<CAPS>

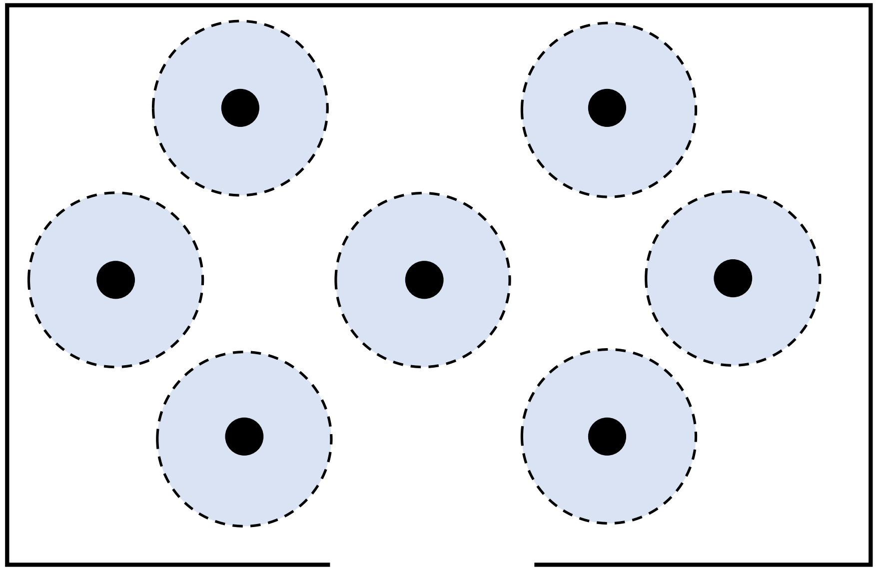
Session 2 – Rule-Based Decision Making in a Fuzzy World

Leo Klenner, Henry Fung, Cory Combs

Outline

1. The Room Clearing Problem
2. Recap and Conceptual Trade-Offs in Reinforcement Learning
3. Rule-Based Decision Making and Finite State Machines
4. Primer on Multi-Agent Learning
5. Decision Rules for Navigation of a Contested Environment

The Room Clearing Problem



> How to maximize coverage of a room?

> How to navigate through a complex location?

The Room Clearing Problem

- How to maximize coverage of a room?
  - Agents have limited field of vision
  - Limited number of agents
  - Obstacles in rooms; modular layout
  - Agents need to coordinate, often without direct contact

- How to navigate through a complex location?
  - Unknown territory
  - Need to discern all relevant features (e.g. doors, etc.)
  - Balance trade-offs between shortest path and best path
  - Agents need to coordinate, often without direct contact

Recap - Modeling the World as an MDP

> What is a **Markov Decision Process** and what features does it have?

> How should we understand the concept of a **state**?

Trade-Offs in RL 1 – Online versus Offline Learning

- **Online RL**
  - Prior knowledge from training is used by the agent
  - Typically used on large, static data set
- **Offline RL**
  - Agent has no prior knowledge of the environment
  - The agent learns to improve its policy and make optimal action through direct interaction with the environment
  - Allows for continuous learning as new information become available
  - Typically used on incremental, fluid data set

Online versus Offline RL in Target Acquisition

> Learning how to classify synthetic apperture radar data

Offline

Data

Classifier

Online

Data

Classifier

Target

Trade-Offs in RL 2 – Exploration versus Exploration

- Dilemma for an algorithm that aims to **acquire knoweldge** (explore) and at the same time, **maximize its reward** (exploitation)

- Example
  - You are in a casino filled with **slot machines**.
  - Each of the slot machines have different odds of winning. How do you maximize your cumulative winnings?
    - Strategy 1: **Play only one machine**. But you won‘t be able to find out about the odds of winning of other machines (maybe there are better machines out there?)
    - Strategy 2: **Play and find out the odds of many different machines**. But you won‘t be able to always play on the known best machine to get more winnings.

