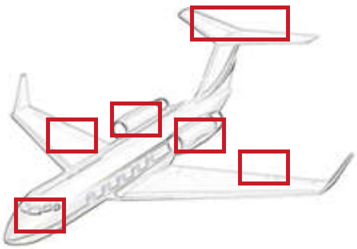
An Introduction to Gossip Protocols

François Taïani

Traditional Distribution

- Typical distributed protocols:
  - ➔ do not rely on 'chance'
  - ➔ provide **deterministic** guarantees



- But
  - ➔ either rely on a **central** core of servers
  - ➔ or **not very scalable** ($O(n^k)$, k= 2,3)

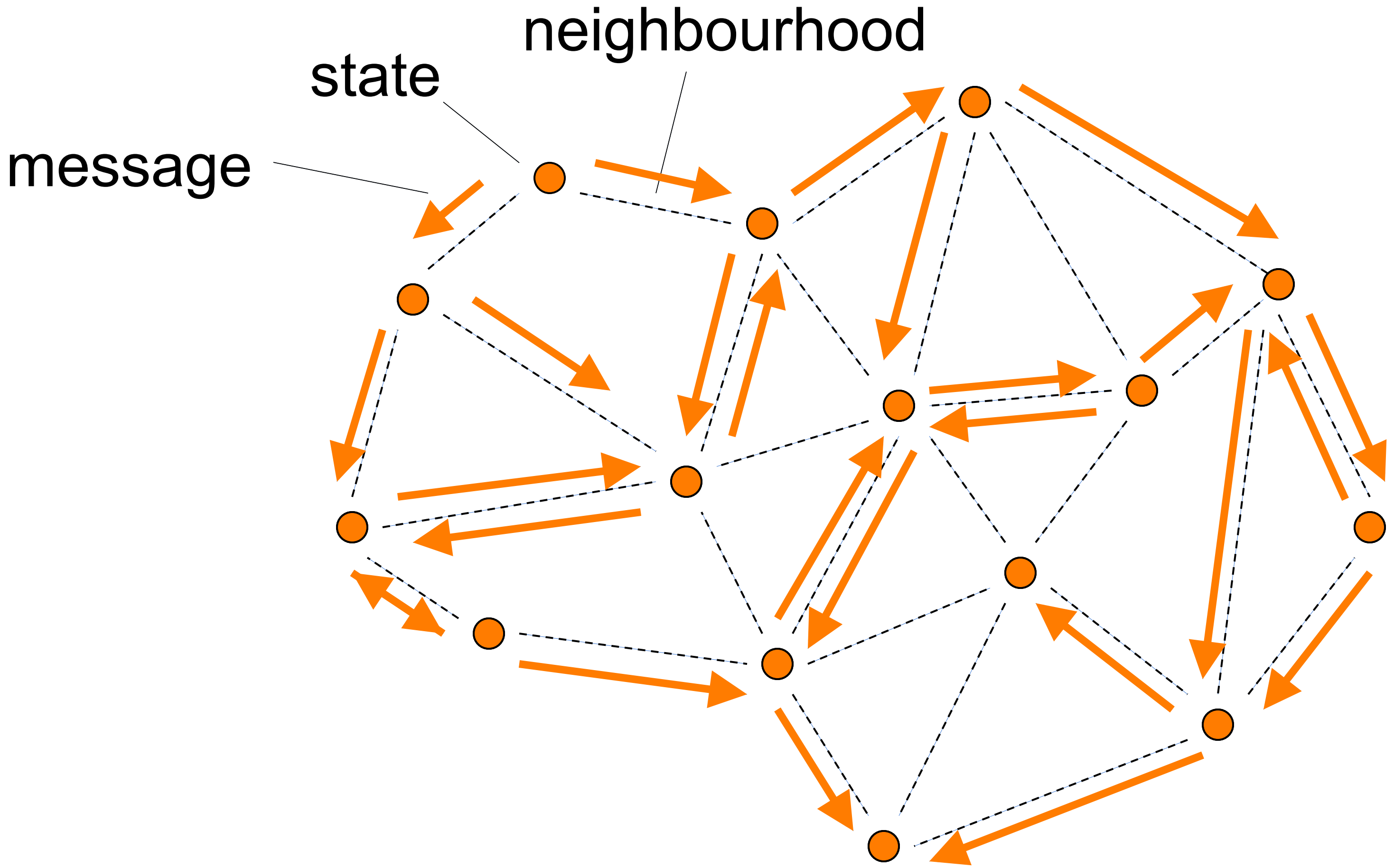
Gossip Protocols

- Alternative: **probabilistic** approach
- Gossip (aka epidemic) Protocols
  - ➔ Introduce some '**chaos**'
  - ➔ Goal: system to **converge** to a desirable outcome
  - ➔ But some nodes might be left out
- Trading determinism for **scalability** & **robustness**

