PAAJSFDCICOP

Python And Asynchronous JavaScript

For

Distributed Computation

In

Combinatorial Optimization Problems

Ed Sykes & Jan-Klaas Kollhof

Intro

- How it all began
- Real World Ants
- Simulating Ants
- Ants for solving TSPs
- Distributed TSP solution
- Considerations
- Conclusions

Many Moons Ago

- ACCU 2006 presentation on jsolait
- Idea of AJSFDSC was born
- Ed was interested in learning python
- Jan was interested in learning some AI
- Decided to have a go with ant systems
- Submission of topic to ACCU 2007

Motivation

- learning python (Ed)
- learning some AI (Jan)
- learn about distributed systems
- gain knowledge and experience to share with others
- become famous and dominate the world
- enjoy doing the above

Ants in Nature

ANTS IN NATURE

- Types of self organising behaviour
- How ants find food
- Implications for solving difficult problems

ANTS IN NATURE

- Ants have long fascinated humans
- Simple agents solving complex problems
- What governs them?
- How do they plan and execute?
- Knowing how they works helps AI

Ants in Nature

Self organise to create routes

Weaver Ants

Ants in Nature

Self organising to find food – Leafcutter ants

Ants in Nature

Self organise to divide labour

Pheidole Ants

Ants in Nature

How ants organise to find food sources

- Direct contact vs stigmergy
- Laying and following of pheromones
- Probabilistic behaviour in following
- Autocatalytic system

Ants in Nature

Injecting Obstacles

An example with real ants.

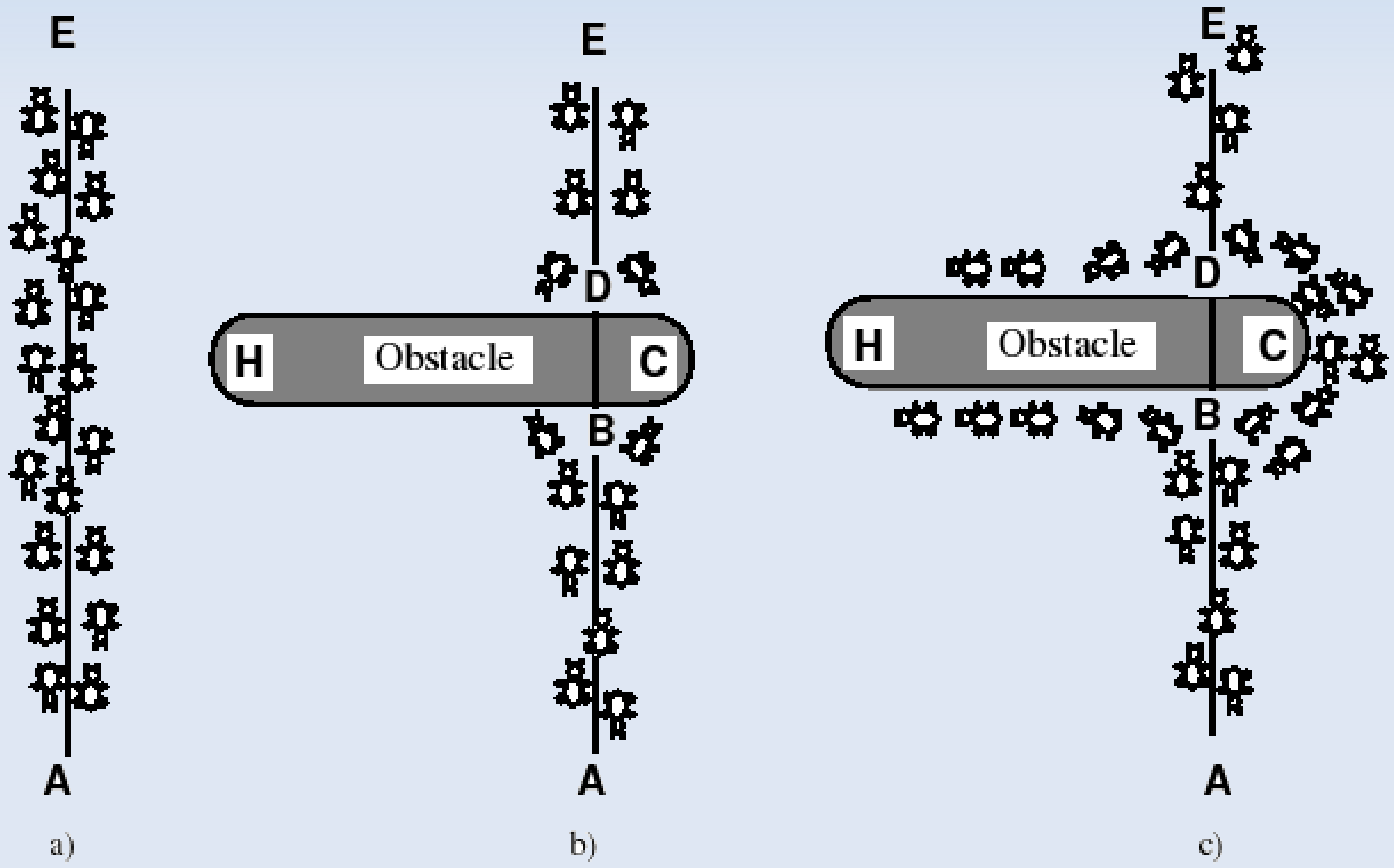
a) Ants follow a path between points A and E.

b) An obstacle is interposed; ants can choose to go around it following one of the two different paths with equal probability.

c) On the shorter path more pheromone is laid down.

