AIRFLOW SUMMIT

Lessons learned for scaling up Airflow 3
in Public Cloud

Przemek Więch, Augusto Hidalgo, Michał Modras

3.0

Nice to meet you!

Przemek Więch

Software Engineer at Google in Warsaw since 2011

Cloud Composer team since 2019

PhD in Computer Science
from the Warsaw University of Technology

Nice to meet you!

Michał Modras

Engineering Manager @ Google Cloud Composer team since 2022

Member of Airflow Summit organization committee

Agenda

- Performance testing framework
- Test results
    - Airflow 3 vs Airflow 2
    - Tweaking various parameters
- Conclusions

Performance Framework

3.0

Goals

- Compare Airflow 2 and Airflow 3
- Test scaling limits

- Make the tests
  - easy to run
  - reproducible

- Continuously monitor Airflow code performance

Testing framework – AIP-59

- Test performance of Airflow components
- Detect performance changes between Airflow versions
- Improve transparency of performance changes in Apache Airflow releases
- Identify changes that have impact on performance
- Let users test their setup

Testing framework - concepts

- **Instance** – definition of Airflow installation setup
- **Performance DAG** – definition of DAG that is executed during the test
- **Test suite** – combination of Instance and Performance DAG

Testing framework - instance

- Instance type – e.g. Docker Compose, Kubernetes
- number of schedulers and workers
- component CPU and memory
- machine types

Instance configuration file can include Jinja style placeholders that are populated in Test suite
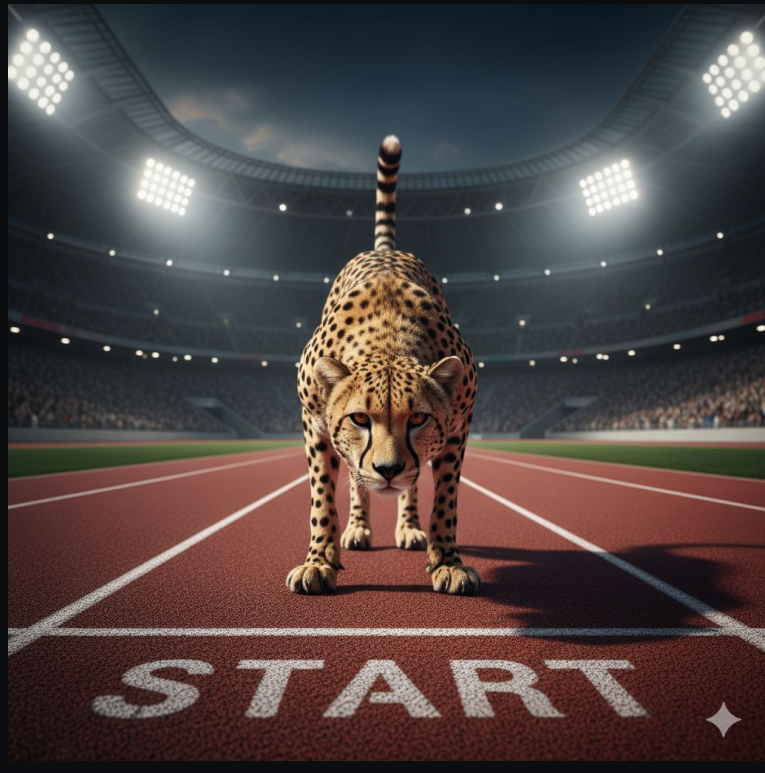
Instance config

Sample configuration with template parameters

```
{
  "environment_type": "VANILLA_GKE",
  "project_id": "{{project_id}}",
  "airflow_image_tag": "{{airflow_image_tag}}",
  "docker_image": "{{docker_image}}",
  "env_variable_sets": {{env_variable_sets}},
  "helm_chart_sets": {{helm_chart_sets}},
  "force_routing": {{force_routing}},
  "values_file": "{{values_file}}",
  "cluster_config": {
    "name": "{{cluster_id}}",
    "locations": [
      "{{zone_id}}"
    ],
    "node_pools": [
      {
        "name": "default-pool",
        "config": {
          "machine_type": "{{machine_type}}",
          "disk_size_gb": {{disk_size_gb}},
          "oauth_scopes": ["https://www.googleapis.com/auth/cloud-platform"],
        },
        "initial_node_count": {{node_count}}
      }
    ],
    "default_max_pods_constraint": {
      "max_pods_per_node": {{max_pods_per_node}}
    },
    "network_config": {
      "network": "projects/{{project_id}}/global/networks/{{network_id}}",
      "subnetwork": "projects/{{project_id}}/regions/{{location_id}}/subnetworks/{{subnetw
    },
    "ip_allocation_policy": {
      "use_ip_aliases": {{use_ip_aliases}}
    },
    "private_cluster_config": {
```

