Co-dfns
Status Report

Aaron W. Hsu
awhsu@indiana.edu

Dyalog “Conference” 2014
Co-dfns

Demos/Fun

Current Status

Motivation
Why?
Why?

Fostering Research

Tool of Thought

Modern Architectures

Push APL to the ⍢≡

More POWER!

Massive Parallelism
What?
Current Status

ARRAY '14
Black Scholes
Runtime design
C Generation
Mystika
Type system
A little more...;-)
Optimizations
Optimizations
Demos/Fun

Co-dfns

Current Status

Motivation
Choose your own adventure demo

Basic Compiler Example
Black Scholes Benchmark

Special Sauce

Mystika
Compiler Design/Development/Architecture
Brief Interlude

'Tis the dream of each programmer
Before his life is done,
To write three lines of APL
And make the darn thing run.
Fun Stuff

Compiler Sans Parser → 187 lines
Runtime System → 500-900 lines
A compiler with almost no branching/recursion
A **real** solution to APL Type Systems
Leverages Index-of and Key/Rank
Plenty of Function Trains
Trivially switch from GPU to CPU and back
When?
Thank You

Gratipay.com/arcfide
Github.com/arcfide
Sacrideo.us

SEND MORE CODE!
Community-ready parser
Intermediate Reuse

\[ D_1 \leftarrow (\log S \div X) + (r + 2 \div v \times 2) \times T \]

\[ \text{int } p(A \ast z, A \ast l, A \ast r) \]

\[ \text{div}(t1, s, x) \]

\[ \text{log}(t1, \text{NULL}, t1) \]

\[ T_1 \leftarrow \log S \div X \]

\[ D_1 \leftarrow T_1 \times T \]

\[ T_1 \leftarrow r + 2 \div v \times 2 \]

\[ D_1 \leftarrow T_1 + D_1 \]
Frame Size Reduction

Register allocation for array variables
Reduce function call overhead
Better memory locality

\{X \leftarrow \alpha \times \omega \; \diamond \; Y \leftarrow 2 \times X \; \diamond \; S \rho Y\}
Scalar Function Fusion

D1←(⊗S÷X)+(r+2÷v*2)×T

D1←{S←⍵⌷,S ⋄ X←⍵⌷,X ⋄ T←⍵⌷,T
(⊗S÷X)+(r+2÷v*2)×T
}¨⍳×/⍴T
Stack Functions/Operators

g←{a←7×⍵ ◊ b←3×⍺ ◊ f←{ω+a+b} ◊ _ f _}

f←{ ⍝ Depth 1
    ω+(0 ⎕ENV 1)+(1 ⎕ENV 1)
}

g←{a←7×⍵ ◊ b←3×⍺ ◊ f(_ _ _ (⎕ENV 0))}
Lazy Data Copying

Structure primitives don't touch data values
Don't copy data until needed (copy on write)
Allows very fast manipulation of arrays
Should be possible without reference counting