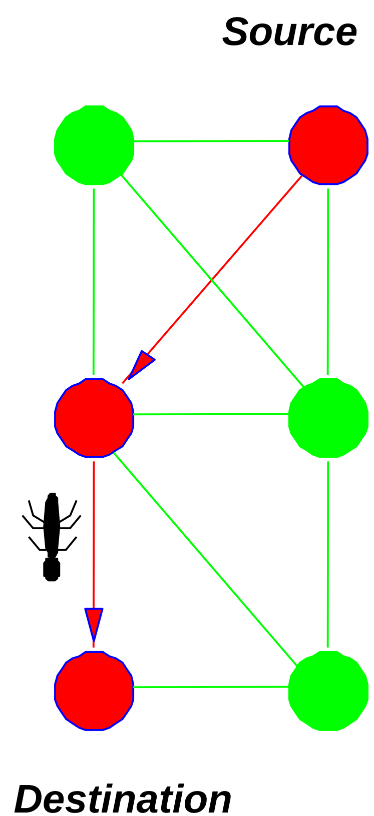
ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Distributed Intelligent Systems – W2:

Multi-Agent Systems based on Ant Trail Laying/Following Mechanisms: Algorithms and Applications

Outline

- Moving beyond the original AS
  - Ant Colony System (ACS)
  - ACS with local search for TSP: ACS-3-Opt
- Trail laying and following mechanisms applied to network routing
  - ABC
  - AntNet
- ACO summary



Extending Ant System: The Ant Colony System Algorithm

Constructive Heuristic and Local Search

Current wisdom says that a very good strategy for the approximate solution of NP-hard combinatorial optimization problems is the coupling of:

- **a constructive heuristic** (i.e. generate solutions from scratch by iteratively adding solution components)
- **local search** (i.e., start from some initial solution and repeatedly tries to improve by local changes)

These two methods are highly complementary. The problem is to find good couplings: ACO appears (as shown by experimental evidence) to provide such a good coupling.

2 Extensions of AS

- **Ant Colony System (ACS)** – improved constructive heuristic

  (Gambardella & Dorigo, 1996; Dorigo & Gambardella, 1997)
  - Different transition rule
  - Different pheromone trail updating rules: global and local
  - Use of a candidate list for the choice of the next city

- **ACS-3-opt** – constructive heuristic + local search

  (Gambardella & Dorigo, 1996; Dorigo & Gambardella, 1997)
  - Standard ACS + local search
  - In case of TSP problems, 2-opt (2 edges exchanged), 3-opt (3 edges exchanged), and Lin-Kernighan (variable number of edges exchanged) are used as local search algorithms

Ant Colony System

Loop * t=0; t:=t+1 *

Place one ant on each node *there are *n* nodes *

For k := 1 to *m* * each ant builds a solution, in this case m=n*

For step := 1 to *n* * each ant adds a node to its path *

Choose the next city to move by applying a probabilistic ***solution construction rule***

End-for

End-for

Update pheromone trails

Until End_condition * e.g., t=tmax *

Different Transition Rule

An ant k on city i chooses the city j to move according to the following rule:

$$j = \begin{cases} \arg\max_{u \in J_i^k} \{[\tau_{iu}(t)][\eta_{iu}]^{\beta}\} & if \quad q \le q_0 \\ J \quad if \quad q > q_0 \end{cases}$$

With $J \in J_i^k$ being a city that is randomly selected according to:

$$p_{iJ}^k(t) = \frac{[\tau_{iJ}(t)][\eta_{iJ}]^{\beta}}{\sum_{l \in J_i^k} [\tau_{il}(t)][\eta_{il}]^{\beta}}$$

- **q:** uniform distributed random variable [0,1]
- **q_0:** parameter between 0 and 1, controls exploration/exploitation ($q_0 < 0$ as AS)
- **$q \le q_0$:** deterministic rule, exploitation of the current knowledge of the problem (problem heuristic knowledge + learned knowledge)
- **$q > q_0$:** probabilistic rule, more exploration, roulette wheel like in the original AS

Virtual Pheromone: Global Update with Elitism

AS: all ants can update pheromones trails in the same way

EAS: all ants update pheromones trails; extra amount for the best tour

ACS: the global update is performed exclusively by the ant that generated the best tour from the beginning of the trial; it updates only the edges of the best tour T^+ of length L^+ since the beginning of the trial (*best-so-far*, saving of computing time, no major difference with *best-of-iteration*)

Update rule for (i,j) edges belonging to T+:

$$\tau_{ij}(t+1) \leftarrow (1-\rho)\tau_{ij}(t) + \rho\Delta\tau_{ij}(t)$$

with $\Delta\tau_{ij}(t) = 1/L^+$

Note 1: the result is a more directed, greedy search; ants are encouraged to search for path in the vicinity of the best tour found so far.

Note 2: notice the weighted sum of old and new pheromones, different from AS.

Virtual Pheromone: Local Update

All ants can perform a local update. When an ant k in city i select city $j \in J_i^k$ the pheromone concentration on edge (i, j) is updated as follows:

$$\tau_{ij}(t+1) \leftarrow (1-\xi)\tau_{ij}(t) + \xi\tau_0$$

ξ: parameter; $\xi = 0.1$ from experimental finding, Bonabeau et al. book $\xi = \rho$

τ_0: parameter, also representing the initial pheromone quantity on all edges (like in AS). From experimental finding: $\tau_0 = (nL_{nn})^{-1}$; n = number of cities, L_{nn} = length of the tour produced by the nearest neighbor heuristic only

Note: application of the local update rule make the pheromone level decreases each time that an edge is visited → indirectly favor exploration of not yet visited edges → avoid stagnation and convergence to a common path → increase probability that one of the m ants finds a even better T^+

