# Pathfinding through congruences

Alexander J. T. Gurney

University of Pennsylvania

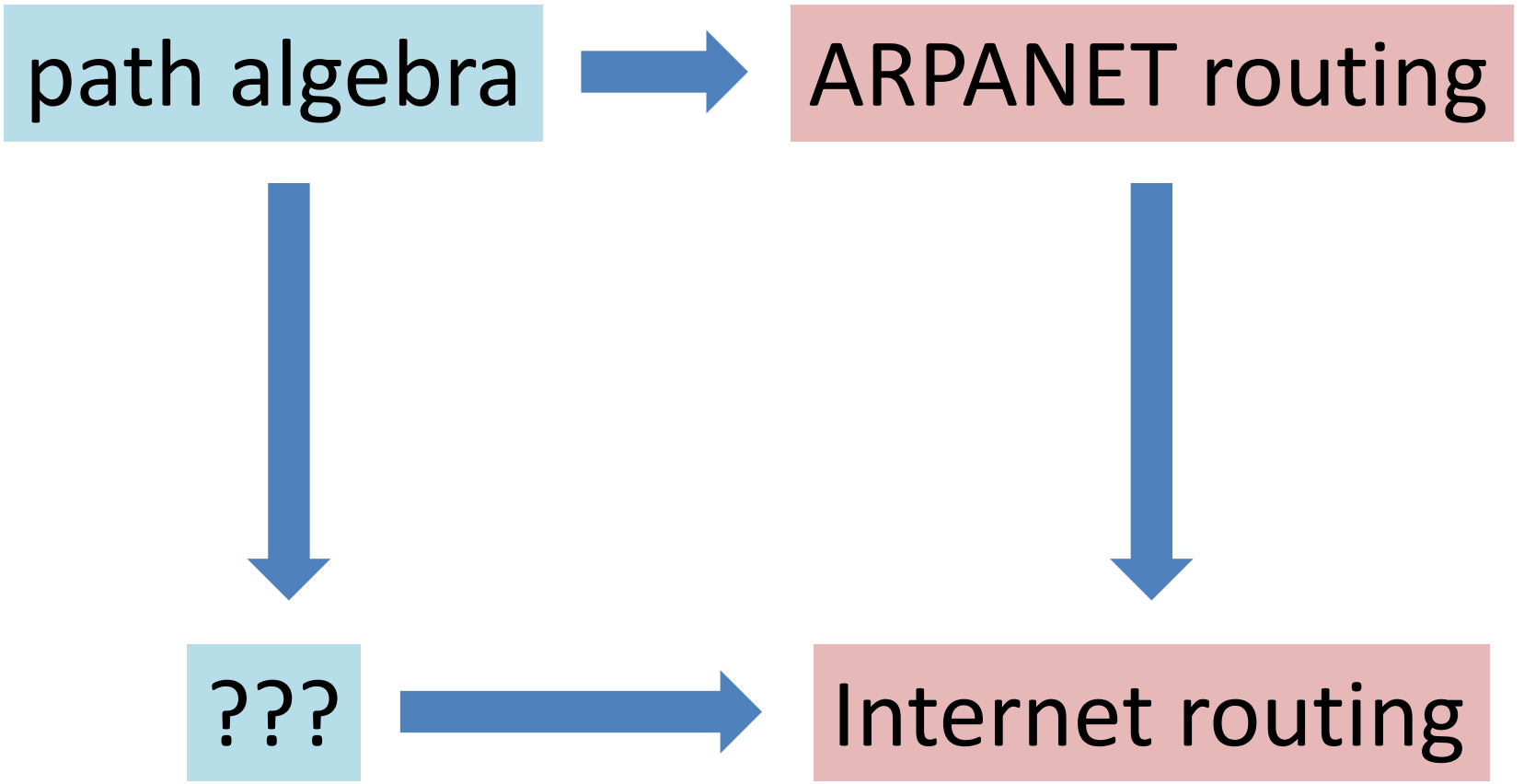
Timothy G. Griffin

University of Cambridge

12th RAMICS – 3 June 2011 – Rotterdam

# Acknowledgements


- Engineering and Physical Sciences Research Council (UK) EP/F002718/1
- National Science Foundation (USA) CNS-0845552


path algebra
ARPANET routing
???
Internet routing

# Congruences

1. Can be defined for our new structures
2. Can be used to model certain aspects of Internet routing
   - Multipath
   - Filtering
   - Error handling
   - …

