

# Fighting Spam with the NeighborhoodWatch DHT

---

**Adam Bender**

Derek Monner

Nate Goergen

Neil Spring

Bobby Bhattacharjee

**University of Maryland**

Rob Sherwood

**Deutsche Telekom**

# Spam is bad

---

- Methods of fighting email spam
  - ✦ Blacklisting known senders
  - ✦ Content-based filtering
  - ✦ Greylisting
  - ✦ Proof of work
  - ✦ DomainKeys and Sender Policy Framework

# Spam is bad

---

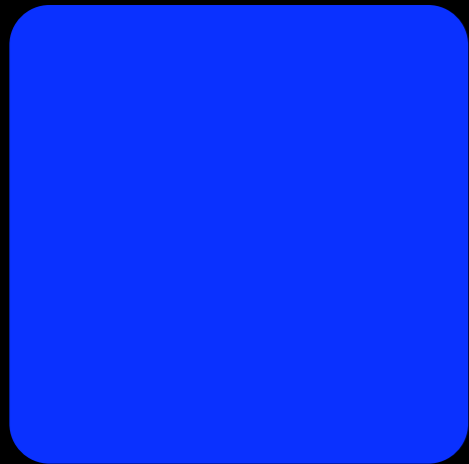
- Methods of fighting email spam
  - ✦ **Blacklisting known senders**
  - ✦ Content-based filtering
  - ✦ Greylisting
  - ✦ Proof of work
  - ✦ DomainKeys and Sender Policy Framework

# DNS-based blacklist

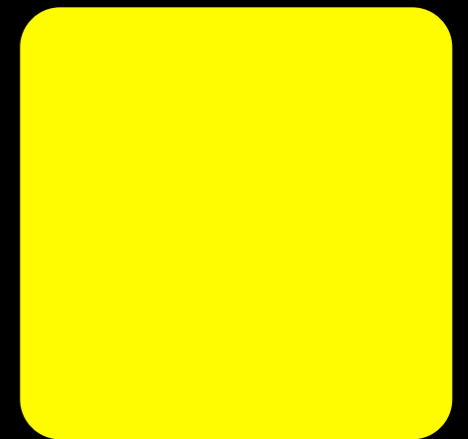
---

1.2.3.4

Mail source



Mail  
destination



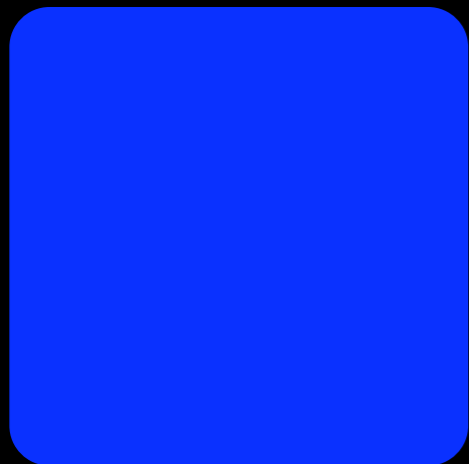
DNSBL

# DNS-based blacklist

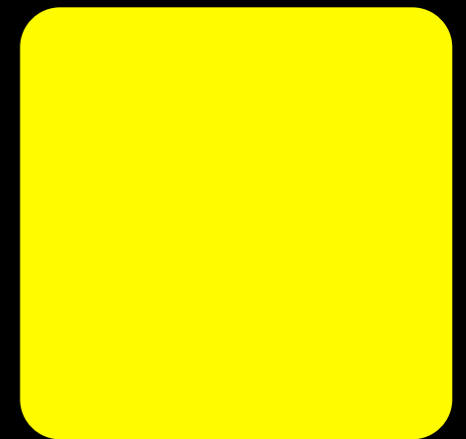
---

1.2.3.4

Mail source



Mail  
destination



DNSBL

# DNS-based blacklist

---

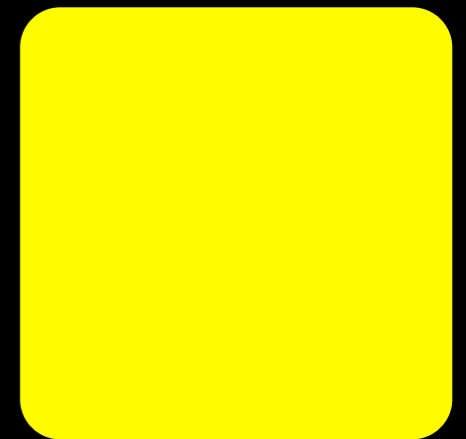
1.2.3.4

Mail source



Mail  
destination

DNS query: 1.2.3.4 blocked?



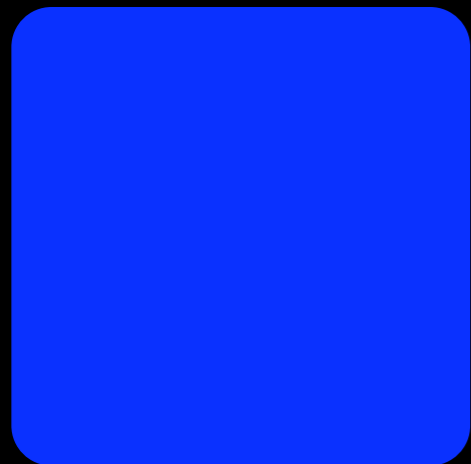
DNSBL

# DNS-based blacklist

---

1.2.3.4

Mail source

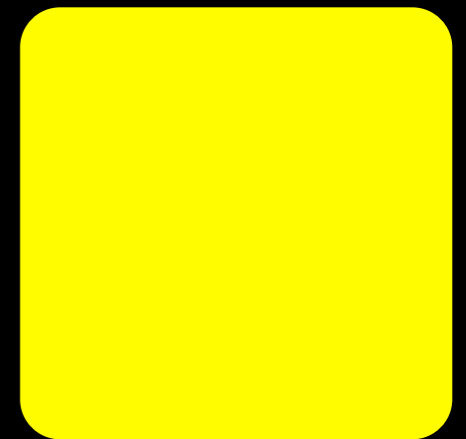


Mail  
destination

DNS query: 1.2.3.4 blocked?



