SpeckNet

Research Consortium in Speckled Computing

Location discovery in SpeckNets

Ryan McNally

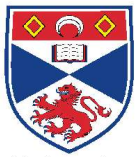
University of Edinburgh

s9902505@sms.ed.ac.uk

UNIVERSITY of GLASGOW

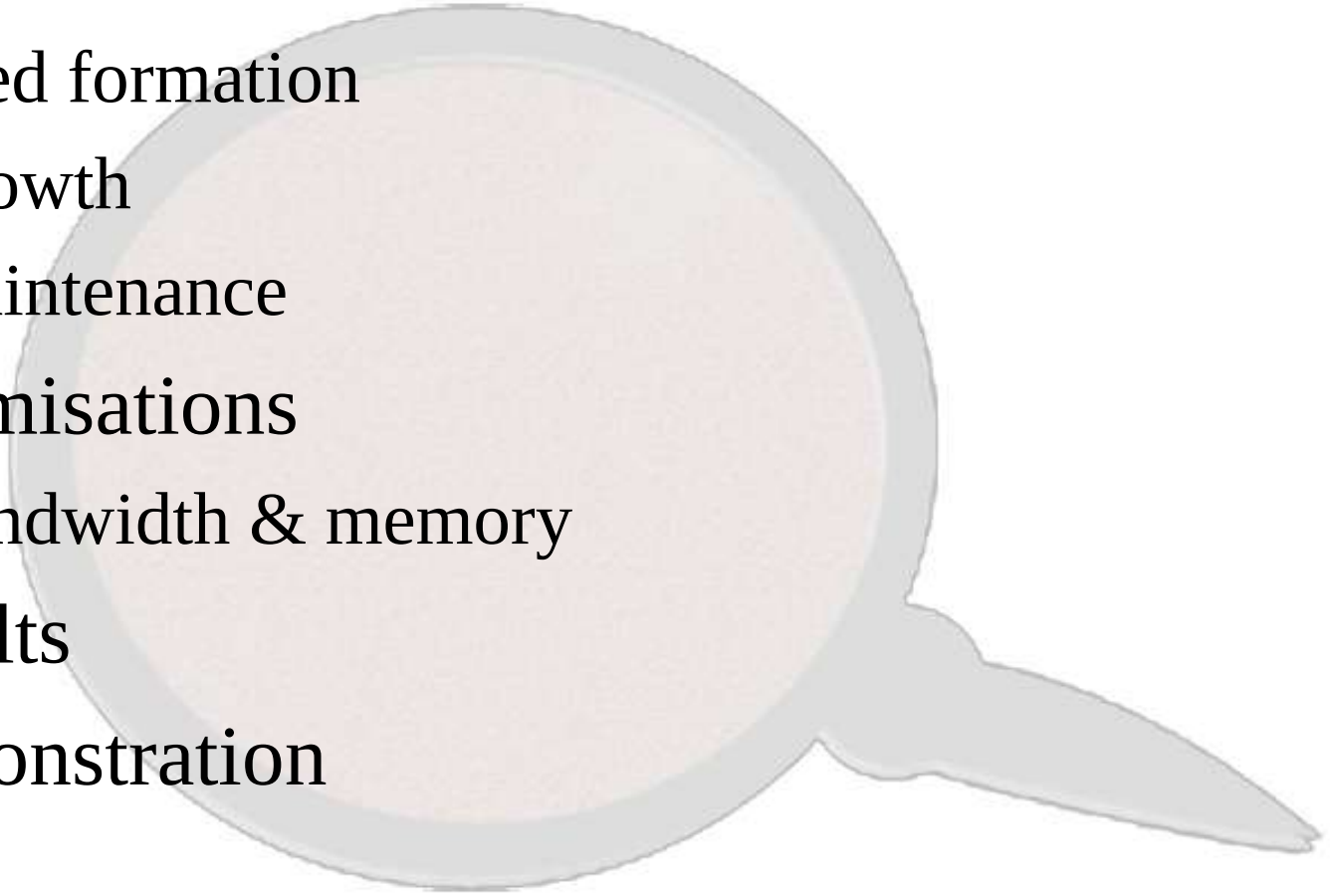
NAPIER UNIVERSITY EDINBURGH

UNIVERSITY OF STRATHCLYDE



University of St Andrews

- Algorithm outline
  - Seed formation
  - Growth
  - Maintenance
- Optimisations
  - Bandwidth & memory
- Results
- Demonstration

