Playing With Mazes

David B. Suits
Department of Philosophy
Rochester Institute of Technology
Rochester NY 14623

david.suits@rit.edu

6. Maze Games

So far I have been talking of mazes as mazes — a series of paths from points to points. The series of paths may be simple or complex, but in any case the object is to move from some given point to some other. Although we may delight in the cleverness of a maze (perhaps it is embedded in a complex drawing, for example), as a puzzle or game it is rather restricted and can quickly become boring. If we wish to make use of the power of a computer in constructing and refereeing maze games, we ought to look to something other than the mere display of a maze which we are to traverse from entry to exit.

6.1 Blind Mazes

The first enhancement to computer maze games is straightforward: do not represent to the player his position relative to the rest of the maze — or else make this information at least partly hidden.

Such a maze game might, for example, display the perimeter of the maze and the position of the player inside the maze relative to the perimeter, but not display the individual cells within the maze, nor, of course, the paths from cell to cell. Such a maze we can call a **blind maze**. The player specifies the direction of desired movement from the present cell: up, right, down, or left (or else north, east, south or west). The player is moved accordingly if the present cell is open in the chosen direction. Otherwise, the player bumps up against a wall and remains in the present cell. (As an option, a wall which is bumped up against might then be displayed.) Allowing the player a maximum number of attempts to move in order to reach the exit cell would add an additional challenge. By adding a computer-simulated opponent (or by allowing for a second human player as an opponent), the game can be made even more challenging: one player tries to reach the exit cell (or goal cell) within a certain time, or within a certain number of moves; the opponent tries to catch the player (or get to the goal cell first). Both players move around the maze under constraints of darkness.

Instead of providing a "bird's eye view" of the maze, the player might be shown only the "player's eye view". You will see only what is directly in front of you: either a wall, or else an opening to another cell. Only by turning around in your cell will you be able to learn where the cell's exits are located. In such a case, you might issue commands such as *rotate right*, *rotate left*, or *move forward*. Whether you are moving north, east, south or west will be known by the computer, which will keep track of your position and orientation, but can be known by you only if you keep track for yourself. The "player's eye view" might be enhanced by showing a perspective view of your position. Thus, if you are facing an exit from the cell, you will be able to see through into the next cell; if that cell has an opening on the facing wall, then you will be able to see through to the cell beyond that; and so on.

Adding an opponent (human or computer) who also roams the maze, will add challenge to such a game. In fact, such a multiplayer maze game would be a good candidate for a multi-computer game.

6.2 Wrap-Around Mazes

Certain very simple modifications to the traditional maze can be implemented. Consider, for example, the very simple maze in figure 23. One cell (cell 4) is established as the exit cell: moving north from that cell will constitute exiting the maze (and, presumably, winning the game). But in cell 9, movement is allowed only southwards. And in cell 10 there are walls to the east and west. Of course, there must be a wall on the east side of cell 10, because if there were not, then cell 10 would be the exit cell (or another exit cell). But wait! Why must moving east from cell 10 (assuming there were an opening there) really constitute an exit from the maze? Perhaps, instead, we can allow the maze to "wrap around" to the other side, such that moving east from cell 10 would put you into cell 6 (or 11). Allowing for such a possibility immediately transforms the very simple 5×5 maze into an effective 5×∞ maze. Allowing for wrap-around in the vertical direction as well will transform the maze into a ∞×∞ maze. If the player is given information only about the present cell (and, perhaps, about any cell which can be seen through an exit from the present cell), then even a simple maze will be very difficult to traverse, unless, by keeping careful records, the player begins to spot a repetition of patterns in the cells moved through.

Such an infinite wrap-around maze is extremely easy to implement on even a very small computer (I have put such maze games on programmable calculators), especially where the cells are internally represented as an ordered set of four walls, where, say, a "1" represents a wall and a "0" represents an opening. The computer need only keep track of which cell the player is in and the orientation of the player within the cell. Thus,

cell(*i*,1) = status of north side of cell *i*.
cell(*i*,2) = status of east side of cell *i*.
cell(*i*,3) = status of south side of cell *i*.
cell(*i*,4) = status of west side of cell *i*.

To move east from cell *i*, then, is allowed only if cell(*i*,2)=0, and a successful move then places the player in cell *i*+1.

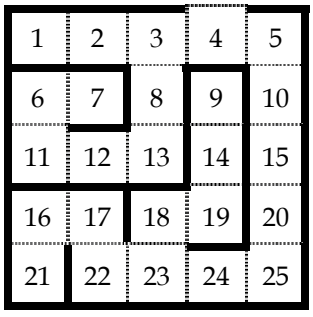

Figure 23. A simple 5×5 maze. Moving north from cell 4 will exit the maze, unless it is a wrap-around maze, in which case moving north from cell 4 would land you in cell 24.

