Reproduction Package for Ready to Leap (by Co-Design)? Join Order Optimisation on Quantum Hardware

MANUEL SCHÖNBERGER, Technical University of Applied Sciences Regensburg, Germany
WOLFGANG MAUERER, Technical University of Applied Sciences Regensburg and Siemens AG, Corporate Research, Germany

This document provides detailed instructions for reproducing experiments for *Ready to Leap (by Co-Design)? Join Order Optimisation on Quantum Hardware*, based on our reproduction package.

1 OVERVIEW

Our reproduction package includes

(1) a Docker production script (*Dockerfile*)
(2) source code for all experiments (in the *base* and *base/Scripts* directories)
(3) all data artifacts as collected by our experiments, including solutions obtained on remote quantum devices (*base/ExperimentalAnalysis* and *base/TranspilationExperiment* directories)

We discuss hardware and estimated reproduction time in Sec. 2. We specify reproduction parameters in Sec. 3 and our recommended settings in Sec. 4. Finally, we explain usage instructions in Sec. 5.

2 HARDWARE AND ESTIMATED REPRODUCTION TIME

As we provide a fully automated reproduction script, manual reproduction effort is very limited. The limited instruction steps are outlined in Sec. 5.

Reproduction time depends on the applied reproduction settings, which are outlined in Sec. 3. Using our recommended settings, the entire process may take several hours, depending on the used hardware, where no strict requirements have to be met. To provide an estimation for a concrete setup, we have run the entire reproduction process, as detailed in Sec. 5, on an octa-core AMD Ryzen 7 PRO 5850U CPU with 32 GB of DDR4 RAM, where a full reproduction based on our recommended settings finished after roughly 15 hours.

3 REPRODUCTION SETTINGS

Reproduction settings can be specified in the configuration file (base/config.py), which includes the following parameters:

- **dwave-processing-queries**(*str*): Queries to be used in D-Wave experiments. Choose 'collected' for queries already included in the reproduction package (and used for the experiments), or 'generate' for generating a new batch of queries. Default: 'collected'.
  Note: Generating new queries (option 'generate') requires specifying Gurobi license credentials (gurobi-wlsaccessid, gurobi-wlssecret and gurobi-licenseid, see settings below), for constructing corresponding QUBO encodings based on the Gurobi MILP optimisation library.
- **dwave-processing** (*str*): Mode for running D-Wave experiments. Choose 'collected' for processing data already collected on a DWave QPU, 'qpu' for running a new batch of experiments on a remote, actual D-Wave QPU, or 'cpu' for running experiments on a local simulator. Default: 'collected'.
  Note: Accessing a real QPU (option 'qpu') requires including a license file (dwave.conf) with valid credentials in the licenses directory. Experimental data generated by a local simulator may substantially differ in quality from data obtained by a real remote QPU, which was used for the paper.

- **ibmq-processing** (*str*): Mode for running IBM Q experiments. Choose 'collected' for processing data already collected on an IBM Q QPU (and included in the reproduction package), 'qpu' for running a new batch of experiments on a remote, actual IBM Q QPU, or 'cpu' for running experiments on a local simulator. Default: 'collected'.
  Note: Accessing a real QPU (option 'qpu') requires specifying a valid access token, IBM Q hub, IBM Q group, IBM Q project and IBM Q backend (see settings ibmq-token, ibmq-hub, ibmq-group, ibmq-project, ibmq-backend, explained below). Experimental data generated by a local simulator may substantially differ in quality from data obtained by a real remote QPU, which was used for the paper.
- **ibmq-token** (*str*): IBM Q token, required for accessing IBM Q devices. Default: "".
- **ibmq-hub** (*str*): IBM Q hub, specifying the hub for the IBM Q device. Default: "open".
- **ibmq-group** (*str*): IBM Q group, specifying the group for the IBM Q device. Default: "open".
- **ibmq-project** (*str*): IBM Q project, specifying the project for the IBM Q device. Default: "main".
- **ibmq-backend** (*str*): Backend identifier for the IBM Q device. Default: "".
- **codesign-queries** (*str*): Queries to be used in DB/QPU co-design experiments. Choose 'collected' for queries already included in the reproduction package (and used for the experiments), or 'generate' for generating a new batch of queries. Default: 'collected'.
  Note: Generating new queries (option 'generate') requires specifying Gurobi license credentials (gurobi-wlsaccessid, gurobi-wlssecret and gurobi-licenseid, see settings above), for constructing corresponding QUBO encodings based on the Gurobi MILP optimisation library.
- **qiskit-circuit-transpilation**: Mode for experiments involving quantum circuit transpilation, using the qiskit transpilation library. Choose 'collected' for processing data already collected (and included in the reproduction package), or 'retranspile' for running a new batch of circuit transpilations. Default: 'collected'.
- **tket-circuit-transpilation** (*str*): Mode for experiments involving quantum circuit transpilation, using the tket transpilation library. Choose 'collected' for processing data already collected (and included in the reproduction package), or 'retranspile' for running a new batch of circuit transpilations. Default: 'collected'.
  Note: Retranspiling circuits with the tket library (option 'retranspile') requires specifying credentials for IBM Q and IonQ access (see settings ibmq-token and ionq-token), for fetching the required passes.
- **ionq-token** (*str*): IonQ token, required for accessing IonQ devices. Default: "".
- **gurobi-wlsaccessid** (*str*): ID for the Gurobi Web License Service. Default: "".
- **gurobi-wlssecret** (*str*): Secret for the Gurobi Web License Service. Default: "".
- **gurobi-licenseid** (*int*): License ID for the Gurobi Web License Service. Default: 0.

