Epidemic Algorithms for replicated Database maintenance

Alan Demers et al
Xerox Palo Alto Research Center, PODC 87

Presented by: Harshit Dokania

Motivation

- Replicated Databases

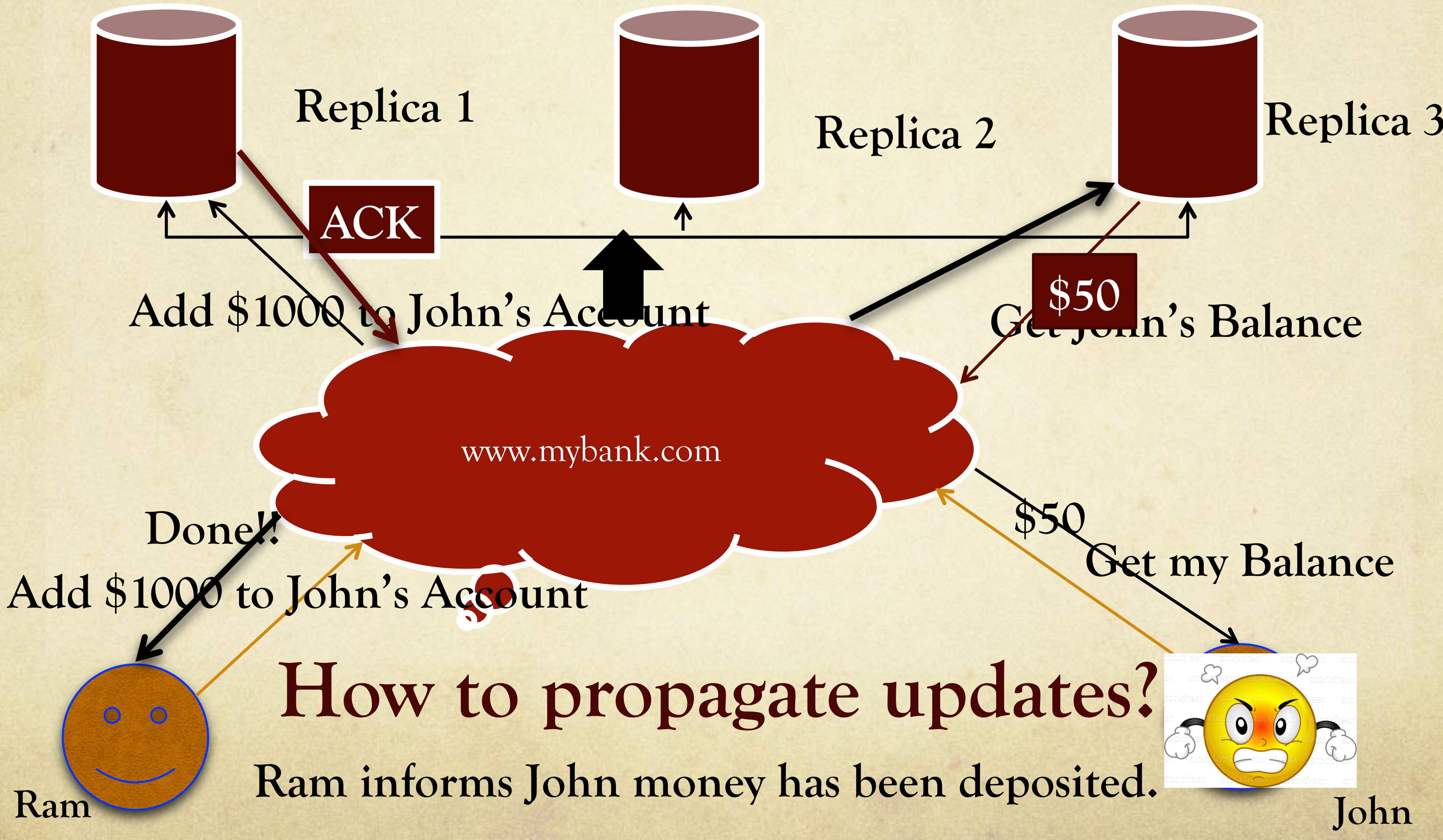
  Availability, Fault – tolerant, Faster reads

  Consistency

    Eventual Consistency : Epidemic Algorithms

      Faster reads, writes

Why Consistency ?



How to propagate updates?

Ram informs John money has been deposited.

How to Propagate Updates?

