Mehmet E. Aydin

Mehmet Emin Aydin

Collaboration in Multi Agent Systems and Learning Swarms

27 April 2019

ICATCES 2019, Alanya, Turkey

UWE Bristol | University of the West of England

Introduction

- Artificial Intelligence / Machine Learning is in rise due to compelling performance in solving large-scale problems.

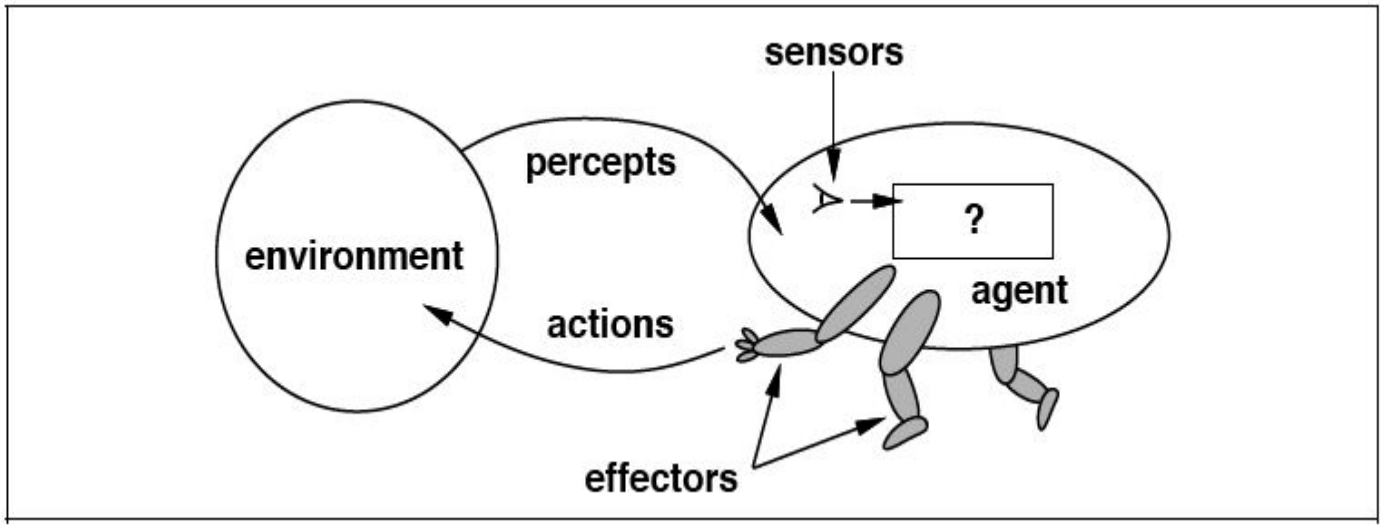
- Processing power and substantial increase in memory space encourages to attempt solving many complex and large-scale problems, which could not have been tackled with AI/ML in the past.
- Distributed AI has gained attention in 1990s, when the concept of intelligent agent was introduced. This approach helps:
  - Apply divide-and-conquer approach
  - Implement distributed, but collaborating intelligence
  - Avoid any prospective infrastructure limitations

Intelligent Agents and Multi-agent Systems

UWE Bristol | University of the West of England

Intelligent Agents

"An agent is anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors"
- (Artificial Intelligence: A Modern Approach by Stuart Russell and Peter Novig)



- An Agent is a piece of software designed to intelligently behave perceiving its environment through its sensors and acting accordingly through its actuators

Intelligent Agents

- Characteristics of agents:
  - Autonomy
    - An agent is expected to behave autonomously and independently.
  - Proactivity
    - An agent is able to initiate all of its activities regardless of external stimulus
  - Reactivity
    - An agent is able to react upon any stimulus risen from the environment
  - Social Ability
    - Agents are expected to interact and collaborate with other peer agents towards any objectives.

Multi-agent Systems

- A system consists of a number of homogenous or heterogeneous agents interacting for common objectives. (Social Ability)

- An agent needs to be able to corporate, compete and/or negotiate in order to successfully interact with the other peer agents.

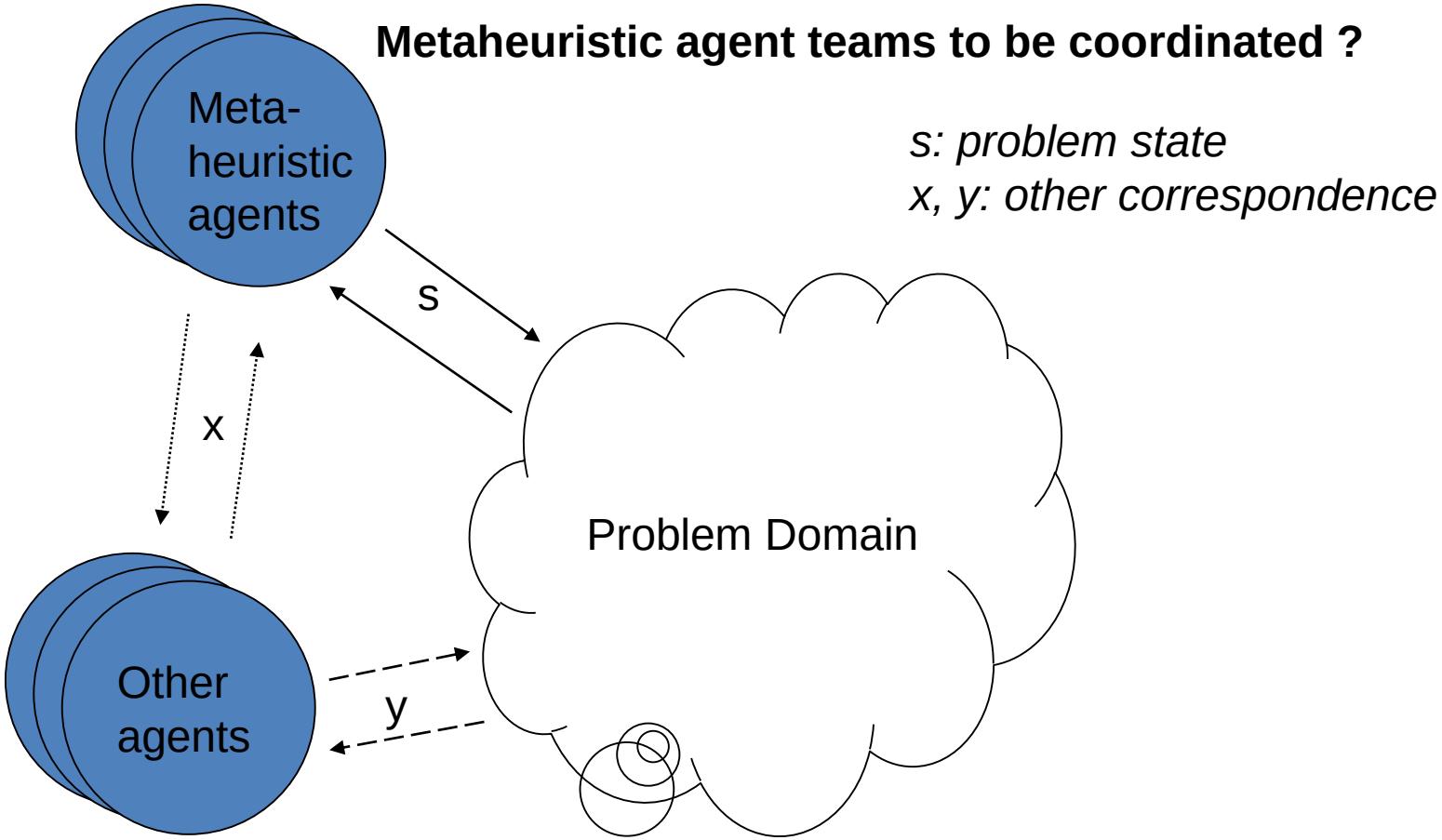
Multi-agent Systems

- **Agent Types:**
  - Agents are usually named based on their functionalities. That affects their classification.
  - Examples
    - Utility agent : Operates for some utility actions such as print out.
    - Learning agents: Agents devised with learning mechanisms such as NN, or Q learning
    - Moderator agents: Agents function as manager/negotiator etc.
    - Metaheuristic agents: Agents devised with heuristic search facilities