4 RECOMMENDED SETTINGS

While our reproduction packages allows parameter specifications to reproduce all steps of our experiments, most of these steps require varying kinds of third party licenses (*e.g.*, Gurobi, IBM-Q, D-Wave). For our recommended settings, we hence suggest to

(1) consider queries as already provided in the reproduction package itself (the queries are identical to the ones used to obtain our experimental results)
(2) consider data collected on the remote quantum systems (IBM-Q and D-Wave), as included in the reproduction package
(3) consider data collected by the tket transpiler, as included in the reproduction package

(4) retranspile the quantum circuits for solving the included queries with the qiskit transpilation library.

This setup avoids providing any licenses, and allows to reproduce experiments for transpiling IBM Q quantum circuits and DB-QPU co-design (Figure3 and Figure 5 in our paper respectively). The default values in the configuration file (base/config.py) correspond to our recommended setup. The remaining figures are produced based on data previously collected using the required third party licenses.

More comprehensive settings require to obtain licenses, where Sec. 3 outlined the respective licensing parameters and third party library dependencies for varying settings. In case queries are to be regenerated (value "generate" for the *dwave-processing-queries* or *codesign-queries*), Gurobi web license information [3] has to be specified (a free academic web license is sufficient), to create the required quantum encodings. For accessing D-Wave [1] or IBM Q [4] remote quantum devices, paid licenses have to be provided: For IBM Q access, the credentials have to be set as values for the respective configuration parameters, whereas for acessing D-Wave devices, a *dwave.conf* file has to be placed in the *licenses* directory. To transpile quantum circuits with the tket optimiser, IBM Q [4] and IonQ [5] credentials have to be specified (free licenses are sufficient), to fetch the respective transpilation passes.

5 USAGE INSTRUCTIONS

In this section, we outline the specific steps to work with our reproduction package. For running experiments based on our recommended settings, no adjustments to the contents of the reproduction package are required. If any further experiments that require third party access are desired, the required parameters (*base/config.py*) have to be set accordingly, alongside the required access credentials (for Gurobi, IBM Q, IonQ access), and license files (for D-Wave access) must be placed in the *licenses* directory, before proceeding with the following steps.

To use the docker container, a docker installation is required [2]. The docker image may be built using our provided production script, using the following command:

```
docker build -t sigmod-repro .
```

To create a docker container for the docker image, the following command may be used:

```
docker run –name sigmod-repro -it sigmod-repro [<option>]
```

Available options are

(1) *ibmq_only*: Runs only IBM Q experiments
(2) *dwave_only*: Runs only D-Wave experiments
(3) *codesign_only*: Runs only DB-QPU co-design experiments
(4) *all*: Runs all of the above
(5) *bash*: Launches an interactive shell

For a full reproduction, we recommend running

```
docker run –name sigmod-repro -it sigmod-repro all
```

After the experiments are finished, plots for all figures depicting experimental data (Figure 2, Figure 3 and Figure 5 in our paper) can be found as PDF files inside the *plots* directory. Note that Figure 4 of our paper does not show any experimental results, but rather depicts resource requirements that are clear a priori. Hence, our reproduction package does not produce a PDF file for Figure 4.

REFERENCES

[1] D-Wave. 2023. D-Wave Leap. https://cloud.dwavesys.com/leap
[2] Docker Inc. 2023. Docker. https://www.docker.com/
[3] Gurobi. 2023. Gurobi Web License Service. https://www.gurobi.com/features/web-license-service/
[4] IBM Quantum. 2022. Cloud access to quantum computers provided by IBM. https://quantum-computing.ibm.com
[5] IonQ Quantum. 2022. IonQ technology. https://ionq.com/technology