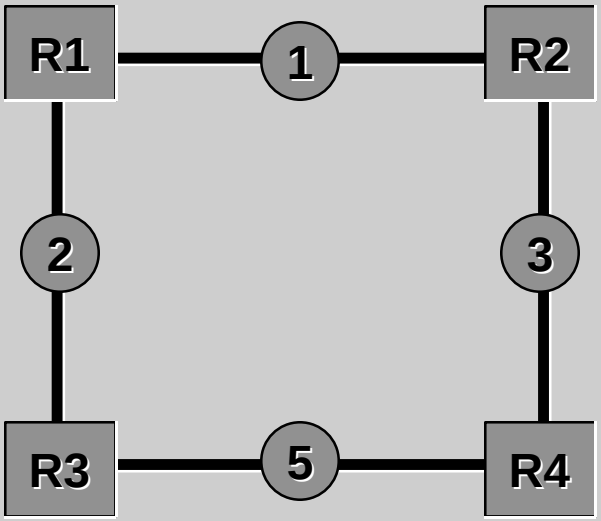
Finding The Shortest Path

V1.2: Geoff Bennett

FORE SYSTEMS

The Distance Matrix

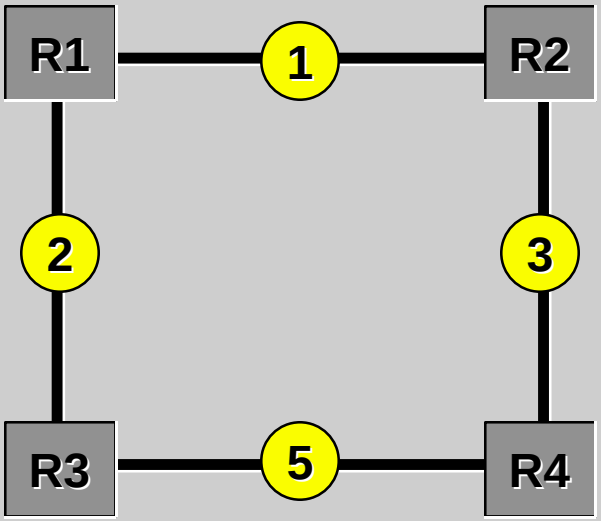
To calculate the Shortest Path we need to be able to store route costs in a convenient way.
The data structure used by modern Routing Protocols is the Distance Matrix.



Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

This is the Distance Matrix that I developed in the tutorial “Representing Network Link Costs”.



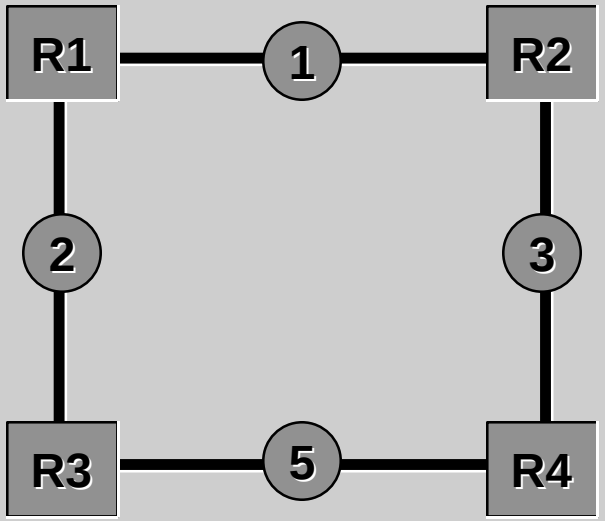
Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

The circled numbers represent the *Link Cost*.

In the tutorial “Overview of Routing Metrics” I discuss the relative merits of the various cost metrics used by modern Routing Protocols.

For this example, I’ll use a simple, single digit representation.