Response: **Yes** or **No**



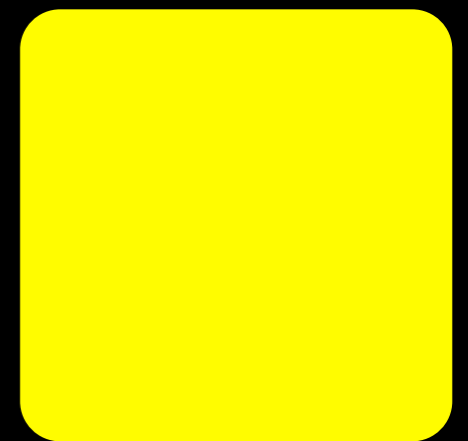
DNSBL

# Under attack

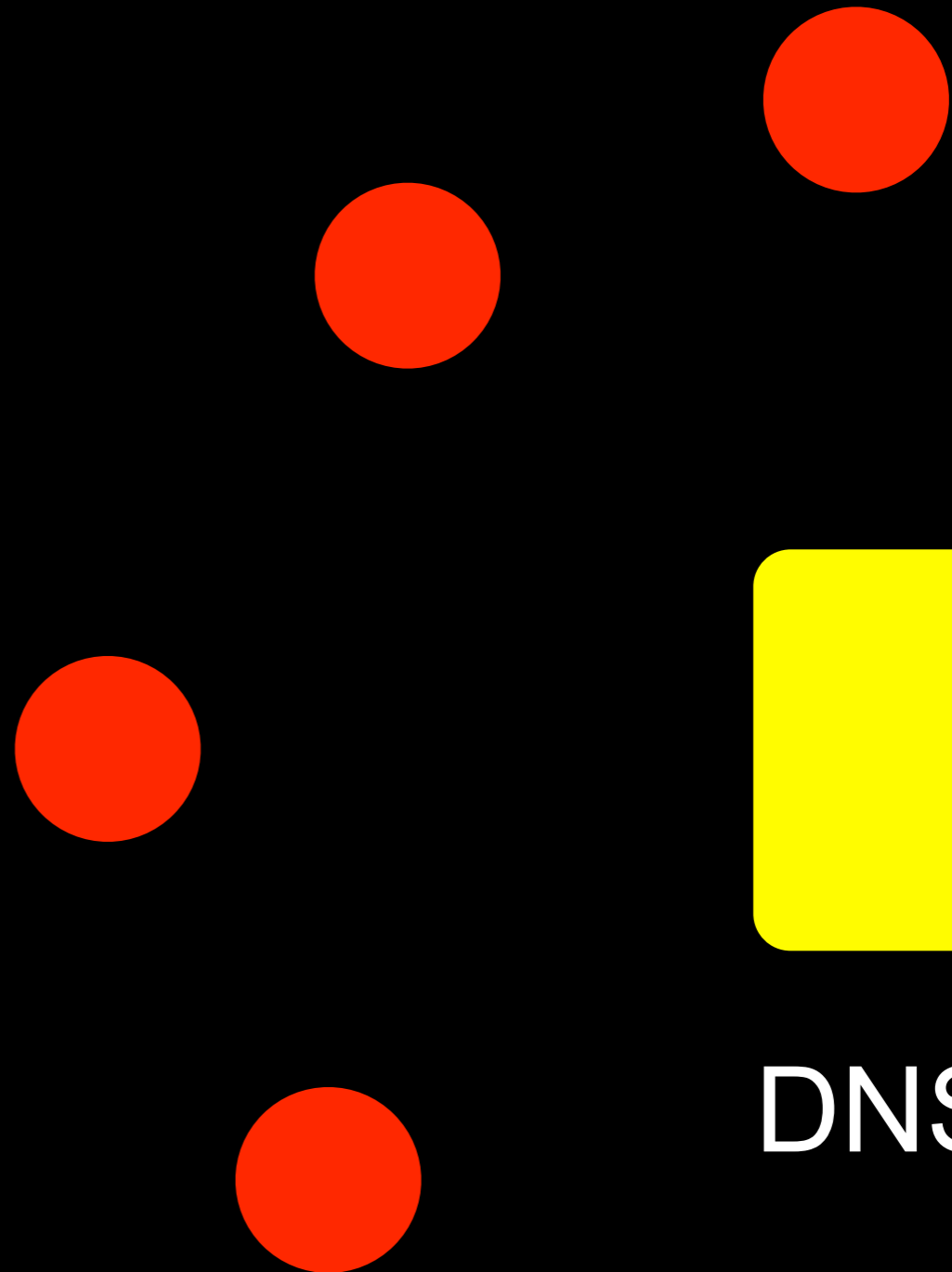
---



Mail  
destination

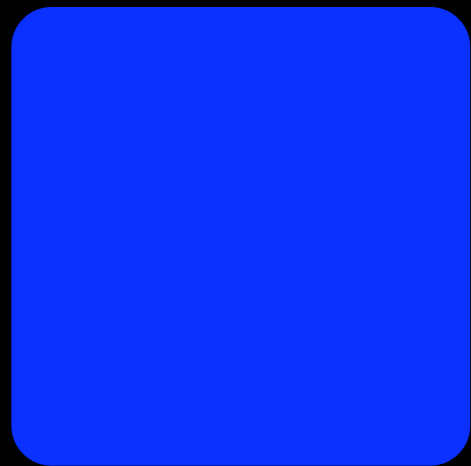


DNSBL

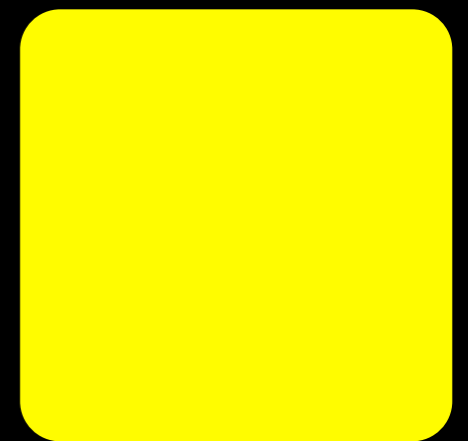


# Under attack

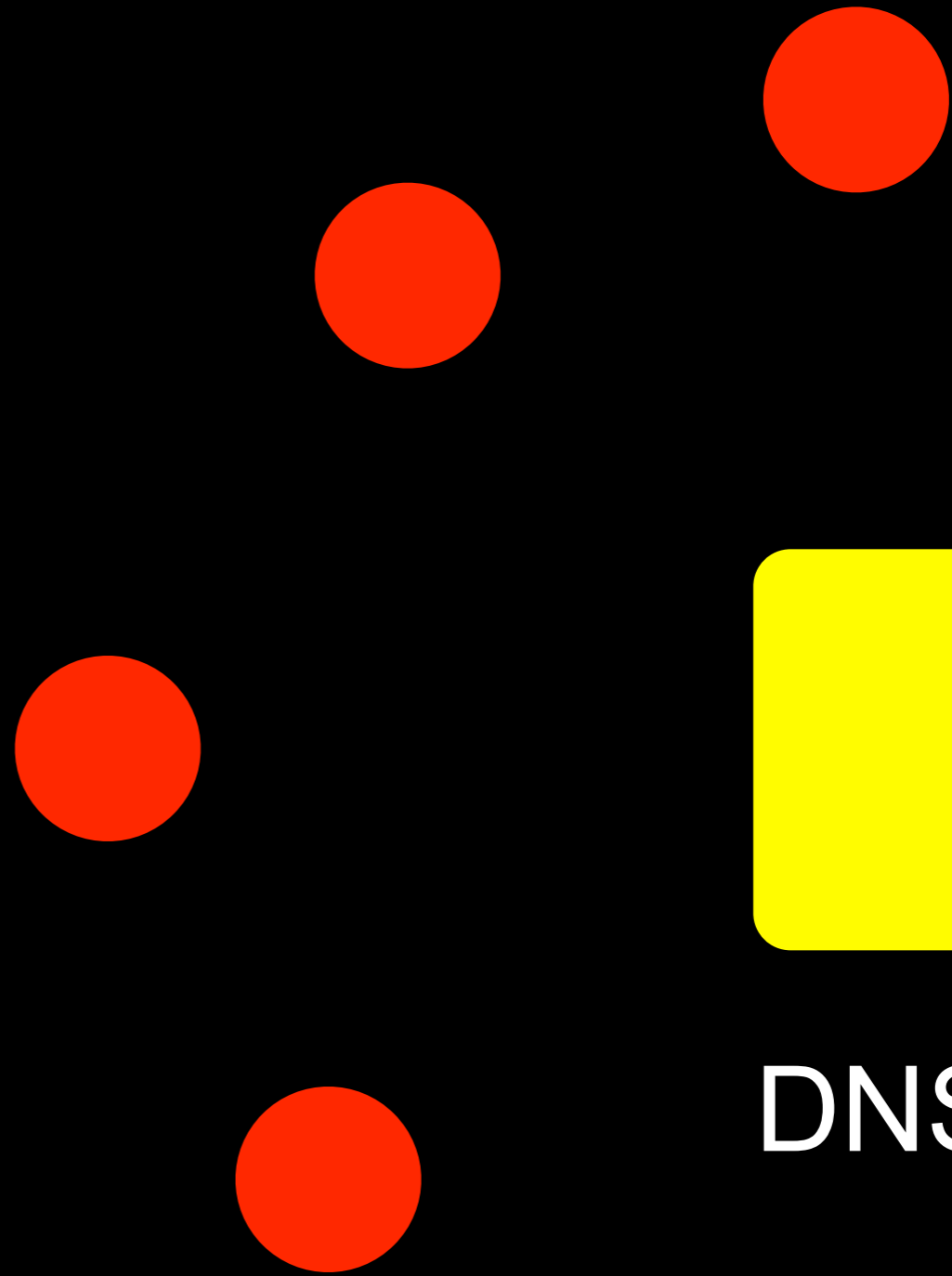
---



Mail  
destination



DNSBL

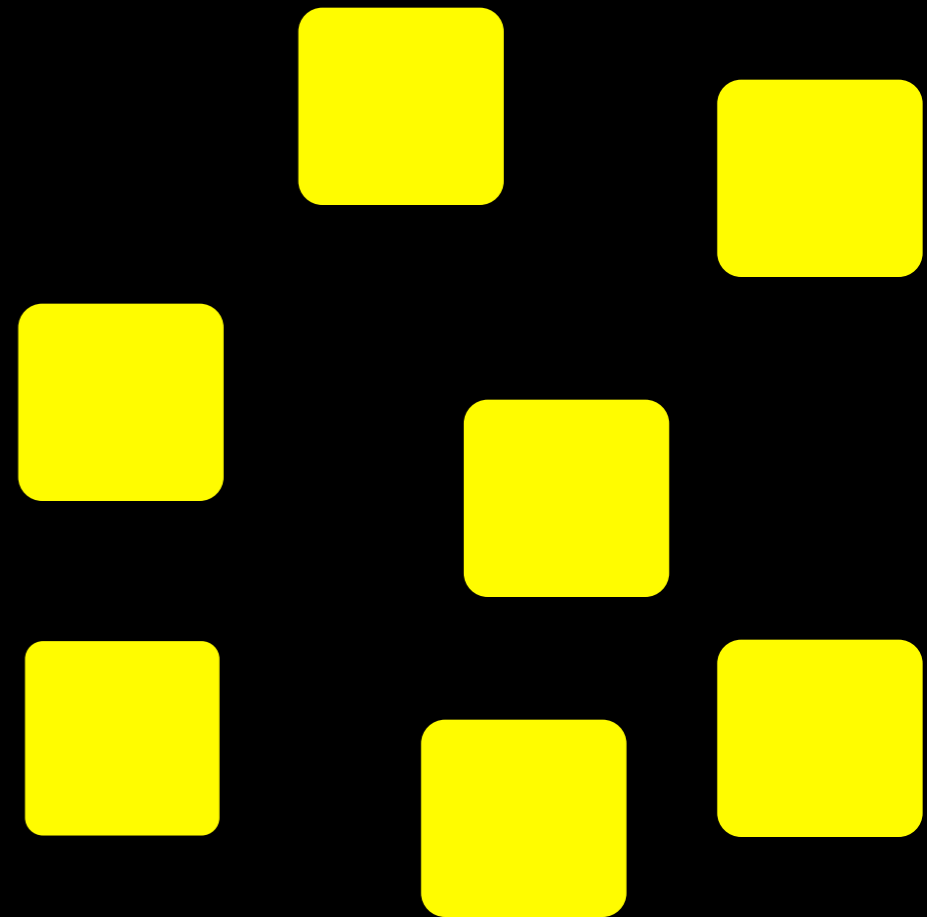


# Replication

---



Mail  
destination



DNSBL

Distributed hash tables  
provide efficiency, scalability,  
and fault-tolerance

# Who should join?

---

## Single administration

- Spammers cannot join

## Multiple administrations

- Resources from many organizations
- Multiple blacklists in single repository

# Who should join?

---

## Single administration

- Spammers cannot join

## Multiple administrations

- Resources from many organizations
- Multiple blacklists in single repository

# Who should join?

---

Spammers may infiltrate the DHT

## Single administration

- Spammers cannot join

## Multiple administrations

- Resources from many organizations
- Multiple blacklists in single repository

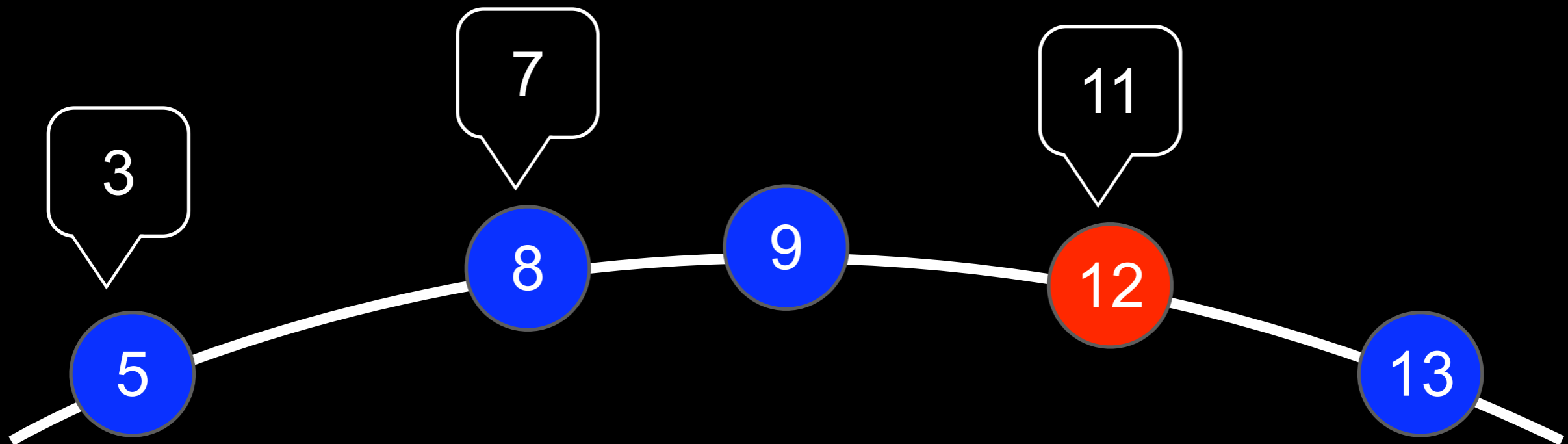
# What can bad nodes do?

---

- Ignore queries
- Misroute queries
- Falsify routing updates
- Refuse to store or return items
- Return incorrect items
- Prevent new nodes from joining

