

Database Seminar

Scalable Network Distance Browsing in Spatial Databases

Markus Cadonau

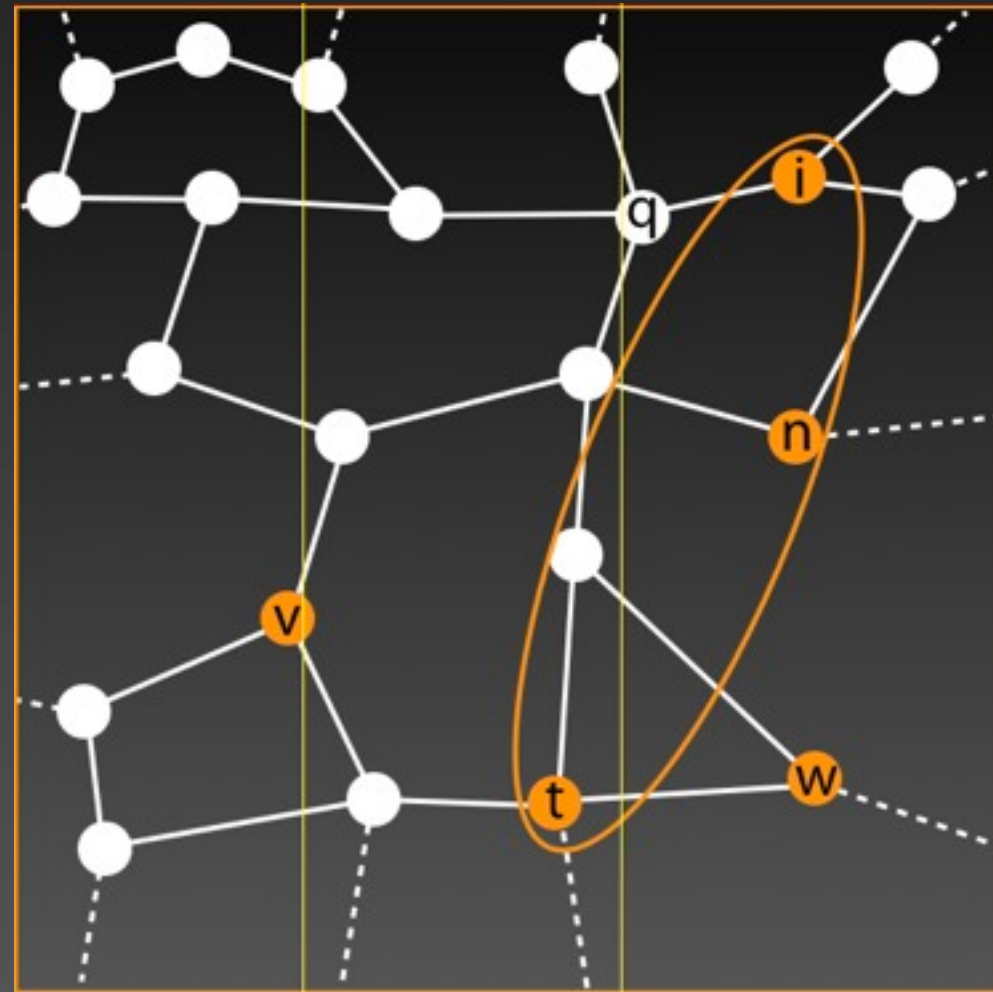
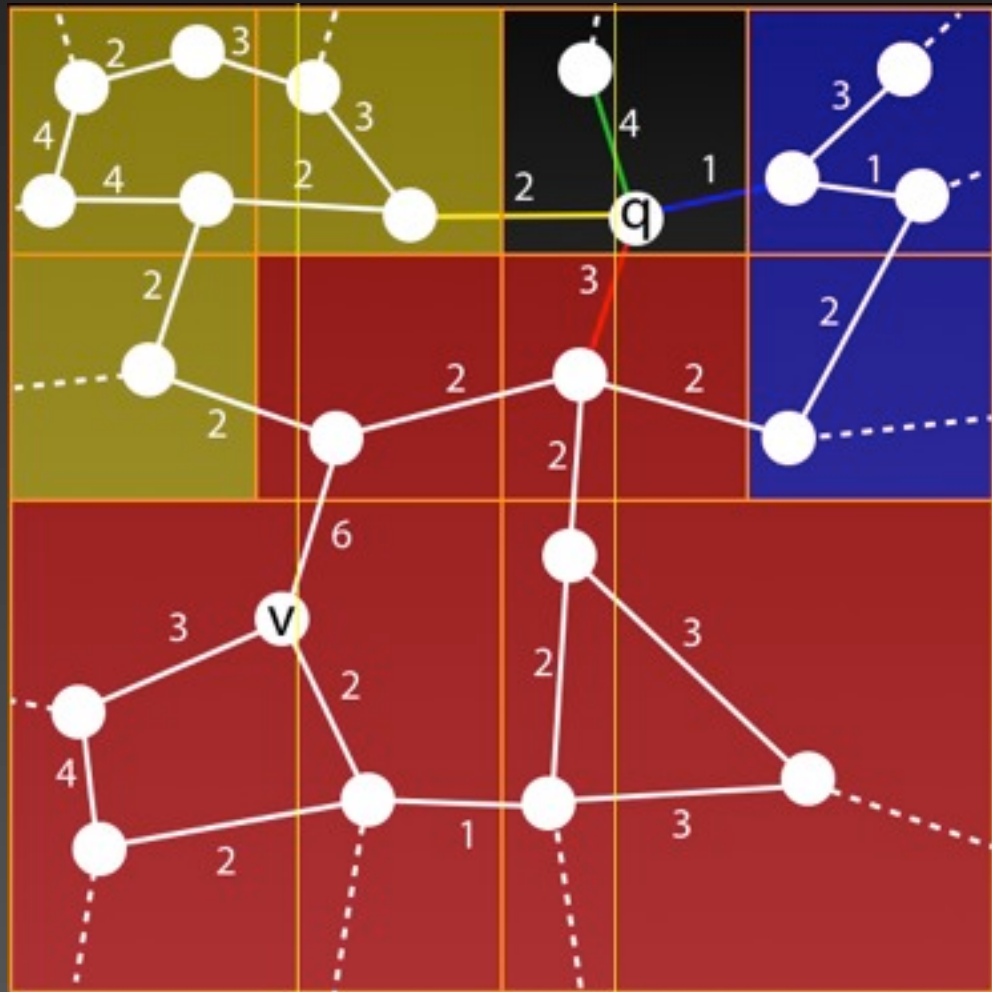
8 May 2010

How can spatial coherence be used for efficient nearest neighbor search in static road networks?

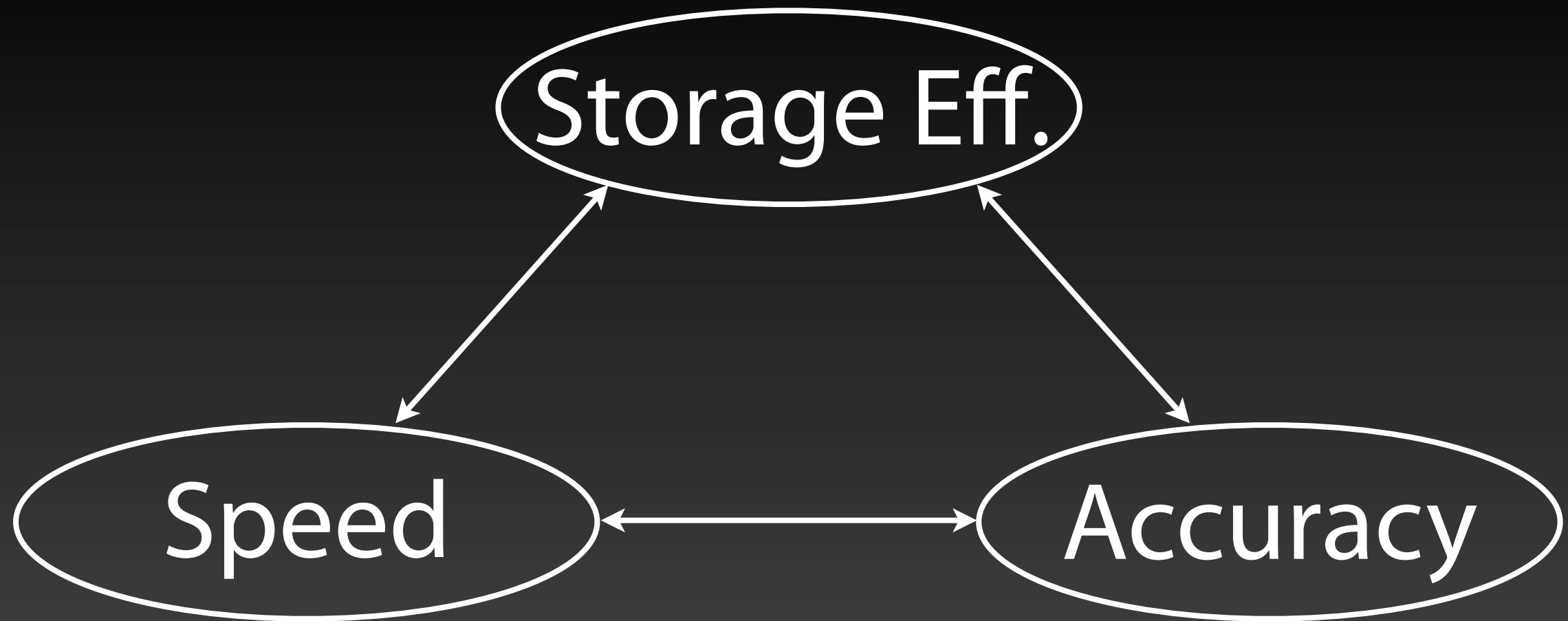
Shortest Path Quadrees

+

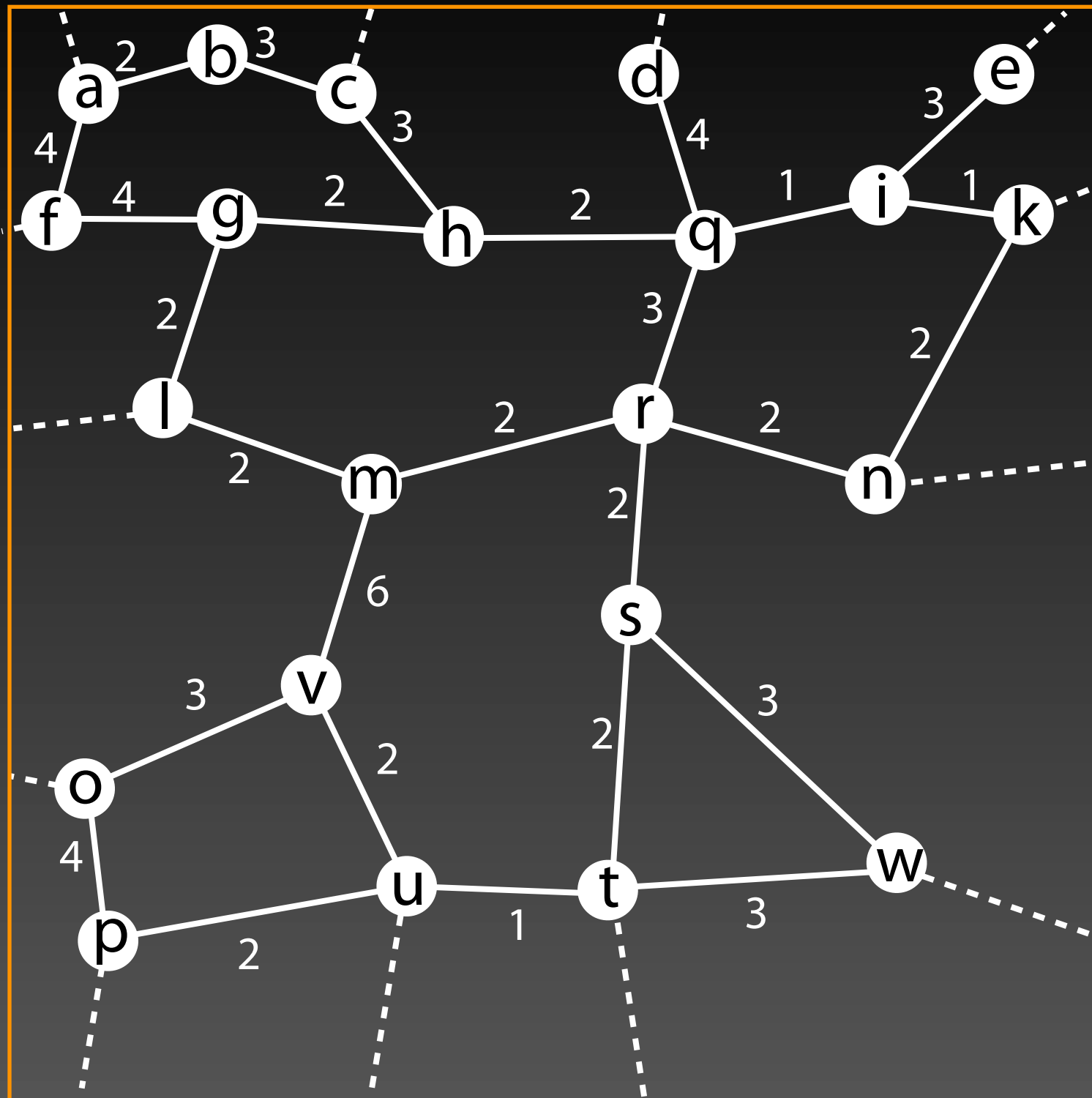
Best-First k-Nearest Neighbor



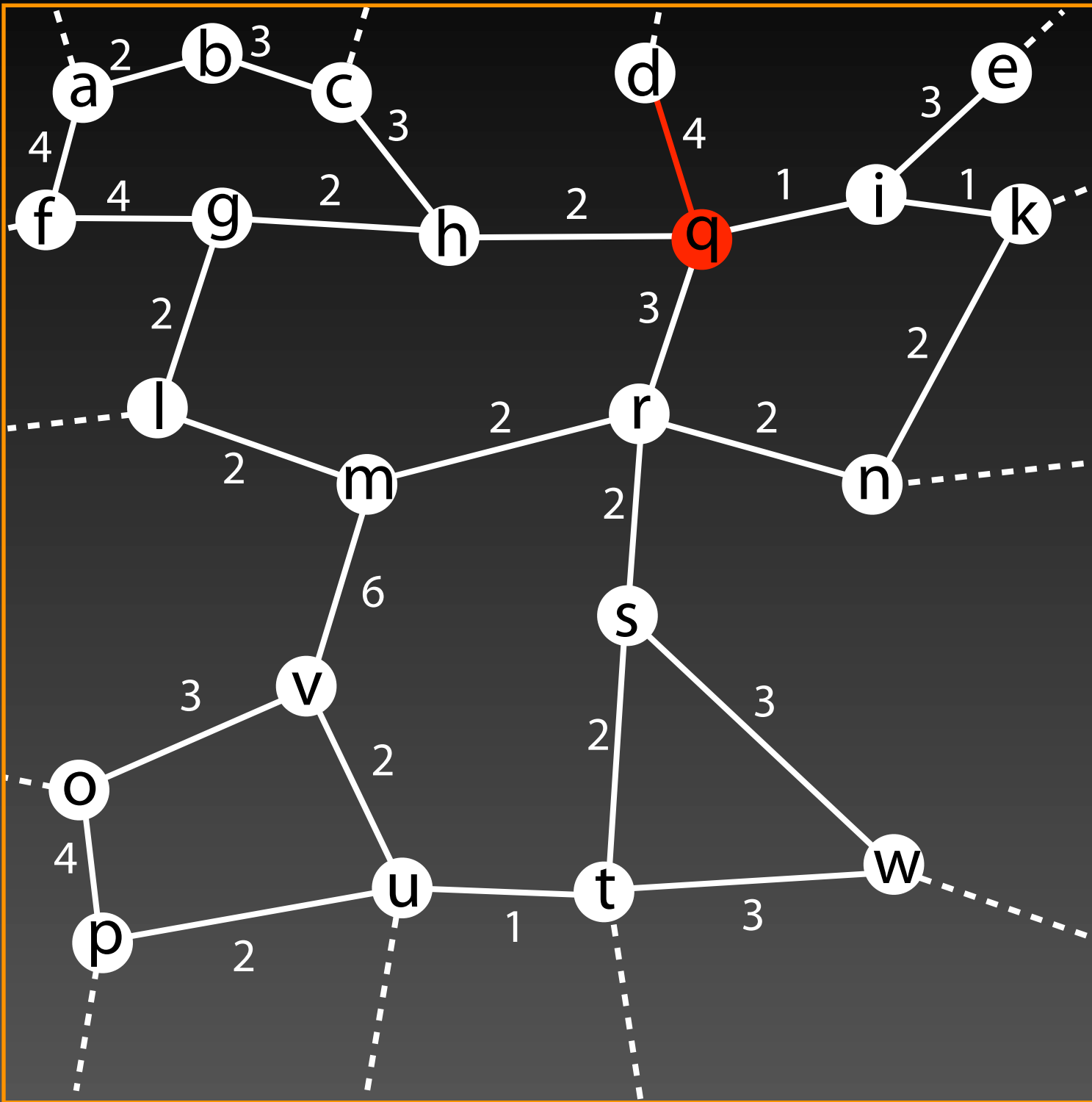
Trade-Offs



Road Network



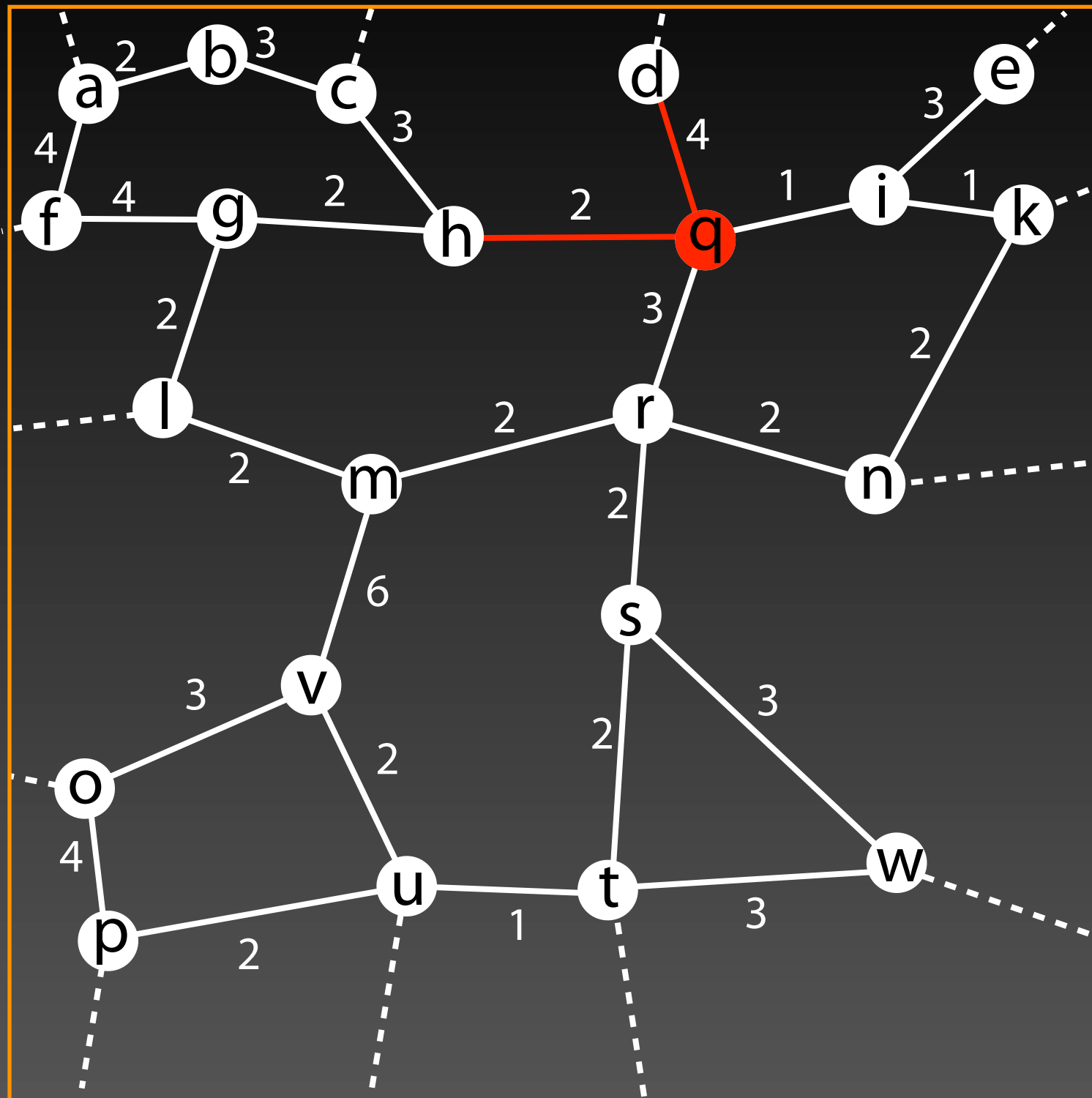
Shortest Paths / kNN (Dijkstra)



Queue:

(d,4)

Shortest Paths / kNN (Dijkstra)

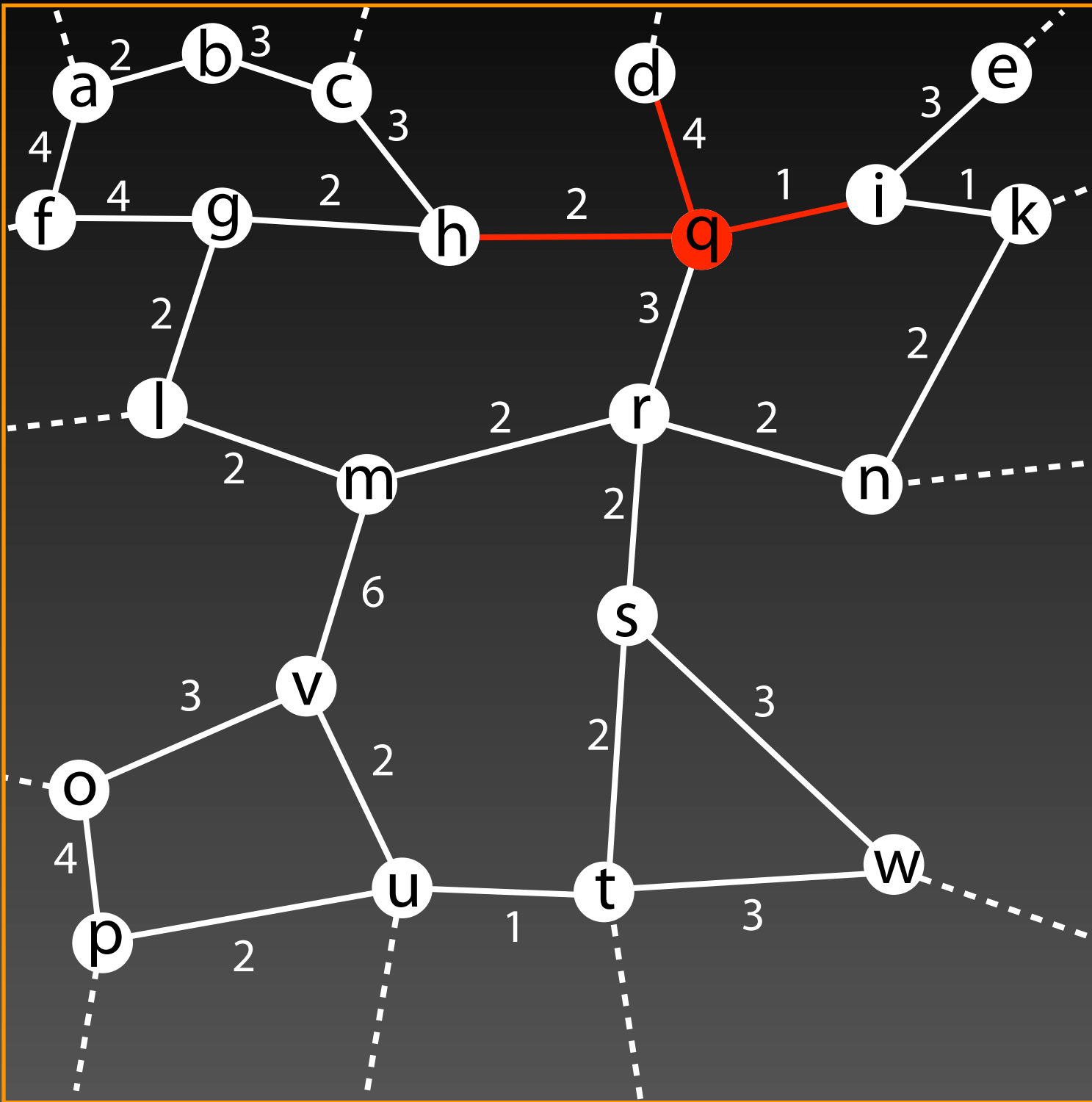


Queue:

(h,2)

(d,4)

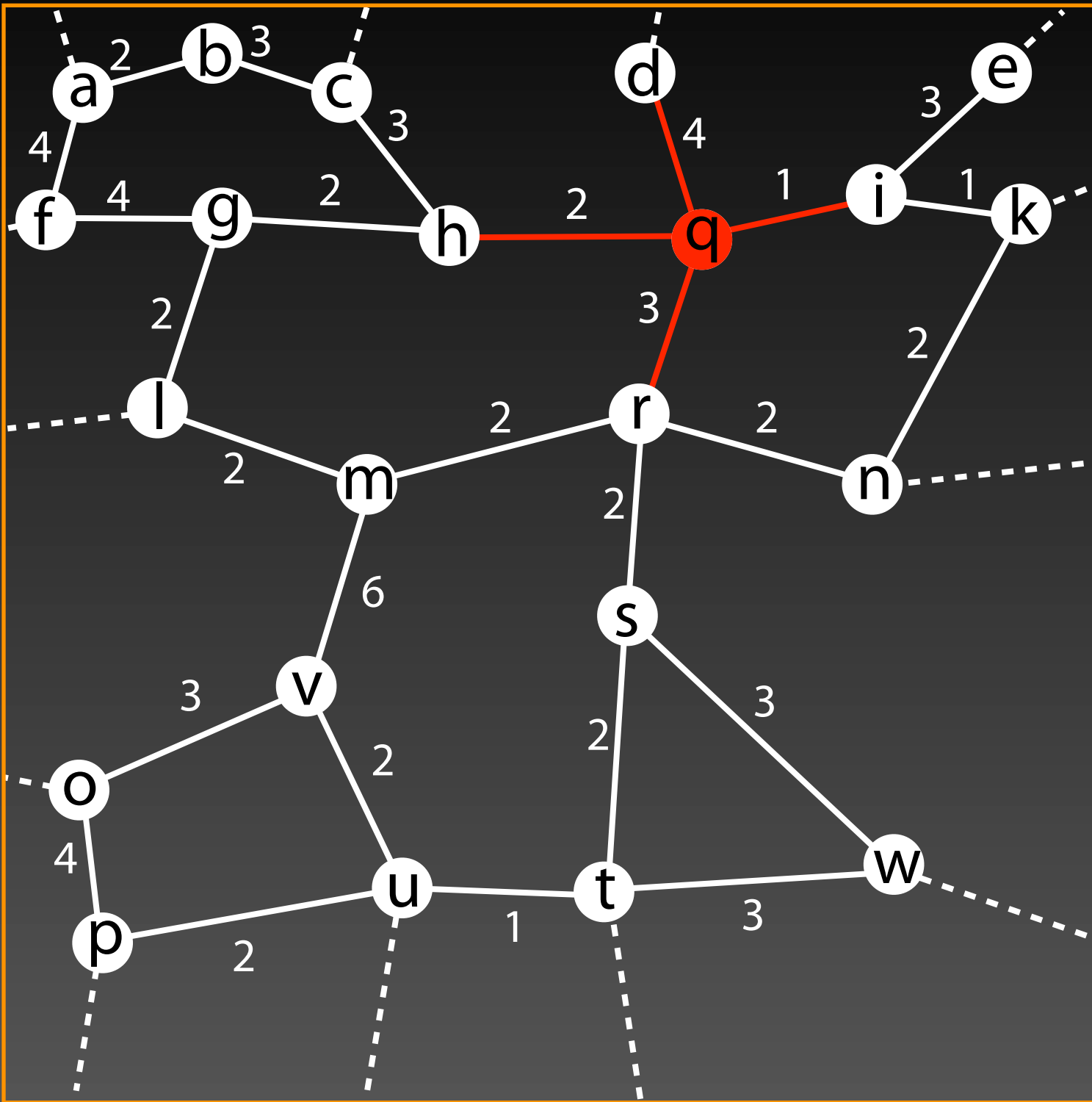
Shortest Paths / kNN (Dijkstra)



Queue:

(i,1)
(h,2)
(d,4)

Shortest Paths / kNN (Dijkstra)



Queue:

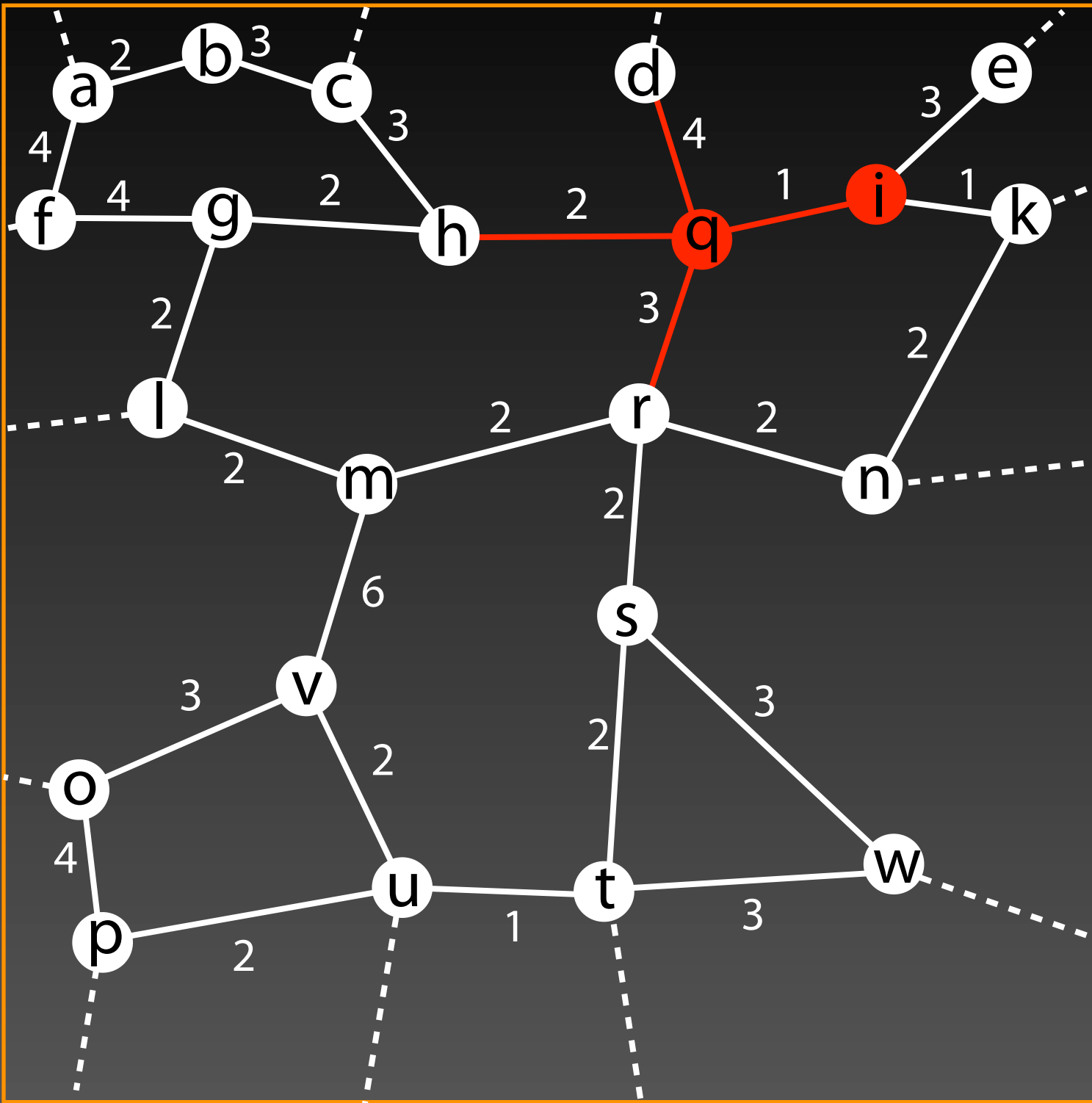
(i,1)

(h,2)

 $(r, 3)$

(d,4)

Shortest Paths / kNN (Dijkstra)



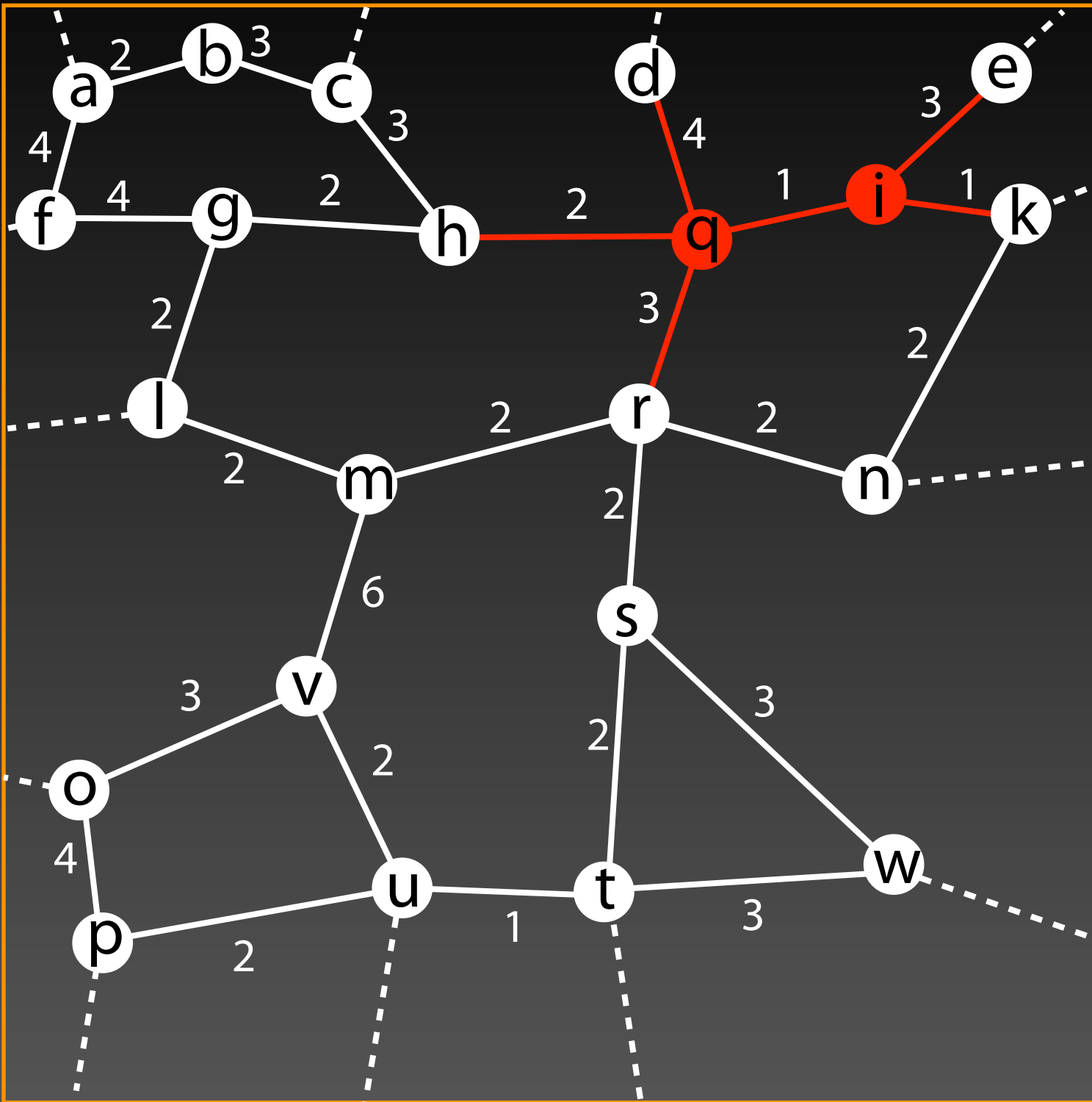
Queue:

(h,2)

 $(r, 3)$

(d,4)

Shortest Paths / kNN (Dijkstra)



Queue:

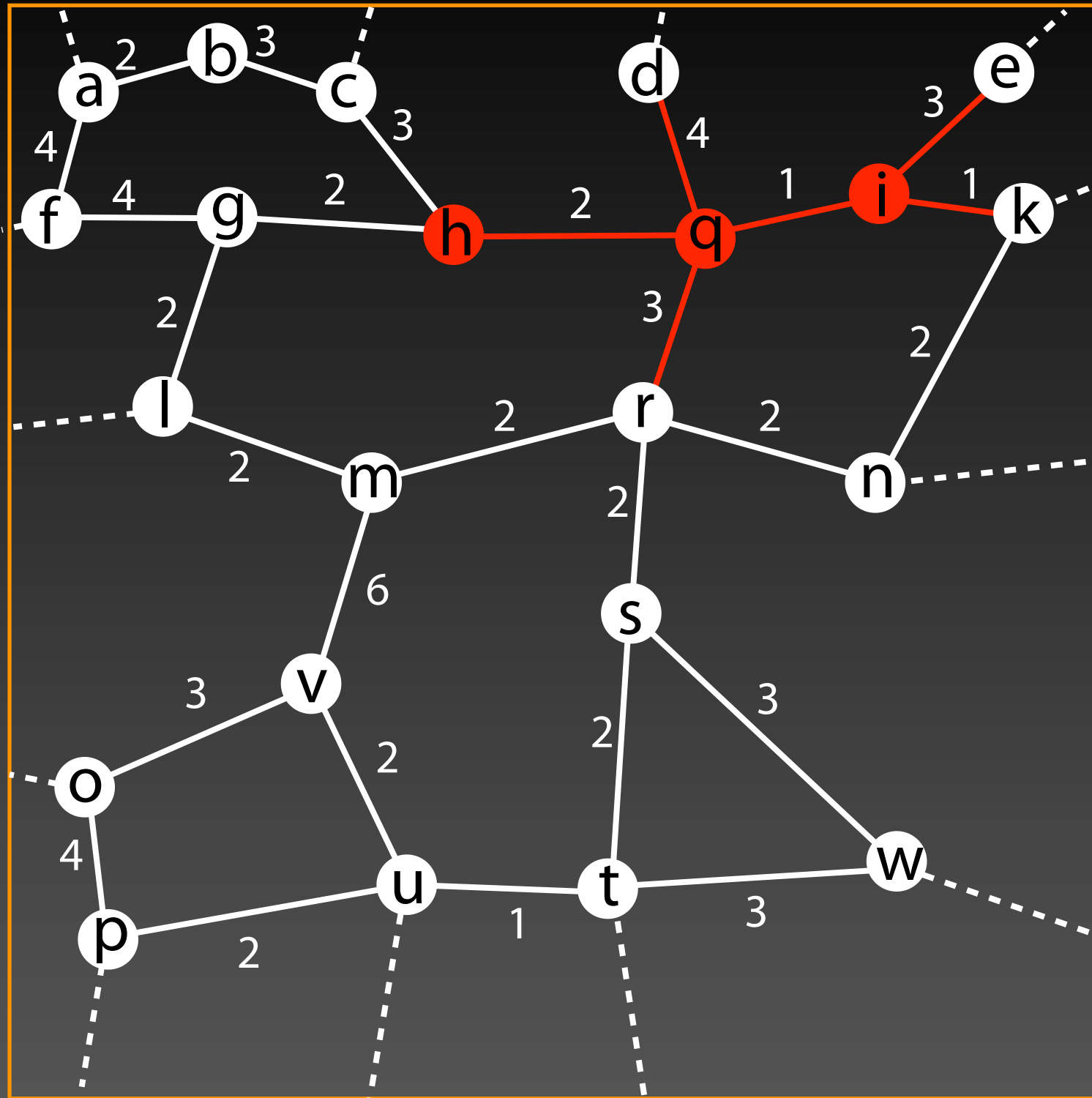
(h,2)

 $(k, 2)$ $(r, 3)$

(d,4)

(e,4)

Shortest Paths / kNN (Dijkstra)



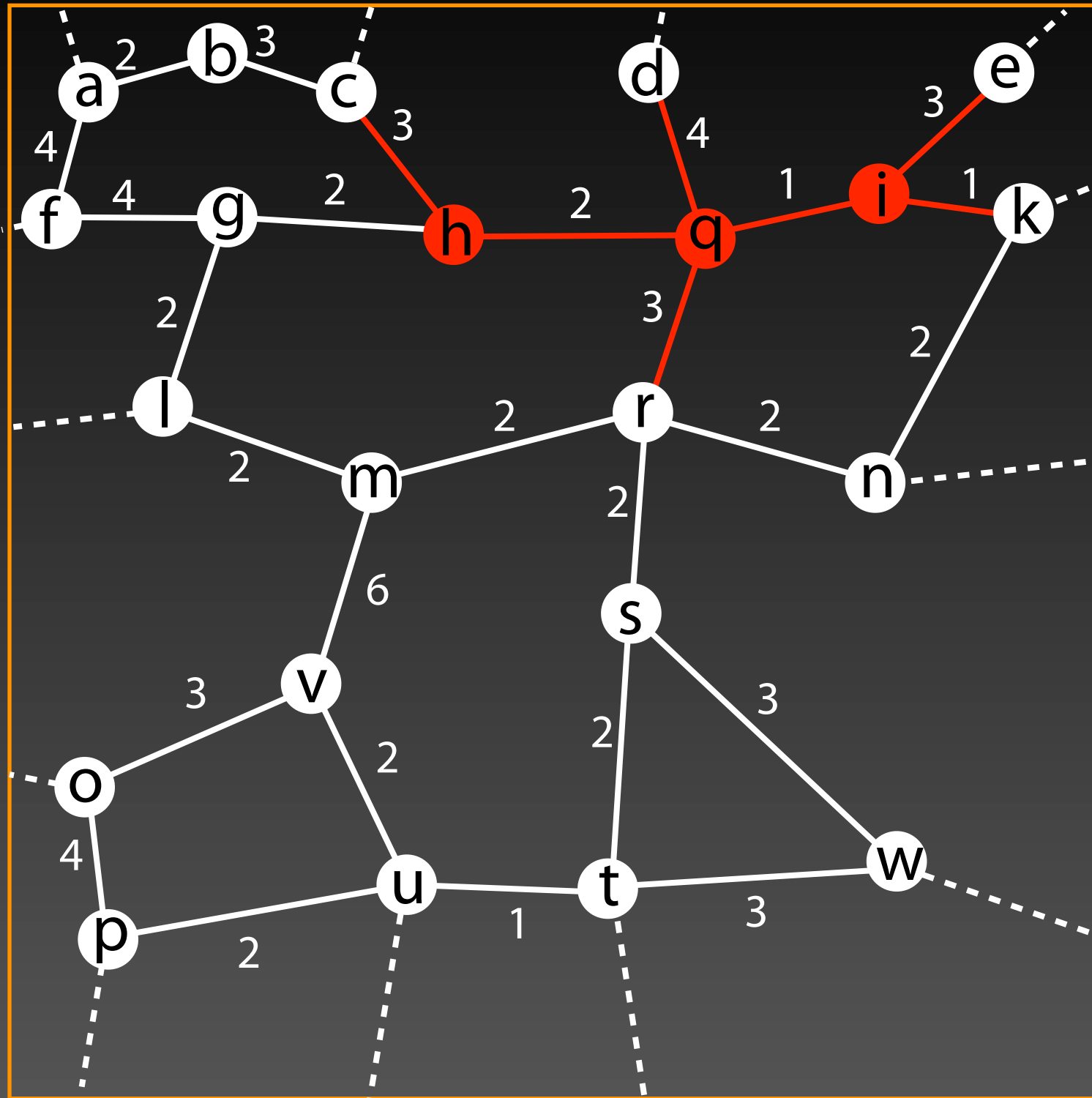
Queue:

 $(k, 2)$ $(r, 3)$

(d,4)

(e,4)

Shortest Paths / kNN (Dijkstra)



Queue:

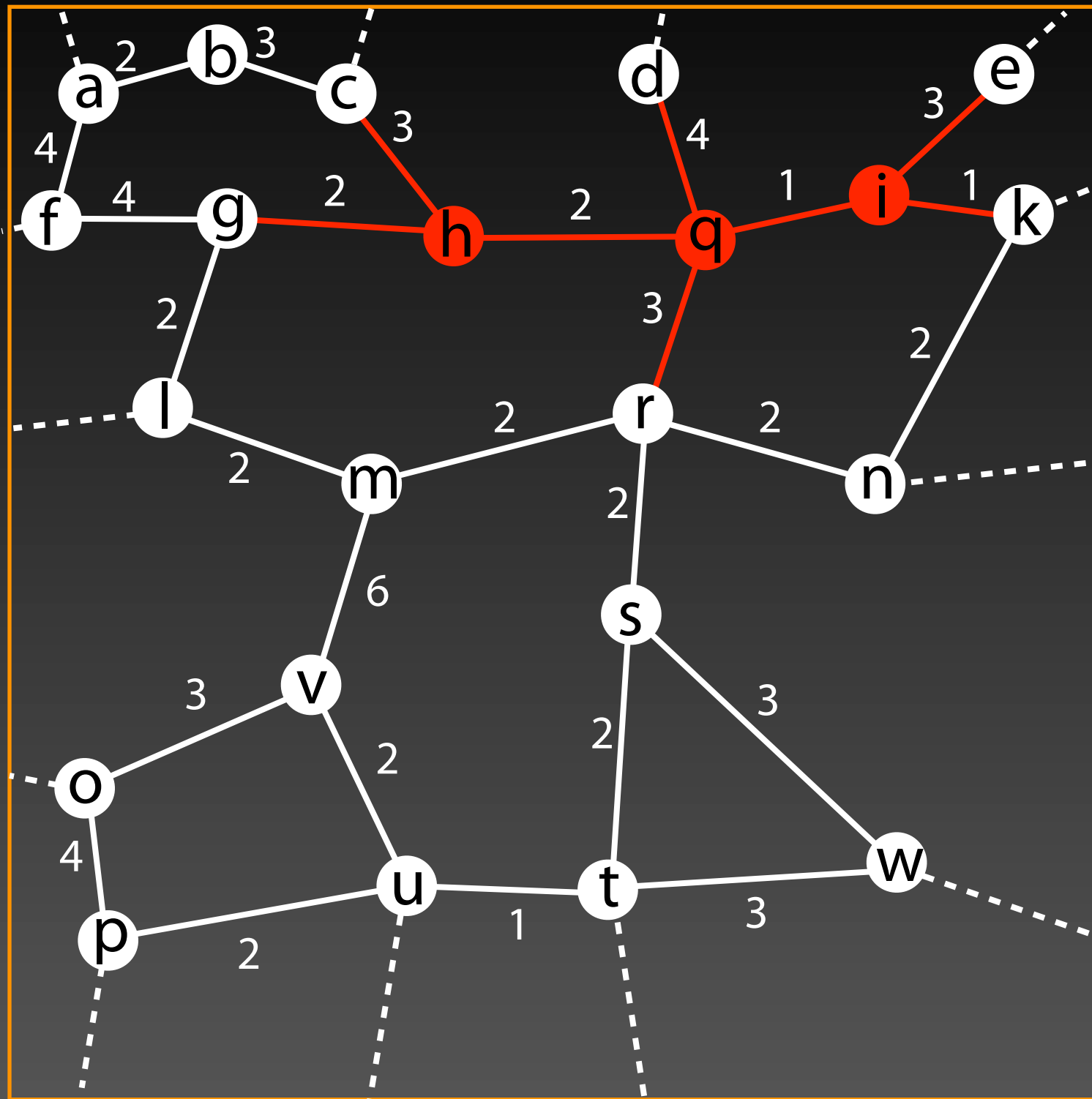
 $(k, 2)$ $(r, 3)$

(d,4)

(e,4)

$(c, 5)$

Shortest Paths / kNN (Dijkstra)



Queue:

 $(k, 2)$ $(r, 3)$

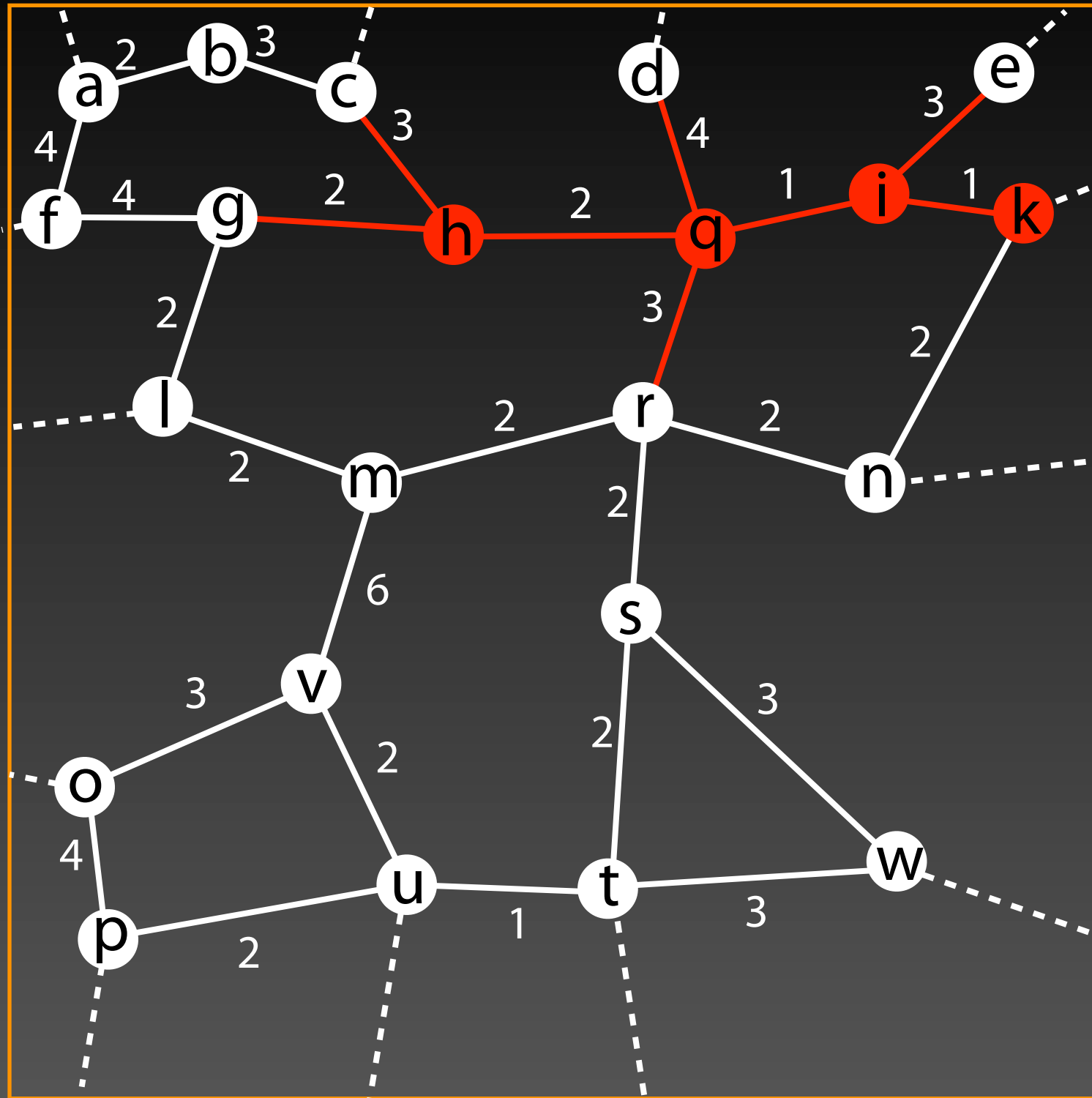
(d,4)

(e,4)

(g,4)

(c,5)

Shortest Paths / kNN (Dijkstra)



Queue:

 $(r, 3)$

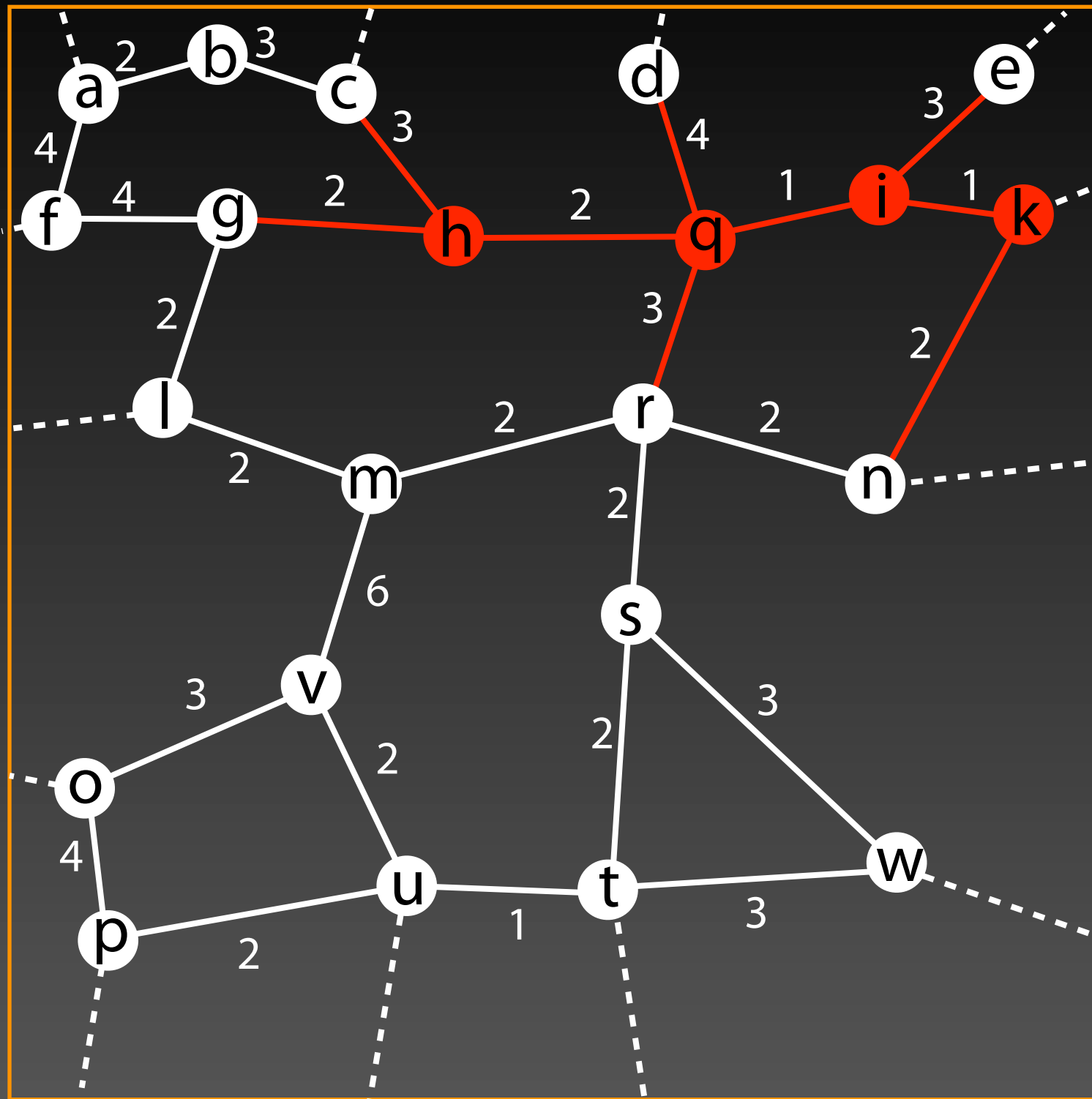
(d,4)

(e,4)

(g,4)

$(c, 5)$

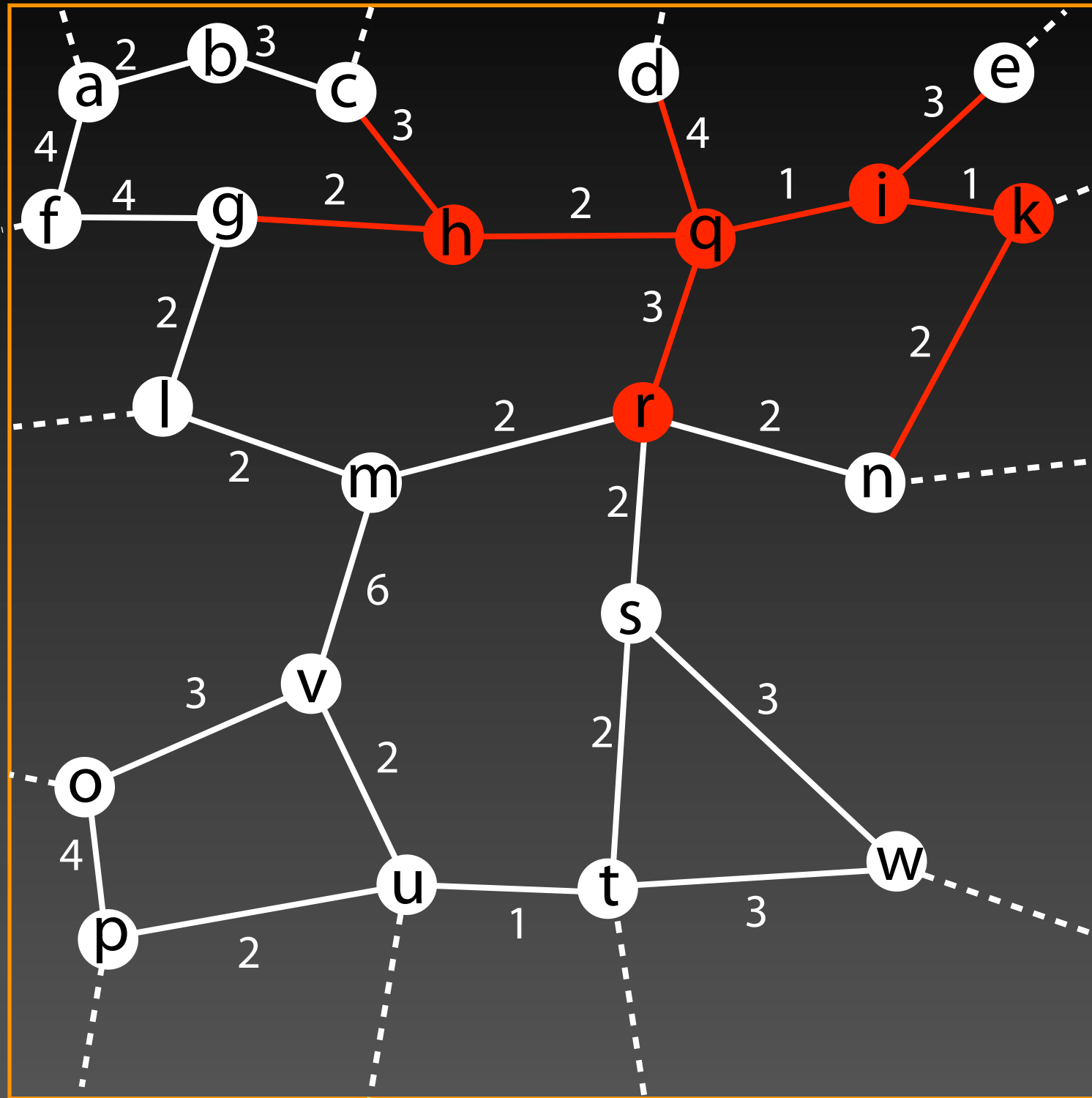
Shortest Paths / kNN (Dijkstra)



Queue:

(r,3)
(d,4)
(e,4)
(g,4)
(n,4)
(c,5)

Shortest Paths / kNN (Dijkstra)



Queue:

(d,4)

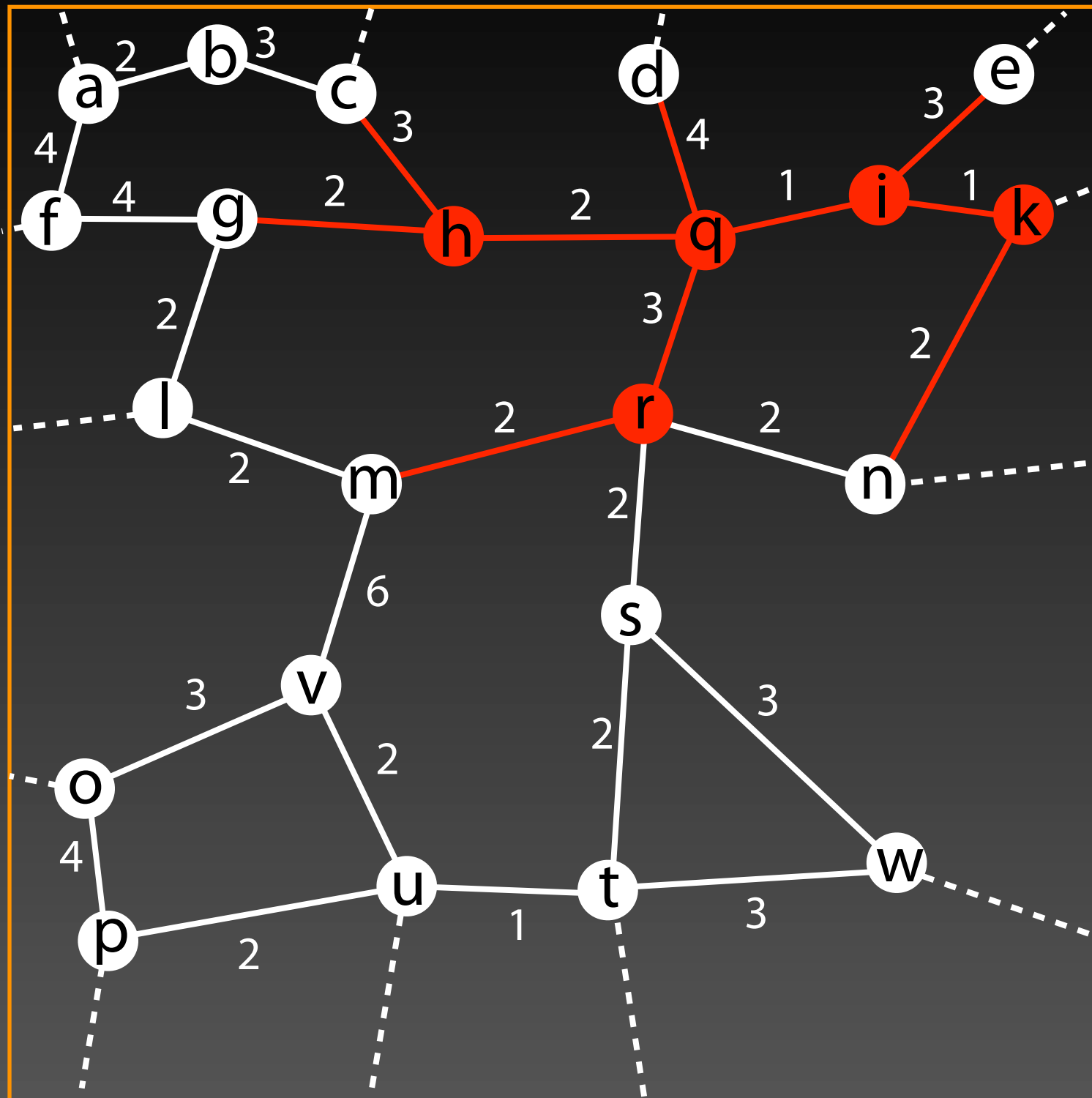
(e,4)

(g,4)

(n,4)

$(c, 5)$

Shortest Paths / kNN (Dijkstra)



Queue:

(d,4)

(e,4)

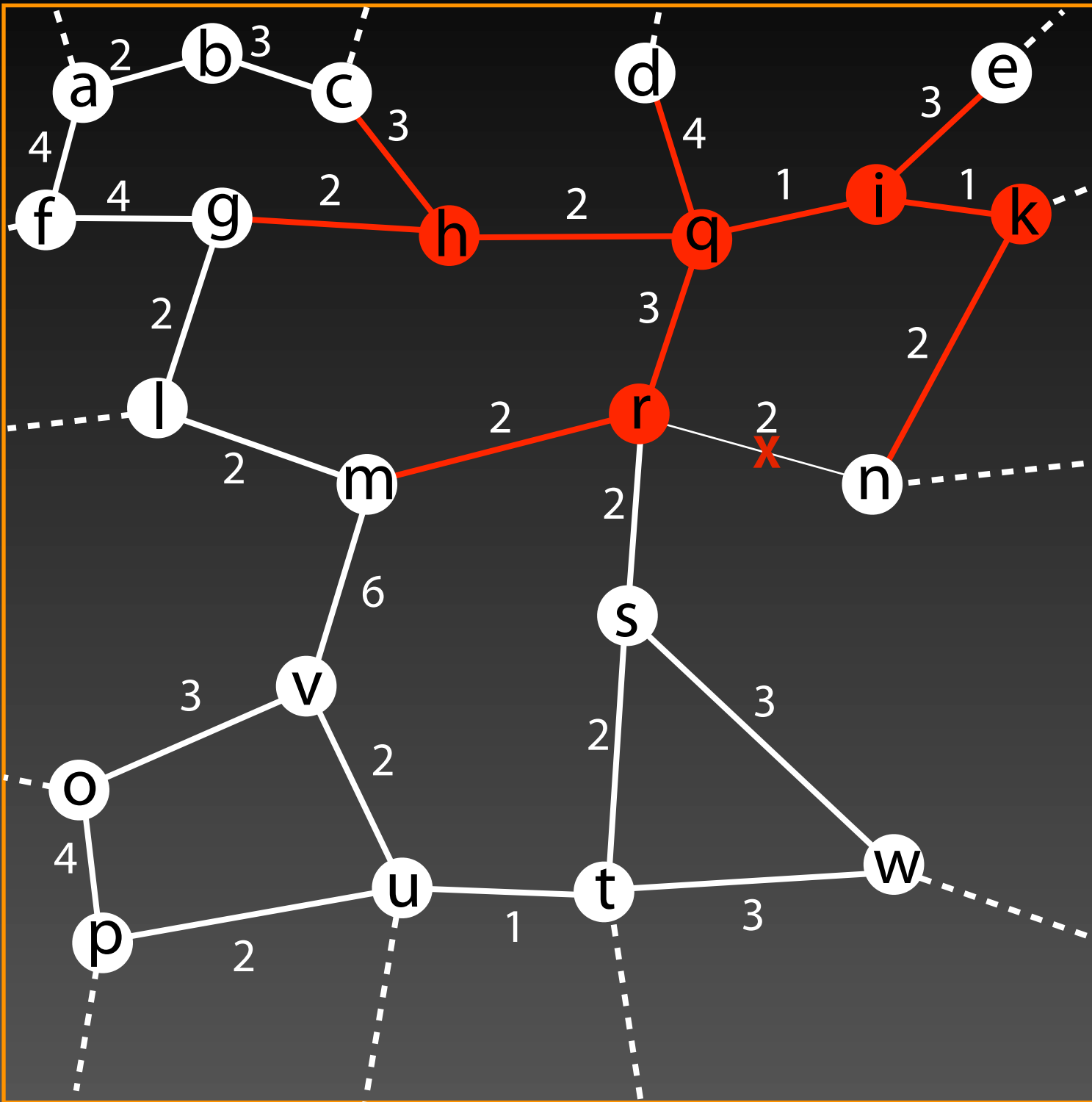
(g,4)

(n,4)

(c,5)

(m,5)

Shortest Paths / kNN (Dijkstra)



Queue:

(d,4)

(e,4)

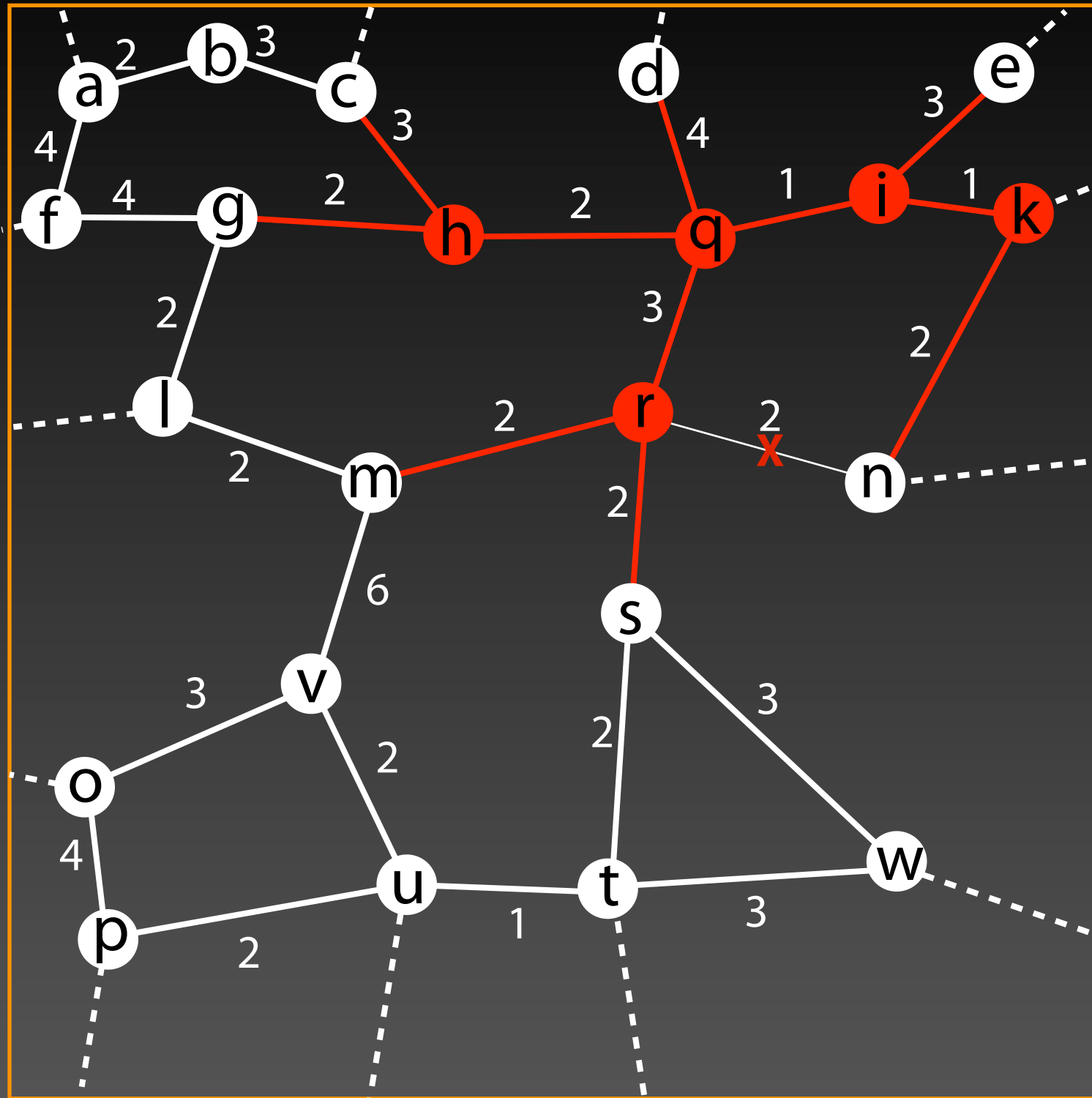
(g,4)

(n,4)

$(c, 5)$

$(m, 5)$

Shortest Paths / kNN (Dijkstra)



Queue:

(d,4)

(e,4)

(g,4)

