**CSCI 446 - Artificial Intelligence: Project 2**

*Search, Constraint Satisfaction, and Graph Coloring*

**Report**

< Version 0.01 >

Emmanuel Armstrong

Matthew Rohrlach

Nicolas Moore

2016–9–15

**Surviving Wumpus World using First-Order Logic**

October 24, 2016

**Nicolas Moore** (nicolas.moore88@gmail.com)

**Matthew Rohrlach** ([mattrohrlach@gmail.com](mailto:mattrohrlach+ml447@gmail.com))

**Emmanuel Armstrong** ([emmanuelrarmstrong@gmail.com](mailto:emmanuelrarmstrong@gmail.com))

1. **Abstract:**

The Wumpus World is a problem presented in Artificial Intelligence that consists of creating a cell based world and to create an agent to be sent into this world. The goal is for the agent to traverse the world and find the gold while avoiding deadly obstacles such as pits and monsterous Wumpuses. In this paper we used First-Order logical operations to create the reasoning inference method for the agents. The reasoning algorithm would use this logic to make decisions based off of what it knows about the world, and what it can sense from each square it is in. There is also a reactionary agent that only makes decisions based on what it senses at that particular moment. The reasoning agent was able to solve almost every solvable world given to it, even at the largest world size, which is 25x25. The reactionary could solve the smaller worlds very well, but as the world size increased, its solvability decreased drastically.

1. **Problem Statement/Hypothesis:**

The purpose of this project is to create a knowledge-based, logically-reasoning agent with the ability to make decisions towards the successful traversal of varying “Wumpus worlds”. The agent must use a  first-order logic reasoning system to decide the next appropriate action. The goal of the agent is to reach the cell containing the “gold” before succumbing to the hazards placed randomly throughout the world. The first task of our software will be to generate the Wumpus worlds . Each Wumpus world will consist of a grid in sizes ranging from 5 by 5 to 25 by 25 cells.

The next task will consist of placing this reasoning agent within the world. A reactive agent, described below, will also be placed into their own instance of the same Wumpus world; to provide a comparison. Both agents will then traverse the world until they die, determine they cannot reach the gold, or reach the gold. Of the two, the reasoning agent will traverse the world by maintaining a knowledge base of first-order logical facts in clause form; then use that knowledge to make inferences about the unknown aspects of the Wumpus world and what action would be most-appropriate to take next. In comparison, the reactive agent will traverse the Wumpus world by performing random moves to cells it believes to be safe, or by moving to an unsafe cell if there are not any safe cells available.

Further details regarding the implementation of the Wumpus world generator, reasoning agents, and reactive agents are contained in Section 3 of this document. Our experimental designs, the structure of the tests we will perform, the parameters of such tests, and the comparisons that we will draw between the reasoning agents and the reactive agents may be found in Section 4 of this document.

It is expected that the reasoning inference will be able to solve most of the worlds that are solvable, and also get a high percentage of Wumpuses killed. The reactionary agent is expected to perform much worse, and while it should still be able to solve a fair amount of the worlds, it will do so much less efficiently than the reasoning agent.

1. **Algorithms Implemented:**

In this project, software elements were written such that every class would represent an object in a practical example of a Wumpus world. For example, the world will be written as a self-contained class that accepts and carries out actions affected upon it by the agent class. The agent class will share no methods with the world class, but both classes will maintain certain elements in redundancy; such as the current position of the agent. The agent class will contain instances of logically-reasoning or reactive “inference” classes to act as decision-makers. When the appropriate inference class has reached a decision, it will pass this decision in the form of a string object to the agent; who will then pass this decision back to the Wumpus world class to be parsed into an action. Stored within the reasoning inference class will be both a knowledge bank and a set of first-order logical rules. By updating this knowledge bank at every movement, a list of visited, safe-yet-unvisited (e.g. no suspicions), and suspicious cells can be maintained by the agent’s sense of reasoning. To follow the project rules, these fragments of knowledge will then be applied in first-order logic rules in clause form to determine an appropriate action. This object-based design, and the familiarities of the project authors, left the Java language as the appropriate choice of programming language.

* 1. **Wumpus World Generator**

As mentioned in Section 1, the Wumpus world generator builds Wumpus worlds of sizes ranging from 5 by 5 to 25 by 25 cells. Each increment in grid size will occur by 5 cells (i.e. 5 by 5 becomes 10 by 10, 15 by 15 becomes 20 by 20, etc.). A cell consists of an Integer key, five boolean values that respectively determine if the cell is empty, contains a pit, contains an obstacle, contains a Wumpus, or contains gold. A cell also includes a list of cell keys that are connected to itself. As input, the Wumpus world generator’s constructor will require an agent, either reasoning or reactive, an integer denoting the side dimension of the Wumpus world, measured in cells, an integer to denote the probability of generating a pit in a cell, an integer to denote the probability of generating an obstacle in a cell, and an integer to denote the probability of generating a Wumpus inside a cell.