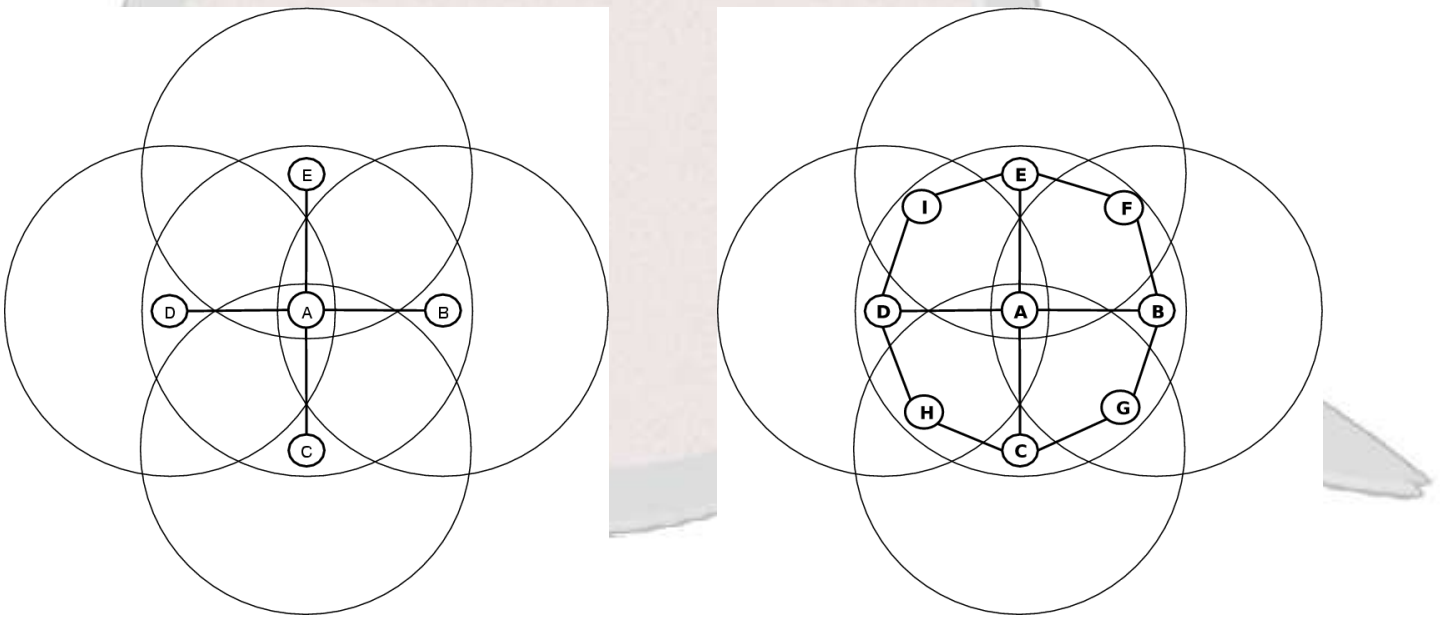


Crystal Growth

- A decentralised algorithm for computing the position of specks with respect to the network
  - Will differ by some network-wide affine transformation to actual location
- Each speck uses only local information
- Information being broadcast is not solely useful for the purposes of location discovery
  - Message routing, etc

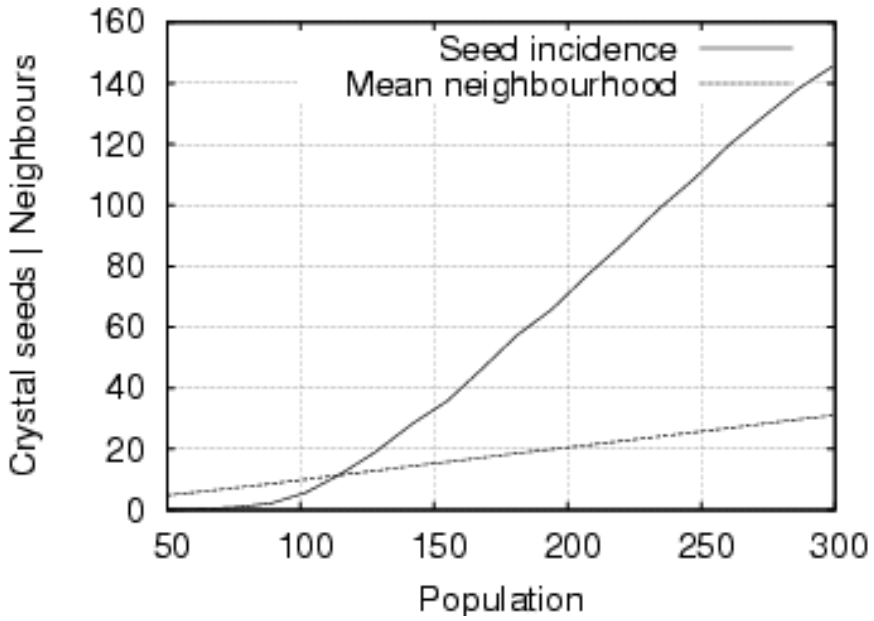
Seed formation

- Specks start life with no knowledge of their location
- Need to find a way to calculate likely positions from neighbourhood information



Seed formation

- Need only share neighbourhood data with neighbours
- 2D case commonly occurring
- 3D case rare

