

Pursuit

Zach Bearinger & Tom Hadlaw



October 30, 2014

Acadia University

# Introduction

Tracking something you are looking to see how their actions effect the environment. If you are trying to follow someone through populated areas it would be productive to ask people if they saw the person recently. Depending on a person’s loyalty they may choose to protect your prey. This would involve the townsfolk to lie in an effort to send you in the wrong direction. We plan to use Madkit to create a simulation of a tracker chasing their prey through a series of ‘towns’ each populated with a differing number of agents with varying degrees of loyalty to the runner.

# Objectives

## Short Term

* (S1) Create a AI interface that will be able to transfer between points on a graph
* (S2) Create AI’s that have allegiance either to the runner or chaser AI
* (S3) Create a Chaser AI that will poll each point to determine where to go next
* (S4) Provide a visual representation of the simulation

## Long Term

* (L1) Allowing the runner and chasers presence to affect the AI’s at each point for positive or negative
* (L2) Supplying a much larger graph for the runner and chaser to move within
* (L3) Seeing the difference having multiple chasers has on the system
* (L4) Possible implementation within a video game environment

# Literature Review

Our project will utilize several different types of AI and their interactions. One key aspect which has had a lot of research is having AI traversing a graph. There are two algorithms which focus on fast path finding cooperation push and swap, and push and rotate. Push and Swap has two distinct operations. The push operation will move the input agent towards its going until its path is obstructed by another agent. The swap operation will direct the AI’s towards the nearest intersection to allow one to pass another. Push and Swap only provides an incomplete solution and so work began on push and rotate. Push and rotate works on the same principals as push and swap. However the rotate function cycles segments of the graph to maneuver agents. Using this method from tests provides a complete solution for maneuvering agents around a graph. However as our simulation will allow multiple agents at the same position, we will use a maneuvering system similar to the wumpus world problem.

The chasing AI will poll each location it visits akin to IBM’s Jeopardy competitor. Watson’s end result is an AI that will answer if its leading solution passes a confidence interval. The important factors to this project is how Watson determines its confidence interval. Researchers determined that for a quiz game like Jeopardy there was a correlation between how precise each answer was against how many questions were attempted. From this conclusion they developed a method which checks the information it knows with supporting evidence. Initially the system will deduct a number hypothetical solutions to the question. From there depending on the supporting evidence for each solution the system will rank the solutions and if the leading solution surpasses the others by a confidence interval. To develop the confidence interval IBM used machine-learning by applying a series of training questions with known answers.

# Methodology

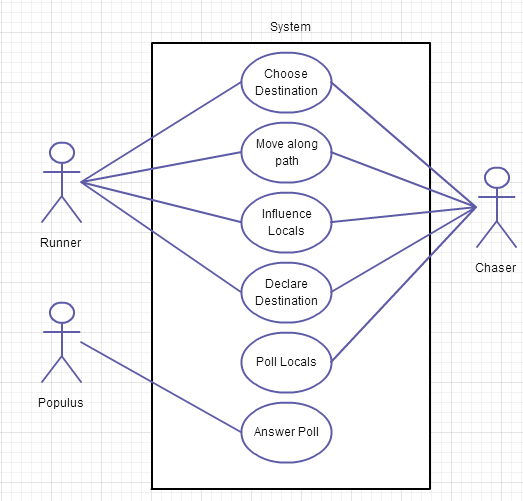


Figure Use case for agents within the simulation

## Short Term

(S1) This interface will allow agents the ability to transfer from one point on the graph to another. In order for the AI to make this decision it will use a move function. This function will take into consideration the distance between points as well as its current state as to where the agent will move to. The agent will also alert all of the agents at a point its next destination.

(S2) These AI’s will have a loyalty that will either be towards the runner or chaser. This value will be able to change depending on who visits the town. The agent will respond when questioned by the chaser with either the truth or a falsehood. This will depend on who the agent is loyal to. If the Agent tells the truth then it will respond with which direction the runner departed. Otherwise it will tell the chaser the runner went to any other connected city.

(S3) Using the interface from S1 we will create a faster moving AI that can poll each point. It will receive the runners graph as to see where the runner can move between. However for movement all points on the graph are connected. This AI will start a poll in each city, once the AI has passed a confidence interval it will move to the next node. This will require a minimum poll length which will differ as each point has a different population.

(S4) Using Madkit’s built in simulation tools we will develop a visualization of the simulation.

(S5) Movement and polling will take time, this will allow the chaser to move faster than the runner. However it also means that the runner can create a gap if the chaser takes too long to poll a node.

## Long Term

(L1) At the declaration of the where the runner or chaser intends to go next it will allow the Agents in the point to be influenced. The parameters for how they change will be global that we can manipulate to monitor how that affects the simulation.

(L2) We will construct a much larger graph to see the interplay between the runner and chaser on a much larger scale.

(L3) By implementing multiple chasers and seeing the how the effectiveness will change by having a group of co-operating chasing AI.

(L4) In the future we are looking at implementing this system into a video game to have chasing agents try to intercept a player as they move around in the world.

# Summary

This simulation will provide insight into continual effect on a system by multiple agents. From here we can create larger simulations will see the effectiveness of different searching algorithms and effects that the agents have on the system.

# References

Luna, R., & Bekris, K. E. (2011b). Push and Swap: fast cooperative path-finding with completeness

guarantees. *In Proceedings of the Twenty-Second international joint conference on Artificial*

*Intelligence – Volume One*, IJCAI, pp. 294–300. AAAI Press.

De Wilde, B., Ter Mors, A. W., & Witteveen, C. (2013). Push and Rotate: Cooperative multi-agent

path planning. *In Proceedings of the twelfth international conference on autonomous agents*

*and multiagent systems*, AAMAS, pp. 87–94, Saint Paul, Minnesota, USA.

Shapiro, Stuart C., and Michael Kandefer. "A SNePS approach to the wumpus world agent or Cassie meets the wumpus." *IJCAI-05 Workshop on Nonmonotonic Reasoning, Action, and Change (NRAC’05): Working Notes*. 2005.

Ferrucci, David, et al. "Building Watson: An overview of the DeepQA project."*AI magazine* 31.3 (2010): 59-79.