[H0N05A] MODELLING OF COMPLEX SYSTEMS

MCS PROJECT PART 1: SAVING ARKHAM - REPORT

November 21, 2015

Joran Van de Woestijne

Vincent Van Gestel

INTRODUCTION

This report will describe our solution to the Arkham game. In the first section we will describe the predicates and functions added to the given vocabulary. In the section *Verifications* we discuss the results of the verification tests. Finally, we shall present some concluding thoughts about the project and our solution as well as estimate the time spent on the project.

EXPANDED VOCABULARY

Monster_Move(t, l, d) is the amount of monsters travelling from location l to district d at time t. This function will determine the movement of the monsters through Arkham. Whenever t is a monster turn, all monsters will move to a connected district. The monsters leaving a location is bound by the amount of monsters present in said location and the monsters arriving in a district is bound by three, as there can be no more than three monsters in one location at a time.

Is_Connected(l1, l2) is true if and only if location $l1$ has a connection with location $l2$. This predicate is different from **Has_Connection(l1, l2)** in the sense that **Is_Connected(l1, l2)** describes a symmetric connection instead of a directed one.

C_Win(t) is true if and only if there is a cause to win at time t. If all win conditions are met in the next turn, this will be true and the player will win the game at time $t+1$.

C_Lose(t) is true if and only if there is a cause to lose at time t. This predicate describes the exact opposite of the previous one. If all lose conditions are met in the next turn, the player will lose the game that next turn.

Reachable(l1, l2) is true if and only if location $l2$ can be reached from location $l1$. This predicate is used in the first verification, where it is enforced for all locations.

VERIFICATIONS

The first verification checks whether it is possible to reach all locations in Arkham. This is checked for the `Easy` and `Unplayable` structures. The **Reachable** predicate is enforced for all locations, hence any structure with unreachable locations will be unsatisfiable. This is the case for the `Unplayable` structure. If we investigate this structure, we confirm the presence of an unreachable section.

The third verification fails for the `Unlosable` structure. This verification is intended to check if it is impossible to lose in a given structure. It is however always possible to lose a game (with open gates). Verification seven proves that it is impossible to win a game without attacking. If there is a constant flow of monsters into the city (i.e. there is an open gate present in the city) there will be state where the investigator can no longer move and lose the game. `Unlosable` has open gates, so if the investigator restrains himself from attacking long enough, he will eventually lose the game. The structure `Unlosable` is thus losable.

In verification four, we can find that the investigator can live for seven turns in the `BoxedIn` structure. At turn eight, he will always be in a lost state. The investigator loses due to an excess of monsters. At turn seven the city will be flooded by monsters and no more can spawn without breaking the limit set on the amount of monsters present in the districts.

All other verifications behave as expected and don't require additional explanations.

Conclusion

The project was fun to work on, yet took substanially more time to finish than estimated in the assignment. This is largely due to the difficulties of finding bugs. Given that some statements require other statements to work properly, it can be quite difficult to find the exact location of an error. A lot of time is also spent waiting for idp to find a solution. Even for "small" structures like `Easy`, it can take a suprisingly long time to process. Making small adjustments to the code takes more time than it should, due to that reason.

In the end it took us around **30 hours** of work to finish the entire project.