Candidate List

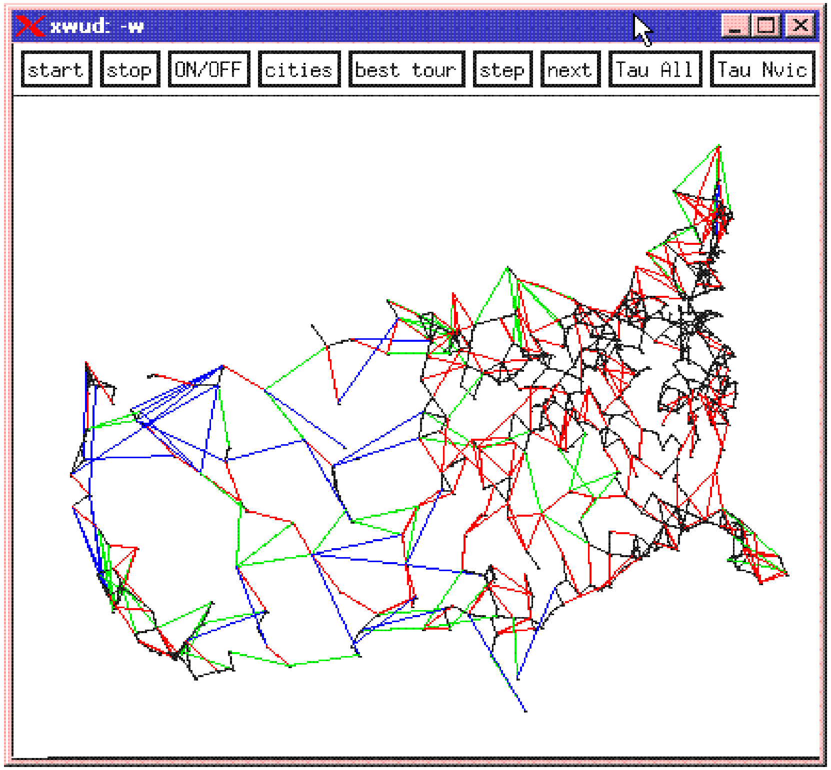
- A *candidate list* is a list of cities of length *cl* (*cl* = algorithmic parameter) to be visited from a given city; cities in the candidate list are ranked according to the inverse of their distance, the list is scanned sequentially.
- An ant first restrict the choice of the next city to those in the candidate list; it uses standard ACS transition rule to select a city.
- Once all the *cl* closest cities in the candidate list for a given city *i* have been visited, the next city *j* is selected from the closest of the yet unvisited cities.

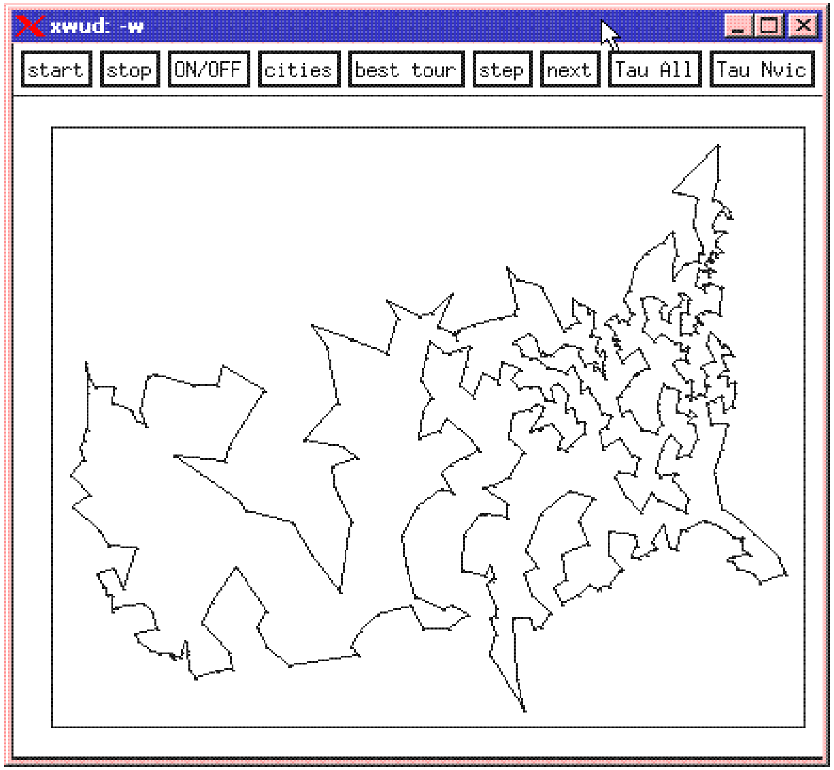
ACS for TSP – Comparison with Other Algorithms

ACS ran for 1250 iterations (end criterion) using 20 ants (25'000 tours generated); average over 15 runs for all implementations; see Bonabeau et al. book and further pointers slide for more results

Problem	Ant Colony System (Dorigo & Gambardella,1997)		Genetic Algorithms (Whitley *et al.*, 1989)		Simulated Annealing (Lin *et al.*, 1993)	
	Shortest tour lenght	# of tours before best tour found	Shortest tour lenght	# of tours before best tour found	Shortest tour lenght	# of tours before best tour found
Eil50 (50 cities)	425	1830	428	25000	443	68512
Eil75 (75 cities)	535	3480	545	80000	580	173250
KroA100 (100 cities)	21282	4820	21761	103000	N/A	N/A

ACS for TSP – Results on ATT532 Problem





2 Extensions of AS

- **Ant Colony System (ACS)**
  (Gambardella & Dorigo, 1996; Dorigo & Gambardella, 1997)
  - Different transition rule
  - Different pheromone trail updating rule
  - Use of local update of pheromone trail
  - Use of a candidate list for the choice of the next city

- **ACS-3-opt**
  (Gambardella & Dorigo, 1996; Dorigo & Gambardella, 1997)
  - Standard ACS + local search
  - In case of TSP problems, 2-opt (2 edges exchanged), 3-opt (3 edges exchanged), and Lin-Kernighan (variable number of edges exchanged) are used as local search algorithms

ACS + Local Search

Loop * t=0; t:=t+1 *

Place one ant on each node * there are *n* nodes *

For k := 1 to *m* * each ant builds a solution, in this case m=n *

For step := 1 to *n* * each ant adds a node to its path *

Choose the next city to move by applying a probabilistic ***solution construction rule***

