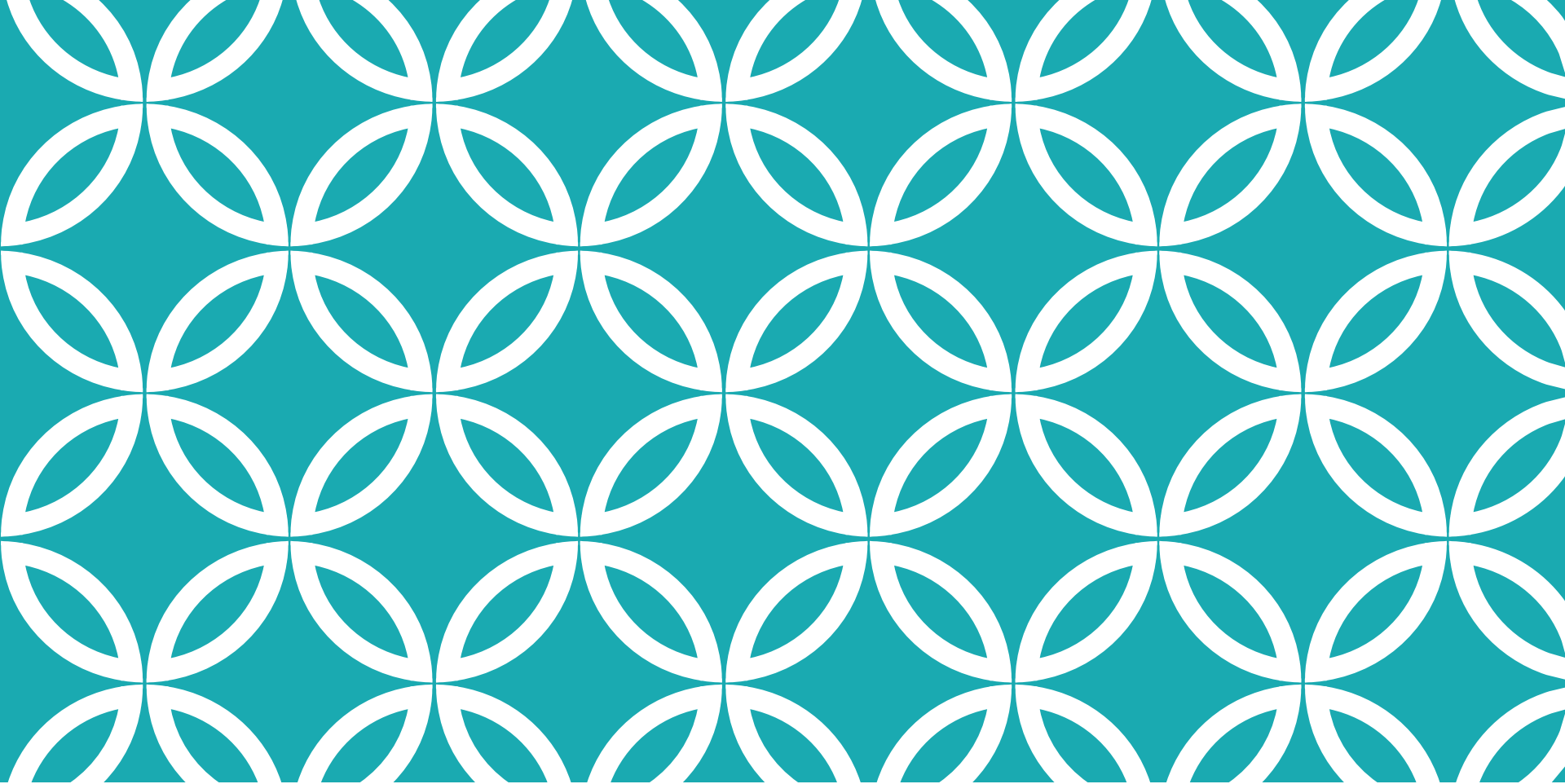


# COOPERATIVE AND ADAPTIVE ALGORITHMS

WEEK 7



# BIOLOGICALLY INSPIRED SEARCH

Ant Colony Optimization  
(ACO)

(Based on material from  
Text and some referenced  
presentations)

# OUTLINE

Introduction,

ACO,

ACO Applications

Adaptation

Cooperation,

References.

# ANTS

There are about        living insects

About 2% of all insects are  $10^{18}$  social

About 50% of all social insects are ants

The total weight of ants is about the total weight of humans

Ant colony size can be as few as 30 ants and as large as million ants


Ants colonize the world since 100,000,000 years, humans only since 50,000 years

# INTRODUCTION

Ants use Stigmergy that was discovered by by Pierre-Paul Grasse [1, 2] who was studying termites.

## Properties of Stigmergy

- indirect agent interaction modification of the environment
- environmental modification serves as external memory
- work can be continued by any individual



Ants walking, to or from, a food source deposit a chemical substance on its way,

This substance is referred to as the *pheromone*.

# INTRODUCTION

Other ants can smell this pheromone,

Its presence will influence the ants' choice of a certain path,

Ants tend to follow strong pheromone concentrations.

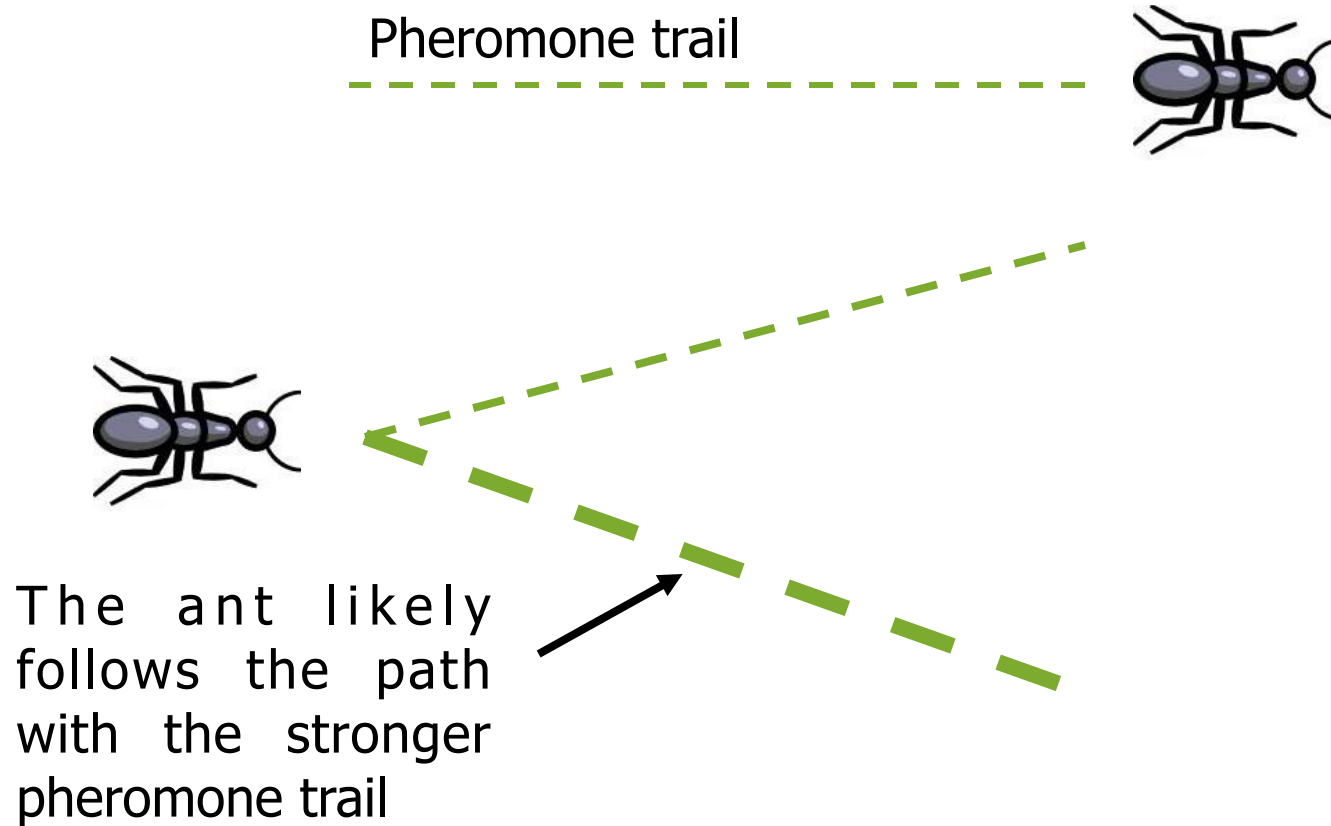
# INTRODUCTION

*Pheromone trails*, is formed by the pheromone deposited on the ground,

Pheromone trails help the ants to reach good food sources that have been previously identified by other ants.



# INTRODUCTION



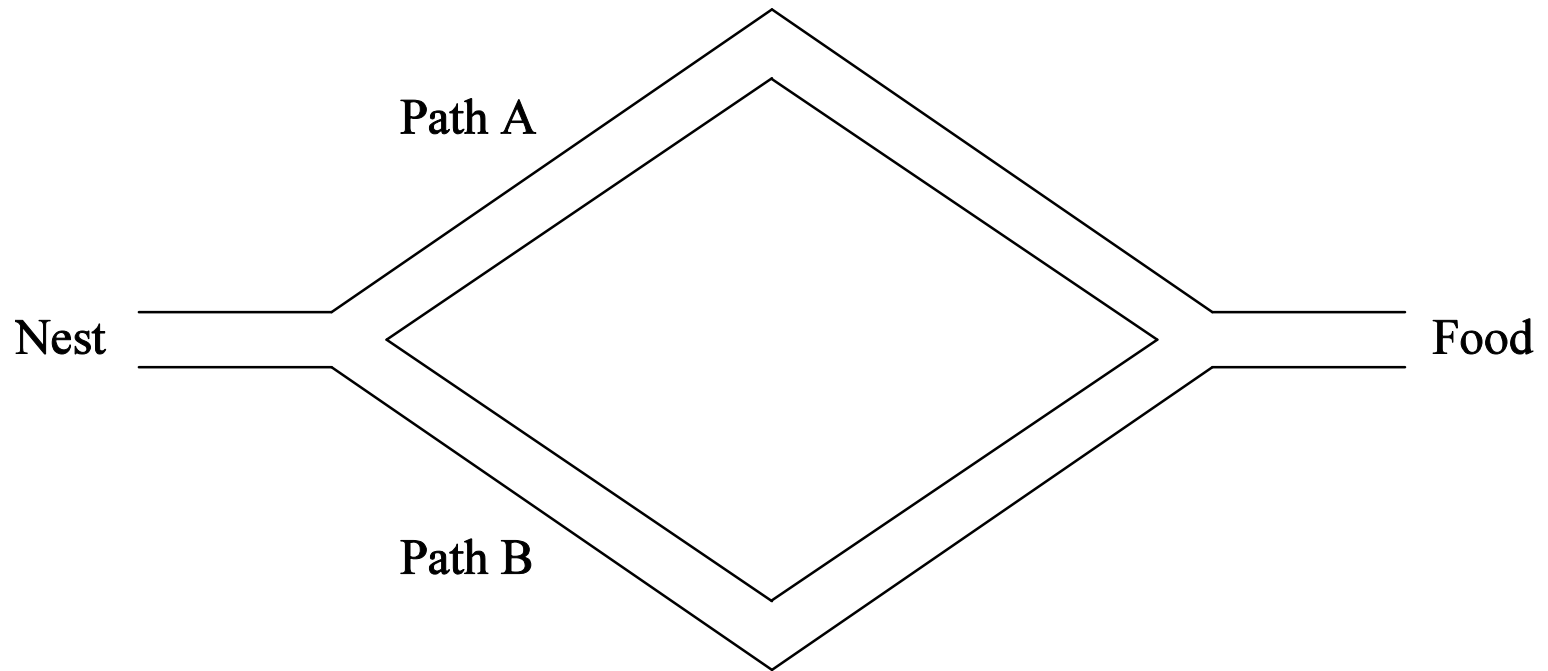
# INTRODUCTION

A *binary bridge experiment* was done by Deneubourg et al. [3],

The ants nest was connected to a food source through two equal length bridges,

The ants behaviour was observed until they converge towards the use of the same bridge.

