CS 421 Lecture 15: APL

- Lecture outline
  - Functional programming
  - APL

Functional programming

- "The assignment statement splits programming into two worlds. The first world comprises the right sides of assignment statements. This is an orderly world of expressions, a world that has useful algebraic properties... It is the world in which most useful computation takes place.
- "The second world ... is the world of statements. … This world of statements is a disorderly one, with few useful mathematical properties."

John Backus (creator of Fortran), "Can Programming be liberated from the von Neumann Style? A Functional Style and its Algebra of Programs."

Turing Award lecture, 1977.

APL

- 1960 – Ken Iverson – “A Programming Language”
- Computations on matrices using operators that have matrix arguments.
- Defined a set of operators on matrices, plus a typeface for those operators, and built terminals

APL operations

- Binary operations on numbers extended naturally to matrices
  - Comparison and boolean ops treated as arithmetic
- Reduction operations: +/, ×/, ^/, ...
  - For vectors, put operator between every element
  - For matrices, reduce each row
- Compression: B / V
  - selects elements (or rows) of V where B = 1
- No precedence rules
  - evaluate right-to-left

APL font

ABCDEFGHIJKLMNOPQRSTUVWXYZ

0123456789

≥≤≠<>=ιρ⌀^/÷←*×+−

APL operators

- Comparison
  - ≥≤≠<>=
- Arithmetic
  - ^÷*×+−
- Assignment
  - ←
- Index generation
  - ι
- Dimension (monadic) or restructure (dyadic)
  - ρ
- Transpose
  - ∅
- Compression (dyadic), reduction
  - /

APL examples

- $1+M$
- $(+/V)\div n$
- $(+/V)\div\rho V$
- $(((V\div 2)\times 2)=V)/V$

APL examples

- $prime\ n = \wedge/(0 \neq n \div (1 + \iota(n-2)))$

APL examples

- Subscripting: V[V'] – elements of V in positions given by V'.
- *reverse V* = *V*[1+(ρ*V*)−ιρ*V*]

APL examples

- Dyadic ρ – “restructure”
  - VρA returns a value with shape V, values drawn from A
- `2 3 ρ ι6`
- `2 3 ρ ι5`
- `(2ρn) ρ 1,nρ0`

APL examples

- Assignment
  - $\leftarrow$
- Transpose
  - $\varnothing$
- $(\varnothing M) = M \leftarrow (2\rho n)\rho\iota n$

APL examples (in OCaml)

```
let zero = newint 0;;
let four = newint 4;;
let a = rho(newveci [2;3]) (indx (newint 6));;
let v = newveci [2;4;6];;
let c = newveci [1;0];;
let d = newveci [1;0;1];;
a *@ a
v -@ one
a >@ four
!+v
```

- Arithmetic operators: `*@ -@ >@`
- Reduction: `+`

APL examples (in OCaml)

```
maxR a
d % v
c % a
shape a
ravel a
rho (shape a) v
rho (shape v) c
a ^@ c
```

- Max, min reductions : `maxR, minR`
- Compression: `%`
- Catenation: `^@`

APL examples (in OCaml)

```
indx (newint 5)
trans a
v @@ (indx two)
a @@ one
(trans a) @@ (indx two)
```

- Subscript: `@@`

APL examples (in OCaml)

```
let incr a = a +@ (newint 1);;
let fac n = !* (indx n);;
let avg v = (!+v) /@ (shape v);;
let reverse v =
  let sz = (shape v) @@ one
  in v @@ (incr (sz -@ (indx sz)));;

let prime n = !& (zero <>@ (n %@ (incr
  (indx (n -@ two)))));;
```

APL reference

- Posted alongside lecture slides on the web site
- May come in handy for a future MP…