How to Build a State Machine on MongoDB and Redis

Niki Castle @ Appboy

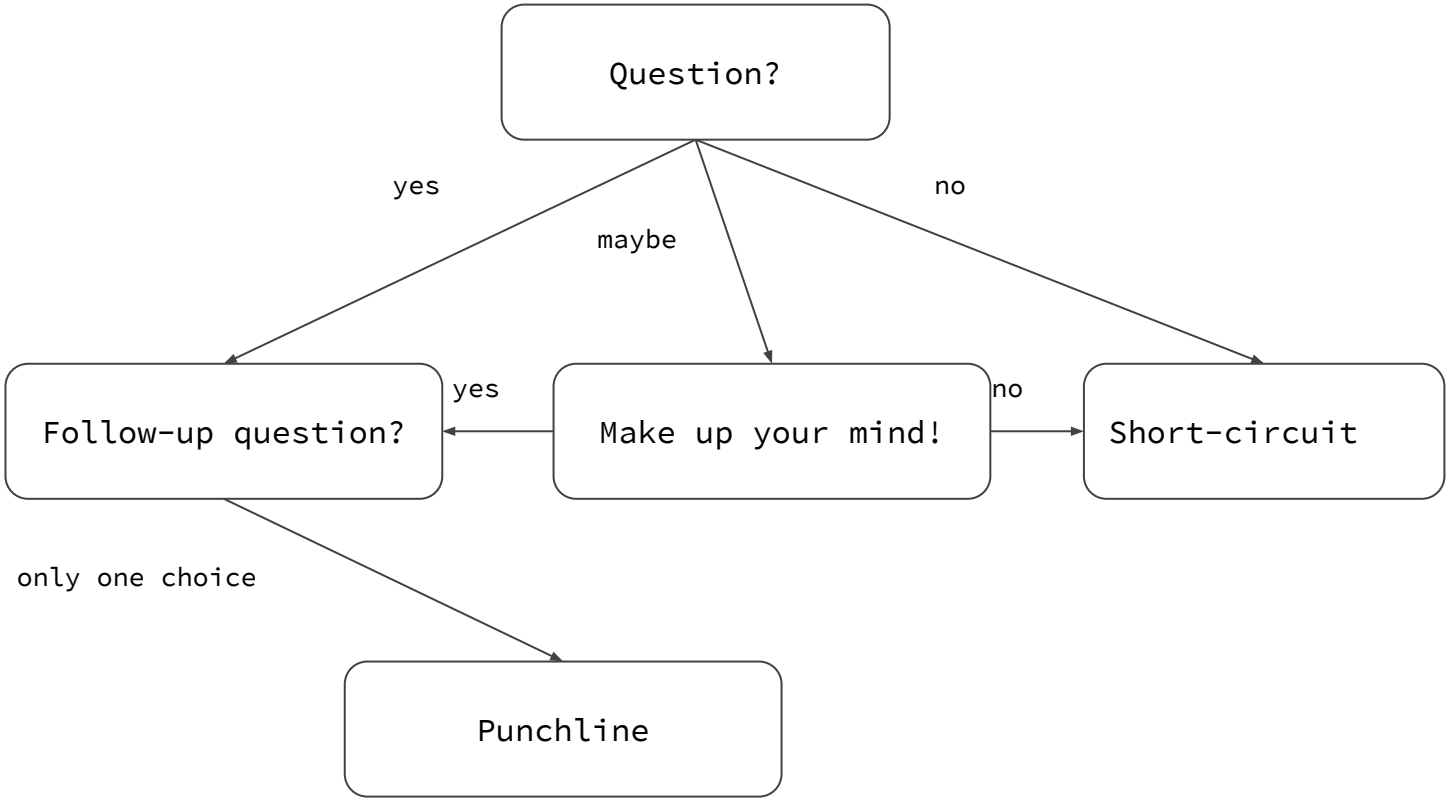
What is Appboy?

- CRM platform
- Message sender
- Analytics provider

Goals:

- Try really hard to send exactly one message!
- Never, ever, send the wrong message!

