GATES

A General AB TEsting Service

Ilija Vukotic
University of Chicago

GDB 2019-02-16

Content

- What is A/B testing?
- Why do we need it?
- How do we technically do it?
- An example of A/B testing workflow.

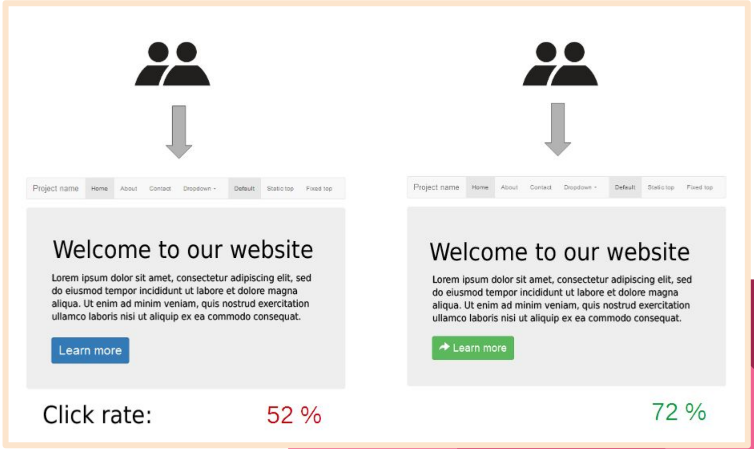
What is A/B testing?

From Wikipedia, the free encyclopedia

A/B testing (**bucket tests** or **split-run testing**) is a randomized experiment with two variants, A and B It includes application of statistical hypothesis testing or "two-sample hypothesis testing" as used in the field of statistics. A/B testing is a way to compare two versions of a single variable, typically by testing a subject's response to variant A against variant B, and determining which of the two variants is more effective.

Most often used in website optimization. Google even optimizes the shade of each button on a page. There is a whole industry that simplifies and speeds up A/B testing (Optimizely, VWO,...).

Much less often seen outside of software development as experiments are much more expensive.


Project name
Home
About
Contact
Dropdown
Default
Static top
Fixed top
Welcome to our website
Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.
Learn more
Click rate:
52 %
72 %

Why do we need it?

Experiments have large data stores collecting data from different computing systems: job scheduling, data distribution, FTS, PerfSONAR, etc.

While that is great for monitoring, accounting, and finding issues, it is not sufficient for the system optimization.

One can try to guess what kind of effect a change will made, but without validation it does not mean much.

We need a way to quickly test different options and get actionable answers.

Why do we need it?

Example 1

ATLAS created "C3PO" - a system to detect popular datasets and replicate for greater availability. Once deployed we did not know if it had any impact - positive or negative.

Simple approach would be to turn it ON for a week and OFF for a week and check if tasks using these datasets were processed faster during the ON week. But too many things change in one week to get reliable results (site availability, tasks in the pipeline, etc.).

It took three months to create a system to do A/B test*.

* Once a replication decision was made, replication was done for datasets whose hash was ending with 1 and not done if it ended with 0. Then Rucio data was joined with task data to extract result.

Why do we need it?

Example 2

We can set most of our computing elements to do copy-to-scratch or direct-access. Most sites don't know what kind of effect will this have on:

- Job error rate
- Job CPU efficiency
- Their infrastructure (LAN traffic, disk servers)

Often, site admins don't have time/expertise to make an informed decision.

nb. In this case you would want to have an A/B test with tunable weights
(eg. copy-to-scratch:direct-access = 80:20)

Why do we need it?

Other examples:

