# MazeSolver.

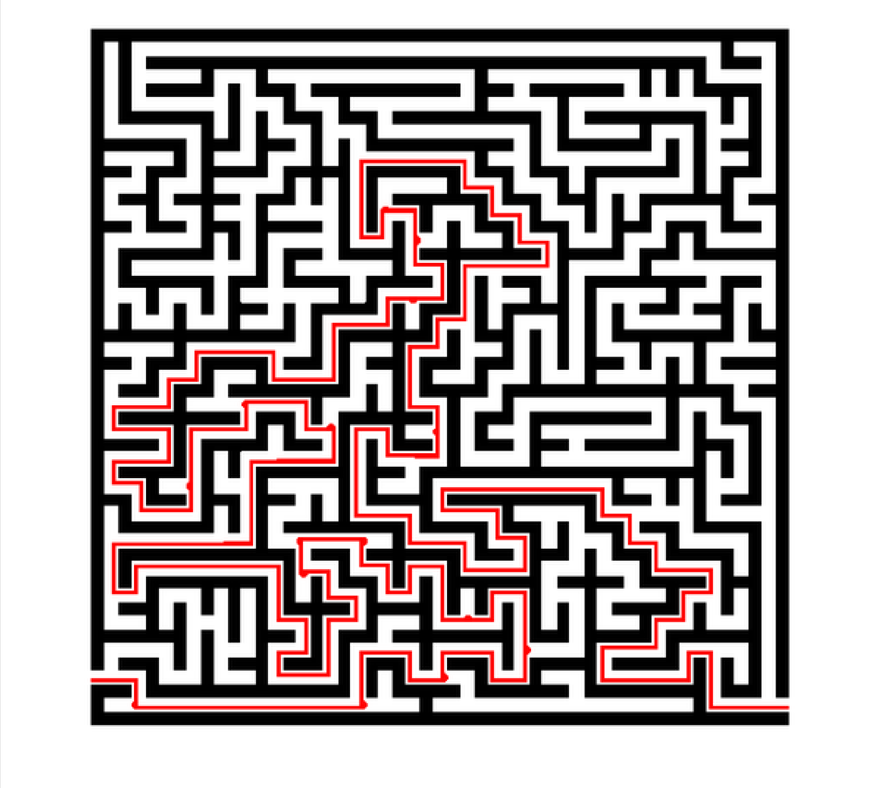
## Finding Solutions to Complex Mazes with Artificial Intelligence and Computer Vision

Jay Ricco
Computer Science
Mentor: Nate Derbinsky, Ph.D.