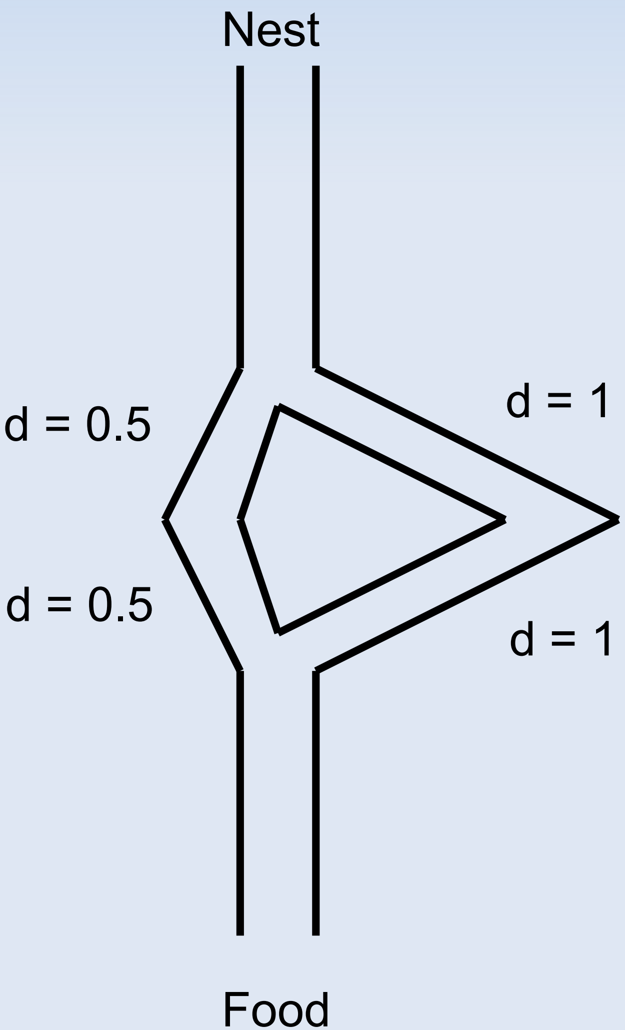
Ants in Nature

Injecting Obstacles



Ants in Nature

Binary branching



Ants in Nature

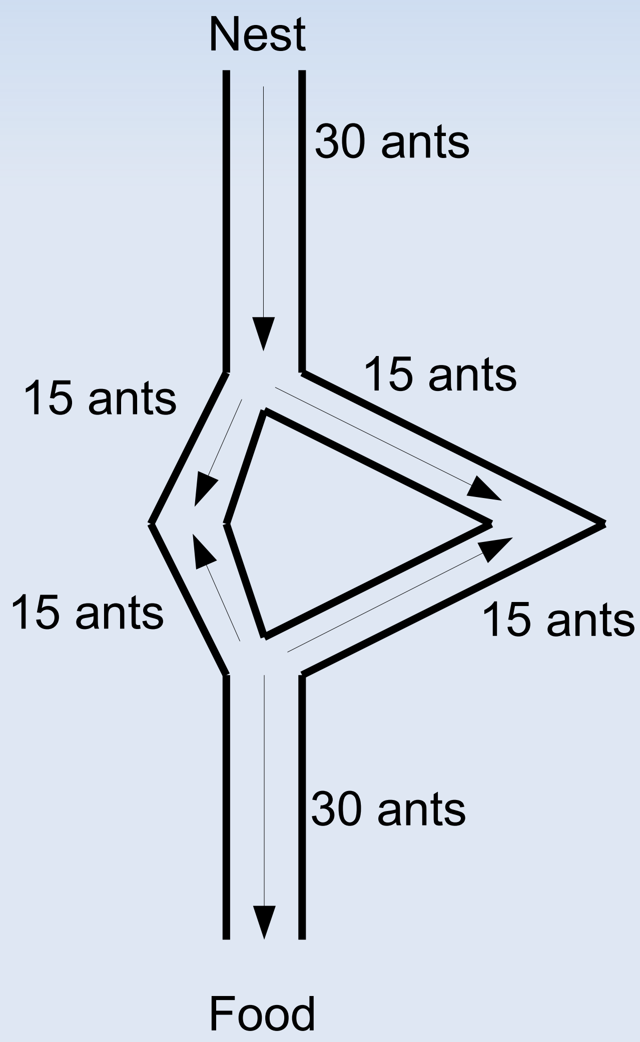
Ant behaviour

- In each time tick ants move a distance of 1
- in each time tick ants lay a pheromone deposit of 1

