AN OPTIMIZED SOLUTION FOR WATER-JUG AND 8-PUZZLE PROBLEM

[1]PeddyReddy. Swathi, SQL Server DBA/Salesforce Admin, Buffini & Company, Carlsbad,CA, USA

[2]Dr. Narsimha Rao, Professor, Kalinga University, India

Abstract

To solve a problem using a production system, we must specify the global database the rules, and the control strategy. For the 8 puzzle problem that correspond to these three components. These elements are the problem states, moves and goal. In this problem each tile configuration is a state. The set of all configuration in the space of problem states or the problem space, there are only 3, 62,880 different configurations o the 8 tiles and blank space. This paper provides an Optimized solution for Water-jug and 8-puzzle problem.

Index Terms: Artificial Intelligence, algorithm, Machine Learning.

I. The definitions of AI

a) "The exciting new effort to make computers think . . . *machines with minds,* in the full and literal sense" (Haugeland, 1985) "The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning..."(Bellman, 1978)	b) "The study of mental faculties through the use of computational models" (Charniak and McDermott, 1985) "The study of the computations that make it possible to perceive, reason, andact" (Winston, 1992)
c) "The art of creating machines that perform functions that require intelligence when performed by people" (Kurzweil, 1990) "The study of how to make computers do things at which, at the moment, people are better" (Rich and Knight, 1 99 1)	d) "A field of study that seeks to explain and emulate intelligent behavior in terms of computational processes" (Schalkoff, 1 990) "The branch of computer science that is concerned with the automation of intelligent behavior" (Luger and Stubblefield, 1993)

The definitions on the top, **(a)** and **(b)** are concerned with **reasoning**, whereas those on the bottom**, (c)** and **(d)** address **behavior.**The definitions on the left, **(a)** and **(c)** measure success in terms of human performance, and those on the right, **(b)** and **(d)** measure the ideal concept of intelligence called rationality

II. History of AI

Important research that laid the groundwork for AI:

- In 1931, Goedel layed the foundation of Theoretical Computer Science**1920-30s**:

He published the first universal formal language and showed that math itself is either flawed or allows for unprovable but true statements.

- In 1936, Turing reformulated Goedel's result and church's extension thereof.
- In 1956, John McCarthy coined the term "Artificial Intelligence" as the topic of the **Dartmouth Conference**, the first conference devoted to the subject.
- In 1957, The **General Problem Solver (GPS)** demonstrated by Newell, Shaw & Simon
- In 1958, John McCarthy (MIT) invented the Lisp language.
- In 1959, Arthur Samuel (IBM) wrote the first game-playing program, for checkers, to achieve sufficient skill to challenge a world champion.
- In 1963, Ivan Sutherland's MIT dissertation on Sketchpad introduced the idea of interactive graphics into computing.
- In 1966, Ross Quillian (PhD dissertation, Carnegie Inst. of Technology; now CMU) demonstrated semantic nets
- In 1967, Dendral program (Edward Feigenbaum, Joshua Lederberg, Bruce Buchanan, Georgia Sutherland at Stanford) demonstrated to interpret mass spectra on organic chemical compounds. First successful knowledge-based program for scientific reasoning.
- In 1967, Doug Engelbart invented the mouse at SRI
- In 1968, Marvin Minsky & Seymour Papert publish Perceptrons, demonstrating limits of simple neural nets.
- In 1972, Prolog developed by Alain Colmerauer.
- In Mid 80's, Neural Networks become widely used with the Backpropagation algorithm (first

described by Werbos in 1974).

- 1990, Major advances in all areas of AI, with significant demonstrations in machine learning, intelligent tutoring, case-based reasoning, multi-agent planning, scheduling, uncertain reasoning, data mining, natural language understanding and translation, vision, virtual reality, games, and other topics.
- In 1997, Deep Blue beats the World Chess Champion Kasparov
- In 2002,iRobot, founded by researchers at the MIT Artificial Intelligence Lab, introduced **Roomba**, a vacuum cleaning robot. By 2006, two million had been sold.

III. Water-jug problem & 8-puzzle problem

The General Problem Solver (GPS) was the first useful AI program, written by Simon, Shaw, and Newell in 1959. As the name implies, it was intended to solve nearly any problem**.**

Newell and Simon defined each problem as a space. At one end of the space is the starting point; on the other side is the goal. The problem-solving procedure itself is conceived as a set of operations to cross that space, to get from the starting point to the goal state, one step at a time.

The General Problem Solver, the program tests various actions (which Newell and Simon called operators) to see which will take it closer to the goal state. An operator is any activity that changes the
state of the system. The General Problem Solver always chooses the operation that appears to bring itcloser to its goal.

Example: Water Jug Problem

Consider the following problem:

A Water Jug Problem: You are given two jugs, a 4-gallon one and a 3-gallon one, a pump which has unlimited water which you can use to fill the jug, and the ground on which water may be poured. Neither jug has any measuring markings on it. How can you get exactly 2 gallons of water in the 4-gallon jug?