Testing framework - performance DAG

Example configuration options:

- PERF_DAG_FILES_COUNT
- PERF_DAGS_COUNT
- PERF_TASKS_COUNT
- PERF_SHAPE – "no_structure", "linear", "binary_tree", "star", "grid"
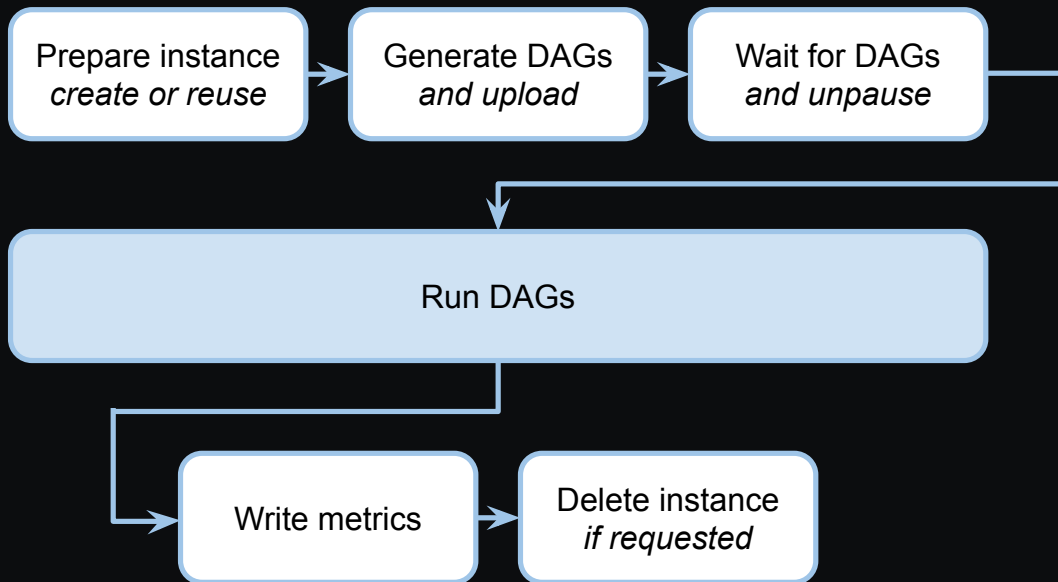- PERF_OPERATOR_TYPE – "python", "bash", etc.

Test suite

- multiple tests in a single "study"

- load template values from file

- override at any level

```json
{
  "study_components": [
    {
      "component_name": "vanilla_gke_airflow_2",
      "args": {
        "environment_specification_file_path": "configurations/config_generic.json",
        "jinja_variables": {
          "airflow_image_tag": "2.10.5"
        }
      }
    },
    {
      "component_name": "vanilla_gke__airflow_3",
      "args": {
        "environment_specification_file_path": "configurations/config_generic.json",
        "jinja_variables": {
          "airflow_image_tag": "3.0.0"
        }
      }
    }
  ],
  "default_args": {
    "results_project_id": "airflow-performance-tests",
    "results_dataset": "airflow-performance-tests-dataset",
    "jinja_variables": {
      "location_id": "us-central1",
      "machine_type": "n1-standard-8",
      "node_count": "6",
      "values_file": "values/environment.yaml"
    }
  },
  "default_flags": [
    "delete_if_exists",
    "delete_upon_finish"
  ],
  "default_attempts": 3
}
```

Testing framework - test suite

Prepare instance *create or reuse* → Generate DAGs *and upload* → Wait for DAGs *and unpause* → Run DAGs → Write metrics → Delete instance *if requested*

Results – metrics

- Resource utilization
    - CPU
    - Memory
    - Network

- Monitored components
    - Worker
    - Scheduler
    - Triggerer
    - database

- Durations
    - Total time
    - DAG run time
    - Task run time

AIRFLOW SUMMIT

How are the results stored?

