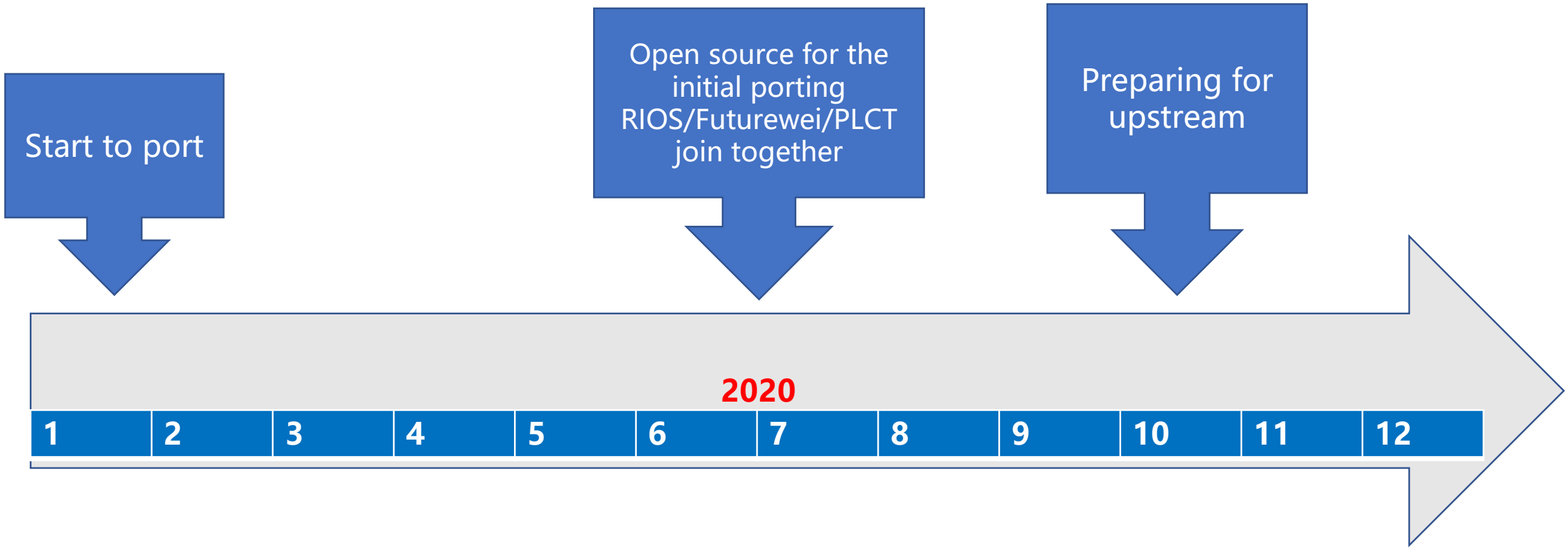
V8 for RISC-V: Working Toward a Tier-1 Platform

PLCT V8 Team
yahan@iscas.ac.cn

Retrospect： V8 for RISC-V in 2020



The porting process can be referred to:
1. Unlocking JavaScript V8 on RISC-V by Peng Wu, https://www.youtube.com/watch?v=MdHL9yGQer8
2. V8 for RISCV, where to com and where to go by PLCT lab, https://github.com/isrc-cas/PLCT-OpenDay-2020/blob/main/V8forRV-PLCTOD2020.pdf

Retrospect: V8 for RISC-V in 2020

Contents for today’ s presentation

- upstreaming process
- daily maintain after upstream
- node.js status
- feedback from tuning V8 to the ISA
- future work

The upstream process

Contribute your code to v8

- Get source code:

