Shortest Path First

Dijkstra's Famous Algorithm

“The question of whether computers can think is like the question of whether submarines can swim”

Edsger Wybe Dijkstra

Dijkstra's SP Algorithm

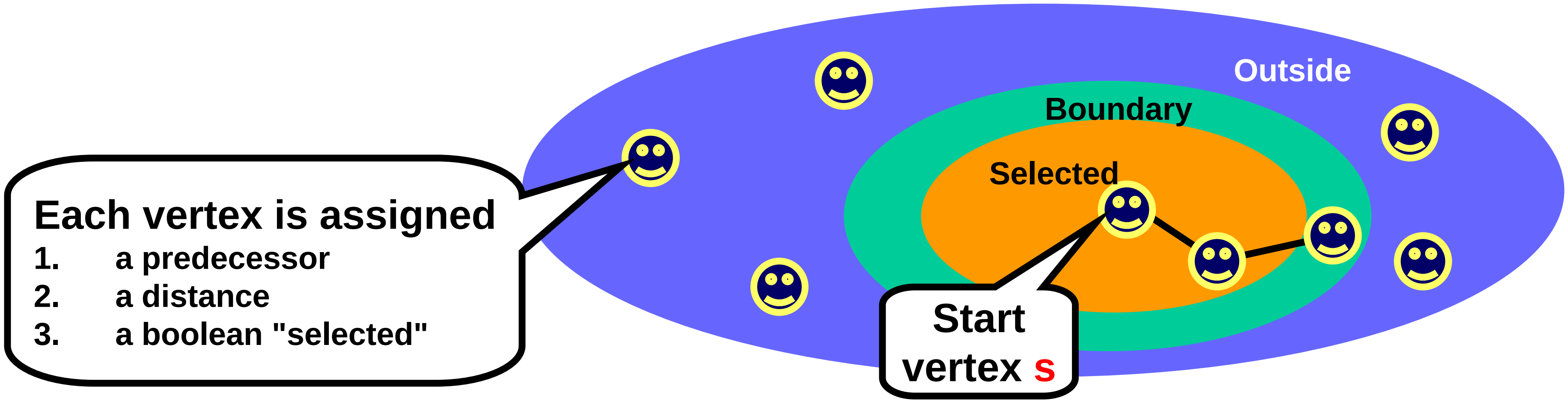
- Famous paper "A note on two problems in connection with graphs" (1959)
- Single source SP problem in a directed graph
- Important applications include
  - Network routing protocols (OSPF, IS-IS)
  - Traveller‘s route planner

Terms

- Graph G(V,E) consists of vertices V and edges E
- Edges are assigned costs c
- "Length" of graph c(G) = sum of all costs
  - Assumed to be positive ("Distance Graph")
- "Distance" between two vertices d(v,v') = min{c(p)}, p…path
  - Can be infinite
- p with c(p) = d(v,v') is called shortest path sp(v,v')

Definitions

- Select start vertex s
- Three sets of vertices:
  - Selected (sp already calculated)
  - Boundary (currently subject of calculation)
  - Outside (not yet examined)



The Algorithm

Initialize Vertices
v.predecessor = none
v.distance = ∞
v.selected = false

Select S
s.predecessor = s
s.distance = 0
s.selected = true

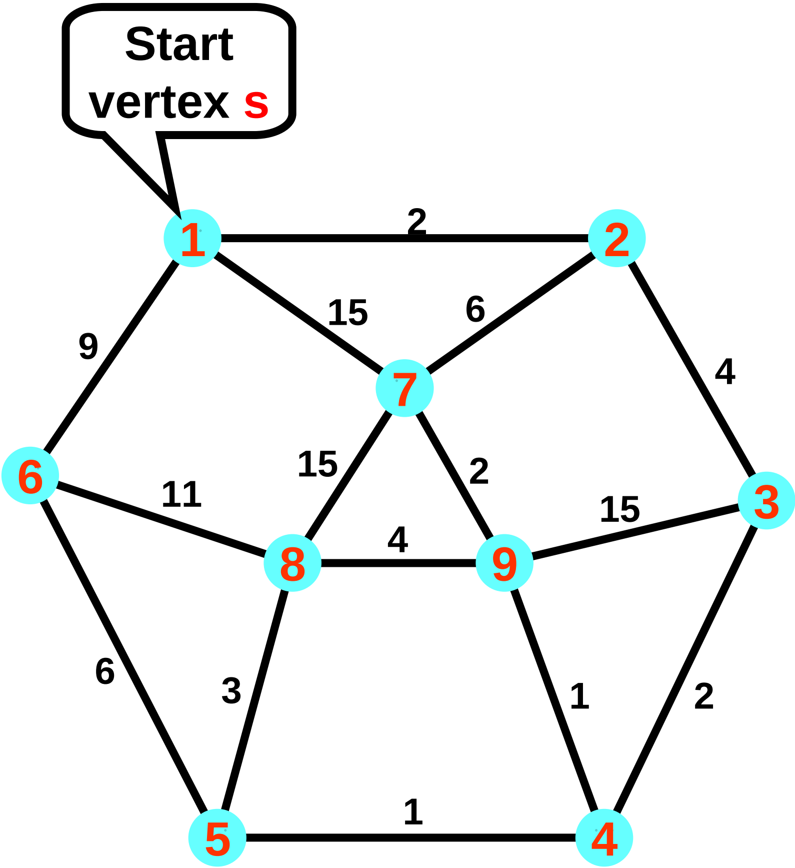
Add neighbors of S to boundary

Select V with lowest distance from boundary

Add neighbors of V to boundary

For these neighbors calculate distance using V as predecessor
Previous vertices might get better total distance