Distance Matrix

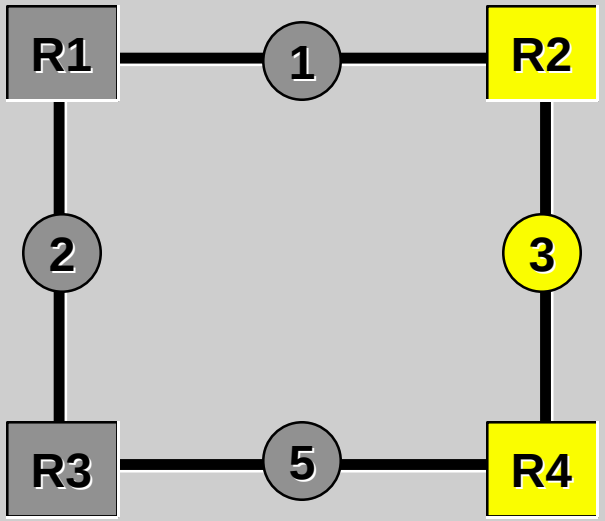
$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

As humans we can easily recognise the shortest path between any two points. In this case the “shortest path” is the one having the lowest *total* link cost.

To calculate the “Shortest Path First”, routers can use iterative (ie. repetitive) algorithms.

Using such algorithms, computers can still calculate the shortest paths in very complex networks when human estimates would become unreliable.

Unfortunately, in the smaller examples that I’ll be using, the iterative algorithms look rather tedious.



Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

Note that the Routing Protocols that are used to build the distance matrix are based on a *reliable* update technique. This means that a pair of routers is able to form a “partnership”, known as an *adjacency*.

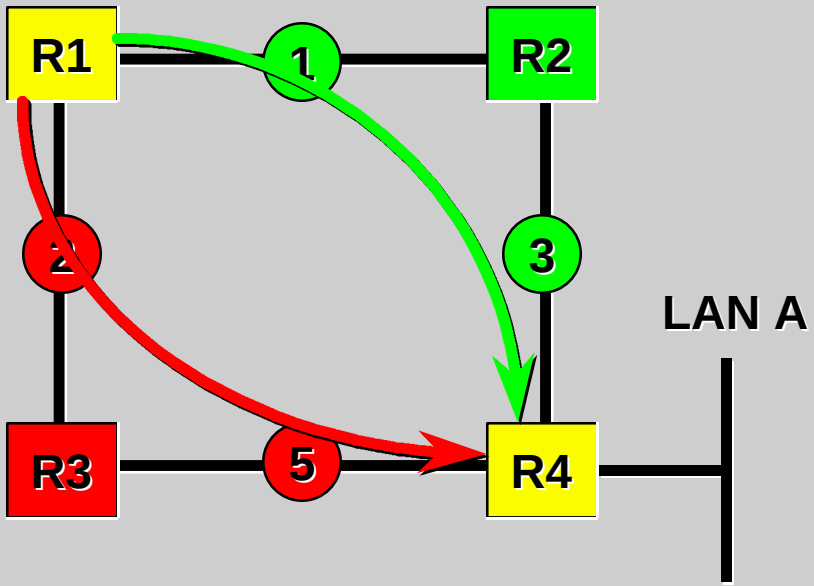
The benefit of the adjacency is that R2, for example, is able to confidently advertise the cost of its link to R4.

This differs from the *relative* update mechanism used by Bellman-Ford algorithms.

FORE SYSTEMS

The Dijkstra Algorithm

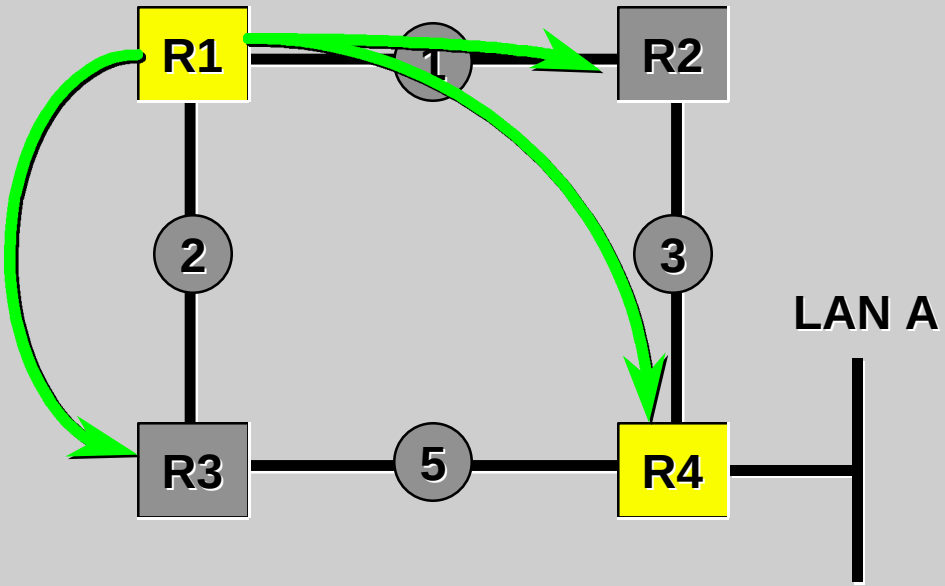
Modern routing protocols such as OSPF and Integrated ISIS use an SPF algorithm developed by Dijkstra.