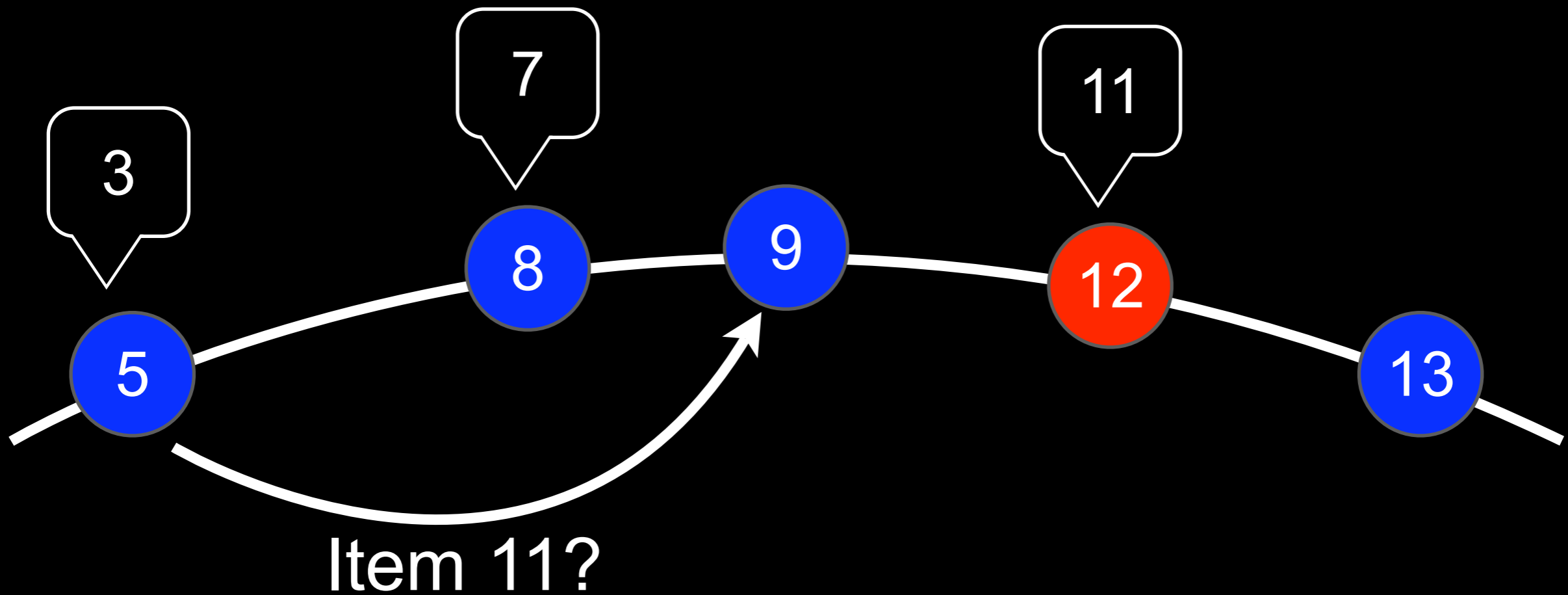
# Returning false responses

---



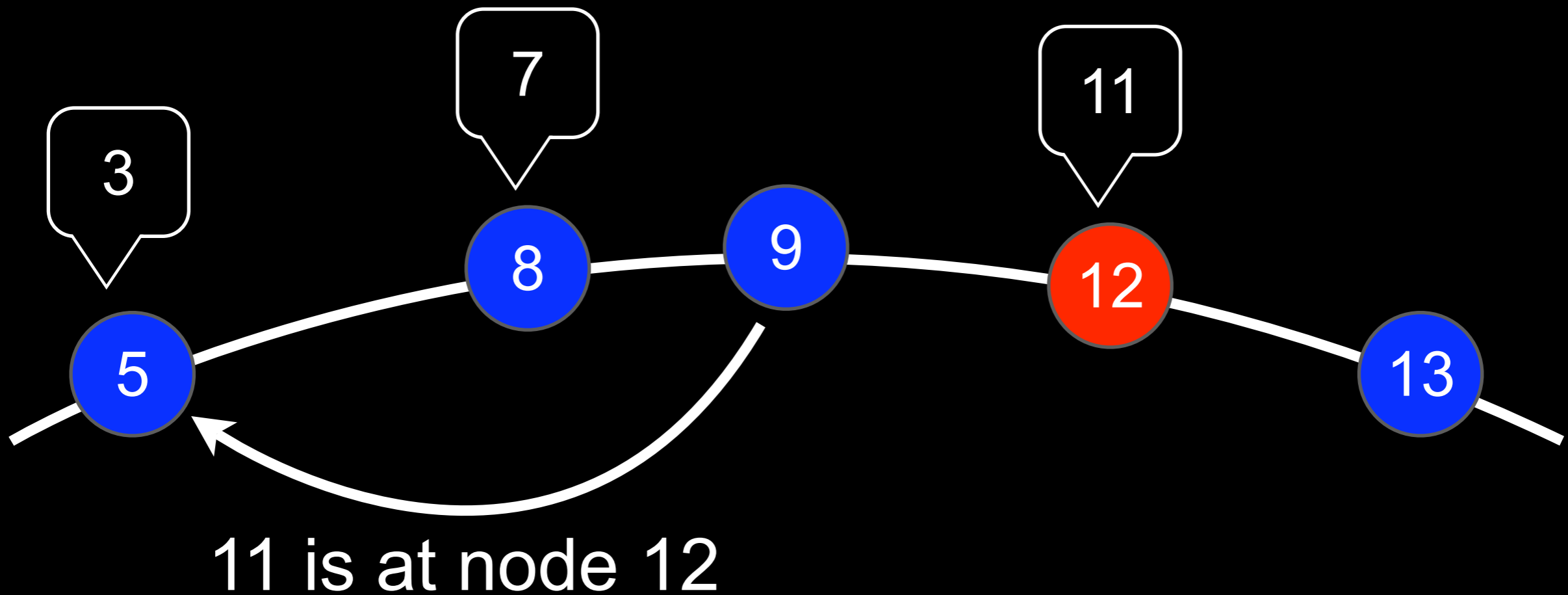
# Returning false responses

---



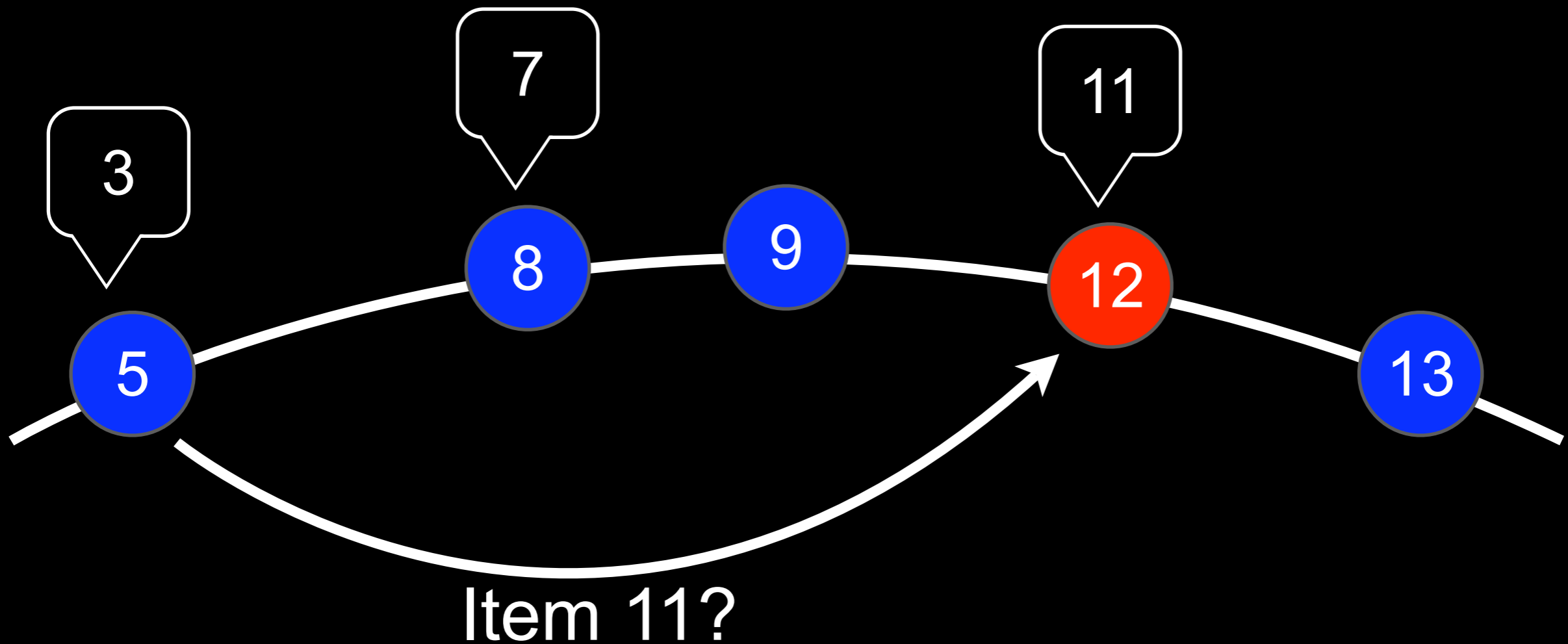
# Returning false responses

---



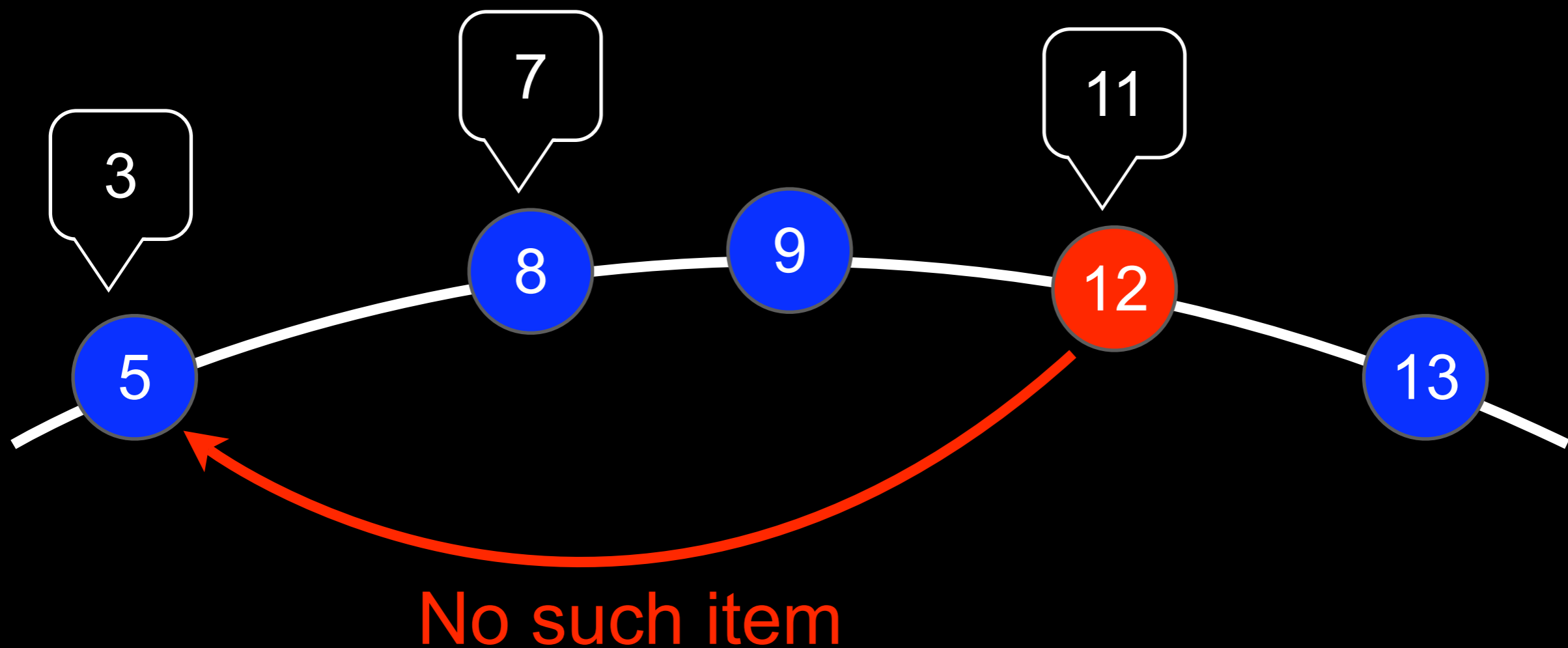
# Returning false responses

---



# Returning false responses

---



# NeighborhoodWatch

---

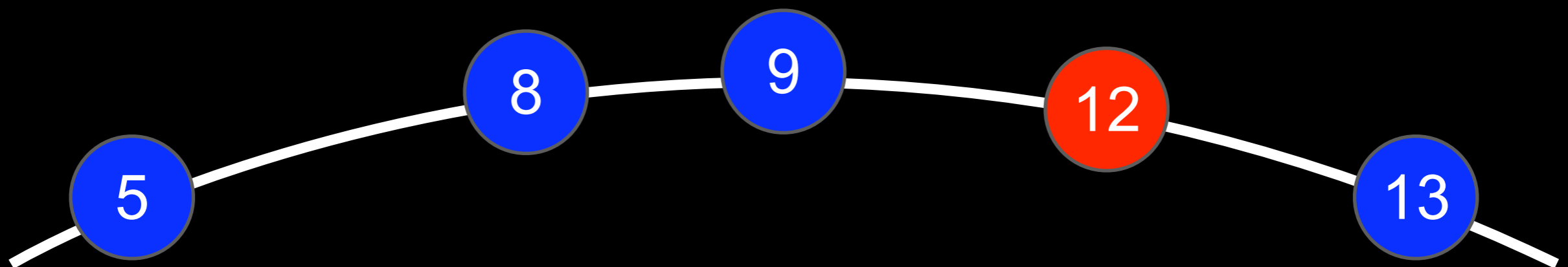
Invariant: bad nodes cannot corrupt **long sequences** of nodes in the ID-space

Misbehavior can be **proven**  $\mapsto$  bad nodes are evicted

# Neighborhoods

---

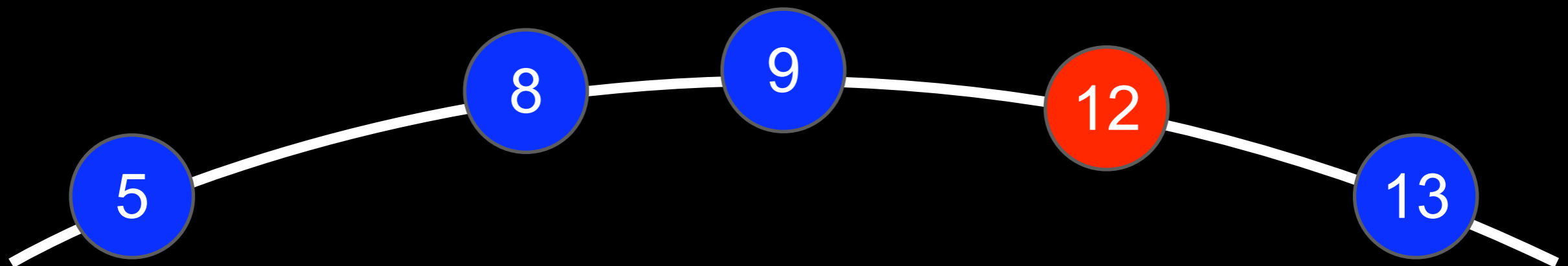
Bad nodes avoided by contacting neighbors



# Neighborhoods

---

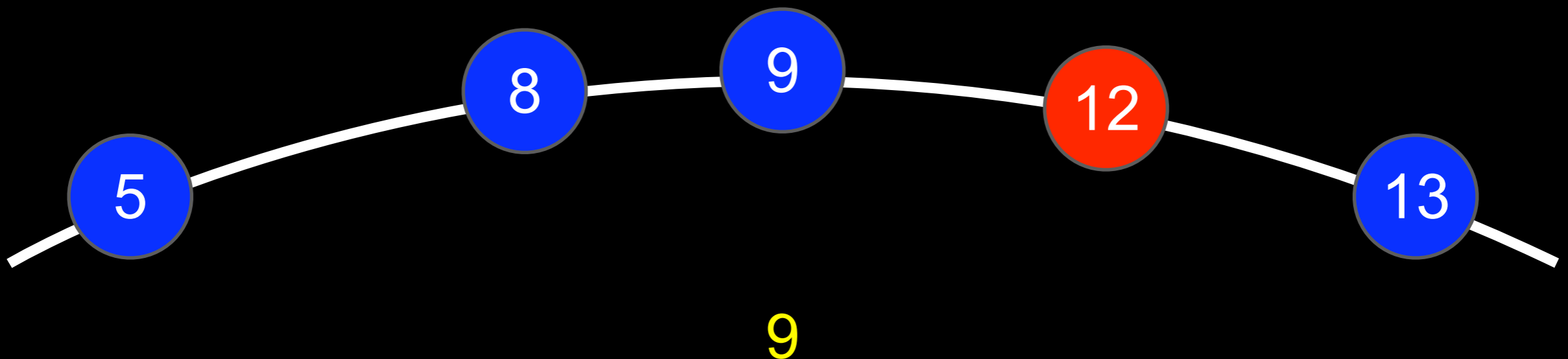
**Neighborhood certificates** (nCerts)  
identify neighborhoods



# Neighborhoods

---

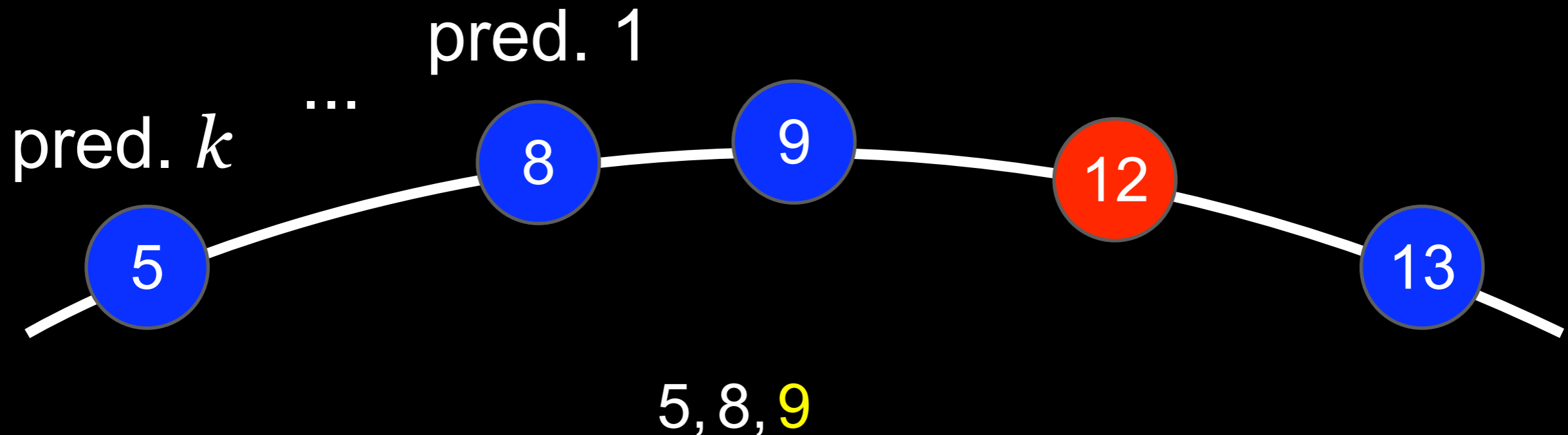
**Neighborhood certificates** (nCerts)  
identify neighborhoods



# Neighborhoods

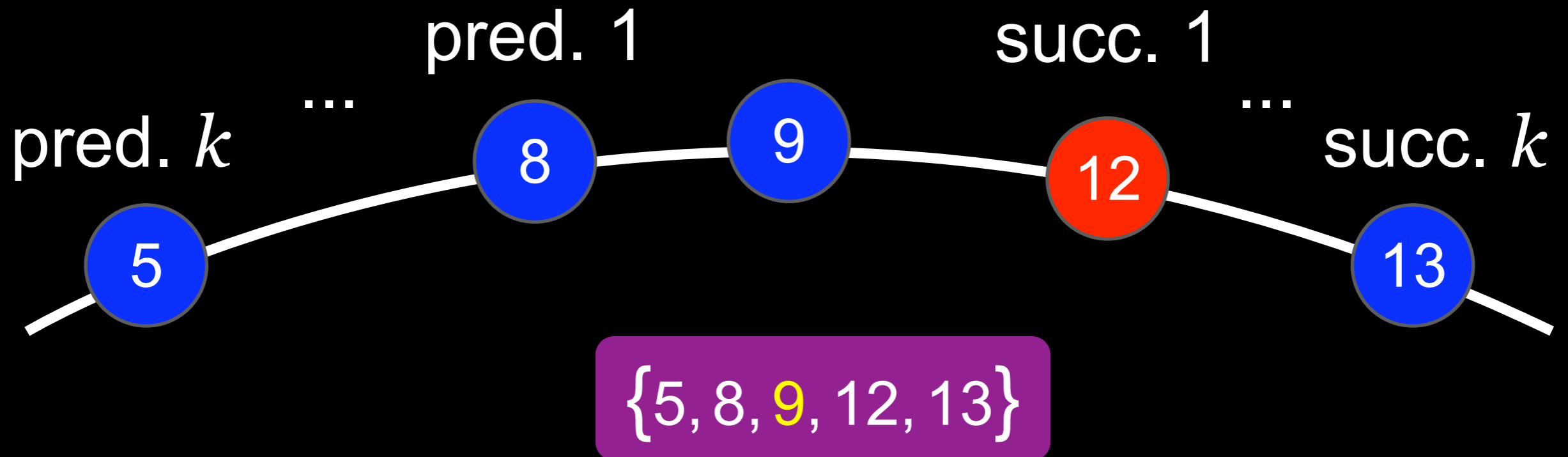
---

Neighborhood certificates (nCerts)  
identify neighborhoods



# Neighborhoods

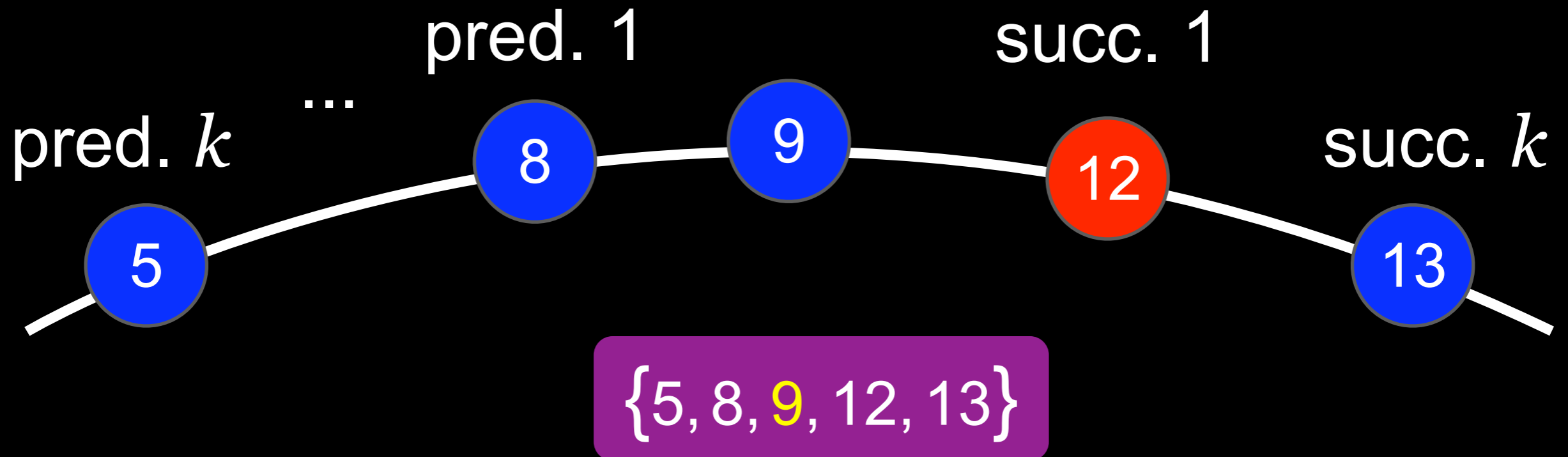
Neighborhood certificates (nCerts)  
identify neighborhoods



