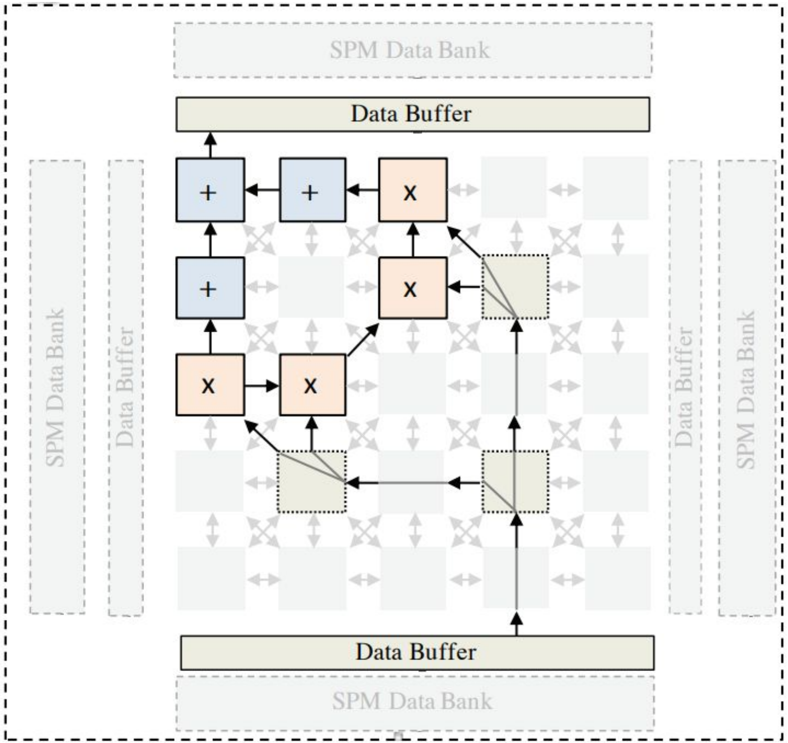
Designing CGRAs with Deep Reinforcement Learning

Jackson Woodruff, Chris Cummins
University of Edinburgh, Meta AI

CGRAs

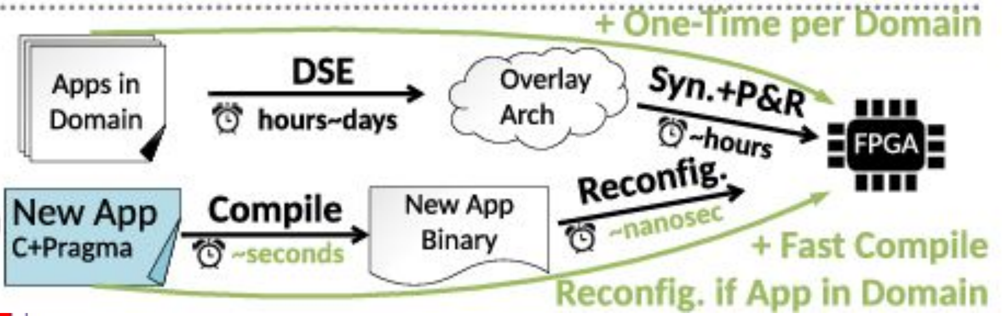


(Tan 2021)

CGRA Use Cases

(Relatively) Easy Design Process

Library-Based Acceleration (e.g. NXP PowerQuad)



(Liu, Micro 2022)