Ants in Nature

Binary branching

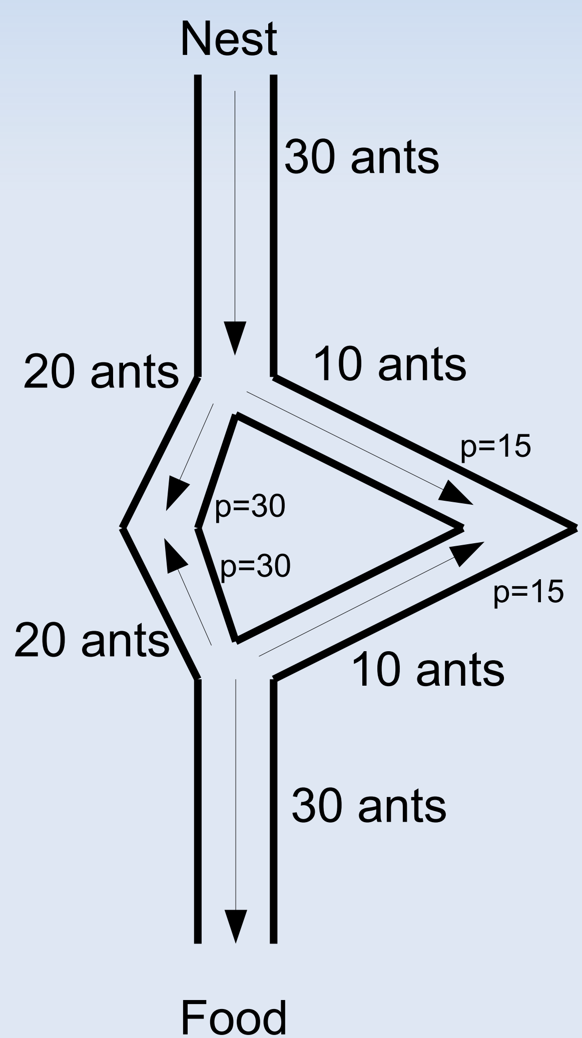
t = 0



Ants in Nature

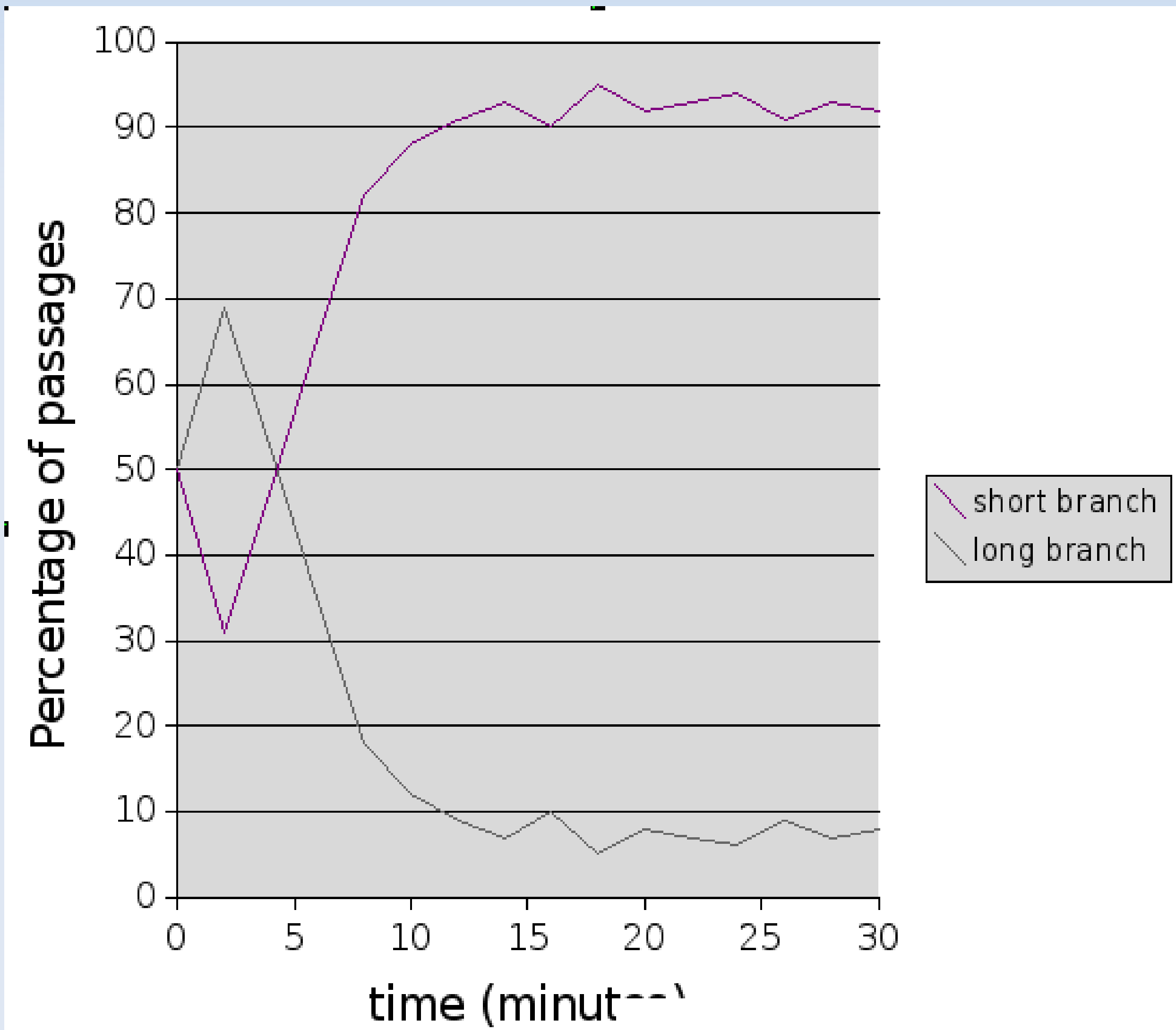
Binary branching

t = 1