# EQUAL LENGTH BRIDGES



Courtesy of Prof. A. P. Englebrecht, author of text

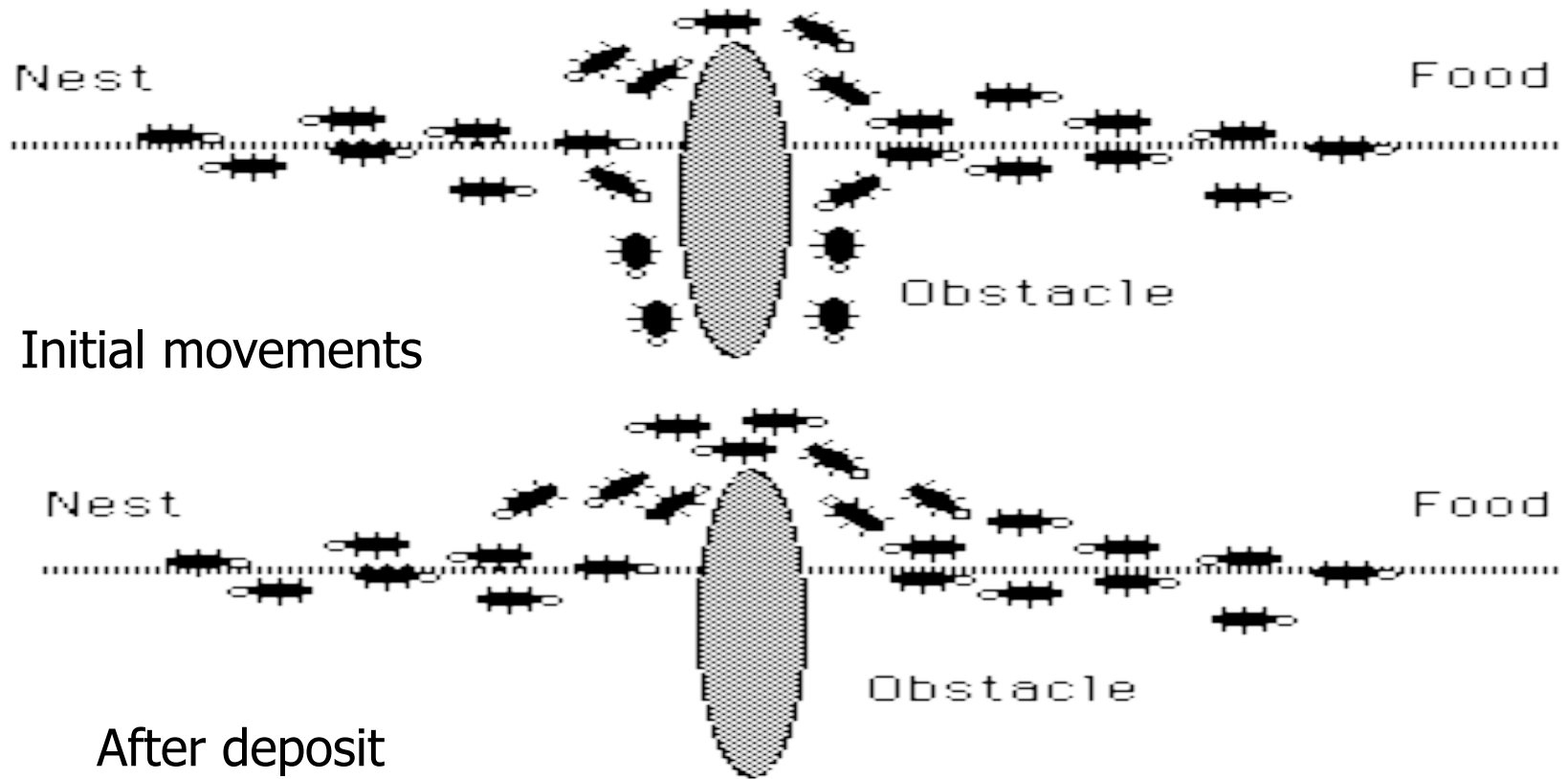
# INTRODUCTION

Initially, the ants randomly choose the bridge to follow,

After a while, the ants will tend to follow the bridge with stronger pheromone trails,

The selection of one bridge is due to random fluctuations causing it to have higher pheromone concentration.

# SHORTEST PATH AROUND AN OBSTACLE



# BRIDGES WITH NON-EQUAL LENGTH

Another experiment was carried by Goss et al. [4] where the bridges were not of equal length,

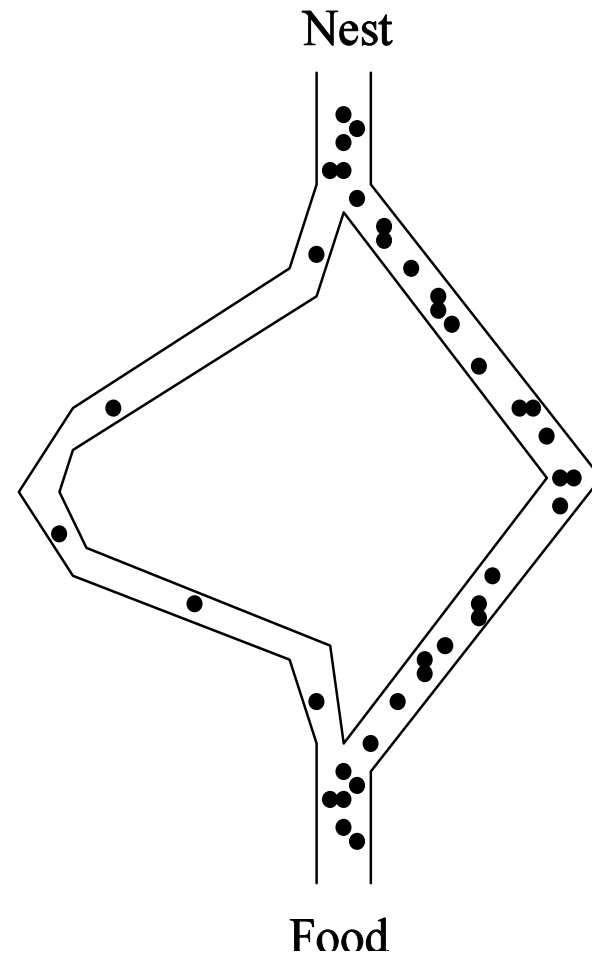
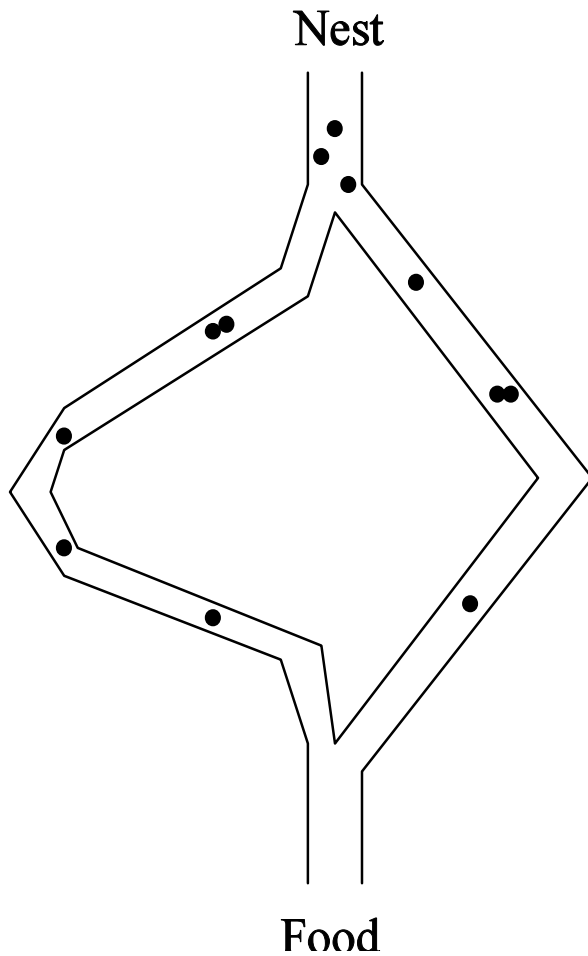
One bridge was significantly shorter than the other,

In this case, the ants following the shorter bridge by chance will be the first to reach the food source.