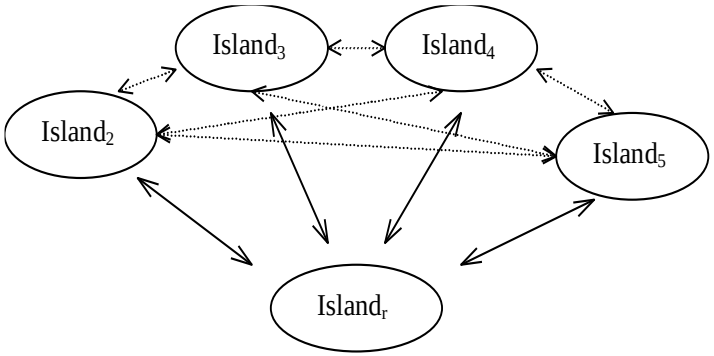
Metaheuristic-agent Teams

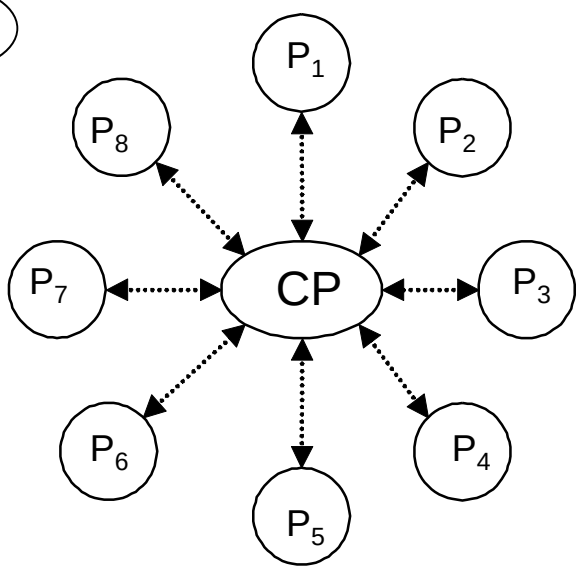
- Metaheuristics are computational methods /frameworks used iteratively to solve optimisation problems. (e.g. GA, SA, TS, etc.)
- Metaheuristic agent teams are multi-agent systems identified to describe teams of search agents used to operate for optimisation.
- They are specified to implement metaheuristics for solving large-scale optimisation problems.
- One of the major problems with metaheuristics is that there is no guarantee to find optimum solutions within reasonable time, as they usually provide with local optimum.
- One way to overcome this problem is to diversify the search conducted with the heuristics.

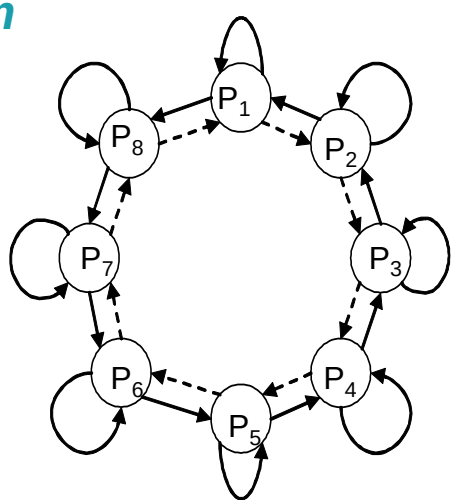
Metaheuristic Agents

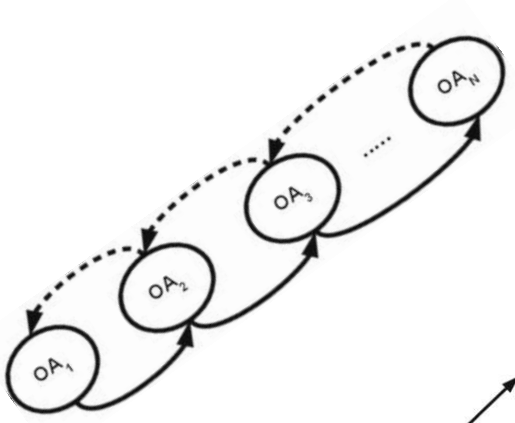


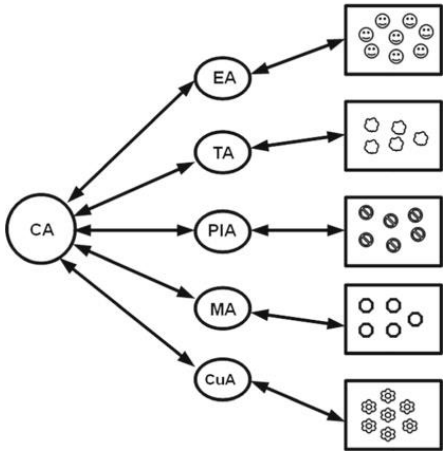
Collaboration is a problem

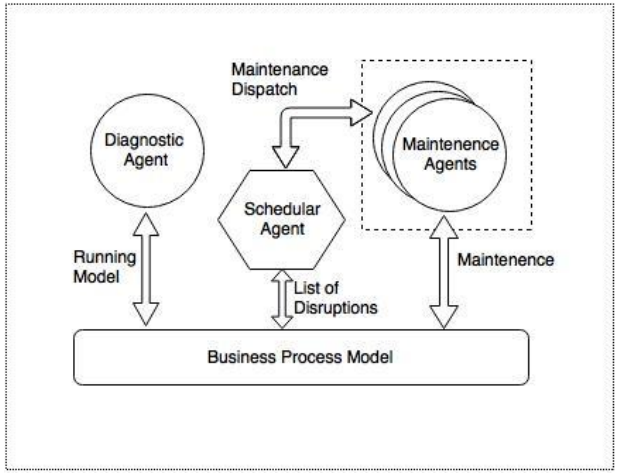












UWE Bristol | University of the West of England

Metaheuristic Agents

- Coordinating metaheuristic agents
  - Coordination languages/protocols
    - Linda, ACL etc.
  - MAS Middleware
    - JADE, DREAM etc.
  - Architecture based coordination
    - Possible Architectures
      - Multi Island organisations
      - Centralist organisations
      - Ring Topologies
  - Swarm Intelligence
    - ACO, PSO, BCO etc.

Collaboration with Swarm Intelligence

UWE Bristol | University of the West of England

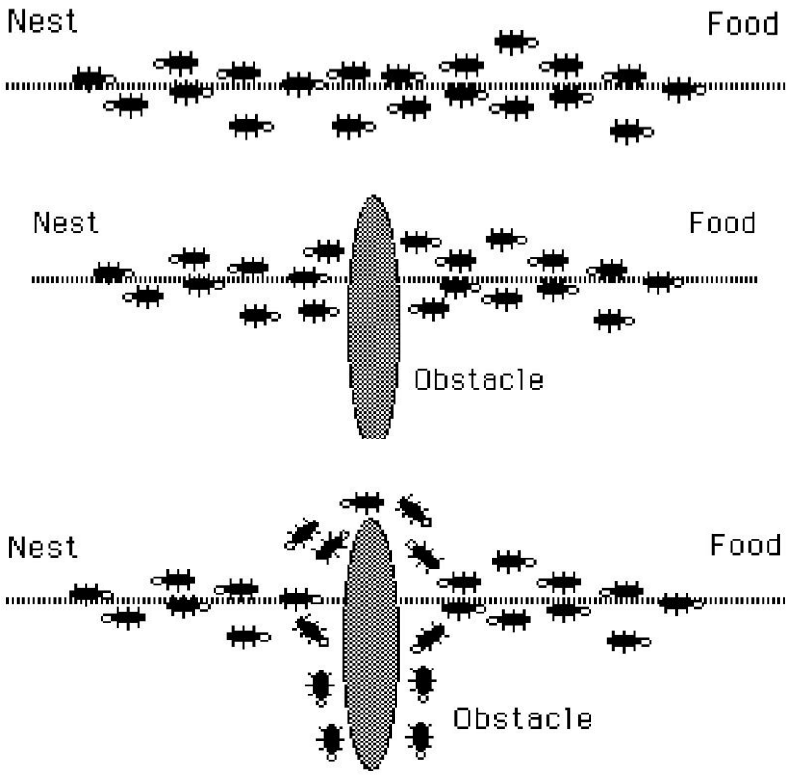
Swarm Intelligence

- Swarm Intelligence is a nature-inspired discipline of modelling and developing collective behaviours of decentralised, self-organising independent entities.
- It imposes harmonizing the agents so as to collaborate in developing collective behaviours/intelligence.
- Entities/agents are working towards global aims with local information/action and a particular scheme of interaction.
- Agents/entities work in populations such as colonies, flocks etc.

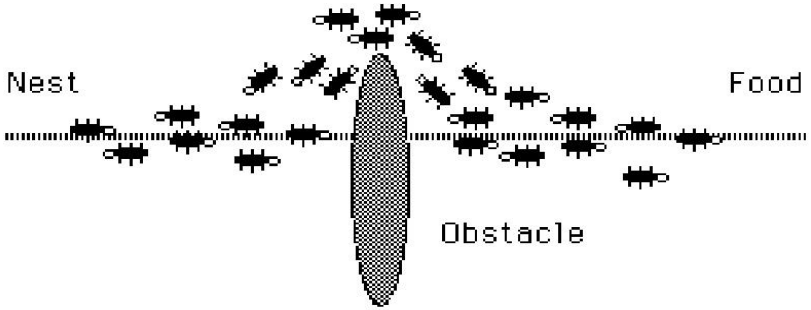
Swarm Intelligence

- Characteristics:
  - Self-Organizing; paths to the solution emerges rather than predefined.
  - Decentralization; no central control or leadership needed for the colony
  - Flexibility; the colony can look at internal perturbations and external changes
  - Robustness; tasks are completed even if some individuals fail.