End-for

Apply local search

End-for

Update pheromone trails

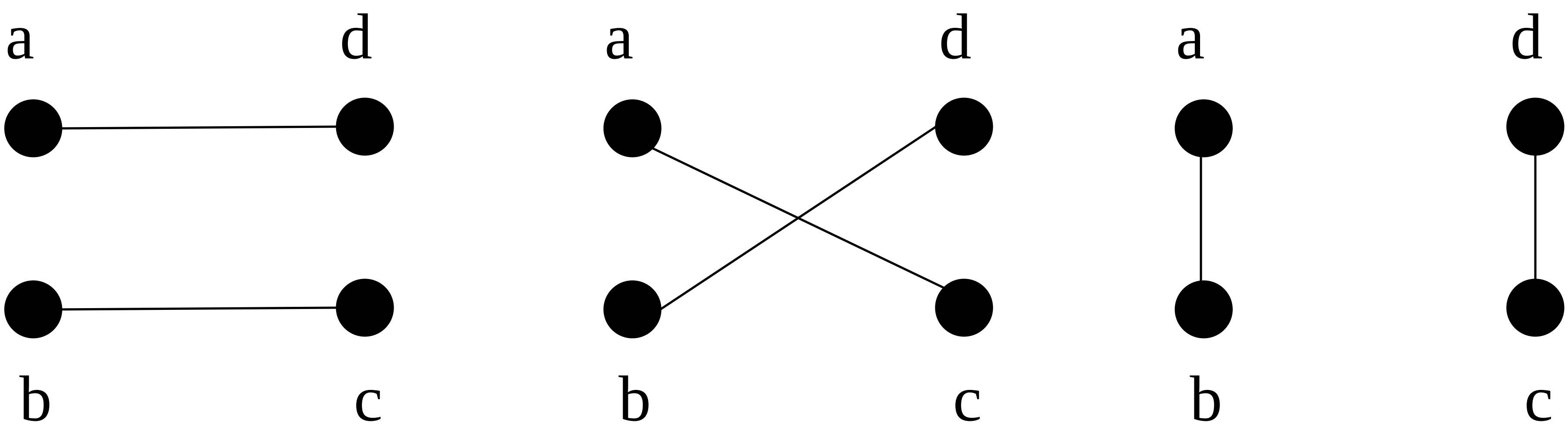
Until End_condition * e.g., t=tmax *

k-opt Heuristic

- Take a given tour and delete (up to) k mutually disjoint edges
- Each fragment endpoint can be connected to $2k-2$ other possibilities: of $2k$ total fragment endpoints available, the two endpoints of the fragment under consideration are disallowed.
- Reassemble the remaining fragments into a tour, leaving no disjoint subtours (that is, don't connect fragment's endpoints together).
- Do this systematically: generate the set of all candidates solutions possible by exchanging in all possible ways (up to) k edges

2-Opt

2*2-2 = 2 alternative options

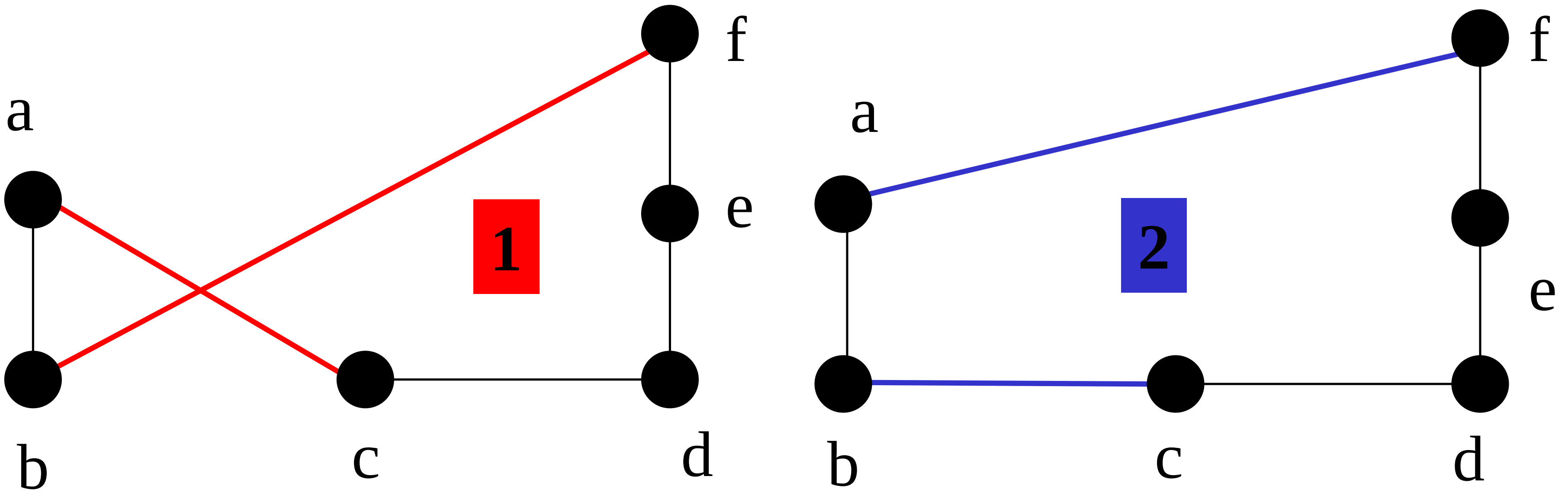


Original
(connected graph,
single tour)

Variant 1
(connected graph,
single tour)

Variant 2
(disconnected graph,
2 sub-tours)

Example of Local Search: 2-Opt



- 2-opt swapping: (b,f) and (a,c) replaced by (a,f) and (b,c)
- Tour 2 shorter than tour 1
- Note on pheromones update:
  - candidate tours at iteration t are not marked with pheromones immediately, they are just built based on pheromones of iteration t-1
  - pheromones are updated on the edges of the already locally optimized solutions (i.e. evaporation, local and global update after local search).

For TSP Problems: Local Search using 3-Opt

- For each ant k, at each iteration of ACS, up to three edges at the time are exchanged iteratively until a local optimum is reached while all other sub-tour orientations are maintained unchanged; full 3-opt, 2*3-2 = 4 alternative combinations for reconnecting a disconnected edge; in total 8 valid permutation of 3 edges at the time, including 4 degenerating to exchanges of 2 edges only (2-opt exchanges must be considered in 3-opt search as well)
- In ACS-3-opt restricted permutation: only moves that *do not revert the order* in which the cities are visited, e.g.: (k,l), (p,q), (r,s) → (k,q), (p,s), (r,l)
- Computational speed-up obtained by using nearest neighbor list, 2.5-opt algorithm, etc. See for instance [Bentley 1992].

Advantages and Drawbacks of Local Search

- Local search is complementary to ant pheromone mechanisms, so probability it achieves a major impact on a given problem is high
- The quality of the achieved solution is in particular improved; the computational cost is increased
- Local search lacks of good starting solutions on which it can perform combinatorial optimization; these solutions are provided by artificial ants using pheromone mechanisms
- Depending on targeted performance metrics (wished solution quality vs. and desired computational cost) an appropriate balance between local search and constructive heuristic has to be chosen

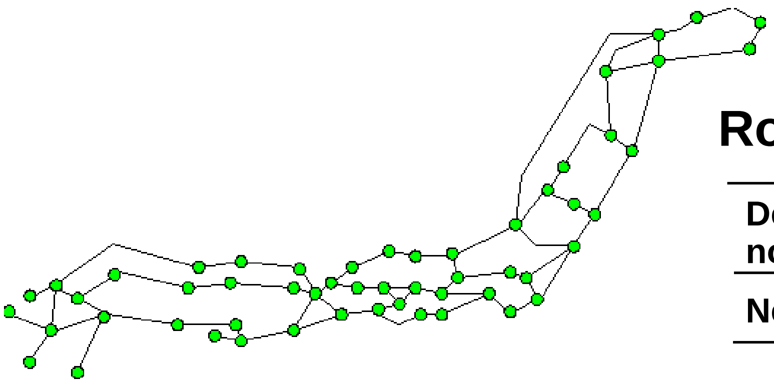
ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Disal

Trail Laying/Following Mechanisms applied to Communication Networks: Ant-Based Routing Algorithms

The Routing Problem

- The practical goal of routing algorithms is to build routing tables



Routing table of node k (N-nodes net)

Destination node	**1**	**...**	***j***	**...**	***k*-1**	***k*+1**	**...**	***N***
Next node	i_1	**...**	i_j	**...**	i_{k-1}	i_{k+1}	**...**	i_N