Example



vertex number | distance | predecessor

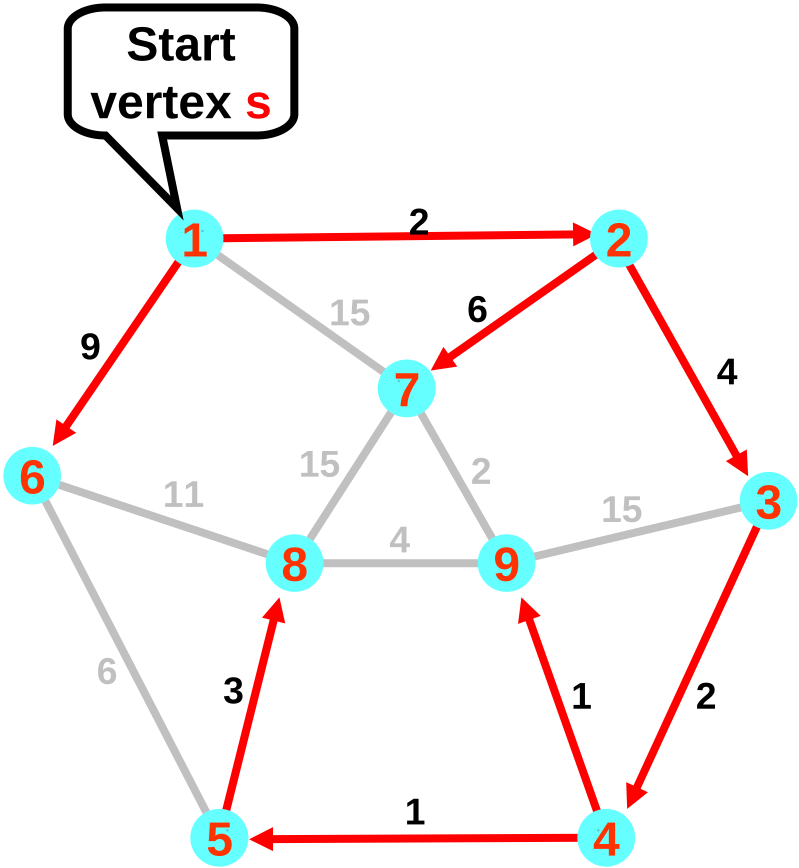
Selected

vertex number	distance	predecessor
1	0	1
2	2	1
3	6	2
7	8	2
4	8	3
6	9	1
9	9	4
5	9	4
8	12	5

Boundary

2	2	1	6	9	1	7	15	1			
6	9	1	7	8	2	3	6	2			
6	9	1	7	8	2	4	8	3	9	21	3
6	9	1	4	8	3	9	10	7	8	23	7
6	9	1	8	23	7	9	9	4	5	9	4
9	9	4	5	9	4	8	20	6			
5	9	4	8	13	9						
8	12	5									

