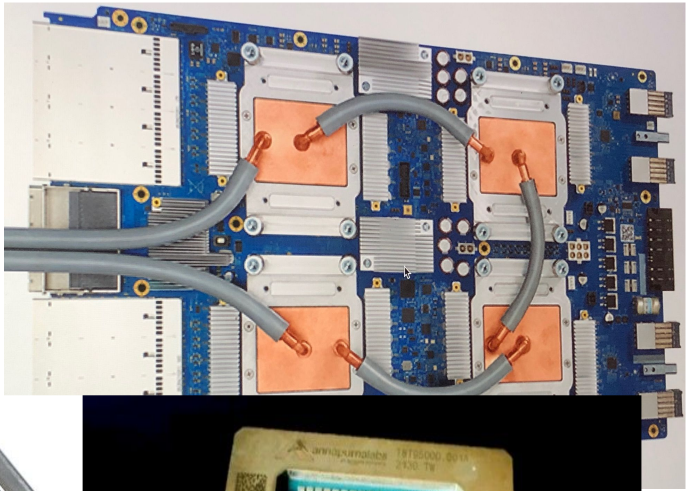
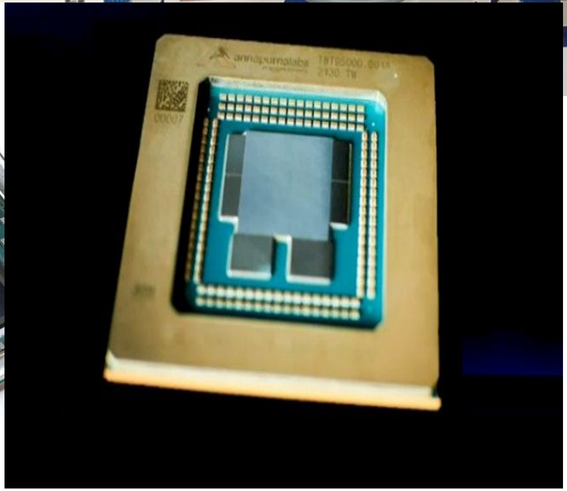
ELE 548: Project Proposal

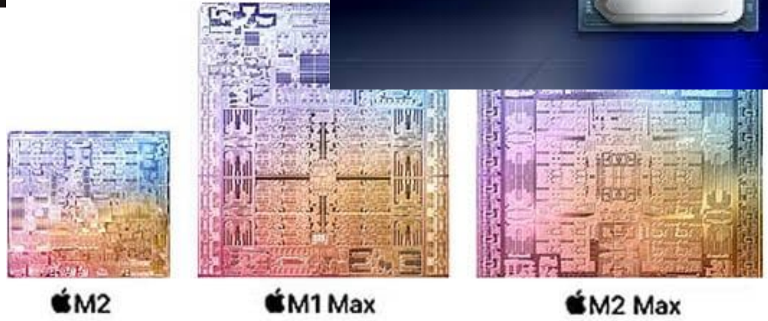
- Calvin Higgins

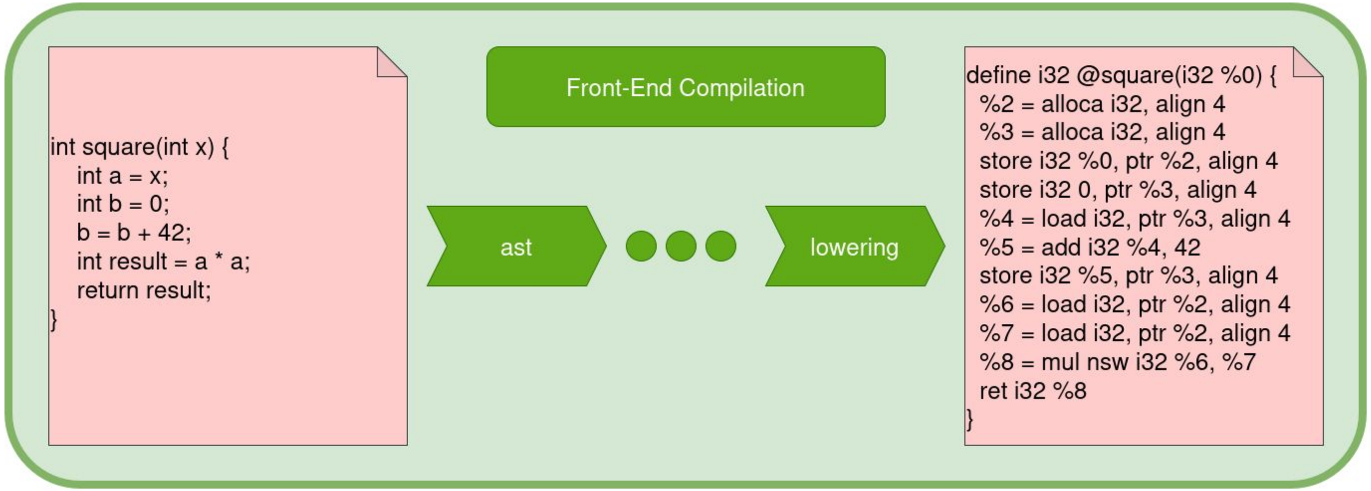
Modern Hardware is Heterogeneous?!



