Multi-Agent Systems on Sensor Networks: A Distributed Reinforcement Learning Approach

Chen-Khong Tham & Jean-Christoph Renaud

Presented by Howard Luu

Presentation Structure

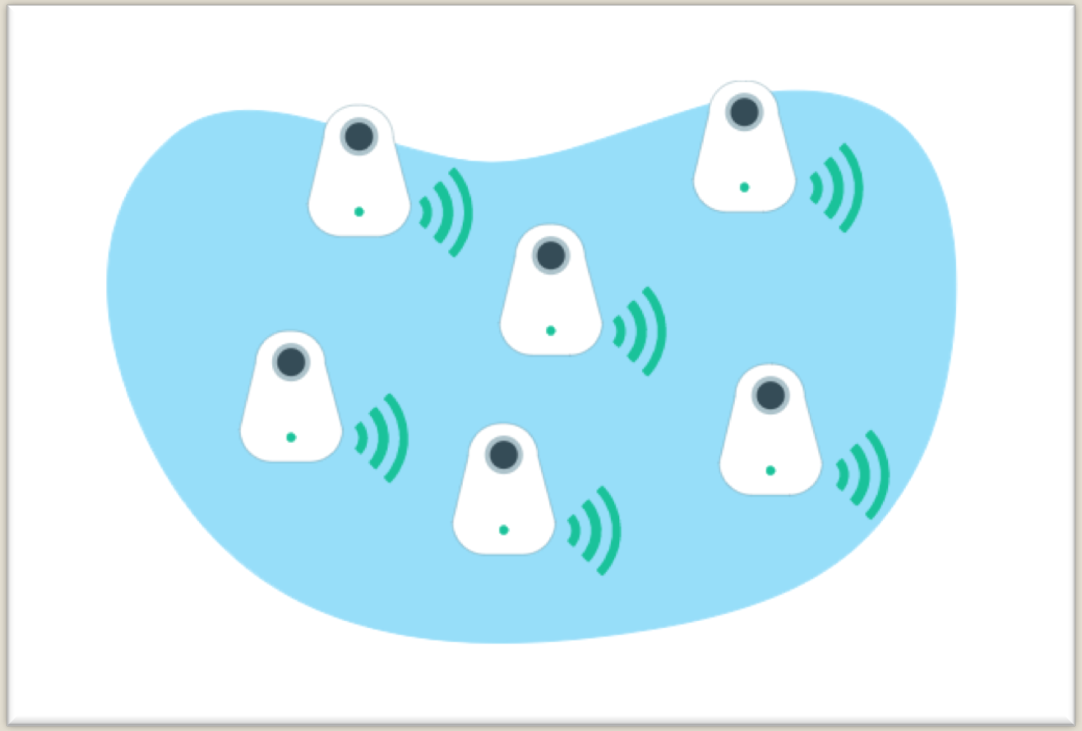
- Paper Objective
- Background
  - Multi-Agent Systems (MAS) & Wireless Sensor Networks (WSN)
  - Reinforcement Learning
- Distributed Reinforcement Learning (DRL) Approaches
- Simulations and Results
- Conclusion