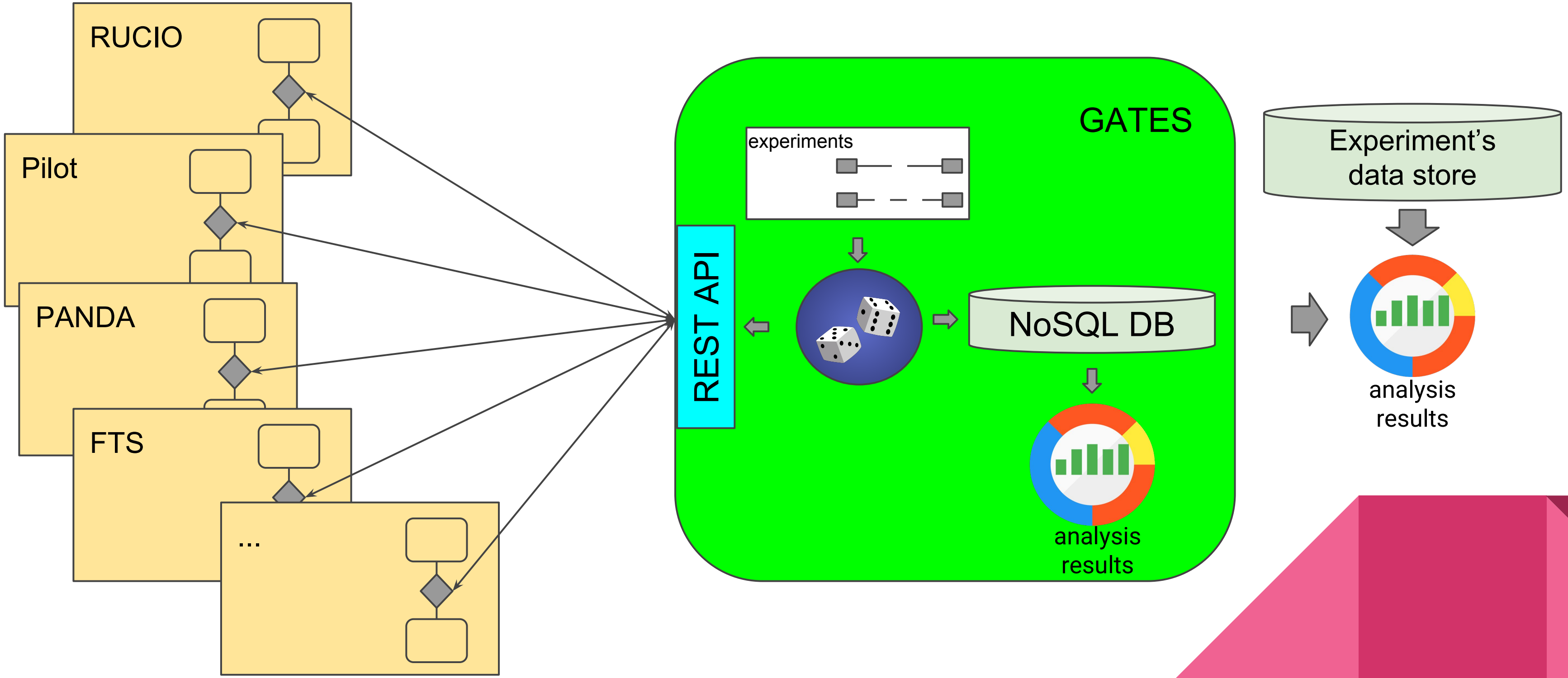
- Local vs Remote access
- Different job scheduling algorithms
- One data mover vs another
- Different FTS scheduling algos
- FTS limits per site/link
- Is a Rucio rule helping or not?
- Is there an impact from a different memory allocator (tcmalloc, jemalloc)?
- Is it better to oversubscribe servers: jobs/core=1,1.1 or 2?
- ...

How do we make it simple and fast to do A/B ?

Requirements for an A/B testing service:

- General (avoid experiment specific dependencies)
- Simple to instrument code (shell, python, c/c++)
- Fast and reliable (millions of experiments per day, with ms latency)
- Flexible (accommodate all different test options)
- Easy to correlate hypothesis with outcomes

Proposing GATES



How users do it? Variant 1.

- Come up with the hypothesis to test: eg. is copy-to-scratch better than direct-access?
- Define a new experiment in the GATES web front end. Sets: test name, weights for different options, etc.
- In ATLAS this setting is done in AGIS. We would add an option “test”.
- Rucio Mover is doing the operation. Once it finds setting “test” it contacts GATES with two parameters (jobid and testname). GATES returns randomly selected option and records selection, jobid, testname in the DB.
- Week later user downloads the test data from GATES, correlates it with data from the Panda DB, gets an answer which option is better.

How users do it? Variant 2.

- Come up with the hypothesis to test: eg. what is optimal size of TTreeCache size (10, 20, 30 MB)?
- Define a new experiment in GATES web front end. Sets: test name, weights for the 3 different options, and result.
- This setting is applied as a job configurable option.
- In job configuration, user’s code throws random number, measures times and reports to GATES (testname, selection, result). GATES just saves all in the DB.
- While jobs are running user gets plots of results in real time at GATES website.

Finally

- This is only a proposal for now.
- To do it quick and dirty ~ 1-3 months, full time developer
- To do it right 3-9 months (security, documentation, nice web interface).
- To educate people how to design experiments ~ lifetime :) While A/B tests are simple, one has to be careful:
  - To understand the problem
  - Develop a good hypothesis
  - Have enough statistics (specially with multivariant experiments)