Cost Modelling for Register Allocation and Beyond

UCDAVIS
COLLEGE OF ENGINEERING

College of Engineering
University of California, Davis

UCDAVIS

COST MODELLING FOR REGISTER ALLOCATION AND BEYOND

Presented by Aiden Grossman

May 7, 2023

Working under Mircea Trofin and Ondrej Sykora.

WHY FAST, ACCURATE, AND STATIC COST MODELS?

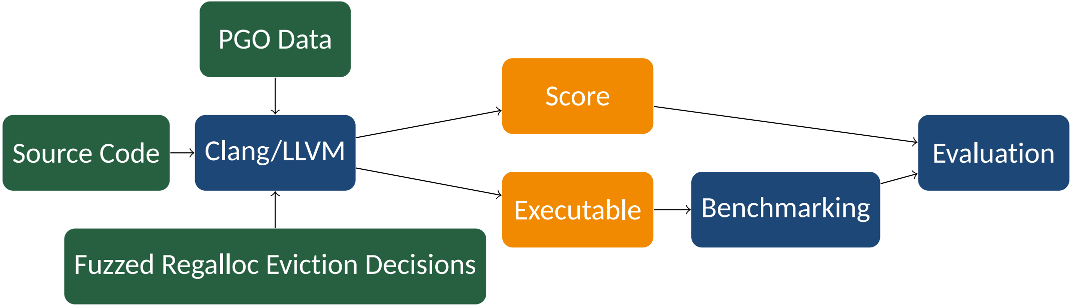
- Benchmarking is expensive and noisy.
- Cost models might be called millions of times during training.
- Having deterministic results can make ML training easier.
- The better the accuracy, the more "deterministic" model training can be.
- Higher accuracy leads to better deployed models.

CURRENT LINEAR MODEL

Currently, we're using a linear model, implemented in
`llvm/lib/CodeGen/RegAllocScore.cpp`

- Counts the number of a couple memory specific instructions, weights by latencies and MBB frequency.
- Does produce some signal.
- Leaves a lot to be desired in terms of "determinism" in training.
- Still can produce performant heuristic-replacing models. One is currently deployed in Google Search.

EVALUATING COST MODELS - PROCESS



EVALUATING COST MODELS - METRICS

- Polarity correct - polarity prediction around an arbitrary pivot point.
- Mean difference
- Ordering (tau coefficient)

LINEAR MODEL PERFORMANCE

UCDAVIS

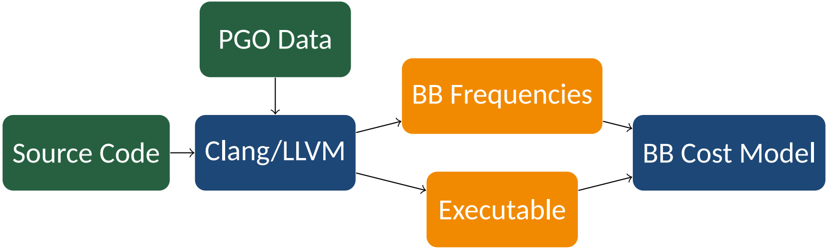
- Does not perform particularly well.
- Polarity correct metric hovers around the 50-60% mark.
- Average difference is a little under 5%.
- Tau coefficient for standard benchmarks hovers around 0.
- Fitting new weights greatly improves performance but offers no generalization.

USING SOA BB COST MODELS

- New BB cost models are quite accurate[1] and reasonably fast.
- Models many more properties than the simple linear model (like instruction ordering).
- Learned models are also highly performant[2].

[1] Abel and Reineke, “uiCA”.
[2] Sykora et al., *GRANITE*.

BB COST MODELS - PROCESS



PERFORMANCE OF SOA BB COST MODELS

- Significantly better than the linear model on all metrics.
- Percent error drops by up to 50%.
- Polarity accuracy increases even in hard to model cases.
- Actual accuracy of ordering is about the same.

LIMITATIONS OF THIS WORK

- The current evaluation framework only works on small benchmarks.
- Only a small variety of benchmarks have been tested.
- Achieving ideal execution conditions while running non-trivial benchmarks is difficult.
- Only fuzzed part of the register allocator.

WHERE/WHY DO THESE MODELS FAIL

- These models all assume ideal execution environments.
- Ideal conditions are rare and non-ideal conditions can change results by multiple orders of magnitude.
- Presence of L1 cache misses significantly impacts the performance of the linear model.
- Anything beyond the stream of instructions in a BB is not modelled (i.e., branching, function call overhead).

FUTURE DIRECTIONS - BETTER LEARNED BB MODELS

UCDAVIS

- Learned cost models are more adaptable to new (micro)architectures.
- Ground truth data has a lot of collection nuances.[3][4]
- Should be landing changes soon in `llvm-exegesis` to alleviate this problem.
- Assembly fuzzing might alleviate models learning false patterns[5].

---

[3] Abel and Reineke, “uiCA”.
[4] Chen et al., “BHive”.
[5] Ritter and Hack, “AnICA”.

MORE PROFILE INFORMATION

- Modelling non-ideal execution completely statically is essentially impossible.
- Collecting profile information and tagging specific instructions should massively increase accuracy.
- Building data collection pipelines and integrating this data into LCMs is an open scientific/engineering problem.

ARTIFACTS AND Q&A

- Artifacts available at
  `https://github.com/boomanaiden154/regalloc-cost-model-evaluation`
- Questions?