Ant Routing Simulation

B.A. Bagula, H.A.C. de Villiers, J. du Toit, A.E. Krzesinski, M. Loubser,

J.G. van der Horst

Department of Computer Science
University of Stellenbosch
7600 Stellenbosch
South Africa

This research was funded by grants from the South African National Research Foundation, Siemens Telecommunications and Telkom SA Limited.

Introduction

A swarm is a decentralized, autonomous multi-agent system consisting of many simple, cooperative agents which behave according to simple rules & use local information in a common environment to achieve global goals.

Swarm intelligence attempts to design algorithms or distributed problem solving devices inspired by the collective behaviour of social insect colonies & other animal societies.

The distributed nature of ant colonies & their ability to adapt when conditions change makes them a promising paradigm for designing optimal routing algorithms for modern telecommunication networks.

Introduction

We simulate an ant routing algorithm which applies swarm intelligence to find optimal routes in a telecommunication network.

We investigate

- *the efficiency of two versions of the algorithm in finding optimal routes*
- *the properties of the routes*
- *how the ant routing algorithms respond to link failure*
- *path discovery when two previously disconnected routing areas are connected.*

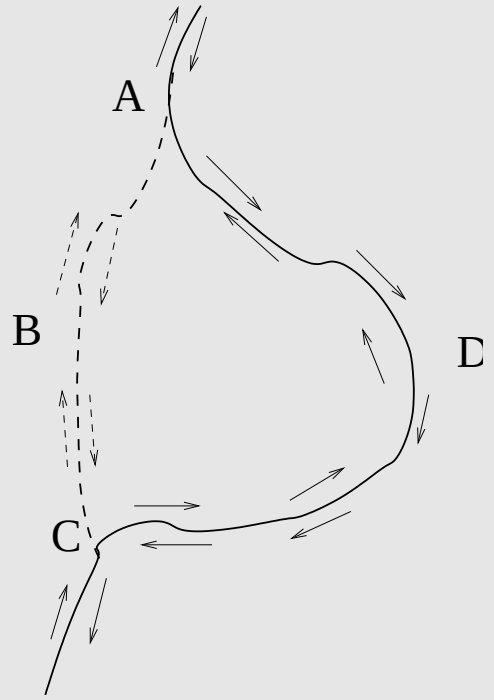
Introduction

Ant colonies are an example of swarm intelligence exhibiting

- *Cooperative behaviour: ants cooperate using a chemical called pheromone as a medium of communication.*
- *Stigmergetic communication: when ants forage, they randomly wander and lay a pheromone trail which leads other ants to a source of food. Many ants may discover different routes to the same food.*
- *Autocatalytic behaviour: the number of ants that travel a path determines the strength of the pheromone trail. The ants which travel the shortest path reinforce the path with more pheromone which aids other ants to follow. After an initial randomization the ants finally arrive at the shortest path.*

Ants find efficient routes

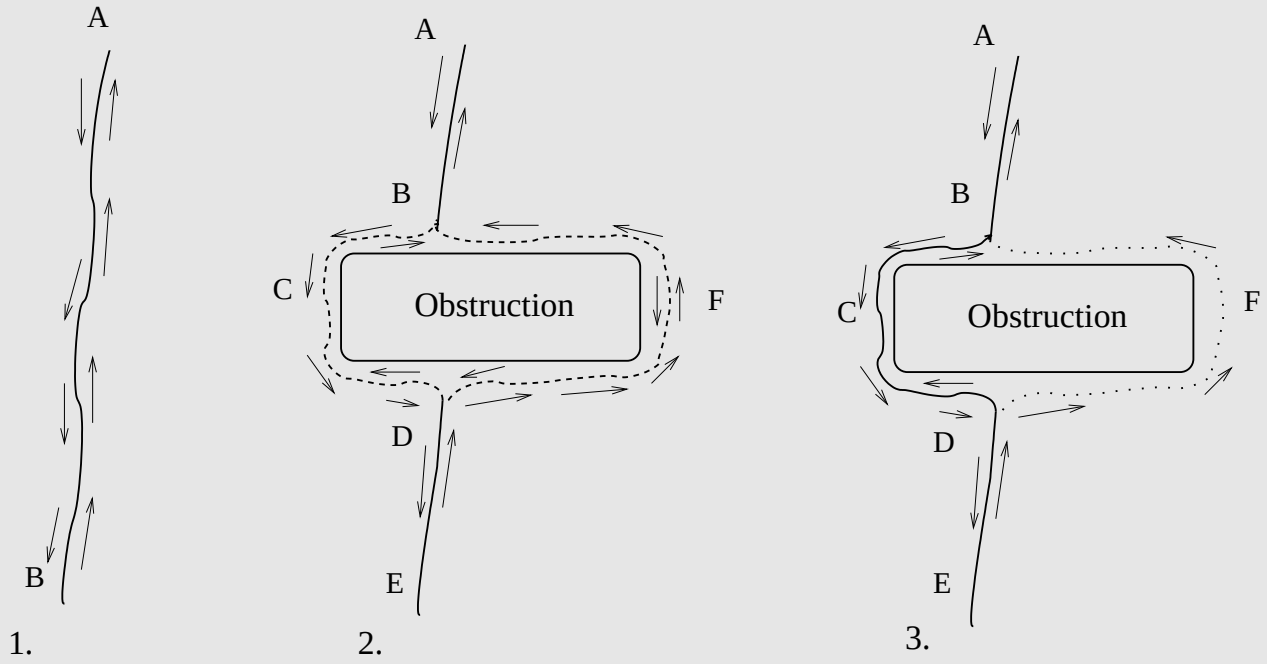
The number of ants that travel a path determines the strength of the pheromone trail.