When R1 runs Dijkstra, it will know for example that the shortest route for a packet that is addressed to LAN A is the green route I've drawn on the diagram. This has a cost of 4 (1 + 3).

R1 knows that the red route to LAN A is also available, but the red route has a cost of 7.

If the green route ever becomes unavailable, then R1 can immediately switch to the red route.

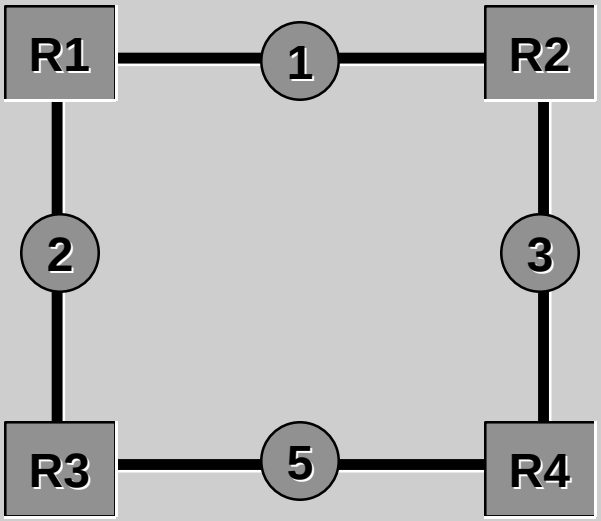


R1 will run Dijkstra for each possible router destination in the internetwork.

This is one reason why many people feel that Dijkstra is a processor-intensive task.

In fact, in a well designed OSPF network it is possible to limit the scope of Dijkstra calculations by reducing the number of routers or links in a given OSPF area.

Stage 1: Building the Distance Matrix



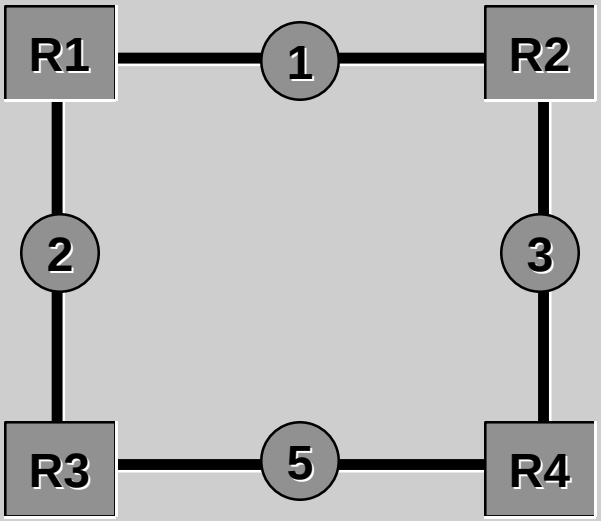
Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

The first thing we have to do is to create the distance matrix for the network.

In OSPF, this is done by the Link State Advertisement protocol. Essentially each router simply advertises the cost of each directly attached link.

Stage 2: Define a Data Structure



Distance Matrix

∞	1	2	∞
1	∞	∞	3
2	∞	∞	5
∞	3	5	∞

The Distance Matrix is actually the same for every router in the network.

But any single computation of Dijkstra tells us the shortest route to a given network.