  1.git clone https://chromium.googlesource.com/chromium/tools/depot_tools.git

  2.export PATH=/path/to/depot_tools:$PATH

  3. mkdir v8 & cd v8

  3. fetch v8 & gclient syn

- Sign the Google Individu
- Init git setup for chroiu

  visit https://chromium

The upstream process

Contribute your code to v8

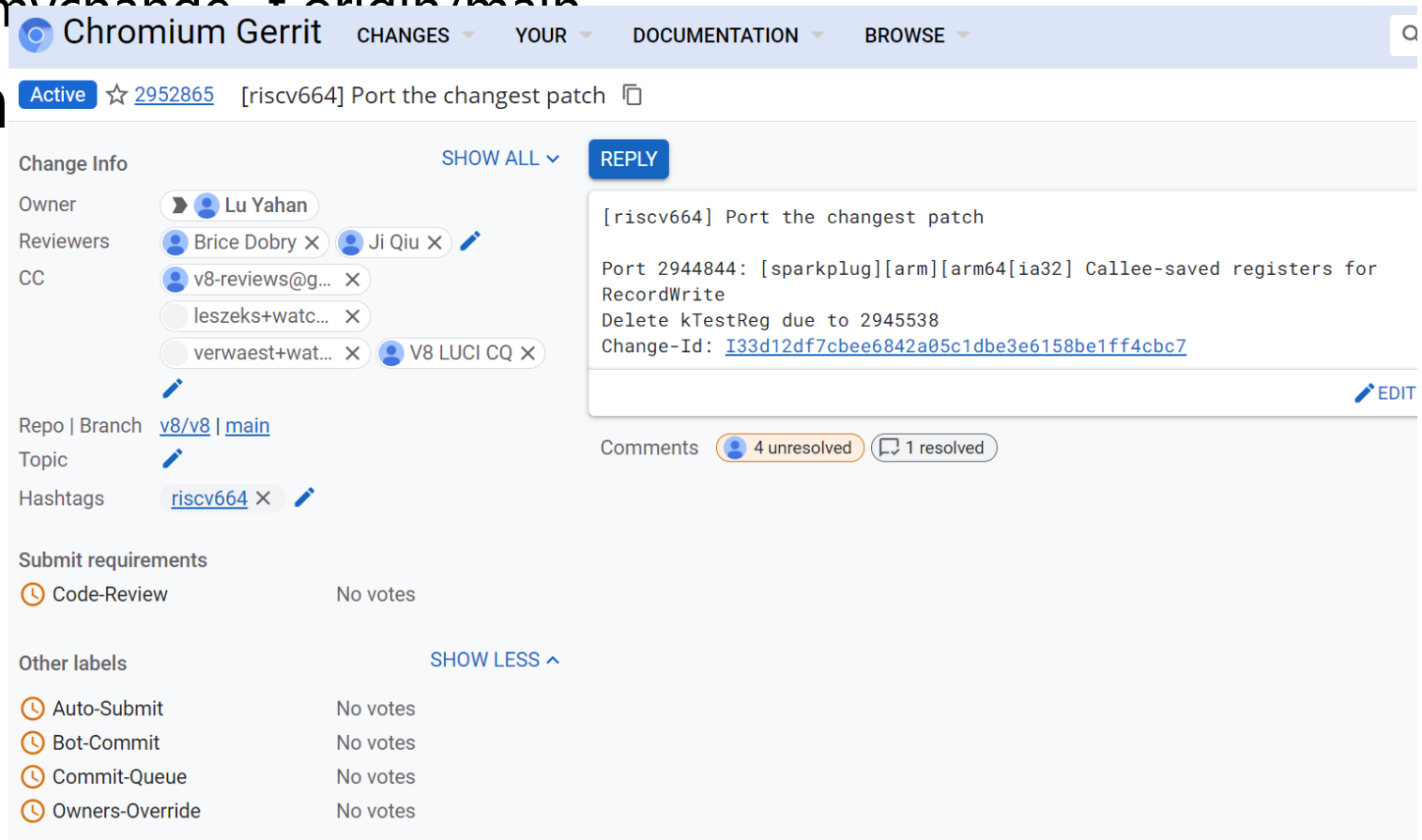
- Checkout your branch

  git checkout -b mychange -t origin/main

- Upload your cl(ch

  git cl upload

- Review your code



The upstream process

Review status

- WIP(work in process)
- Active
  - If you are a commiter, you can add flag CQ(Commit-Queue)+1 to trigger test.
  - OWNERS review code.

    If your code is good for owner, owner will reply LGTM and CQ+2 and trigger test again.

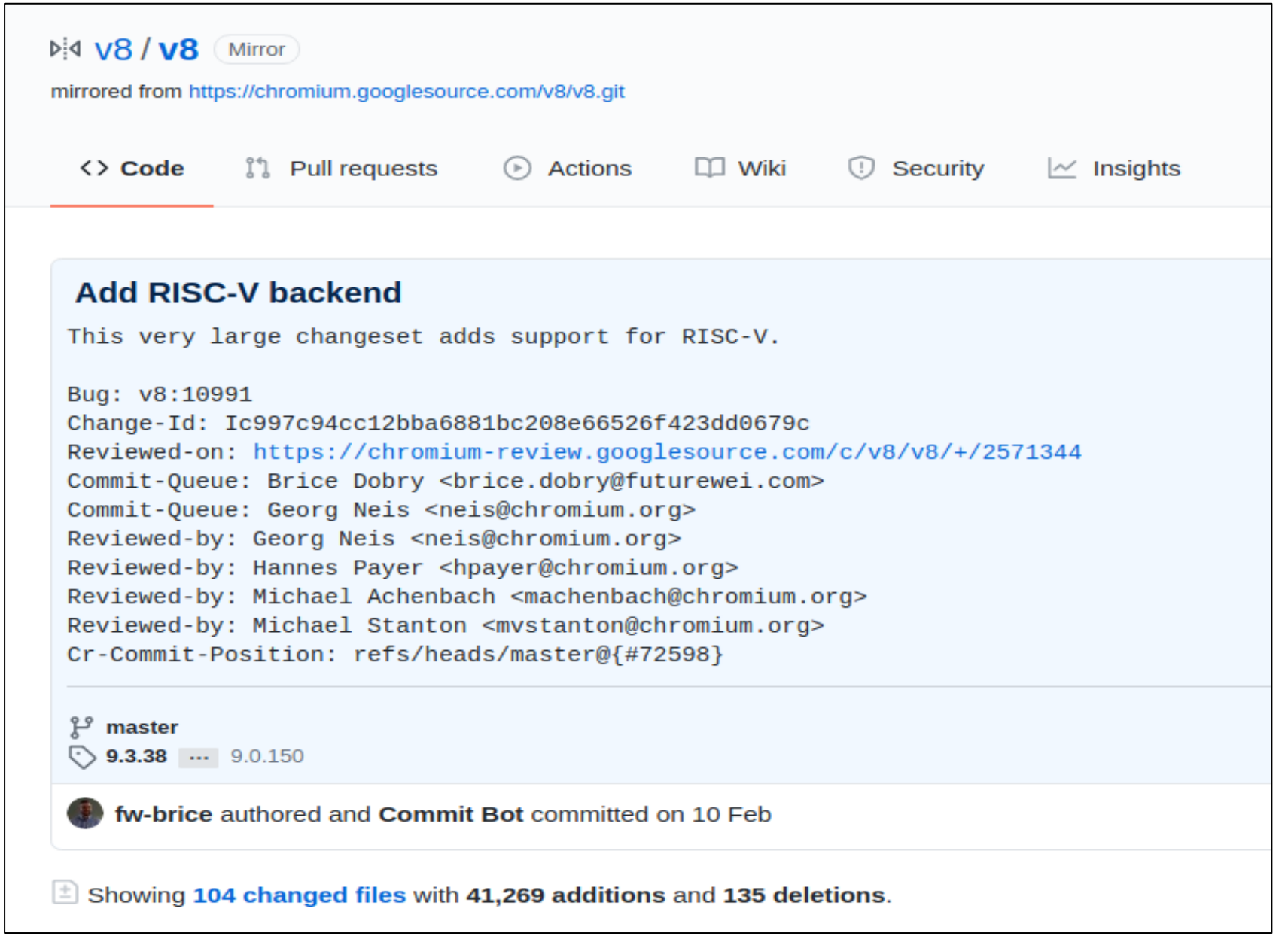
    If all test all green, your code will be merge.
- Merged

  The cl had been merged into upstream
- Abanded

  The cl had been rejected.