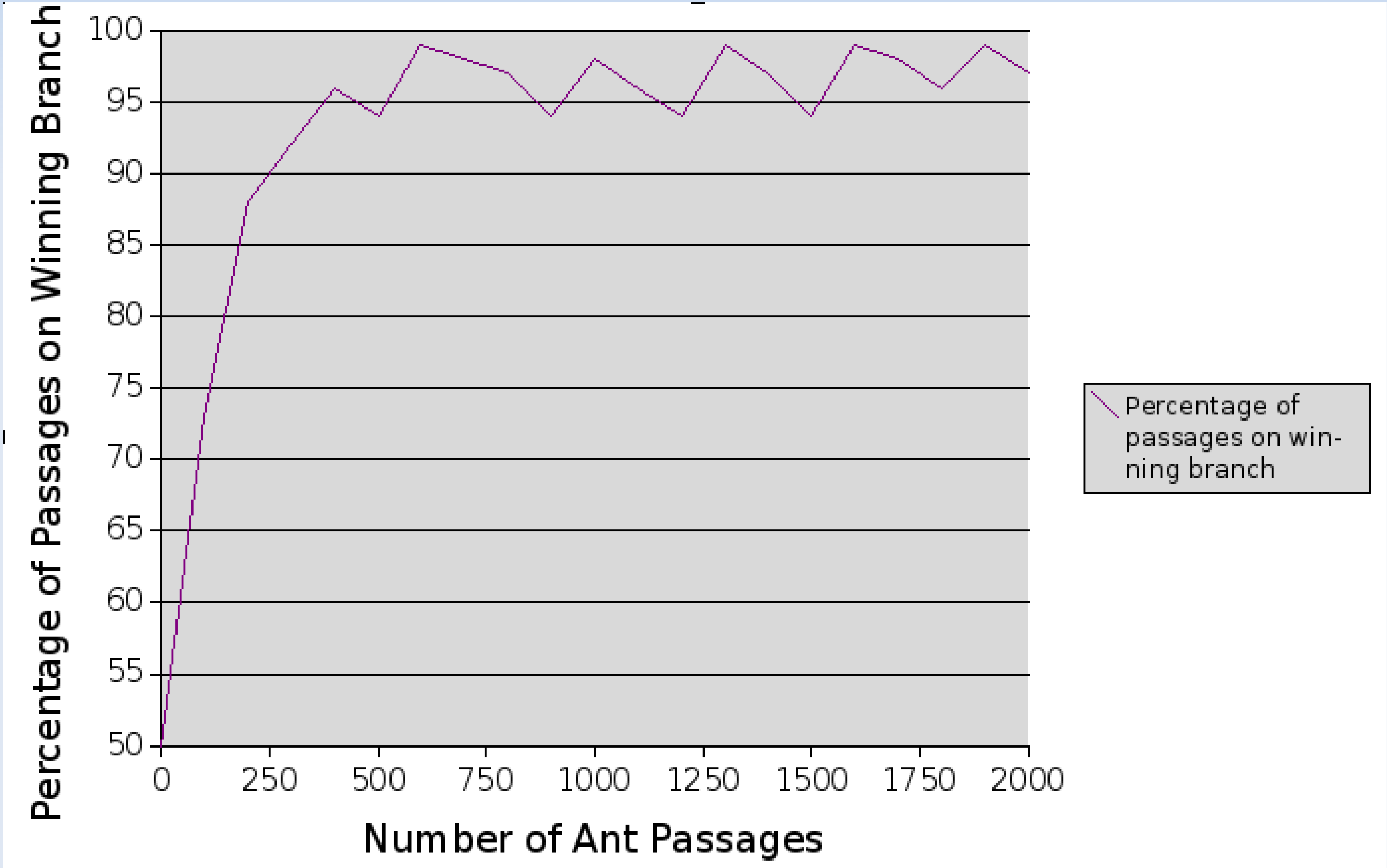
Ants in Nature

Short / long branch choice over time



Ants in Nature

Convergence on shortest branch



Ants in Nature

How do ants help us?