# Neighborhood Cert. Authority

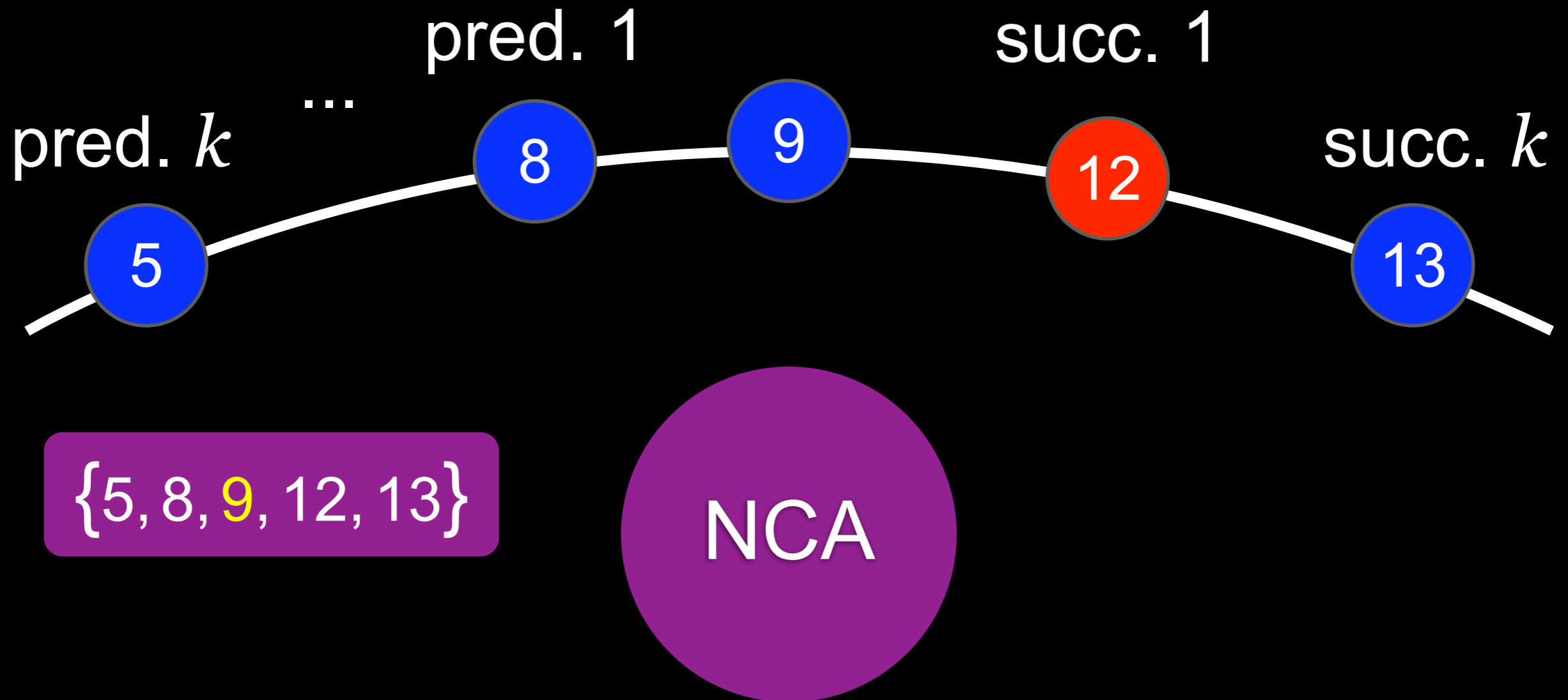
---

Neighborhood certificate authority (NCA) creates nCerts. It is the **only** trusted entity in the system.



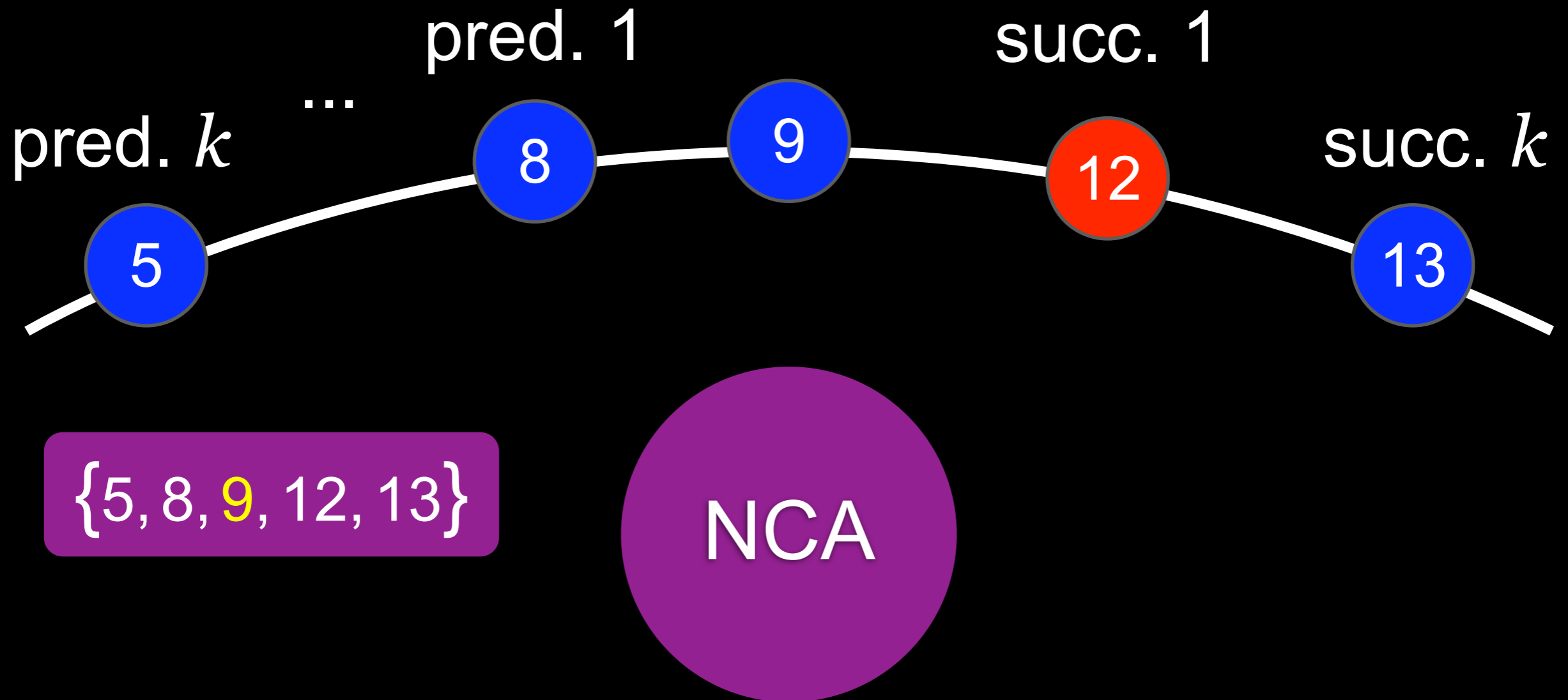
# Neighborhood Cert. Authority

Neighborhood certificate authority (NCA) creates nCerts. It is the **only** trusted entity in the system.



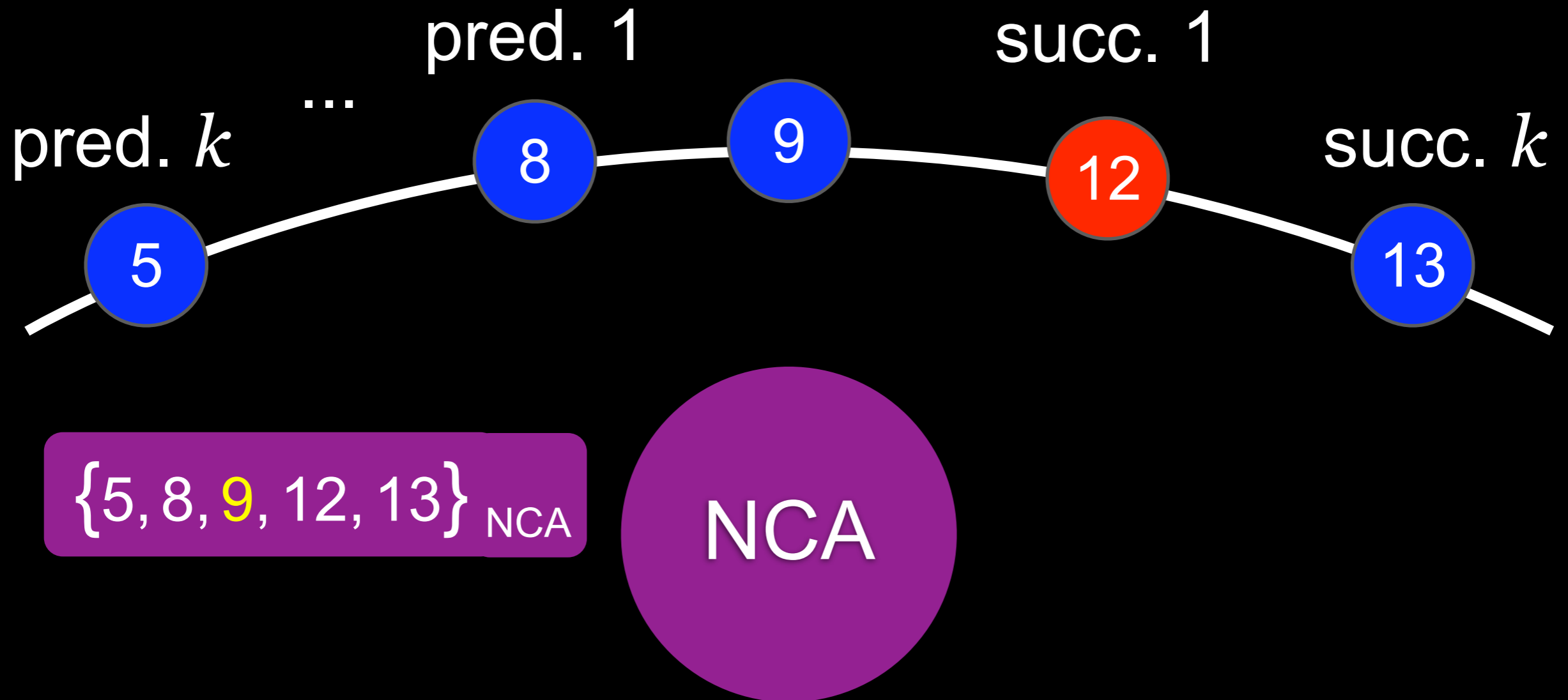
# Neighborhood Cert. Authority

NCA signs and distributes nCerts on demand when existing nCerts expire or nodes join.



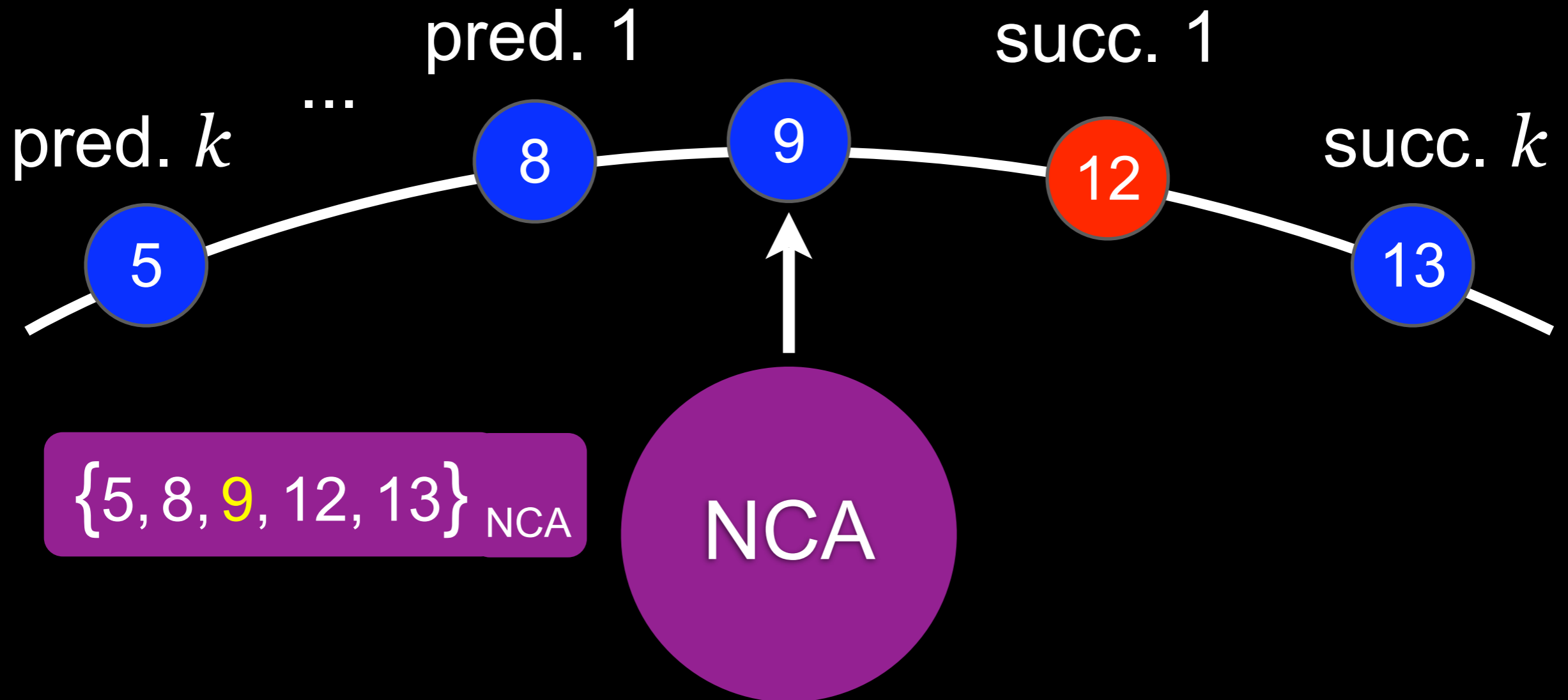
# Neighborhood Cert. Authority

NCA signs and distributes nCerts on demand when existing nCerts expire or nodes join.