# INTRODUCTION

They will be also the first to reach the nest as they will take the same path home; since it will have more pheromone,

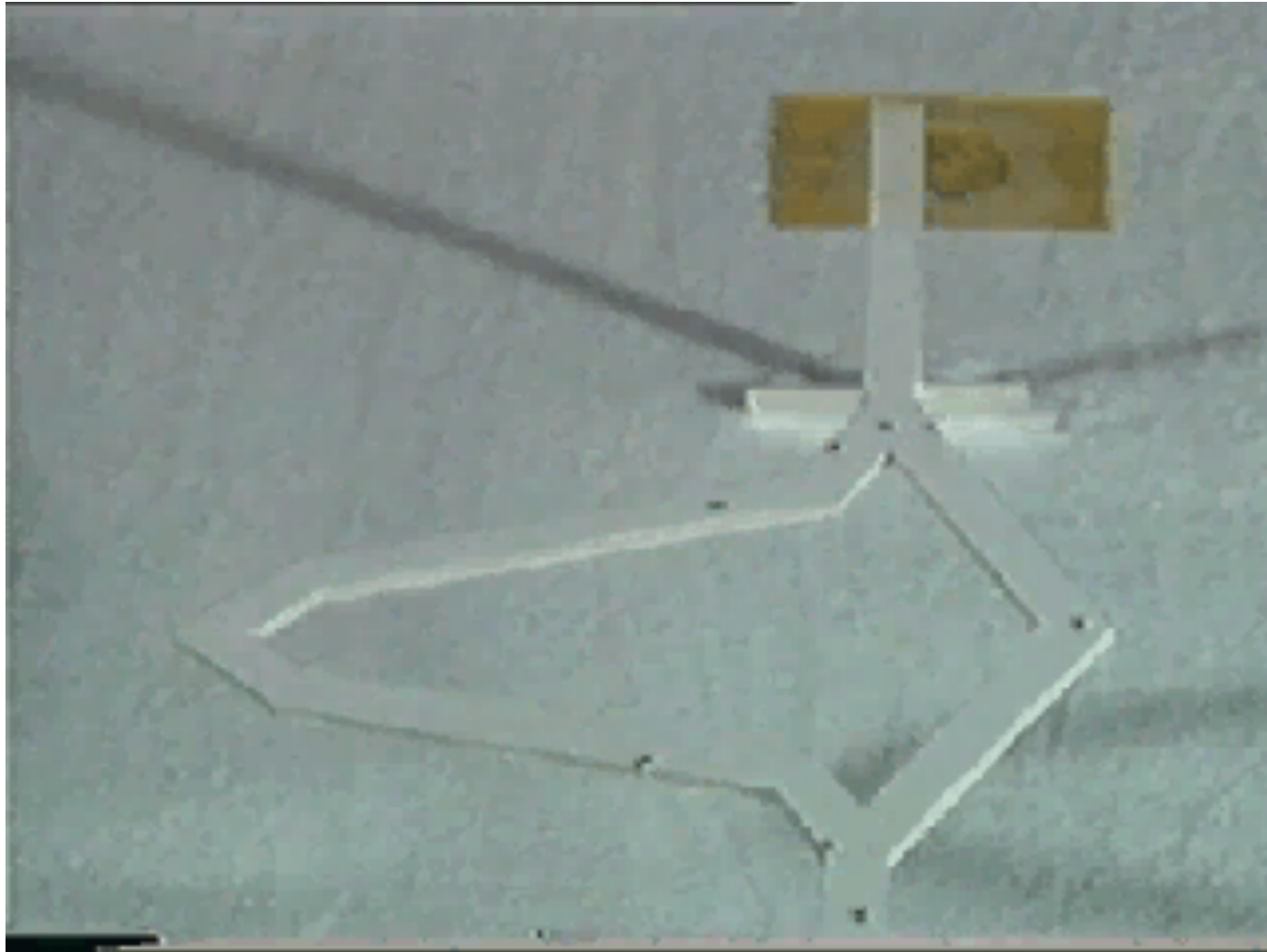
The ants finally converged to the shorter path due to the pheromone depositing mechanism.



Courtesy of Prof. A. P. Englebrecht, author of text



# INTRODUCTION



# ROLE OF PHEROMONE

Pheromone trail acts as a collective memory for the ants to communicate through by sensing and recording their foraging experience

Pheromone evaporates over time introducing some changes in the environment.

# ACO

Introduction,

Simulation,

Algorithm,

Parameters

Variants on ACO

# ACO - INTRODUCTION

Introduced by M. Dorigo in 1992 [5],

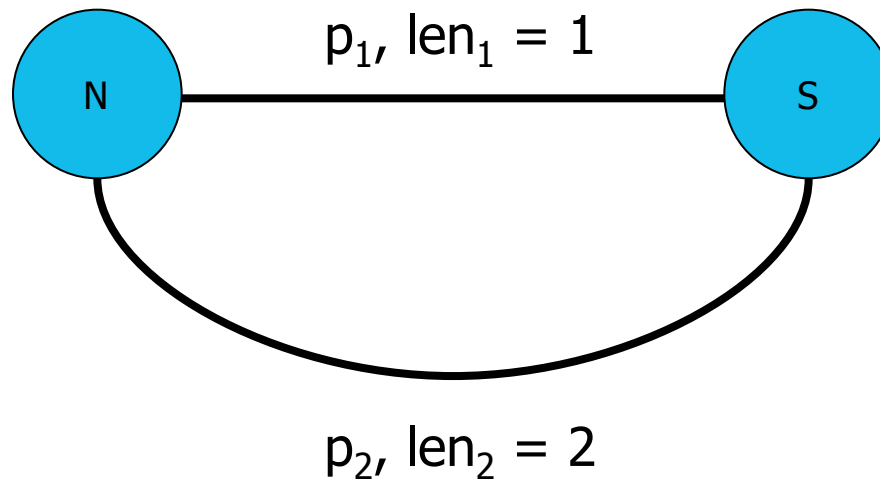
Inspired by the foraging behaviour of real ants,

Successfully applied in many applications including:

- Traveling Salesman Problem (TSP),
- Packet Routing in Network Telecommunications,
- Scheduling Problems,
- Vehicle Routing Problems.

# ACO - SIMULATION

To simulate the behaviour of ants: assume we have a nest and a food source connected through two paths with different lengths.



# ACO - SIMULATION

Assign initial artificial pheromone values for every path,  $\tau_1$  and  $\tau_2$ ,

Initially, both values are equal:

$$\tau_1 = \tau_2 = C > 0$$

# ACO - SIMULATION

Place  $m$  ants at the nest,

For each ant  $k$ ,

- Traverses path 1 with a probability:

$$p_1 = \frac{\tau_1}{\tau_1 + \tau_2}$$

- Traverses path 2 with a probability:

$$p_2 = 1 - p_1$$

# ACO - SIMULATION

An evaporation phase is applied:

- To simulate the evaporation of real pheromone,
- To avoid quick convergence to sub-optimal paths.

$$\tau_i = (1 - \rho) \times \tau_i, \quad \rho \in (0, 1)$$

$\rho$  specifies the rate of evaporation, when equal to 1, the move becomes random.

Each ant leaves more pheromone on its traversed path:

$$\tau_i = \tau_i + \frac{1}{len_i}$$



# REAL AC VRS      ARTIFICIAL AC

Real ant and  
pheromone



Artificial ant (agent) and  
pheromone (value)

Food



Solution

Continuous



Discrete

Pheromone update  
while moving



Pheromone update after  
traverse

Solutions are evaluated  
implicitly



Explicit function to  
evaluate solutions

# ACO - ALGORITHM

Let  $G=(N,E)$  denote a graph, where  $N$  is the set of nodes (vertices) and  $E$  is the set of arcs (edges),

Each arc  $(i, j)$  is associated with a value  $d_{ij}$  denoting the *distance* between nodes  $i$  and  $j$ ,

A simple ant algorithm could be used to find the shortest path between two given nodes in the graph.

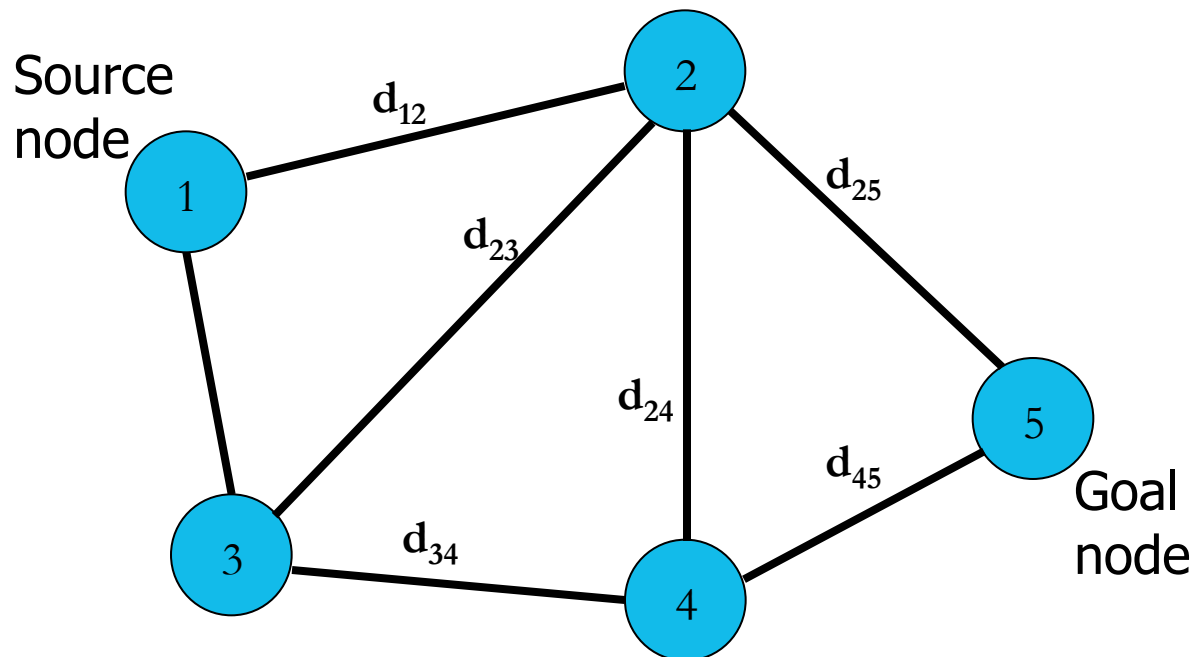
# ACO - ALGORITHM

Let each arc  $(i, j)$  be associated with a value  $\tau_{ij}$  called the *artificial pheromone*,

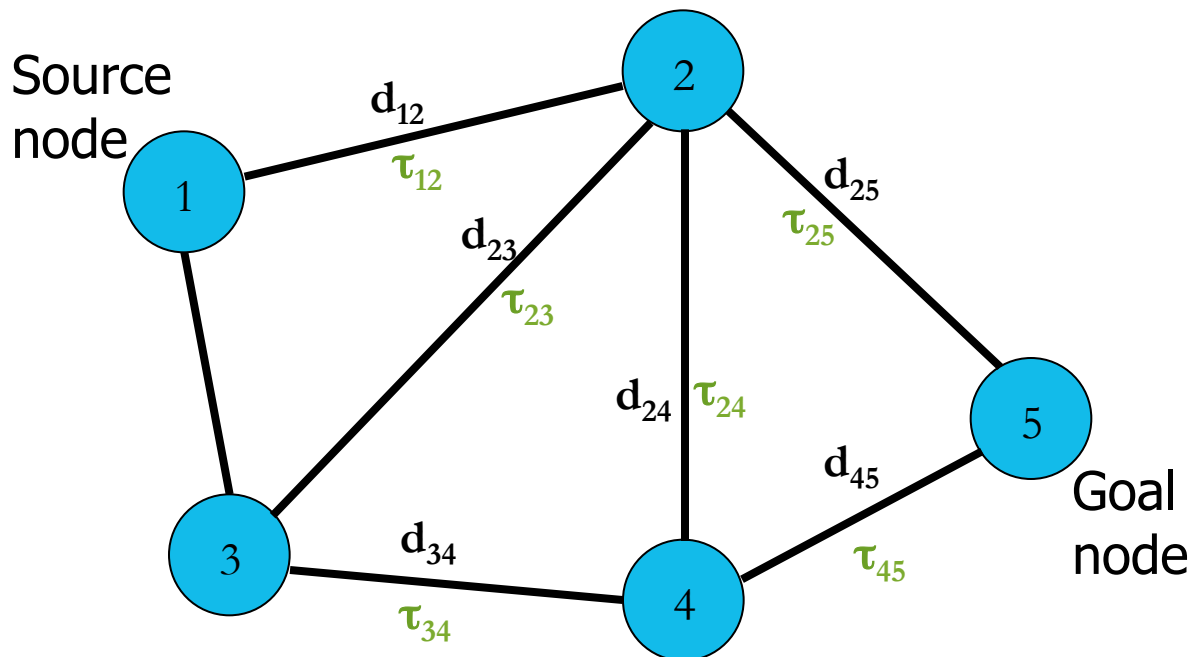
At the beginning, the same small amount of artificial pheromone is placed on all arcs,

Place a group of  $m$  ants at the source node.

