Rook Jumping Maze Design Considerations

Todd W. Neller[1], Adrian Fisher[2], Munyaradzi T. Choga[1], Samir M. Lalvani[1], and Kyle D. McCarty[1]

[1] Gettysburg College, Dept. of Computer Science, Gettysburg, Pennsylvania, 17325, USA, `tneller@gettysburg.edu`,
WWW home page: `http://cs.gettysburg.edu/~tneller`
[2] Adrian Fisher Design Ltd., Portman Lodge, Durweston, Dorset, DT11 0QA England, `adrian@adrianfisherdesign.com`,
WWW home page: `http://www.adrianfisherdesign.com`

Abstract. We define the Rook Jumping Maze, provide historical perspective, and describe a generation method for such mazes. When applying stochastic local search algorithms to maze design, most creative effort concerns the definition of an objective function that rates maze quality. We define and discuss several maze features to consider in such a function definition. Finally, we share our preferred design choices, make design process observations, and note the applicability of these techniques to variations of the Rook Jumping Maze.

1 Introduction

In this Section, we will define the Rook Jumping Maze, provide a suitable notation for discussing maze features, and describe the history of the maze. In Section 2, we will outline a general optimization process for generating such mazes, whereas Section 3 will define a number of specific features suitable for defining an objective function for such optimization. Section 4 shares observations concerning the outcomes of algorithmic choices in Section 3, experiences in the business of maze design, and the "Aha!" moments enjoyed when solving novel quick mazes (a.k.a. logic mazes). Existing and potential variations of Rook Jumping Mazes are described in Section 5, followed by conclusions in Section 6.

1.1 Definitions

Figure 1 provides an example of a Rook Jumping Maze.[3] Let $r_{\max}$ and $c_{\max}$ be the number of rows and columns, respectively. In this case, $r_{\max} = c_{\max} = 5$. A state s of the maze (i.e., current location) is denoted by the row-column coordinate (r, c), where $r \in \{1, \ldots, r_{\max}\}$ and $c \in \{1, \ldots, c_{\max}\}$. For example, a maze puzzler located at $(1, 1)$ is located in the upper-left corner cell of the grid. The set of all states is denoted S. Let functions $\mathrm{row} : S \to \mathbb{N}$ and $\mathrm{col} : S \to \mathbb{N}$ map a state to its row and column, respectively.

[3] Minimum 13-move solution for Figure 1: down, right, left, up, down, left, right, up, left, left, right, down, up

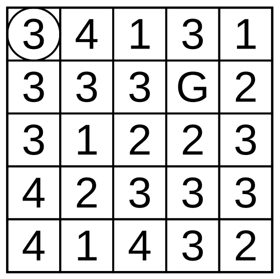


Fig. 1. Example Rook Jumping Maze. Starting at the circled cell, each *jump number* indicates the exact number of cells one may move in a straight line horizontally or vertically. The object is to find a path to the goal marked “G”.

The circled starting state of this example maze, denoted s_{start}, is $(1, 1)$. The goal state of this example maze, denoted s_{goal} and marked with a “G”, is $(2, 4)$.

Each state of the maze has an associated *jump number* that provides the exact number of cells one may move horizontally or vertically in a straight line to change states. In Figure 1, the first move from $(1, 1)$ may either be 3 cells right to $(1, 4)$, or 3 cells down to $(4, 1)$. From $(4, 1)$, there is only one legal *forced move* 4 cells right to $(4, 5)$. From $(4, 5)$, one may move 3 cells left to $(4, 2)$ or 3 cells up to $(1, 5)$. A jump must be in a single orthogonal direction, and may neither stop short of the number of required cells at edges, nor may it wrap around edges toroidally. Variations are discussed in Section 5.

Let *jump function* $j : S \to \mathbb{N}$ map a state to its jump number. Define $j(s_{\text{goal}}) = 0$. Let the *successor function* $\sigma : S \to 2^S$ map a state to its possible successor states, that is:

$$\sigma(s) = \left\{ s' \in S \;\middle|\; \begin{array}{l} s' = (\text{row}(s) + j(s), \text{col}(s)), \text{or} \\ s' = (\text{row}(s) - j(s), \text{col}(s)), \text{or} \\ s' = (\text{row}(s), \text{col}(s) + j(s)), \text{or} \\ s' = (\text{row}(s), \text{col}(s) - j(s)) \end{array} \right\}$$