The cooler and more efficient gaming CPU
Intel Core Ultra 200S Series
intel
CORE
ULTRA





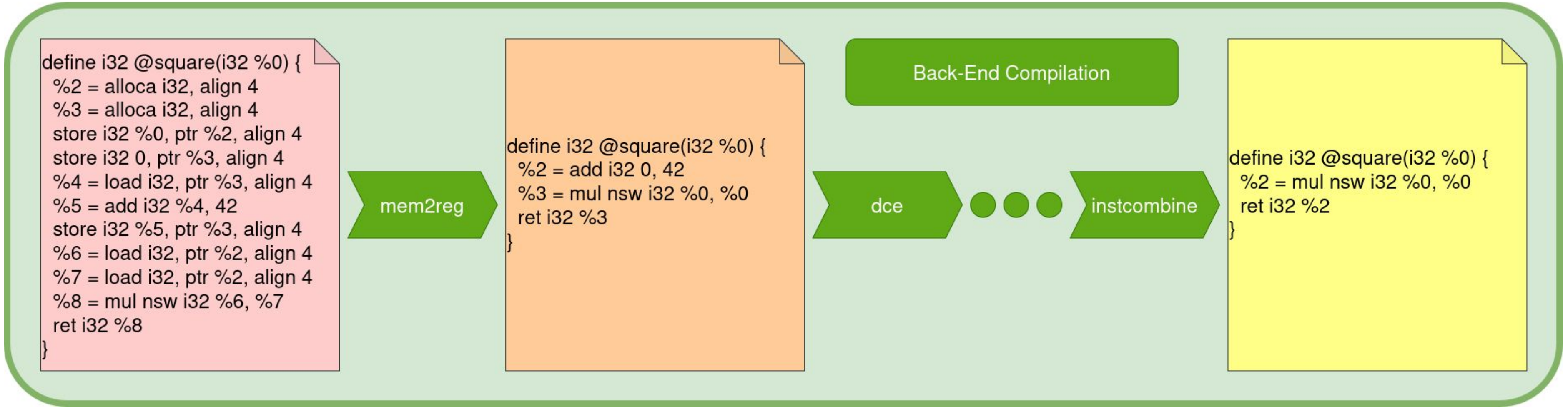


```
int square(int x) {
   int a = x;
   int b = 0;
   b = b + 42;
   int result = a * a;
   return result;
}
```

```
define i32 @square(i32 %0) {
  %2 = alloca i32, align 4
  %3 = alloca i32, align 4
  store i32 %0, ptr %2, align 4
  store i32 0, ptr %3, align 4
  %4 = load i32, ptr %3, align 4
  %5 = add i32 %4, 42
  store i32 %5, ptr %3, align 4
  %6 = load i32, ptr %2, align 4
  %7 = load i32, ptr %2, align 4
  %8 = mul nsw i32 %6, %7
  ret i32 %8
}
```

Optimal pass order for every architecture???

HUGE effort to avoid regressions!



