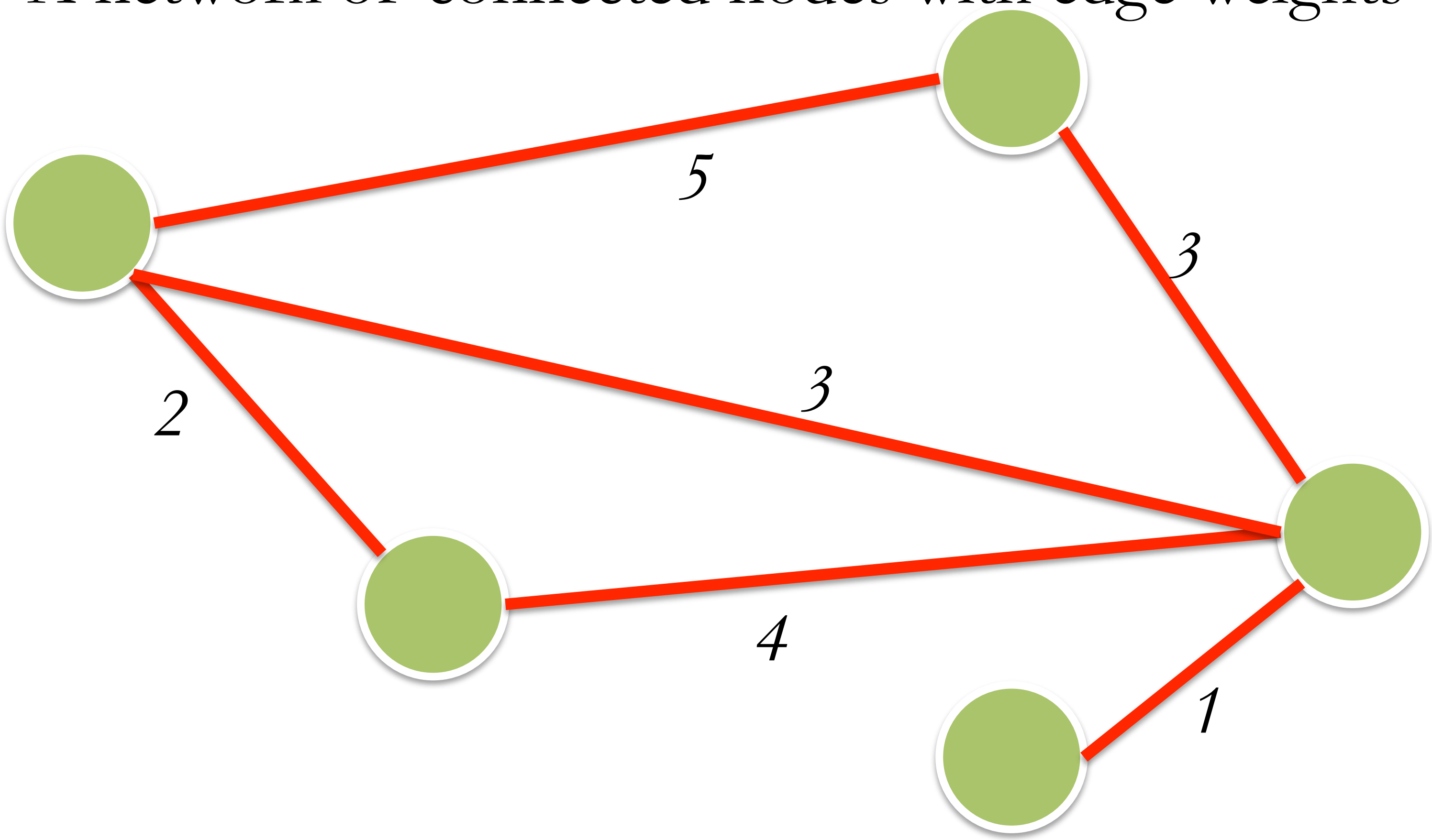
An Axiomatic Approach to Routing

Omer Lev, Moshe Tennenholtz and Aviv Zohar

TARK 2015

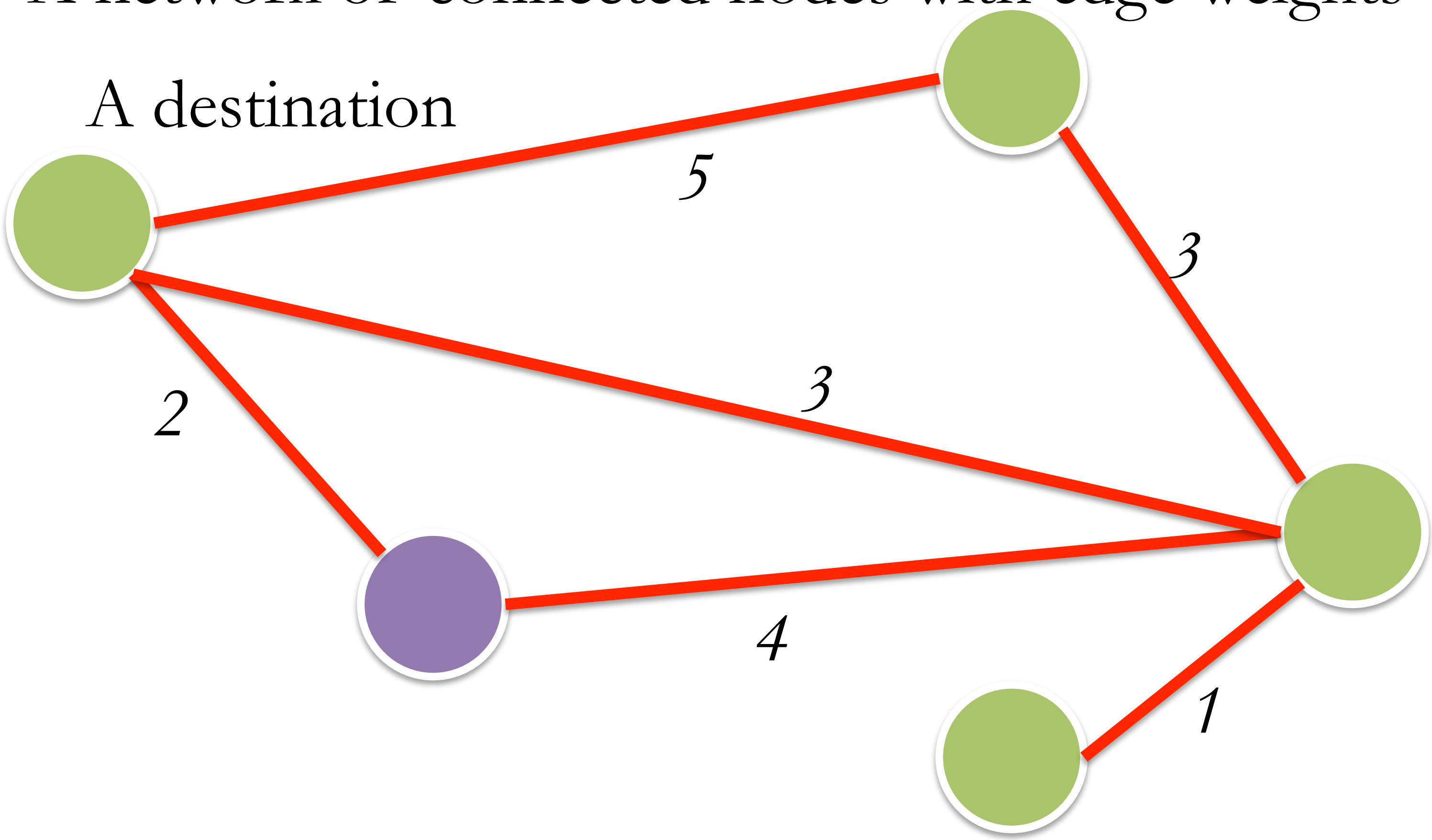
What is routing?

A network of connected nodes with edge weights



What is routing?

