## Conclusion

In this work, we develop a prototype of bot for StarCraft: Brood War which learns its decision-making processes from demonstrations. We decompose the problem of playing RTS game using our framework based on Multi-agent system. Presentation of RTS domain is part of chapter {link} and concepts of MAS are in chapter {link}. We developed our framework in a way that Inverse Reinforcement Learning and other techniques can be integrated into it. Description of the framework and techniques integrations is in chapter {link}. Overview of techniques with current challenges in RTS games is given in chapter {link}. Inverse Reinforcement Learning as one of the novel techniques how to learn decision-making from demonstration has own dedicated chapter {link}. Using the right set of replays to learn decision-making trough gameplay observation our bot was able to learn strategy which can beat built-in AI for Zerg in some 1on1 map scenarios as we show in the last chapter. The bot also shows the ability to adapt to the situation to some degree. In the final chapter; we also present some examples of decomposition and decision-making problem statements for IRL.

### Discussion and future work

With the current state of bot it seems that we are still not at the end of the road to introduce game AI which could be a severe threat to the present state of the art bot and maybe even for humans. However, results present great promises. The latest version of bot shows some level of adaptive behavior and is even able to learn some winning strategy. However, there are still challenges to be addressed before bot developers can take full advantage of IRL technique or presented framework. In the case of our framework, we see great promise in the implementation of following features:

* It would be good to have some local belief mediator for shared desires between the contractor and committed agents to the abstract developer from accessing global beliefs to find information which can be shared locally.
* Another great functionality greatly increasing the level of flexibility for plan definition would be the ability to use different techniques to select appropriate implementation of the module.

For Inverse Reinforcement Learning usage in the domain of RTS, we see following directions for future work:

* Exploring usage of infinite states MDPs.
* Combining IRL with other algorithms to adjust policy and reward function.
* Use learned MDPs in other domains or at least other RTS.

We are also interested in possibilities of usage of symbolic regression in combination with our framework to model the system. In our case model of bot is provided by the user which may introduce bias and can limit the performance of the system as some of the important domain knowledge may not be know by a designer. An instance of symbolic regression could eliminate those issues by finding a structure which can fit dataset by composing system from blocks of behaviors.