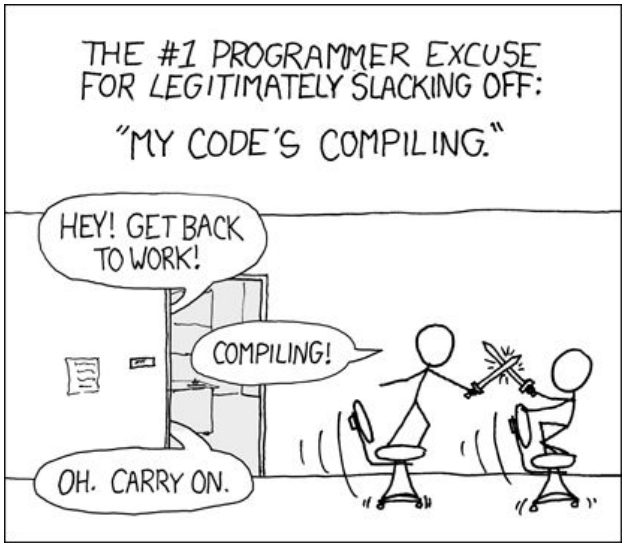
The Path to Instant Rebuilds

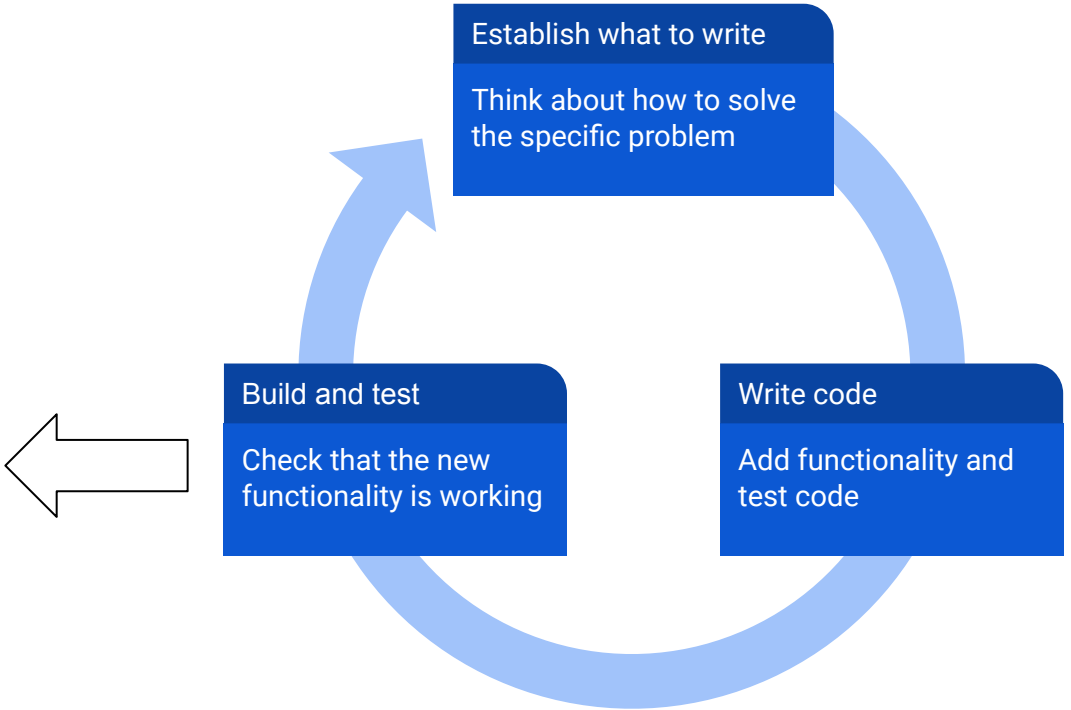
Less Context Switching and Sub Minute Submit Times

DevLin2022

Software Development Loop



https://xkcd.com/303/

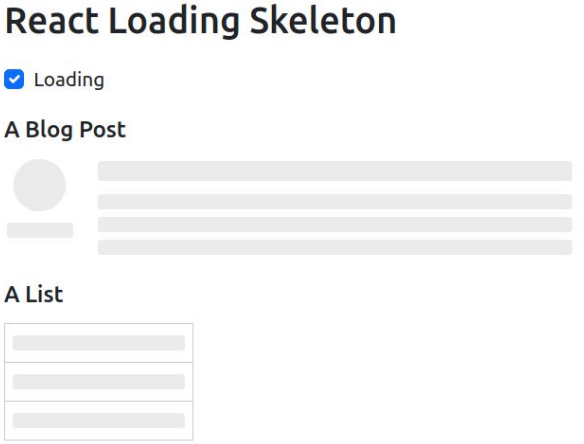


Build Startup Time

- Time before the first compile command is running
- Example: Chromium project
  - 30000 source files
  - 10s startup time with Make
  - Created Ninja
  - 1s startup time acceptable
  - https://ninja-build.org/manual.html#_introduction

Startup Time

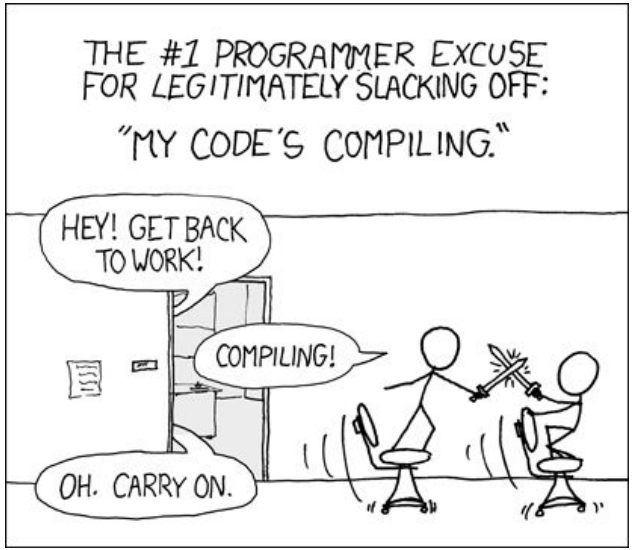
- Example: Web skeleton loading
  - Don’t let the user loose focus or go back
- Example: Chrome browser startup time
  - Regression test
- Solution: Choose your build system wisely