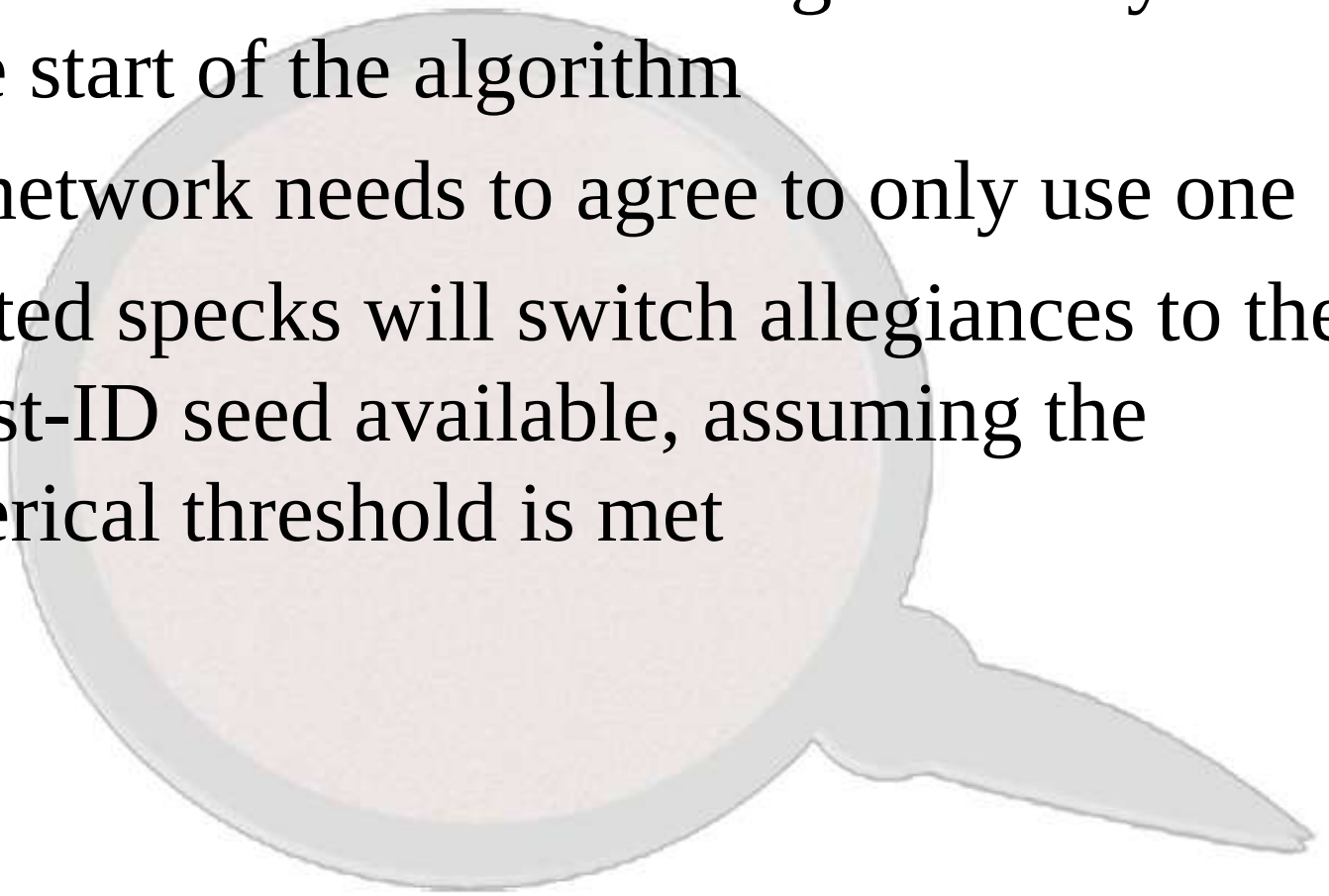


Growth

- A specks S monitors its neighbourhood for located specks
- When there is some threshold number of specks that are located using the same seed, S will also commit itself to a location
  - A simple mean of the other specks' locations
- Priority is given to the seed with the lowest ID number

Battling seeds

- It is common for there to be great many seeds at the start of the algorithm
- The network needs to agree to only use one
- Located specks will switch allegiances to the lowest-ID seed available, assuming the numerical threshold is met

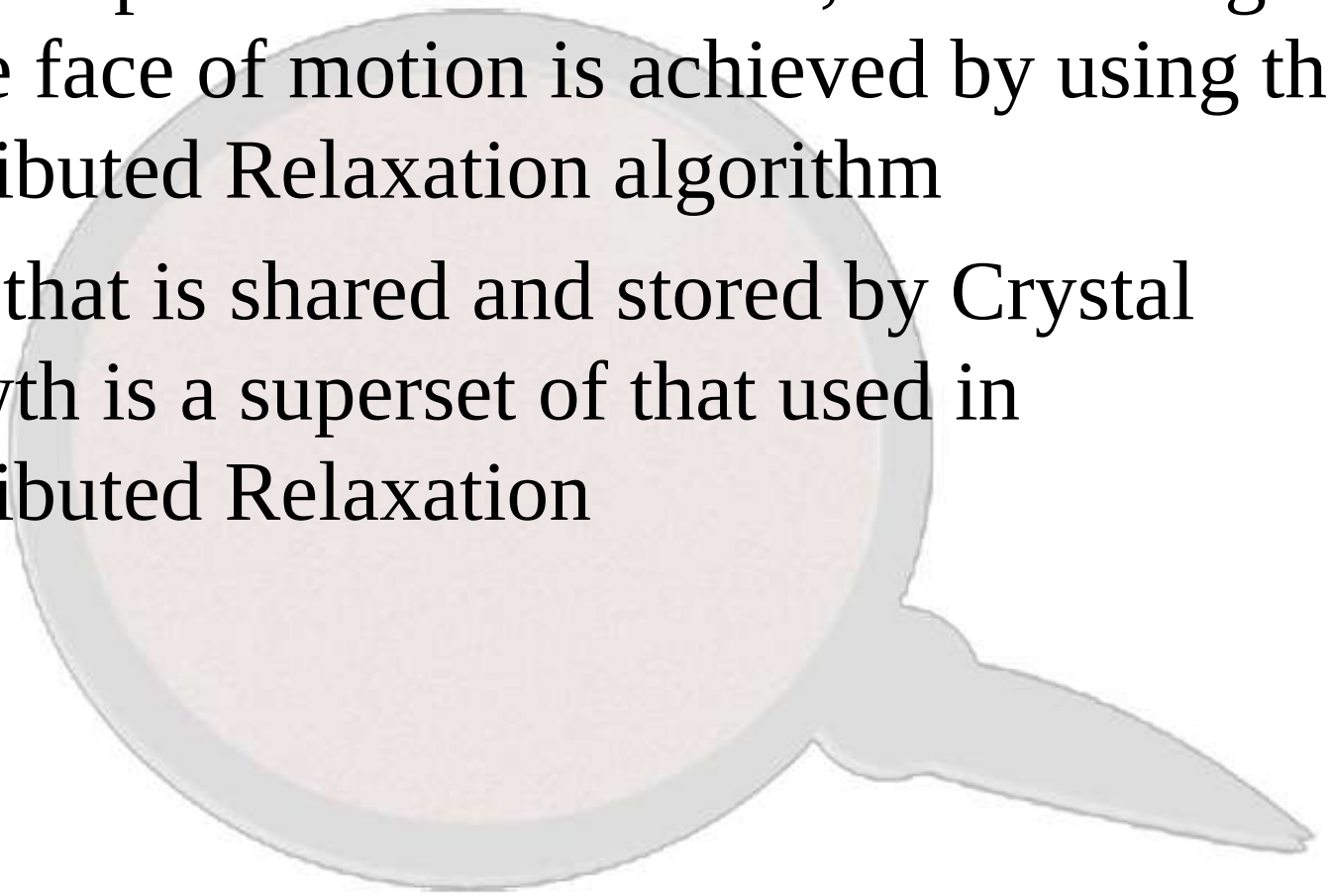


Guaranteeing growth

- The growth of a system is dependent on the threshold number of located neighbours being compatible with the local density of the network
  - Threshold too high = no growth
  - Threshold too low = accuracy suffers
- Solution is to start with a high threshold, and gradually reduce it when no growth is detected