- CSV files
- BigQuery tables

- Low-level metrics
- High-level statistics

Row	id	run_at	configuration	configuration_id	airflow_sched...	airflow_sc...	airflow_sche...	airflow_sch...	airflow_sche...	airflow_sched
2	415...	2025-09-24T09:15:31	{"AIRFLOW__CORE__STORE_SERIALIZED_DAGS": "Unknown", "airflow_version": "3.0.0", "composer_api_endpoint":	5bb3ea69238...	0.442061658...	0.44	112390144.0	112.39 MB	695173120.0	695.17 MB
3	8a2...	2025-09-26T16:46:09	{"AIRFLOW__CORE__STORE_SERIALIZED_DAGS": "Unknown", "airflow_version": "2.10.5", "composer_api_endpoint":	ecfcf653894a...	1.1774829652...	1.18	155245468.9...	155.25 MB	941590098.5...	941.59 MB
4	3c5...	2025-09-24T09:15:...	{"AIRFLOW__CORE__STORE_SERIALIZED_DAGS": "Unknown", "airflow_version": "2.10.5", "composer_api_endpoint":	fd6eb285e31...	0.423571309...	0.42	155912192.0	155.91 MB	833375232.0	833.38 MB
5	b5ff...	2025-09-22T14:57:...	{"AIRFLOW__CORE__STORE_SERIALIZED_DAGS": "Unknown", "airflow_version": "2.10.5", "composer_api_endpoint":	bf2afe8d8ab8...	0.596778456...	0.60	103871093.1...	103.87 MB	582683761.7...	582.68 MB
6	93a...	2025-09-25T09:02:19	{"AIRFLOW__CORE__STORE_SERIALIZED_DAGS": "Unknown", "airflow_version": "3.0.0", "composer_api_endpoint":	27493d941c1...	1.7942445586...	1.79	110429440.0	110.43 MB	1011907072.0	1.01 GB

Performance Test Results

3.0

Baseline scenario

Instance setup

- Google Kubernetes Engine (GKE)
- 6 workers (2 vCPU, 7.5 GB memory, 20 GB disk)
- 2 schedulers (1 vCPU, 4 GB memory, 5 GB storage)
- Postgres database

Tests overview

- Airflow 3.0.0
- Airflow 2.10.5

- 10 DAGs with z tasks each ($n = 10z$)
  - n = 100, 1k, 5k, 10k
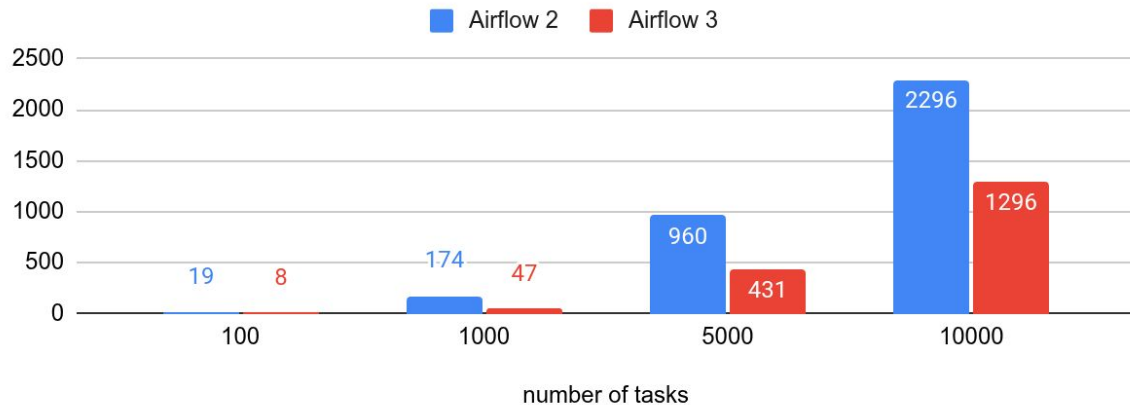 - No-op python operator tasks
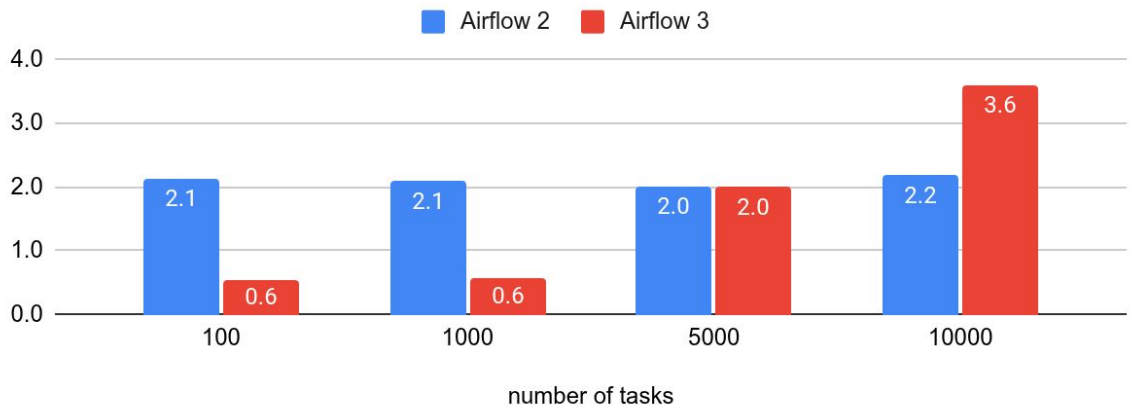
Airflow 2 vs Airflow 3