The longer route ADC is replaced by the shorter route ABC.

Ant routes adapt

When a path is disrupted, the ants explore until the path is found again, or an alternative path is found.



When multiple paths are found, the shortest path is selected.

IP routing

Each router in an IP network contains a routing table.

The routing table contains one entry for each destination in the router's domain.

Each entry specifies the next router to be visited on the shortest path to a given destination.

The ant routing algorithm

The basic mechanisms used in ant routing are

- *Each routing table contains multiple entries for each destination in the router's domain*
  - *each entry specifies the probability of choosing this neighbour as the next hop along the shortest path to the destination.*
- *The next-hop probabilities are initially equal and are updated by ant packets which visit the router.*
- *Each source node sends out ant packets based on the entries in its routing table. The ant packets explore the routes in the network. The ant packets remember their outbound routes.*
- *. . .*

The ant routing algorithm

- *When an ant packet reaches the destination node, the ant packet returns to the source node along the same route.*
- *The ant packet updates the routing table at each node on the return path.*

  The rules for updating the routing tables are

  - *increase the probability of the hop where the ant packet has immediately come from*
  - *decrease the probabilities of the other hops.*
- *. . .*

The ant routing algorithm

- *The route with higher probability is always favoured.*

  More ant packets use that route, increase its use and attract more ant packets.

  The positive feed-back quickly identifies the best paths.
- *If the network load or configuration changes, the ant packets identify and enforce new optimal paths.*

Thus ant routing is dynamic, robust and scalable.

Bi-directional exploration

There are two castes of ant packets

- *`AntOut` packets are sent from a router to find a destination*
- *`AntBack` packets retrace the route from the destination to the originating node.*

AntOut packets

When an AntOut packet passes through a node

- *the address of the node and a timestamp are written into the AntOut packet header*
- *the AntOut packet updates the routing table*
  - *routing probabilities for previously unknown destinations are added to the routing table*
- *the next hop is selected probabilistically and the AntOut packet is transmitted to the next node*
- *the probability that an AntOut packet will return to the node that it has just come from is decreased to accelerate the convergence of the routing algorithm.*

This process is repeated until the AntOut packet reaches its destination or its TTL expires.

AntBack packets

At the destination node an AntBack packet is generated

- *the list of nodes visited by the AntOut packet is written into the AntBack packet header*
- *the AntBack packet follows the route back to the source of the AntOut packet*
- *the AntBack packet updates the routing tables in the nodes along its path*
  - *uni-directional: AntOut packets do not update the routing tables*
  - *bi-directional: AntOut and AntBack packets update the routing tables.*

Explorer packets

The routers are initialized with a non-existent router in their routing tables

- *the routers to create Explorer packets which wander through the network until their TTLs expire*
- *as the number of entries in the router table increases, AntOut packets with valid destinations are generated*
- *the number of Explorer packets decreases and their role is filled by AntOut packets*
- *all ant packets contribute to the mapping of the network, even those ant packets whose TTLs expire are useful.*

Routing table updates

A scale factor α is used to adjust the magnitude of a routing table update

$$\alpha = 0.1 + 0.2/\sqrt{2\tau + 2}$$

where τ is the link propagation delay.

The scale factor is a monotone decreasing function of τ so that nearby nodes have a greater effect on the entries in the routing table than distant nodes.

The scale factor α has a lower limit of 0.1 so that distant nodes can influence the routing table.

The value of α is sufficiently small to avoid instability in the network which might occur by over-reacting to a single ant packet.

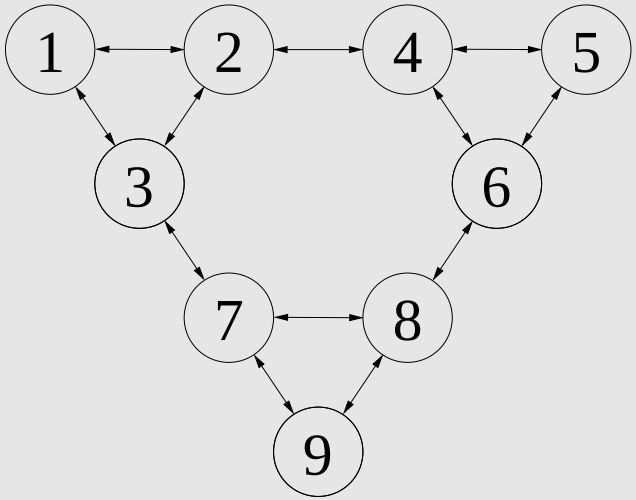
Simulating ant routing algorithms

The ant routing algorithms were implemented in Objective C and simulated using the Swarm libraries.