https://www.npmjs.com/package/react-loading-skeleton

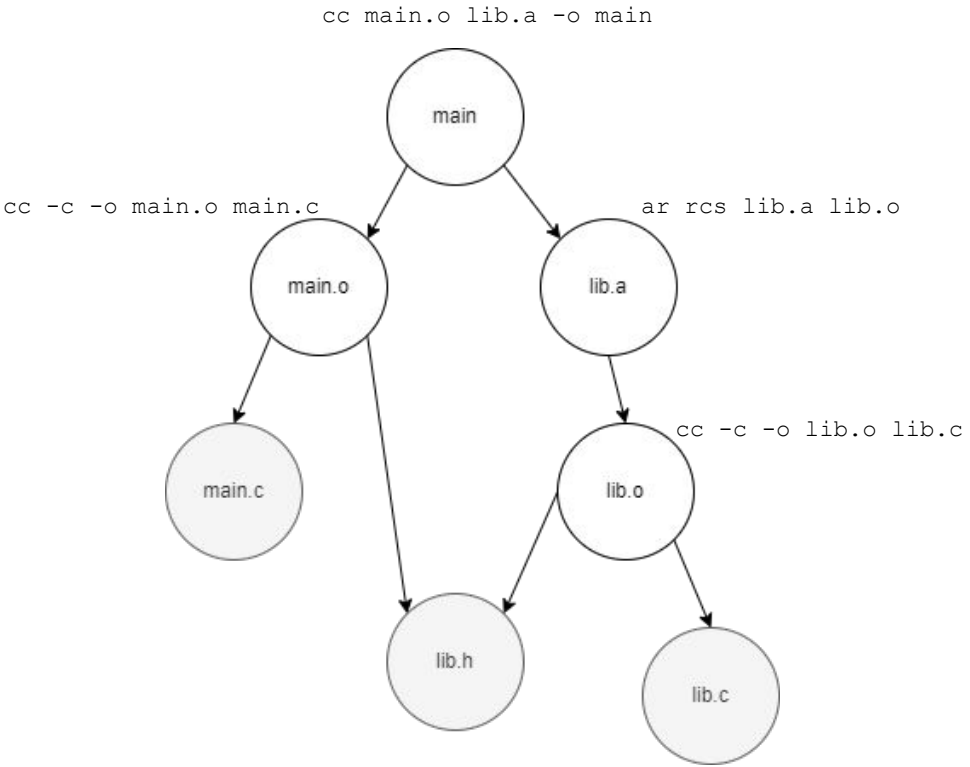
MEROTON
SPEED UP YOUR BUILDS

Build and Test Time



How to Build and Test

- Example C program
  - main.c, lib.c, lib.h
- Build system
  - Describe the dependency graph
  - Examples: Make, Scons, Bazel
- NOTE: Everyone builds the same

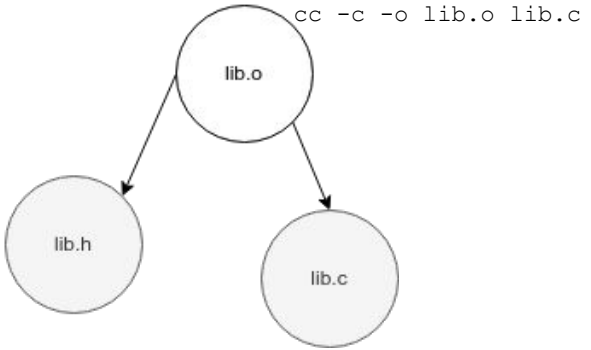


Remote Cache

- Action description
  - Command line
  - Environment variables
  - Input files and content

- Store the results on a cache server
  - More than 99% cache hit, built by your colleagues or CI
  - **Reduce build time by 90%**

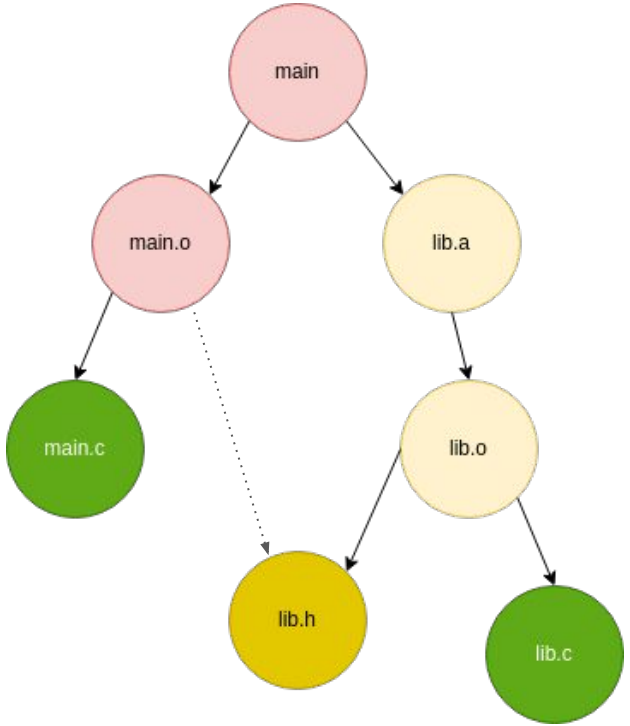
- Deterministic actions
  - No random numbers
  - No date or time



Underdescribed Graph - Cache Poisoning

- What if the graph is underdescribed?
  - This is a bug.
  - Builds might be correct by luck.
  - Workaround: **make clean && make all**