Test total duration

Task average duration
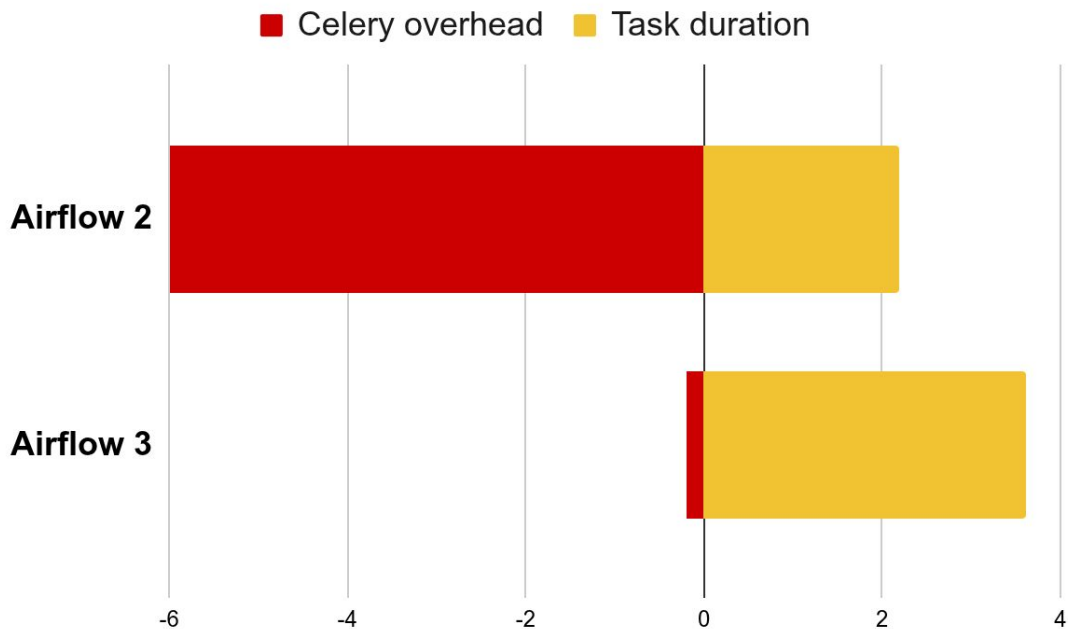
Airflow 2 vs Airflow 3

Memory consumption

Worker memory [GB]