Swarm provides data structures and scheduling facilities suited to simulating the multi-agent environment.

- *We model networks whose links have unlimited capacity so that there is no competition for resources between the ant packets and the data packets.*

Network convergence

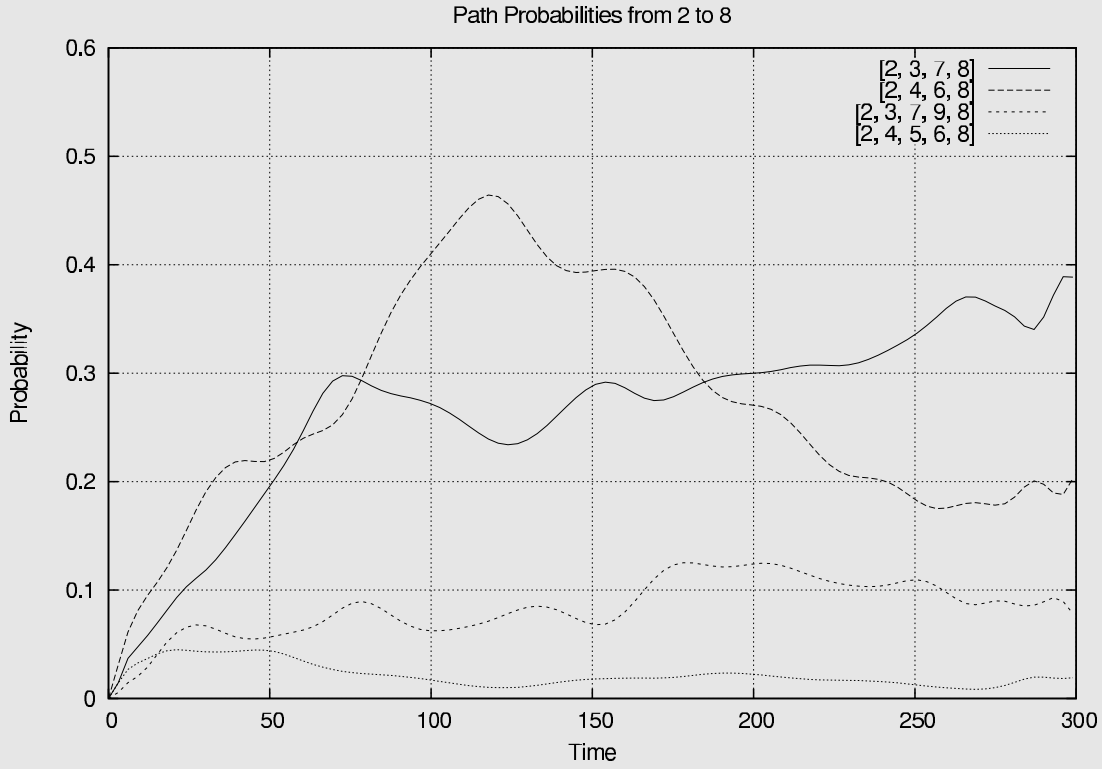


We investigate whether the ant routing algorithm

- *effectively find paths in the network*
- *allows competing equal-cost paths to be exploited equally, so that efficient multi-path routing emerges.*

The bi-directional algorithm: short term behaviour

Four paths are found from node 2 to node 8.

