
AWS
re:Invent
NOV. 28 – DEC. 2, 2022 | LAS VEGAS, NV

API205

Accelerating workloads using parallelism with AWS Step Functions & Lambda

Brian Zambrano (he/him/they)

Serverless Specialist
Solutions Architect
AWS

Justin Callison (he/him/they)

General Manager,
AWS Step Functions
AWS

Adam Wagner (he/him/they)

Principal Serverless Specialist
Solutions Architect
AWS

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

- Data processing
- Step Functions overview
- Parallel data processing
- Distributed Map overview
- Distributed Map demo
- Distributed Map deep dive
- When to use Distributed Map

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Data processing has evolved



Spreadsheets



Bespoke code

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Data processing is evolving



Spreadsheets



Code (developer)

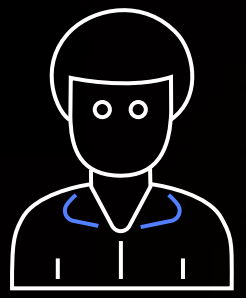


Big data frameworks

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Data processing is evolving



Spreadsheets



Code (other)



Code (developer)



Big data frameworks

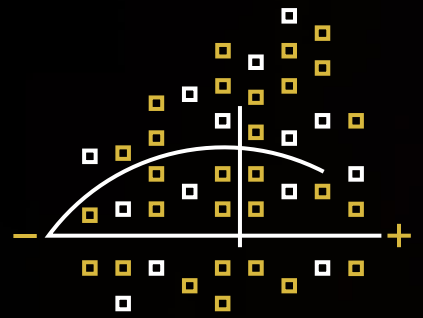
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Step Functions overview

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS Step Functions is a serverless workflow service



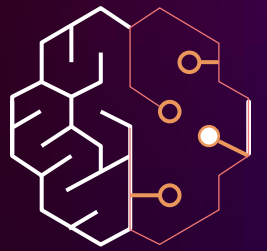
Data processing



Application orchestration



Security and IT automation



Machine learning

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Directly compose applications from over 220 AWS services and 10,000 API actions



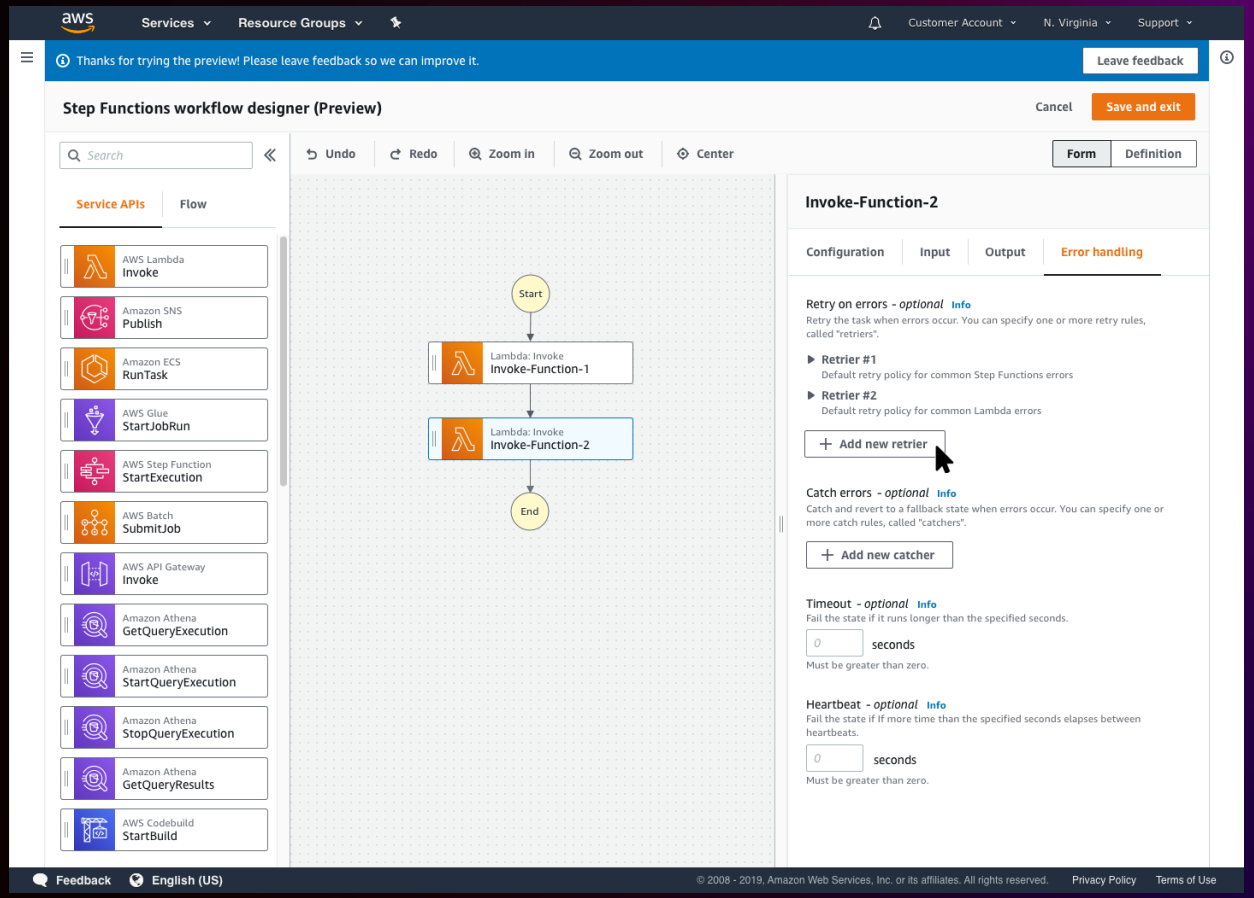
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS Step Functions

The workflows you build with Step Functions are called state machines, and each step of your workflow is called a state

When you execute your state machine, each move from one state to the next is called a state transition

You can reuse components, easily edit the sequence of steps, or swap out the code called by task states as your needs change



AWS STEP FUNCTIONS WORKFLOW STUDIO

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Integration types to suit your needs



Optimized integrations

Customized to simplify the usage of 17 AWS services

Supported integration patterns:

- Request response
- Wait for a callback (.waitForTaskToken)
- Run a job (.sync)

AWS SDK integrations

Call 220 AWS services directly (10,000+ API actions)

Supported integration patterns:

- Request response
- Wait for a callback (.waitForTaskToken)

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

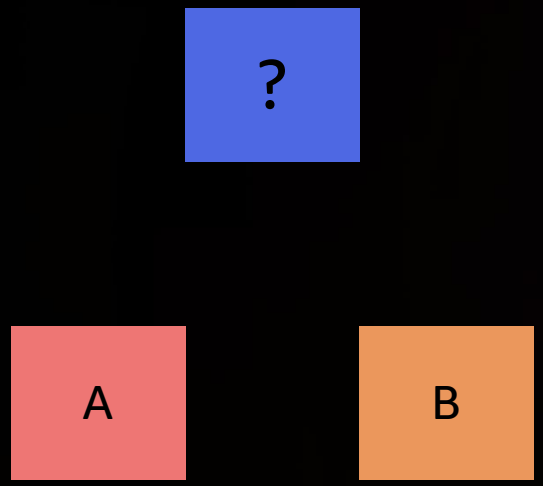
Select the best workflow for the job

Standard workflows	Express workflows
Ideal for **long-running**, **durable**, and **auditable** workflows	Ideal for **high-volume**, **event-processing**, and **short duration** workflows
Duration up to 1 year	Duration <5 minutes
Free to wait	
Pricing = Per state transition Free tier: 4,000 state transitions/mo	Pricing = Request + Duration

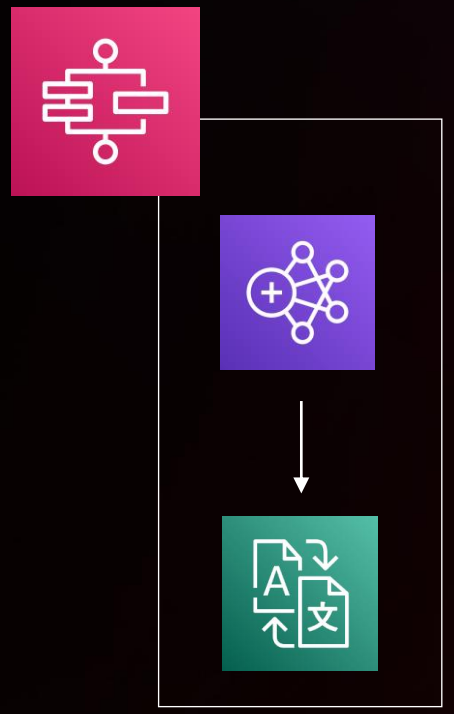
…or use together

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

States allow you to make decisions, perform actions, and accelerate processing



Choice state



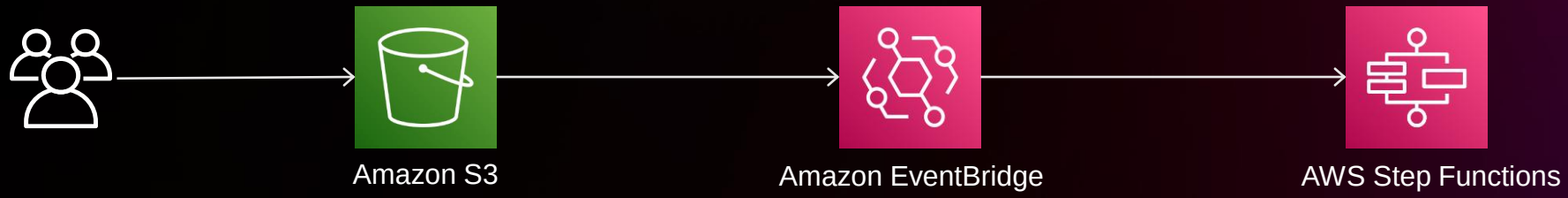
Invoke services, including cross-account