Paper Objective

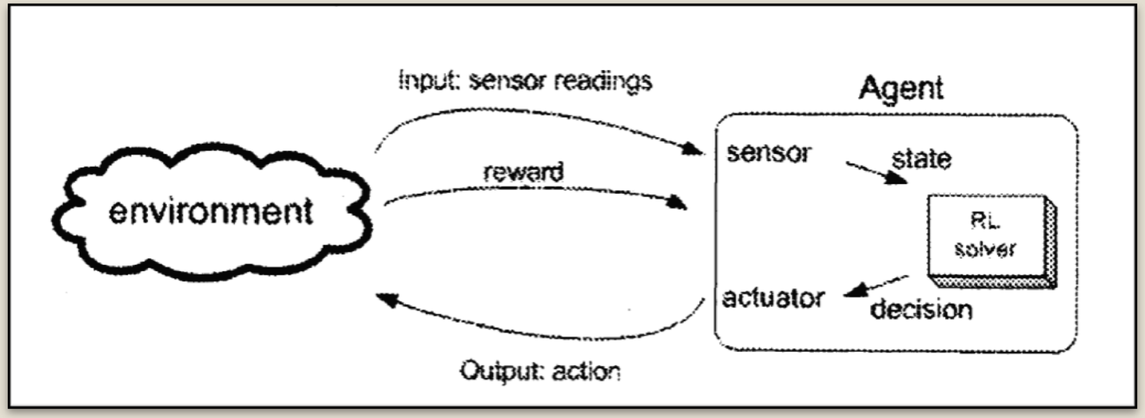
- Goal
  - Use distributed reinforcement learning (DRL) in sensor networks
  - Decentralized cooperation among independent sensors
- Introduce three DRL approaches
  - *IndLearners*
  - Distributed Value Function (*DVF*) DRL
  - Optimistic DRL (*OptDRL*)
- Apply approaches in a simulation comparing:
  - Policy convergence
  - Energy consumption
  - Memory consumption

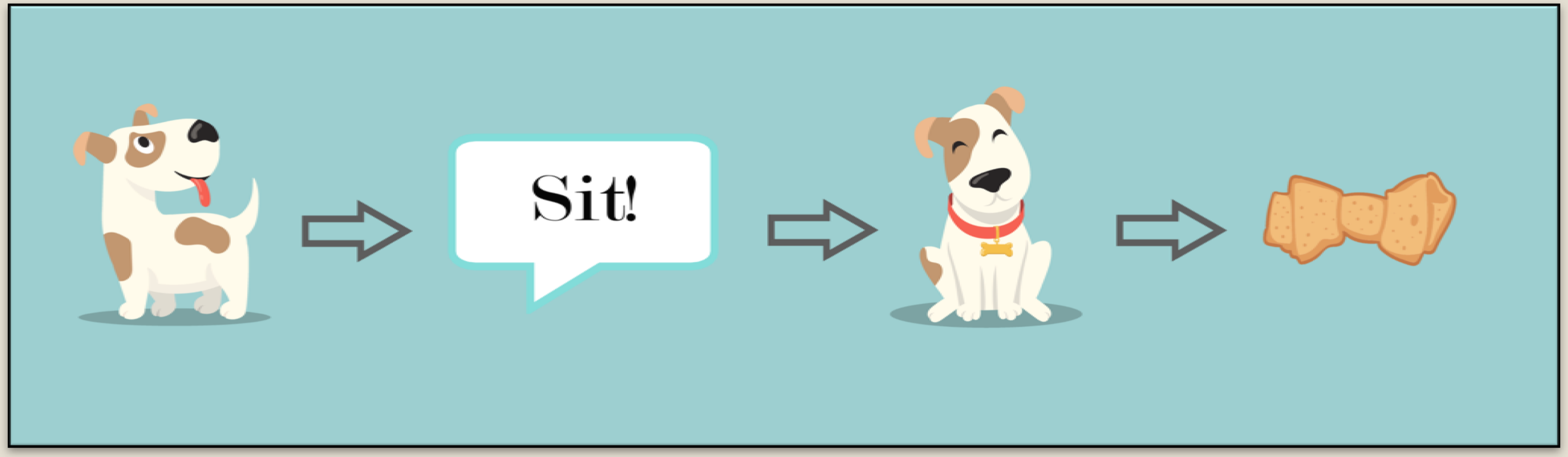
Multi-Agent Systems (MAS) in Wireless Sensor Networks (WSN)

- Multi-agent systems
  - Independent agents each acting upon the environment
  - Frequently changing environment dynamics
- Wireless sensor networks
  - Limited energy per sensor
  - Sensors can communicate with one another
    - Incurs energy costs
  - May or may not have a central hub
    - Centralized or decentralized
- Decentralization
  - Reduces energy consumption
  - Scalable
  - Natural implementation



