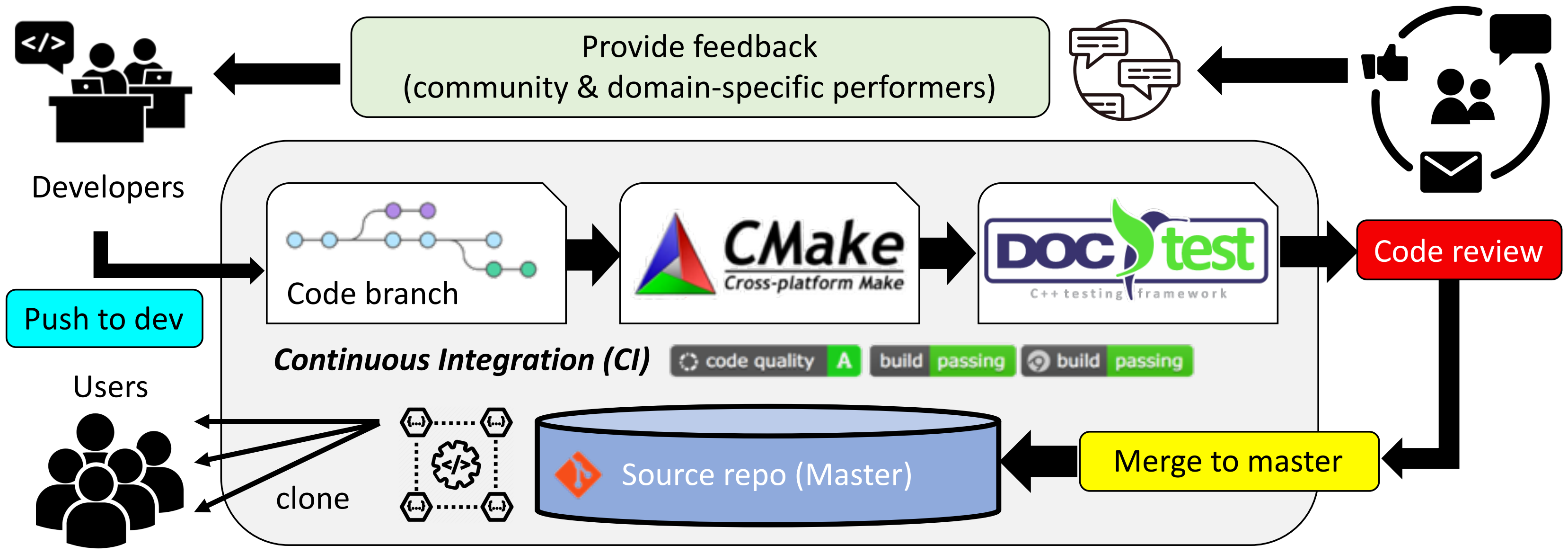
Let's go even further:
- 39th ICCAD to include 30% tool papers
- Code review as a main judge
- TPC will include code reviewers
- Software patches are contributions
- 1st ACM/IEEE CADSys conference

Open-Source EDA Projects Activities



We need to make ourselves open so we can engage more talented people to contribute to this community

A Healthy Open-source Development Cycle



1. Understand your users and what you are aiming for
2. Things to know in creating a repository
3. Prepare an informative README and documentation
4. Set up a contribution guideline
5. Iterate the feedback loop

Understand What Your Users Need

- ❑ **Roughly speaking …**
  - ❑ Developers take your project to do "derived" work
    - For example, a parallel programming library
  - ❑ End users take your project to do "standalone" work
    - For example, a C++ debugger or a performance profiler

Developers are normally respectful

Talk technically;
Care API and reference;
Write code and software;

Talk generally;
Care doc and usability;
Use software and tools;

End-users are often friendly …

Open-source owners can be both developers and end-users, but it's important to understand the target users of your projects

Code of Conduct

- ❑ **It is a free world, especially in open source**
  - ❑ You cannot force others to use your tools
  - ❑ No ones owe you to use your tools
- ❑ **Put respect to the highest standards**
  - ❑ Nobody is ever going to be the top coder in the world
  - ❑ Open source means open collaboration
    - Minimize risk, shared effort, quick prototyping
  - ❑ Respect users' need and their intent
  - ❑ Respect opportunities and opponents

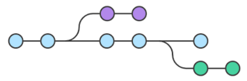
Google

"Don't be evil"

Never ignore the importance of respect even though the project is free

Things to Know in Creating a Repository

- ❑ **A repository helps store and manage code with**
  - ❑ Git version control (branch capabilities)
  - ❑ Cloud-based service (GitHub, Bitbucket, GitLab)
  - ❑ Issue tracker, open forum, contribution environment
- ❑ **Name your project wisely**
  - ❑ Precise, specific, no jargon
  - ❑ Keep the name to be 7-10 words
- ❑ **Tag your project to the right search categories**
  - ❑ Language, functionality, algorithm, library
- ❑ **Attach a proper license to your project**
  - ❑ MIT, BSD, Apache, GPL, etc.



