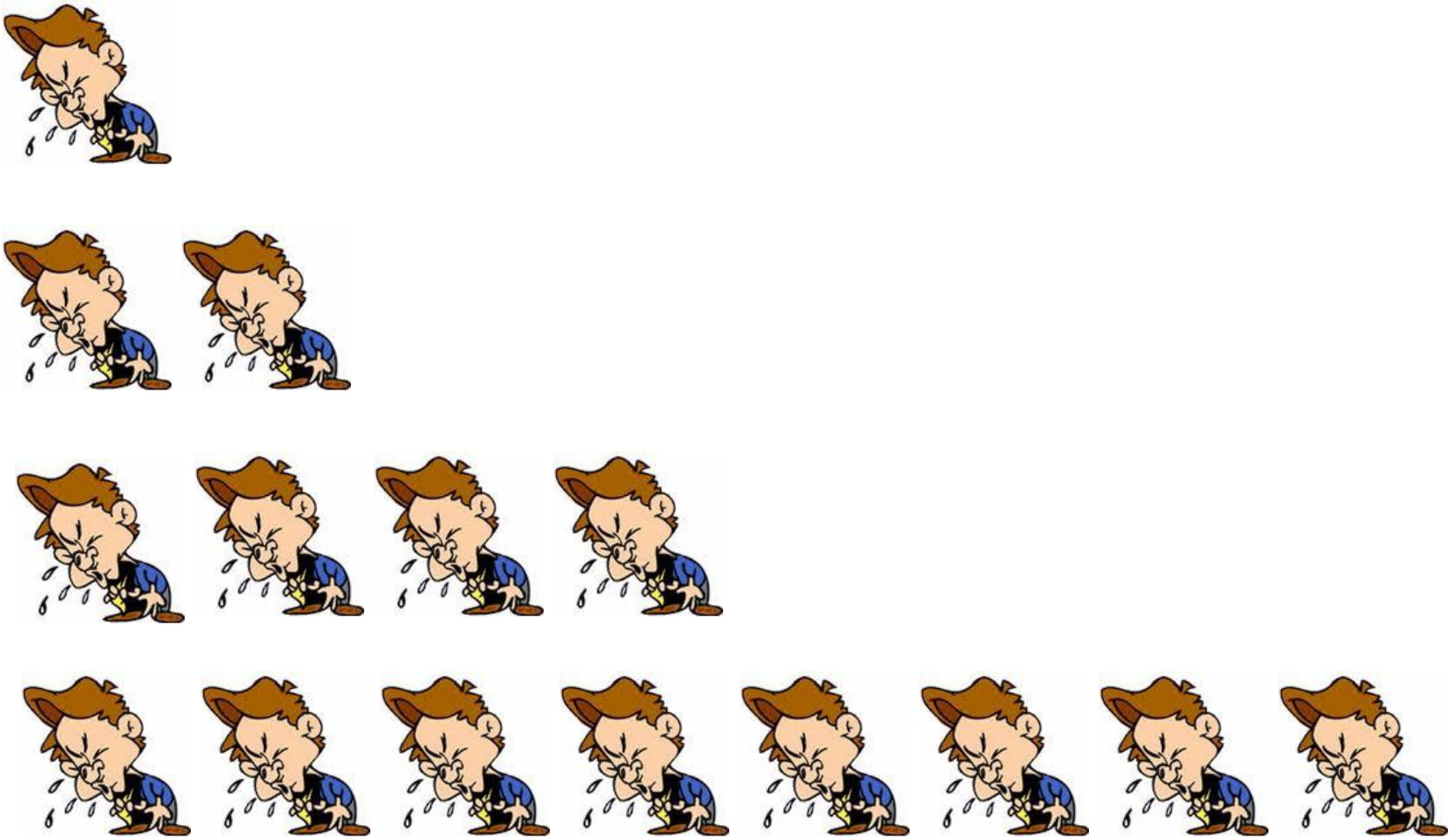
EPIDEMIC TECHNIQUES

Kaushik Parasam
kp397@cornell.edu
03 Nov, 2011

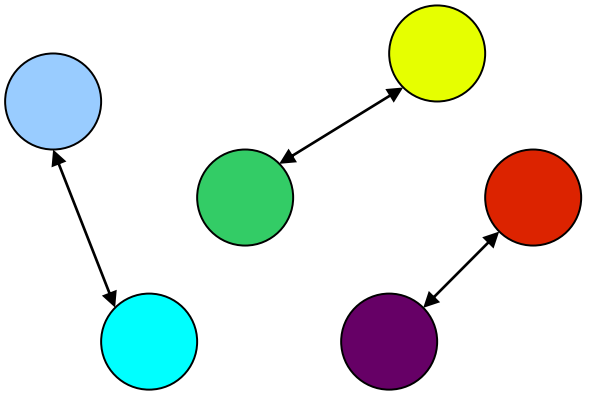
Epidemics



Gossip Communication
a.k.a Epidemic protocol, Rumor Spreading

Gossip protocol is a style of computer-to-computer communication protocol extensively used in distributed systems

- Periodic exchange of bounded size messages
- Pairwise interaction
- Randomization
- *Eventual* consistency
- Exponential (1.8^k) spread in log(n) time.



Keys to success

- Simplicity
- Limited resource usage
- Robustness to failures
- Tunable system behavior

Where does it fit ?

Replication

EPIDEMIC ALGORITHMS FOR REPLICATED DATABASE MAINTENANCE

Alan Demers, Dan Greene, Carl Hauser, Wes Irish, John Larson,
Scott Shenker, Howard Sturgis, Dan Swinehart, and Doug Terry

**MANAGING UPDATE CONFLICTS IN BAYOU,
A WEAKLY CONNECTED REPLICATED STORAGE SYSTEM**

Doug Terry, Marvin M Theimer, Karin Petersen, Alan Demers,
Mike J Spreitzer and Carl Hauser

Xerox Palo Alto Research Center

Epidemic Algorithms For Replicated Database Maintenance

Alan Demers Professor at Cornell University

Dan Greene At Xerox PARC – Vehicle networks

Carl Hauser Associate Professor, Washington State University

Wes Irish Coyote Hill Consulting

Scott Shenker Professor at UCBerkeley

Doug Terry Microsoft Research

John Larson, Howard Sturgis, Dan Swinehart

Takeaways

Takeaways

- Novel method – distributed systems
- Epidemiology works !
- Small work, large impact
- Many implementations down the line...
    - Aggregation in Large Dynamic Networks [1]
    - Kelips: P2P DHT [2]
    - Storage systems [3]
    - T-Man: Fast topology construction [4]

    and many more …....

- Security ?
- Evaluation ? Not much...