# Engineering a theory

- Path algebra has been hugely successful and has many important results
- Try to take advantage of this:
  - reuse results where possible
  - otherwise, recapitulate the ideas in spirit
- Avoid ad-hoc definitions, special cases, baroque structures

# Comparison

**Path algebra**

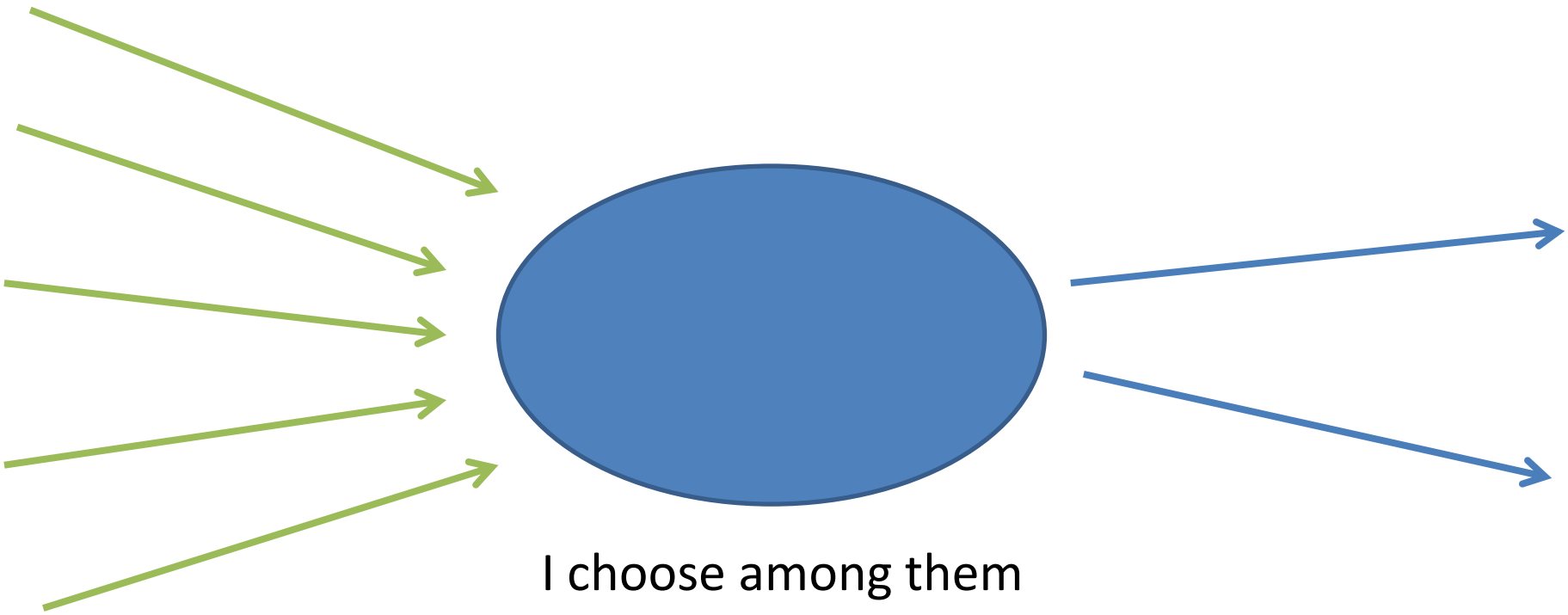
- Semirings
- Products, quotients, ...
- Labelled graphs
- Matrices
- Fixed points via Knaster, Tarski, Kleene theorems
- Generic algorithms
- Distributed algorithms

**Internet routing algebra**

- Order transformations
- Products, quotients, ...
- Labelled graphs
- Matrices (sort of)
- Fixed points via Banach theorem
- Generic algorithms
- Distributed algorithms