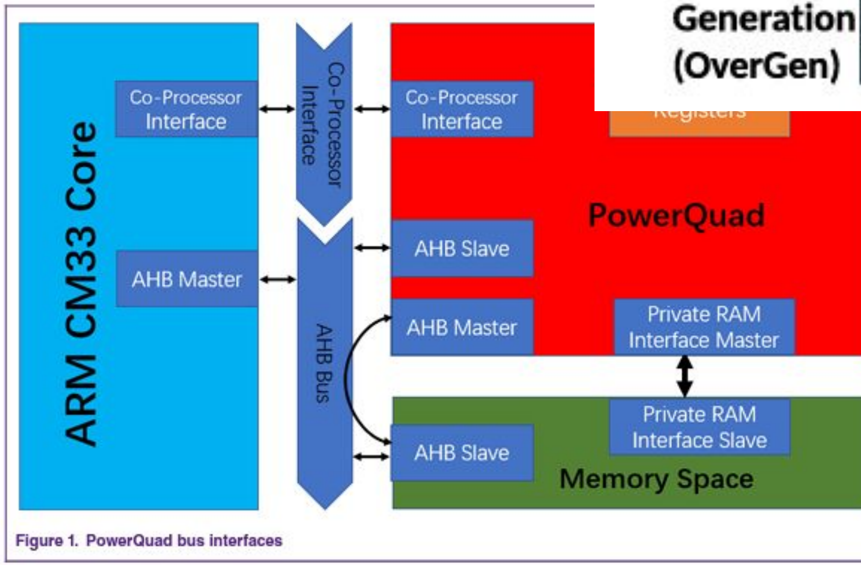


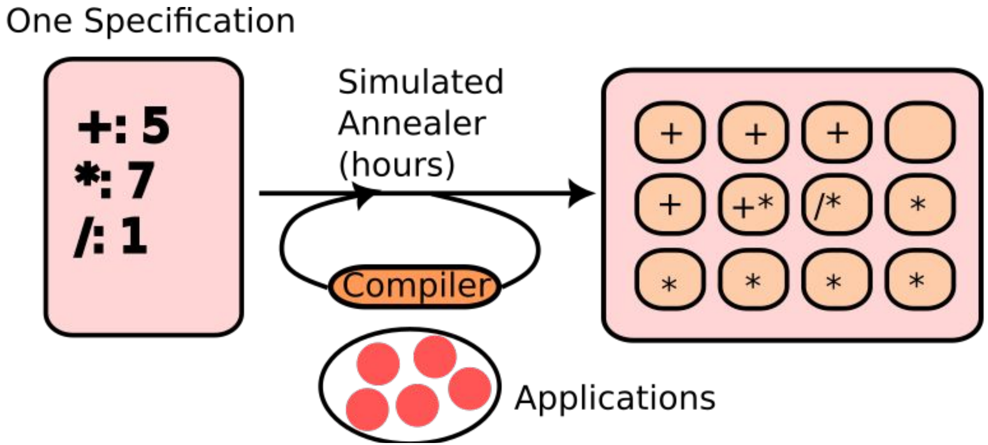
Figure 1. PowerQuad bus interfaces

(NXP PowerQuad Documentation)

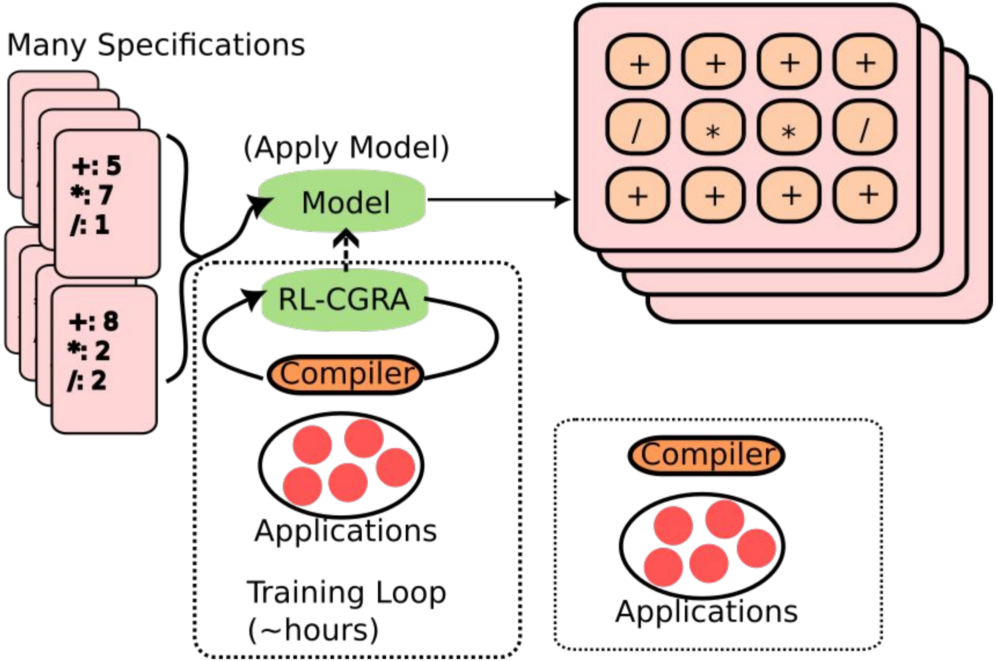
Why Design CGRAs?