Swarm Intelligence



- Ant colony optimization
- Ant colonies (society)



Swarm Intelligence

Particle Swarm Optimization
(Bird flocks)

Leading bird is followed to gain the food

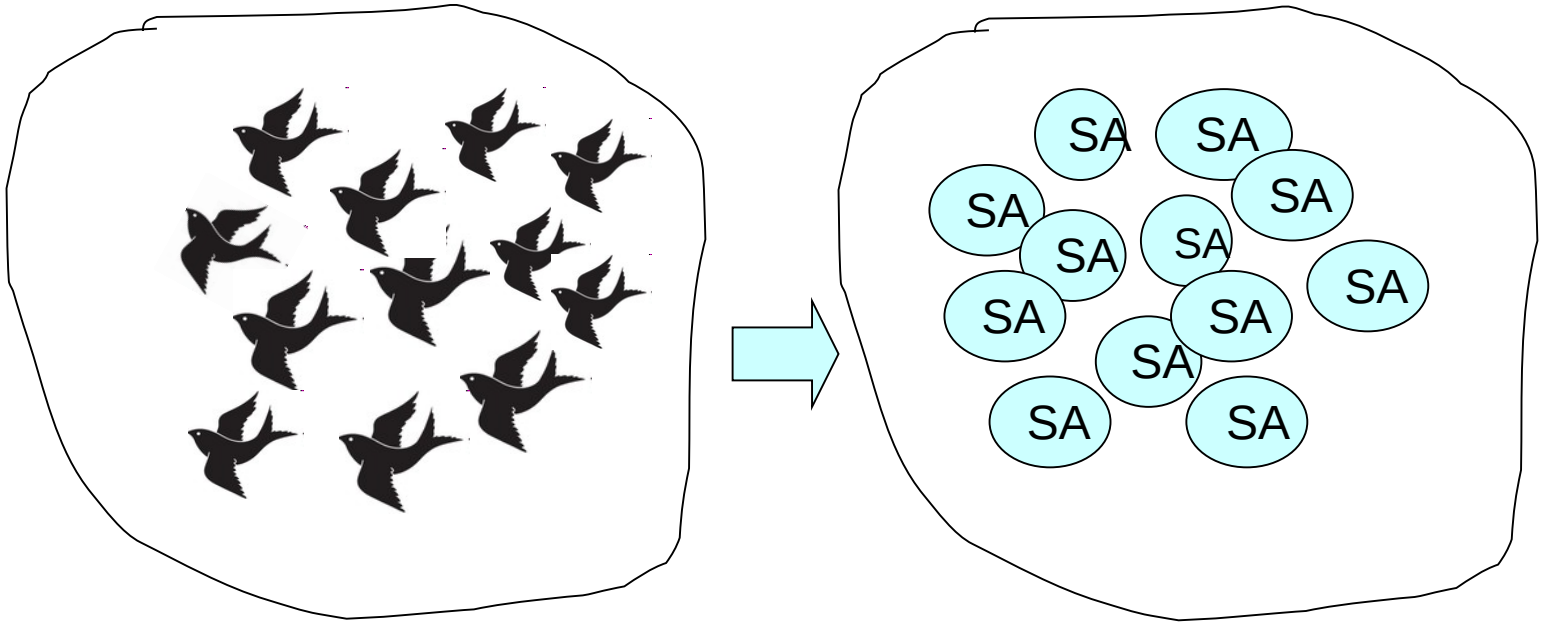


Birds share the best experience towards the destination

UWE Bristol | University of the West of England

A Swarm of SA

- Simulated Annealing (**SA**) is a probabilistic search framework used for solving optimisation problems



How to organise the swarm of SA? **PSO**

Particle Swarm Optimisation (PSO)

- PSO is originally devised by Kennedy, Eberhart (1995) and was first intended for simulating social behaviour.
- The algorithm was simplified and it was observed to be performing optimization.
- Best known information is considered and exchanged
- Cognitive and Social skills are accounted in interaction
- Key roles are played for self organisation.

Particle Swarm Optimisation (PSO)

A population based algorithms made of particles, each is defined with two key vectors;

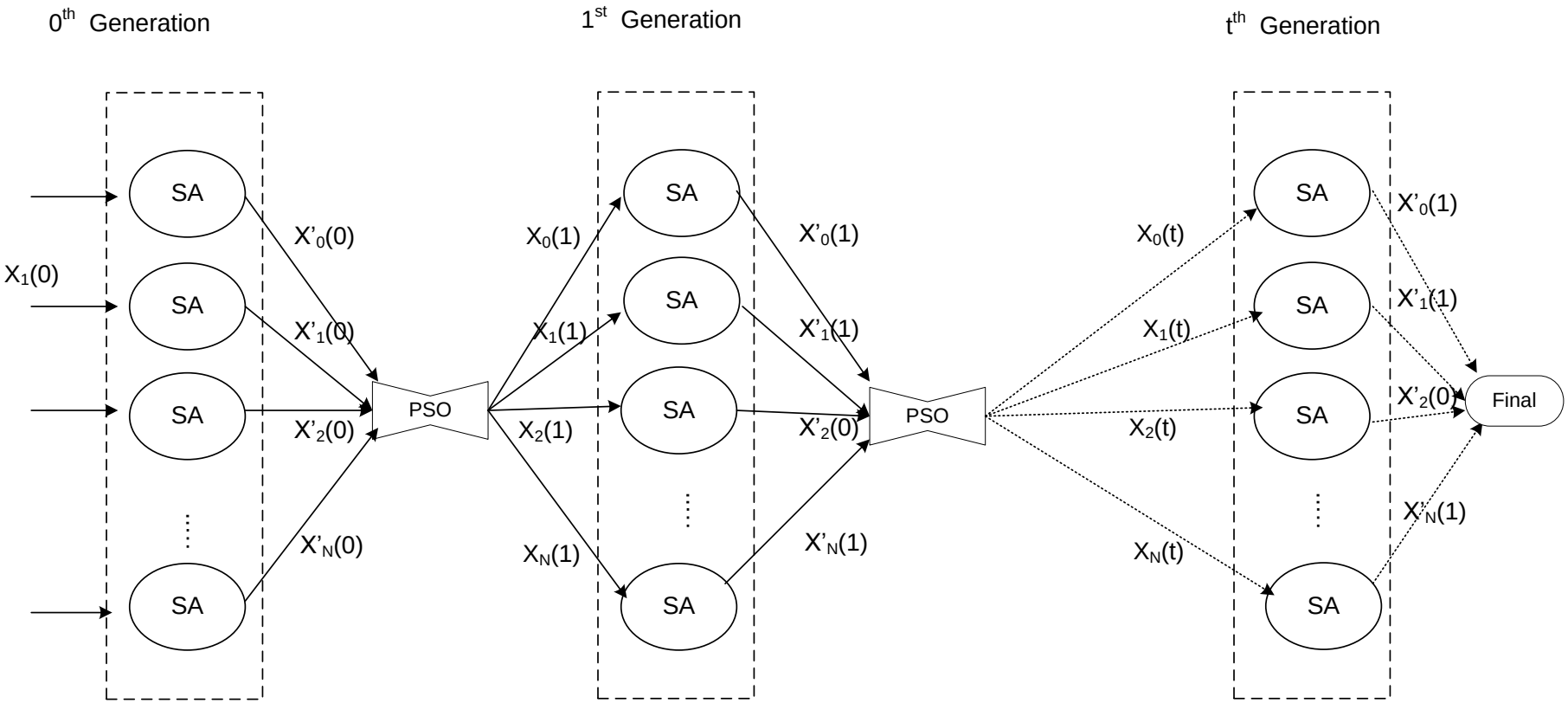
position (X_i), velocity (V_i) $i=1,2,\ldots,popSize$

a personal best (P_i)*, and global best* (G_i) *for each particle,*

1. Initialise ***X*** and ***V***
2. Repeat the following steps until STOP is true
   a. Check the personal best (*P*)
   b. Check the global best (*G*)
   c. Update the velocity (*V*)
   d. Update the position (*X*)