# Internet routing

- Fundamentally *not* a “shortest path” problem
- Instead, must find *stable paths*
  - compare: neural network stability; circuit stability; finding Nash equilibria; stable model semantics
- Because network participants have conflicting definitions of “good” path
- Not well modelled by semiring axioms



I choose among them

Paths come in

I communicate my choice

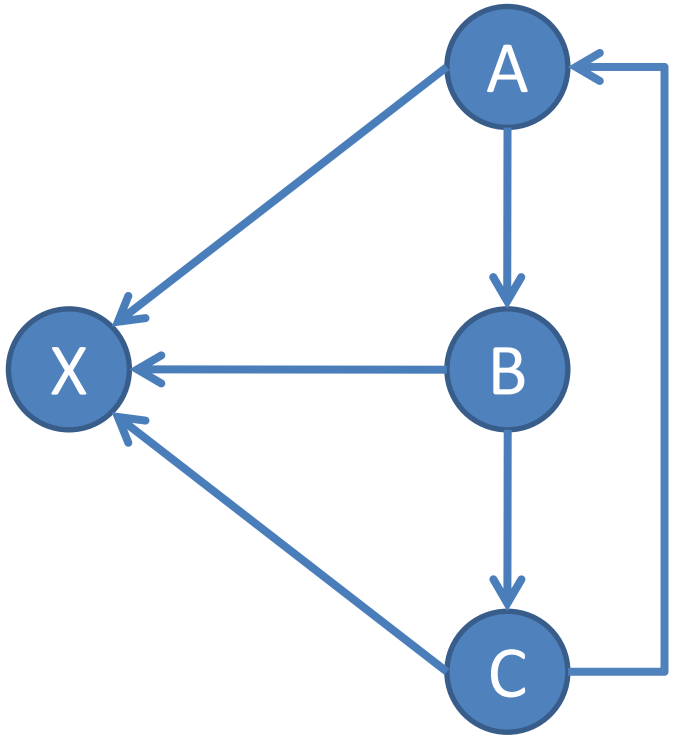
# An abstract model

- Have a set of paths (or path weights) S
- Order them according to a preorder ≤
- Extend them according to functions in a set F
  - these label the arcs of the graph

- If $s < f(s)$ for all s in S, and all f in F, then the routing problem has a **unique solution** (fixed point of the simultaneous update operator)

# Inflationary condition

- This “s < f(s)” means I can put *all paths* in some order, where
  - a path appears before any extension of that path
  - a better path appears before a worse path (for the same source and destination)