The upstream process

- at first: downstream maintain on GitHub
  - https://github.com/riscv/v8
- launch upstream at 2020 October
- many rounds of rebase and bug fix
- skipped several test cases for the regression routine
- finished at 2021 February 10



https://github.com/v8/v8/commit/ffd9e82dd5dd314bad48ab602517bd378b6d2184

Daily maintain after upstream

- latest changes port
  - for passing build
  - for new features
- ISA and ABI related configurations and fix
- bug fix
- arch-related optimization

Daily maintain after upstream

- latest changes port
  - for passing build
  - for new features

1. Port the latest changes to pass build
   1. 2690182[*]
   2. 2914246
   3. 2894036
2. Port the latest changes to introduce new feature
   1. OSR shadow stack (2831171)
   2. add static interface descriptors (2834632)
   3. The newly introduced compiler Sparkplug port (2763963)
   4. use EnterFrame and LeaveFrame with StackFrame (2848735)
   5. wasm atomic implementation (2881494)

[*] This is the change number in the Chromium’ s Gerrit system.
e.g., https://chromium-review.googlesource.com/c/v8/v8/+/2690182

Daily maintain after upstream

- latest changes port: (1) for passing build (2) for new feature
- ISA and ABI related configurations and fix
- bug f
- arch-

1. add toolchain configurations for RISC-V (2725484, 2754588)
2. make t6 as a call target register (2814726)
3. rename riscv flags (2878150)
4. C-ext configuration (2876854)
5. fix disassemble of illegal instruction (2874399)

Daily maintain after upstream

- latest changes port: (1) for passing build (2) for new feature
- ISA and ABI related configurations
- bug fix
- arch-relate

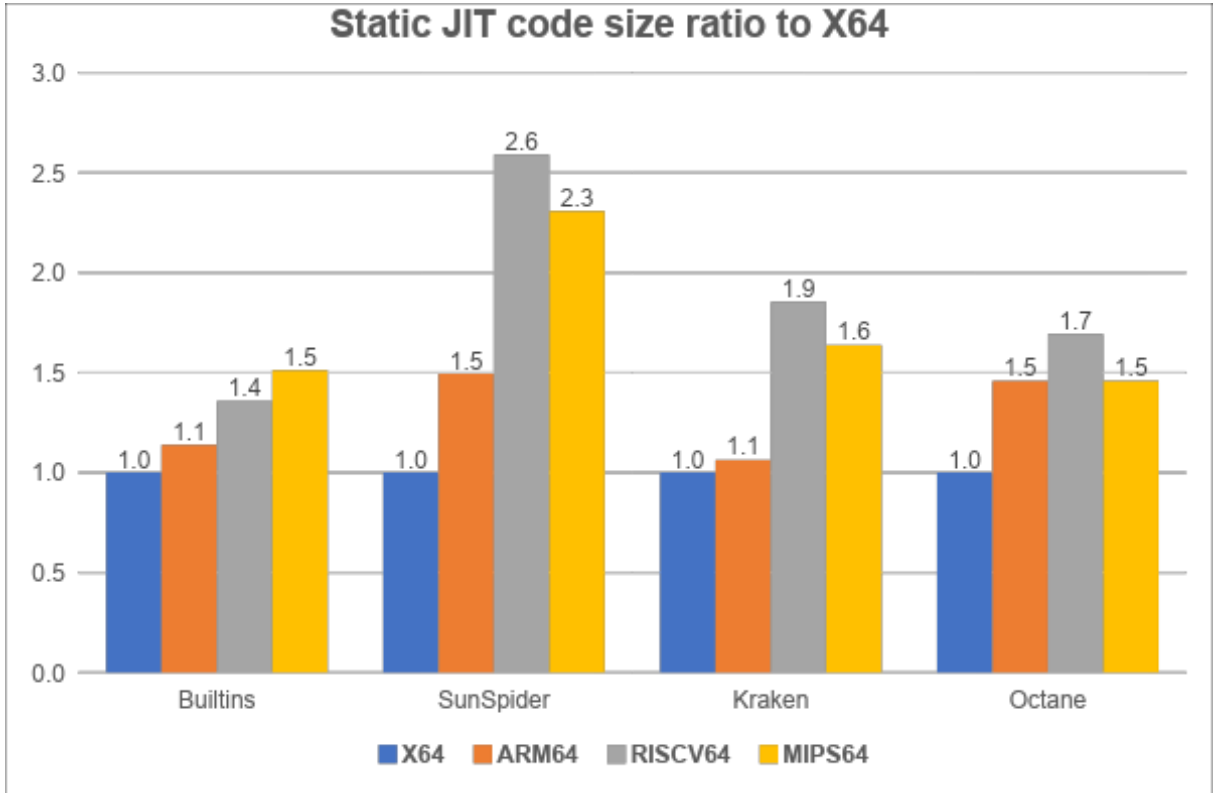
1. NaN processing issue (2814725)
2. Record correct offset in StoreTaggedPointer (2814723)
3. CPU profiler related bug (also help MIPS change the bug) (2902341)
4. typo fix (2917597)
5. unaligned mem access (2887023)
6. fix constant pool (2839546)
7. skip atomic test (2853282)
8. skip cctests (2881505)
9. skip inspector test (2894025)