Epsilon-Decreasing with Softmax

> Exploit current strategy with probability 1-epsilon. Explore a new option with probability of epsilon, with epsilon *decreasing over time*.

> We do a lot of exploration in the first few "training sessions", and then "stick" to one option and exploit in the later training sessions.

> When we explore a new option, we don't just pick an option at random, but instead we estimate the outcome of each option, and then pick the one with the best estimated outcome ("softmax").

What is a Rule-Based System?

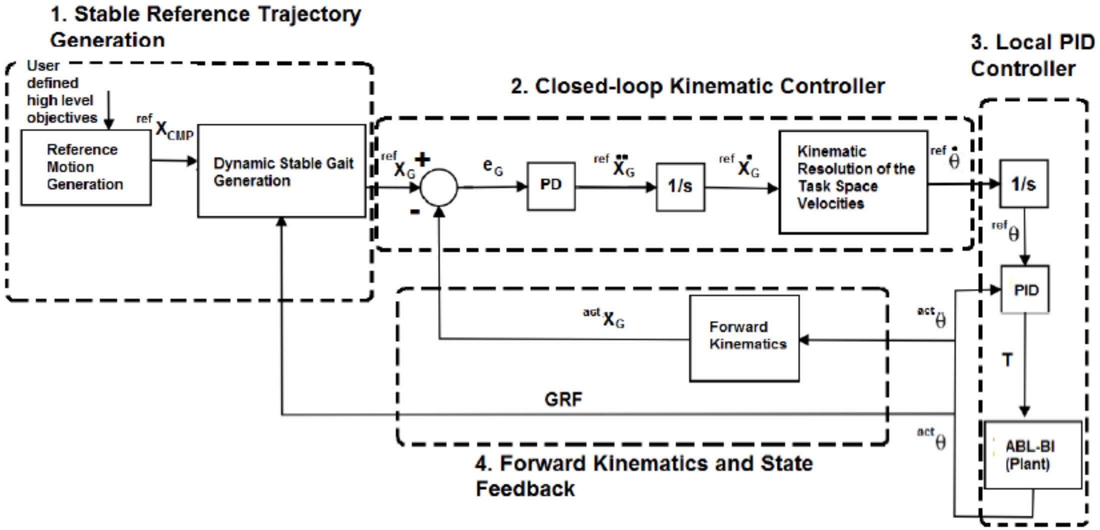
> Codify human strategies into a set of instructions that are executed by the computer

> For example, we want to build an AI to play chess. We will first “tell” the computer the basic rules of chess.

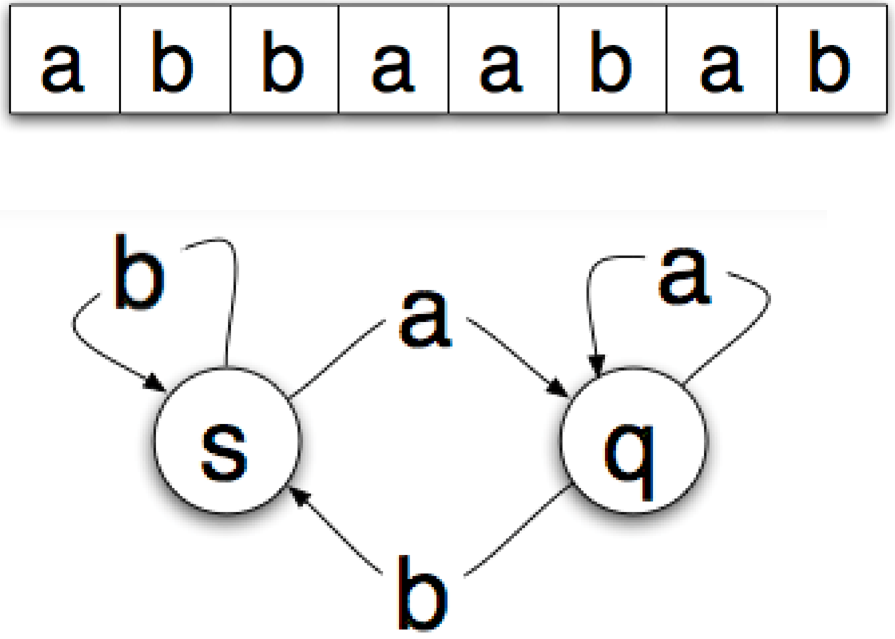
> Next, we interview the best chess players in the world (experts) and codify their strategies into the computer.