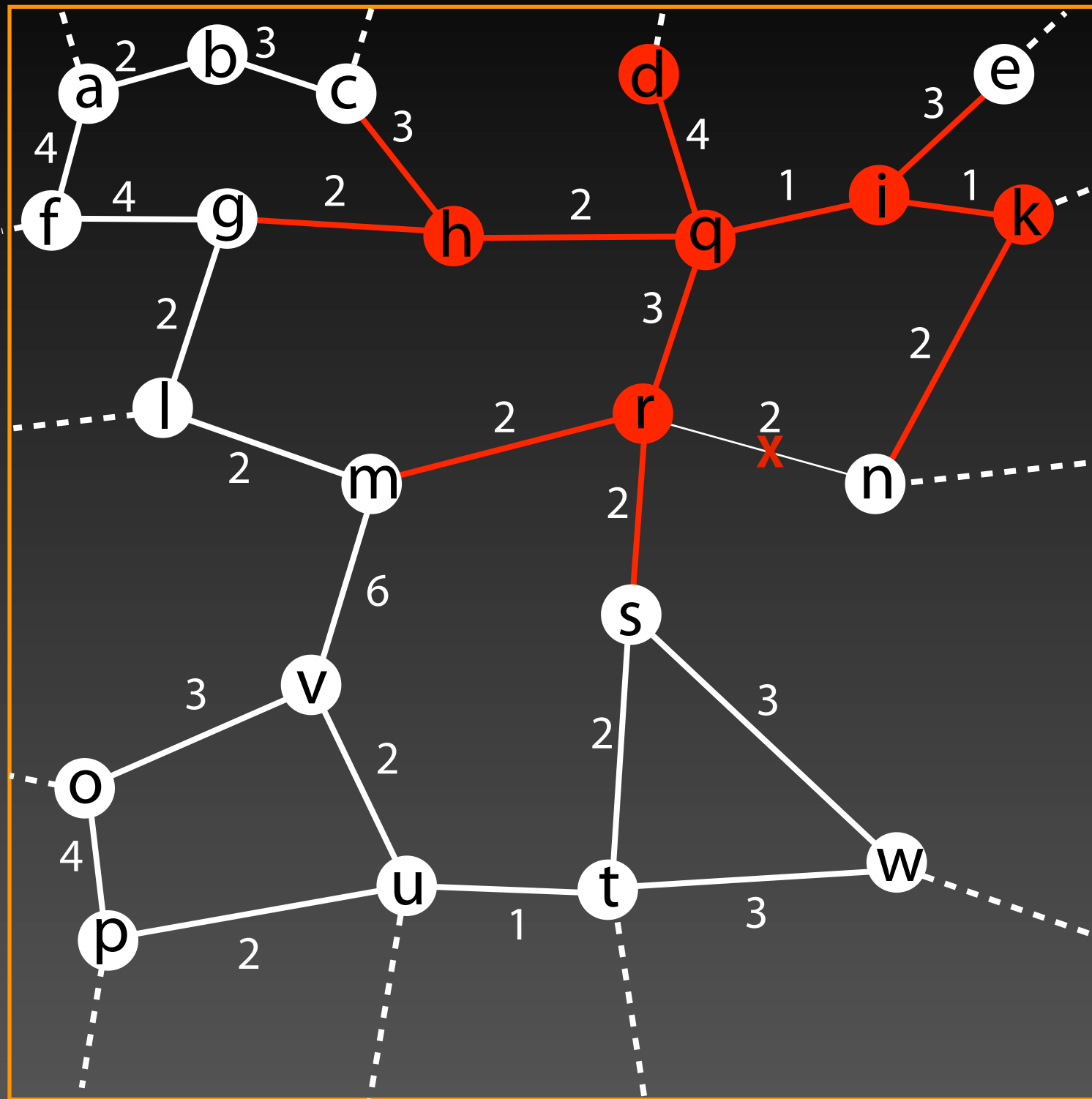
(n,4)

$(c, 5)$

(m,5)

$(s, 5)$

Shortest Paths / kNN (Dijkstra)



Queue:

(e,4)

(g,4)

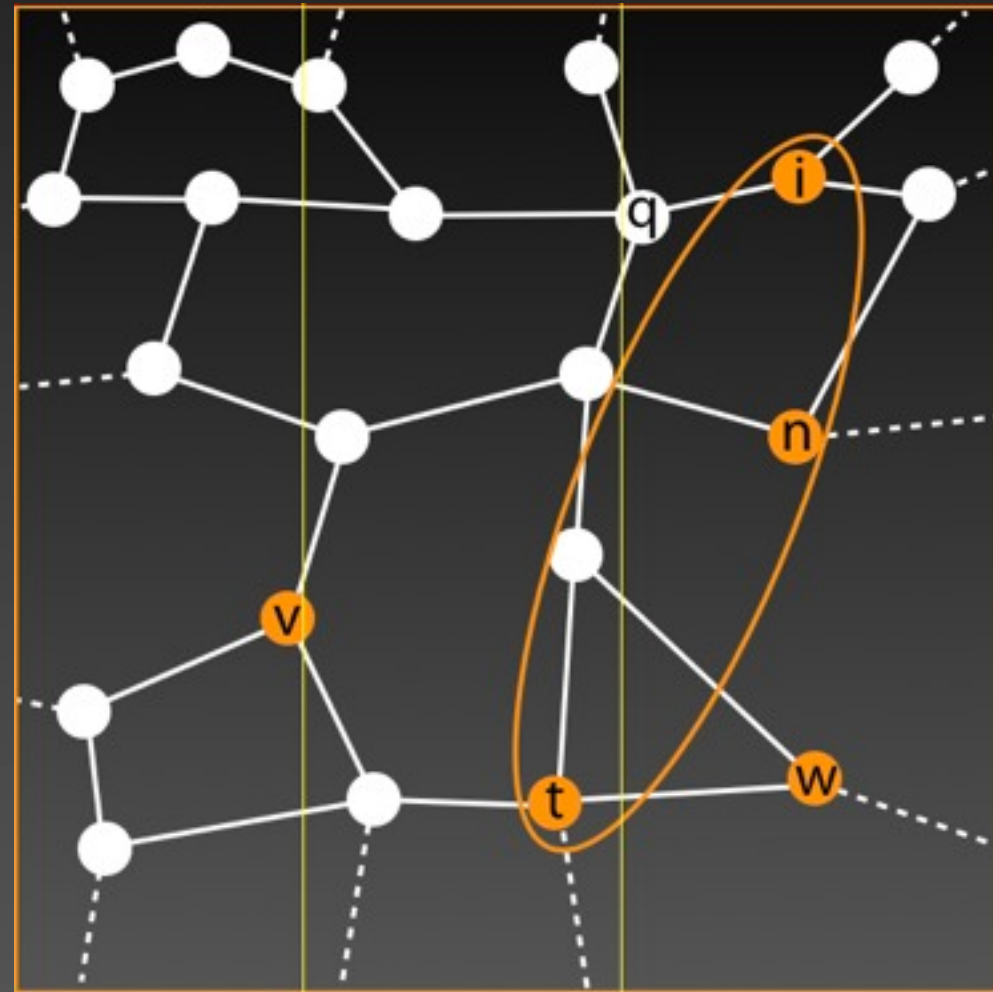
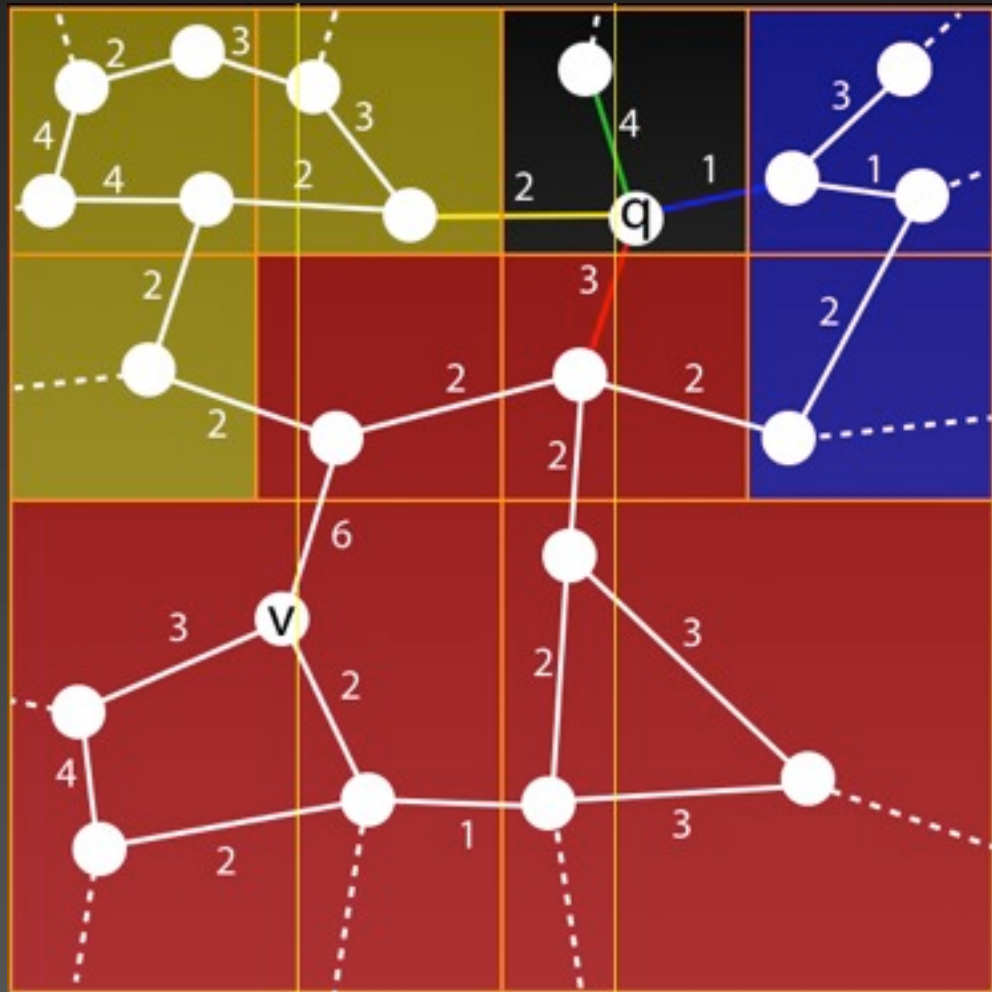
$(n, 4)$

(c,5)

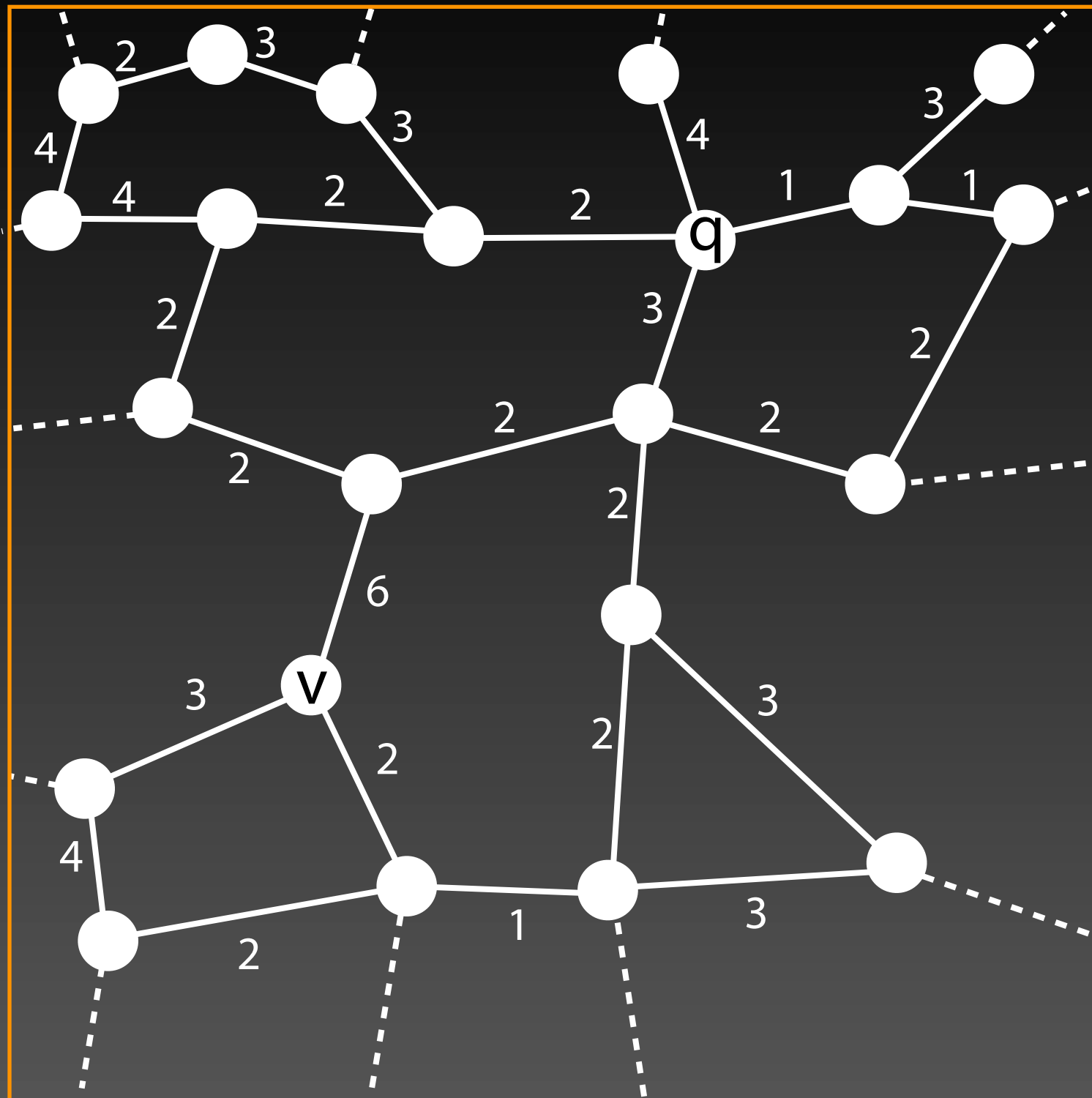
(m,5)

$(s, 5)$

Shortest Path Quadtrees + Best-First k-Nearest Neighbor



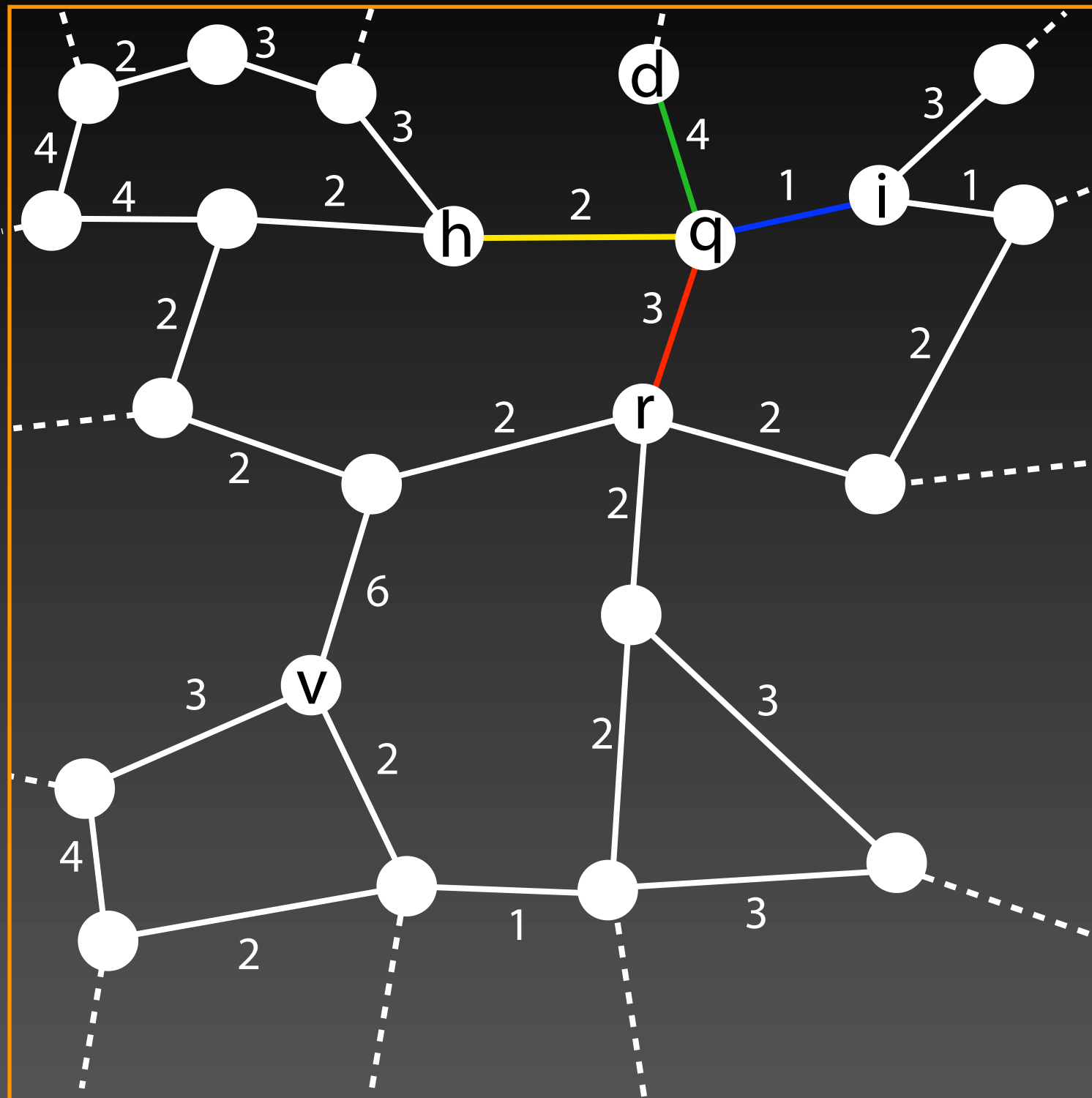
Precomputing Shortest Path Quadrees



Idea:

Static underlying
Road Network

Precomputing Shortest Path Quadrees



Edges:

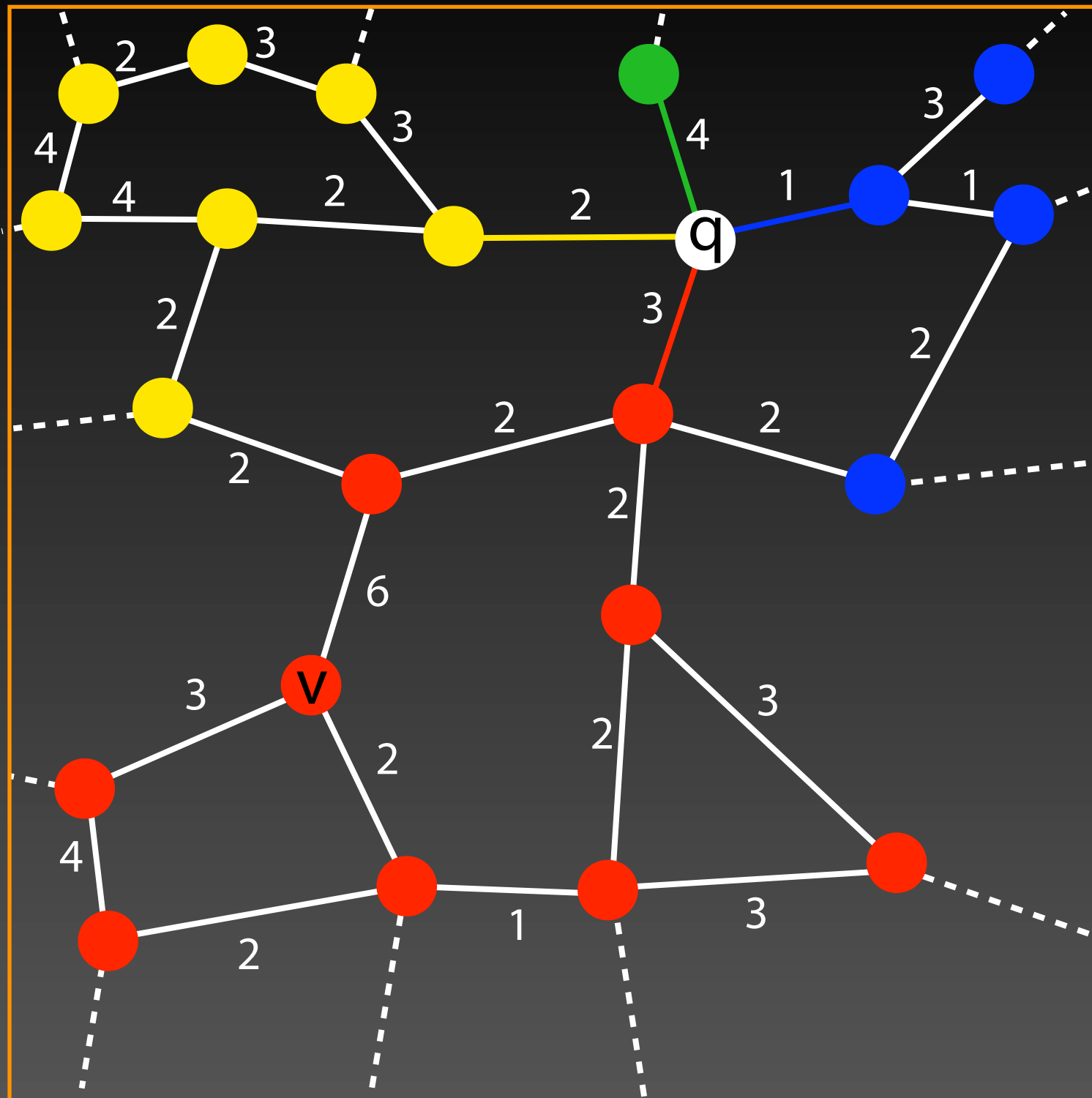
$\alpha(q, d; 4)$

$\beta(q, h; 2)$

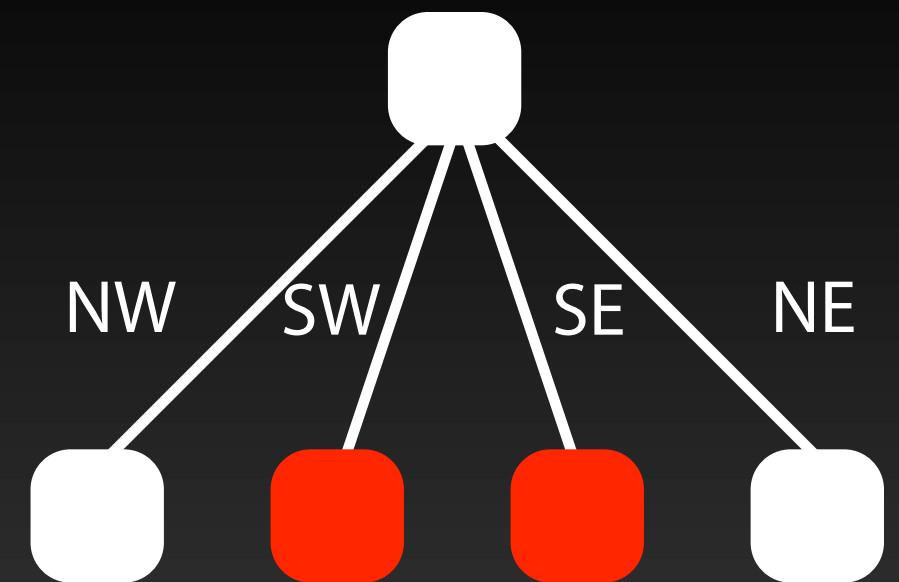
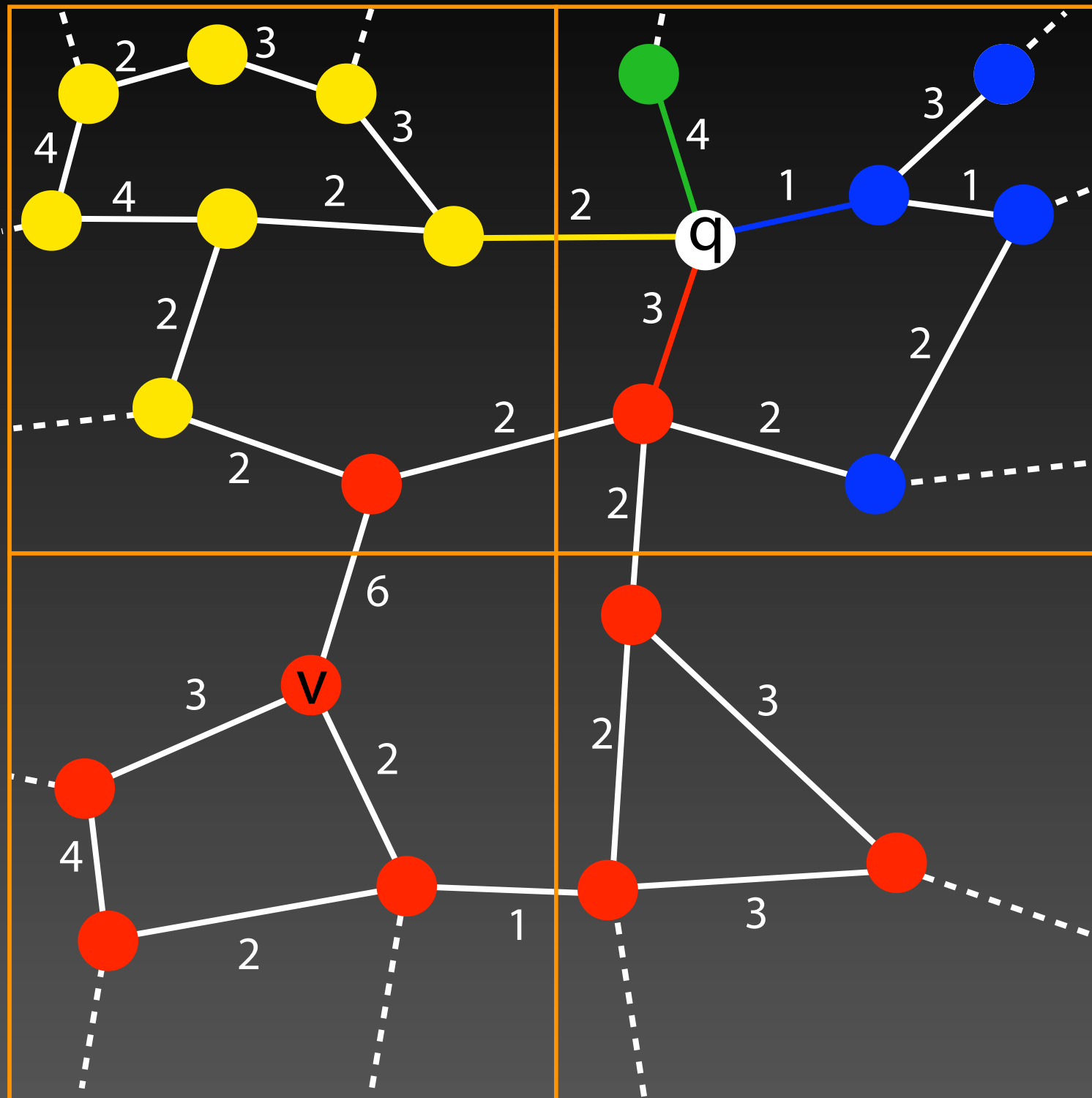
$\gamma(q, i; 1)$

$\delta(q, r; 3)$

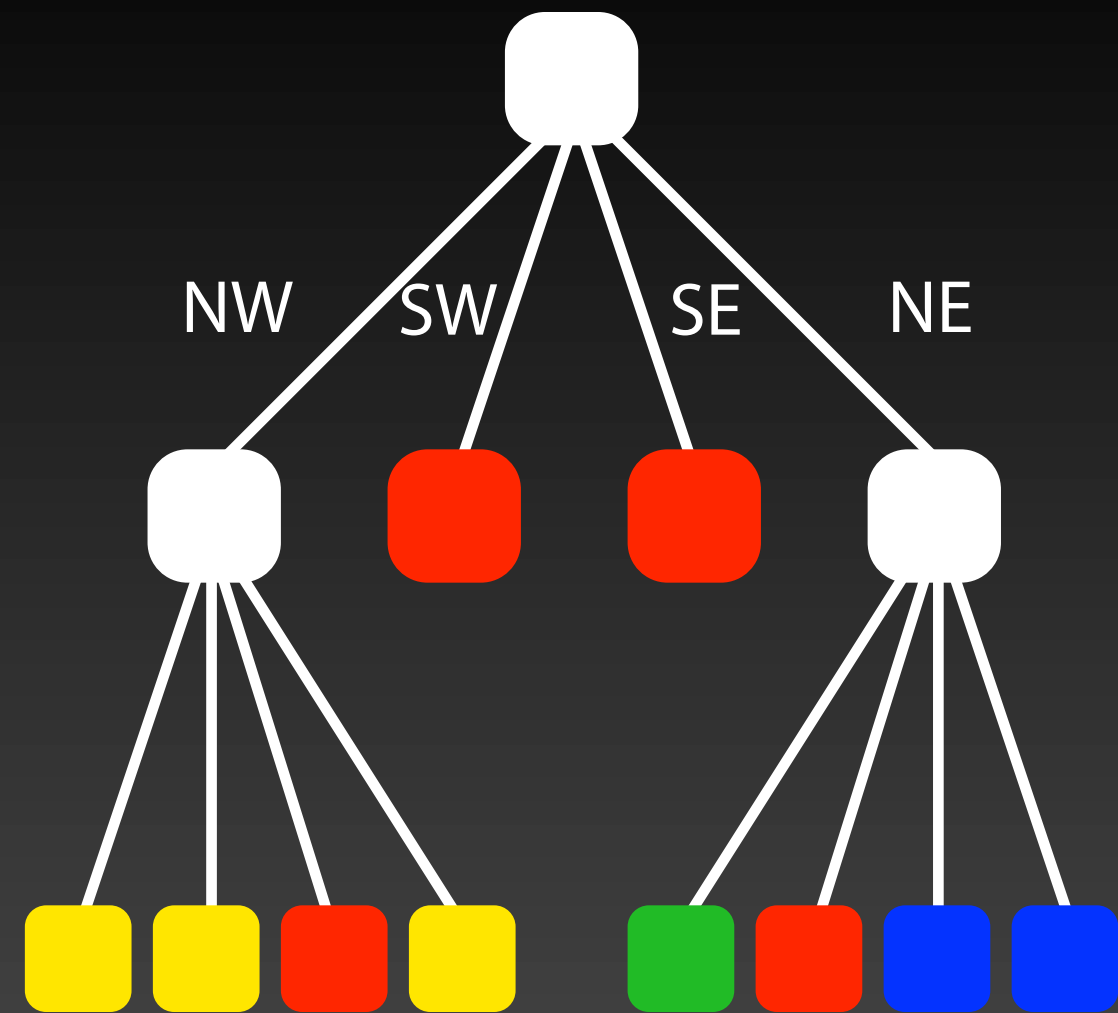
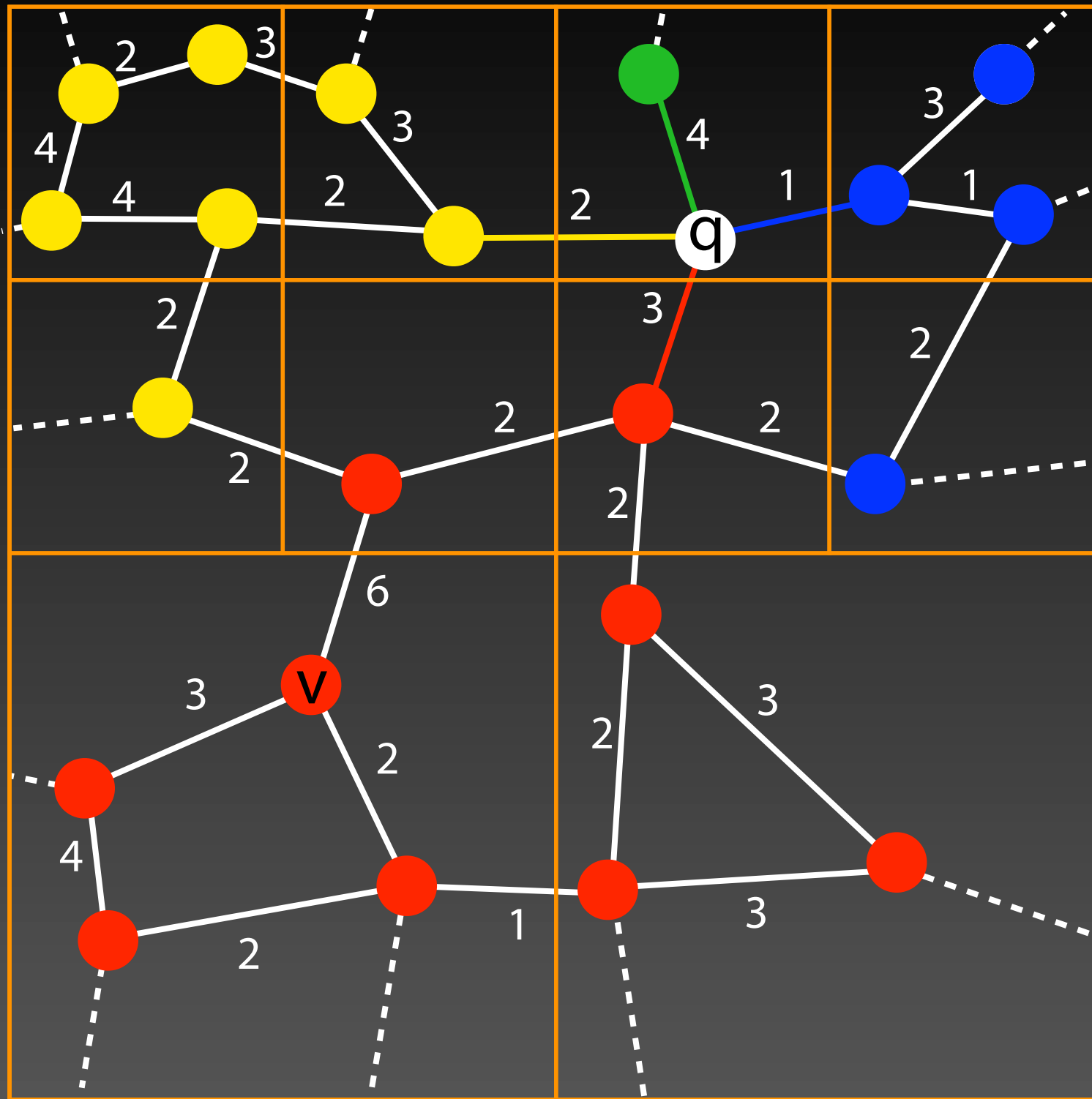
Precomputing Shortest Path Quadtrees