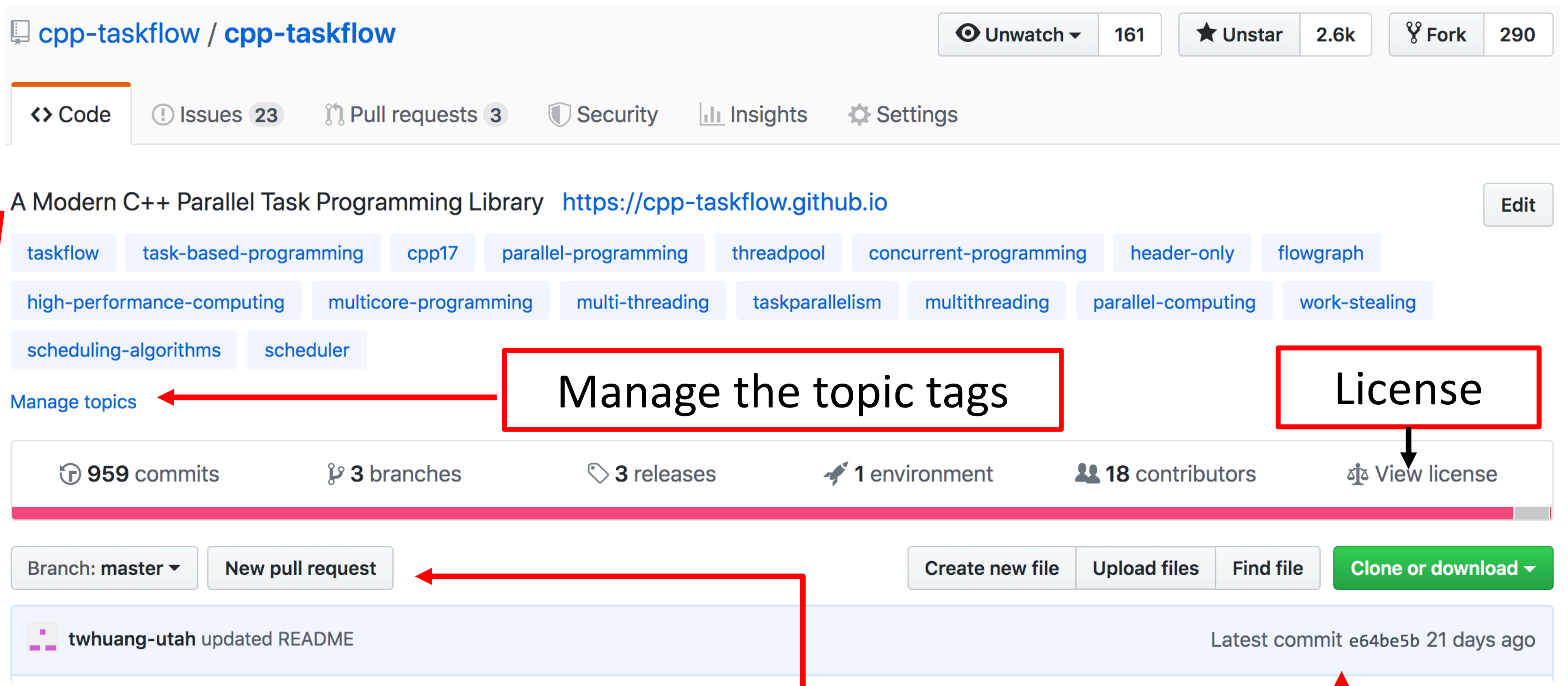
GitHub

GitLab

Bitbucket

open source initiative **Approved License**®

Example: Cpp-Taskflow's Front Page



cpp-taskflow / cpp-taskflow

Unwatch 161 | Unstar 2.6k | Fork 290

<> Code | Issues 23 | Pull requests 3 | Security | Insights | Settings

A Modern C++ Parallel Task Programming Library https://cpp-taskflow.github.io

Edit

taskflow task-based-programming cpp17 parallel-programming threadpool concurrent-programming header-only flowgraph high-performance-computing multicore-programming multi-threading taskparallelism multithreading parallel-computing work-stealing scheduling-algorithms scheduler

Manage topics

959 commits | 3 branches | 3 releases | 1 environment | 18 contributors | View license

Branch: master | New pull request | Create new file | Upload files | Find file | Clone or download

twhuang-utah updated README — Latest commit e64be5b 21 days ago

Manage the topic tags

License

Default branch (master)

Project name
(6 specific words)

Project activities
(keep your project active by new commits every few days)

Similar ideas apply to other platforms (GitLab, Bitbucket) as well.

Cpp-Taskflow: https://github.com/cpp-taskflow/cpp-taskflow

Comparison of Popular Licenses

Terms and Use		GNU GPLv3	Apache License 2	MIT License
Permissions	● Commercial use	✓	✓	✓
	● Distribution	✓	✓	✓
	● Modification	✓	✓	✓
	● Patent use	✓	✓	
	● Private use	✓	✓	✓
Conditions	● Disclose source	✓		
	● License & copyright	✓	✓	✓
	● Same license	✓		
	● State changes	✓	✓	
Limitations	● Liability	✓	✓	✓
	● Trademark use		✓	
	● Warranty	✓	✓	✓

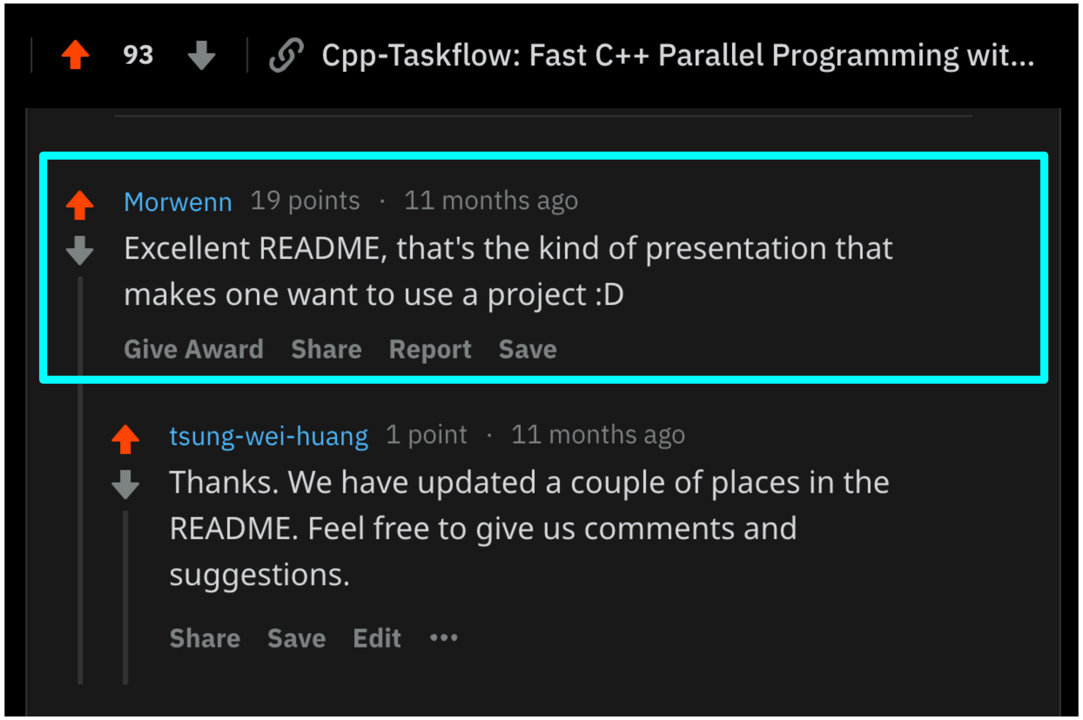
Do NOT ever create your own open-source licenses; always use existing licenses

Open-source License: https://choosealicense.com/licenses/

Understand the targeted users of your open-source projects, attach a proper license, and name your project concisely

Prepare for an “Effective” README

❑ The **most important** component in your project



93 Cpp-Taskflow: Fast C++ Parallel Programming wit...

Morwenn 19 points · 11 months ago
Excellent README, that's the kind of presentation that makes one want to use a project :D
Give Award Share Report Save

tsung-wei-huang 1 point · 11 months ago
Thanks. We have updated a couple of places in the README. Feel free to give us comments and suggestions.
Share Save Edit •••

Keep in mind thousands of projects are being created everyday; the majority people glance and leave.

HOOK YOUR USER

Points to take care:
- What/Why/Where
- Code example
- Installation guide
- System environment
- Doc & API reference
- Reward contributors

Cpp-Taskflow

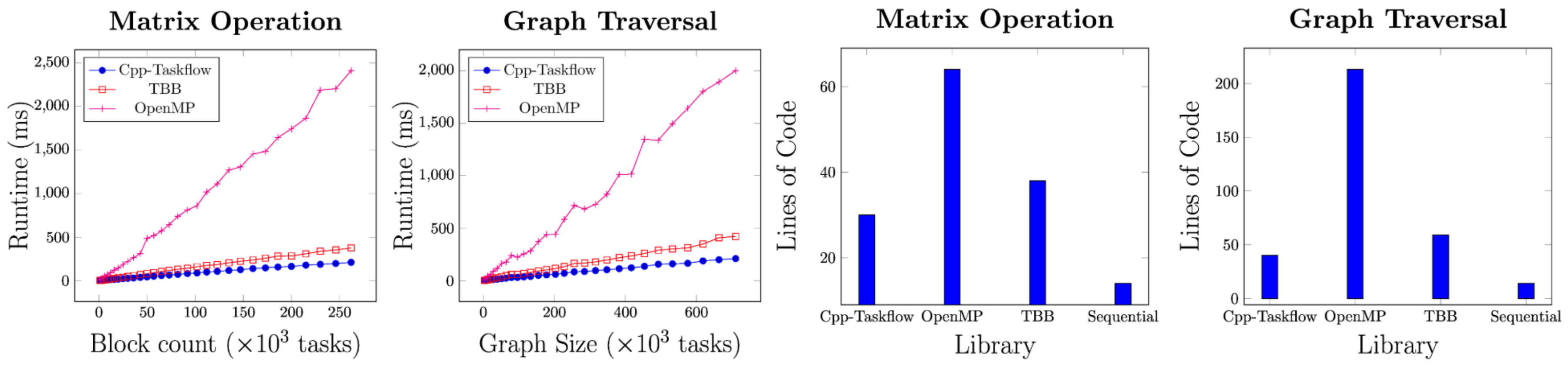
code quality A | build passing | build passing | c++ 17 | download latest | API docs | cite IPDPS

Cpp-Taskflow

A fast C++ *header-only* library to help you quickly write parallel programs with complex task dependencies

Why Cpp-Taskflow?