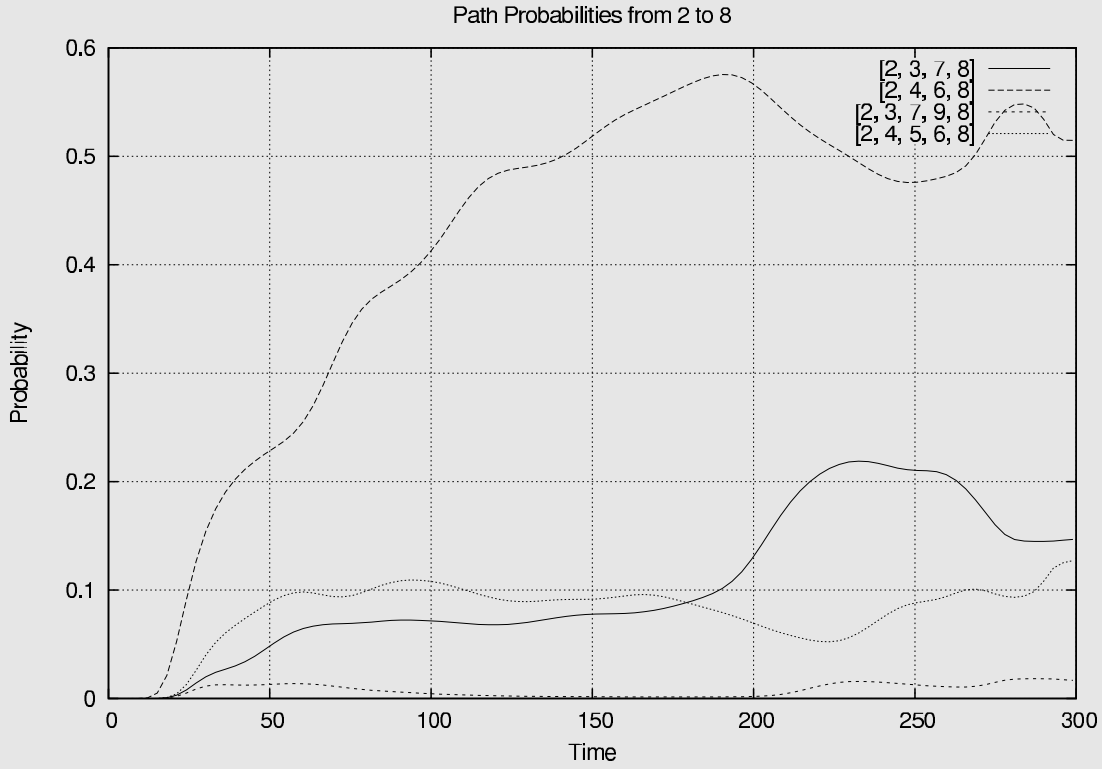


The bi-directional algorithm

- ✓ *finds the (2,8) routes quickly*
- ✓ *distributes the flow over the two shortest paths.*

The uni-directional algorithm: short term behaviour

Four paths are found from node 2 to node 8.



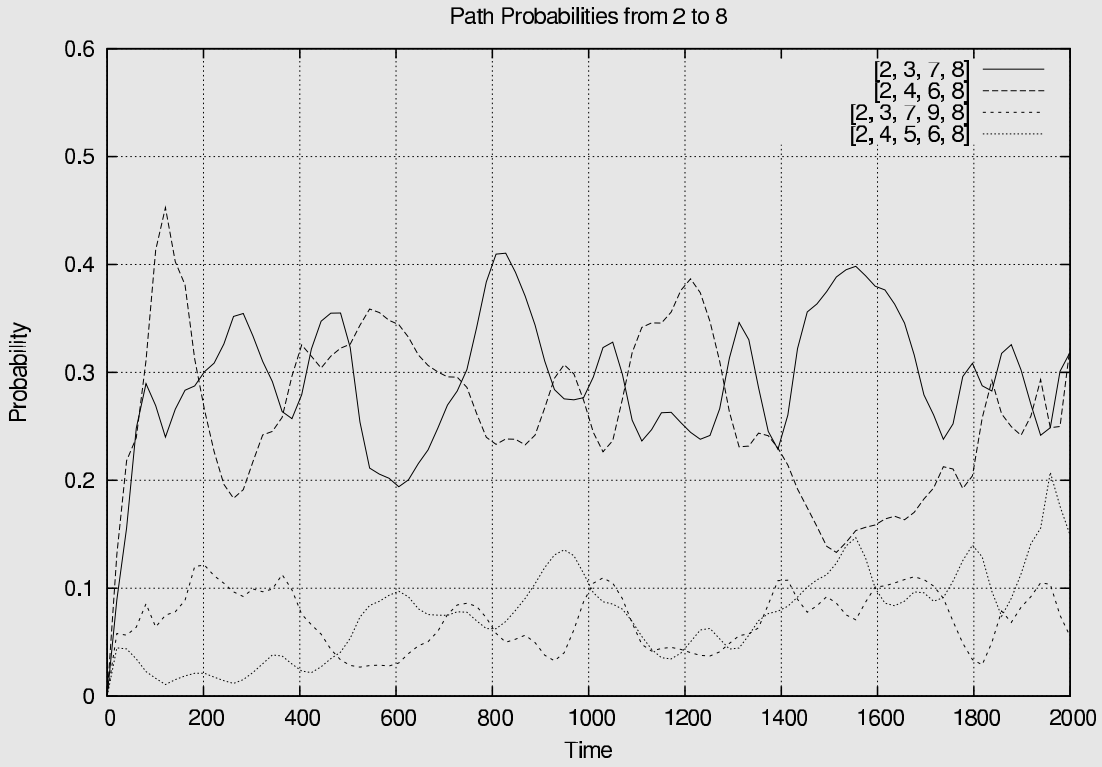
The uni-directional algorithm

✓ *finds the (2,8) routes quickly*

✗ *gives preference to one of the routes.*

The bi-directional algorithm: long term behaviour

Four paths are found from node 2 to node 8.

