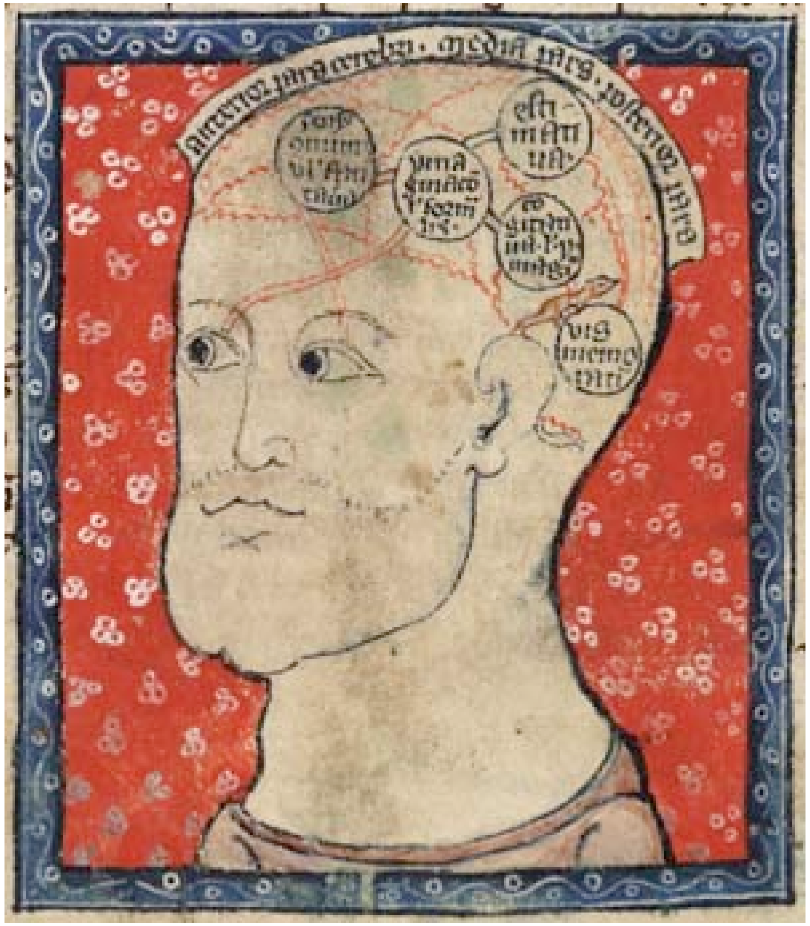
Algebraic metalanguages for routing policy specification

Alexander Gurney
Computer Laboratory, University of Cambridge

Multi-Service Networks 2006
Cosener's House, Abingdon, UK

The Metarouting Idea

- A language for policy
- Build up complex protocol descriptions, as simple algebras combined in known ways
- Susceptible to proof
- Suitable for implementation
- One language, many meanings

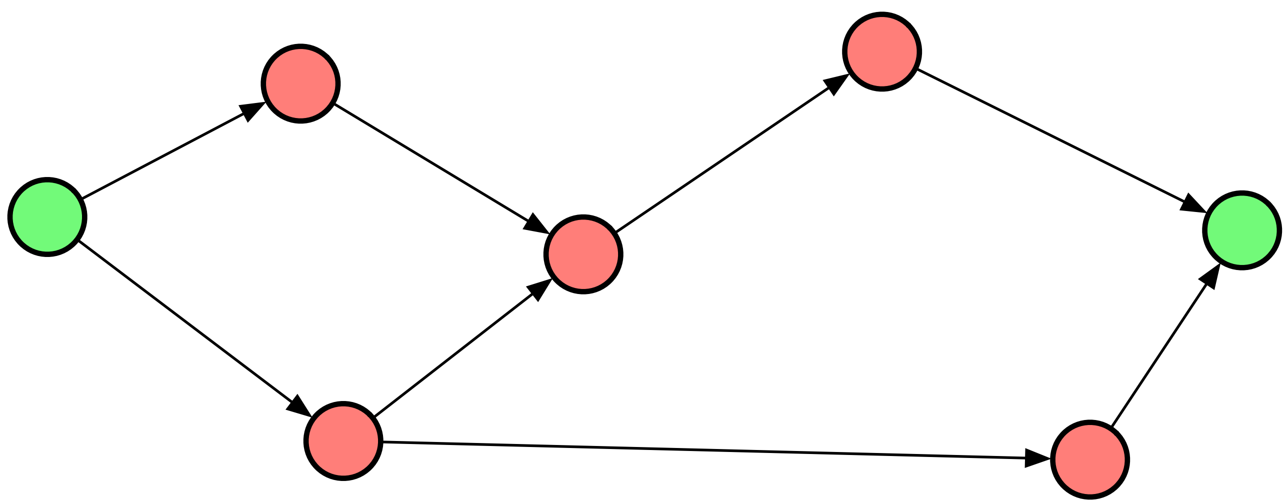


Meanwhile, in the world of mathematics

- The shortest path problem, with numbers (Dijkstra, Bellman / Ford, Moore / Shannon; and related works back to at least 1871)
- Generalised shortest paths (Gondran / Minoux; Carré; Berge; ...)
- All kinds of fun algebra: monoids, semirings, semilattices and other ordered structures, actions, representations
- “If the algebra has property X then we can use algorithm Y to find an optimal solution” for various values of “X”, “Y”, and “optimal”

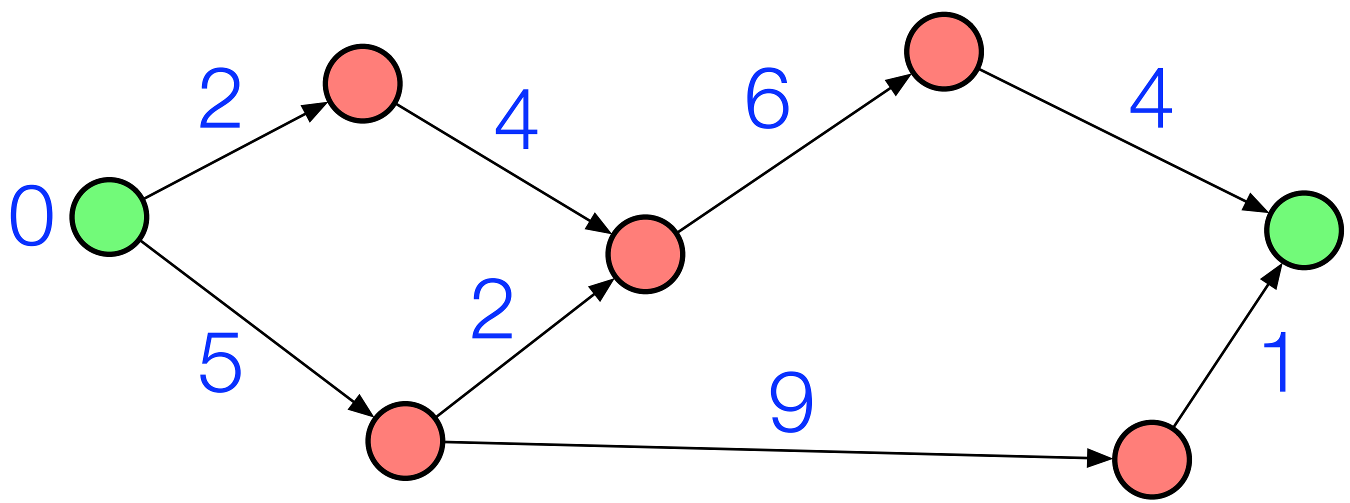
Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, ∧, ⊕, ⊤, 0): two monoids (S, ∧, ⊤), (S, ⊕, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: **($\mathbf{N}^{\infty}$, min, +, ∞, 0)**, (℘(A), ∩, ∪, A, ∅), ($\mathbf{R}^{+}$, max, min, 0, ∞)