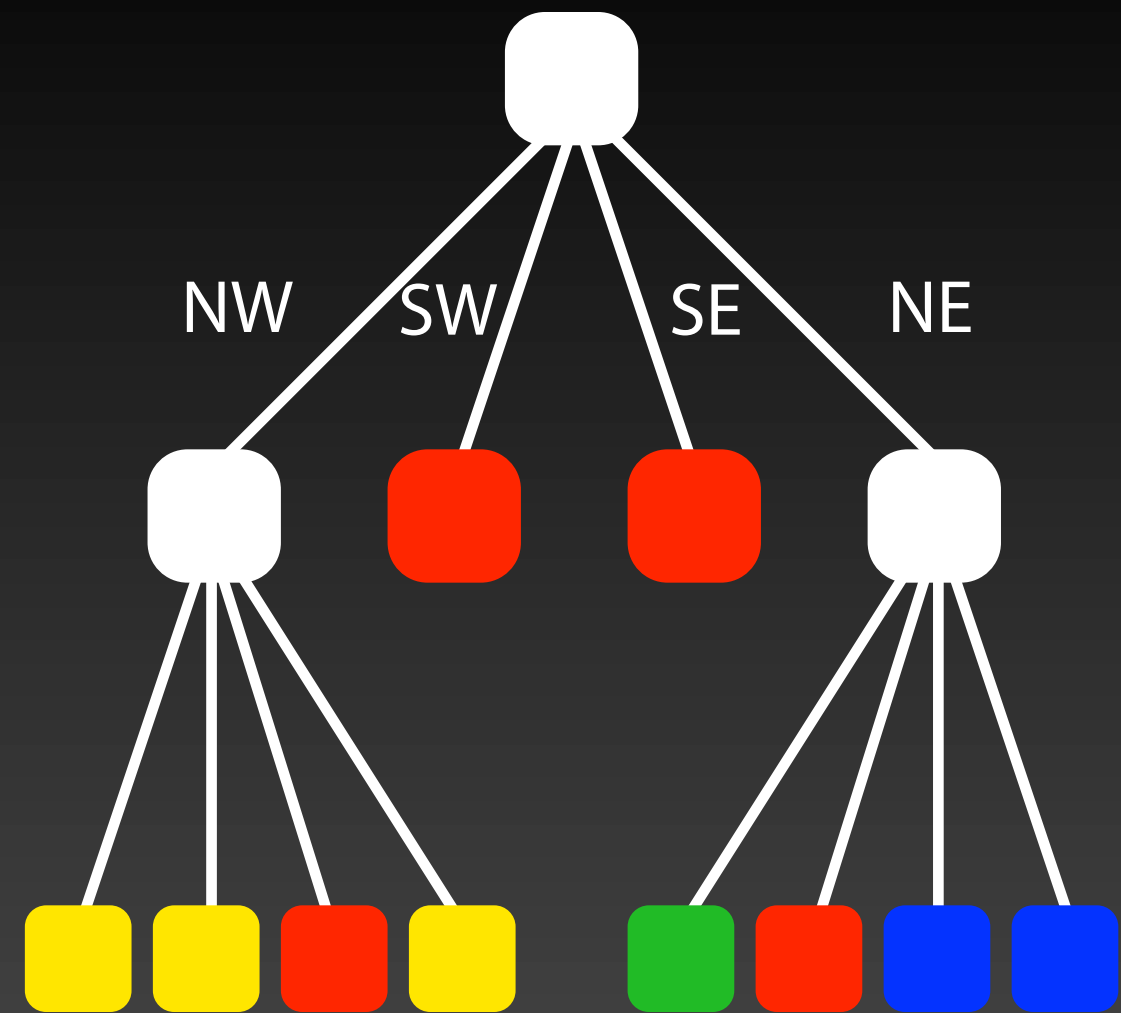
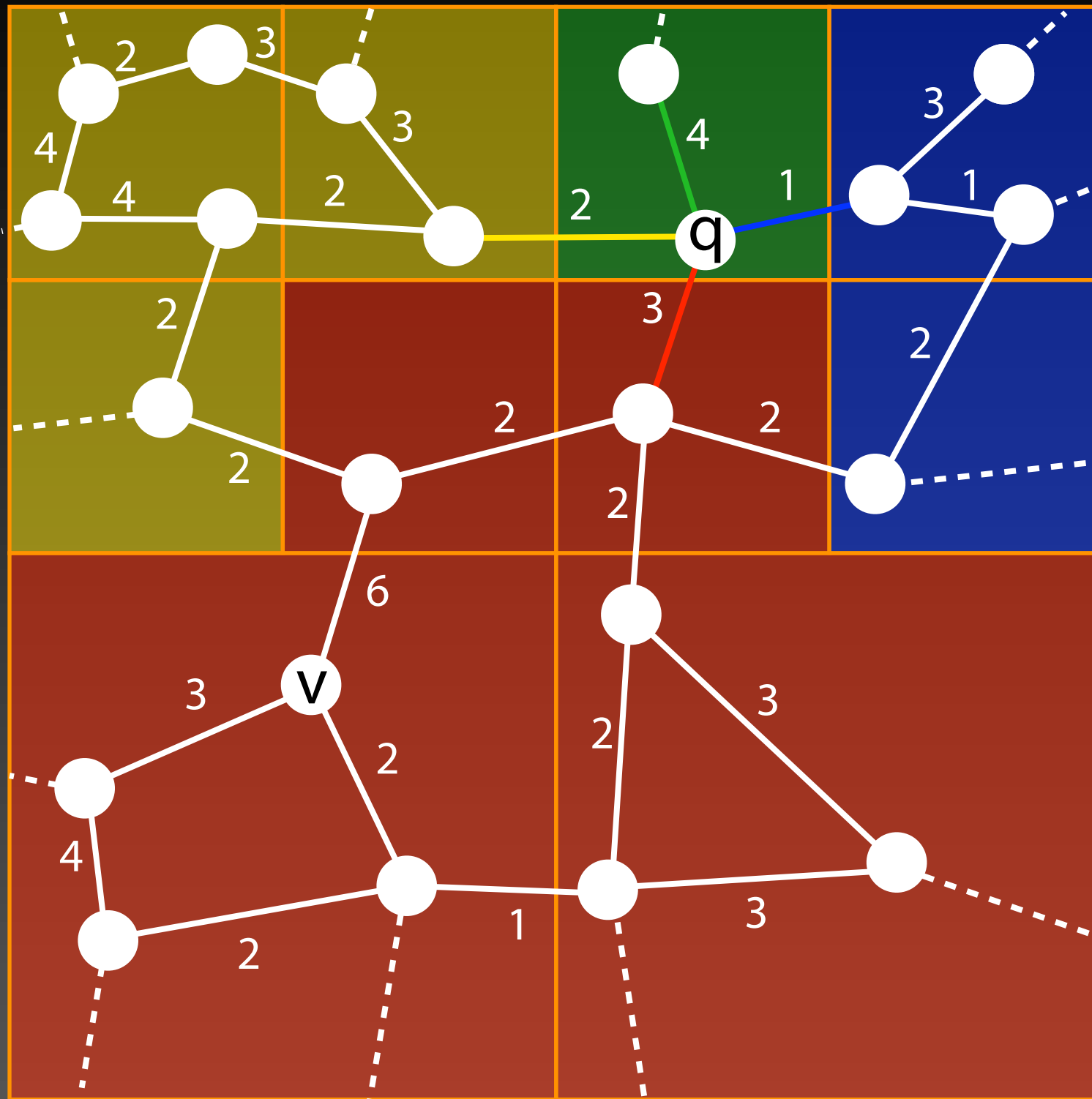
Precomputing Shortest Path Quadtrees



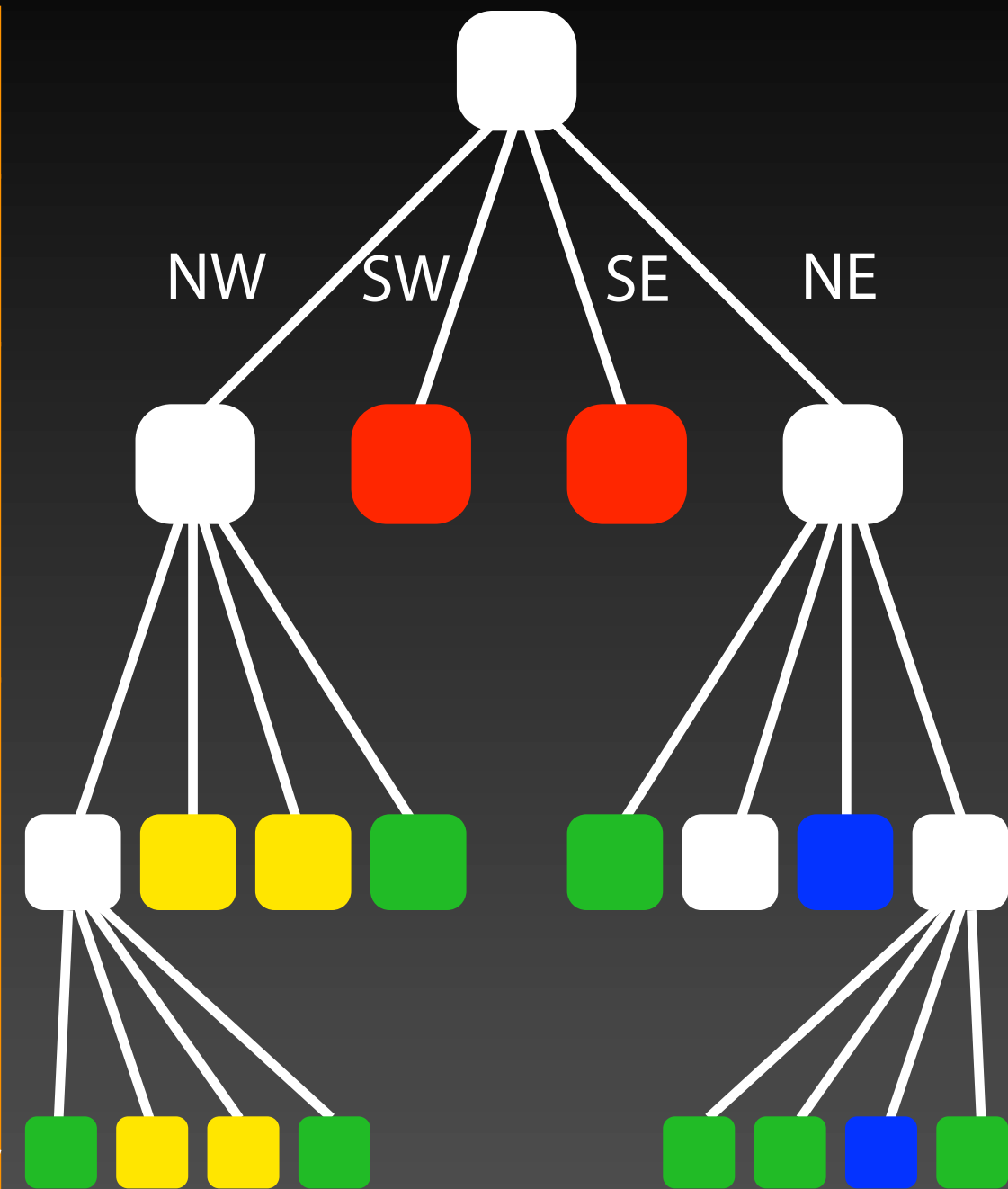
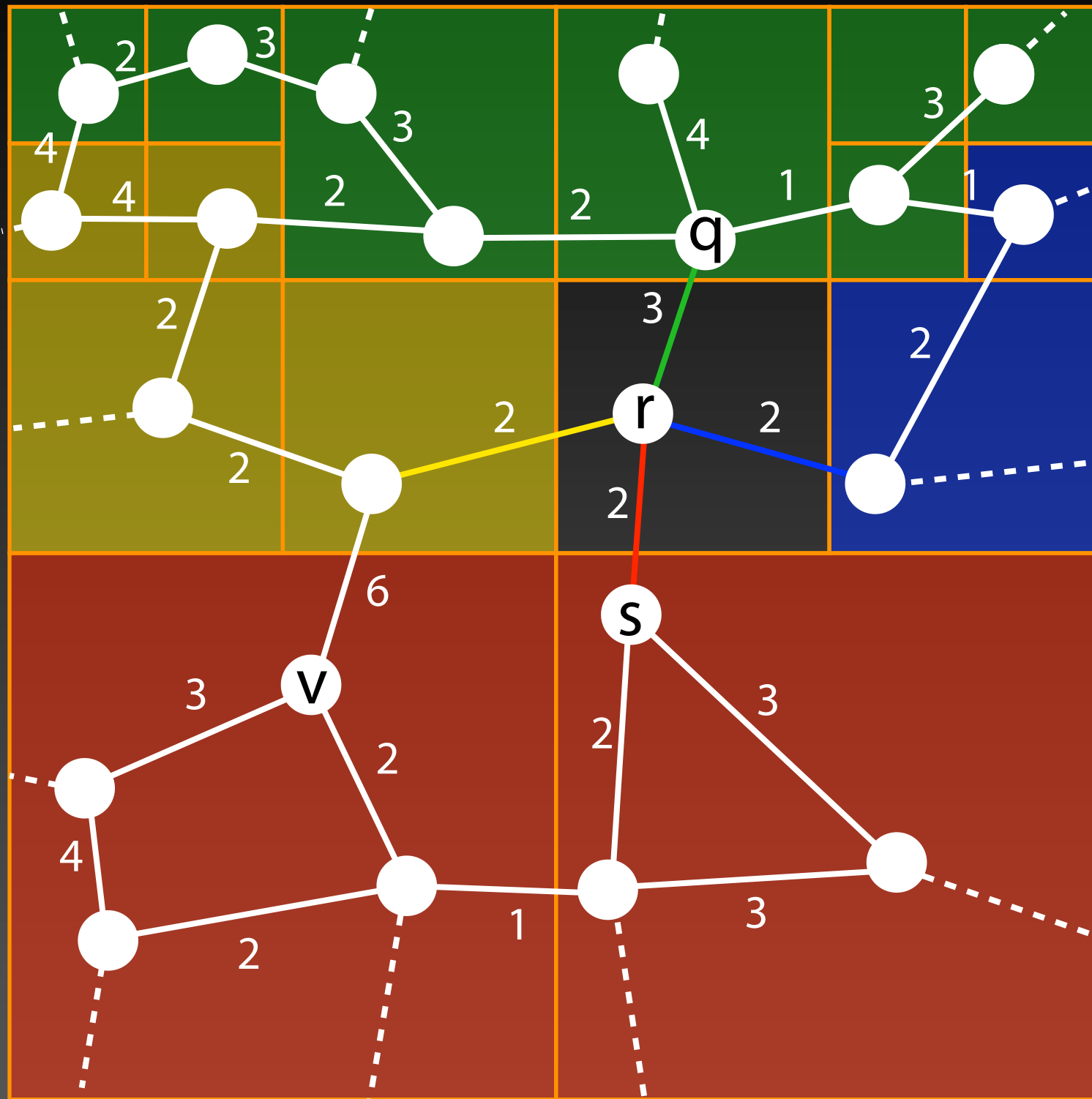
Precomputing Shortest Path Quadtrees



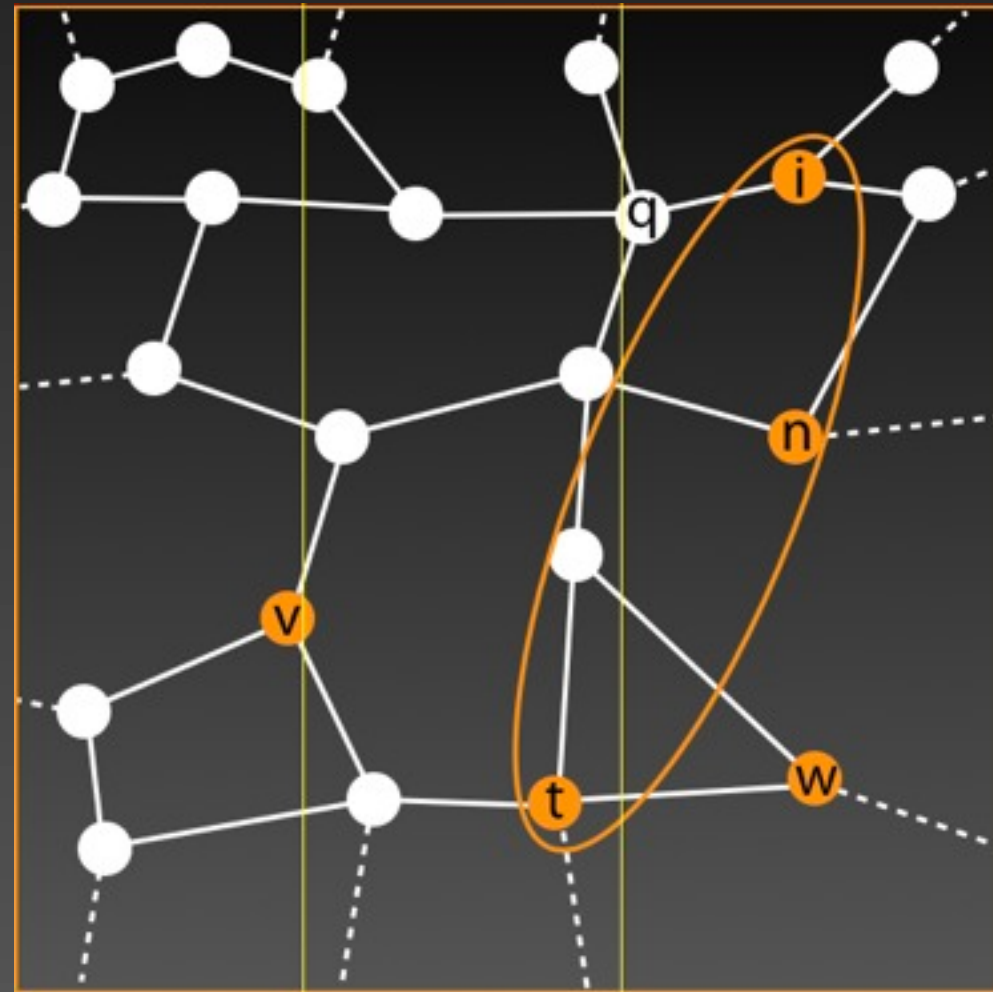
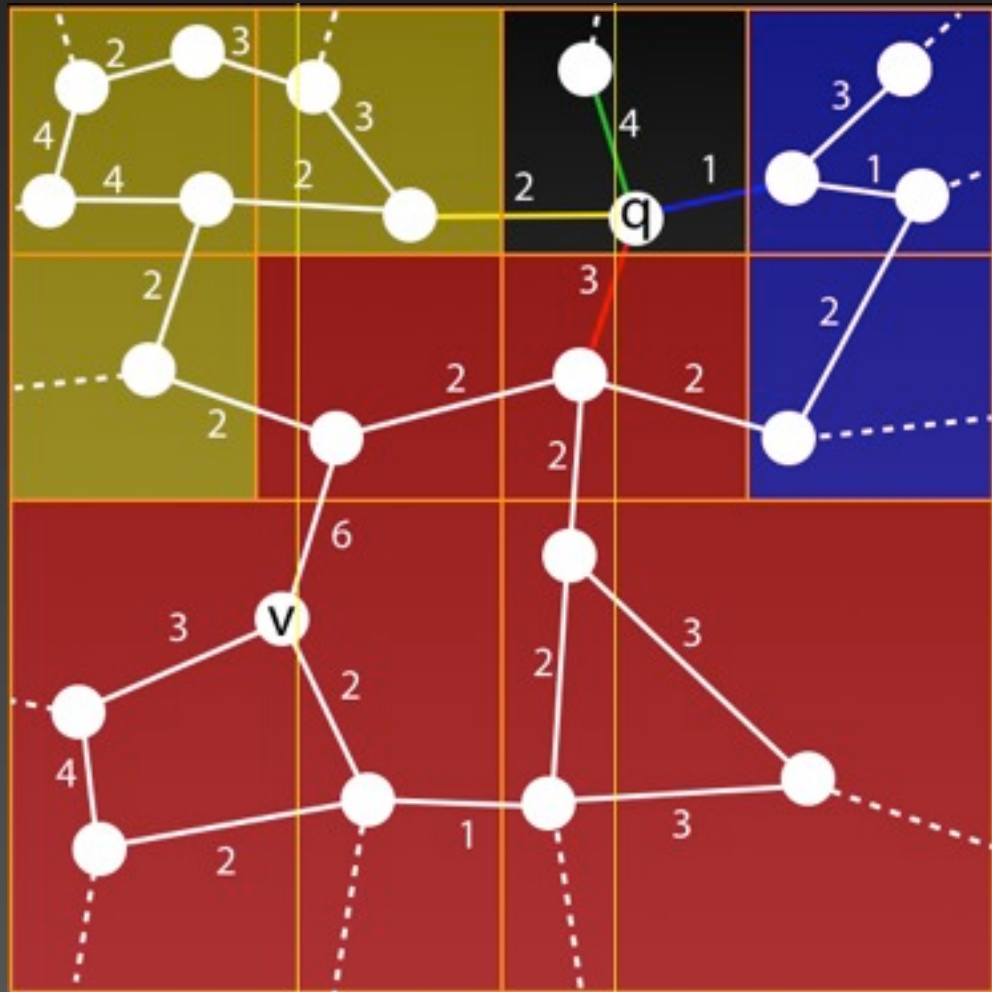
Precomputing Shortest Path Quadtrees



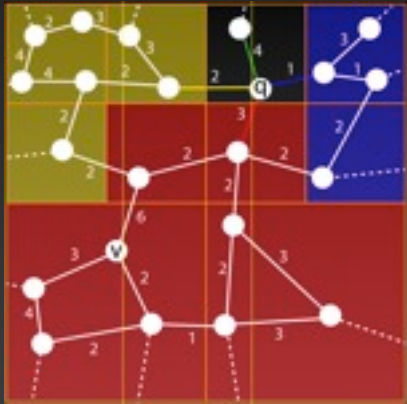
Precomputing Shortest Path Quadtrees



Shortest Path Quadtrees + Best-First k-Nearest Neighbor

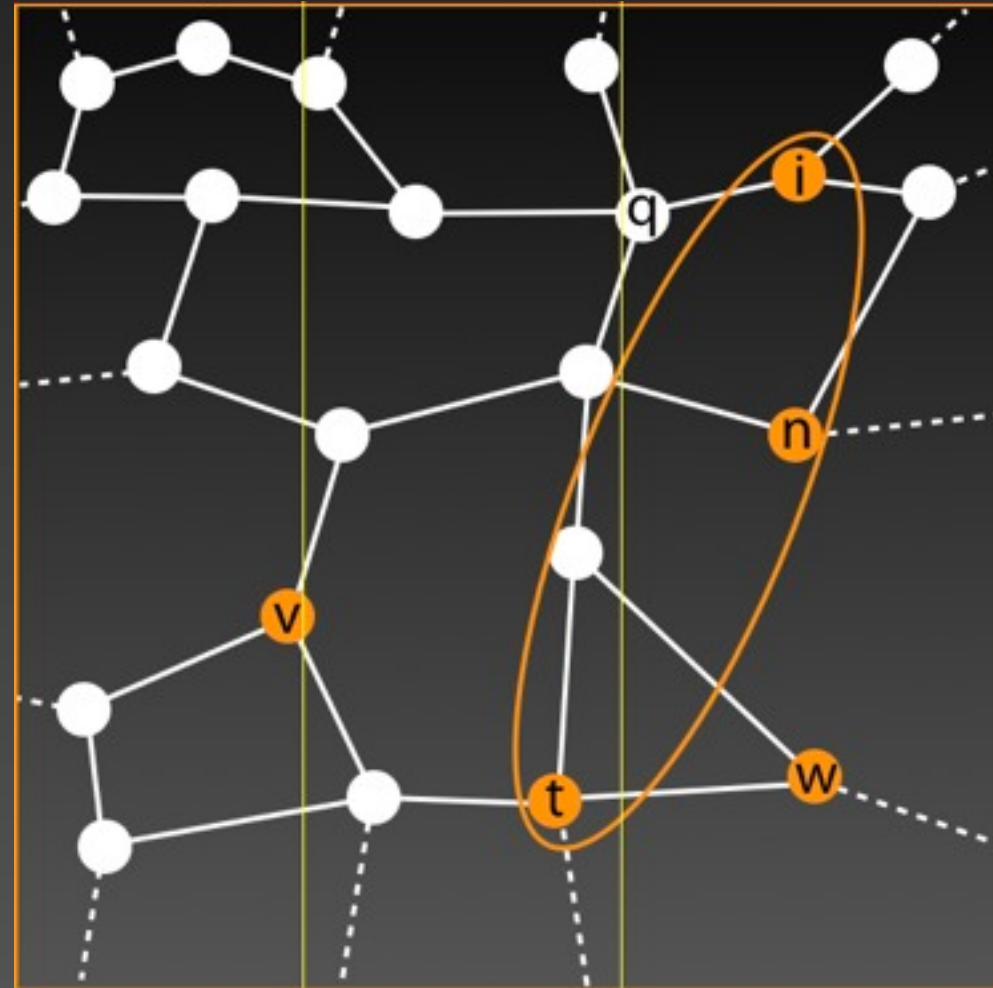


Shortest Path Quadtrees + Best-First k-Nearest Neighbor

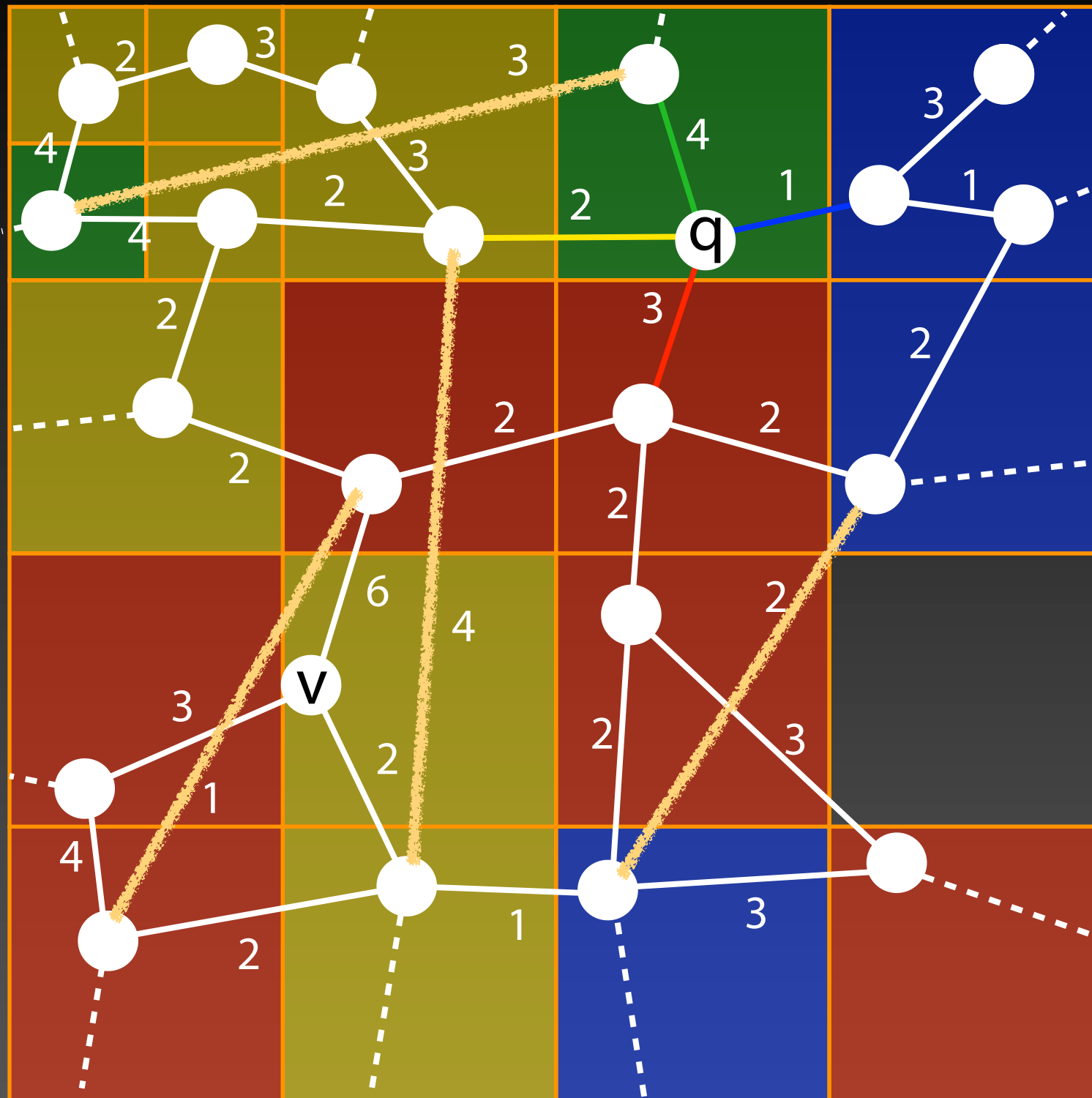


Issues:

- Non-Planar Networks
- Multiple Shortest Paths



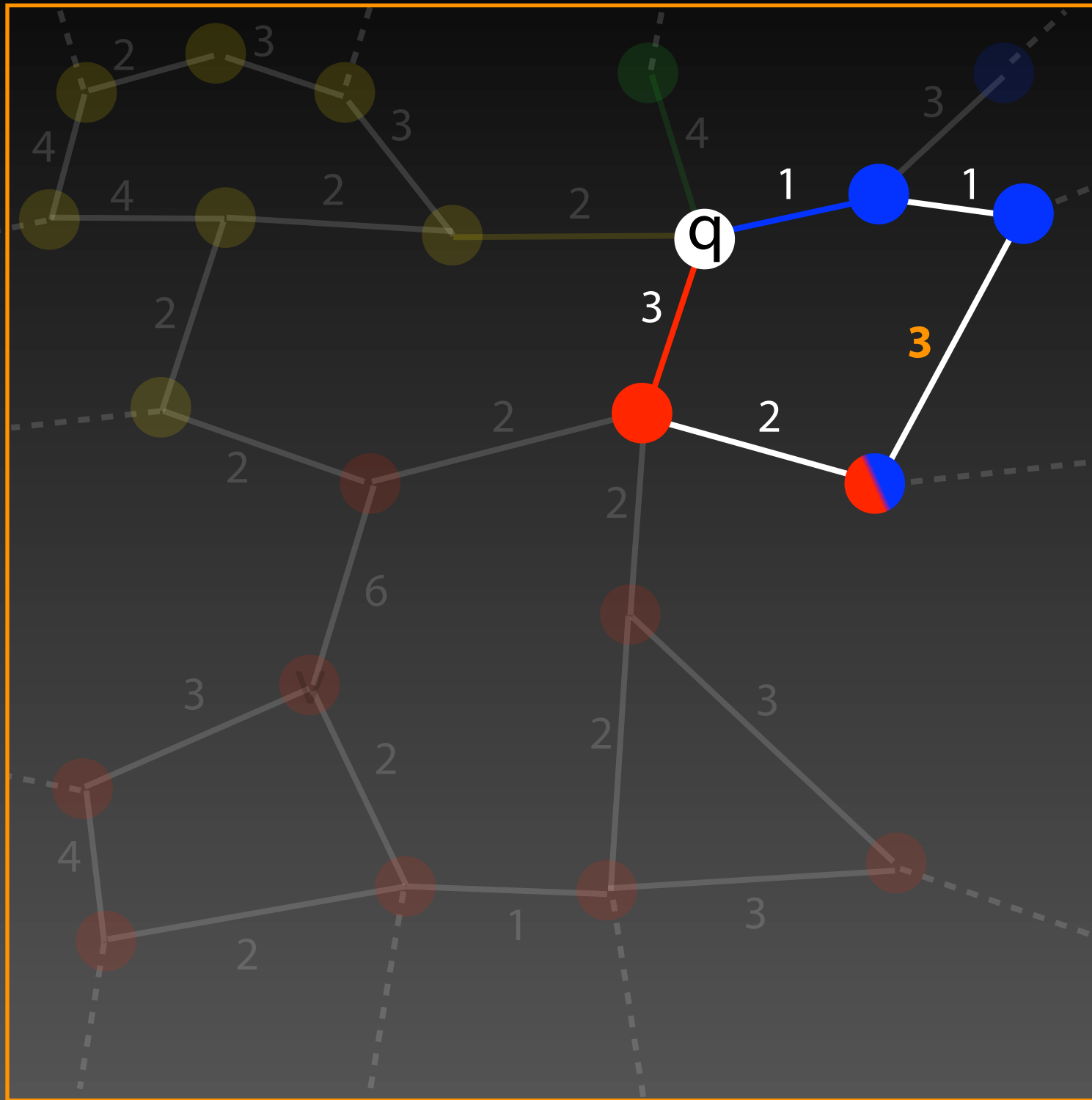
Non-Planar Networks



Tunnels/Bridges:

- Regions not contiguous
- More storage required
- Bigger effort to retrieve shortest paths
- kNN will still work

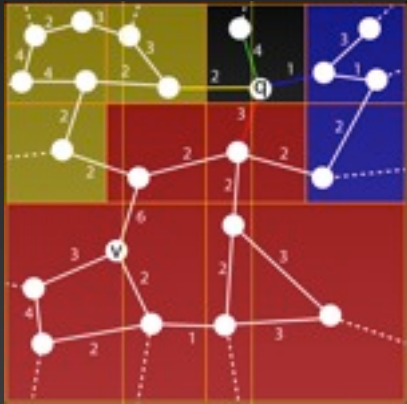
Multiple Shortest Paths



Requirements/
Assumptions:

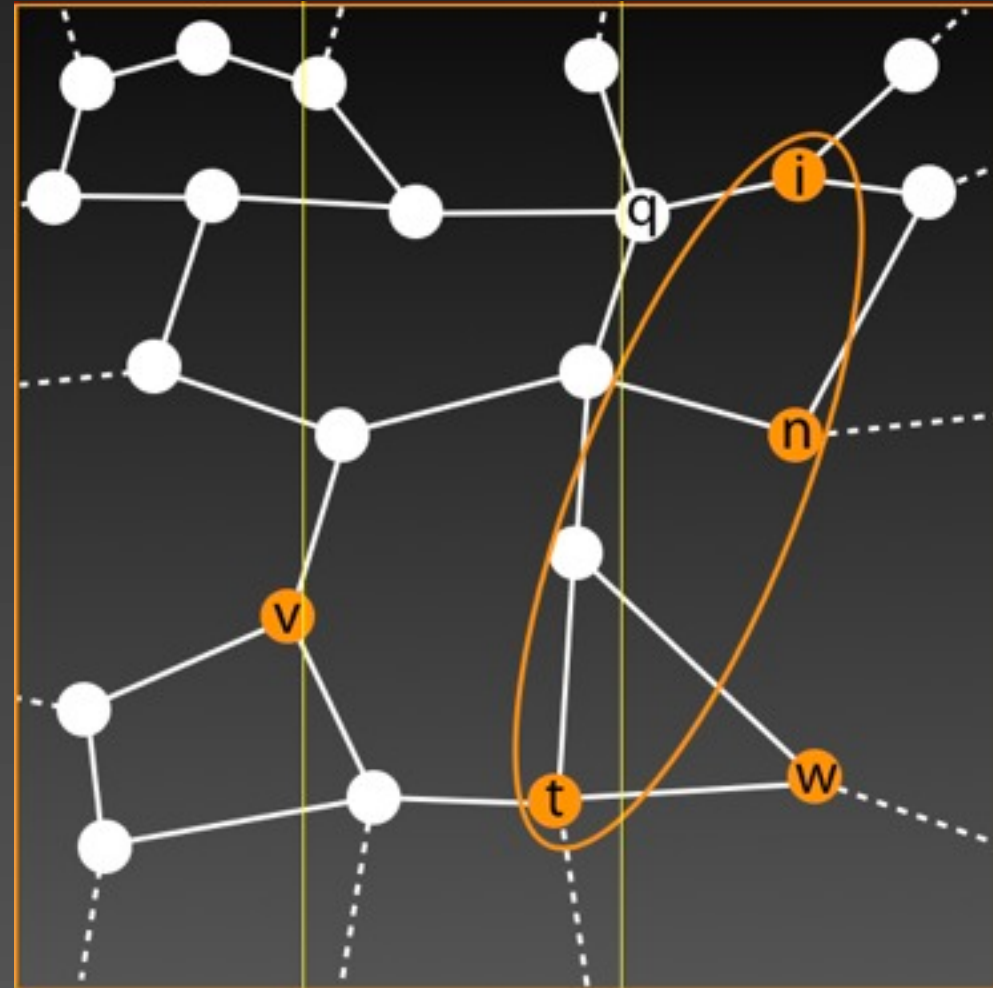
- Unique shortest paths
- Disjoint regions
- Achieved by additional criteria (e.g. less intersections, tolls)

Shortest Path Quadtrees + Best-First k-Nearest Neighbor

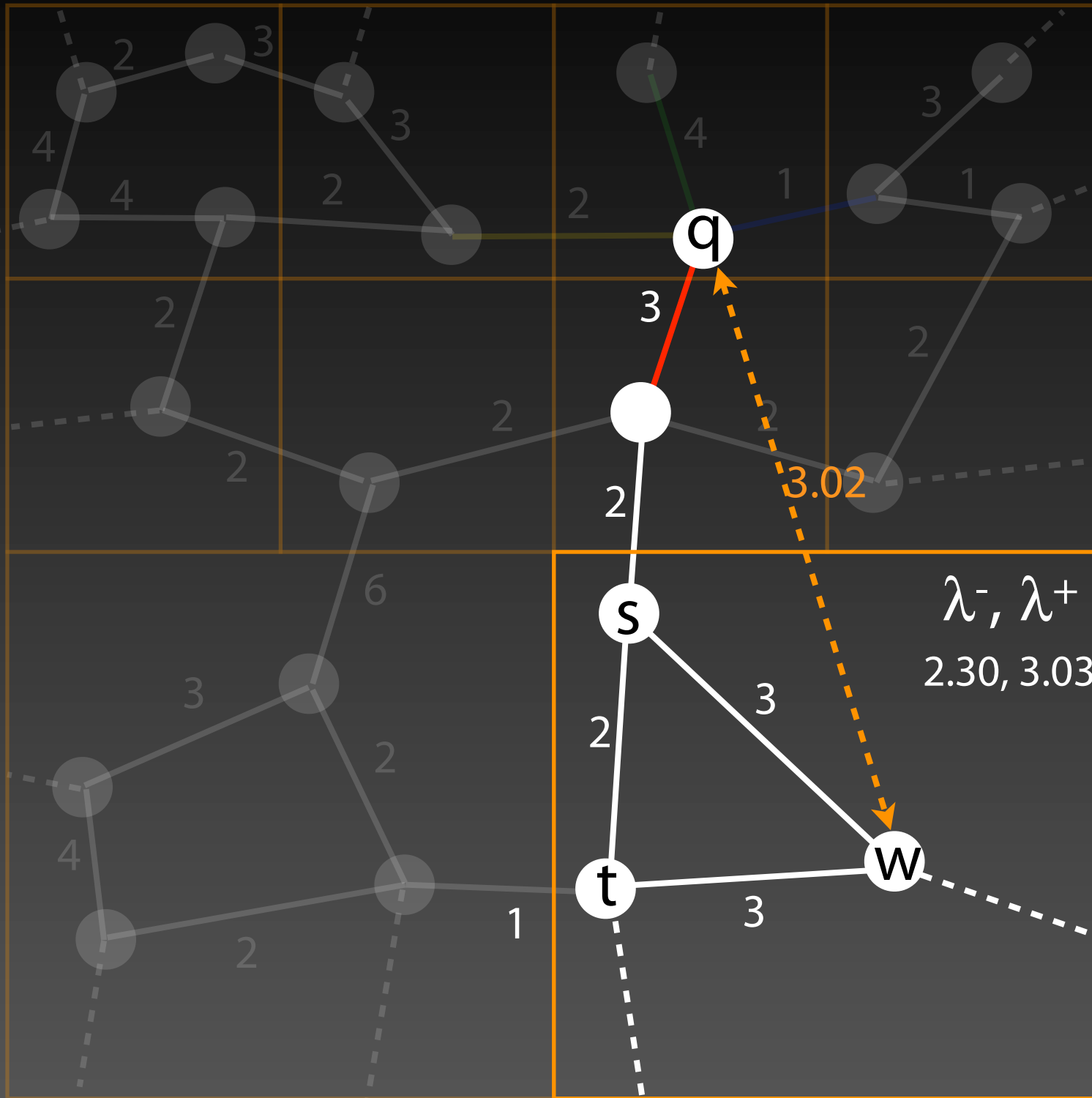


Issues:

- Non-Planar Networks
- Multiple Shortest Paths



Ratios of Network to Spatial Distances



Distances:

$\varepsilon(q,w) = 8$ (Network)

$\sigma(q,w) = 3.02$ (Spatial)

$$\lambda = \varepsilon / \sigma \quad (\text{Ratio})$$

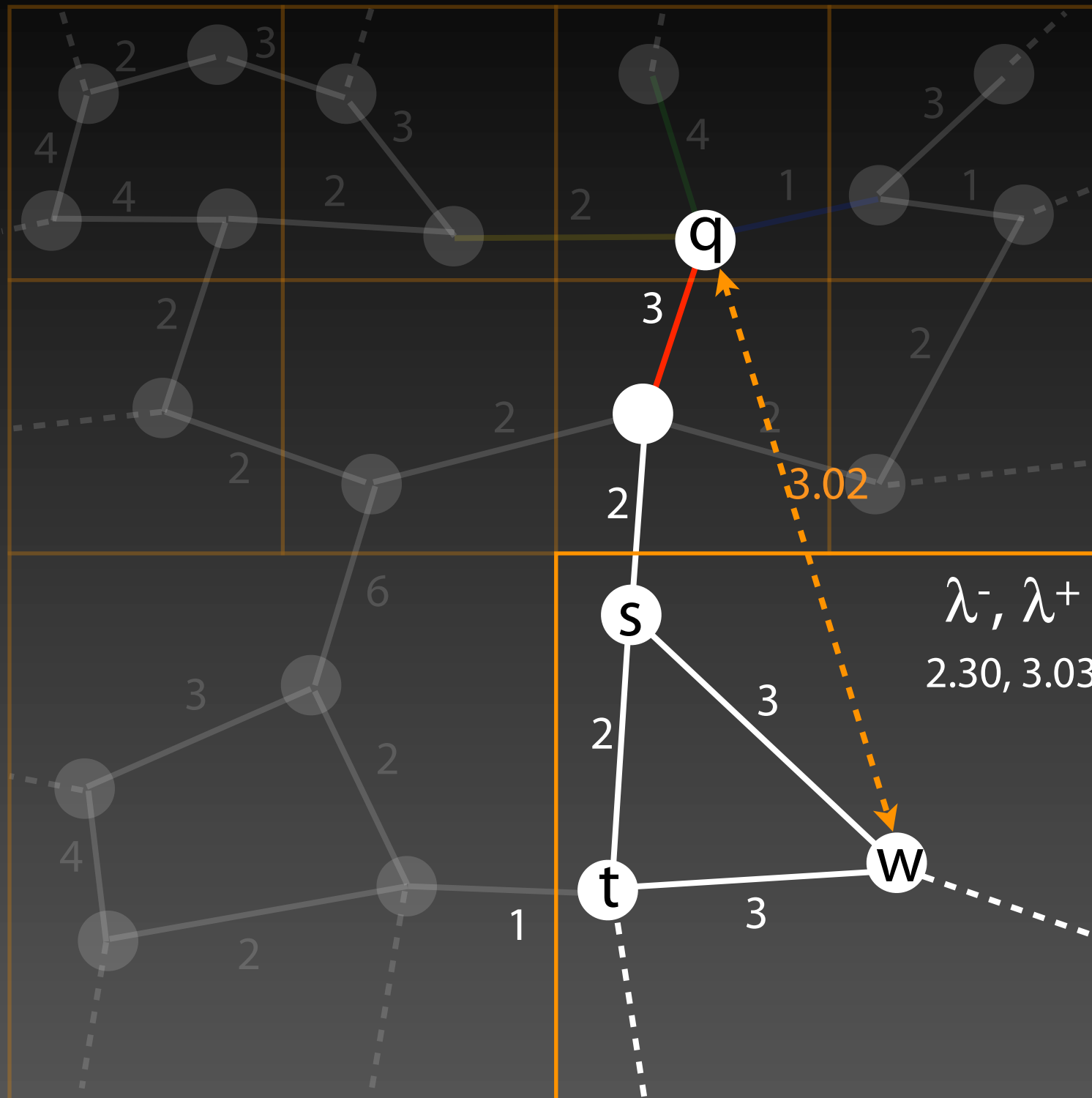
$$\lambda(q,w) = 8/3.02 = 2.65$$

$$\lambda(q,s) = 5/1.65 = 3.03$$

$$\lambda(q,t) = 7/3.05 = 2.30$$

$$(\lambda^-, \lambda^+) = (2.30, 3.03)$$

Network Distance Interval



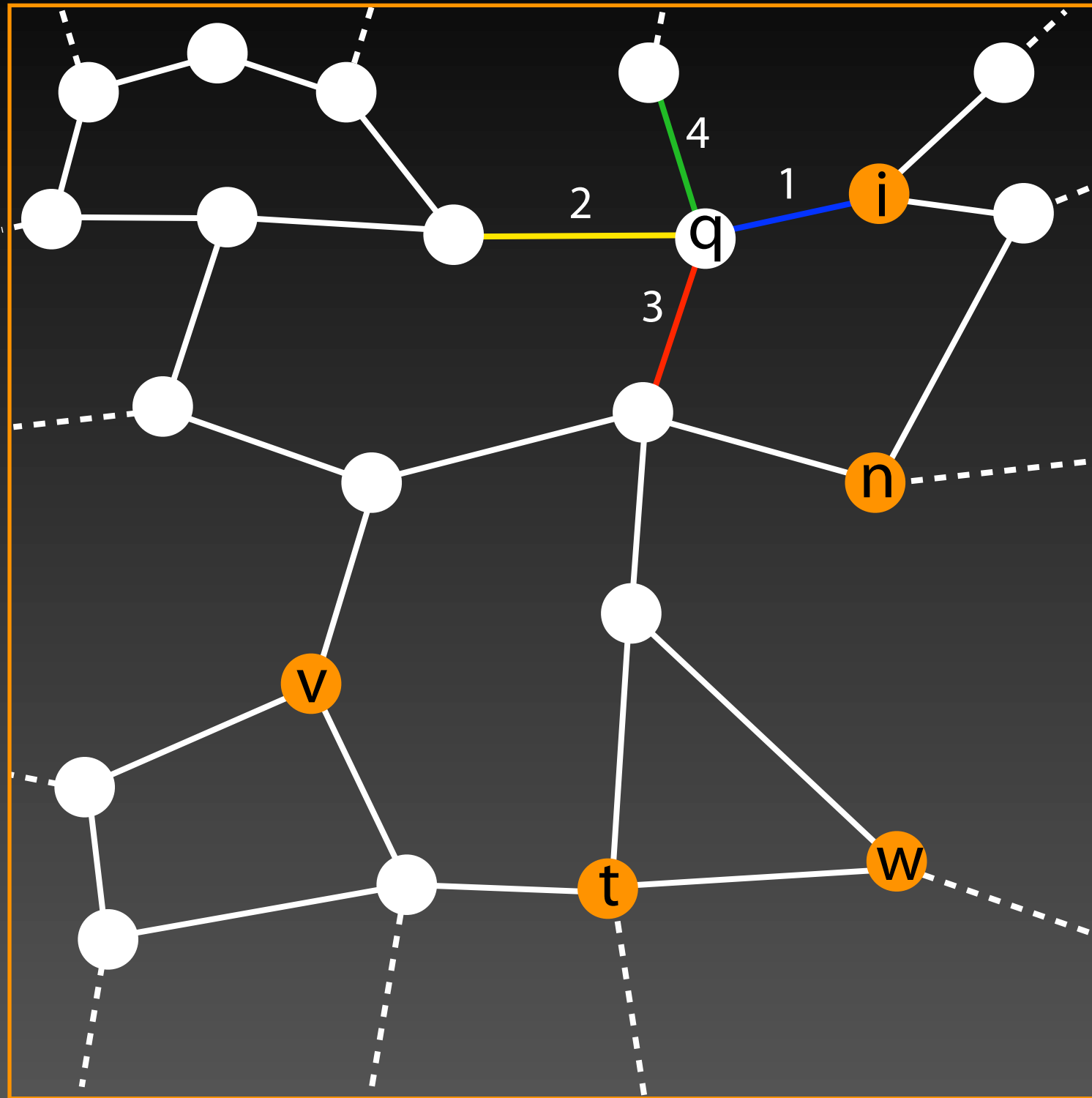
Distance Estimate q, w :

$$3.02 \times (\lambda^-, \lambda^+) = 3.02 \times (2.30, 3.03) =$$

$$w(\delta^-, \delta^+) = w(6.95, 9.15)$$

δ^- Min. Network Distance
 δ^+ Max. Network Distance

Finding k-Nearest Neighbors

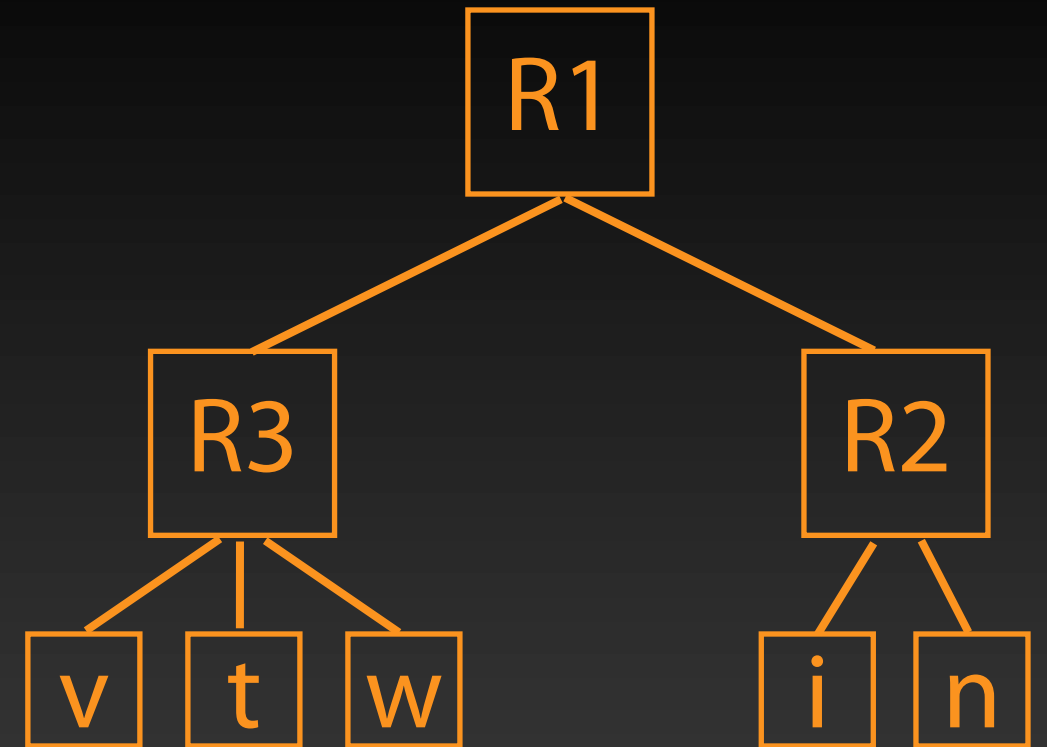
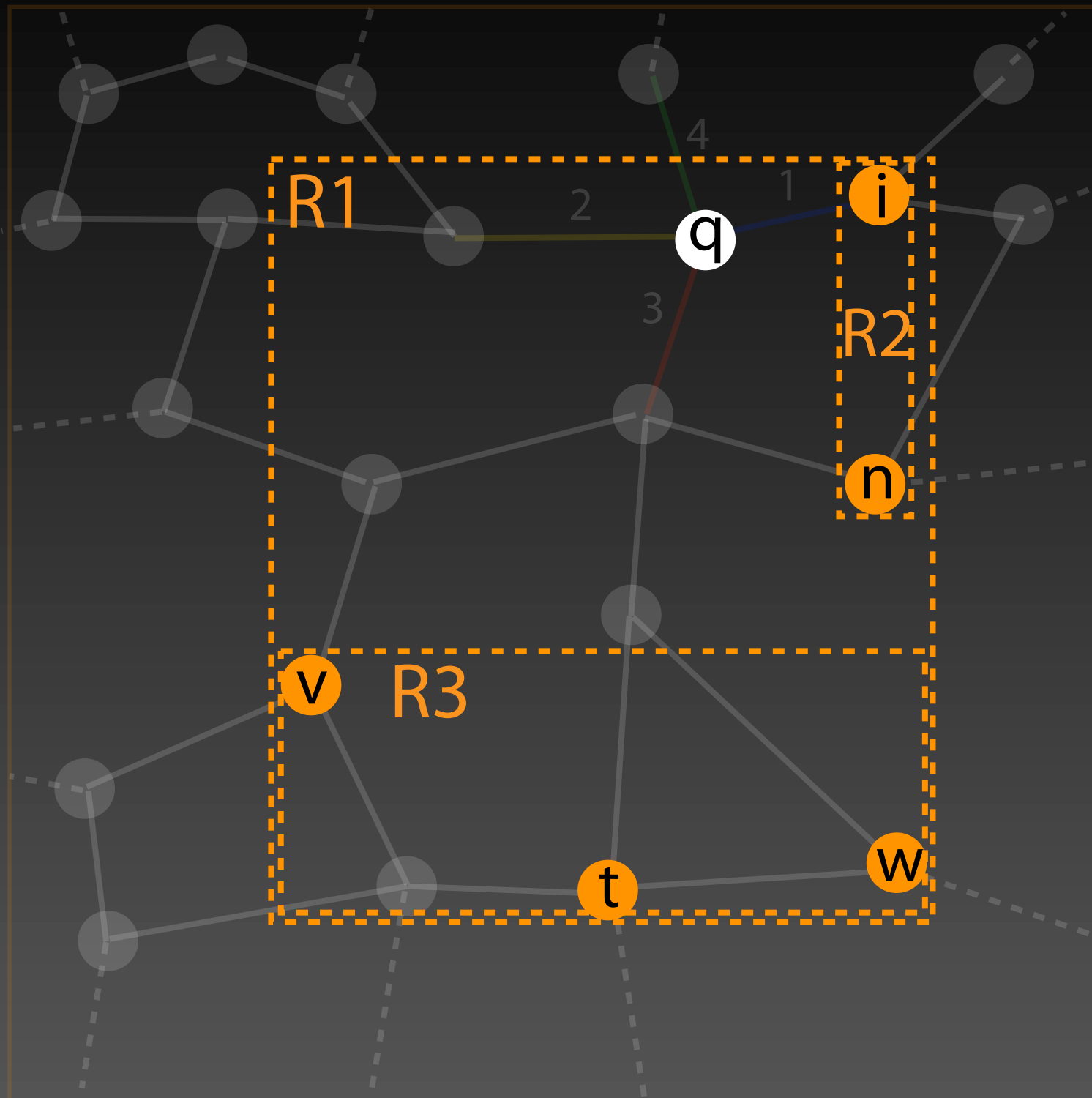


Goal:

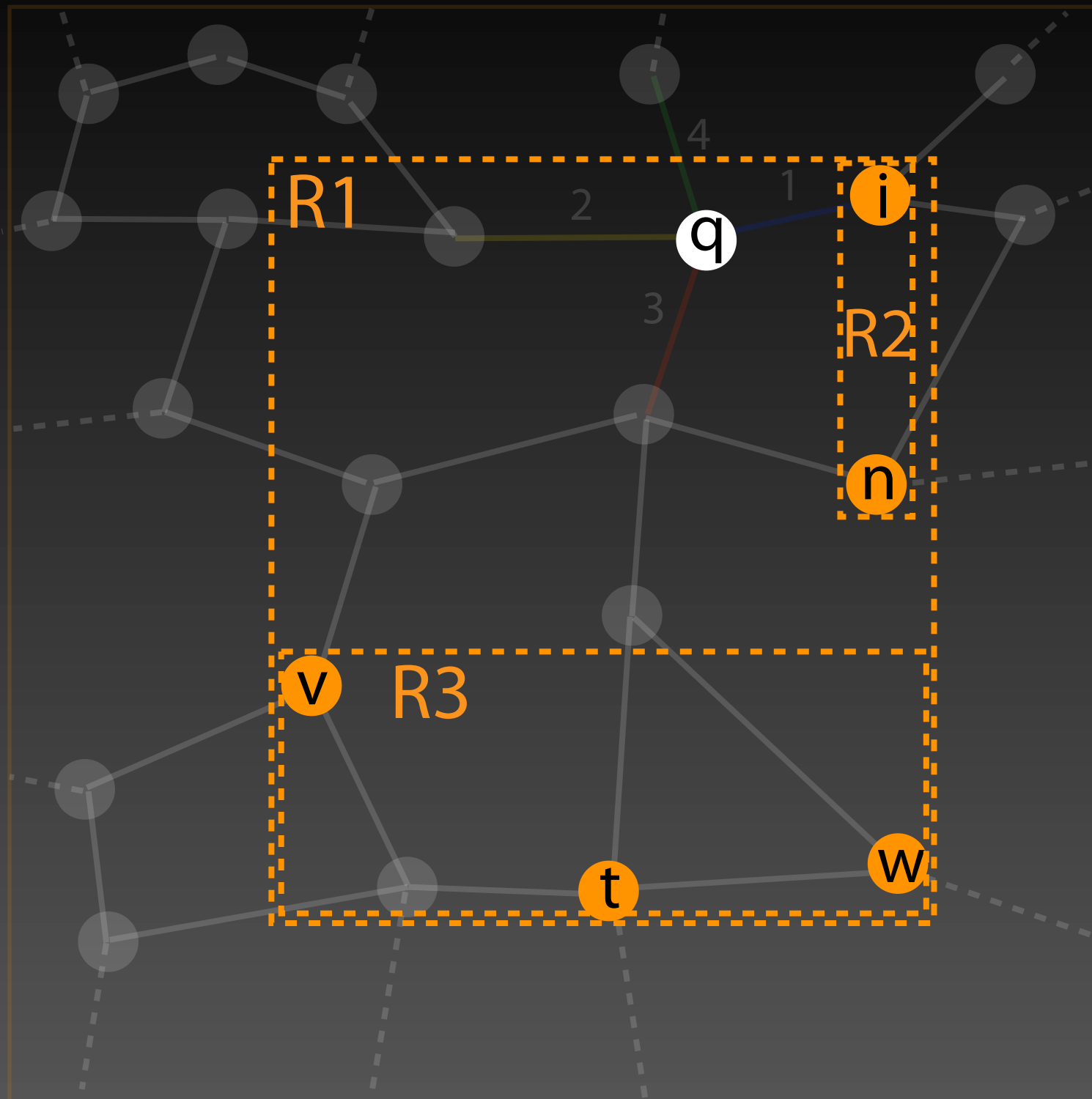
- Find 3 nearest neighbors
- 5 Elements in object set (i,n,t,v,w)

Finding k-Nearest Neighbors

Search Hierarchy



Finding k-Nearest Neighbors



Priority Queues

x: Range or Object

y: Node on Shortest Path/
Status of Refinement

ϵ : Network Distance $\epsilon(x, y)$

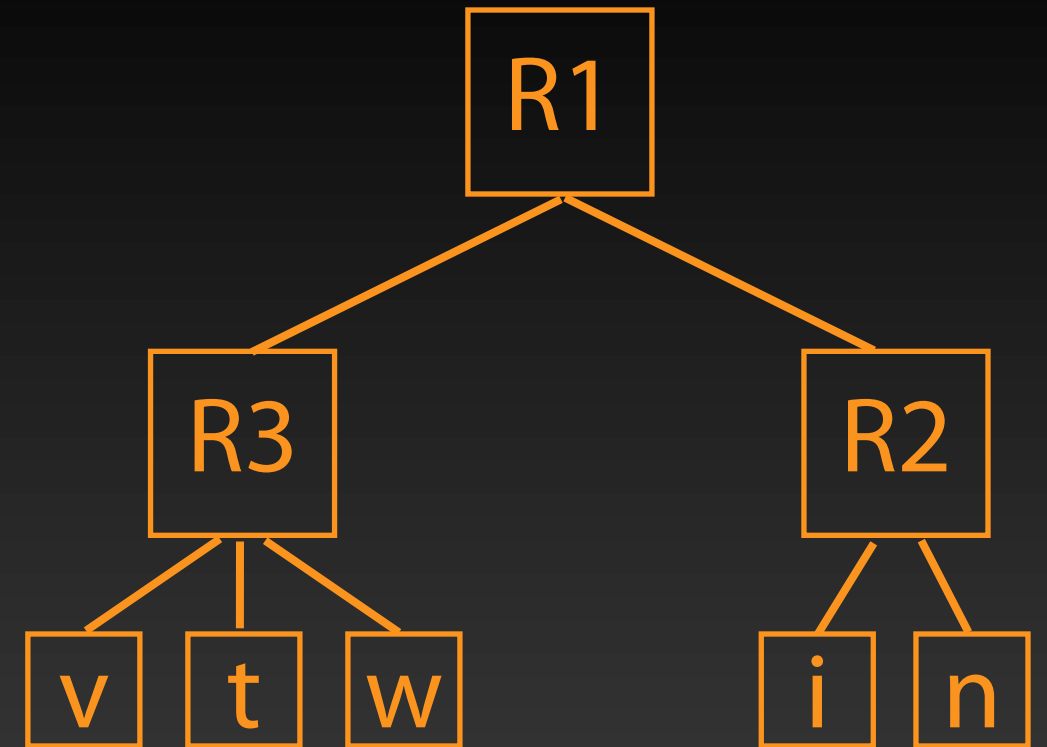
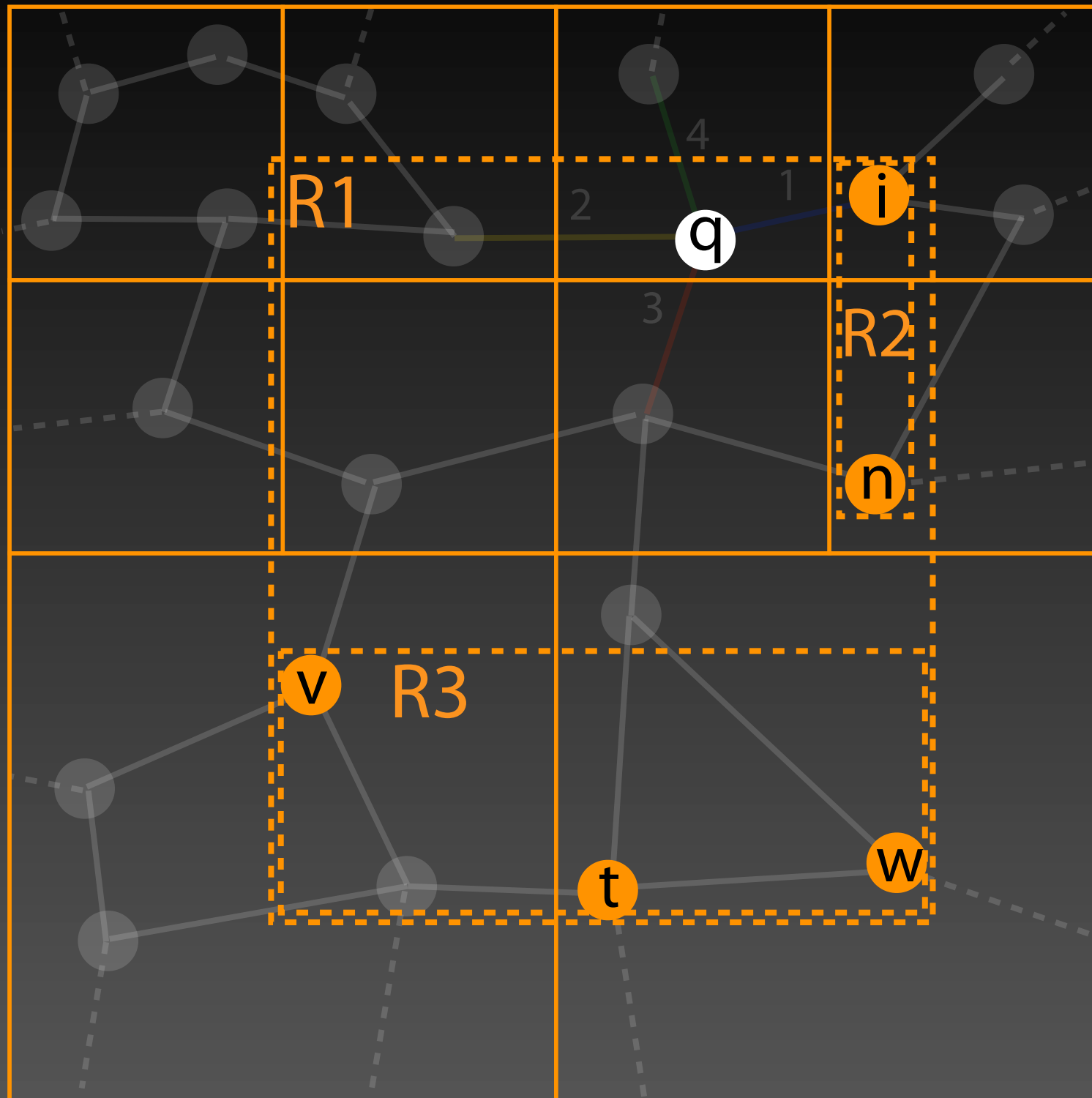
Queue: $x(\delta^-, \delta^+, [y, \epsilon])$

kNN: $z(-, \delta^+) \leq D_k$

Ascending Order

D_k : δ^+ of k-th Nearest Neighbor

Finding k-Nearest Neighbors

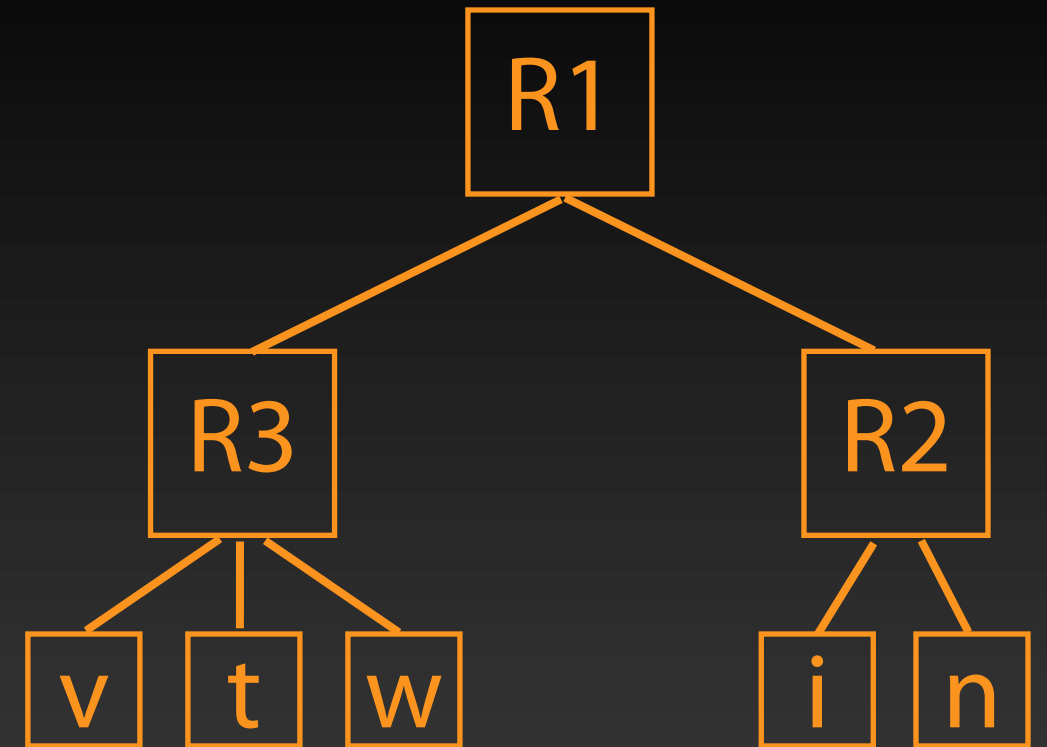
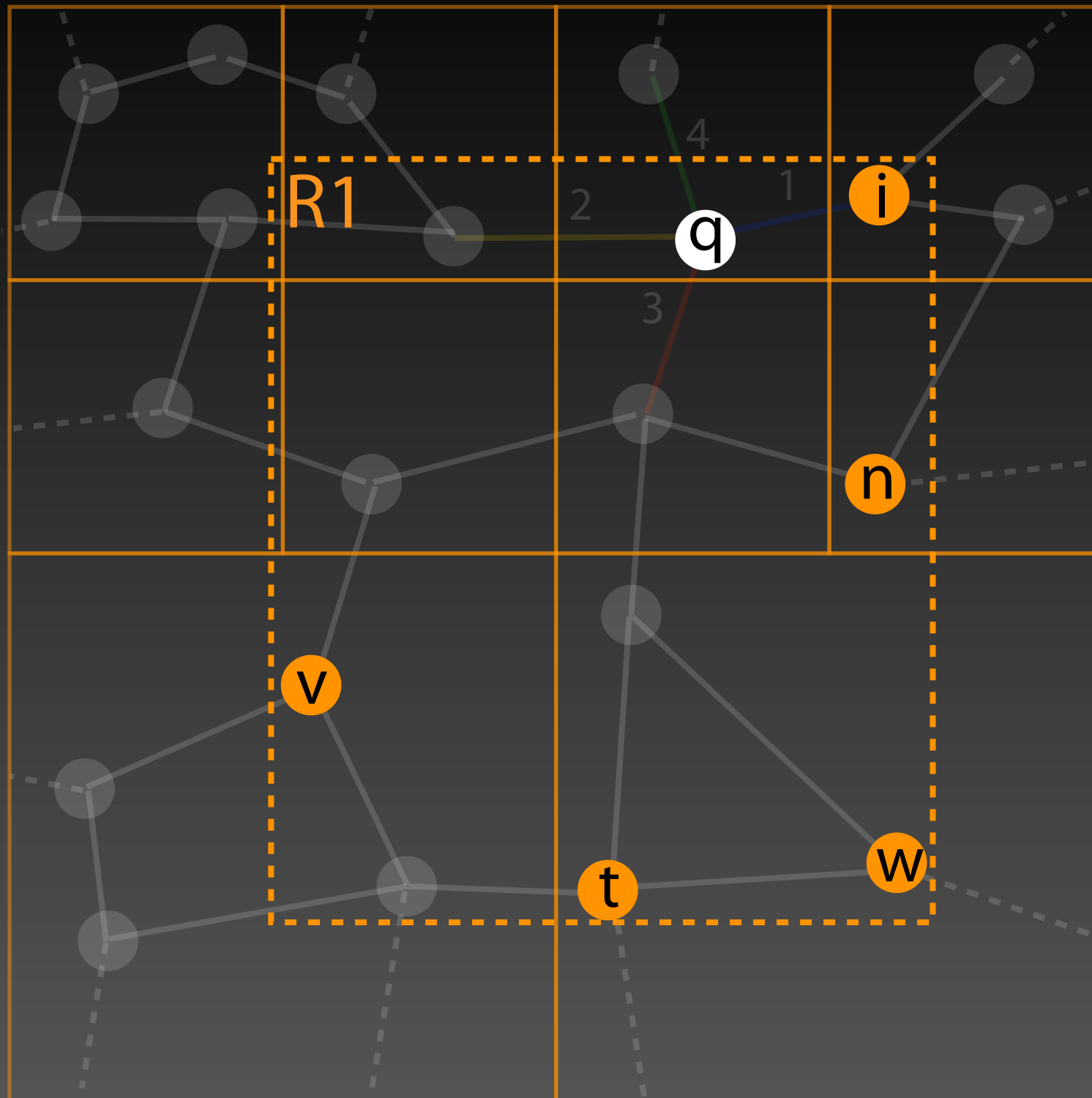


Queue:

kNN:

$\leq \infty$

Finding k-Nearest Neighbors

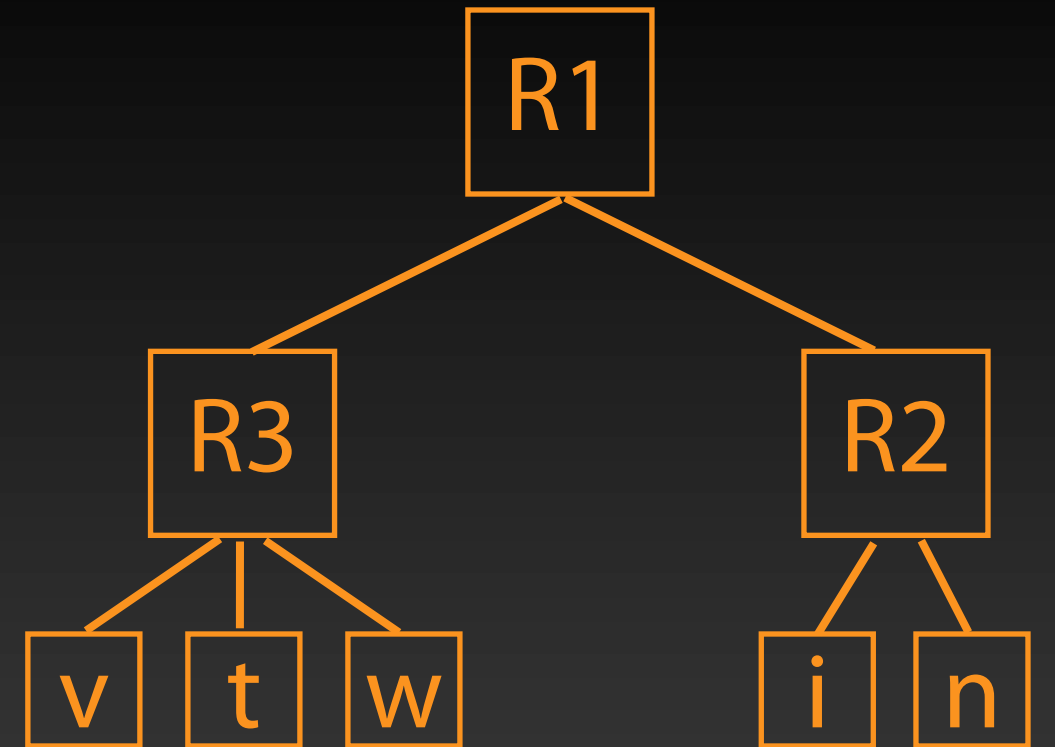
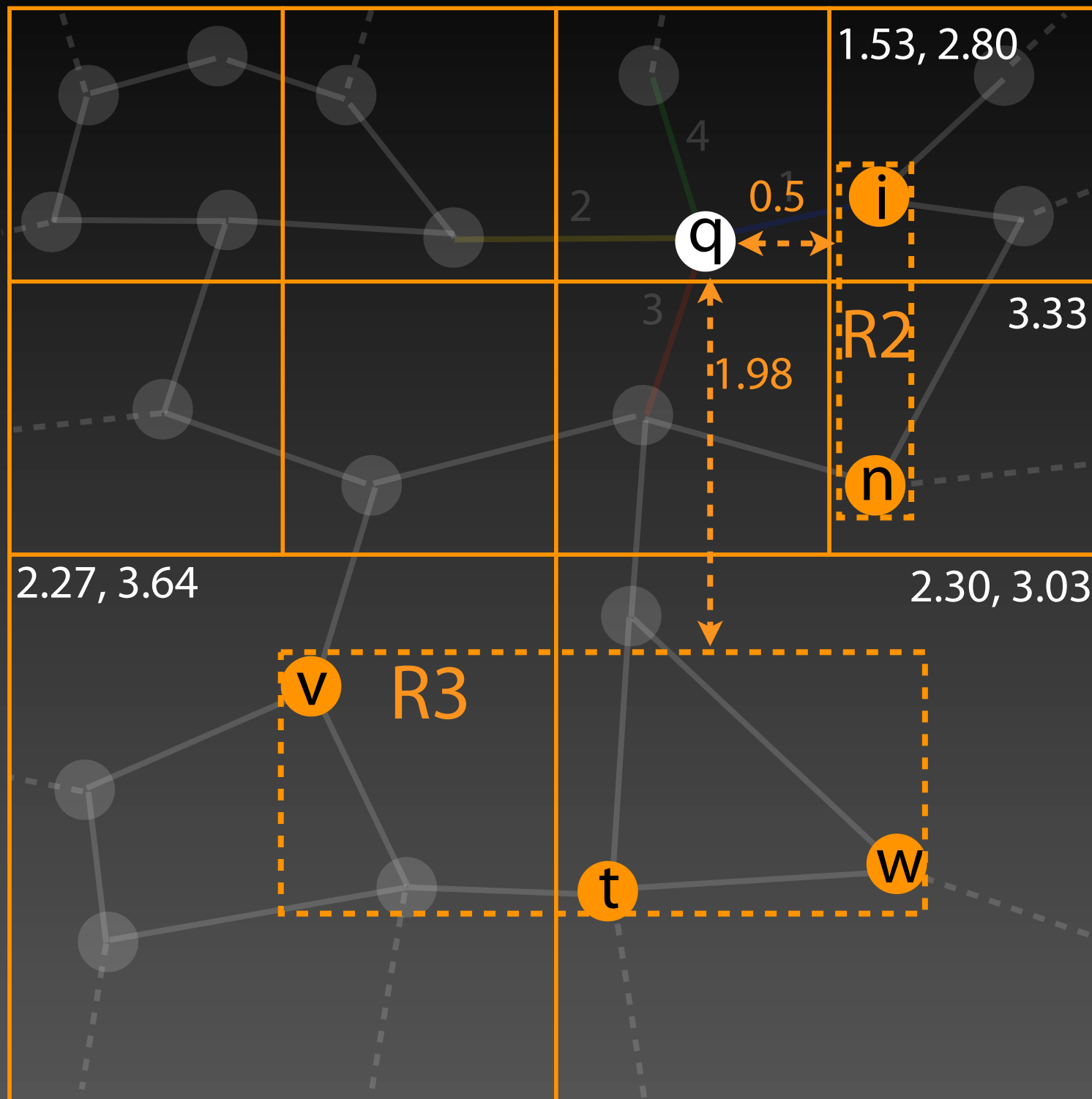


Queue: $R1(0,-)$

kNN:

$\leq \infty$

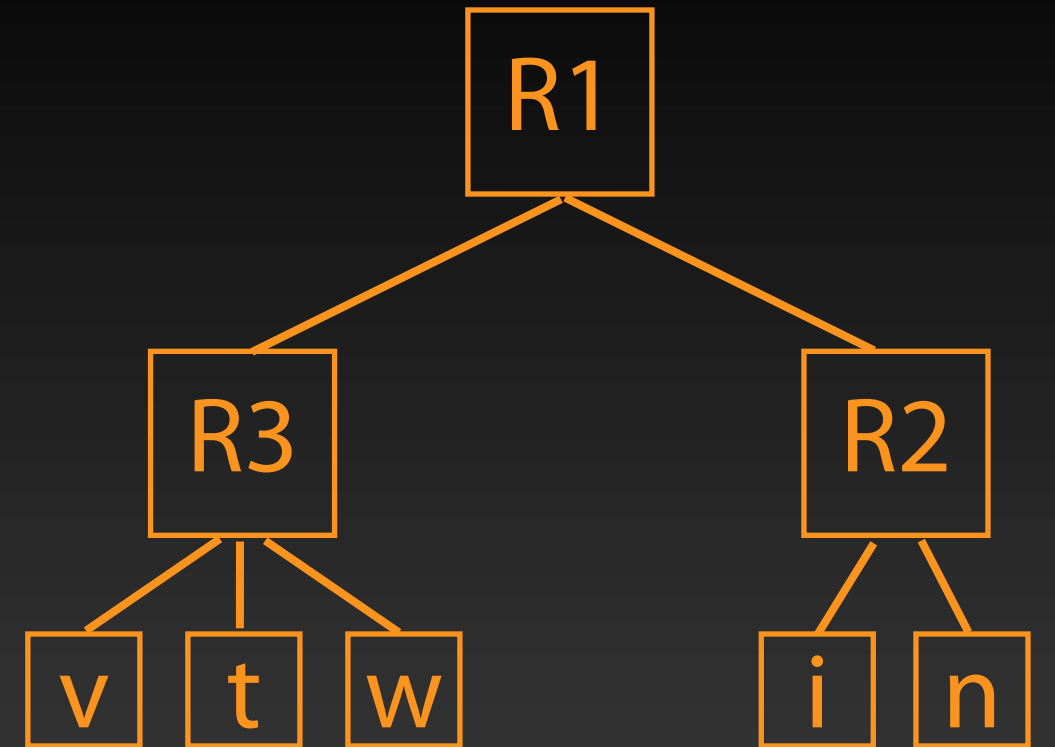
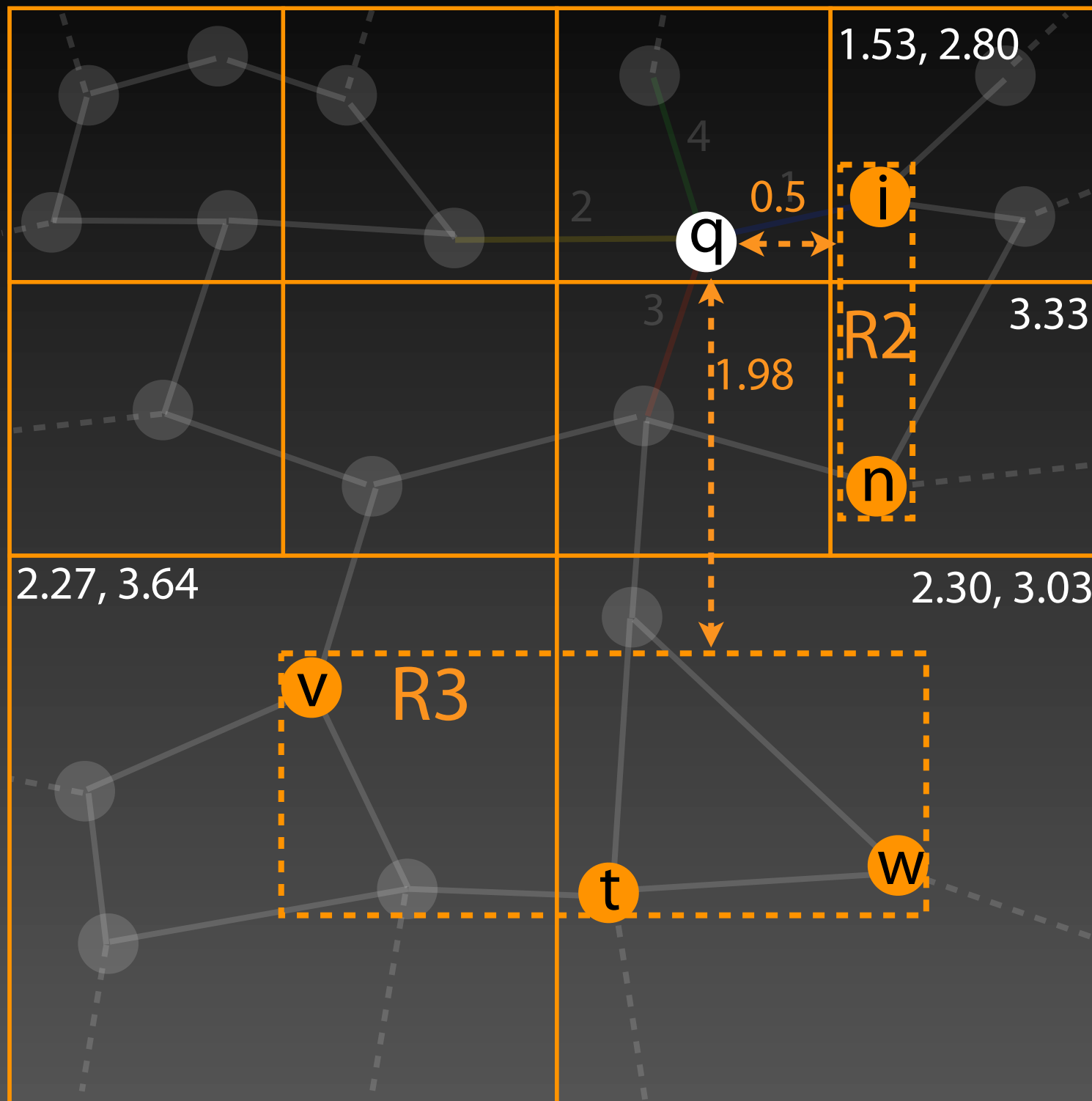
Finding k-Nearest Neighbors



Queue: R2(0.76,-), R3(4.49,-)

kNN: $\leq \infty$

Finding k-Nearest Neighbors

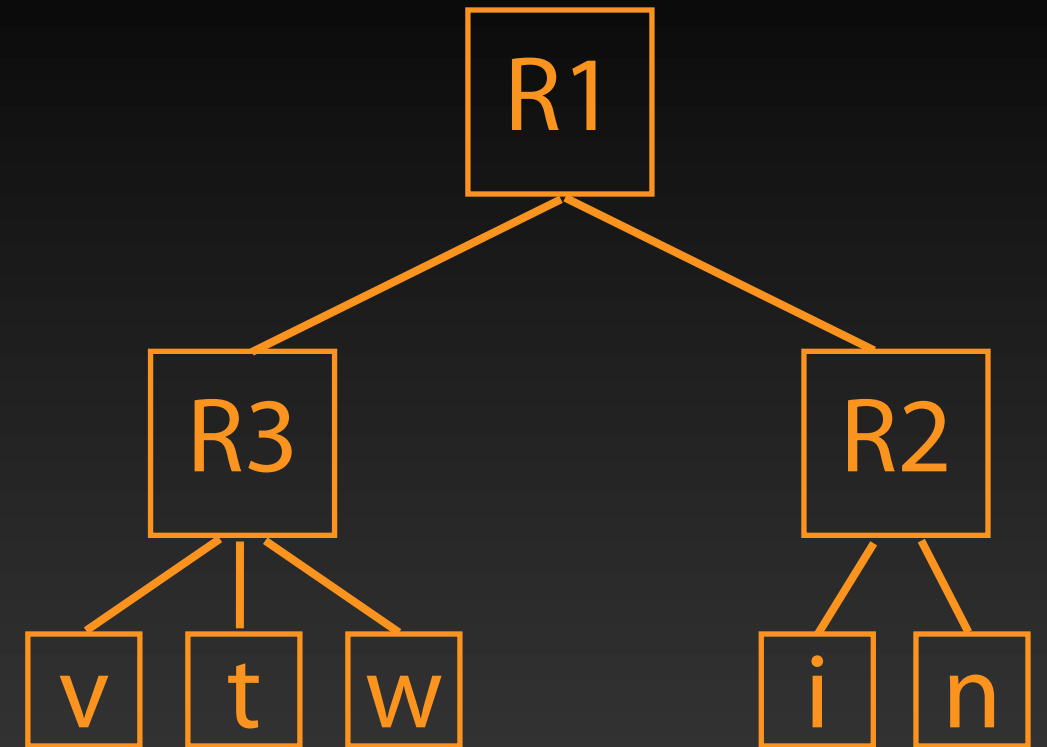
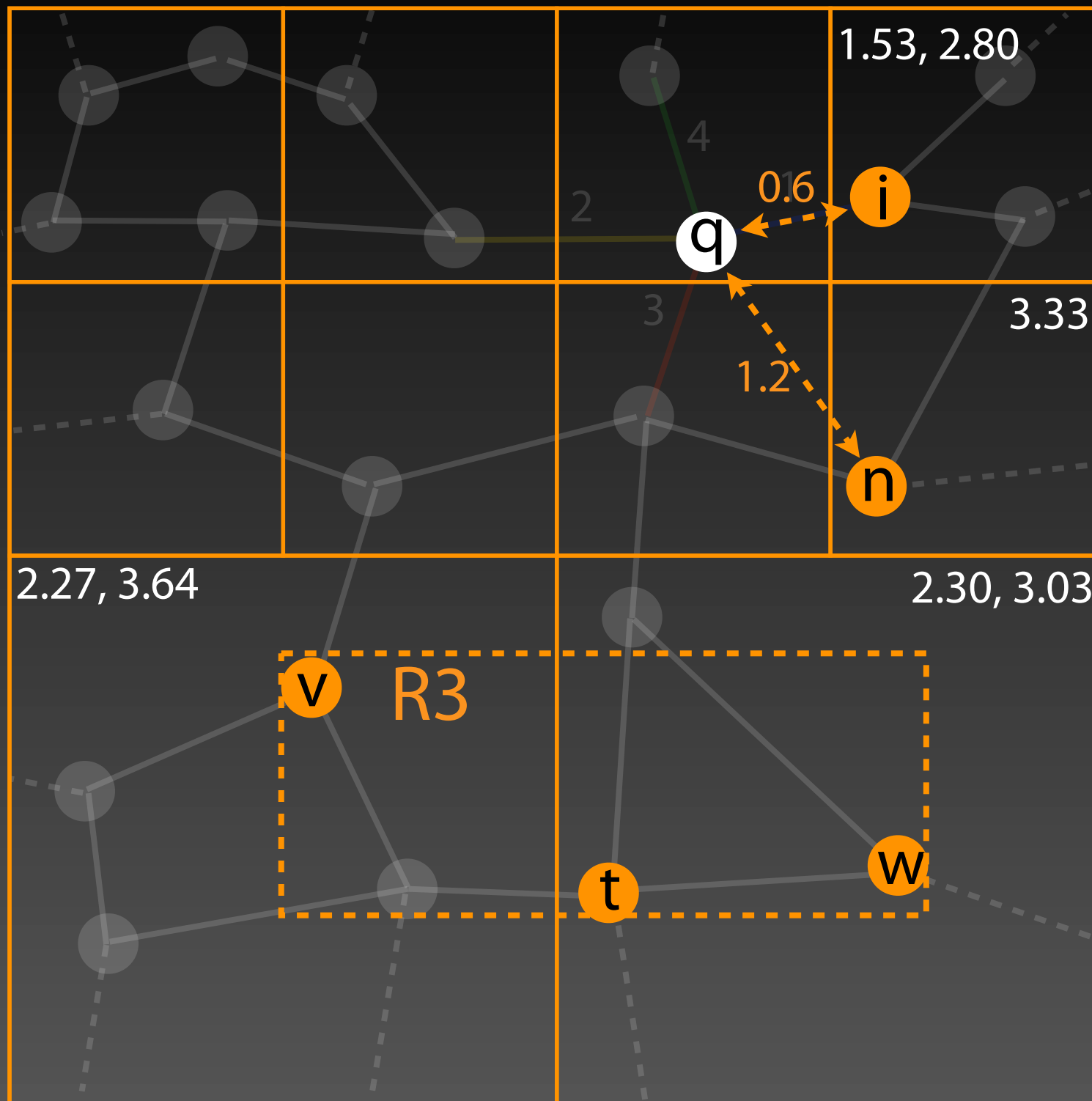


Queue: **R2**(0.76,-), **R3**(4.49,-)

kNN: $\leq \infty$

Current Working Set: **R2**

Finding k-Nearest Neighbors

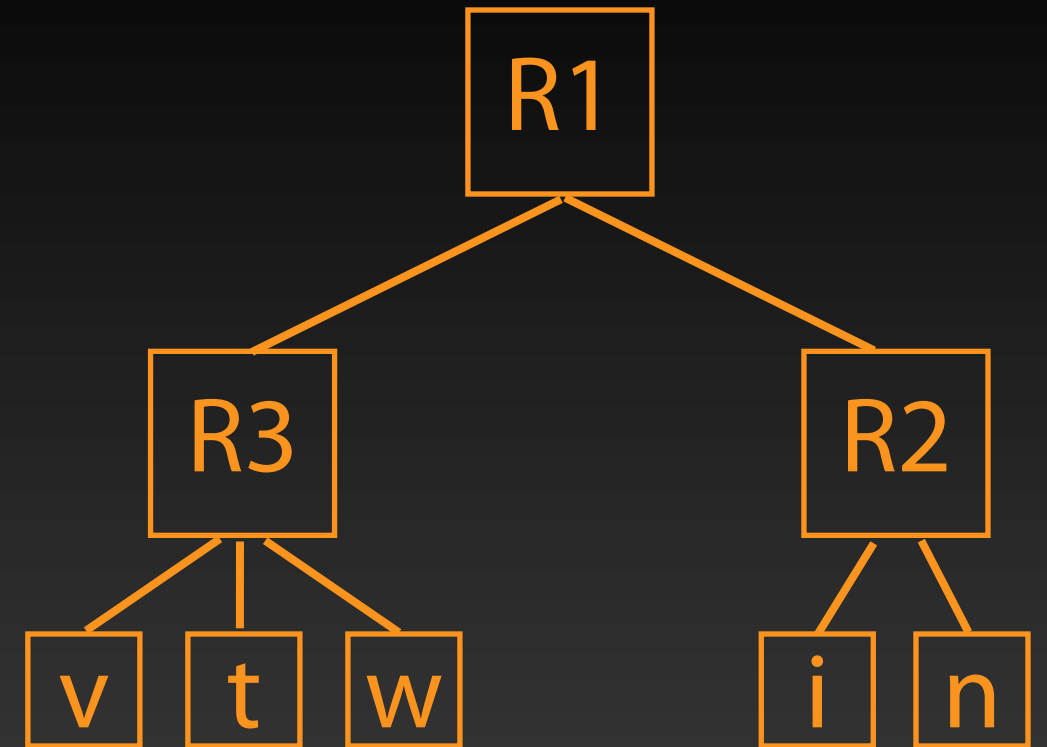
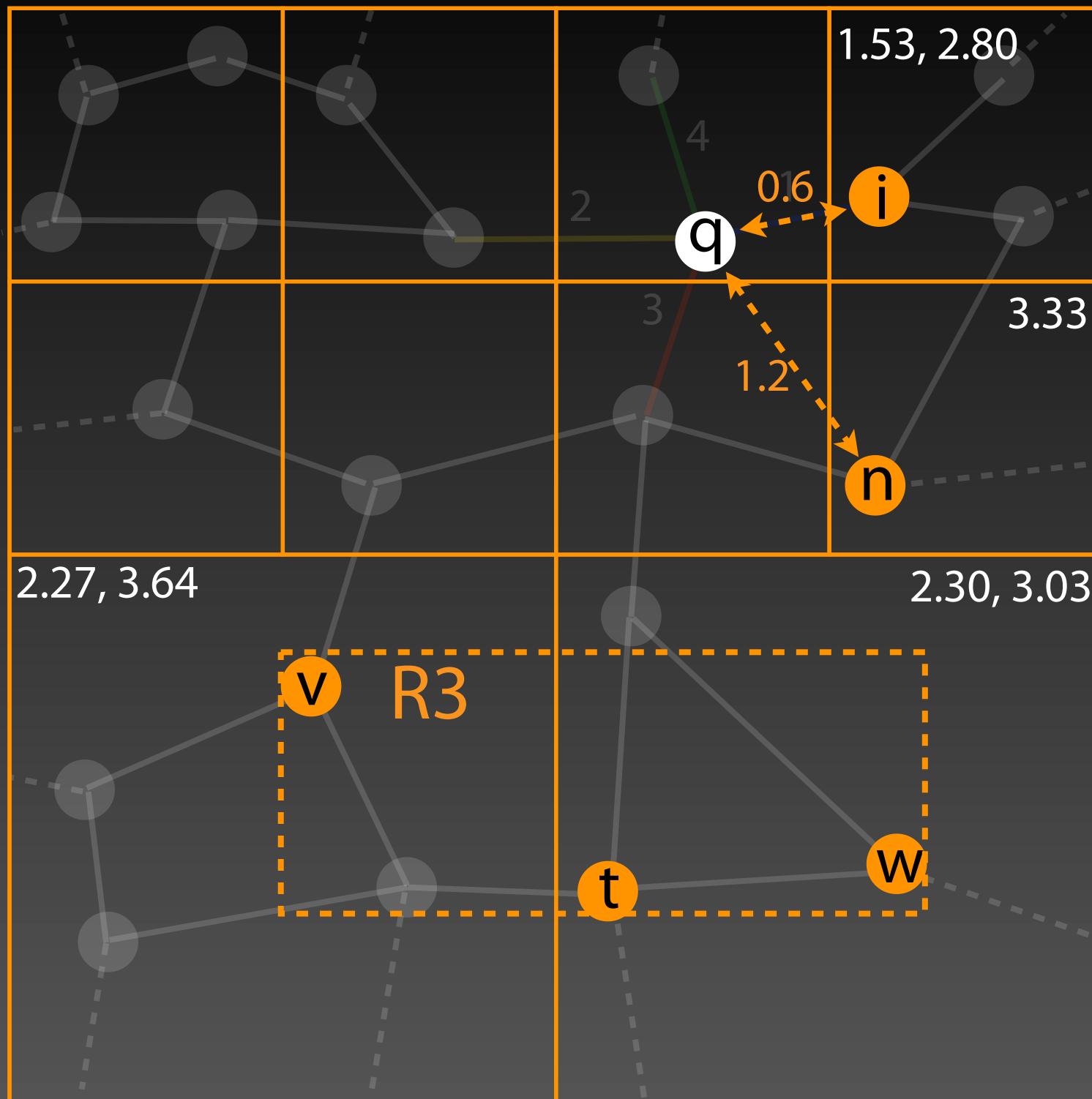


Queue: $i(0.92, 1.68)$, $R3(4.49, -)$

kNN: $i(-, 1.68)$ $\leq \infty$

Current Working Set: $R2$

Finding k-Nearest Neighbors

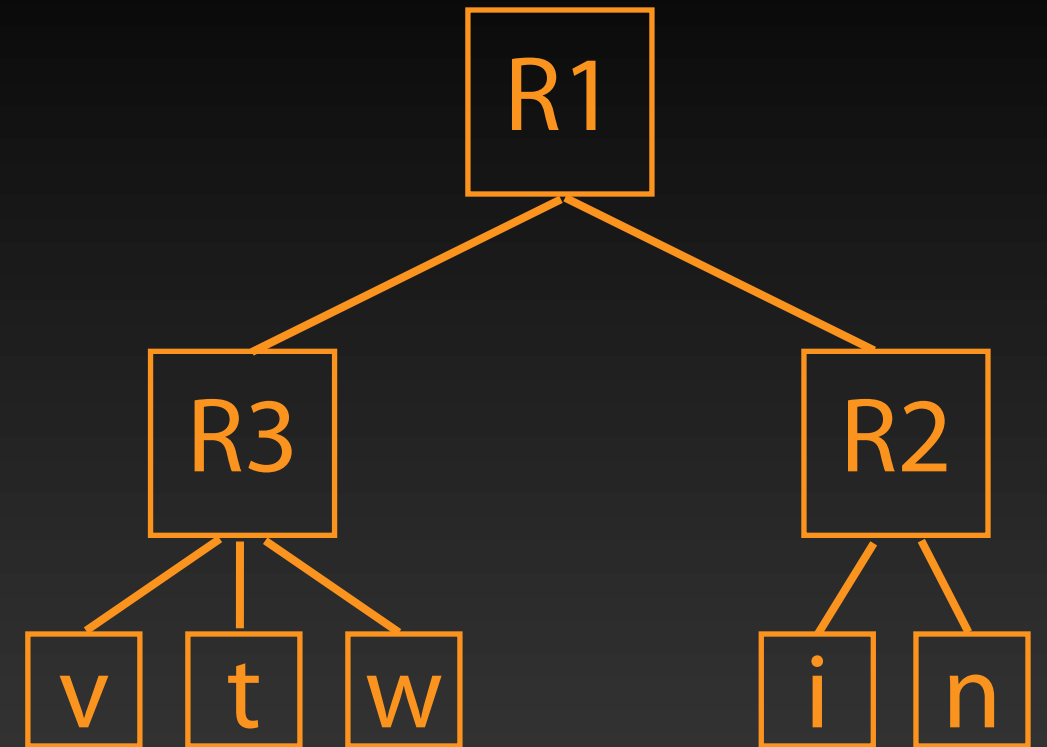
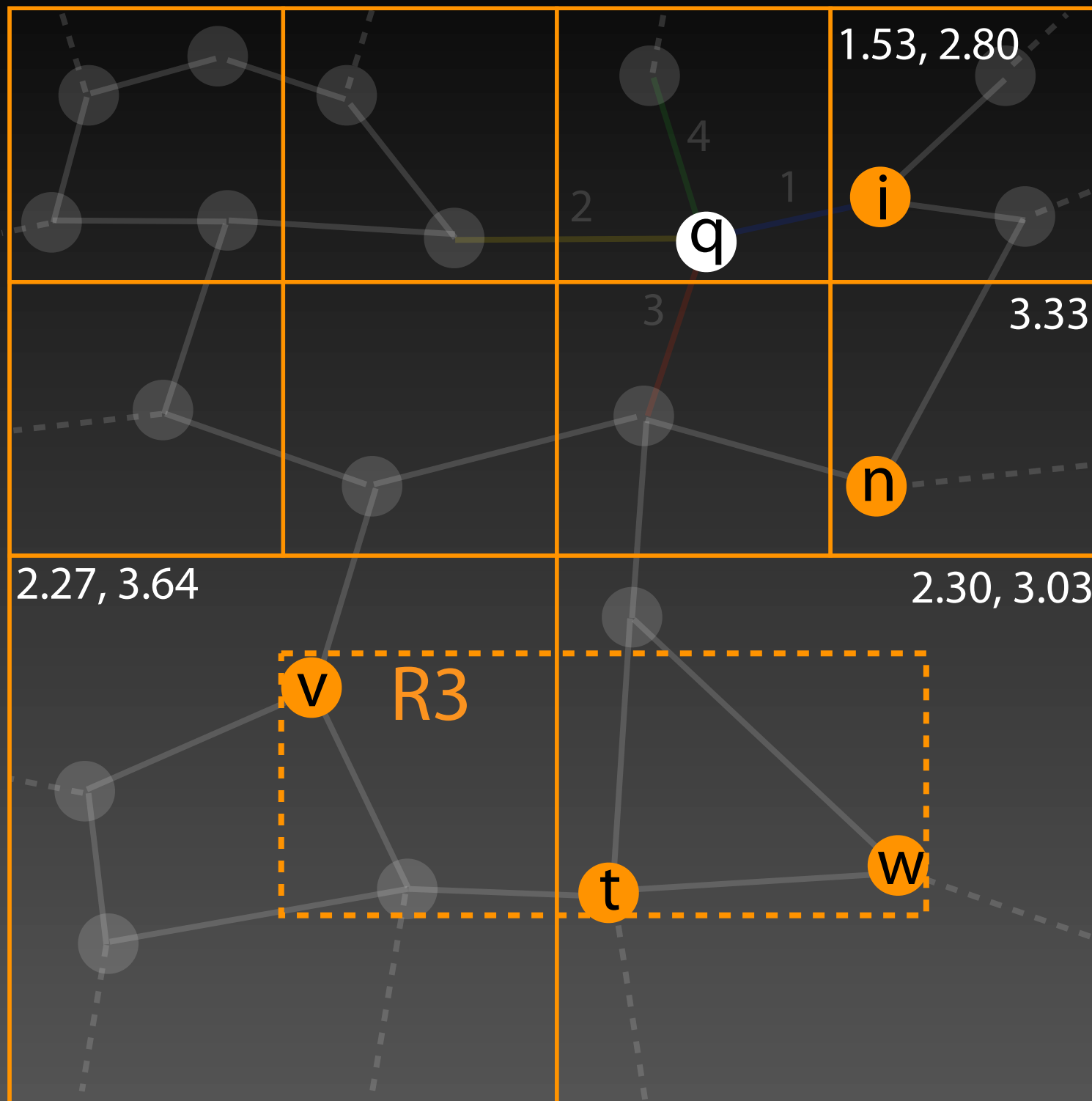


Queue: $i(0.92, 1.68)$, $n(4, 4)$,
 $R3(4.49, -)$

kNN: $i(-, 1.68)$, $n(-, 4) \leq \infty$

Current Working Set: **R2**

Finding k-Nearest Neighbors

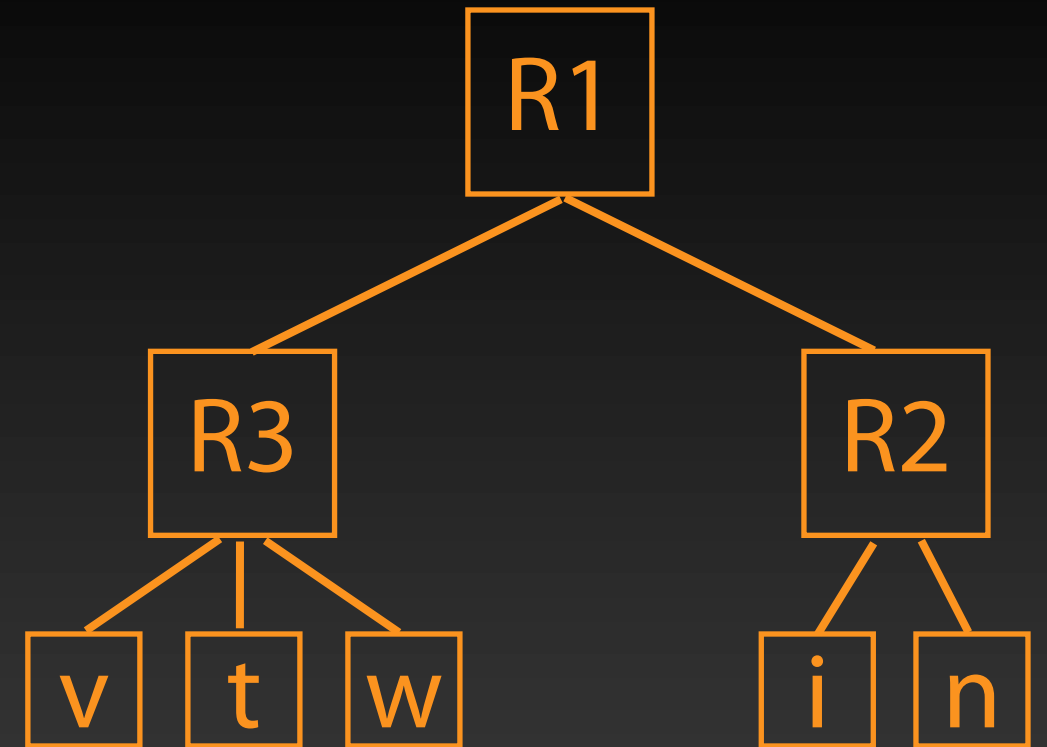
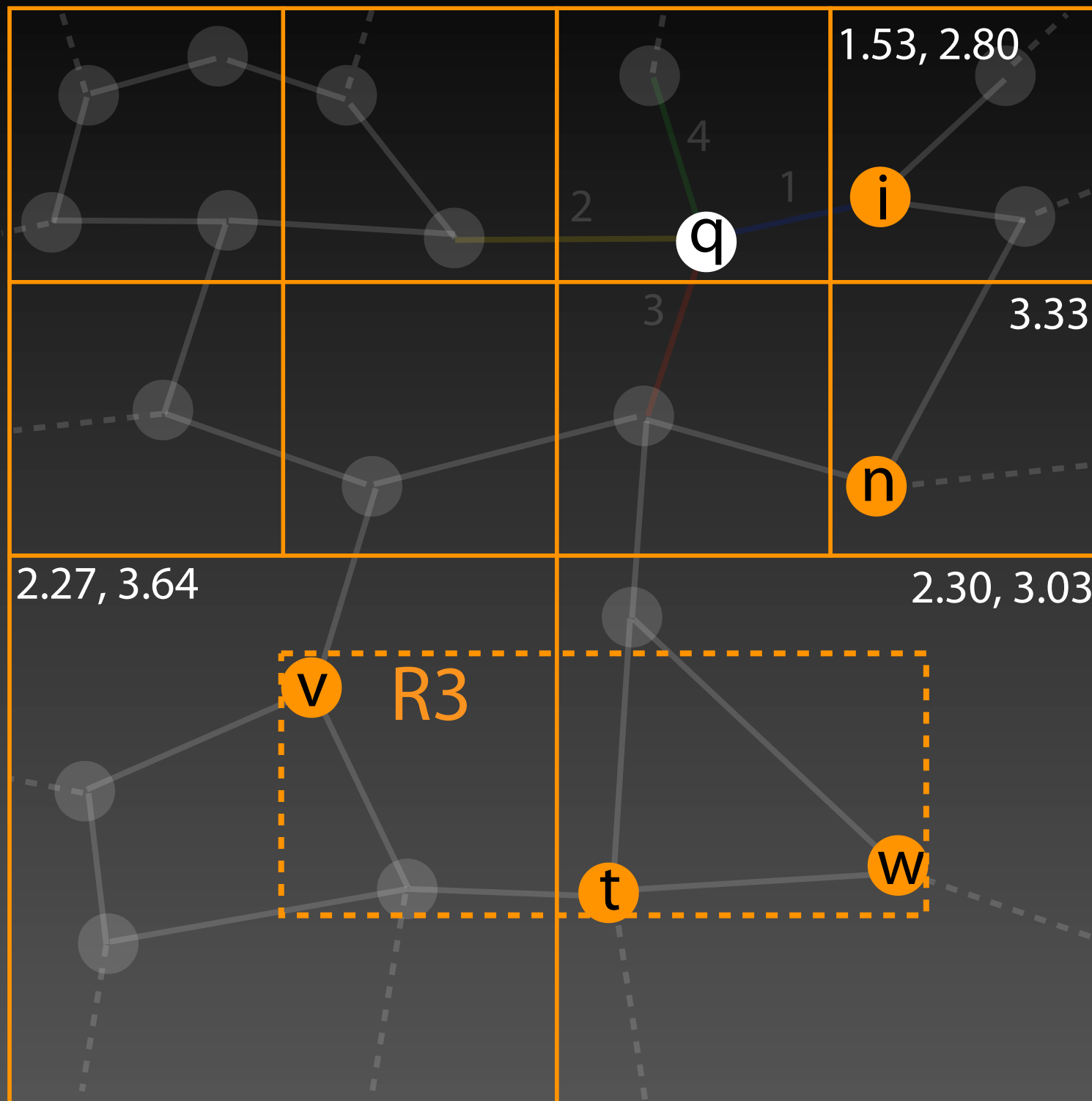


