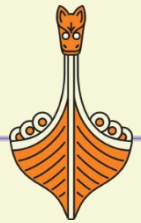
DYALOG
Deerfield Beach
2013



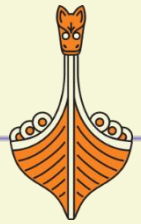
The Compiler Project

or

Reducing Interpreter Overhead

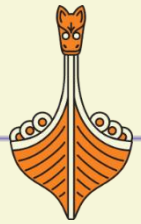
Jay Foad

Dyalog



Goals

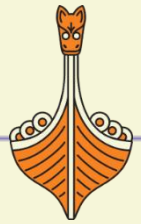
- Reduce interpreter overhead
- Enable high level optimisations



Motivating examples

```
loop←{⍵=0:0 ⋄ ∇⍵-1}
```

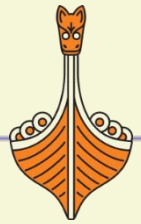
```
mean←{(+⌿⍵)÷≢⍵}
```



Interpreter overhead

Extra parentheses

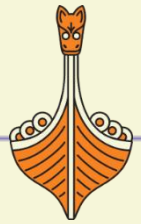
```
mean←{(((+)⌿(⍵))÷(≢(⍵)))}
```



Interpreter overhead

Blank lines and comments

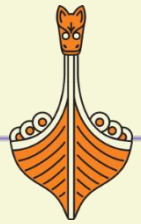
```
mean←{
      ⍝ Calculate the mean
      (+⌿⍵)÷≢⍵
}
```



Interpreter overhead

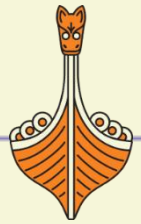
Local names

```
mean←{
      sum←+⌿⍵
      num←≢⍵
      sum÷num
}
```