- Harness millions of years of evolution to help solve a problem
- Once we understand how a self organising system work we can implement simple agents easily
- Distributed computation is attractive in today's multicore environment

PAAJSFDCICOP

Ant Simulation

Ant Simulation

- Technology and Tools we used
- Development steps
- Frustration

Ant Simulation

- Python
- WX-Python
- Numpy

- Trac + SVN for source control and documentation

- TDD, Pair programming

Ant Simulation

- Randomly walking ants
- Ants avoid walls
- Central ant nest
- Ants detect food and lay down pheromones
- Ants follow pheromones
- Saturation, evaporation of pheromones
- 2 types of phreomones
- Ants converge to a ”good route” !

Ant Simulation

Demo

Ant Simulation

Frustration

- Pheromone detection
- Different type of ”ant visions”
- Balancing of the system
- Tiredness from post work hacking
- Intermittent nature of sessions

Ant System for TSP

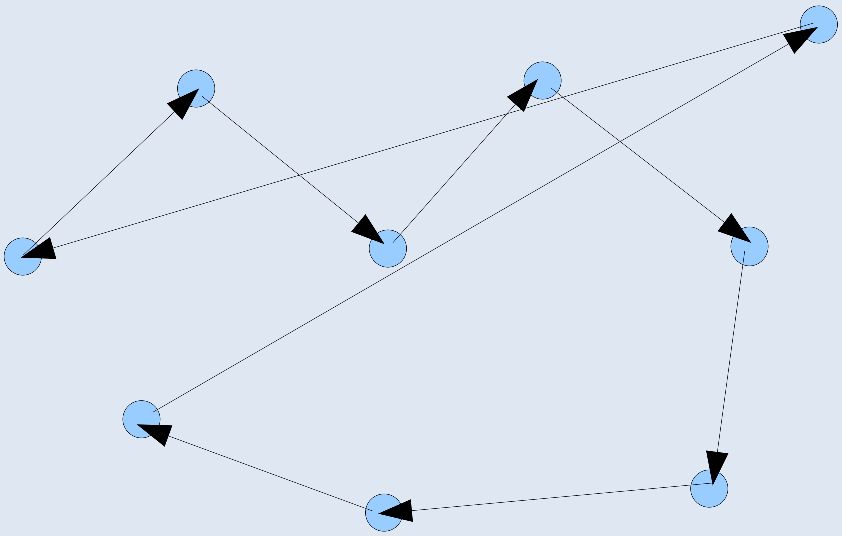
Ant System for TSP

- Travelling Salesman Problem
- The Ant System algorithm
- Implemention

Ant System for TSP

Travelling Salesman Problem

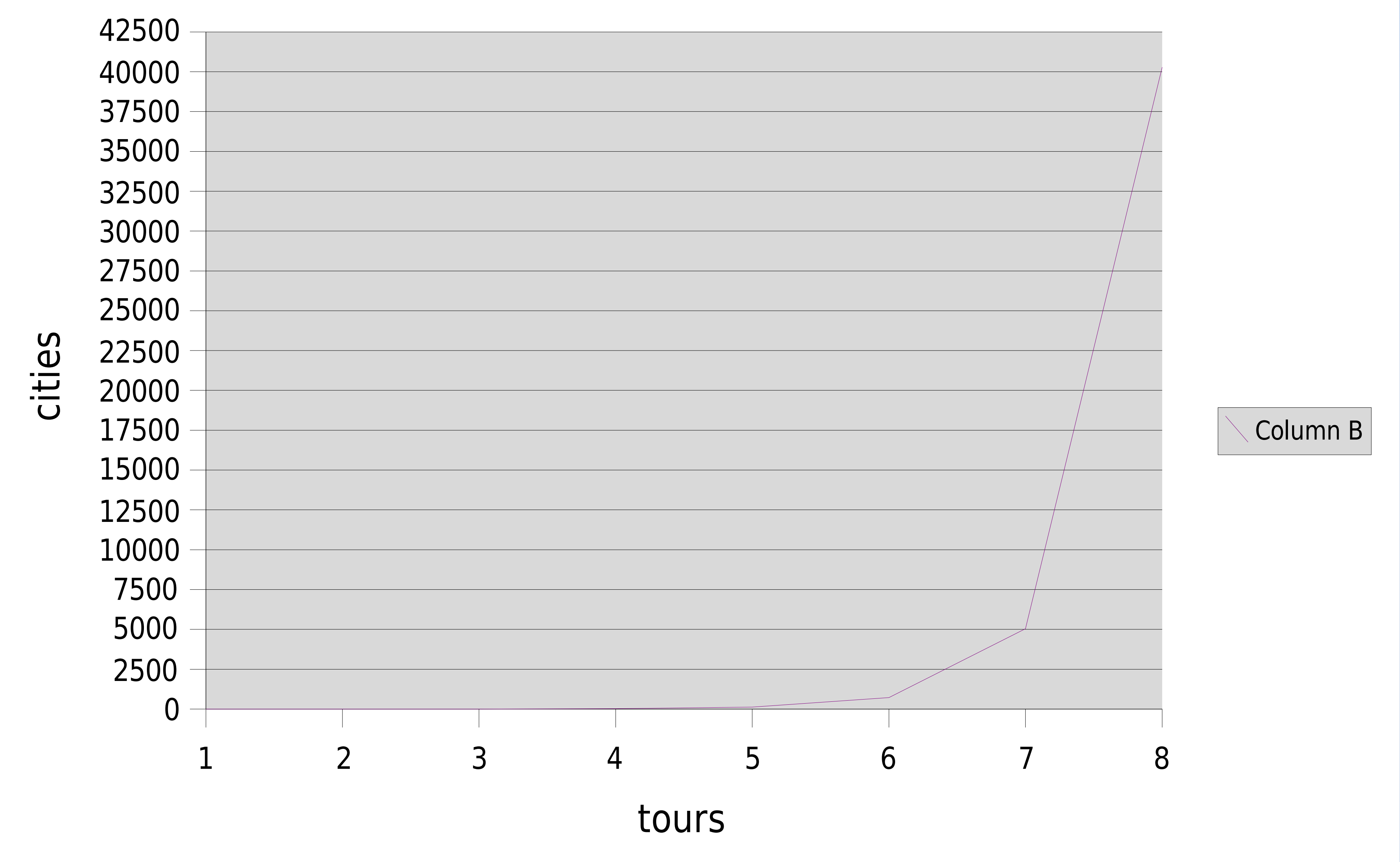
Given a set of n towns, the TSP can be stated as the problem of finding a minimal length closed tour that visits each town once.