■ Airflow 2 ■ Airflow 3

number of tasks	Airflow 2	Airflow 3
100	5.57	6.36
1000	6.58	6.82
5000	8.61	8.61
10000	9.57	10.15
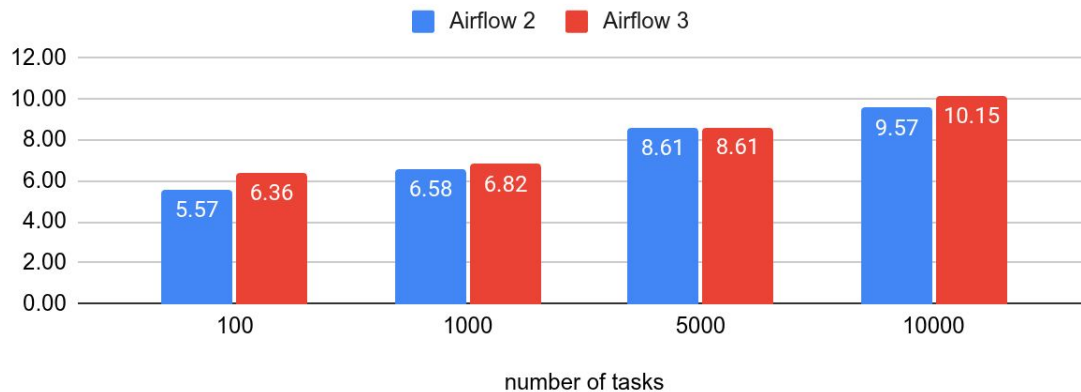
Scheduler memory [GB]

■ Airflow 2 ■ Airflow 3

number of tasks	Airflow 2	Airflow 3
100	0.52	0.48
1000	0.53	0.51
5000	0.58	0.65
10000	0.64	0.83
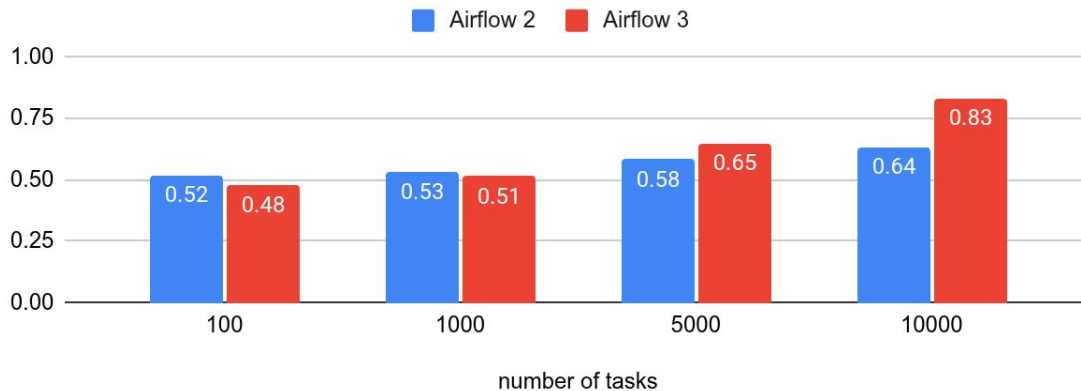
Airflow 2 vs Airflow 3

CPU usage

Worker CPU

■ Airflow 2 ■ Airflow 3

number of tasks	Airflow 2	Airflow 3
100	1.7	0.9
1000	5.5	5.5
5000	12.2	22.4
10000	16.9	28.3

number of tasks

Scheduler CPU

■ Airflow 2 ■ Airflow 3

number of tasks	Airflow 2	Airflow 3
100	0.3	0.3
1000	0.5	1.0
5000	0.6	1.6
10000	0.8	1.6