Result



Selected		
1	0	1
2	2	1
3	6	2
7	8	2
4	8	3
6	9	1
9	9	4
5	9	4
8	12	5

- **Single source SP**
- **Minimal length**
- **Complete**

Performance

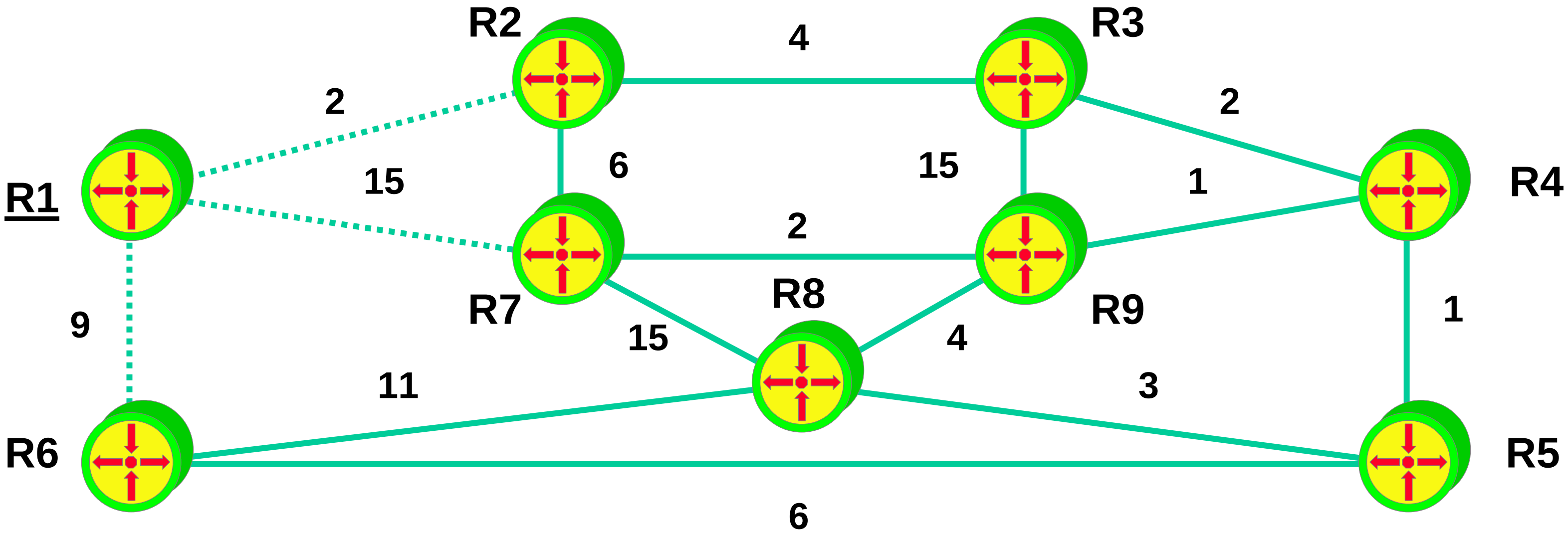
- Greedy algorithm
- Most critical: Implementation of boundary data structure
  - No explicit structure: $O(|V|^2)$
  - Fibonacci heap: $O(|E|+|V| \log |V|)$
- Alternatives
  - Bellman-Ford (RIP) algorithm
  - Floyd-Warshall algorithm
  - A* algorithm
    - Extends SPF with a estimation function to enhance performance in certain situations

About E. W. Dijkstra

- Born in 1930 in Rotterdam
- Degrees in mathematics and theoretical physics from the University of Leyden and a Ph.D. in computing science from the University of Amsterdam
  - Programmer at the Mathematisch Centrum, Amsterdam, 1952-62
  - Professor of mathematics, Eindhoven University of Technology, 1962-1984
  - Burroughs Corporation research fellow, 1973-1984
  - Schlumberger Centennial Chair in Computing Sciences at the University of Texas at Austin, 1984-1999
  - Retired as Professor Emeritus in 1999
  - 1972 recipient of the ACM Turing Award, often viewed as the Nobel Prize for computing
- Died 6 August 2002

Edsger W. Dijkstra
(1930-2002)

Select root (R1)