Motivation

- Clearinghouse servers on Xerox Corporate Internet (CIN)
    - Hundreds of ethernets connected by gateways and phone lines
    - Hierarchical name space – domains
- Remailing - a big overhead

$$\forall\, s, s' \in S : s.ValueOf = s'.ValueOf$$

The Two Considerations

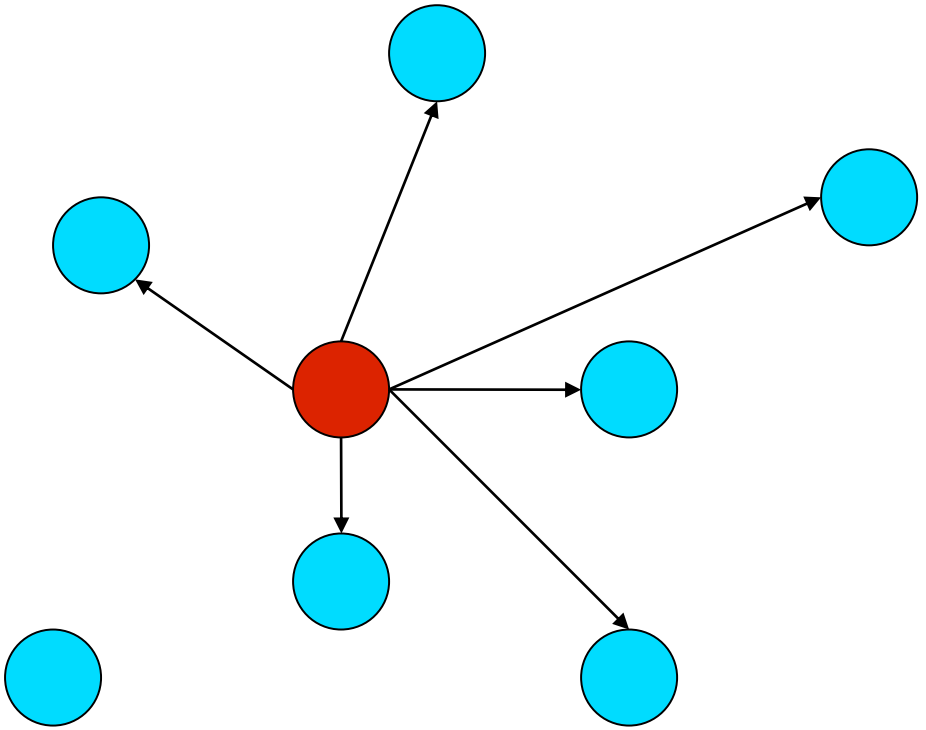
- Time required for an update to propogate
- Network traffic

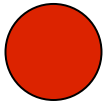
Epidemiology

- Infective – Knows the update and *spreads* it
- Susceptible – Can get *infected* with the update
- Removed – Knows the update but *not* willing to spread it

Under the hood

Direct Mail – Updates immediately mailed from an entry site to all other sites.



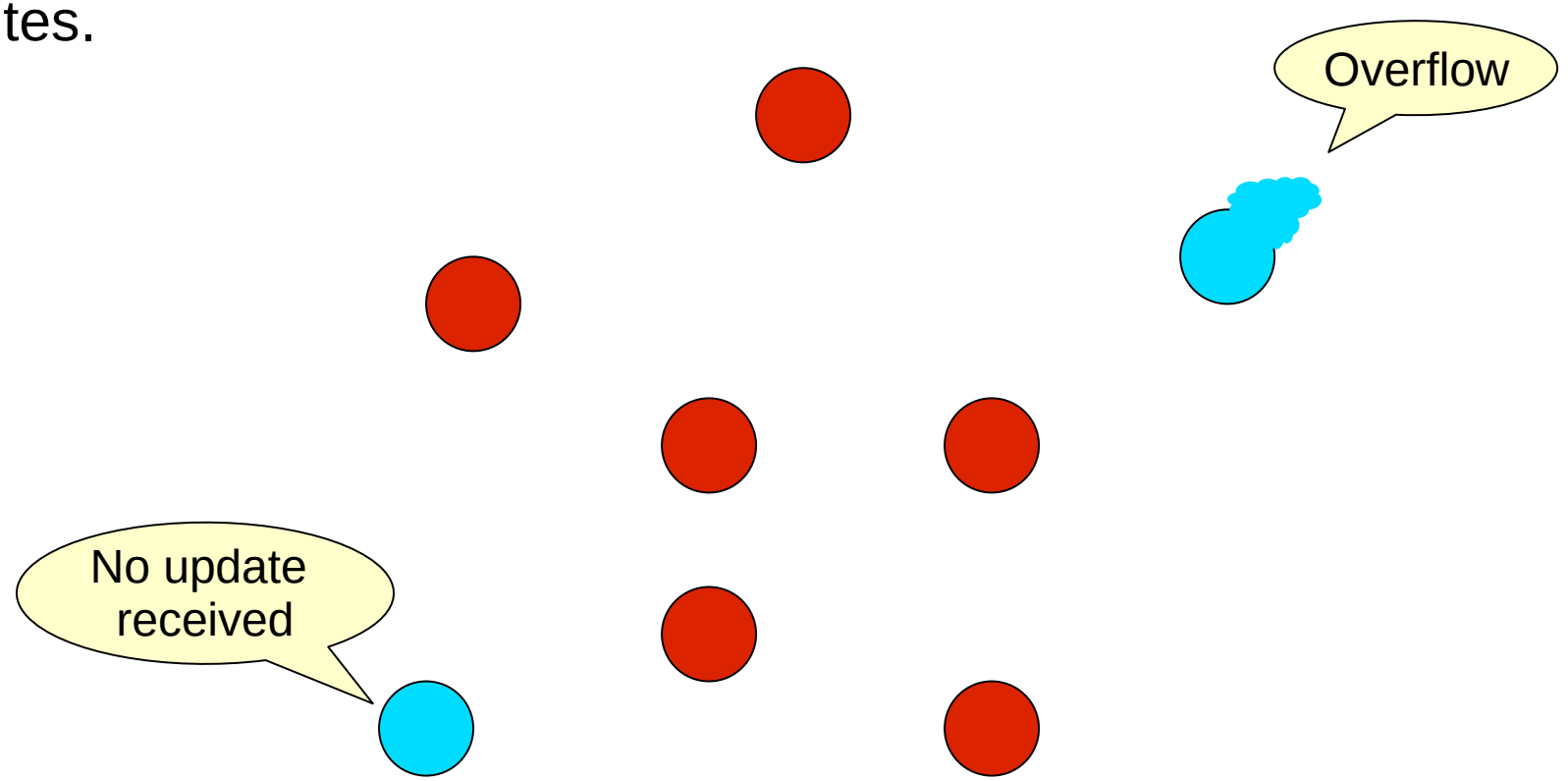


- Infected

- Susceptible

Under the hood

Direct Mail – Updates immediately mailed from an entry site to all other sites.



- *Lacks node topology*
- *Message is lost*
- *Traffic proportional to the number of sites * average distance between sites*

- Infected
- Susceptible

Under the hood

Anti-Entropy – Periodically pick a random site and exchange messages

Variants:

- Push

```
IF s.ValueOf.t > s'.ValueOf.t THEN
  s'.ValueOf <- s.ValueOf
```

- Pull

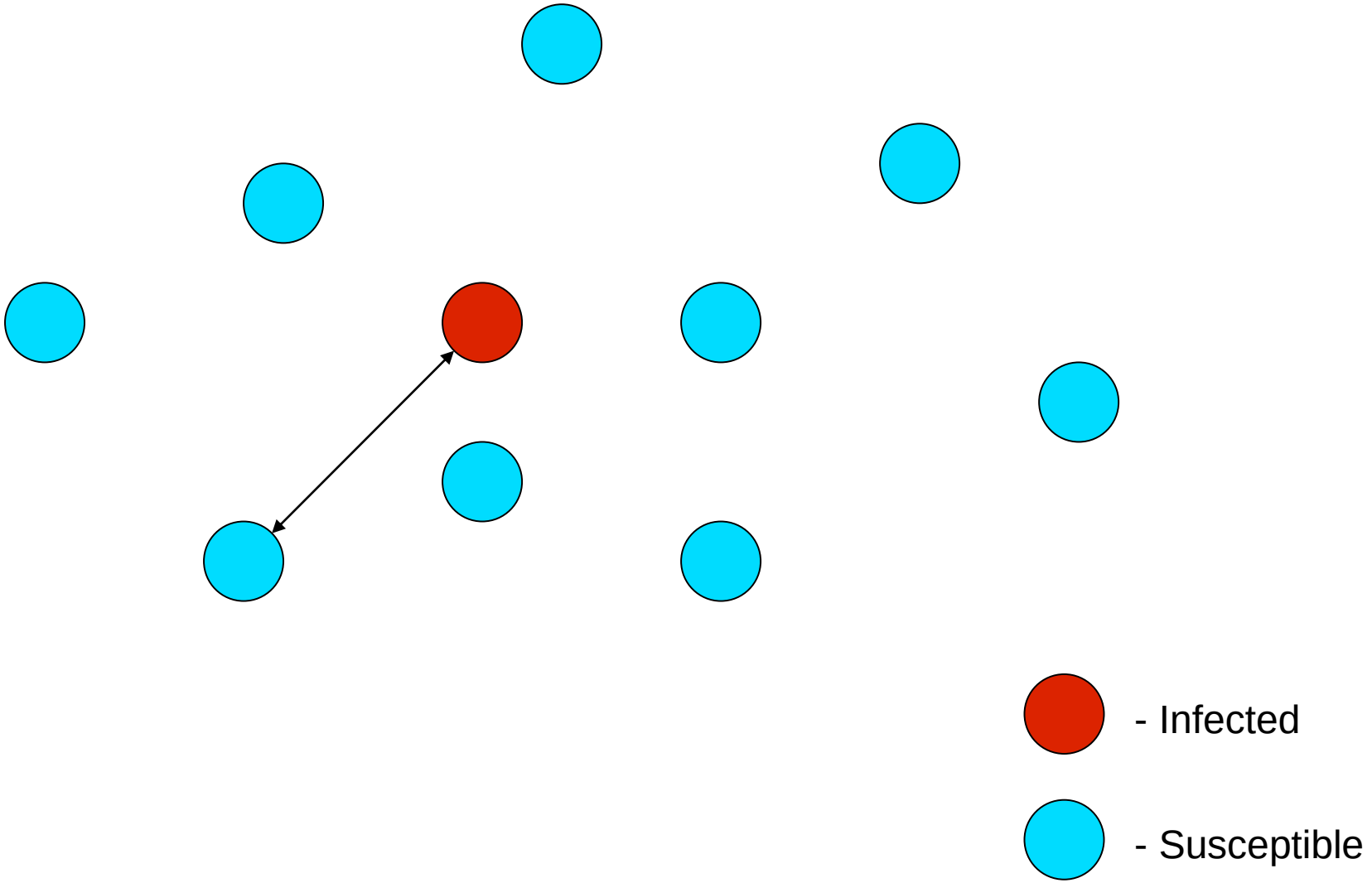
```
IF s.ValueOf.t < s'.ValueOf.t THEN
    s.ValueOf <- s'.ValueOf
```

- Push-pull

```
SELECT TRUE FROM
s.ValueOf.t > s'.ValueOf.t => s'.ValueOf <- s.ValueOf
s.ValueOf.t < s'.ValueOf.t => s.ValueOf <- s'.ValueOf
ENDCASE -> NULL:
```