# The Problem

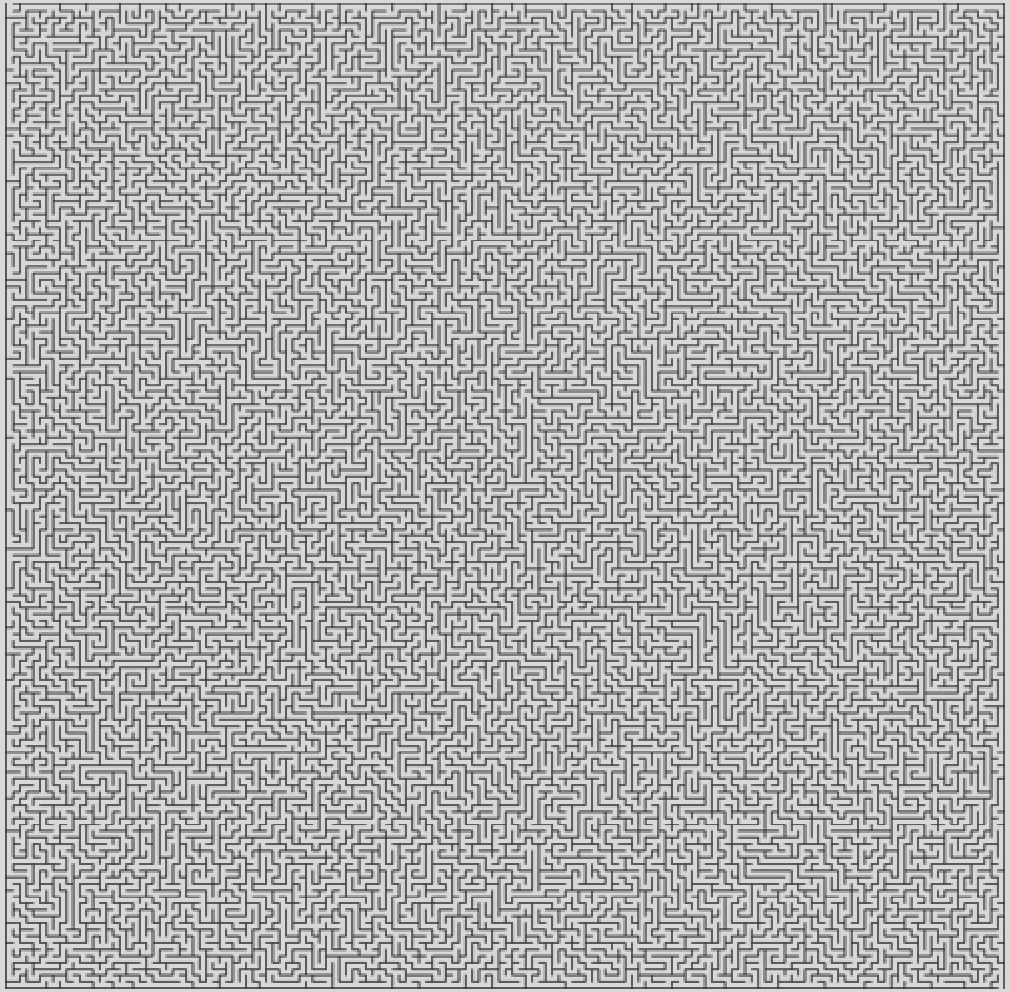
Input greyscale jpeg of rectangular maze:


Receive this:



# The Problem

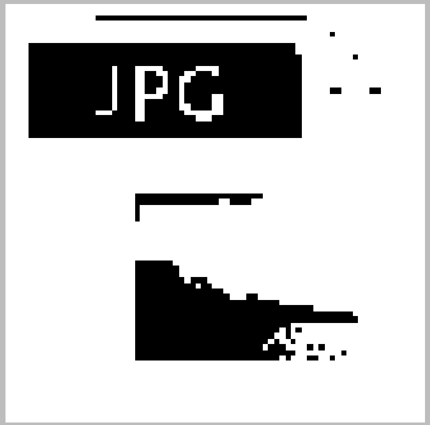
## Scalable



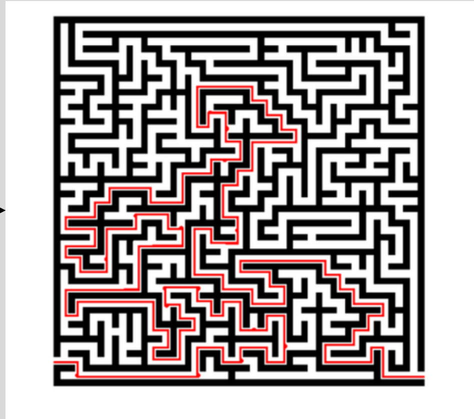
# The Approach

Improve data for max efficiency – not naïve

Evaluate data & Provide result


010110111…



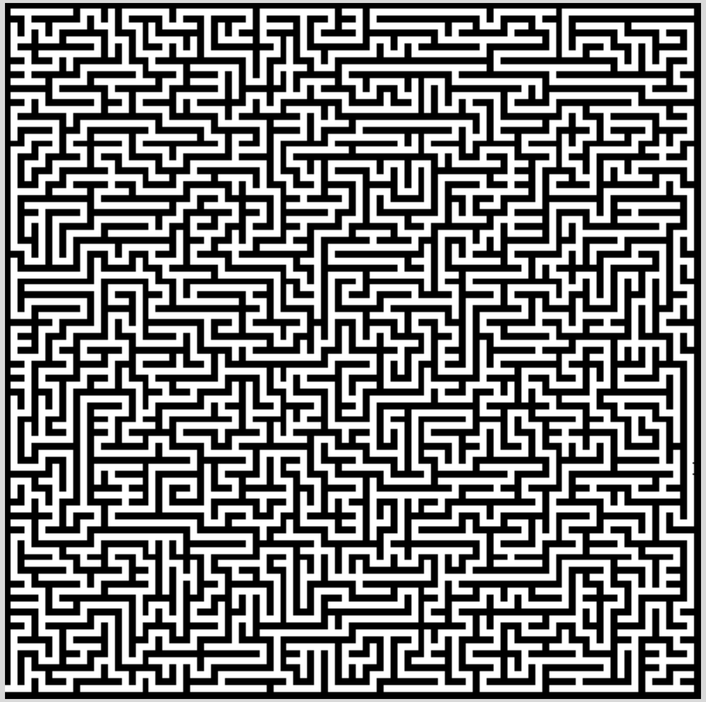
# The Approach – Efficiency

- Input is inefficient in it’s current format
- Convert data to allow for an efficient solution with optimal return
- Computational vision techniques allow for this conversion
- OpenCV is an open-source C++ computer vision library (http://opencv.org)
    - Stores images as 2D matrices
    - Contains many functions for easy manipulation of image data

We will convert our data by:

**Downsampling & Binning**

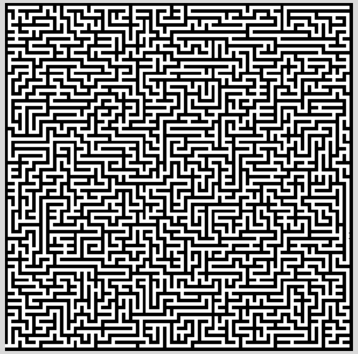
# The Approach – Downsampling



Jay Ricco

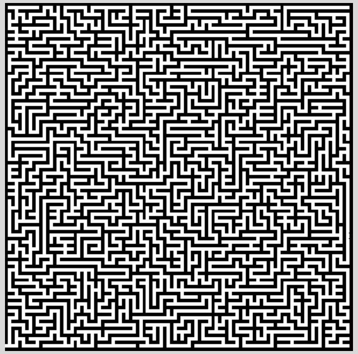
# The Approach – Downsampling

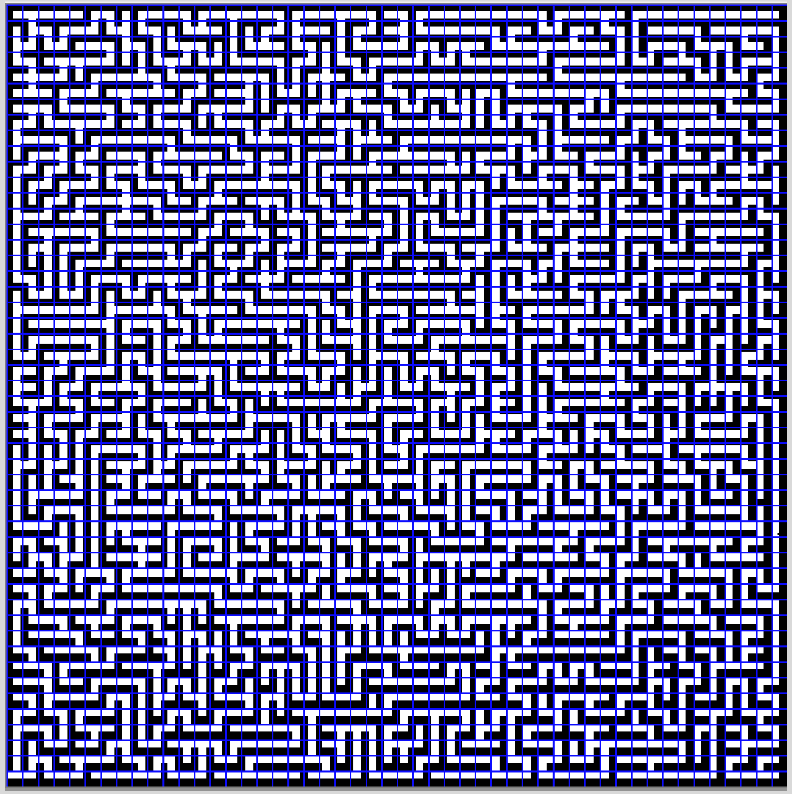
We want to import our image



# The Approach – Downsampling

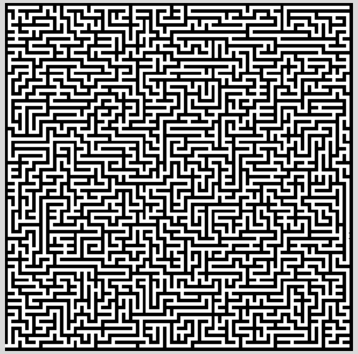
We want to import our image



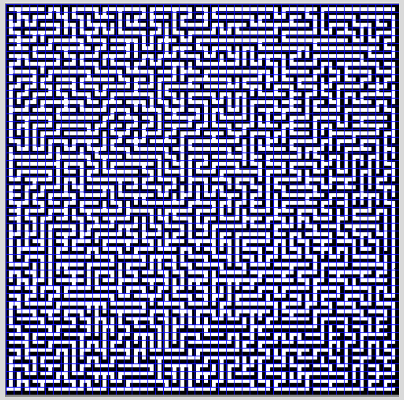


# The Approach – Downsampling

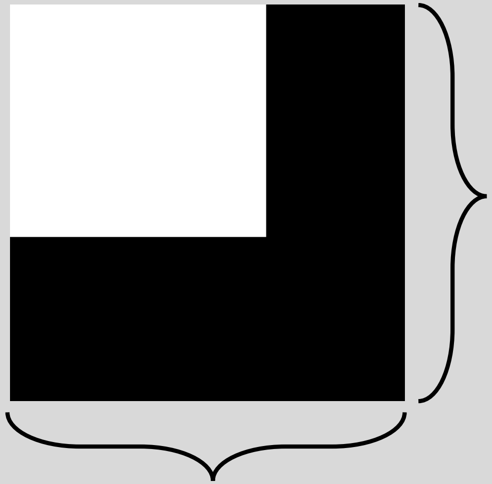
We want to import our image



Partition and sample uniformly



Obtain a set of the samples



height modifier

width modifier

# The Approach – Efficiency

- We now have tiles of the image, lets bin the data!
- Analyze each tile
  - Based on result, decide if tile is **walkable**, or **unwalkable**. (i.e. a wall or a corridor)
- Binning the values allows for more efficient data storage
  - Especially efficient for us as possibilities are binary
- Store the entire maze as a string of 1's and 0's
- Essentially performing "**lossy**" compression on data

