Analysis and Verification of Quantum Communication Protocols in UPPAAL

This folder contains artifacts for "Analysis and Verification of Quantum Communication Protocols in UPPAAL" paper for CAV'26.

Download the entire artifact: artifact.zip

Overview

The folder contains the following artifacts:

- bsdc-enumerated.xml - a self-contained BSDC model for 2 qubits with enumerated quantum states.
- bsdc-density-matrix.xml - a full BSDC model for 4 qubits with noise and distillation, shifts using density matrices and Armadillo library. The model is polished for readability.
- dmqs - C++ library linking with Armadillo, required by bsdc-density-matrix.xml and bsdc-experiments.xml.
- runner - Python library to manage large experiments in UPPAAL. Below are instructions on how to reproduce the figures from the paper, and a preview of the models.

Instructions

1. Download and install UPPAAL:

- Either version 5.0 or 5.1 from UPPAAL.org
- UPPAAL graphical interface requires Java Runtime Environment (JRE or JDK) installed; we recommend OpenJDK 25 from:
  - Linux distribution
  - Adoptium (may require administrative rights)
  - Microsoft (possible to install for local user environment)
- Obtain an academic license key:
  - Register at veriaal.dk
  - Or use our key: dd5bd740-8262-4fce-b6ee-5b2549c8a3c0

2. Reproduce snippets in Fig.3 by opening `bsdc-enumerated.xml` in UPPAAL:

- In Editor:
  - Select `Quantum2` and inspect the path shown in Quantum of Fig.3.
  - Select `SenderTimeslotted` and observe the beginning matching in Sender of Fig.3.
  - Select `ReceiverTimeslotted` and observe the beginning matching in Receiver of Fig.3.
  - Select `Monitor` and check that is equal to Monitor of Fig.3.
- In Verifier, check that the two properties are satisfied:

  - `A[] not deadlock`
  - `A[] not monitor.Error`
- Alternatively, check the properties on a command line:

```
verifyta bsdc-enumerated.xml
```

3. Reproduce snippets in Fig.4 by opening `bsdc-density-matrix.xml` in UPPAAL:

- In Editor:
- Select `Qstate4` and inspect the snippet show in Quantum of Fig.4.
- Select `Distiller` and inspect the snippet show in Distiller of Fig.4.
- Select `SenderShifting`and ovserve the middle matching in the snippet show in Sender of Fig.4

4. Reproduce Fig.5 and Fig.6

- Download and install Docker from docker.com
  - If you are using Docker on Ubuntu, you will have to install `docker-compose` (`sudo apt-get install docker-compose`)
- In Text editor:
  - If using own UPPAAL license key, change `UPPAAL_KEY` in `docker-compose-yml` on line 15
  - Depending on the number of cores in your system, consider changing the number of threads used to run the experiment in `config/distillation.py` and `config/replication.py` on line 4.
- In Terminal:
  - Run the following commands (Took 1 min 45 seconds).
    ```
    docker compose build
    docker compose up -d
    ```
  - Run the following command.
    ```
    docker exec -ti uppaal-bsdc-sim /bin/bash
    ```
  - Run the following command.
    ```
    source activate
    ```
  - Run the following command to reproduce Fig.5 (Took 17 min and 51 seconds on AMD Ryzen 7 9700X).
    ```
    uv run runner.py --config config/distillation.py --run --export
    ```
- In Artifact folder.
  - Open with a browser or any installed svg viewer the Fig.5 result `errors_0.2.svg` found in the `data/distillation` folder.
- In Terminal:
  - Run the following command to reproduce Fig.6 (Took 18 seconds on AMD Ryzen 7 9700X).
    ```
    uv run runner.py --config config/replication.py --run --export
    ```
- In Artifact folder on the host system.
  - Open with a browser or any installed SVG viewer Fig.6 result `Bits_per_Timeslot_Sent.svg` and `Errors_per_Qubit_Sent.svg` in the `data/replication` folder on the host system.