Ant System for TSP

- euclidean distances: $d_{ij}=[(x_i-x_j)^2 + (y_i-y_j)^2]^{1/2}$
- problem is NP-Hard
- solutions = n!
- Brute force won't work!

Ant System for TSP



Ant System for TSP

- Each ant is a simple agent
- Each ant completes a tour
- At every city an ant chooses a non visited edge probabilistically (based on distance and pheromone)
- For each ant that completed a tour, the length is used to update the pheromone on the edges of that tour.
- After all ants have completed a tour the pheromone is evaporated

Ant System for TSP

Transition function

- The probability of choosing an edge between cities i and j for ant k at time t ($p^k_{ij}(t)$) is:
  - visibility η_{ij} is $1/d_{ij}$ where d is distance between i and j with i and j being cities
  - desirability at time t (trail) $\tau_{ij}(t)$ is the pheromone on the edge between i and j with i and j being cities
  - Two co-effecients α and β which control the relative importance of visibility and desirability
  - The sum of the probabilities of all the other eligible edges

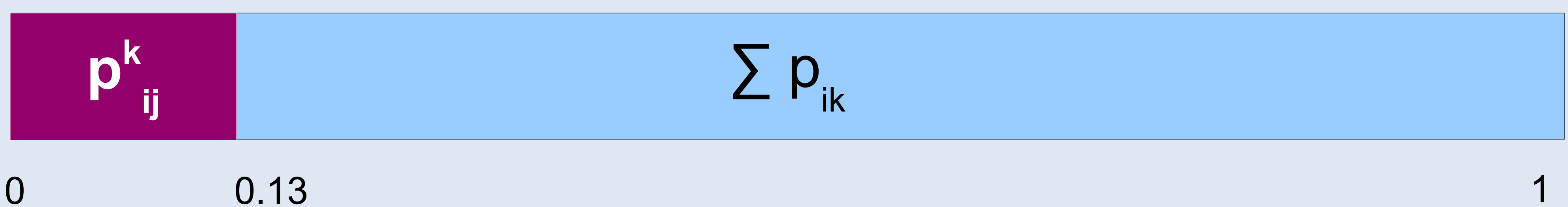
Ant System for TSP

Probability Transition Function

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

Ant System for TSP

probablity of moving to city i from city j in terms of the sum of the probabilities of moving to the other cities fromi



Ant System for TSP

Updating the pheromones

- For each ant that has completed a tour
  - Calculate the pheromone delta to be applied to each edge by normalising the tour length
  - for each edge in the tour apply the pheromone delta
- Evaporate the pheromone on each edge using a coefficient

Ant System for TSP

Updating the pheromones – Formal

New pheromone τ_{ij} at t+n is the pheromone at t $\tau_{ij}(t)$ multiplied by an evaporation coeffient ρ plus the pheromone delta $+\Delta\tau_{ij}$

i.e. $\tau_{ij}(t+n)=\rho.\tau_{ij}(t)+\Delta\tau_{ij}$

Ant System for TSP

Updating the pheromones – Formal

Pheromone delta

$$\Delta\tau_{ij} = \sum_{k=1}^{m} \Delta\tau_{ij}^{k}$$

where $\Delta\tau^{k}_{ij}$ is the quantity per unit of length of trail substance (pheromone in real ants) laid on edge (i,j) by the k-th ant between time t and t+n;

Ant System for TSP

Updating the pheromones – Formal

the pheromone delta for an ant k for the edge between i and j ($\Delta\tau^k_{ij}$)is given by:

$$\Delta\tau^k_{ij} = \begin{cases} Q / L & \text{if k - th ant uses edge (i, j) in its tour (between time t and t + n)} \\ 0 & \text{otherwise} \end{cases}$$

where Q is a constant and Lk is the tour length of the k-th ant.