Swarm of SA agents

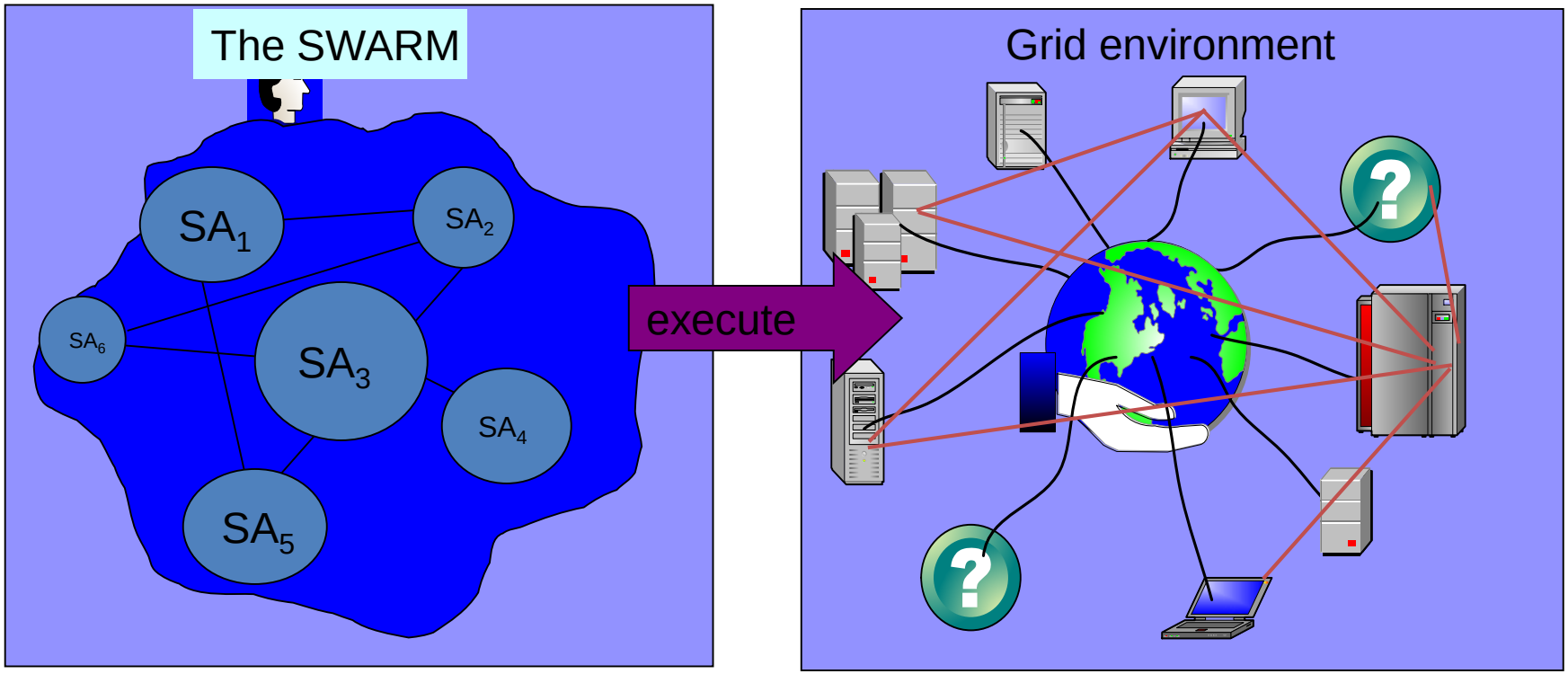
The whole process is computationally very expensive



- Each SA agent takes one solution and tries to improve it, then, submits to PSO for collaboration

UWE Bristol | University of the West of England

Swarm of SA agents



- Distributed objects are used for development
- POPC++ is a distributed object based programming tool.
- Once developed, the system deployed on compute cluster.

UWE Bristol | University of the West of England

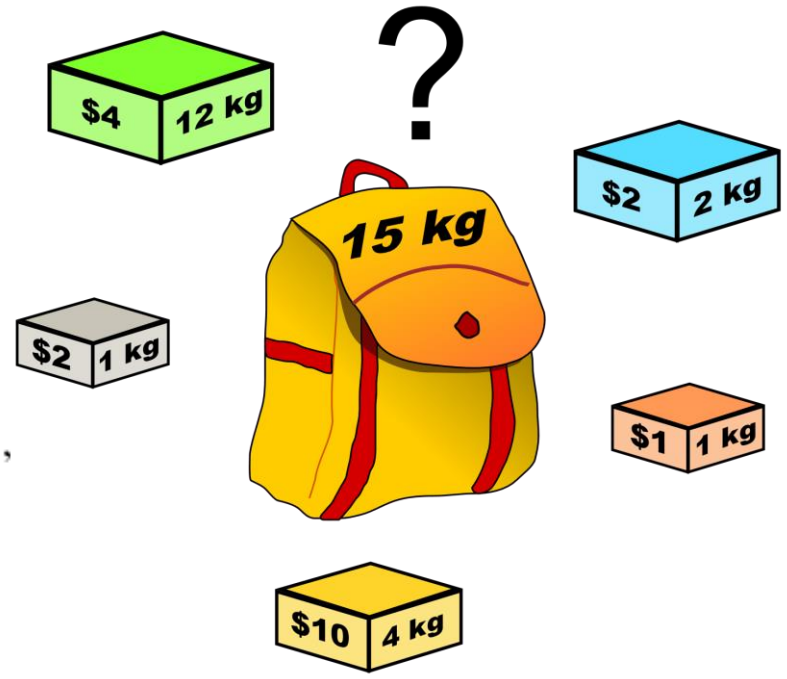
An Application

- Multidimensional Knapsack Problem is an NP-Hard combinatorial problem requires optimising the total gain of items fit in each of a number of knapsacks.

$$\text{maximise} \quad \sum_{j=1}^{n} p_j x_j,$$

$$\text{subject to} \quad \sum_{j=1}^{n} r_{ij} x_j \leq b_i, \quad i = 1, \ldots, m,$$

$$x_j \in \{0, 1\}, \quad j = 1, \ldots, n.$$



Numerical Results

Swarm Size	ESA		BCO		PSO	
	MKP6	MKP7	MKP6	MKP7	MKP6	MKP7
5	0.03495	0.03748	0.02808	0.04077	0.00257	0.00307
10	0.01183	0.02170	0.02021	0.03270	0.00214	0.00175
15	0.00899	0.01528	0.01694	0.02782	0.0017	0.00112
20	0.01052	0.01407	0.01530	0.01906	0.00203	0.00064
30	0.00762	0.00961	0.01344	0.01516	0.00098	0.00014
40	0.00768	0.00821	0.01226	0.01736	0.00122	0.00030
50	0.00633	0.00865	0.01093	0.01979	0.00061	0.00028
Mean	**0.01256**	**0.01643**	**0.01674**	**0.02467**	**0.00161**	**0.00104**

BCO: Bee Colony Optimization
ESA: Simple Replacement
PSO: Particle Swarm Optimization
Agent teams are **homogenous**

} Coordination policies

UWE Bristol | University of the West of England

Numerical Results

	Swarm Size	TEAM		TS		VNS	
		%	CPU	%	CPU	%	CPU
MKP 6	30	0.27	3.01	0.39	2.75	1.12	2.67
	20	0.35	2.14	0.52	2.06	1.16	1.93
	10	0.49	1.23	0.64	1.17	1.40	1.19
MKP 7	30	0.63	3.24	0.91	2.98	1.99	3.08
	20	0.84	2.34	1.16	2.22	2.21	2.21
	10	0.96	1.29	1.32	1.24	2.49	1.25