Parallel processing

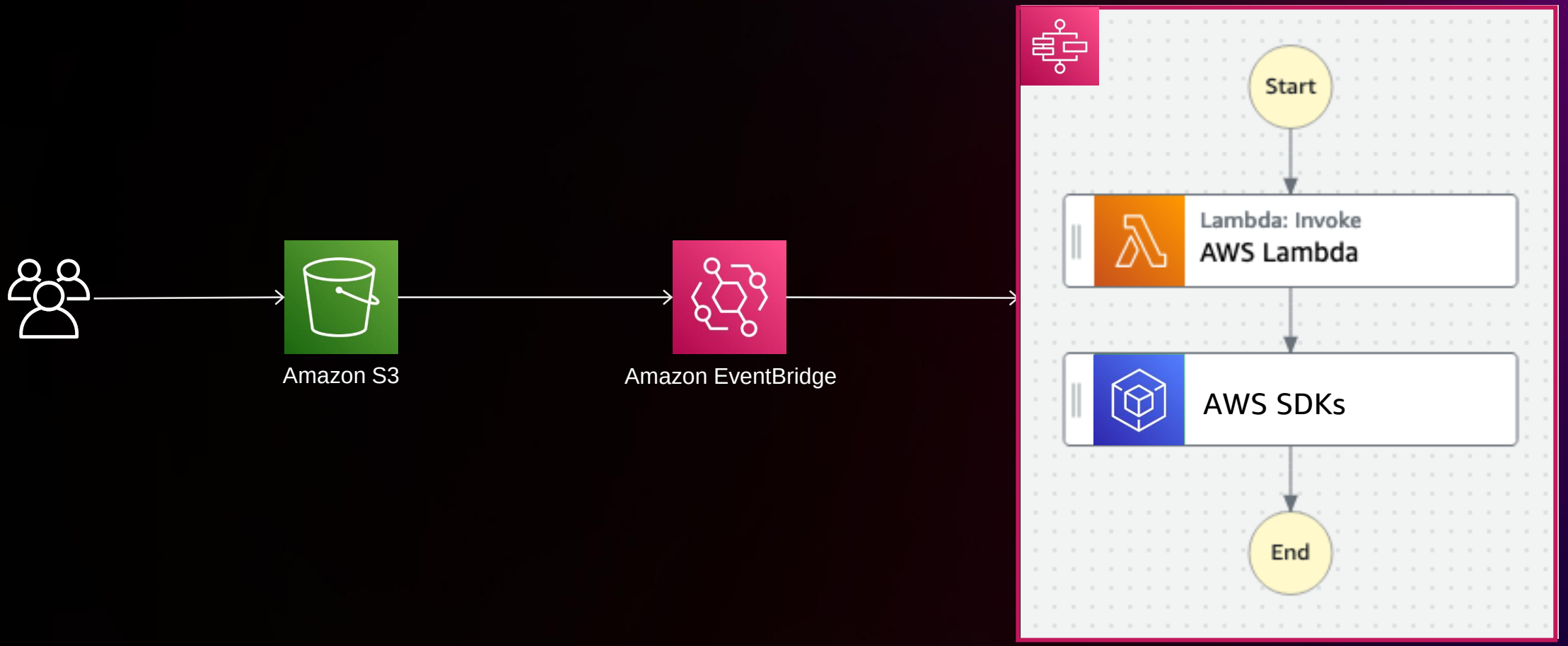
aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Processing an object



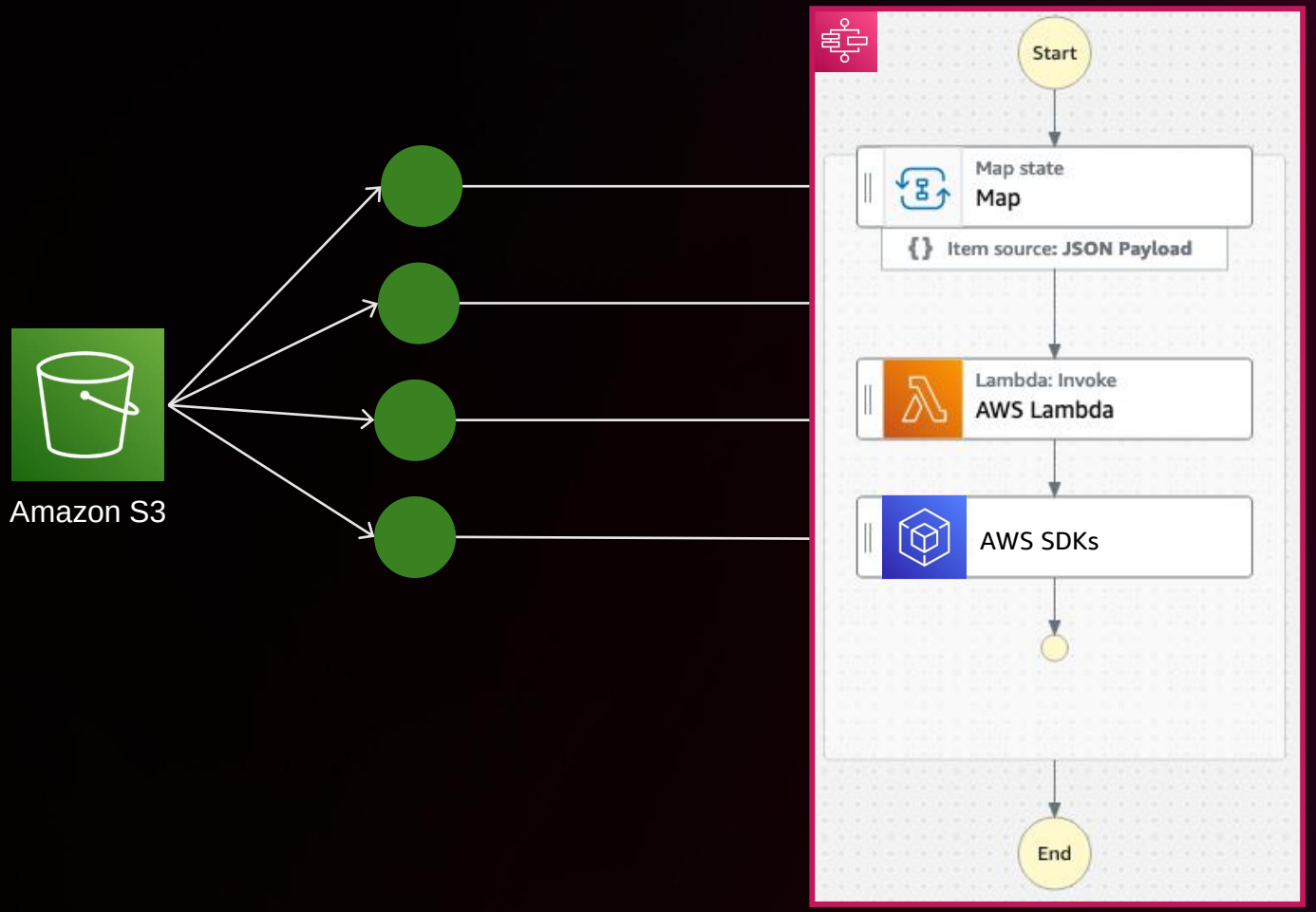
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Processing an object



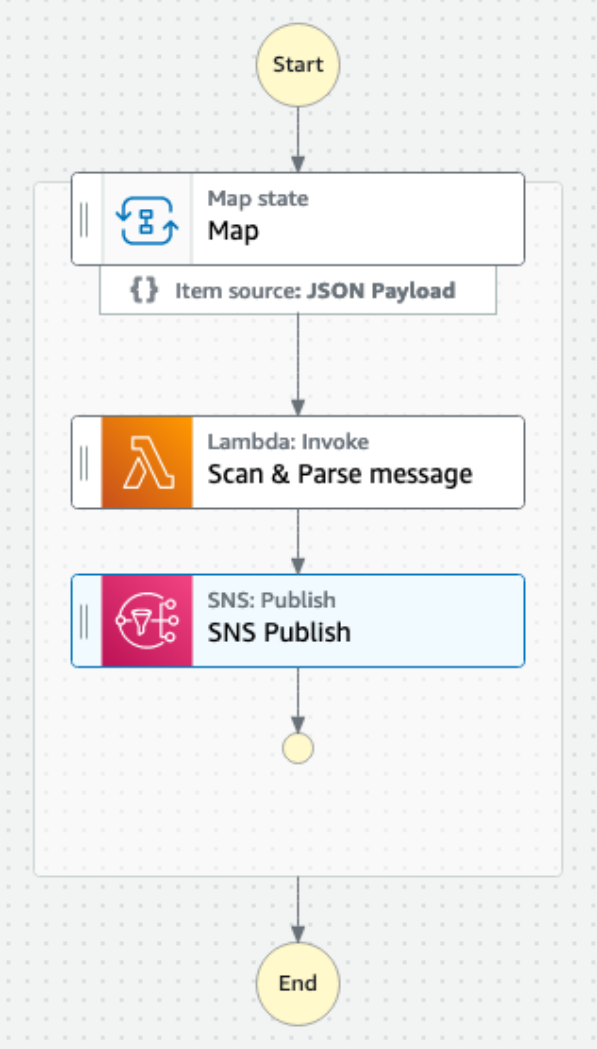
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Processing many objects