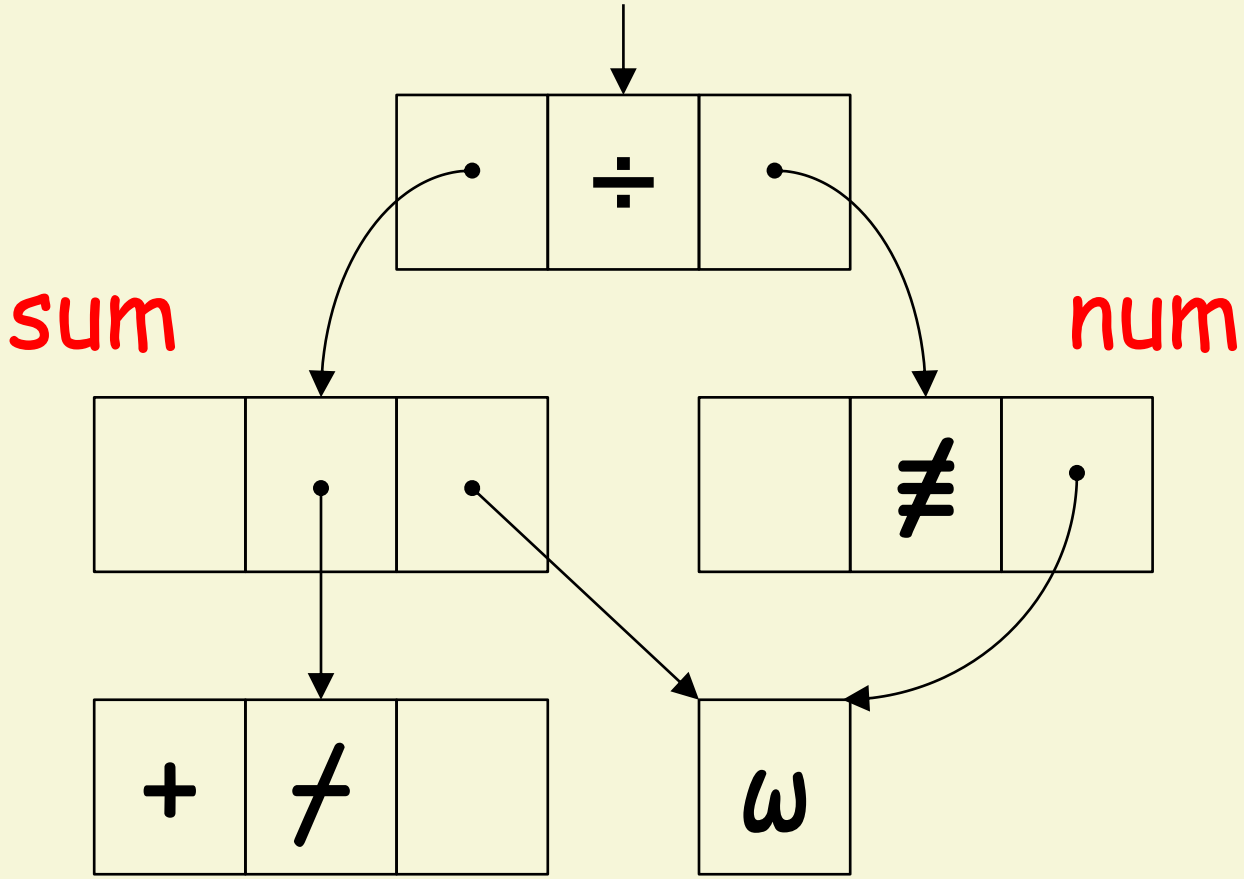


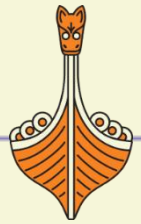
Parsing APL

```
A  B  C
1  2  3
1  +  3
⊃  ⍛  3
+  /  3
⊃  ⍤  ≡
```



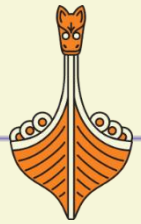
Compiling: parse tree





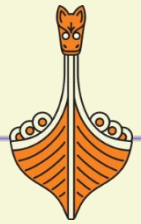
Compiling: bytecode

```
0002: 00002004 rel Larg
0003: 000069C4 cpy PFUNCTION, rawlst[3] // +≠
0004: 00000545 cpy slot[0], Rarg
0005: 00000003 eval
0006: 00000444 mov Rarg, slot[0]
0007: 00002465 mov slot[1], Rslt
0008: 00001F03 eval 0x1F // ≠
0009: 00002424 mov Larg, slot[1]
000A: 00006044 mov Rarg, Rslt
000B: 00000503 eval 0x05 // ÷
000C: 00000002 ret
```



Speed of compiled code

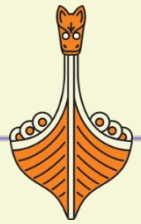
Expression	Before (ns)	After (ns)	Factor
`mean 1 2 3 4`	1966	825	2.38
`root 10`	1155	772	1.49
`easter 2013`	13420	9384	1.43



Limitations

Global names

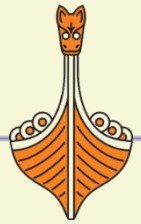
```
compidn←{
      base←days 1970 1 1
      stamp←⍬⍴2↓⎕FRDCI ⍺ ⍵
      base+stamp÷×/1 3/24 60
}
```



Limitations

Execute ⍎ and system functions

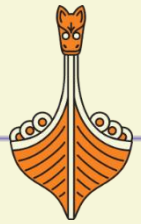
```
time←{
      t←⎕AI
      r←⍎⍵
      ⎕←⎕AI-t
      r
}
```



Limitations

Namespace references

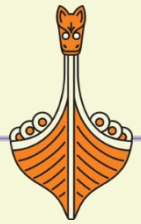
```
run←{
      ⍵.f ⍵.x
}
```



Limitations

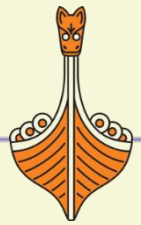
Selective assignment

```
stars←{
      t←⍵
      ((' '≠t)/⍳t)←'*'
      t
}
```



