



A Look at Array Languages

APL vs BQN vs J

<https://github.com/codereport/Content>

<https://github.com/codereport/array-language-comparisons/>

Conor Hoekstra



code_report

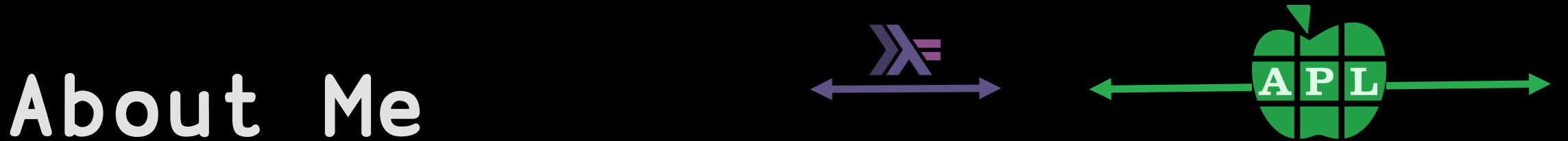
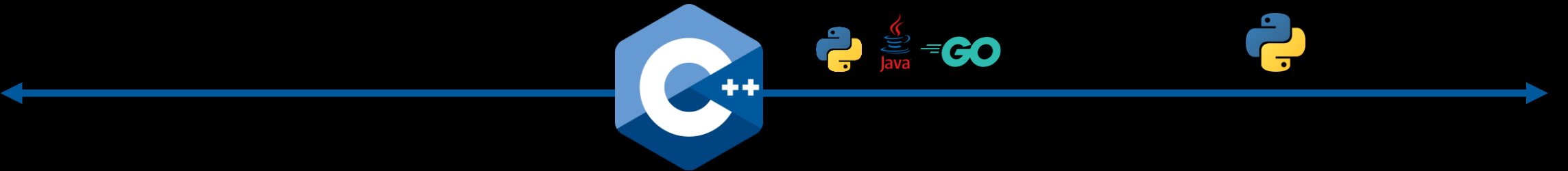
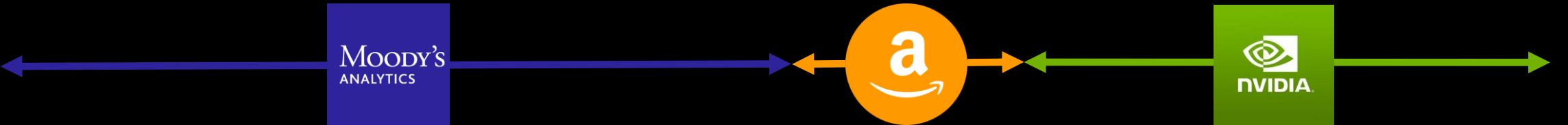
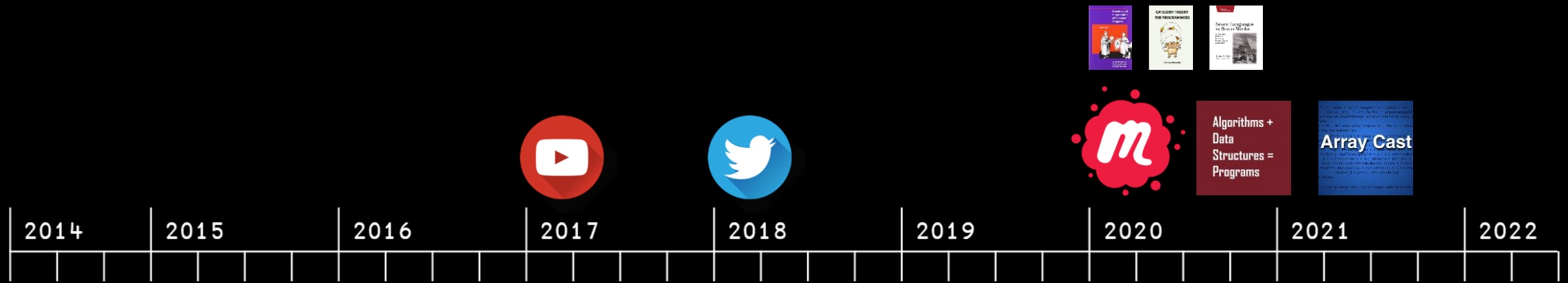


codereport

A Look at Array Languages

APL vs BQN vs J (vs Julia vs R vs NumPy)





