underlying game. However, this is a first step towards representations with less information loss, such that optimal performance in the metagame can result in strong performance in the underlying game.

2. CURRENT RESULTS

As a first experiment, we instantiated a special case of the metagame upon real-time strategy game StarCraft. The special case has a single state, i.e., each player maps only the initial game state to an algorithm that plays an entire match. This metagame corresponds to a normal-form game, whose payoff matrix indicates expected relative performance among algorithms. We estimated the payoff matrix with average results of matches among StarCraft bots which served as our algorithms.

We noticed that some algorithms interact in cyclical ways, similarly to rock-paper-scissors. In fact, our experiments confirmed insights of computer rock-paper-scissors tournaments [2]: it is useful to deviate from equilibrium to exploit sub-optimal opponents, but, against strong competitors, a player must protect itself against exploitation by playing the equilibrium strategy. These results are in [6].

We also built a functional StarCraft bot, named MegaBot, which uses the single-state metagame framework of [6], to participate in AI tournaments. MegaBot has a portfolio of three algorithms, which are themselves other StarCraft bots, specifically, three non-dominant bots of AIIDE 2015 tournament[1], in order to test whether MegaBot performs well by learning how to properly select algorithms rather than due to a powerful portfolio. In StarCraft AI tournaments, we do not know the metagame payoff matrix beforehand, thus we learn its values via minimax-Q's update rule.

MegaBot placed among the top 50% bots in two competitions. It has outperformed each of its portfolio components and received an honorable mention for its learning curve (measured in rate of victories per round). This indicates that the metagame is a useful approach for algorithm selection in adversarial settings. MegaBot did not score better because no component of its portfolio could defeat the strongest competitors.

3. NEXT STEPS

The presented metagame model assumes that players know each other's algorithm portfolio. However, in a realistic setting, the agent is aware of the opponent's presence but does not know his possible behaviors. Formally, the agent plays an incomplete-information stochastic game, in which it only knows its own actions. Next steps of this research involve the study of our proposed model for this situation: an extension of the adversarial multi-armed bandit [1] - which corresponds to a normal-form game with unknown opponent actions - to a multi-state problem. The Exp3 method of [1] replaces the usual equilibrium calculation in the single-state adversarial bandit, exhibiting theoretical performance guarantees. In our multi-state case, we extend Exp3 by incorporating the value of future states in action-values used for policy calculation, as in traditional reinforcement learning methods. To the best of our knowledge, both the model of stochastic games with incomplete information and the proposed method to handle it (named SG-Exp3, from Stochastic Game Exp3) are novel.

We want to investigate whether SG-Exp3 bounds agent's losses, extending the guarantees of Exp3 from adversarial multi-armed bandits to stochastic games. Experimental results may be useful in this sense: a robust performance of SG-Exp3 may indicate that further investigation of its theoretical properties can be fruitful.

An interesting direction of research, possibly out of the current thesis' scope, is metagame creation, that is, to automatically place decision points in relevant portions of the underlying game's state space. Automated creation might help mitigating a metagame limitation: it is a "lossy" abstraction of the underlying game so that its solution is not valid for the underlying game. In other words, it might be possible for a player to perform low-level underlying game actions to exploit a metagame player. In this sense, a long-term goal is to construct increasingly precise metagames, that remain simple to solve, but maintain the underlying game's strategic structure, so that optimal metagame strategies result in strong underlying game performance. Moreover, this methodology would fit the leading game-solving paradigm used in poker [5], in which automatic abstractions are generated, solved and the resulting strategy is ported to the original game.

[1] `https://www.cs.mun.ca/~dchurchill/starcraftaicomp/2015/`

Acknowledgments

We acknowledge CNPq for the PhD scholarship and CAPES and FAPEMIG for support in this research.

REFERENCES

[1] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. Gambling in a rigged casino: The adversarial multi-armed bandit problem. In *Foundations of Computer Science. Proceedings, 36th Annual Symposium on*, pages 322–331. IEEE, 1995.

[2] D. Billlings. First International RoShamBo programming competition. `https://webdocs.cs.ualberta.ca/~darse/rsbpc1.html`, 1999.

[3] M. L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the 11th International Conference on Machine Learning*, pages 157–163, New Brunswick, NJ, 1994. Morgan Kaufmann.

[4] J. R. Rice. The algorithm selection problem. *Advances in computers*, 15:65–118, 1976.

[5] T. Sandholm. The State of Solving Large Incomplete-Information Games, and Application to Poker. *AI Magazine*, 31(4):13–32, 2010.

[6] A. Tavares, H. Azpúrua, A. Santos, and L. Chaimowicz. Rock, Paper, StarCraft: Strategy Selection in Real-Time Strategy Games. In *12th Artificial Intelligence and Interactive Digital Entertainment Conference (AIIDE)*, pages 93–99, 2016.

[7] L. Xu, F. Hutter, H. H. Hoos, and K. Leyton-Brown. SATzilla: portfolio-based algorithm selection for SAT. *Journal of Artificial Intelligence Research*, 32:565–606, 2008.