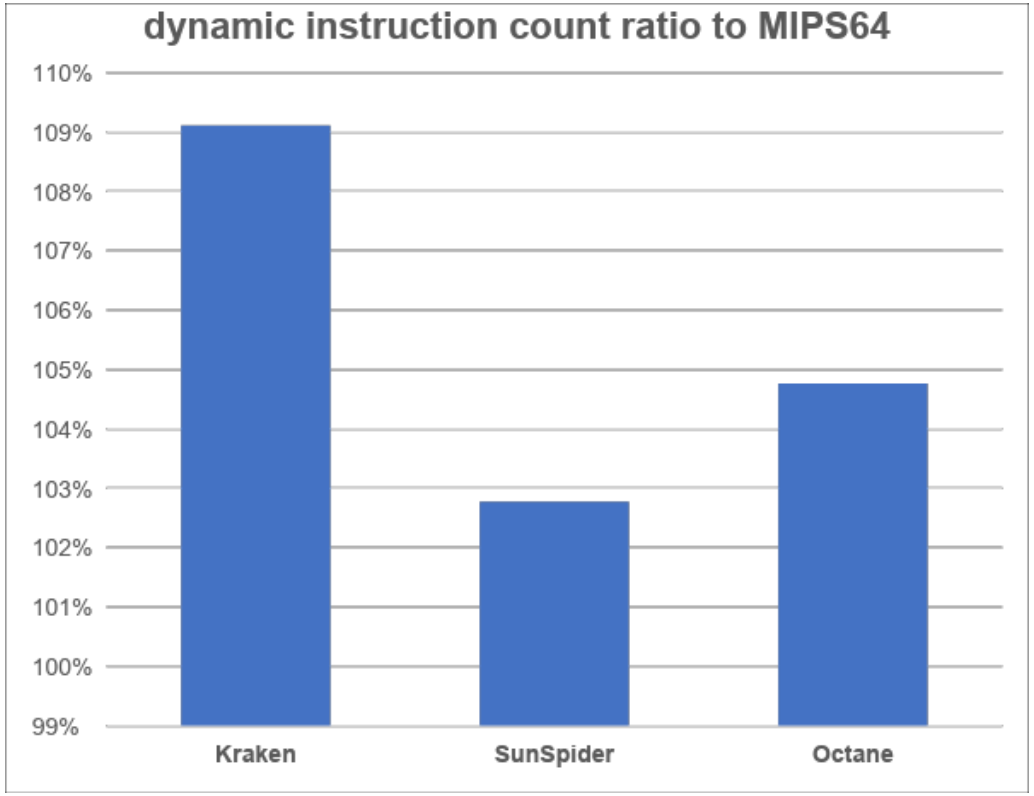
Daily maintain after upstream

- latest changes port: (1) for passing build (2) for new feature
- ISA and ABI related configurations
- bug fix
- arch-related optimization

1. Port PC-relative bultin-to-builtin calls (2814722)
2. enable constant pool (2814724)
3. remove of unnecessary not (2847673)
4. add/sub with immediate (2848732)
5. add RISCVCmpZero to gen bnez/beqz with zero_reg (2814562)
6. fold imm to the load/store offset (2814563)

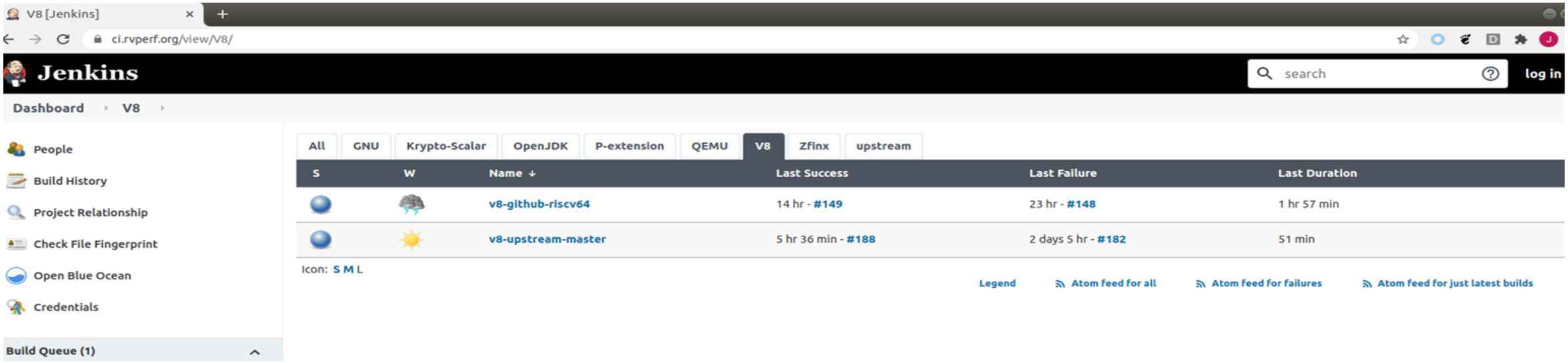
Current status-performance





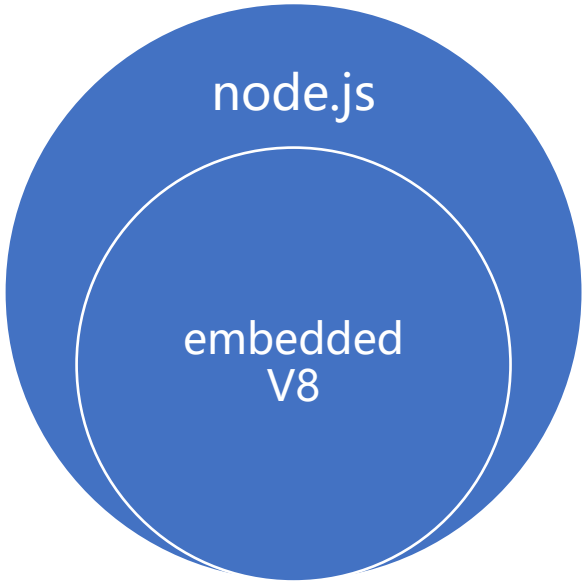
dynamic instruction count： collected from the V8 simulation run

