# Ant Colony Optimization

- Real ants
- Stigmergy
- Autocatalyzation
- Ant System
- Ant Colony System

# Overview

"Ant Colony Optimization (ACO) studies artificial systems that take inspiration from the *behavior of real ant colonies* and which are used to solve discrete optimization problems."

***-Source: ACO website, http://iridia.ulb.ac.be/~mdorigo/ACO/about.html***

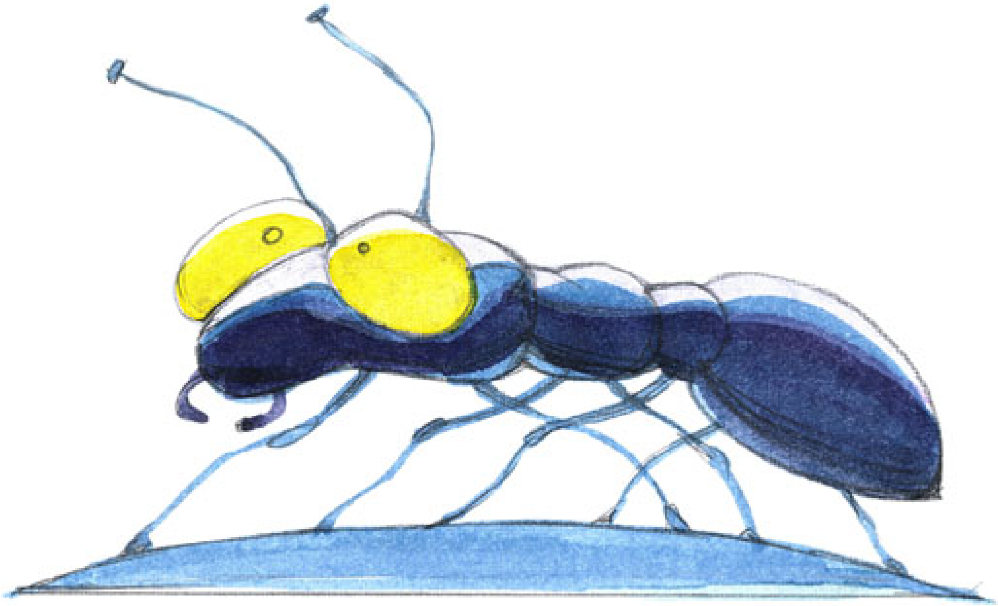
Almost blind.

Incapable of achieving complex tasks alone.

Rely on the phenomena of swarm intelligence for survival.

Capable of establishing shortest-route paths from their colony to feeding sources and back.

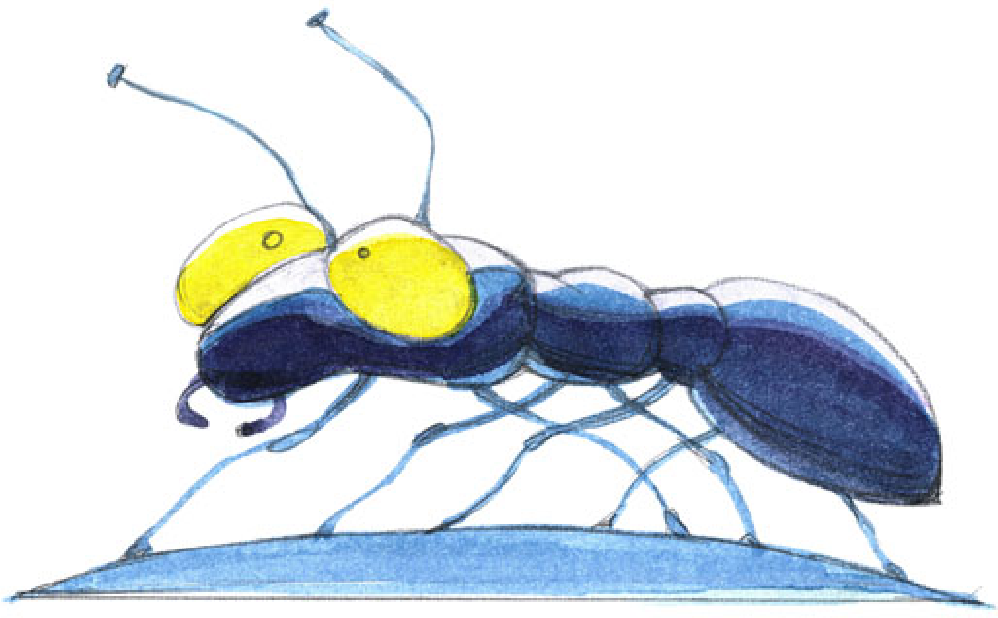
Use stigmergic communication via pheromone trails.



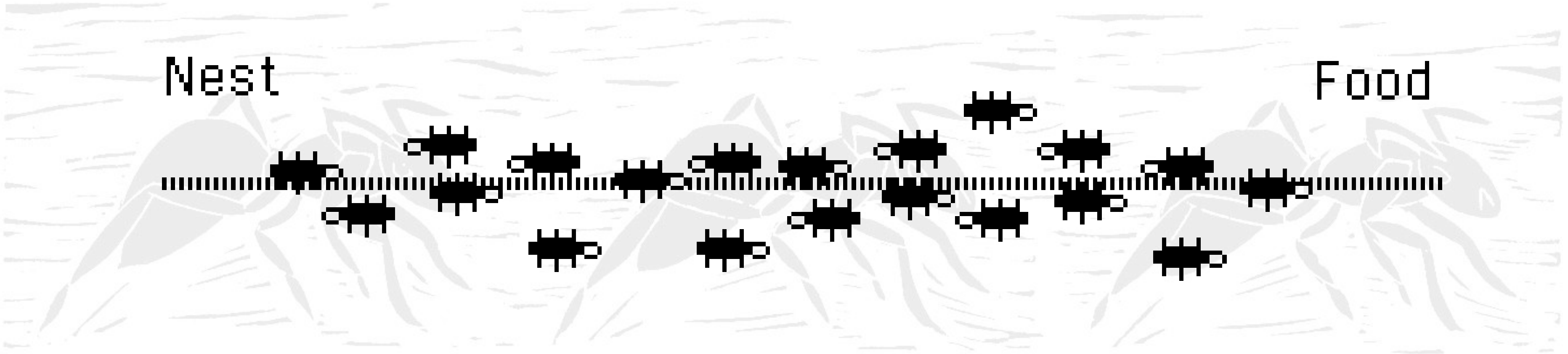
Follow existing pheromone trails with high probability.