The Wumpus world constructor will initialize a cell array with size equal to the side dimension squared. A helper function called by the constructor will perform the work of actually building the Wumpus world. Given the integer probability values, the helper function will generate cells incrementally until the constraints of the world are met. During the generation of a cell, a key value will be assigned to the cell and, based on the aforementioned probability values, the helper function will determine whether or not to place a pit, an obstacle, or a Wumpus in the given cell. The boolean values for the cell will be set accordingly, and the cell will be added to the cell array stored by the world. The southwest cell will always be made empty, so that it may be used as a safe starting point for the agent. Once all the cells have been created, each cell will then have its list of connected cell keys populated, and a list of empty cells, not containing the first cell, will be created. One cell from the list of empty cells will be chosen at random and the gold will be placed in that cell. After the gold is placed, the agent will be placed in the southwest corner cell.

The Wumpus world will be responsible for keeping track of the agent's current location. The Wumpus world will also be responsible for accepting and executing the actions of the agent, including arrows fired.

* 1. **Reasoning Inference**

When the current position is passed to the reasoning inference class contained within the agent class, the object will automatically update the knowledge base of that agent with the implication of the current position being safe. The knowledge base consists of “knowledge fragment” objects, each representing a cell that the agent is aware of. Each of these fragments will maintain a list of connected fragments, drawn from the physical cell’s list of connected cells. Each fragment will also contain lists that represent the suspicions that fragment has about its neighbors Wumpus and pit states, respectively. The smell or wind of the current position will affect both this current fragment’s suspiciousness in neighboring lists, as well as the lists maintained by this fragment itself.

At every movement, the knowledge bank of the inference class will be updated in a linear process. After the current position has been matched to the fragment that stores information known about that position, the fragment will remove itself from the suspicions of every neighboring cell. If a previous cell had a smell or breeze, such a report would imply that a neighboring cell had a hazard, and all non-safe neighbors would be suspected of harboring that hazard. With suspicion removed from the current position, if there is only a single remaining suspicious neighbor in a list, a conclusion can be drawn about a hazard within that neighbor. A conclusion drawn about a pit would be permanent, but, in the case of a Wumpus, a conclusion would be tentative. A Wumpus may be killed by an arrow fired from a distance, so this conclusion would be subject to change in the face of an updated smell report at a neighbor. A conclusion about an obstacle could only be drawn by an attempt to move into a cell, so this conclusion would not be drawn through suspicions. When the knowledge bank is properly maintained, it will provide a “frontier” of safe, non-visited cells and safe, visited cells. The distinction between the two is such that a visited cell is a guaranteed traversable path, but an obstacle might still exist in a non-visited cell. All cells blocked with obstacles will be permanently marked as non-traversable (e.g. not safe).

The decision-making process will be structured through the aforementioned first-order logic rules. The current plan of implementation is to use these rules to send the agent into safe, non-visited cells until the gold is found or there are no more safe non-visited cells. The visited cells of the world will function as a road between such safe, non-visited cells that may exist at opposite ends of the world. When there are no more safe, non-visited cells to explore, the agent will use the first-order logic rules to choose a Wumpus-containing cell with an unvisited, unmarked neighbor. The agent will then fire-into and traverse this cell, ideally opening a new avenue of movement. If no such cells exist, the agent will use the aforementioned logic to find any unvisited cell that may be visited without guaranteed failure. If no such cells still exist, the agent will report their failure to the world. When the gold has been reached, regardless of the path taken, the world will recognize this fact and automatically report success to the agent.

* 1. **Reactionary Inference**

To act as a control in experiments concerning the reasoning agent, a reactive agent will be introduced as an alternative choice in decision-making. The reactive agent will contain a reactionary inference class, designed not to store knowledge, but to act upon the best conclusion that can be drawn about the agent’s immediate surroundings. For example, if the agent reaches a cell with both a smell and wind, and the current position has three neighbors, the best course of action would be to return to the previous cell. Such an action would be governed purely by the agent’s reaction to an immediate condition; not by conclusions drawn in the context of broader knowledge.

To accomplish this task, the agent will consult a set of logical rules that, in contrast to the reasoning agent, will be run on current conditions only.