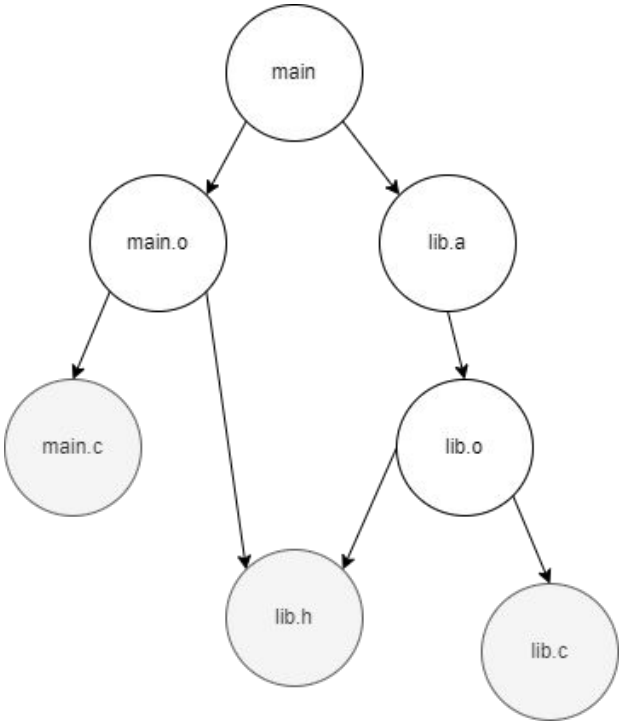
- Avoid cache poisoning
  - Sandbox each action
  - … or use remote execution



Remote Execution

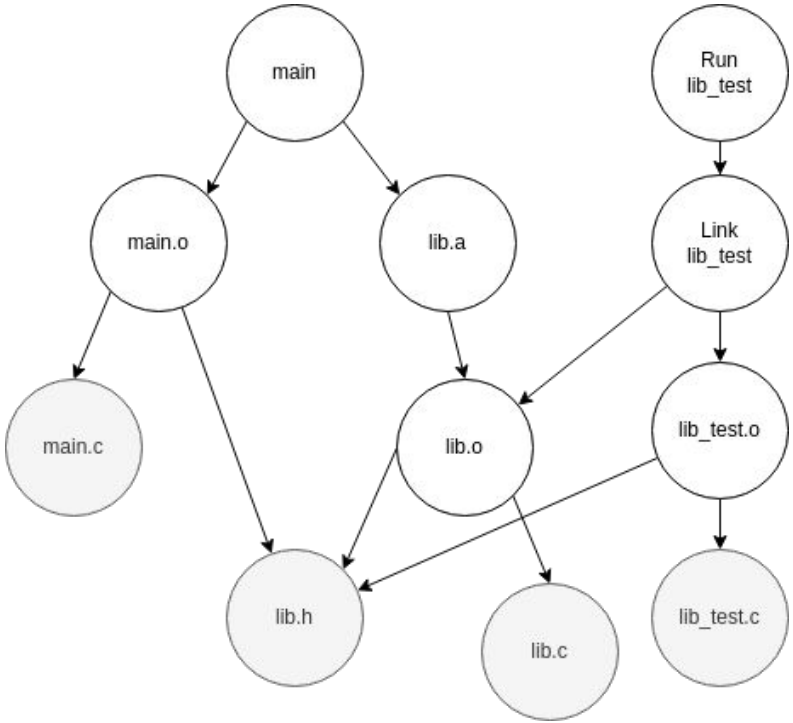
- Compilations can be done in parallel
  - make -j1000
- Underdescribed actions won't build
  - Cannot access local host
  - In practice sandboxed environment

- Move the CPUs to the cluster
  - Small developer machines
  - Large cluster
  - Time share the resources
  - **Reduce build time by another 90%**



No Difference Between Build and Test

- Both are spawning actions
- Run the test in parallel with building main



Remote Execution API

https://github.com/bazelbuild/remote-apis

- Clients
  - Bazel - Google (Blaze)
  - Pants - Twitter
  - Buck - Facebook
  - Goma - Chromium (with ninja)
  - …
- Servers
  - Buildbarn
  - Buildfarm
  - bazel-remote (cache only)
  - …