Cpp-Taskflow is by far faster, more expressive, and easier for drop-in integration than existing parallel task programming libraries such as OpenMP Tasking and Intel TBB FlowGraph in handling complex parallel workloads.



Cpp-Taskflow lets you quickly implement task decomposition strategies that incorporate both regular and irregular compute patterns, together with an efficient *work-stealing* scheduler to optimize your multithreaded performance.

Github: https://github.com/cpp-taskflow/cpp-taskflow

Document your Project

- ❑ **As important as other development facets**
  - ❑ Reminds you of what you code
  - ❑ Reduce users' time spent on understanding your code
- ❑ **But… what is the problem?**
  - ❑ The main reason code goes undocumented is time
  - ❑ Code abstraction happens before documentation

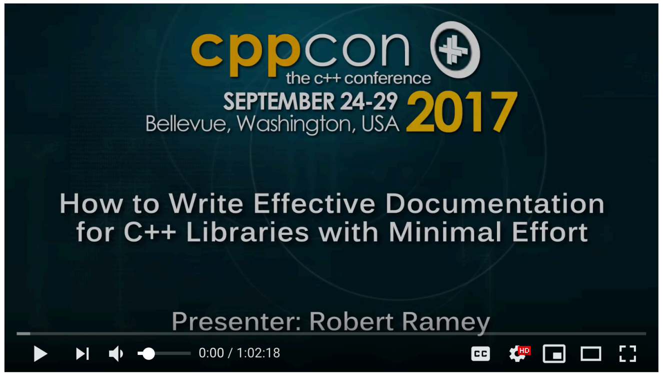
"An incredible 93% of people reported being frustrated with incomplete or confusing documentation," Robert Ramey

- ❑ **A suggested solution**
  - ❑ Craft code and documentation together (e.g., Doxygen)

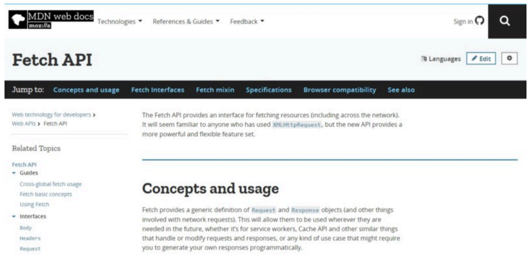
"If you spent 6 hours on writing code, spend at least another 6 hours on documenting your code," C++ Conference Keynote

Resources to Document Your Code

- ❑ **Good code does need good documentation**
  - ❑ Never forego the need of doc
- ❑ **Some popular examples**
  - ❑ MDN
  - ❑ Dijango
  - ❑ Stripe
  - ❑ Doxygen
- ❑ **My personal taste**
  - ❑ C++ reference
  - ❑ Boost documentation


cppcon the c++ conference
SEPTEMBER 24-29 2017
Bellevue, Washington, USA
How to Write Effective Documentation for C++ Libraries with Minimal Effort
Presenter: Robert Ramey
0:00 / 1:02:18

CppCon 2017: Robert Ramey “How to Write Effective Documentation for C++ Libraries...”



Fetch API

Jump to: Concepts and usage | Fetch Interfaces | Fetch mixin | Specifications | Browser compatibility | See also

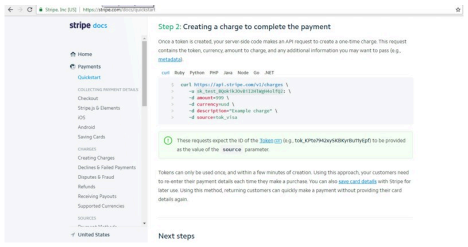
Web technology for developers › Web APIs › Fetch API

The Fetch API provides an interface for fetching resources (including across the network). It will seem familiar to anyone who has used `XMLHttpRequest`, but the new API provides a more powerful and flexible feature set.

Related Topics
Fetch API
- Guides
  - Cross-global fetch usage
  - Fetch basic concepts
  - Using Fetch
- Interfaces
  - Body
  - Headers
  - Request

Concepts and usage

Fetch provides a generic definition of `Request` and `Response` objects (and other things involved with network requests). This will allow them to be used wherever they are needed in the future, whether it's for service workers, Cache API and other similar things that handle or modify requests and responses, or any kind of use case that might require you to generate your own responses programmatically.



Step 2: Creating a charge to complete the payment

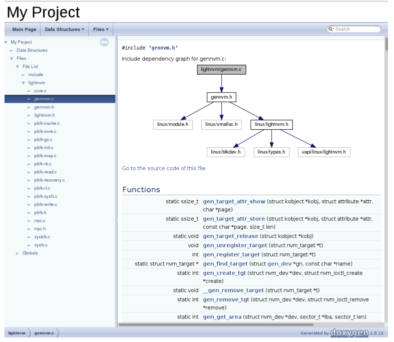
Once a token is created, your server-side code makes an API request to create a one-time charge. This request contains the token, currency, amount to charge, and any additional information you may want to pass (e.g., metadata).

```
$ curl https://api.stripe.com/v1/charges \
>    -u sk_test_BQokikJOvBiI2HlWgH4olfQ2: \
>    -d amount=999 \
>    -d currency=usd \
>    -d description="Example charge" \
>    -d source=tok_visa
```

These requests expect the ID of the Token (e.g., tok_KPte7942xySKBKyrBu11yEpf) to be provided as the value of the `source` parameter.

Tokens can only be used once, and within a few minutes of creation. Using this approach, your customers need to re-enter their payment details each time they make a purchase. You can also save card details with Stripe for later use. Using this method, returning customers can quickly make a payment without providing their card details again.

Next steps



```
#include "gennvm.h"
```

Include dependency graph for gennvm.c:

Go to the source code of this file.

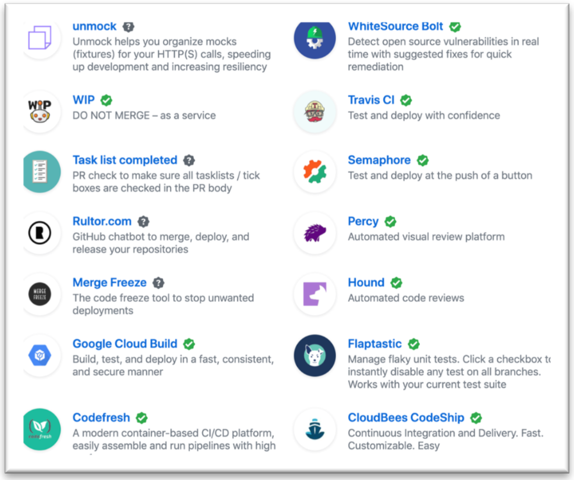
Functions

static ssize_t	gen_target_attr_show (struct kobject *kobj, struct attribute *attr, char *page)
static ssize_t	gen_target_attr_store (struct kobject *kobj, struct attribute *attr, const char *page, size_t len)
static void	gen_target_release (struct kobject *kobj)
void	gen_unregister_target (struct nvm_target *t)
int	gen_register_target (struct nvm_target *t)
static struct nvm_target *	gen_find_target (struct gen_dev *gn, const char *name)
static int	gen_create_tgt (struct nvm_dev *dev, struct nvm_ioctl_create *create)
static void	__gen_remove_target (struct nvm_target *t)
static int	gen_remove_tgt (struct nvm_dev *dev, struct nvm_ioctl_remove *remove)
static int	gen_get_area (struct nvm_dev *dev, sector_t *lba, sector_t len)