About Me

Conor Hoekstra / @code_report





2319. Check if Matrix Is X-Matrix

A square matrix is said to be an **X-Matrix** if both of the following conditions hold:

1. All the elements in the diagonals of the matrix are non-zero.
1. All other elements are 0.

Given a square matrix, return **true** if grid is an X-Matrix. Otherwise, return **false**.

<https://leetcode.com/problems/check-if-matrix>

2	0	0	1
0	3	1	0
0	5	2	0
4	0	0	2

5	7	0
0	3	1
0	5	0

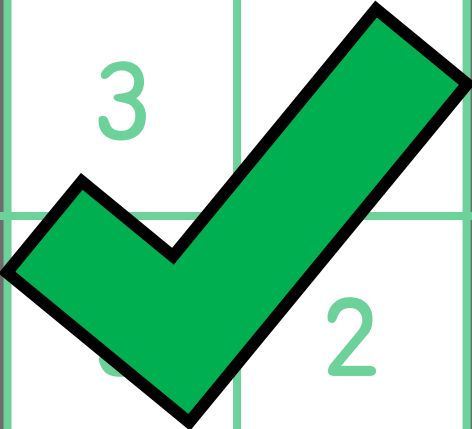
2	0	0	1
0	3	1	0
0	5	2	0
4	0	0	2

5	7	0
0	3	1
0	5	0


2	0	0	1
0	3	1	0
0	5	2	0
4	0	0	2

5	7	0
0	3	1
0	5	0

2	0	0	1
0	3		0
0		2	0
4	0	0	2



5	7	0
0		1
0	5	0





```
mat ← 4 4p2 0 0 1 0 3 1 0 0 5 2 0 4 0 0 2
```



mat ← 4 4 2 0 0 1 0 3 1 0 0 5 2 0 4 0 0 2

2	0	0	1
0	3	1	0
0	5	2	0
4	0	0	2



mat

2	0	0	1
0	3	1	0
0	5	2	0
4	0	0	2



\neq mat

4



\neq mat

1 2 3 4



$(\iota \neq \text{mat}) \circ ., \iota \neq \text{mat}$

1 1	1 2	1 3	1 4
2 1	2 2	2 3	2 4
3 1	3 2	3 3	3 4
4 1	4 2	4 3	4 4



$$(\text{⍳mat}) \circ . = \text{⍳mat}$$

1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1



○ . = ~ ≠ mat

1	0	0	0
0	1	0	0
0	0	1	0
0	0	0	1



$\phi \circ . = \sim \nabla \neq \text{mat}$

0 0 0 1

0 0 1 0

0 1 0 0

1 0 0 0



$\phi \circ . = \sim \iota \neq \text{mat}$

$\circ . = \sim \iota \neq \text{mat}$

0 0 0 1

1 0 0 0

0 0 1 0

0 1 0 0

0 1 0 0

0 0 1 0

1 0 0 0

0 0 0 1



$(\phi \circ . = \sim \iota \neq \text{mat}) \quad \lceil \quad \circ . = \sim \iota \neq \text{mat}$

1	0	0	1
0	1	1	0
0	1	1	0
1	0	0	1



$(\phi \circ . = \sim \iota \neq \text{mat}) \vee (\vdash \circ . = \sim \iota \neq \text{mat})$