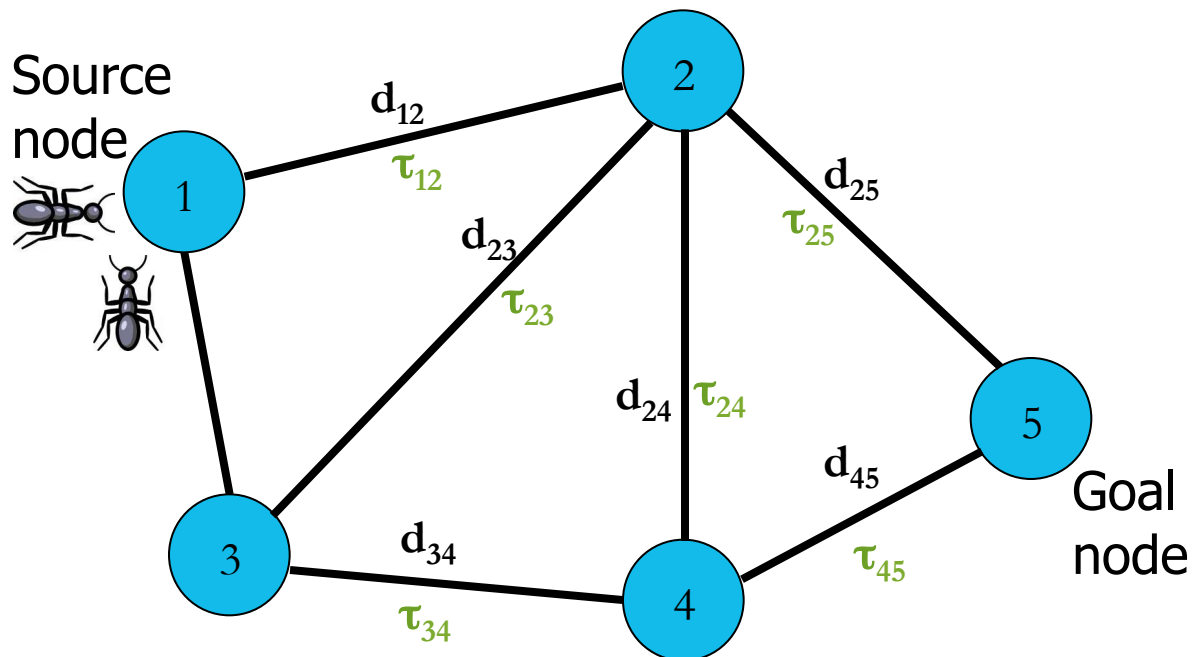
# ACO - ALGORITHM



# ACO - ALGORITHM



# ACO - ALGORITHM



# ACO - ALGORITHM

At each node  $i$ , the ant has a choice to move to any of the  $j$  nodes connected to it,

Each node  $j \in N_i$  connected to  $i$  has a probability to be selected by ant  $k$  equal to:

$$p_{ij}^k = \begin{cases} \frac{\tau_{ij}^\alpha / d_{ij}^\beta}{\sum_{n \in N_i} \tau_{in}^\alpha / d_{in}^\beta} & \text{if } j \in N_i \\ 0 & \text{if } j \notin N_i \end{cases}$$

# ACO - ALGORITHM

$\alpha$  and  $\beta$  are chosen to balance the *local* vs. the *global* search ability of the ant.

Evaporate the artificial pheromone:

$$\tau_i = (1 - \rho) \times \tau_i, \quad \rho \in (0, 1]$$

The ant deposits extra pheromone on the arc it chooses:

$$\tau_{ij} = \tau_{ij} + \Delta\tau$$

This will increase the probability of a subsequent ant choosing the same arc and referred to as *online step\_by\_step* pheromone update.



# ACO - ALGORITHM

Different approaches exist for choosing the value of  $\Delta\tau$  :

- *Ant density model*, adding  $Q$ , a constant value, hence the final pheromone added to the edge will be proportional to the number of ants choosing it. This doesn't take the edge length into account,
- *Ant quantity model*, adding  $Q/d_{ij}$ , taking the edge length into account, hence enforcing the ant local search ability.

# ACO - ALGORITHM

Another approach is the *online delayed* pheromone update, sometimes referred to as the *ant cycle model*,

After the ant builds the solution, it traces the path backward and updates the pheromone trails on the visited arcs according to the solution quality.

# ACO - ALGORITHM

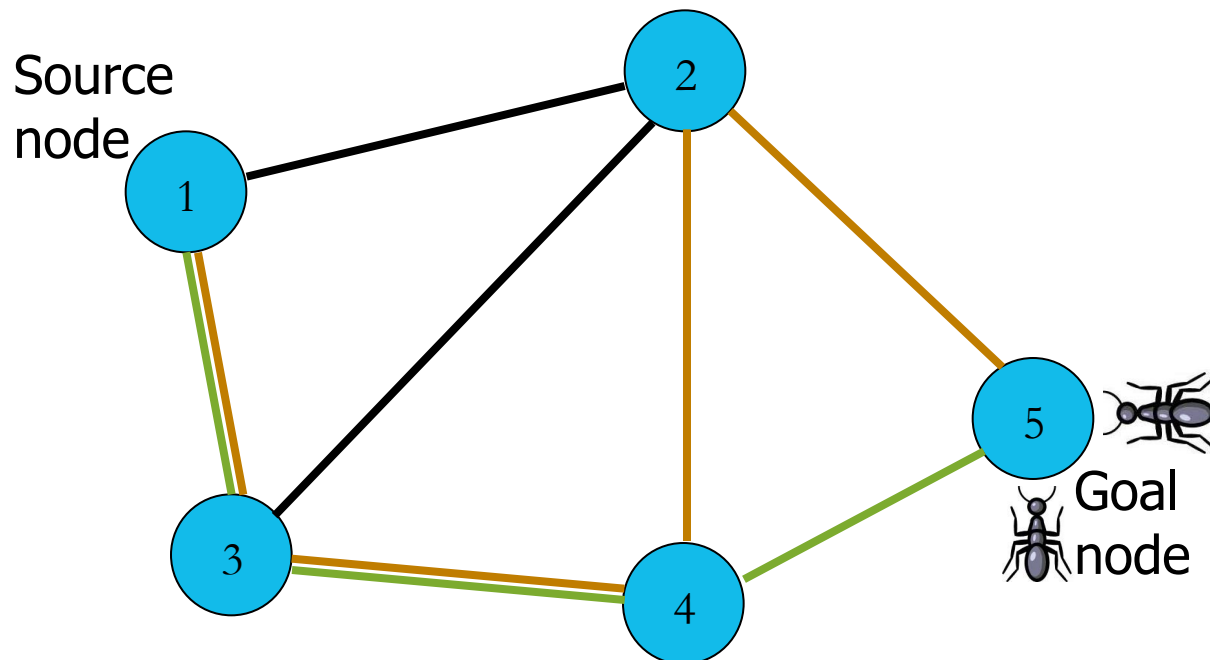
The online delayed pheromone update is done by adding (done after the evaporation phase) :

$$\Delta \tau_{ij} = \frac{Q}{L^k}$$

for every arc  $(i, j)$  on the path, where  $L^k$  is the length of the path found by ant  $k$ .

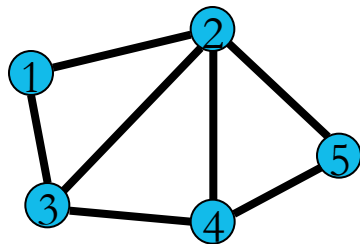
Repeat the process for different initialization of pheromone

# ACO - ALGORITHM

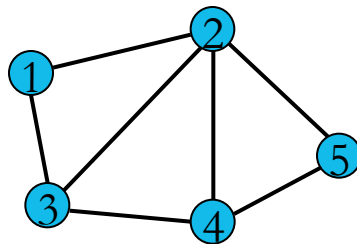


# ACO - ALGORITHM

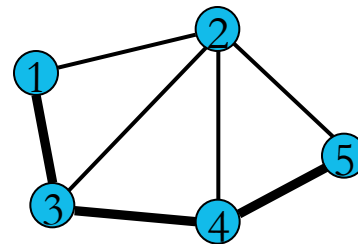
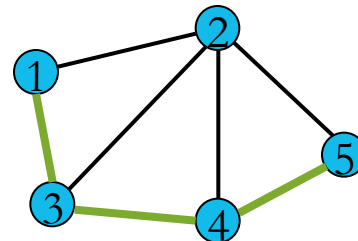
Start



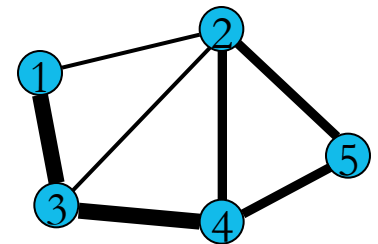
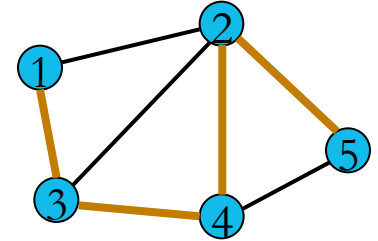
Evaporation



Solution 1



Solution 2



# ACO METAHEURISTIC

Algorithm Ant colony optimization  
metaheuristic

Set parameters, initialize pheromone trails  
while termination conditions not met do

ConstructAntSolutions

ApplyLocalSearch (optional)  
UpdatePheromones

end while

# ACO ALGORITHM

Termination conditions:

- Max number of iterations reached
- Acceptable solution reached
- All ants (or most of them) follow the same path, i.e stagnation.

# PARAMETERS

Number of ants: more ants more computations, but also more exploration.

Max number of iterations: has to be enough to allow convergence.

Initial pheromone: constant, random, max value, small value.

Pheromone decay parameter  $\rho$



# COMPONENTS

Transition rule: probability of selection for the ant

Pheromone evaporation rule

Pheromone update rule

Problem heuristic if used

Quality of solution measure

Memory or list for constraints (Tabu list)

Termination Criteria

# ANT SYSTEM ALGORITHM

*AS* :

- Proposed by Dorigo et al. [6],
- Uses the same basic steps outlined in ACO,
- The online delayed pheromone update is adopted using all the solutions of the current iteration.

$$\Delta \tau_{ij} = \frac{Q}{L^k}$$

# MAX-MIN ANT SYSTEM ALGORITHM

## MMAS:

- Proposed by Stutzle and Hoos [7] to overcome stagnation.
- The update is done using the best solution (best ant) in the current iteration or the best solution over all, also decay in the update of the pheromone

$$\tau_{ij}(t+1) = (1 - \rho)\tau_{ij}(t) + \rho\Delta\tau_{ij}^{best}(t)$$

# MMAS

- The values of the pheromone are restricted between  $\tau_{min}$  and  $\tau_{max}$ , allows high exploration in the beginning ( $\tau_{max}$ ) and more intensification later.
- The values of  $\tau_{min}$  and  $\tau_{max}$  are chosen experimentally, although they could be calculated analytically if the optimal solution is known,
- Improved the performance significantly over AS.

