

Timothy Olson, Kirk Baly, Owen Prough, Alex Youngman

Prof. James Caristi

Research in Computer Science (CS 492)

4/22/10

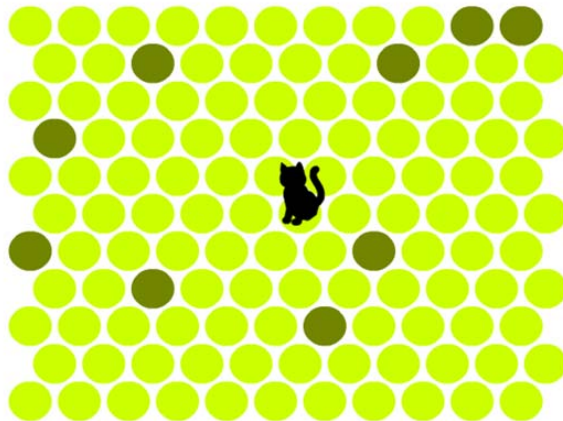
### *Circle the Cat: A Modern Adaptation and Analysis of a Classic Mathematical Game*

#### **Abstract**

Circle the Cat is an adaptation of the classic mathematical puzzle Quadraphage in which a cat attempts to escape a hexagonal board, but after each turn one square is blocked. We have programmed a simulation of this classic puzzle. The result of our development allows for the implementation of various strategies to computationally examine their effectiveness. The code itself, though extensive, is open enough for additions and extensions to be added fairly easily. Analysis of the simpler strategies has allowed for improvement of both Cat and Player AI systems.

#### **Introduction**

We developed a game based on a flash game found on the internet called Circle the Cat. Developed by Taro Ito, Circle the Cat is based on a classic mathematical game called Quadraphage. Fig. 1 shows a screenshot of the original version.



**Figure 1: Original game from <http://members.shaw.ca/gf3/circle-the-cat.html>**

The object of the game is to prevent the cat from reaching a boundary space by "blocking" spaces in such a manner as to halt its motion toward freedom. The blocker (user) and the cat alternate turns, with the blocker taking the first move. The blocker chooses one space per turn to block, after which the cat moves to any one of the six adjacent spaces, provided that the space in question is not blocked.

Quadrphage (Latin for "square eater") is a game played on a chess board of arbitrarily large size. One player is a chess king (capable of moving one space per turn), and the other is a quadrphage (who devours one square per turn). The objective of the king is to avoid being encircled by the quadrphage. A generalized version of this game is "Angel and the Devil" in which the Angel is similar to the king, but can move a specified number of squares per turn (not constrained to one), while the Devil fulfills the role of the quadrphage.

### Scope

The online version of the game does not permit the user to configure any initial settings. That is, the board is a fixed size, the initially blocked spaces are randomly distributed, the cat has but one strategy, and there is no artificial blocker intelligence. Our goal was to remedy these shortcomings by developing our own version from the ground up. We chose to program in Java because it is easily ported to a wide variety of operating systems. Our development scheme was intended to allow for easy modification of the code and straightforward implementation of new options and strategies.

We intended to create a model of the board such that the initial height, width, number and location of blocked spaces, cat position, etc. could be specified by the user at runtime. This would make analysis of winning (or losing) strategies given any initial conditions possible. We also intended to model the original cat's intelligence, as well as implement custom strategies developed by our research team. Another of our goals was to develop an artificial intelligence for the blocker to facilitate automated game play.

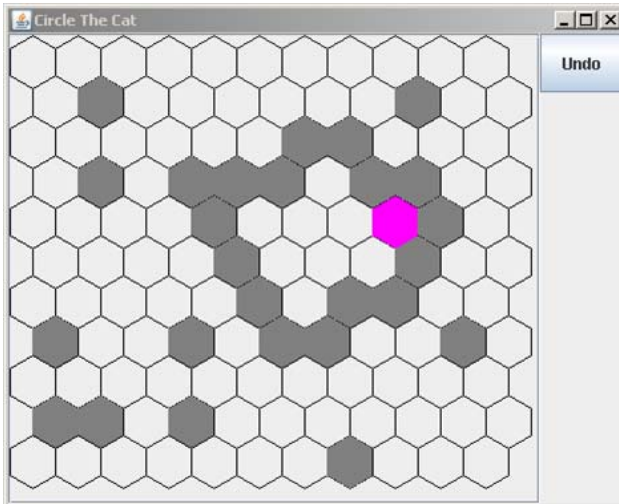
### Results to Date

We have successfully modeled the board in such a way as to allow runtime user configuration of board settings. The user can initialize the game using command line flags to set various parameters ahead of time, or change the settings through the user interface. Fig. 2 shows an example of the Graphical implementation of the initial configuration window. Three different interfaces have been created: Batch mode, Text mode, and Graphical mode. The batch interface is designed to facilitate analysis of computer vs. computer games. It writes the initial board parameters and results to a file. There is no user interaction (except to start the game). Text mode was the first playable interface; it models the board as 0's for free spaces, X's for blocked spaces, and a