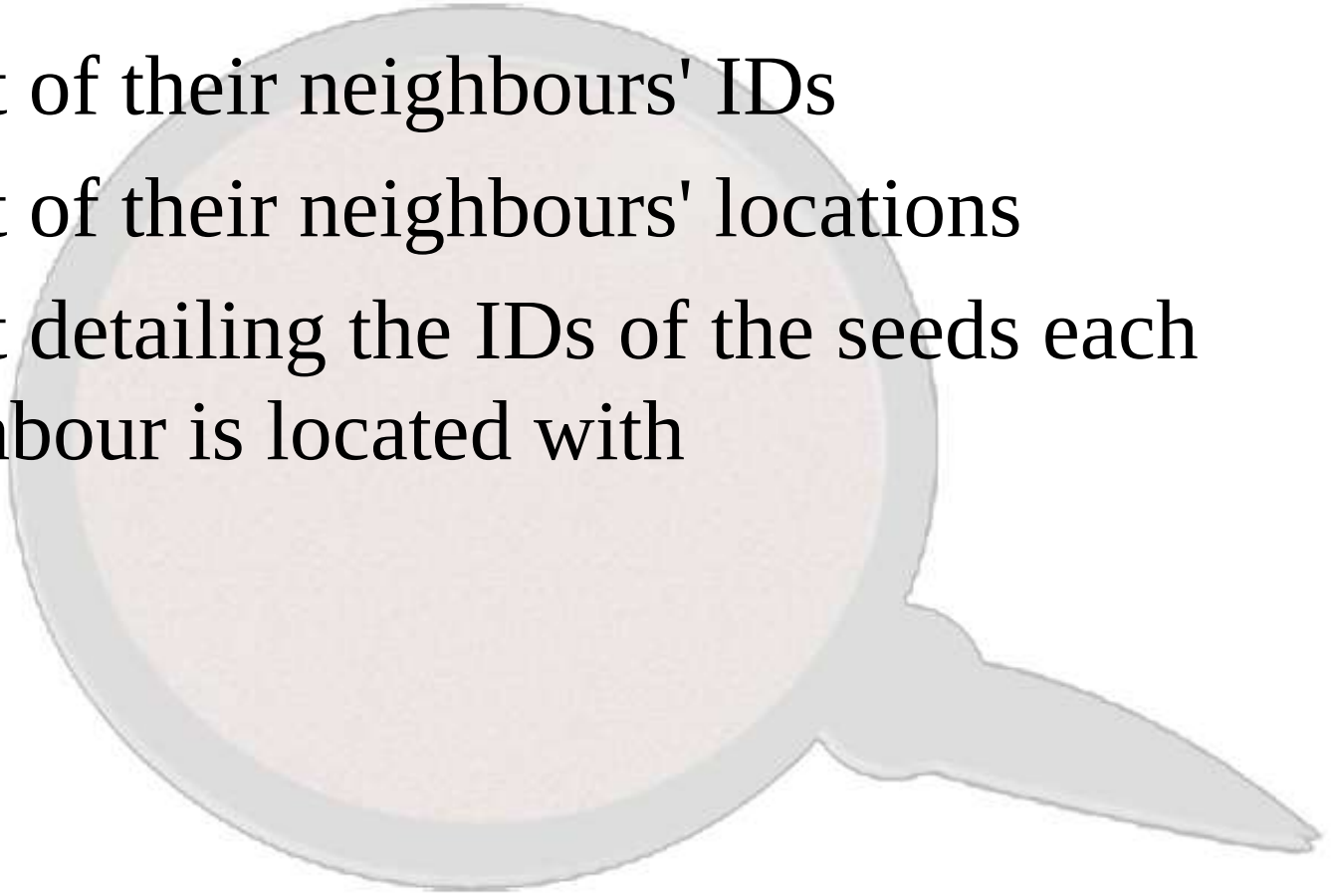
Location maintenance

- Once a speck has it's location, maintaining it in the face of motion is achieved by using the Distributed Relaxation algorithm
- Data that is shared and stored by Crystal Growth is a superset of that used in Distributed Relaxation



Implementation - Messages

- Their ID
- A list of their neighbours' IDs
- A list of their neighbours' locations
- A list detailing the IDs of the seeds each neighbour is located with