Ant System for TSP

Implementing the ant system

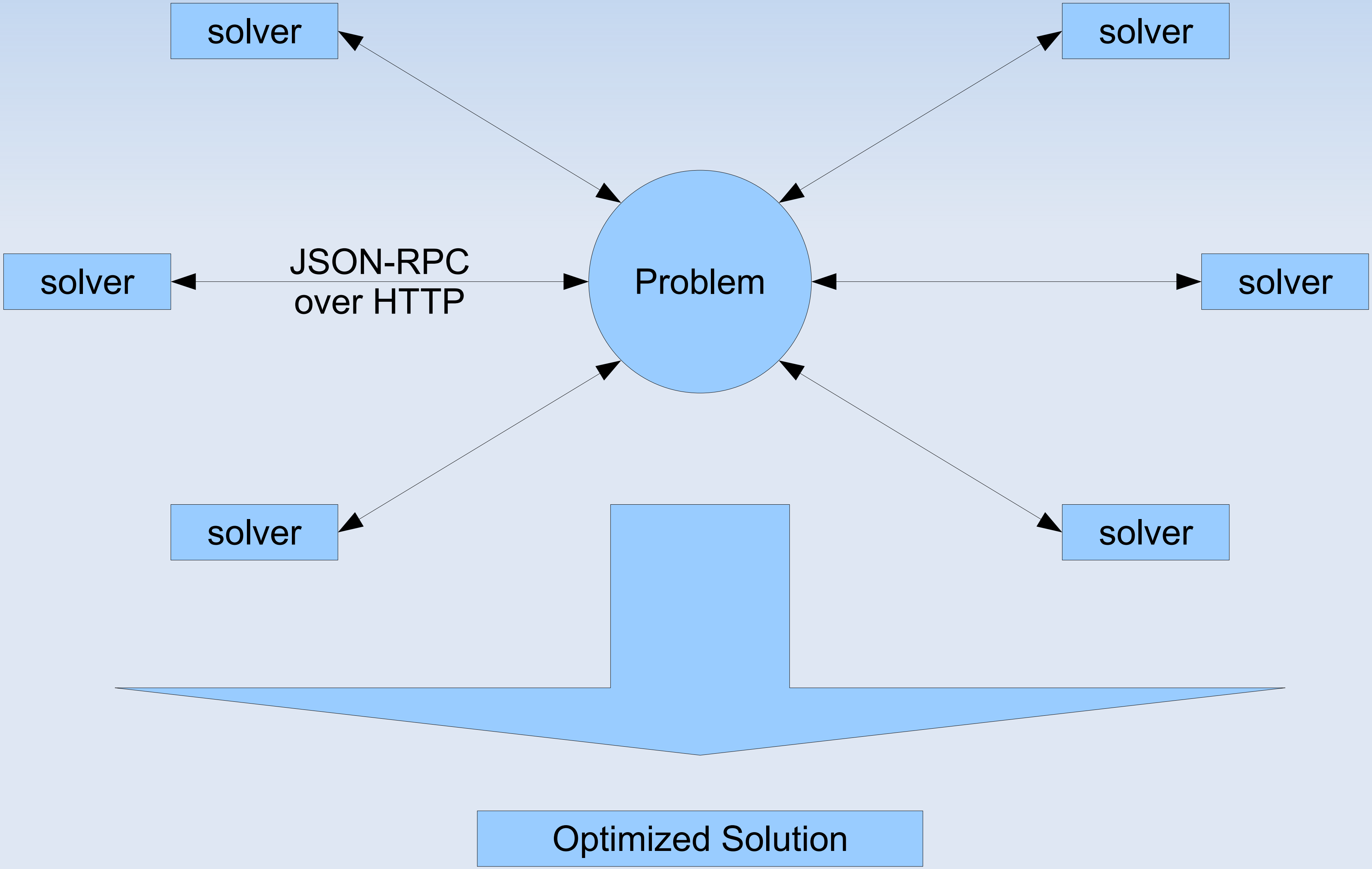
- Implemented in python, numpy for pheromones
- Used TDD
- Didn't pair on this
- First version was single threaded
- Then created a multi-threaded version
- Locks