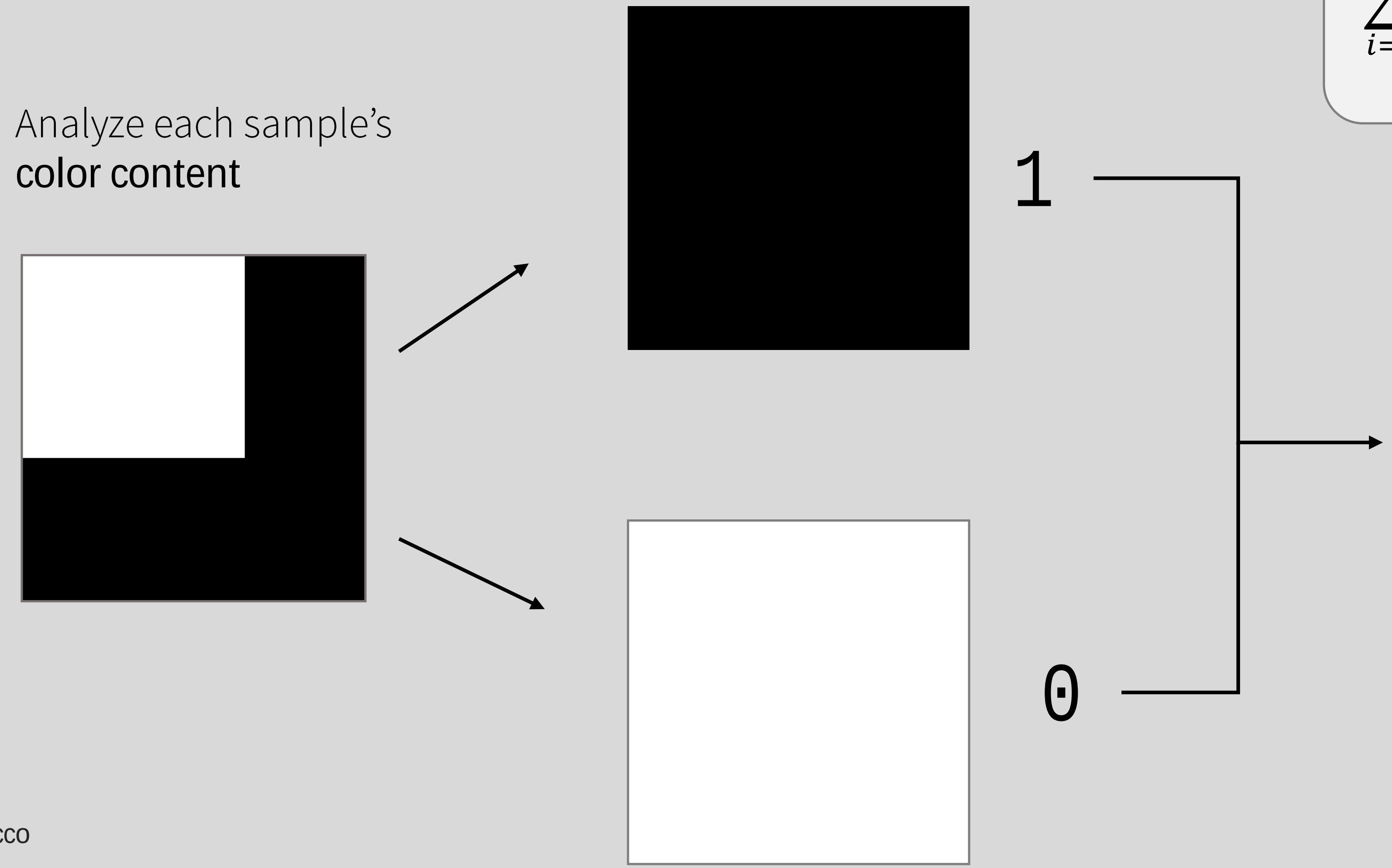
# The Approach – Binning

Bin the value

$$\sum_{i=0}^{n} \frac{P(i)}{n} < T$$

Where $n$ is the number of pixels per tile, $P(i)$ is the intensity value of the $i^{th}$ pixel, and $T$ is some threshold value.

Analyze each sample's
**color content**




01101001001010010100 1...

# The Approach – A* Search Algorithm

- A* is a graph-based heuristic search algorithm. (*Hart, Peter et al. 1968*)
  - Graph is what we're searching through
  - Heuristic is how we do it (cost based)

- Best-first search based on calculated movement cost
  - Lower cost is better

- Traverses the graph using a priority queue
  - If we get stuck, we can try paths we refused previously

- Looks for the lowest cost path between two points.

# The Approach – A* Search Algorithm

- A* uses a global cost function:

$$f(s) = g(s) + h(s)$$

Where $g(s)$ is the cost from the start to state $s$,
$h(s)$ is distance between state $s$ and the goal,
and $f(s)$ is the total worth of the movement from the current state to $s$.