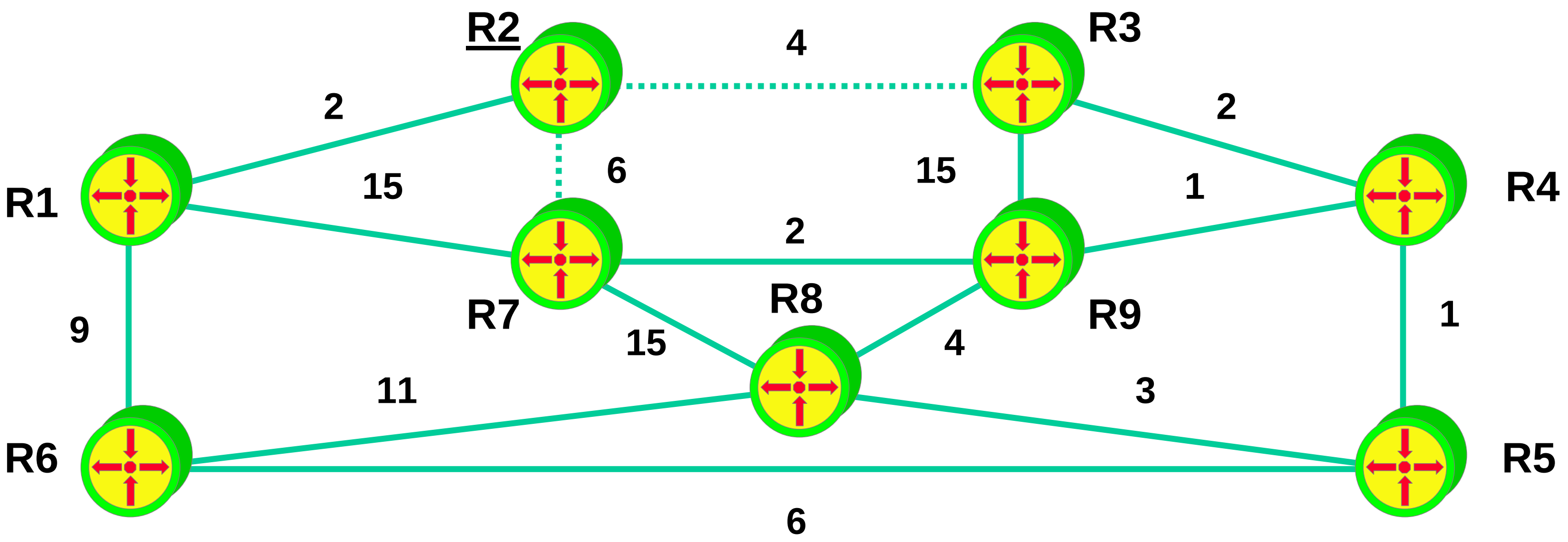
Selected

R1	0	R1

Boundary

R2	2	R1	R6	9	R1	R7	15	R1

Select router with lowest cost in boundary (R2), calculate cost for neighbours R3, R7



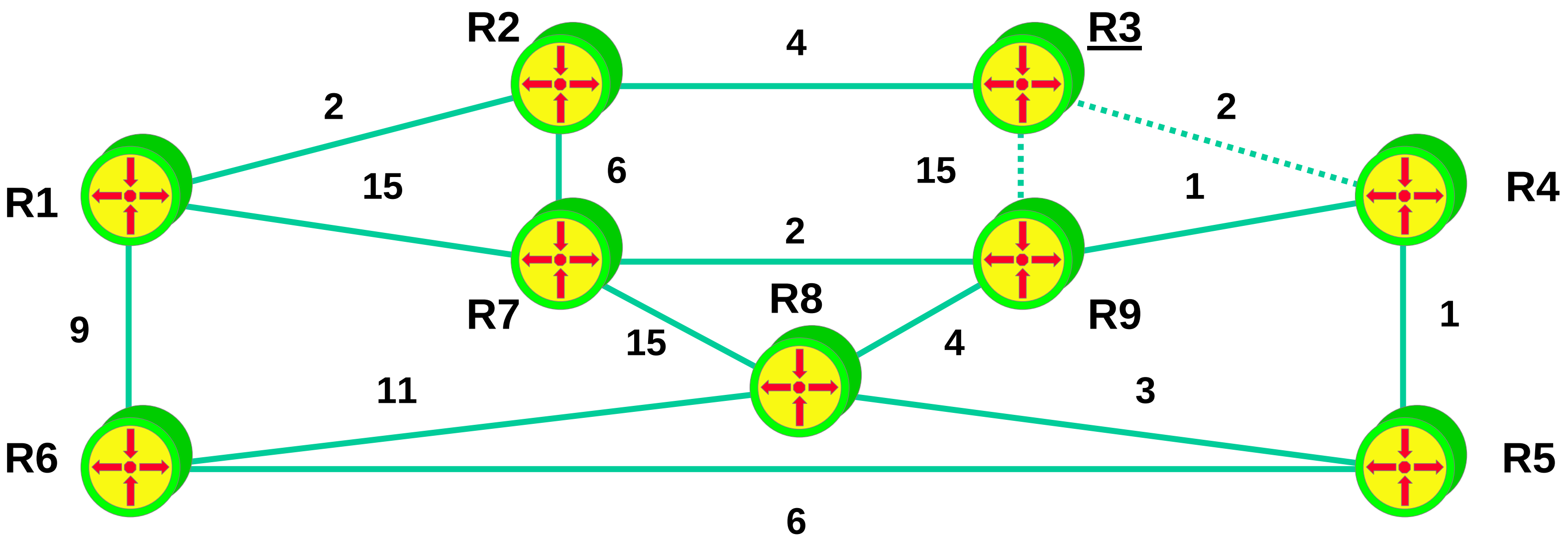
Selected

R1	0	R1
R2	2	R1

Boundary

R2	2	R1	R6	9	R1	R7	15	R1
R6	9	R1	R7	8	R2	R3	6	R2

Select router with lowest cost in boundary (R3), calculate cost for neighbours R9, R4