number of tasks
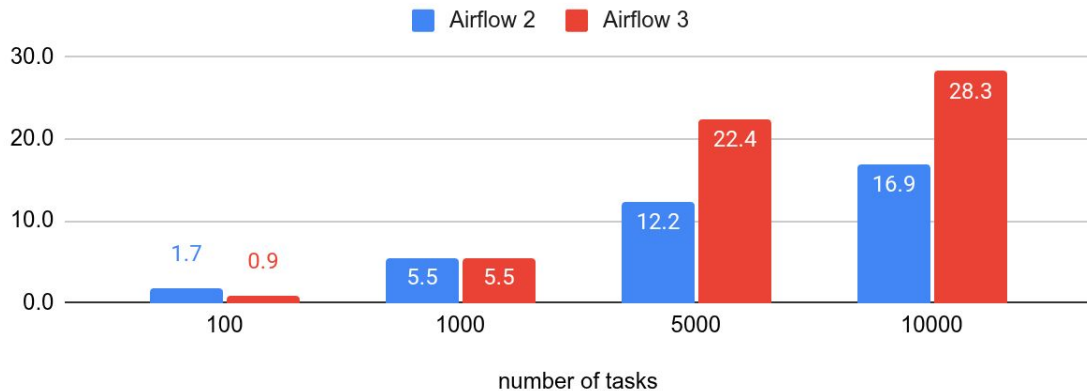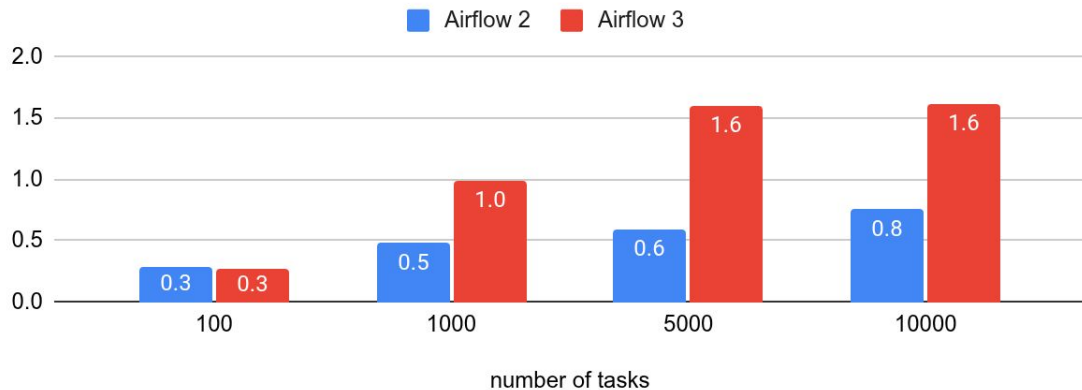
First conclusions

- **Airflow 3** overall performs better than Airflow 2
- **Airflow 3** utilizes resources better (faster to finish using more CPU)
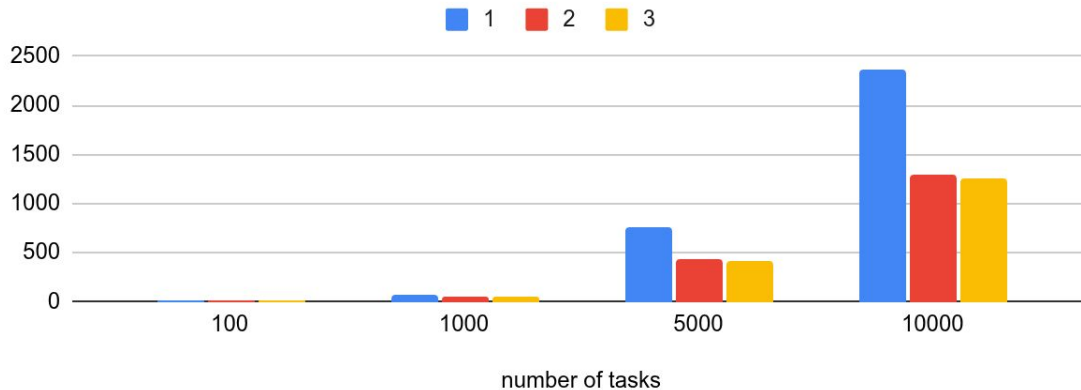- **Airflow 3** uses a bit more memory

Number of schedulers
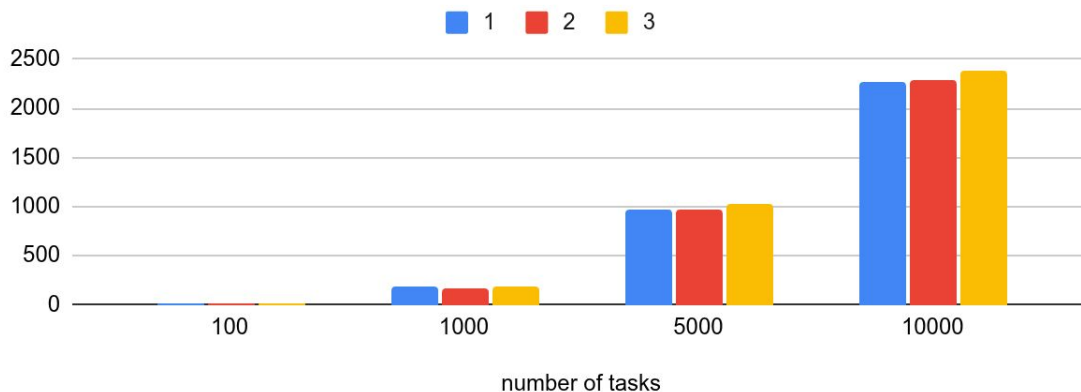
Running with 1, 2 or 3 schedulers