Current status-stability and the CI system



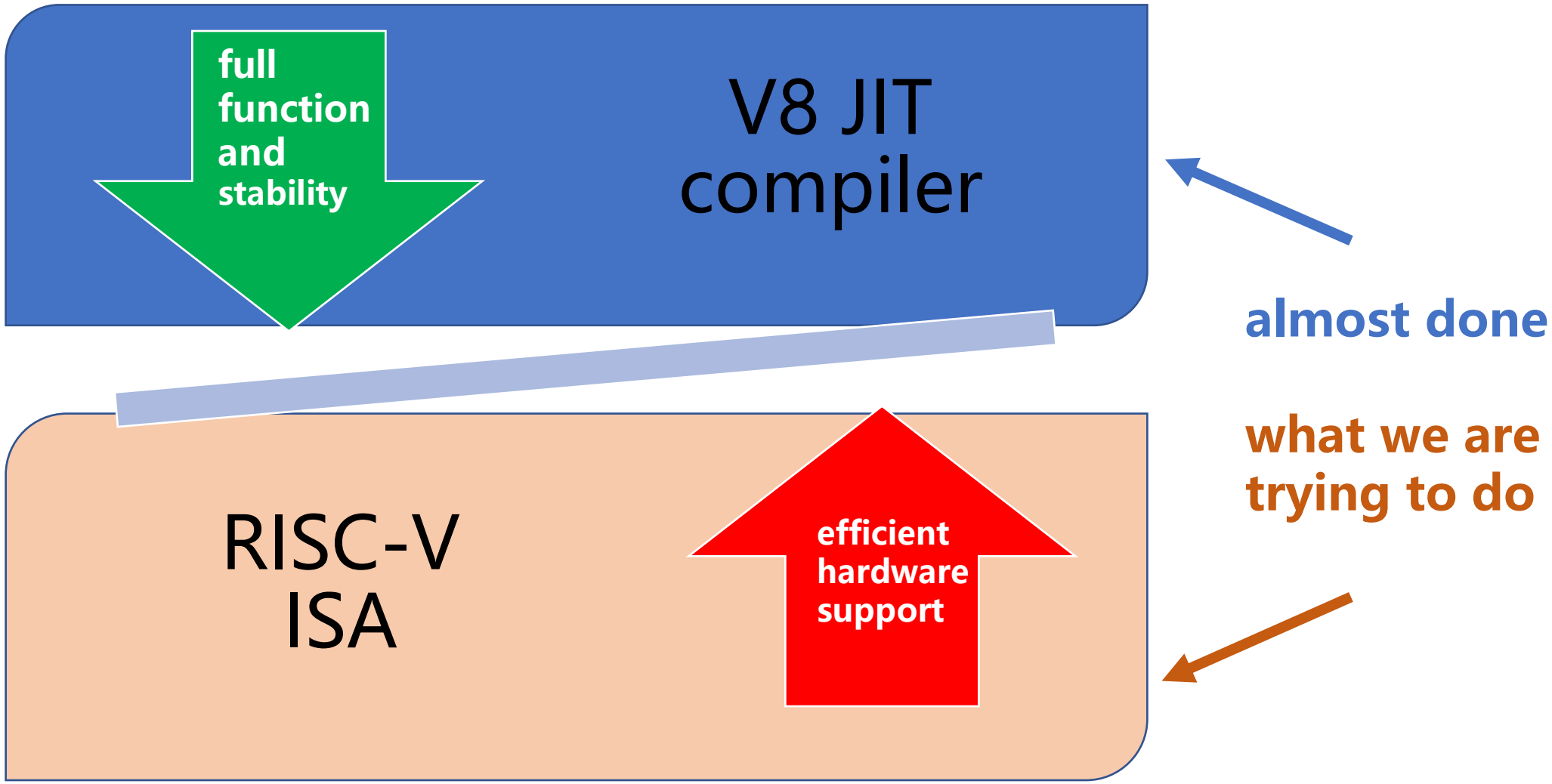
- no full CI on upstream
  - new commits may break the RISC-V target
- PLCT lab had set up a website for the RISC-V software
  - master branch: always keep up with upstream master
  - riscv64 branch: the latest commit that is good for RISC-V (pass all the regression test suit)

node.js support



- a downstream version was ported：
  - https://github.com/v8-riscv/node/tree/riscv64
- committed the RISC-V 64G ISA build configurations
  - https://github.com/nodejs/node/commit/1d21a8d140f78d4408c5375caedc917a83902a80
- node.is megered v8-9.1, node.js for riscv64 has been build pass