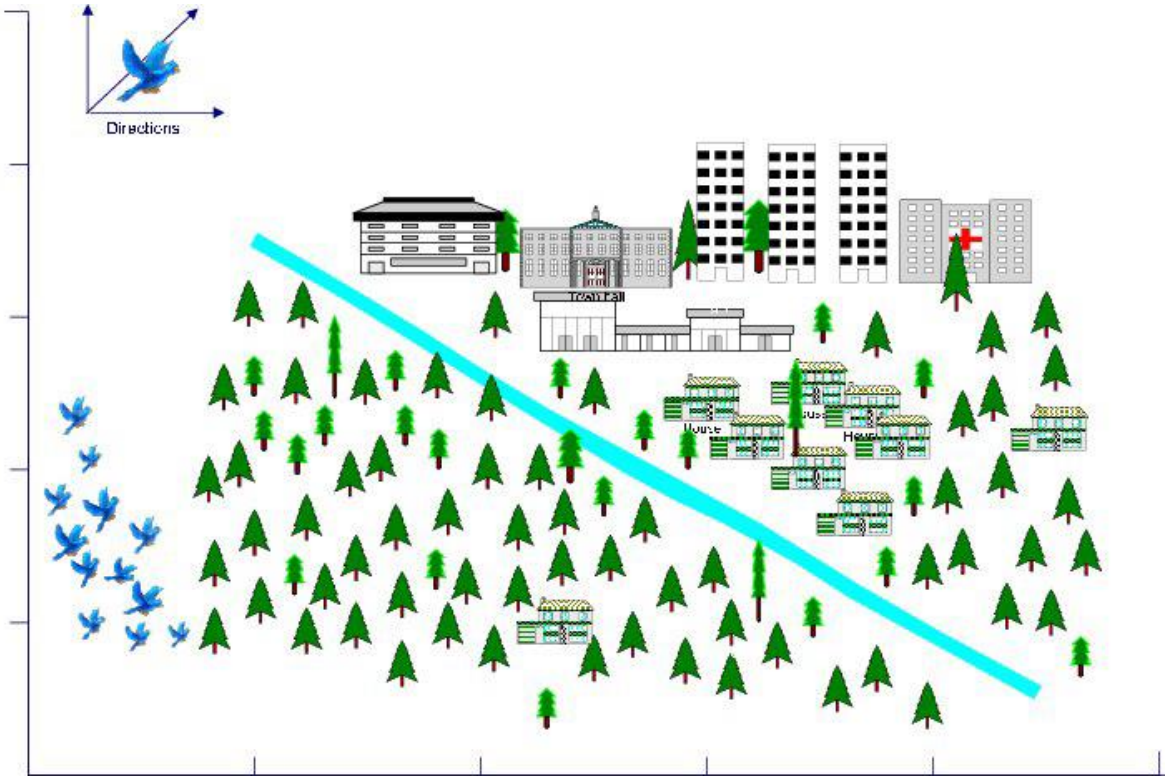
TS: Homogenous team of *tabu search* (TS) agents
VNS: Homogenous team of *variable neighbourhood search* (VNS) agents
TEAM: Heterogeneous team of TS and VNS agents
PSO is the coordination policy applied all cases.

- *Collaboration among the agents have been customised to make it fit in the circumstances of the domain*
- *Swarms are also pre-conditioned in this respect.*
- *Can an adaptive approach be developed to coordinate MAS and swarms*

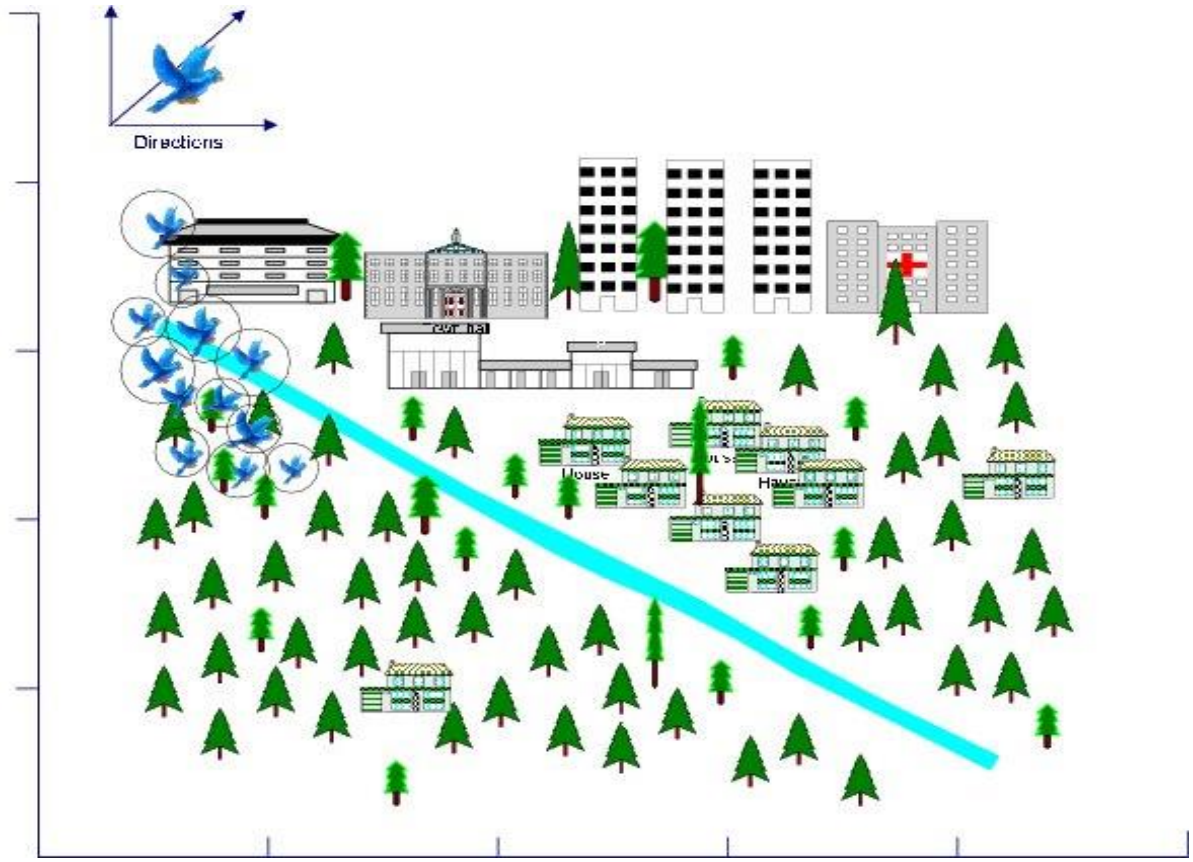
Learning Swarms

UWE Bristol | University of the West of England

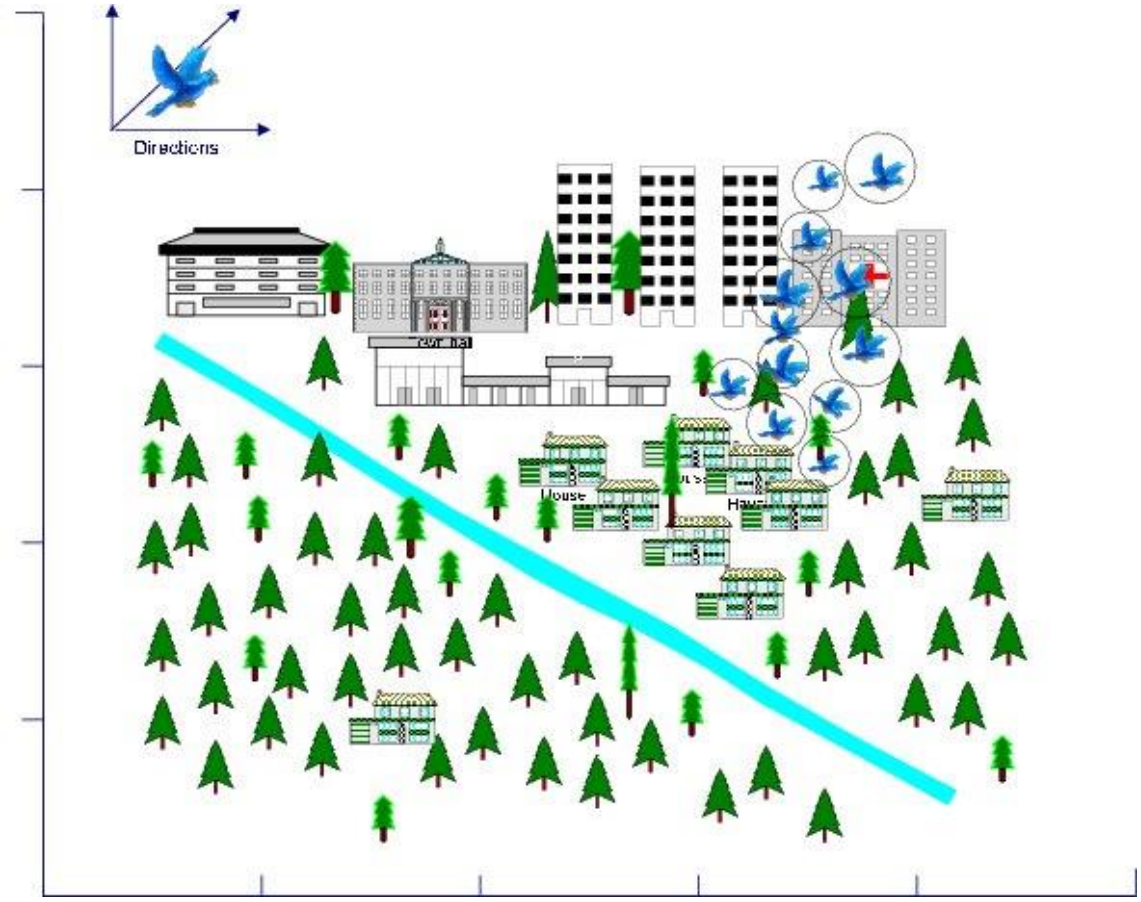
Scenario (initial stage)



Scenario (…early stages)



Scenario (a mature stage)



UWE Bristol University of the West of England

Introduction

- This is a proof-of concept study for demonstrating the viability of building collaboration among multiple agents through standard Q learning algorithm embedded in particle swarm optimisation.