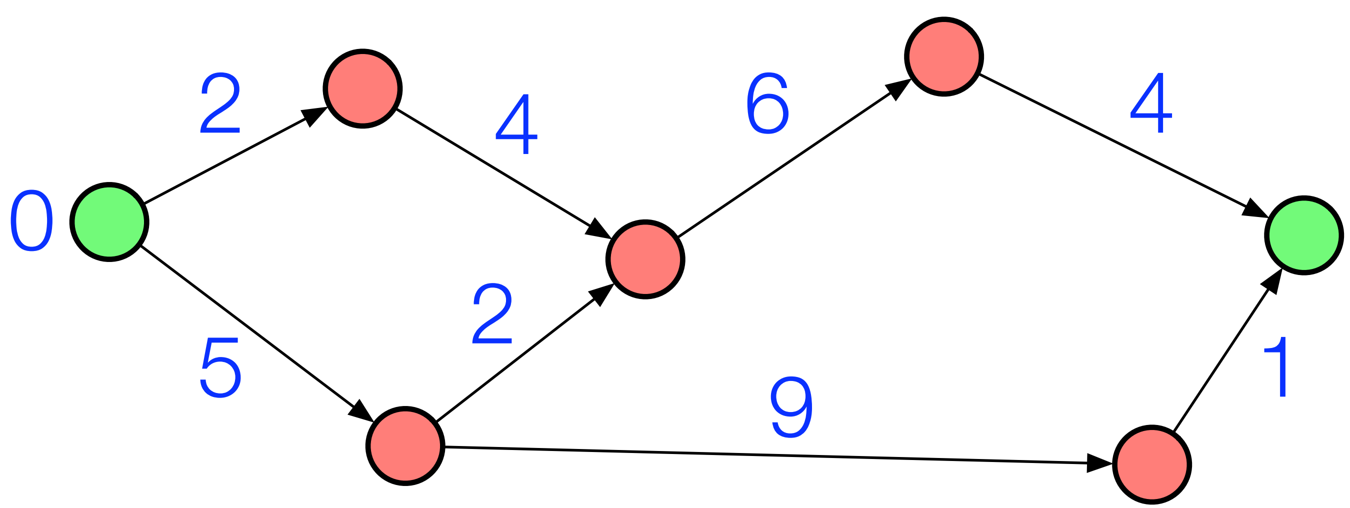
Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, ∧, ⊕, ⊤, 0): two monoids (S, ∧, ⊤), (S, ⊕, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: ($\mathbf{N}^{\infty}$, min, +, ∞, 0), (℘(A), ∩, ∪, A, ∅), ($\mathbf{R}^{+}$, max, min, 0, ∞)



Model I: Path algebra

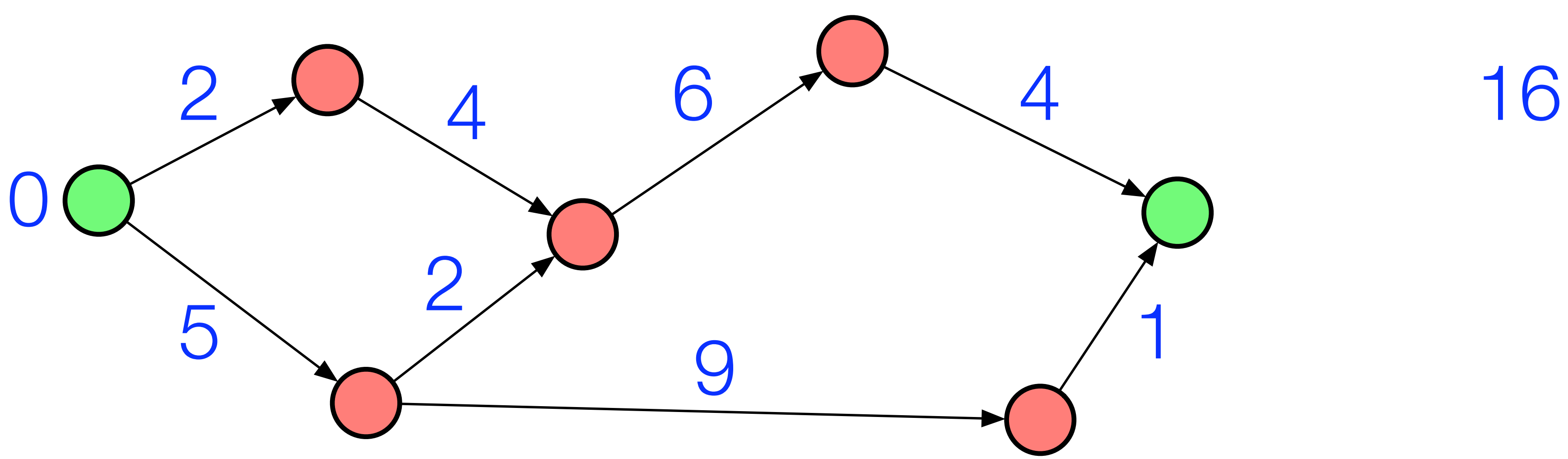
- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, ∧, ⊕, ⊤, 0): two monoids (S, ∧, ⊤), (S, ⊕, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: **($\mathbf{N}^{\infty}$, min, +, ∞, 0)**, (℘(A), ∩, ∪, A, ∅), (**$\mathbf{R}^{+}$**, max, min, 0, ∞)



0+2+4+6+4

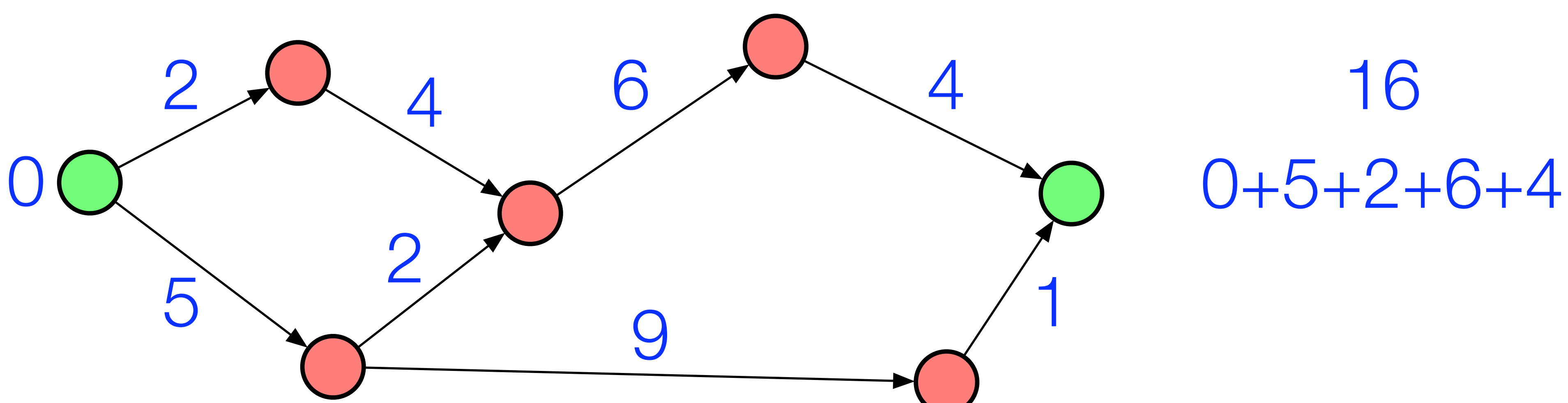
Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, ∧, ⊕, ⊤, 0): two monoids (S, ∧, ⊤), (S, ⊕, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: **($\mathbf{N}^{\infty}$, min, +, ∞, 0)**, (℘(A), ∩, ∪, A, ∅), (**$\mathbf{R}^{+}$**, max, min, 0, ∞)