> If certain conditions are met, a particular strategy (which contains a sequence of moves) will be executed by the computer.

> Rule-based algorithms can be used to control very complex systems such as a bipedal walking robot!



What is a Finite State Machine (FSM)?

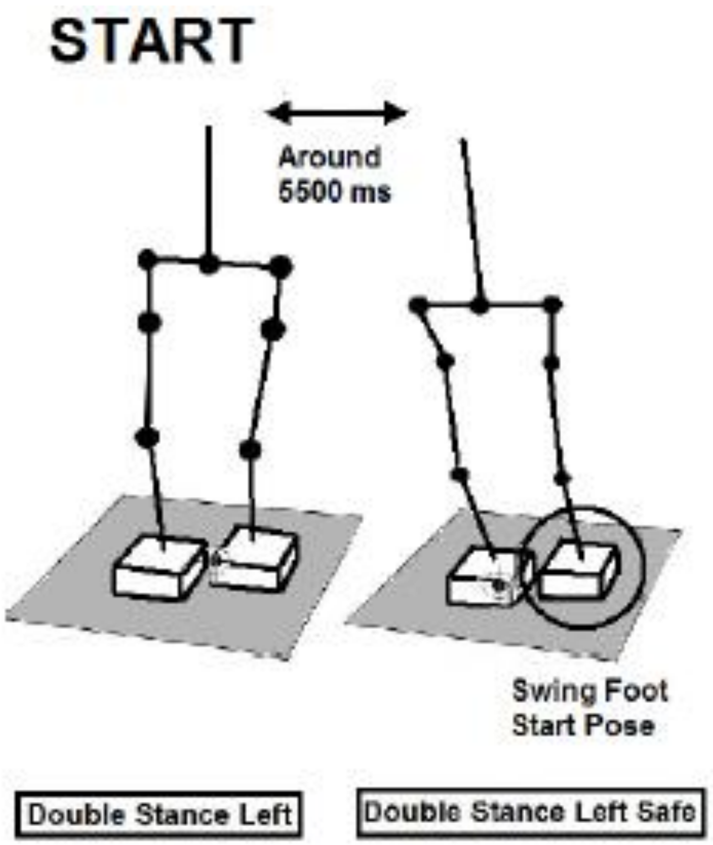


- Consider a machine that reads out letters.
- Circles are states, arrows are transition.
- For each state, there can only be one transition.
- One application of FSM: How can we tell if the tags are in a particular order?
- Answer: If FSM successfully makes it to the final state, then the tags are in the correct order.

```
<html>
  <head> </head>
  <body> </body>
</html>
```