- It presents a combination of two computational paradigms to manage building collaboration among multiple agents.

- Multi-agent systems (MAS) is an up-to-date artificial intelligence paradigm , which attracts much attention for modelling intelligent solutions in rather a distributed form. It imposes formulating limited capacity items as proactive and smart entities, which autonomously act and accumulate experience to exploit ahead in fulfilling duties more and more efficiently. In this way, a more comprehensive and collective intelligence can be achieved. This paradigm has proved success so many times in a wider problem solving horizon

Collaboration in MAS

- Collaboration of the agents remains as one of crucial problem within the field.
- Various approaches are proposed so far; depending on the nature of the problem to solve, the agents can be organised in various ways:
  - Hierarchical, static, blind
  - Flat, dynamic, and informed
  - Learning, deliberative etc.
- Collaboration requires agreement and conscious of knowledge exchange (approximation to equilibrium)

Reinforcement Learning

- Reinforcement learning (RL) is a class of learning in which unsupervised learning rules work alongside with a reinforcement mechanism to reward an agent based on its action selection activity to respond the stimulus from its own environment. It can be also called as semi-supervised learning since it receives a reinforcement point, either immediate or delayed, fed back from the environment.

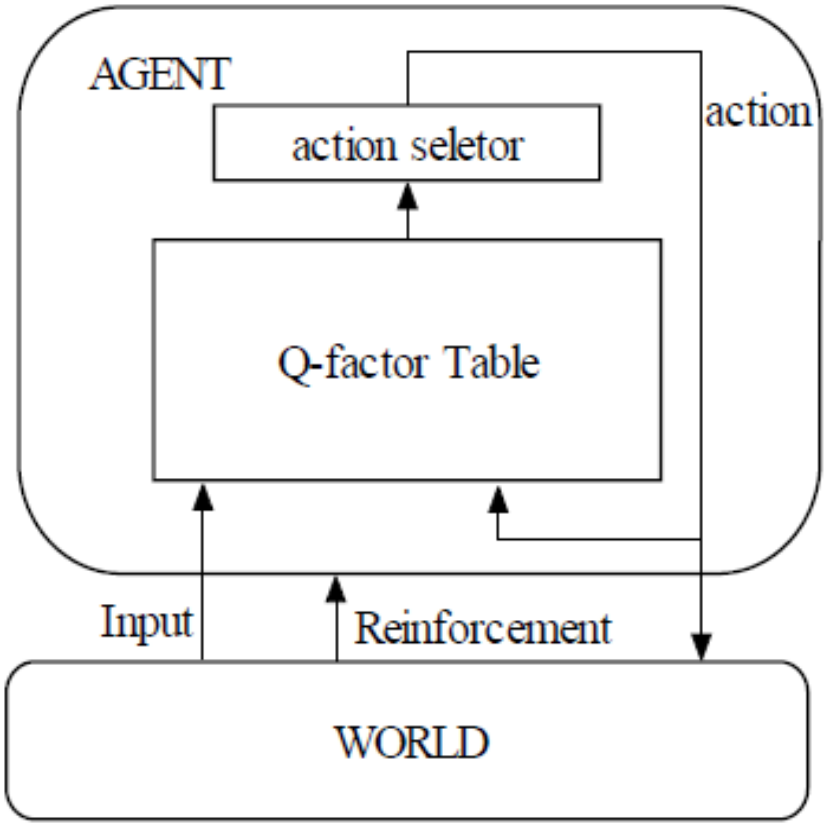
- The idea is to produce rewards for actions taken by the agents given the circumstances of the environment.

$$\mathbf{R} : S \times A \longrightarrow \mathbb{R}$$

$$\max \mathbf{R} = \sum_{i=1}^{\infty} r_i$$

Q Learning

- Q learning is a reinforcement learning algorithm that is developed based on temporal-difference handled with asynchronous dynamic programming.
- It provides rewards for agents with the capability of learning to act optimally in Markovian domains by experiencing the consequences of actions, without requiring them to build map of the respective domain.



Q Learning

- The main idea behind Q learning is to use a single data structure called the utility

$$Q(x,a) = E(r + \gamma e(y)|x,a)$$

$$e(y) = \max\{Q(y,a)\} \qquad for \quad \forall a \in A$$

$$Q^{t+1}(x,a) = Q^t(x,a) + \beta(r + \gamma e(y) - Q^t(x,a))$$

Particle Swarm Optimisation (PSO)

- PSO is a population-based optimization technique inspired of social behaviour of bird flocking and fish schooling. PSO inventors have implemented such natural processes to solve the optimization problems in which each single solution, called a particle, joins the other individuals to make up a swarm (population) for exploring within the search space. Each particle has a fitness value calculated by a fitness function, and a velocity of moving towards the optimum. A basic PSO algorithm builds each particle based on, mainly, two key vectors; position vector,

$$
\begin{aligned}
x_{i,k}(0) &= x_{min} + (x_{max} - x_{min}) \times r_1, \\
v_{i,k}(0) &= v_{min} + (v_{max} - v_{min}) \times r_2, \\
\mathbf{v_i}(t+1) &= \delta w_t \Delta \mathbf{v_i}(t) \\
\Delta \mathbf{v_i} &= c_1 r_1 (\mathbf{y_i}(t) - \mathbf{x_i}(t)) + c_2 r_2 (\mathbf{g}(t) - \mathbf{x_i}(t)) \\
\mathbf{x_i}(t+1) &= \mathbf{x_i}(t) + \mathbf{v_i}(t+1)
\end{aligned}
$$

PSO + Q Learning

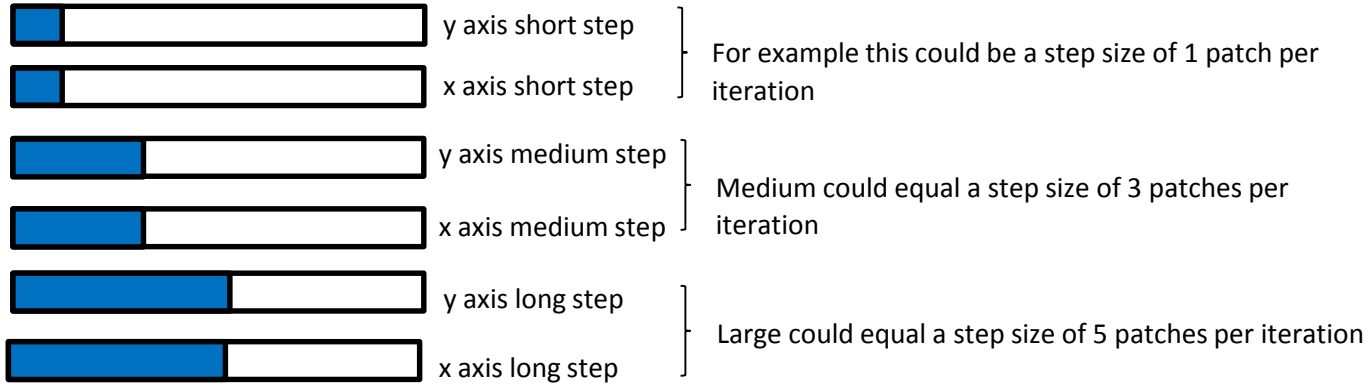
- Since the swarm intelligence framework preferred in this study is PSO, each individual to form up the swarm will be identified as a particle as is in particle swarm optimisation.
- Q learning is embedded within PSO in the following way:
  - A set of actions will be available to select each time: $\boldsymbol{\Delta} = \{\delta_j | j = 1, ..., A\}$
  - Then, the opted action will produce a results: $f(Q, x_i, a) = \pi\delta_j$
  - Finally, the positions will be updated as follows:

$$\mathbf{x_i}(t+1) = \mathbf{x_i}(t) + f(Q, x_i, a)$$
$$f(Q, x_i, a) = \{\hat{x}_i | \max[Q(x_i, a)] \quad for \quad \forall a \in A\}$$