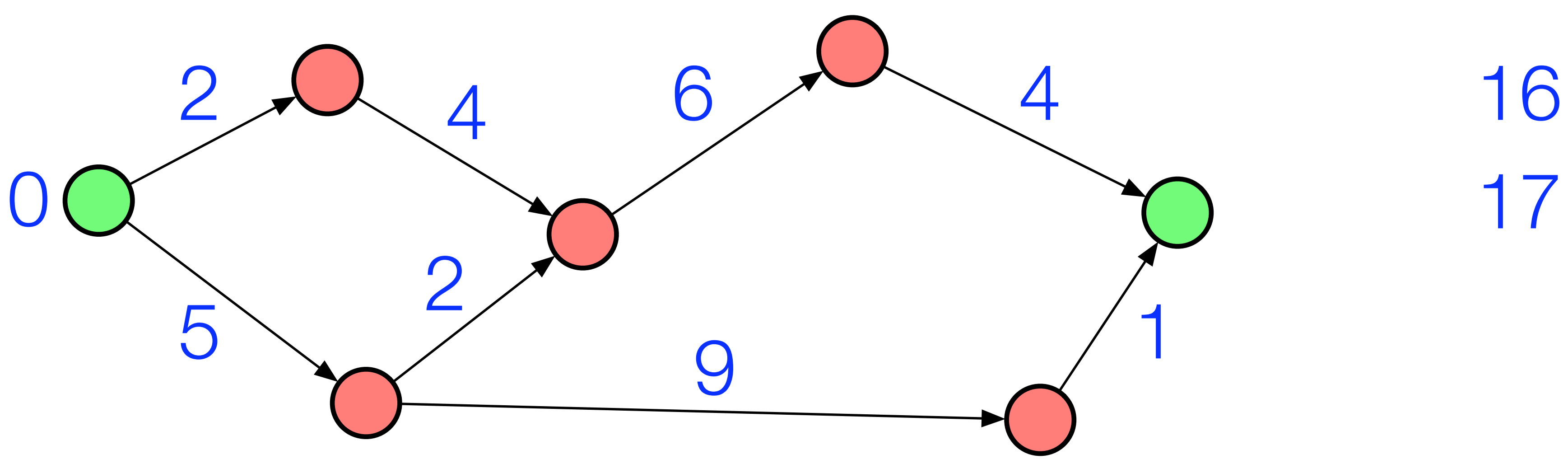
Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, ∧, ⊕, ⊤, 0): two monoids (S, ∧, ⊤), (S, ⊕, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: **($\mathbf{N}^{\infty}$, min, +, ∞, 0)**, (℘(A), ∩, ∪, A, ∅), ($\mathbf{R}^{+}$, max, min, 0, ∞)



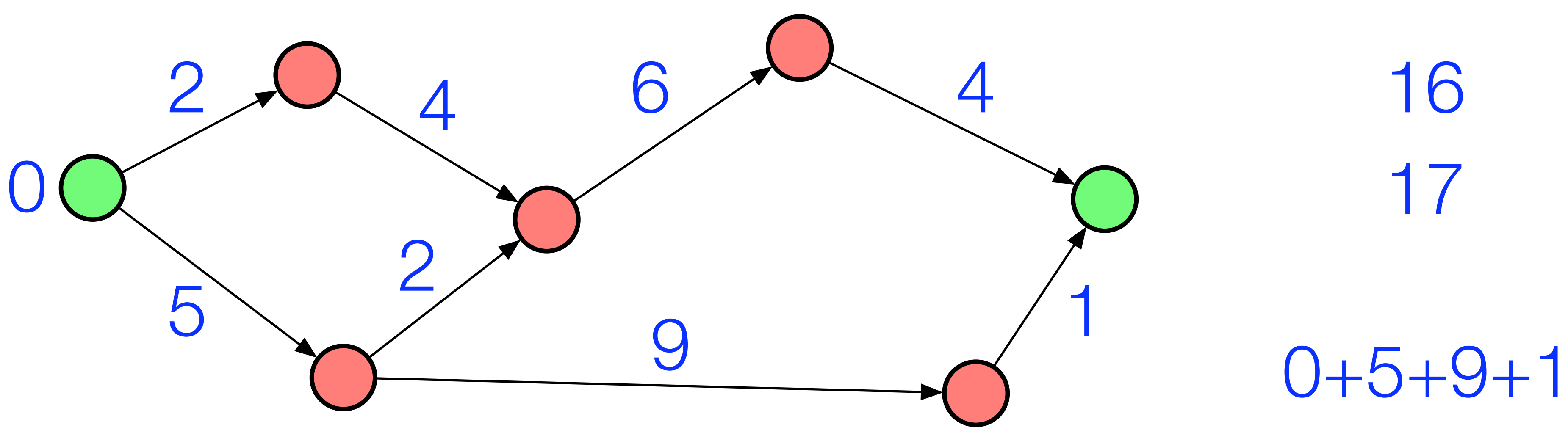
Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, ∧, ⊕, ⊤, 0): two monoids (S, ∧, ⊤), (S, ⊕, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: ($\mathbf{N}^\infty$, min, +, ∞, 0), (℘(A), ∩, ∪, A, ∅), ($\mathbf{R}^+$, max, min, 0, ∞)



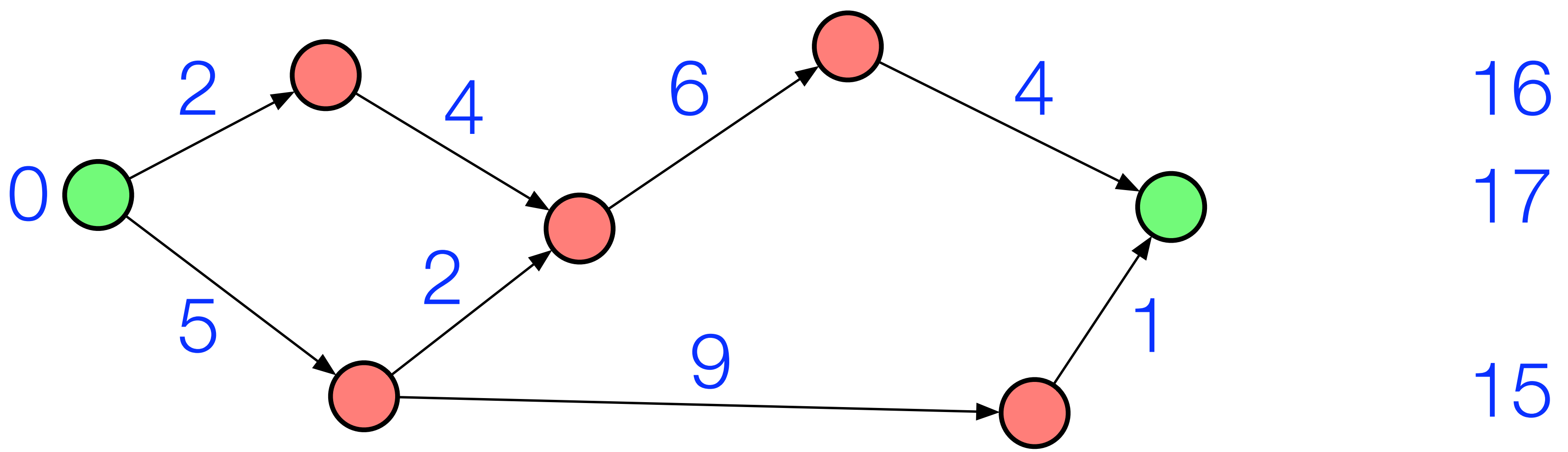
Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring $(S, \wedge, \oplus, \top, 0)$: two monoids $(S, \wedge, \top)$, $(S, \oplus, 0)$ on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: **$(\mathbf{N}^{\infty}, \min, +, \infty, 0)$**, $(\wp(A), \cap, \cup, A, \emptyset)$, $(\mathbf{R}^{+}, \max, \min, 0, \infty)$