- Direct Mail
- Anti-entropy - Simple Epidemic

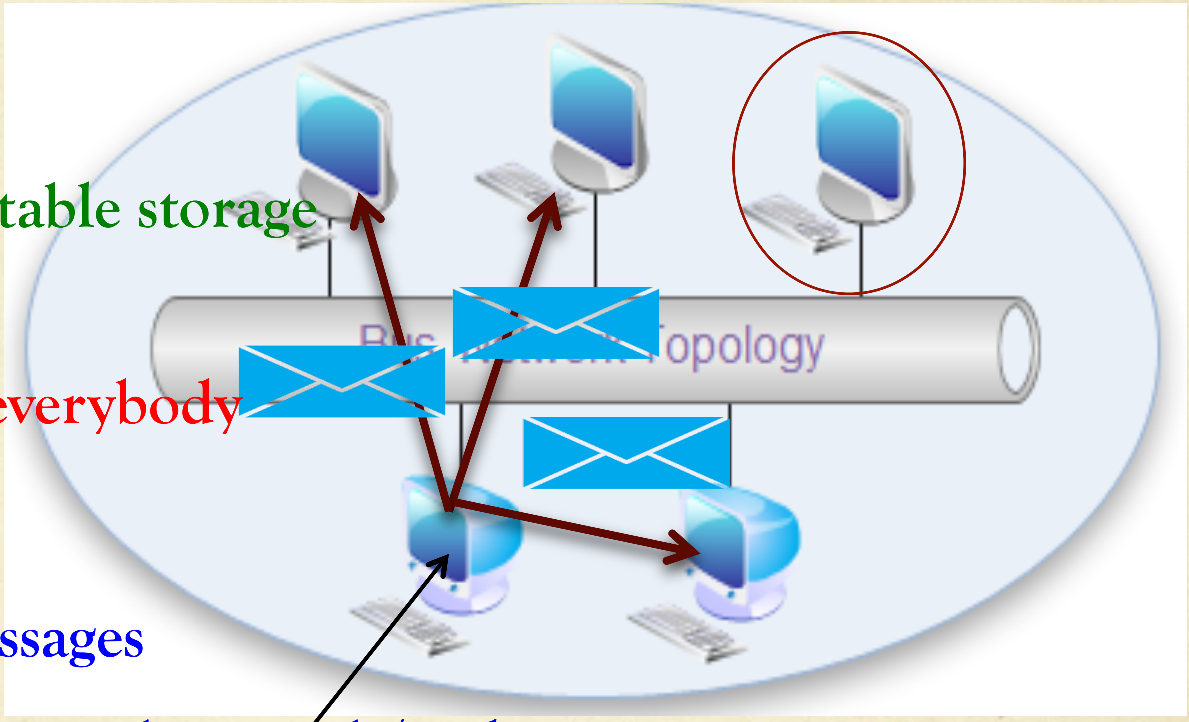
  infected or susceptible sites

- Rumor Mongering – Complex Epidemic

  infected, susceptible and removed

Goal: For all s, s’ € S: s.valueOf = s’.valueOf

Direct Mail



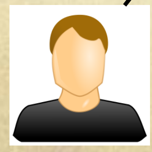
- Messages are queued, stable storage

  Reliable but…
- Not everybody knows everybody
- Queue overflows

Source site = O(n) messages

Traffic = # sites * Average distance b/w the sites

Anti-Entropy in Background

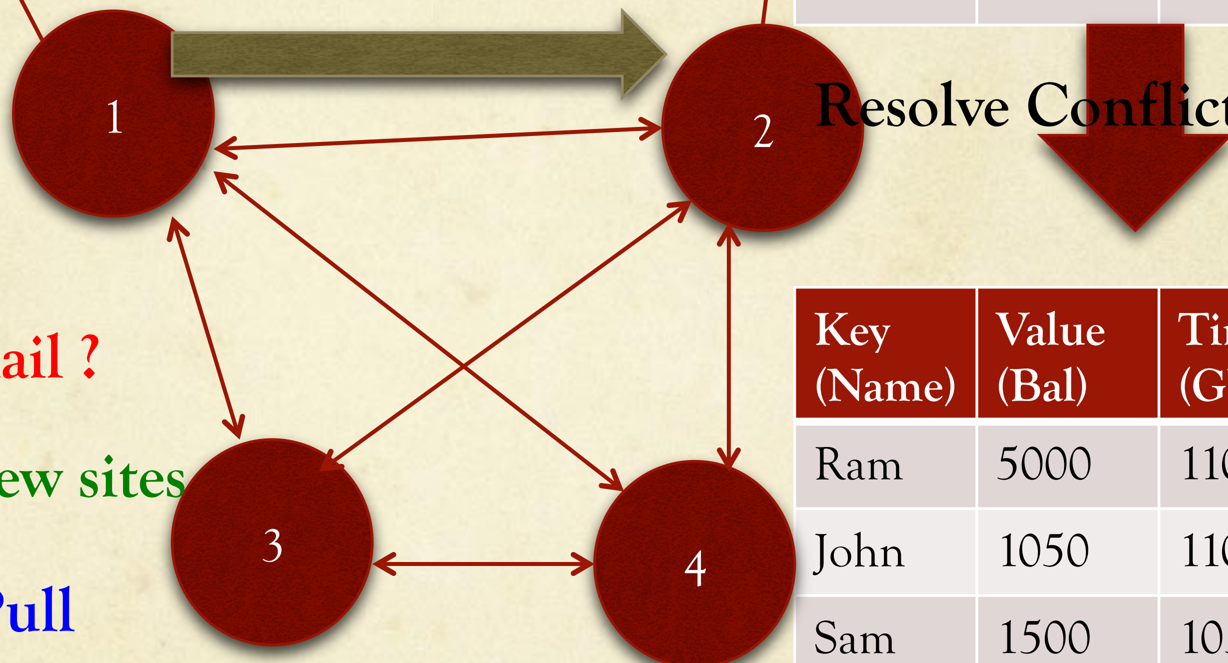


Update[v : V] = s.ValueOf <—(v, GMT)

Anti-Entropy

Key (Name)	Value (Bal)	Time (GMT)
Ram	5000	110
John	1050	110
Sam	1200	100

Key (Name)	Value (Bal)	Time (GMT)
Ram	6000	100
John	50	100
Sam	1500	105



Key (Name)	Value (Bal)	Time (GMT)
Ram	5000	110
John	1050	110
Sam	1500	105

Slower than Direct Mail ?

Distribute update to few sites

Push, Pull, Push - Pull

Push vs Pull

- **Pull**

  p_i= Probability of a site remaining susceptible after the i^{th} cycle

  Only if it Selects susceptible site in i+1st cycle

  $p_{i+1} = (p_i)^2 \approx 0$

- **Push**

  Only if No infectious site chose to contact susceptible site

  $p_{i+1} = p_i(1\text{-}1/n)^{n(1-p_i)} = p_i e^{-1} \approx 0$ (less rapidly)

  But Anti-Entropy is Expensive!!!

Anti-Entropy is Expensive

- Usually Databases are in "nearly" complete agreement

  Then why send entire Database across network ?