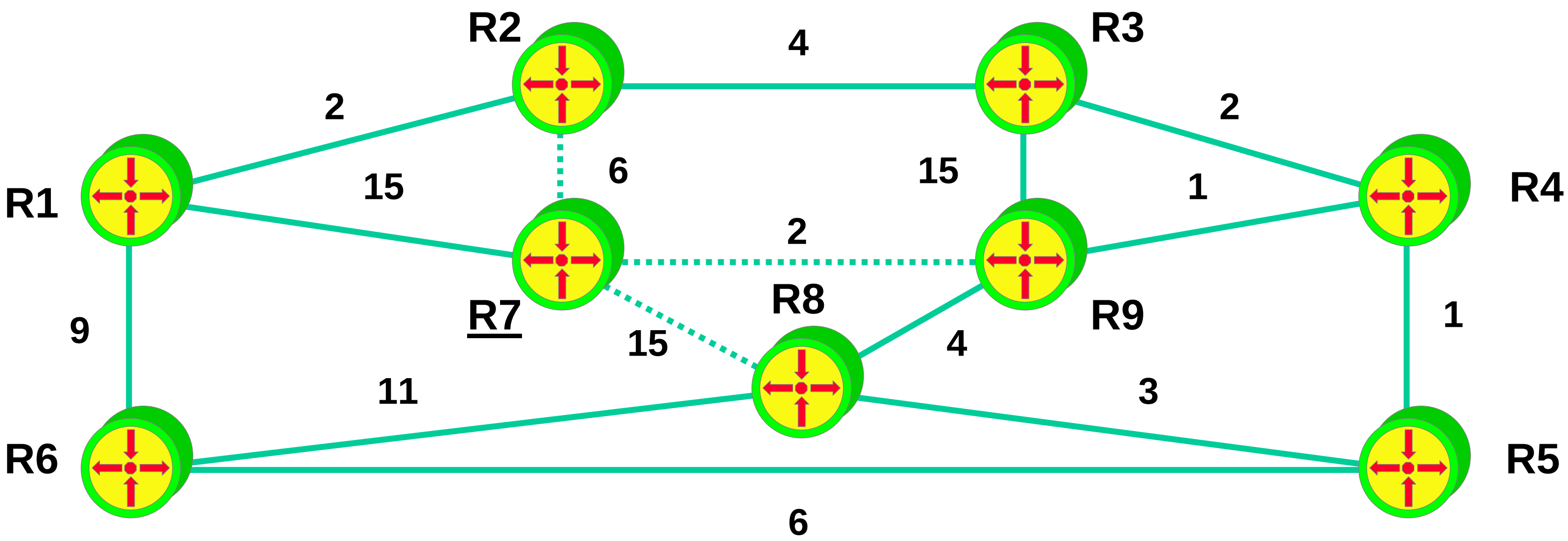
Selected

R1	0	R1
R2	2	R1
R3	6	R2

Boundary

R2	2	R1	R6	9	R1	R7	15	R1			
R6	9	R1	R7	8	R2	R3	6	R2			
R6	9	R1	R7	8	R2	R9	21	R3	R4	8	R3

Select one router with lowest cost in boundary (R7), calculate cost for neighbours R8, R9



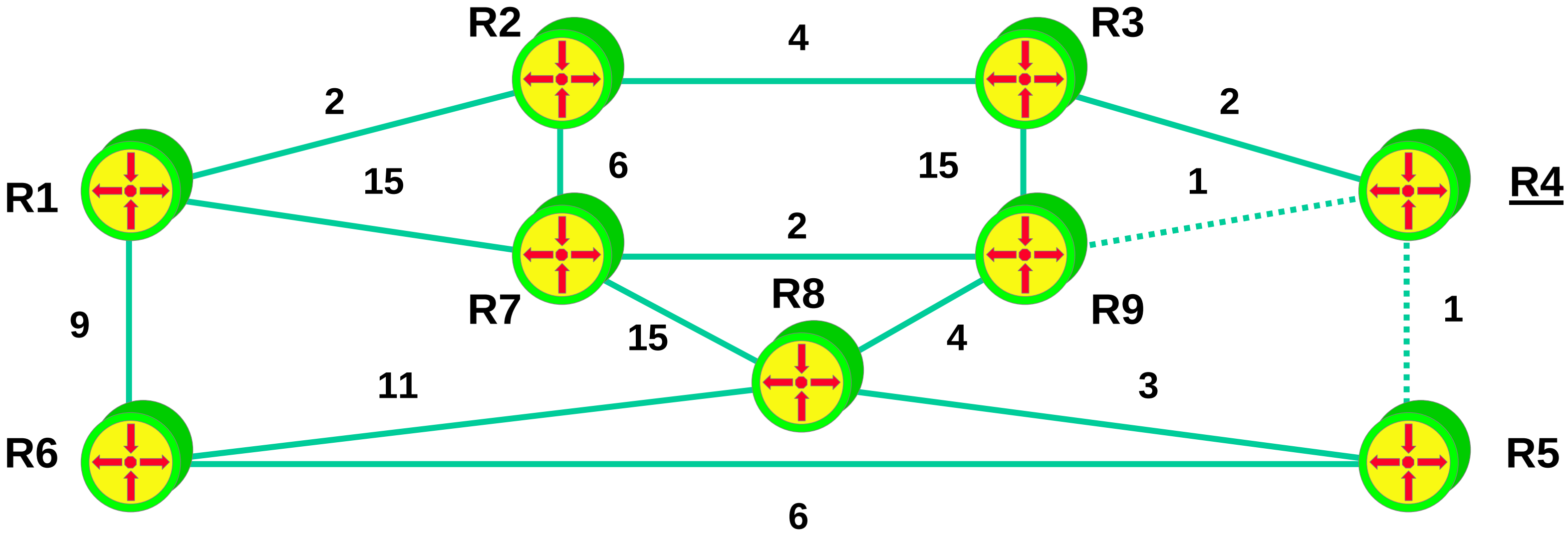
Selected

R1	0	R1
R2	2	R1
R3	6	R2
R7	8	R2

Boundary

R2	2	R1	R6	9	R1	R7	15	R1			
R6	9	R1	R7	8	R2	R3	6	R2			
R6	9	R1	R7	8	R2	R9	21	R3	R4	8	R3
R6	9	R1	R4	8	R3	R9	10	R7	R8	23	R7