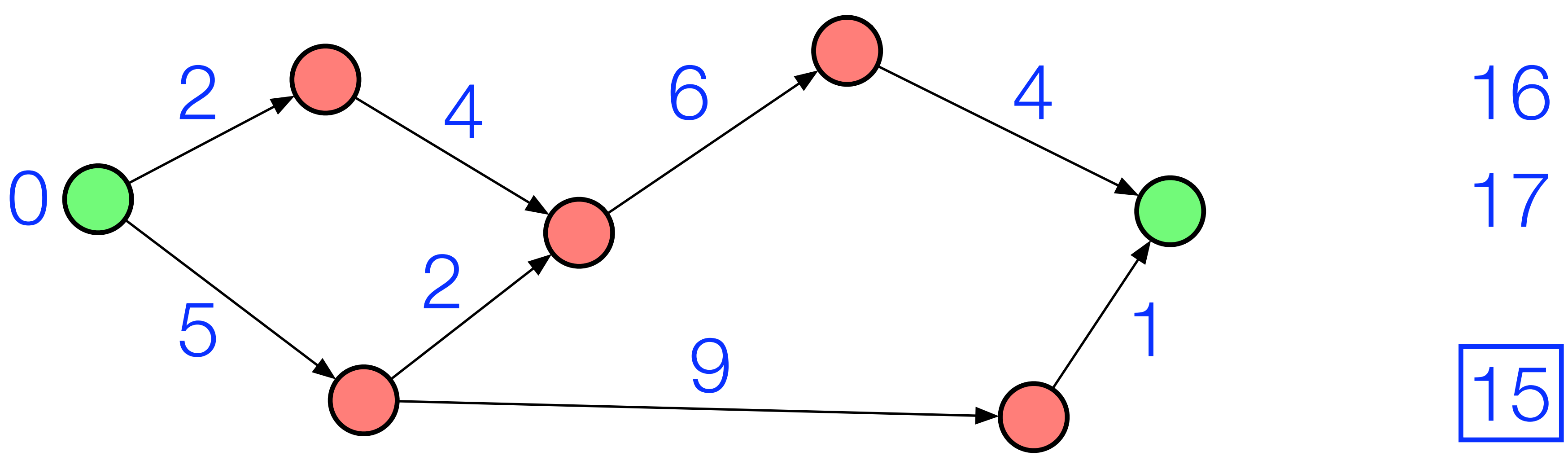
Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, ∧, ⊕, ⊤, 0): two monoids (S, ∧, ⊤), (S, ⊕, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: **($\mathbf{N}^{\infty}$, min, +, ∞, 0)**, (℘(A), ∩, ∪, A, ∅), ($\mathbf{R}^{+}$, max, min, 0, ∞)



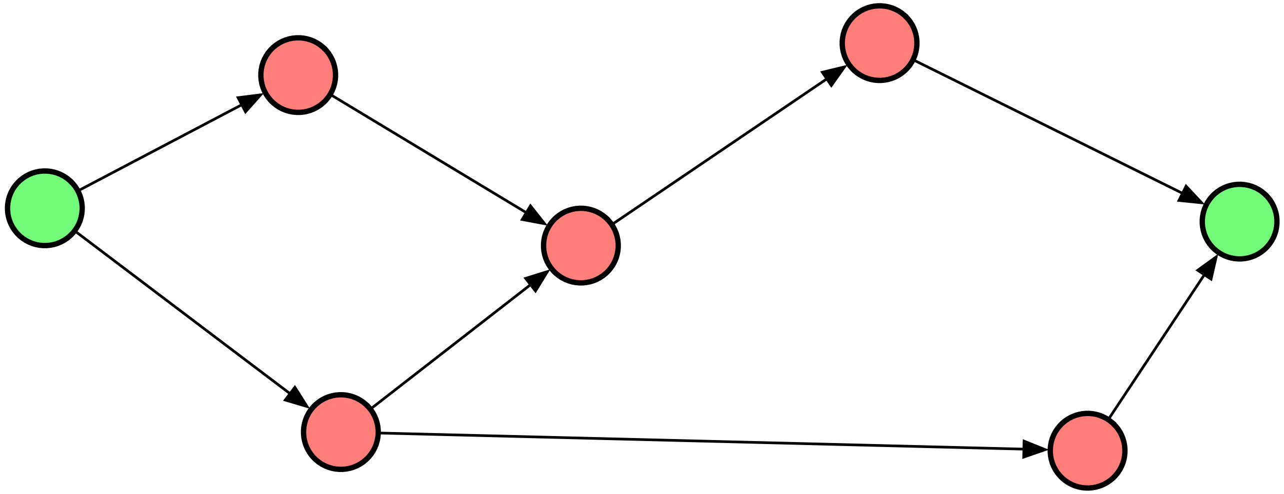
Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, ∧, ⊕, ⊤, 0): two monoids (S, ∧, ⊤), (S, ⊕, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: **($\mathbf{N}^{\infty}$, min, +, ∞, 0)**, ($\wp$(A), ∩, ∪, A, ∅), ($\mathbf{R}^{+}$, max, min, 0, ∞)



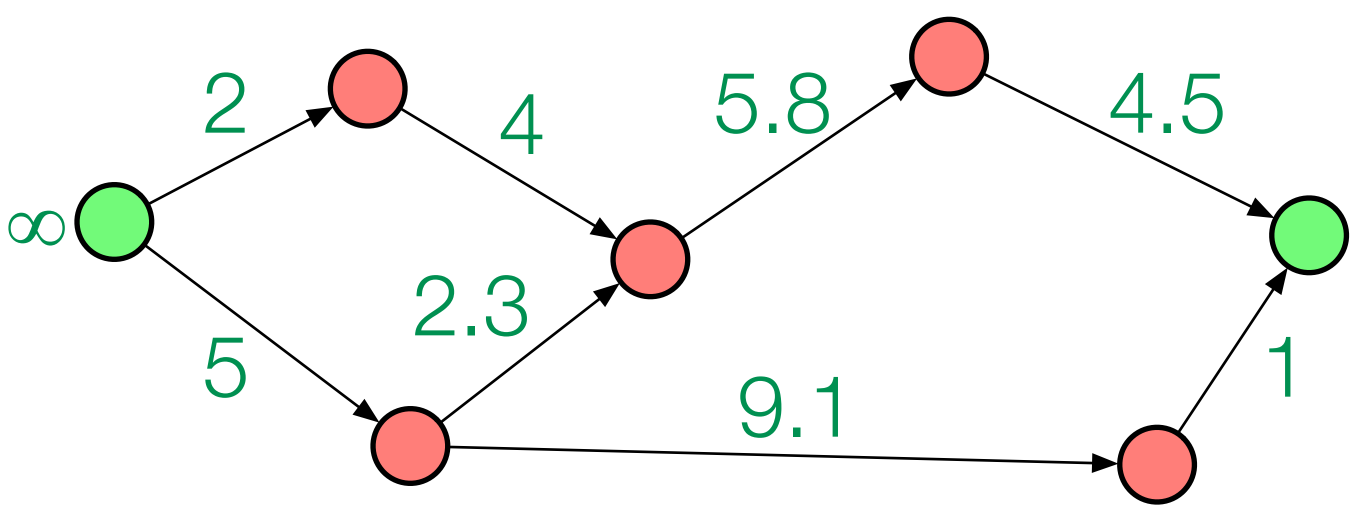
Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, $\wedge$, $\oplus$, $\top$, 0): two monoids (S, $\wedge$, $\top$), (S, $\oplus$, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: ($\mathbf{N}^\infty$, min, +, ∞, 0), ($\wp$(A), $\cap$, $\cup$, A, $\varnothing$), **($\mathbf{R}^+$, max, min, 0, ∞)**



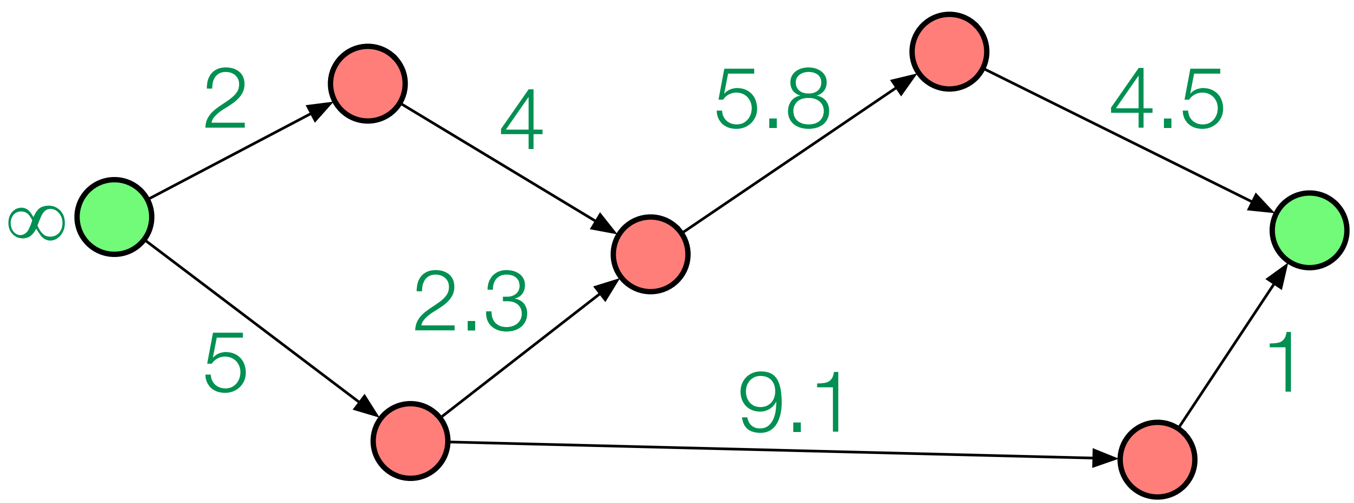
Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, $\wedge$, $\oplus$, $\top$, 0): two monoids (S, $\wedge$, $\top$), (S, $\oplus$, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: ($\mathbf{N}^\infty$, min, +, ∞, 0), ($\wp$(A), $\cap$, $\cup$, A, $\varnothing$), ($\mathbf{R}^+$, max, min, 0, ∞)