Example: Overlay Multicast

- **Deterministic** options
  - Flooding → highly inefficient
  - Spanning tree → very efficient, but complex, costly, brittle
- **Gossip** approach: for each new broadcast, each node
  - on first reception randomly pick k targets in the rest of the system (**k** = ‘fanout’), and forward msg to them
  - (discard duplicate reception)

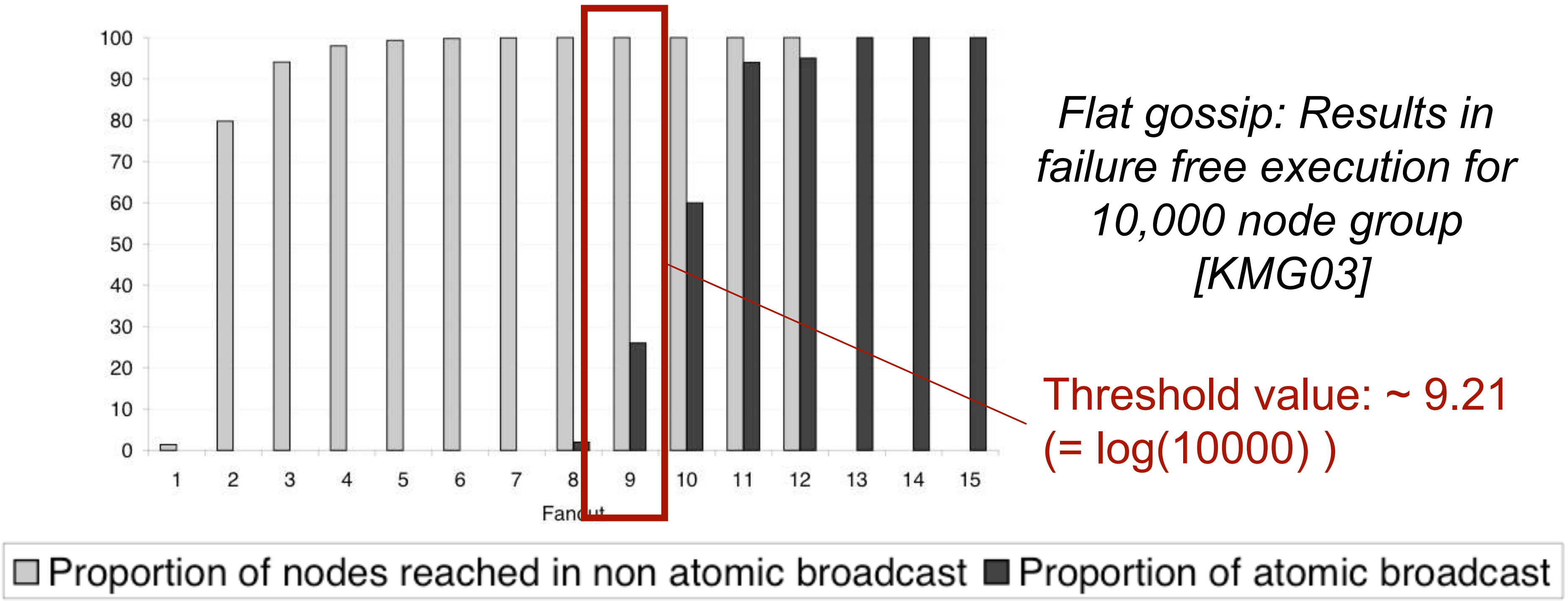
Example 1: Multicast



- How to choose **k** (fanout) to reach everyone?