- Routing is difficult because costs are dynamic
- Adaptive routing is difficult because changes in the control policy determine changes in the costs and vice versa

Two Main Algorithms up to Date

- ABC (Ant-Based Control), [Schoonderwoerd, Holland, et al., 1996]
  - Target: telephone network (symmetric level of congestion on one given source-destination pair)
  - Test: UK telephone network
- AntNet [DiCaro and Dorigo, 1998]
  - Target: packet-switching network style Internet
  - Tests: more exhaustive on several networks

Ant-Based Control (ABC) Algorithm

ABC Algorithm - Definitions

- d = destination; s = source, n = neighbor node, t = real time
- Assumption: same level of traffic congestion s-d and d-s (ok for telephone networks)
- N nodes total, k_i = neighboring nodes to node i
- Routing table node i (time-variant matrix with k_i rows and N-1 columns):

$$R_i = [r^i_{n,d}(t)]_{k_i, N-1}$$

$r^i_{n,d}(t)$: For ants: probability that an ant with destination d will be routed from i to neighbor n

For calls: deterministic path (pick up the higher value for choosing the route from i to neighbor n)

$\sum_n r^i_{n,d}(t) = 1$ Sum of all possible routes to neighbors at a given node = 1

ABC – Updating Rules

- Ants launched from any node (exist an optimal rate) continuously; travel from s → d
- Ants die when they reach d
- For routing table updating: s is viewed as d (ant has only information about the traffic at visited nodes; information used by future ants and calls)
- i-1 = neighbor the ant came from before joining i
- Each visited node's routing table updated according to:

$$r^{i}_{i-1,s}(t+1) = \frac{r^{i}_{i-1,s}(t) + \delta r}{1 + \delta r}$$

Reinforce

$$r^{i}_{n,s}(t+1) = \frac{r^{i}_{n,s}(t)}{1 + \delta r} \quad \text{for } n \neq i\text{-}1$$

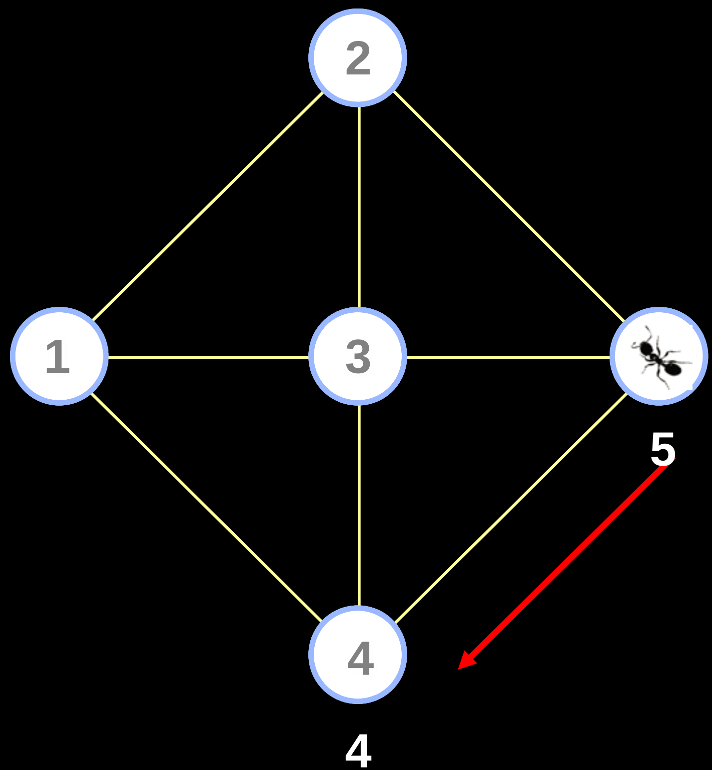
Decay

$\sum_{n} r^{i}_{n,d}(t) = 1$ for all n

Preserved !

δr = reinforcement parameter

A Simple Network Example with 5 Nodes



Routing table of node 4

Destination Node

4	1	2	3	5
1	0,8	0,3	0,1	0,1
3	0,1	0,4	0,8	0,1
5	0,1	0,3	0,1	0,8

Neigboring Nodes

Ant destination

Probabilities for the next hop of the ant

Table entries to be updated

Ex.: ant with node 5 as source and node 2 as destination

ABC Algorithm – Definitions

- Node i has (maximal) capacity C_i (max number of connections, static) and spare capacity S_i (capacity available for new connections, dynamic)
- Once call set-up between d and s, each node in the route is decreased in its spare capacity by a given amount
- If no spare capacity left for at least one of the node in the route under construction, the call is rejected
- When a call terminates (hanging up or rejection), the corresponding reserved capacity for each of the nodes in the route is made available again for other nodes

ABC – Reinforcement

$$\delta r = \frac{a}{T} + b$$

T = absolute time spent in the network

a,b = parameters

Idea: ants have an age, the older they are and the less influence they have on the routing table; ants age faster if they pick up congested routes

ABC – Enforced Delay & Noise

- Delay imposed on ant reaching a given node i:

$$D_i = ce^{-dS_i}$$

c,d parameters

S_i spare capacity

Idea: less congested nodes delay less ants

- Tunable noise parameter g
  - g = probability to chose route at random
  - 1-g = probability to choose route according to routing tables

Idea: increase exploration

ABC - Sample Results

Call failure percentage with different algorithms – static call probabilities

- 30-nodes BT network
- 10 runs
- 15’000 time steps total

	Mean	Standard dev.
Without load balancing (fixed, shortest routes)	12.57%	2.16%
Original mobile agents	9.19%	0.78%
Improved mobile agents	4.22%	0.77%
Ants (0% noise)	1.79%	0.54%
Ants (5% noise)	1.99%	0.54%

ABC - Sample Results

Call failure percentage with different algorithms – dynamic call probabilities

- 30-nodes BT network
- 10 runs
- 15’000 time steps total
- after 7’500 steps different set of call probabilities

	Mean	Standard dev.
Without load balancing (fixed, shortest routes)	12.53%	2.04%
Original mobile agents	9.24%	0.80%
Improved mobile agents	4.41%	0.85%
Ants (0% noise)	2.72%	1.24%
Ants (5% noise)	2.56%	1.05%

AntNet Algorithm

Two Main Algorithms up to Date

- ABC (Ant-Based Control), [Schoonderwoerd, Holland, et al., 1996]
  - Target: telephone network;
  - Test: UK telephone network
- AntNet [DiCaro and Dorigo, 1998]
  - Target: packet-switching network style Internet
  - Tests: more exhaustive on several networks