Model I: Path algebra

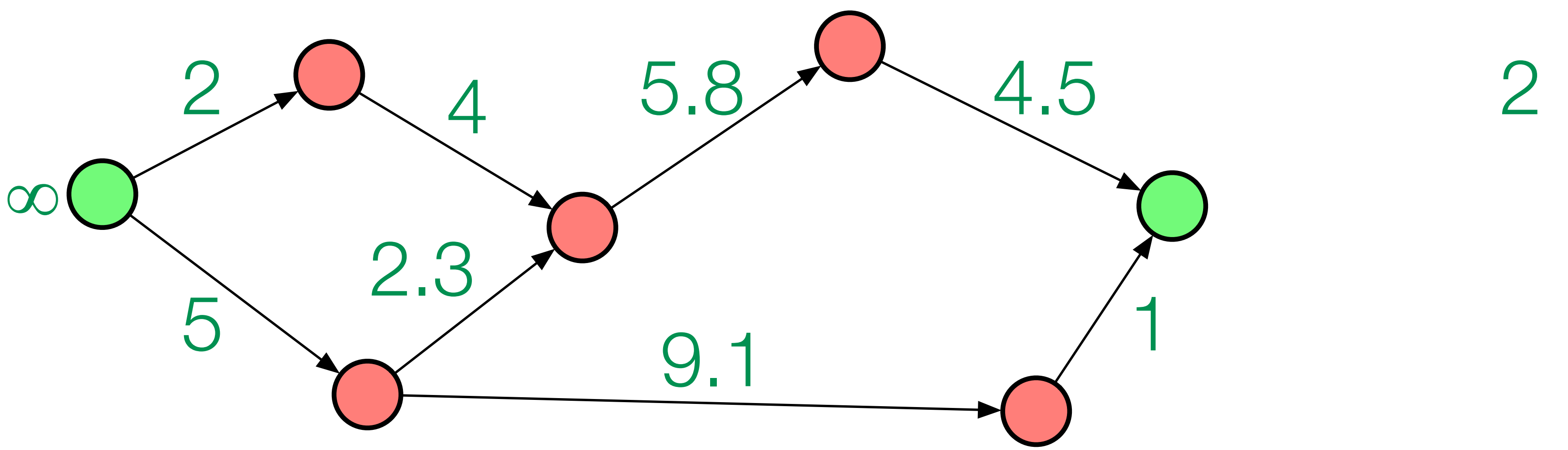
- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, ∧, ⊕, ⊤, 0): two monoids (S, ∧, ⊤), (S, ⊕, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: ($\mathbf{N}^{\infty}$, min, +, ∞, 0), (℘(A), ∩, ∪, A, ∅), ($\mathbf{R}^{+}$, max, min, 0, ∞)



{2, 4, 5.8, 4.5}

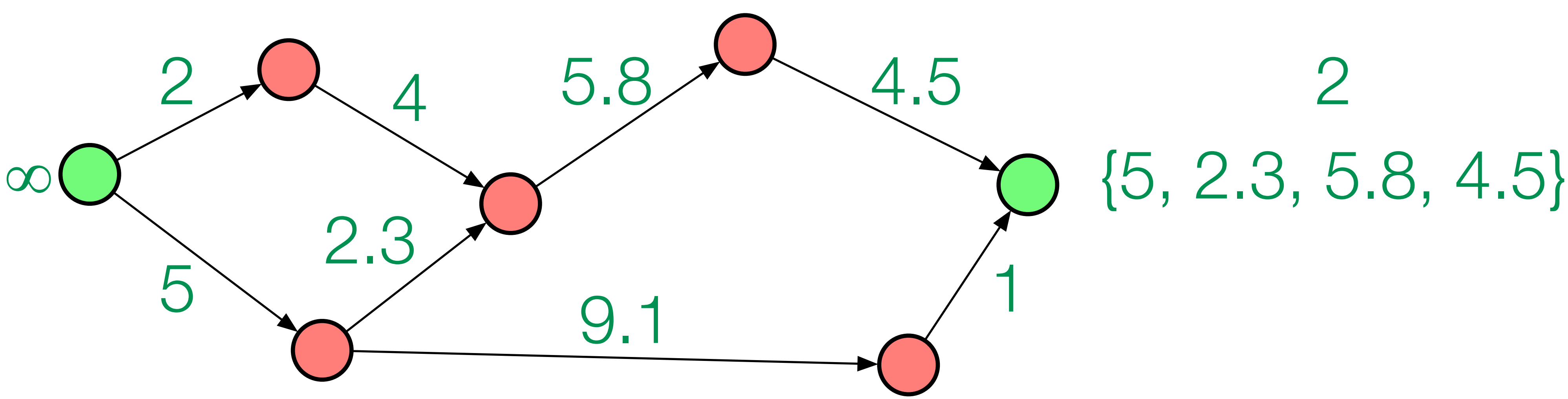
Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, ∧, ⊕, ⊤, 0): two monoids (S, ∧, ⊤), (S, ⊕, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: ($\mathbf{N}^{\infty}$, min, +, ∞, 0), (℘(A), ∩, ∪, A, ∅), ($\mathbf{R}^{+}$, max, min, 0, ∞)



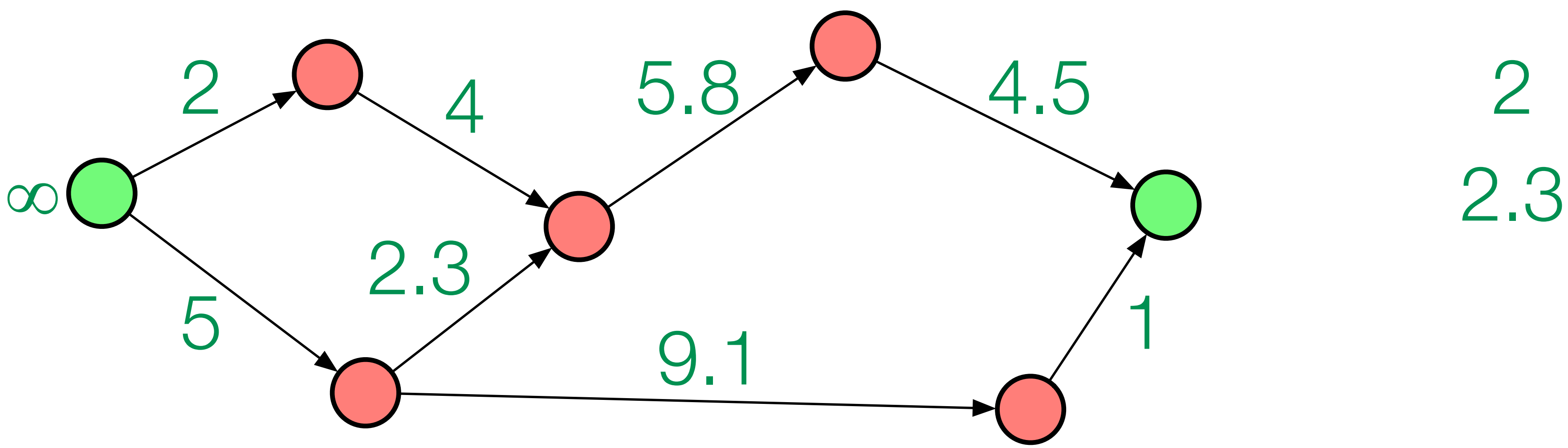
Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, ∧, ⊕, ⊤, 0): two monoids (S, ∧, ⊤), (S, ⊕, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: ($\mathbf{N}^{\infty}$, min, +, ∞, 0), (℘(A), ∩, ∪, A, ∅), **($\mathbf{R}^{+}$, max, min, 0, ∞)**