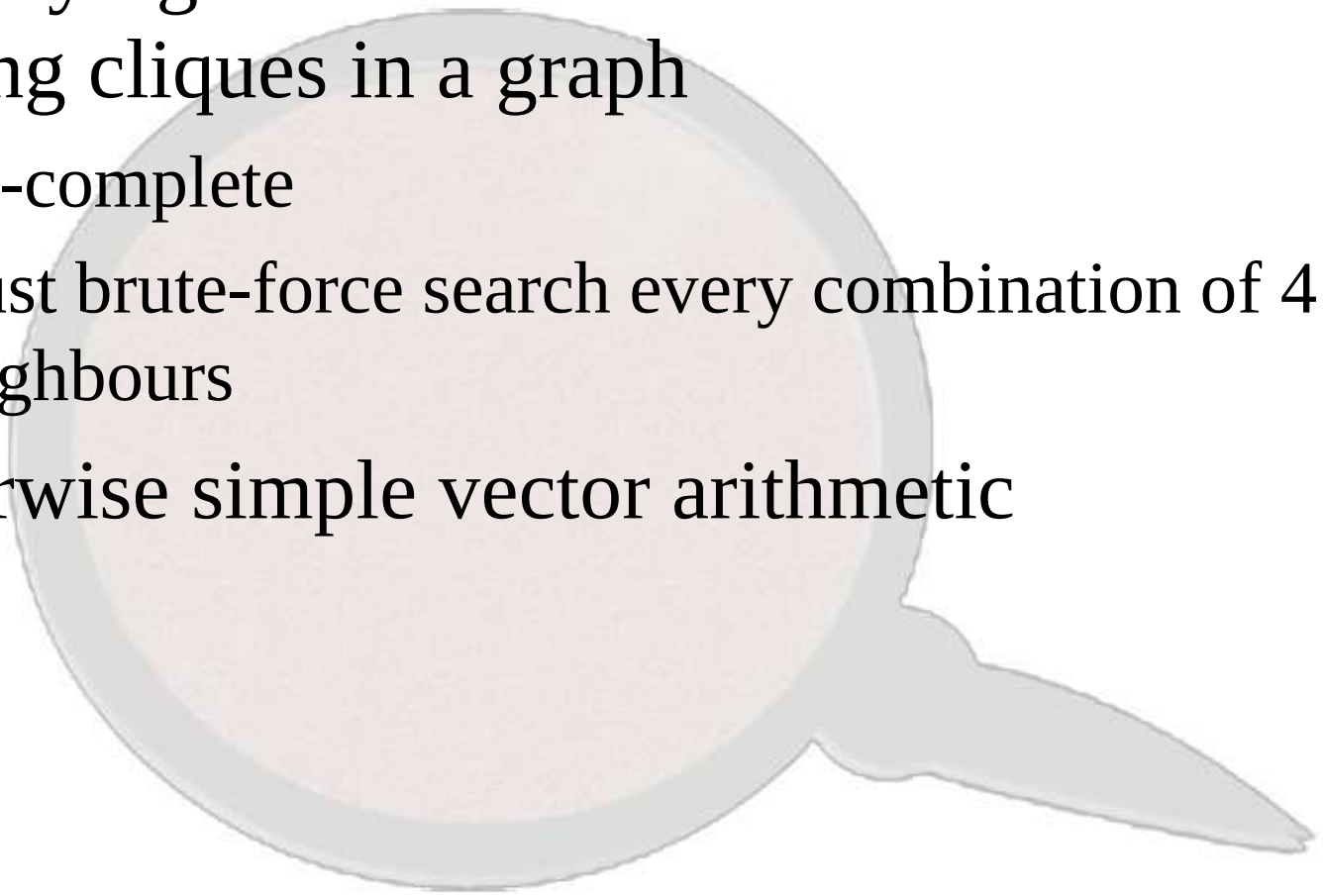


Implementation – Storage

- Until a seed is found, each speck must maintain a list of their neighbours, and their neighbours' neighbourhoods
- After a seed is found, each speck must maintain
  - A list of their neighbours' IDs
  - A list of their neighbours' locations
  - A list detailing the ID of the seed each neighbour is located with
    - Can be discarded after the network agrees one a seed

Implementation – Computation

- Identifying a seed formation is akin to finding cliques in a graph
  - NP-complete
  - Must brute-force search every combination of 4 neighbours
- Otherwise simple vector arithmetic

