Introduction

In a search game, a seeker searches for a hider in some space. In some versions of the problem, the hider is stationary, and other times the hider is allowed to move but has a maximum speed, \( v \). The seeker will always have maximum speed 1. The space in which the game takes place can also be varied. In all cases, the seeker is denied some information about the game, whether it is the hider’s location or information on the search space itself. The goal of the hider is to avoid capture if possible, and if he can’t avoid capture then to maximize the time until capture. In contrast, the seeker wishes to minimize the time until guaranteed capture.

Search games have been studied on different types of graphs including stars in (1), and trees in (2). In these cases, only the case where the hider is immobile is investigated. Presented here are some results on the search space.

For clarity’s sake, we will refer to the hider as he, and the seeker as she. The search spaces to be investigated here are stars. A star is a graph in which all the edges emanate from one vertex. We will call a star an \( n \)-star if it has exactly \( n \) edges, and we will call the center vertex from which all edges emanate the origin. We will assume that all legs have length 1. Some examples of stars are given below in Figure 1.

Result

It is possible for the seeker to guarantee to catch the hider on an \( n \)-star if \( v < 1/n \) and \( n \) is odd or \( w < 1/n \) and \( n \) is even. Otherwise, it is impossible for the seeker to guarantee capture of the hider. The lengths which the hider can occupy is called uncleared. The lengths to which the hider can either be at the center or not. In order to keep the rate of clearing as large as possible, we want the number of completely cleared legs, \( e \), or the number of completely uncleared legs, \( f \), to be as small as possible. The upper bound on \( \max(e, f) \) is shown in 3 for the case of a 7-star.

Optimality of the Algorithm

The given algorithm guarantees capture of the hider and does so in the least amount of time. The optimal algorithm will achieve the highest clearing rate at all possible points. When looking at the clearing rate, we need to look at what the seeker does, and where the seeker can gain back length where he can be.

Conclusion

We have described what is optimal in all the cases for a seeker looking on a star with legs of equal length. A natural question which arises from these results is how do the results change when the search space is a tree or even any graph. Searching on a cycle has been studied (3). Cycles have mainly been studied where the hider has a faster maximum speed than the seeker. So to further these results to apply to graphs in general one would have to determine the behavior of searching on a cycle for a slower hider. Another possible extension is to find a strategy for an omnipotent hider who could always avoid capture when his speed is above the given bounds.

References


Acknowledgments

I would also like to thank my amazing thesis advisor, Ran Libeskind-Hadas for listening to my progress every week and constantly providing feedback.

For Further Information

• My email address is jiglesias@hmc.edu
• To obtain more information on the topics discussed on this poster, http://www.math.hmc.edu/seniorthesis/archives/2012/jiglesias.