What emerges is a form of *autocatalytic* behavior: the more ants follow a trail, the more attractive that trail becomes for being followed.

The process is thus characterized by a positive feedback loop, where the probability of a discrete path choice increases with the number of times the same path was chosen before.

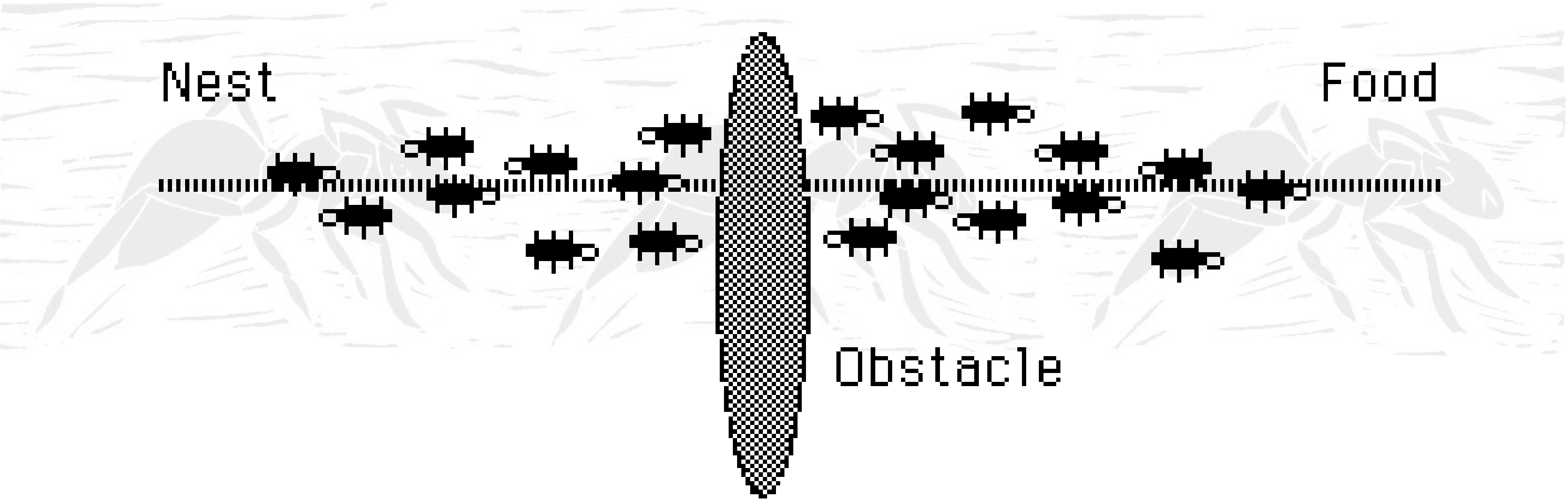


# Naturally Observed Ant Behavior



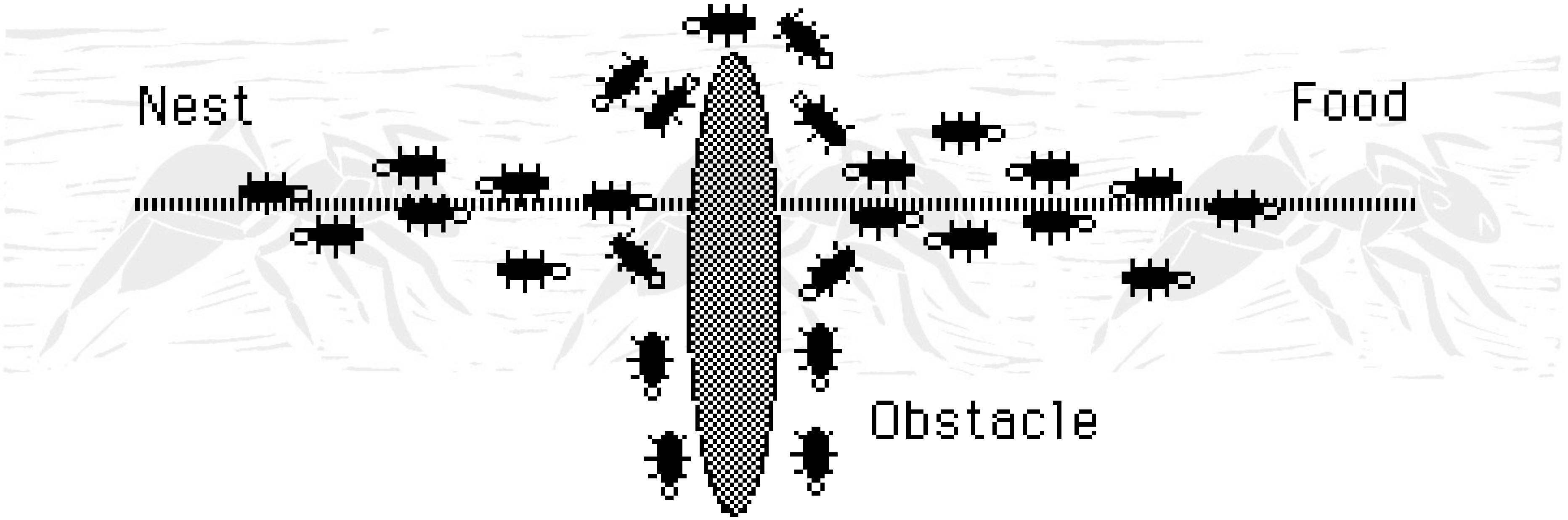

**All is well in the world of the ant.**

# Naturally Observed Ant Behavior



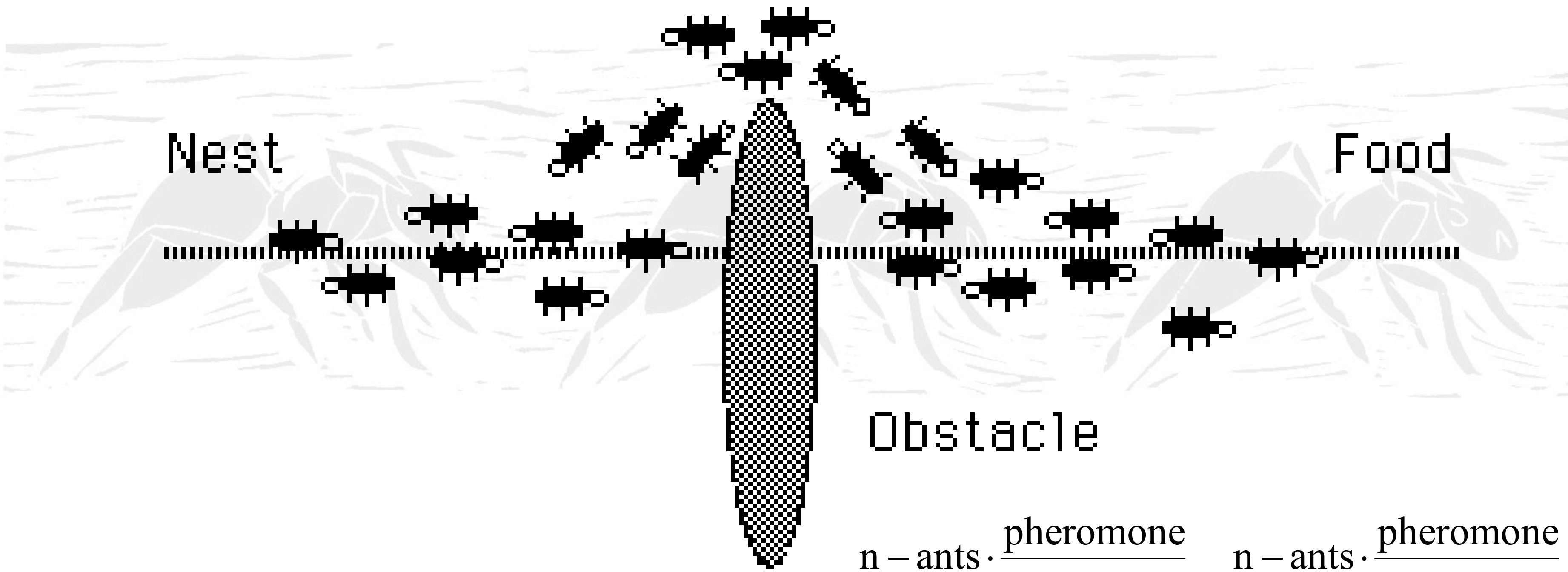

**Oh no! An obstacle has blocked our path!**

# Naturally Observed Ant Behavior




**Where do we go? Everybody, flip a coin.**

# Naturally Observed Ant Behavior




$$\frac{n-\text{ants}\cdot\dfrac{\text{pheromone}}{\text{x - distance}}}{\dfrac{\text{distance}_{\text{longer path}}}{\text{time}}} < \frac{n-\text{ants}\cdot\dfrac{\text{pheromone}}{\text{x - distance}}}{\dfrac{\text{distance}_{\text{shorter path}}}{\text{time}}}$$

**Shorter path reinforced.**

# “Stigmergic?”

- Stigmergy, a term coined by French biologist Pierre-Paul Grasse, is interaction through the environment.
- Two individuals interact indirectly when one of them modifies the environment and the other responds to the new environment at a later time. This is stigmergy.