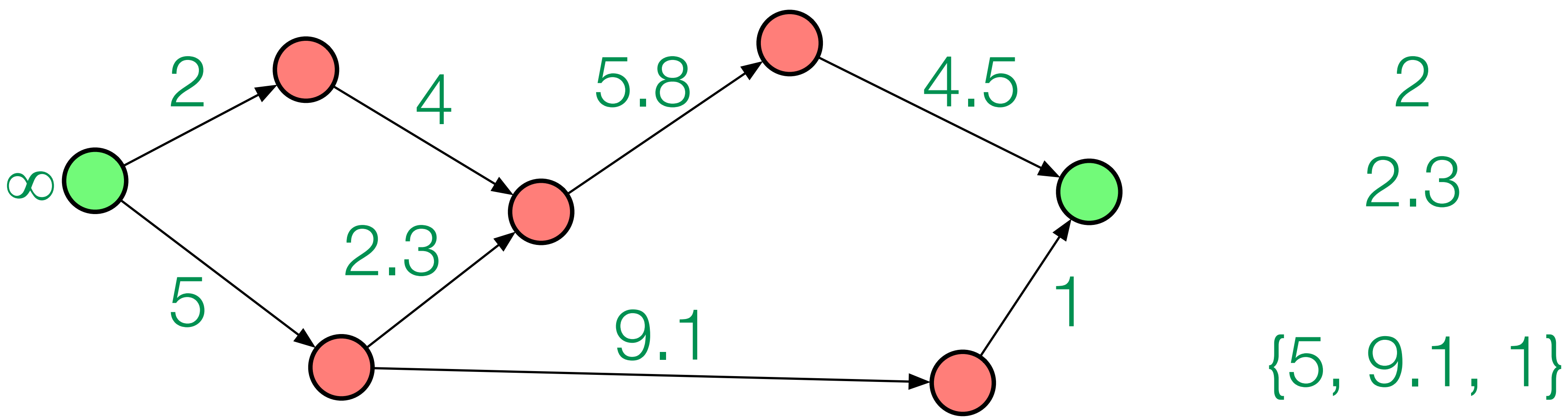
Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, ∧, ⊕, ⊤, 0): two monoids (S, ∧, ⊤), (S, ⊕, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: ($\mathbf{N}^\infty$, min, +, ∞, 0), (℘(A), ∩, ∪, A, ∅), **($\mathbf{R}^+$, max, min, 0, ∞)**



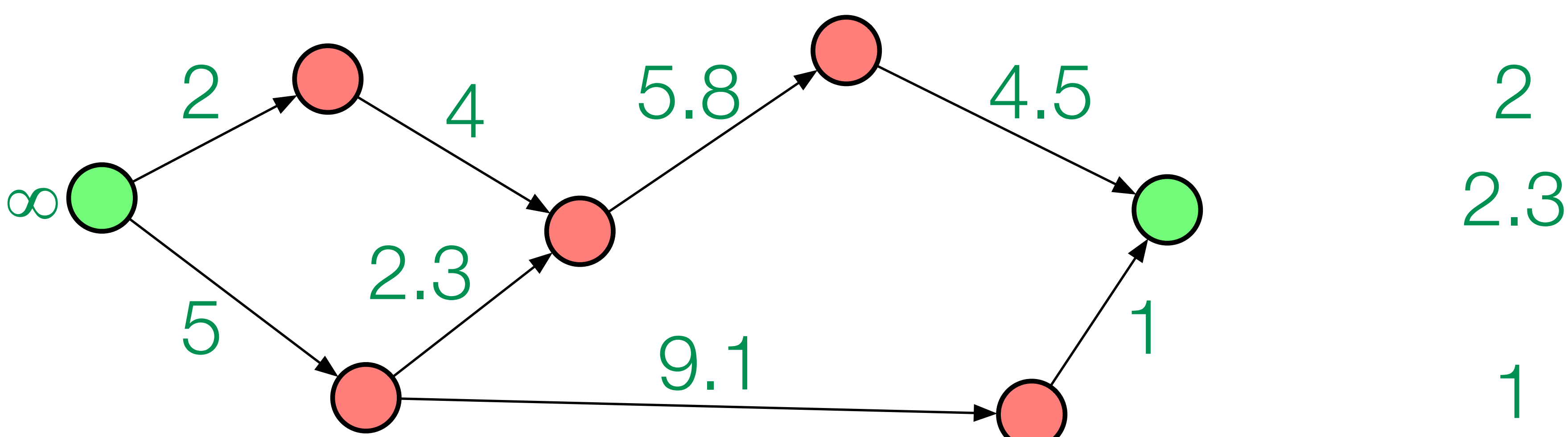
Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, ∧, ⊕, ⊤, 0): two monoids (S, ∧, ⊤), (S, ⊕, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: ($\mathbf{N}^\infty$, min, +, ∞, 0), (℘(A), ∩, ∪, A, ∅), ($\mathbf{R}^+$, max, min, 0, ∞)



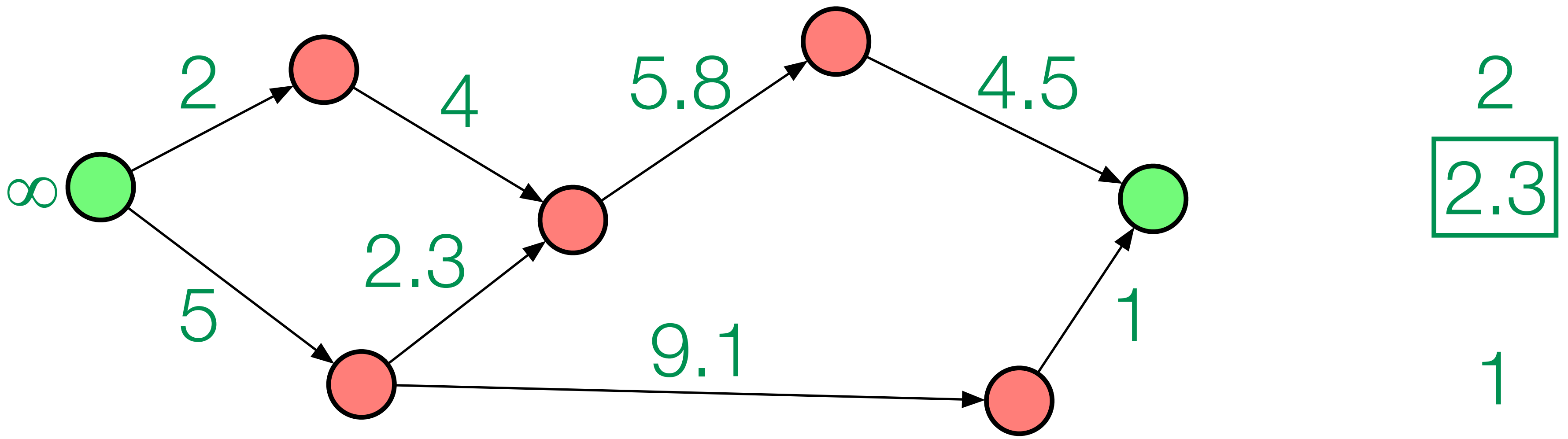
Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, $\wedge$, $\oplus$, $\top$, 0): two monoids (S, $\wedge$, $\top$), (S, $\oplus$, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: ($\mathbf{N}^{\infty}$, min, +, ∞, 0), ($\wp$(A), $\cap$, $\cup$, A, $\emptyset$), ($\mathbf{R}^{+}$, max, min, 0, ∞)



Model I: Path algebra

- Monoid (M, +, 0): closed, associative, identity
- Semiring (S, ∧, ⊕, ⊤, 0): two monoids (S, ∧, ⊤), (S, ⊕, 0) on the same set
- Path algebra: has commutativity, distributivity, idempotence ($a + a = a$)
- Examples: ($\mathbf{N}^{\infty}$, min, +, ∞, 0), (℘(A), ∩, ∪, A, ∅), **($\mathbf{R}^{+}$, max, min, 0, ∞)**



