Angry-HEX: An Angry Birds-playing Agent based on HEX-Programs

Francesco Calimeri[1] and **Stefano Germano**[1,3] and **Nikolaus Funk**[2] and **Bianca Löhnert**[2] and **Hesham Morgan**[2,4] and **Christoph Redl**[2] and **Zeynep G. Saribatur**[2] and **Peter Schüller**[2]

calimeri@mat.unical.it, stefanogermano0@gmail.com, e1425838@student.tuwien.ac.at
e1525825@student.tuwien.ac.at, heshammse@gmail.com, redl@kr.tuwien.ac.at
zeynep@kr.tuwien.ac.at, peter.schueller@tuwien.ac.at

[1] Computer Science Group – Department of Mathematics and Computer Science – University of Calabria, (Rende, Italy)
[2] Knowledge-Based Systems Group – Institut für Logic and Computation – Technische Universität Wien, (Vienna, Austria)
[3] Information Systems Group (ISG) – Department of Computer Science – University of Oxford, (Oxford, UK)
[4] Computer Science Engineering Group (CSEN) – Media Engineering and Technology (MET) – German University in Cairo, (Cairo, Egypt)

Abstract

A distinctive characteristic of our agent is that it uses a declarative, logic-programming based module for reasoning about the target to shoot and the level to play, implemented by means of so-called HEX-programs [Eiter *et al.*, 2016], i.e., by means of Answer Set Programming (ASP) with external sources and other extensions.

The Angry-HEX agent

Our agent, called Angry-HEX, builds on the *Base Framework* provided by the organizers and extends it with declarative means for decision making models by means of an *Answer Set Programming (ASP)*. Declarative logic programming controls two different layers for Angry-HEX: the **Tactics** layer, which plans shots, and decides how to complete a level; and the **Strategy** layer, which decides the order of levels to play and repeated attempts to solve the same level.

Tactics is declaratively realized by HEX-programs, i.e., an extension of ASP to incorporate external sources of computation via so-called *external atoms*. It is implemented using the HEXLITE solver and computes optimal shots based on information about the current scene and on domain knowledge modelled within the HEX-program. Its *input* comprises scene information encoded as a set of logic program facts (position, size and orientation of pigs, ice, wood and stone blocks, slingshot, etc.); its *output* are *answer sets* that contain a dedicated atom describing the target to hit, and further information about the required shot. Geometric calculations are integrated via external atoms.

The agent of this year significantly differs from previous years' agents ([Calimeri *et al.*, 2016]). The aim of this year's agent was to plan multiple actions in advance to have better strategies planned over predicated successor level states. HEXLITE [Schüller, 2019] is used instead of DLVHEX, which makes the setup of the agent easier and gives higher performance but supports less expressive external computations. The number of possible actions for each turn of the agent is exponential in the number of blocks of the level, and some levels have more than 50 blocks. Simulating physics for such levels is also expensive. Therefore, our agent implements an abstract model where the level is converted into a graph representation, where every object is represented as a node and the distance between objects is represented as the weight of the edges between the nodes. The agent searches over all possible shots, finds the optimal solution over all possile actions while minimizing on the number of actions, and maximizing on the number of blocks eliminated in a given level. Our agent predicts up to 4 successor states at every state of the level, using a planning problem that is formulated in the HEX language, where external computations represent the trigonometric equations that are used to build the abstract graph.

The **Strategy** layer decides, at the end of each level, which level to play next. This layer is also realized declaratively as an (ordinary) ASP program encoding our strategy on three priority levels: (1) each available level is played once; (2) levels where the agent score differs most from the current best score are selected; (3) levels where Angry-HEX achieved a score higher than the current best scores and that have the minimum difference from the best score, are selected. For each level, the **Strategy** layer keeps tracks of previously achieved scores and previously selected initial target objects.

References

[Calimeri *et al.*, 2016] F Calimeri, M Fink, S Germano, A Humenberger, G Ianni, C Redl, D Stepanova, A Tucci, and A Wimmer. Angry-HEX: An artificial player for angry birds based on declarative knowledge bases. *IEEE Trans. Comput. Intellig. and AI in Games*, 8(2):128–139, 2016.

[Eiter *et al.*, 2016] T Eiter, M Fink, G Ianni, T Krennwallner, C Redl, and P Schüller. A Model Building Framework for Answer Set Programming with External Computations. *Theory and Practice of Logic Programming*, 16(04):418–464, 2016.

[Schüller, 2019] P Schüller. The Hexlite solver. In *European Conf. on Logics in Artificial Intelligence*, pages 593–607, 2019.