Basically that means that we can't use a Distance matrix as the data structure to store our routes, or any intermediate calculations.

FORE SYSTEMS

r=R1

Node	R1	R2	R3	R4
l()	0	∞	∞	∞
Permanent?	yes	no	no	no

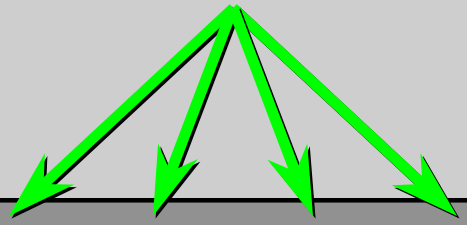
Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

This is the table I'll use to store the Dijkstra calculations.

FORE SYSTEMS

r=R1



Node	R1	R2	R3	R4
l()	0	∞	∞	∞
Permanent?	yes	no	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

The whole table is calculated from the point of view of one of the routers in the network, in this case R1.

Each column represents a router that we're trying to find a route to.

FORE SYSTEMS

r=R1

Node	R1	R2	R3	R4
l()	0	∞	∞	∞
Permanent?	yes	no	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

This notation, is the cost label for a given router, shown as l().

FORE SYSTEMS

r=R1

Node	R1	R2	R3	R4
l()	0	∞	∞	∞
Permanent?	yes	no	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

The cost labels will be optimised as the calculation proceeds.

At some stage (decided by the algorithm), the labels become permanent. If the labels are not permanent, we run the calculation again.

When the label *does* become permanent, then the label value is the cost of the shortest route to this router.

FORE SYSTEMS

r=R1

Node	R1	R2	R3	R4
l()	0	∞	∞	∞
Permanent?	yes	no	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

The table is run from the point of view of one router, and each loop looks at one destination router at a time.

This box tells us the last label which we made permanent.

FORE SYSTEMS

r=R1

Node	R1	R2	R3	R4
l()	0	∞	∞	∞
Permanent?	yes	no	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

With any algorithm, we need a fixed point of reference, so we may as well start from our own router.

In Dijkstra, we set the label *value* for our own router to 0, and say that it's *permanent*.

FORE SYSTEMS

r=R1

Node	R1	R2	R3	R4
l()	0	∞	∞	∞
Permanent?	yes	no	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

The objective of Dijkstra is to keep on calculating until these labels become permanent.

When all the labels are permanent, the numbers in the l() row will indicate the *cost* of the shortest path *from* R1 *to* the appropriate router destination.

So how do we achieve this?

Stage 3: Running The Algorithm

r=R1

Node	R1	R2	R3	R4
l()	0	∞	∞	∞
Permanent?	yes	no	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

Let's move onto the first destination candidate - R2. We'll try to make the label for R2, l(2) permanent.

Dijkstra Pass 1

r=R1

First Value

Node	R1	R2	R3	R4
l()	0	∞	∞	∞
Permanent?	yes	no	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

Dijkstra says that we do this by comparing two values.

The first is the current value of the label for the node that we're trying to make permanent, in this case R2. The current value for this label is infinity.

Dijkstra Pass 1

r=R1

Node	R1	R2	R3	R4
l()	0	∞	∞	∞
Permanent?	yes	no	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

Second Value = This + This

The second value is the sum of the value of the last label that was made permanent (in this case R1, which is zero), plus the distance from this node to the node we're considering (in this case R2, which is 1).

The second value is, therefore (0+1=1).

Dijkstra Pass 1

r=R1

Node	R1	R2	R3	R4
l()	0	1	∞	∞
Permanent?	yes	no	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

We set the new label value for R2 to the *smaller* of the two values - infinity or 1.

Now we repeat this comparison for all the remaining nodes with non-permanent labels.

Dijkstra Pass 1

r=R1

Node	R1	R2	R3	R4
l()	0	1	∞	∞
Permanent?	yes	no	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

First Value **Second Value = This + This**

The R3 comparison value is the smaller of infinity (the current value for the R3 label) and 2 (the sum of the R1 label, 0 and the distance from R1 to R3, 2).