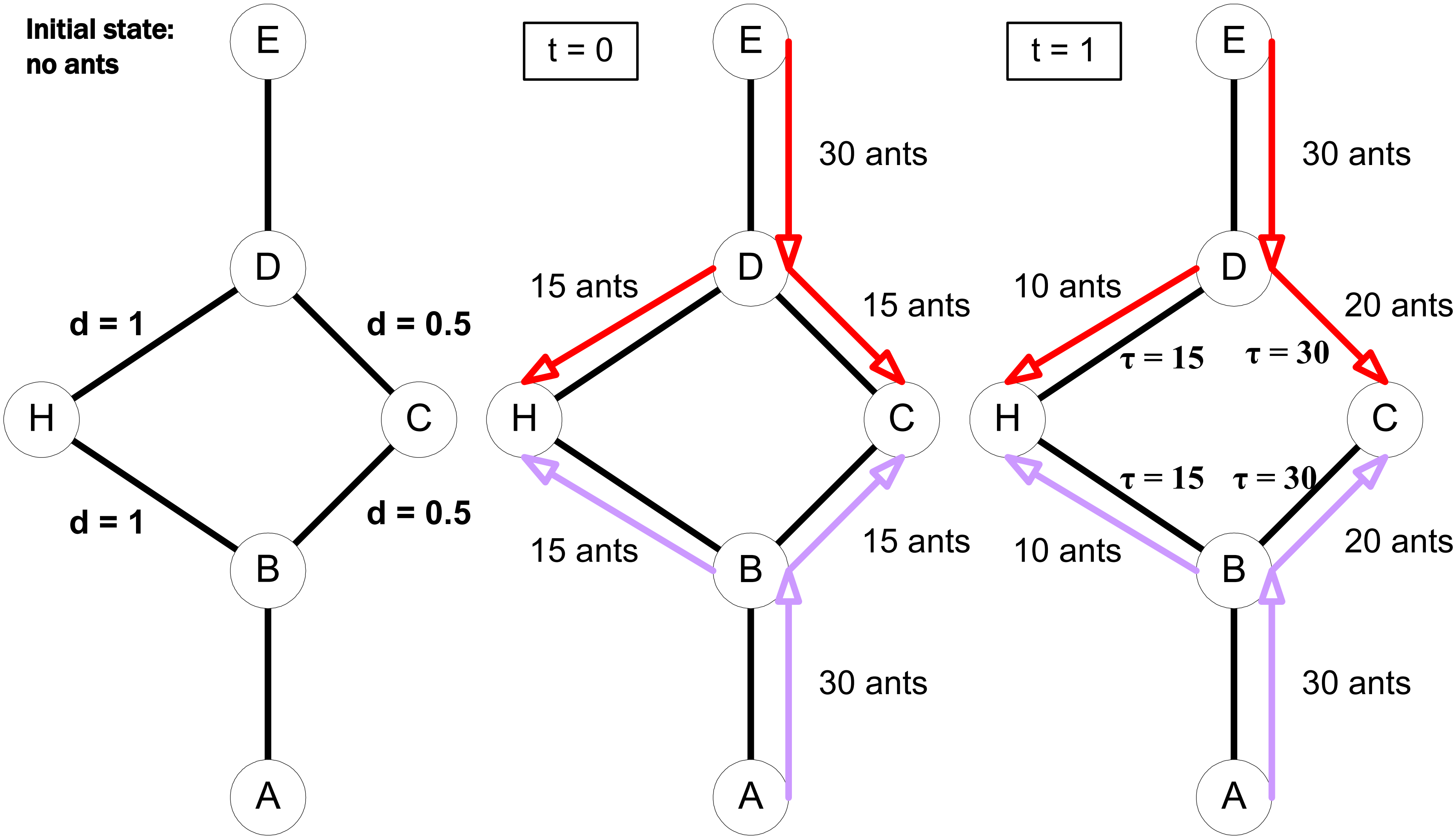
# Stigmergy

**Real ants use stigmergy. How again?**

**PHEROMONES!!!**

# Autocatalyzation

What is *autocatalytic behavior*?


Initial state:
no ants
E
D
H
C
B
A
d = 1
d = 0.5
d = 1
d = 0.5
t = 0
30 ants
15 ants
15 ants
15 ants
15 ants
30 ants
t = 1
30 ants
10 ants
20 ants
τ = 15
τ = 30
τ = 15
τ = 30
10 ants
20 ants
30 ants

(a) (b) (c)

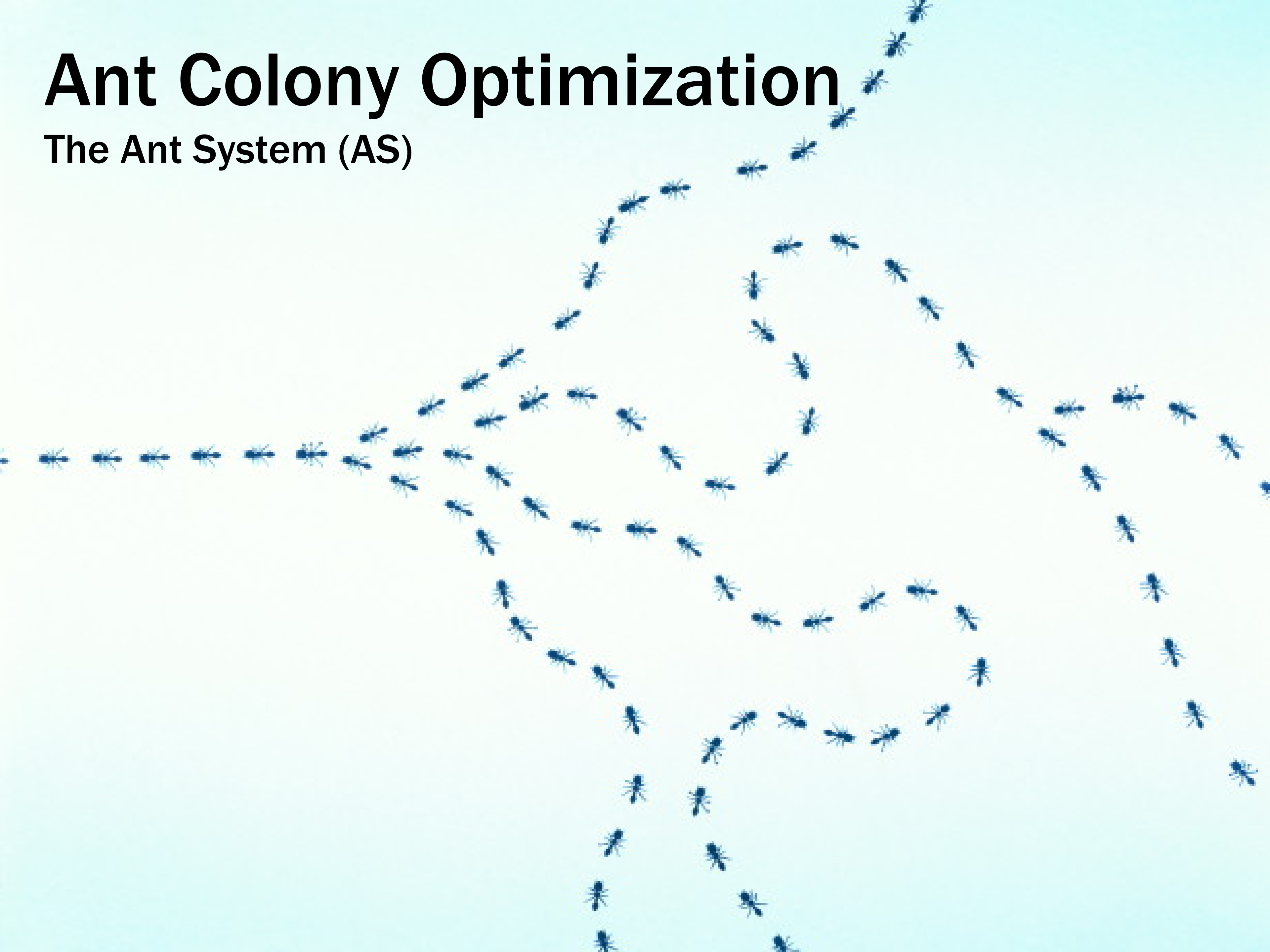
# Autocatalyzation

This is why ACO algorithms are called autocatalytic positive feedback algorithms!