Conclusions

- **Airflow 3** benefits from having 2 schedulers

- **Airflow 2** does not benefit from having more than 1 scheduler

- No benefit from having third scheduler

Airflow 3 – test duration [seconds]

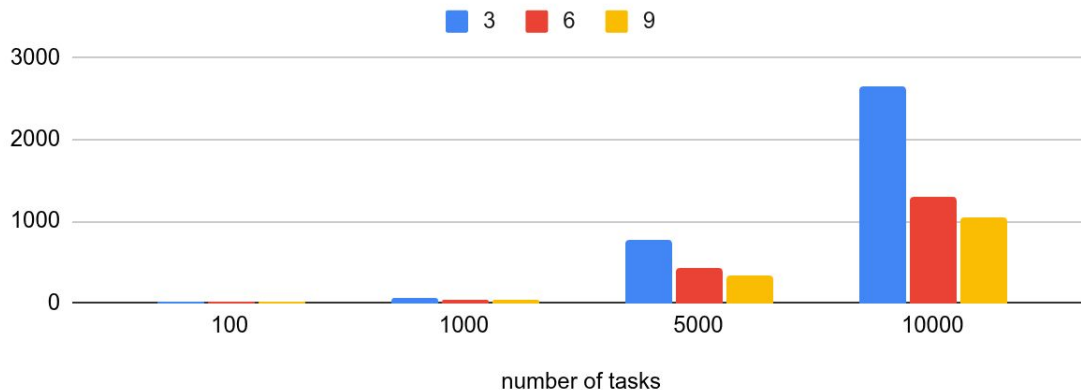## Airflow 2 – test duration [seconds]

Number of workers

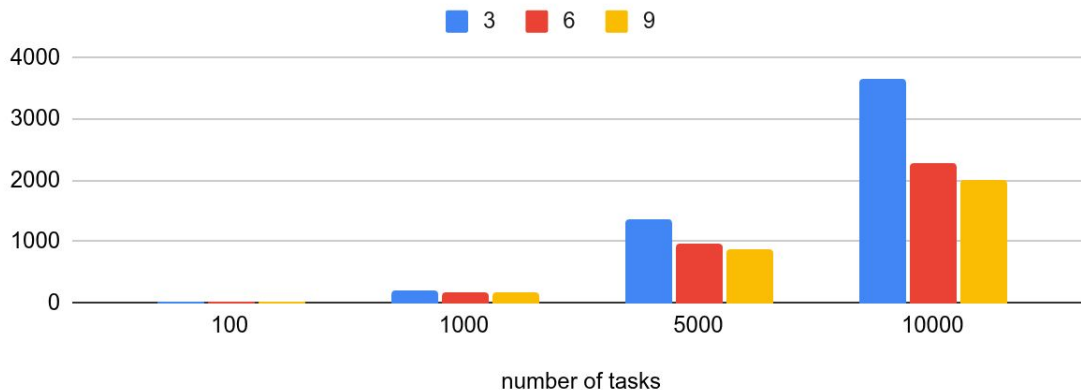Running with 3, 6 or 9 workers

Conclusions

- **Airflow 3** scales better with number of workers

- 3 → 6 workers – linear scaling
- 6 → 9 workers – smaller benefit

Airflow 3 – test duration [seconds]

Airflow 2 – test duration [seconds]

Conclusions

3.0

Conclusions

- Performance tests can show performance improvement and degradation
- The developed framework makes it possible to run various scenarios

- **Airflow 3** is more performant than **Airflow 2**
- **Airflow 3** better utilizes resources

- 3 schedulers are unnecessary, unless used for availability
- Scaling workers is linear until a certain point

Future

- Publish performance framework code
- Run performance tests on each release
- Run performance tests daily

Questions?

pwiech@oogle.com
michalmodras@google.com
augustoh@google.com