- Exchange Database only if checksum of Database disagree

  Time to update to all sites > Interval between the updates

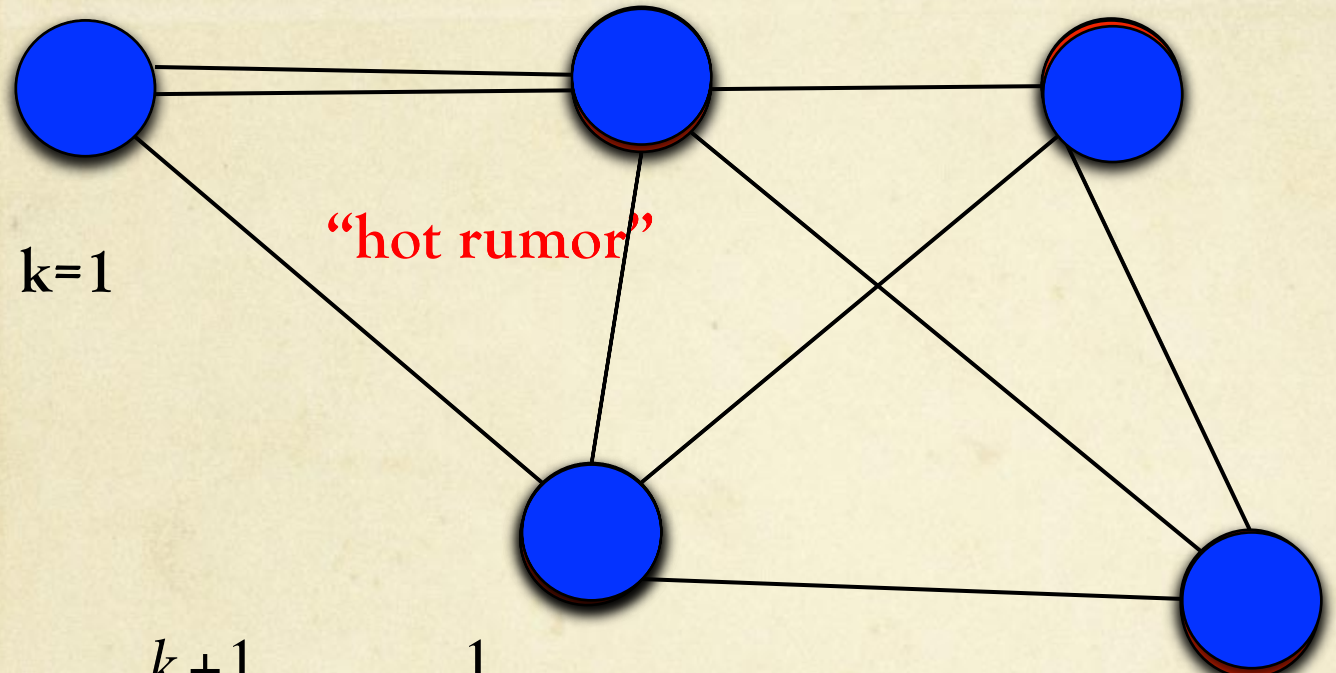
- Recent update list that contains all new changes for time window t'

  t' MUST > Time required to distribute the update

- Exchange recent update list, then compare checksums

  Checksums agree, Less traffic, Less Database comparison

Rumor Mongering



$$i(s) = \frac{k+1}{k}(1 - s) + \frac{1}{k}\log s$$

Convergence, i = 0

$$s = e^{-(k+1)(1-s)}$$

s = ? Residue

Counter vs Probability

- Become removed after k unnecessary contacts

Response and Counters

Counter k	Residue s	Traffic m	Converge 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

Blind and Probabilistic

Counter k	Residue s	Traffic m	Converge 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

Convergence

Residue

Push, RM

Traffic

Push vs Pull

- Numerous updates

Susceptible can find infective with high Probability

Counter k	Residue s	Traffic m	Convergence t_{ave} | t_{last}
1	$3.1 * 10^{-7}$	2.70	9.97 17.63
2	$5.8 * 10^{-4}$	4.49	10.07 15.39
3	$4.0 * 10^{-6}$	6.09	10.08 14.00

Pull, Response and Counters

Exchange counters If both know the update Increment the site with smaller counter

Push gets better than Pull with connection limit. How?

Two sites contact the same recipient One gets rejected Still gets the update

Two sites contact the same infected site One gets rejected Only 1 site updated

Direct Mail vs Rumor

- Both has no guarantee

  - Anti-entropy is the key

- But what if Anti- entropy finds conflicts ?
  - Let it be…
  - Clearinghouse uses Direct Mail for redistribution