Which k? Bimodal behaviour

- (Kemarrec, Massoulie, Ganesh / Microsoft Research)



Flat gossip: Results in failure free execution for 10,000 node group [KMG03]

Threshold value: ~ 9.21 (= log(10000))

Gossip: Key Ingredients

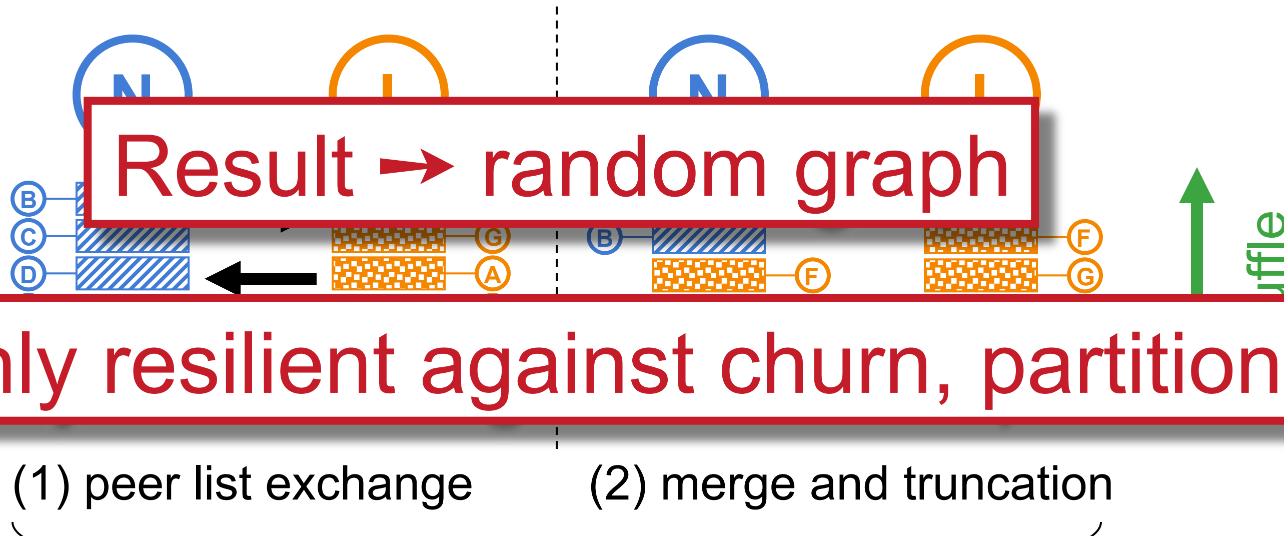
State → Stochastic Data exchange → New State

- What **state** to maintain?
  - → msg already seen (multi-cast)
  - → sensor value (aggregation, averaging)
  - → neighbours list (topology construction, membership)
  - → user profile (social networks)
- **When** to gossip? (periodic vs. event based)
- With **which probability** and **with whom** to gossip?
- Which information to **exchange**?
- How to **compute** new state?

Ex. 2: Peer Sampling

- "Random Peer Sampling Service" [JGK04]
  - ➔ Membership service: who is in the system?
- **State**: list of k neighbours (k = system param)
- Main idea: **Periodically**: each node **n**
  - ➔ picks one neighbour **i**
  - ➔ sends own list of neighbours, receives that of **i**
  - ➔ **i** and **n merge**, **shuffle**, **truncate** the 2 lists

Random Peer Sampling (cont.)



Result → random graph

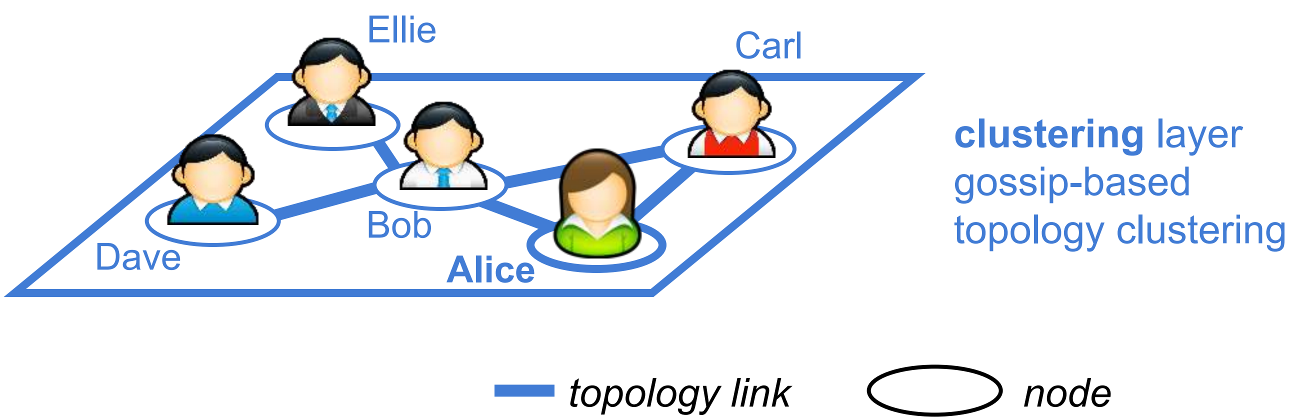
Highly resilient against churn, partition