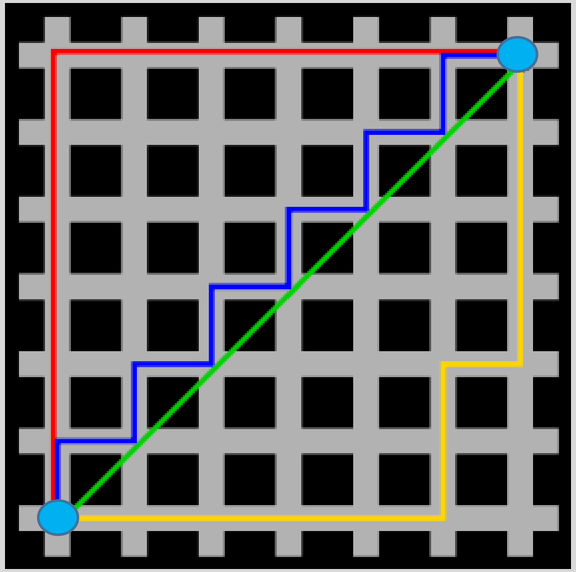
- Priority queue holds possible movements in order, based off of $f(s)$

- A* is *complete*, will always find an answer if it exists

- In our case, the heuristic is *admissible*: we will always have the shortest path to the goal.

# The Approach – A* Search Algorithm

- $h(s)$ found by Manhattan Distance → the Euclidean $l1$ norm.
- $h(s) = \ ||s\ - c||$
  Where $c$ is the current state.
- Ideal for our search space.

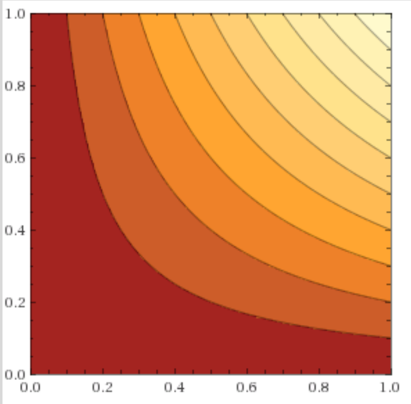


# The Approach – Theoretical Time Complexity

- **Time Complexity:**
  - Downsampling Algorithm: $O(x * y)$, linear depending on number of tiles
  - A* in absolute worst case is $O(b^d)$, for us it's $O(N^2)$,

  Where $b^d$ is the number of options per intersection to the power of the max depth,