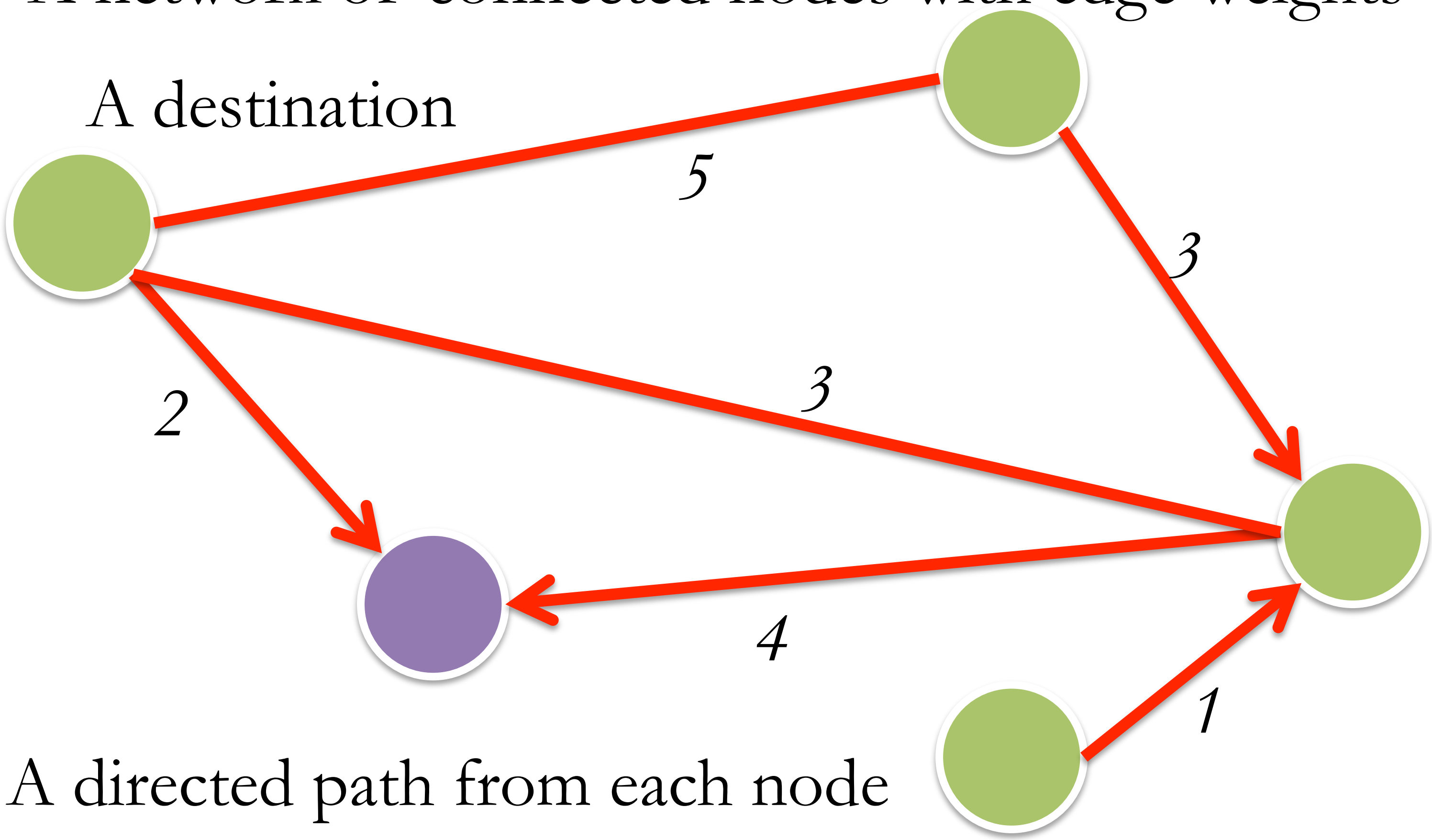
A network of connected nodes with edge weights



What is routing?

A network of connected nodes with edge weights

A destination

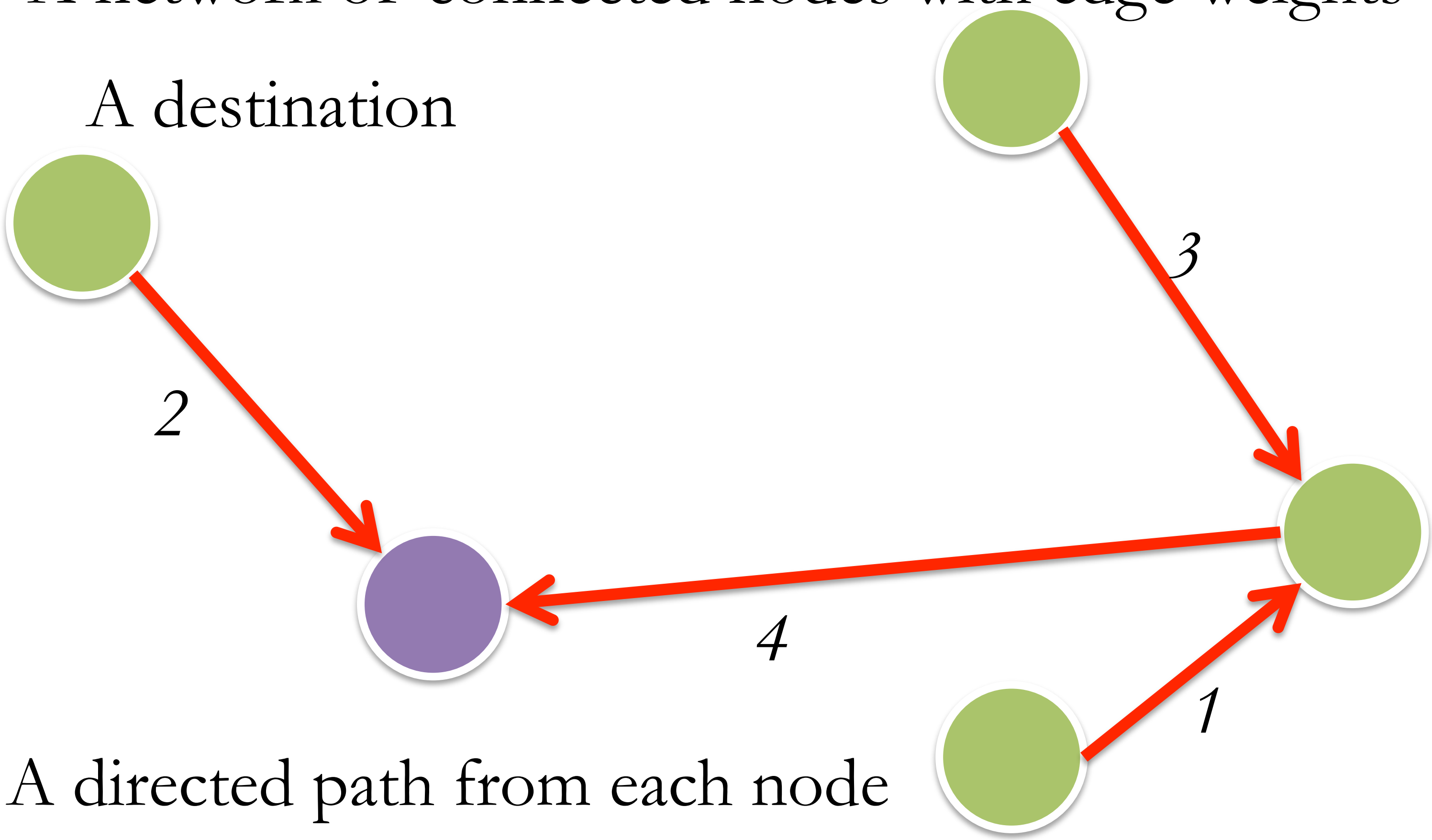


A directed path from each node

What is routing?

A network of connected nodes with edge weights

A destination



A directed path from each node

Creating a tree

What is routing?

Moving packets in a computer network: internal (central planner); or wide-area, internet like, network.

Flow of management authority in an organization.

