# Decentralized POMDP (Dec-POMDP)
# A Brief Overview

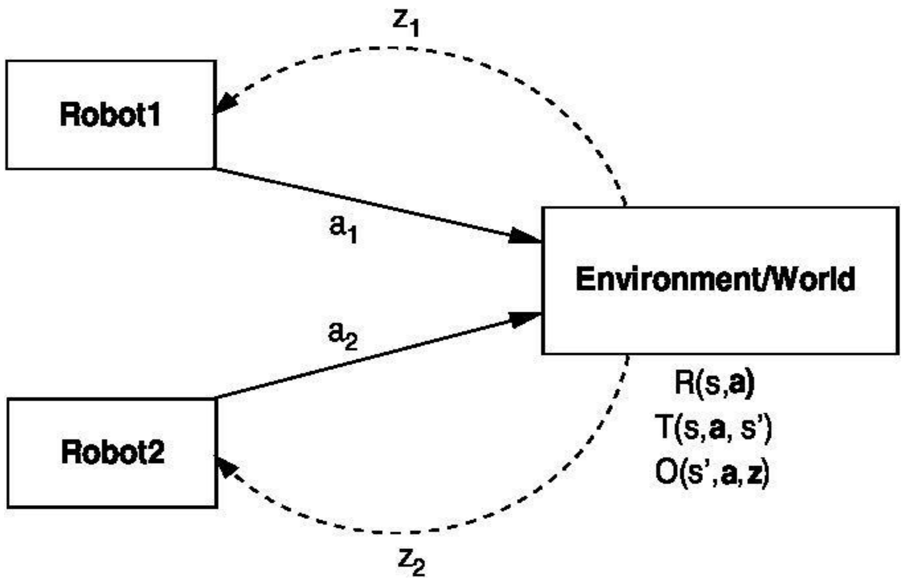
**Prof. Mohan Sridharan**
**Chair in Robot Systems**
University of Edinburgh, UK
https://homepages.inf.ed.ac.uk/msridhar/
*m.sridharan@ed.ac.uk*

# Dec-POMDP Motivation

- Many real-world problems can be formulated as multiagent and multirobot decision making problems.
- Still need to model uncertainty and make reliable decisions.
- Need "policies" for the individual members that helps achieve overall objective of team.
- Computationally expensive; workarounds exist.

# Dec-POMDP overview

- Defined by the **tuple**: $<D, S, A_i, Z_i, T, O, R, h, I>$




- *D={1,…, n}*: finite set of robots.
- *S*: finite set of states for the environment.
- $A_i$: finite set of actions for each agent.
- $Z_i$: finite set of observations for each agent.