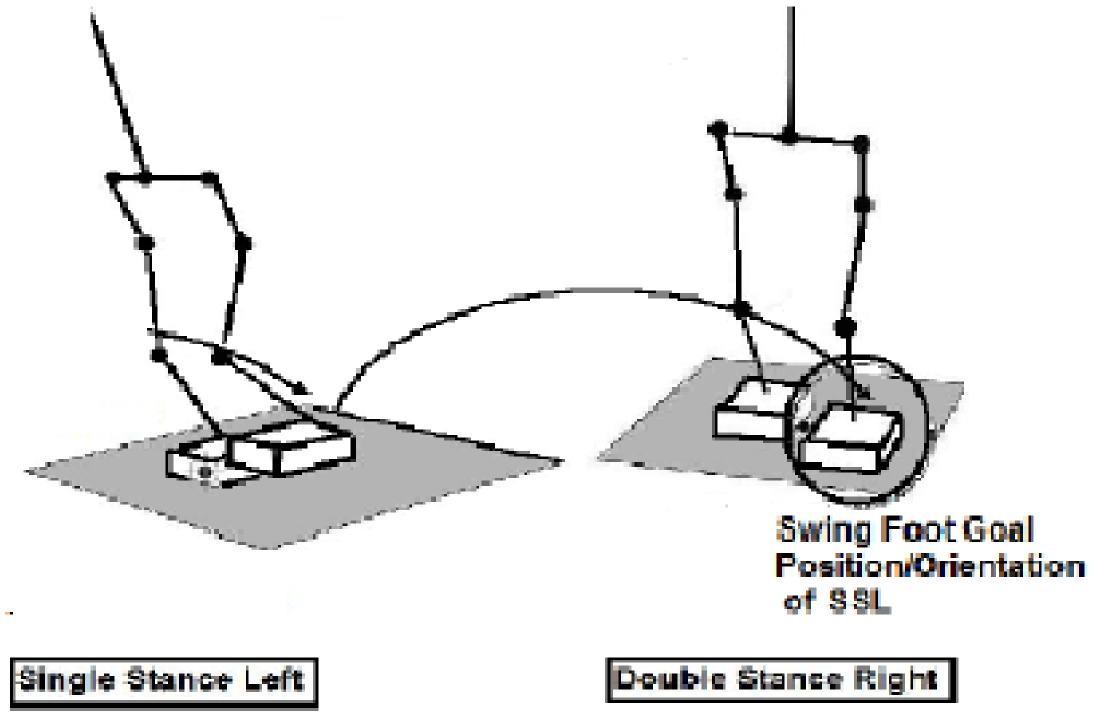
FSM Example: States and State Transitions I

- Double stance left (DBL):
  - State: Both feet on the ground, left foot forward
  - Action: Shift weight to left foot.
  - State transition: once weight is on left foot, transition
- Double stance left safe (DBL_safe):
  - State: Both feet on the ground, left foot forward, weight is on left foot
  - Action: Lift swing (right) leg upwards
  - State transition: swing leg leaves the ground



FSM Example: States and State Transitions II

- Single Stance Left (SSL):
  - State: Left feet on the ground, right feet has left the ground
  - Action: Swing right leg forward to take a step, adjust torso to maintain balance.
  - State transition: swing leg touches down.
- Double Stance Right (DBR):
  - State: Both feet on the ground, right foot forward
  - Action: Shift weight to right foot.
  - State transition: weight is on right foot