Small diameter (good for multicast)

Ex. 3: Topology Construction

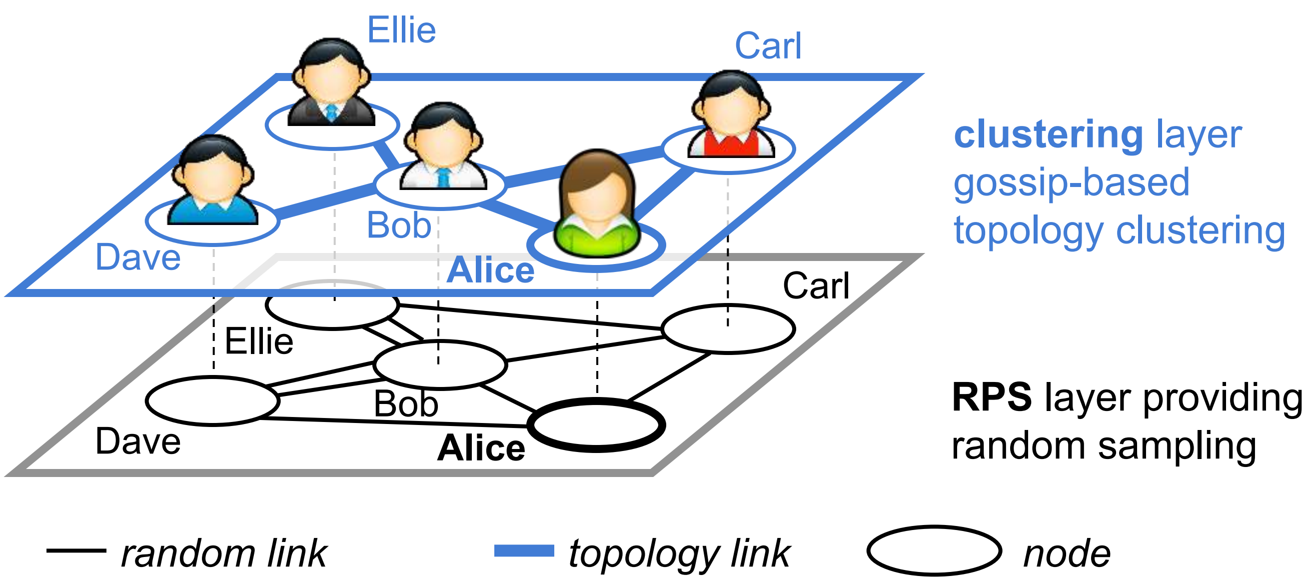
E.g. T-Man [JMB09]

- State: each node has a position (e.g. in R^3)
- Metrics: Euclidian distance between node
- Goal: find *n* closest neighbours