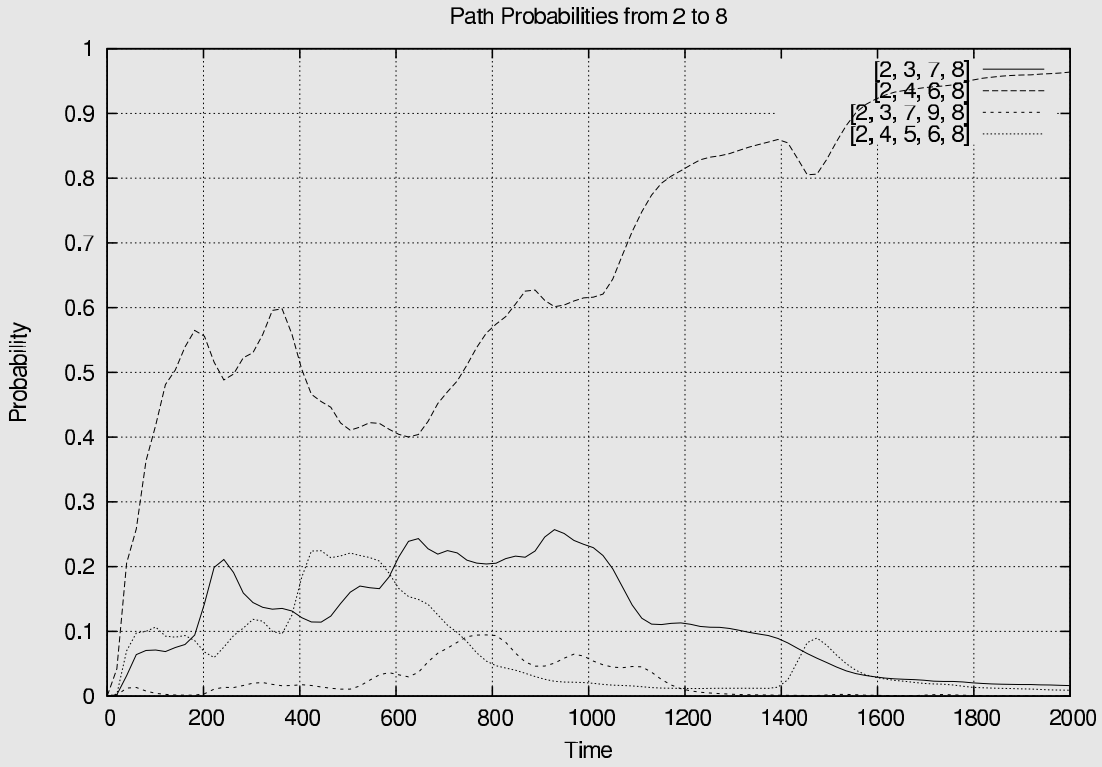


The bi-directional algorithm

- ✓ *favours the shortest paths*
- ✓ *also uses the longer paths.*

The uni-directional algorithm: long term behaviour

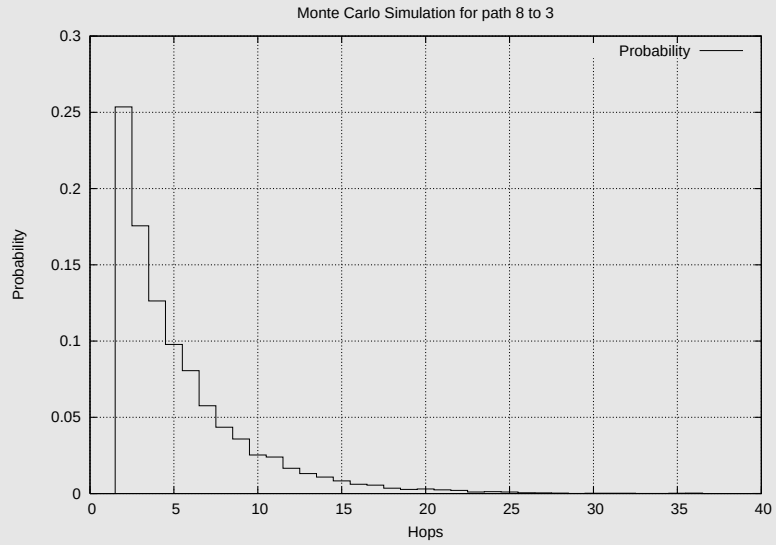
Four paths are found from node 2 to node 8.