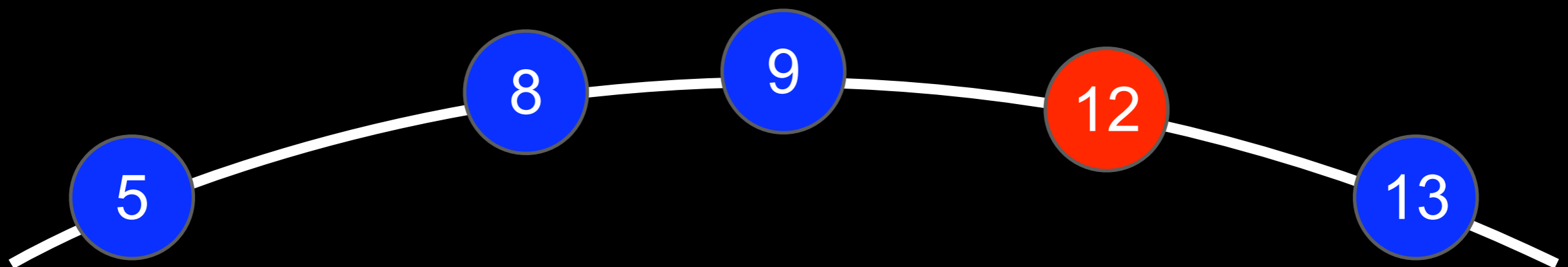
# Neighborhood Cert. Authority

NCA signs and distributes nCerts on demand when existing nCerts expire or nodes join.