Worst case : DM Manages to deliver an update half the sites

Later AE-DM generates $O(n^2)$ messages

or

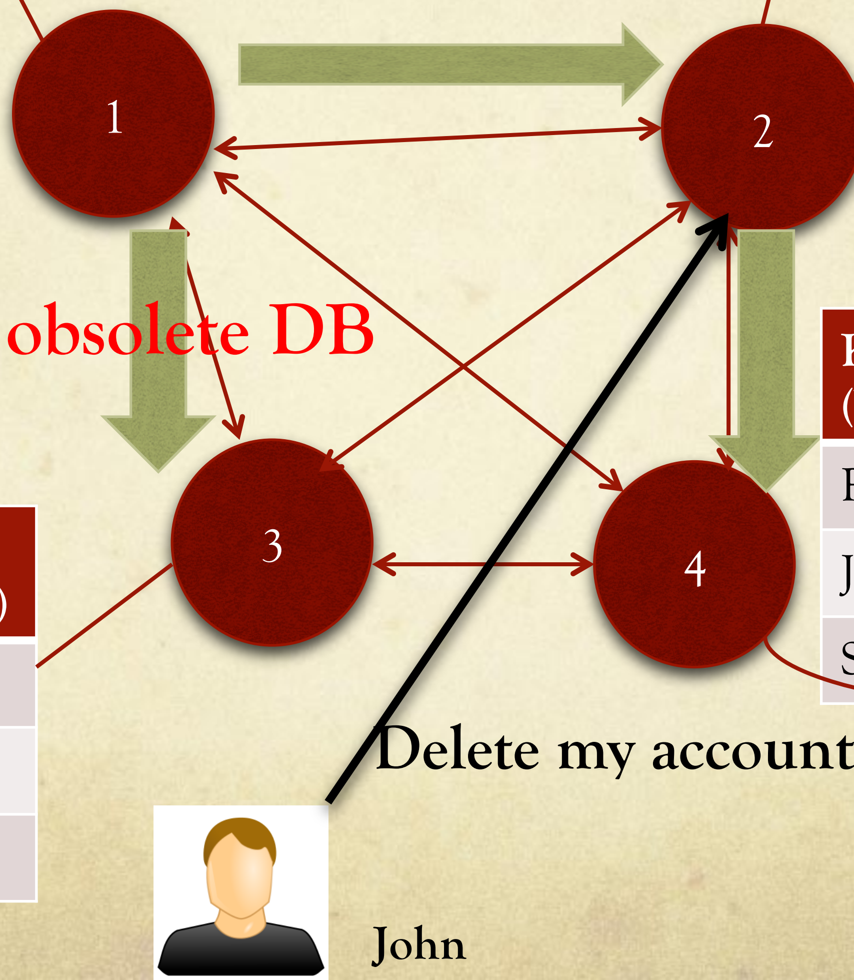
AE-RM generates $O(n)$ messages

How do we delete?

Key (Name)	Value (Bal)	Time (GMT)
Ram	5000	110
John	1050	110
Sam	1200	100

Key (Name)	Value (Bal)	Time (GMT
Ram	5000	110
John	1050	110
Sam	1200	100

Resurrection with obsolete DB



Key (Name)	Value (Bal)	Time (GMT)
Ram	5000	110
John	1050	110
Sam	1200	100

Key (Name)	Value (Bal)	Time (GMT)
Ram	5000	110
John	1050	110
Sam	1200	100

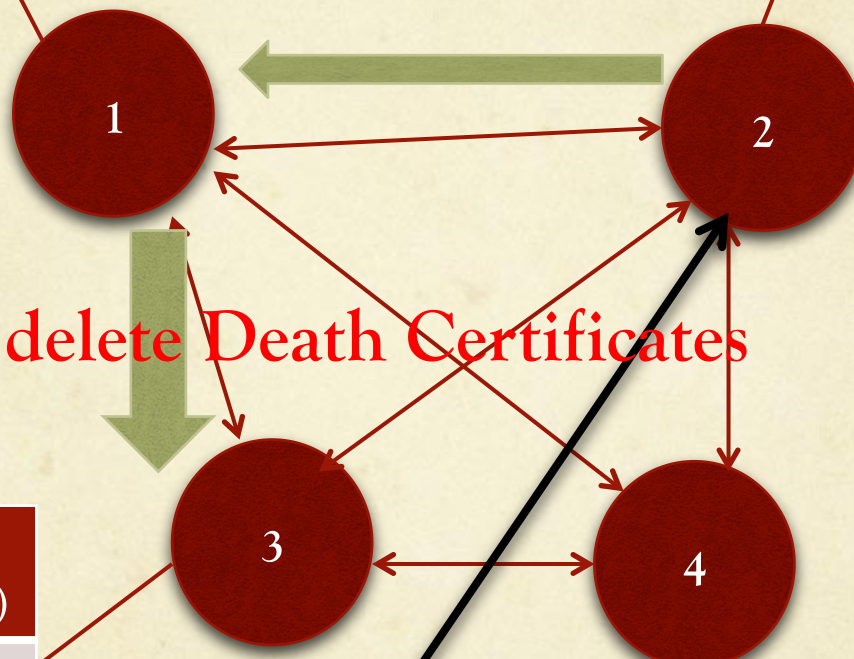
Death Certificates

Delete
John, 120

Key (Name)	Value (Bal)	Time (GMT)
Ram	5000	110
Sam	1200	100
Sam	1200	100

Key (Name)	Value (Bal)	Tim (GM
Ram	5000	110
John	1050	110
Sam	1200	100

Delete
John, 120

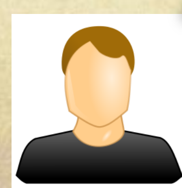


But when do we delete Death Certificates

Looks Easy !!!