# ANT COLONY SYSTEM ALGORITHM

## ACS:

- Proposed by Gambardella and Dorigo [8],
- Based on AS but uses a different transition rule based on Elitist strategy
- If  $q < q_0$  ( $q$  is a random number,  $q_0$  is in  $(0,1)$ )

$$j = \arg \max_{j \in N_i^k} \{ \tau_{ij}(t) \eta_{ij}^\beta \}$$

# ACS

- Else use:

$$p_{ij}^k(t) = \begin{cases} \frac{[\tau_{ij}(t)]^\alpha [\eta_{ij}]^\beta}{\sum_{k \in \text{allowed}_k} [\tau_{ik}(t)]^\alpha [\eta_{ik}]^\beta} & \text{if } j \in \text{allowed}_k \\ 0 & \text{otherwise} \end{cases}$$

This creates bias towards choices of better quality,  $q_0$  is a user specified parameter to control this bias.

# ACS

- Uses the offline pheromone update, where all the ants update their last traversed edge only,

$$\tau_{ij} = (1 - \rho_2)\tau_{ij} + \rho_2\tau_0$$

- Introduced a new pheromone update rule (using the best solution only).

$$\tau_{ij}(t + 1) = (1 - \rho_1)\tau_{ij}(t) + \rho_1\Delta\tau_{ij}^{best}(t)$$

Best here can be iteration best or global best

# OUTLINE

- Introduction,
- ACO,
- ACO Applications,
- Adaptation,
- Cooperation,
- References.



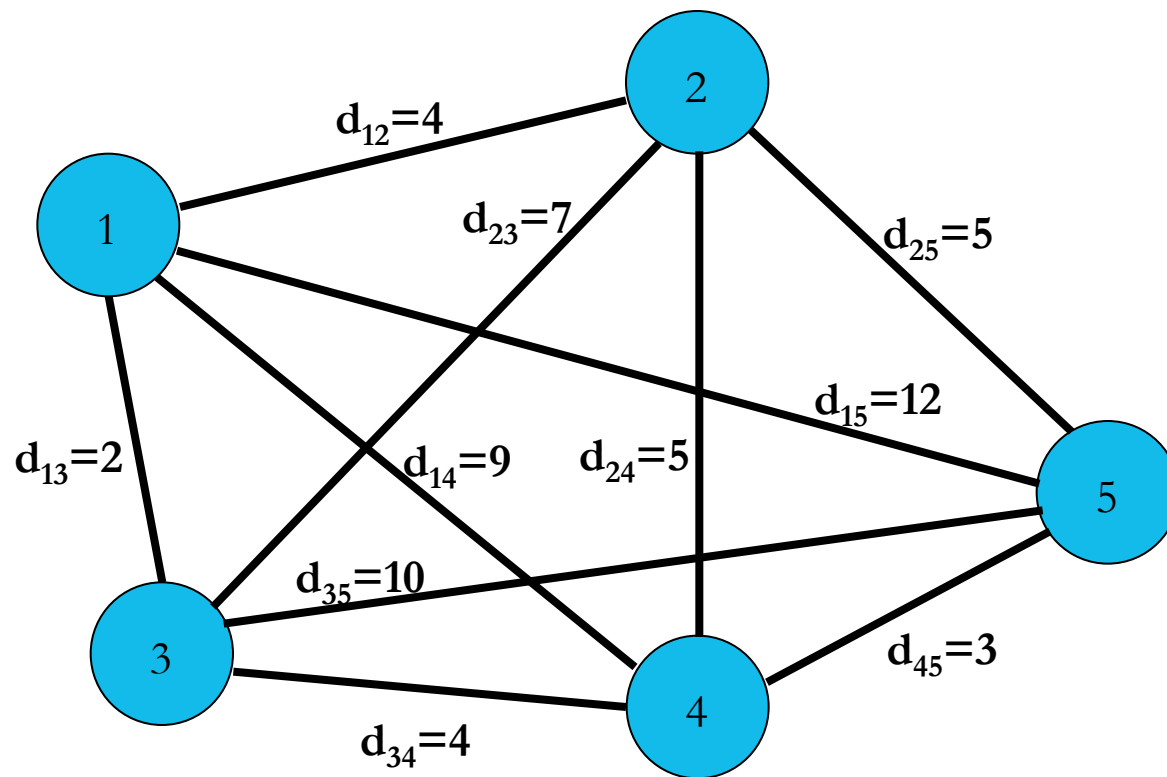
# ACO APPLICATIONS

- Travelling Salesman Problem (TSP),
- Assembly Line Balancing (ALB),
- Cell Assignment in PCS Networks (CA).
- Other

# TSP

- Initialize the pheromone trails,
- While termination condition not met
  - ConstructAntSolutions,
  - ApplyLocalSearch {Optional},
  - UpdatePheromones,
- end

# TSP



# TSP

- A small artificial pheromone value is placed on all the edges,
- $M$  ants are placed on the graph divided among all the nodes,
- $M$  (number of ants) roundtrips are created in  $n$  (number of cities) steps.

# TSP

- The ConstructAntSolutions step:
  - For each ant:
    - A new node is randomly chosen according to the selection probability,
    - This node is added to the ants' *tabu list* (so as not to be re-selected because every city is visited once).
- After the completion of the roundtrips, the tabu lists are emptied,

# TSP

- Usually, the online step-by-step update rule is not adopted,
- The ApplyLocalSearch step:
  - These are optional actions referred to as *daemon actions*,
  - One action is to apply a local search method in order to improve the constructed solution,
  - Another action is to add extra pheromone (*delayed update*) based on the collection of some global information.

# TSP

- The UpdatePheromones step:
  - Apply pheromone evaporation,
  - Update the pheromone trails using a chosen set of good solutions, different approaches use (*Ant cycle model*):
    - All the solutions found in the current iteration,
    - The best solution found in the current iteration,
    - The best solution found so far,

# TSP

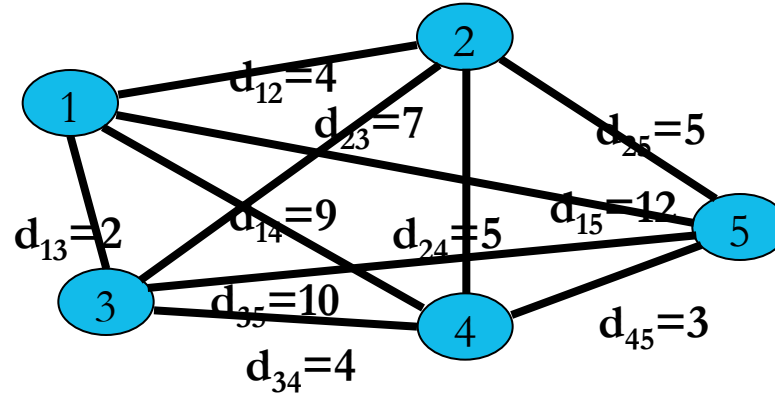
- A simple iteration:
  - Assume that an ant  $k$  was placed at node 1,
  - The neighbours of node 1 are :

$$N_1 = \{2, 3, 4, 5\}$$

- Initially, all the edges have the same pheromone value (assume  $\tau = 1$ ).



# TSP



- Assuming  $\alpha$  and  $\beta$  are both equal to 1,
- Then:

$$\sum_{n \in N_1} \tau_{1n}^{\alpha} / d_{1n}^{\beta} = \frac{1}{4} + \frac{1}{2} + \frac{1}{9} + \frac{1}{12} \cong 0.95$$

$$p_{12}^k = \frac{0.25}{0.95} \cong 0.26, p_{13}^k = \frac{0.5}{0.95} \cong 0.53$$

$$p_{14}^k = \frac{0.11}{0.95} \cong 0.12, p_{15}^k = \frac{0.08}{0.95} \cong 0.08$$

# TSP

- Node 3 has the highest probability of being selected,
- If node 3 gets selected, ant  $k$  moves to that node and continues with the next iteration where:

$$N_3 = \{2, 4, 5\}$$

# TSP

■ Hence:

$$\sum_{n \in N_3} \tau_{1n}^{\alpha} / d_{1n}^{\beta} = \frac{1}{7} + \frac{1}{4} + \frac{1}{10} \cong 0.49$$

$$p_{32}^k = \frac{0.14}{0.49} \cong 0.29, p_{34}^k = \frac{0.25}{0.49} \cong 0.51$$

$$p_{35}^k = \frac{0.1}{0.49} \cong 0.20$$

■ Another node is selected, and so on ...

# TSP

- If ant  $k$  completes the tour:

$$\textit{Tour} = \{1, 3, 2, 4, 5\}$$

$$\textit{Cost} = 29$$

- First, all the pheromone trails get evaporated,

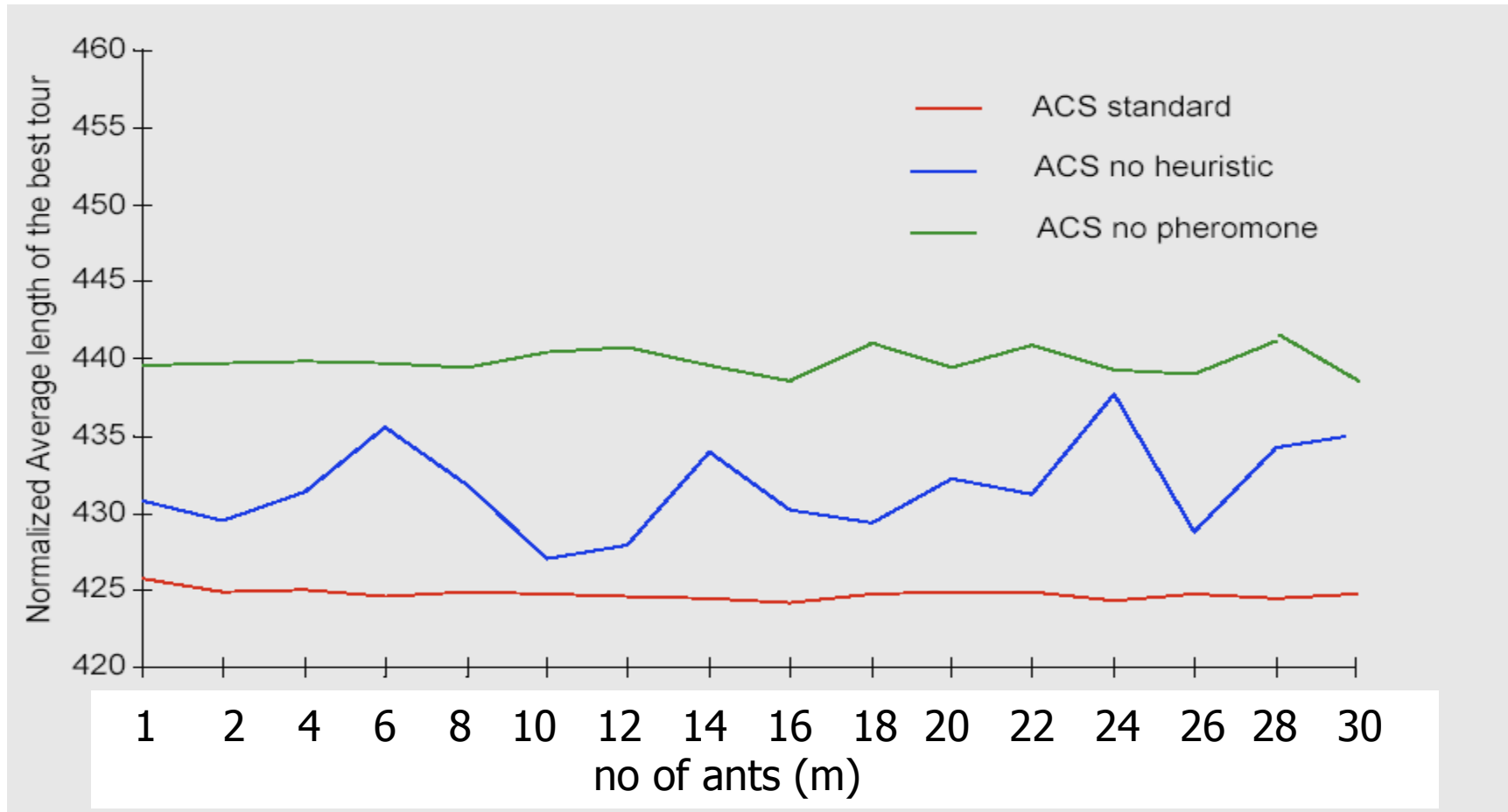
# TSP

- And then, if this ant is selected to update the pheromone trails, it enforces the edges

$\{1, 3\}, \{3, 2\}, \{2, 4\},$   
 $\{4, 5\}, \{5, 1\}$

with the value  $Q/29$ .

# The importance of the pheromone and the heuristic function [8]



Comparison between ACS standard, ACS with no heuristic, and ACS in which ants neither sense nor deposit pheromone. Problem: Oliver30. Averaged over 30 trials, 10,000/ $m$  iterations per trial.

# THE IMPORTANCE OF THE PHEROMONE AND THE HEURISTIC FUNCTION [8]

The results show that not using pheromone deteriorates the performance,

ACS without heuristics performs better than ACS without pheromone. This may be due that ACS with pheromone and no heuristics is still guided by the global update rule (reflecting the importance of the solution).

ACS without pheromone reduces to a stochastic multi-greedy algorithm

ACS with both is better confirming the role of cooperation.

# FROM[7]

Number of ants = n

Number of iterations = 10000

Benchmark	Optimal	AS	MMAS	ACS
eil51	<b>426</b>	437.3	<b>427.6</b>	428.1
kroa100	<b>21282</b>	22471.4	<b>21320.3</b>	21420.0
d198	<b>15780</b>	16702.1	<b>15972.5</b>	16054.0



# COMPARISON RESULTS FROM [16]

Comparison of ACS with the genetic algorithm (GA), evolutionary programming (EP), simulated annealing (SA), and the annealing-genetic algorithm (AG), a combination of genetic algorithm and simulated annealing.

The best integer tour length, the best real tour length (in parentheses) and the number of tours required to find the best integer tour length (in square brackets).

The best result for each problem is in boldface.

# COMPARISON RESULTS FROM [16]

Problem name	ACS	GA	EP	SA	AG	Optimum
Oliver30	<b>420</b>	421	<b>420</b>	424	<b>420</b>	<b>420</b>
(30-city problem)	(423.74)	(N/A)	(423.74)	(N/A)	(N/A)	(423.74)
	[830]	[3,200]	[40,000]	[24,617]	[12,620]	
Eil50	<b>425</b>	428	426	443	436	<b>425</b>
(50-city problem)	(427.96)	(N/A)	(427.86)	(N/A)	(N/A)	(N/A)
	[1,830]	[25,000]	[100,000]	[68,512]	[28,111]	
Eil75	<b>535</b>	545	<b>542</b>	580	561	<b>535</b>
(75-city problem)	(542.31)	(N/A)	(549.18)	(N/A)	(N/A)	(N/A)
	[3,480]	[80,000]	[325,000]	[173,250]	[95,506]	
KroA100	<b>21,282</b>	21,761	N/A	N/A	N/A	<b>21,282</b>
(100-city problem)	(21,285.44)	(N/A)	(N/A)	(N/A)	(N/A)	(N/A)
	[4,820]	[103,000]	[N/A]	[N/A]	[N/A]	

It is clear that ACS and EP greatly outperform GA, SA, and AG.  
N/A means not available in the literature

# ACS ON LARGER TSP PROBLEMS

ACS performance for some bigger geometric problems (over 15 trials). We report the integer length of the shortest tour found, the number of tours required to find it, the average integer length, the standard deviation, the optimal solution (for fl1577 we give, in square brackets, the known lower and upper bounds, given that the optimal solution is not known), and the relative error of ACS.

Problem name	ACS best integer length (1)	ACS number of tours generated to best	ACS average integer length	Standard deviation	Optimum (2)	Relative error $\frac{(1)-(2)}{(2)} * 100$
d198 (198-city problem)	15,888	585,000	16,054	71	15,780	0.68 %
pcb442 (442-city problem)	51,268	595,000	51,690	188	50,779	0.96 %
att532 (532-city problem)	28,147	830,658	28,523	275	27,686	1.67 %
rat783 (783-city problem)	9,015	991,276	9,066	28	8,806	2.37 %
fl1577 (1577-city problem)	22,977	942,000	23,163	116	[22,204 – 22,249]	3.27+3.48 %

# OUR OWN COMPARISON

AS and SA are performing the same number of FEVs ( $1000 \cdot n$ )

TS is performing (45500, 115500, 564000, 676000)

Benchmark	Optimal	TS(with AC)	SA (1000,0.01)		AS #ants=n #iterations=1000
burma14	30.8785	30.8785	31.1463	200	30.8785
ulysses22	75.6651	75.9832	77.5532	300	77.3018
att48	33524	38446	41604	700	35224
eil51	426	457.66	513.67	700	460.71 (437-510000)

# ACO APPLICATIONS

- Travelling Salesman Problem (TSP),
- Assembly Line Balancing (ALB),
- Cell Assignment in PCS Networks (CA).
- Timetabling Problems

# ALB

An assembly line is made of a  $m$  workstations arranged in series and in parallel.

The production item consists of a set  $V$  of  $n$  tasks with precedence relations that need to be assigned to the workstations.

Each task  $j$  requires processing time  $t_j > 0$  and space  $a_j$

Each station  $k$  has a cycle time  $c_k$ , to carryout the subset of  $V$ ,  $S_k$  of tasks assigned to it (workload) and space area  $A_k$ .

Usually the cycle time is the same for all workstation  $c$ , which determines the production rate  $r=1/c$ . The cycle time represents an upper limit of the total time required by the tasks assigned to the workstation.

$I_k = c - t(S_k)$  is the idle time of workstation  $k$ . Total idle time is sum over all stations

# ALB

■ The main objective is to assign the tasks to workstations to satisfy one or more of the following:

1. Minimize the number of workstations given a fixed value for the cycle time  $c$  and space  $A$
2. Minimize the cycle time  $c$  given a fixed number of workstations  $m$  and  $A$
3. Minimize total space required given  $c$  and  $m$
4. Give  $m$ ,  $c$  and  $A$  find a feasible solution (assignment)

Multi-objectives:

1 & 2 given  $A$

1 & 3 given  $c$

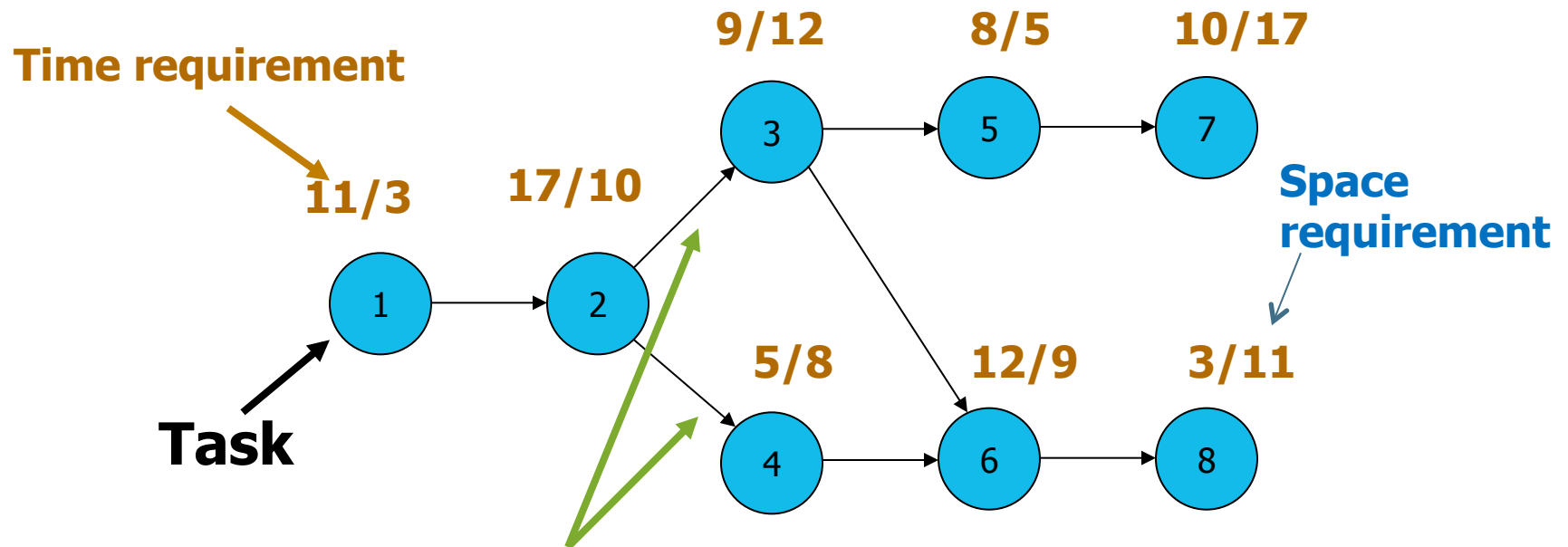
2 & 3 given  $m$

1 & 2 & 3

A simplified version of the above is not to consider the space requirement

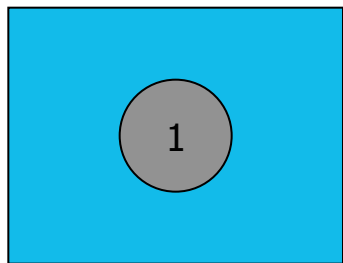
# ALB

Example [9],  $C=20$ ,  $A=20$

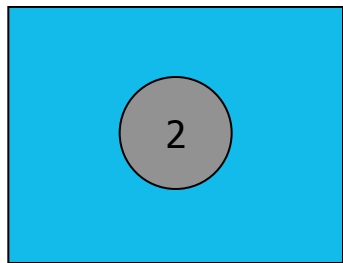




ALB  $C=20$ ,  $A=20$

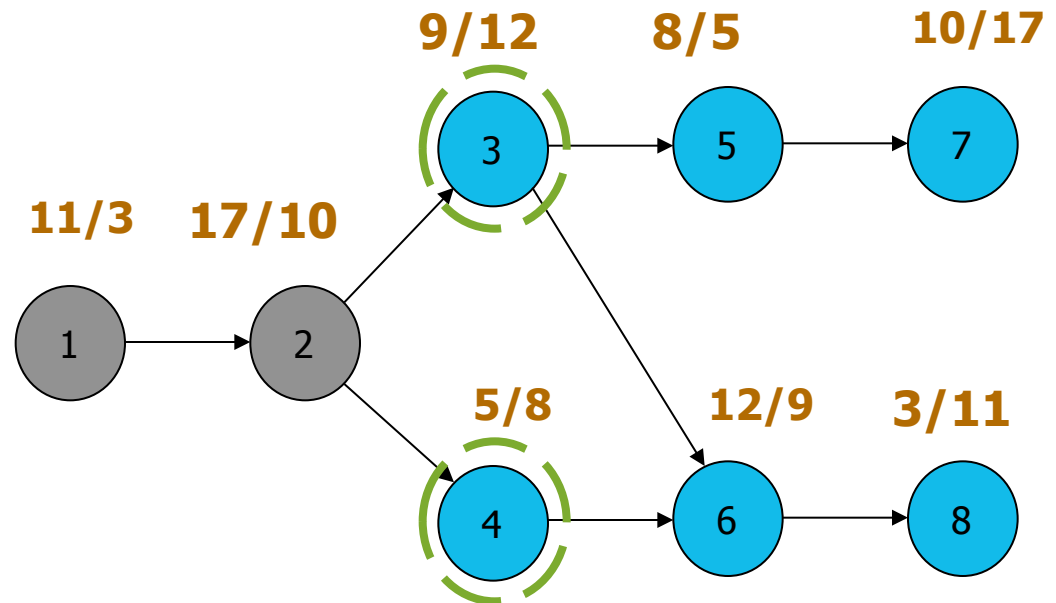


**w1**



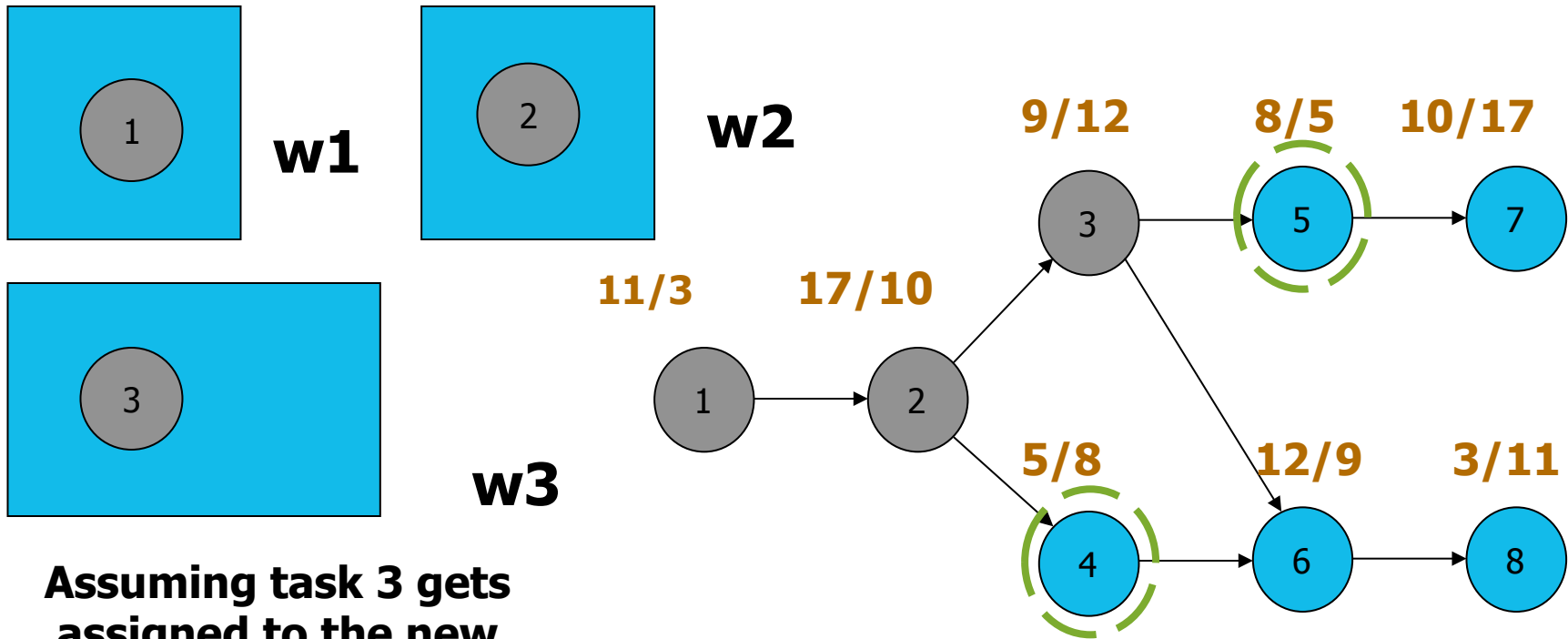
**w2**

**Tasks 1 and 2 get  
assigned to two  
workstations**



**Next tasks to  
assign** **Neither can be  
assigned to w2 or w1**

# ALB



Assuming task 3 gets assigned to the new workstation w3

4 or 5 could be assigned to w3, greedy selects 5

Next tasks to assign

# ALB

Continuing this way (without checking space constraint) we get the following solution

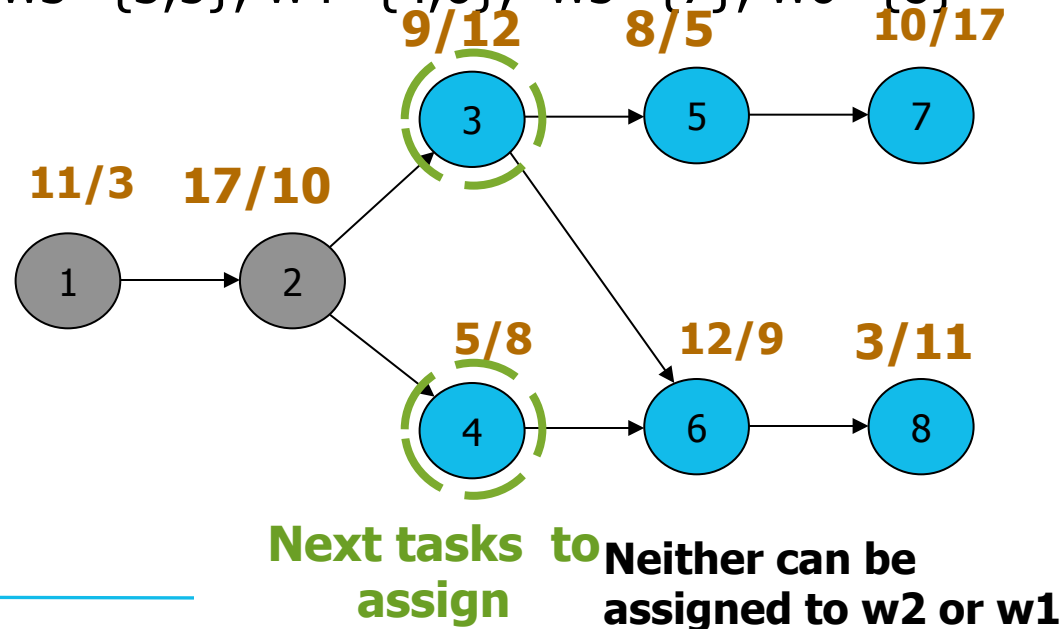
$$w1=\{1\}, w2=\{2\}, w3=\{3,5\}, w4=\{4,6,8\}, w5=\{7\}, m=5$$

If we check space  $w4$  doesn't satisfy the space limit.

A solution to satisfy space will be

$$w1=\{1\}, w2=\{2\}, w3=\{3,5\}, w4=\{4,6\}, w5=\{7\}, w6=\{8\}$$

with  $m=6$



# ALB-ACO

- How to apply ACO to solve the problem:
  - In TSP, each ant determines which city to add next to the solution,
  - In this problem, each ant determines which assignment (task/workstation) to add next to the solution,
  - Each different assignment gets a pheromone trail parameter. A pheromone trail is associated with each assignment,  $\tau_{jk}$  denotes the pheromone trail associated with task  $j$  getting assigned to workstation  $k$ ,
  - Let  $T$  be the set of tasks that:
    - Are not yet assigned to a work station,
    - Whose predecessors are all assigned to work stations,

# ALB- ACO

- Select max no of iterations, itmax
- Select number of Ants, antmax
- While number of iterations < itmax
  - While current ant number < antmax
    - Initialize ant parameters (pheromone, update,..etc)
    - While ( $\tau$  is not empty)
      - Select task  $j$  to be assigned to workstation  $k$  using transition rule.
      - If none of the tasks can be fit in the current workstation due to time or space capacity)  
Open a new workstation,
    - Update problem data
  - End ( $\tau$  is empty):
    - Update Pheromone trail using update rules and evaporation rule
    - Evaluate Solution
  - Retain best solution of all ants
  - Update pheromone trail based on best solution (if desired)
- Return best solution

# ALB-ACO

## Transition rules

- Simple ratio

$$p_{jk} = \begin{cases} \frac{\tau_{jk}^{\alpha}}{\sum_{n \in T} \tau_{nk}^{\alpha}} & \text{if } j \in T \\ 0 & \text{if } j \notin T \end{cases}$$

- Using heuristics

$$p_{jk} = \begin{cases} \frac{[\tau_{jk}]^{\alpha} [\eta_j]^{\beta}}{\sum_{l \in T} [\tau_{lk}(t)]^{\alpha} [\eta_j]^{\beta}} & \text{if } j \in T \\ 0 & \text{otherwise} \end{cases}$$

# ALB-ACO

■ Heuristic can be related to utilization

$$\eta_j = (1/c) \left( \sum_{l \in k} t_l + t_j \right)$$

Or if space is checked

$$\eta_j = (1/c) \left( \sum_{l \in k} t_l + t_j \right) + (1/A) \left( \sum_{l \in k} a_l + a_j \right)$$

Pheromone update and evaporation can be simple

$$\tau_{jk} = (1 - \rho) \tau_{jk} + \Delta \tau_{jk}$$

Where  $\Delta \tau_{jk}$  can be Density, quantity or online delayed using utilization as solution quality

# ALB-ACO

Can use pheromone update based on best solution

$$\tau_{jk} = (1 - \rho)\tau_{jk} + \rho\Delta\tau_{jk}^{best}$$





# CHARACTERISTICS OF PROBLEMS

Combinatorial Optimization problems.

