L11: Algebraic Path Problems with applications to Internet Routing

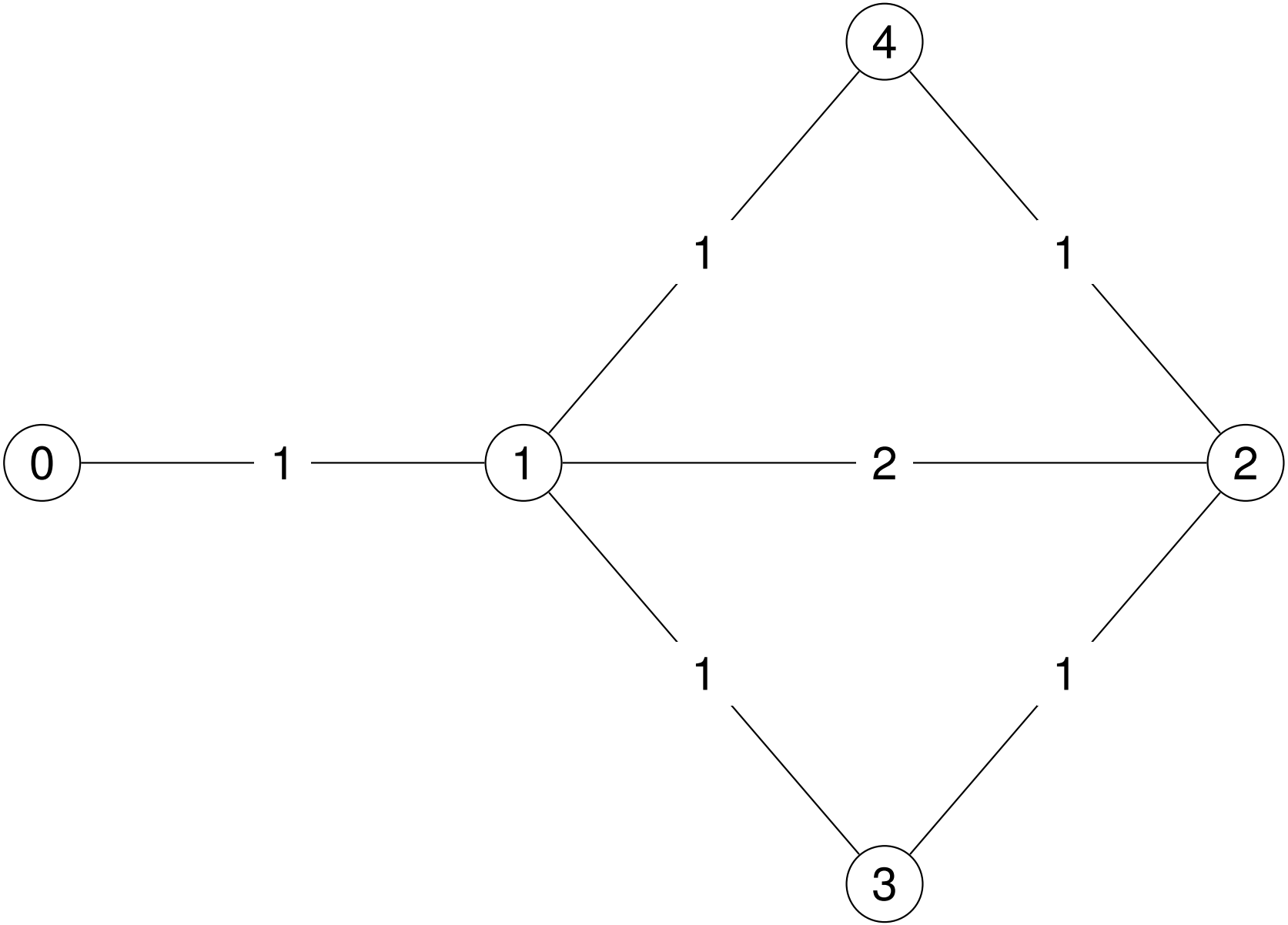
Lecture 01

Timothy G. Griffin

timothy.griffin@cl.cam.ac.uk
Computer Laboratory
University of Cambridge, UK

Michaelmas Term, 2014

Let's start with shortest paths!