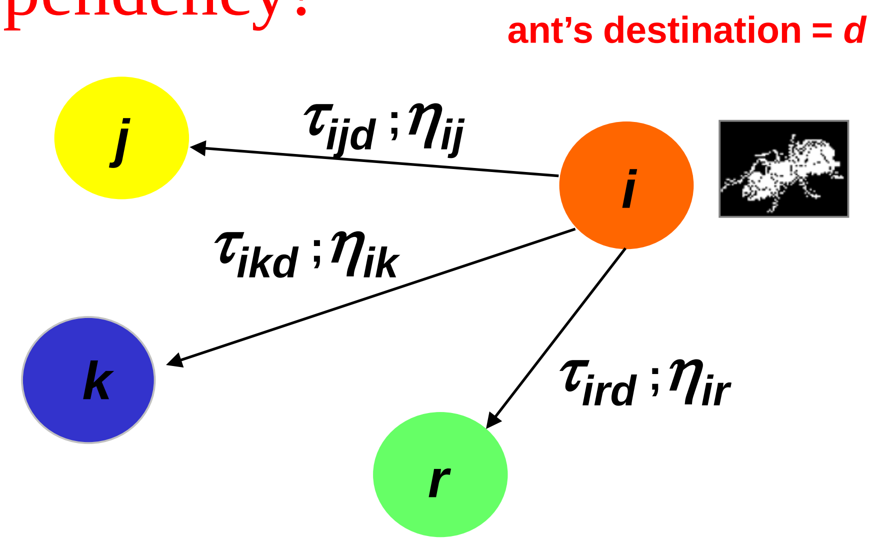
AntNet: The Algorithm

- **Ants** are launched at regular instants, asynchronously from each node to randomly chosen destinations; modulation of ant rate as a function of traffic
- **Ants** build their paths probabilistically with a probability function of:
  - (i) **artificial pheromone values** (stored in the routing tables R), and
  - (ii) **heuristic values** (length of queues, stored in the trip vectors Γ)
- **Ants** memorize visited nodes and elapsed times
- Once reached their destination nodes, **ants** retrace their paths backwards, and update the pheromone trails and trip vectors

Using Pheromones and Heuristic to Choose the Next Node

Additional (real) time-dependency!

$$p_{ijd}^{k}(t) = f\left(\tau_{ijd}(t), \eta_{ij}(t)\right)$$



- τ_{ijd} is the pheromone trail (multiple pheromone trails for the same link i,j!); normalized to 1 for all possible neighboring nodes
- η_{ij} is an heuristic evaluation of link (*i*,*j*) which introduces problem specific information (e.g., in AntNet η_{ij} is $\propto$ to the inverse of link (*i*,*j*) queue length)
- unvisited next nodes first; cycling ants do not update pheromones

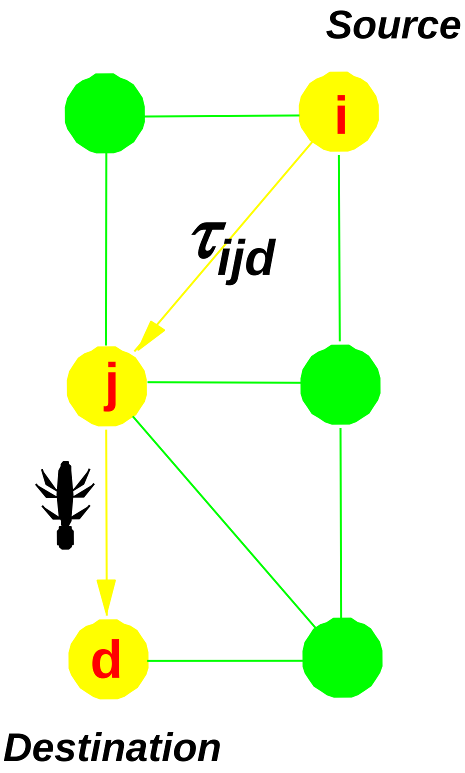
Ants' Pheromone Trail Depositing

$$\tau_{ijd}^{k} \leftarrow (1-\rho)\cdot\tau_{ijd}^{k} + \Delta\tau_{ijd}^{k}$$

where the (*i,j*)'s are the links visited by ant *k*, and

$$\Delta\tau_{ijd}^{k} = quality^{k}$$

where *quality*^{k} is set proportional to the inverse of the time it took ant *k* to build the path from *i* to *d* via *j*



AntNet: Data Structures at Nodes

- **Routing table R_i:**

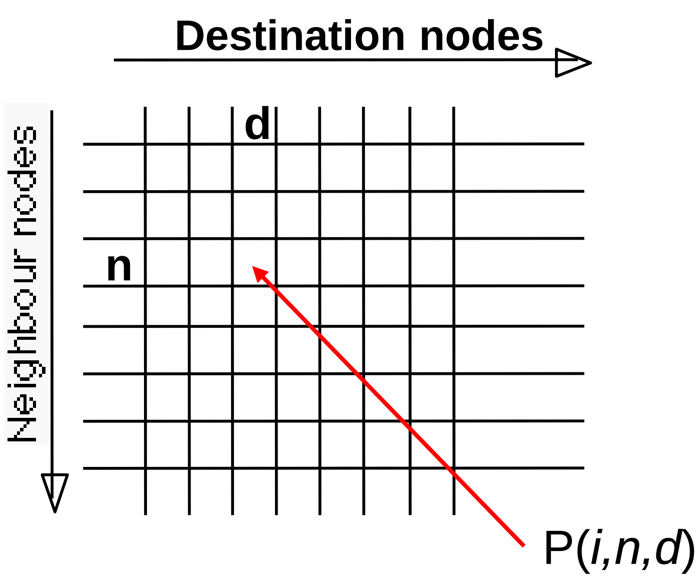
  Memorizes probabilities of choosing each neighbor nodes for each possible final destination

- **Trips vector Γ_i:**

  contains statistics about ants' trip times from current node *i* to each destination node *d* (means and variances); used for calculating pheromone reinforcement ("quality^k")

Trips vector of node *i*				
$\mu(i,1)$ $\sigma^2(i,1)$	…	$\mu(i,d)$ $\sigma^2(i,d)$	…	$\mu(i,N)$ $\sigma^2(i,N)$

Routing table of node *i*



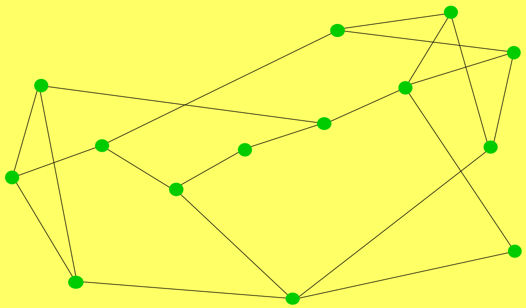
AntNet: the Role of F-ants and of B-ants

- **F-ants** collect implicit and explicit information on available paths and traffic load
  - implicit information, through the arrival rate at their destinations
  - explicit information, by storing experienced trip times
- **F-ants** share queues with data packet
- **B-ants** are F-ants which reached their destination; they fast backpropagate info collected by F-ants to visited nodes and update routing tables R & trip vectors Γ; have a stack memory with visited nodes
- **B-ants** use higher priority queues (usually available on real network for control packages)