1. **Experimental approach:**

In order to compare the reasoning and reactive agents, a set of one hundred Wumpus worlds will be generated for each size: 5 by 5, 10 by 10, 15 by 15, 20 by 20, and 25 by 25 cells. Each agent will traverse the same set of Wumpus worlds to ensure that no agent has an easier set of Wumpus worlds relative to the other agents. During an agent’s traversal of the Wumpus world, many metrics, described below, will be collected in order to form a means for comparing the agents.

* 1. **Planned Parameter Tuning**

For this version of the Wumpus World, there will be specific parameters that must to be tuned to achieve accurate results.  The three primary parameters that will be used in this example will be: the max number of actions that the agent is allowed in each world, the number of worlds to send the agent through in each world size, and the probability that a hazard will be placed in a given cell.  These variables will be very important in producing accurate results from the agent that can be compared to the results of other agents ran through the world.

The “max number of actions” given to each agent will provide an adequate sample of decisions made in order to illustrate how each agent deals with more complex worlds.  This parameter will also provide a good level for comparing the reactionary inference to the reasoning inference.  The reasoning inference should solve harder worlds much more consistently, and this parameter will allow better illustrations of its efficiency.

The “number of worlds” parameter is fairly straightforward.  The more worlds of a given size that an agent is sent through, the more data that will be collected to compare and interpret.  The experimentation done in this project implementation will run one hundred different worlds of each world size.  The data gathered from each set of worlds will be averaged and normalized into a single metric to be compared between agents.

Finally, the probability that a hazard will be placed within a cell will be used to fine-tune the difficulty of the worlds to be explored.  This will help to illustrate the efficacy of each agent in regards to world-solving.  If the program reaches a the point where no agents are completing worlds, it will be clear that the difficulty is set too high. If this situation occurs, this parameter can be tuned once more to find the optimal difficulty for accurate results and data gathering. Determining the parameter values that produce the strongest results will be a major focus of this experimentation.

* 1. **Variables to Track for Comparison**

The main metric that will be tracked for comparison will be the average score obtained by each agent when a set of worlds has been completed, under a given hazard probability setting.  This average will be saved into the points variable. A higher average score in points will obviously indicate a more successful agent, while a lower score will show a less successful solution.   Because of this, the agents are able to be compared, and a conclusion about efficiency made with reasonable confidence.

The size of the world that the agents are running on will also be tracked.  This variable is very important for accurate data collection.  Each agent must be positive that it is being run on a world of the same size, or else the results would be very different for each.

The final variable to be tracked is the probability of hazard spawns, in other words, pitProbability and wumpusProbability respectively.  These variables will be tracked for comparison once the agents have completed their exploration of every world.  The agents must have worlds that are generated with similar levels of difficulty for proper testing.  If one agent has a world with twice as many Wumpuses or pits as a different one then it will provide very skewed results for comparison.

**4.3  How Comparisons are Created**

As the agent moves through the world its score will be constantly monitored.  There will exist a set starting number of points and each action made will have an effect on the score.  At the end of each iteration, the final value stored in the points variable will be saved and the next world will be initialized.  At the point where all of the worlds of a particular size have been completed, the stored points will be averaged together for each survivor. The agent will know when points are edited due to its senses in the world, as well as the default actions that are always documented.  For example, when a scream is detected, the agent will be told to add 10 points to its points variable.  This is performed for every action and reaction that an agent makes.

The probabilities for hazard spawns will be saved each time a world is generated.  The program will also compare these probabilities for each iteration.  This is to guarantee the worlds are all of a similar difficulty, thus providing accurate result collection.  This probability will stay the same for every world until the program ends, and then will be adjusted accordingly for future tests.

The program will also keep track of the size of the world that is being tested for this set of iterations. The world size will be saved as a variable; and this variable will remain the same for the set number of iterations before increasing to match the following world size.   This process will repeat until the max size of 25x25 is reached.  When this has occurred, the agents will be run for the set number of actions, and the final averages calculated.

The strongest agents, those with the highest average scores for all worlds, will be deemed the best.  The size of the worlds will also affect the difficulty set on the agents.  If the worlds are bigger there are more possible cells for hazards to appear in, however, there are more possible routes through the world that could be utilized by an agent.  Accordingly, the project will take this into account when setting the hazard probabilities in different-size worlds.

In the final report, these various conclusions will be tabulated and formed into paragraphs that include the reasoning taken in choosing their parameters; as well as supporting charts and graphs to illustrate their meaning within the scope of the project as a whole.