Let the *predecessor function* $\pi : S \to 2^S$ map a state to its possible predecessor states, that is:

$$\pi(s) = \{s' \in S | s \in \sigma(s')\}$$

Define a *path* of length n from s_{from} to s_{to} as a sequence of states $(s_1, s_2, \ldots, s_n)$ such that $s_1 = s_{\text{from}}$, $s_n = s_{\text{to}}$, and for all $1 \leq i < n$, $s_{i+1} \in \sigma(s_i)$. The *optimal* or *shortest solution path* is a path of minimal length from s_{start} to s_{goal}. Let $|p|$ denote the length of path p. Let $P_{s_{\text{from}}, s_{\text{to}}}$ be the set of all paths from s_{from} to s_{to}. Then an optimal solution path p^* is $\arg\min_{p \in P_{s_{\text{start}}, s_{\text{goal}}}} |p|$.

1.2 History

The origin of Rook Jumping Mazes is unknown, but some attribute its creation to the great puzzle innovator Sam Loyd. Loyd’s 1898 Queen Jumping Maze, which additionally allows diagonal moves, is shown in Figure 2.[4] It appears on page 106 of the *Cyclopedia of Puzzles* [1], a collection of Loyd's work compiled by his son.[5]

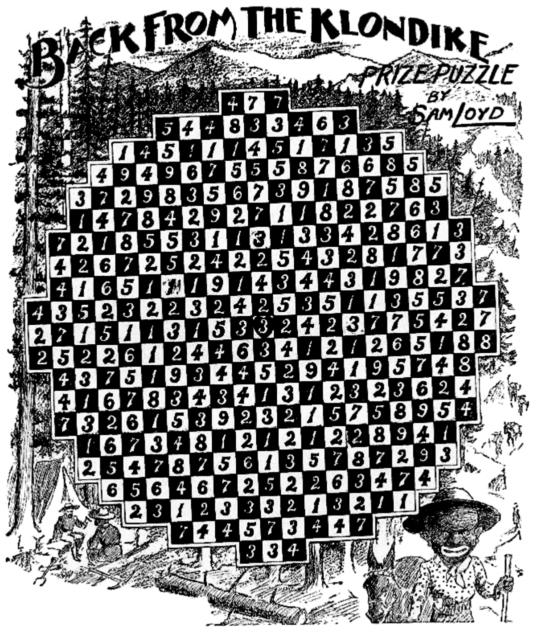


Fig. 2. Loyd's puzzle "Back from the Klondike"

The puzzler is directed to a heart-marked start location at the center of a gridded circle. The object is to find a path to a cell from which one can jump one cell beyond the circle's edge. Loyd writes that the puzzle "... was built purposely to defeat Euler's [working backwards] rule and out of many attempts is probably the only one which thwarts his method."

Leading modern maze designer Adrian Fisher started creating floor mazes with colored plastic tiles in 1984. From the outset, the colored tiles allowed him to create mazes based on directed graphs, initially using sequences of colored maze paths, and later using numbers and arrow to constrain movement.

Using colored plastic tiles, he created his first human-size Rook Jumping Maze in 1991 at Paradise Park, Hawaii. The logic was also one-way, this time with each cell being the junction, and each "path" being an imaginary hop through the air of a given distance (as stated in the cell), with the player choosing which direction to jump. Since then, Fisher has supplied hundreds of plastic tile floor mazes worldwide to science

[4] See also: `http://en.wikipedia.org/wiki/Back_from_the_Klondike`
[5] Public domain scans available from `http://www.mathpuzzle.com/loyd/`

centers, children's museums, schools and farm attractions (often in conjunction with cornfield maize mazes). One of Fisher's Rook Jumping Mazes can be seen in [2].

Robert Abbott is another prominent modern maze designer who has created Rook Jumping Mazes such as the 7×7 "Jumping Jim" [3, pp. 14–15], and the 8×8 "Number Maze" [4, pp. 36–37] Rook Jumping Maze variant with a "no-U-turn rule" disallowing reverse moves, e.g., a left jump immediately following a right jump.

2 Generation of Rook Jumping Mazes

In our experience, stochastic local search [5] algorithms have provided a satisfactory means of generating Rook Jumping Mazes such as that of Figure 1. Even simple algorithms yield good results, freeing us as designers to focus on the relative (un)desirability of various maze features.

