An Optimising Compiler from Haskell to Java Bytecode

Keith Collister, kc506

Success Criteria

- ✔ Subset of Haskell into Java bytecode
- ~ Non-strict evaluation
- ✘ At least one optimisation

```
class Num a where
    (+) :: a -> a -> a

instance Num Int where
    x + y = ...

instance Num Float where
    ...
```

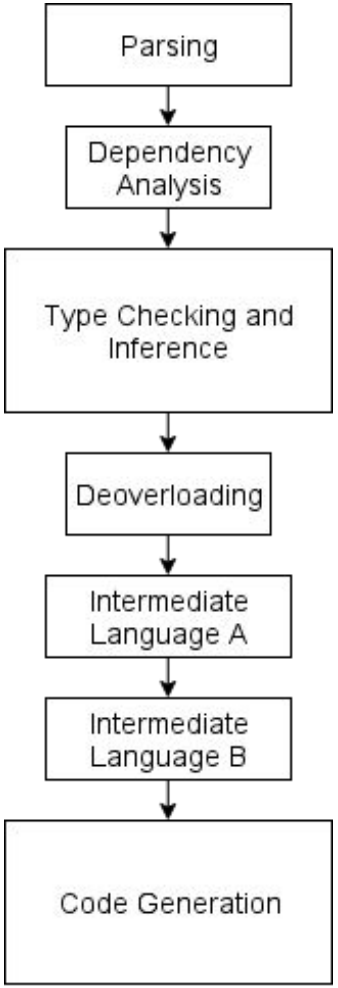
```
data Num a = DNum a
(+) :: Num a -> a -> a -> a
(+) (DNum f) = f

dNumInt = DNum addIntPrim
addIntPrim :: Int -> Int -> Int
addIntPrim = ...

dNumFloat = ...
```

Current Progress

- Lexing+parsing (modified existing parser)
- Type checking+inference
- 2 intermediate languages
- Executable Java 8+ Bytecode
- 5000 LOC + 1000 LOC tests

Extensions

✔ Datatypes
✔ Typeclasses
~ Monads
✘ IO
✘ Strictness Analysis



Remaining work

- Non-strict evaluation (< 1 week)
- Optimisations (2/3 weeks)
  - Peephole
  - Unreachable code/procedure elimination
  - Lambda+Let lifting
  - Strictness Analysis if there’s time
- Evaluation
- Dissertation