Can represent a problem instance with an adjacency matrix

$$\mathbf{A} = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{ccccc} 0 & 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} \infty & 1 & \infty & \infty & \infty \\ 1 & \infty & 2 & 1 & 1 \\ \infty & 2 & \infty & 1 & 1 \\ \infty & 1 & 1 & \infty & \infty \\ \infty & 1 & 1 & \infty & \infty \end{bmatrix} \end{array}$$

But what problem are we solving?

Classic: globally optimal path weights

We want to find $\mathbf{A}^*$ such that

$$\mathbf{A}^*(i,\ j) = \min_{p \in P(i,\ j)} w(p),$$

where $P(i,\ j)$ is the set of all paths from i to j.

In the example:

$$\mathbf{A}^* = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{ccccc} 0 & 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 1 & 3 & 2 & 2 \\ 1 & 0 & 2 & 1 & 1 \\ 3 & 2 & 0 & 1 & 1 \\ 2 & 1 & 1 & 0 & 2 \\ 2 & 1 & 1 & 2 & 0 \end{bmatrix} \end{array}$$

An Algorithm: Dijkstra's

Input : adjacency matrix **A** and source vertex $i \in V$,
Output : the i-th row of **R**, where $\mathbf{R}(i,\ j)$ is the shortest distance from i to j in the graph represented by **A**.

(1) **for each** $q \in V$ **do** $\mathbf{R}(i,\ q) \leftarrow \infty$
(2) $S \leftarrow \{\}$; $\mathbf{R}(i,\ i) \leftarrow 0$
(3) **while** $S \neq V$ **do**
(4) find $q \in V - S$ such that $\mathbf{R}(i,\ q)$ is minimal
(5) $S \leftarrow S \cup \{q\}$
(6) **for each** $j \in V - S$ **do**
(7) $\mathbf{R}(i,\ j) \leftarrow \mathbf{R}(i,\ j)\ \min\ (\mathbf{R}(i,\ q) + \mathbf{A}(q,\ j))$

Run this $|V|$ times to get $\mathbf{R} = \mathbf{A}^*$.

But wait! What about the PATHS???

A bit of notation

Assume X and Y are sets of paths over E.

$$X \diamond Y \quad \equiv \quad \{pq \mid p \in X,\ q \in Y\}$$

Dijkstra's Algorithm Augmented With Paths

Input : adjacency matrix **A** and source vertex $i \in V$,
Output : the i-th row of **R** as before. Now with $\mathbf{P}(i,\ j)$ the set of **all** paths from i to j of distance $\mathbf{R}(i, j)$

(1) **for each** $q \in V$ **do** $\mathbf{R}(i,\ q) \leftarrow \infty;\ \mathbf{P}(i,\ q) \leftarrow \{\}$
(2) $S \leftarrow \{\};\ \mathbf{R}(i,\ i) \leftarrow 0;\ \mathbf{P}(i,\ i) \leftarrow \{\epsilon\}$
(3) **while** $S \neq V$ **do**
(4) find $q \in V - S$ such that $\mathbf{R}(i,\ q)$ is minimal
(5) $S \leftarrow S \cup \{q\}$
(6) **for each** $j \in V - S$ **do**
(7) **if** $\mathbf{R}(i,\ j) = \mathbf{R}(i,\ q) + \mathbf{A}(q,\ j)$
(8) **then** $\mathbf{P}(i,\ j) \leftarrow \mathbf{P}(i,\ j) \cup (\mathbf{P}(i,\ q) \diamond \{(q,\ j)\})$
(9) **else if** $\mathbf{R}(i,\ j) > \mathbf{R}(i,\ q) + \mathbf{A}(q,\ j)$
(10) **then** $\mathbf{R}(i,\ j) \leftarrow \mathbf{R}(i,\ q) + \mathbf{A}(q,\ j)$;
(11) $\mathbf{P}(i,\ j) \leftarrow \mathbf{P}(i,\ q) \diamond \{(q,\ j)\}$