- Low power
- Flexibility
- Easy Programmability
- Mature Toolchains

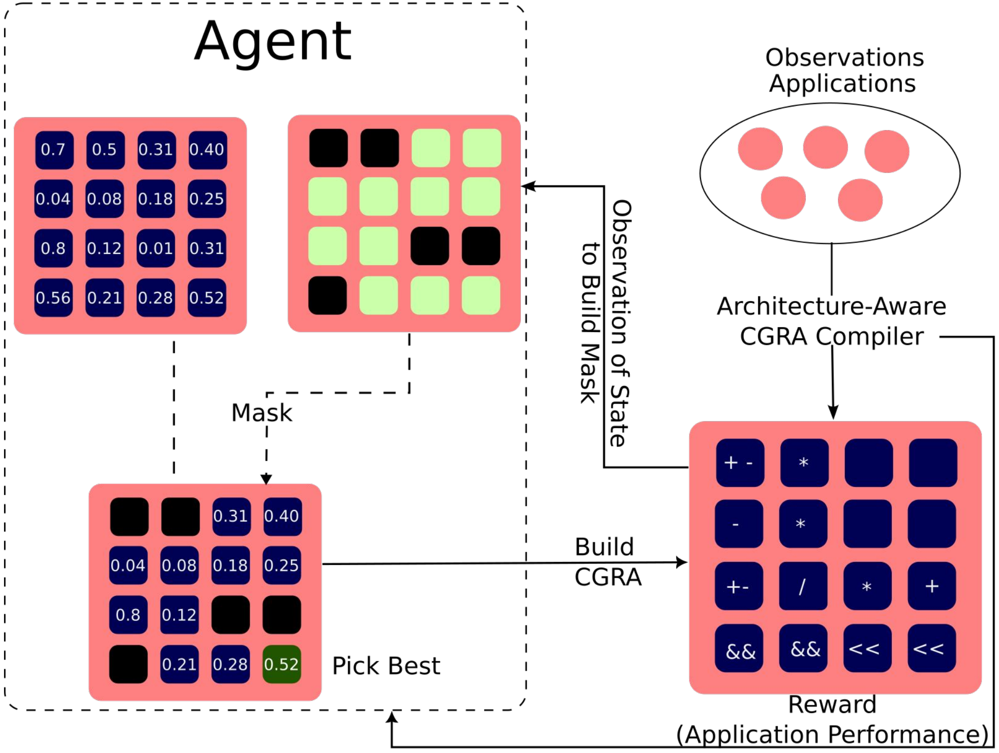
Simulated Annealing-Based Design



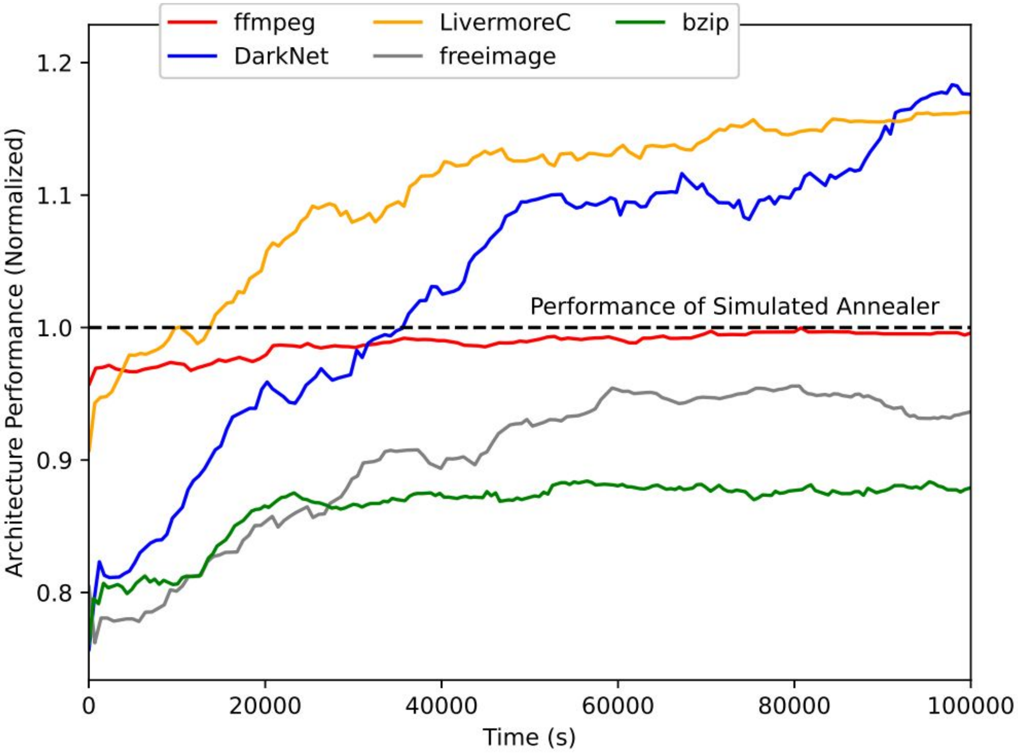
RL-CGRA


Many Specifications
+: 5
*: 7
/: 1
+: 8
*: 2
/: 2
(Apply Model)
Model
RL-CGRA
Compiler
Applications
Training Loop
(~hours)
+ + + +
/ * * /
+ + + +
Compiler
Applications

RL-CGRA Agent Design



RL-CGRA Results



RL-CGRA Open Questions

- What about applications is causing performance differences?
- Is it possible to do compiler-directed learning with code features in reasonable times?
    - What features?
- Can these techniques apply to finer-grained architectures?
- What is best format of compiler feedback?

Conclusions

- Explores RL agents for CGRA design
- Integrate compiler-feedback directly into hardware design toolchains
- Learn from previous experience doing so

References

C. Tan, C. Xie, A. Li, K. J. Barker and A. Tumeo, "OpenCGRA: An Open-Source Unified Framework for Modeling, Testing, and Evaluating CGRAs," *2020 IEEE 38th International Conference on Computer Design (ICCD)*, Hartford, CT, USA, 2020, pp. 381-388, doi: 10.1109/ICCD50377.2020.00070

S. Liu *et al*., "OverGen: Improving FPGA Usability through Domain-specific Overlay Generation," *2022 55th IEEE/ACM International Symposium on Microarchitecture (MICRO)*, Chicago, IL, USA, 2022, pp. 35-56, doi: 10.1109/MICRO56248.2022.00018.