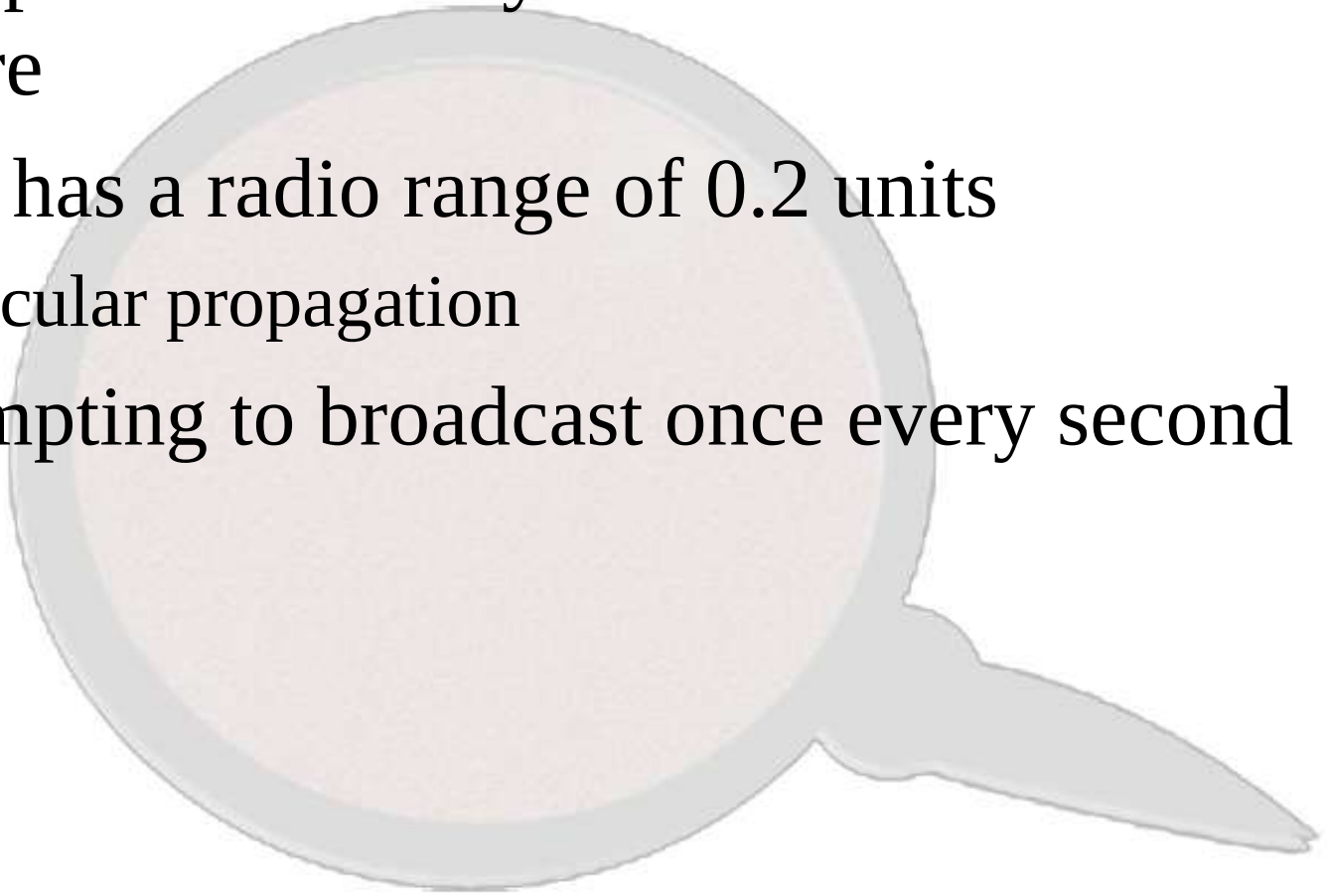


Optimisations

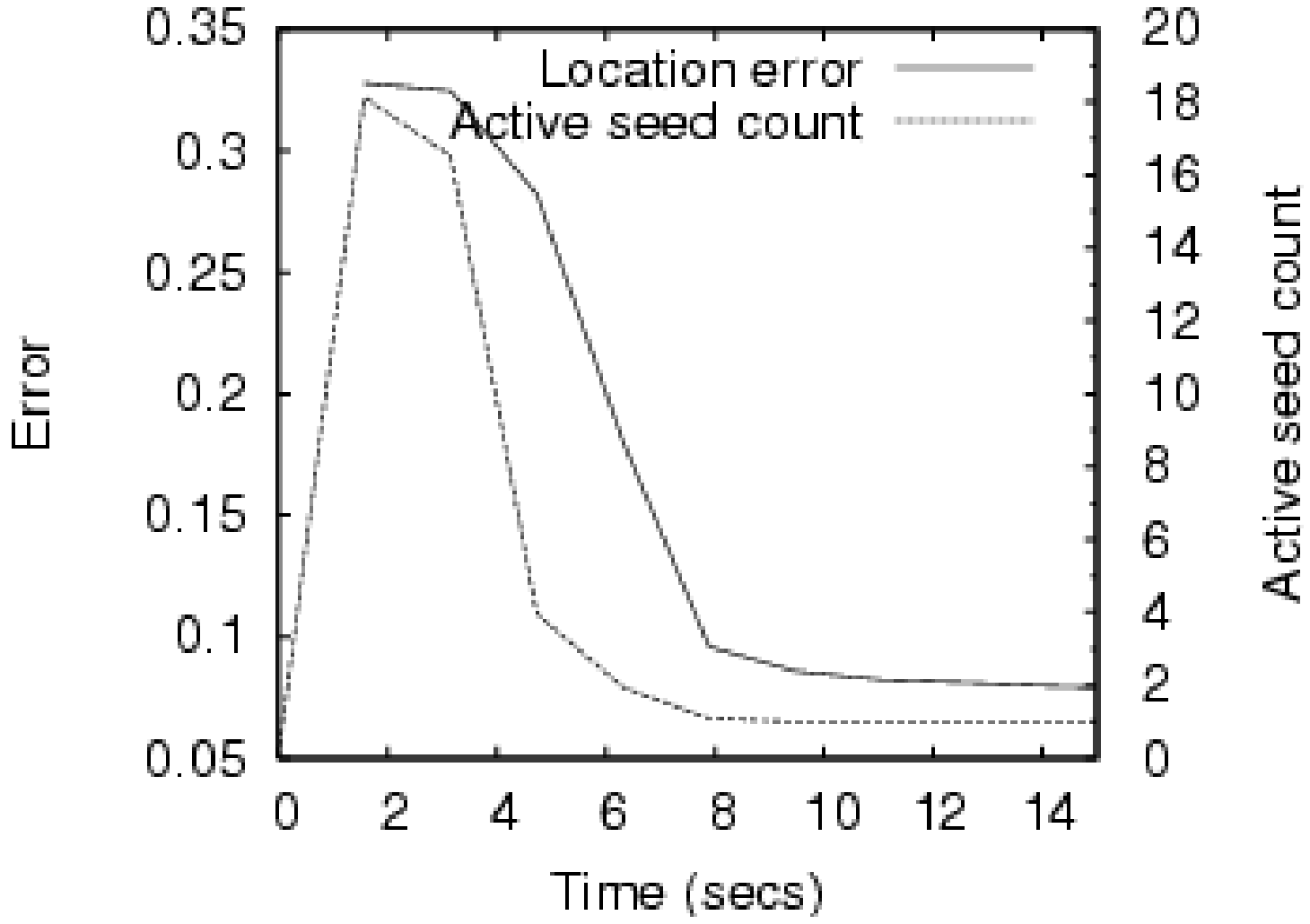
- Can reduce memory and bandwidth usage by:
  - Limiting the precision of locations
  - Limiting the number of neighbours that we broadcast full details for
- The theoretical accuracy limit is determined by the local density
  - for anything but the highest densities, around 8 bits per component is more than sufficient