Solution(s)

$$\mathbf{R} = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{ccccc} 0 & 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} 0 & 1 & 3 & 2 & 2 \\ 1 & 0 & 2 & 1 & 1 \\ 3 & 2 & 0 & 1 & 1 \\ 2 & 1 & 1 & 0 & 2 \\ 2 & 1 & 1 & 2 & 0 \end{bmatrix} \end{array}$$

$$\begin{aligned} \mathbf{P}(0,0) &= \{\epsilon\} \\ \mathbf{P}(0,1) &= \{(0,1)\} \\ \mathbf{P}(0,2) &= \{(0,1,2),\ (0,1,3,2),\ (0,1,4,2)\} \\ \mathbf{P}(2,1) &= \{(2,1),\ (2,3,1),\ (2,4,1)\} \\ \mathbf{P}(2,0) &= \{(2,1,0),\ (2,3,1,0),\ (2,4,1,0)\} \\ \vdots & \quad \vdots \quad \vdots \end{aligned}$$

Note : could use just the next hop to implement hop-by-hop forwarding.

Let's enrich the metric to Widest Shortest-Paths

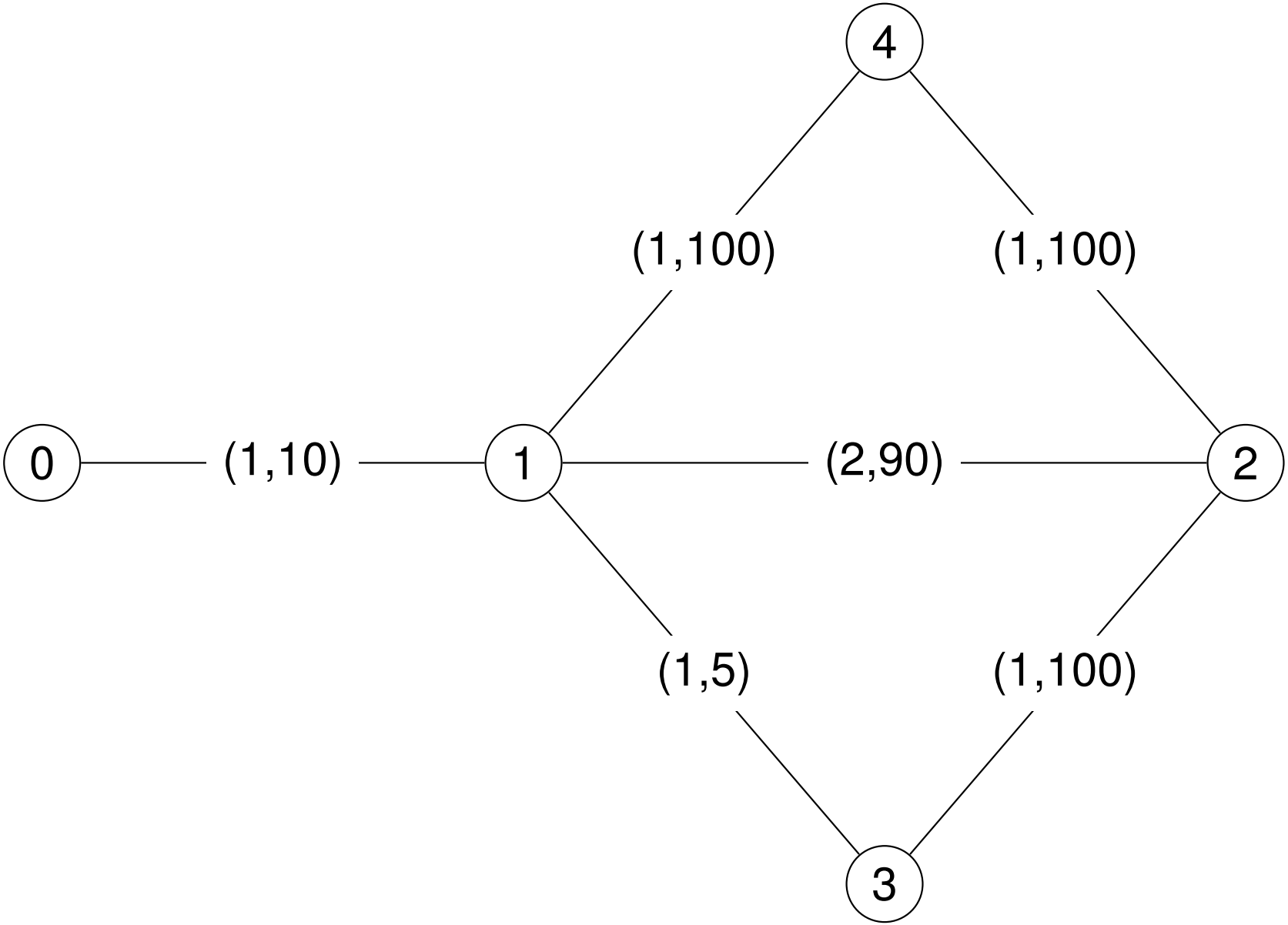
shortest paths	widest shortest paths
$\mathbb{N} \cup \{\infty\}$	$S_{\text{wsp}} \equiv (\mathbb{N} \times \{1, \ldots, \top\}) \cup \{\infty\}$
min	$\circ$
$+$	$\bullet$
0	$(0, \top)$