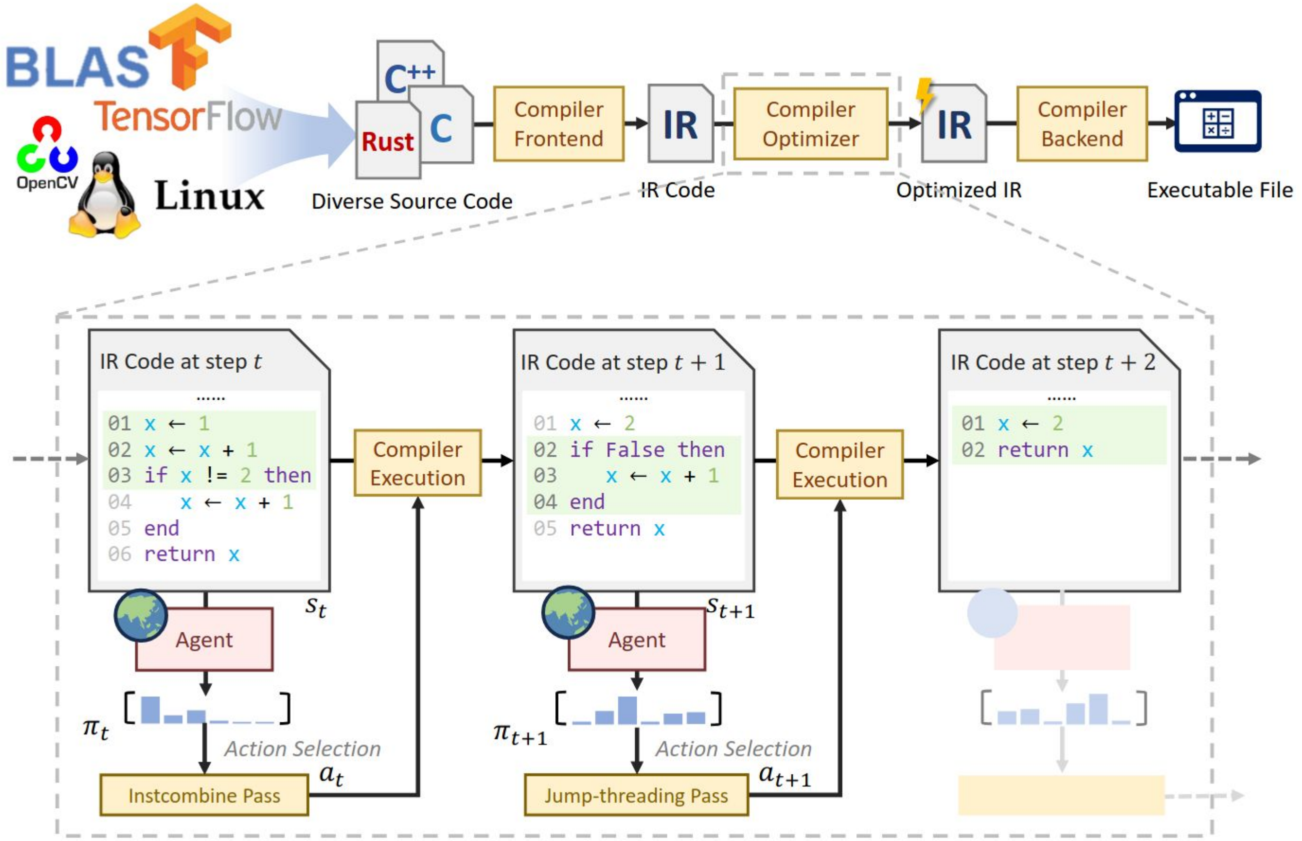
```
define i32 @square(i32 %0) {
  %2 = alloca i32, align 4
  %3 = alloca i32, align 4
  store i32 %0, ptr %2, align 4
  store i32 0, ptr %3, align 4
  %4 = load i32, ptr %3, align 4
  %5 = add i32 %4, 42
  store i32 %5, ptr %3, align 4
  %6 = load i32, ptr %2, align 4
  %7 = load i32, ptr %2, align 4
  %8 = mul nsw i32 %6, %7
  ret i32 %8
}
```

```
define i32 @square(i32 %0) {
  %2 = add i32 0, 42
  %3 = mul nsw i32 %0, %0
  ret i32 %3
}
```

```
define i32 @square(i32 %0) {
  %2 = mul nsw i32 %0, %0
  ret i32 %2
}
```

Solution:

Automatically order passes with reinforcement learning!



```
01 x ← 1
02 x ← x + 1
03 if x != 2 then
04     x ← x + 1
05 end
06 return x
```

```
01 x ← 2
02 if False then
03     x ← x + 1
04 end
05 return x
```