Simulation setup

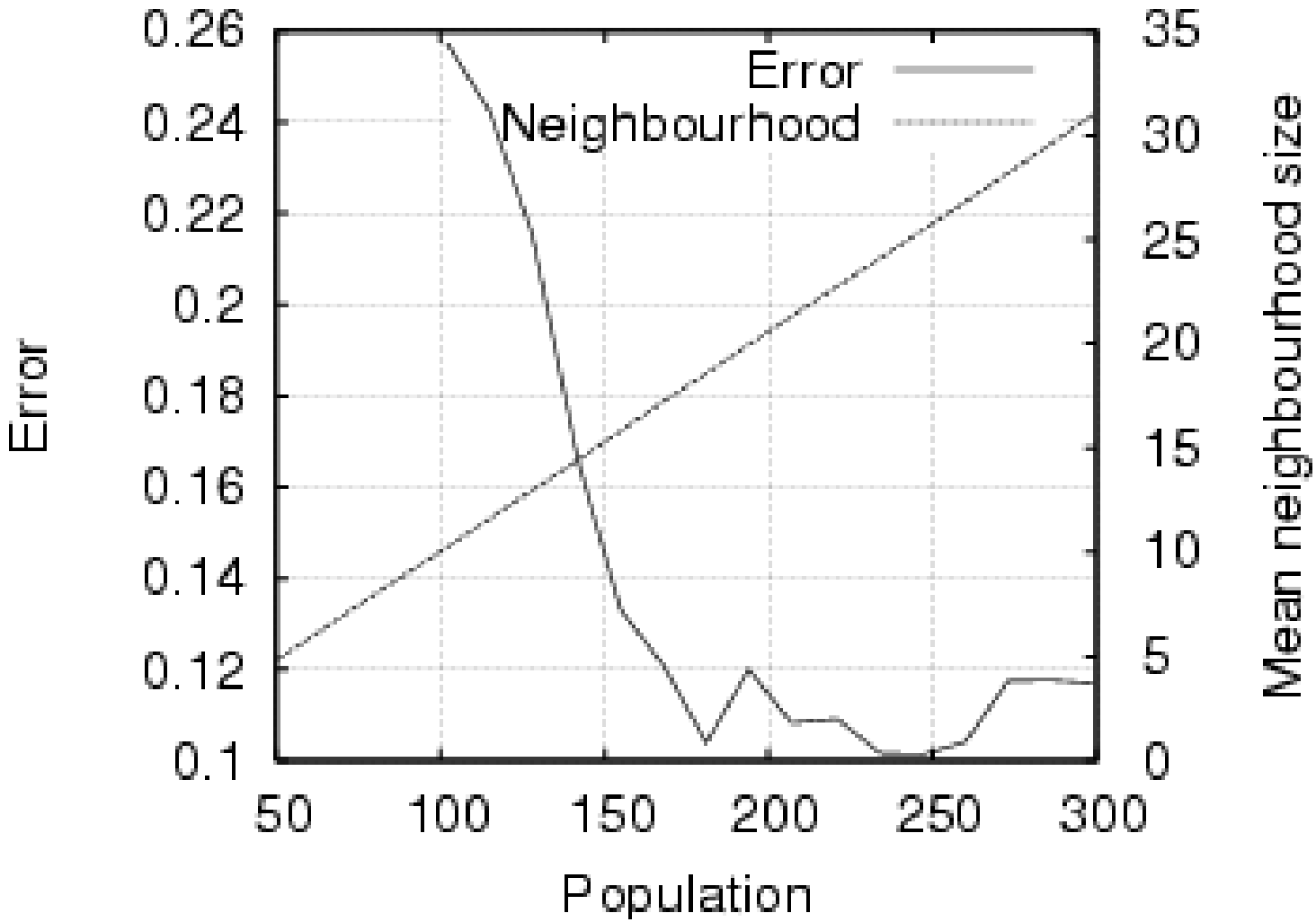
- 200 specks randomly distributed on a unit square
- Each has a radio range of 0.2 units
  - Circular propagation
- Attempting to broadcast once every second