# Dec-POMDP overview

- Transition function

  $$T(s'|s,a) \to [0,1]$$

- Observation function

  $$O(o|s',a) \to [0,1]$$

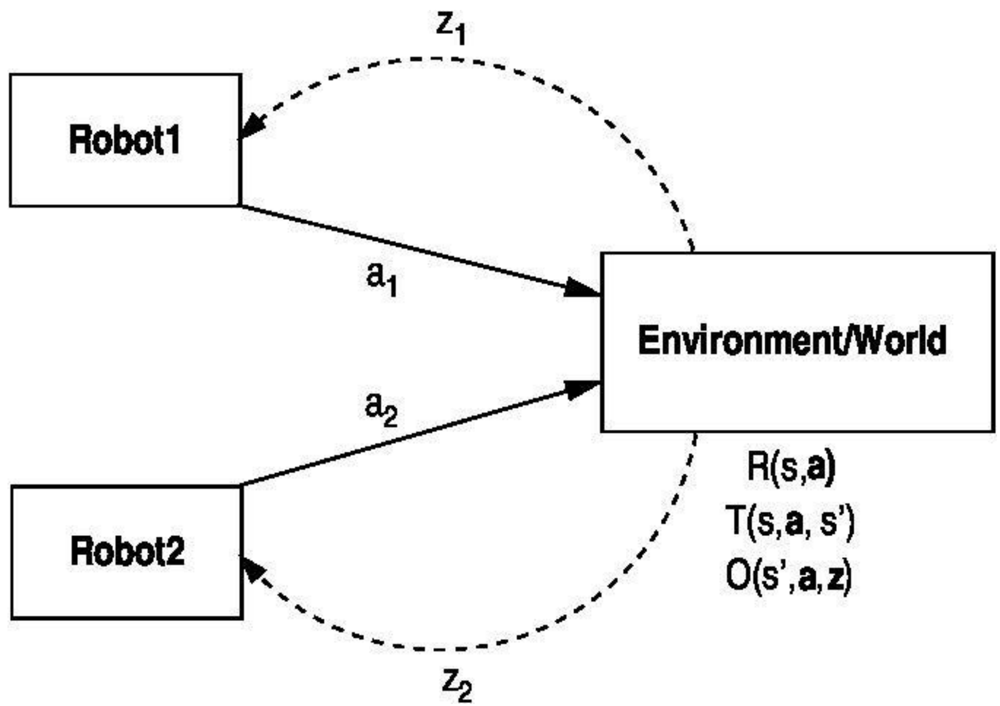
- Reward function

  $$R(s,a) \to \Re$$

- *h*: horizon of the problem.

- Initial state distribution

  $$I \in p(S)$$



# Dec-POMDP

- Considers joint actions and observations.
- Every time step:
  - Joint action **a** is taken, and influences environment. Robots only known their individual action choice.
  - Robots receive joint observation **z** from environment. Each robot receives own component: $z_i$ only.
  - Each robot assumes to act based on own observation.
  - No explicit communication (can be relaxed); implicit through states, actions, and observations.

- Need to determine joint policy (one for each robot) that maximizes expected cumulative reward.
- Policy computed offline, executed online (similar to POMDPs).

# “Decentralized Tiger” Problem

- Two robots standing in a corridor with two doors: one door hides a treasure, the other hides a tiger!
- Robots need to compute a policy for identifying the door with the tiger, and avoid it ☺
- Each robot acts on the environment and receives an observation. The true underlying state does not change but is unknown.
- Once a door is opened, the states (and belief state) are reset again.

# “Decentralized Tiger” Problem

- Two robots. States:
  - tiger-left ($s_L$), tiger-right ($s_R$), terminal-state?
- Initial belief: [0.5 0.5]
- Actions (for each robot):
  - open-left ($a_{OL}$), open-right ($a_{OR}$), listen ($a_{LI}$)
- Observations:
  - Hear tiger behind left door ($z_{HL}$), Hear tiger behind right door ($z_{HR}$)
- Transition function:
  - State does not change until door opened. Once opened, episode ends and belief resets. T : <$a_1$,…, $a_n$> : <start-state> : <end-state> : %f

# “Decentralized Tiger” Problem

- Observation function: can be same for both robots.
  - If state is $s_L$, and action $a_{LI}$ is executed, each agent observes $z_{HL}$ with probability 0.85
  - Probability that both hear tiger behind left door= 0.85*0.85 = 0.7225
  - O : <$a_1$,…, $a_n$> : <end-state> : <$z_1$, …, $z_m$> : %f
- Reward function:
  - One robot opens door with treasure: 10; both robots open door with treasure: 20
  - Listening only has low cost: -2
  - One robot opens door with tiger: -100
  - Both robots open door with tiger: -50 (only one robot gets eaten?) ☺
  - R : <$a_1$,…, $a_n$> : <start-state> : <end-state> : %f … %f
- No way of observing attack by tiger! ☹
- Policy should maximize probability of acting jointly.

# Dec-POMDP Solvers

- Exact solvers:
  - Dynamic programming (bottom-up, backward search).
  - Heuristic search (top-down, forward search).
  - Policy iteration.
- Approximate solvers:
  - Memory-bounded dynamic programming (top-down + bottom-up).
  - Joint equilibrium search.
  - Communication! ☺
- Combinatorial optimization: cross-entropy, genetic algorithms.
- State of the art efficient Dec-POMDPs solvers being developed; used for real-world multiagent planning.

# That's all folks ☺