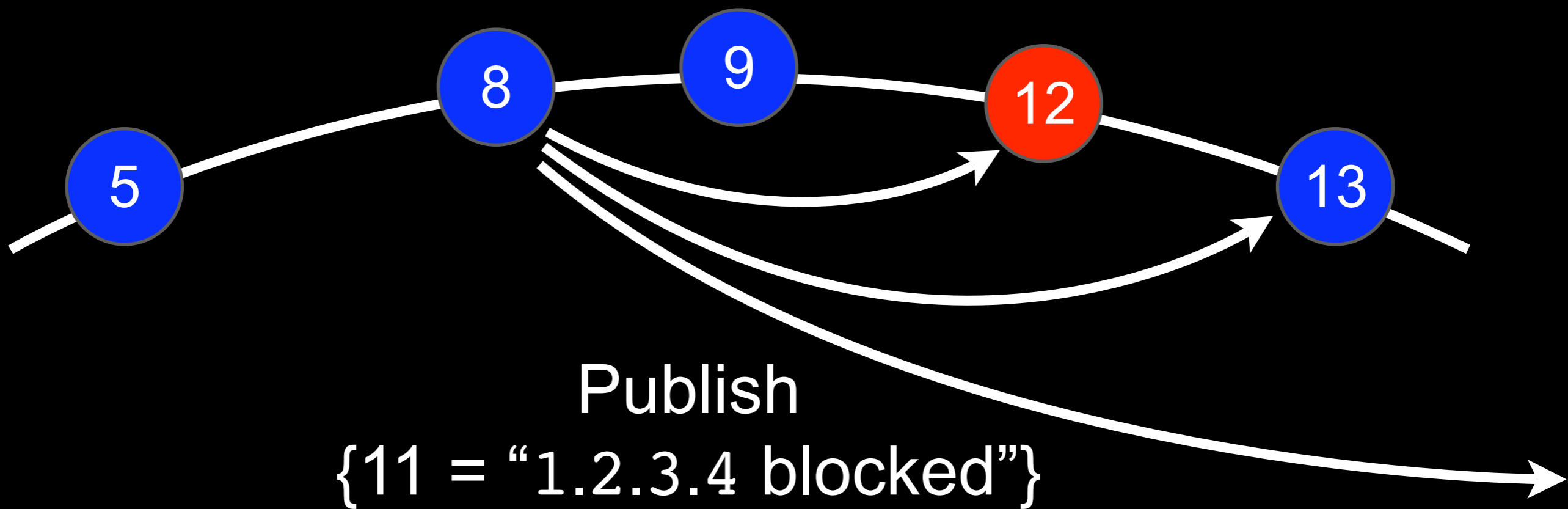
# Replicate to publish

---



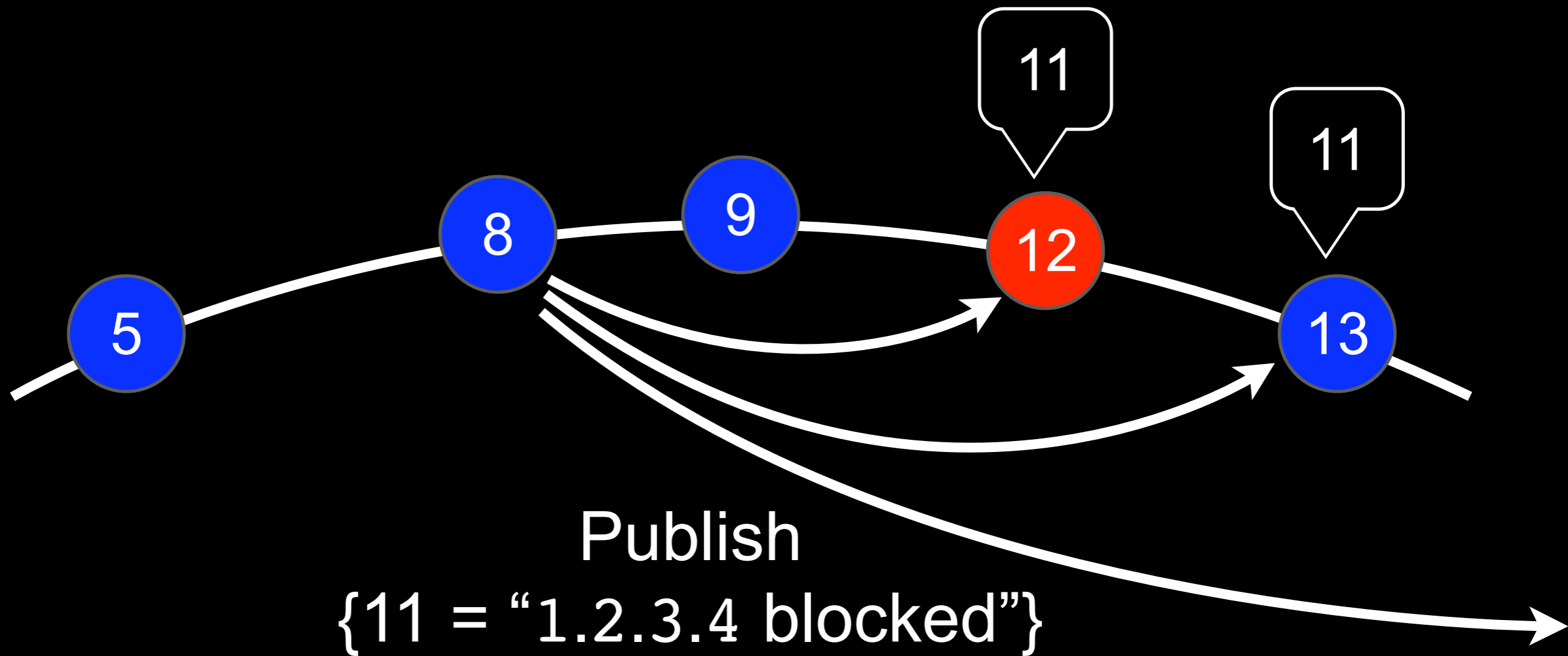
# Replicate to publish

---



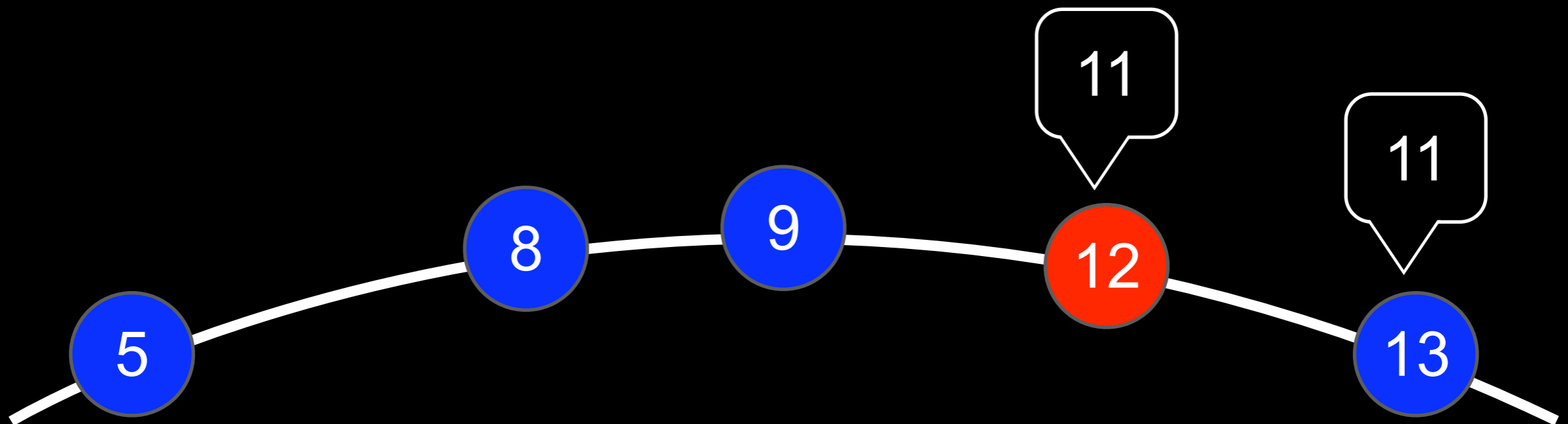
# Replicate to publish

---



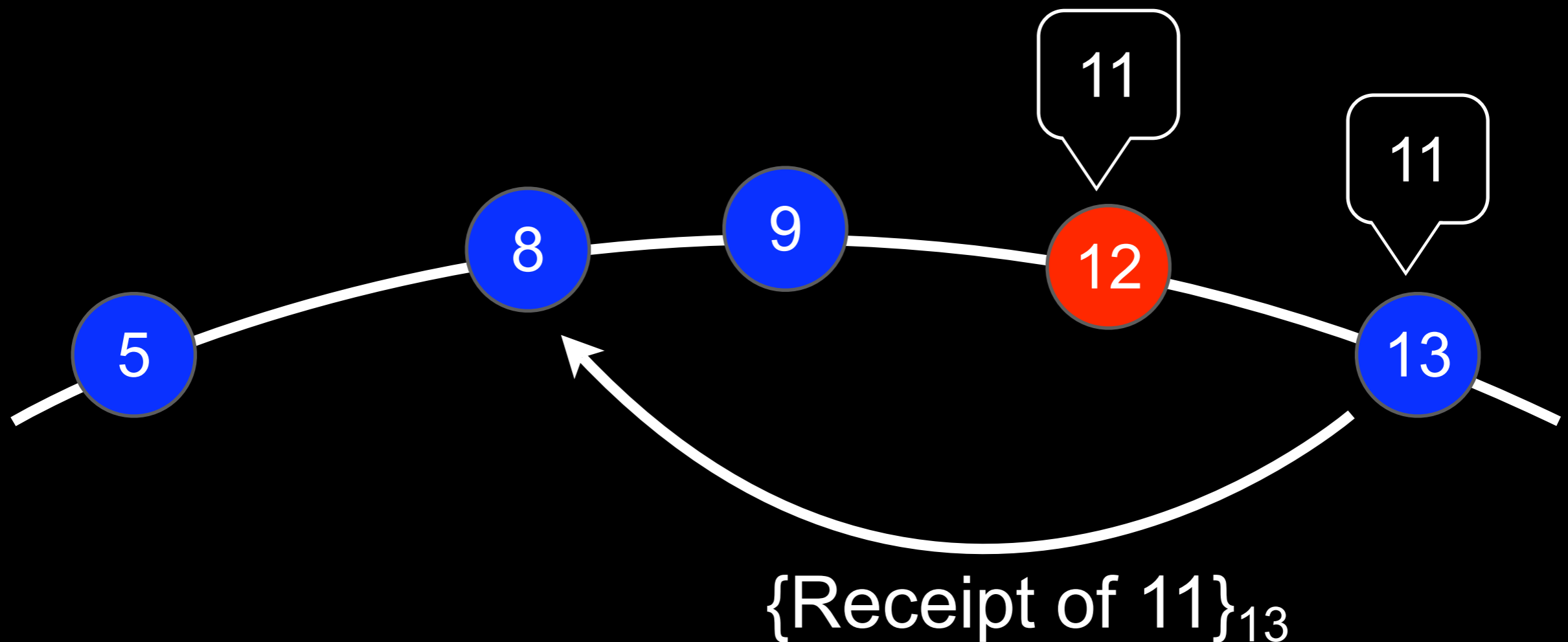
# Return receipts

---



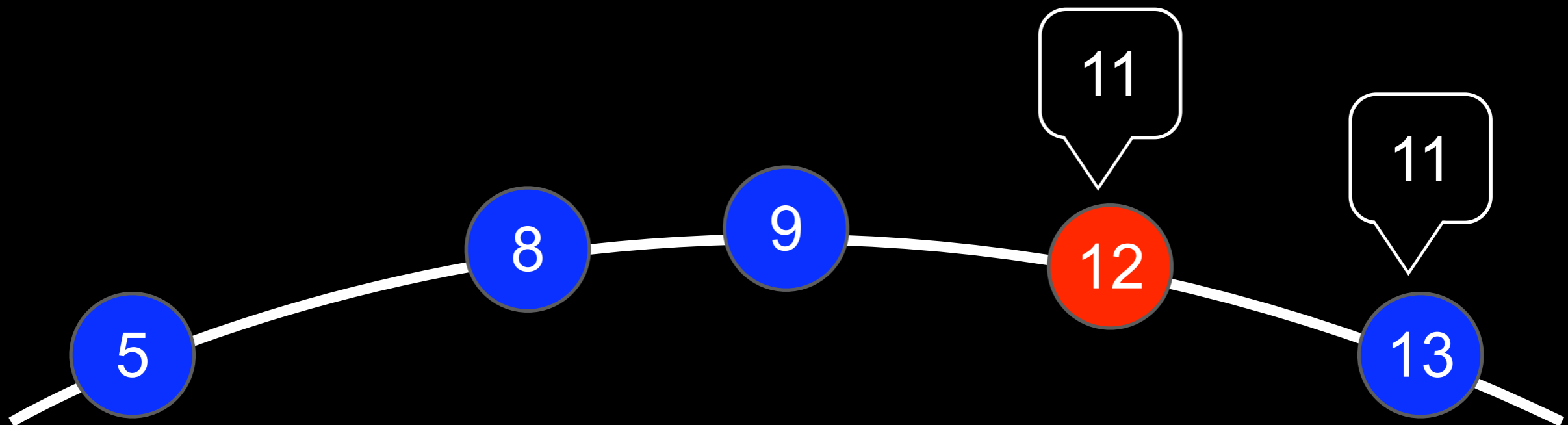
# Return receipts

---



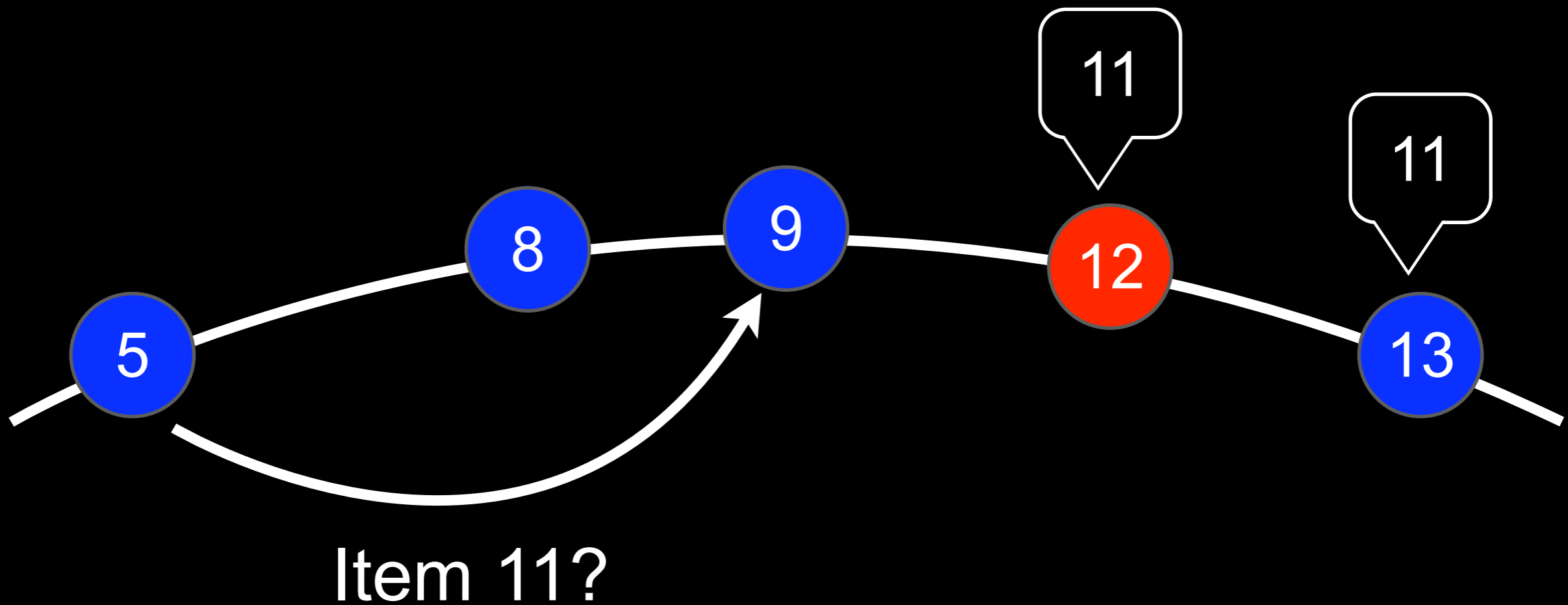
# Retrieve using nCerts

---

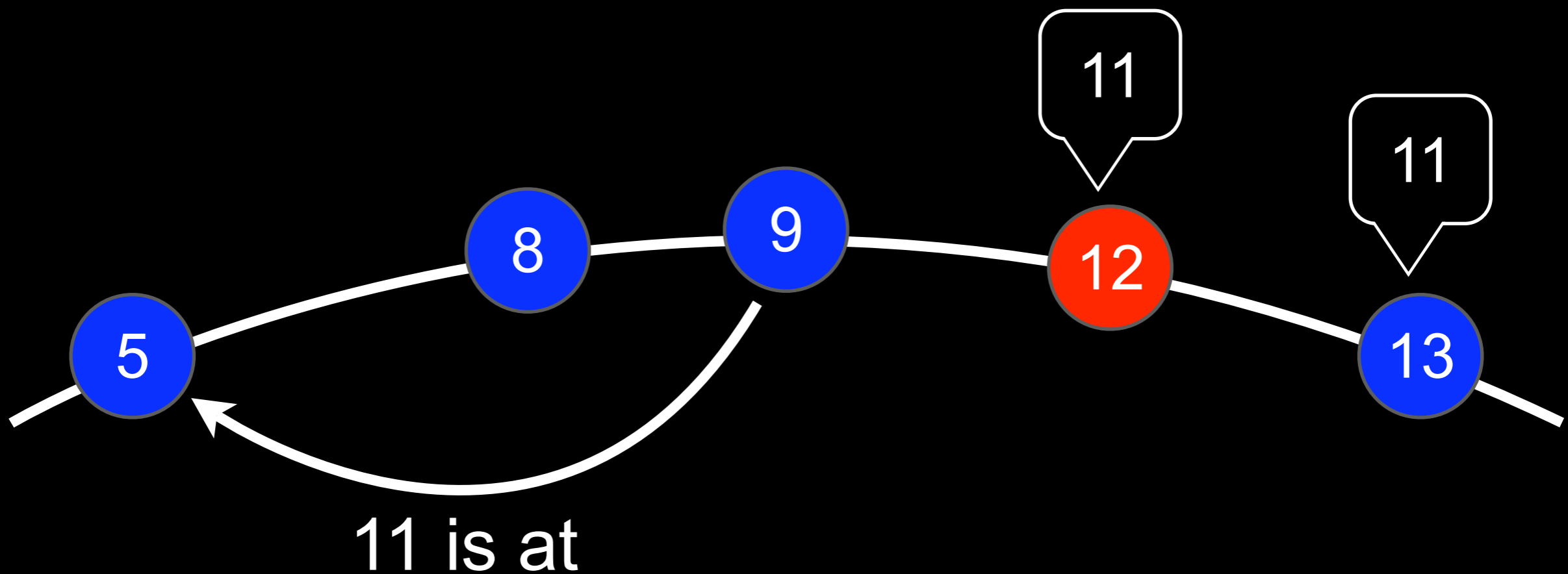


# Retrieve using nCerts

---



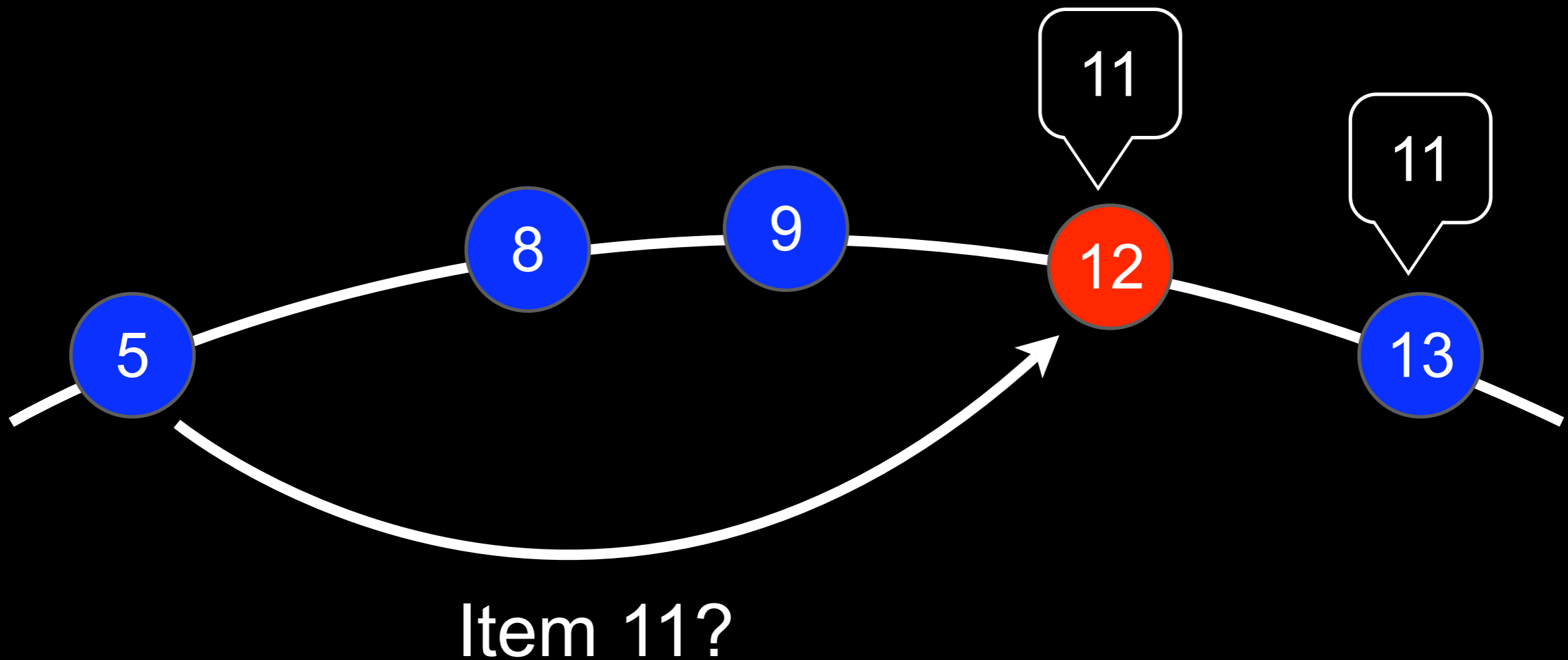
# Retrieve using nCerts



$\{8, 9, 12, 13, 17\}_{NCA}$

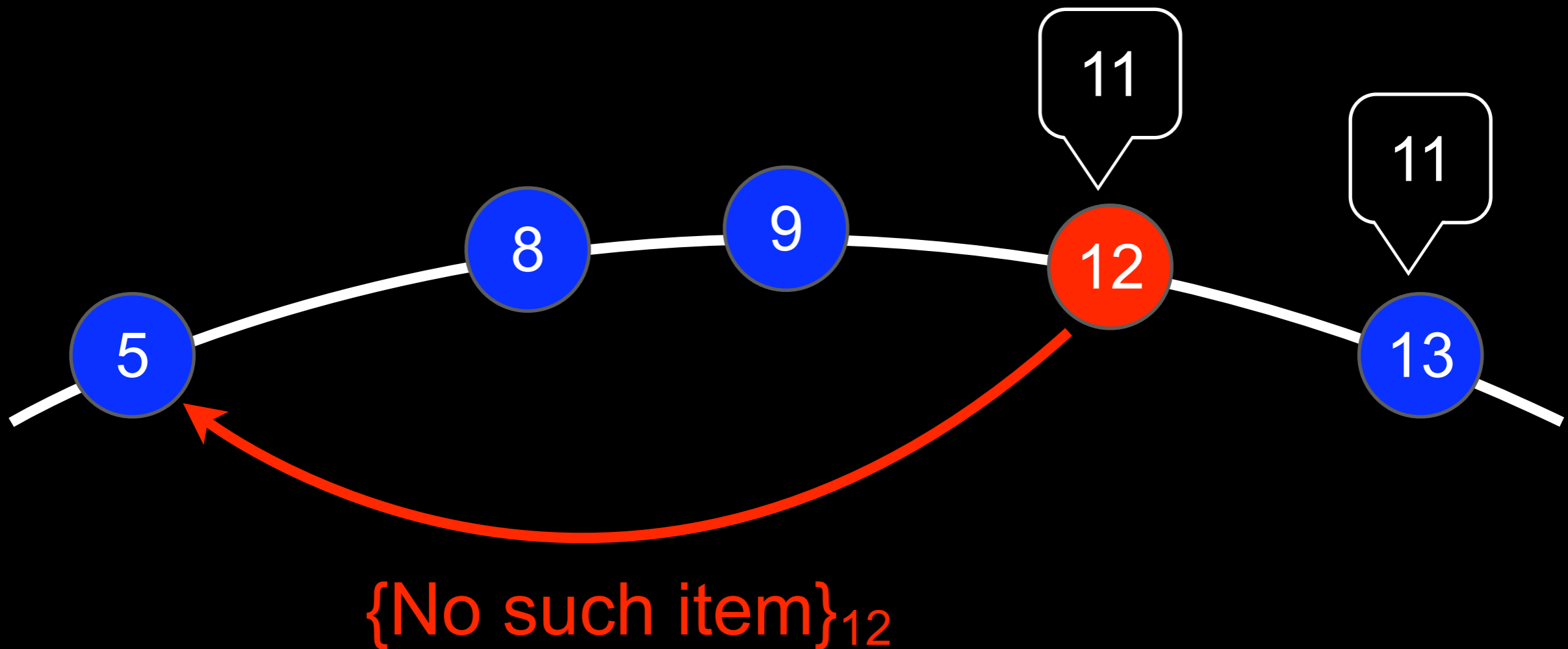
# Retrieve using nCerts

---



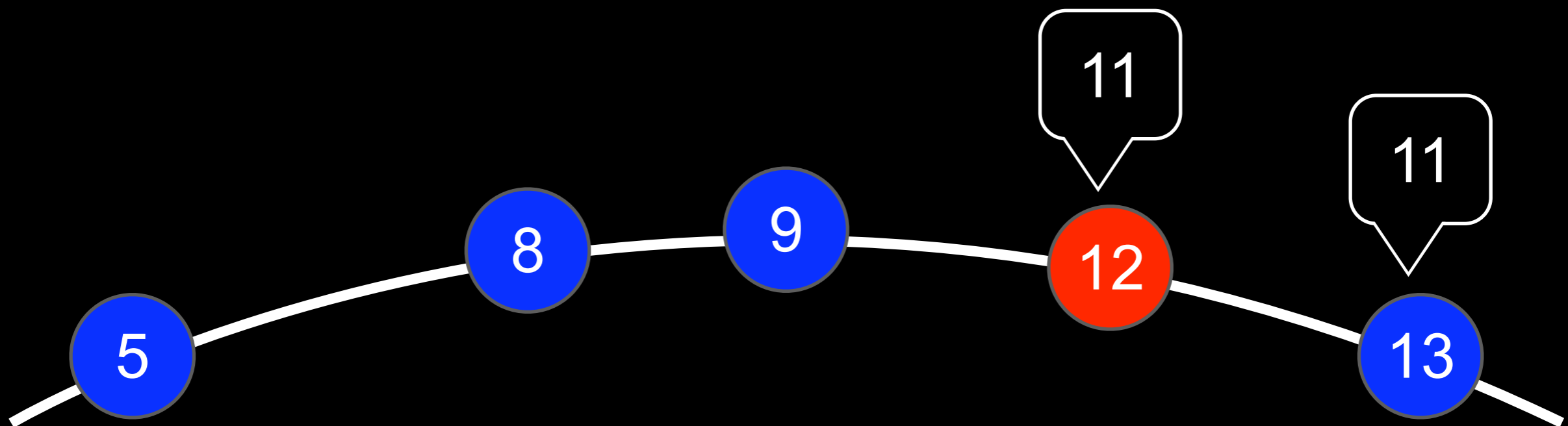
# Retrieve using nCerts

---



# Lookup in NeighborhoodWatch

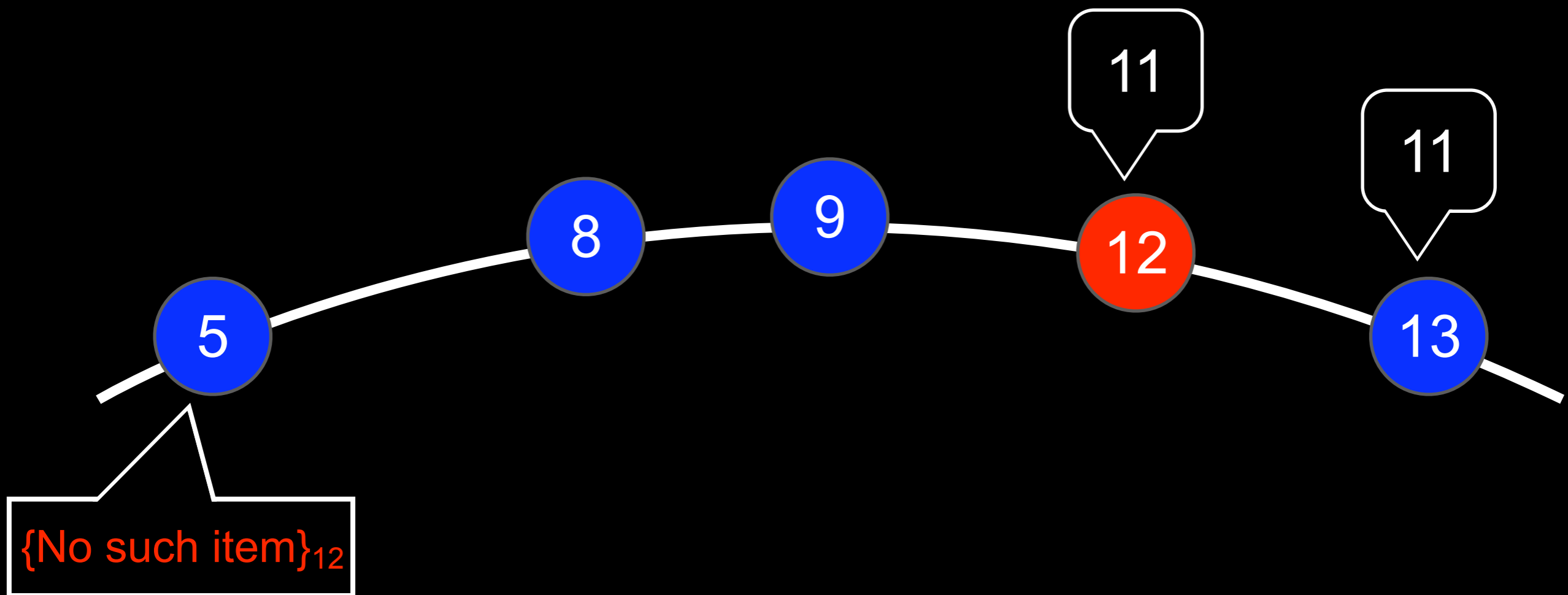
---



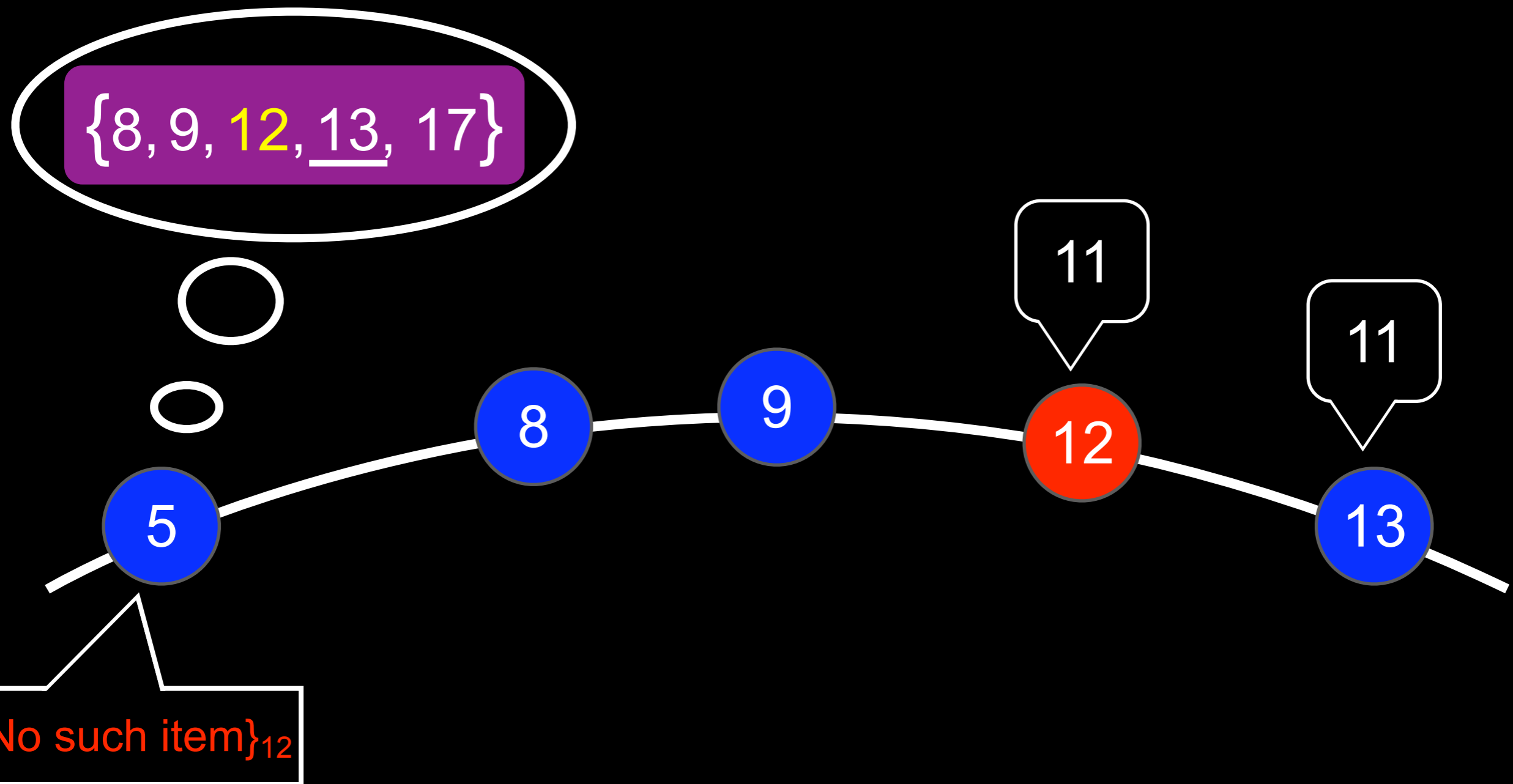
{No such item}<sub>12</sub>

# Lookup in NeighborhoodWatch

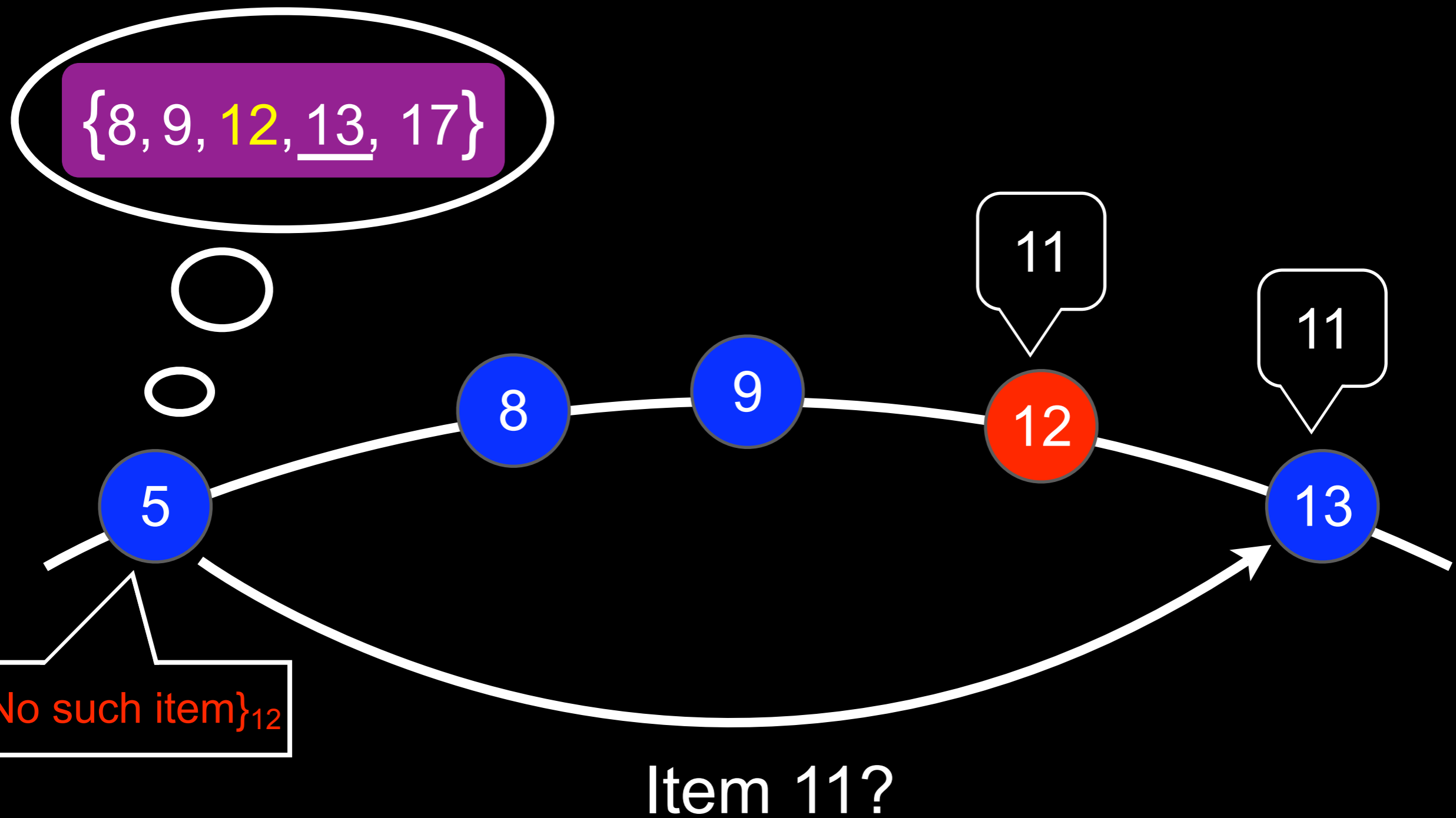
