High Qubit simulation with PyQrack OpenCL Hybrid POCL and ZSWAP

Full Connectivity MARP analysis with ThereminQ-tensors

A Gateway to more flexible and High-Fidelity OpenCL QC Simulators

Introduction

Quantum computing (QC) circuit simulations on GPUs often demand significant memory.

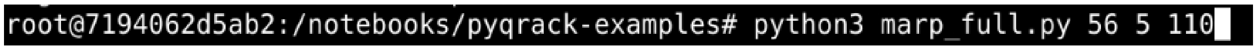
Execution time for these simulations is typically less critical compared to memory requirements, especially at high qubit counts and full connectivity. The primary concern is whether the available memory can accommodate the workload.

Recent advancements in multicore and high-bandwidth CPU systems suggest that relying solely on vendor-specific GPU solutions may not fully utilize available performance. To address this, we aim to expand the memory available for running MARP circuit experiments.

Setup

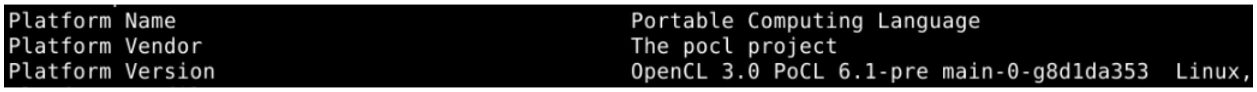
Common frameworks for QC simulation include CUDA and OpenCL. In this context, we use OpenCL.

- **PyQrack Examples:** Available at GitHub, providing implementations for both 2D and full MARP connectivity circuits.



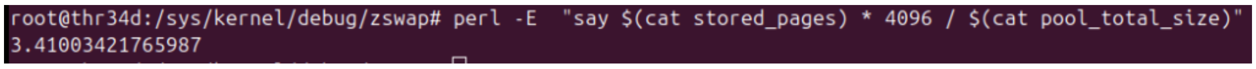
```
root@7194062d5ab2:/notebooks/pyqrack-examples# python3 marp_full.py 56 5 110
```

- **POCL:** Enables adding a CPU device to the OpenCL stack. The latest version (v6.1 pre) is on GitHub, while Linux distributions often carry v5.x or older.



```
Platform Name                                   Portable Computing Language
Platform Vendor                                 The pocl project
Platform Version                                OpenCL 3.0 PoCL 6.1-pre main-0-g8d1da353  Linux,
```

- **ZSWAP:** A kernel extension for memory compression before swapping, achieving compression ratios up to 20x. Typical PyQrack workloads see about 3x compression. Documentation is available here.



```
root@thr34d:/sys/kernel/debug/zswap# perl -E  "say $(cat stored_pages) * 4096 / $(cat pool_total_size)"
3.41003421765987
```

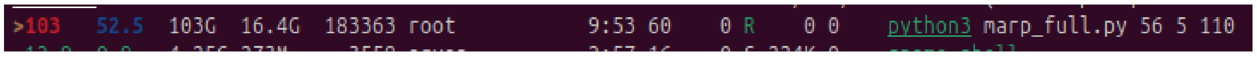
Note: to start containerized VNC WebUI environments, use the following command:

- *docker run --gpus all --ipc=host --device=/dev/dri:/dev/dri -d twobombs/thereminq-tensors:jupyter-pocl*
- *Access the environment via a browser at http://localhost:6080*

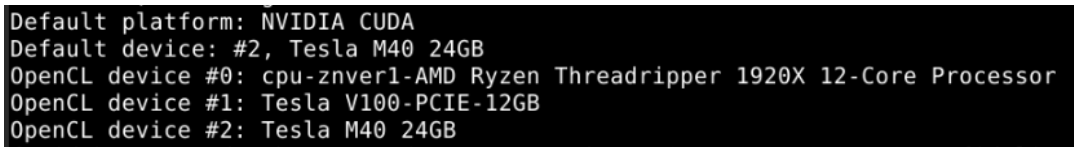
Runtime

Running marp_full.py on a system with OpenCL POCL and ZSWAP compressed memory results in faster execution times. This configuration avoids intensive swap utilization seen in CPU-only setups without ZSWAP and prevents VRAM OOM exceptions common in GPU-only setups.