1	0	0	1
0	1	1	0
0	1	1	0
1	0	0	1



$(\phi \vdash) \circ . = \approx \neq \text{mat}$

1	0	0	1
0	1	1	0
0	1	1	0
1	0	0	1



mat

2	0	0	1
0	3	1	0
0	5	2	0
4	0	0	2



1lmat

1	0	0	1
0	1	1	0
0	1	1	0
1	0	0	1



1Lmat

$(\phi \vdash) \circ . = \sim \neq \text{mat}$

1 0 0 1

1 0 0 1

0 1 1 0

0 1 1 0

0 1 1 0

0 1 1 0

1 0 0 1

1 0 0 1



$(1\text{Lmat}) \equiv (\phi\lceil\vdash) \circ . = \sim \neg \text{mat}$

1



$(1 \text{ Lmat}) \equiv (\phi \vdash) \circ . = \sim \neq \text{mat}$



$$\{ (1 \mid \omega) \equiv (\phi \vdash) \circ . = \sim \neq \omega \}$$



checkMatrix $\leftarrow \{ (1 \downarrow \omega) \equiv (\phi \uparrow \vdash) \circ . = \sim \wr \neq \omega \}$



CheckMatrix $\leftarrow \{ (1 \mid \mathbb{X}) \equiv \phi \circ \lceil = \ulcorner \sim \Downarrow \neq \mathbb{X} \}$



```
checkMatrix =. {{ (1<.y)-:(>.|.)=/~i.#y }}
```



`checkMatrix ← {(1⊔ω)≡(ϕ⌈⊔)∘.=⌘⌈≠ω}`



`CheckMatrix ← {(1⊔⊞)≡ϕ⊔⌈=⌈~⌈≠⊞}`



`checkMatrix =. {{ (1<⌈y)-⌈:(>⌈⌈.)=⌈~⌈.⌈y }}`



$$\{ (1 \lfloor \omega) \equiv (\phi \lceil \vdash) \circ . = \ddot{\sim} \wr \neq \omega \}$$



$$\{ (1 \lfloor \mathbb{X}) \equiv \phi \circ \lceil = \lceil \sim \updownarrow \neq \mathbb{X} \}$$



$$\{ \{ (1 < . y) - : (> . | .) = / \sim i . \# y \} \}$$



$$\{ (1 \mid \omega) \equiv (\phi \vdash) \circ . = \sim \wr \neq \omega \}$$



$$\{ (1 \mid \mathbb{X}) \equiv \phi \circ \vdash = \lceil \sim \updownarrow \neq \mathbb{X} \}$$



$$\{ \{ (1 < . y) - : (> . | .) = / \sim i . \# y \} \}$$



$$\{(1 \mid \omega) \equiv (\phi \vdash) \circ . = \ddot{\sim} \wr \neq \omega\}$$



$$\{(1 \mid \mathbb{X}) \equiv \phi \circ \vdash = \ulcorner \sim \updownarrow \neq \mathbb{X}\}$$



$$\{\{ (1 < . y) - : (> . | .) = / \sim i . \# y \} \}$$



Rank Polymorphism Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory



Rank Polymorphism

Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory



$$\{(1 \mid \omega) \equiv (\phi \vdash) \circ . = \ddot{\sim} \wr \neq \omega\}$$



$$\{(1 \mid \mathbb{X}) \equiv \phi \circ \vdash = \ulcorner \sim \updownarrow \neq \mathbb{X}\}$$



$$\{\{ (1 < . y) - : (> . | .) = / \sim i . \# y \} \}$$

Scalar (R0) \downarrow Matrix (R2)



$\{ (1 \downarrow \omega) \equiv (\phi \uparrow \vdash) \circ . = \sim \wr \neq \omega \}$



$\{ (1 \downarrow \mathbb{X}) \equiv \phi \circ \uparrow = \uparrow \sim \updownarrow \neq \mathbb{X} \}$



$\{ \{ (1 < . y) - : (> . | .) = / \sim i . \# y \} \}$



Matrix (R2) \uparrow Matrix (R2)