Can replace $+$ by $\bullet$ and min by $\circ$ in both Dijkstra and Bellman-Ford.

$$(a,\ b) \circ (c,\ d) \quad = \quad \begin{cases} (a,\ \ b \text{ max } d) & (a = c) \\ (a,\ b) & (a < c) \\ (c,\ d) & (c < a) \end{cases}$$

$$(a,\ b) \bullet (c,\ d) \quad = \quad (a + c,\ b \text{ min } d)$$

Add bandwith to link weights



Weights are globally optimal

Widest shortest-path weights computed by Dijkstra and Bellman-Ford

$$\mathbf{R} = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \\ 4 \end{array} \begin{array}{c} \begin{array}{ccccc} 0 & 1 & 2 & 3 & 4 \end{array} \\ \begin{bmatrix} (0,\top) & (1,10) & (3,10) & (2,5) & (2,10) \\ (1,10) & (0,\top) & (2,100) & (1,5) & (1,100) \\ (3,10) & (2,100) & (0,\top) & (1,100) & (1,100) \\ (2,5) & (1,5) & (1,100) & (0,\top) & (2,100) \\ (2,10) & (1,100) & (1,100) & (2,100) & (0,\top) \end{bmatrix} \end{array}$$

Four optimal paths of weight $(3,10)$. Do our algorithms find all of them?

$$\begin{aligned} \mathbf{P}_{\text{optimal}}(0,2) &= \{(0,1,2),\ (0,1,4,2)\} \\ \mathbf{P}_{\text{optimal}}(2,0) &= \{(2,1,0),\ (2,4,1,0)\} \end{aligned}$$

Surprise!