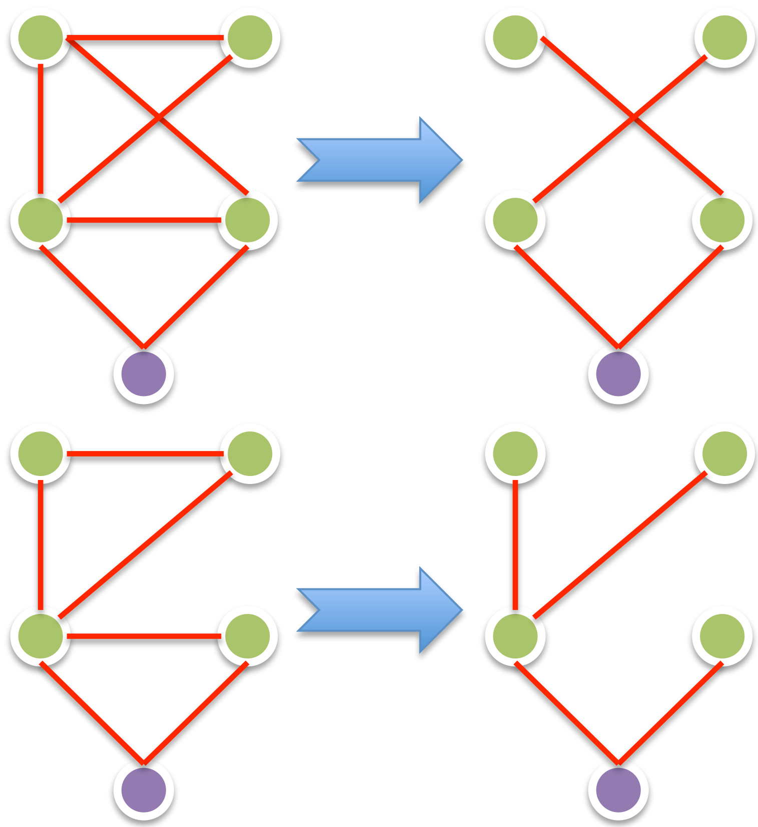
Moving information in a social structure.

What is the axiomatic approach?

Defining the properties of the system we desire, and build a system using them, instead of accepting the problems and limitations of a system we have thought of.

Axiom 1: Robustness

Removing an edge not in the routing tree does not change it. Removing one in it will not change the routing of nodes which haven't used it.



Axiom II: Scale invariance

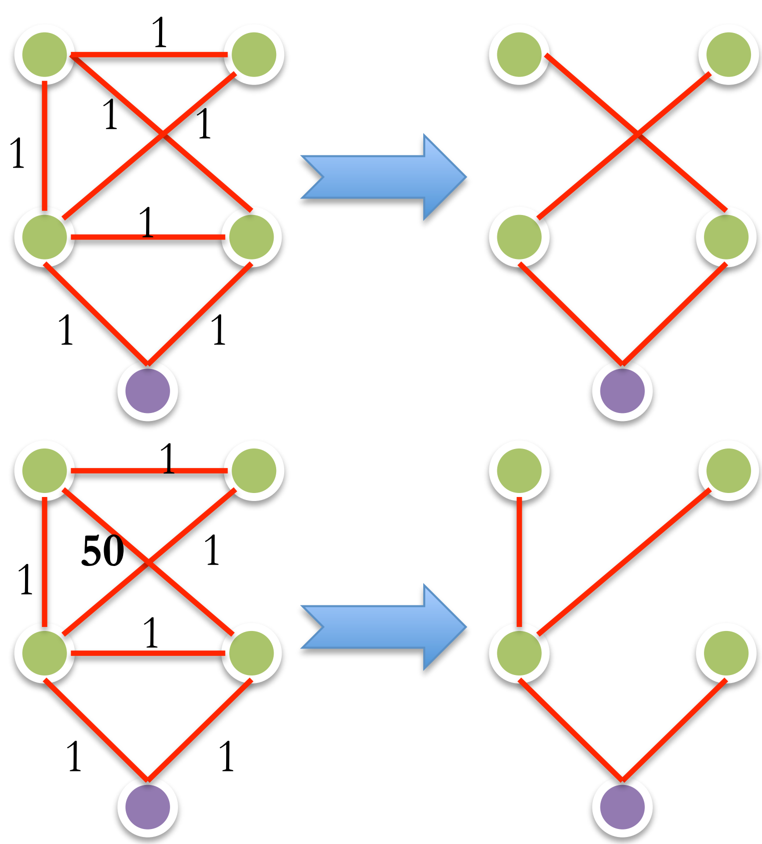
Multiplying all edges' weight does not change the routing tree.

Axiom III: Shift invariance

Adding a fixed amount to all edges' weight does not change the routing tree.

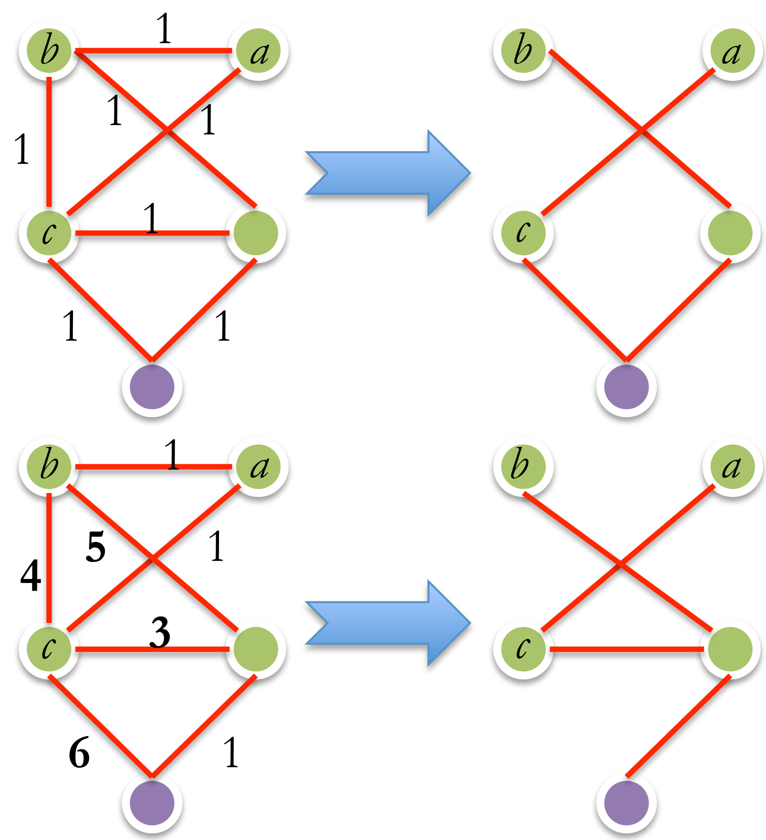
Axiom IV: Monotonicity

Increase an edge's weight by enough, and it will no longer be a part of the routing tree.



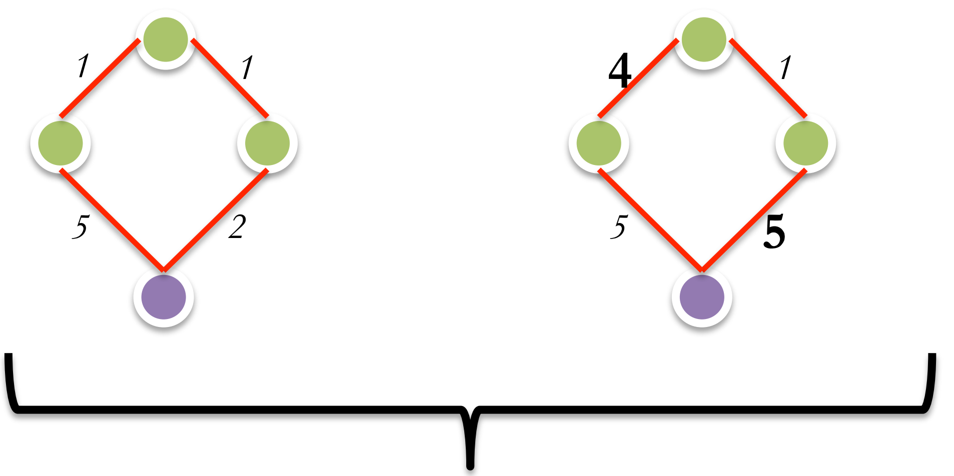
Axiom V: First hop ("economic")

The edge beginning the path from a node to the destination will not change if the weights of the potential edges do not change