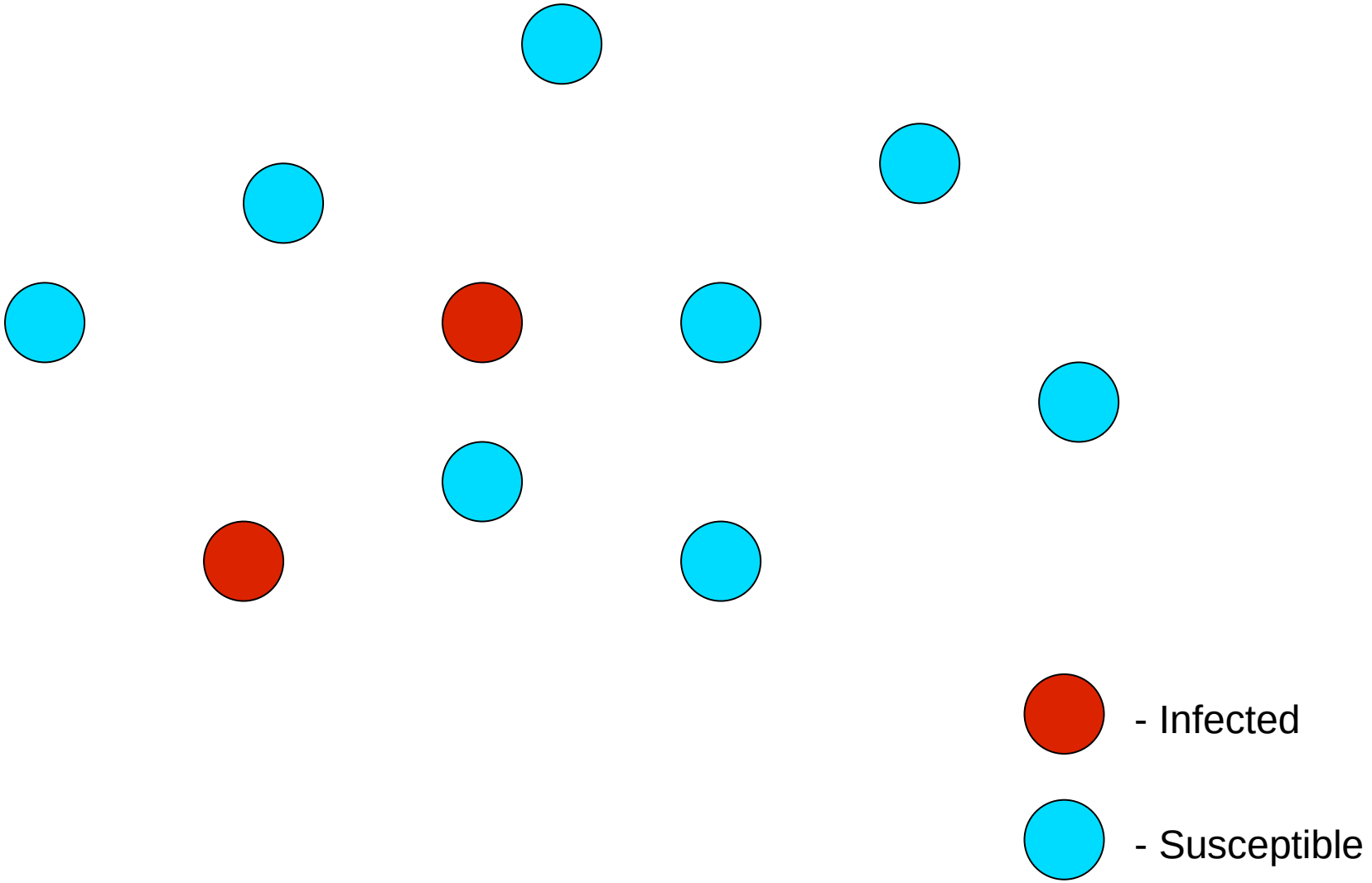
Under the hood

Anti-Entropy



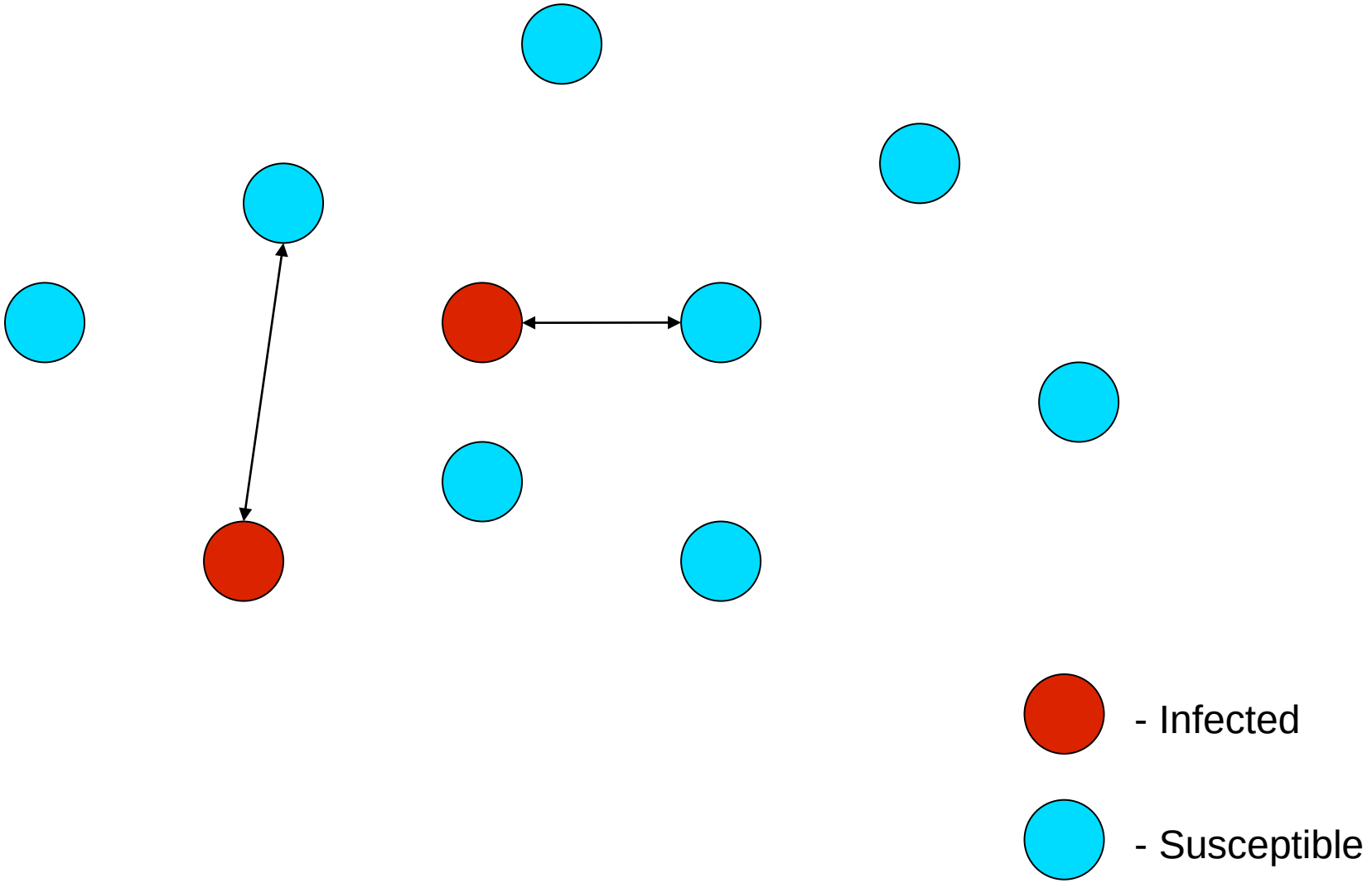
Under the hood

Anti-Entropy



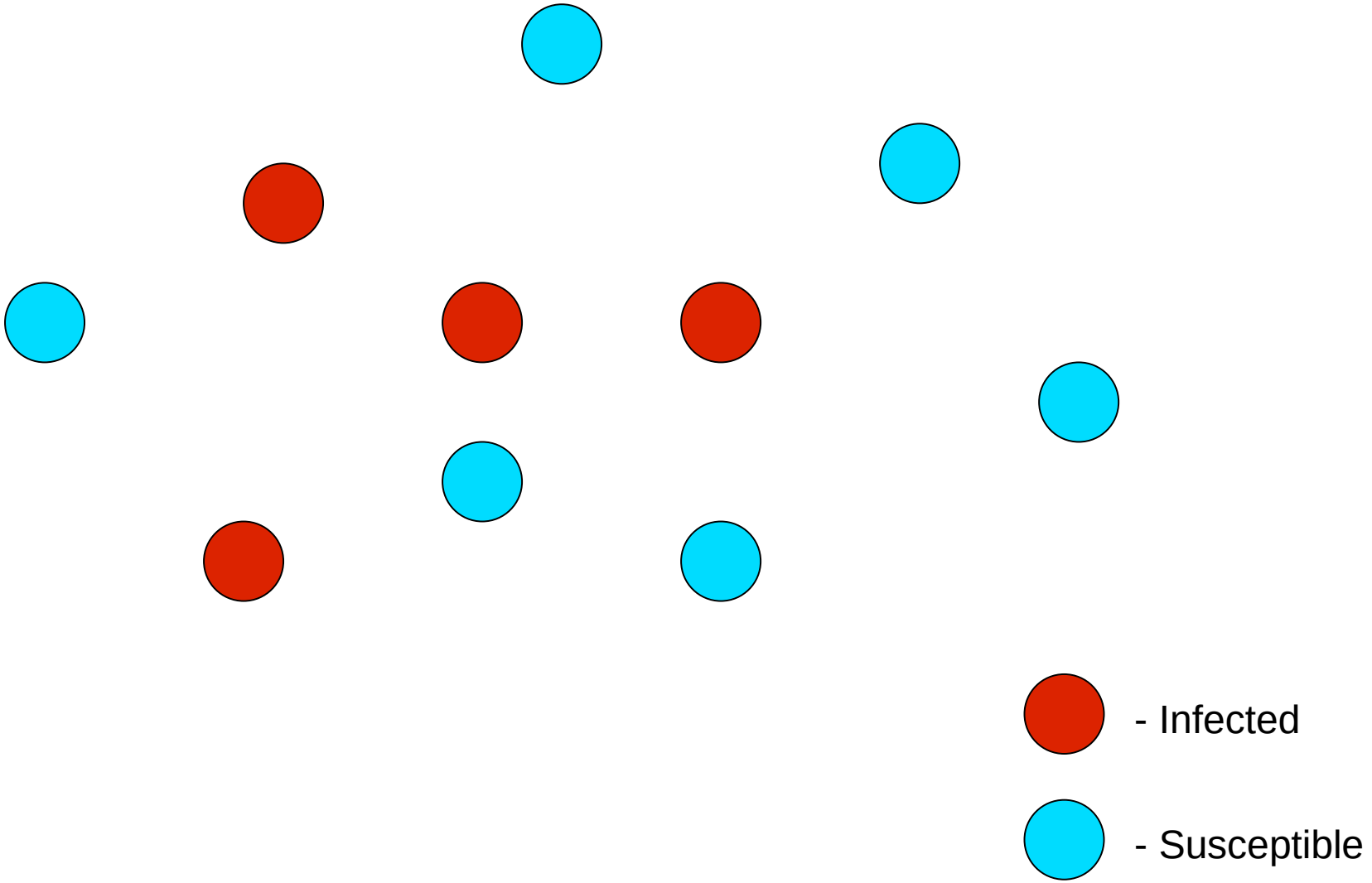
Under the hood

Anti-Entropy



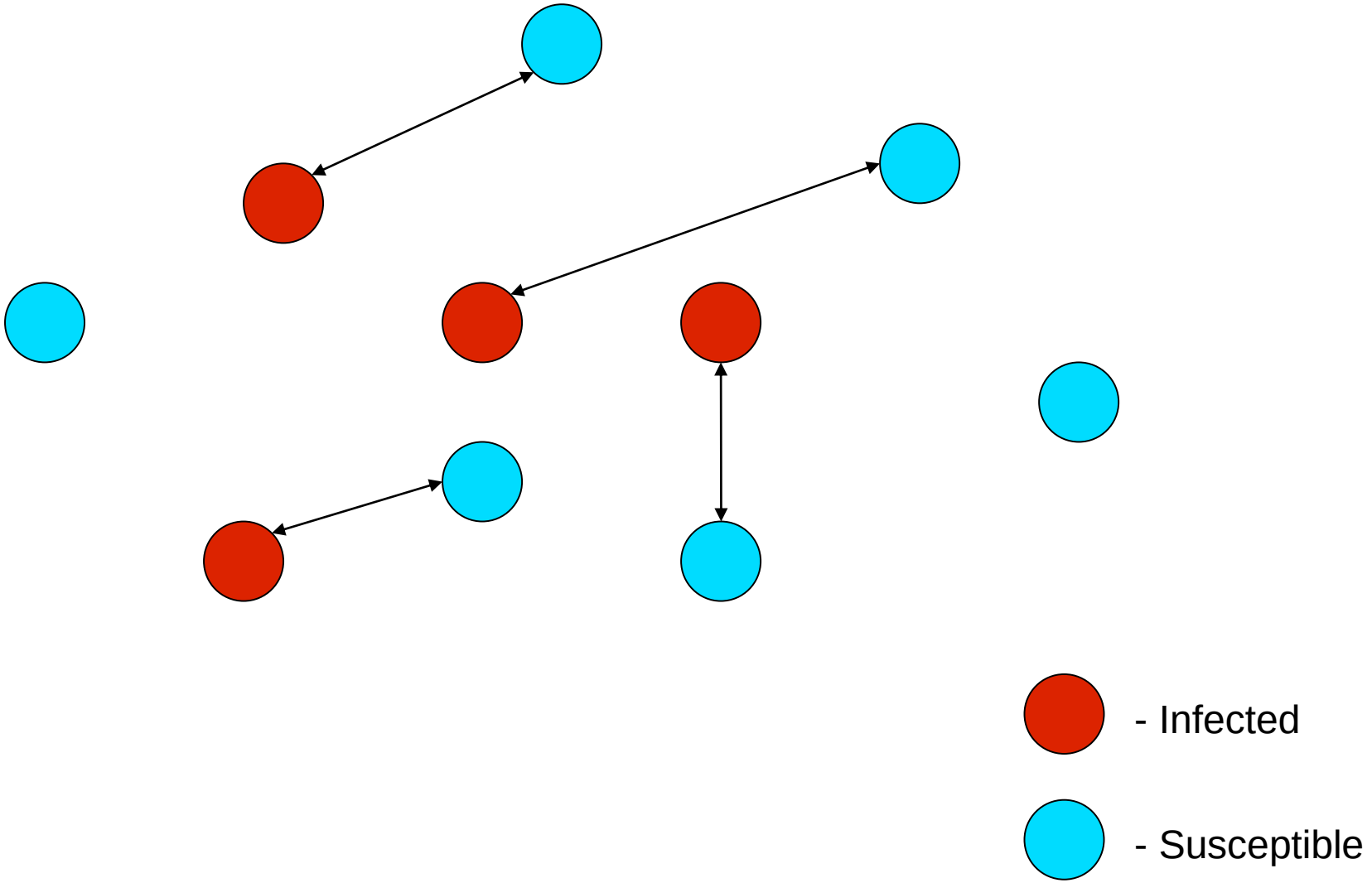
Under the hood