Poll - Build System

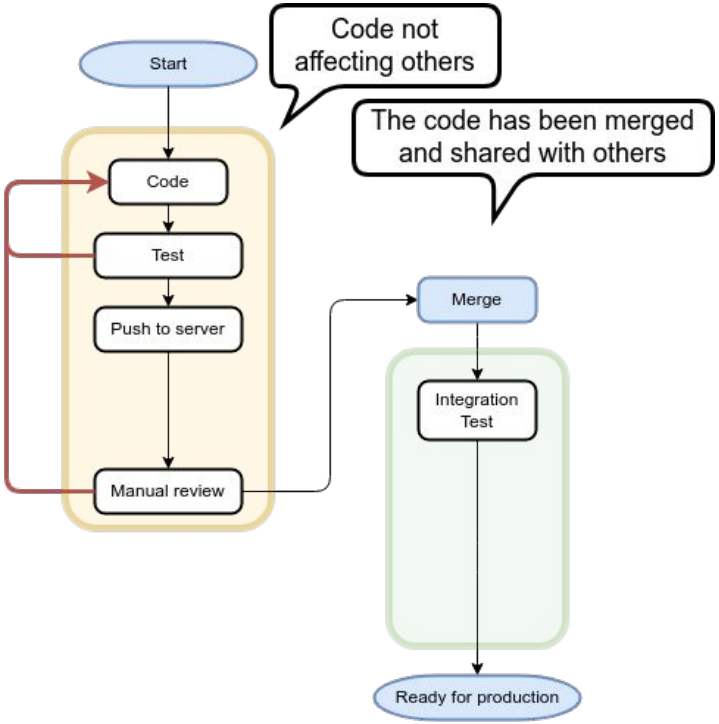
Do you use

- Remote cache
- Remote execution

Continuous Integration

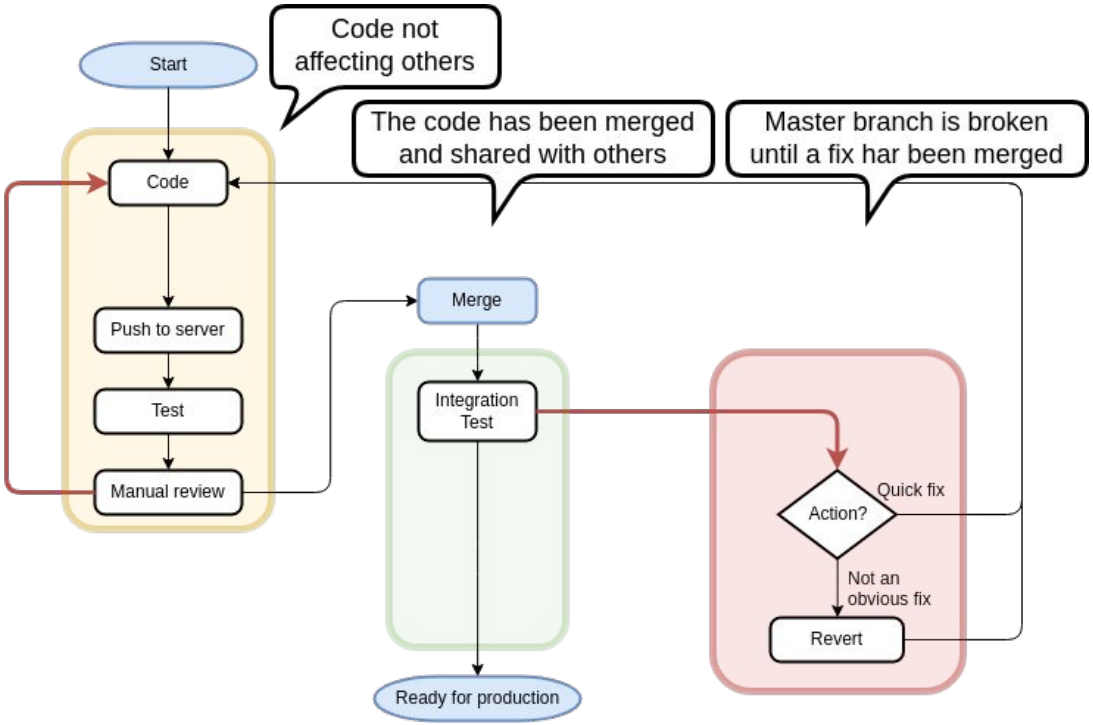
Code Integration

- Two phases
  - Presubmit
    - Assigned to a task
  - Post submit
    - Task done



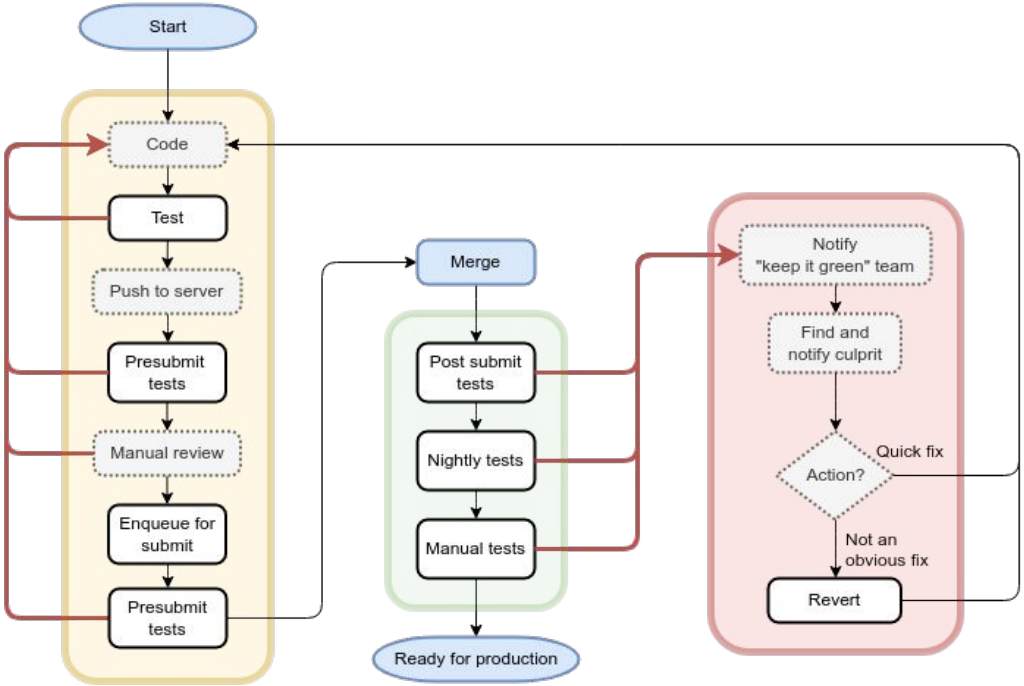
Code Integration

- Three phases
  - Presubmit
  - Post submit
  - Broken state
- Keep repo green
  - “All” tests should always pass



Move Tests Earlier

- Three phases
  - Presubmit
  - Post submit
  - Broken state
- Keep repo green
  - “All” tests should always pass
- Automated tests
  - When to run?
  - Machine cost