“And by state machine, you mean…”



Canvas

User Journey Tool

Motivation

Examples

Constraints

User State

Side-by-side: Linking two steps

Before:

- Build your first campaign
- Build your second campaign
- Set the second campaign to exclude people who haven’t received the first campaign
- Make sure your time windows are right
- Hope for the best
- Tag your steps so at least you can get some basic journey grouping

After:

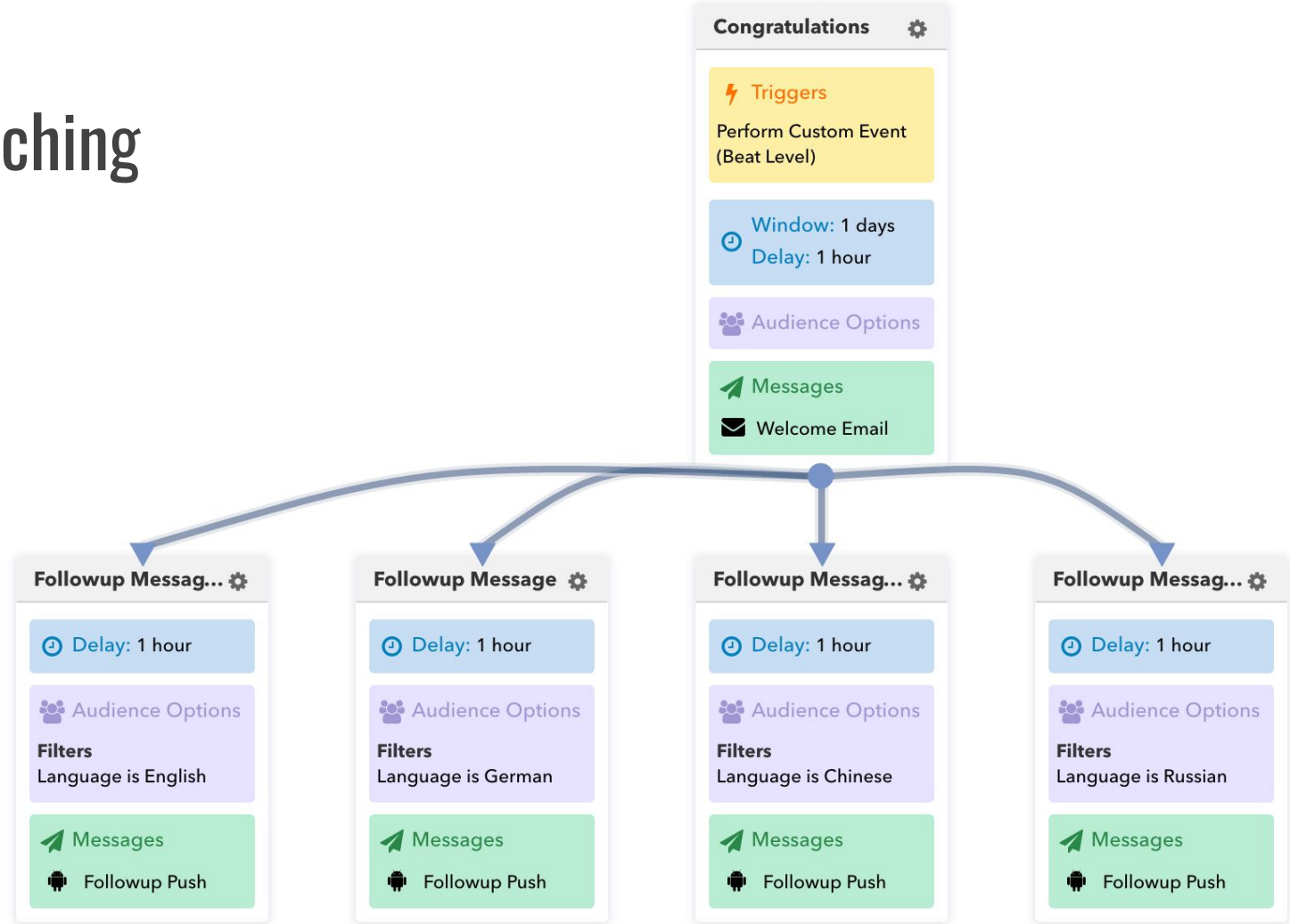
- Make a Canvas
- Build two steps
- Link them together -- we’ll handle eligibility for you!
- View the entire journey on one page, including analytics

The Base Case

---

1. Do trigger action within window
2. Wait specified delay
3. Receive first message
4. Wait specified delay (no action necessary)
5. Check filters
6. Receive second message

With Branching



The (Main) Constraints

State tracking:

- Time
- User attributes
- Actions performed
- Actions NOT performed

1 Mongo write to commit finalized state

Delay between executing state changes

User State

For each user:
{ <journey_id>, <entry_timestamp>: [next available steps] }
[user attributes]

Branching! (Redis as Lock)

In Theory

In Practice

In Theory

Super simple!