Anti-Entropy



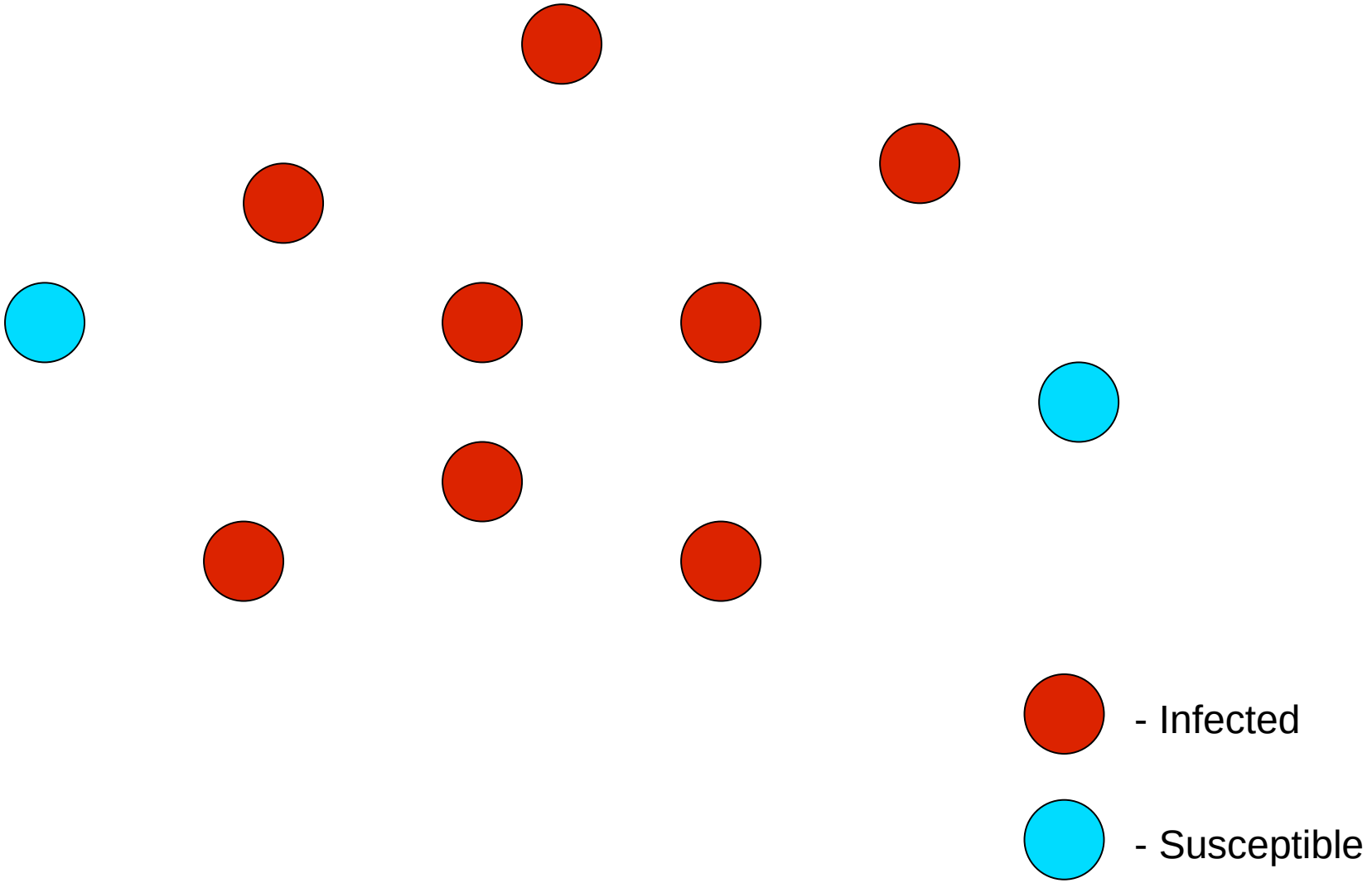
Under the hood

Anti-Entropy



Under the hood

Anti-Entropy



Under the hood

Anti-Entropy:

- Eventually everyone is infected
- Efficient and reliable, but propagates slower than direct mail

Problems ?