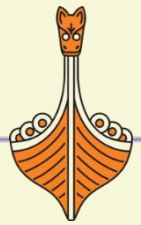
Gallery

```
easter←{                          ⍝ Easter Sunday in year ⍵.
    G←1+19|⍵                      ⍝ year "golden number" in 19-year Metonic cycle.
    C←1+⌊⍵÷100                    ⍝ Century: for example 1984 → 20th century.

    X←¯12+⌊C×3÷4                  ⍝ number of years in which leap year omitted.
    Z←¯5+⌊(5+8×C)÷25              ⍝ synchronises Easter with moon's orbit.

    S←(⌊(5×⍵)÷4)-X+10             ⍝ find Sunday.
    E←30|(11×G)+20+Z-X            ⍝ Epact.
    F←E+(E=24)∨(E=25)^G>11        ⍝   (when full moon occurs).

    N←(30×F>23)+44-F              ⍝ find full moon.
    N←N+7-7|S+N                   ⍝ advance to Sunday.

    M←3+N>31                      ⍝ month: March or April.
    D←N-31×N>31                   ⍝ day within month.
    ↑10000 100 1+.×⍵ M D          ⍝ yyyymmdd.
}
```



Gallery: local dfns

```
packU←{⎕IO←0      ⍝ Unique packer.
     cmp←{
          u←∪⍵,⍵
          (⍴⍵)u(u⍳⍵,⍵)
     }
     exp←{
          (0⊃⍵)⍴(1⊃⍵)[2⊃⍵]
     }
     α←1 ⋄ α:cmp ⍵ ⋄ exp ⍵
}
```



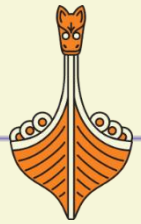
User interface

- Disabled off by default
- Enable auto-compilation:

```
      400⌶2
```

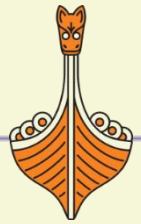
- Compile specified functions:

```
      2(400⌶)'foo'
      2(400⌶)⎕NL 3
```



High level optimisations

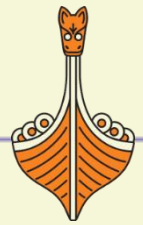
```
⍝ Surface area of k-sphere.
ksphere←{
      n←⍺+1
      pi←(○1)*n÷2
      n×(⍵*⍺)×pi÷!n÷2
}
```



High level optimisations

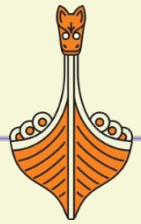
```
f←{
      1+≠ω
}
```

simple scalar
numeric
integer
positive



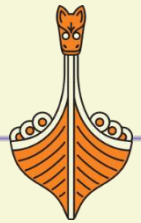
High level optimisations

```
f←{  +/∧\' '=⍵  }

g←{  +/∧\⍵=' '  }
```



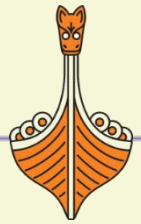
When can I get it?

- In Version 14.0
- Disabled by default

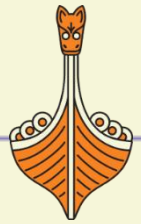


What can I do now?

- Think in a pure functional way
- Use dfns
- Show us your code!

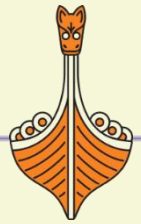


That’s all, folks!



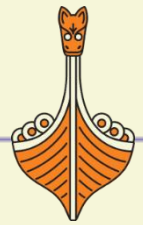
Speed of compilation

```
      )load dfns.dws
      ≠⎕NL 3
197
      +/≠∘⎕CR¨↓⎕NL 3
4562
      iscompiled←1∘(400I)
      compile←2∘(400I)
      +/iscompiled ⎕NL 3
0
      compile time ⎕NL 3
00.00
      +/iscompiled ⎕NL 3
71
```



Functional tradfns

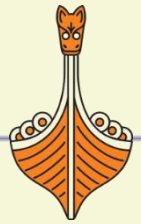
```
∇ r←mean y;sum;num
  sum←+/y
  num←≢y
  r←sum÷num
∇
```



Debugging compiled code

☺ Precise error locations

☹ Can’t suspend in compiled code



Declarations

```
f←{
      :Function foo
      foo 1+⍵
}
```