Fig 2: Initial configuration window

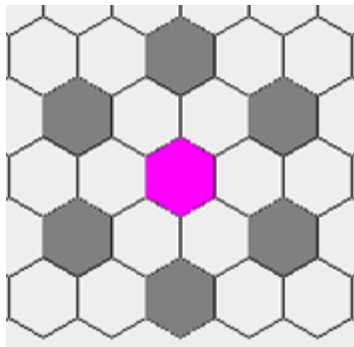
C for the cat. Spaces are blocked by entering the chosen space's coordinates. Some features, such as the ability to play as the cat, are not currently available for this mode. Graphical mode is the most user friendly mode, allowing the user to select spaces simply by clicking on them. The spaces are modeled as hexagons; light gray spaces are free, dark gray spaces blocked. The cat is represented by a bright magenta hexagon (though of course all of the colors can be easily changed).



**Fig. 3: Cat is surrounded and cannot escape. Game over, kitty.**

In both of the user interfaces, the user can choose to undo the previous move or moves. The board can determine when the cat is completely surrounded by an impenetrable wall of blocked spaces (as in Fig. 3), and can also tell when the cat has reached a boundary (Fig. 4). Both situations end the game, at which point the user is prompted to replay using the previous initial configuration or a new configuration, or quit.

blocker. The original cat's intelligence has been modeled by a "greedy" algorithm which chooses the shortest path to the boundary. Ties are broken by randomly selecting one of the

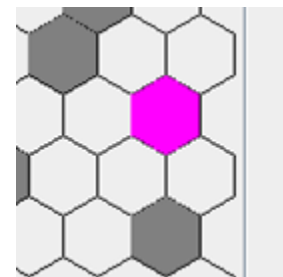


**Fig. 5: Circle of Death. The poor little cat has nowhere to run...**

shortest paths. Path distances are calculated by having each space broadcast its distance from the boundary. Each space then keeps track of the shortest distance it receives and uses that information to determine its own distance from the boundary. A very unintelligent cat strategy was also added in which the cat randomly moves to one of its available neighboring spaces.

A smarter cat searches ahead along the shortest paths to see if there are any obvious traps along the way. Primitive blocker intelligences are also in the early stages of development. One AI randomly blocks spaces on the board unless the cat is within one space of the boundary or if the next move could complete a circle of the cat. Another blocker AI attempts to trap the greedy cat with a "circle of death" (Fig. 5). The technique creates a trap to which the greedy cat is blind. As the cat moves through to the boundary, the player fills in the holes in the circle, thereby defeating the cat.

The state of the board is computed after every move whether by cat or



**Fig. 4: Cat has reached a boundary and escaped. Run little kitty, run!**

### **Future Development**

Some features that have yet to be implemented include improved documentation, such as a user-accessible description of rules and controls, and smarter artificial intelligences for both blocker and cat. It may also be interesting to allow multiple cats to be on the board simultaneously, or to create conditions in which the cat is occasionally able to "eat" a blocked space. Additionally, we have considered implementing some visual enhancements such as improved graphics, perhaps including the ability to use a picture of the user for the cat (particularly if the user is playing as the cat).

We have yet to complete any complex analysis of the strategies or winnability of various situations. One technique to do so would be to start with a large board on which the blocker wins consistently, and then shrink the board until the cat begins to win frequently. Cat strategy analysis could be done using the reverse technique, increasing the board dimensions until the cat no longer wins regularly. It may also be possible to make the cat or blocker learn by repeated play.

### **References**

Berlekamp, Elwyn, John Conway, and Richard Guy. *Winning Ways for Your Mathematical Plays*. 3. A K Peters Ltd, 2001. Print.

"Circle the Cat." <http://members.shaw.ca/gf3/circle-the-cat.html>

Ito, Taro (designer). GameDesign. [www.gamedesign.jp/flash/chatnoir/chatnoir.html](http://www.gamedesign.jp/flash/chatnoir/chatnoir.html) and private correspondence.