Tutorial_2-AWS

January 11, 2023

1 StochSS-Compute on AWS

Follow this tutorial to learn how to launch an AWS EC2 instance and run a simulation on it.

[ ]:
```
import sys, os
sys.path.insert(1, os.path.abspath(os.path.join(os.getcwd(), '../')))

from stochss_compute.cloud import EC2Cluster
from stochss_compute import RemoteSimulation

import gillespy2
```

1.0.1 1. Configuration

1. First, create an AWS account here.
2. In order to make the AWS API calls to your account, you need an AWS access key and access key ID.
   From the IAM dashboard, click 'Manage access keys'.
   Then, under the Access keys tab, click 'Create New Access Key'.
   This file can only be downloaded once, but if something happens you can just make a new one.
   This file contains the Access Key ID and a Secret Access Key.
3. The simplest way to configure API calls is to download and install AWS Command Line Interface.
   Then, run `aws configure`.
   You will be asked for your AWS Access Key ID, your AWS Secret Access Key, and default region name (listed here), such as `us-east-2`.
   If you prefer not to install this, you can set the environment variables `AWS_ACCESS_KEY_ID`, `AWS_SECRET_ACCESS_KEY`, and `AWS_DEFAULT_REGION`.
   For a full list of environment variables you can set, see here.

[ ]:
```
# Uncomment the two lines below if using environment variables to configure AWS API calls.
# from dotenv import load_dotenv # To install: pip install python-dotenv
# load_dotenv() # Loads from a file named .env by default
```

1.0.2 2. Launch

- Instantiate a cluster object.
- `stochss_compute.cloud` will first attempt to re-load an already running cluster by resource name, so you can continue where you left off.

[ ]:
```
cluster = EC2Cluster()
```

- Launch a StochSS-Compute instance, providing an AWS instance type.
- For more information about instance types, see here.
- Make sure you are aware of AWS pricing policies before proceeding. See here and here for more information.

[ ]:
```
cluster.launch_single_node_instance('t2.micro')
```

```
2023-01-11 15:17:46,405 - EC2Cluster - INFO - Launching Network…
2023-01-11 15:17:51,780 - EC2Cluster - INFO - Launching StochSS-Compute server
instance. This might take a minute…
2023-01-11 15:18:23,542 - EC2Cluster - INFO - Instance "i-008f2f04c228c77bf" is
running.
2023-01-11 15:19:24,269 - EC2Cluster - INFO - Waiting on Docker daemon.
2023-01-11 15:20:24,580 - EC2Cluster - INFO - Waiting on Docker daemon.
2023-01-11 15:21:34,763 - EC2Cluster - INFO - Container "sssc" is running.
2023-01-11 15:21:34,764 - EC2Cluster - INFO - Restricting server access to only
your ip.

[SourceIpRequest] http://13.58.169.32:29681/api/v2/cloud/sourceip

2023-01-11 15:21:35,608 - EC2Cluster - INFO - StochSS-Compute ready to go!
```