ABX < AX

ABCX forbidden

BCX < BX

BCAX forbidden

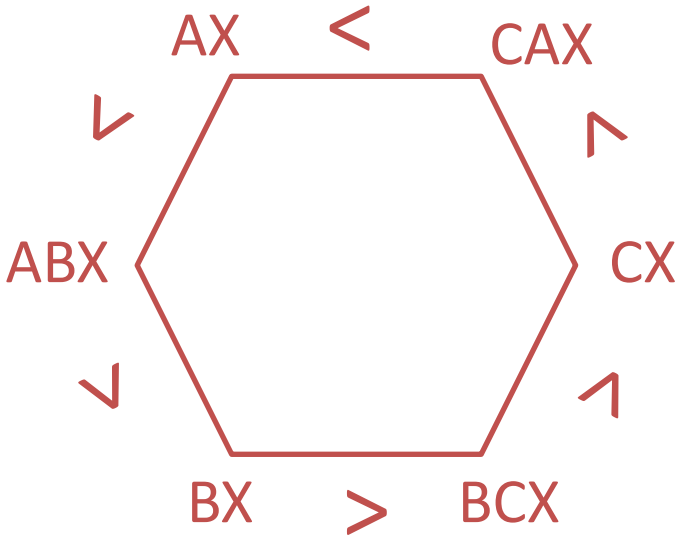
CAX < CX

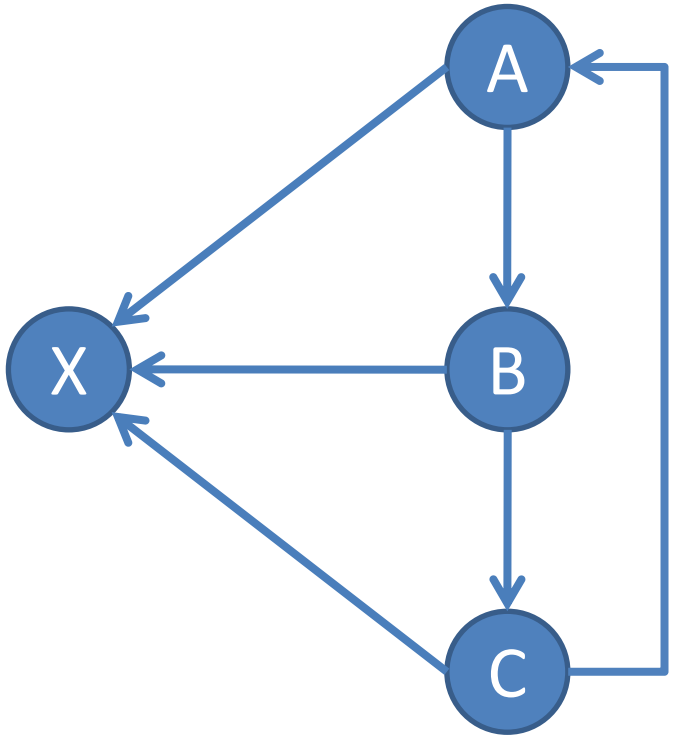
CABX forbidden

AX < CAX

BX < ABX

CX < BCX






ABX < AX  ABCX forbidden

**BX < BCX**  BCAX forbidden

CAX < CX  CABX forbidden

AX < CAX

BX < ABX

CX < BCX

| | |
|---|---|
| X | 1 |
| BX | 1 = X |
| ABX | 1 = BX |
| AX | 0 = X ∧ ¬ABX |
| CAX | 0 = AX |
| CX | 1 = X ∧ ¬CAX |
| BCX | 0 = CX ∧ ¬BX |

# Apply this to Internet routing

What attributes do routes have, and how do we choose good routes?



RFC 4271: A Border Gateway Protocol 4 (BGP-4)

Some routes are excluded from preference consideration:

> If the AS_PATH attribute of a BGP route contains an AS loop, the BGP route should be excluded from the Phase 2 decision function. AS loop detection is done by scanning the full AS path (as specified in the AS_PATH attribute), and checking that the autonomous system number of the local system does not appear in the AS path.

User-defined filters are also invoked.

# Attributes to consider

1. Local preference

```
The tie-breaking algorithm begins by considering all equally
preferable routes to the same destination, and then selects routes
to be removed from consideration. The algorithm terminates as soon
as only one route remains in consideration. The criteria MUST be
applied in the order specified.
```

2. Number of other networks visited
3. Origin code
4. “Multi-exit discriminator”
5. Learned from outside or inside my network?
6. Shortest-path distance to network edge
7. Identifier for the router at the network edge
8. Identifier for the neighbor’s router

c) Remove from consideration routes with less-preferred MULTI_EXIT_DISC attributes. MULTI_EXIT_DISC is only comparable between routes learned from the same neighboring AS (the neighboring AS is determined from the AS_PATH attribute). Routes that do not have the MULTI_EXIT_DISC attribute are considered to have the lowest possible MULTI_EXIT_DISC value.

f) Remove from consideration all routes other than the route that was advertised by the BGP speaker with the lowest BGP Identifier value.

# Things we can do

- Express *arbitrary* preferences among routes we receive
- Filter out routes on import or export
- Alter route attributes
  - OK, only some are *meant* to be altered, and only in certain ways, but...