Dijkstra Pass 1

r=R1

Node	R1	R2	R3	R4
l()	0	1	2	∞
Permanent?	yes	no	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

First Value **Second Value = This + This**

The R4 value is the smaller of infinity (the current value) and infinity (the sum of the R1 label, 0 and the distance from R1 to R4, infinity).

Dijkstra Pass 1

r=R1

Node	R1	R2	R3	R4
l()	0	1	2	∞
Permanent?	yes	no	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

So these are the three calculated figures.

Dijkstra Pass 1

r=R1

Node	R1	R2	R3	R4
l()	0	1	2	∞
Permanent?	yes	yes	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

The *smallest* of the values has its label made *permanent*.

Dijkstra Pass 1

r=R2

Node	R1	R2	R3	R4
l()	0	1	2	∞
Permanent?	yes	yes	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

We also update the marker to show which label was the last to be made permanent.

Dijkstra Pass 1

r=R2

Node	R1	R2	R3	R4
l()	0	1	2	∞
Permanent?	yes	yes	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

This is a single pass of the Dijkstra Algorithm, with a successful attempt to make a label permanent.

We now have to continue until all the labels are permanent. However, after each loop there will be fewer nodes with non-permanent labels to consider.

Dijkstra Pass 2

r=R2

Node	R1	R2	R3	R4
l()	0	1	2	∞
Permanent?	yes	yes	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

First Value **Second Value = This + This**

At the start of the second pass, we see that R3 still has a non-permanent label.

We compare the *current* value of the label (2) with the sum of 1 (the value of the label of the last node made permanent) plus infinity (the distance from R3 to the last node made permanent).

In this comparison, the smaller of 2 and infinity is 2.

Dijkstra Pass 2

r=R2

Node	R1	R2	R3	R4
l()	0	1	2	∞
Permanent?	yes	yes	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

First Value **Second Value = This + This**

R4 also has a non-permanent label.

If we compare its current label value (infinity) with the sum of 1 (the value of the label for the last label made permanent) plus 3 (the distance from R4 to the last node whose label was made permanent, R2), then 4 will become the R4 comparison.

Dijkstra Pass 2

r=R2

Node	R1	R2	R3	R4
l()	0	1	2	4
Permanent?	yes	yes	no	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

The lower of these two values (2 and 4) is obviously 2.

Dijkstra Pass 2

r=R3

Node	R1	R2	R3	R4
l()	0	1	2	4
Permanent?	yes	yes	yes	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

So we make the label for R3 permanent.

Dijkstra Pass 3

r=R3

Node	R1	R2	R3	R4
l()	0	1	2	4
Permanent?	yes	yes	yes	no

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

First Value **Second Value = This + This**

In this pass we see that R4 still has a non-permanent label.

If we compare R4's current label value (4) with the sum of the value of the last label that was made permanent and the distance from R4 to the last node that was made permanent (2 + 5 = 7).

Since 4 is lower than 7, we do not change the value of the label for R4.

Since it is the lowest label value in this pass, we make it permanent.

FORE SYSTEMS

r=R4

Node	R1	R2	R3	R4
l()	0	1	2	4
Permanent?	yes	yes	yes	yes

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

The final step is to make the remaining label, for R4, permanent.

Node	R1	R2	R3	R4
l()	0	1	2	4
Permanent?	yes	yes	yes	yes

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

So this is the completed Dijkstra calculation *for R1* , with all nodes in the table having permanent labels.

Node	R1	R2	R3	R4
l()	0	1	2	4
Permanent?	yes	yes	yes	yes

Distance Matrix

$$\begin{bmatrix} \infty & 1 & 2 & \infty \\ 1 & \infty & \infty & 3 \\ 2 & \infty & \infty & 5 \\ \infty & 3 & 5 & \infty \end{bmatrix}$$

Using this table, R1 can see that the *shortest path* from itself to R2 is 1 unit.
From R1 to R3 is 2 units...
...to R4 is 4 units.

FORE SYSTEMS

The End

This concludes the tutorial.

If you aren't viewing this on the FORE Systems Web Site, then you can become a member of the ATM Academy free of charge and have access to many more of these tutorials.

You can find details at:

http://academy.fore.com/