- Only go down one branch
- Do our best to send a (relevant) message

--> Try to send all of them, then as soon as one succeeds, cancel all the others

In Practice

Somewhat less simple!

Need: a way to cancel steps

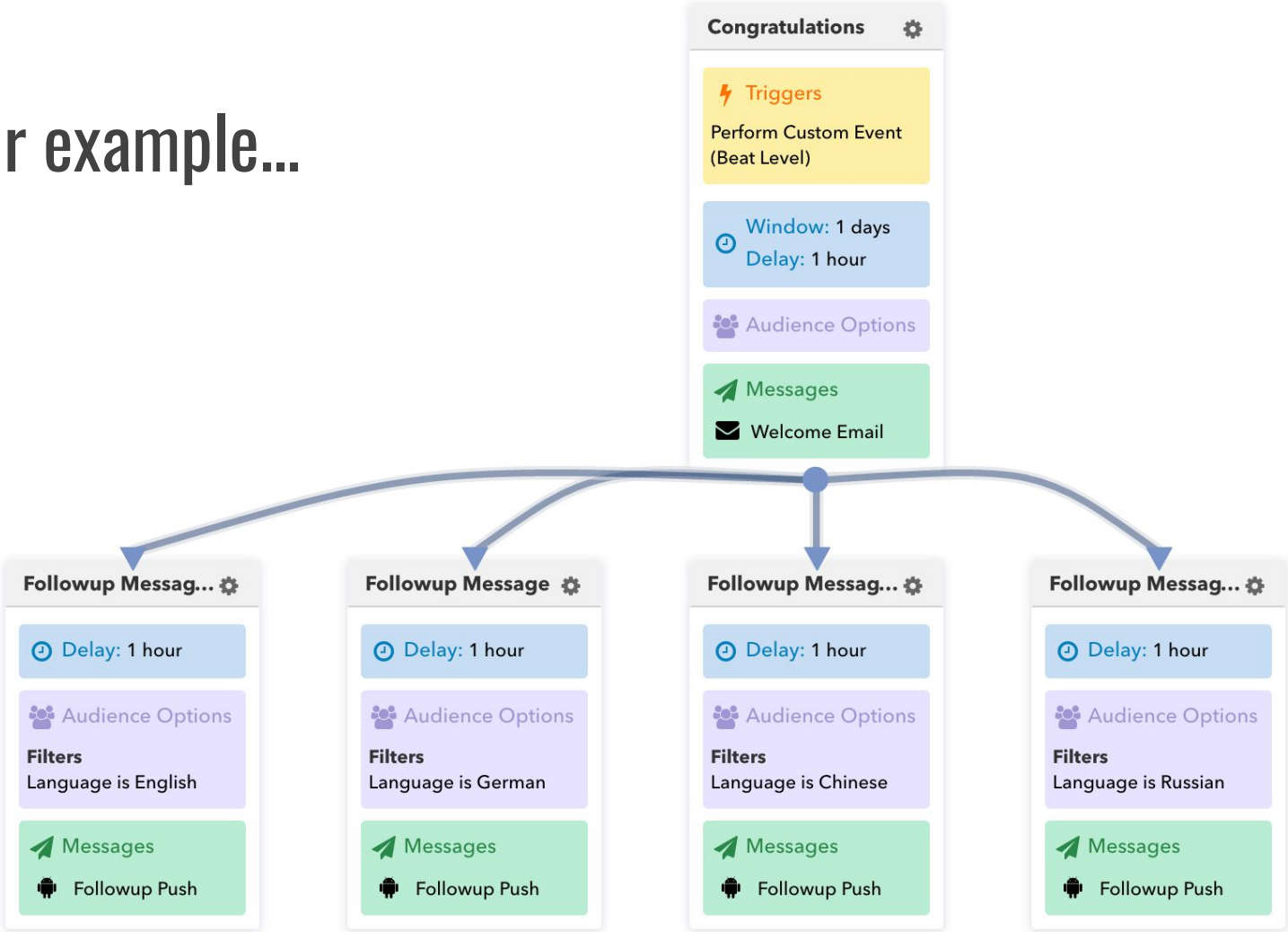
Need: 100% guarantee against >1 branch

Want: 100% guarantee we send something (unless ineligible)

When do we transfer state?