***Remember that!***

# Ant Colony Optimization

## The Ant System (AS)



# Ant System

- First introduced by Marco Dorigo in 1992
- Progenitor to "Ant Colony System," later discussed
- Result of research on computational intelligence approaches to combinatorial optimization
- Originally applied to Traveling Salesman Problem
- Applied later to various hard optimization problems

Would you trust this man?

# Performance Chart

| Problem Name | MST | AS | $ACS_{R\&D}$ | $ACS_D$ | GA | EP | SA | Optimum |
|---|---|---|---|---|---|---|---|---|
| Eil50 (50-city problem) | 615 [1] 44.71% | 450 [36] 5.89% | 463.423 [3] 9.04% | **425** [1,830] 0.00% | 428 [25,000] 0.71% | 426 [100,000] 0.23% | 443 [68.512] 4.24% | **425** [N/A] |
| Eil75 (75-city problem) | 740 [1] 38.31% | 570 [238] 6.5% | 576.749 [10] 7.80% | 545 [3,840] 1.87% | 545 [80,000] 1.87% | 542 [325,000] 1.31% | 580 [173,250] 8.41% | **535** [N/A] |
| KroA100 (100-city problem) | 30517 [1] 43.39% | 22,943 [228] 7.81% | 24497.6 [37] 15.11% | **21,282** [4,820] 0.00% | 21,761 [103,000] 2.25% | N/A [N/A] N/A | N/A [N/A] N/A | 21,282 [N/A] |

**Our Results**

**MST** 2-approximation TSP algorithm

**AS** Ant System ($\alpha = 1, \beta = 5, \rho = .5$)

**$ACS_{R\&D}$** Ant Colony System ($\alpha = 0.1, \beta =2, \rho = .1, m = 50$)

**Published Results**

**$ACS_D$** Ant Colony System

**GA** Genetic Algorithm

**EP** Evolutionary Programming

**SA** Simulated Annealing

# Ants as Agents

Each ant is a simple agent with the following characteristics:

- It chooses the town to go to with a probability that is a function of the town distance and of the amount of trail present on the connecting edge;
- To force the ant to make legal tours, transitions to already visited towns are disallowed until a tour is complete (this is controlled by a tabu list);
- When it completes a tour, it lays a substance called *trail* on each edge $(i, j)$ visited.

The symmetric TSP has a Euclidean based problem space. We use $d_{ij}$ to denote the distance between any two cities in the problem. As such

$$d_{ij} = [(x_i - x_j)^2 + (y_i - y_j)^2]^{1/2}$$

We let $\tau_{ij}(t)$ denote the intensity of trail on edge $(i,j)$ at time $t$. Trail intensity is updated following the completion of each algorithm cycle, at which time every ant will have completed a tour. Each ant subsequently deposits trail of quantity $Q/L_k$ on every edge $(i,j)$ visited in its individual tour. Notice how this method would favor shorter tour segments. The sum of all newly deposited trail is denoted by $\Delta\ \tau_{ij}$. Following trail deposition by all ants, the trail value is updated using $\tau_{ij}(t + n) = p \times \tau_{ij}(t) + \Delta\ \tau_{ij}$, where p is the rate of trail decay per time interval and $\Delta\ \tau_{ij} =$ . $\sum_{k=1}^{m} \Delta \tau_{ij}$

Two factors drive the probabilistic model:

1) ***Visibility***, denoted $\eta_{ij}$, equals the quantity $1/d_{ij}$

2) ***Trail***, denoted $\tau_{ij}(t)$

These two factors play an essential role in the central probabilistic transition function of the Ant System.

In return, the weight of either factor in the transition function is controlled by the variables $\alpha$ and $\beta$, respectively. Significant study has been undertaken by researchers to derive optimal $\alpha:\beta$ combinations.

# Probabilistic Transition Function

$$p_{ij}^{k}(t) = \begin{cases} \dfrac{\left[\tau_{ij}(t)\right]^{\alpha} \cdot \left[\eta_{ij}\right]^{\beta}}{\sum_{k \in allowed_k} \left[\tau_{ij}(t)\right]^{\alpha} \cdot \left[\eta_{ij}\right]^{\beta}} & \text{if } k \in allowed_k \\ 0 & otherwise \end{cases}$$

A high value for $\alpha$ means that trail is very important and therefore ants tend to choose edges chosen by other ants in the past. On the other hand, low values of $\alpha$ make the algorithm very similar to a stochastic multigreedy algorithm.