Combining operations

- $D = (\mathbf{N}^\infty, \min, +, \infty, 0)$, $B = (\mathbf{R}^+, \max, \min, 0, \infty)$

- $D \times B = (\mathbf{N}^\infty \times \mathbf{R}^+, \wedge, \oplus, (\infty, 0), (0, \infty))$, where

  $(d, b) \wedge (e, c) = (\min(d, e), \max(b, c))$

  $(d, b) \oplus (e, c) = (d + e, \min(b, c))$

- $D \times_{\text{lex}} B = (\mathbf{N}^\infty \times \mathbf{R}^+, \wedge_{\text{lex}}, \oplus, (\infty, 0), (0, \infty))$, where

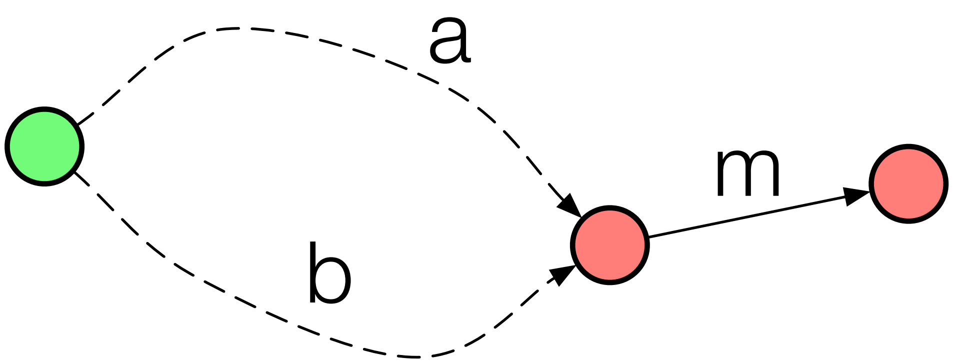
  $(d, b) \wedge_{\text{lex}} (e, c) = (d, b)$ if $d < e$ or $[d = e$ and $b > c]$; (b, c) otherwise

  $\oplus$ is as before

  We obtain $<$ from min by defining $a \leq b$ iff $a = \min(a, b)$

Dijkstra's algorithm and the prefix property

- To get the right answer out of Dijkstra, we need each prefix of a shortest path to also be a shortest path
- In semiring language: $a = a \wedge b \Rightarrow (m + a) = (m + a) \wedge (m + b)$, for all a, b, m



- This is the case when we have distributivity: $(m + a) \wedge (m + b) = m + (a \wedge b)$

Property preservation

- If A and B are distributive, then so is $A \times B$; but $A \times_{lex} B$ may not be
- So if we have a whole load of distributive semirings A, B, C, and D, then we know we can run Dijkstra correctly on $A \times B \times C \times D$
- Similar rules exist for other properties and other operations – so we can deduce facts like “we can’t do Dijkstra, but we can do Bellman-Ford”

Expressiveness issues

- Consider the ASPATH attribute of BGP (a list of numbers; shorter lists are preferred, but we don't care about the contents). How can we encode this?

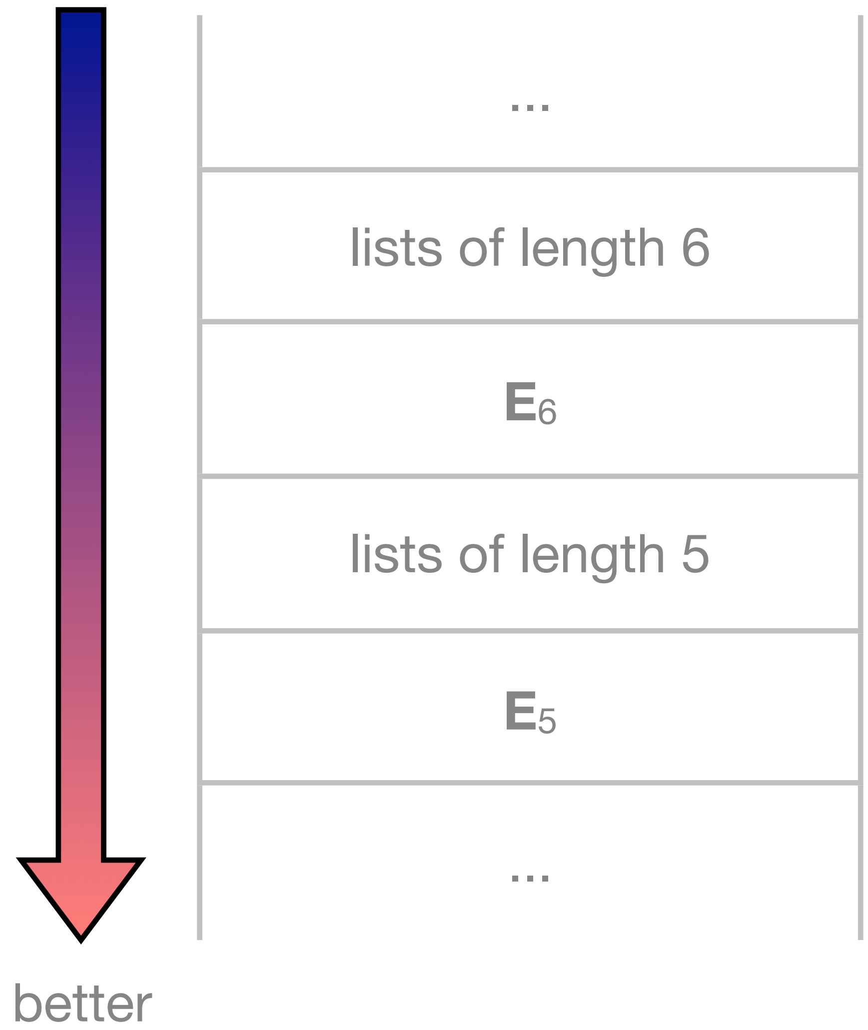
- Surely S must be the set of lists, and $\oplus$ is the append operation...

  ...and we can say by convention that we will only put one-element lists on the arcs...

  ...but what should $\wedge$ be for two lists of equal length? It has to be something different from either operand, so maybe we can have a special "equal length" symbol, "**E**". But then what is **E** $\wedge$ *a* ? We actually need an $\mathbf{E}_k$ for each *k*. And we can extend $\oplus$ to work on these, too.

Expressiveness issues

- We now have a consistent algebra
- But we've lost touch with reality
- $\mathbf{E}_1 = \mathbf{E}_1 \wedge [1]$, so $\mathbf{E}_1 < [1]$
- $\mathbf{E}_k$ tells us nothing about the actual path. And we prefer these to concrete lists!



Multiple equivalent paths

- S now consists of sets of lists (where all lists in a set are of the same length)
- $A \oplus B = \{ \text{append}(a, b) \mid a \in A, b \in B \}$; identity $\{ [\,] \}$
- $A \wedge B = \{ x \in A \cup B \mid \forall y \in A \cup B\text{: } |x| \leq |y| \}$; identity $\emptyset$
- Only ever put $\{ [n] \}$ on the arcs
- Is this really ‘natural’? Can it be derived automatically?

Model II: Routing algebras

- (S, $\leq$) where $\leq$ is a preference relation (reflexive, transitive, total)
- Label set L; application function $\oplus : L \times S \to S$
- Very general (even more so if we extend $\leq$ to a preorder)

  We can encode ASPATH very easily, along with our other examples

- Price to pay: not so algebraically nice

Model III: Functional path algebras

- A hybrid of $(S, \oplus, \wedge, 0, 1)$ and $(S, \leq, L, \oplus)$

- (M, F) where M is a commutative monoid and F a set of functions $M \rightarrow M$

- Elements of F go on the arcs

- If everything in F is a homomorphism, then these look a lot like path algebras. But we do not require this.