Summary - Problems with Latency

MEROTON
SPEED UP YOUR BUILDS

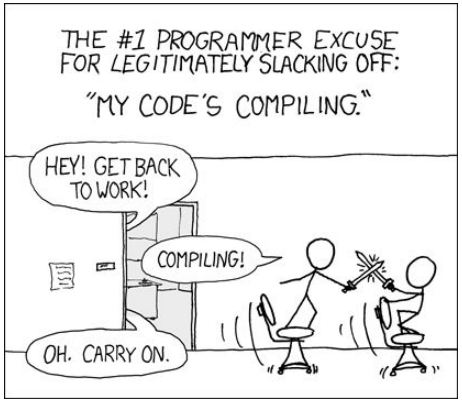
- Long startup times
  - Lost focus
- Developers need to juggle multiple tasks in parallel
  - Annoying to wait hours for fixing minor things
- Lower willingness to fix typos
  - Increased technical debt
- Reusing the same commit for multiple things
  - More difficult to review

- Extends the critical path
  - Slower feature growth in product

Poll - Presubmit times

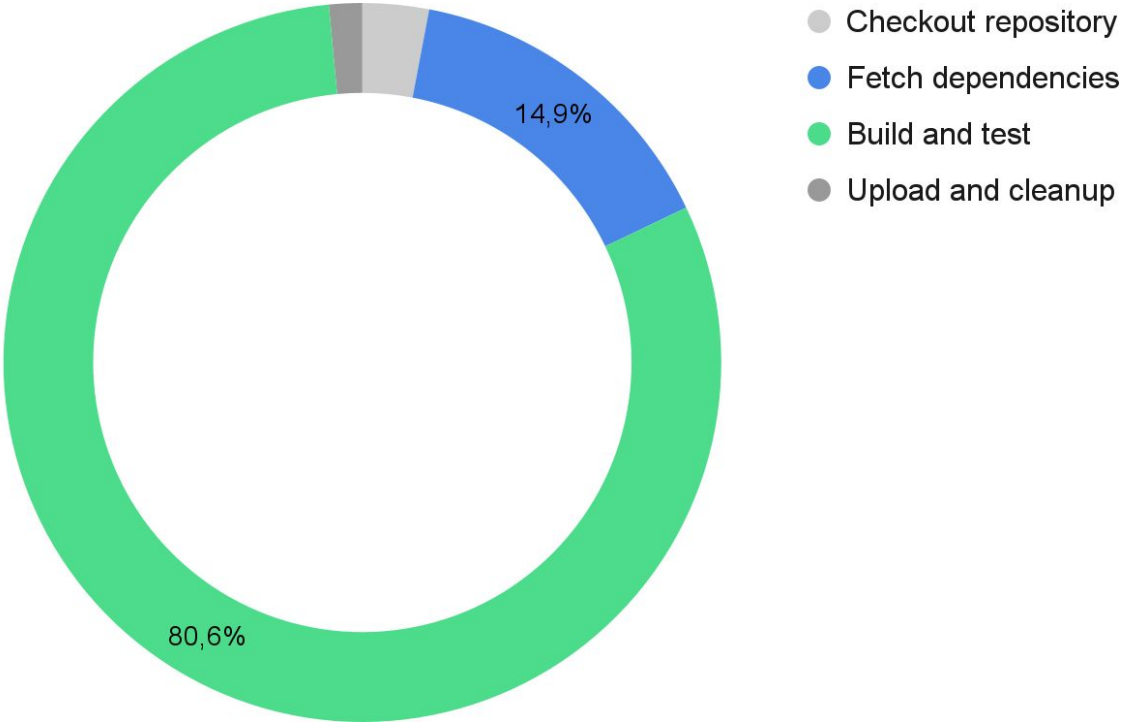
Average time from triggering presubmit checks until merge

- Have at least some tests
- <1h
- <30m
- <10m
- <1m



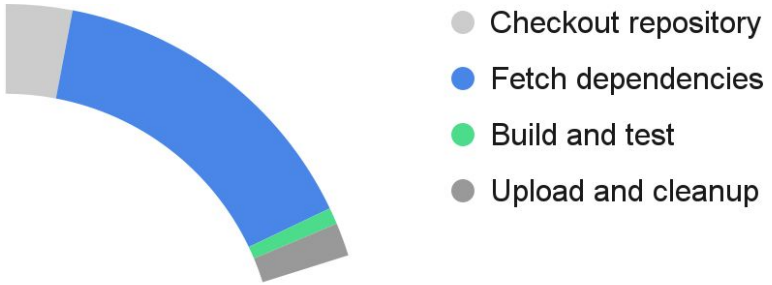
CI Time - Locally

- Disk and network intensive
  - During startup
- CPU intensive
  - When compiling