1 L X

map (map (min 1)) x





Rank Polymorphism

Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory



Rank Polymorphism

Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory



$$\{(1 \mid \omega) \equiv (\phi \vdash) \circ . = \ddot{\sim} \wr \neq \omega\}$$



$$\{(1 \mid \mathbb{X}) \equiv \phi \circ \vdash = \ulcorner \sim \updownarrow \neq \mathbb{X}\}$$



$$\{\{ (1 < . y) - : (> . | .) = / \sim i . \# y \} \}$$



$$\{(1 \mid \omega) \equiv (\phi \vdash) \circ . = \sim \neg \neq \omega\}$$



$$\{(1 \mid \mathbb{X}) \equiv \phi \circ \vdash = \lceil \sim \updownarrow \neq \mathbb{X}\}$$



$$\{\{(1 < . y) - : (> . | .) = / \sim i . \# y\}\}$$

1	2	3	4	5	6
1	2	3	4	5	6

1 2 3 4 5 6

1

2

3

4

5

6

	1	2	3	4	5	6
1	2	3	4	5	6	7
2	3	4	5	6	7	8
3	4	5	6	7	8	9
4	5	6	7	8	9	10
5	6	7	8	9	10	11
6	7	8	9	10	11	12

	Name		Reduce		Scan		Outer Product
APL	Operators Adverbs	●	<code>/</code> (reduce) <code>≠</code> (reduce first)	●	<code>\</code> (scan) <code>≡</code> (scan first)	●	<code>∘.</code> (outer product)
J	Adverbs & Conjunctions	●	<code>/</code> (insert)	●	<code>\</code> (prefix)	●	<code>/</code> (table)
BQN	Modifiers	●	<code>`</code> (fold) <code>~</code> (insert)	●	<code>`</code> (scan)	●	<code>⌈</code> (table)
Q	Iterators	●	<code>/</code> over	●	<code>\</code> scan	●	<code>/: \:</code>
Julia	Functions	●	reduce	●	accumulate	●	broadcast
NumPy	Functions	●	<code>_.reduce()</code>	●	<code>_.accumulate()</code>	●	<code>_.outer()</code>
R	Functions	●	Reduce	●	Reduce(accumulate=TRUE)	●	outer
Nial	Transformers	●	REDUCE	●	ACCUMULATE	●	OUTER
Futhark	Functions SOAC	●	foldl/r reduce(_comm)	●	scan	●	outer_product
SaC		●	-	●	-	●	-



Rank Polymorphism

Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory



Rank Polymorphism Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory



$$\{(1 \mid \omega) \equiv (\phi \vdash) \circ . = \ddot{\sim} \wr \neq \omega\}$$



$$\{(1 \mid \mathbb{X}) \equiv \phi \circ \vdash = \ulcorner \sim \updownarrow \neq \mathbb{X}\}$$



$$\{\{ (1 < . y) - : (> . | .) = / \sim i . \# y \} \}$$



$$\{(1 \sqsubset \omega) \equiv (\phi \vdash) \circ . = \sim \wr \neq \omega\}$$



$$\{(1 \sqsubset \mathbb{X}) \equiv \phi \circ \vdash = \lceil \sim \updownarrow \neq \mathbb{X}\}$$



$$\{\{(1 < . y) - : (> . | .) = / \sim i . \# y\}\}$$



$$\{(1 \mid \omega) \equiv (\phi \vdash) \circ . = \sim \wr \neq \omega\} \quad \phi$$



$$\{(1 \mid \mathbb{X}) \equiv \phi \circ \vdash = \lceil \sim \updownarrow \neq \mathbb{X}\} \quad \Sigma$$



$$\{\{(1 < . y) - : (> . | .) = / \sim i . \# y\}\} \quad S$$



$$\{(1 \mid \omega) \equiv \phi \circ \lceil \sim \circ . = \sim \lceil \neq \omega\} \quad D+W$$



$$\{(1 \mid \omega) \equiv (\phi \lceil \vdash) \circ . = \sim \lceil \neq \omega\} \quad \phi$$