Reinforcement Learning Overview





Markov Decision Process (MDP)

- Markov Property
  - Future outcomes based only on current state
- MDP = RL problem that has Markov property
  - Defined as a 4-tuple

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$$

$\mathcal{S}$ is a discrete set of states

$\mathcal{A}$ is a discrete set of actions

$$\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$$

$$P^a_{ss'} = Prob(s_{t+1} = s' | s_t = s, a_t = a)$$

$$\mathcal{R} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$$

$$R^a_{ss'} = E[r_{t+1} | s_{t+1} = s', s_t = s, a_t = a]$$

Policies and Value Functions

$$policy\ \pi : \mathcal{S} \times \mathcal{A} \rightarrow [0..1]$$

$$R_t = \sum_{k=0}^{\infty} \gamma^k r_{t+k+1}$$

$$\begin{aligned} V^\pi(s,a) &= E^\pi\{R_t | s_t = s\} \\ &= E^\pi\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s\} \end{aligned}$$

$$\begin{aligned} Q^\pi(s,a) &= E^\pi\{R_t | s_t = s, a_t = a\} \\ &= E^\pi\{\sum_{k=0}^{\infty} \gamma^k r_{t+k+1} | s_t = s, a_t = a\} \end{aligned}$$

$$Q^{\pi^*}(s,a) = \max_{\pi} Q^\pi(s,a)\ \forall\ (s,a) \in \mathcal{S} \times \mathcal{A}$$

- Policy
- Discounted rewards
- State-value function
- Action-value function
- Policy convergence

Q-Learning

- Hold a 2D lookup table of [state, action] Q-values
- Update Q-values base on:

$$Q_{t+1}(s_t, a_t) = (1-\alpha)Q_t(s_t, a_t) + \alpha\Big(r_{t+1}(s_{t+1}) + \gamma \max_{a \in A} Q_t(s_{t+1}, a)\Big)$$

Fully Distributed Q-Learning (*IndLearning*)

- Rudimentary approach
- Used as baseline comparison
- No communication among agents
  - Each agent acts according to independent Q-learning
- Not guaranteed to converge on an optimal policy

$$Q^i_{t+1}(s^i_t, a^i_t) = (1-\alpha)Q^i_t(s^i_t, a^i_t) + \alpha\Big(r^i_{t+1}(s^i_{t+1}) + \gamma \sum_{j \in Neigh(i)} f^i(j) V^j_t(s^j_{t+1})\Big)$$

$$V^i_{t+1}(s^i_t) = \max_{a \in A^i} Q^i_{t+1}(s^i_t, a)$$

$$f^i(j) = \begin{cases} \frac{1}{|Neigh(i)|}, & \text{if } Neigh(i) \neq 0; \\ 1, & \text{otherwise.} \end{cases}$$

Distributed Value Function (*DVL*) DRL

- Agents communicate information about value functions
- Communicates with all other nodes via the value function
  - Results in something like global reward but decentralized
- Algorithm

$$\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$$

$$\mathcal{S} = \prod_{i=1}^{m} S^i$$

$$\mathcal{A} = \prod_{i=1}^{m} A^i$$

$$\mathcal{P} : \mathcal{S} \times \mathcal{S} \rightarrow [0, 1]$$

$$\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$$

Multi-Agent MDP (MAMDP)