Four **optimal** paths of weight $(3, 10)$

$$\begin{aligned} \mathbf{P}_{\text{optimal}}(0,2) &= \{(0,1,2),\ (0,1,4,2)\} \\ \mathbf{P}_{\text{optimal}}(2,0) &= \{(2,1,0),\ (2,4,1,0)\} \end{aligned}$$

Paths computed by **Dijkstra**

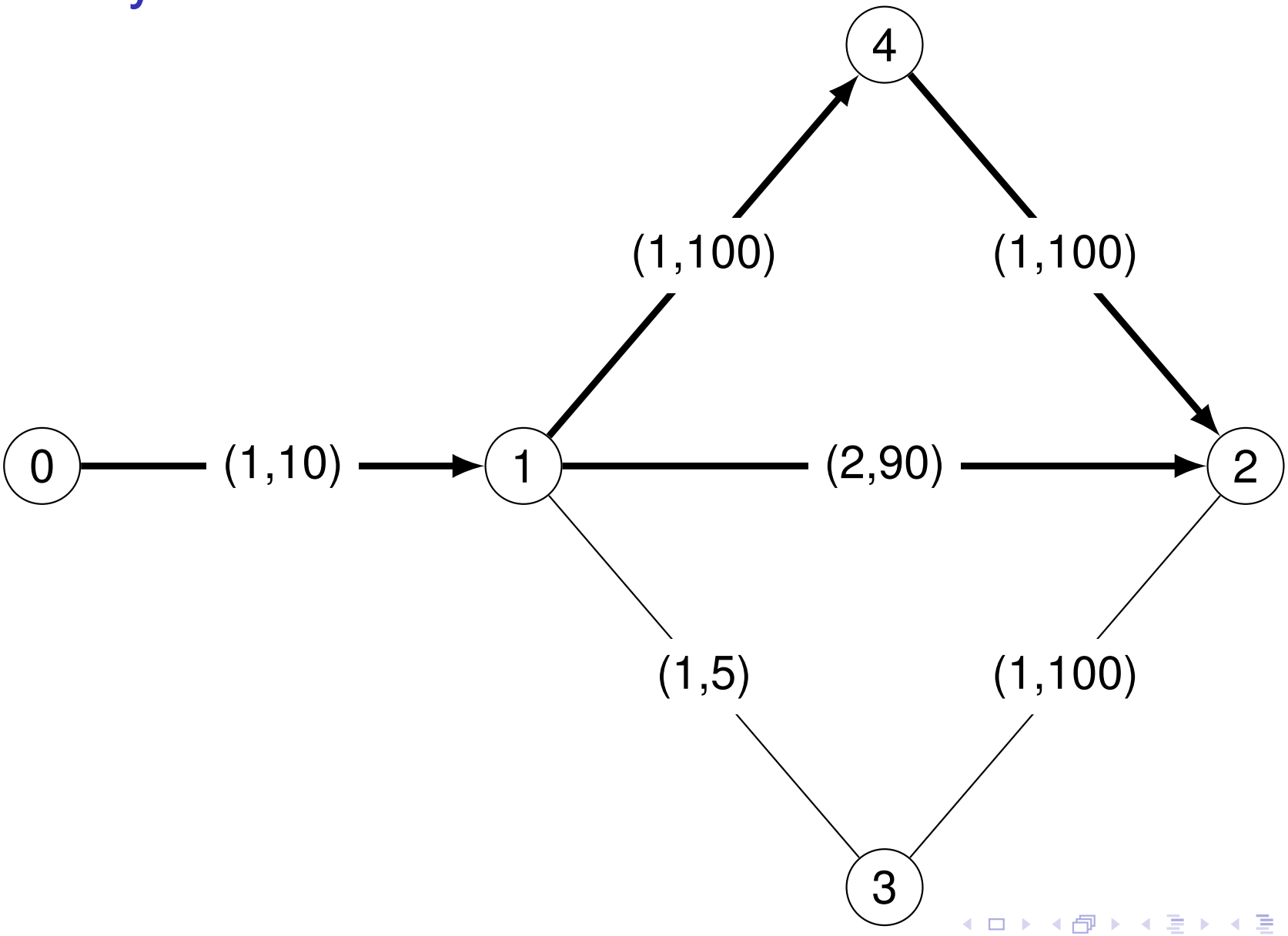
$$\begin{aligned} \mathbf{P}_{\text{Dijkstra}}(0,2) &= \{(0,1,2),\ (0,1,4,2)\} \\ \mathbf{P}_{\text{Dijkstra}}(2,0) &= \{(2,4,1,0)\} \end{aligned}$$

Notice that 0's paths cannot both be implemented with next-hop forwarding since $\mathbf{P}_{\text{Dijkstra}}(1,2) = \{(1,4,2)\}$.