“If you write good documentation, most likely you will write a good scientific paper,” my manager at Citadel

Grow your Project Community

- ❑ **Attract people to contribute**
  - ❑ Turn end-users to developers
  - ❑ Getting pull requests is not easy
  - ❑ Proof of your project creditability
- ❑ **A good contribution environment**
  - ❑ Template, code review, refactor
  - ❑ Continuous integration
    - Ensure each change doesn't break



unmock
Unmock helps you organize mocks (fixtures) for your HTTP(S) calls, speeding up development and increasing resiliency

WhiteSource Bolt ✓
Detect open source vulnerabilities in real time with suggested fixes for quick remediation

WIP ✓
DO NOT MERGE – as a service

Travis CI ✓
Test and deploy with confidence

Task list completed
PR check to make sure all tasklists / tick boxes are checked in the PR body

Semaphore ✓
Test and deploy at the push of a button

Rultor.com
GitHub chatbot to merge, deploy, and release your repositories

Percy ✓
Automated visual review platform

Merge Freeze
The code freeze tool to stop unwanted deployments

Hound ✓
Automated code reviews

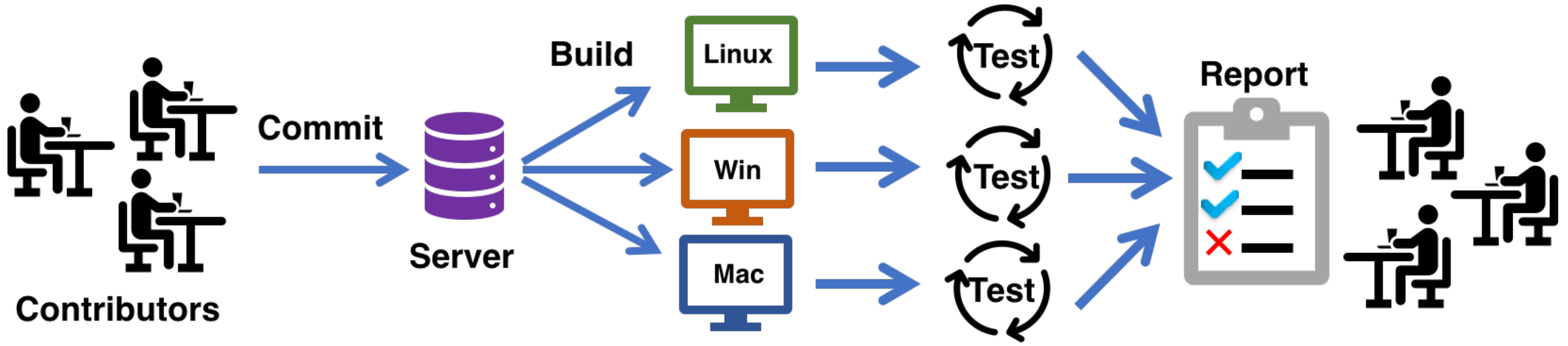
Google Cloud Build ✓
Build, test, and deploy in a fast, consistent, and secure manner

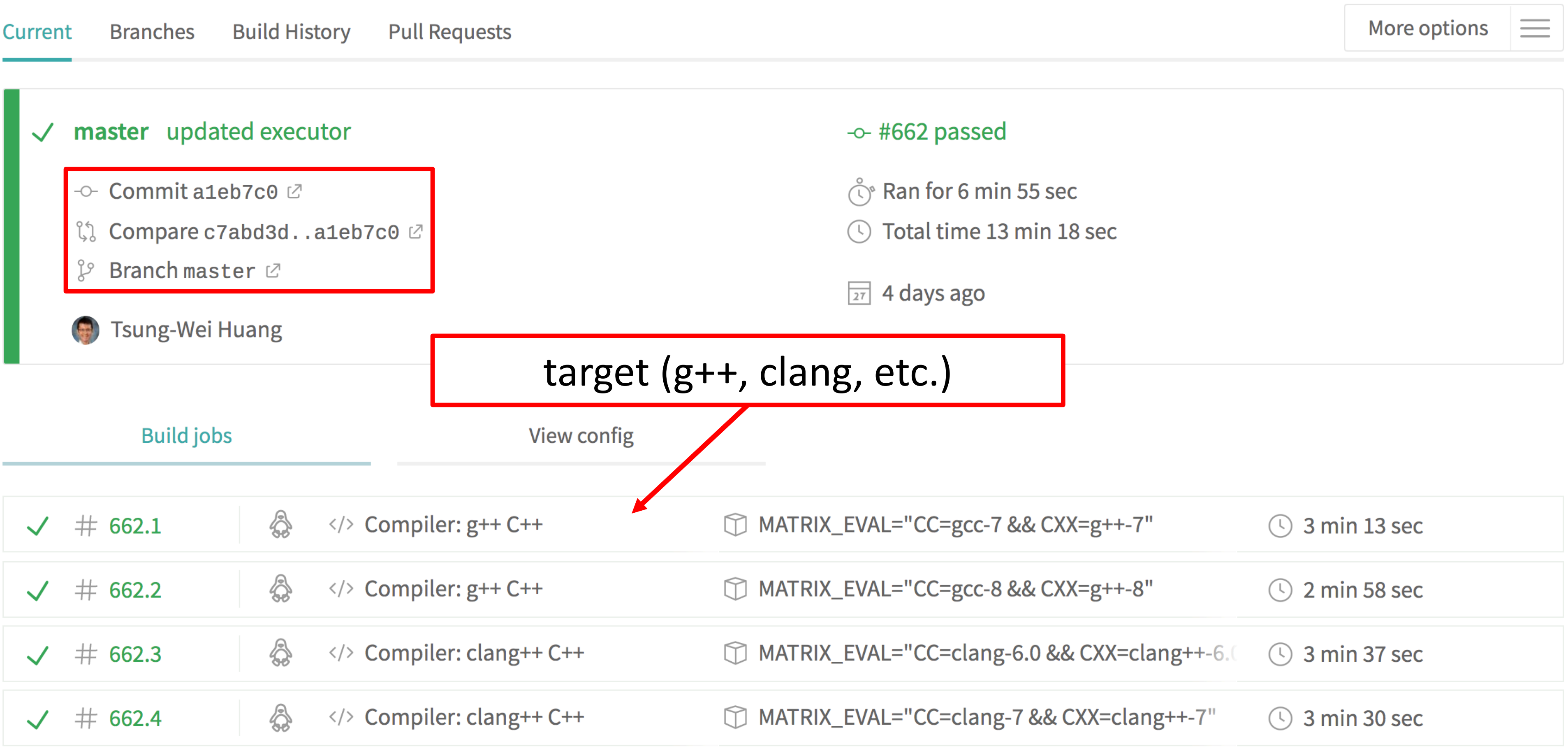
Flaptastic ✓
Manage flaky unit tests. Click a checkbox t instantly disable any test on all branches. Works with your current test suite

Codefresh ✓
A modern container-based CI/CD platform, easily assemble and run pipelines with high

CloudBees CodeShip ✓
Continuous Integration and Delivery. Fast. Customizable. Easy