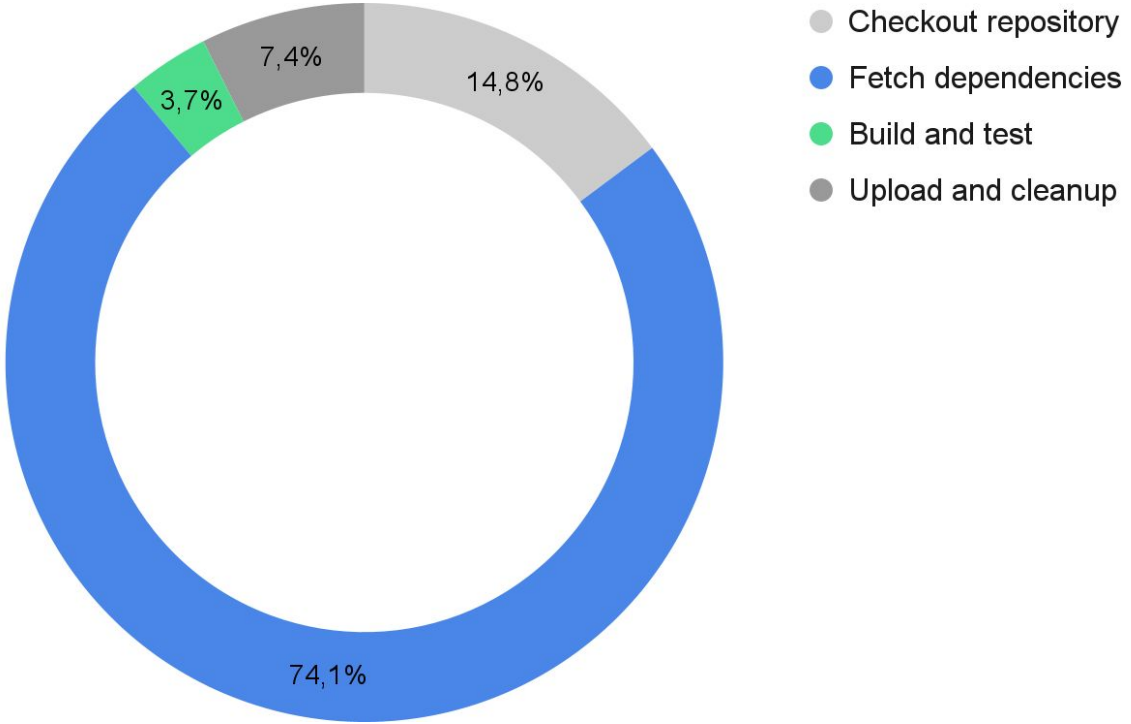
CI Time - Remotely

- CPU intensive tasks minimized



CI Time - Remotely

- Disk and network intense
  - Not much CPU needed
- Next step
  - Reuse the workspace
  - Requires fully specified build graph



Resolving Red Repo State

Rebuild Locally

- Reproduce the error locally
- Start debugging
- Fix and test it

Rebuild Locally - Mimic the CI Pipeline

- CI Tools
  - Jenkins pipelines
  - GitHub actions
  - Gitlab pipelines
- Transfering artifacts
  - Deliverables between steps store externally

or…

- Shell scripts
- Defined process or helper scripts
  - … to reproduce certain CI steps locally

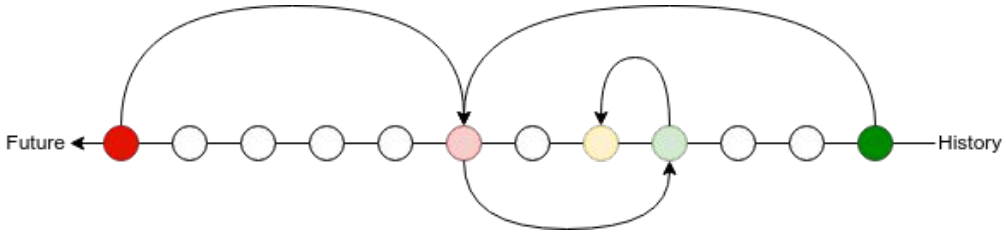
Visualize Individual Test Results over Time

MEROTON
SPEED UP YOUR BUILDS

Results for abseil-cpp@master	1649c03 2022-11-02 19:45:24	a87df8e 2022-11-02 14:12:54	0064d9d 2022-11-01 21:27:28	2b403ec 2022-11-01 19:13:10	d819278 2022-10-31 20:32:21	ea882fb 2022-10-31 19:21:32	f5fd4cc 2022-10-31 17:27:59	37a1aa0 2022-10-31 16:24:14	1db72eb 2022-10-28 20:22:03	fd9fbe7 2022-10-28 05:22:33	8279400 2022-10-27 22:45:48	1b6a9e9 2022-10-27 10:11:42	bc09731 2022-10-26 19:52:27	4691321 2022-10-26 17:15:07	2fc358d 2022-10-25 21:56:36	90184f6 2022-10-25 20:08:45
bazel test ... -//absl/cleanup/...																
- //absl/...	220/220	219/220	219/220	219/220	219/220	219/220	220/220	220/220	220/220	220/220	220/220	220/220	220/220	220/220	220/220	220/220
+ //absl/algorithm/...	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2
+ //absl/base/...	20/20	20/20	20/20	20/20	20/20	20/20	20/20	20/20	20/20	20/20	20/20	20/20	20/20	20/20	20/20	20/20
+ //absl/container/...	23/23	23/23	23/23	23/23	23/23	23/23	23/23	23/23	23/23	23/23	23/23	23/23	23/23	23/23	23/23	23/23
+ //absl/debugging/...	5/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5	5/5
+ //absl/flags/...	11/11	11/11	11/11	11/11	11/11	11/11	11/11	11/11	11/11	11/11	11/11	11/11	11/11	11/11	11/11	11/11
- //absl/functional/...	4/4	3/4	3/4	3/4	3/4	3/4	4/4	4/4	4/4	4/4	4/4	4/4	4/4	4/4	4/4	4/4
//absl/functional:any_invocable_test	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass
//absl/functional:bind_front_test	Pass	Fail	Fail	Fail	Fail	Fail	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass
//absl/functional:function_ref_test	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass
//absl/functional:function_type_benchmark	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass
+ //absl/hash/...	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3
+ //absl/log/...	14/14	14/14	14/14	14/14	14/14	14/14	14/14	14/14	14/14	14/14	14/14	14/14	14/14	14/14	14/14	14/14
+ //absl/memory/...	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2
//absl/meta:type_traits_test	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass
+ //absl/numeric/...	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3
+ //absl/profiling/...	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3	3/3
+ //absl/random/...	27/27	27/27	27/27	27/27	27/27	27/27	27/27	27/27	27/27	27/27	27/27	27/27	27/27	27/27	27/27	27/27
+ //absl/status/...	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2	2/2
+ //absl/strings/...	59/59	59/59	59/59	59/59	59/59	59/59	59/59	59/59	59/59	59/59	59/59	59/59	59/59	59/59	59/59	59/59
+ //absl/synchronization/...	8/8	8/8	8/8	8/8	8/8	8/8	8/8	8/8	8/8	8/8	8/8	8/8	8/8	8/8	8/8	8/8
+ //absl/time/...	7/7	7/7	7/7	7/7	7/7	7/7	7/7	7/7	7/7	7/7	7/7	7/7	7/7	7/7	7/7	7/7
+ //absl/types/...	10/10	10/10	10/10	10/10	10/10	10/10	10/10	10/10	10/10	10/10	10/10	10/10	10/10	10/10	10/10	10/10
//absl/utility:utility_test	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass	Pass
bazel test -c opt //absl/cleanup/...																
- //absl/...				0/1			0/1					1/1				1/1
//absl/cleanup:cleanup_test				Fail			Fail					Pass				Pass