Axiom VI: Path cardinal invariance

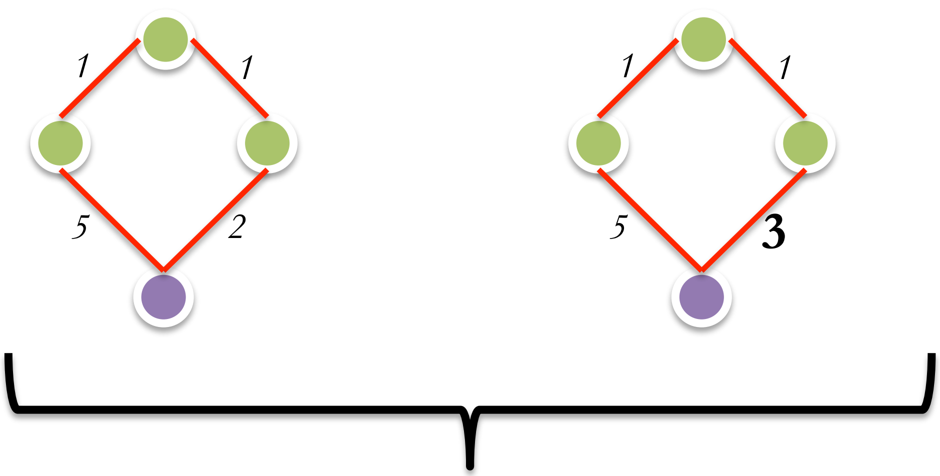
The routing choice between 2 parallel paths does not change as long as the sum of edges' weights remains the same.



Same routing tree

Axiom VII: Path Ordinal invariance

The routing choice between 2 parallel paths does not change as long as an edge's weight does not change its ranking in a list of all edges in the parallel path.



Same routing tree

Routing algorithms: Minimal Spanning Tree

Produce a tree with the smallest possible sum of edges' weight.

Theorem I

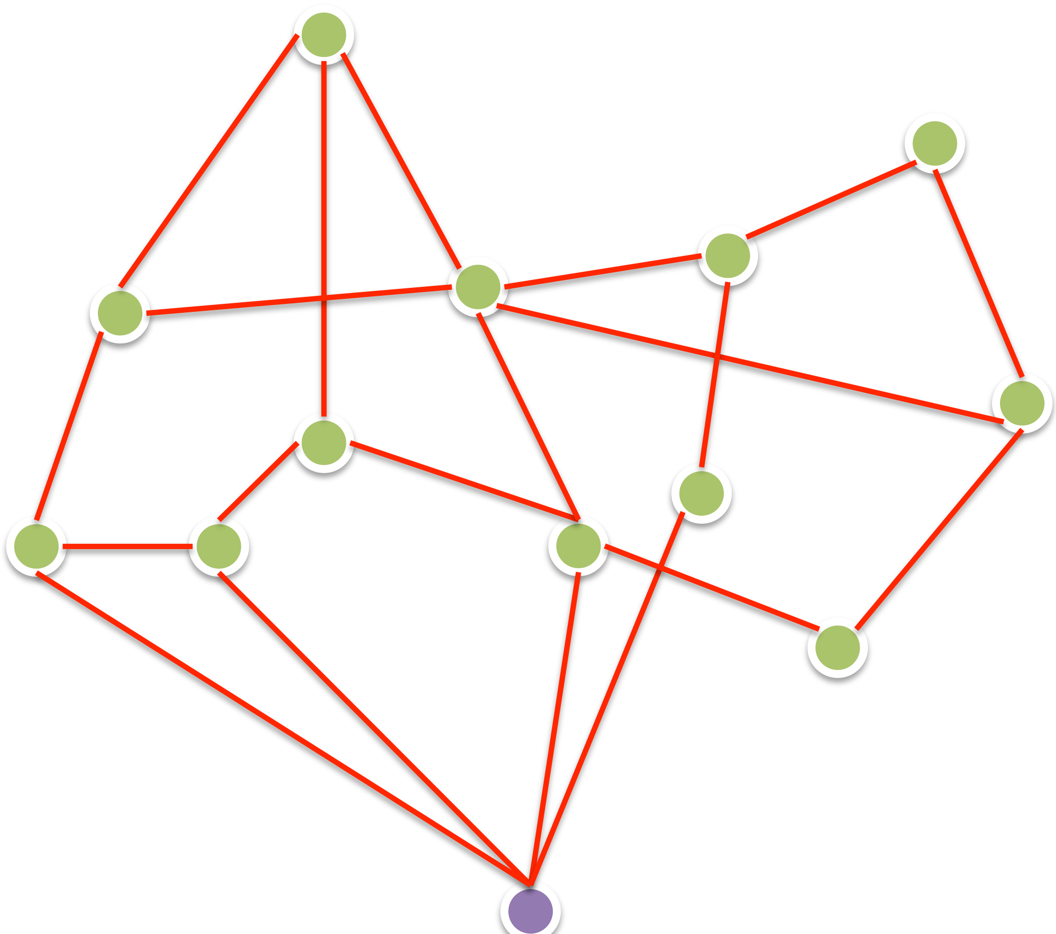
Axioms 1-5 (robustness, scale + shift invariance, monotonicity and first hop) uniquely define the Minimal Spanning Tree algorithm.

Proof sketch:
Kruskal reminder

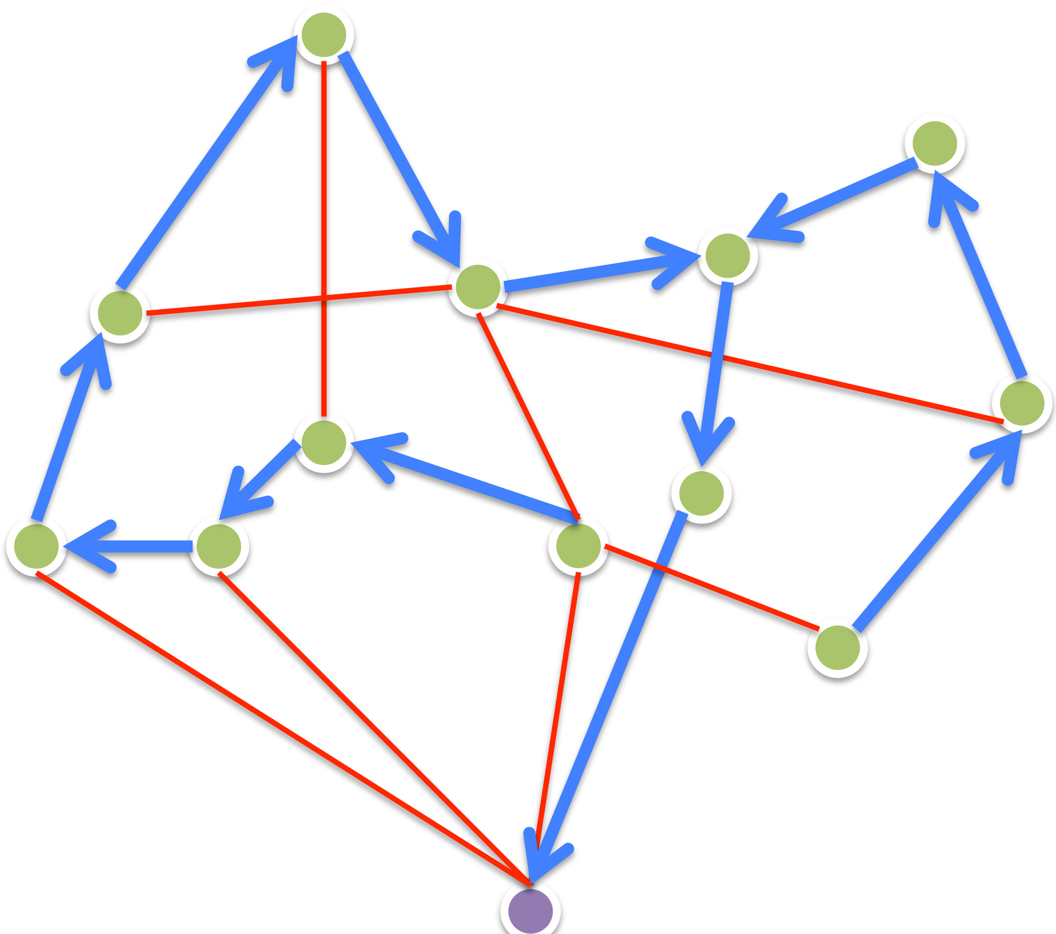
To find a minimal spanning tree:

- Order edges according to their weight.
- Pick *n-1* edges from lightest to heaviest, skipping on those creating a cycle.