Possible Improvements:

- Checksums
- Inverted Index

Connection Limit, Hunting

Under the hood

Rumor mongering (Complex epidemic)

- When a site receives a new update – Hot rumor
- Periodically choose another site at random to infect other sites
- If enough number of sites have seen the rumor – Remove

Blind vs. Feedback

- Probability 1/k
- Depending on receiver

Counter vs. Coin

- Loses interest after k unnecessary contacts
- Probability on k coin tosses

Table 1. Performance of an epidemic on 1000 sites using response and counters.

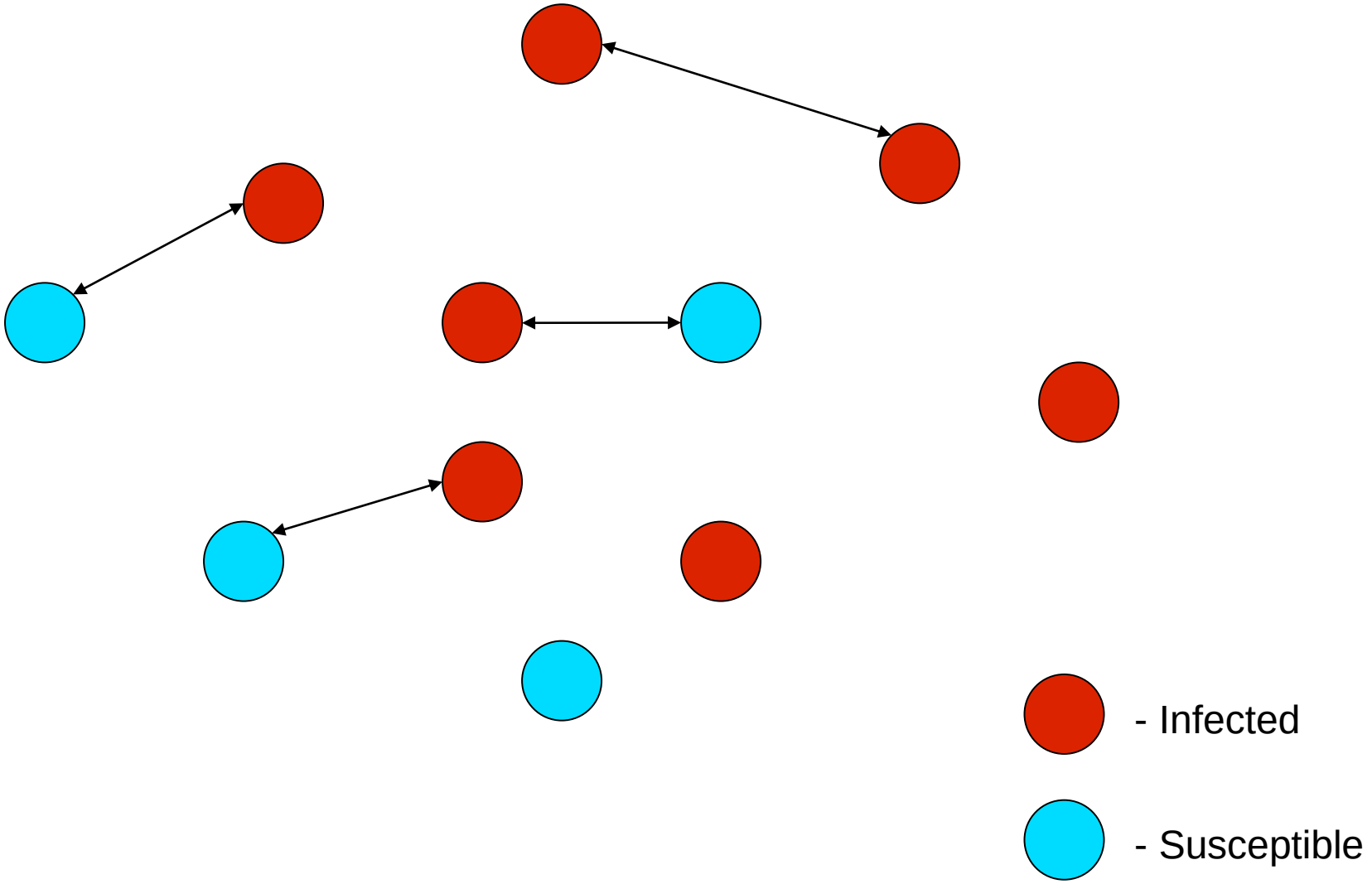
Counter	Residue	Traffic	Convergence	
k	s	m	t_{ave}	t_{last}
1	0.176	1.74	11.0	16.8
2	0.037	3.30	12.1	16.9
3	0.011	4.53	12.5	17.4
4	0.0036	5.64	12.7	17.5
5	0.0012	6.68	12.8	17.7

Table 2. Performance of an epidemic on 1000 sites using blind and probabilistic.

Counter	Residue	Traffic	Convergence	
k	s	m	t_{ave}	t_{last}
1	0.960	0.04	19	38
2	0.205	1.59	17	33
3	0.060	2.82	15	32
4	0.021	3.91	14.1	32
5	0.008	4.95	13.8	32