Continuous integration tools





Current Branches Build History Pull Requests More options

✓ master updated executor

Commit a1eb7c0

Compare c7abd3d..a1eb7c0

Branch master

Tsung-Wei Huang

#662 passed

Ran for 6 min 55 sec

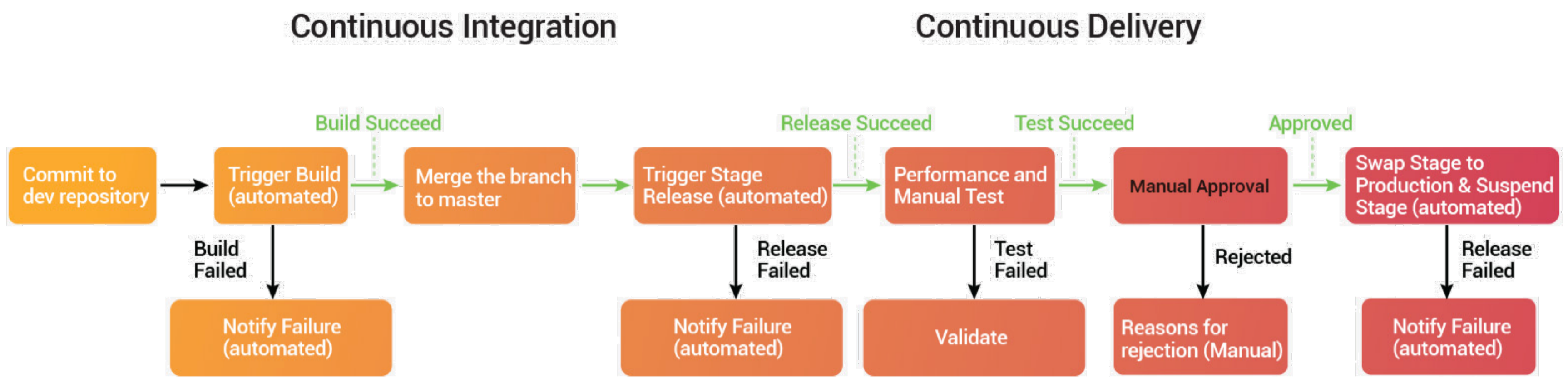
Total time 13 min 18 sec

4 days ago

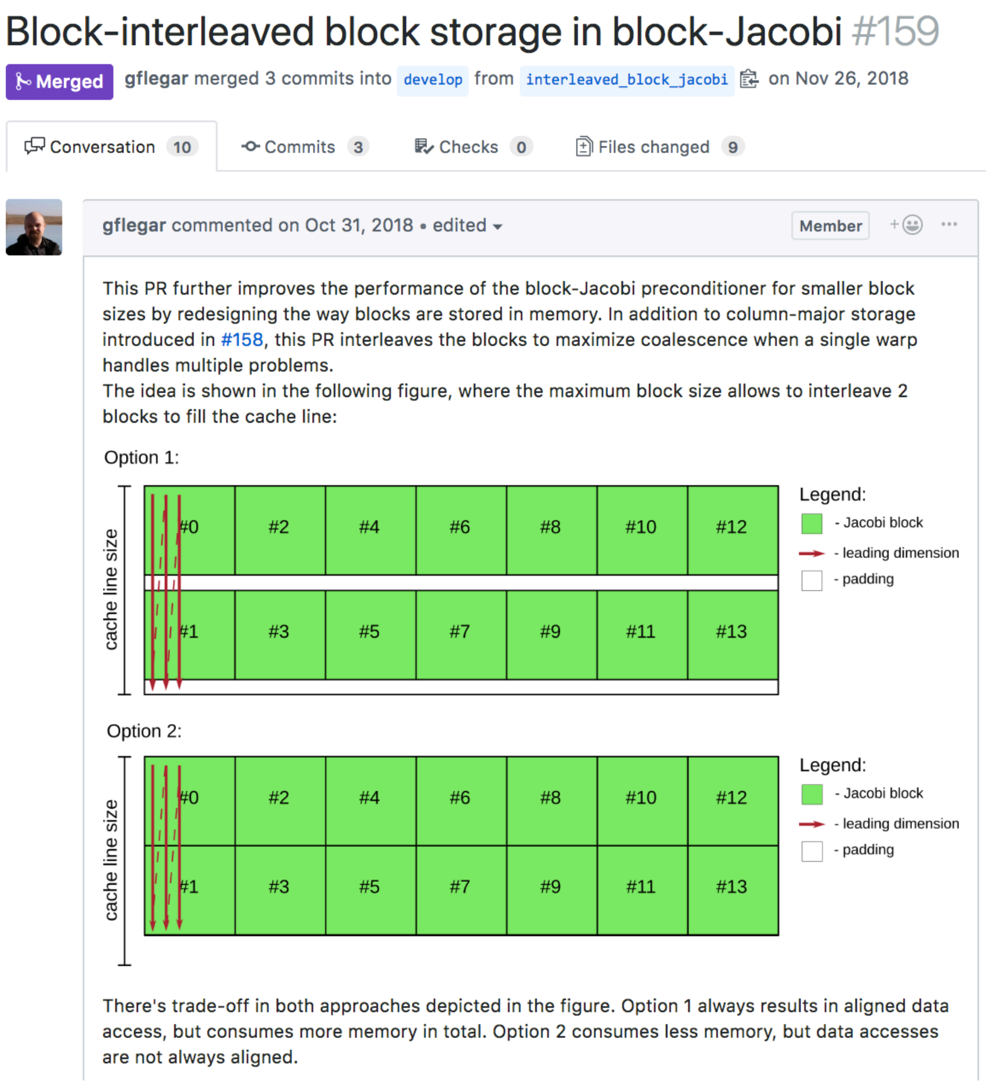
target (g++, clang, etc.)

Build jobs View config

✓	# 662.1	Compiler: g++ C++	MATRIX_EVAL="CC=gcc-7 && CXX=g++-7"	3 min 13 sec
✓	# 662.2	Compiler: g++ C++	MATRIX_EVAL="CC=gcc-8 && CXX=g++-8"	2 min 58 sec
✓	# 662.3	Compiler: clang++ C++	MATRIX_EVAL="CC=clang-6.0 && CXX=clang++-6.	3 min 37 sec
✓	# 662.4	Compiler: clang++ C++	MATRIX_EVAL="CC=clang-7 && CXX=clang++-7"	3 min 30 sec


Continuous Integration
Continuous Delivery
Commit to dev repository
Trigger Build (automated)
Build Succeed
Merge the branch to master
Trigger Stage Release (automated)
Release Succeed
Performance and Manual Test
Test Succeed
Manual Approval
Approved
Swap Stage to Production & Suspend Stage (automated)
Build Failed
Notify Failure (automated)
Release Failed
Notify Failure (automated)
Test Failed
Validate
Rejected
Reasons for rejection (Manual)
Release Failed
Notify Failure (automated)

Iterate Feedback Loop



Block-interleaved block storage in block-Jacobi #159

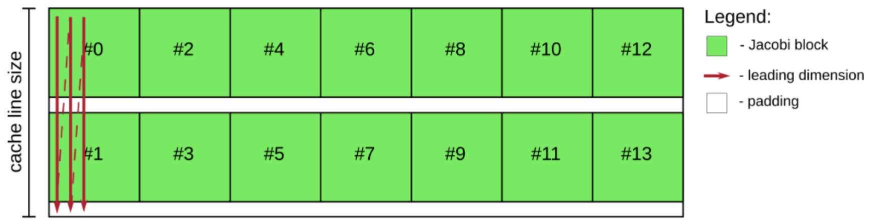
Merged gflegar merged 3 commits into `develop` from `interleaved_block_jacobi` on Nov 26, 2018