Queue: $n(4,4)$, $R3(4.49,-)$

kNN: $i(-,1.68)$, $n(-,4) \leq \infty$

Current Working Set: i

Finding k-Nearest Neighbors

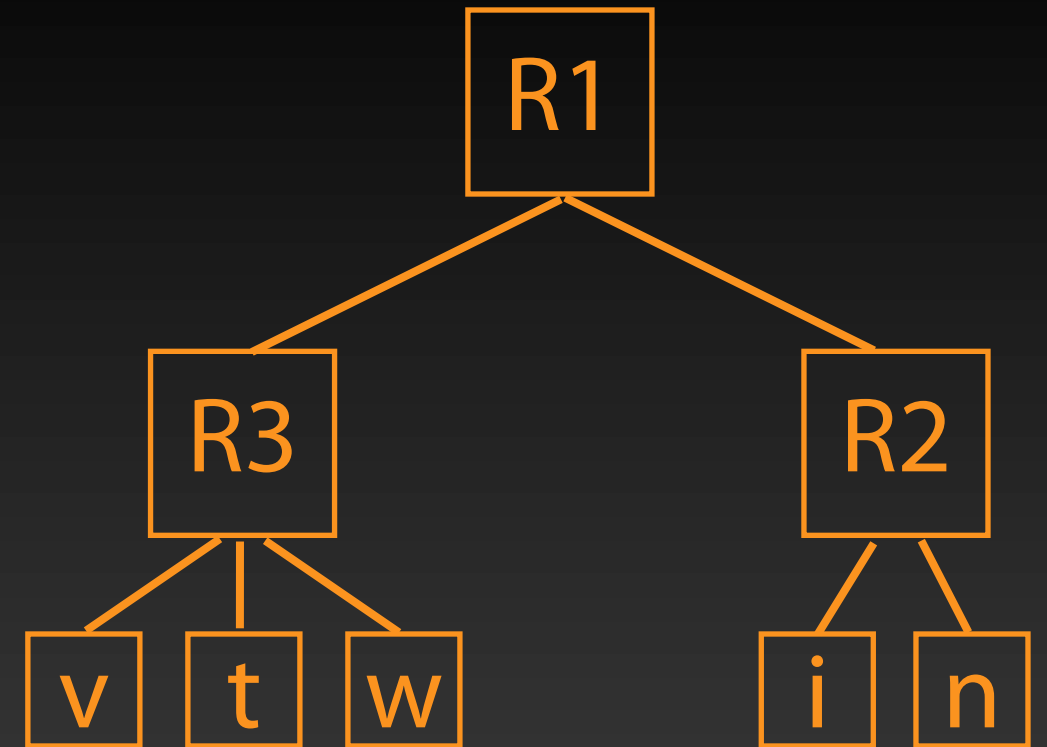
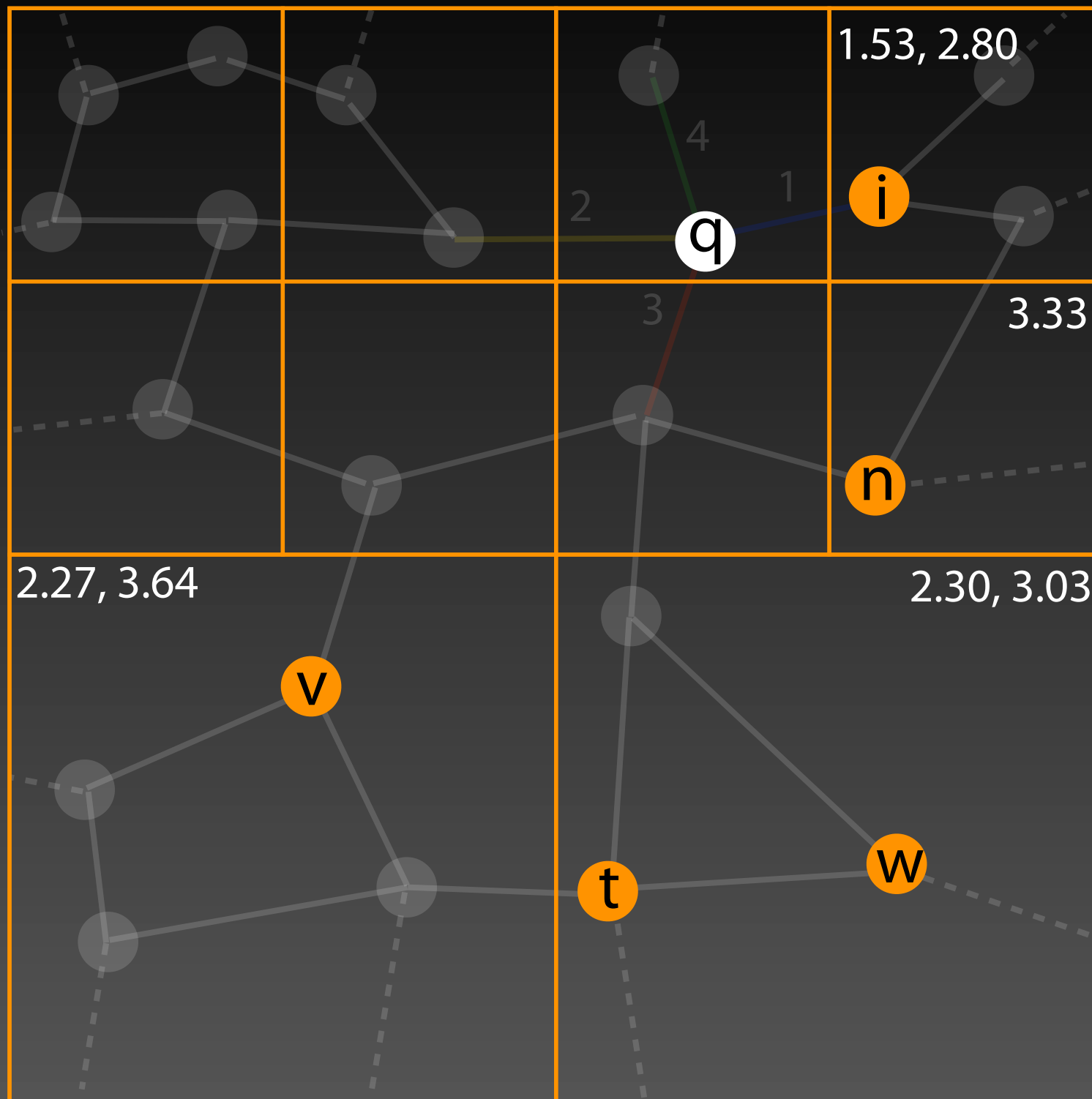


Queue: R3(4.49,-)

kNN: $i(-, 1.68), n(-, 4) \leq \infty$

Current Working Set: n

Finding k-Nearest Neighbors

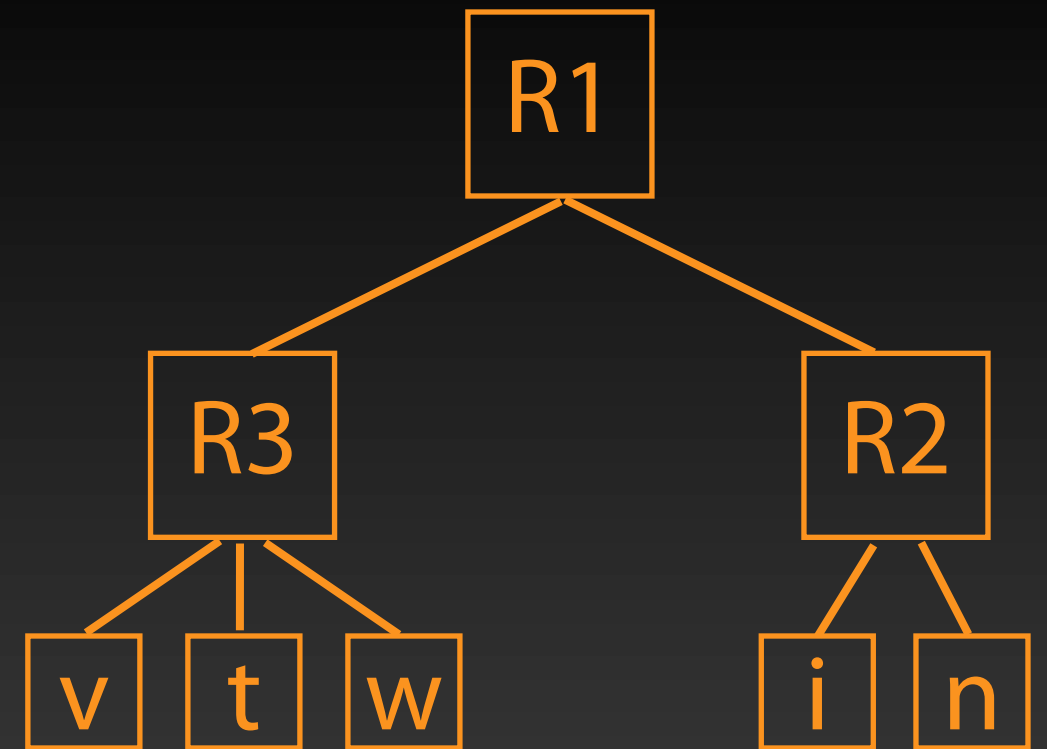
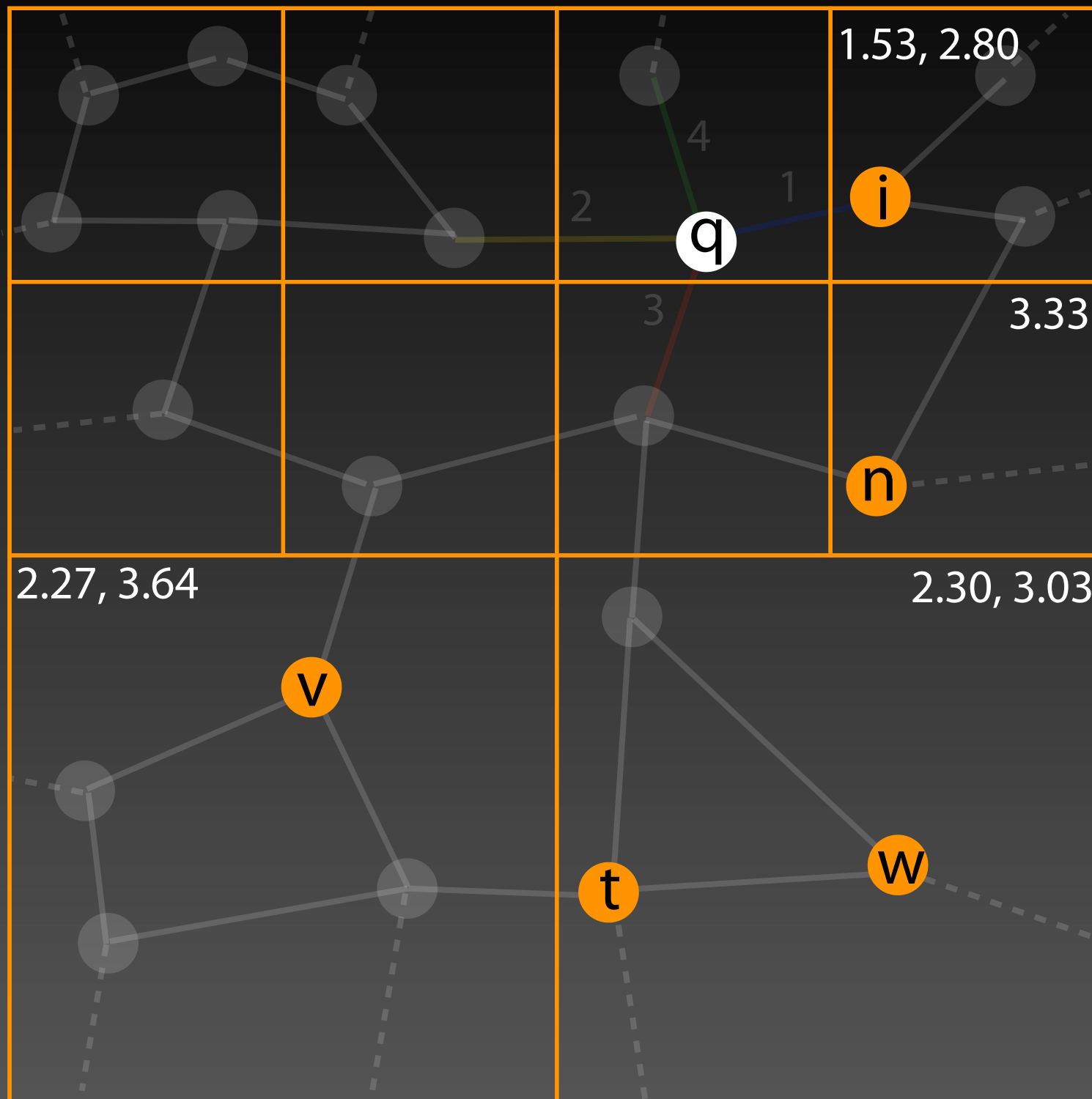


Queue: $v(8.88, 14.23)$

kNN: $i(-, 1.68), n(-, 4),$
 $v(-, 14.23) \leq 14.23$

Current Working Set: **R3**

Finding k-Nearest Neighbors

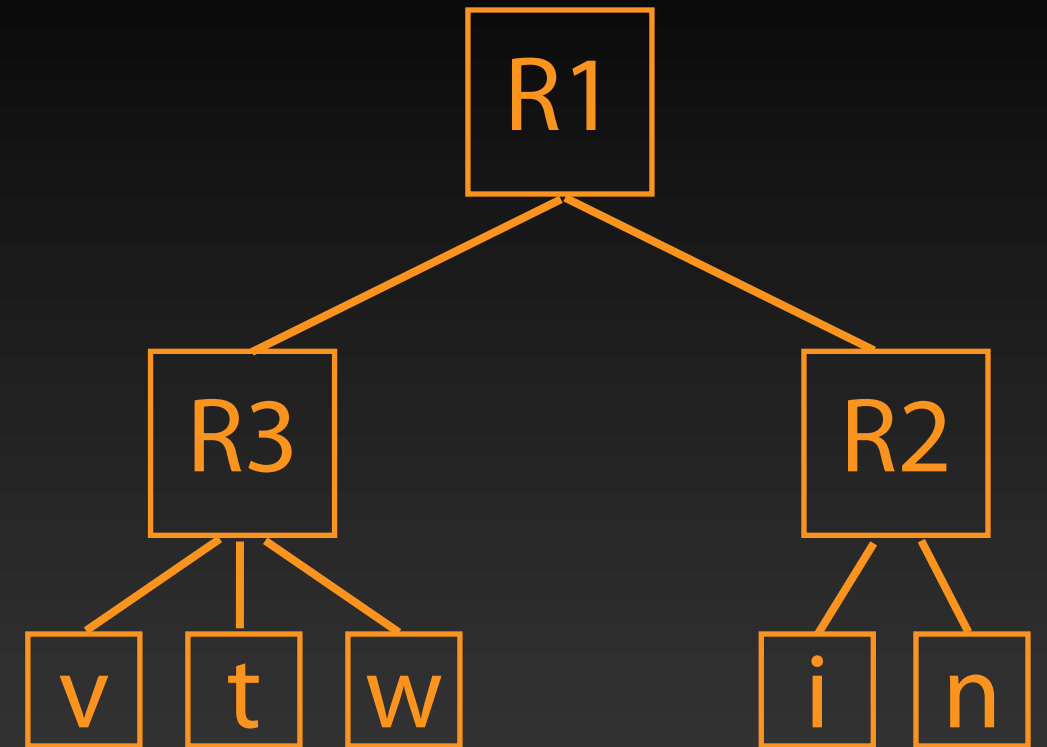
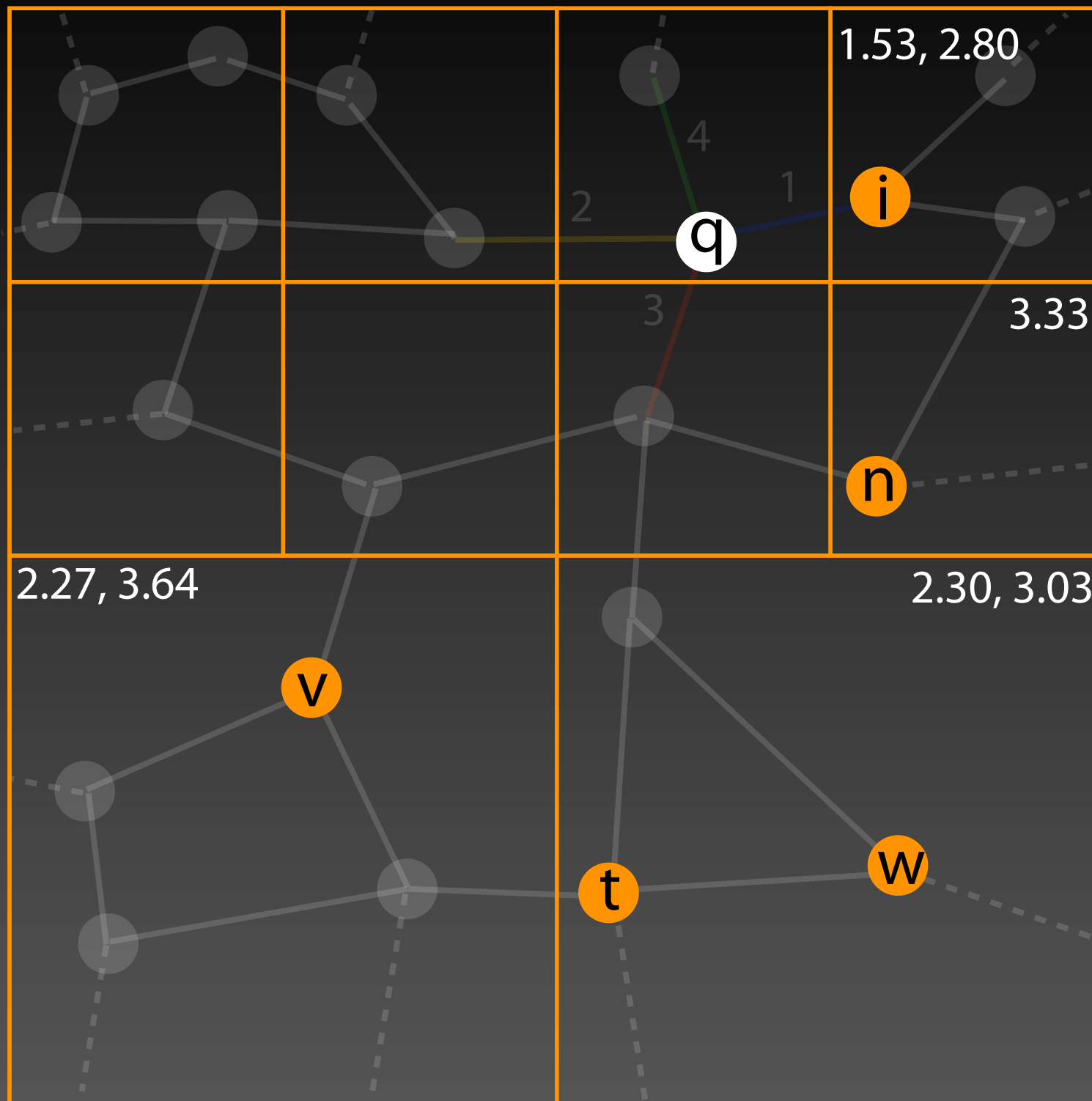


Queue: $t(7.00, 9.24)$,
 $v(8.88, 14.23)$

kNN: $i(-, 1.68)$, $n(-, 4)$,
 $t(-, 9.24) \leq 9.24$

Current Working Set: **R3**

Finding k-Nearest Neighbors

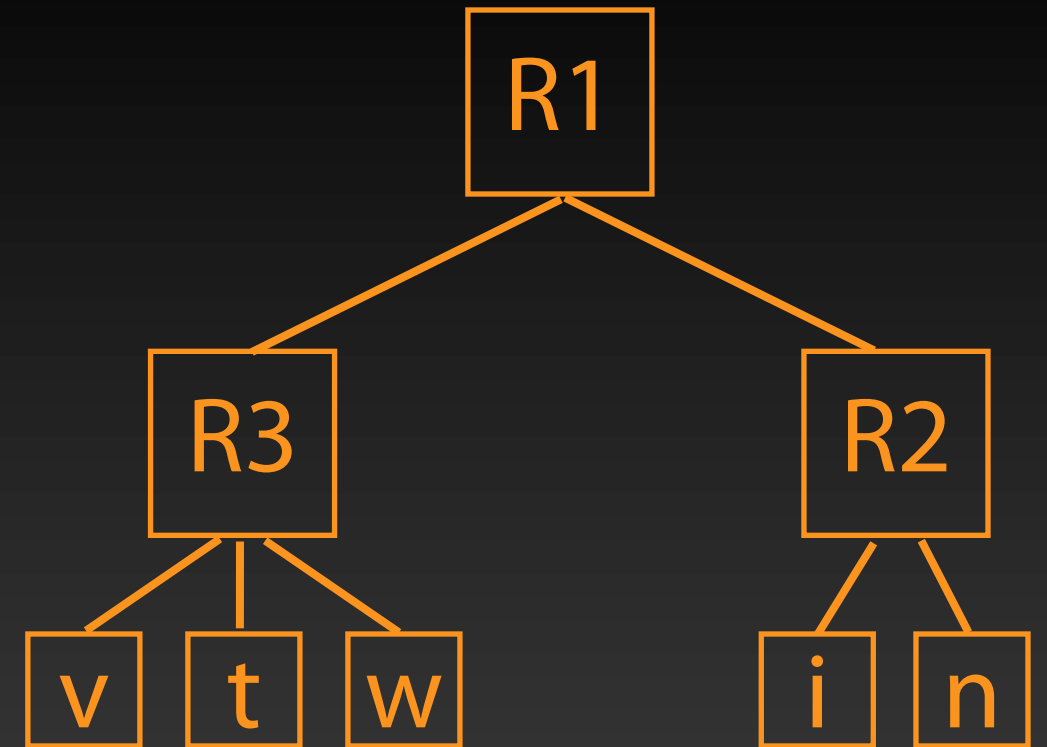
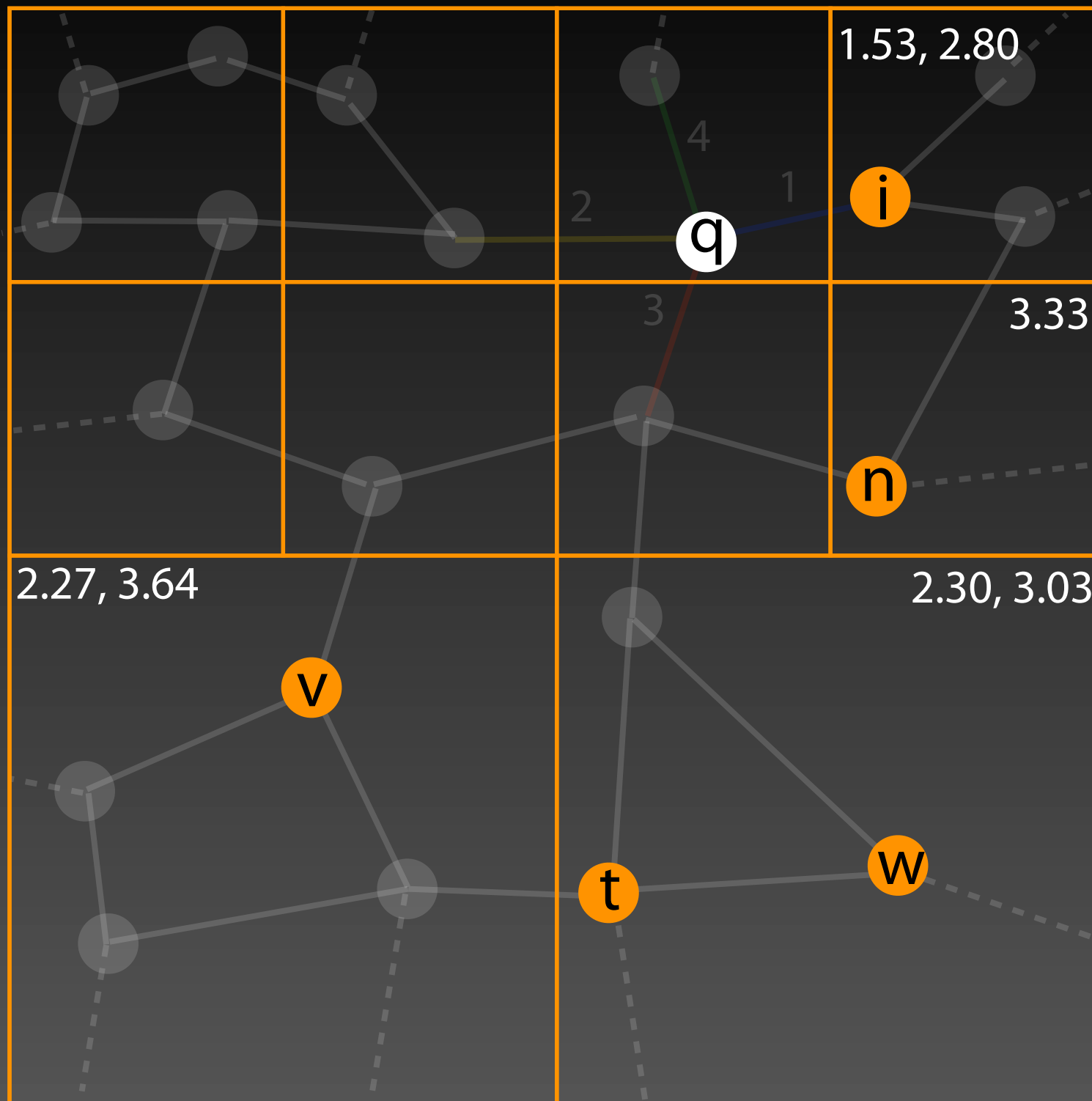


Queue: $w(6.95, 9.15)$, $t(7.00, 9.24)$
 $v(8.88, 14.23)$

kNN: $i(-, 1.68)$, $n(-, 4)$,
 $w(-, 9.15) \leq 9.15$

Current Working Set:

Finding k-Nearest Neighbors

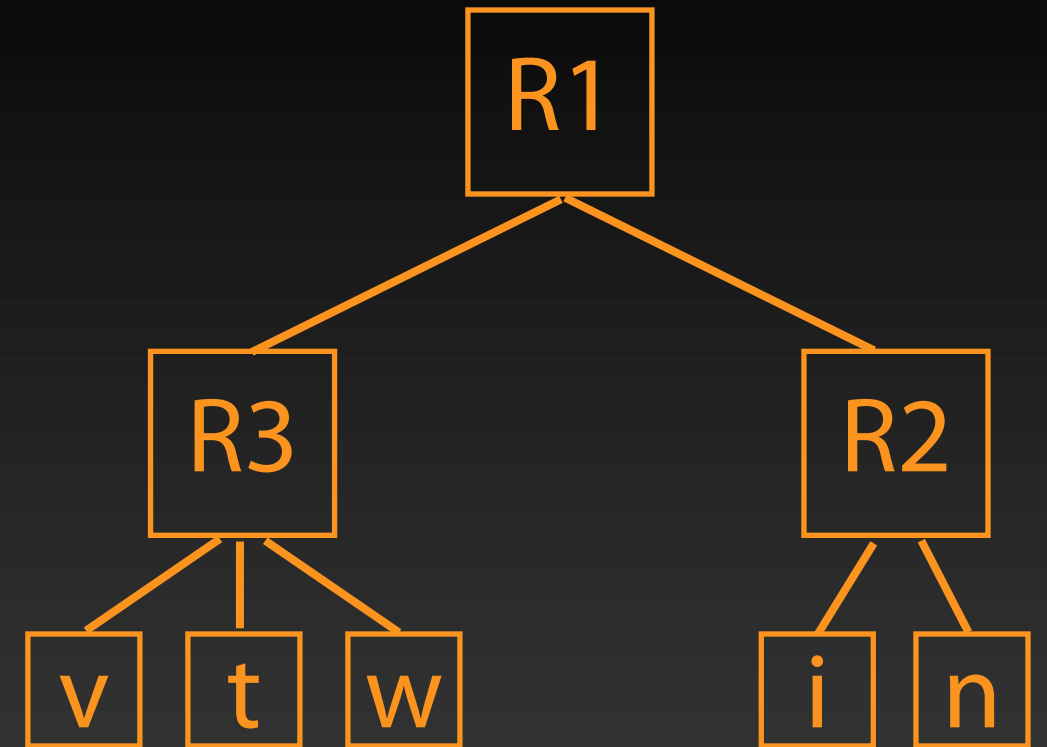
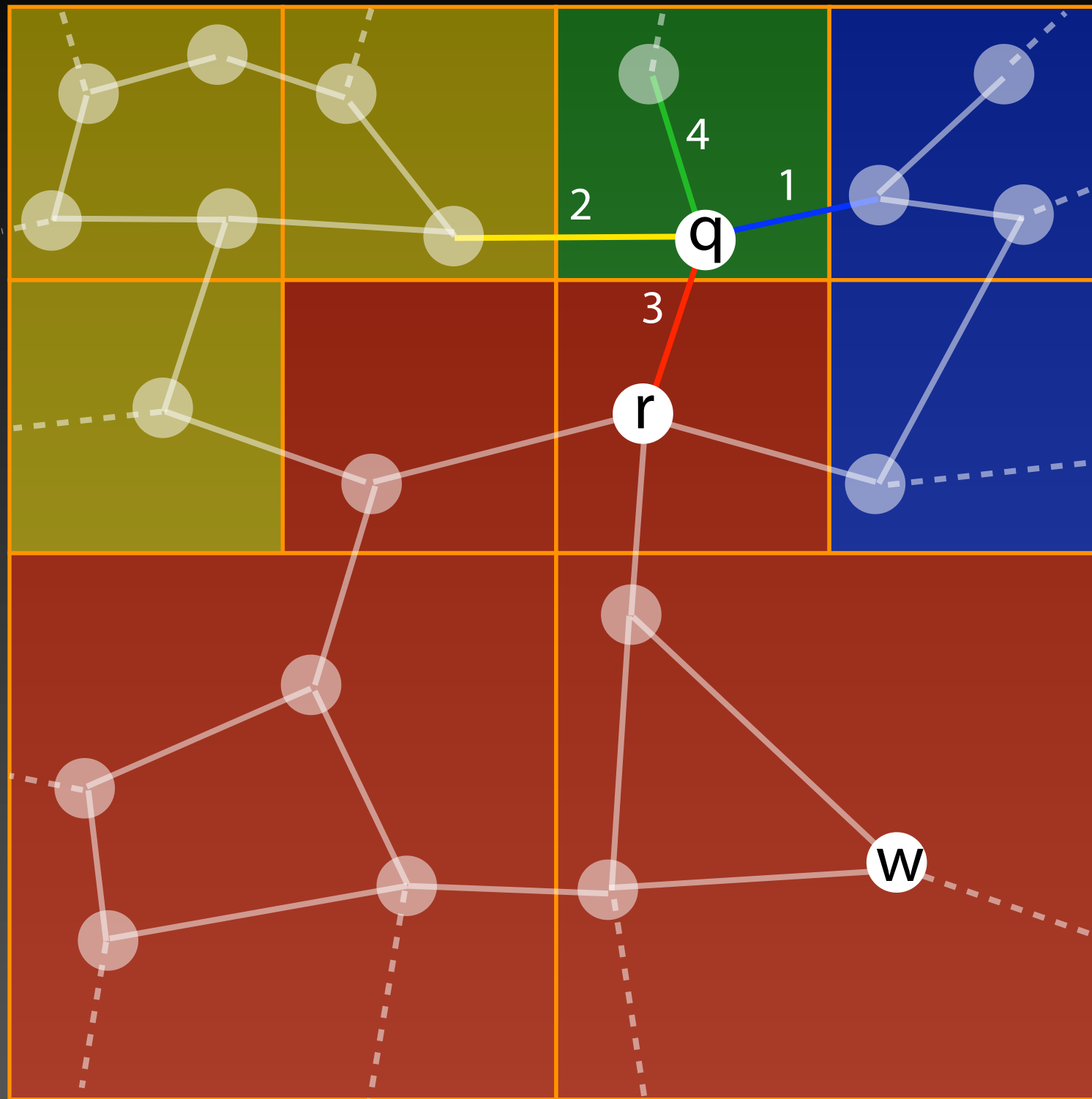


Queue: $w(6.95, 9.15)$, $t(7.00, 9.24)$
 $v(8.88, 14.23)$

kNN: $i(-, 1.68)$, $n(-, 4)$,
 $w(-, 9.15) \leq 9.15$

Current Working Set: w

Finding k-Nearest Neighbors

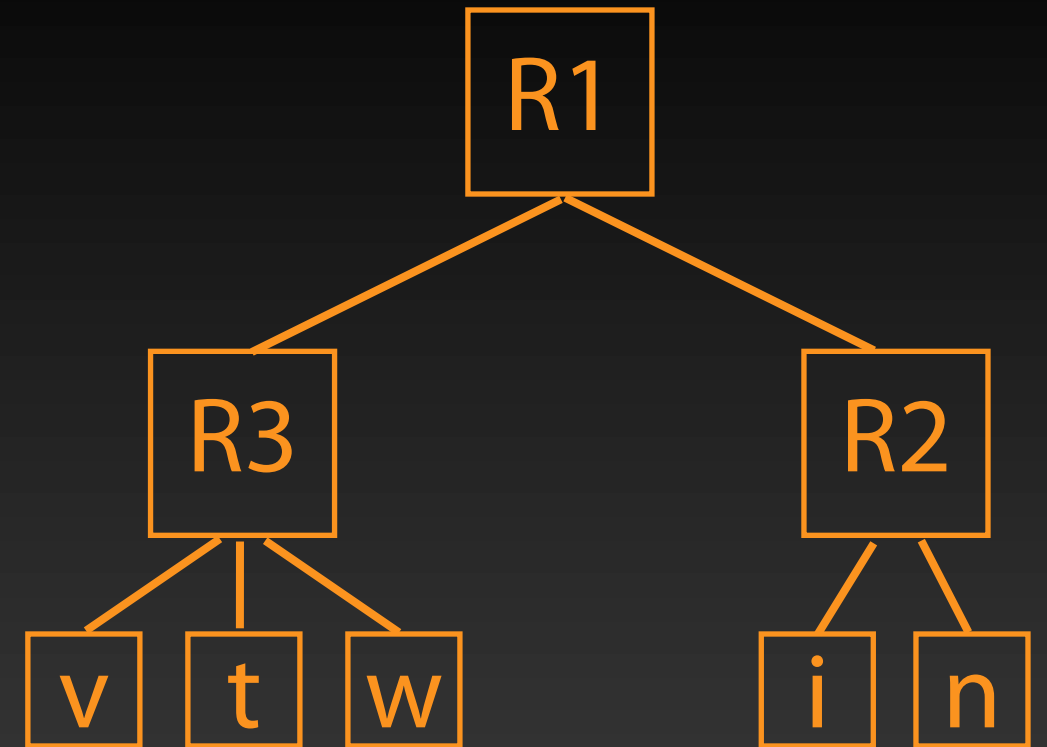
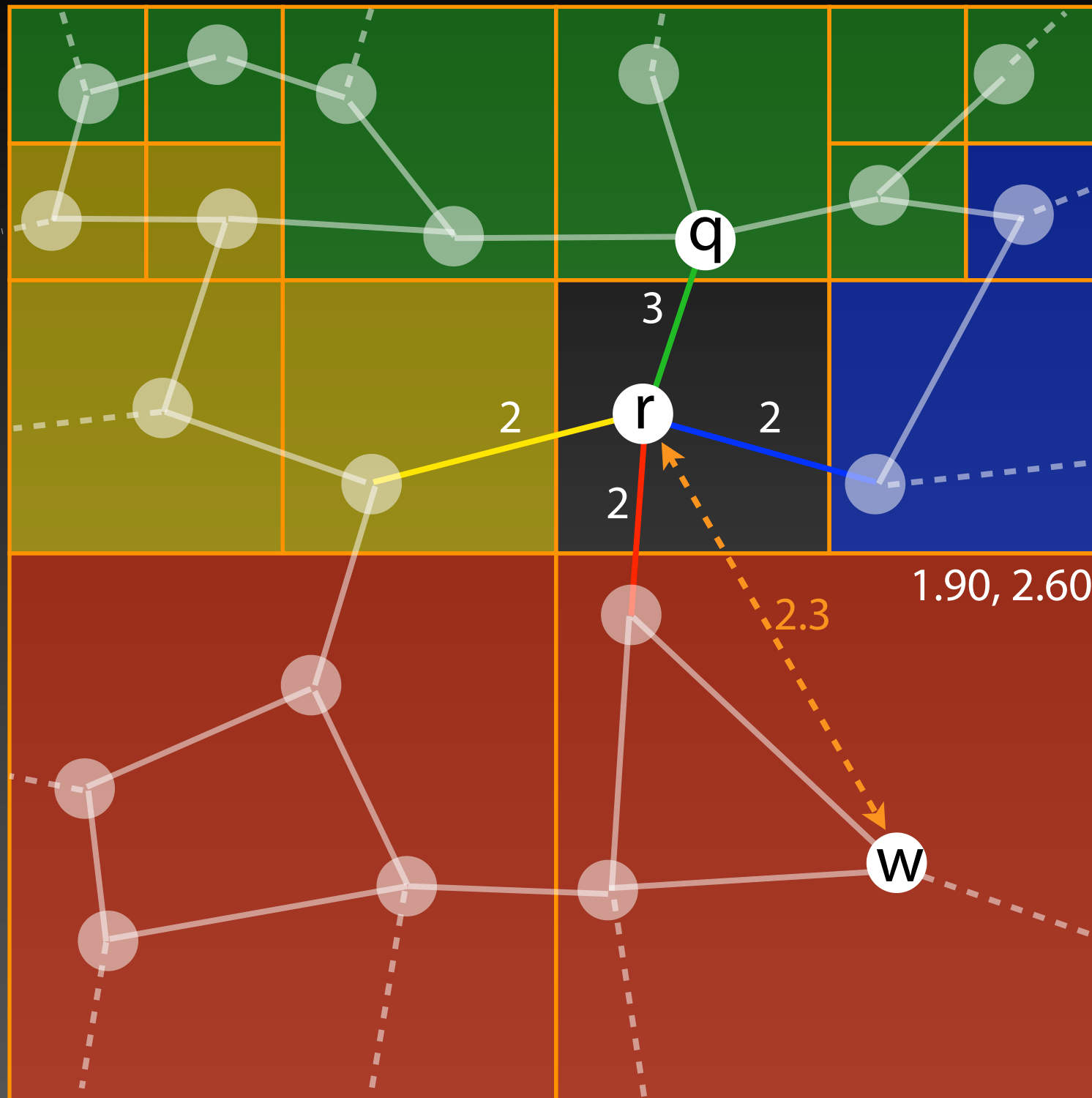