1. **Results:**

Ensure solvability: true

Sound: false

World square dimensions: All x All

Decision maker chosen: All

Number of worlds: 50

Max actions per world: Default

Probability of bottomless pit appearance: 33 percent

Probability of obstacle appearance: 33 percent

Probability of Wumpus appearance: 33 percent

=======Results of REASONING 5 x 5 (max 75 actions per world)========

====POINT METRICS====

Total times gold found: 49/50

Average points per explorer: 666.12

====EXPLORATION METRICS====

Total cells explored: 488

Average cells explored per explorer: 9.76

====DEATH METRICS====

Total times died to Wumpus: 1

Total times died to pit: 13

Total times died: 14

Total times max actions reached: 1

Average deaths to Wumpus per explorer: 0.02

Average deaths to pit per explorer: 0.26

Average deaths per explorer: 0.28

====ARROW METRICS====

Total Wumpuses killed: 36

Average arrows remaining per explorer: 56.451612903225815 percent

Average arrow accuracy per explorer: 66.66666666666666 percent

=======Results of REASONING 10 x 10 (max 300 actions per world)========

====POINT METRICS====

Total times gold found: 50/50

Average points per explorer: -372.68

====EXPLORATION METRICS====

Total cells explored: 1950

Average cells explored per explorer: 39.0

====DEATH METRICS====

Total times died to Wumpus: 5

Total times died to pit: 54

Total times died: 59

Total times max actions reached: 0

Average deaths to Wumpus per explorer: 0.1

Average deaths to pit per explorer: 1.08

Average deaths per explorer: 1.18

====ARROW METRICS====

Total Wumpuses killed: 218

Average arrows remaining per explorer: 46.45941278065631 percent

Average arrow accuracy per explorer: 70.3225806451613 percent

=======Results of REASONING 15 x 15 (max 675 actions per world)========

====POINT METRICS====

Total times gold found: 50/50

Average points per explorer: -1143.72

====EXPLORATION METRICS====

Total cells explored: 3872

Average cells explored per explorer: 77.44

====DEATH METRICS====

Total times died to Wumpus: 4

Total times died to pit: 83

Total times died: 87

Total times max actions reached: 0

Average deaths to Wumpus per explorer: 0.08

Average deaths to pit per explorer: 1.66

Average deaths per explorer: 1.74

====ARROW METRICS====

Total Wumpuses killed: 465

Average arrows remaining per explorer: 48.631239935587764 percent

Average arrow accuracy per explorer: 72.88401253918495 percent

=======Results of REASONING 20 x 20 (max 1200 actions per world)========

====POINT METRICS====

Total times gold found: 47/50

Average points per explorer: -1878.32

====EXPLORATION METRICS====

Total cells explored: 6696

Average cells explored per explorer: 133.92

====DEATH METRICS====

Total times died to Wumpus: 2

Total times died to pit: 101

Total times died: 103

Total times max actions reached: 3

Average deaths to Wumpus per explorer: 0.04

Average deaths to pit per explorer: 2.02

Average deaths per explorer: 2.06

====ARROW METRICS====

Total Wumpuses killed: 752

Average arrows remaining per explorer: 53.30337078651686 percent

Average arrow accuracy per explorer: 72.37728585178056 percent

=======Results of REASONING 25 x 25 (max 1875 actions per world)========

====POINT METRICS====

Total times gold found: 45/50

Average points per explorer: -3404.96

====EXPLORATION METRICS====

Total cells explored: 10581

Average cells explored per explorer: 211.62

====DEATH METRICS====

Total times died to Wumpus: 15

Total times died to pit: 133

Total times died: 148

Total times max actions reached: 5

Average deaths to Wumpus per explorer: 0.3

Average deaths to pit per explorer: 2.66

Average deaths per explorer: 2.96

====ARROW METRICS====

Total Wumpuses killed: 1141

Average arrows remaining per explorer: 53.259898779398625 percent

Average arrow accuracy per explorer: 72.67515923566879 percent

=======Results of REACTIONARY 5 x 5 (max 75 actions per world)========

====POINT METRICS====

Total times gold found: 38/50

Average points per explorer: 145.18

====EXPLORATION METRICS====

Total cells explored: 460

Average cells explored per explorer: 9.2

====DEATH METRICS====

Total times died to Wumpus: 3

Total times died to pit: 24

Total times died: 27

Total times max actions reached: 12

Average deaths to Wumpus per explorer: 0.06

Average deaths to pit per explorer: 0.48

Average deaths per explorer: 0.54

====ARROW METRICS====

Total Wumpuses killed: 55

Average arrows remaining per explorer: 46.808510638297875 percent

Average arrow accuracy per explorer: 73.33333333333333 percent

=======Results of REACTIONARY 10 x 10 (max 300 actions per world)========

====POINT METRICS====

Total times gold found: 32/50

Average points per explorer: -1686.2

====EXPLORATION METRICS====