Conversation 10 · Commits 3 · Checks 0 · Files changed 9

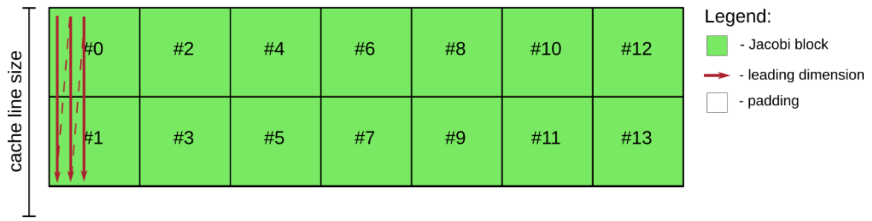
gflegar commented on Oct 31, 2018 • edited · Member

This PR further improves the performance of the block-Jacobi preconditioner for smaller block sizes by redesigning the way blocks are stored in memory. In addition to column-major storage introduced in #158, this PR interleaves the blocks to maximize coalescence when a single warp handles multiple problems.
The idea is shown in the following figure, where the maximum block size allows to interleave 2 blocks to fill the cache line:

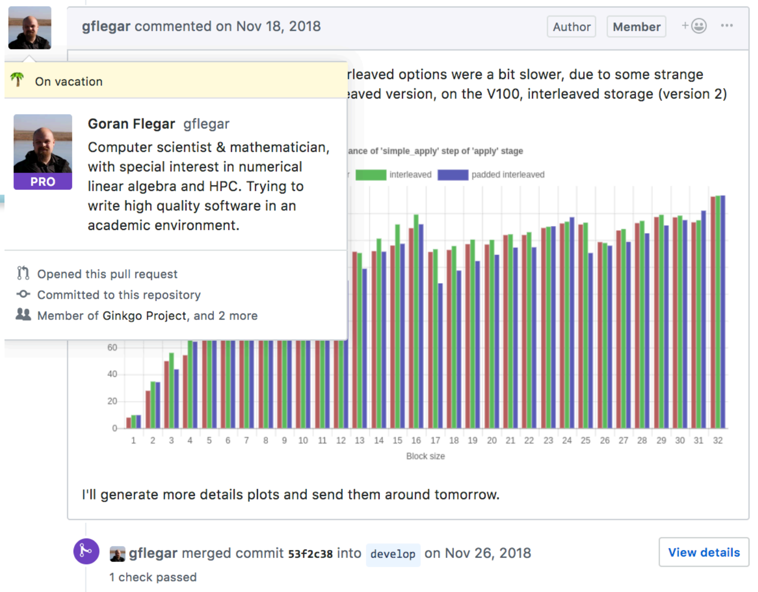
Option 1:



Option 2:



There's trade-off in both approaches depicted in the figure. Option 1 always results in aligned data access, but consumes more memory in total. Option 2 consumes less memory, but data accesses are not always aligned.



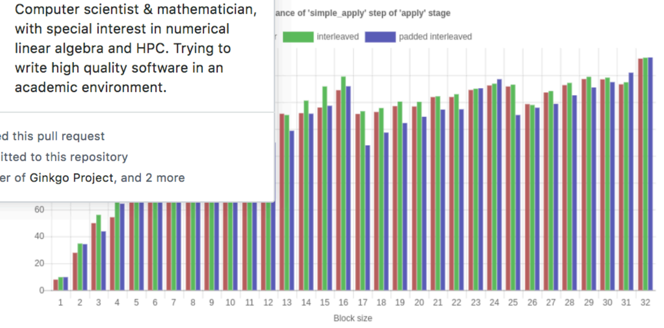
gflegar commented on Nov 18, 2018 · Author · Member

On vacation

Goran Flegar gflegar
Computer scientist & mathematician, with special interest in numerical linear algebra and HPC. Trying to write high quality software in an academic environment.

Opened this pull request
Committed to this repository
Member of Ginkgo Project, and 2 more

rleaved options were a bit slower, due to some strange
eaved version, on the V100, interleaved storage (version 2)



I'll generate more details plots and send them around tomorrow.

gflegar merged commit **53f2c38** into `develop` on Nov 26, 2018 · View details
1 check passed

A good software patch has
- Motivation
- Technical explanation
- Performance evaluation
- Rigorous code review
- Code refactoring
- Multiple feedback loops

Similar to the scientific journal contributions

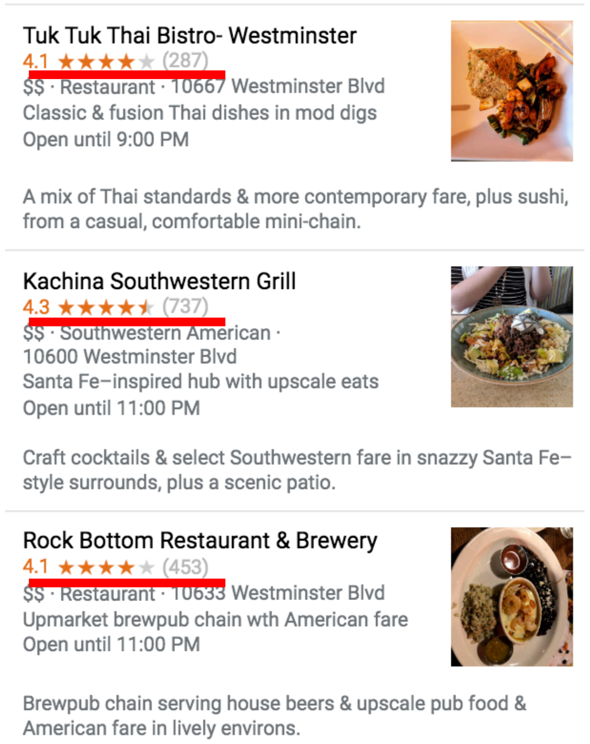
Ginkgo: https://github.com/ginkgo-project/ginkgo

Effective README and Documentation are key to engage people to use and contribute to your open-source projects

How to Attract Users?

- ❑ I was finding a place to eat …

Google review



Tuk Tuk Thai Bistro- Westminster
4.1 ★★★★★ (287)
$$ · Restaurant · 10667 Westminster Blvd
Classic & fusion Thai dishes in mod digs
Open until 9:00 PM

A mix of Thai standards & more contemporary fare, plus sushi, from a casual, comfortable mini-chain.