AntNet: Experimental setup

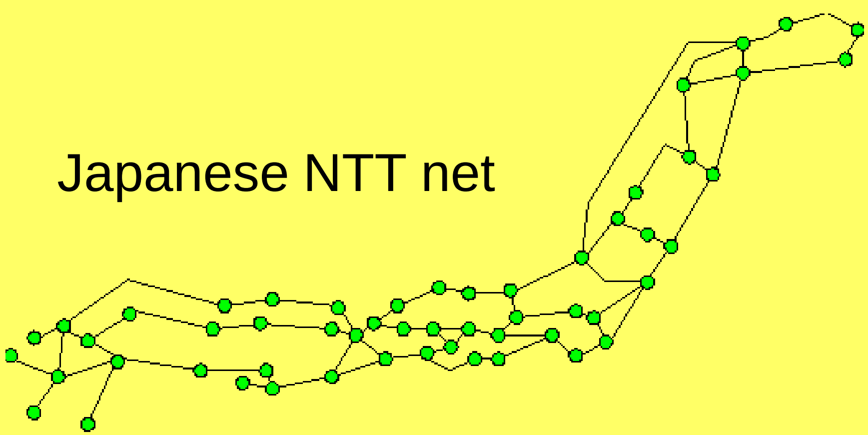
- **Many topologies**
- **Realistic simulator** (discrete events, not standard)
- **Many traffic patterns**
- **Comparison with many state-of-the-art algorithms**
- **Performance measures**

Experimental Setup: Network Topologies

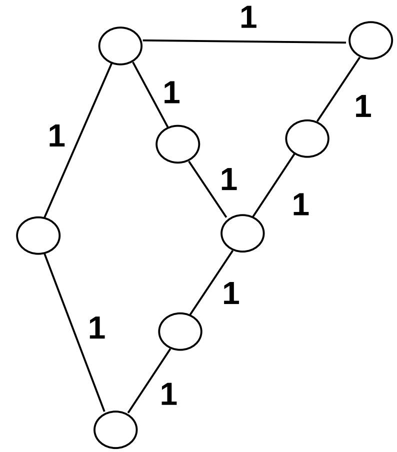


American NSF net

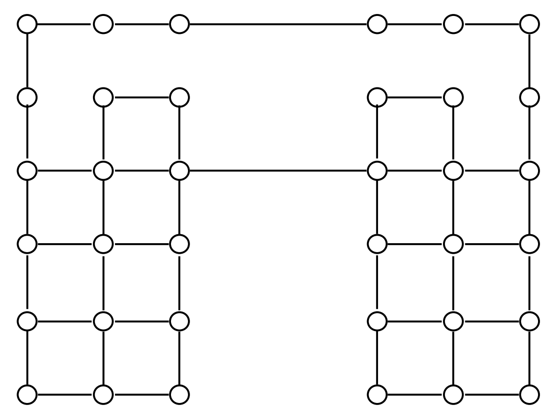
Japanese NTT net



Simple net



6x6 grid net



Experimental Setup: Traffic Characteristics

Traffic patterns are obtained by the combination of spatial and temporal distributions for sessions

- Spatial distributions
  - Uniform (U)
  - Random (R)
  - Hot Spots (HS)
- Temporal distributions
  - Poisson (P)
  - Fixed (F)
  - Temporary (TMP)

Experimental Setup: Experiments Design

- Experiment duration:
  - Each experiment, lasting 1000 sec, is repeated 10 times
  - Before feeding data, routing tables are initialized by a 500 sec phase
- Experiment typology:
  - Study of algorithms behavior for increasing network load
  - Study of algorithms behavior for transient saturation
  - Evaluation of influence of control packet traffic on total traffic

Competing Algorithms

AntNet was compared with:

- OSPF (Open Shortest Path First, current official Internet routing algorithm)
- SPF (Shortest Path first)
- ABF (Adaptive Bellman-Ford)
- Q-routing (asynchronous on-line BF)
- PQ-R (Predictive Q-routing)
- Daemon: approximation of an ideal algorithm

  It knows at each instant the status of all queues and applies shortest path at each packet hop

Measures of Performance

Standard measures of performance are

- Throughput (bits/sec): quantity of service
- Average packet delay (sec): quality of service

Good routing:

- Under high load: increase throughput for same average delay
- Under low load: decrease average delay per packet

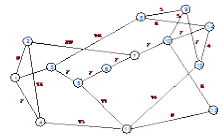
How to Read Results

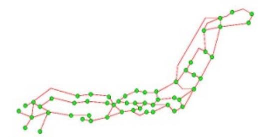
- Routing is a multi-objective problem (maximizing throughput and minimizing delay)
- Max throughput is the main criterion:

  non max throughput means
  - retransmissions,
  - error notification
  - augmented congestion
- Average packet delay has inherent very high variance

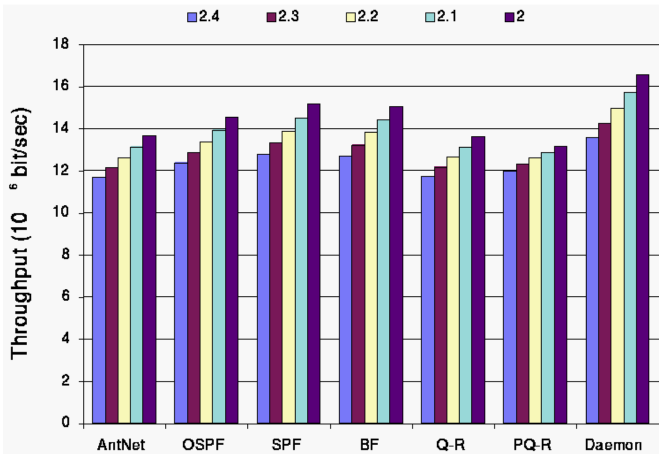
NSFNET & NTTnet (increasing UP traffic)

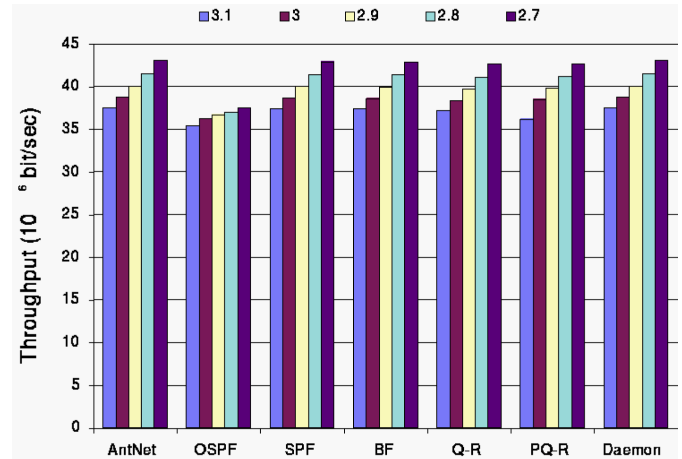
From Di Caro and Dorigo, 1998,
Journal of Artificial Intelligence Research