Simulations

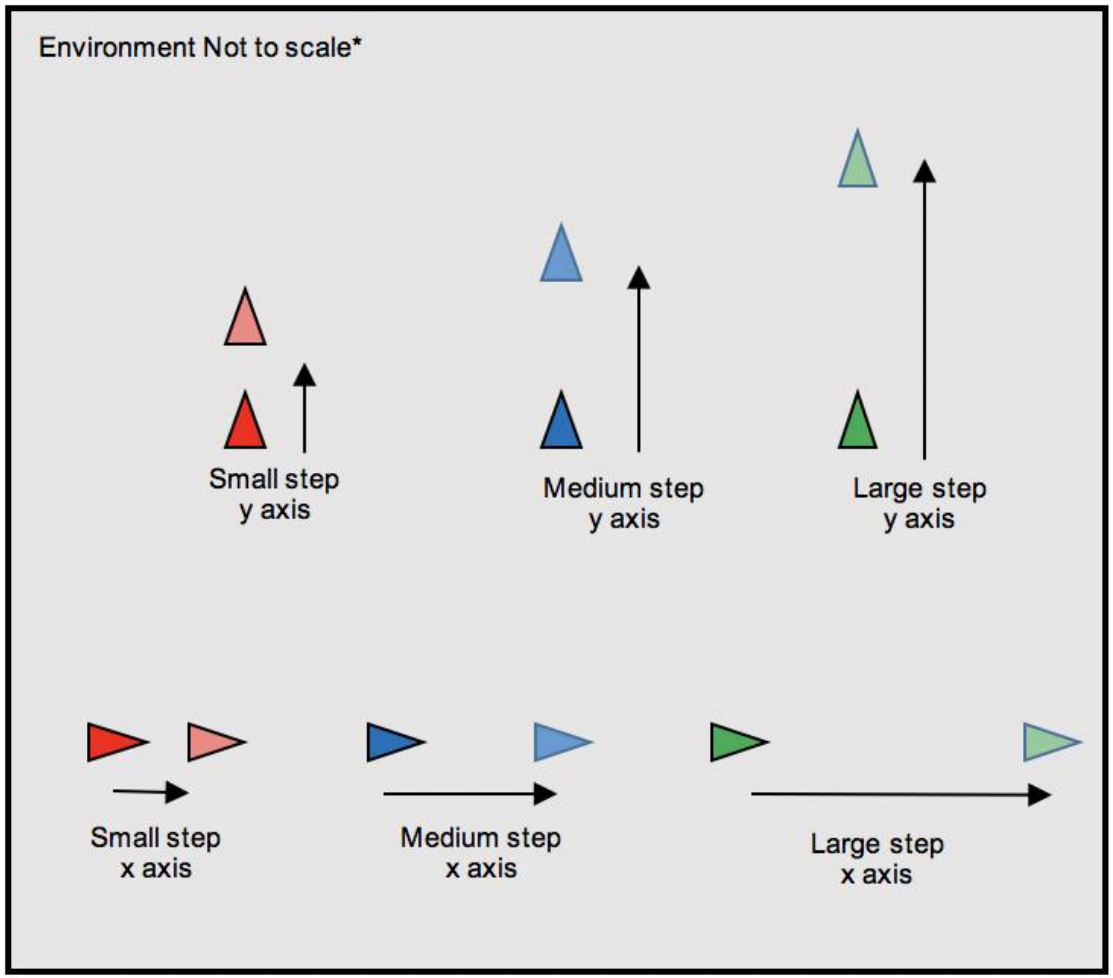
- A simulation was developed using an agent-based programming/simulation language, so-called NetLogo, to simulate the previously introduced scenario:

- The set of actions to take are designed as follows:



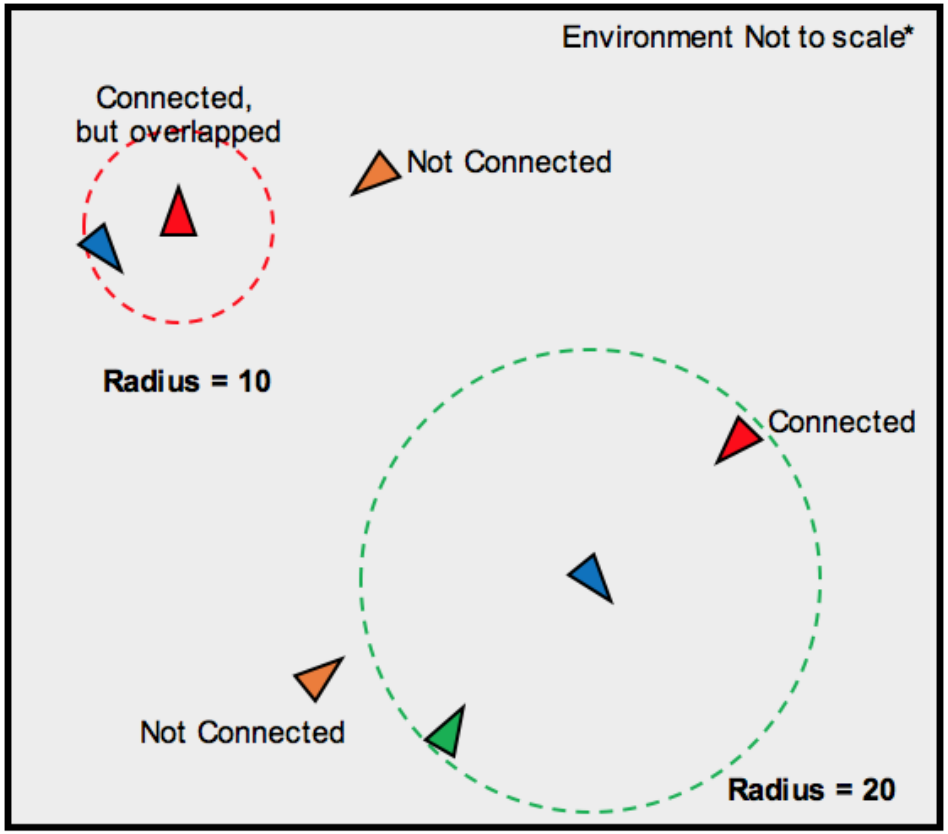
Simulations

1. Short step along the x axis
2. Short step along the y axis
3. Medium step along the x axis
4. Medium step along the y axis
5. Large step along the x axis
6. Large step along the y axis

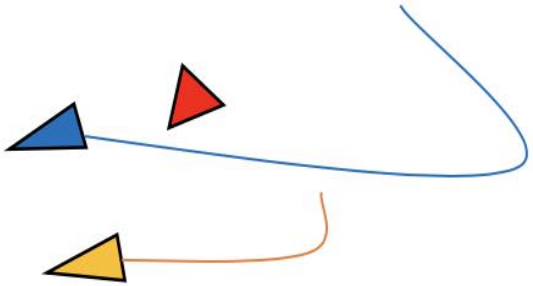


Simulations

- Assumptions:
  - remain connected
  - not overlapped,
  - Not violating each-other's territories
  - Once disconnected, will be punished
  - Otherwise rewarded

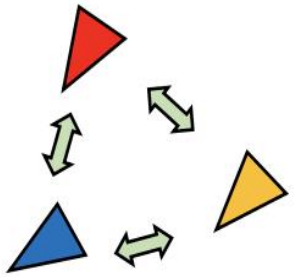


Simulations



Standard PSO movement



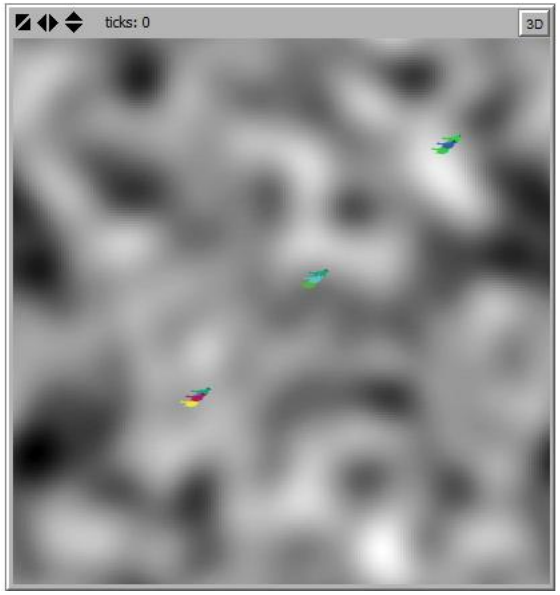


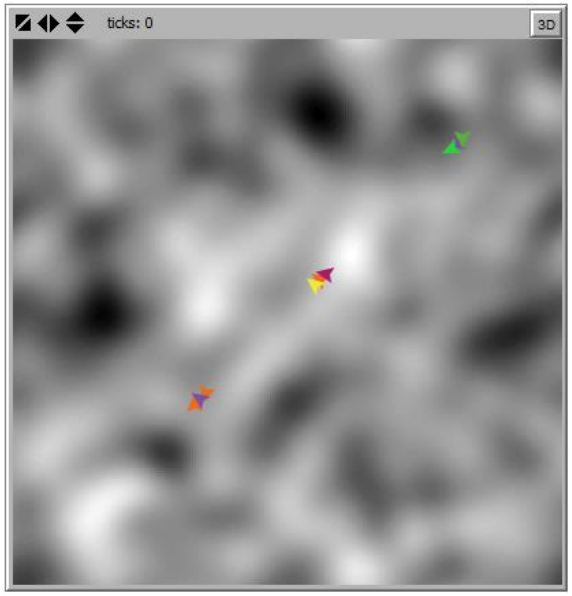
With Q-Learning Integration

$$r = \begin{cases} 100, & \text{if} \quad \sum_{k=1}^{N_i} d(\mathbf{x_i}, \mathbf{x_k}) = N_i\epsilon \\ N_i\epsilon - \sum_{k=1}^{N_i} d(\mathbf{x_i}, \mathbf{x_k}), & \text{if} \quad N_i\epsilon > \sum_{k=1}^{N_i} d(\mathbf{x_i}, \mathbf{x_k}) \\ -100, & \text{if} \quad \sum_{k=1}^{N_i} d(\mathbf{x_i}, \mathbf{x_k}) \leq 0 \end{cases}$$