feedback from tuning V8 to the ISA



Feedback1 : li_ptr macro generates more instruction

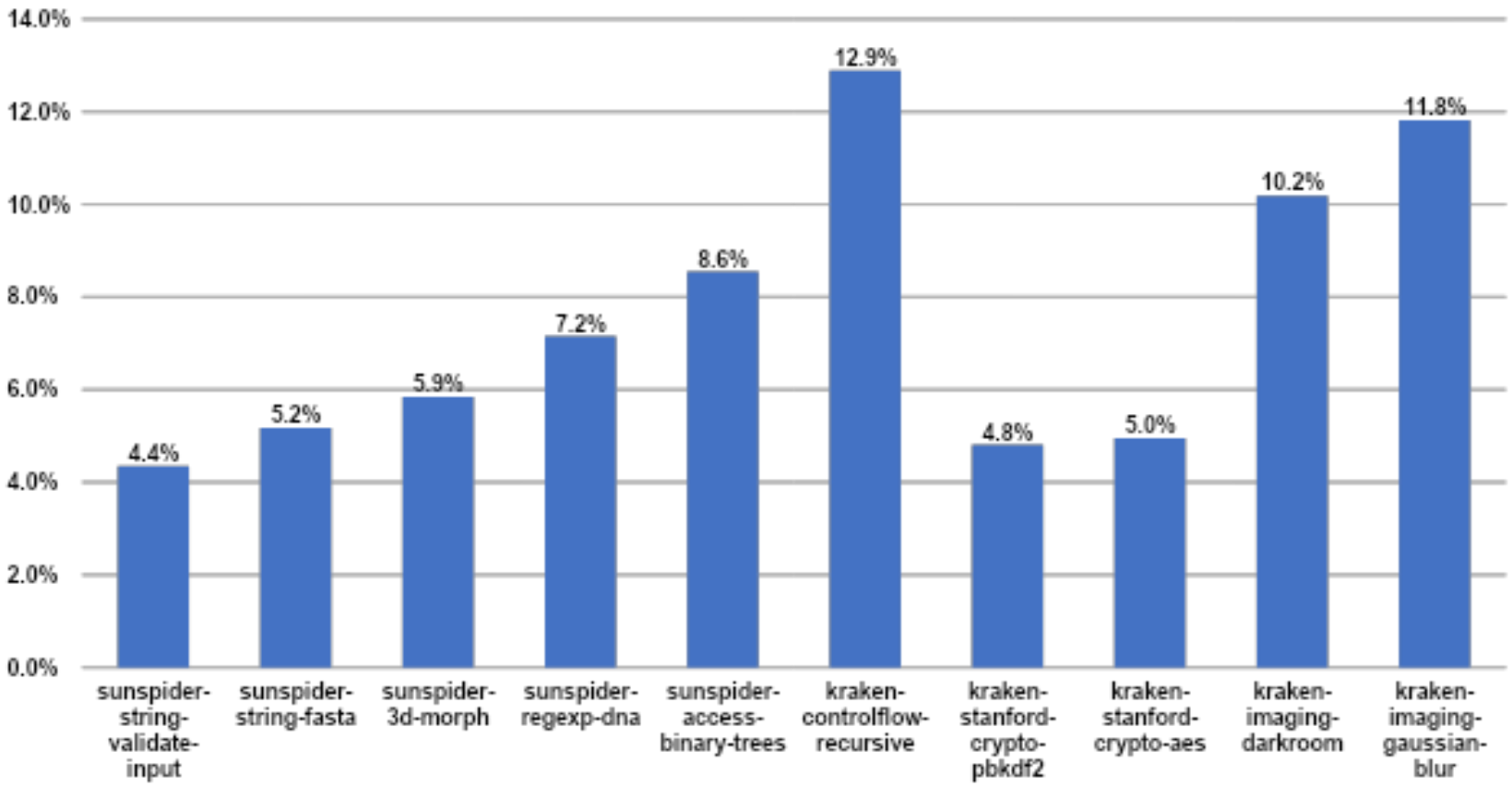
ISA	Instruction sequence	counts

RISCV64	int64_t a6 = imm & 0x3f; // bits 0:5. 6 bits
	int64_t b11 = (imm >> 6) & 0x7ff; // bits 6:11. 11 bits
	int64_t high_31 = (imm >> 17) & 0x7fffffff; // 31 bits
	int64_t high_20 = ((high_31 + 0x800) >> 12); // 19 bits
	int64_t low_12 = high_31 & 0xfff; // 12 bits
	lui(rd, (int32_t)high_20);
	addi(rd, rd, low_12); // 31 bits in rd.
	slli(rd, rd, 11); // Space for next 11 bis
	ori(rd, rd, b11); // 11 bits are put in. 42 bit in rd
	slli(rd, rd, 6); // Space for next 6 bits
	ori(rd, rd, a6); // 6 bits are put in. 48 bis in rd
		6

MIPS64	lui(rd, (j.immediate() >> 32) & kImm16Mask);
	ori(rd, rd, (j.immediate() >> 16) & kImm16Mask);
	dsll(rd, rd, 16);
	ori(rd, rd, j.immediate() & kImm16Mask);
		4

li_ptr: load a 48bit immediate into a register in a fix length sequence
RISC-V: I-Type has only 12bit imm filed
MIPS64: I-Type has 16 bit imm field