State Representation and Initial State :

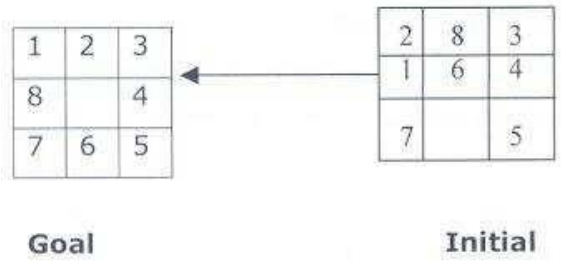
We will represent a state of the problem as a tuple (x, y) where x represents the amount of water in the 4-gallon jug and y represents the amount of water in the 3-gallon jug. Note $0 \leq x \leq 4$, and $0 \leq y \leq 3$. Our initial state: (0, 0)

Goal Predicate - state = (2, y) where $0 \leq y \leq 3$.

Operators -we must defi ne a set of operators that will take us from one state to another:

1. Fill 4-gal jug	(x,y) x < 4		→ (4,y)
2. Fill 3-gal jug	(x,y) y < 3		→ (x,3)
3. Empty 4-gal jug on ground	(x,y) x > 0		→ (0,y)
4. Empty 3-gal jug on ground	(x,y) y > 0		→ (x,0)
5. Pour water from 3-gal jug	(x,y)		→! (4, y - (4 - x))
to ll 4-gal jug	0 < x+y	4 and y > 0	
6. Pour water from 4-gal jug	(x,y)		→ (x - (3-y), 3)
to ll 3-gal-jug	0 < x+y	3 and x > 0	
7. Pour all of water from 3-gal jug	(x,y)		→ (x+y, 0)

8 Puzzle Problem.



The 8 puzzle consists of eight numbered, movable tiles set in a 3x3 frame. One cell of the frame is always empty thus making it possible to move an adjacent numbered tile into the empty cell. Such a puzzle is illustrated in following diagram.

The program is to change the initial configuration into the goal configuration. A solution to the problem is an appropriate sequence of moves, such as “move tiles 5 to the right, move tile 7 to the left, move tile 6 to the down, etc”.

Solution:

Once the problem states have been conceptually identified, we must construct a computer representation, or description of them. this description is then used as the database of a production system. For the 8-puzzle, a straight forward description is a 3X3 array of matrix of numbers. The initial global database is this description of the initial problem state. Virtually any kind of data structure can be used to describe states.

A move transforms one problem state into another state. The 8-puzzle is conveniently interpreted as having the following for moves. Move empty space (blank) to the left, move blank up, move blank to the right and move blank down,. These moves are modeled by production rules that operate on the state descriptions in the appropriate manner.

The rules each have preconditions that must be satisfied by a state description in order for them to be applicable to that state description. Thus the precondition for the rule associated with “move blank up” is derived from the requirement that the blank space must not already be in the top row.

- A* begins at a selected node. Applied to this node is the "cost" of entering this node (usually zero for the initial node). A* then estimates the distance to the goal node from the current node. This estimate and the cost added together are the heuristic which is assigned to the path leading to this node. The node is then added to a priority queue, often called "open".
- The algorithm then removes the next node from the priority queue (because of the way a priority queue works, the node removed will have the lowest heuristic). If the queue is empty, there is no path from the initial node to the goal node and the algorithm stops. If the node is the goal node, A* constructs and outputs the successful path and stops.

- The algorithm also maintains a 'closed' list of nodes whose adjoining nodes have been checked. If a newly generated node is already in this list with an equal or lower cost, no further processing is done on that node or with the path associated with it. If a node in the closed list matches the new one, but has been stored with a *higher* cost, it is removed from the closed list, and processing continues on the new node.

IV. Conclusion

If the node is not the goal node, new nodes are created for all admissible adjoining nodes; the exact way of doing this depends on the problem at hand. For each successive node, A* calculates the "cost" of entering the node and saves it with the node. This cost is calculated from the cumulative sum of costs stored with its ancestors, plus the cost of the operation which reached this new node. This paper provided an Optimized solution for Water-jug and 8-puzzle problem

1) F. Facchinei and C. Kanzow, Generalized Nash equilibrium problems, Ann. Oper. Res., vol. 175, no. 1, pp. 177–211, 2010.

2) Y. Nesterov, Stable traffic equilibria: Properties and applications, Optim. Eng., vol. 1, no. 1, pp. 29–50, 2000.

3) A. Kannan, U. V. Shanbhag, and H. M. Kim, Addressing supply-side risk in uncertain power markets: Stochastic Nash models, scalable algorithms and error analysis, Optim. Methods Software, vol. 28, no. 5, pp. 1095–1138, 2013.

4) J. S. Pang, G. Scutari, D. P. Palomar, and F. Facchinei, Design of cognitive radio systems under temperature-interference constraints: A variational inequality approach, IEEE Trans. Signal Process., vol. 58, no. 6, pp. 3251–3271, 2010.

5) T. Alpcan and T. Basar, A game-theoretic framework for congestion control in general topology networks, in Proc. 41st IEEE Conf. on Decision and Control, 2002, Las Vegas, NV, USA, 2002,
pp. 1218−1224.

6) D. Fudenberg and D. K. Levine, The Theory of Learning in Games. Cambridge, MA, USA: MIT Press, 1998.