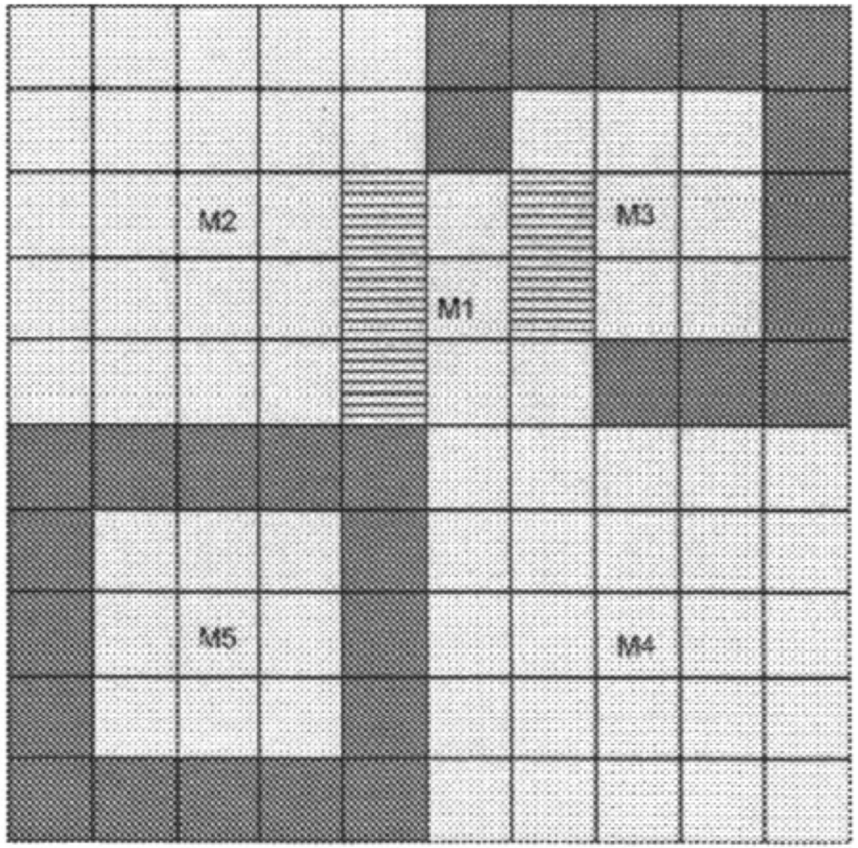
- Extends the basic MDP from RL to multiple agents
  - Considers the state and action of every agent
- MAS state and action = vector of individual states and actions
- Similar 4-tuple to MDP

$$q^i_{t+1}(S_t, a^i_t) = \max\{q^i_t(S_t, a^i_t), (1-\alpha)q^i_t(S_t, a^i_t) + \alpha\big(r^i_{t+1}(S_{t+1}) + \gamma \max_{a \in A^i} q^i_t(S_{t+1}, a)\big)\}$$

$$\Pi^i_{t+1}(S_t) = a^i_t \text{ iff } \max_{a \in A^i} q^i_t(S_t, a) \neq \max_{a \in A^i} q^i_{t+1}(S_t, a)$$

Optimistic DRL (*OptDRL*)

- Uses two equations to ensure convergence of optimal policy
  - First equation
    - Q-function
    - Assumes other agents act optimally
  - Second equation
    - Updates the policy only when there is an improvement
    - Introduces coordination among agents



Simulations

- 5 stationary agents illuminating a 10x10 room
- Goals
  - Fully illuminate a room
  - Minimize energy consumption
- Agent actions
  - Lights off (No energy)
  - Lights low (Some energy)
  - Lights high (High energy)
- Agent states
  - Light level of the 5x5 grid around the agent
- Optimal policy
  - M1 turns off its light
  - All other agents turn on lights high

Results

- Converged policies
  - *IndLearners*: all agents lights high (not optimal)
  - *DVF* and *OptDRL*: optimal
- Convergence time
  - *DVF*: 4400 iterations
  - *OptDRL*: 1200 iterations
- Energy consumption
  - *OptDRL* uses more energy for communication and computation than DVF
- Memory requirements

	Expression	Actual values
IndLearners	$\lvert s^i \rvert \times \lvert A^i \rvert$	$2^{25} \times 3$
DVF	$\lvert s^i \rvert \times \lvert A^i \rvert + \lvert s^i \rvert$	$2^{25} \times 4$
OptDRL	$\lvert S \rvert \times \lvert A^i \rvert + \lvert S \rvert$	$2^{100} \times 4$