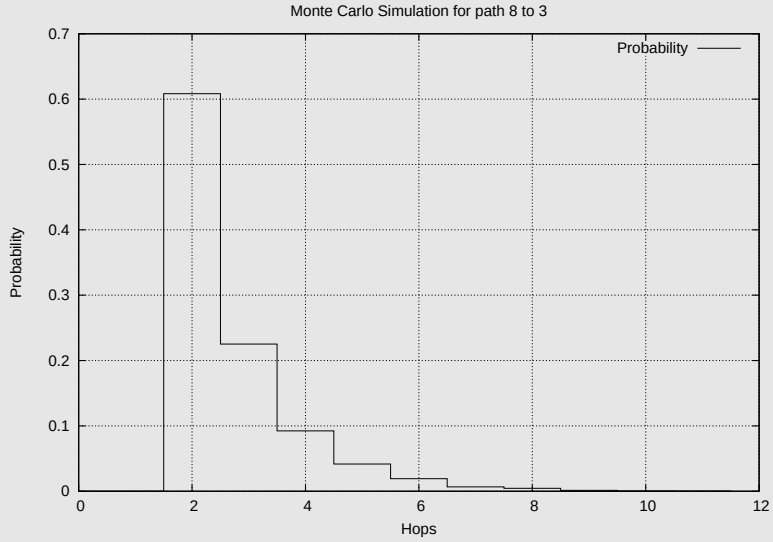
The uni-directional algorithm

✗ *fails to distribute the flow across the two shortest paths.*

Evolution of path length distribution

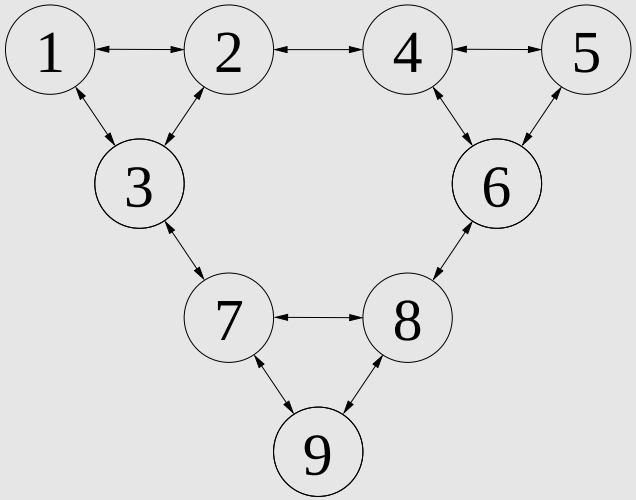


At time $t = 20$.



At time $t = 300$.

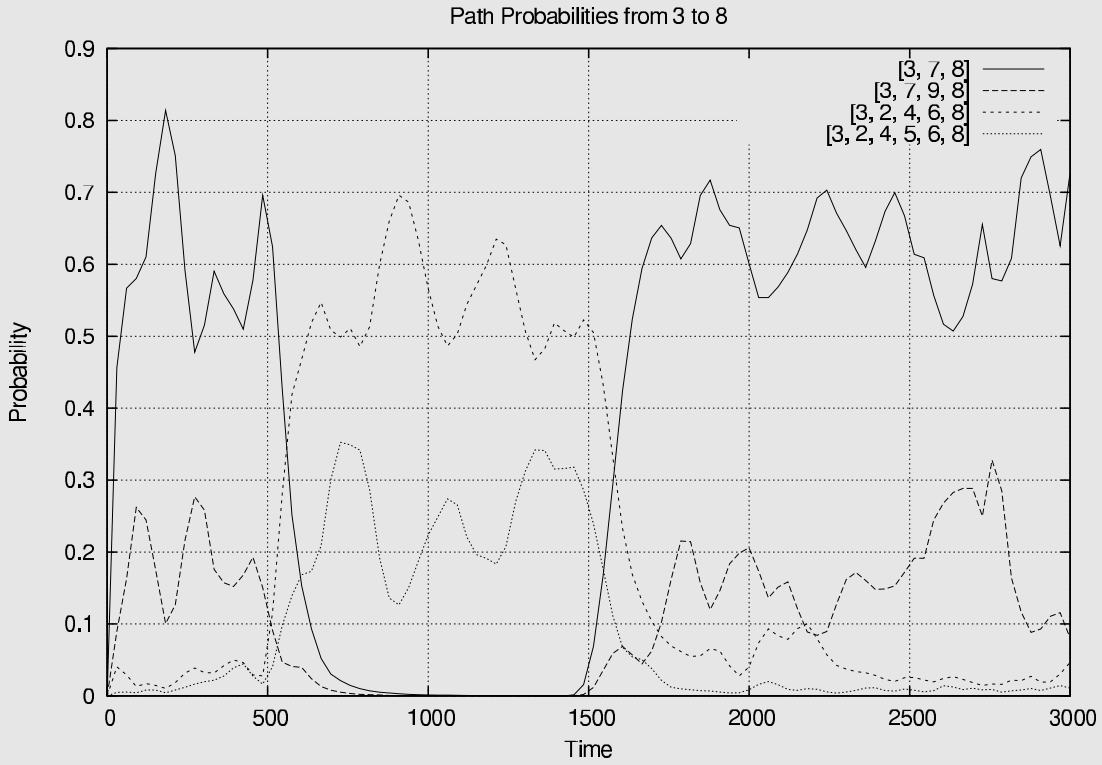
Link failure and recovery



The link from node 3 to node 7 fails at time step 500. The link is restored at time step 1500.