---



# Lookup in NeighborhoodWatch

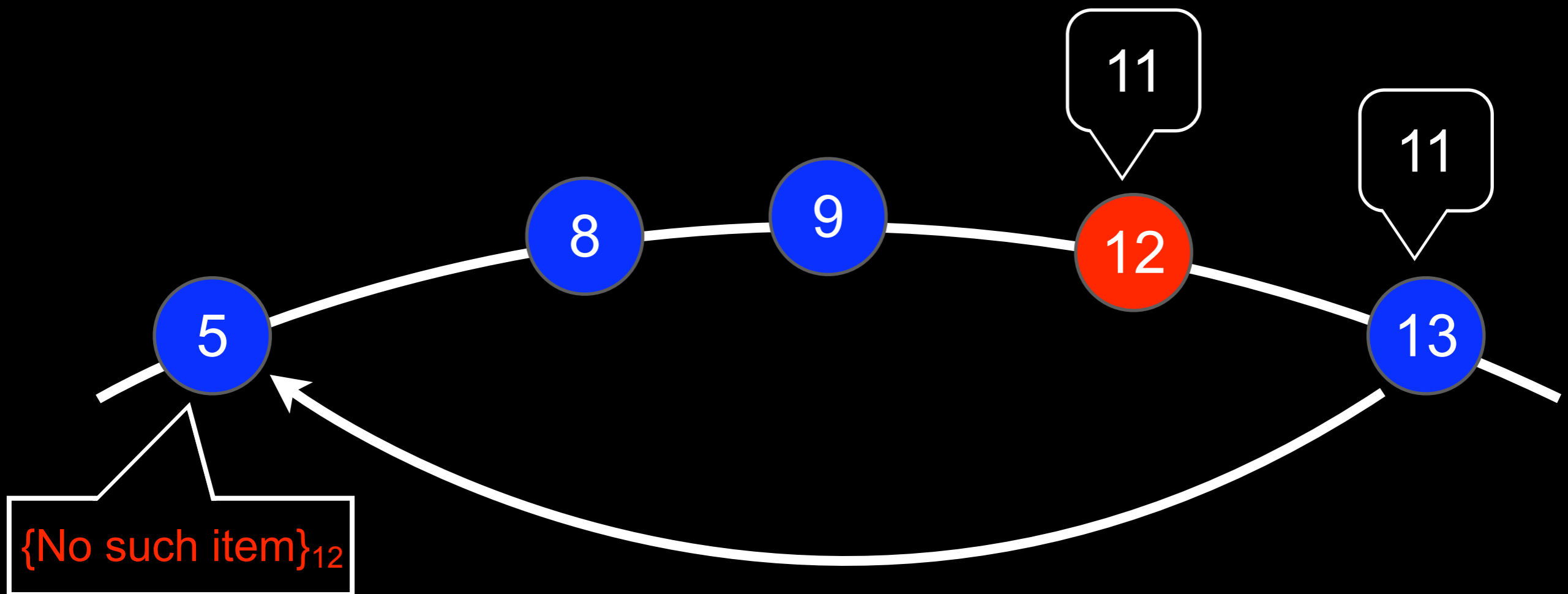


# Lookup in NeighborhoodWatch



# Lookup in NeighborhoodWatch

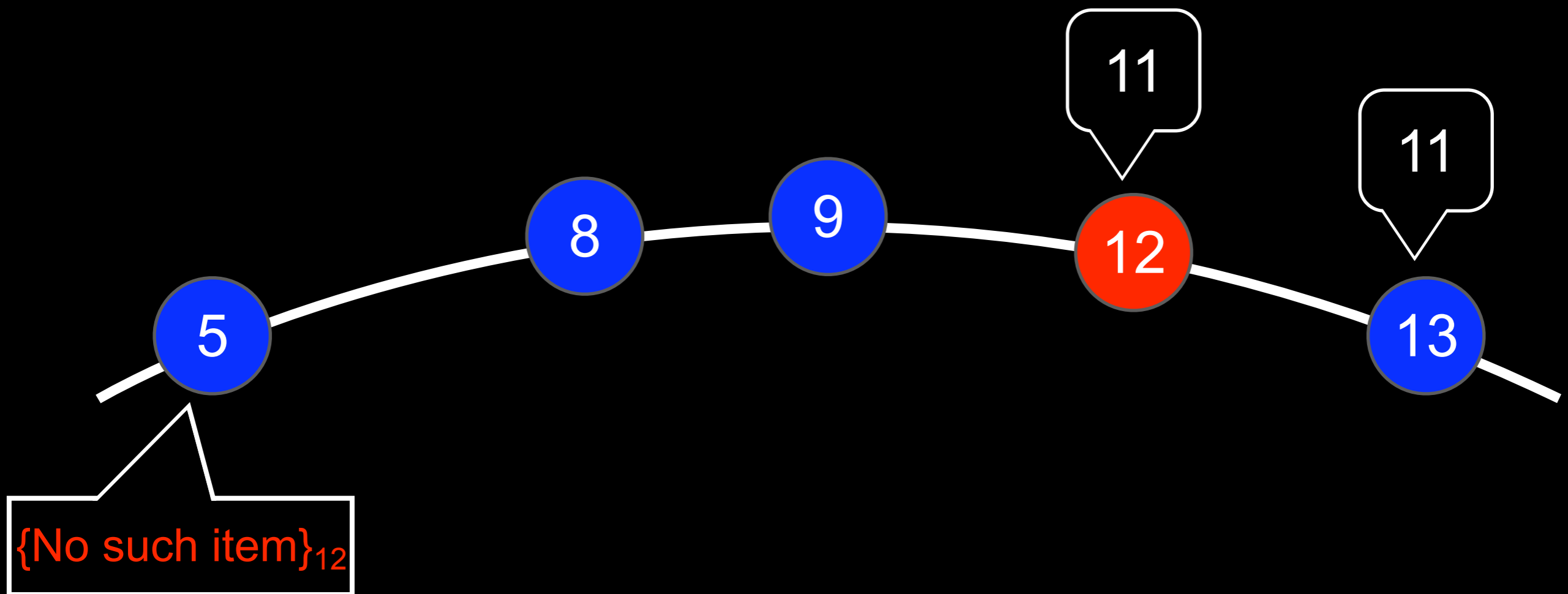
---



Item 11 = “1.2.3.4 blocked”

# Receipts expose bad nodes

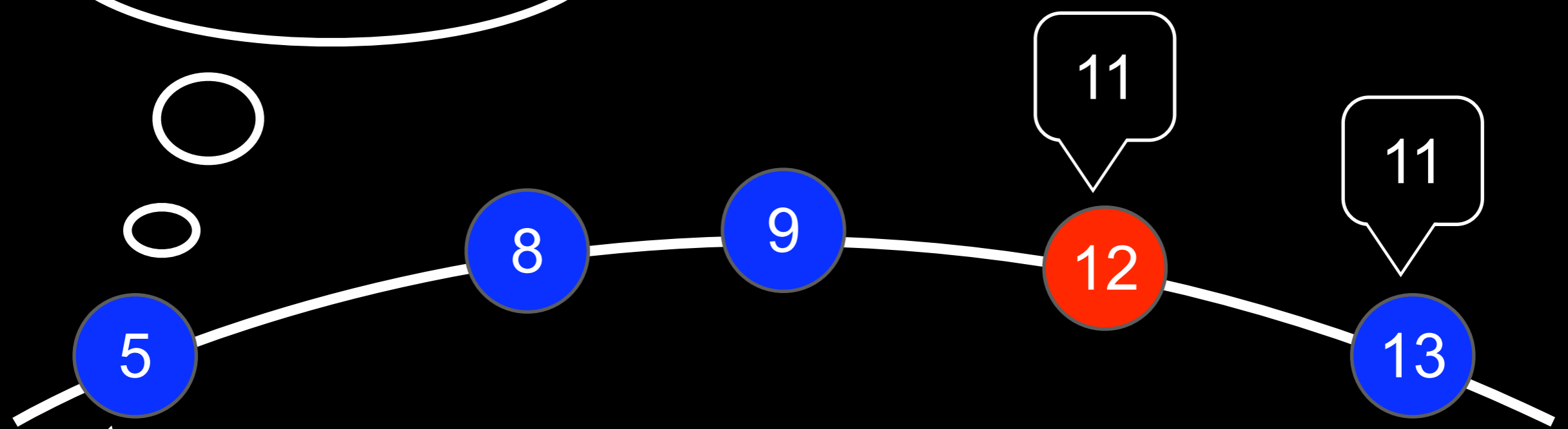
---



# Receipts expose bad nodes

---

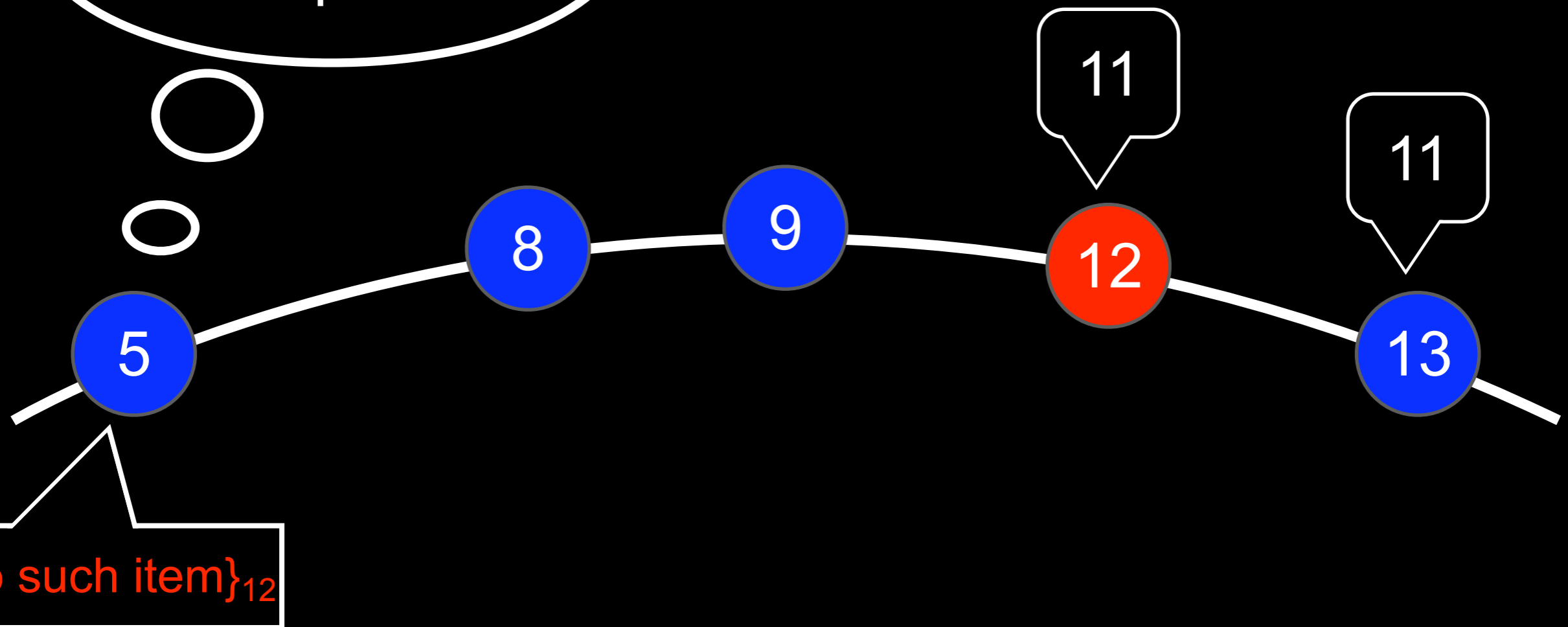
12 likely bad!



`{No such item}_12`

# Receipts expose bad nodes

Find receipt  
and expose 12



# Receipts

---

- Receipts stored in DHT at Hash(item.id, node.id)
  - ✦ Easy to find
  - ✦ Independent of storing node
- If a node suspects a “no such item” response is false, looks for receipt
- Presenting a receipt and a signed, conflicting response to the NCA results in eviction

# Ensuring the invariant

---

- Invariant: adversary cannot corrupt a sequence of  $k+1$  consecutive nodes
- Admission control
  - ✦ Nodes need expensive, long-lived certificate