# Avoiding nondeterminism

- Routers can be configured to prefer *older* routes at a certain stage of the decision process
- Nowadays operators are advised not to use this capability
- We assume that for any two routes, we can reliably choose between them

# Avoiding context dependence

- The decision to prefer *a* over *b* should be based only on *a* and *b*, not on the presence of any other route *c*.
- Easier to reason about
- Easier to implement
- Easier to ensure correct
- Not always true in practice, *but* we wish it was

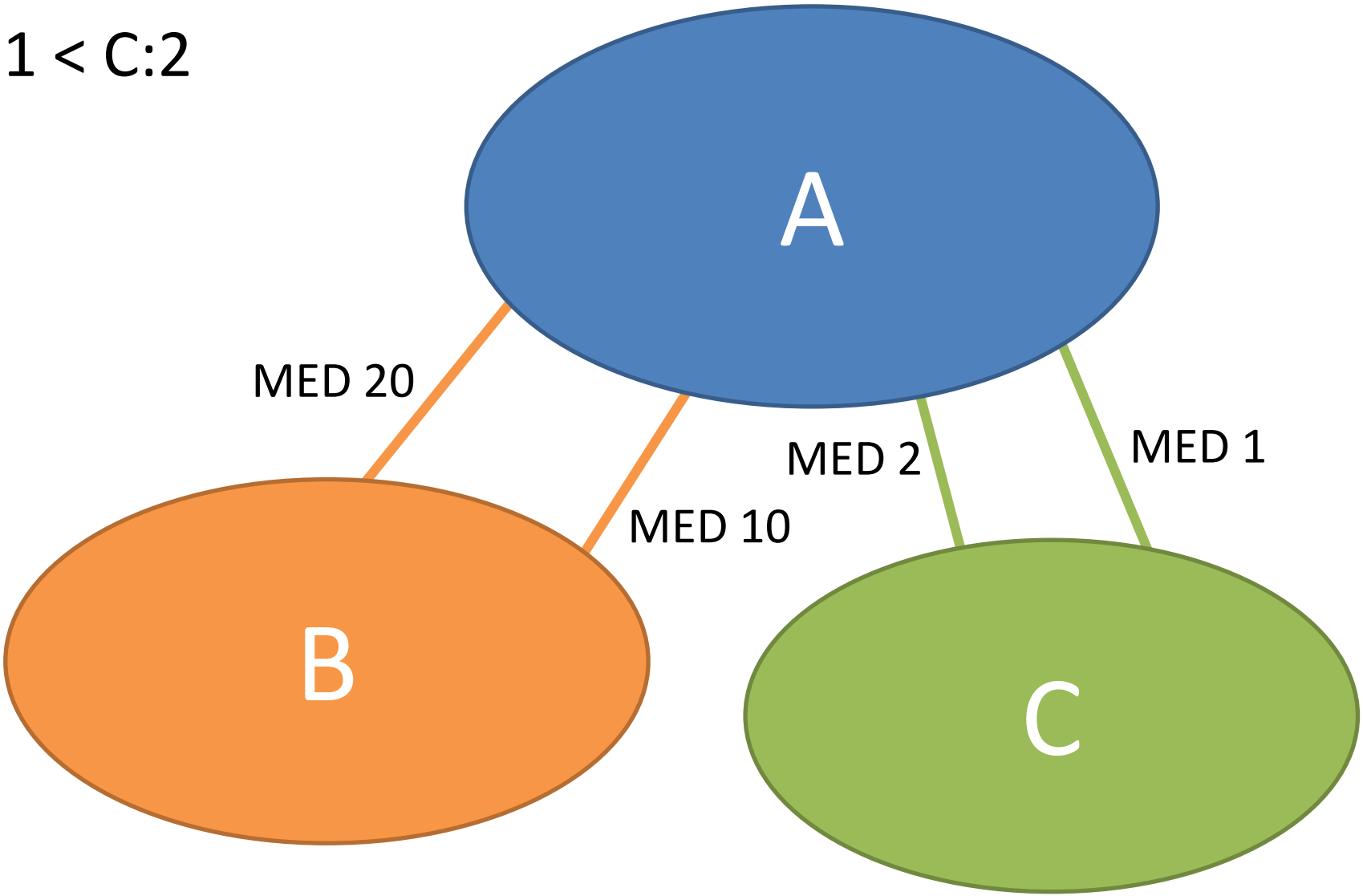
# Incomparability

c) Remove from consideration routes with less-preferred MULTI_EXIT_DISC attributes. MULTI_EXIT_DISC is only comparable between routes learned from the same neighboring AS (the neighboring AS is determined from the AS_PATH attribute). Routes that do not have the MULTI_EXIT_DISC attribute are considered to have the lowest possible MULTI_EXIT_DISC value.