Embed routing algebra in functional path algebra

- $(S, \leq, L, \oplus) \rightarrow ((\wp_{\leq}(S), \cup_{\leq}), F_L)$, where

  $\wp_{\leq}(S) = \{ A \subseteq S \mid \min_{\leq}(S) = S \}$

  $A \cup_{\leq} B = \min_{\leq}(A \cup B)$

  $F_L = \{ \lambda S . \min_{\leq}\{ l \oplus s \mid s \in S \} \mid l \text{ in } L \}$

- Multipath routing with a partial order, but secretly based on a far more general order

- Arc labels are better than before – they seem like single elements

The ASPATH example

- Routing algebra is $(S, \leq, L, \oplus)$ where

  S = lists of AS numbers (and no list has the same number twice), plus **E**

  $\leq$ orders lists by length; **E** is topmost

  L = AS numbers

  $n \oplus ns = n{:}ns$ (unless n is in ns or $n{:}ns$ is too long, when we return **E**)

The ASPATH example

- As a functional path algebra: $((\wp_{\leq}(S), \cup_{\leq}), F_L)$

  Each element of $\wp_{\leq}(S)$ is a set of lists; all lists have the same length (and there is also an element {**E**})

  $p \cup_{\leq} q$ is the set of shortest lists in $p \cup q$

  so { [2, 1], [5, 1] } $\cup_{\leq}$ { [6, 1] } = { [2, 1], [5, 1], [6, 1] };

  and { [2, 1] } $\cup_{\leq}$ { [4] } = { [4] }

  Each element of F is a function f_k, adding k to each list in the given set

  f_6 { [1, 4], [3, 6] } = min { [6, 1, 4], **E** } = { [6, 1, 4] }

Canonical constructions

- Direct and lexical products
- Parallel sum A || B: $a + b = \mathbf{E}$
- Layered sum A ◁ B: $a + b = a$
- Local preference: $F = \{ \lambda x.a \mid a \}$
- Origin preference: $F = \{ \mathrm{id} \}$
- many more

The metalanguage

- Borrow syntax from maths (but this will definitely be *syntax*)

- Expressions E ::= atom | unary(E) | (E binary E)

  unary ::= LP, OP, FLIP, FLATTEN, ...

  binary ::= $\times$, $\times_{lex}$, $||$, $\lhd$, ...

- For each kind of structure, an evaluation function, appropriately typed

  written as $(S, \leq)$ $[\![\ldots]\!] = \ldots$

- Notational convenience: $(A_1, \ldots, A_k)$ $[\![\ldots]\!] = (A_1 [\![\ldots]\!], \ldots, A_k [\![\ldots]\!])$

Sample rules

- $\leq [\![A \,||\, B]\!] = \leq [\![A]\!] \cup \leq [\![B]\!]$

- $(M, +) [\![A \lhd B]\!] = (M [\![A]\!] \cup M [\![B]\!] , \oplus)$

  where $a \oplus a' = a +_{[\![A]\!]} a'$; $b \oplus b' = b +_{[\![B]\!]} b'$; $a \oplus b = a$

- $F [\![LP(A)]\!] = \{ \lambda x.a \mid a \in M [\![A]\!] \}$

- $a \oplus_{[\![LP(A)]\!]} a' = a$

Integration of properties

- We can make sense of properties in the same framework
- Want DISTRIB ⟦ A ⟧ iff A is distributive
- Prove DISTRIB ⟦ A x B ⟧ = (DISTRIB ⟦ A ⟧ and DISTRIB ⟦ B ⟧)
- and so forth

Scoped product derivation

- $A \ \Theta \ B = (OP(A) \times_{lex} B) +_F (A \times_{lex} LP(B))$
- $M [\![A \ \Theta \ B]\!]$

  $= M [\![OP(A) \times_{lex} B]\!] \cup M [\![A \times_{lex} LP(B)]\!]$

  $= (M [\![OP(A)]\!] \times M [\![B]\!]) \cup (M [\![A]\!] \times M [\![LP(B)]\!])$

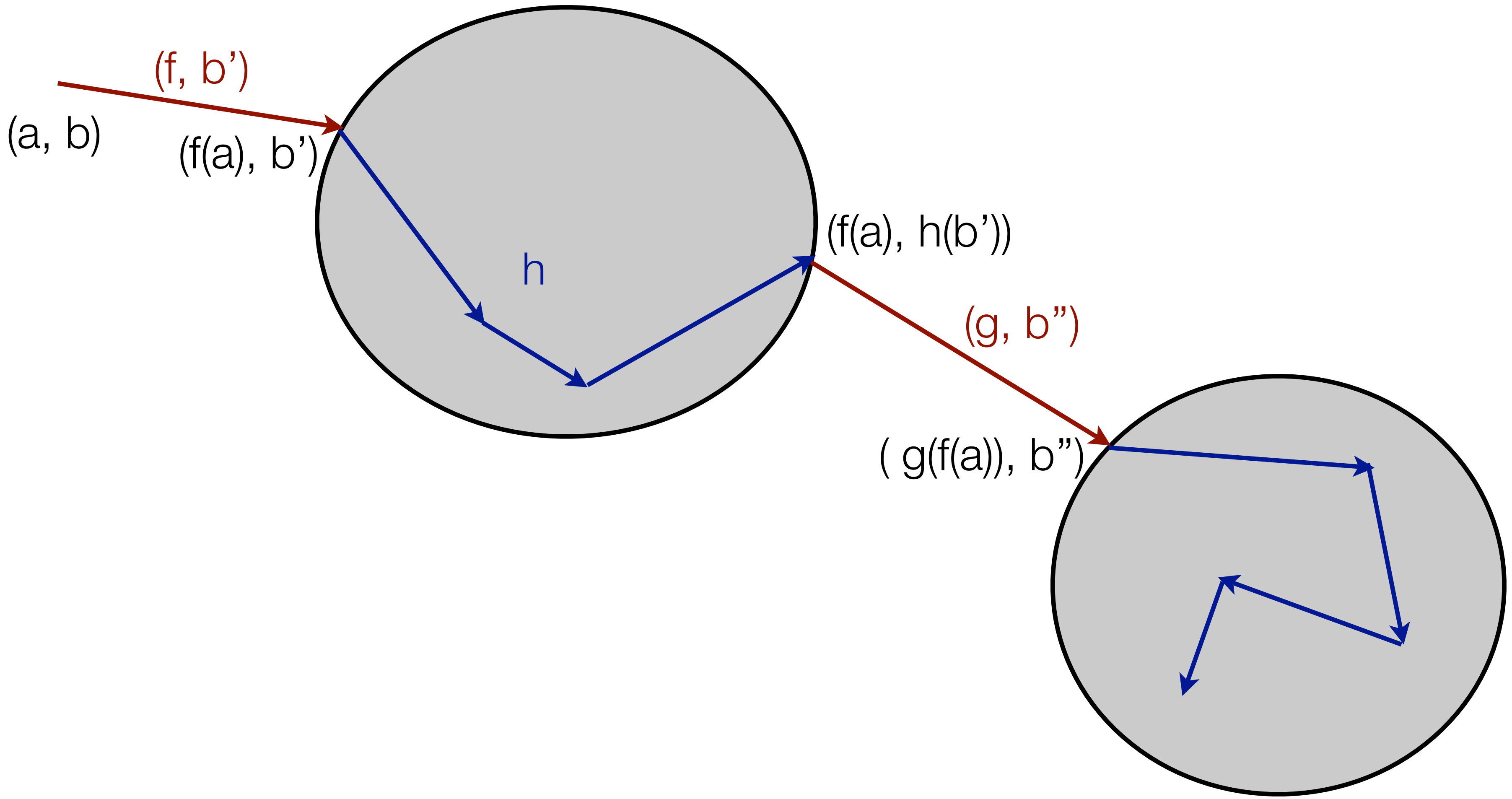
  $= (M [\![A]\!] \times M [\![B]\!])$
- $F [\![A \ \Theta \ B]\!]$

  $= F [\![OP(A) \times_{lex} B]\!] \cup F [\![A \times_{lex} LP(B)]\!]$

  $= (F [\![OP(A)]\!] \times F [\![B]\!]) \cup (F [\![A]\!] \times F [\![LP(B)]\!])$

  $= \{ (id, f) \mid f \in F [\![B]\!] \} \cup \{ (g, \lambda x.b) \mid g \in F [\![A]\!] , b \in M [\![B]\!] \}$

Scoped product



Future directions

- Generate programs / configuration files by the same means
- Handle more complex policy interactions
- Maths: find good operators from the category theory zoo
- Deeper understanding of algorithms
- Modality, migration, other protocol aspects