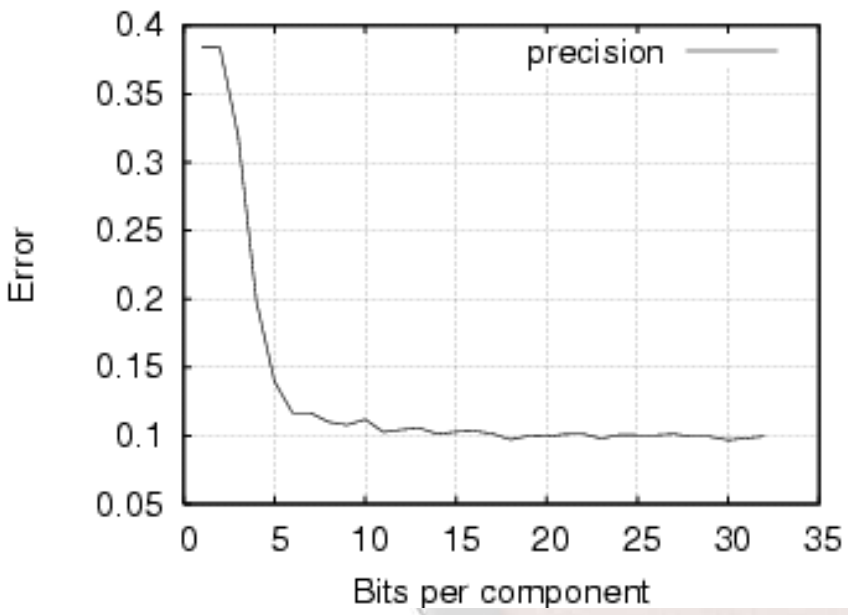
Results – Time

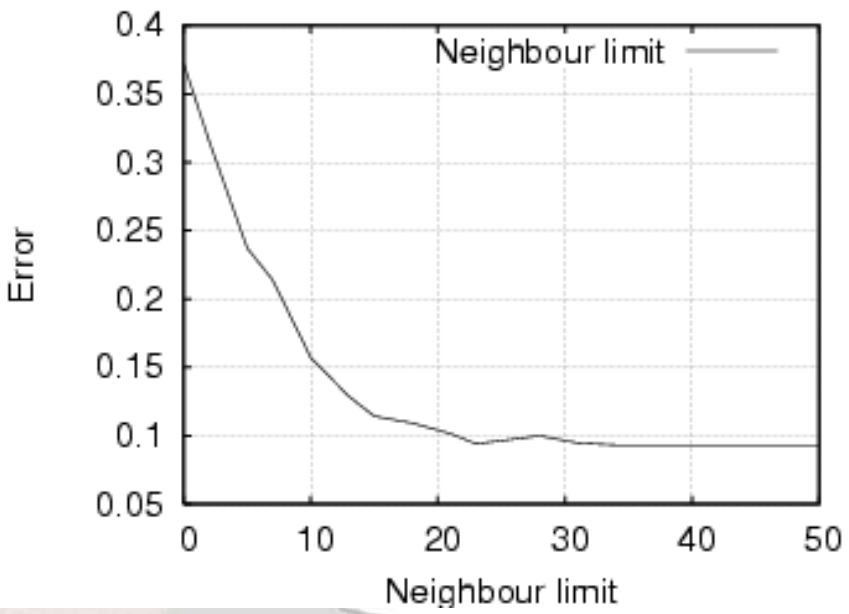


Results – Speck population

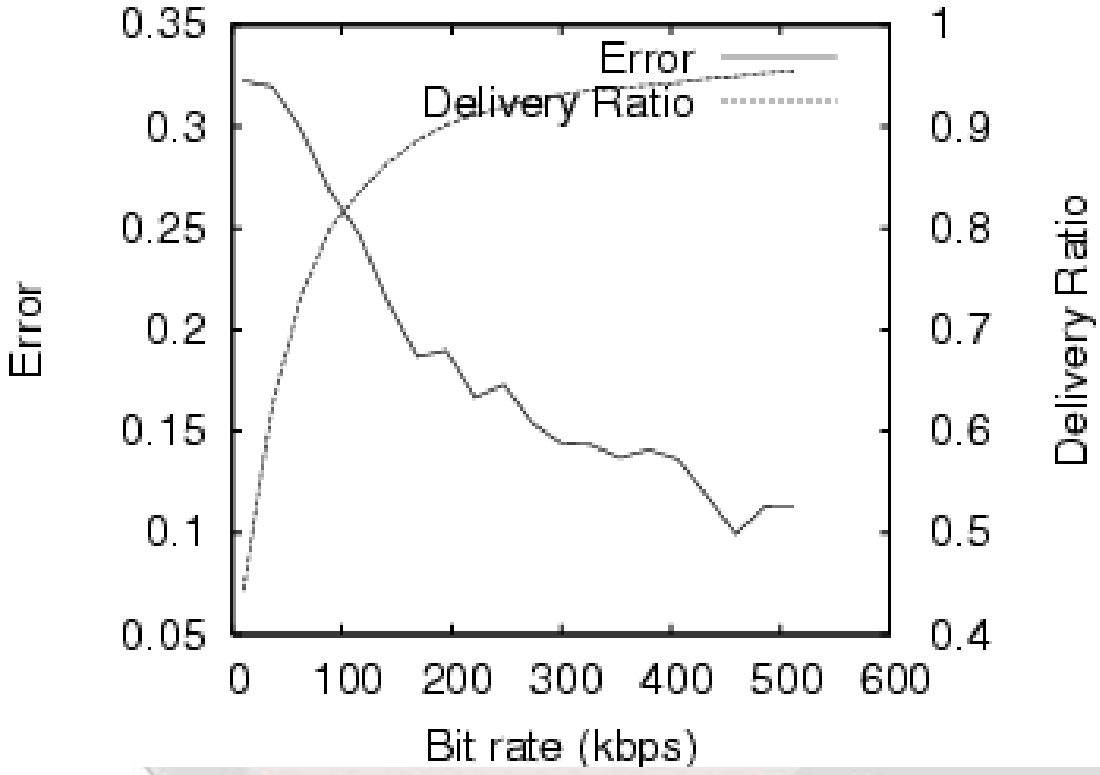


Optimisations



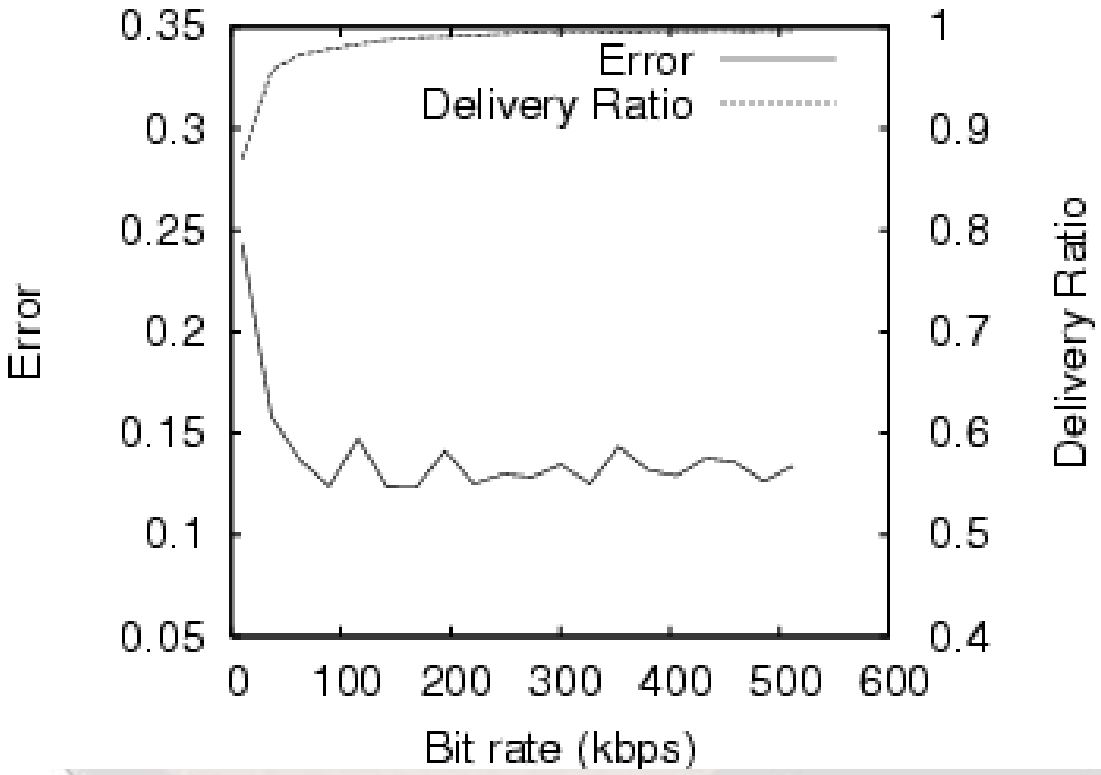


Bit rate



Doesn't look so good. The message rate is too high – saturating the carrier.

Bit rate – more sane



Specks broadcasting every 9 seconds here

Questions