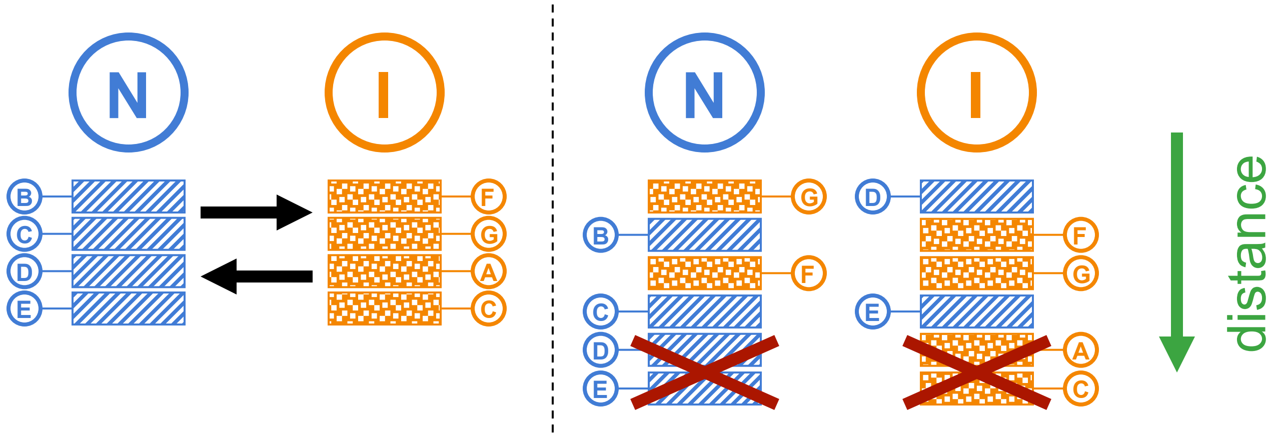
T-Man: Architecture

- Uses RPS service
- Periodically: each node **n**
  - ➔ merges RPS neighbours with clustering neighbours
  - ➔ **sorts** nodes in list according to distance
  - ➔ keeps k closest neighbours



T-Man: Swap Mechanism

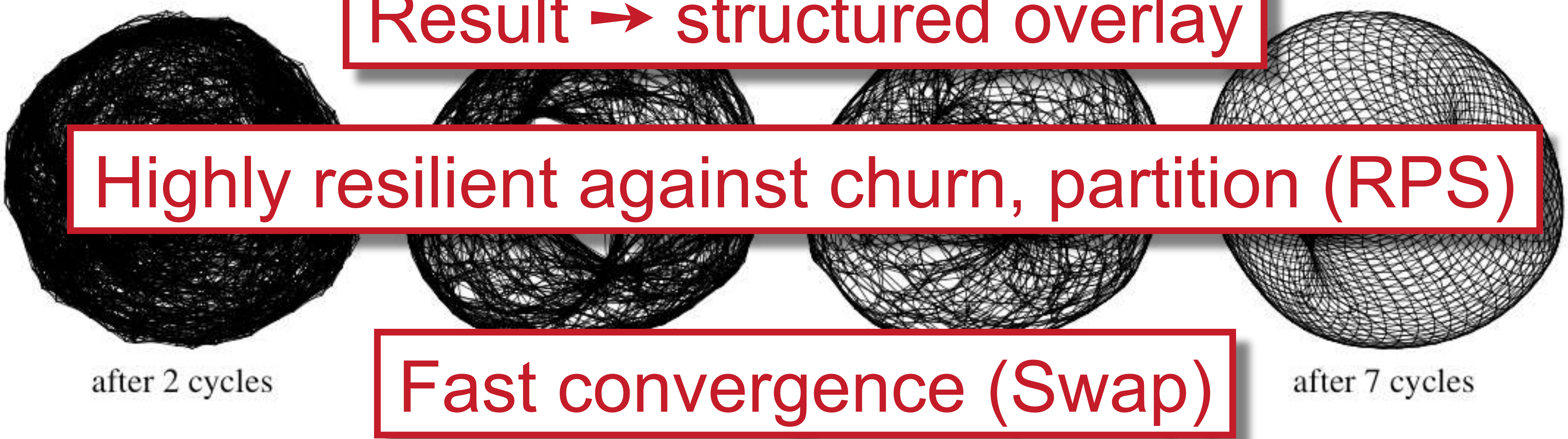
- To speed convergence each node **n**
  - ➔ picks one node **i** in clustering neighbours
  - ➔ sends own list of neighbours, receives that of **i**
  - ➔ **i** and **n** merge, **sort**, keep k closest neighbours



(1) peer list exchange (2) merge and truncation

T-Man in Action

- (taken from [JMB09])