$$\{(1 \mid \mathbb{X}) \equiv \phi \circ \lceil = \lceil \sim \updownarrow \neq \mathbb{X}\} \quad \Sigma$$



$$\{(1 \mid \mathbb{X}) \equiv \lceil \circ \phi = \lceil \sim \updownarrow \neq \mathbb{X}\} \quad S$$



$$\{\{(1 < . y) - : (> . | .) = / \sim i . \# y\}\} \quad S$$

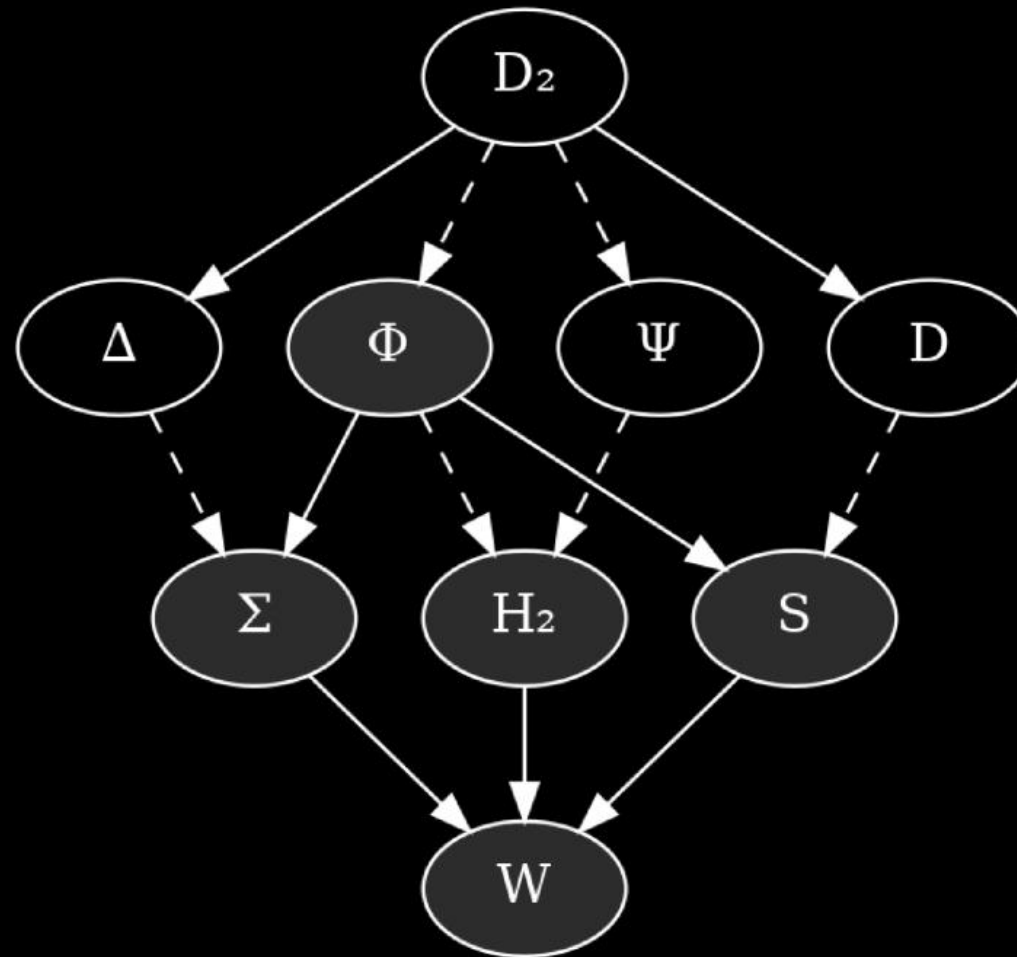


Figure 4.2: D_2 combinator hierarchy with Δ , Σ and H_2 .