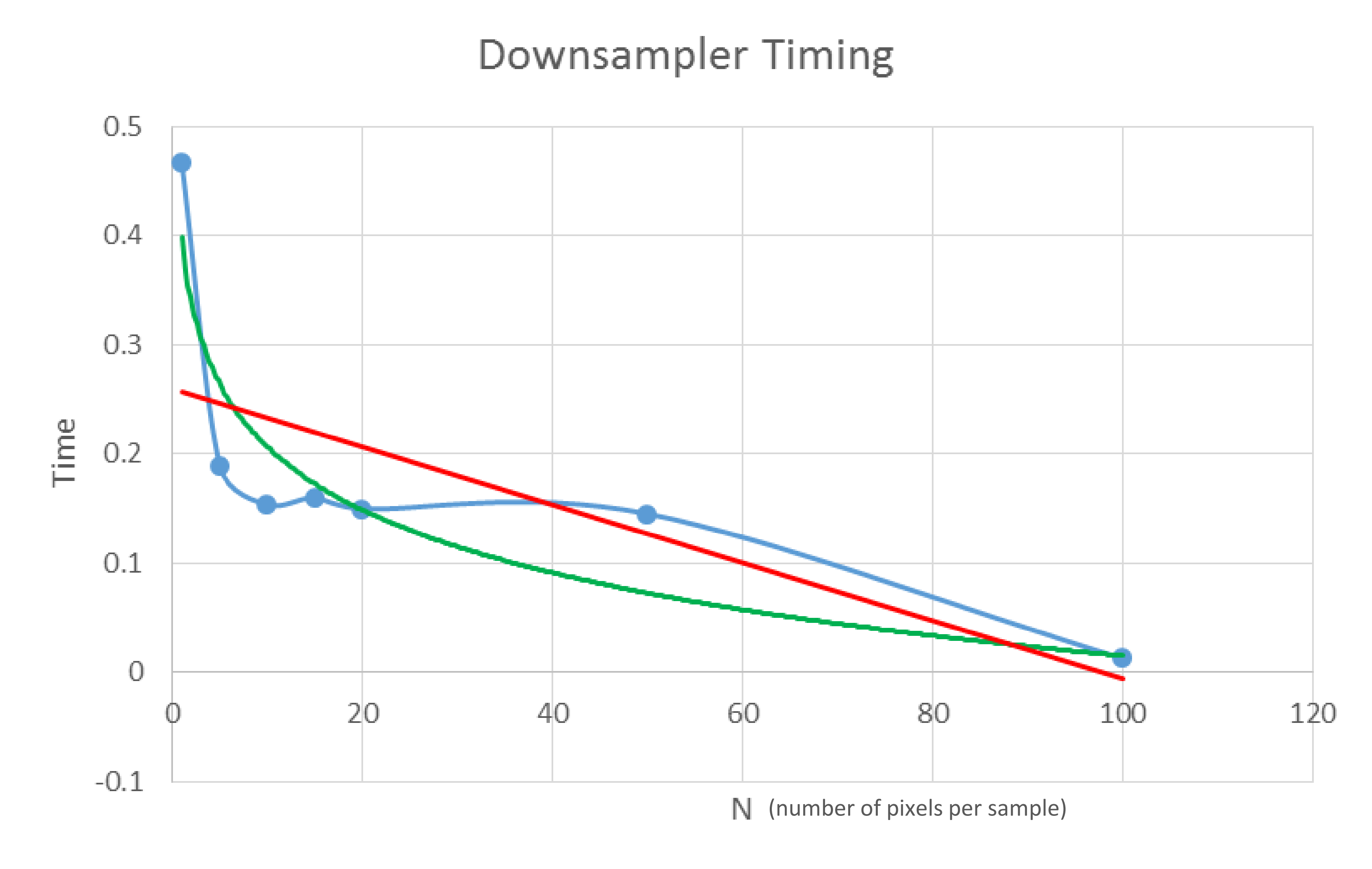
  And $N$ is the number of possible states in the search space.




Contour plot: x*y

MazeSolver.

# The Approach – Empirical Specifics




- Actual Data
- Logarithmic Trend line
- Linear Trend line

Jay Ricco

# Summary

- Input our image file
- Improve on naïve by increasing efficiency
  - Downsampling
  - Binning

Larger granule size = more efficient

- Use A* and an admissible heuristic to find most optimal path efficiently

# Summer Research

## Goal: General Intelligence

- Gathering inspiration from humans: **memory**
- Allow for the recognition and storing of an environment
- Allows for building upon previous experience
- Without memory, AI agents would not be knowledgeable