Total cells explored: 1348

Average cells explored per explorer: 26.96

====DEATH METRICS====

Total times died to Wumpus: 15

Total times died to pit: 83

Total times died: 98

Total times max actions reached: 18

Average deaths to Wumpus per explorer: 0.3

Average deaths to pit per explorer: 1.66

Average deaths per explorer: 1.96

====ARROW METRICS====

Total Wumpuses killed: 223

Average arrows remaining per explorer: 30.324909747292416 percent

Average arrow accuracy per explorer: 57.77202072538861 percent

=======Results of REACTIONARY 15 x 15 (max 675 actions per world)========

====POINT METRICS====

Total times gold found: 23/50

Average points per explorer: -6639.76

====EXPLORATION METRICS====

Total cells explored: 3346

Average cells explored per explorer: 66.92

====DEATH METRICS====

Total times died to Wumpus: 23

Total times died to pit: 278

Total times died: 301

Total times max actions reached: 27

Average deaths to Wumpus per explorer: 0.46

Average deaths to pit per explorer: 5.56

Average deaths per explorer: 6.02

====ARROW METRICS====

Total Wumpuses killed: 546

Average arrows remaining per explorer: 27.48031496062992 percent

Average arrow accuracy per explorer: 59.28338762214984 percent

=======Results of REACTIONARY 20 x 20 (max 1200 actions per world)========

====POINT METRICS====

Total times gold found: 19/50

Average points per explorer: -11377.12

====EXPLORATION METRICS====

Total cells explored: 5139

Average cells explored per explorer: 102.78

====DEATH METRICS====

Total times died to Wumpus: 36

Total times died to pit: 454

Total times died: 490

Total times max actions reached: 31

Average deaths to Wumpus per explorer: 0.72

Average deaths to pit per explorer: 9.08

Average deaths per explorer: 9.8

====ARROW METRICS====

Total Wumpuses killed: 865

Average arrows remaining per explorer: 31.540565177757518 percent

Average arrow accuracy per explorer: 57.58988015978696 percent

=======Results of REACTIONARY 25 x 25 (max 1875 actions per world)========

====POINT METRICS====

Total times gold found: 23/50

Average points per explorer: -15369.24

====EXPLORATION METRICS====

Total cells explored: 7451

Average cells explored per explorer: 149.02

====DEATH METRICS====

Total times died to Wumpus: 32

Total times died to pit: 623

Total times died: 655

Total times max actions reached: 27

Average deaths to Wumpus per explorer: 0.64

Average deaths to pit per explorer: 12.46

Average deaths per explorer: 13.1

====ARROW METRICS====

Total Wumpuses killed: 1268

Average arrows remaining per explorer: 35.53124100201555 percent

Average arrow accuracy per explorer: 56.632425189816885 percent

1. **Discussion of Algorithmic Behavior:**

As expected the reasoning agent was able to solve worlds much more efficiently than the reactionary. It was also able to solve more of the solvable worlds, due to the fact that the reactionary agent ran out of moves before it could solve some of the worlds. Once the program got to the larger worlds this was even more apparent.

The reasoning agent was able to discern where Wumpuses were spawned and either dodge or kill them most of the time. Even on the largest maps the average death due to a Wumpus was 0.3 times. On the opposite side they were able to kill almost all the Wumpuses on each map. Pits proved to be a bit more of an issue for the reasoning agent, due to the fact that there is no way to destroy the pit, as there is with a Wumpus. However, even then they were able to avoid the pits most of the time. In the largest worlds the average number of deaths to a pit was only 2.66. This was in a set of 50 worlds of each size where all worlds were solvable.

The reactionary agent had a much harder time solving worlds. Due to the fact that it made decisions based only off of what is known at that moment it took a lot more moves to get to the gold, and many times this would cause it to run out of moves before finding the gold. While it got killed by more Wumpuses than the reasoning did, it still did a fairly good job of avoiding and/or killing them. The average number of deaths by a Wumpus was still under 1 for each explorer, even on the largest worlds. However pits proved to be even more of a problem for these agents than they were for the reactionary agent. By the time it was ran on the larger worlds the average deaths due to pits was 12.46 for each explorer. This was in a set of 50 worlds of each size where all worlds were solvable.

1. **Summary:**

The Wumpus World problem turned out to be a very good look into logic based reasoning and problem solving. It relies on the understanding of this logic to be implemented correctly, both from the programmer, and the agent that is created.

**References:**