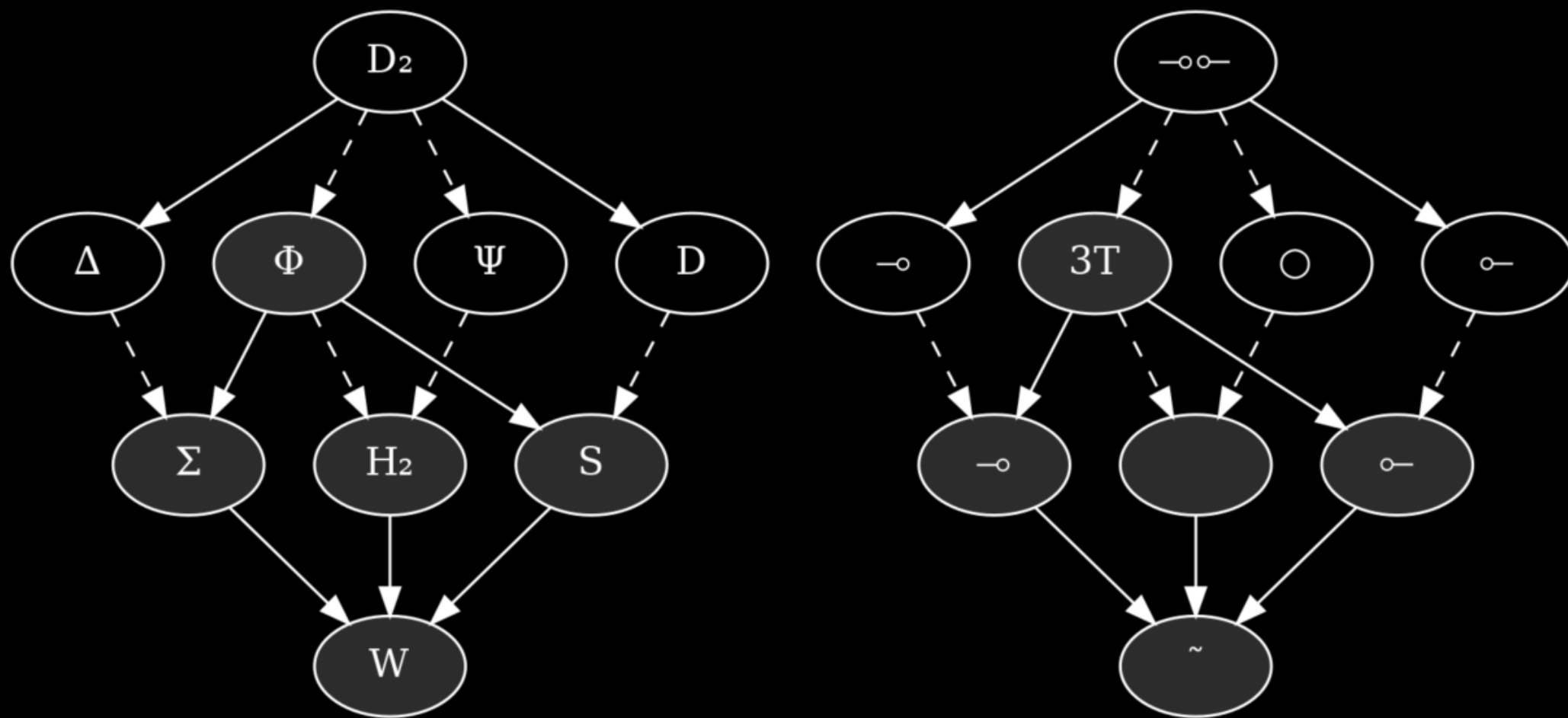


Figure 4.3: D_2 combinator hierarchy vs BQN spellings.