Select router with lowest cost in boundary (R4), calculate cost for neighbours R9, R5



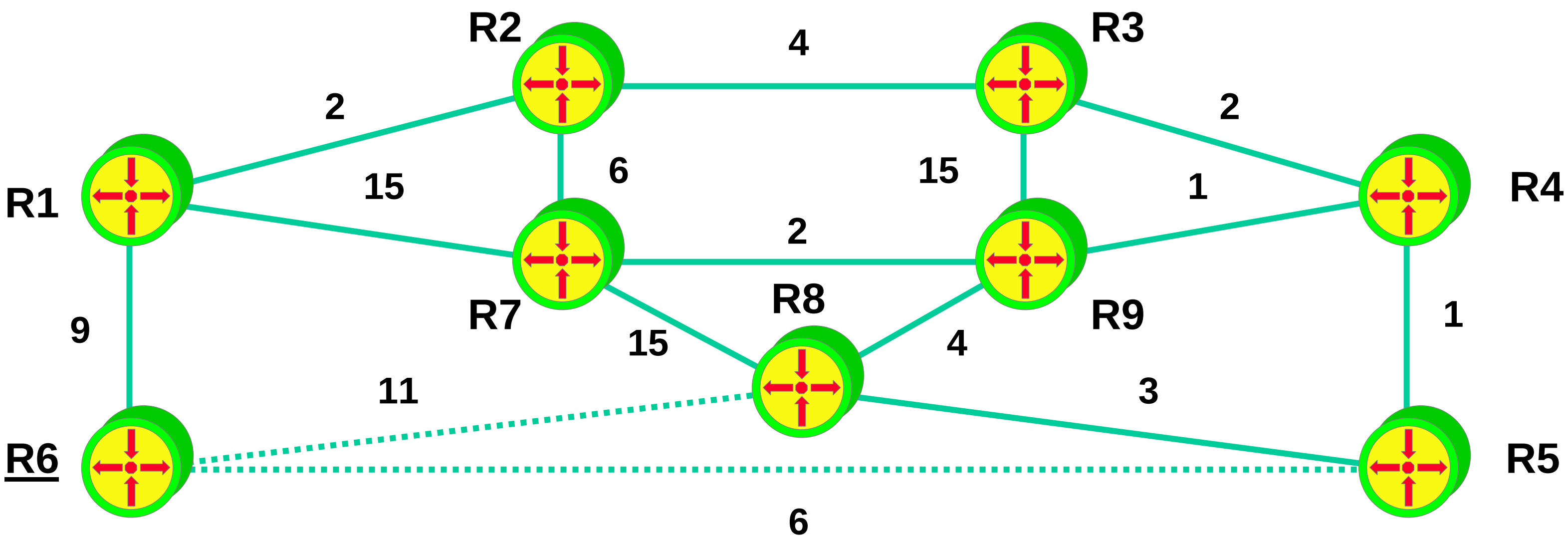
Selected

R1	0	R1
R2	2	R1
R3	6	R2
R7	8	R2
R4	8	R3

Boundary

R2	2	R1	R6	9	R1	R7	15	R1			
R6	9	R1	R7	8	R2	R3	6	R2			
R6	9	R1	R7	8	R2	R9	21	R3	R4	8	R3
R6	9	R1	R4	8	R3	R9	10	R7	R8	23	R7
R6	9	R1	R8	23	R7	R9	9	R4	R5	9	R4

Select one router with lowest cost in boundary (R6), calculate cost for neighbours R5 and R8



