Abstract
In a multi-agent environment we can expect to encounter agents that are not only autonomous, but also capable of learning from the behaviour of others.

In this setting, the way one chooses to act can influence the decisions of other agents, suggesting solutions or showing weaknesses. One’s learning process cannot therefore be decoupled from what one teaches others through his own actions, even without explicit communication.

We present a learning algorithm that can model the phases of the interaction with a wide class of agents, including adaptive ones, and suggest the best strategy in order to achieve the design goal of our agent, in the setting of stochastic games.

Model-based learning
When facing agents with bounded resources, we can try to infer a model of the strategy they employ in the game, and compute the best response to their actions that will bring our agent closer to its goal. If necessary, we can update the model after each step to ensure that it is still consistent with the development of the game.

Models have limited expressive power, and can only represent certain classes of agents. No general model exists yet for adaptive agents.

The question
In front of such adaptive opponents, how can an agent learn to act best, in order to achieve its design objective?

Would it be possible to model the adaptive process of another agent, use this knowledge to develop a corresponding game strategy, and possibly also for cooperation or exploitation of specific behaviours?

Our approach
We use a hierarchical model to collect information on the way the opponent plays and adapts to our agent’s actions.

On the bottom layer of the hierarchy, we model the visible states of the opponent, the stationary behaviours that it exhibits while playing. On top of these, we build a graph of the relations among these states, and the cause-effect dependencies from our actions.

Building the model
The method that we are currently researching uses finite state machines to model the states of the opponent. We compute a best response to the model while we play, until we can consider the machine complete, e.g. until we cannot improve our strategy by modifying it. Complete machines are added to the set of visible states of the opponent.

If at some point the current model does not reflect the behaviour of the opponent, we start looking for a new machine. This might be already in our set, or we might need to restart the inference procedure above. We save information about this transition from one state to another. We also record our actions preceding it: in the future we might execute the sequence again, trying to reproduce the same reaction in the opponent!

As the interaction develops and we infer more machines and actions, we should be able to build a complete set of behaviours of the other agent, and a set of macro-actions that we can use to influence its state.

Moreover, the Dog is an adaptive agent, and it might develop a new strategy to achieve is goal (and sleep) at any time during the ‘game’. For example, it could start chasing the Robot around the grid, spreading dirt of course, until the Robot stops in a square for some time.

We can assign a value to these states, depending on their utility toward our goal, and their stability. We will be able to learn which macro-actions are effective, and discover the best states in the model. If we are lucky, we might find an equilibrium, a state from which both players have no incentive to deviate.

Concluding (buzz)words
The specific context of our research is multi-agent reinforcement learning in repeated games. Both the model and the algorithm are still under development. Extensive empirical evaluation will be necessary to ascertain if the method achieves at least best response against the target class of relatively stationary opponents, and avoids exploitation.