1.0.3 3. Run

- Create your model

[ ]:
```
def create_michaelis_menten(parameter_values=None):
    # Intialize the Model with a name of your choosing.
    model = gillespy2.Model(name="Michaelis_Menten")

    """
    Variables (GillesPy2.Species) can be anything that participates in or is␣
 ↪produced by a reaction channel.

    - name: A user defined name for the species.
    - initial_value: A value/population count of species at start of simulation.
    """
    A = gillespy2.Species(name="A", initial_value=301)
    B = gillespy2.Species(name="B", initial_value=120)
    C = gillespy2.Species(name="C", initial_value=0)
    D = gillespy2.Species(name="D", initial_value=0)
```

```
    # Add the Variables to the Model.
    model.add_species([A, B, C, D])

    """
    Parameters are constant values relevant to the system, such as reaction␣
↪kinetic rates.

    - name: A user defined name for reference.
    - expression: Some constant value.
    """
    rate1 = gillespy2.Parameter(name="rate1", expression=0.0017)
    rate2 = gillespy2.Parameter(name="rate2", expression=0.5)
    rate3 = gillespy2.Parameter(name="rate3", expression=0.1)

    # Add the Parameters to the Model.
    model.add_parameter([rate1, rate2, rate3])

    """
    Reactions are the reaction channels which cause the system to change over␣
↪time.

    - name: A user defined name for the reaction.
    - reactants: A dictionary with participant reactants as keys, and consumed␣
↪per reaction as value.
    - products: A dictionary with reaction products as keys, and number formed␣
↪per reaction as value.
    - rate: A parameter rate constant to be applied to the propensity of this␣
↪reaction firing.
    - propensity_function: Can be used instead of rate in order to declare a␣
↪custom propensity function in string format.
    """
    r1 = gillespy2.Reaction(
            name="r1",
            reactants={'A': 1, 'B': 1},
            products={'C': 1},
            rate='rate1'
        )

    r2 = gillespy2.Reaction(
            name="r2",
            reactants={'C': 1},
            products={'A': 1, 'B': 1},
            rate='rate2'
        )

    r3 = gillespy2.Reaction(
```

```
            name="r3",
            reactants={'C': 1},
            products={'B': 1, 'D': 1},
            rate='rate3'
        )

    # Add the Reactions to the Model.
    model.add_reaction([r1, r2, r3])

    # Define the timespan of the model.
    tspan = gillespy2.TimeSpan.linspace(t=100, num_points=100)

    # Set the timespan of the Model.
    model.timespan(tspan)
    return model
```

- Run it

[ ]:
```
model = create_michaelis_menten()
```

[ ]:
```
simulation = RemoteSimulation(model, server=cluster, solver=gillespy2.
↪TauHybridSolver)
```

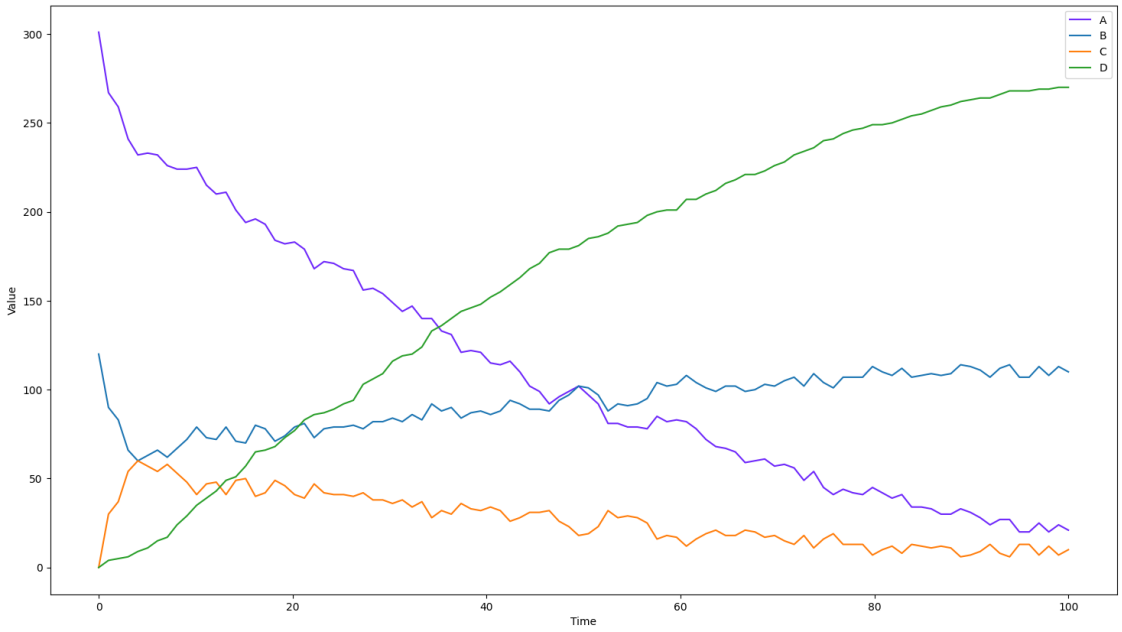
[ ]:
```
results = simulation.run()
```

```
[SimulationRunRequest] http://13.58.169.32:29681/api/v2/simulation/gillespy2/run
```