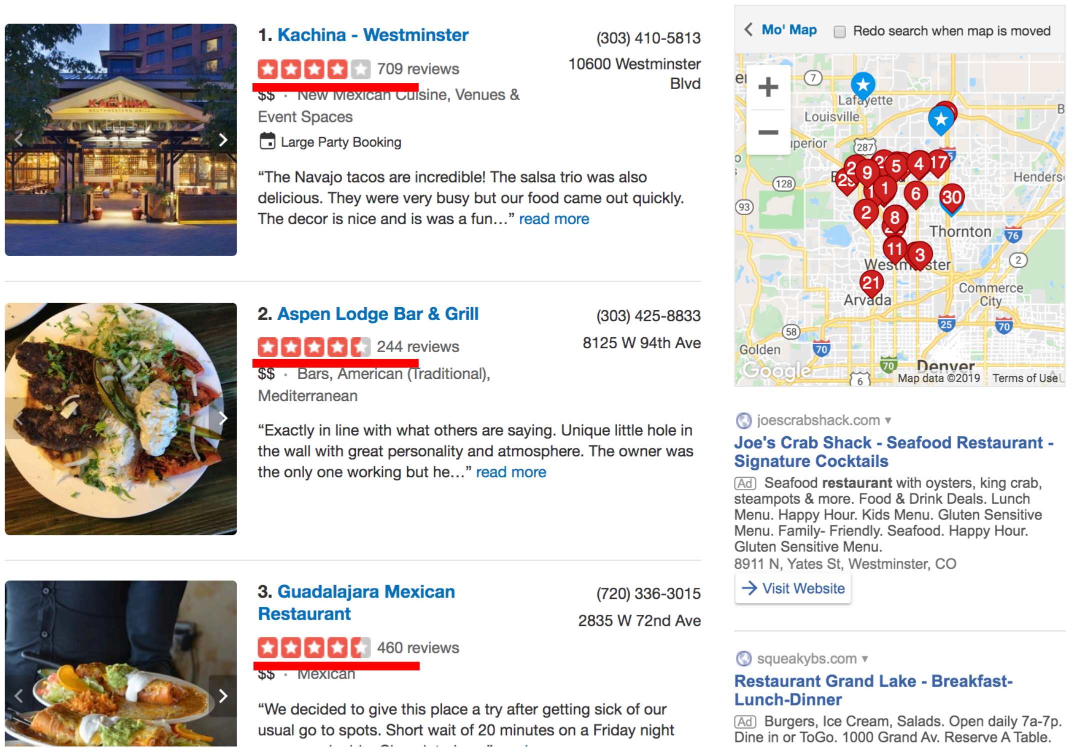
Kachina Southwestern Grill
4.3 ★★★★★ (737)
$$ · Southwestern American ·
10600 Westminster Blvd
Santa Fe–inspired hub with upscale eats
Open until 11:00 PM

Craft cocktails & select Southwestern fare in snazzy Santa Fe–style surrounds, plus a scenic patio.

Rock Bottom Restaurant & Brewery
4.1 ★★★★★ (453)
$$ · Restaurant · 10633 Westminster Blvd
Upmarket brewpub chain wth American fare
Open until 11:00 PM

Brewpub chain serving house beers & upscale pub food & American fare in lively environs.

Yelp rating



1. Kachina - Westminster
709 reviews
(303) 410-5813
10600 Westminster Blvd
$$ · New Mexican Cuisine, Venues & Event Spaces
Large Party Booking

"The Navajo tacos are incredible! The salsa trio was also delicious. They were very busy but our food came out quickly. The decor is nice and is was a fun…" read more

2. Aspen Lodge Bar & Grill
244 reviews
(303) 425-8833
8125 W 94th Ave
$$ · Bars, American (Traditional), Mediterranean

"Exactly in line with what others are saying. Unique little hole in the wall with great personality and atmosphere. The owner was the only one working but he…" read more

3. Guadalajara Mexican Restaurant
460 reviews
(720) 336-3015
2835 W 72nd Ave
$$ · Mexican

"We decided to give this place a try after getting sick of our usual go to spots. Short wait of 20 minutes on a Friday night

joescrabshack.com
Joe's Crab Shack - Seafood Restaurant - Signature Cocktails
Ad Seafood **restaurant** with oysters, king crab, steampots & more. Food & Drink Deals. Lunch Menu. Happy Hour. Kids Menu. Gluten Sensitive Menu. Family- Friendly. Seafood. Happy Hour. Gluten Sensitive Menu.
8911 N, Yates St, Westminster, CO
Visit Website

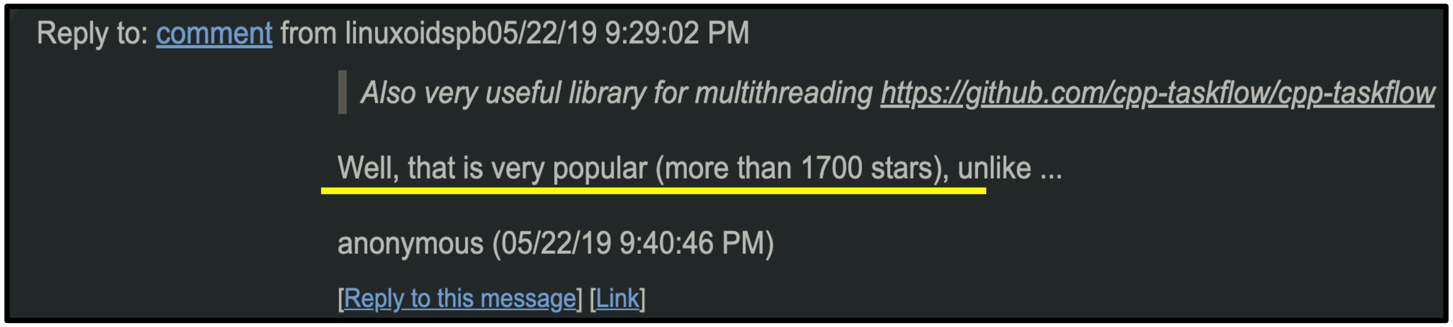
squeakybs.com
Restaurant Grand Lake - Breakfast-Lunch-Dinner
Ad Burgers, Ice Cream, Salads. Open daily 7a-7p. Dine in or ToGo. 1000 Grand Av. Reserve A Table.

"Too many places… Where do we go?"
"Let's go to the one with the highest star in the rating app!"

Advertise Your Project

No one knows you until you let others know …

- ❑ **Many users use your project because of stars** Star
  - ❑ Stars are the popularity and credibility of your project
  - ❑ Stars are an indicator of the number of potential users


Reply to: comment from linuxoidspb05/22/19 9:29:02 PM

Also very useful library for multithreading https://github.com/cpp-taskflow/cpp-taskflow

Well, that is very popular (more than 1700 stars), unlike ...

anonymous (05/22/19 9:40:46 PM)

[Reply to this message] [Link]