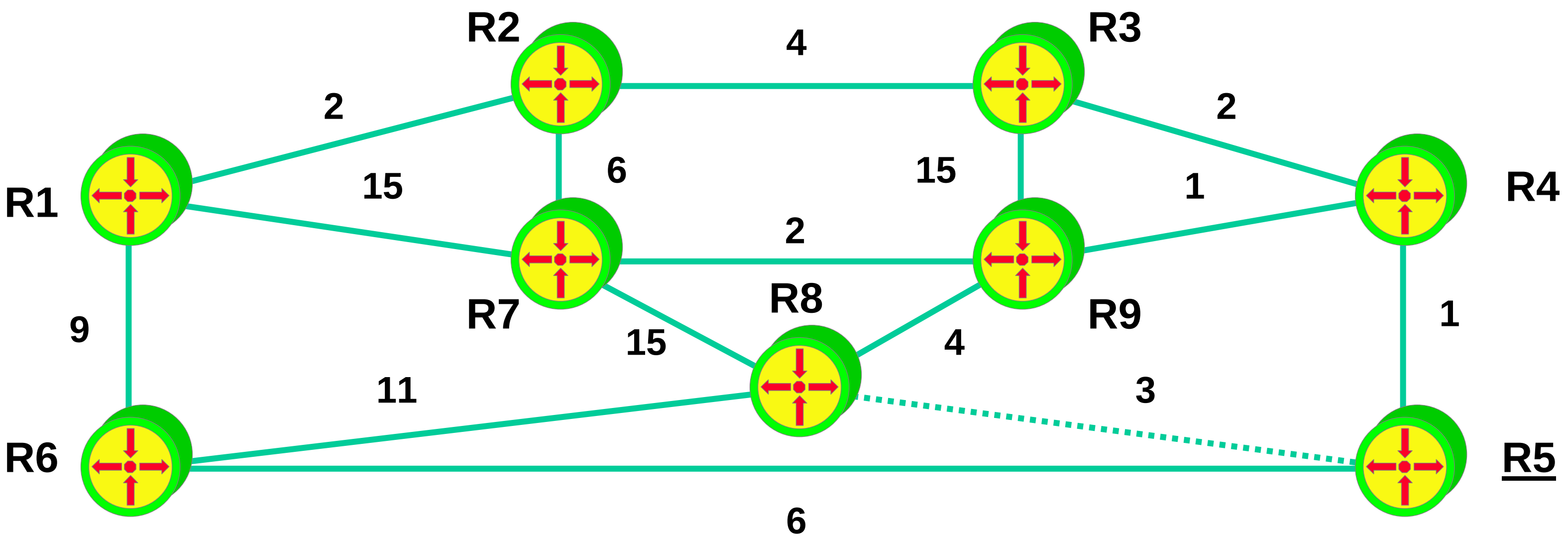
Selected

R1	0	R1
R2	2	R1
R3	6	R2
R7	8	R2
R4	8	R3
R6	9	R1

Boundary

R2	2	R1	R6	9	R1	R7	15	R1			
R6	9	R1	R7	8	R2	R3	6	R2			
R6	9	R1	R7	8	R2	R9	21	R3	R4	8	R3
R6	9	R1	R4	8	R3	R9	10	R7	R8	23	R7
R6	9	R1	R8	23	R7	R9	9	R4	R5	9	R4
R9	9	R4	R8	20	R6	R5	9	R4			

Select one neighbour with lowest cost in boundary (R5), calculate cost for neighbour R8