Adding a POCL ZSWAP-enabled device to the OpenCL stack effectively increases the available system RAM for the process by 3x creating a software-based GPU with compressed vRAM.



```
>103   52.5   103G  16.4G  183363 root            9:53 60     0 R     0 0    python3 marp_full.py 56 5 110
```

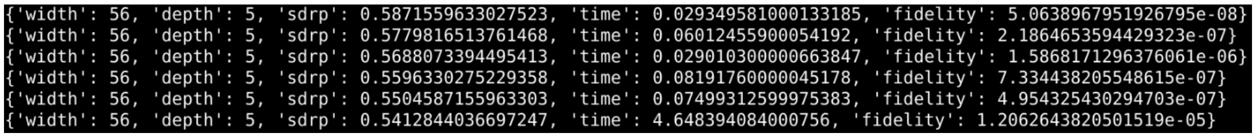


```
Default platform: NVIDIA CUDA
Default device: #2, Tesla M40 24GB
OpenCL device #0: cpu-znver1-AMD Ryzen Threadripper 1920X 12-Core Processor
OpenCL device #1: Tesla V100-PCIE-12GB
OpenCL device #2: Tesla M40 24GB
```

Benchmarks:

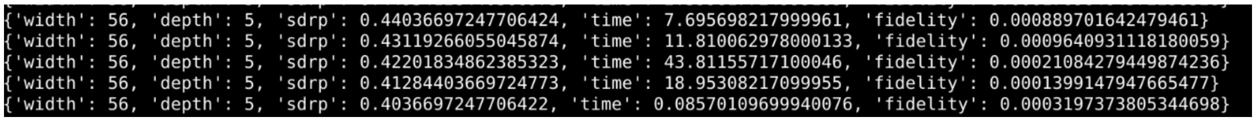
CPU only results

- 24 thread CPU with 32 GB RAM ZSWAP: high SDRP, low fidelity



```
{'width': 56, 'depth': 5, 'sdrp': 0.5871559633027523, 'time': 0.029349581000133185, 'fidelity': 5.0638967951926795e-08}
{'width': 56, 'depth': 5, 'sdrp': 0.5779816513761468, 'time': 0.06012455900054192, 'fidelity': 2.1864653594429323e-07}
{'width': 56, 'depth': 5, 'sdrp': 0.5688073394495413, 'time': 0.029010300000663847, 'fidelity': 1.5868171296376061e-06}
{'width': 56, 'depth': 5, 'sdrp': 0.5596330275229358, 'time': 0.08191760000045178, 'fidelity': 7.334438205548615e-07}
{'width': 56, 'depth': 5, 'sdrp': 0.5504587155963303, 'time': 0.07499312599975383, 'fidelity': 4.954325430294703e-07}
{'width': 56, 'depth': 5, 'sdrp': 0.5412844036697247, 'time': 4.648394084000756, 'fidelity': 1.2062643820501519e-05}
```

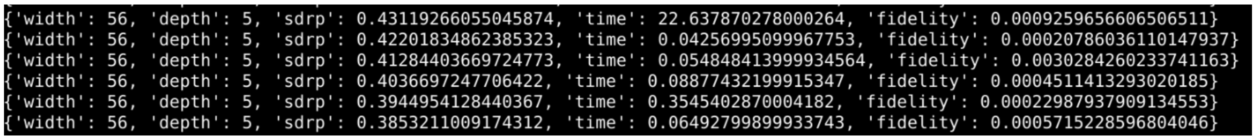
- 96 thread CPU with 64 GB RAM ZSWAP: mid SDRP, mid fidelity



```
{'width': 56, 'depth': 5, 'sdrp': 0.44036697247706424, 'time': 7.695698217999961, 'fidelity': 0.000889701642479461}
{'width': 56, 'depth': 5, 'sdrp': 0.43119266055045874, 'time': 11.810062978000133, 'fidelity': 0.0009640931118180059}
{'width': 56, 'depth': 5, 'sdrp': 0.42201834862385323, 'time': 43.81155717100046, 'fidelity': 0.00021084279449874236}
{'width': 56, 'depth': 5, 'sdrp': 0.41284403669724773, 'time': 18.95308217099955, 'fidelity': 0.0001399147947665477}
{'width': 56, 'depth': 5, 'sdrp': 0.4036697247706422, 'time': 0.08570109699940076, 'fidelity': 0.0003197373805344698}
```

GPU only results

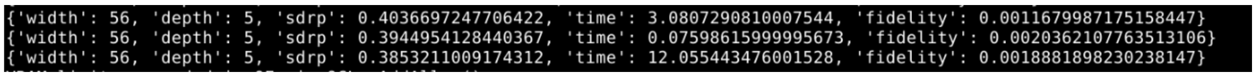
- 24GB + 16GB vRAM OCL: low SDRP, low fidelity



```
{'width': 56, 'depth': 5, 'sdrp': 0.43119266055045874, 'time': 22.637870278000264, 'fidelity': 0.0009259656606506511}
{'width': 56, 'depth': 5, 'sdrp': 0.42201834862385323, 'time': 0.04256995099967753, 'fidelity': 0.00020786036110147937}
{'width': 56, 'depth': 5, 'sdrp': 0.41284403669724773, 'time': 0.054848413999934564, 'fidelity': 0.0030284260233741163}
{'width': 56, 'depth': 5, 'sdrp': 0.4036697247706422, 'time': 0.08877432199915347, 'fidelity': 0.0004511413293020185}
{'width': 56, 'depth': 5, 'sdrp': 0.3944954128440367, 'time': 0.3545402870004182, 'fidelity': 0.00022987937909134553}
{'width': 56, 'depth': 5, 'sdrp': 0.3853211009174312, 'time': 0.06492799899933743, 'fidelity': 0.0005715228596804046}
```