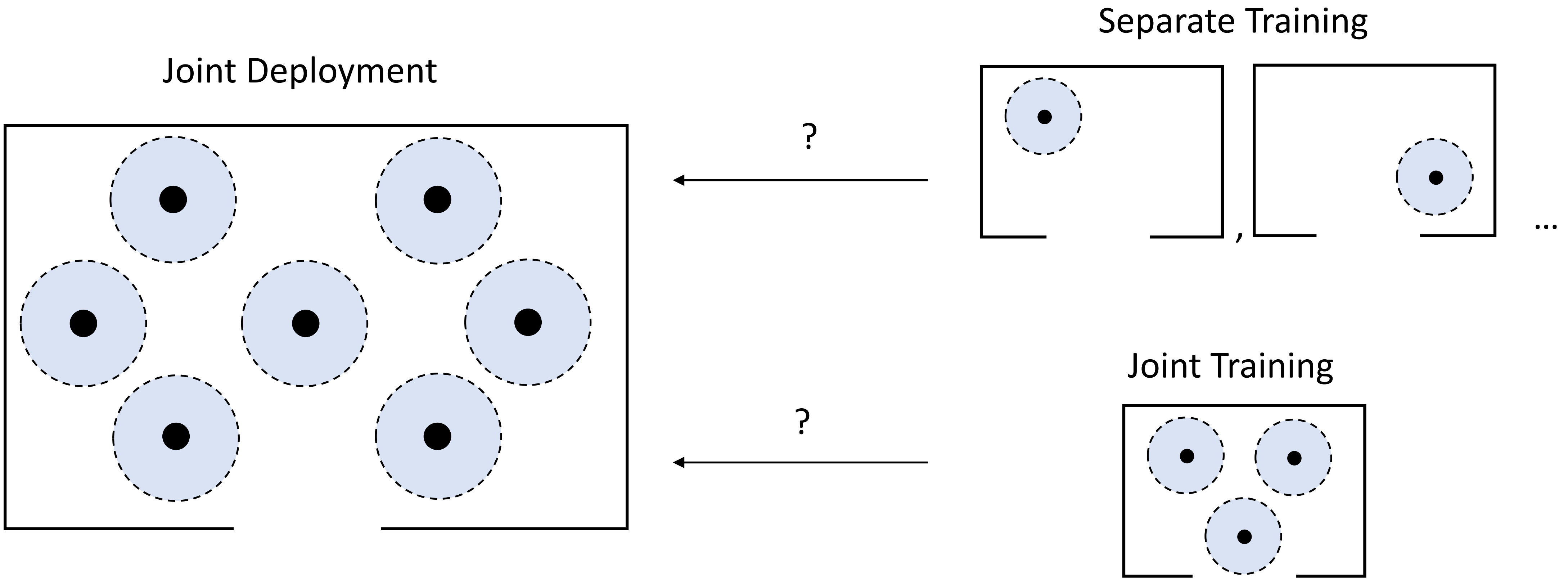


Finding Essential Properties of a Team

> Think of Army captains Mark Nutsch and Jason Emory in Afghanistan

  > Specific team size?
  > Ability to integrate new team members? During training? During deployment?
  > Ability to cope with other changes in the team’s structure?
  > Heterogenous or homogenous team members?

Primer on Multi-Agent Learning (MAL)



Commonalities of MAL

- Break the learning problem into seperate roles that are semi-independent
- Learn roles seperately through interaction with each other

- Problem:
  - No representation of how roles relate to the team structure
  - No principle for assigning new roles automatically to new individuals