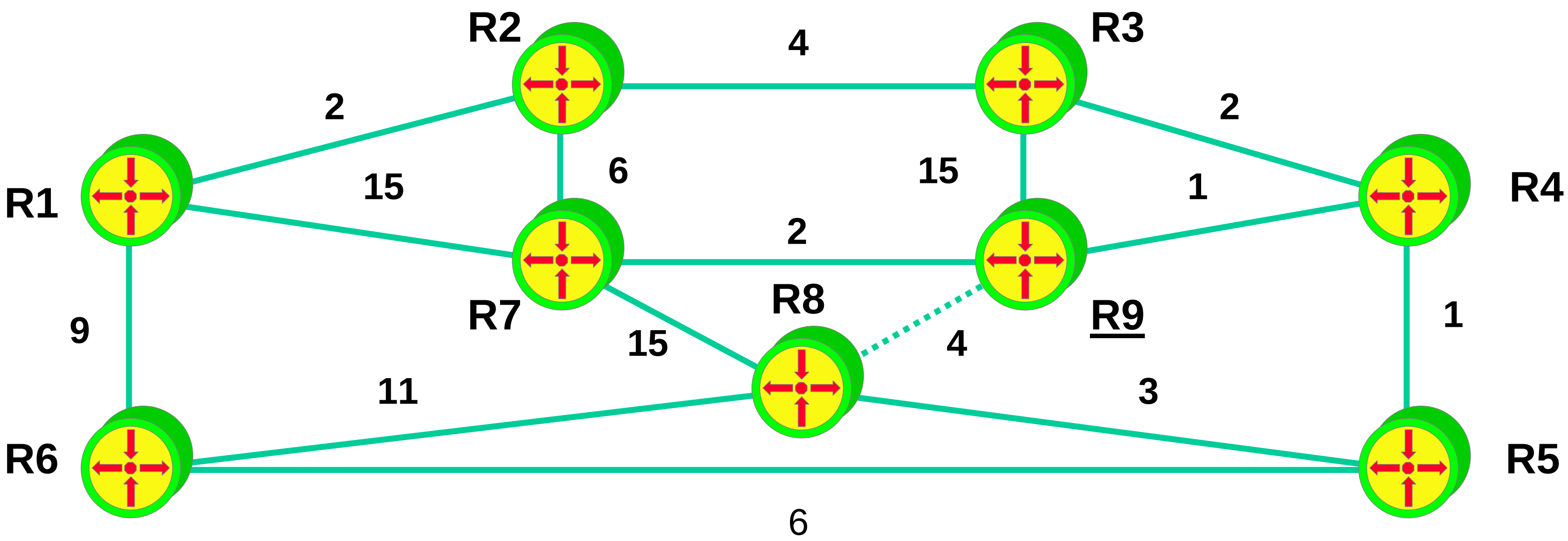
Selected

R1	0	R1
R2	2	R1
R3	6	R2
R7	8	R2
R4	8	R3
R6	9	R1
R5	9	R4

Boundary

R2	2	R1	R6	9	R1	R7	15	R1			
R6	9	R1	R7	8	R2	R3	6	R2			
R6	9	R1	R7	8	R2	R9	21	R3	R4	8	R3
R6	9	R1	R4	8	R3	R9	10	R7	R8	23	R7
R6	9	R1	R8	23	R7	R9	9	R4	R5	9	R4
R9	9	R4	R8	20	R6	R5	9	R4			
R9	9	R4	R8	12	R5						

Select router with lowest cost in boundary (R9), calculate cost for neighbours R8



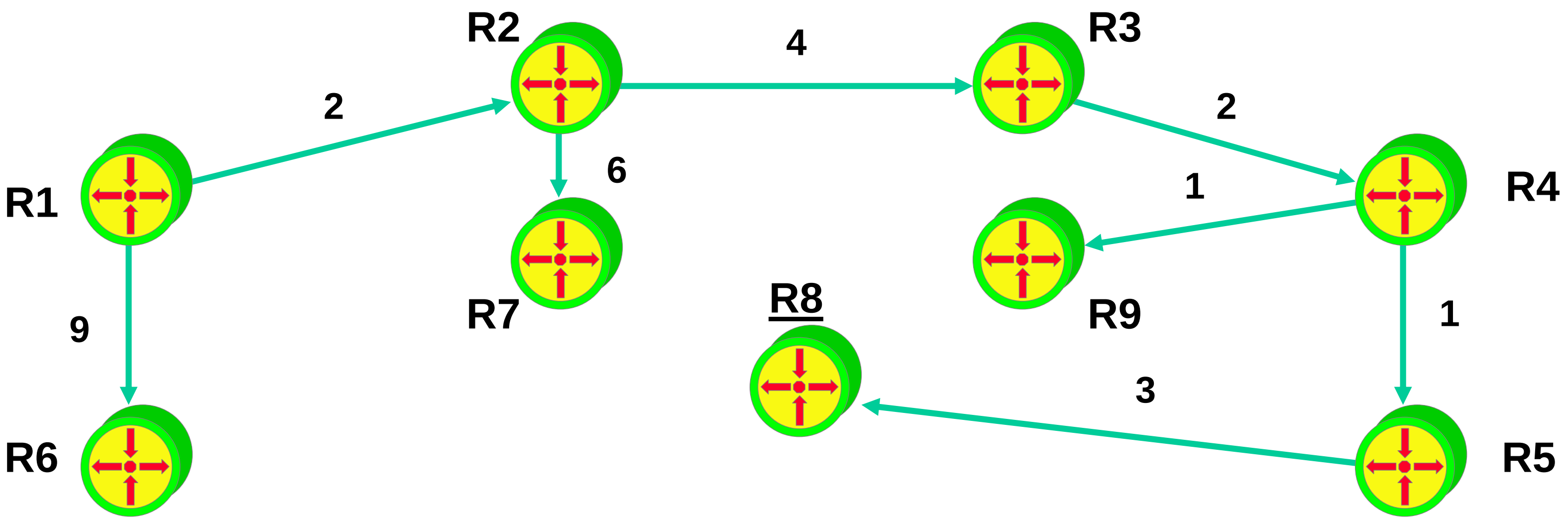
Selected

R1	0	R1
R2	2	R1
R3	6	R2
R7	8	R2
R4	8	R3
R6	9	R1
R5	9	R4
R9	9	R4

Boundary

R2	2	R1	R6	9	R1	R7	15	R1			
R6	9	R1	R7	8	R2	R3	6	R2			
R6	9	R1	R7	8	R2	R9	21	R3	R4	8	R3
R6	9	R1	R4	8	R3	R9	10	R7	R8	23	R7
R6	9	R1	R8	23	R7	R9	9	R4	R5	9	R4
R9	9	R4	R8	20	R6	R5	9	R4			
R9	9	R4	R8	12	R5						
R8	12	R5									

Select last router in boundary (R8), algorithm terminated, all shortest paths found



Selected

R1	0	R1
R2	2	R1
R3	6	R2
R7	8	R2
R4	8	R3
R6	9	R1
R5	9	R4
R9	9	R4
R8	12	R5

Boundary

R2	2	R1	R6	9	R1	R7	15	R1			
R6	9	R1	R7	8	R2	R3	6	R2			
R6	9	R1	R7	8	R2	R9	21	R3	R4	8	R3
R6	9	R1	R4	8	R3	R9	10	R7	R8	23	R7
R6	9	R1	R8	23	R7	R9	9	R4	R5	9	R4
R9	9	R4	R8	20	R6	R5	9	R4			
R9	9	R4	R8	12	R5						
R8	12	R5									