Queue: $w(6.95, 9.15)$, $t(7.00, 9.24)$
 $v(8.88, 14.23)$

kNN: $i(-, 1.68)$, $n(-, 4)$,
 $w(-, 9.15) \leq 9.15$

Current Working Set: w

Finding k-Nearest Neighbors

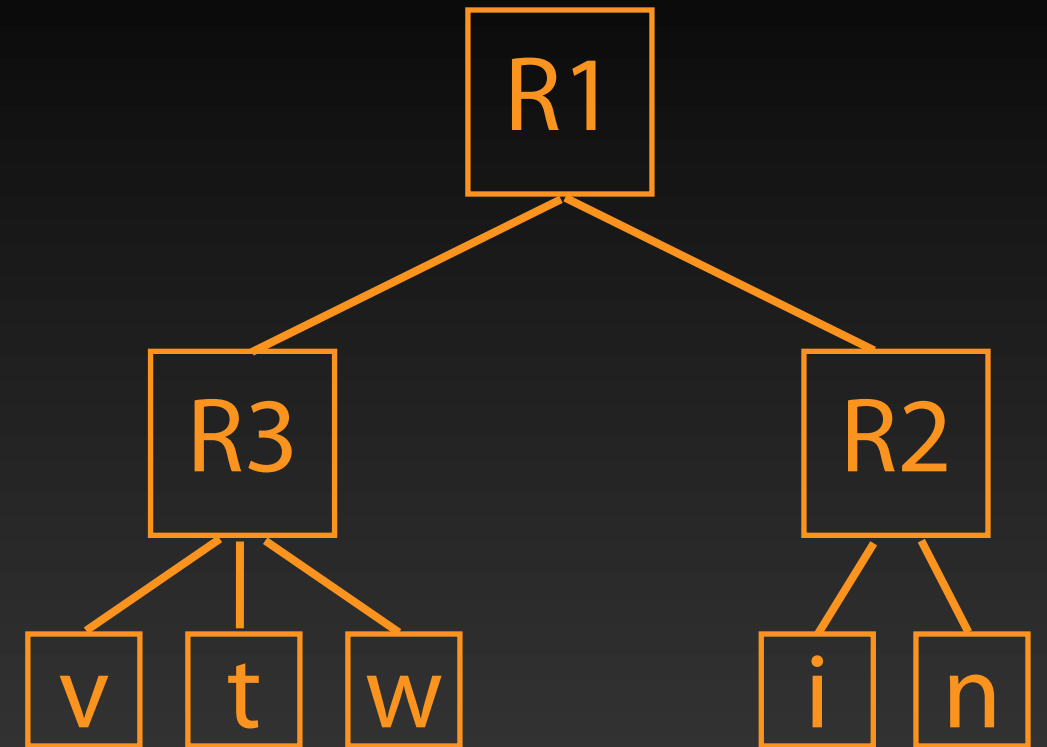
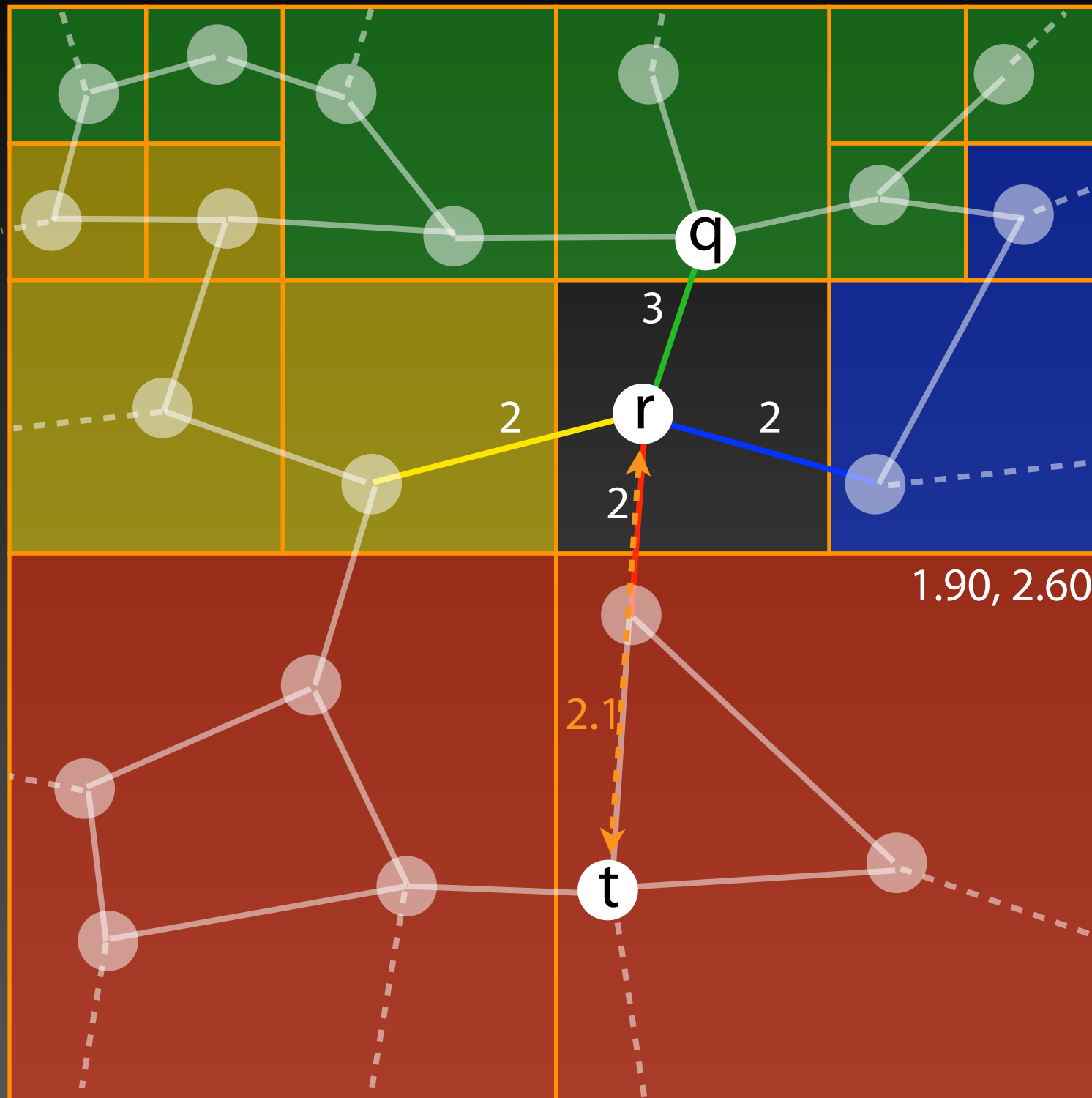


Queue: $t(7.00, 9.24)$, $w(7.37, 8.98, r)$
 $v(8.88, 14.23)$

kNN: $i(-, 1.68)$, $n(-, 4)$,
 $w(-, 8.98) \leq 8.98$

Current Working Set: **w**
 $3 + 2.3 \times (1.90, 2.60)$

Finding k-Nearest Neighbors

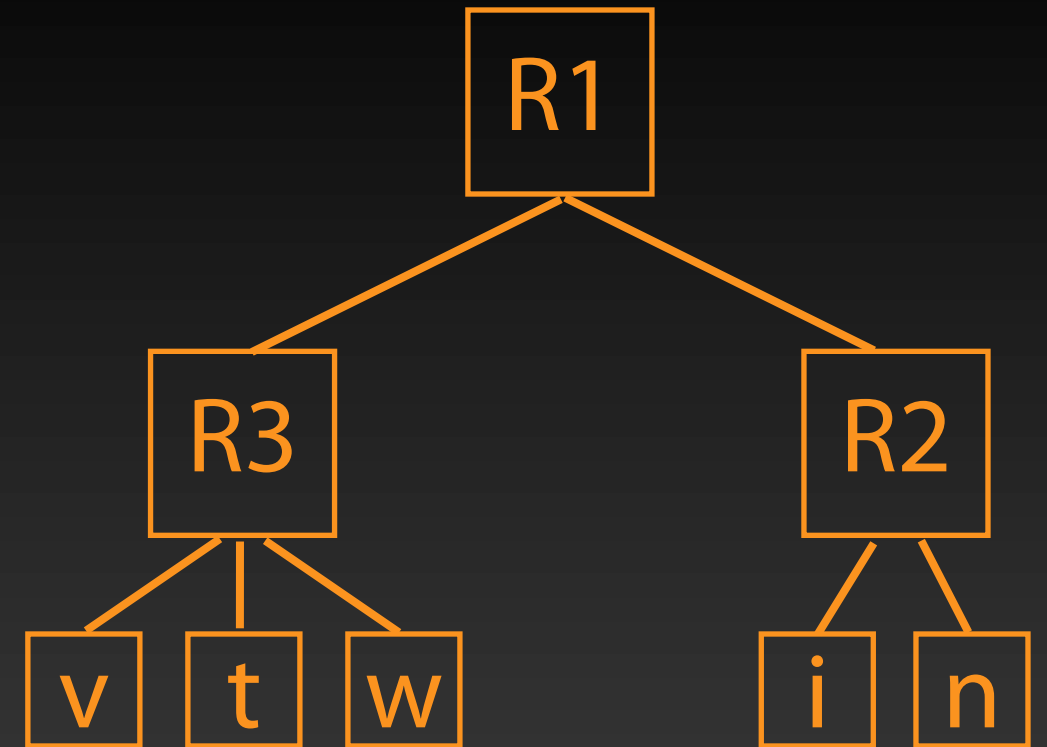
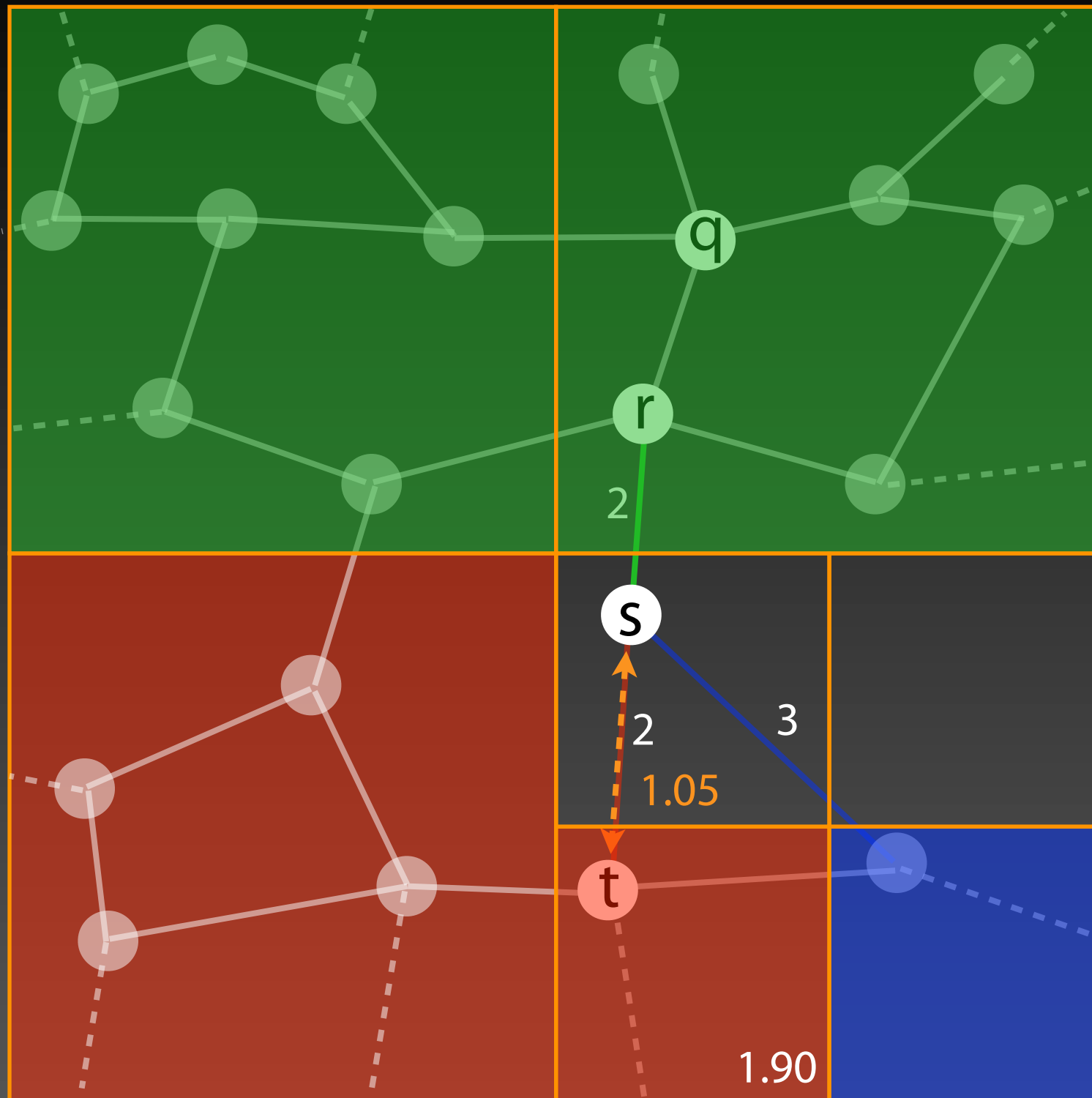


Queue: $t(7.00, 8.46, r)$, $w(7.37, 8.98, r)$
 $v(8.88, 14.23)$

kNN: $i(-, 1.68)$, $n(-, 4)$,
 $t(-, 8.46) \leq 8.46$

Current Working Set: t
 $3 + 2.1 \times (1.90, 2.60)$

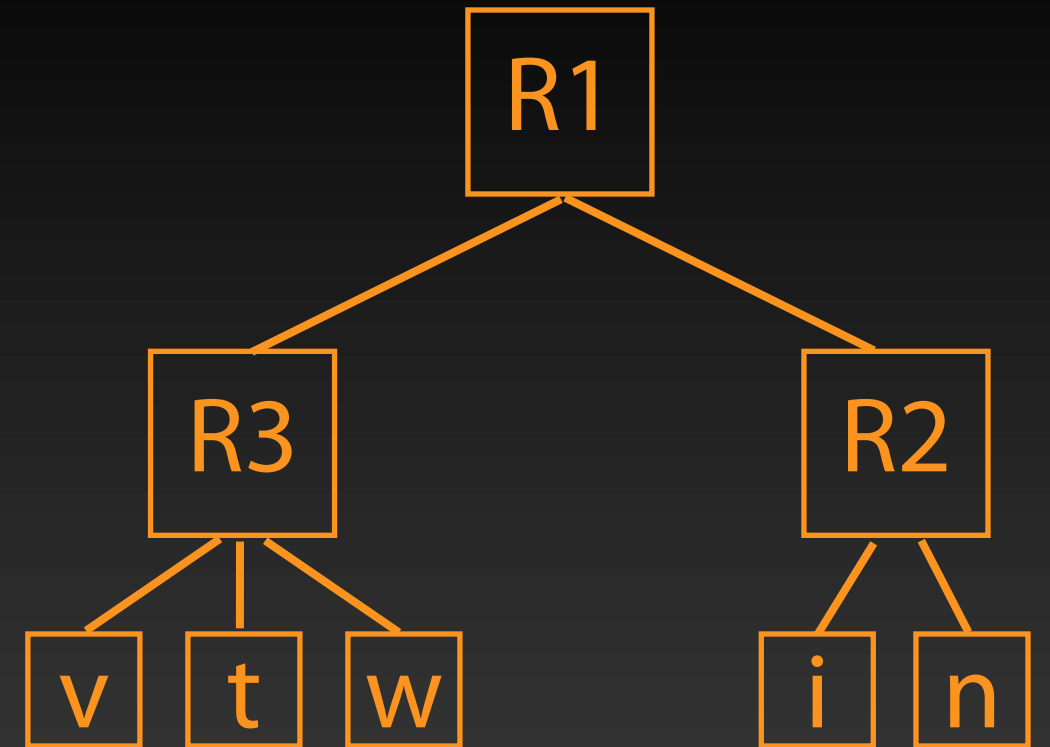
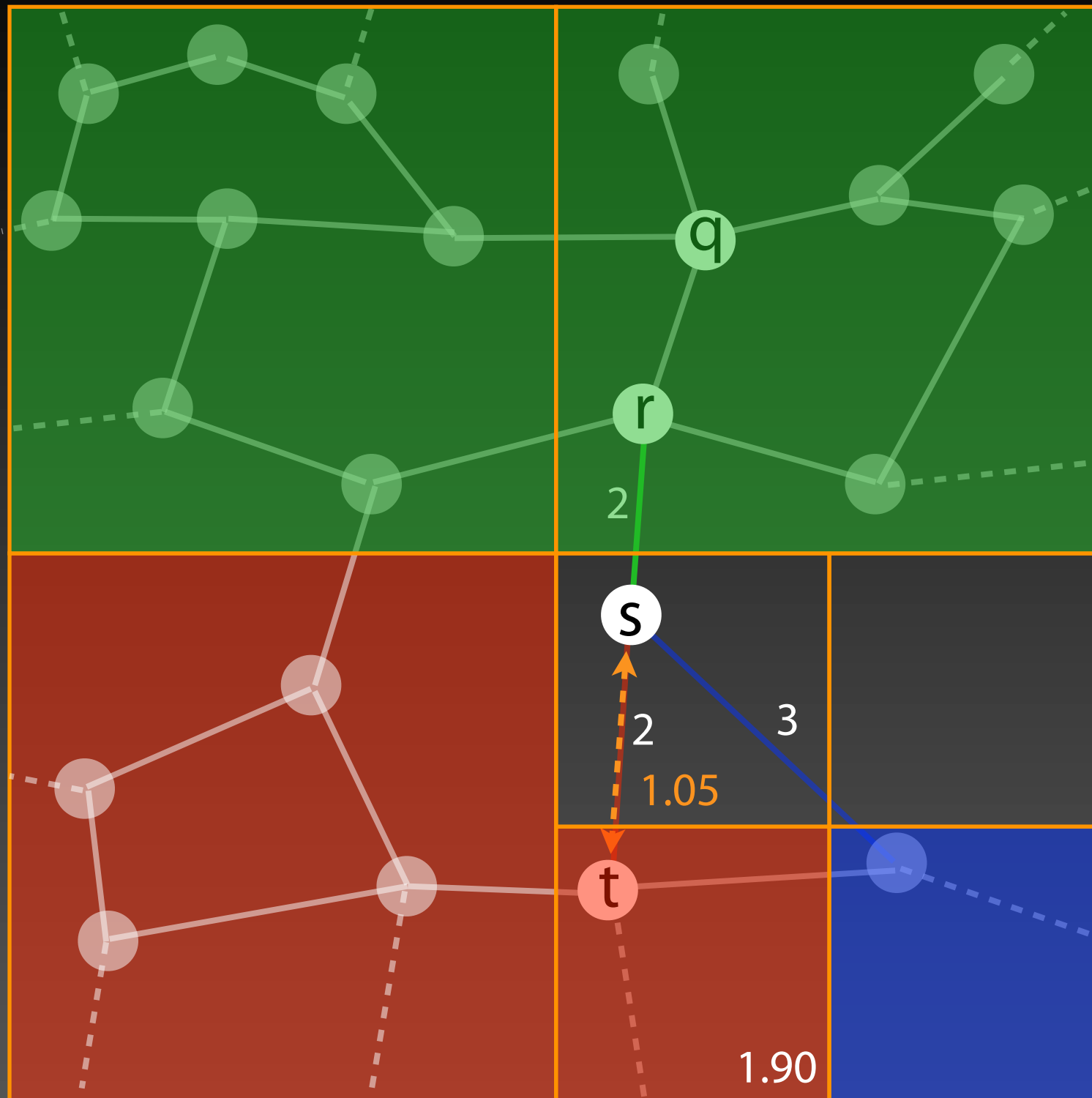
Finding k-Nearest Neighbors



Queue: $w(7.37, 8.98, r)$
 $v(8.88, 14.23)$
 kNN: $i(-, 1.68), n(-, 4),$
 $t(-, 7.00) \leq 7.00$

Current Working Set: t
 $5 + 1.05 \times 1.90$

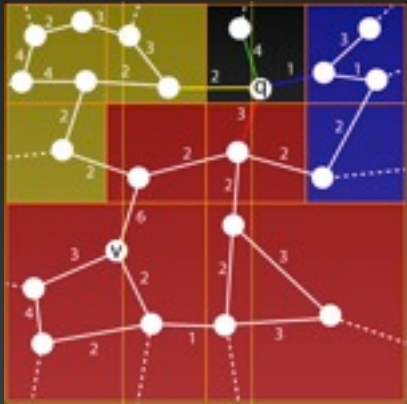
Finding k-Nearest Neighbors



Queue: $w(7.37, 8.98, r)$
 $v(8.88, 14.23)$
 kNN: $i(-, 1.68), n(-, 4),$
 $t(-, 7.00) \leq 7.00$

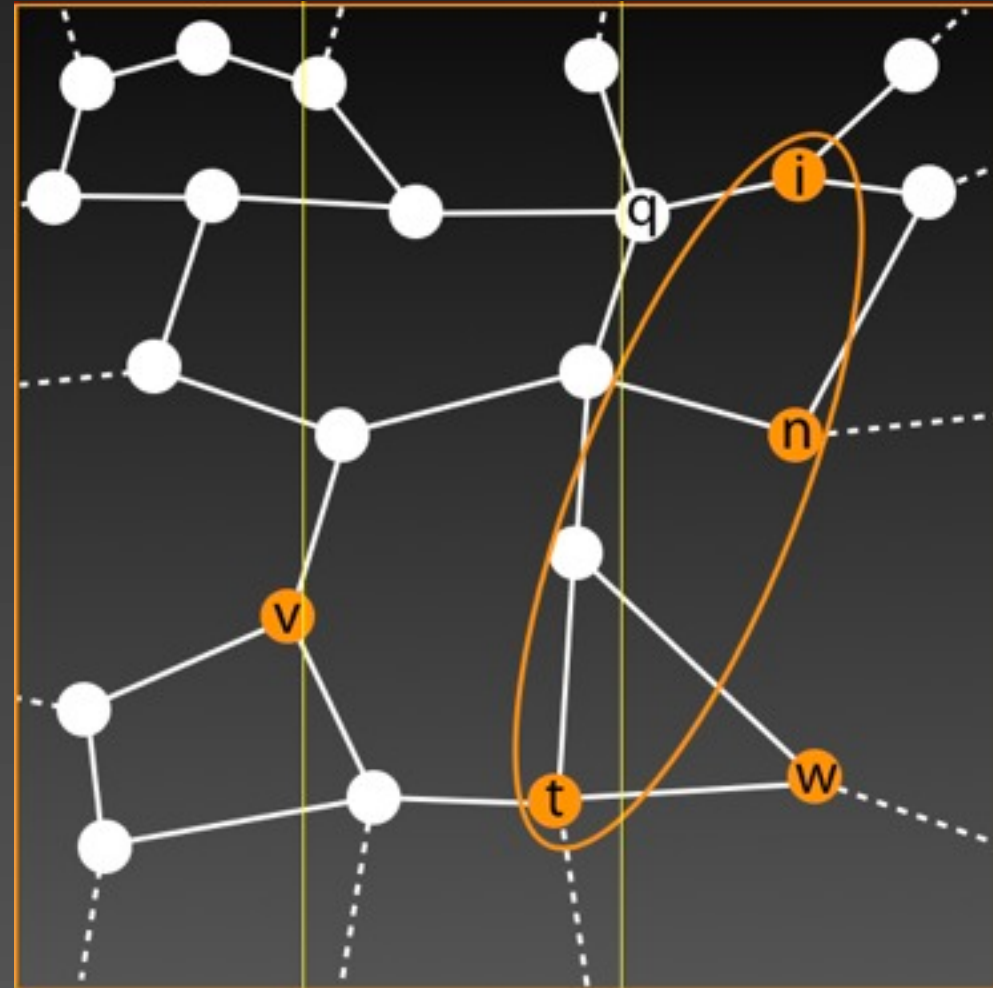
Current Working Set:
 Disregard rest of Queue

Shortest Path + Best-First Quadtrees k-Nearest Neighbor



Issues:

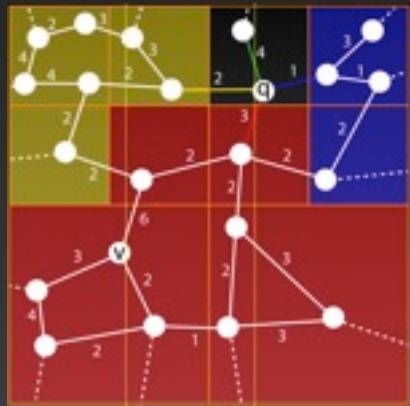
- Non-Planar Networks
- Multiple Shortest Paths



Shortest Path Quadrees

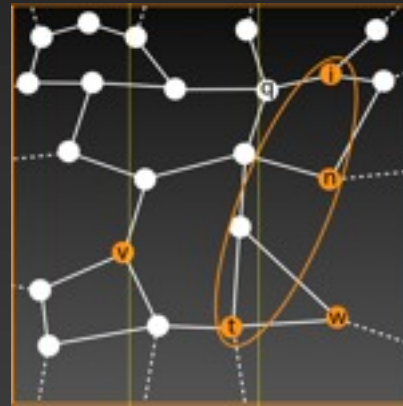
+

Best-First k-Nearest Neighbor



Issues:

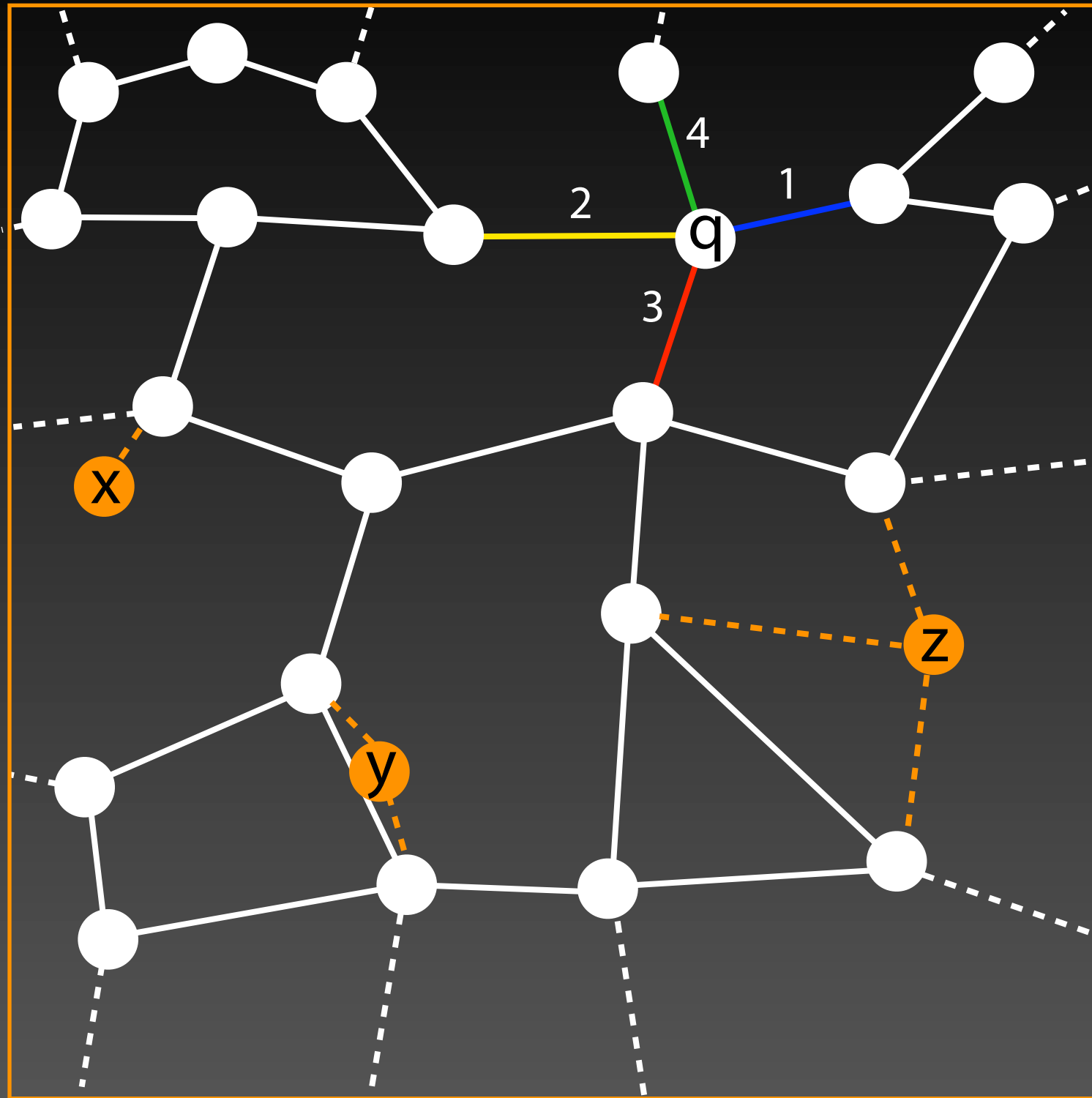
- Non-Planar Networks
- Multiple Shortest Paths



Issues:

- Objects Not on Vertices
- Similar Distances of Objects
- Big k

Objects Not on Vertices



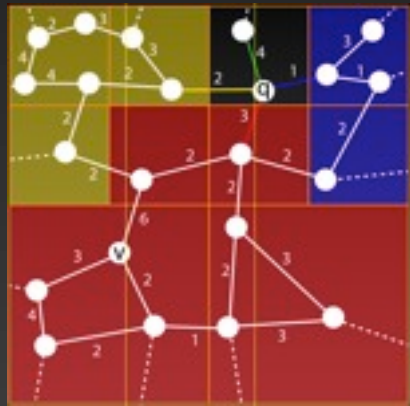
Effects:

- Associate objects with node(s)
- Adjust kNN algorithm
- Computational effort bigger

Shortest Path Quadrees

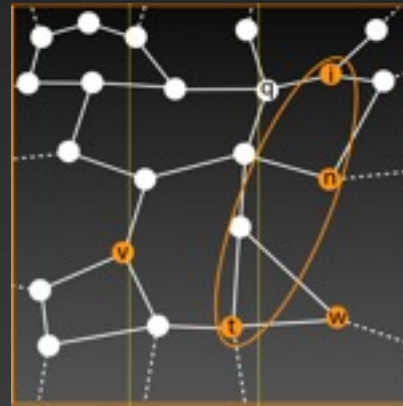
+

Best-First k-Nearest Neighbor



Issues:

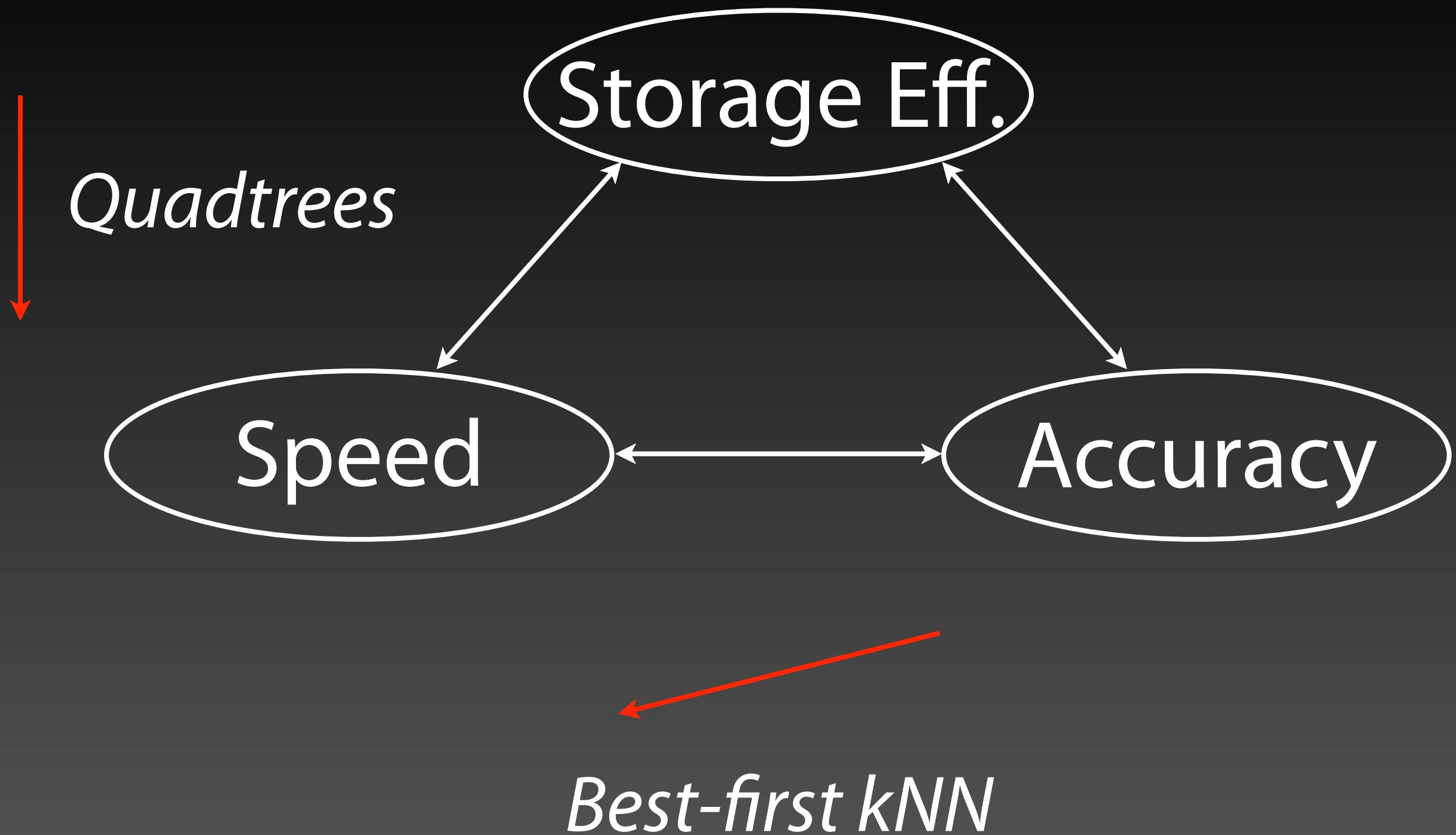
- Non-Planar Networks
- Multiple Shortest Paths



Issues:

- Objects Not on Vertices
- Similar Distances of Objects
- Big k

Trade-Offs



Trade-Offs

Quadrees

- *Store Trees*
- *Retrieve Shortest Paths Based on Intermediate Edges*

Best-first kNN

- *Store Distance Ratios*
- *K Nearest Neighbors Ordered by Estimates*
- *Typically Less Blocks and Nodes Visited*