NSF net 

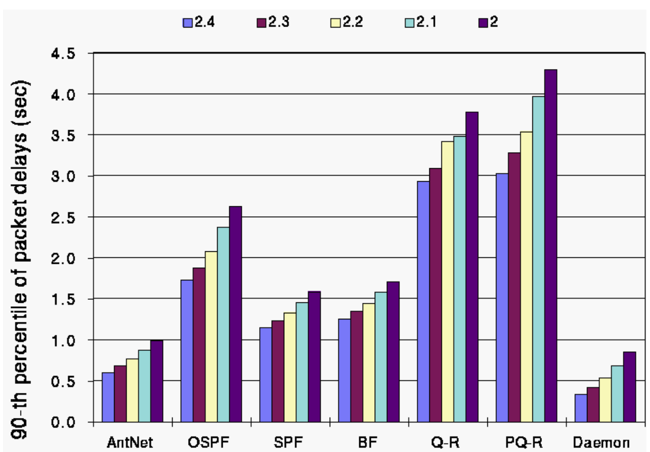
NTT net 

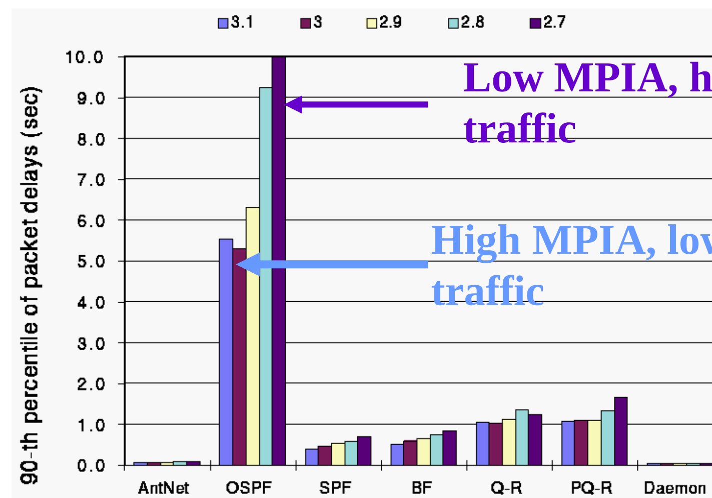
Throughput (b/s)





Avg packet delay





Low MPIA, higher traffic

High MPIA, lower traffic

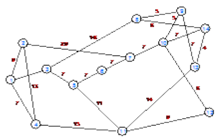
Increasing Uniform-Poisson (UP) traffic
UP traffic increased by reducing the Mean Session Inter Arrival (MPIA) time [s]

NSFNET & NTTnet (UP plus transient HS)

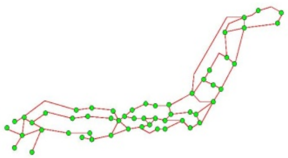
Throughput (b/s)