Hybrid

- CPU 32 GB RAM ZSWAP POCL, 24 GB + 16 GB vRAM: low SDRP, higher fidelity



```
{'width': 56, 'depth': 5, 'sdrp': 0.4036697247706422, 'time': 3.0807290810007544, 'fidelity': 0.0011679987175158447}
{'width': 56, 'depth': 5, 'sdrp': 0.3944954128440367, 'time': 0.07598615999995673, 'fidelity': 0.0020362107763513106}
{'width': 56, 'depth': 5, 'sdrp': 0.3853211009174312, 'time': 12.055443476001528, 'fidelity': 0.0018881898230238147}
```

Notably, the fidelity difference between CPU/GPU-only setups and the hybrid stack is substantial.

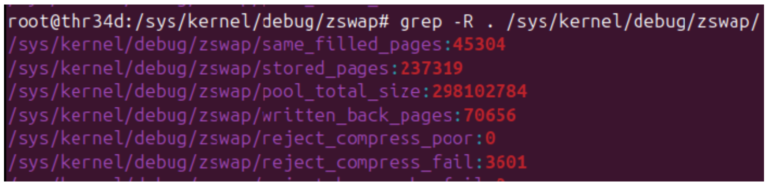
Conclusion

The elements of this stack have been under development for a while. Despite some initial errors, the stack runs successfully on an experimental basis. Future analysis will benchmark the stack's usefulness for combined QC/AI workloads.

disclaimer : these results are preliminary and could be subject to changes by review

Appendix I : runtime notes

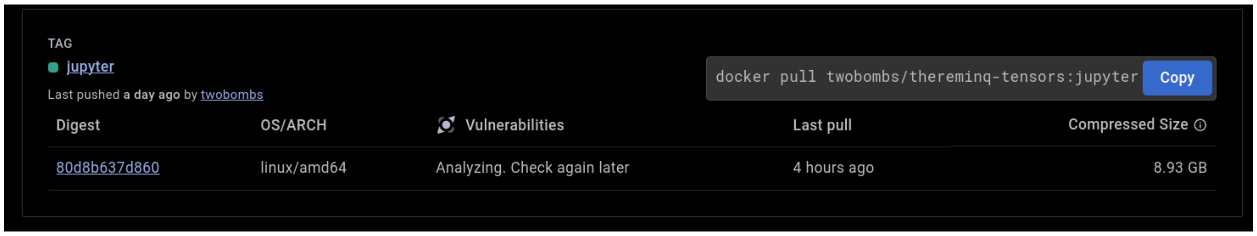
- Typical ZSWAP utilization demonstrates a relatively low amount of compression fail pages compared to the total stored pages.



```
root@thr34d:/sys/kernel/debug/zswap# grep -R . /sys/kernel/debug/zswap/
/sys/kernel/debug/zswap/same_filled_pages:45304
/sys/kernel/debug/zswap/stored_pages:237319
/sys/kernel/debug/zswap/pool_total_size:298102784
/sys/kernel/debug/zswap/written_back_pages:70656
/sys/kernel/debug/zswap/reject_compress_poor:0
/sys/kernel/debug/zswap/reject_compress_fail:3601
```

Appendix II

- The ThereminQ-tensors docker image used is :jupyter modified to run POCL



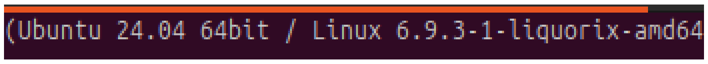
TAG
jupyter
Last pushed **a day ago** by twobombs

```
docker pull twobombs/thereminq-tensors:jupyter
```

Digest	OS/ARCH	Vulnerabilities	Last pull	Compressed Size
80d8b637d860	linux/amd64	Analyzing. Check again later	4 hours ago	8.93 GB

Appendix III

- The used kernel is the optimized but unmodified Liquorix kernel on Ubuntu 24.04



```
(Ubuntu 24.04 64bit / Linux 6.9.3-1-liquorix-amd64
```

For more information and resources, visit:

- ThereminQ-tensors GitHub
- Docker Hub
- Liquorix Kernel