Proof sketch

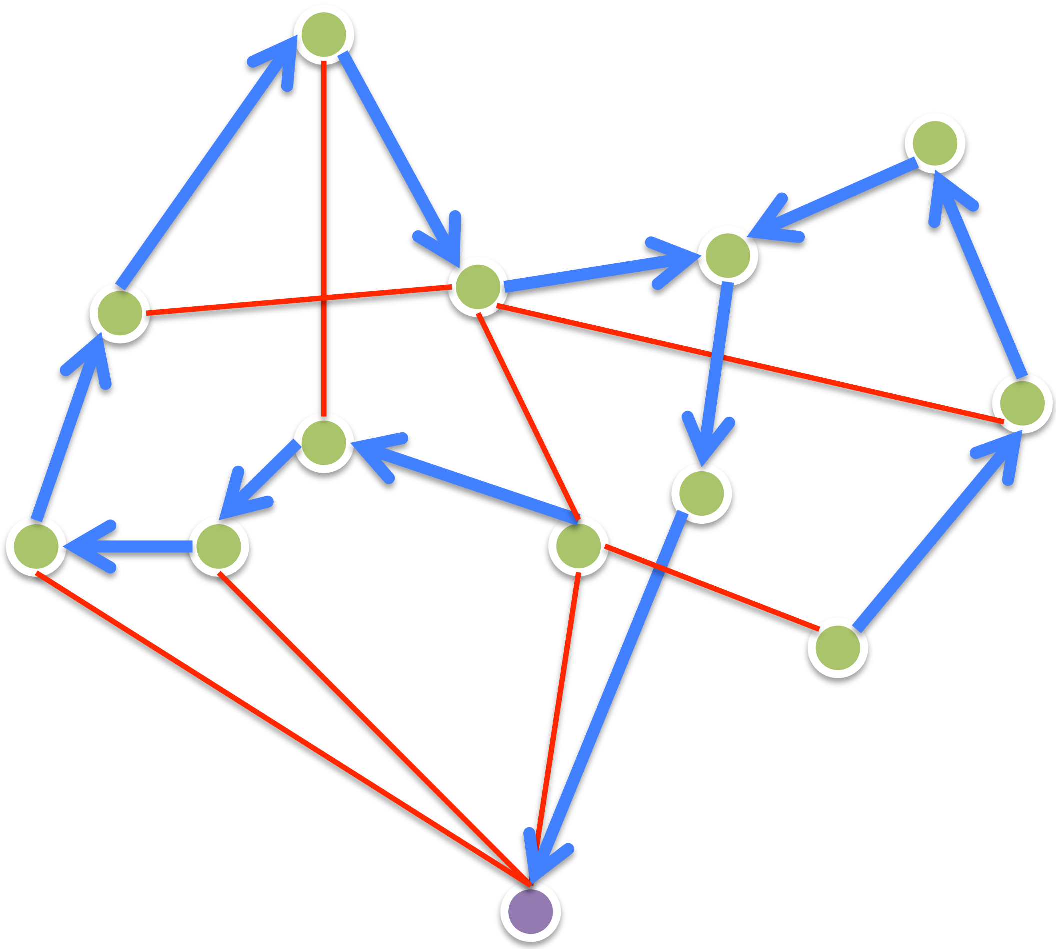


Proof sketch



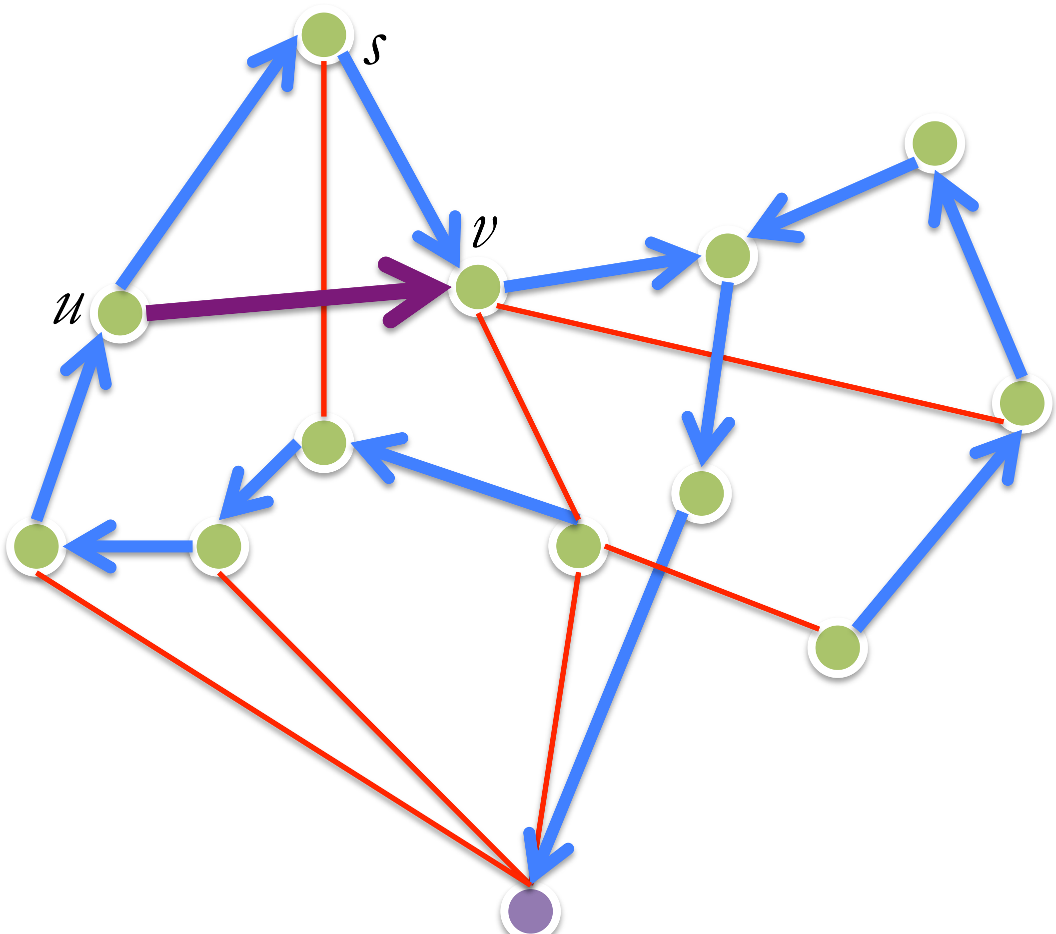
Proof sketch:

Assumption

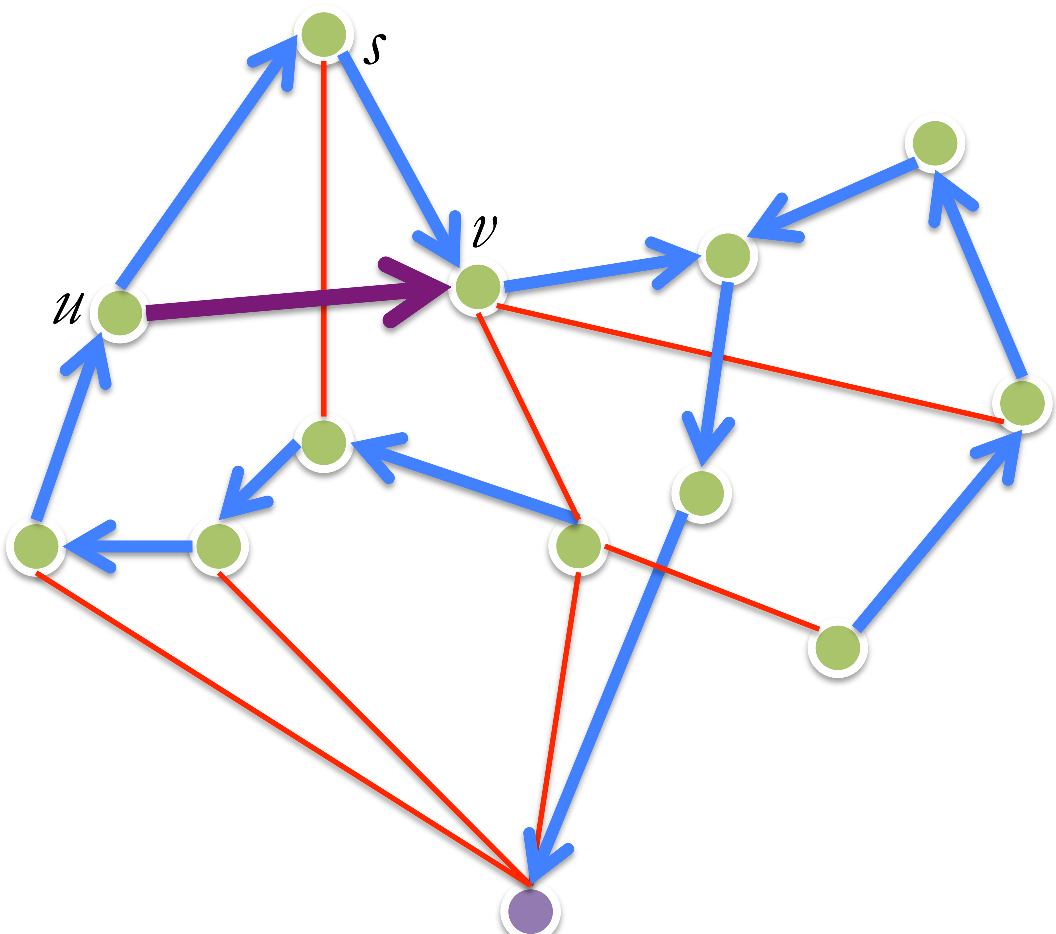


Proof sketch:

Assumption

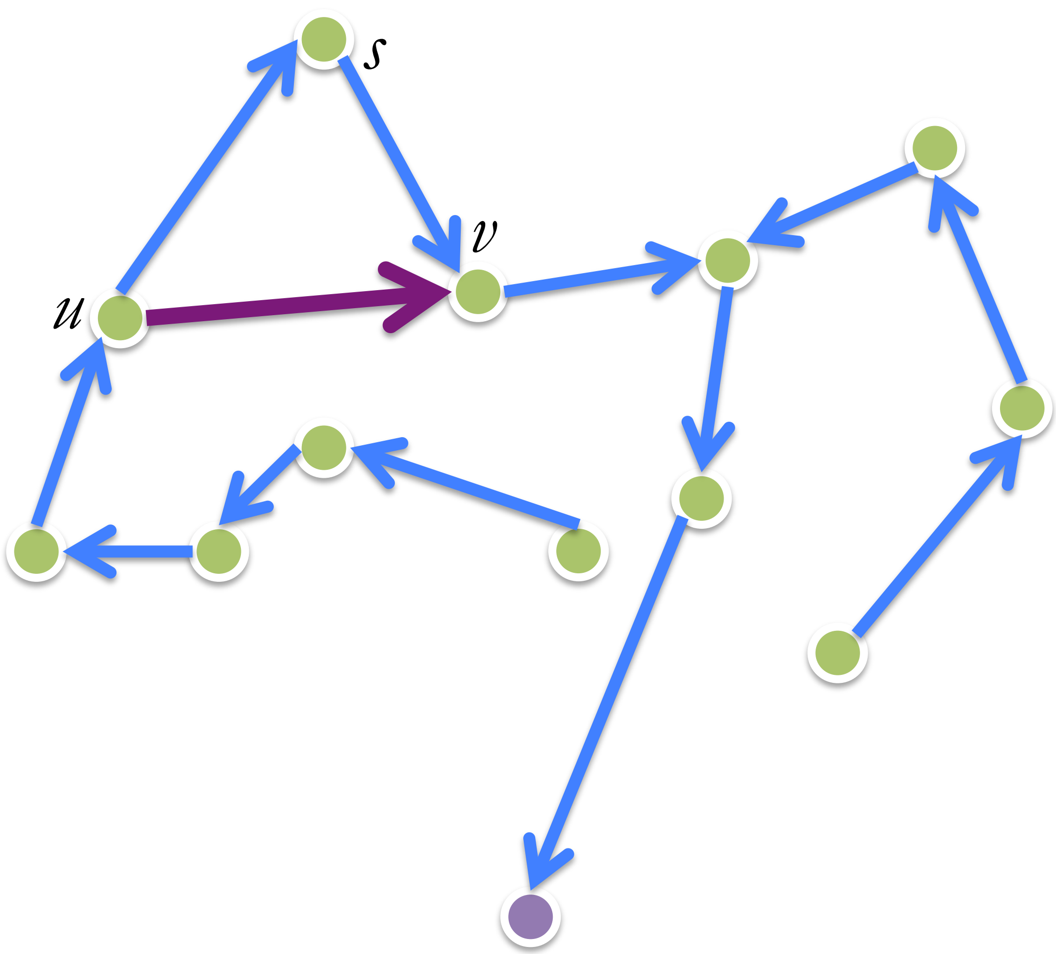


Proof sketch:
Applying robustness

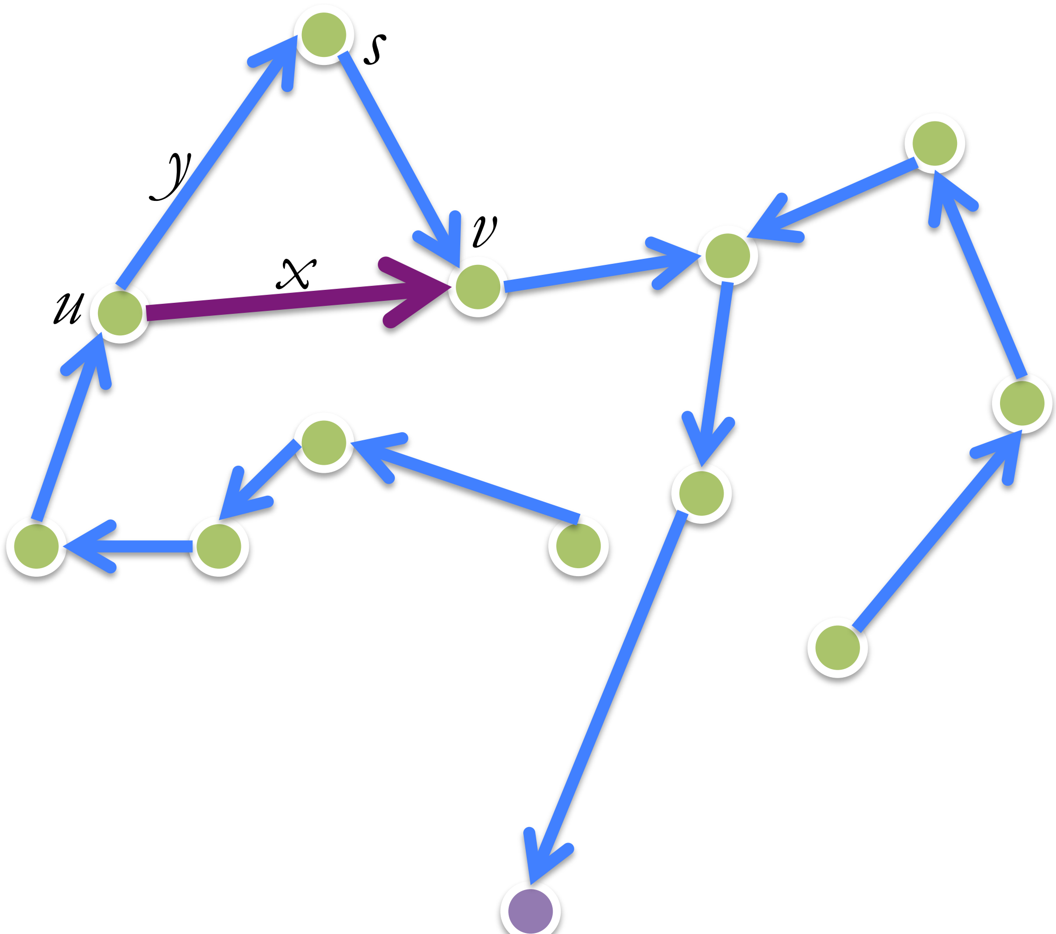


Proof sketch:

Applying robustness



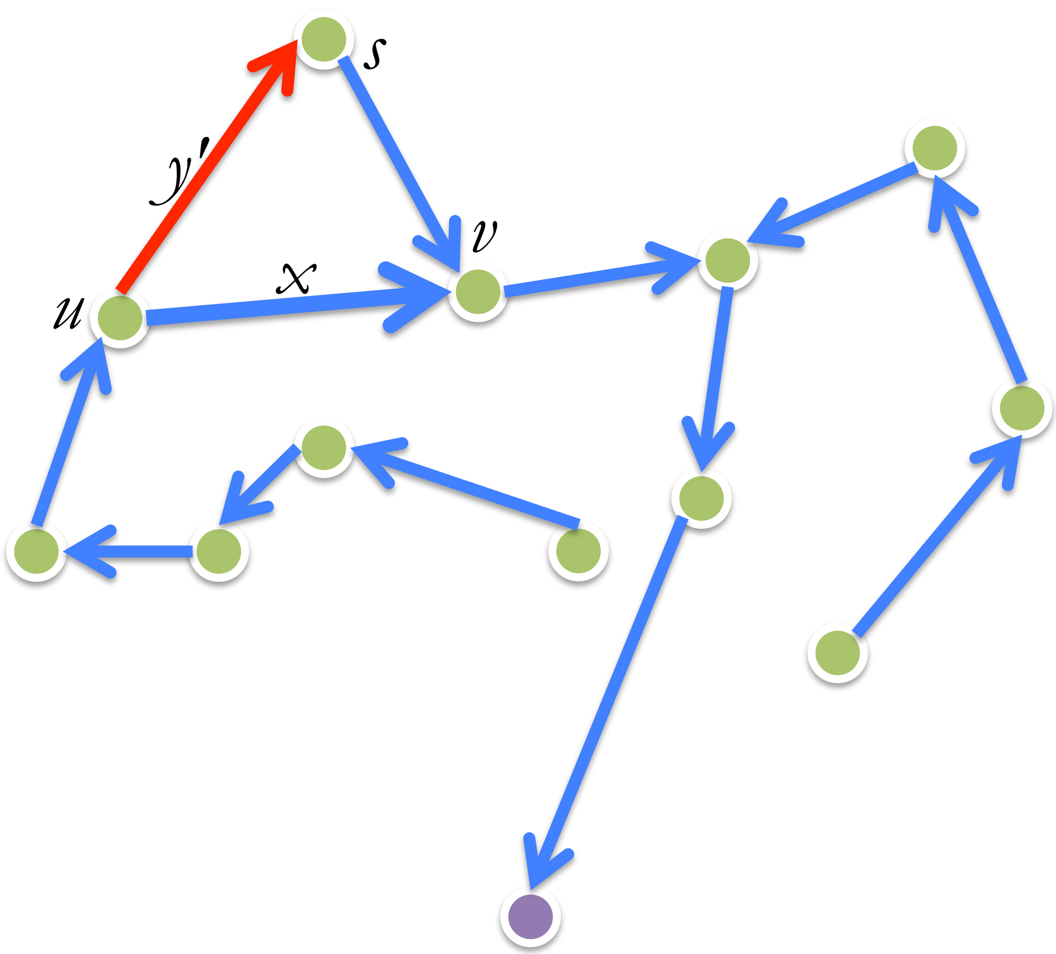
Proof sketch:
Applying monotonicity



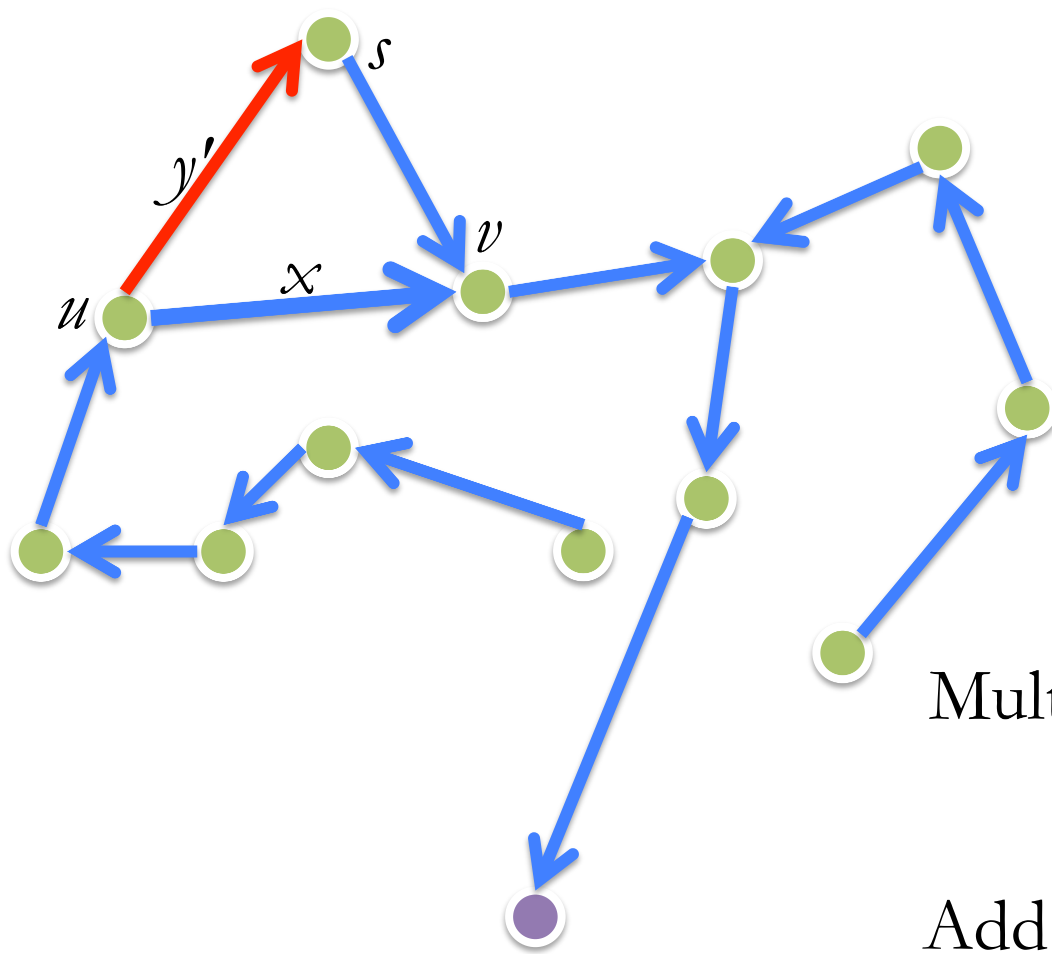
From the assumption $x<y$

Proof sketch:

Applying monotonicity



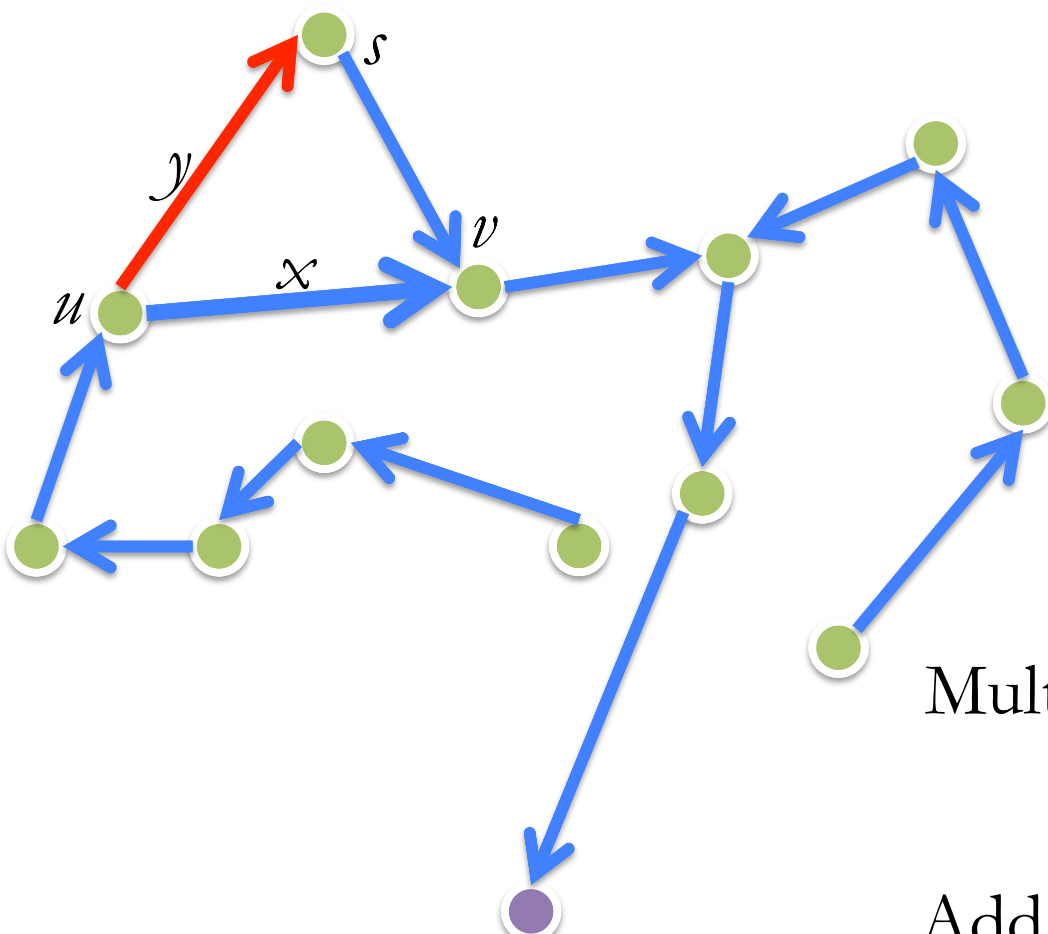
Proof sketch:
Applying scale and shift invariance



Multiply by $\dfrac{y-x}{y'-x}$

Add $y - \dfrac{y-x}{y'-x}y'$

Proof sketch:
Applying scale and shift invariance

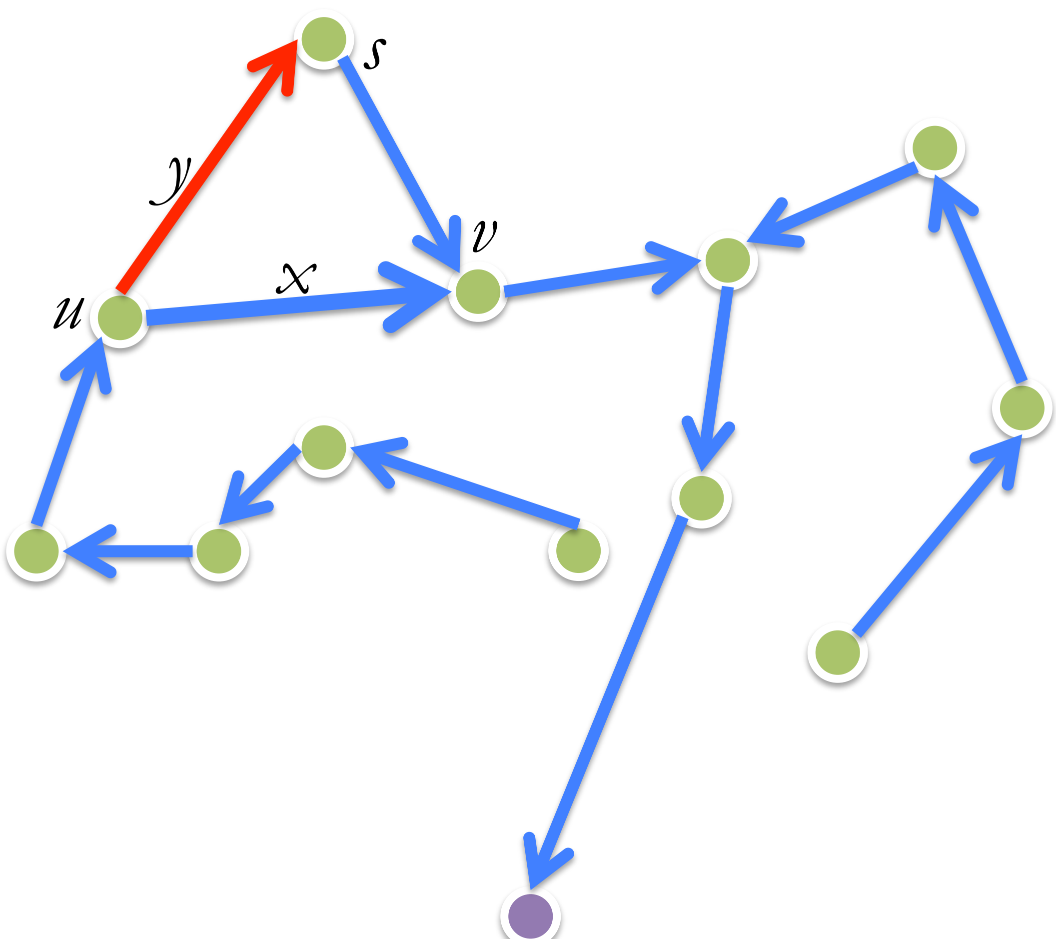


Multiply by $\dfrac{y-x}{y'-x}$

Add $y - \dfrac{y-x}{y'-x}y'$

Proof sketch:

Contradicting first hop



Routing algorithms: Shortest path

Each node is connected to the destination with the lightest path possible.

Theorem 2

Axioms 1-2, 4 and 6 (robustness, scale invariance, monotonicity and path cardinal invariance) uniquely define the shortest path routing algorithm.

(we lost shift invariance and first hop)

Routing algorithms: Weakest link

A path from a node to the destination has the value of the lightest edge on it. Each node is connected to the destination using the heaviest path available to it.

Theorem 3

Axioms 1-4 and 7 (robustness, scale invariance, shift invariance, inverse monotonicity and path ordinal invariance) uniquely define the weakest link routing algorithm.

(we lost first hop, and path cardinal invariance)

Future directions

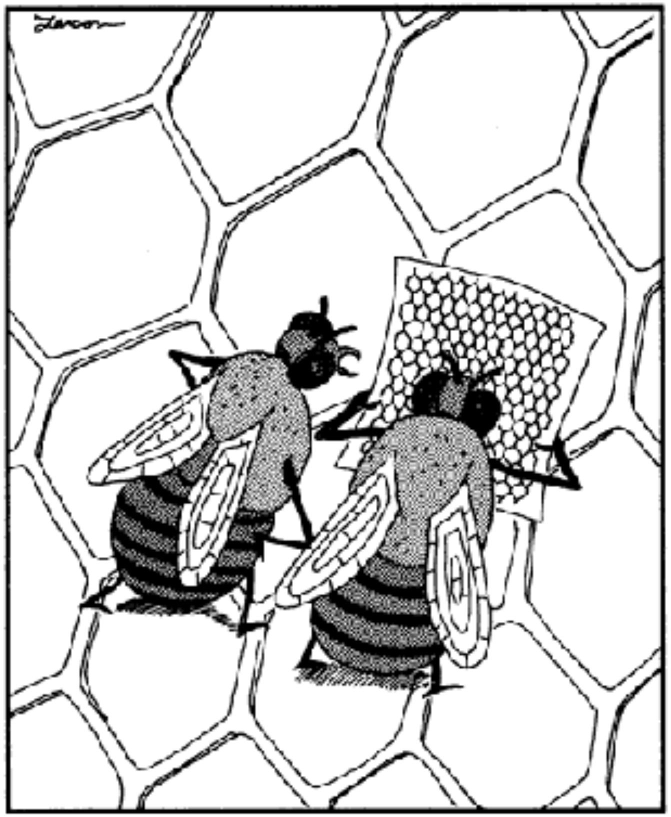
More axioms! More routing algorithms!

Social / organizational structure properties and implications

Characterizing families of algorithms implementing a property (e.g., "robust algorithms").

Formulating more economic and real-world constraints as axioms

Sometimes, even the best routing instructions don't work…



"Face it, Fred—you're lost!"

Thanks for listening