- Time
- User attributes
- Actions performed
- Actions NOT performed

Back to our example...



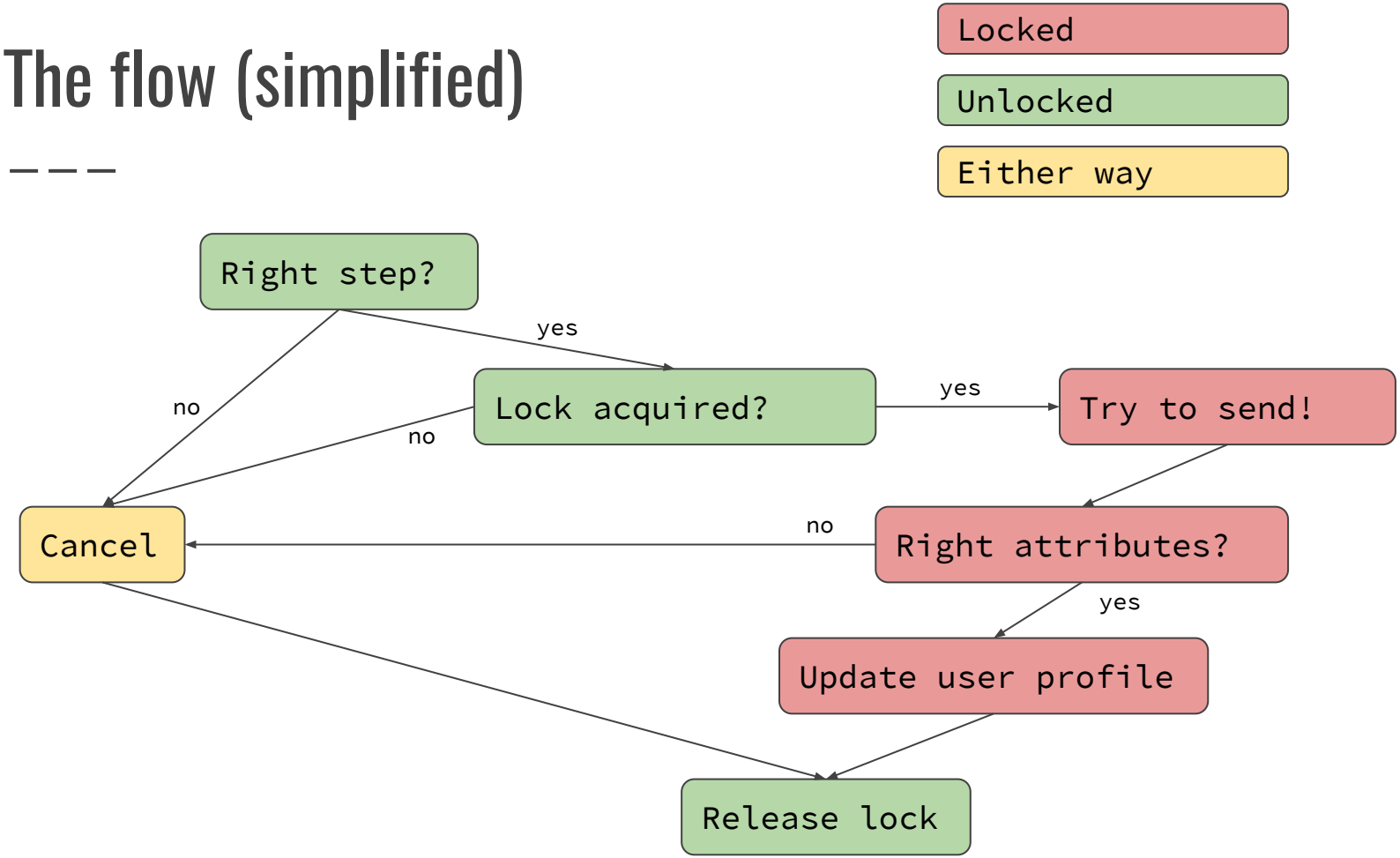
Redis as Lock

Unique instance of user in journey

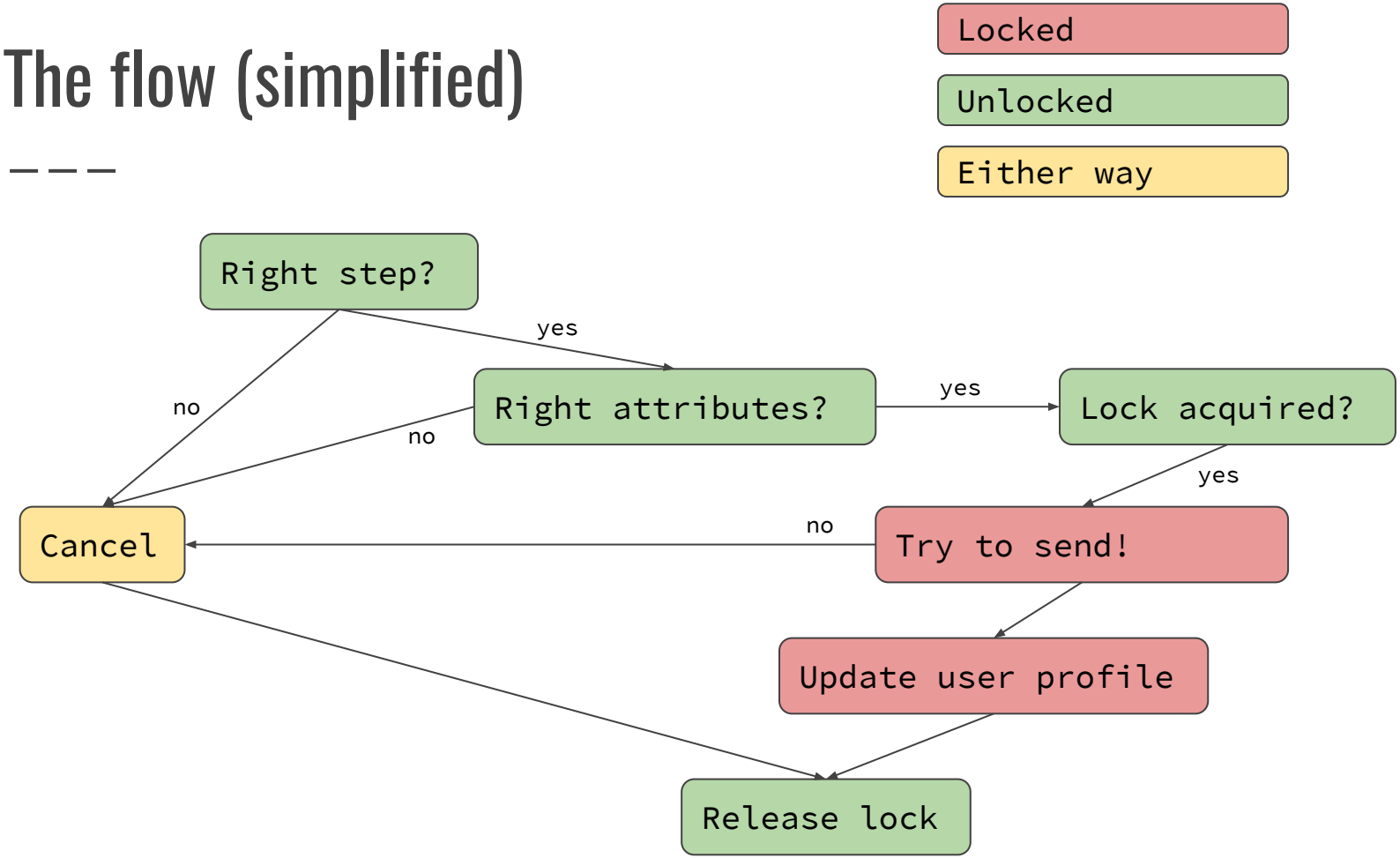
Key (lock name): “<journey_id>:<user_id>:<entry_id>”
Value: “<branch_id>"

1. Acquire lock
2. Send message
3. Release lock

The flow (simplified)



The flow (simplified)



Batching! (Redis as Buffer)

The Problem

The Options

The Solution

The Problem

```
#jobs = #users * #branches
```

:(

Batching Options

---

1 job per user per many branches

(many ~= 10)

1 job per branch per many users

(many ~= 100)

Before and After

---

1. Process event for user
2. Enqueue dispatch job for user for desired send time

1m users = 1m jobs

1. Process event for user
2. Add user to Redis set (Key: “branch_id:timestamp”)
3. (Enqueue batch job)
4. Batch job runs
5. Pull 200 users from Redis set
6. Enqueue dispatch job for those users for desired send time

1m users = 5000 jobs

Questions?

niki@appboy.com

(We’re hiring! appboy.com/careers)