# Summer Research

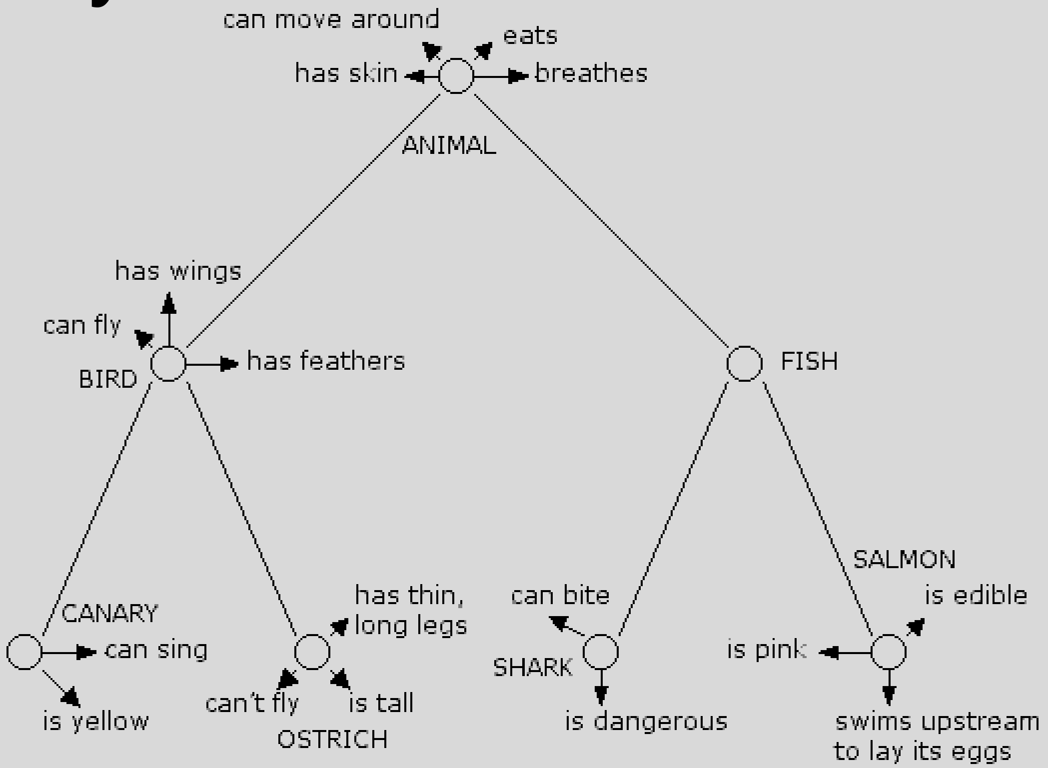
## Memory

- Semantic (Factual)
- Episodic (Little movie in your head)
- **Issue:** Effective and efficient acquisition, storing, and retrieval of large amounts of knowledge

# Summer Research

## Semantic Memory

- Gathering facts about environment
- Efficient retrieval of these facts
- Forgetting over time
- **CONTEXT!!!**

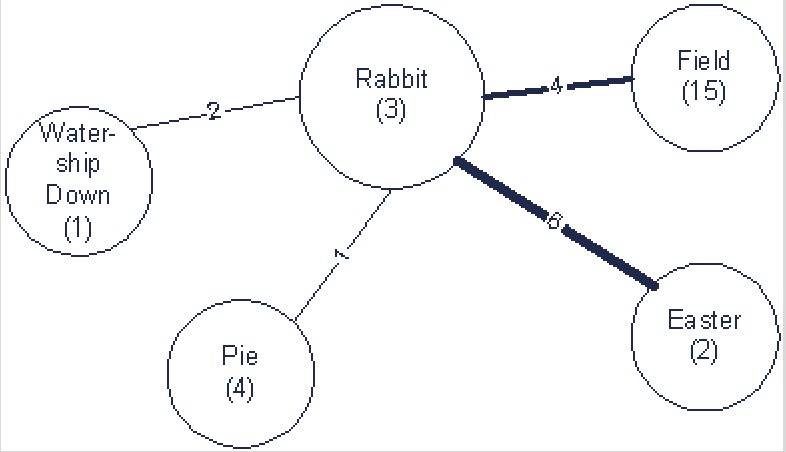



From Collins, A. M. and Quillian, M. R. (1969).
Retrieval Time from Semantic Memory.
*Journal of Verbal Learning and Verbal Behavior*, *8* (2) 240-247.

# Summer Research

## Semantic Memory Context via Spreading Activation

- Based on recently accessed memories
- Relation to other memories
- Allow for forgetting of unused information
- Graph of activation allows for situational context



# Questions?

**E-mail**:
riccoj@wit.edu

**Website**:
jayricco.github.io