- Eventually all routes should be in a linear order (at least in theory)
- But intermediate attributes require us to admit incomparable routes

B:10 < B:20

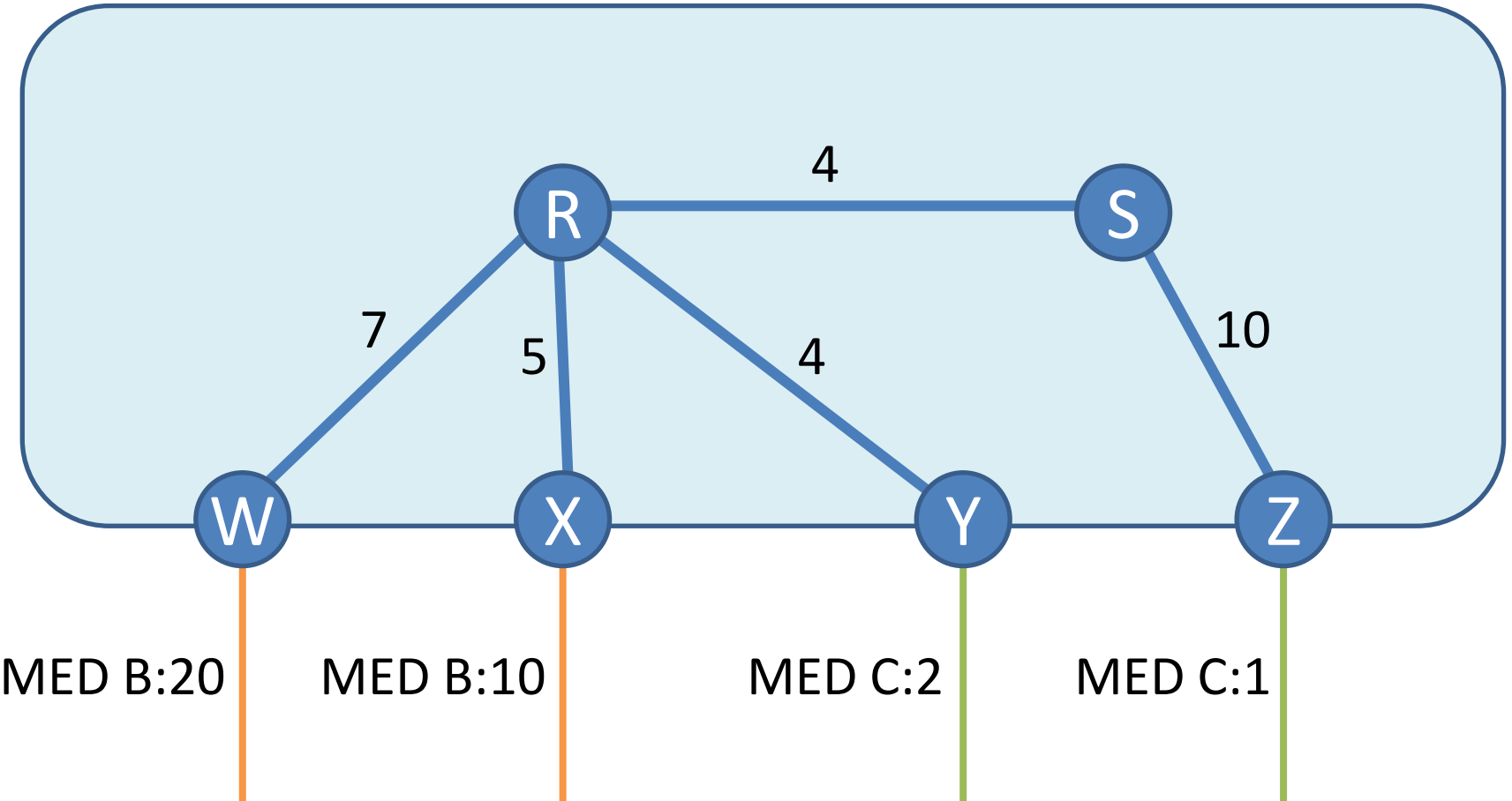
C:1 < C:2



Look at MED first, then (internal) shortest paths if necessary.