li_ptr macro generates more instruction-2

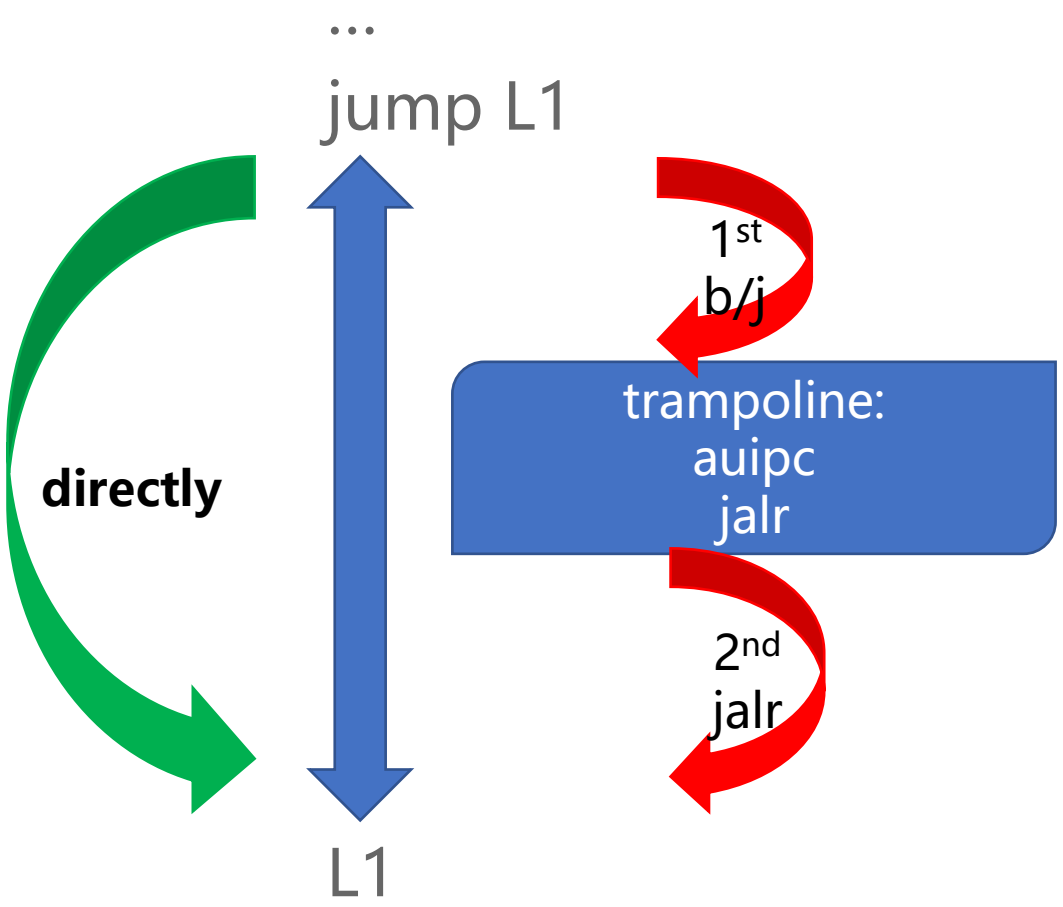


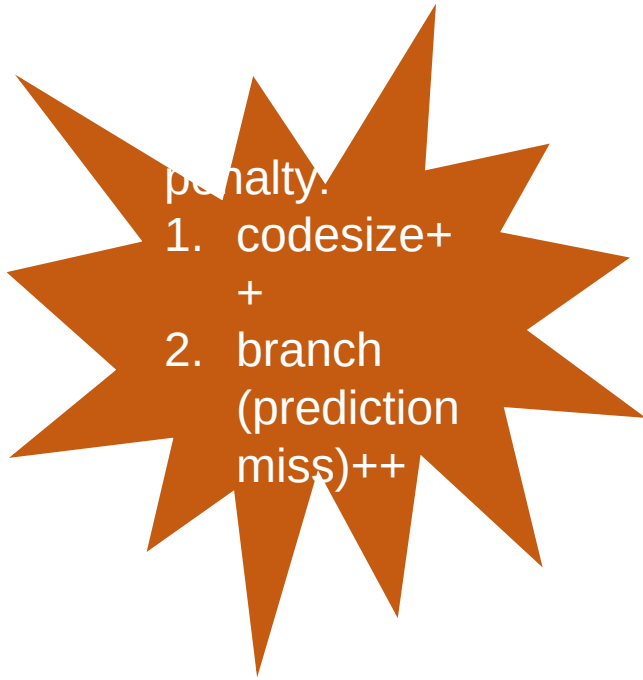
Feedback2: short offsets of the direct branch and jump

Trampoline must be emitted

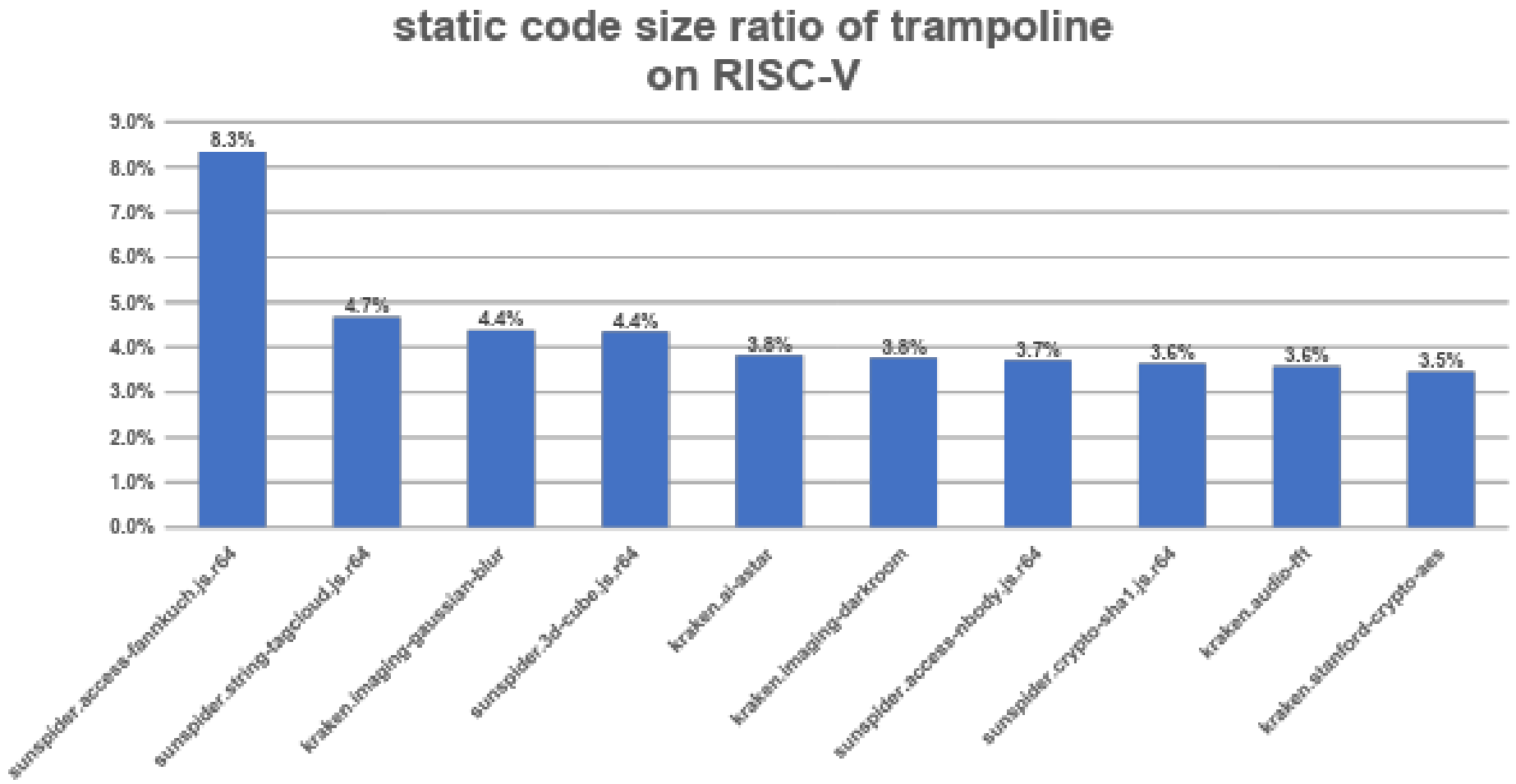
When the distance is within the offset range of jal and beq.