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

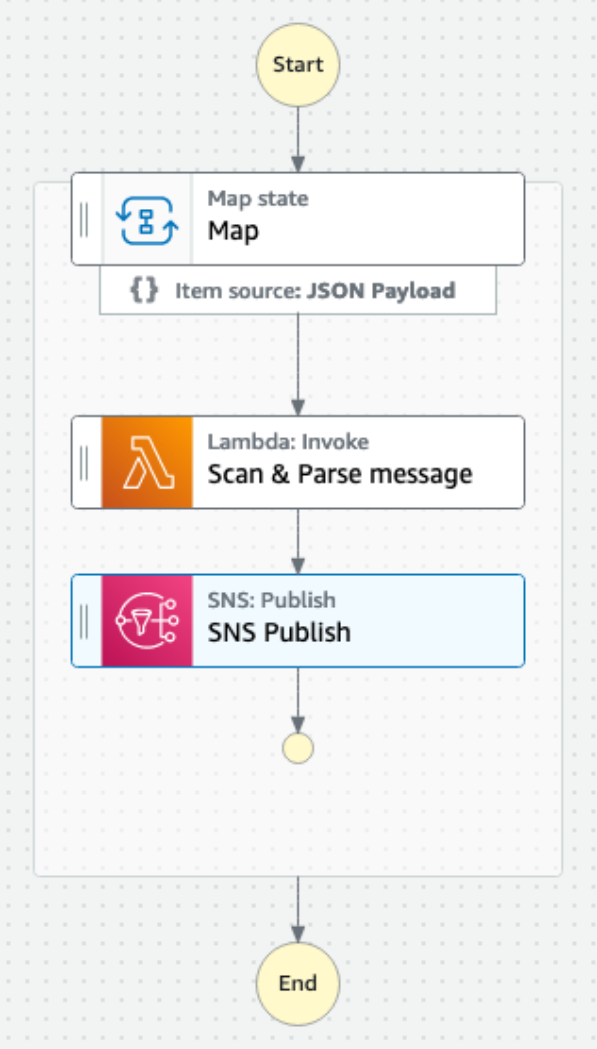
Inline Map state



- Enables dynamic parallelism
- Execute the same workflow steps for a collection of items
- Achieve parallel processing without the complexity

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Inline Map state



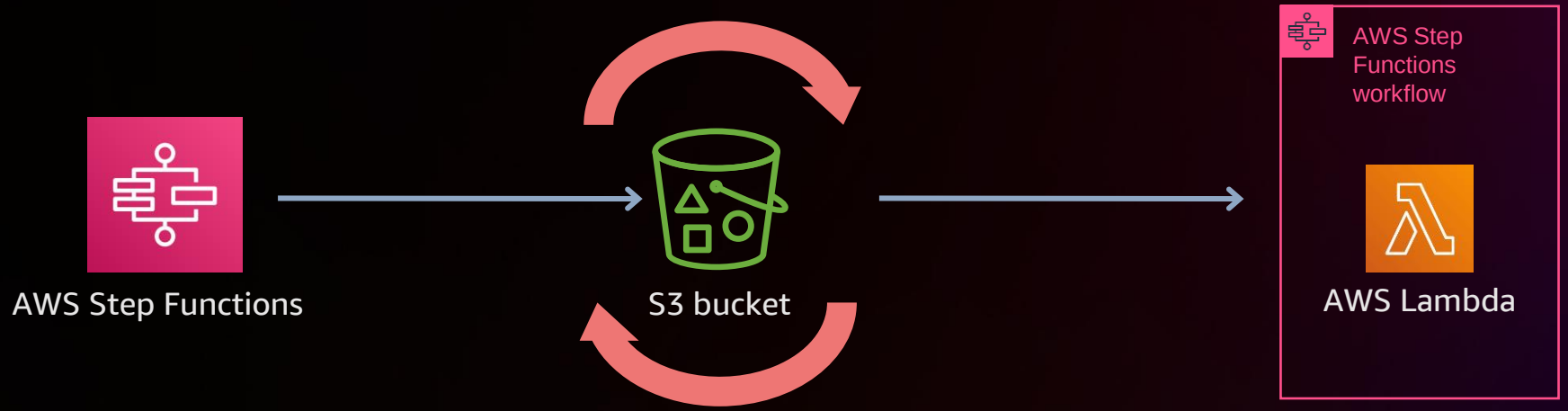
- Map iterations are started within the same execution history, which has a 25K limit
- Supports up to 40 concurrent iterations
- Input and output must be under 256 KB

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Distributed Map

Unrivalled parallelism for large-scale workloads

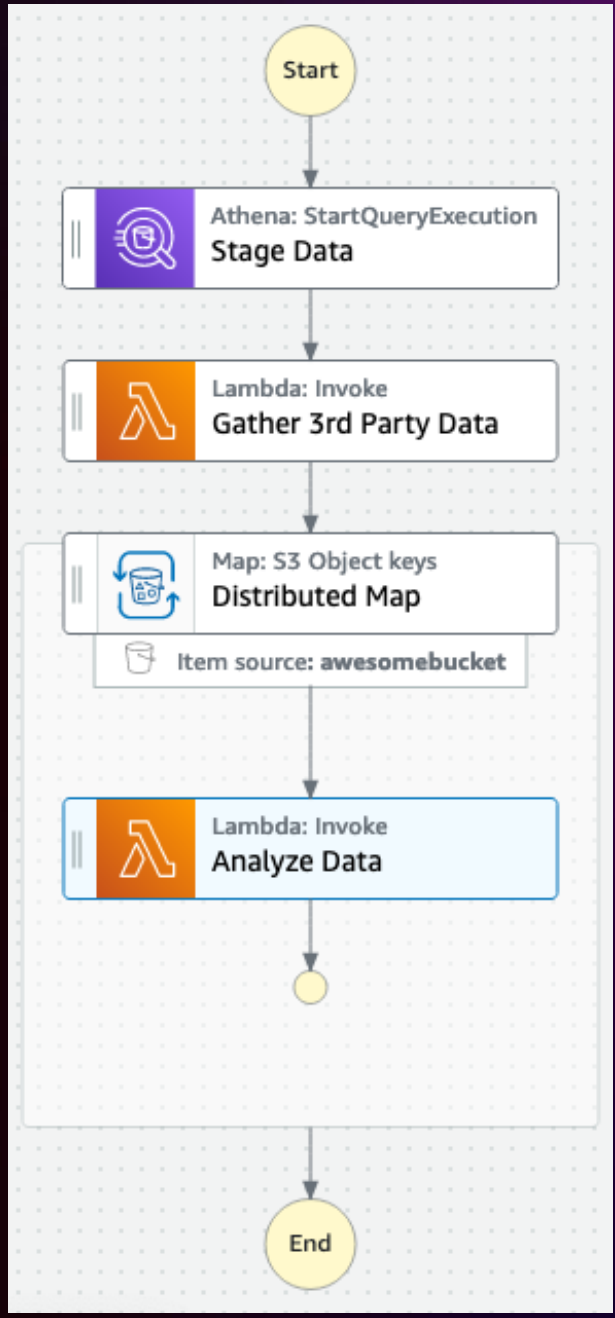
- Fan-out/fan-in pattern for **high-concurrency** workloads
- Up to **10K** concurrency
- Iterate over objects in an **S3 bucket**, process using a workflow
- Operator **dashboard** for tracking progress
- Works with **Express or Standard** Workflows



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

The new distributed mode for Map state

- Iterations of the Map state are invoked as individual state machine executions, avoiding 25K history event limits for the parent state machine
- Iterate over objects, S3 Inventory manifest, or JSON/CSV file in S3 as input and export execution outputs to S3 to bypass input and output limits of 256 KB
- Specify MaxConcurrency to control throughput and avoid overwhelming other services
- Control your workload with batch sizes and error thresholds, and choose between Standard or Express Workflows for your iterations



aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

The new distributed mode for Map state

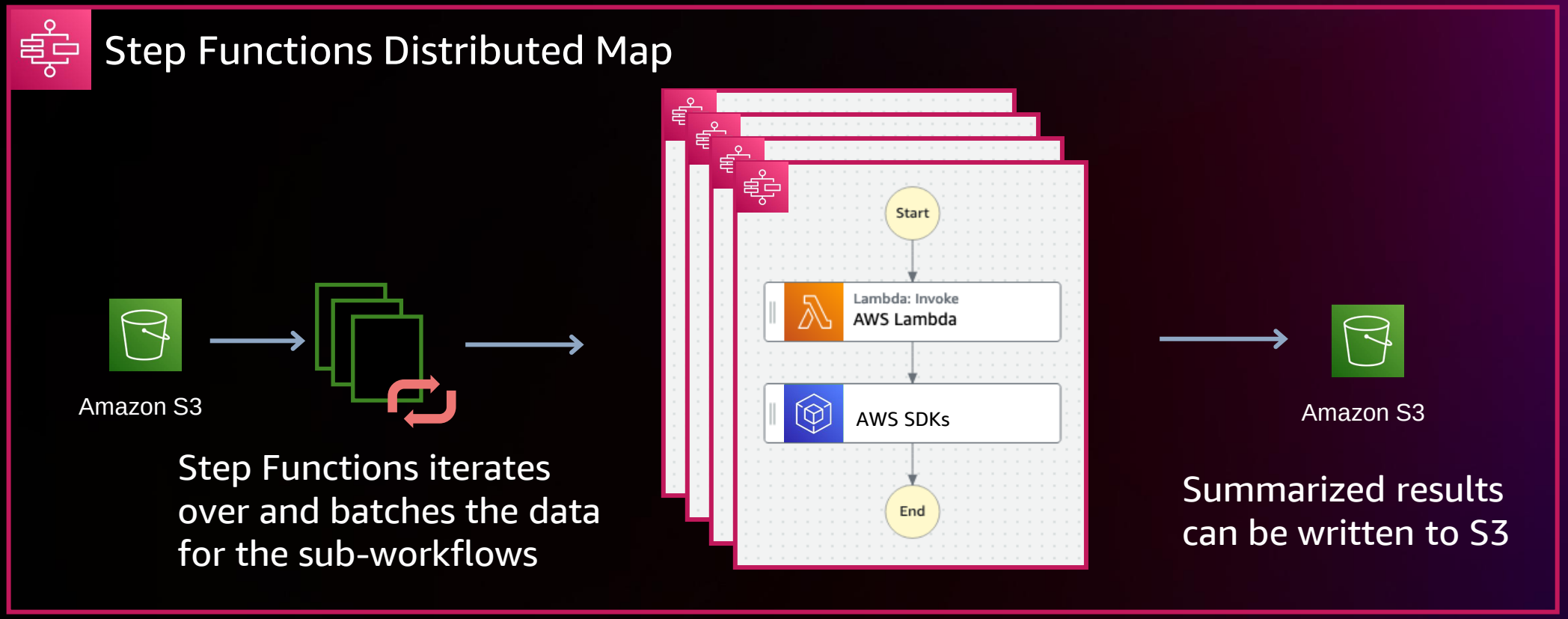
- Iterations of the Map state are invoked as individual state machine executions, avoiding 25K history event limits for the parent state machine
- Iterate over objects, S3 Inventory manifest, or JSON/CSV file in S3 as input and export execution outputs to S3 to bypass input and output limits of 256 KB
- Specify MaxConcurrency to control throughput and avoid overwhelming other services
- Control your workload with batch sizes and error thresholds, and choose between Standard or Express Workflows for your iterations

```
"Map": {
  "Type": "Map",
  "MaxConcurrency": 5,
  "End": true,
  "ItemBatcher": {
    "MaxItemsPerBatch": 1,
  },
  "ItemReader": {
    "Resource": "arn:aws:states:::aws-sdk:s3:listObjectsV2",
    "Parameters": {
      "Bucket": "mydataset",
      "Prefix": "pictures_dataset"
    }
  },
  "ItemSelector": {
    "index.$": "$$.Map.Item.Index",
    "value.$": "$$.Map.Item.Value"
  },
  "ResultWriter": {
    "Resource": "arn:aws:states:::aws-sdk:s3:putObject",
    "Parameters": {
      "Bucket": "outputdata",
      "Key": "test_1"
    }
  },
  "ItemProcessor": {
    "ProcessorConfig": {
      "Mode": "DISTRIBUTED",
      "ExecutionType": "EXPRESS"
    },
    "StartAt": "Process Data",
    "States": {
      ...
```

https://states-language.net/spec.html

aws © 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Processing many objects

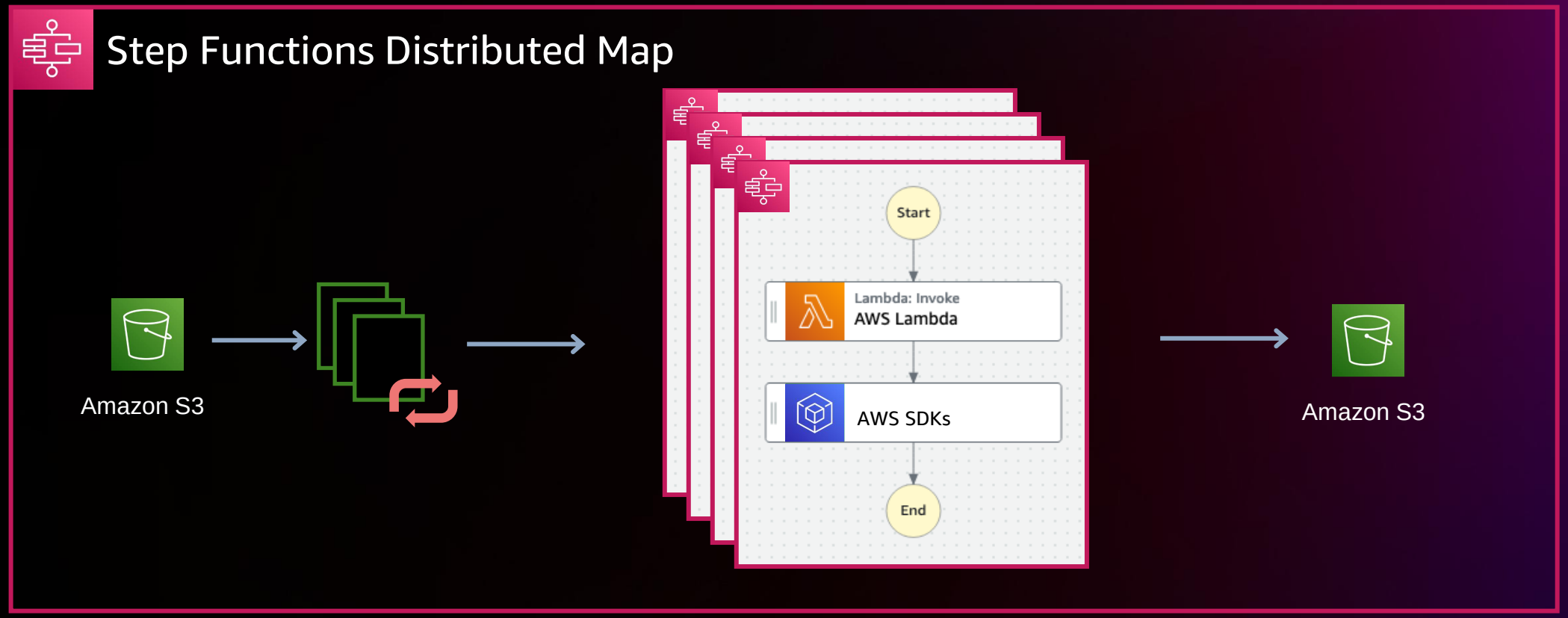


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Deep dive into Distributed Map

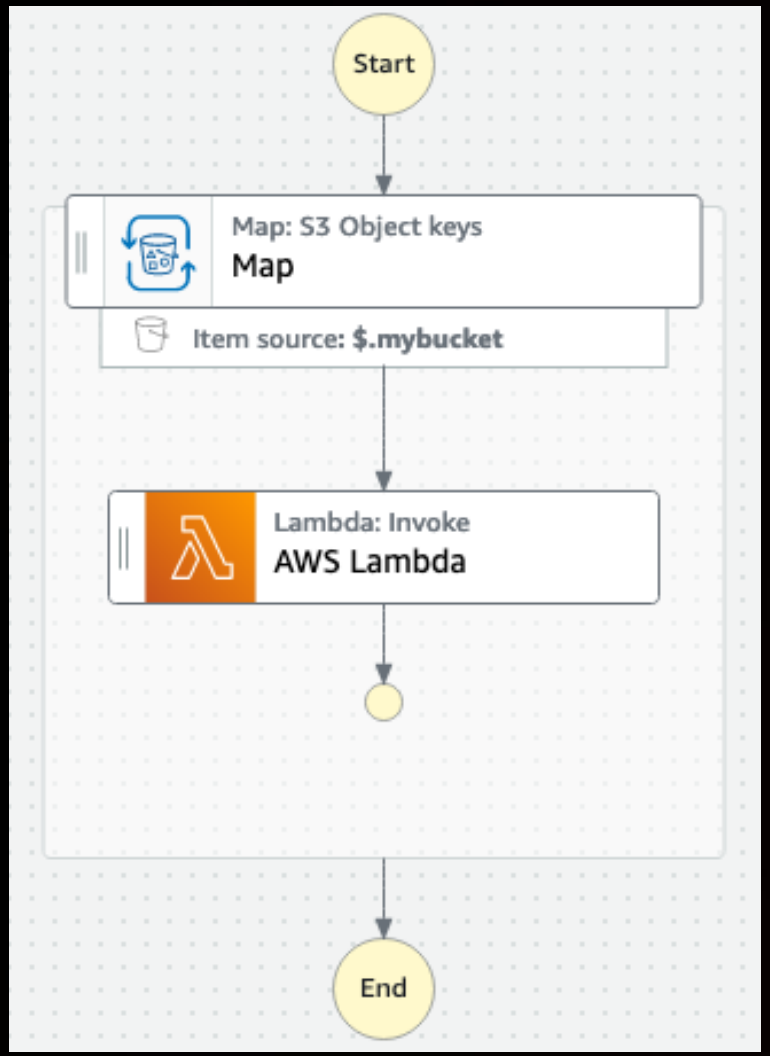
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Distributed Map



