Query-Directed Adaptive Heap Cloning for Optimising Compilers

Compiler Optimisations

Tool Framework

Heap Cloning

```c
int main()
    { int* buffer1 = getMem();
        return malloc;
    }

    int* getMem(){
        return malloc;
    }

top-down
    Heap-Aware Pointer Analysis
    WOPT
    Compile
    Executable Program
```

Motivation

Very costly to reason about full heap cloning on large size program with high-density call graph

Compiler optimisations of diverse programs benefit in different precision levels of heap cloning

Approach

QUDA: Query-Directed Adaptive Heap Cloning

QUDA performs heap cloning on parts of the program according to clients’ need in a demand-driven fashion

An Example

QUDA heap cloning (1st iteration)

QUDA heap cloning (2nd iteration)

QUDA heap cloning (3rd iteration)

Results

Analysis overhead compared to Fulcra (state-of-the-art)

Heap objects reduced by QUDA over Fulcra

QUDA analysis time per iteration over the total

QUDA alias queries to be answered at each iteration

QUDA Framework in the Open64 Compiler

QUDA is implemented in industry-strength compiler Open64 scales up to 200K lines of code. It has the same precision as the state-of-the-art, but is significantly more scalable. For 10 SPEC2000 C benchmarks and 5 C applications (totalling 840 KLOC) evaluated, QUDA takes only 4+ minutes but exhaustive heap cloning takes 42+ minutes to complete.