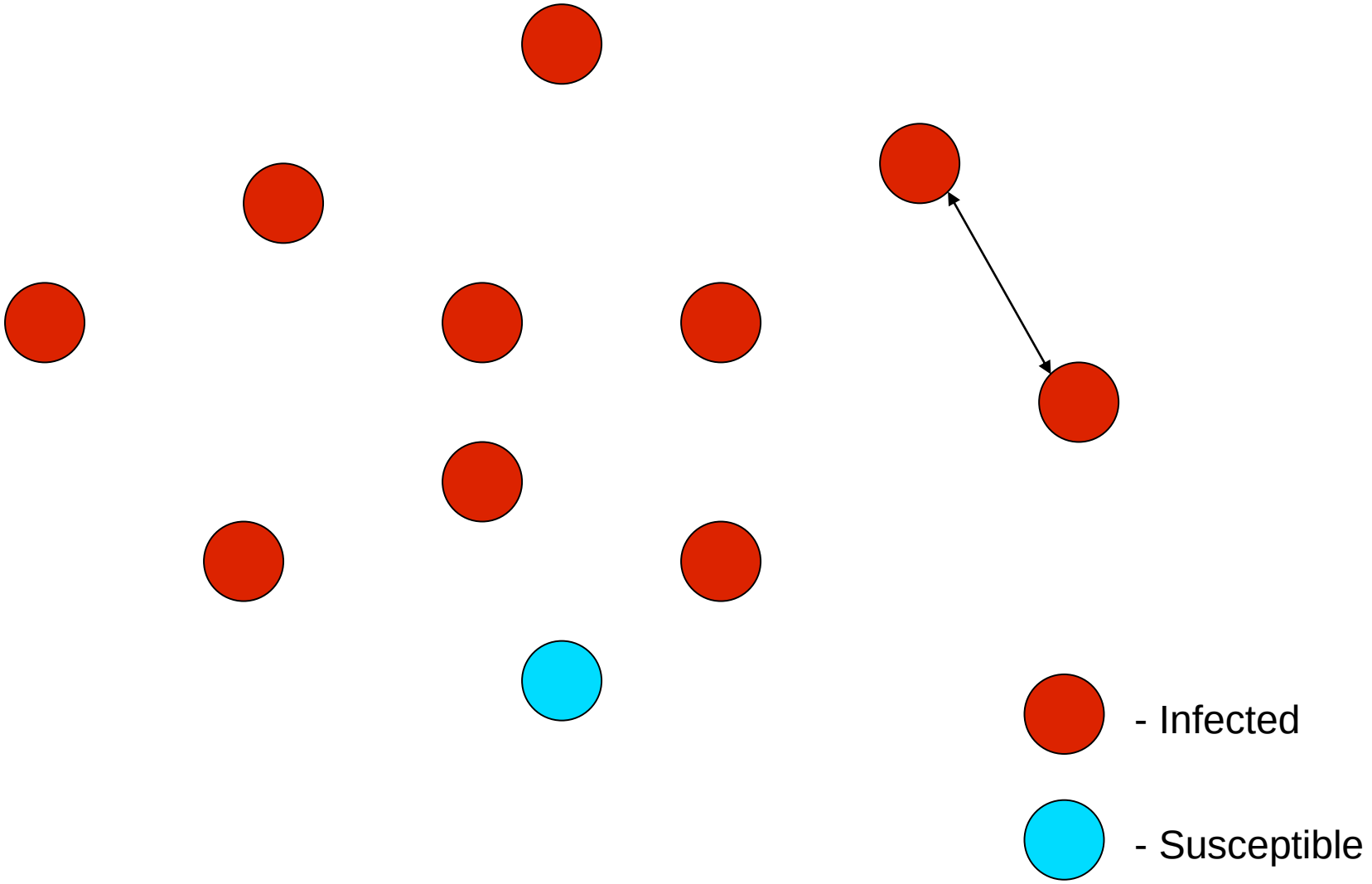
Under the hood

Rumor mongering



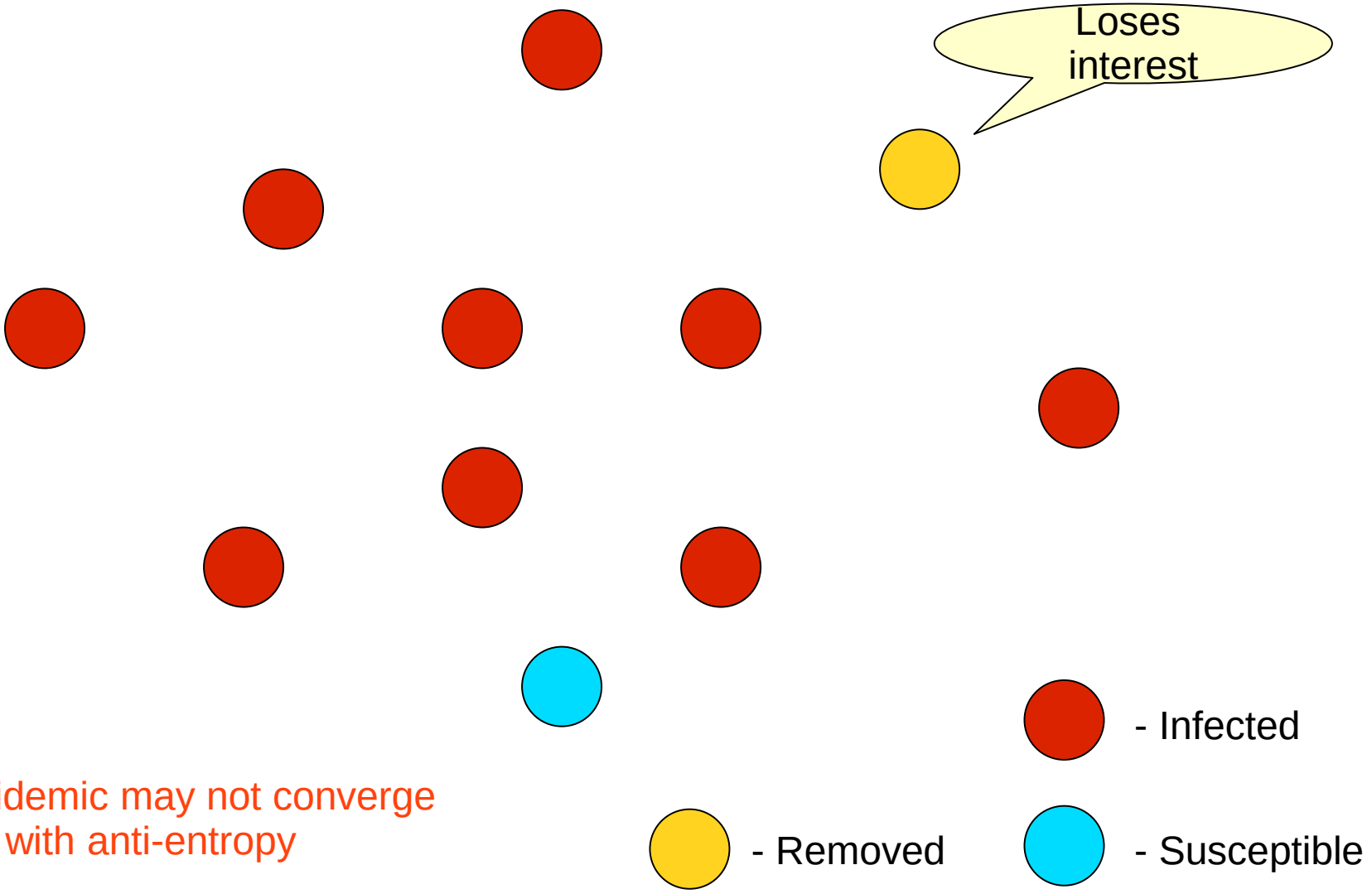
Under the hood

Rumor mongering



Under the hood

Rumor mongering



Complex epidemic may not converge
- Backed up with anti-entropy

Under the hood

Death certificates

- Deletion - harder to manage
- Usually done with a timestamp

Spatial distribution

Biased Gossip

- Favoring nearby sites over farther ones ? But...
- d^{-a}

Table 4. Simulation results for anti-entropy, no connection limit.

Spatial Distribution	t_{last}	t_{ave}	Compare Traffic		Update Traffic	
			Average	Bushey	Average	Bushey
uniform	7.81	5.27	5.87	75.74	5.85	74.43
$a = 1.2$	10.04	6.29	2.00	11.19	2.61	17.52
$a = 1.4$	10.31	6.39	1.93	8.77	2.49	14.10
$a = 1.6$	10.94	6.70	1.71	5.72	2.27	10.88
$a = 1.8$	11.97	7.21	1.52	3.74	2.07	7.68
$a = 2.0$	13.32	7.76	1.36	2.38	1.89	5.87

Table 5. Simulation results for anti-entropy, connection limit 1.

Spatial Distribution	t_{last}	t_{ave}	Compare Traffic		Update Traffic	
			Average	Bushey	Average	Bushey
uniform	11.00	6.97	3.71	47.54	5.83	75.17
$a = 1.2$	16.89	9.92	1.14	6.39	2.69	18.03
$a = 1.4$	17.34	10.15	1.08	4.68	2.55	13.68
$a = 1.6$	19.06	11.06	0.94	2.90	2.32	10.20
$a = 1.8$	21.46	12.37	0.82	1.68	2.12	7.03
$a = 2.0$	24.64	14.14	0.72	0.94	1.94	4.85

Results with anti-entropy

Table 6. Simulation results for *push-pull* rumor mongering.