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

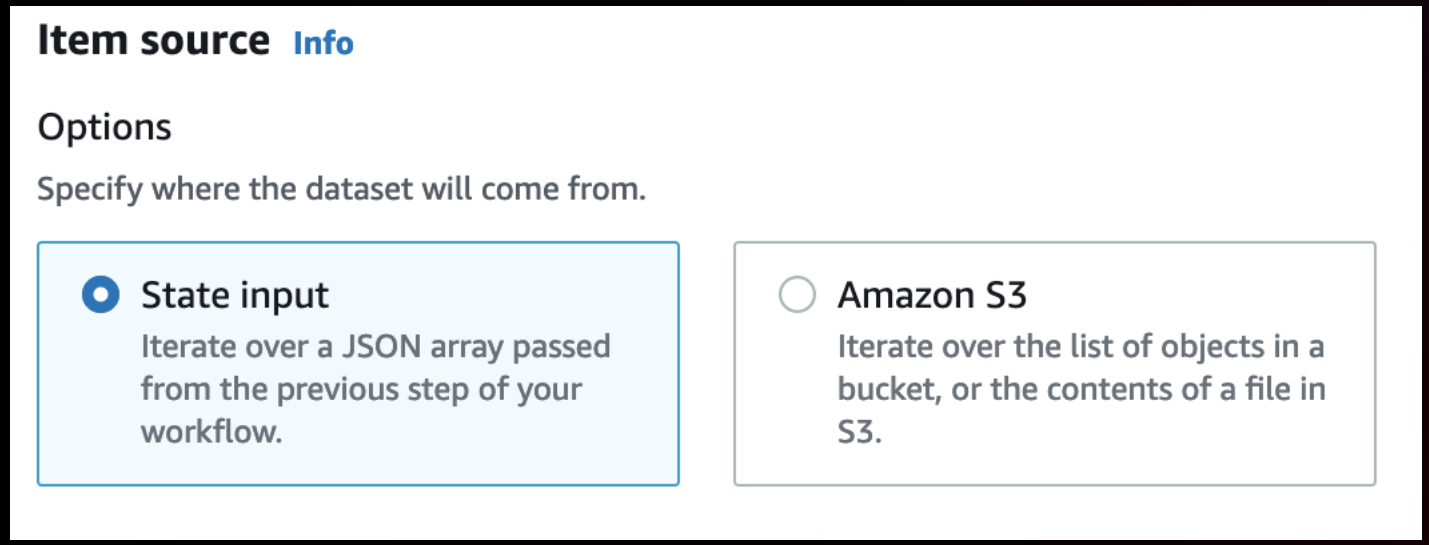
Distributed Map – Input options



- State input
  - Iterate over a JSON input from previous step
- S3 objects
  - List objects
  - S3 Inventory file (CSV only)
- S3 object contents
  - JSON file
  - CSV file

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

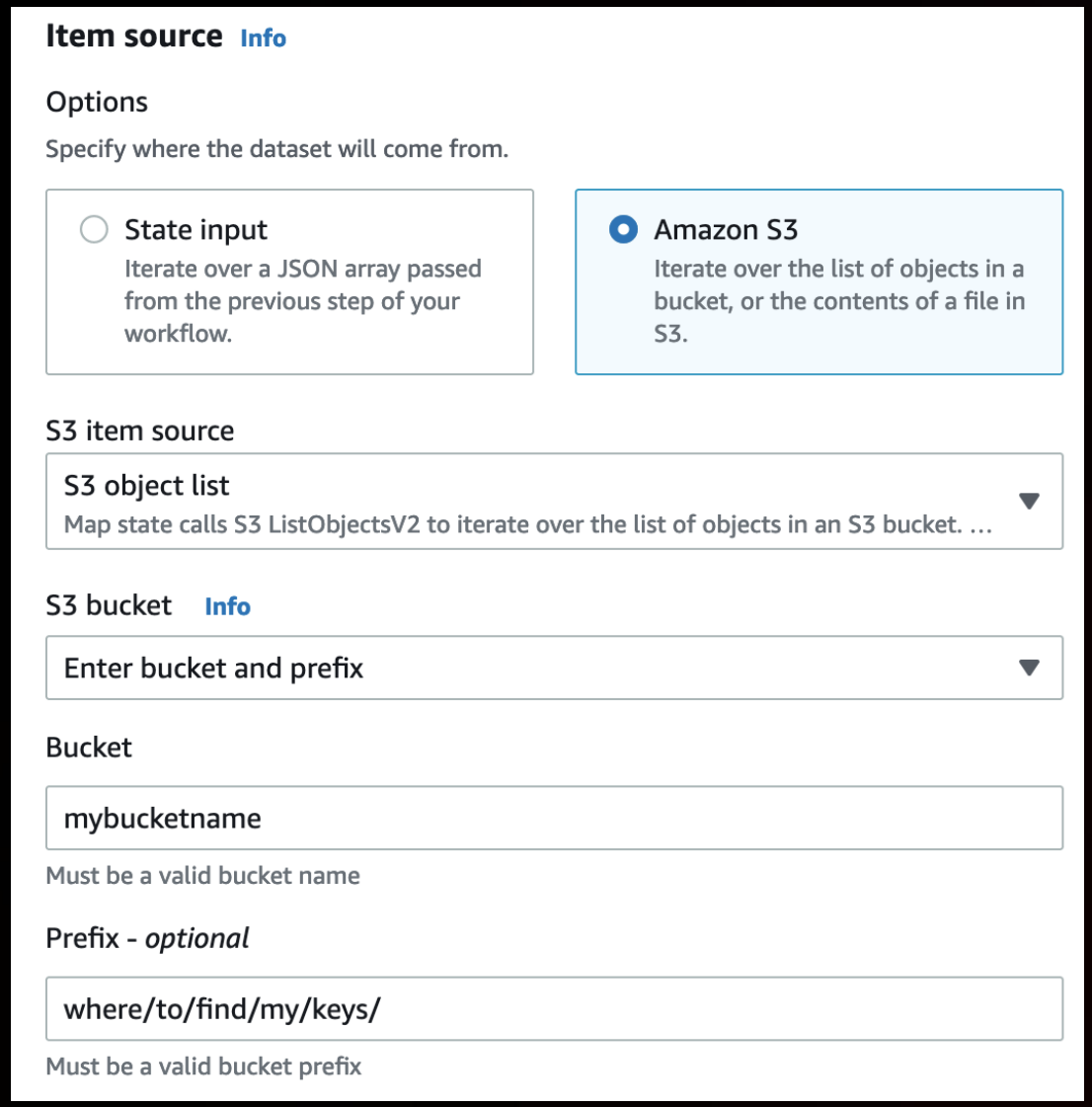
Distributed Map – State input



- Works the same way as Inline Map has before
- Pass an array of items in the state input to iterate over

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Distributed Map – S3 objects

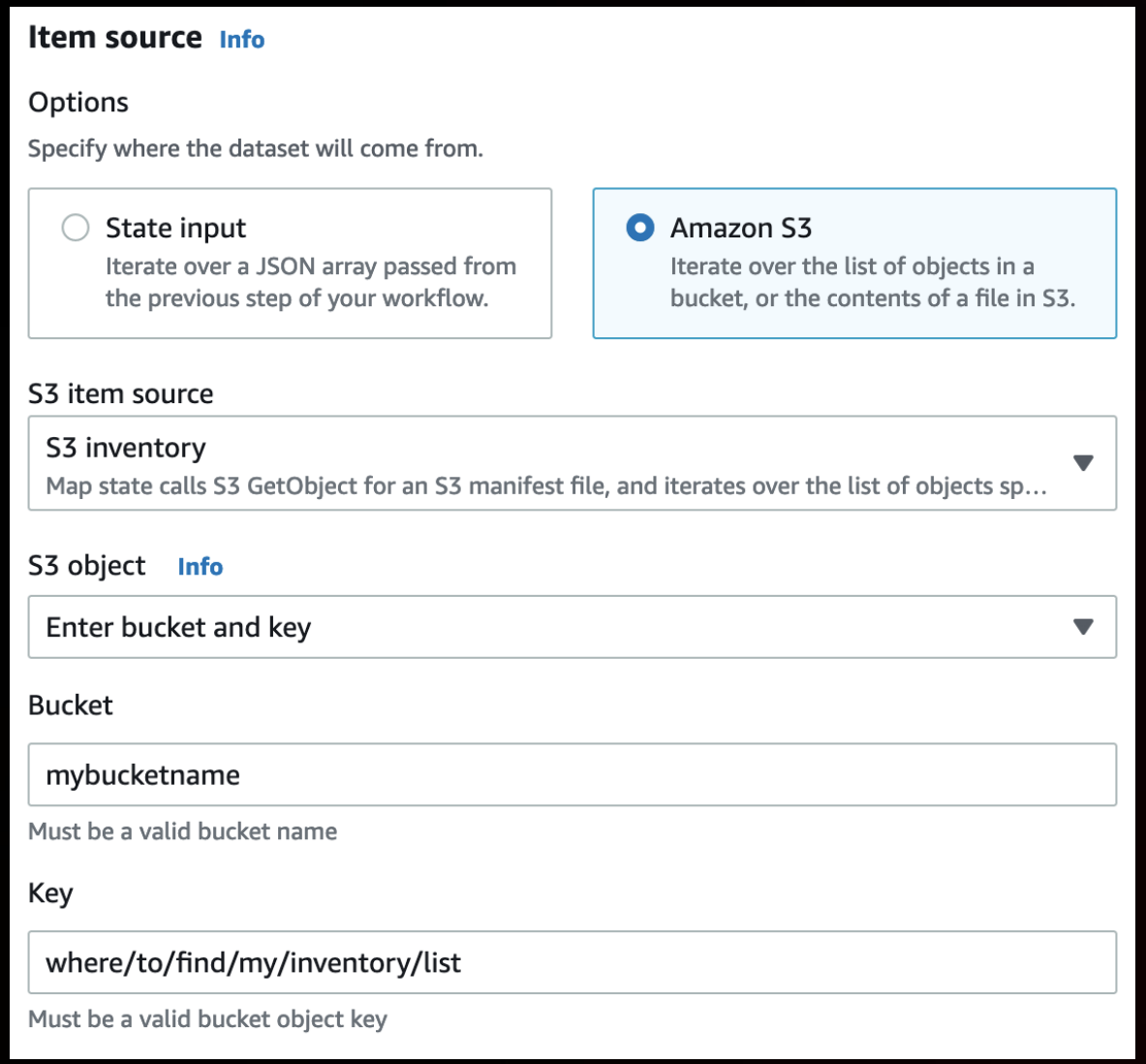



```
"ItemReader": {
  "Resource": "arn:aws:states:::s3:listObjectsV2",
  "ReaderConfig": {
    "MaxItems": 1000
  },
  "Parameters": {
    "Bucket": "mybucketname",
    "Prefix": "where/to/find/my/keys/"
  }
},
```