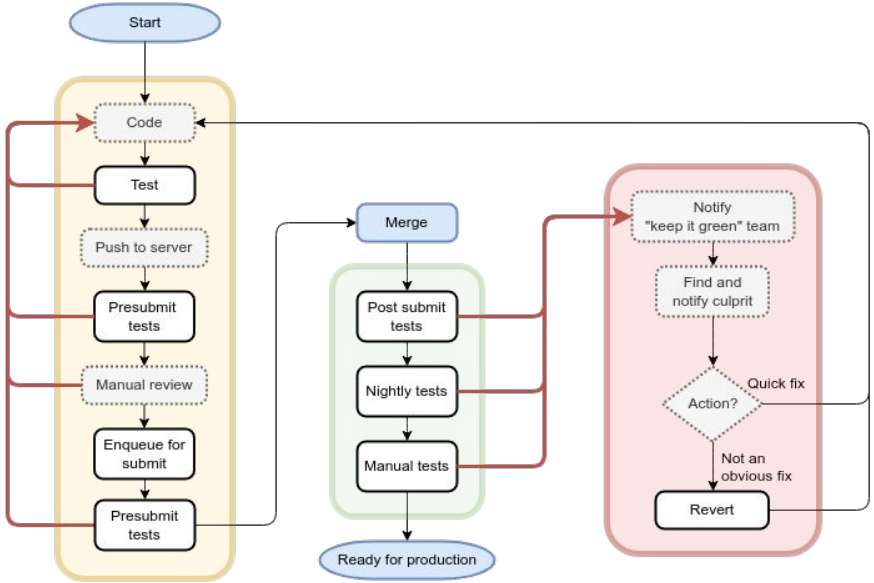
Interval Halving

- Interval halving to find the bad commit
  - git bisect start HEAD <good-commit> –
  - git bisect run ./run_test.sh
  - git bisect reset
- Start automatically
  - … for nightly tests
- Review suggested revert
  - Fooled by a flaky test?
  - Is there an easy fix instead?
- Manual inspection in case of slow and expensive tests
  - Timeouts
  - Bricking hardware



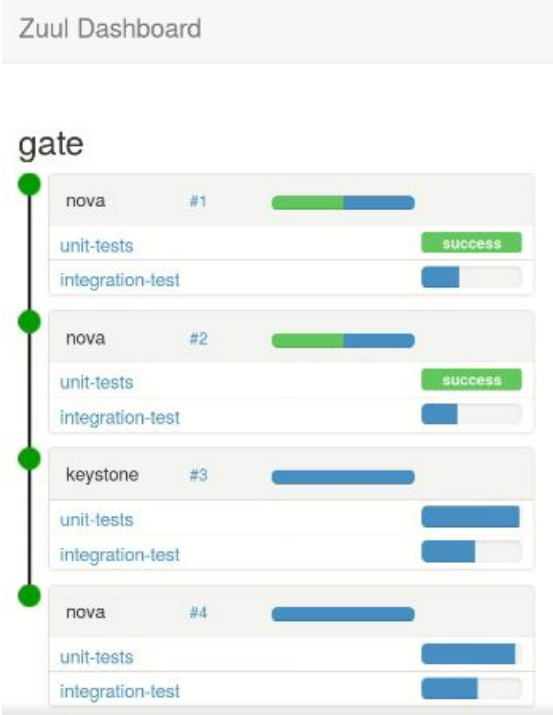
Keep Master Green

- Test what will be the merge result
- Move tests early
  - Includes integration tests



Speculative Gate Queue

- Test the future state after merge
  - Test main+1
  - Test main+1+2
  - Test main+1+2+3
  - Test main+1+2+3+4

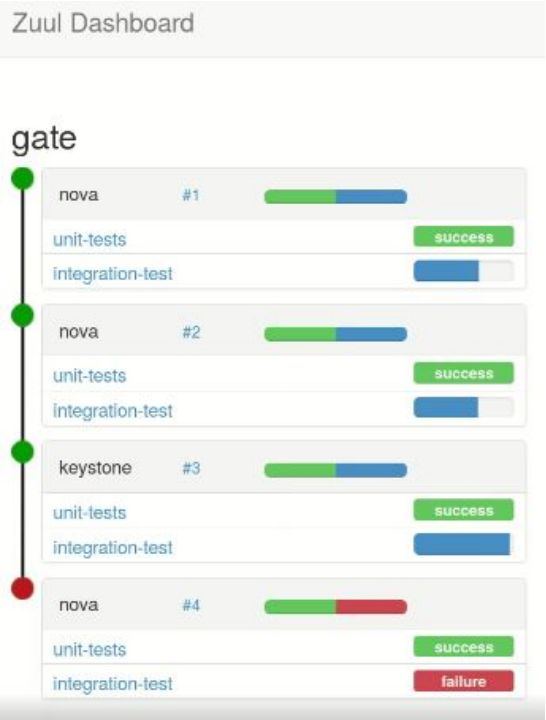


https://zuul-ci.org/

Speculative Gate Queue

- Test the future state after merge
  - Test main+1
  - Test main+1+2
  - Test main+1+2+3
  - Test main+1+2+3+4 - Failing

- Is #4 bad?