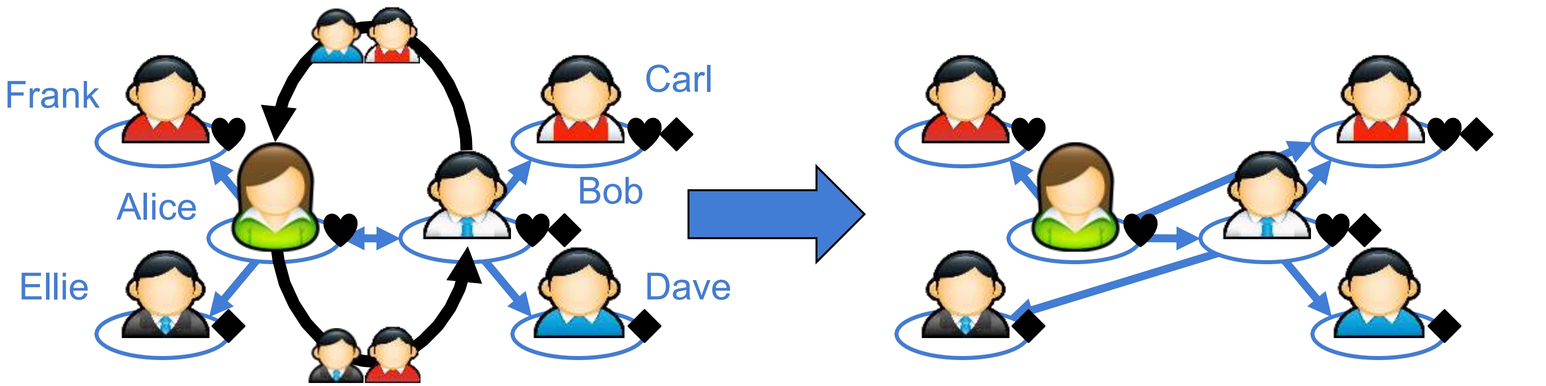
Result → structured overlay

Highly resilient against churn, partition (RPS)

Fast convergence (Swap)

Not limited to Euclidian space

- E.g. with user profiles (here metric = overlap)



❶ exchange of neighbors lists

❷ neighborhood optimization

Other Uses of Gossip

- Self-organisation in WSN: Who should do what?
- Topology construction in fixed overlay: fastest routing
- Knowledge aggregation: e.g. average temperature
- Information dissemination (failure detection, AWS S3)

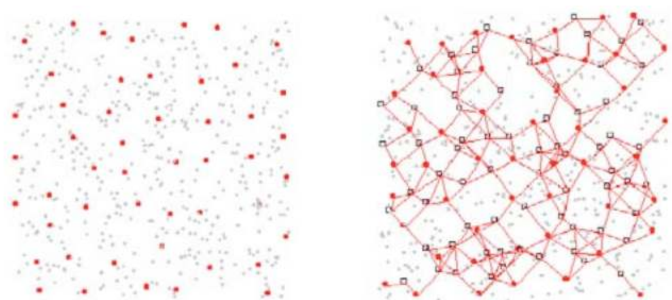
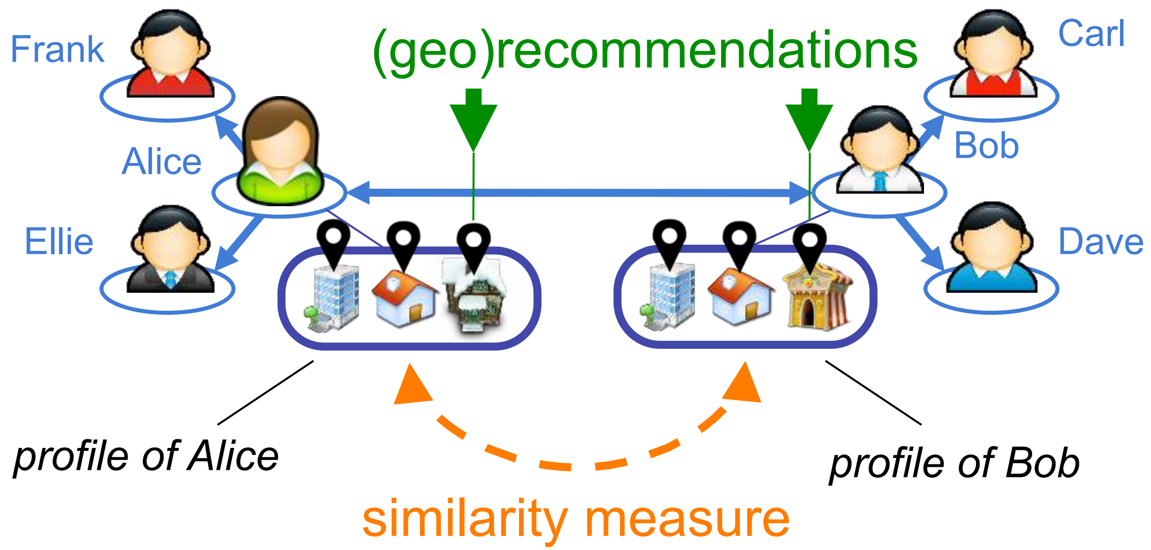