- List objects from an S3 bucket
- Specify the bucket name and prefix
- Can optionally limit the number of items

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Distributed Map – S3 objects




```
"ItemReader": {
  "Resource": "arn:aws:states:::s3:getObject",
  "ReaderConfig": {
    "MaxItems": 1000,
    "InputType": "MANIFEST"
  },
  "Parameters": {
    "Bucket": "mybucketname",
    "Key": "where/to/find/my/inventory/list"
  }
},
```

- Optionally use an S3 Inventory list
- Inventory list is a scheduled alternative to synchronous list API calls

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Distributed Map – S3 objects

Input

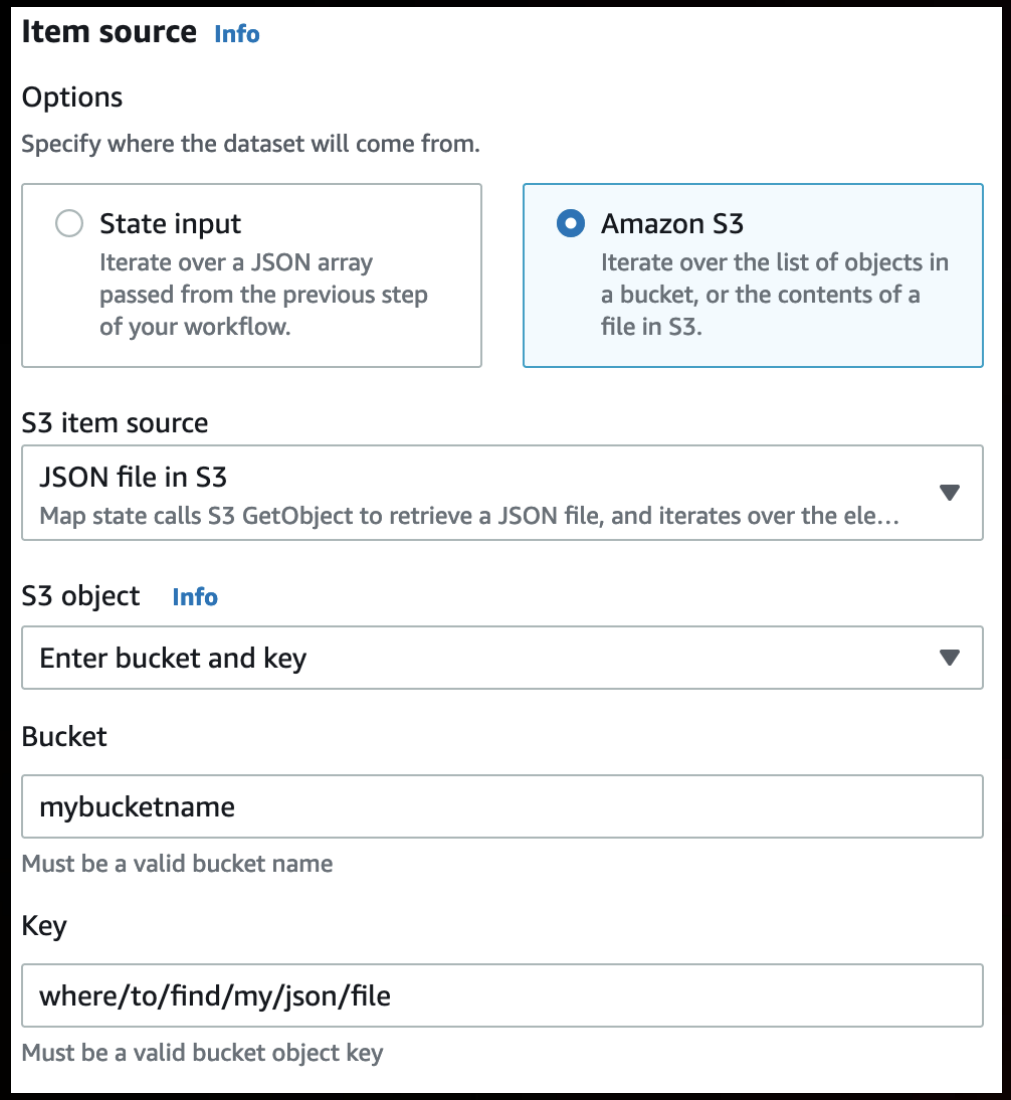

```
{
  "Etag": "\"5b925b0890d6b6eb59b6a205ffa85e4b\"",
  "Key": "data/simple-json-dataset/7_Tty4c.jpg.json",
  "LastModified": 1668886888,
  "Size": 23171,
  "StorageClass": "STANDARD"
}
```


- Passes the key to the object rather than the object contents
- List objects – lists all objects in a Bucket + Prefix
  - Output matches the S3 ListObjectsV2 API
  - Use ItemSelector to pass additional information such as the bucket name

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Distributed Map – S3 object contents

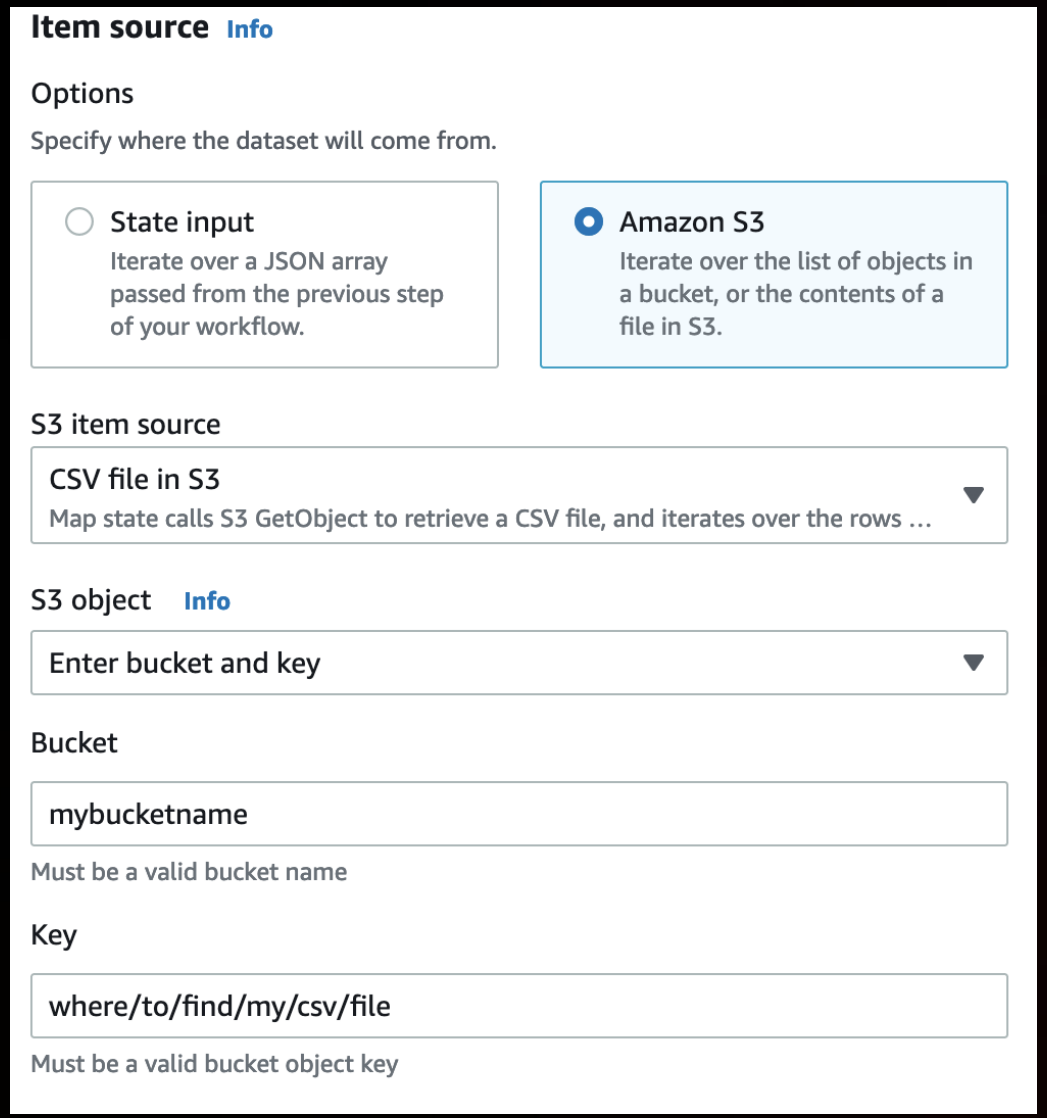



```
"ItemReader": {
  "Resource": "arn:aws:states:::s3:getObject",
  "ReaderConfig": {
    "InputType": "JSON"
  },
  "Parameters": {
    "Bucket": "mybucketname",
    "Key": "where/to/find/my/json/file"
  }
},
```