Rank Polymorphism Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory



Rank Polymorphism Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory



$$\{(1 \mid \omega) \equiv (\phi \vdash) \circ . = \ddot{\sim} \wr \neq \omega\}$$



$$\{(1 \mid \mathbb{X}) \equiv \phi \circ \vdash = \ulcorner \sim \updownarrow \neq \mathbb{X}\}$$



$$\{\{(1 < . y) - : (> . | .) = / \sim i . \# y\}\}$$



$$\{(1 \mid \omega) \equiv (\phi \vdash) \circ . = \sim \wr \neq \omega\}$$



$$\{(1 \mid \mathbb{X}) \equiv \phi \circ \vdash = \vdash \sim \updownarrow \neq \mathbb{X}\}$$



$$\{\{(1 < . y) - : (> . | .) = / \sim i . \# y\}\}$$



ϕ



ϕ

J

|.



ϕ

1

2

3



ϕ

1

2

3

J

|.

1

2

3



ϕ

1

2

3

→

3

2

1



ϕ

1

2

3

→

3

2

1



|.

1

2

3

→

3

2

1



ϕ

1

2

3

4

5

6



ϕ

1

2

3

4

5

6



|.

1

2

3

4

5

6

 ϕ

1

2

3

 \rightarrow

3

2

1

4

5

6

6

5

4

 ϕ

1

2

3

 \rightarrow

4

5

6

4

5

6

1

2

3



|. .

1

2

3

 \rightarrow

4

5

6

4

5




6

1

2

3

Leading
Axis

	Reverse Columns	Reverse Rows
	\ominus	ϕ $\ominus \circ 1$
	ϕ	ϕ° $\phi \ominus 1$
	$.$	$. " 1$



Rank Polymorphism Operators

(Modifiers | Adverbs / Conjunctions)

Combinators

Leading Axis Theory



Arrays



```
auto check_matrix(auto grid) -> bool {  
    for (int i = 0; i <= grid.size(); ++i) {  
        for (int j = 0; j <= grid.size(); ++j) {  
            if (i == j or (grid.size() - i == j)) {  
                if (grid[i][j] == 0) {  
                    return false;  
                }  
            } else if (grid[i][j] != 0) {  
                return false;  
            }  
        }  
    }  
    return true;  
}
```



```
auto check_matrix(auto grid) -> bool {  
    for (int i = 0; i < grid.size(); ++i) {  
        for (int j = 0; j < grid.size(); ++j) {  
            if (i == j or (grid.size() - i == j)) {  
                if (grid[i][j] == 0) {  
                    return false;  
                }  
            } else if (grid[i][j] != 0) {  
                return false;  
            }  
        }  
    }  
    return true;  
}
```



```
auto check_matrix(auto grid) -> bool {  
    for (int i = 0; i < grid.size(); ++i) {  
        for (int j = 0; j < grid.size(); ++j) {  
            if (i == j or (grid.size() - i == j + 1)) {  
                if (grid[i][j] == 0) {  
                    return false;  
                }  
            } else if (grid[i][j] != 0) {  
                return false;  
            }  
        }  
    }  
    return true;  
}
```



$$\{ (1 \mid \omega) \equiv (\phi \vdash) \circ . = \sim \wr \neq \omega \}$$



$$\{ (1 \mid \mathbb{X}) \equiv \phi \circ \vdash = \lceil \sim \updownarrow \neq \mathbb{X} \}$$



$$\{ \{ (1 < . y) - : (> . | .) = / \sim i . \# y \} \}$$



Thank you!

<https://github.com/codereport/Content>

<https://github.com/codereport/array-language-comparisons/>

Conor Hoekstra



code_report



codereport



Questions?

<https://github.com/codereport/Content>

<https://github.com/codereport/array-language-comparisons/>

Conor Hoekstra



code_report



codereport



Bonus

<https://github.com/codereport/Content>

<https://github.com/codereport/array-language-comparisons/>

Conor Hoekstra



code_report



codereport



```
function checkmatrix(grid)
    i = size(grid, 1) |> I |> Matrix
    min.(grid, 1) == max.(i, reverse(i, dims=1))
end
```



```
check_matrix <- function(grid) {  
  n = nrow(grid)  
  i = diag(n)  
  return(all(pmin(grid, 1) == pmax(i, i[n:1,])))  
}
```