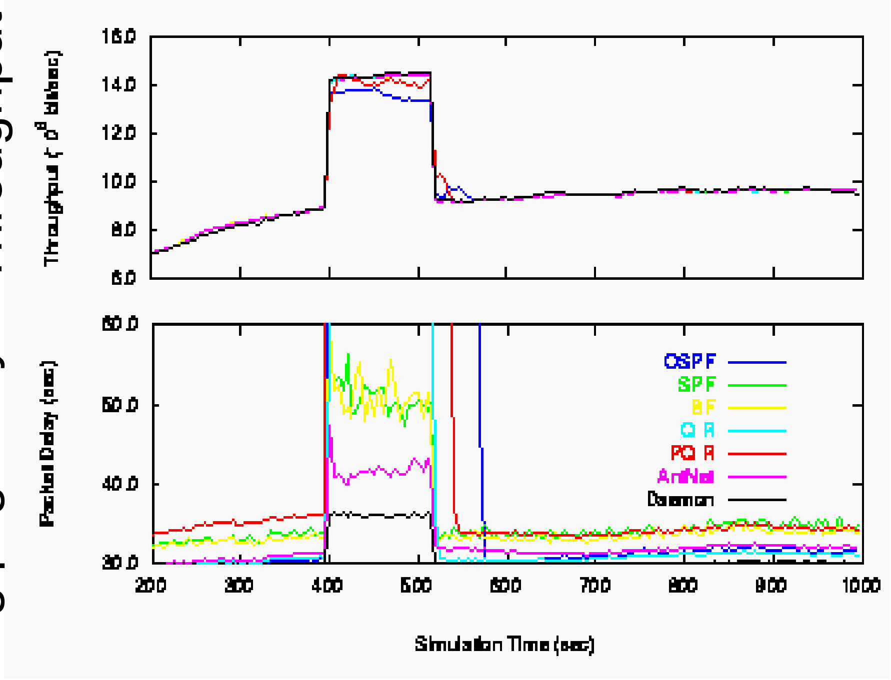
Avg pkg delay

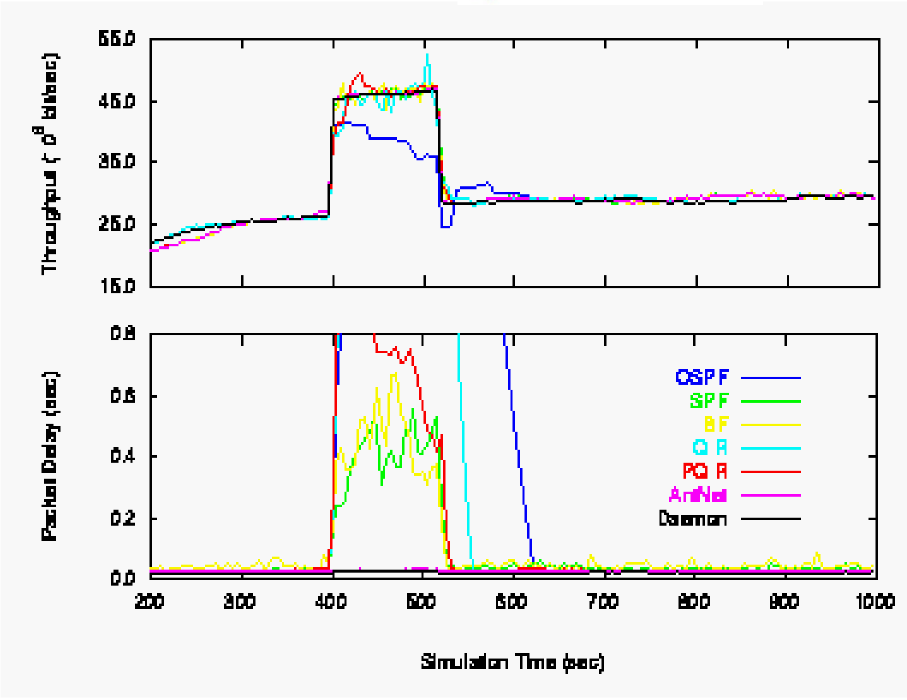
NSF net



NTT net







Data averaged over a 5 seconds sliding window

From Di Caro and Dorigo, 1998

Routing Overhead

From Di Caro and Dorigo, 1998,
Journal of Artificial Intelligence Research

	AntNet	OSPF	SPF	BF	Q-R	PQ-R	Daemon
SimpleNet	0.33	0.01	0.10	0.07	1.49	2.01	0.00
NSFNET-UP	2.39	0.15	0.86	1.17	6.96	9.93	0.00
NSFNET-RP	2.60	0.16	1.07	1.17	5.26	7.74	0.00
NSFNET- UP-HS	1.63	0.15	1.14	1.17	7.66	8.46	0.00
NTTnet-UP	2.85	0.14	3.68	1.39	3.72	6.77	0.00
NTTnet- UP-HS	3.81	0.15	4.56	1.39	3.09	4.81	0.00

Ratio (10^{-3}) between bandwidth occupied by the routing packets and the total available network bandwidth

From Di Caro and Dorigo, 1998

ACO Summary

- ACO metaheuristic
- Overall performance in the literature
- ACO theory
- Applications using ACO

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Why Do Ant-Based Systems Work?

Three important components:

- **TIME**: a shorter path <u>receives pheromone quicker</u> (this is often called: "differential length effect"); on-line set-up (e.g., routing): real time; off-line set-up (e.g., TSP): over multiple iterations

- **QUALITY**: a shorter path <u>receives more pheromone</u>

- **COMBINATORICS**: in most real-world problems a shorter path <u>receives pheromone more frequently</u> because it is likely to have a lower number of decision points

What is a Metaheuristic?

- A **metaheuristic** is a set of algorithmic concepts that can be used to define or organize heuristic methods applicable to a wide set of different problems

- Examples of **metaheuristic** include
  - simulated annealing
  - tabu search
  - iterated local search
  - genetic algorithms
  - particle swarm optimization (later in the course)
  - ant colony optimization

The ACO Metaheuristic

Dorigo, Di Caro & Gambardella, 1999

- Ant System and AntNet have been extended so that they can be applied to any shortest path problem on graphs
- The resulting extension is called ***Ant Colony Optimization metaheuristic***

The ACO-Metaheuristics Procedure

procedure ***ACO-metaheuristics()***

while (not-termination-criterion)

schedule sub-procedures

generate-&-manage-ants()

execute-daemon-actions() *{Optional}*

update-pheromones()

end schedule sub-procedures

end while

end procedure

These are problem specific centralized actions; e.g., local search, select the best ant allowed to deposit extra pheromone

ACO: Quality of Results Obtained

SEQUENTIAL ORDERING PROBLEM (SOP)

Best heuristic currently available — *Gambardella-Dorigo*

QUADRATIC ASSIGNMENT PROBLEM (QAP)

Among best heuristic currently available
on "real-world" problems — *Gambardella-Dorigo-Taillard-Stützle*

ROUTING IN CONNECTION-LESS NETWORKS

Among best heuristics currently available — *Di Caro-Dorigo*

VEHICLE ROUTING PROBLEM (VRP)

Among best heuristics currently available
for vehicle routing problems with time windows — *Gambardella et al.*

SHORTEST COMMON SUPERSEQUENCE PROBLEM (SCS)

Among best heuristics currently available — *Middendorf*

TRAVELLING SALESMAN PROBLEM (TSP)

Good results, although not the best — *Gambardella-Dorigo-Stützle*

GRAPH COLOURING PROBLEM (GCP)

Good results, although not the best — *Hertz*

SCHEDULING PROBLEM

Promising preliminary results on the single machine weighted total tardiness problem — *Dorigo-Stützle*

MULTIPLE KNAPSACK PROBLEM (MKP)

Promising preliminary results — *Michalewicz*

ACO: Theoretical results

- Gutjahr (Future Generation Computer Systems, 2000; Information Processing Letters, 2002) and Stützle and Dorigo (IEEE Trans. on Evolutionary Computation, 2002) have proved **convergence with prob 1 to the optimal solution** of different versions of **ACO**
- Meuleau and Dorigo (Artificial Life Journal, 2002) have shown that there are strong relations between **ACO** and stochastic gradient descent in the space of pheromone trails, which converges to a local optima with prob 1
- Birattari et al. (TR, 2000) have shown the tight relationship between **ACO** and reinforcement learning
- Rubinstein (TR, 2000) has shown the tight relationship between **ACO** and Monte Carlo simulation

ACO: Real-World Applications

- **Sequential ordering in a production line**

  (Gambardella, under evaluation at MCM, Ferrari subcontractor, Italy)
- **Routing of gasoline trucks in Canton Ticino**

  (Gambardella, in use by Pina Petroli, Switzerland)
- **Job-shop scheduling**

  (Bonabeau, in use at Unilever, France)
- **Project scheduling**

  (Kouranos, in use at Intracom S.A , Greece)
- **FaxFactory application**

  (Rothkrantz, Delft Universitaet, in use at KPN, Netherlands)
- **Water management problems**

  (Mariano, Mexican Institute of Water Technology, Mexico)
- **Vehicle routing with time windows**

  (Gambardella, AntOptima, Migros Supermarkets, Switzerland; Number 1 Logistic Group, Italy)

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Natural Artificial vs. Ants

Feature	Natural	Artificial
Memory	yes (but not considered in models of mass recruitment)	yes (node list)
Environment map	to be built (but not considered in models of mass recruitment)	yes (given by the problem)
Physical interference	yes	no (agents), yes (packets)
Synchronicity	no	yes (AS, ACS), no (AntNet, ABC)
Centralized control	no	yes (AS, ACS), no (AntNet, ABC)
Anonymousness	yes	no (agent/packet IDs)
Pheromone modulation	perhaps (but no experimental evidence)	yes (based on solution quality)
Operational space	continuous in time and space	discrete in space (graph) and time (iterations, digital systems)

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Additional Literature – Week 2

Book

- M. Dorigo and T. Stuetzle, “Ant Colony Optimization”, MIT Press, 2004.

Papers

- Gambardella L.M, Taillard E., Dorigo M., “Ant colonies for the Quadratic Assignment Problem”, *J. of the Operational Research Society*, 1999, Vol. 50, pp.167-176.
- Schoonderwoerd R., Holland O., Bruten J., and Rothkrantz L., “Ant-Based Load Balancing in Telecommunications Networks”. *Adaptive Behavior*, Vol. 5, pp. 169-207, 1996.
- Di Caro G. and Dorigo M., “AntNet: Distributed Stigmergic Control for Communications Network”. *Journal of Artificial Intelligence Research*, Vol. 9, pp. 317-365, 1998.