6.3 One-Way Doors

Still another enhancement to a traditional maze is easy to implement, and this too can make even a small maze seem very complex, especially when a "player's eye view" is provided (instead of a "bird's eye view").

If each cell is represented by the status of its four walls (say, "1" for a wall, and "0" for an opening), then one expects that the neighboring cell will also have a corresponding "1" or "0". But suppose it doesn't. Suppose, for example, you are in a cell whose status indicates that the north side is an open door ("0"). Given the "player's eye view", and given that you are facing north, that means that you can look straight ahead into the next cell. Suppose you move ahead into that cell. But suppose that the new cell's status, instead of indicating a "0" for its south side (which, after all, would make sense, since you just came in that way), is actually "1". This means that if you were to turn around and face south, expecting to see through to the cell you had just come from, you would instead see a wall (or a closed door, or whatever it is that the game represents as a barrier). You have just come though a one-way door, and this makes traditional maze search algorithms far less useful, since, after moving through a one-way door into a cell never visited before, there will be no information about how to get back to the cell you came from without initiating a brand new search.

Even a very small maze which wraps around both horizontally and vertically, and which has a few one-way doors, can be an extremely difficult maze to solve.

6.4 Interpreted Mazes

A puzzle or game can be modeled on a maze without the game or puzzle seeming to be a maze. That is to say, the game will have a maze as its underlying structure, but the maze will not be apparent to a player of the game; the player will not think in terms of a maze at all.

Such a game can be constructed by *interpreting* (or translating) the constituents of a maze game into some other idiom. That is, being in a cell will be represented as being in some game situation with a number of options available (one for each open door). To accept a particular option is to move through the door into another cell — into another game situation with a (possibly new) set of options. (Or else each cell is a game, and entry to that cell from a neighbor is allowed only by successfully completing the game.)

For example, instead of geometrical motion through a maze, a game might be represented as a series of choices for trading goods. The player is given some object to begin with and then, for each open door in the cell, he is offered an object in trade. Accepting a trade which leads farther from the goal will result in some sort of net loss (say, in the dollar value of the item presently held), whereas accepting a trade which brings the player closer to the goal will result in a net gain. Backtracking might mean having to trade back for items previously held (presumably for a net loss).

Another example: A maze engine might underlie a bureaucracy game, where the goal is to contact Mr. X. At any given time (i.e., within any given cell) the player is allowed to contact only certain persons in the bureaucracy (various secretaries, undersecretaries, assistants, deputy managers, and so on). Some contacts lead immediately to dead-ends, i.e., closed doors in the cell. (Perhaps there are ten unhelpful persons in the bureau, and perhaps each of the closed doors in each cell is presented to the player as one of those ten chosen at random, so that very often the player is given a choice to contact, say, Assistant Deputy Under Secretary Smith, who turns out always to be most unhelpful. Eventually, the player will simply avoid bothering with Smith — which is to say that, in terms of the underlying maze engine, the player will have learned to recognize one kind of closed door.)

A maze engine might underlie instructional games. In order to gain entrance to a cell, a player has to successfully solve a certain kind of math problem. Easier problems lead farther from the goal, and harder problems lead closer to the goal.

Or each cell might represent a word spelling problem (or a word definition problem, or a language translation problem), where, as in the math game above, successfully completing easy problems takes the player farther from the goal, and successfully completing harder problems brings the player closer to the goal.

In general, each cell of the underlying maze engine might represent some activity or other (perhaps an entire game in itself, such as a Battle-the-Aliens game or a chess game, or…), and the game must be successfully completed in order to advance (i.e., move into the next cell), at which time another series of options is presented. And so on until the goal cell is reached. Probably multiply-connected mazes (i.e., mazes with more than one path from the start cell to the goal cell) would be appropriate for such maze engines.

Bibliography

Allen, S. and Allen, S. A., "Simple Maze Traversal Algorithms", *Byte*, Vol. 4, No. 6 (June, 1979), p. 36.

Even, Shimon, *Graph Algorithms* (Rockville, MD: Computer Science Press, 1979).

Jordon, M. I., "Serial Order: A Parallel Distributed Processing Approach", Technical Report 8604. La Jolla: University of California, San Diego, Institute for Cognitive Science, 1986.

Todd, Peter, "A Sequential Network Design for Musical Applications", in D. Touretsky, G. Hinton and T. Sejnowski, eds., *Proceedings of the 1988 Connectionist Models Summer School* (San Mateo, CA: Morgan Kaufmann Publishers, Inc., 1989), pp. 76–84.