From R:

| | |
|---|---|
| (B:10, 5) > (C:2, 4) | MED incomparable, 5 > 4 |
| (C:2, 4) > (C:1, 14) | C:2 > C:1 |
| (C:1, 14) > (B:10,5) | MED incomparable, 14 > 5 |



# Order-lattice duality

- From the preorder we can form a distributive lattice.
- $\min(A) = \{a \in A \mid \forall b \in A: \neg(b < a)\}$
- Elements of the lattice are sets with $A = \min(A)$
- Meet is $A \wedge B = \min(A \cup B)$

# Multipath routing

- Already here in a limited form
  - intradomain multipath for load balancing
  - ADD-PATHS capability
- Probably will be more widely used in future, including at the interdomain level

# In theory

- Can prove convergence for “min-union” model route choice, if underlying order has the right condition (same as for single-path case)
- “min” operation associated with a congruence $A \sim B \Leftrightarrow \min(A) = \min(B) \Leftrightarrow \uparrow(A) = \uparrow(B)$
- Form the quotient $2^S/\sim$

- From $(S, <, F)$ to $(2^S/\sim, \wedge, F)$

# Reductions and congruences

- “Reduction” operators of Wongseelashote are about putting (collections of) routes into a canonical form
- Each reduction defines a congruence
- Each congruence can be defined by a reduction
- Mediates between “operational” and “algebraic” approaches

# General “reduction”

- Take a semiring $(S, +, \cdot)$ and a function R from S to S.
- Define

  $x \oplus y = R(x + y)$

  $x \otimes y = R(x \cdot y)$

- We need a few more axioms to make this new $(S, \oplus, \otimes)$ into a semiring.

# Reduction axioms

- $R(R(x)) = R(x)$
- $R(x + y) = R(R(x) + y) = R(x + R(y))$
- $R(x \cdot y) = R(R(x) \cdot y) = R(x \cdot R(y))$

- Similar definition for a structure (S, +, F) where the set of functions F (from S to S) replaces the multiplication:

  $R(f(x)) = R(f(R(x)))$

# Error conditions

```
If the AS_PATH attribute of a BGP route contains an AS loop,
the BGP route should be excluded from the Phase 2 decision
function.  AS loop detection is done by scanning the full AS
path (as specified in the AS_PATH attribute), and checking
that the autonomous system number of the local system does not
appear in the AS path.
```

- AS_PATH length is usually the second attribute in the decision chain
- But if you ever see your own number in it, then you throw the route away at once

# AS loop detection

- Not well modelled as an export filter
  - A poisons the route by inserting the number of X; everybody *except* X could learn it
- A bit algebraically awkward
  - We want to combine attributes by lexicographic products, not by some weird operator that mixes the attributes up
  - Especially if the error-causing attribute is a long way down the chain

# Solution: mod out the problems

- Having defined the big long lexicographic product as usual...
- Make all routes with loops equivalent to “infinitely bad” route
- Associated reduction: check for a loop, discard path if found
- Separates concerns for purposes of property inference

# Filtering

- Routes can be filtered out (removed from consideration) before the decision process
- We also filter on output: only tell a route to some neighbors, not all of them

- Let’s not embellish our algebra
- Just define another congruence and reduction

# Filtering

- If F is the set to be filtered out, define

  $A \sim B \Leftrightarrow A/F = B/F$

- Reduction: A goes to A/F

- Separation of concerns: this $\sim$ is only doing one job, so it is simpler to reason about

# Summary

- Handle quite a few problematic cases, with not much extra mathematics
  - don’t need extra stuff in the algebra signature
  - one generic construction
- Heavily inspired by previous theory
  - we hope that the connection to such a rich body of theory will be useful, in ways we do not currently anticipate

# Future directions