Not feasible to solve using classical optimization methods if the problem size is large.

Mainly discrete optimization problems

There some variants of ACO to handle continuous optimization problems.

# ADVANTAGES AND DISADVANTAGES OF ACO

## Advantages:

- ACO is stochastic, population based method similar to GA
- It retains memory of the entire colony instead of just previous generation (as in GA)
- Less affected by poor initial solutions due to combinations of random path selection.
- Proved successful in many application
- Can handle dynamic environments

# ADVANTAGES AND DISADVANTAGES OF ACO

## Disadvantages:

- Mainly experimental, theoretical analysis has been very limited. Some proofs of convergence for some methods under certain conditions exist (MMAS, AS)
- It uses a lot of parameters, selection of values is experimental.
- It may take long time to converge.

# OUTLINE

Introduction,  
ACO,  
ACO Applications,  
**Adaptation**,  
Cooperation,  
References.

# ADAPTATION

Several approaches have been proposed to adapt the parameter values of ACO, which includes:

- ACSGA-TSP, Pilat and White [12],
- Near parameter free ACS, Randall [13].

# ADAPTATION - ACSGA-TSP

In [12], the parameter values in ACS were optimized using a genetic algorithm,

The idea was to have a GA running on top of the ACS to optimize its parameter values,

The optimized parameters were:

- $q_0$ , the parameter determining whether the greedy or probabilistic selection is adopted,
- $\rho$ , the local pheromone updating factor,
- $\beta$ , the relative importance of the visibility heuristic.

# ADAPTATION - ACSGA-TSP

Each ant has its own values of ACS parameters,

Each GA ant was encoded by a 21-bit string (7 bits/parameter) for ranges:

- $q_0 \in [0, 1]$ , double value,
- $\rho \in [0, 1]$ , double value,
- $\beta \in [1, 127]$ , integer.

# ADAPTATION - ACSGA-TSP

The chromosomes were randomly initialized,

The crossover used was the single simple crossover to generate two children,

The mutation used was the bit-wise mutation (mutate every bit based on a given probability).



# ADAPTATION - ACSGA-TSP

**for** each generation **do**:

- Choose the best 4 individuals,
- **for** each of the 4 chosen individuals **do**
  - Run ACS-TSP given  $\beta$ ,  $\rho$ , and  $q_0$  value encoded in each individual and record the result as the fitness of the individual
- **end for**
- The global pheromone update is done by the ant producing the best tour,
- Choose the 2 individuals with the best fitness from chosen 4,
- Produce the 2 children by crossover or copy from the 2 chosen best individuals,
- Mutate the 2 children,
- Replace the worst 2 individuals from the chosen 4 in the population with the 2 children

**end for**

# ADAPTATION - ACSGA-TSP

Comparisons are made between the ACSGA and the ACS system using:

- 20 ants,
- A fixed number of tour constructions,
- Crossover probability of 0.9,
- Mutation probability of 0.01,
- Averages reported over 10 runs.

# ADAPTATION - ACSGA-TSP

Instance	ACS		ACSGA	
	Solution	Iterations	Solution	Iterations
eil51	<b>428.7</b>	2000	432.4	10000
ft70	41144	4000	<b>40868</b>	20000
kroA100	21614	5000	<b>21544</b>	25000
Rat783	12181	100	<b>10057</b>	500

# ADAPTATION – NEAR PARAMETER FREE ACS

In [13], the parameters of ACS were adapted using the same approach for optimizing the problem in hand (an ant approach),

The optimized parameters are:

- $q_0$ , the parameter determining whether the greedy or probabilistic selection is adopted,
- $\rho$ , the local pheromone updating factor,
- $\alpha$ , the global pheromone updating factor,
- $\beta$ , the relative importance of the visibility heuristic.

# ADAPTATION - NEAR PARAMETER FREE ACS

Each ant is allowed to:

- Select the suitable values of its parameters,
- Select the next solution component.

Each ant has its own parameter values adaptively selected using an ant approach:

- A separate pheromone matrix is kept for learning these values,
- No heuristic information (similar to the distance information in TSP) is used.

# ADAPTATION - NEAR PARAMETER FREE ACS

Each parameter is given a suitable range to lie in:

- $q_0 \in [0, 1]$ ,
- $\rho \in [0, 1]$ ,
- $\alpha \in [0, 1]$ ,
- $\beta \in [-5, -1]$ , from [8].

The initial value of each parameter is in the middle of its interval.

# ADAPTATION - NEAR PARAMETER FREE ACS

The pheromone matrix is defined as:

$$v(i, j)$$

Specific parameter

$$1 \leq i \leq 4$$

Range of values

# ADAPTATION - NEAR PARAMETER FREE ACS

For each parameter, the interval is discretized with a step  $P$ , thus dividing the interval into  $w_i, i = 1, \dots, p$

The parameter division,  $w_i$ , is chosen using transition rule based on the pheromone values  $v(i, j)$ .



# ADAPTATION - NEAR PARAMETER FREE ACS

The actual value of each parameter is then calculated according:

$$p_i = l_i + \frac{w_i}{P} (u_i - l_i), \quad 1 \leq i \leq 4$$

The diagram includes four green arrows pointing from text labels to specific parts of the equation:

- An arrow from "Value of parameter i" points to  $p_i$ .
- An arrow from "Lower bound of parameter i" points to  $l_i$ .
- An arrow from "The discretized division of parameter i" points to  $\frac{w_i}{P}$ .
- An arrow from "Upper bound of parameter i" points to  $u_i$ .

# ADAPTATION - NEAR PARAMETER FREE ACS

Comparisons are made between an adaptive version and the ACS system with parameter values from [8] using:

- 10 ants,
- $P = 20$  divisions,
- Maximum of 3000 iterations,
- Applied to TSP with results were reported as deviations from the optimal solution,
- The results reported are the median values of the relative difference of the best known solution

# ADAPTATION - NEAR PARAMETER FREE ACS

Instance	Non-adaptive		adaptive	
	Rel-diff	Time	Rel-diff	Time
hk48	0.08	1.29	<b>0.04</b>	3.41
eil51	2	0.49	<b>0.23</b>	9.75
st70	1.33	43.48	<b>0.89</b>	34.9
d198	<b>0.33</b>	1723.34	1.69	1975.71
ts225	<b>1.15</b>	3019.9	3.53	611.94
pcb442	<b>3.53</b>	38445.9	12.1	199.06

# OUTLINE

Introduction,  
ACO,  
ACO Applications,  
Adaptation,  
Cooperation,  
References.

# COOPERATION

Two approaches have been studied to investigate the cooperation between different ant colonies:

- A *heterogeneous* approach, in which the ants in the two colonies have different behaviours (different optimization criteria),
- A *homogeneous* approach, in which all the ants show a similar behaviour.

# COOPERATION

The heterogeneous approach was proposed by Gambardella et al. [14],

It was applied to a multi-criteria optimization problem (multi-objective),

Each colony was used to optimize a different criterion.

# COOPERATION

Different homogeneous and parallel approaches were taken, similar to what was done with GAs, including:

- A fine-grained implementation, where each processor holds a single ant,
- A coarse-grained implementation, where each processor holds a complete colony,
- A complete survey could be found in [15].

# COOPERATION

In [15], the multi-colony algorithm was studied by having colonies connected in a directed-ring fashion,

Four information exchange approaches were studied:

- All colonies get the same global best solution,
- Circular exchange of locally best solutions,
- Circular exchange of a number of ants (migrants),
- A mixture of the two previous approaches.



# COOPERATION

The multi-colony approach was applied to TSP and compared to the performance of a single colony,

The best exchange method was found to be the circular exchange of locally best solutions,

A multi-colony approach with a moderate number of colonies is better than a single colony.

# REFERENCES

1. P. P. Grasse. "Recherches sur la biologie des termites champignonnistes (*Macrotermitina*)". Ann. Sc. Nat., Zool. Biol. anim., 6, 97, 1944.
2. P. P. Grasse. "La reconstruction du nid et les coordinations interindividu- elles chez *bellicositermes natalensis* et *cubitermes* sp. La theorie de la stig- mergie: essai d'interpretation du comportement des termites constructeurs". Insectes Sociaux, 6, 41, 1959.
3. J. L. Deneubourg, S. Aron, S. Goss and J. M. Pasteels. "The self-organizing exploratory pattern of the Argentine ant". Journal of Insect Behaviour, 3, 159, 1990.
4. S. Goss, S. Aron, J. L. Deneubourg and J. M. Pasteels. "Self-organized shortcuts in the Argentine ant". Naturwissenschaften, 76, 579, 1989.
5. M. Dorigo. "Optimization, Learning and Natural Algorithms". Ph.D. thesis, DEI, Politecnico di Milano, Italy, pp. 140, 1992.

# REFERENCES

6. M. Dorigo, V. Maniezzo and A. Coloni." Ant System: Optimization by a Colony of Cooperating Agents". IEEE Trans. Syst., Man, and Cybern. Part B, vol. 26, no. 1, 1996.
7. T. Stutzle and H. H. Hoos."MAX-MIN Ant System". *Future Generation Comput. Syst.*, vol. 16, no 8, 2000.
8. M. Dorigo and L. M. Gambardella."Ant Colony System: a cooperative learning approach to the traveling salesman problem". IEEE Trans. Evolutionary Computation, vol. 1, no. 1, 1997.
9. J. Bautista and J. Pereira. "Ant Algorithms for A Time and Space Constrained Assembly Line Balancing Problem". European Journal of Operation Research, 177, pp. 2016-2032, 2002.
10. S. J. Shyu, B. M. T. Lin and T. S. Hsiao."An Ant Algorithm for Cell Assignment in PCS Networks". IEEE International Conference on Networking, Sensing and Control, pp. 1081-1086, 2004.

# REFERENCES

11. J. R. L. Fournier and S. Pierre. "Assigning Cells to Switches in Mobile Networks using an Ant Colony Optimization Heuristic". Computer Communications, vol. 28, pp. 65-73, 2005.
12. M. L. Pilat and T. White. "Using Genetic Algorithms to Optimize ACS-TSP", 2002.
13. M. Randall. "Near Parameter Free Ant Colony Optimisation". In M. Dorigo, M. Birattari, C. Blum, L. M. Gambardella, F. Mondada, and T. Stützle, editors, ANTS Workshop, volume 3172 of Lecture Notes in Computer Science, pp. 374–381. Springer, 2004.
14. L. M. Gambardella, E. D. Taillard and G. Aggazi. "MACS-VRPT: A Multiple Ant Colony System for Vehicle Routing Problem with Time Windows". New Ideas in Optimization, McGraw Hill, pp. 63-67, 1999.
15. M. Middendorf, F. Riechle and H. Schmeck. "Information Exchange in Multi Ant Colony Algorithms". Journal of Heuristics, vol. 8, pp. 305-320, 2002.
16. M. Dorigo and L. M. Gambardella. "Ant Colonies for the travelling salesman problem", BioSystems, 1997.