- JSON file containing an array of items

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Distributed Map – S3 object contents




```
"ItemReader": {
  "Resource": "arn:aws:states:::s3:getObject",
  "ReaderConfig": {
    "InputType": "CSV",
    "CSVHeaderLocation": "FIRST_ROW"
  },
  "Parameters": {
    "Bucket": "mybucketname",
    "Key": "where/to/find/my/csv/file"
  }
},
```

- CSV file containing one item per row

aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Processing S3 objects in Lambda

AWS Lambda Powertools for Python makes it easy to process large S3 objects

- Provides an easy-to-use library to fetch parts of an S3 object, while consuming a minimal amount of memory
- Supports decompressing GZIP and deserializing a CSV to a dictionary, as well as custom transformations

```python
from typing import Dict

from aws_lambda_powertools.utilities.streaming.s3_object import S3Object
from aws_lambda_powertools.utilities.typing import LambdaContext


def lambda_handler(event: Dict[str, str], context: LambdaContext):
    s3 = S3Object(bucket=event["bucket"], key=event["key"])
    for line in s3:
        print(line)
```



Only consumes this line of memory usage, not whole file

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

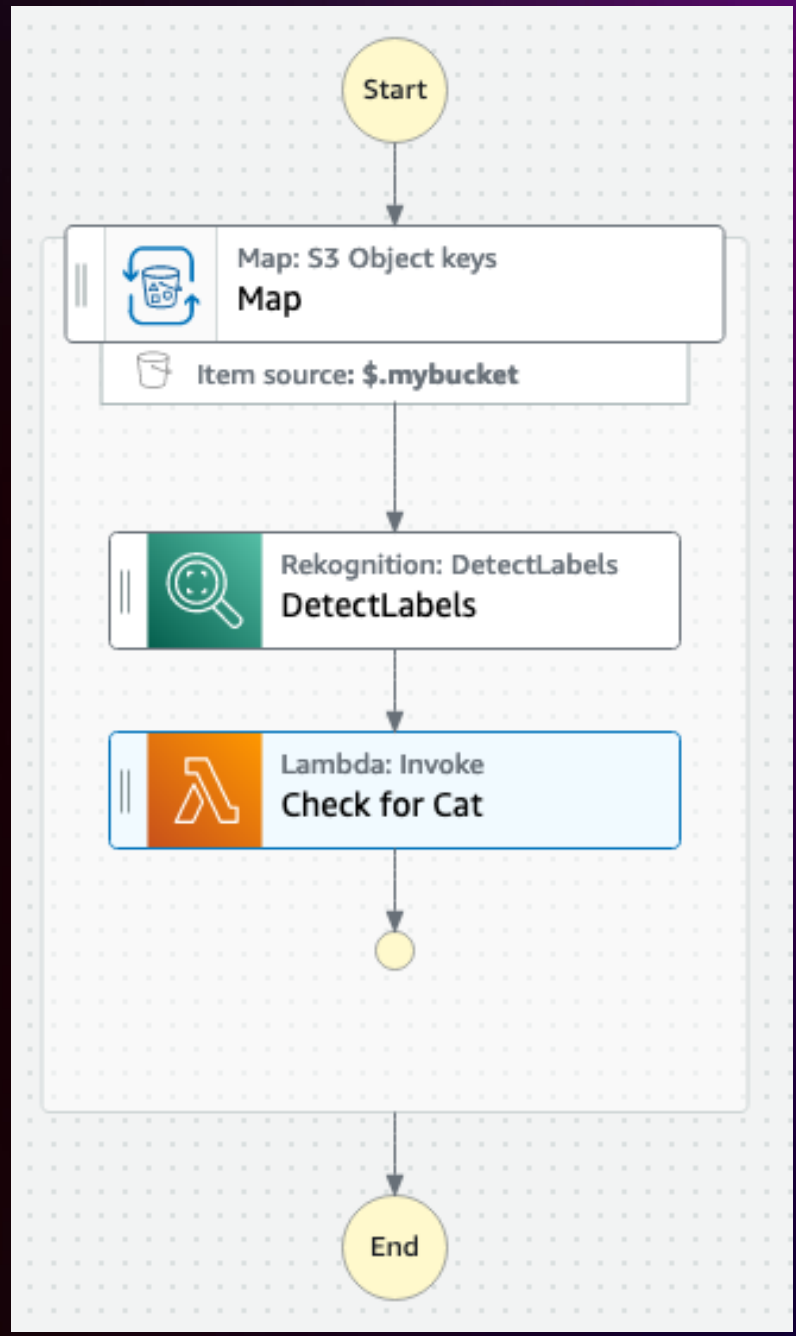
Concurrency

Distributed Map concurrency configuration

- Concurrency

Best practices

- Use concurrency to avoid overloading services that have lower API quotas
- Use the retry logic in Step Functions to retry throttled requests



aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Batching

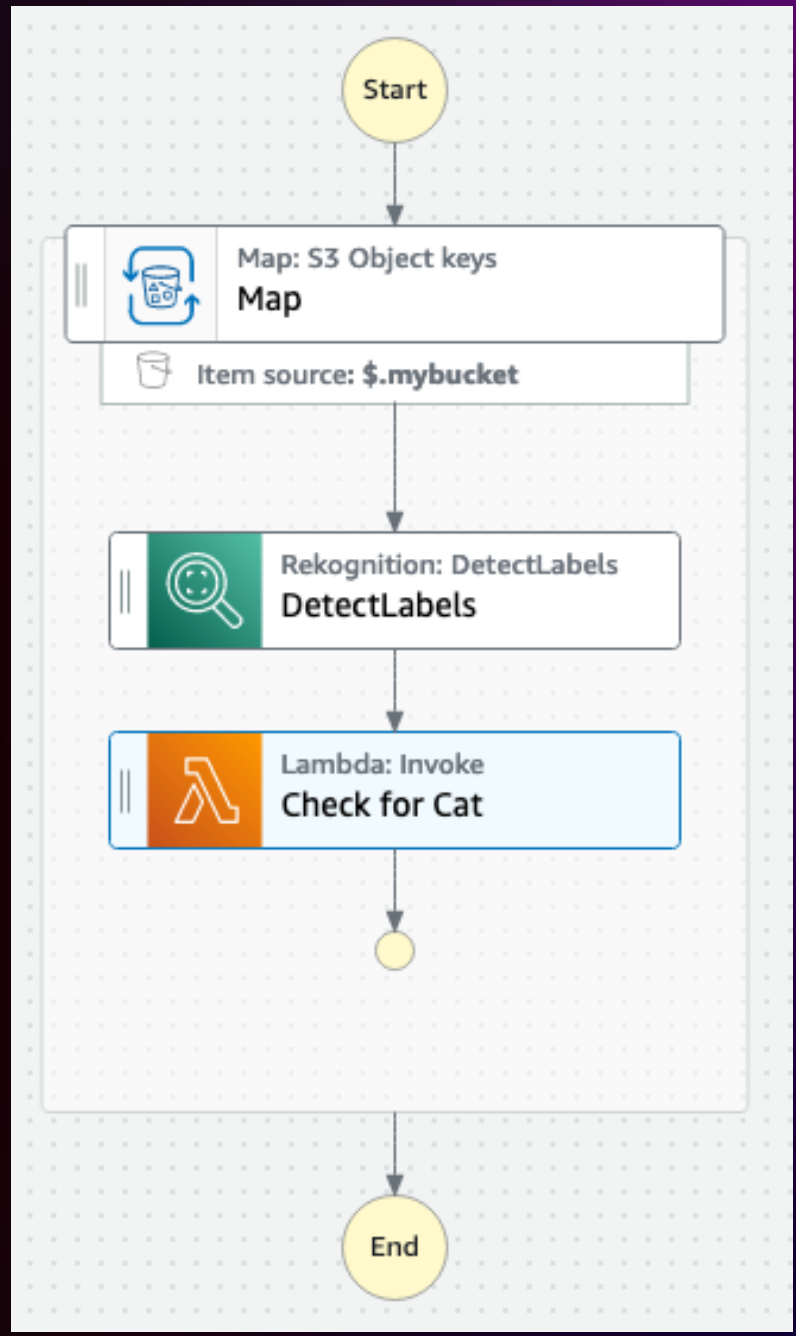
Distributed Map concurrency configuration

- Batch size
  - # of items
  - Size in KB

Use cases

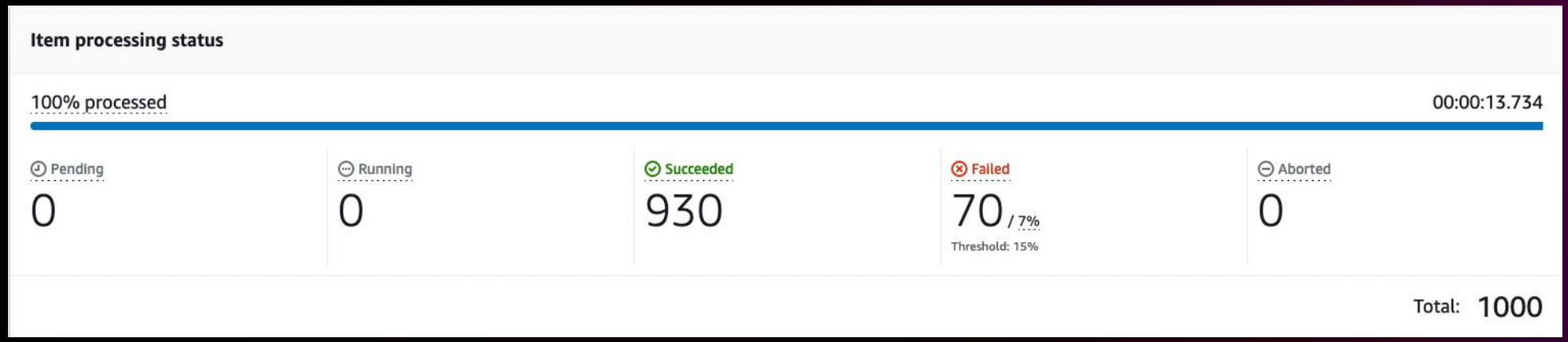
- Use batching to match the functional requirements of downstream systems