# Ant System (AS) Algorithm

1. Initialization
2. Randomly place ants
3. Build tours
4. Deposit trail
5. Update trail
6. Loop or exit

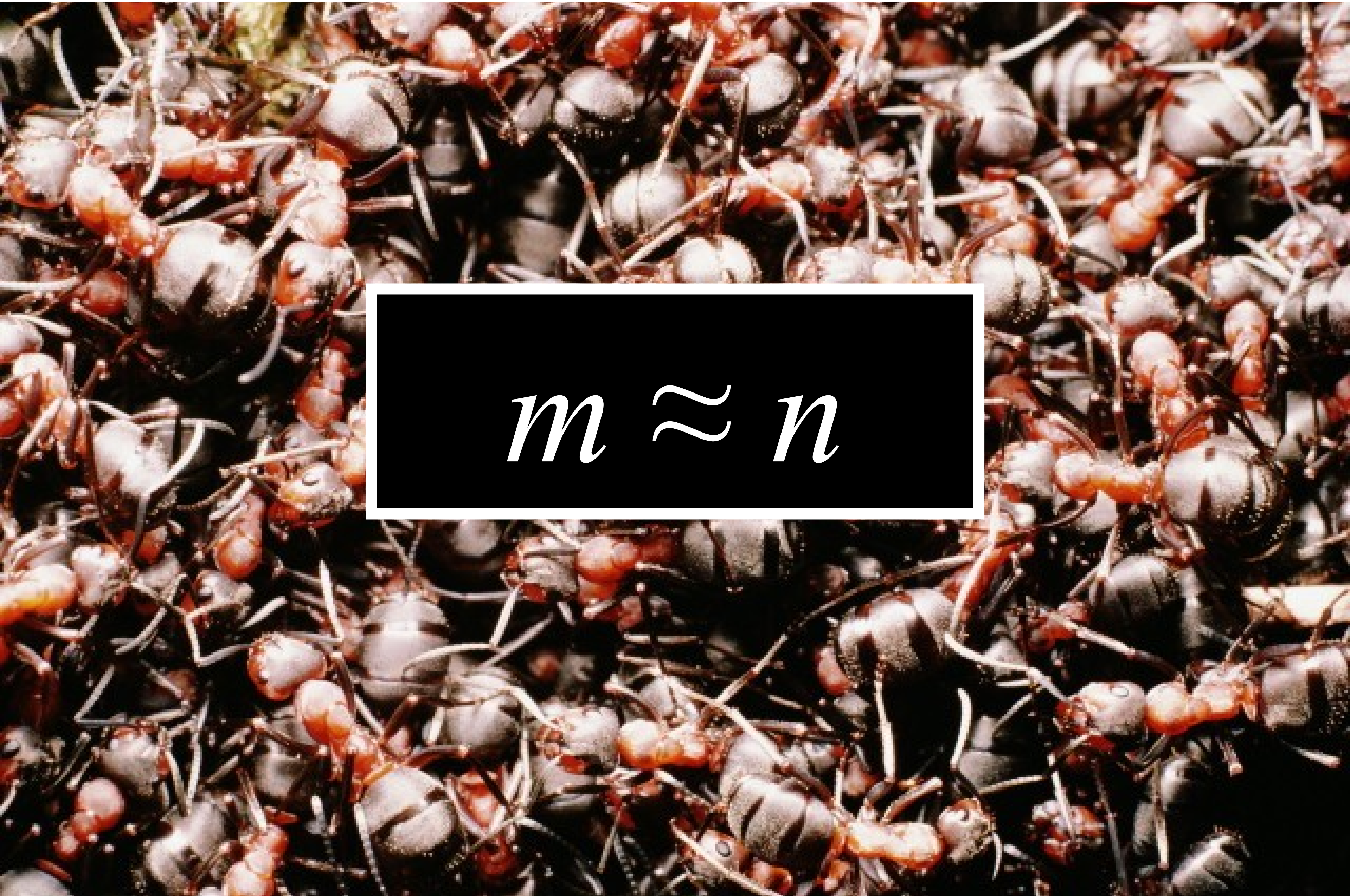
# Computational Complexity

The complexity of this ACO algorithm is $O(NC \times n^2 \times m)$ if we stop the algorithm after $NC$ cycles, where $n$ is the number of cities and $m$ is the number of ants.

Step 1 is $O(n^2 + m)$
Step 2 is $O(m)$
Step 3 is $O(n^2 \times m)$
Step 4 is $O(n^2 \times m)$
Step 5 is $O(n^2)$
Step 6 is $O(n \times m)$

Researchers have found a linear relation between the number of towns and the best number of ants, so the complexity of the algorithm is $O(NC \times n^3)$.

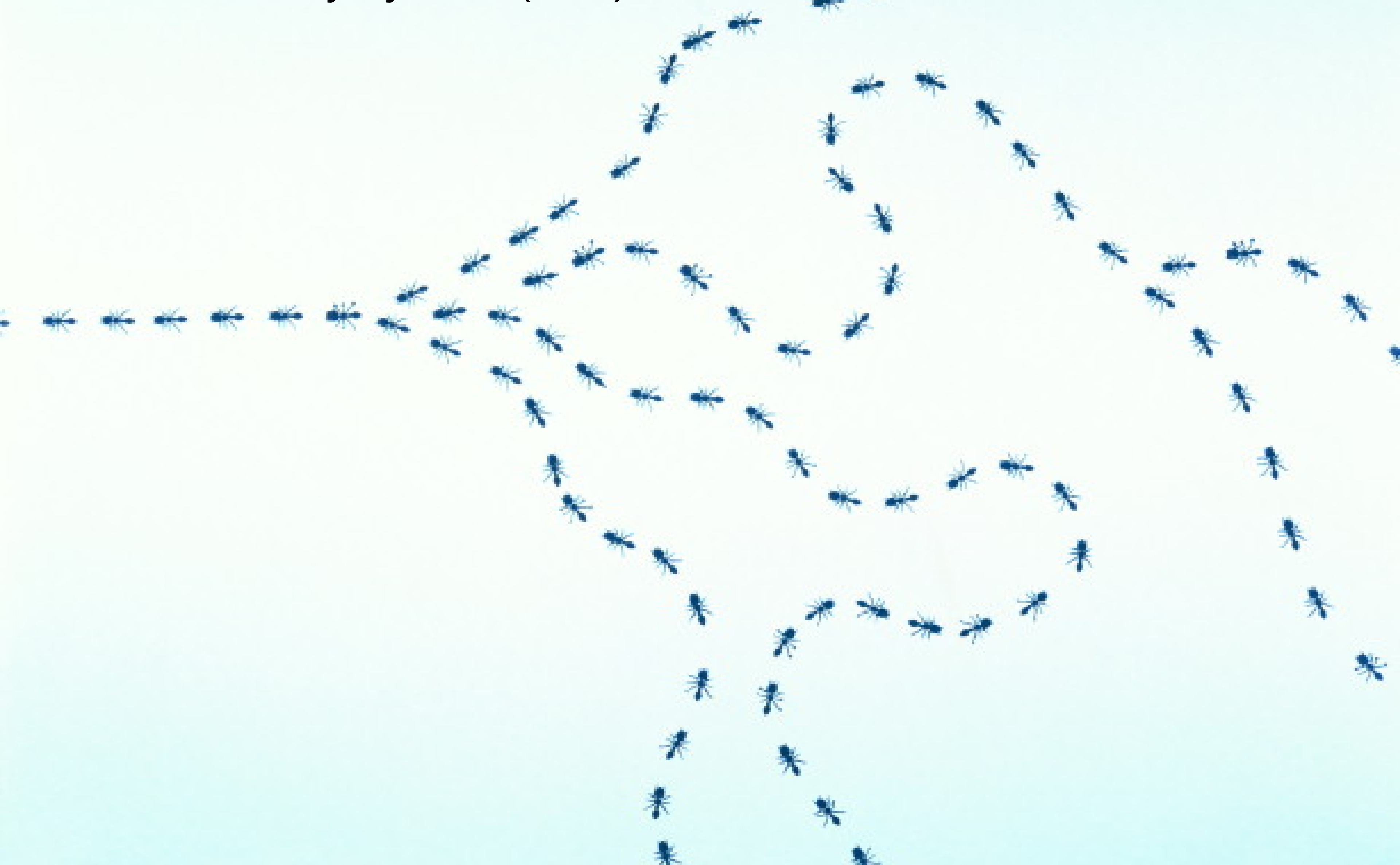
# How many ants do you need?