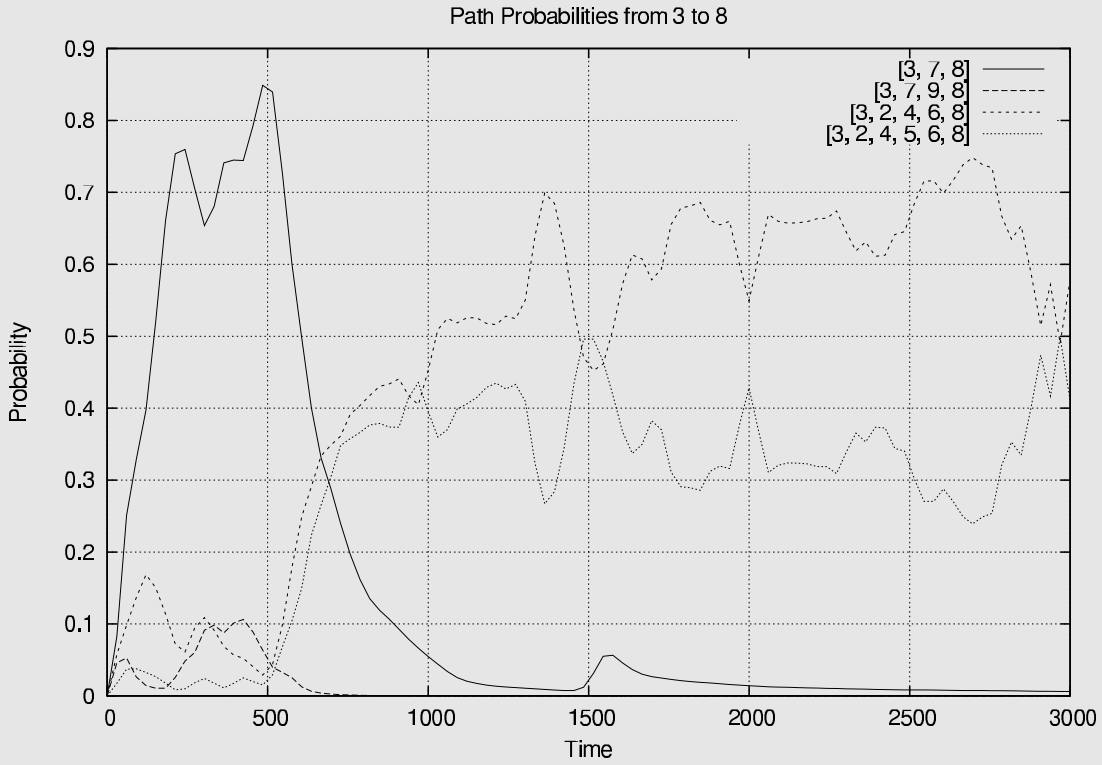
We investigate how effectively the uni- and bi-directional algorithms take advantage of the recovered link.

Link recovery: the bi-directional algorithm



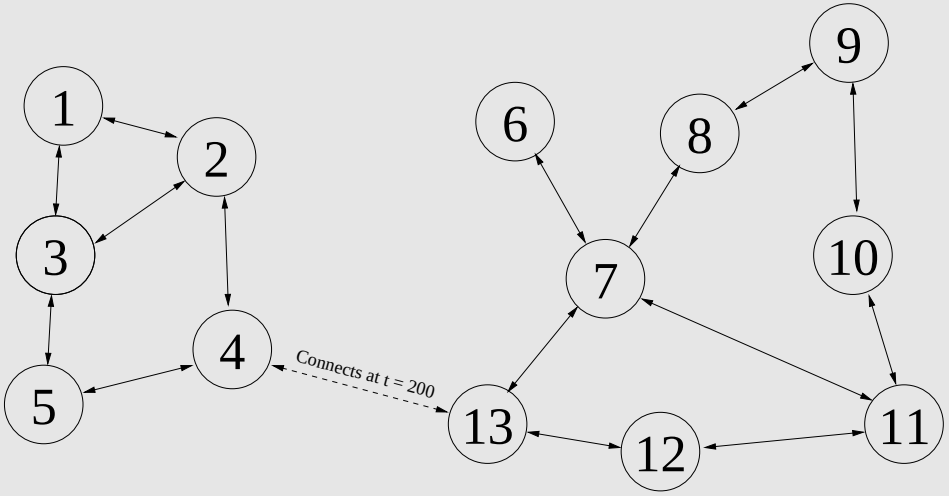
- ✓ *the link failure is discovered immediately*
- ✓ *the restored link is discovered immediately*
- ✓ *the pre-failure state is restored.*

Link recovery: the uni-directional algorithm



- ✗ *the link failure is gradually discovered*
- ✗ *the restored link is not detected*
- ✗ *the pre-failure state is not restored.*