- Optimize the performance of your data processing by tuning the batch size



aws
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Handling failures

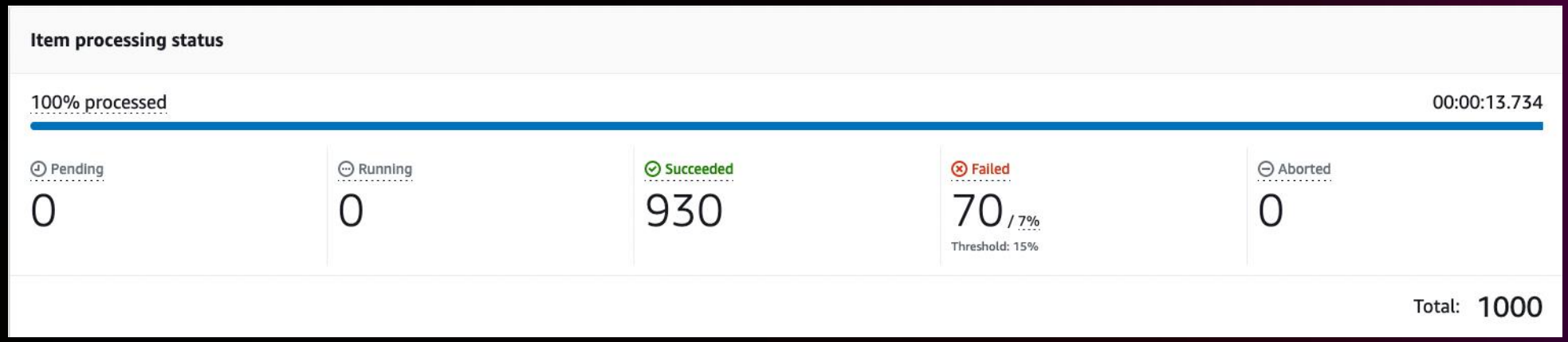


By default, a single child workflow execution failure will fail the whole Map Run, optional failure thresholds can be set

- # of items that failed to process in the child workflow executions
- % of items that failed to process in the child workflow executions

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Handling failures



Enable **Export Map state results to Amazon S3** to get lists delivered to S3 of all the child workflow executions that succeeded and all the ones that failed

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Distributed Map state output

- Specify a ResultsWriter to send results to S3
  - If this is specified, the output from the Map state includes the S3 key to the manifest file for the results
- If no ResultsWriter is specified, the output is an array containing the iteration results


```
{
  DestinationBucket: dist-map-noaa-analysis-s3resultsbucket-1ixrcjw2gg2c4,
  MapRunArn: arn:aws:states:ca-central-1:705158439502:mapRun:TemperatureStatsStateMachine-n6gDsbLNFXyx/DistributedMap:9acd77ce-686a-3d78-a566-ca54d14d7f15,
  ResultFiles: {
    FAILED: [
      {
        Key: results/9acd77ce-686a-3d78-a566-ca54d14d7f15/FAILED_0.json,
        Size: 297338
      }
    ],
    SUCCEEDED: [
      {
        Key: results/10acd77ce-686a-3d78-a566-ca54d14d7f15/SUCCEEDED_0.json,
        Size: 106165175
      }
    ]
  }
}
```


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

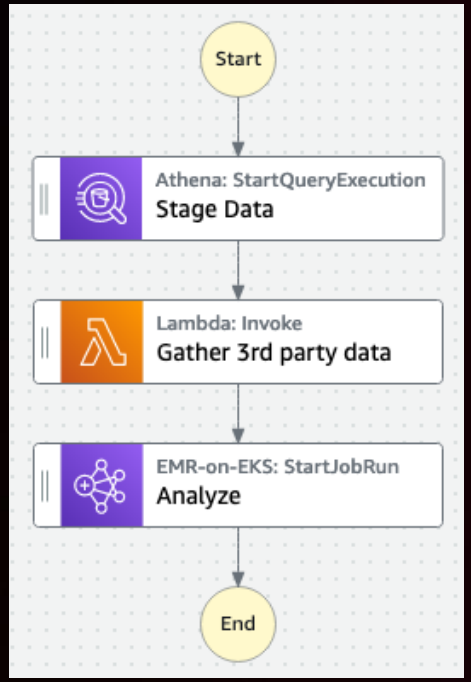
When to use Step Functions Distributed Map vs. AWS big data services

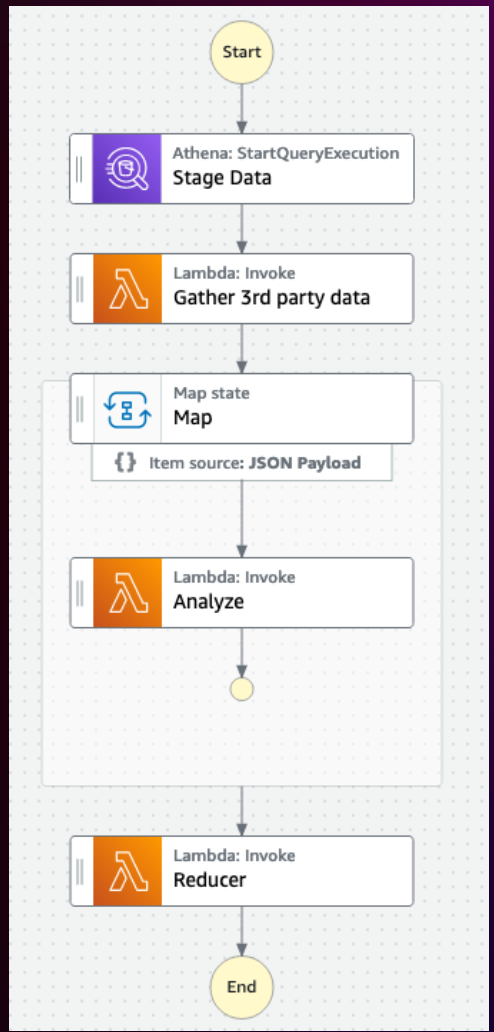
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Big data processing

Step Functions can coordinate big data processing across any AWS service

- Optimized integrations for AWS Glue and Amazon EMR

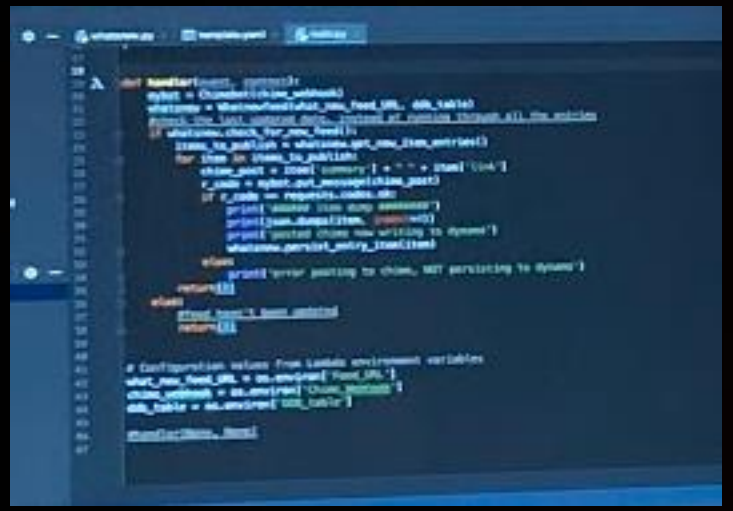




© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Use Distributed Map when you have

Code to process a single object



+

Many, many objects to process



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Getting started resources

Workflows section on Serverlessland.com



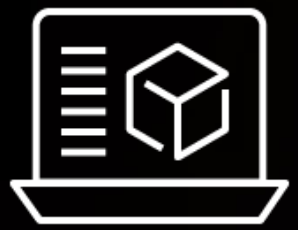
Step Functions workshop



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Continue your AWS serverless learning

Learn at your own pace



Expand your serverless skills with our learning plan on **AWS Skill Builder**

Increase your knowledge



Use our **Ramp-Up Guides** to build your serverless knowledge

Earn AWS serverless badge



Demonstrate your knowledge by achieving **digital badges**

https://s12d.com/serverless-learning

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Related sessions

API309 – Advanced serverless workflow patterns and best practices

API308 – Are you integrating or building distributed applications?

SVS312 – Building Serverlesspresso: Creating event-driven architectures

API307 – Designing event-driven integrations using Amazon EventBridge

SVS209 – Get started building your first serverless, event-driven application

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you!

Brian Zambrano
(he/him/they)
@brianzambrano

Justin Callison
(he/him/they)
@justincallison

Adam Wagner
(he/him/they)
@oathead
@oathead@hachyderm.io



Please complete the session survey in the **mobile app**

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.