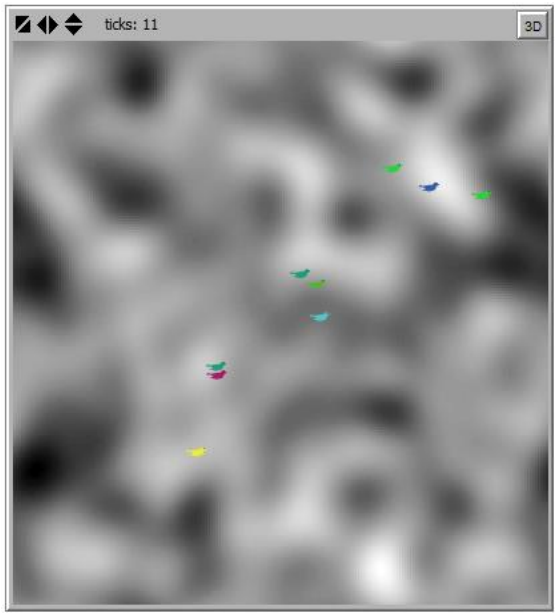
$$N_i = \{\mathbf{x_k} |\ \epsilon > d(\mathbf{x_i}, \mathbf{x_k})\} \quad \forall k \in M$$

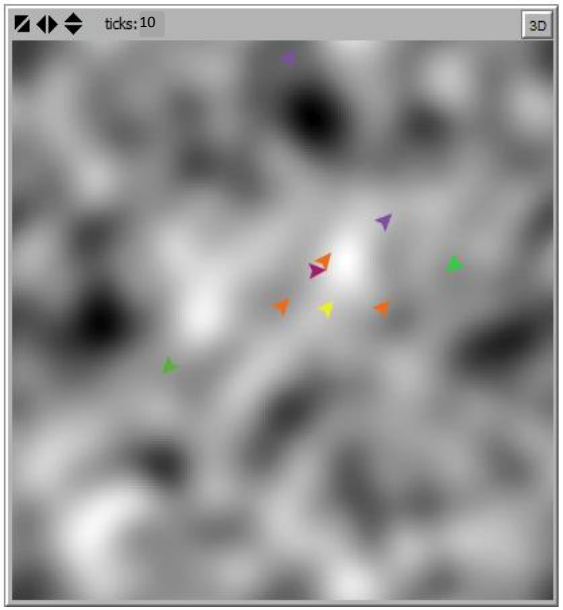
Experimental Results (Initial)



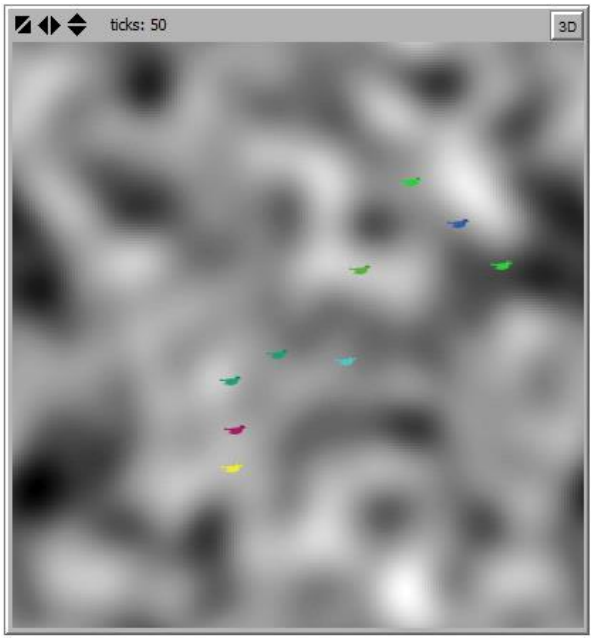


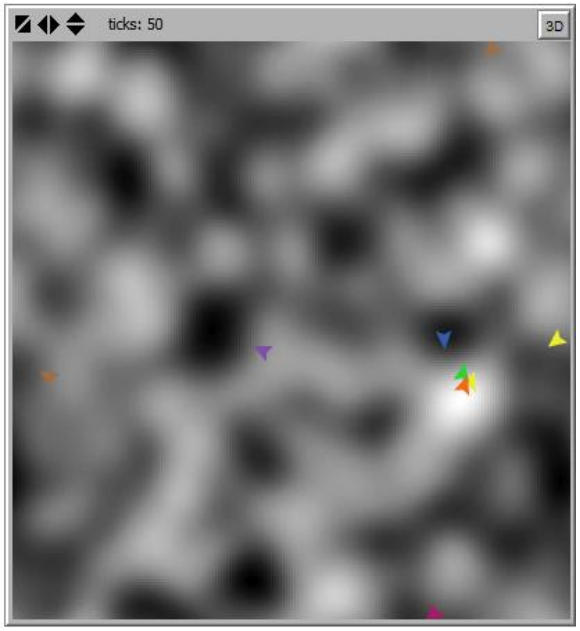
Results (After 10 iterations)



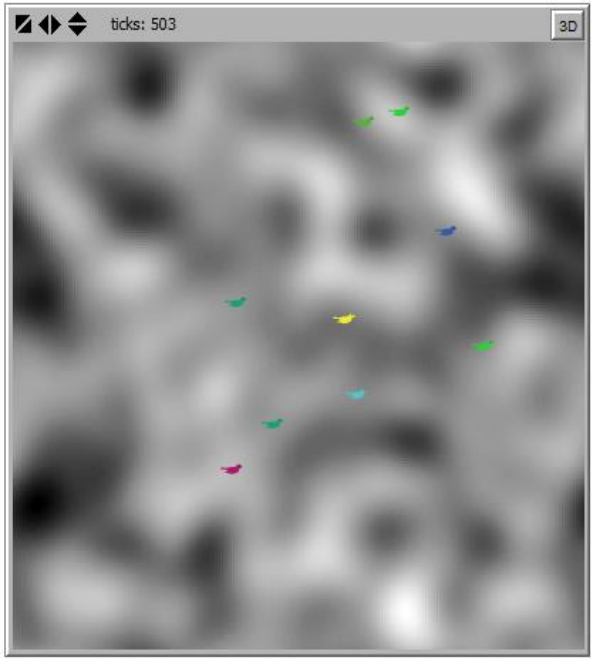


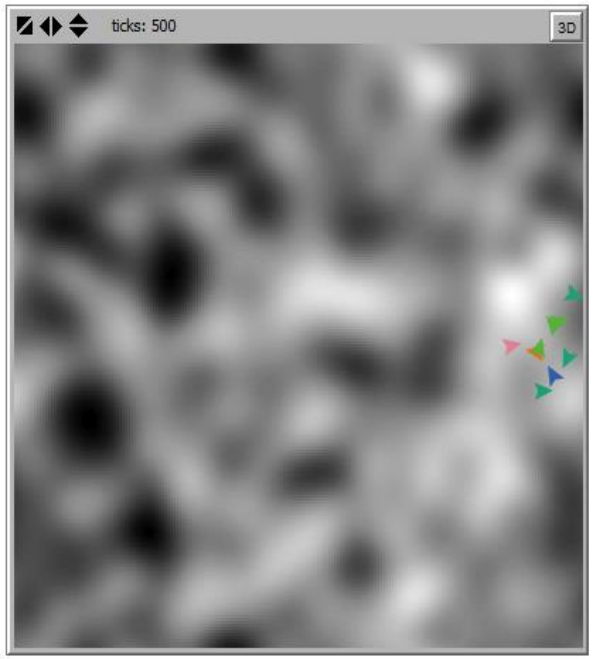
Results (50 iterations)





Results (500 iterations)





Conclusions

- This study demonstrated that multi agent systems can be self-organised through Q learning algorithm, where PSO framework has been used to let individuals be implemented.

- Although the number of actions designed limited, the study has provided a proof-of-concept.

- More complicated set of actions would require further structural bases to enhance Q learning (reinforcement learning) algorithms for implementation and sustaining, which is the primary future work if this study.

Questions?