linux.org.ur: https://www.linux.org.ru/news/development/15005663

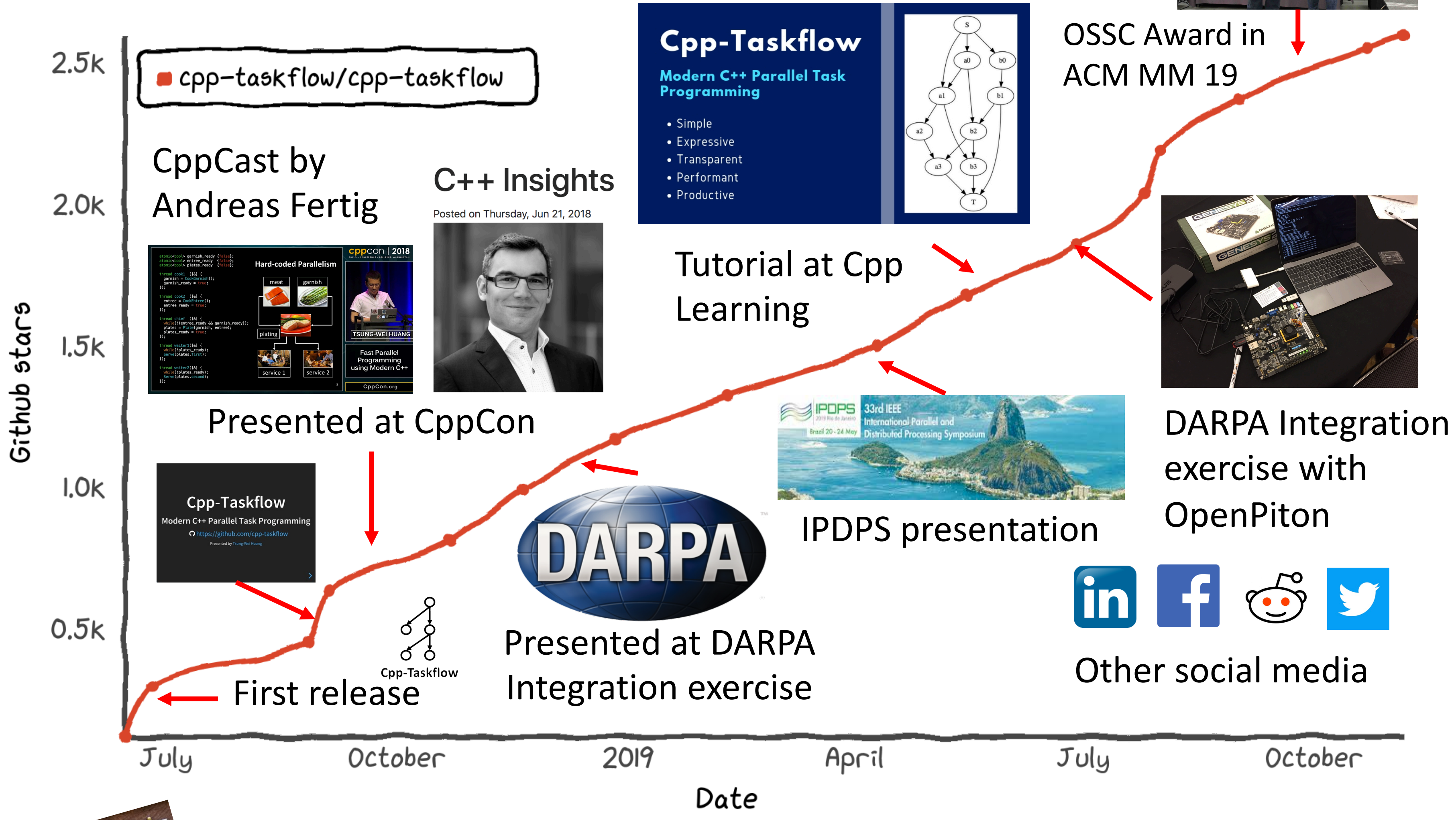
- ❑ **If you have a tasty cake, make it look tasty**
  - ❑ Add logo and badges to your README

Cpp-Taskflow

code quality A | build passing | build passing | c++ 17 | download latest | API documentation | awesome resources

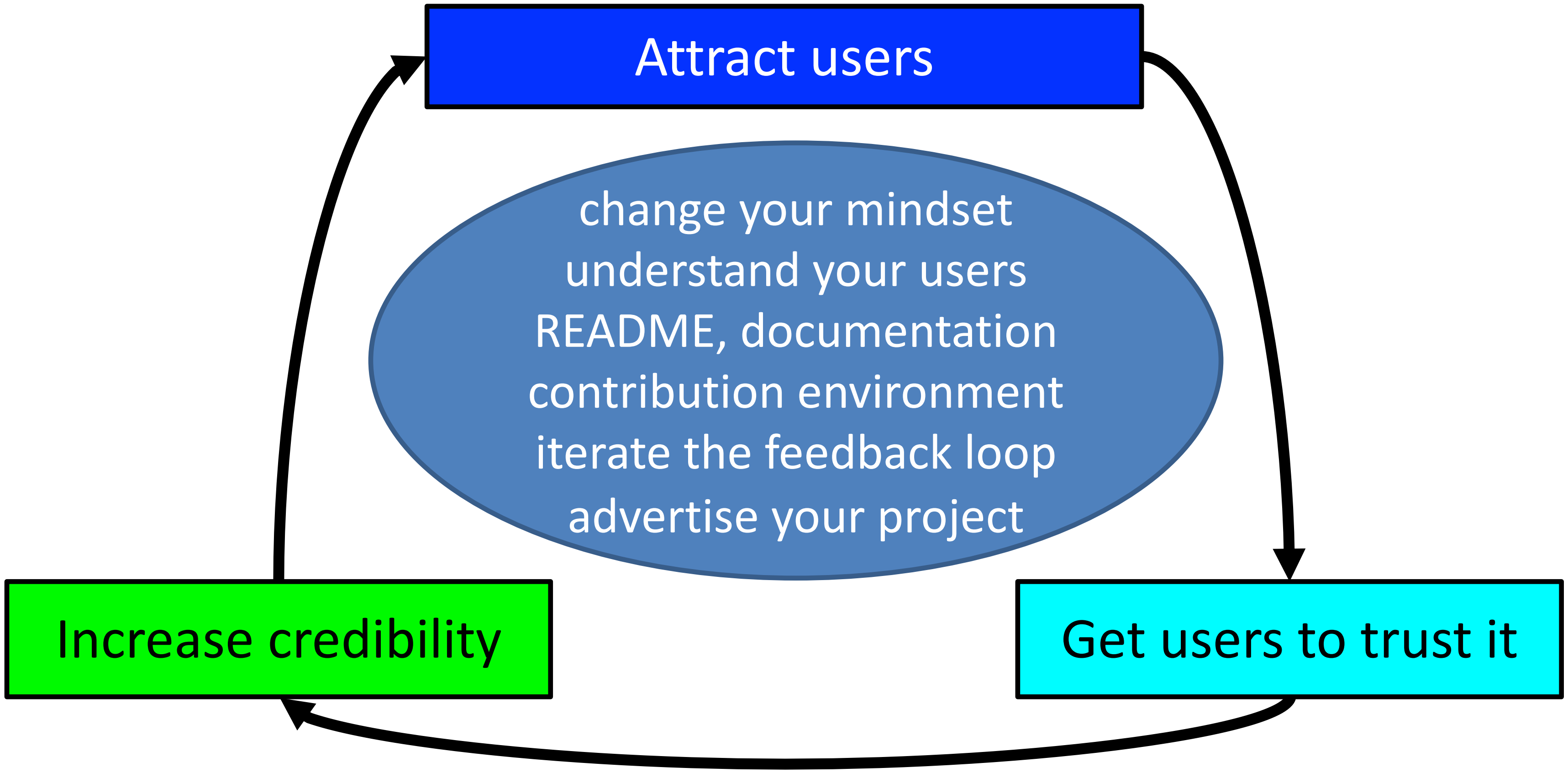
- ❑ **Advertise the project** **multiple times** **for each release**

Cpp-Taskflow's Star History


cpp-taskflow/cpp-taskflow
Github stars
2.5k
2.0k
1.5k
1.0k
0.5k
July
October
2019
April
July
October
Date
OSSC Award in ACM MM 19
CppCast by Andreas Fertig
C++ Insights
Posted on Thursday, Jun 21, 2018
Cpp-Taskflow
Modern C++ Parallel Task Programming
Simple
Expressive
Transparent
Performant
Productive
Tutorial at Cpp Learning
Hard-coded Parallelism
cppcon | 2018
TSUNG-WEI HUANG
Fast Parallel Programming using Modern C++
Presented at CppCon
Cpp-Taskflow
Modern C++ Parallel Task Programming
https://github.com/cpp-taskflow
First release
Cpp-Taskflow
DARPA
Presented at DARPA Integration exercise
IPDPS
33rd IEEE International Parallel and Distributed Processing Symposium
IPDPS presentation
DARPA Integration exercise with OpenPiton
Other social media

Advertising your project is important, but keep in mind it is your project content that makes people use it and like it

Conclusion: The Final Iron Circle



We should work together to change the current crediting system to reward software engineering & scientific software engineers

Thank You (and all our Users) ☺

GitHub: https://github.com/tsung-wei-huang

Twitter: https://twitter.com/twh760812

Website: https://tsung-wei-huang.github.io/