$$m \approx n$$

# Ant Colony Optimization

## The Ant Colony System (ACS)



# AS → ACS

Change to the probabilistic function: drop alpha

$$
p_{ij}^{k}(t) = \begin{cases} \dfrac{\left[\tau_{ij}(t)\right] \cdot \left[\eta_{ij}\right]^{\beta}}{\sum\limits_{k \in allowed_k} \left[\tau_{ij}(t)\right] \cdot \left[\eta_{ij}\right]^{\beta}} & if\ k \in allowed_k \\ 0 & otherwise \end{cases}
$$

# AS ↝ ACS

New state transition rule; used to balance between **exploration** and **exploitation**.

$$s = \begin{cases} \underset{u \in J_k(r)}{\arg\max} \left\{ [\tau(r,u)] \cdot [\eta(r,u)]^{\beta} \right\}_{\alpha} & \text{if } q \le q_0 \quad \text{(exploitation)} \\ S & \text{otherwise} \quad \text{(biased exploration)} \end{cases}$$

Here $q_0$ is a constant parameter, q is a random variable, and S is the outcome of the probabilistic transition function.

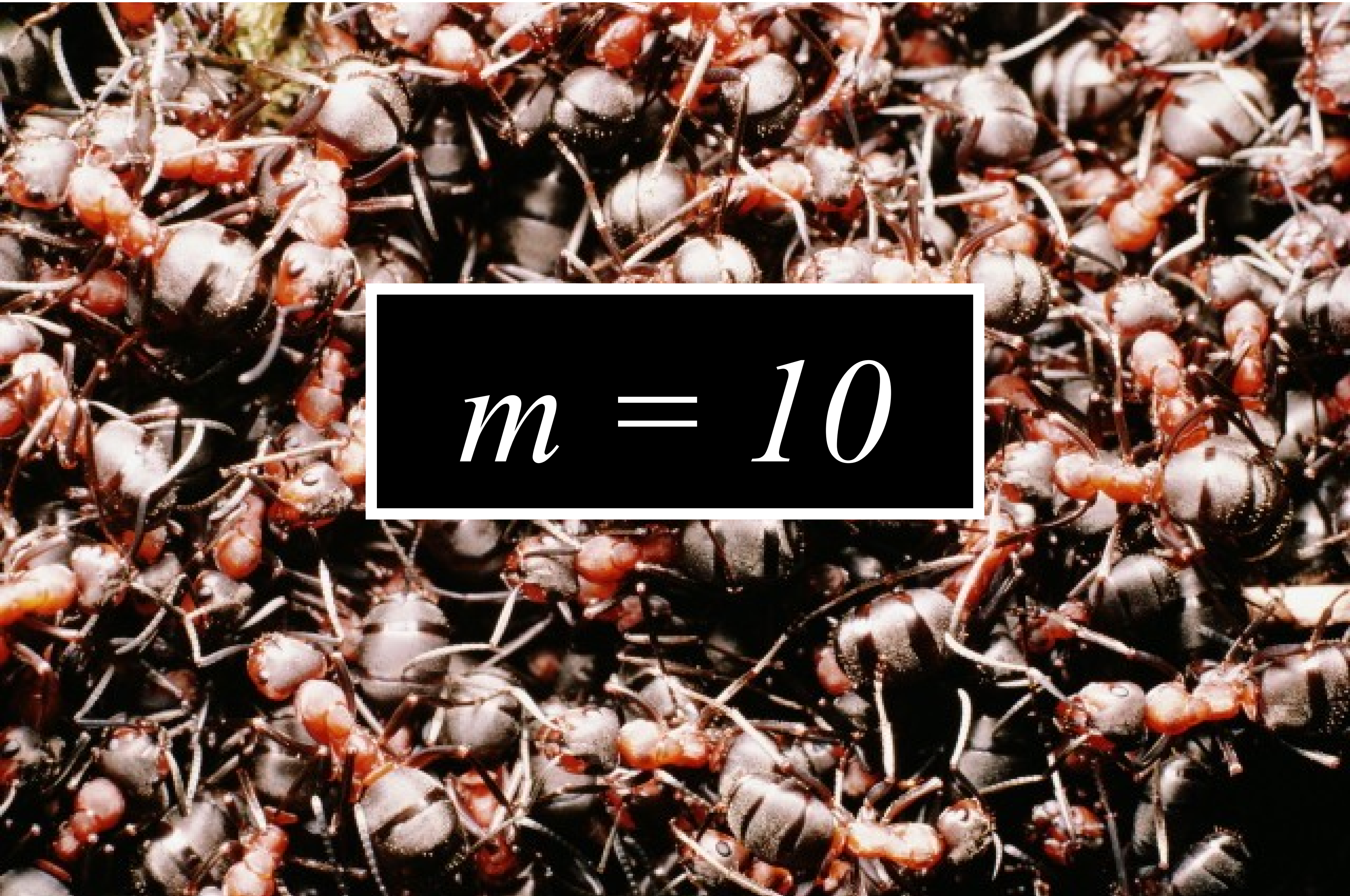