Spatial Dist	k	t_{last}	t_{ave}	Compare Traffic		Update Traffic	
				Avg	Bushey	Avg	Bushey
uniform	4	7.83	5.32	8.87	114.0	5.84	75.87
$a = 1.2$	6	10.14	6.33	3.20	18.0	2.60	17.25
$a = 1.4$	5	10.27	6.31	2.86	13.0	2.49	14.05
$a = 1.6$	8	11.24	6.90	2.94	9.80	2.27	10.54
$a = 1.8$	7	12.04	7.24	2.40	5.91	2.08	7.69
$a = 2.0$	6	13.09	7.74	1.99	3.44	1.90	5.94

Results with rumor-mongering(push-pull)

References:

[1] Gossip-based aggregation in large dynamic networks. Márk Jelasity, Alberto Montresor, and Ozalp Babaoglu. ACM Transactions on Computer Systems, 23(3):219–252, August 2005.

[2] Kelips: Building an Efficient and Stable P2P DHT Through Increased Memory and Background Overhead. Indranil Gupta, Ken Birman, Prakash Linga, Al Demers, Robbert van Renesse. Proc. 2nd International Workshop on Peer-to-Peer Systems (IPTPS '03)

[3] ACM SIGOPS Operating Systems Review - Gossip-based computer networking archive
Volume 41 Issue 5, October 2007

[4] T-Man: Gossip-based fast overlay topology construction. Márk Jelasity, Alberto Montresor, and Ozalp Babaoglu. Computer Networks, 53(13):2321–2339, 2009.

Revisiting CAP Theorem

One-copy serializability

- Stronger consistency

(AP) at stake

- Weaker consistency

Ordering

- Vector clocks
- Logical clocks

Causal  Total Ordering - Bayou

Managing Update Conflicts in Bayou - a Weakly Connected Replicated Storage System

Alan Demers Professor at Cornell University

Carl Hauser Associate Professor, Washington State University

Doug Terry Microsoft Research

Marvin Theimer Amazon Web Services

Mike Spreitzer IBM, Watson Research

Karin Petersen

Takeaways

Takeaways

- Application-specific conflict detection, a caveat ?
- Partial and multi-object updates
- Stable resolutions
- Security

- Not much evaluation results

Purpose

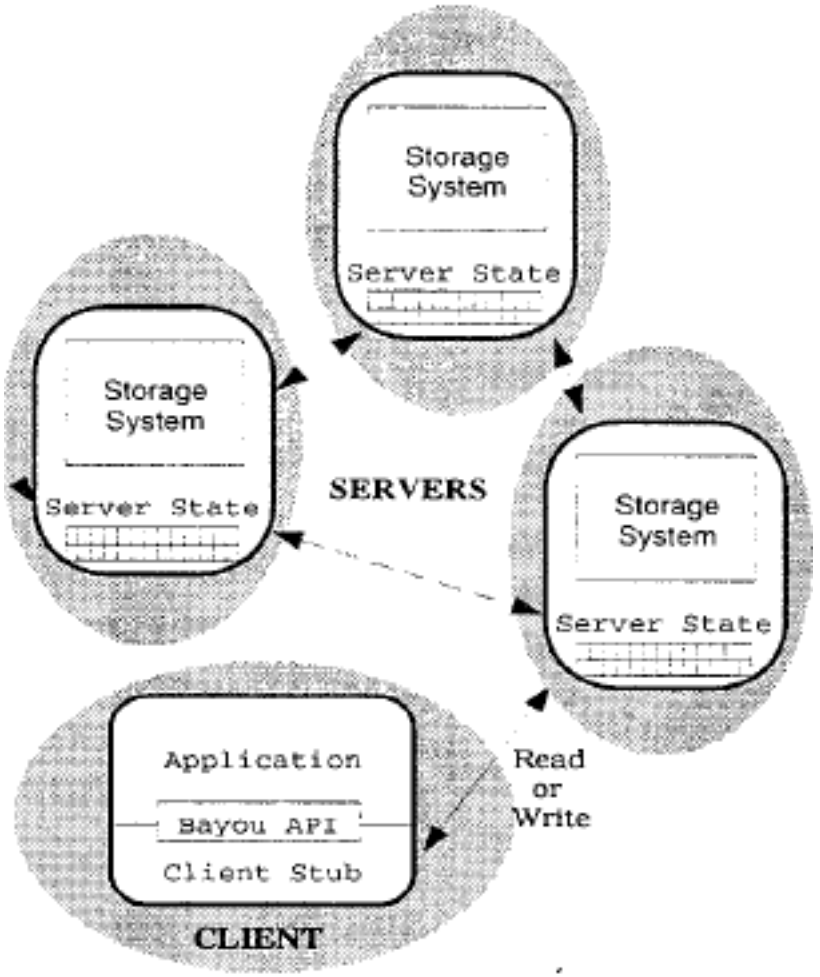
- Mobile, weakly connected
- Non-real time collaborative applications

Bayou Model

- Clients use Bayou's API's
Read-any / write-any

- Replication by anti-entropy

- Session guarantees



Bayou System model

Conflict Detection And Resolution

- Application-specific

Policies to be given by applications to the servers

- Dependency checks

Read-write conflicts detection, a positive

- Merge procedures

Resolve the conflicts

Are these an *overhead* ?

More on Bayou

- Eventual consistency

Notion of logical clocks

- Write stability

- Commitment

Commit procedure
Primary commit scheme

System Implementation

- Write log
- Tuple store
  - Committed / Full views
- Undo log

- Access control - D

Public-key cryptography, ACCs

Table 1: Size of Bayou Storage System for the Bibliographic Database with 1550 Entries

(sizes in Kilobytes)

Number of Tentative Writes	0 (none)	50	100	500	1550 (all)
Write Log	9	129	259	1302	4028
Tuple Store Ckpt	396	384	371	269	1
Total	**405**	**513**	**630**	**1571**	**4029**
Factor to 368K bibtex source	1.1	1.39	1.71	4.27	10.95

Table 2: Performance of the Bayou Storage System for Operations on Tentative Writes in the Write Log
(times in milliseconds with standard deviations in parentheses)

Tentative Writes	0	50		100		500		1550	
	Server running on a Sun SPARC/20 with Sunos								
Undo all	0	31	(6)	70	(20)	330	(155)	866	(195)
(avg. per Write)		.62		.7		.66		.56	
Redo all	0	237	(85)	611	(302)	2796	(830)	7838	(1094)
(avg. per Write)		4.74		6.11		5.59		5.05	
	Server running on a Gateway Liberty Laptop with Linux								
Undo all	0	47	(3)	104	(7)	482	(15)	1288	(62)
(avg. per Write)		.94		1.04		.96		.83	
Redo all	0	302	(91)	705	(134)	3504	(264)	9920	(294)
(avg. per Write)		6.04		7.05		7.01		6.4	

Table 3: Performance of the Bayou Client Operations
(times in milliseconds with standard deviations in parentheses)

Server Client	Sun SPARC/20 same as server		Gateway Liberty same as server		Sun SPARC/20 Gateway Liberty	
Read: 1 tuple	27	(19)	38	(5)	23	(4)
100 tuples	206	(20)	358	(28)	244	(10)
Write: no conflict	159	(32)	212	(29)	177	(22)
with conflict	207	(37)	372	(17)	223	(40)

References: All concepts / results are taken from the following two papers

- Epidemic algorithms for replicated database maintenance. Alan Demers, et al. Proc. 6th ACM PODC, Vancouver BC, 1987.

- SOSP '95 Proceedings of the fifteenth ACM symposium on Operating systems principles ACM New York, NY, USA ©1995 / ACM SIGOPS Operating Systems Review Homepage Volume 29 Issue 5, Dec. 3, 1995

Others:
http://en.wikipedia.org/wiki/Gossip_protocol

Thank You

Questions ?