- Wait for/fetch results

[ ]:
```
results.plot()
```

```
[GET] http://13.58.169.32:29681/api/v2/simulation/gillespy2/38cdac61bb8dee92b4e1
35123743dfe0/1/38cdac61bb8dee92b4e135123743dfe0:1:1fa60a91fc8ca116/status
Simulation is running. Downloading results when complete…
[GET] http://13.58.169.32:29681/api/v2/simulation/gillespy2/38cdac61bb8dee92b4e1
35123743dfe0/1/38cdac61bb8dee92b4e135123743dfe0:1:1fa60a91fc8ca116/status
Results ready. Fetching…
[GET] http://13.58.169.32:29681/api/v2/simulation/gillespy2/38cdac61bb8dee92b4e1
35123743dfe0/1/results
```



1.0.4 4. Clean Up

- Deletes all cluster resources that were created by `launch_single_node_instance()`.

[ ]:
```
cluster.clean_up()
```

```
2023-01-11 15:21:42,513 - EC2Cluster - INFO - Terminating "i-008f2f04c228c77bf".
This might take a minute…
2023-01-11 15:22:28,115 - EC2Cluster - INFO - Instance i-008f2f04c228c77bf"
terminated.
2023-01-11 15:22:28,384 - EC2Cluster - INFO - Deleting
sg-067932d6f693b3b17…
2023-01-11 15:22:28,796 - EC2Cluster - INFO - Security group
sg-067932d6f693b3b17 deleted.
2023-01-11 15:22:29,000 - EC2Cluster - INFO - Deleting
subnet-0346917fa72f649ce…
2023-01-11 15:22:29,325 - EC2Cluster - INFO - Subnet subnet-0346917fa72f649ce
deleted.
2023-01-11 15:22:29,327 - EC2Cluster - INFO - Deleting
subnet-007942a158bbc86b4…
2023-01-11 15:22:29,625 - EC2Cluster - INFO - Subnet subnet-007942a158bbc86b4
deleted.
2023-01-11 15:22:29,710 - EC2Cluster - INFO - Detaching
igw-0a9b54864d2e5e9bc…
2023-01-11 15:22:29,919 - EC2Cluster - INFO - Gateway igw-0a9b54864d2e5e9bc
detached.
2023-01-11 15:22:29,920 - EC2Cluster - INFO - Deleting
```

```
igw-0a9b54864d2e5e9bc…
2023-01-11 15:22:30,124 - EC2Cluster - INFO - Gateway igw-0a9b54864d2e5e9bc
deleted.
2023-01-11 15:22:30,126 - EC2Cluster - INFO - Deleting
vpc-0fad697b6c3e906ad…
2023-01-11 15:22:30,431 - EC2Cluster - INFO - VPC vpc-0fad697b6c3e906ad deleted.
2023-01-11 15:22:30,506 - EC2Cluster - INFO - Deleting "sssc-server-ssh-key".
2023-01-11 15:22:30,507 - EC2Cluster - INFO - Key "sssc-server-ssh-key" deleted.
2023-01-11 15:22:30,642 - EC2Cluster - INFO - Deleting
"/home/mdip/projects/StochSS-Compute/examples/.sssc/sssc-server-ssh-key.pem".
2023-01-11 15:22:30,644 - EC2Cluster - INFO - "/home/mdip/projects/StochSS-
Compute/examples/.sssc/sssc-server-ssh-key.pem" deleted.
```