# AS & ACS

Local updating rule:

$$\tau(r,s) \leftarrow (1-\rho)\cdot\tau_{\alpha}(r,s)+\rho\cdot\Delta\tau(r,s)$$

Here $\Delta tau_0$ is a predetermined constant or function. The edge (r,s) is updated following each iteration of an ant search.

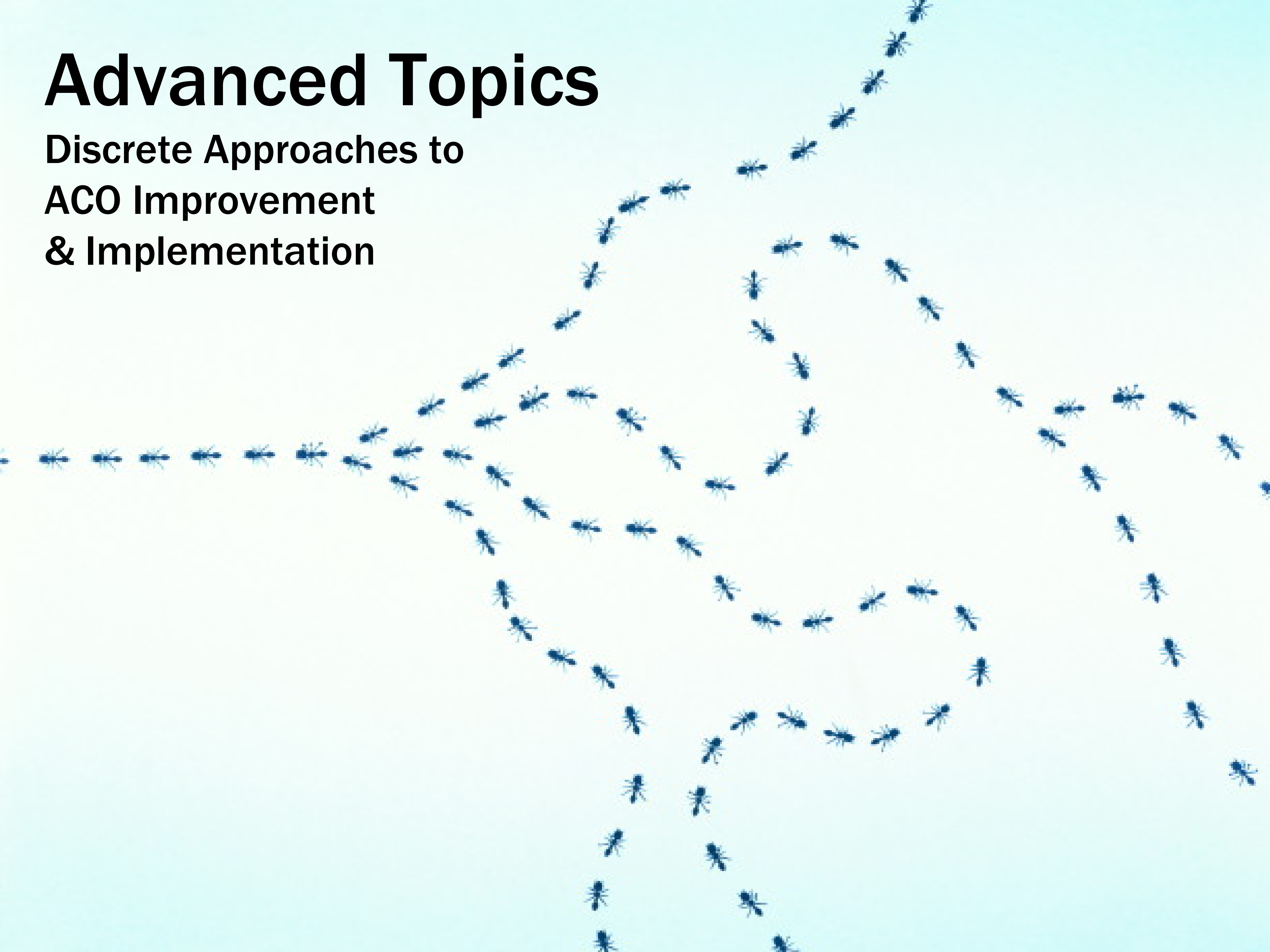
# How many ants do you need?



$$m = 10$$

# Advanced Topics

## Discrete Approaches to ACO Improvement & Implementation



**Check out**

**http://www.conquerware.com/dbabb/academics/research/aco**

**for supplementary materials.**

# Conclusion

The main characteristics of this class of algorithms are a natural metaphor, a stochastic nature, adaptivity, inherent parallelism, and positive feedback. Ants have evolved a highly efficient method of solving the difficult Traveling Salesman Problem. Furthermore, the Ant Colony Optimization can be applied to many other hard problems.

# Questions, Comments?

Thank You