Connecting two routing areas

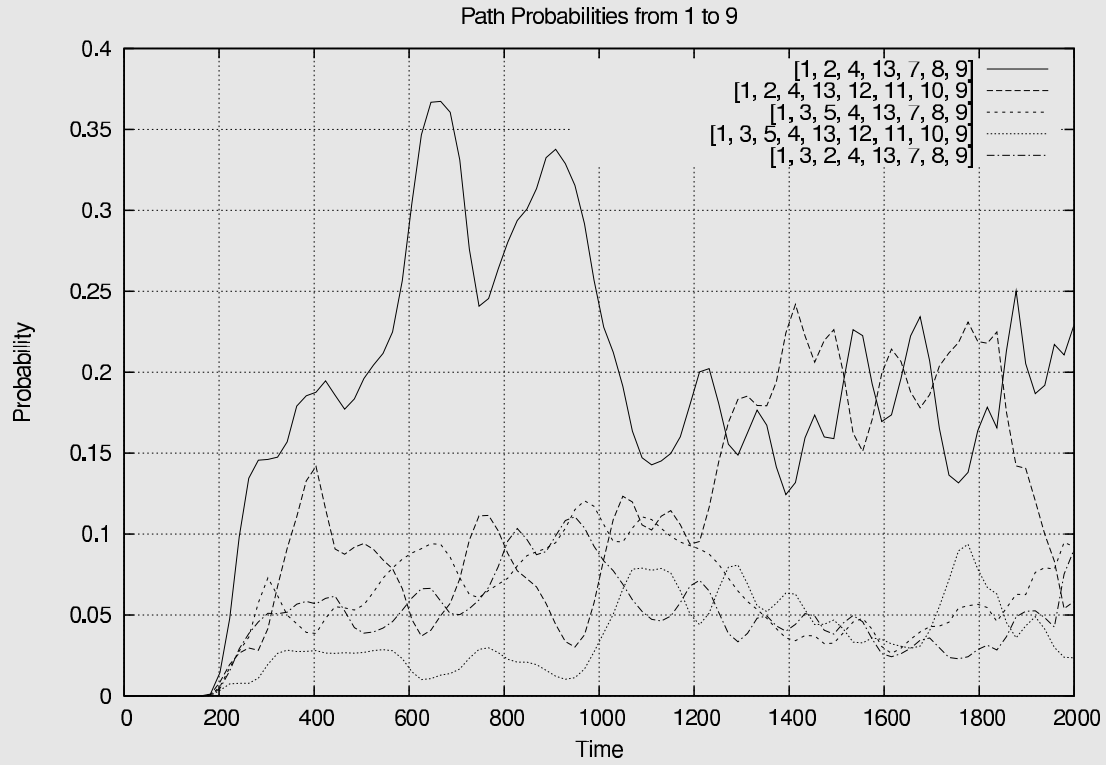


The link (4,13) connecting the two areas is enabled at time step 200.

`Explorer` packets are released every 5 time steps.

We investigate how effectively the uni- and bi-directional algorithms find inter-area routes.

Connecting two areas: the bi-directional algorithm



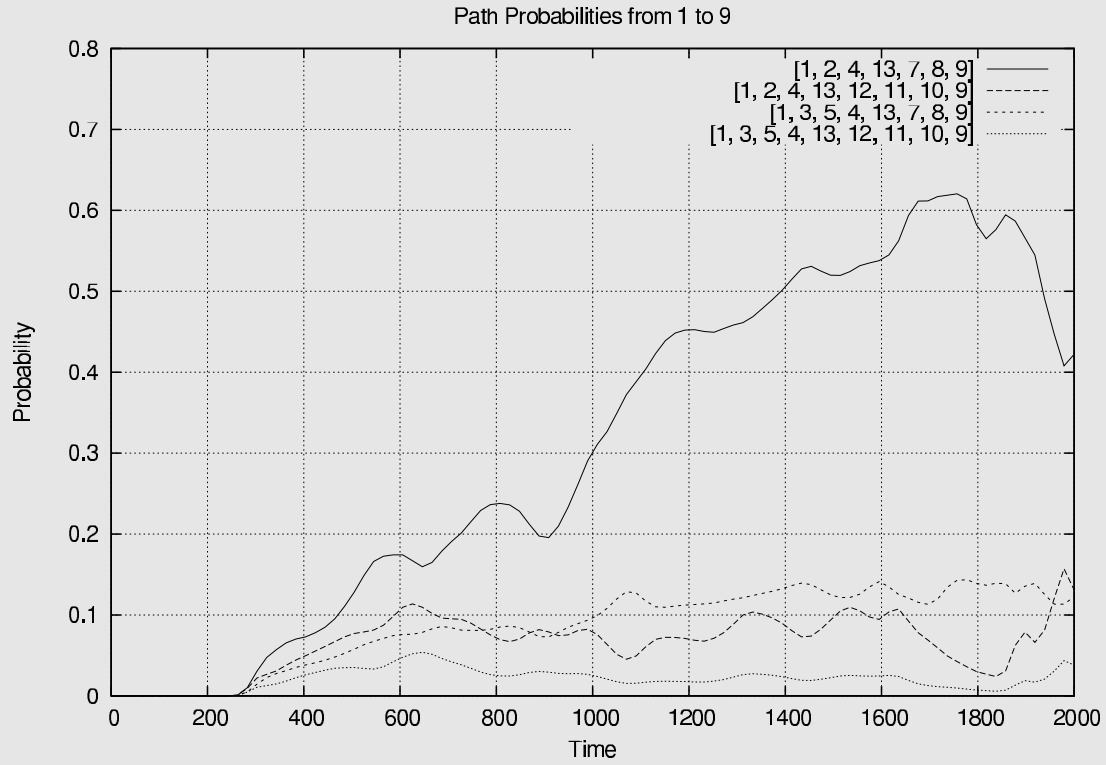
The routing probabilities from node 1 to node 9

$t = 200$ *the areas are connected*

$t = 206$ *inter-area routes are established*

$t = 1200$ *multi-path shortest routes are available.*

Connecting two areas: the uni-directional algorithm



The routing probabilities from node 1 to node 9

$t = 200$ *the areas are connected*

$t = 290$ *inter-area routes are established*

$t = 1600$ *one path dominates.*

Uni- versus bi-directional algorithms

The uni-directional algorithm

- ✓ *the routing probabilities are less volatile*
  - *the volatility is caused by the* `AntOut` *packets*
- ✗ *converges more slowly.*

The bi-directional algorithm

- ✓ *finds efficient multi-paths*
- ✓ *is best suited to environments where the topology and traffic change frequently and significantly.*