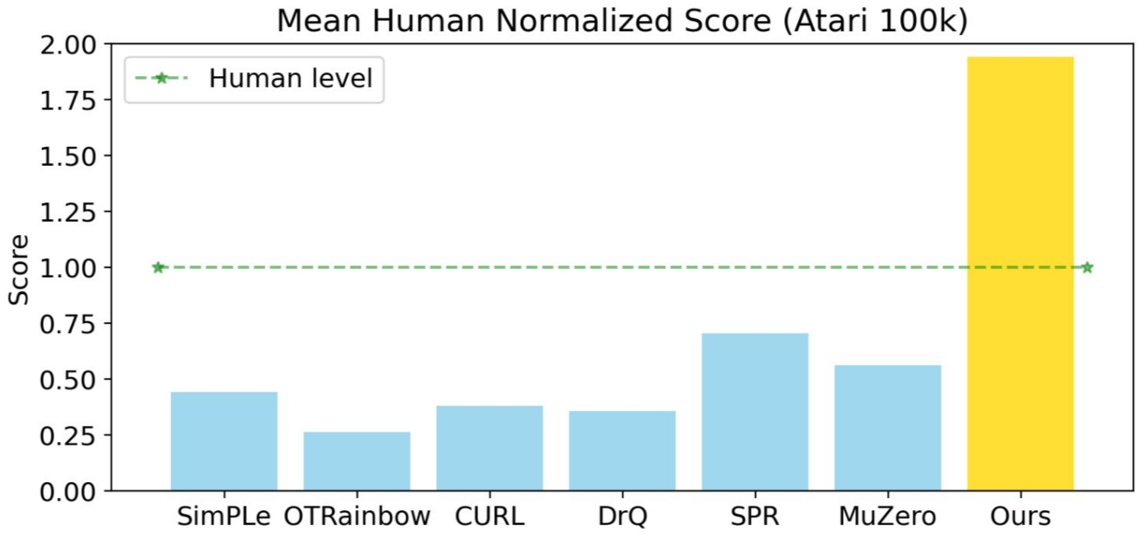
```
01 x ← 2
02 return x
```

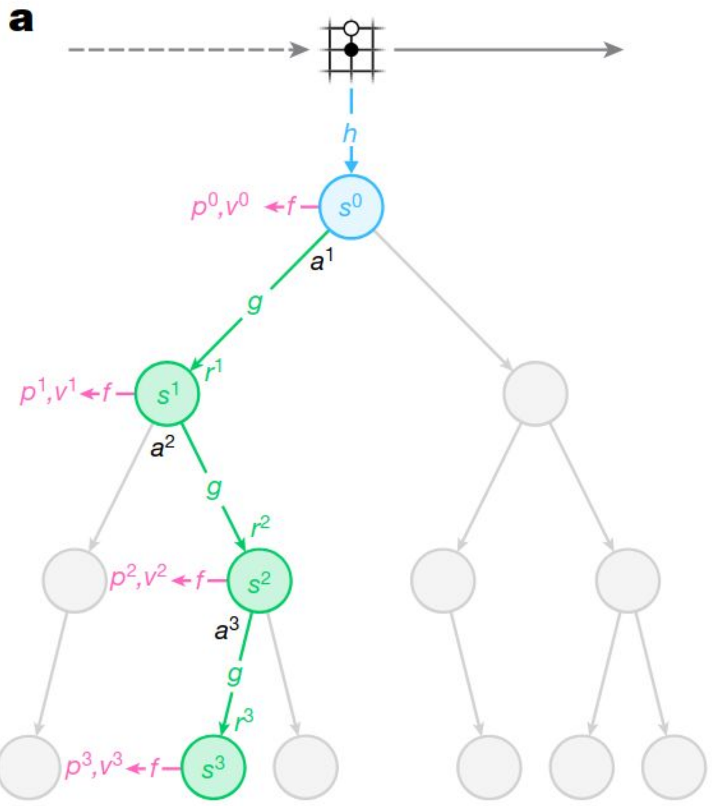
Challenges:

1. High dimensional action space
2. Rewards are sparse
3. Evaluating reward is expensive

Solution:

EfficientZero is a powerful, sample-efficient RL method!





Objective: Minimize binary size

Environment: LLVM pass ordering

Dataset: MiBench (embedded systems)