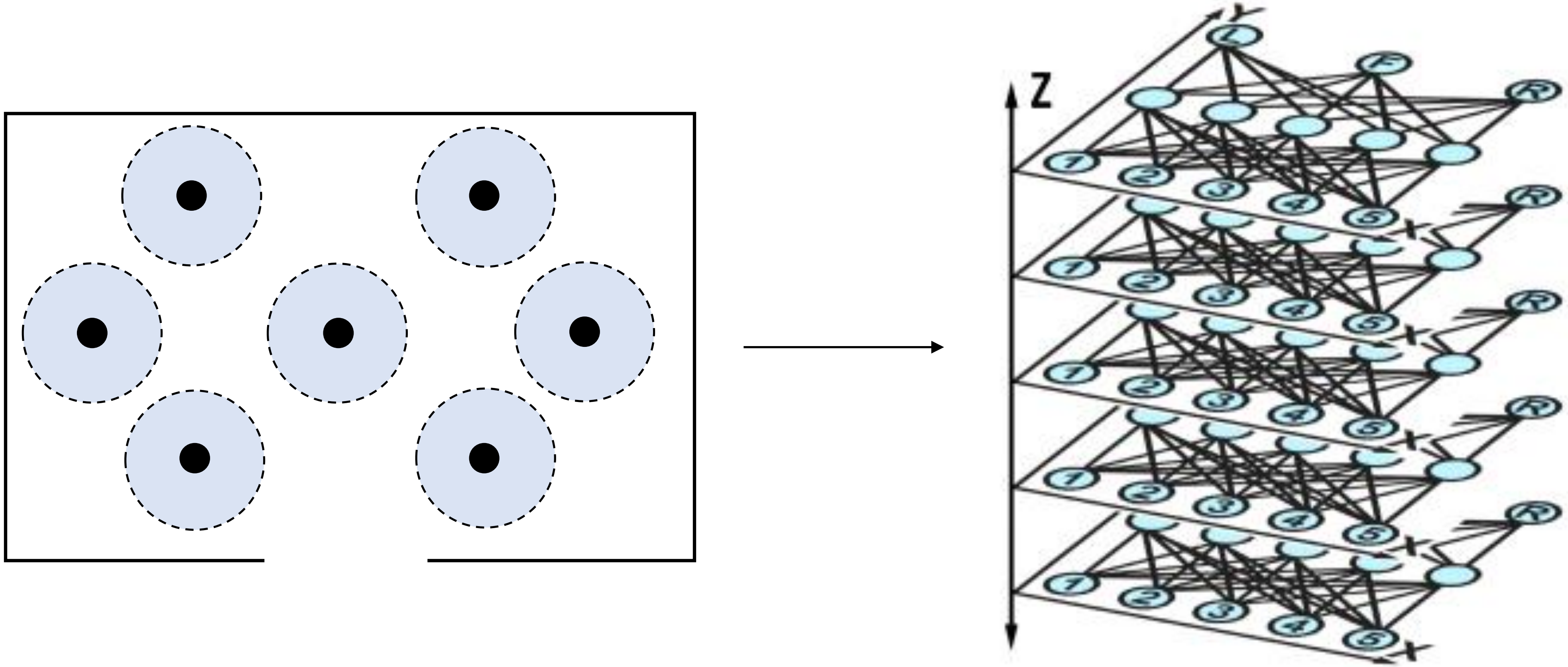
The Problem of Reinvention

- MAL techniques commonly face a *problem of reinvention*
  - Agents are treated as seperate subproblems
  - Agents must seperately discover and represent all aspects of the solution, even though optimally there may be a high degree of overlapping information among the policies of the agent
  - In MARL, each agent learns a separate reward function based upon individual or joint experiences

Policy Geometry MAL

- David B. D‘Ambrosio et al. 2010. *Evolving Policy Geometry for Scalable Multiagent Learning*
- Assumption: The behavior of human agents tend to be related to to their canonical position within a team (e.g. in sports, the offensive positions generally stand in front defensive ones).
- These teams effectively have a *policy geometry*, e.g. a functional relationship between an agent‘s location within the team and that agent‘s behavioral policy.
- Thus, rather than individually learning each agent’s policy, we can instead learn a general relationship between team geometry and policies that can be sampled to derive the policies of individual agents

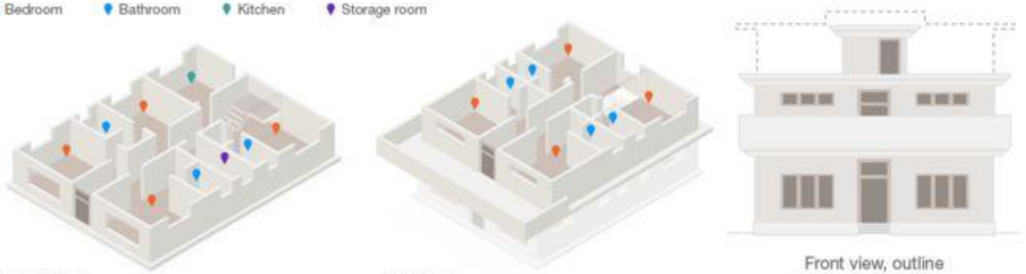
Representing a Policy Geometry



> Agent‘s internal geometry (*x*,*y*), position in the team (*z*)

3D view of building, based on original plans below

Bedroom · Bathroom · Kitchen · Storage room



Ground floor

First floor

Front view, outline shows final building

Original floor plans