Key (Name)	Value (Bal)	Time (GMT)
Ram	5000	110
Sam	1200	100

Delete my account



John

Delete Death Certificates

- Delete them after 30 days
  - Still has risk of resurrection
- Sarin and Lynch propose Chandy-Lamport snapshot algorithm
  - Records Application messages when sees duplicate Marker messages
  - Snapshot ensures Death certificate received at all sites
  - But what when site permanently fails?

Dormant Death Certificates

- Delete death certificates at most sites

  -But retain Dormant death certificates at some retention sites "r"

- Obsolete update encounters Dormant certificate
  Activate Death certificate

  -Original timestamp, Activation timestamp

- Much larger threshold for Dormant Death Certificates

- But lost due to permanent server failures

  $p_{fail} = 2^{-r}$

Spatial Distribution

- **Saves significant amount of traffic**
- **probability of connecting site at distance d= d^{-a}**

  when a = 2 ; convergence is polynomial in log n

  Generalized for CIN with distribution d^{-2D}

  dependent on dimension of mesh 'D'

- **$Q_s(d)$ = sites at distance d or less from s**

  arbitrary network Best Scaling

Using Spatial Distribution

- Anti-Entropy

  Uniform distribution ≈ 80 conversations on critical links

  Expected traffic per link per cycle < 6 conversations

  $Q_s(d)$ adapts well to "local dimension"

- Rumor Mongering

  Making it less sensitive for sudden increases in $Q_s(d)$

  Each site s builds a list of sites sorted by distance

  For a = 2 again complexity is $O(d^{-2d})$

Simulation Results: AE

Push-Pull, No Connection limit

Distribution	t_{last} |	t_{ave}	Compare Traffic Avg |	Critical
Uniform	7.81	5.27	5.87	75.74
a = 1.2	10.04	6.29	2.00	11.19
a = 1.4	10.31	6.39	1.93	8.77
a = 1.6	10.94	6.70	1.71	5.72
a = 1.8	11.97	7.21	1.52	3.74
a = 2.0	13.32	7.76	1.36	2.38

Push-Pull, Connection limit 1

Distribution	t_{last} |	t_{ave}	Compare Traffic Avg |	Critical
Uniform	11.00	6.97	3.71	47.54
a = 1.2	16.89	9.92	1.14	6.39
a = 1.4	17.34	10.15	1.08	4.68
a = 1.6	19.06	11.06	0.94	2.90
a = 1.8	21.46	12.37	0.82	1.68
a = 2.0	24.64	14.14	0.72	0.94

Increase in convergence < factor of 2

Decrease in Average traffic ≈ factor of 4 Frequent anti-entropy

Massive decrease in compare traffic for transatlantic links > factor of 30

Connection limit reduces the compare traffic/cycle but increases the number of cycles

Simulation Results: RM

Distribution	k	t_{last}	t_{ave}	Compare traffic Avg	Compare traffic Bushey	Update traffic Avg	Update traffic Bushey
Uniform	4	7.83	5.32	8.87	114.0	5.84	75.87
a = 1.2	5	10.14	6.33	3.20	18.0	2.60	17.25
a = 1.4	6	10.27	6.31	2.86	13.0	2.49	14.05
a = 1.6	7	11.24	6.90	2.94	9.80	2.27	10.54
a = 1.8	8	12.04	7.24	2.40	5.91	2.08	7.69
a = 2.0	9	13.00	7.74	1.00	3.44	1.90	5.94

Feedback, Counter, Push-Pull, No connection limit

With increase in "a", k increases gradually

Convergence time increases

Decrease in Average traffic ≈ factor of 3

Massive decrease in compare traffic for transatlantic links > factor of 30

Discussion

- Anti- Entropy is robust than Rumor Mongering

  Rumors may become inactive leaving sites susceptible

- Push & Pull much sensitive to spatial distribution than Push-Pull (RM)

  k = 36 for a = 1.2 (Push, Feedback, No connection limit, Counter)

- Anti - Entropy with distribution of d^{-2} was implemented at CIN

  Massive improvement in network load & consistency

Cons/Questions

- Storage, Death Certificates
- Irregular Network Topologies
  - Dynamic Hierarchy