Distributed TSP

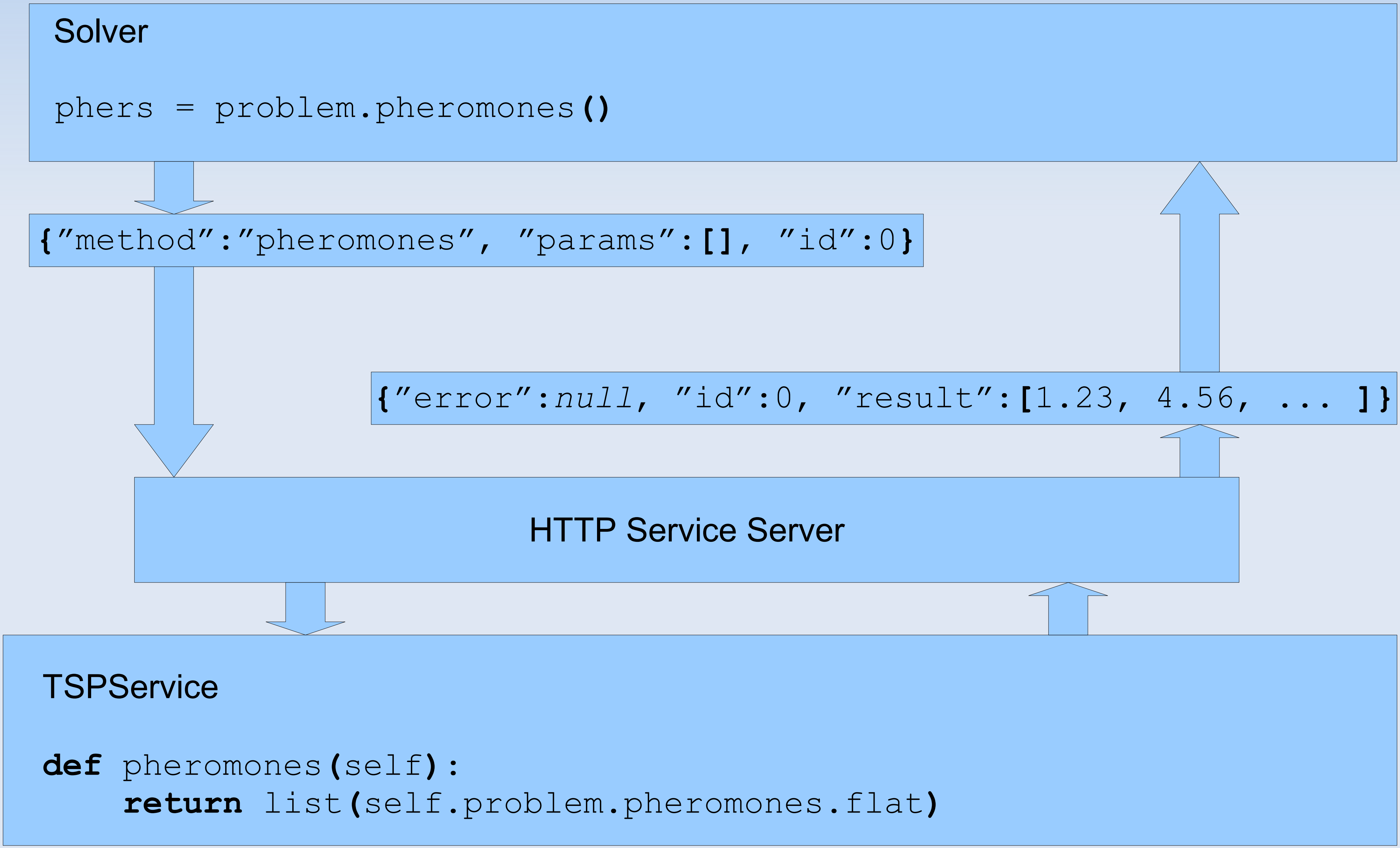
Distributed TSP

- Problem, Solvers, Solution
- Communication
- JavaScript TSP Solver
- JavaScript Challanges
- Considerations

Distributed TSP



JSON-RPC


Solver
phers = problem.pheromones()
{"method":"pheromones", "params":[], "id":0}
{"error":null, "id":0, "result":[1.23, 4.56, ...]}
HTTP Service Server
TSPService
def pheromones(self):
    return list(self.problem.pheromones.flat)

JavaScript TSP Solver

Why ?

JavaScript TSP Solver

Web 4.0 distributed computing!

JavaScript TSP Solver

- Gets pheromones, edges, cities from problem
- Calculates a single solution
- Submits solution to problem
- Repeats all steps above

PAAJSFDCICOP

- SSID: ants
- http://192.168.1.111:8000/tsp-js/tspclient.html
- use FireFox

JavaScript Challenges

- script code runs in the main and only thread in the browser
- Long running code blocks UI

- for(...) loops -> non blocking iterations
- Nested loops

- ”Multitasking” in JavaScript ?!

Considerations

- Python threads on multi core systems
- Communication overhead
- Data conversion overhead
- Effect of multiple Agents
- What is the performance of our algorithm
- Security (malicious agents)
- Optimization as a Services?

Conclusion

- Emergent systems are difficult
- Plenty of work left to do
- Commit to something and you'll get your ass into gear!
- If you have something to say submit a paper!
- Unittesting, SVN, Wiki are good even for a small research project
- Find and be aware of your limitations!
- It was great fun!

References

- http://ants.jsolait.net
- http://json-rpc.org
- http://jsolait.net
- http://jan.kollhof.net
- Swarm Intelligence (ISBN 0-19-513159-2)
  Bonabeau, Dorigo, Theraulaz
- The Ant System: Optimization by a colony of cooperating agents
  Marco Dorigo*,^, Member, IEEE, Vittorio Maniezzo%,^, and Alberto Colorni#
- Ant Colony Optimisation home page:
  http://iridia.ulb.ac.be/~mdorigo/ACO/ACO.html