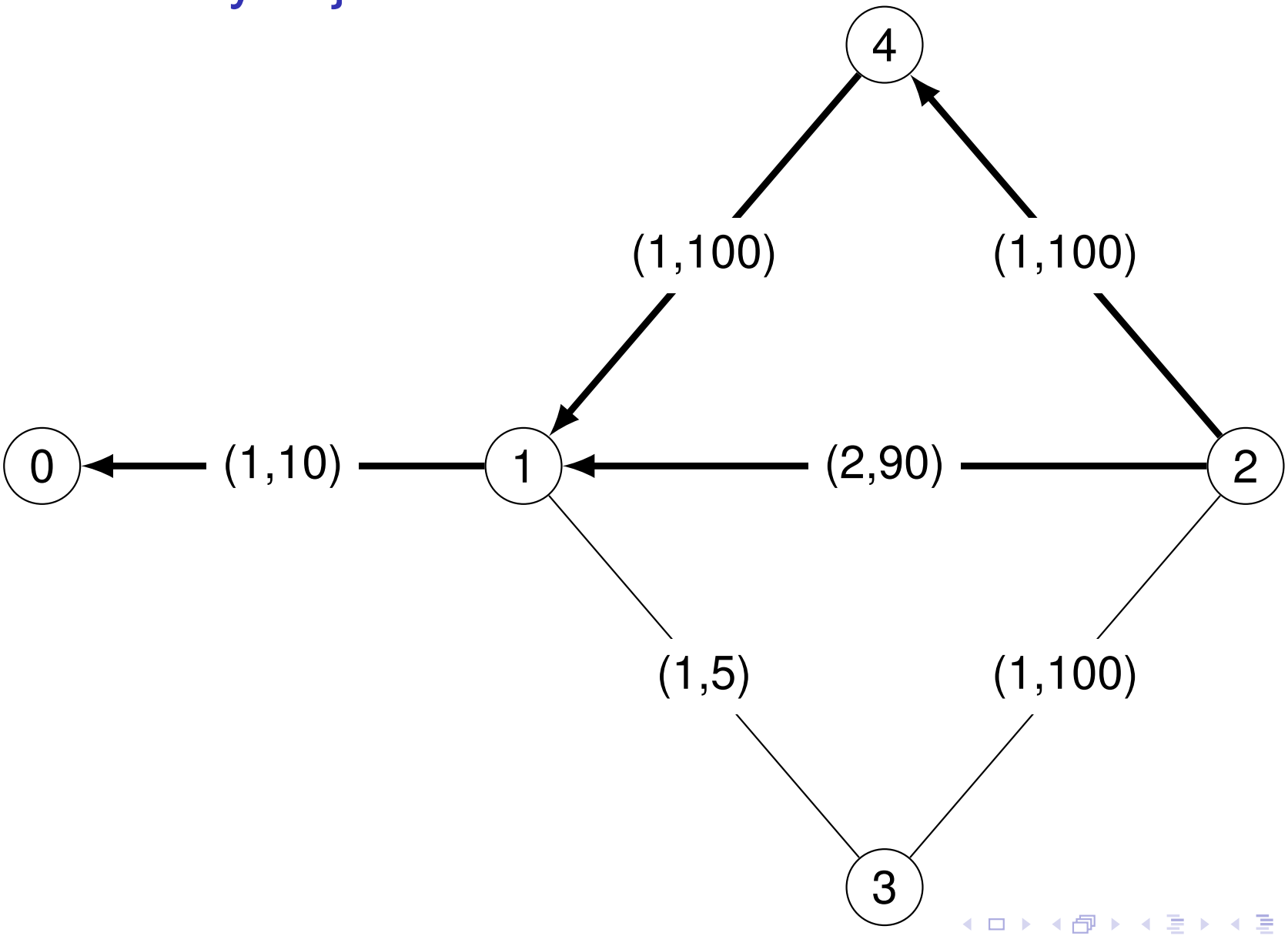
Paths computed by **Distributed Bellman-Ford** (Explained in later lectures)