Else, trampoline is emitted.

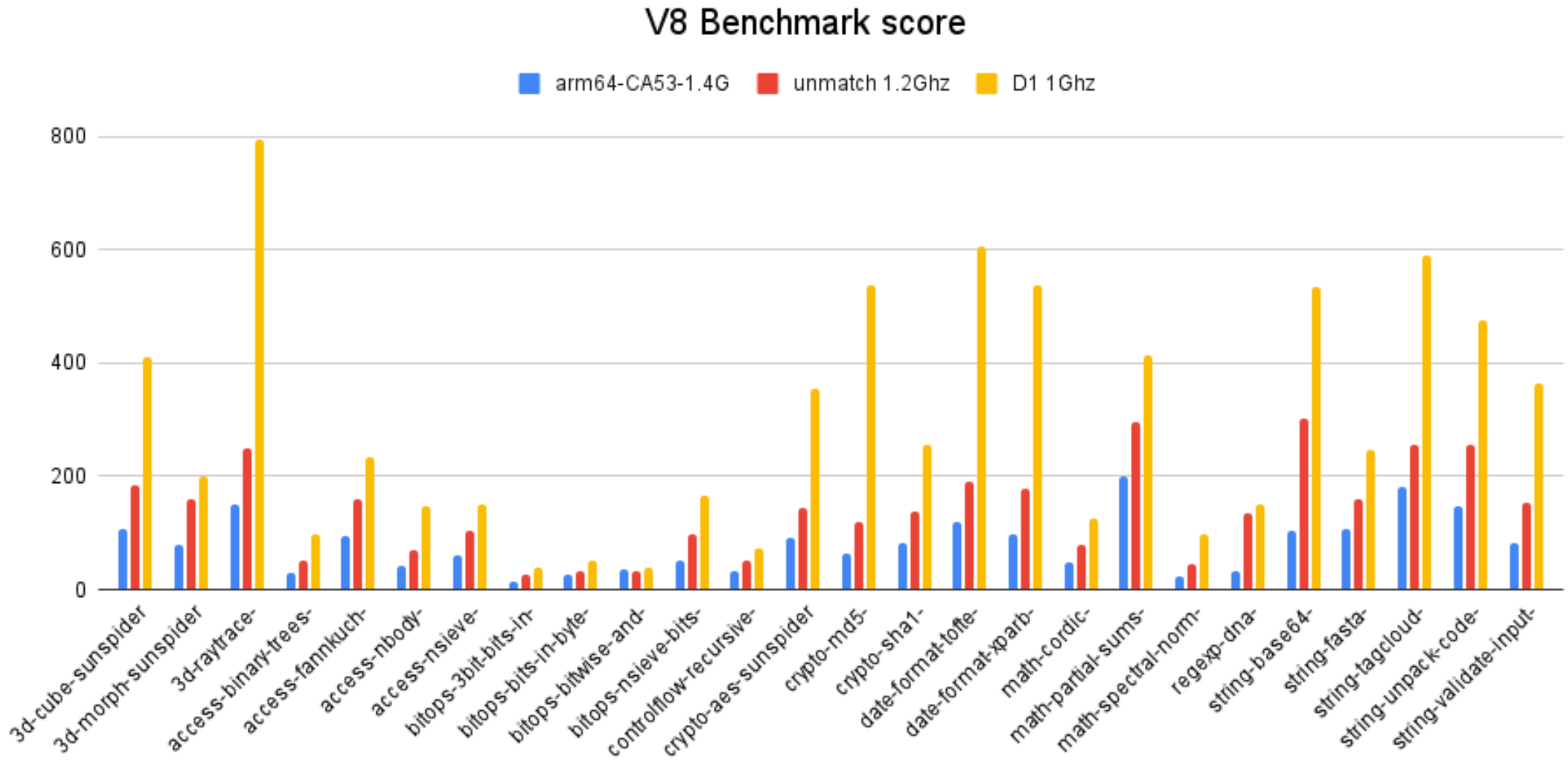




trampolines result in larger code size



V8 Benchmark score



https://github.com/riscv/v8/issues/494

V8 for RISCV Roadmap-Items lists with priority from high to low

- Upstreaming(now porting and upstreaming, keeping pace with master to ensure successfully build)
- Demo and application launch
  - Demo tensorflow.js on RISC-V (a very simple cmd line demo is done by the end of Q1)
  - Embrace node.js community and support all the node.js ecosystem running on RV64GC (upstreaming is launching by the end of Q1) https://github.com/v8-riscv/node/
  - Chromium running on PicoRio with V8(on going by RIOS lab)
- Performance (on going by PLCT)
  - Set up the performance tracking mechanism. (https://ci.rvperf.org/view/V8/ is on progressing)
  - Speed up (blind say 5x) on RV64GC compared with the code we first upstreaming
- ISA-extension
  - C (on going by RIOS lab)
  - V (on going by PLCT)
  - B/P(todo)
- Ecosystem adaption
  - build v8 with LLVM(todo)
- Community develop and enlarge (on going by PLCT)
  - resurrecting more than 10 new interns/contributors/graduates into our project.

done
on progressing
to do

Q&A

Thanks to All the Contributors and Community Members!

Jion us:

https://github.com/riscv/v8/wiki

https://github.com/plctlab/weloveinterns