  - ✦ Node ID = Hash(certicate)
- Choose  $k$  so that for a given DHT size and fraction of bad nodes, the probability of corrupting  $k+1$  consecutive nodes is negligible

# Protecting the NCA

---

- NCA easily distributed - limited state shared across replicas
- Only nodes that are currently in the DHT need to contact the NCA - filter all other traffic
- Alternatively, use the Platypus policy-compliant routing system

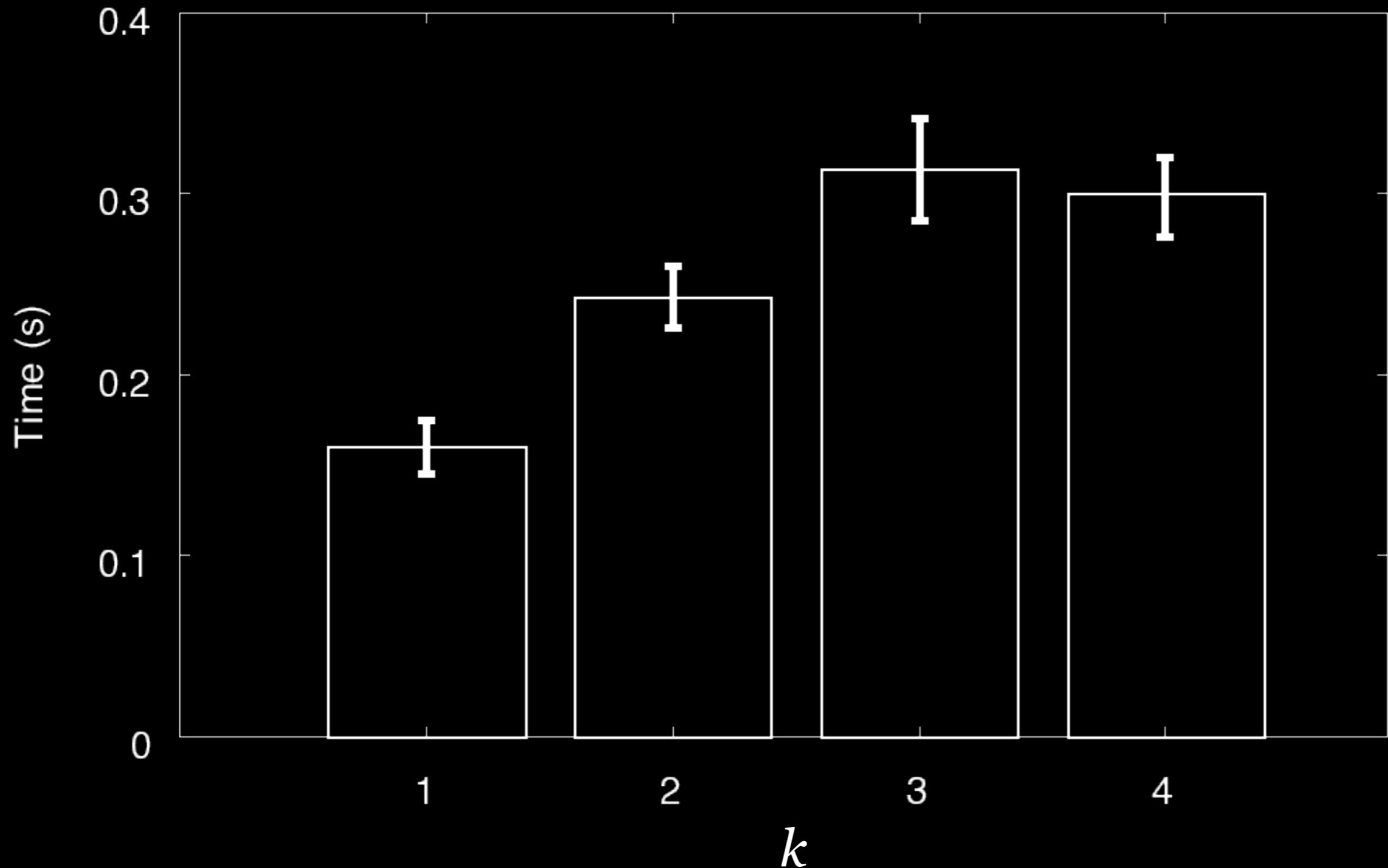
# Implementation

---

- Implemented in Ruby with SWIG wrappers around OpenSSL
- Deployed on PlanetLab
  - ✦ ~75 nodes
  - ✦ 1 NCA, run locally

# Recertification time vs. $k$

---



# Summary

---

- NeighborhoodWatch secure DHT
- Adds security, maintains efficiency
- Distributed blacklist with applications to spam
- Misbehavior can be proven; bad nodes removed from DHT by the NCA
- Novel use of receipts to determine if a node has stored an item