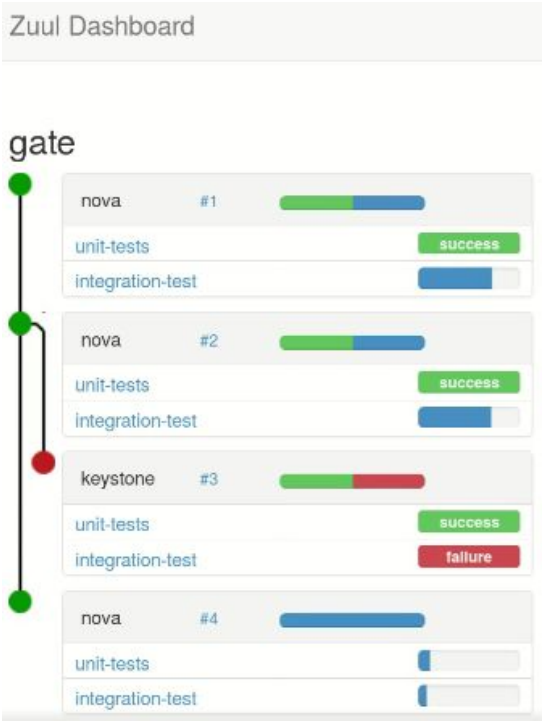


https://zuul-ci.org/

Speculative Gate Queue

- Test the future state after merge
  - Test main+1
  - Test main+1+2
  - Test main+1+2+3 - Failing
  - Test main+1+2+4

- Is #3 bad?

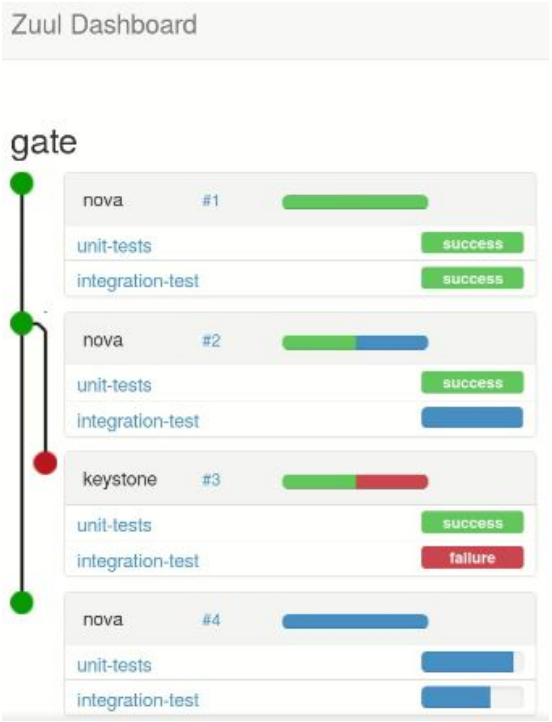


https://zuul-ci.org/

Speculative Gate Queue

- Test the future state after merge
  - Test main+1 - Merging
  - Test main+1+2
  - Test main+1+2+3 - Failing
  - Test main+1+2+4

- Is #3 bad?

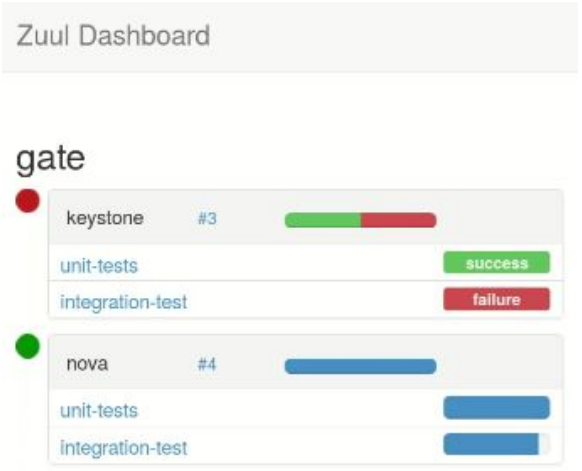


https://zuul-ci.org/

Speculative Gate Queue

- Test the future state after merge
  - Test main+1 - Merged
  - Test main+1+2 - Merged
  - Test main+1+2+3 - Failing
  - Test main+1+2+4

- #3 is bad



https://zuul-ci.org/

Optimal Gate Queue

MEROTON
SPEED UP YOUR BUILDS

- Keeping Master Green at Scale - Uber
  - https://dl.acm.org/doi/pdf/10.1145/3302424.3303970
- Optimistic gate queue is not optimal
  - Changes have a probability of failing
    - Test 1+2+3
    - Test 1+3+4
    - Test 1+2+4
    - Then try 1+2+3+4
  - Correlated with
    - Size of change
    - Certain files
    - Author…

Poll - Gate Queue

Do you use a gate queue

- Tests on push
- Tests after review
- Speculative gate testing after review

Conclusion

- Quick local incremental build -> Keep mind focused
- Remote cache/execution -> Faster builds, for developers and CI
- Fast CI -> Finish one task at a time
- Fast CI -> Encourage small fixes, less technical debt

- Shell scripts -> Locally reproducible builds, now and in the future
- Shell scripts -> Avoid vendor lock in
- Gate queue -> Keep Master Green

References

Tools of interest:

- Bazel - {Fast, Correct} - Choose two
  https://bazel.build/
- Buildbarn
  https://github.com/buildbarn/bb-deployments/
- Zuul CI
  https://zuul-ci.org/

Previous talks on the subject:

- One Minute Presubmits
  https://docs.google.com/presentation/d/14dxac2omYI5Feaoiw-u09qB1fQgJU7ASNJag7YMg3TI/
- Selective testing in Bazel
  - BazelCon 2019 - Selective Testing by Benjamin Peterson
    https://www.youtube.com/watch?v=9Dk7mtIm7_A
  - https://github.com/Tinder/bazel-diff