Figure 2: A 300×300 m area network with an average of 20 neighbors per node representing: (a) the router selection, (b) the energy state distribution.

From [LGKVB06]

Application to Social Networks

- ASAP group, Rennes, Gossple ERC
  - ➔ E.g. (geo)recommendations



Summing Up

- Gossip protocols
  - → Conceptually **simple**
  - → Potentially very **rich**
- Extremely well adapted to **large-scale** distribution
- **Efficient**, **scalable**, **robust**
- No deterministic guarantees (usually)
- Opportunity for **Programmatic / Software Eng.** angle

(Some) References: Protocols

- [BHO99] Birman, K. P., Hayden, M., Ozkasap, O., Xiao, Z., Budiu, M., and Minsky, Y. (1999). **Bimodal multicast.** ACM Trans. Comput. Syst., 17:41–88.
- [GRB01] Gupta, I., Renesse, R. v., and Birman, K. P. (2001). **Scalable fault-tolerant aggrega- tion in large process groups.** DSN '01, pages 433– 442
- [KMG03] Kermarrec A.-M., Massoulie L., Ganesh, A.J., **Reliable Probabilistic Communication in Large-Scale Information Dissemination Systems**, *IEEE Transactions on Parallel and Distributed Systems*, March 2003, (14:3)
- [JGK04] Jelasity, M., Guerraoui, R., Kermarrec, A.-M., and van Steen, M. (2004). **The peer sampling service: experimental evaluation of unstructured gossip-based implementations**. Middleware '04, pages 79–98, New York, NY, USA. Springer- Verlag New York, Inc.

(Some) References: Protocols

- [HHL06] Haas, Z. J., Halpern, J. Y., and Li, L. (2006). **Gossip-based ad hoc routing.** IEEE/ACM Trans. Netw., 14:479–491.
- [JK06] Jelasity, M. and Kermarrec, A.-M. (2006). **Ordered slicing of very large-scale overlay networks.** In Proceedings of the Sixth IEEE International Conference on Peer-to- Peer Computing, pages 117–124, Cambridge, United Kingdom. IEEE Computer Society.
- [JMB09] Mark Jelasity, Alberto Montresor, and Ozalp Babaoglu. 2009. **T-Man: Gossip-based fast overlay topology construction**. *Comput. Netw.* 53, 13 (August 2009), 2321-2339.
- [LGKVB06] Erwan Le Merrer, Vincent Gramoli, Anne-Marie Kermarrec, Aline C. Viana, and Marin Bertier. 2006. **Energy aware self-organizing density management in wireless sensor networks**. *1st Int. workshop on Decentralized resource sharing in mobile computing and networking* (MobiShare '06). ACM

(Some) Refs: Frameworks

- [LTBBK11] Lin S., Taiani F., Bertier M., Blair G. S., Kermarrec A.-M. (2011). **Transparent componentisation: high-level (re)configurable programming for evolving distributed systems**. ACM SAC ’11, pp. 203–208
- [LTB08] Lin, S., Taiani, F., and Blair, G. S. (2008). **Facilitating gossip programming with the gossipkit framework**, DAIS’08, pages 238–252
- [KS07] Kermarrec, A.-M. and van Steen, M. (2007). **Gossiping in distributed systems.** SIGOPS Oper. Syst. Rev., 41:2–7
- [EFL07] Eugster, P., Felber, P., and Le Fessant, F. (2007). **The ”art” of programming gossip-based systems.** SIGOPS Oper. Syst. Rev., 41:37–42.