$$\begin{aligned} \mathbf{P}_{\text{Bellman}}(0,2) &= \{(0,1,4,2)\} \\ \mathbf{P}_{\text{Bellman}}(2,0) &= \{(2,1,0),\ (2,4,1,0)\} \end{aligned}$$

Optimal paths from 0 to 2. Computed by Dijkstra but not by Bellman-Ford



Optimal paths from 2 to 1. Computed by Bellman-Ford but not by Dijkstra



Observations

For distributed Bellman-Ford

$$\begin{array}{rcl} \underline{\text{next-hop-paths}}(\mathbf{A}) & = & \underline{\text{computed-paths}}(\mathbf{A}) \\ & \subseteq & \underline{\text{optimal-paths}}(\mathbf{A}) \end{array}$$

For Dijkstra's algorithm

$$\begin{array}{rcl} \underline{\text{next-hop-paths}}(\mathbf{A}) & \subseteq & \underline{\text{computed-paths}}(\mathbf{A}) \\ & \subseteq & \underline{\text{optimal-paths}}(\mathbf{A}) \end{array}$$

We will see that all of these path sets coincide exactly when the metric is cancellative. That is, when $a \otimes b = a \otimes c$ always implies that $b = c$.

What is going on here???

Help!

- Are the algorithms broken?
- Is the new metric broken?

L11

This course will provide you with the tools to answert these questions!

see also ....

On the Forwarding Paths Produced by Internet Routing Algorithms. Seweryn Dynerowicz and Timothy G. Griffin. To be presented at ICNP 2013 on 10 October, 2013.

The **Tentative** Plan

Classical Semiring-based path finding

1	10 October	:	The Paths Puzzle
2	14 October	:	Semigroups and Order Relations
3	17 October	:	Semirings — Theory
4	21 October	:	Semirings — Constructions
5	24 October	:	Semirings — Examples
6	28 October	:	Semirings — algorithms

The **Tentative** Plan

Non-Classical methods : Local Optimality

7	31 October	:	Internet Routing Protocols : RIP, OSPF, IS-IS
			HW 1 due
8	4 November	:	Internet Routing Protocols : EIGRP, BGP
9	7 November	:	Beyond Semirings — “functions on arcs”
10	11 November	:	Beyond Semirings — Global vs Local optimality
11	14 November	:	Bellman-Ford and Dijkstra revisited
12	18 November	:	Solving the Paths Puzzle
13	21 November	:	Graph (Network) decomposition
			HW 2 due
14	25 November	:	More on Global vs Local optimality
15	28 November	:	Internet’s Route Redistribution and Administrative Distance
16	2 December	:	Metarouting project and open problems
	15 January	:	**HW 3 due**

Our Approach

The Algorithm to Algebra (A2A) method

$$\begin{pmatrix} \text{original metric} \\ + \\ \text{complex algorithm} \end{pmatrix} \rightarrow \begin{pmatrix} \text{modified metric} \\ + \\ \text{generic algorithm} \end{pmatrix}$$

Punch Line

A2A attempts to shift complexity from an algorithm to the metric, which is captured in an algebraic structure — the algebraic properties of that structure will determine what kind of solution is obtained (global or local optima).