Unity discussion points

- Unity has a fixed search process such as TASO [1] or TENSAT [2], to what extent could methods from X-RLFlow for an adaptive search approach?
- How can learned methods influence the cost model of the given system?

- Is the PCG a useful abstraction?
- How can methods like the PCG be incorporated into data-flow programming approaches such as Jax [3]?
- Is there a need for a "-ffast-math DNN compiler"; how can relaxing the strength constraints on transforms open new optimizations?

Bibliography

[1] Z. Jia, O. Padon, J. Thomas, T. Warszawski, M. Zaharia, and A. Aiken, "TASO: optimizing deep learning computation with automatic generation of graph substitutions," in *Proceedings of the 27th ACM Symposium on Operating Systems Principles*, in SOSP '19. Huntsville, Ontario, Canada: Association for Computing Machinery, 2019, pp. 47–62. doi: 10.1145/3341301.3359630.

[2] Y.-C. Yang, P. M. Phothilimthana, Y. R. Wang, M. Willsey, S. Roy, and J. A. Pienaar, "Equality Saturation for Tensor Graph Superoptimization," *ArXiv*, vol. abs/2101.01332, 2021, [Online]. Available: https://api.semanticscholar.org/CorpusID:230523881

[3] R. Frostig, M. Johnson, and C. Leary, "Compiling machine learning programs via high-level tracing," 2018. [Online]. Available: https://mlsys.org/Conferences/doc/2018/146.pdf