In applying stochastic local search (SLS) to maze design, we must, for the moment, step back from maze states (i.e., locations) and individual maze solutions, and consider instead the configuration of the maze itself, i.e., jump numbers and start/goal locations, as a single *configuration state* c in the space of all maze configuration states C. We search through such configuration states in order to find good designs according to a maze-rating measure we subjectively define. Henceforth, we will refer to configuration states as *configurations* to avoid confusion with maze location states.

In general, the goal of SLS is to seek a *configuration* from a set of configurations C that optimizes some measure. We call this measure the configuration's *energy*, denoted $e : C \to \mathbb{R}$, and we seek a configuration with *low* or *minimal* energy. Our task is then to seek a configuration c minimizing $e(c)$, that is, $\arg\min_{c \in C} e(c)$. In practice, it is often the case that we are only able to or only want to find an approximately optimal configuration. For each configuration, there is a neighborhood $N(c)$ which defines those configurations we may look to next after c in our search. We choose a successor configuration from $N(c)$ (which is in some sense "local" to c) stochastically.

One of the simpler algorithms, Hill Descent with Random Uphill Steps, is parameterized by the *number of iterations* and an *uphill step acceptance probability*.[6] We begin with a random configuration c. For each of the given number of iterations, one generates a random neighboring configuration $c' \in N(c)$ as a possible next step in the search space. If this configuration c' has the same or lower energy, we accept the change of c to c'. If the configuration has a higher energy, we accept the change with some small given probability. Otherwise, we disregard the proposed change. At the end of all iterations, the result of the search is the minimum energy configuration encountered.

For our maze generation, the initial random configuration has its start and goal state locations set, and random jump numbers assigned such that at least one legal move is possible from each non-goal location. Thus, the initial SLS configuration is generally a poor-quality maze that possibly has no solution. We generate a random neighboring "local" configuration by choosing a random non-goal location in the maze, and changing the jump number such that a legal move from that location is still possible. While this small change may significantly change the quality of the maze, the overall maze

[6] For rapid generation, we perform 25000 iterations with an acceptance probability of $.005$.

structure is still largely intact, and thus this provides a natural "locality" in the search of all possible maze configurations.

These and other techniques, e.g., simulated annealing, are described in greater detail in [5, 6].[7] The greatest creative work consists of defining the energy function, i.e., the measure to be optimized. In this case, think of *energy* as being a rating of the *badness* of a Rook Jumping Maze configuration. Our design team, consisting of both faculty and students from various disciplines, generated and tested many mazes, sharing and discussing many features we observed as significant to the quality of Rook Jumping Maze design. We next turn our attention to the primary features observed as being relevant to the definition of the energy function.

3 Maze Features

There are a number of features that should be considered when defining the energy function for stochastic local search. In this section, we will define several features and discuss each in turn.

3.1 Goal Reachability, Reaching States, Reachable States, Black Holes, and White Holes

The first and most important feature of a maze is that it can be solved, that is $P_{s_{\mathrm{start}},s_{\mathrm{goal}}} \neq \emptyset$. However, there are further definitions and features to consider, including the number of reaching/reachable states and the existence of black/white holes.[8]

A *reaching state* s is a state from which one can reach the goal, i.e., $P_{s,s_{\mathrm{goal}}} \neq \emptyset$. A *reachable state* s is a state which one can reach from the start, i.e., $P_{s_{\mathrm{start}},s} \neq \emptyset$. A *black hole* $B(s) \subset S$, a "dead-end" of a maze, is a subset of reachable, non-reaching interconnected states defined recursively as follows. Let $s \in B(s)$ be a reachable, non-reaching state. For each $s_1 \in B(s)$, all successors and all reachable, non-reaching predecessors are in $B(s)$ as well. If $s_2 \in \sigma(s_1)$ or ($s_2 \in \pi(s_1)$ and $P_{s_{\mathrm{start}},s_2} \neq \emptyset$ and $P_{s_2,s_{\mathrm{goal}}} = \emptyset$), then $s_2 \in B(s)$. Similarly, a *white hole* $W(s) \subset S$, a back-tracing dead-end of a maze, is a subset of unreachable, reaching interconnected states defined recursively as follows. Let $s \in W(s)$ be an unreachable, reaching state. For each $s_1 \in W(s)$, all predecessors and all unreachable, reaching successors are in $W(s)$ as well. If $s_2 \in \pi(s_1)$ or ($s_2 \in \sigma(s_1)$ and $P_{s_{\mathrm{start}},s_2} = \emptyset$ and $P_{s_2,s_{\mathrm{goal}}} \neq \emptyset$), then $s_2 \in W(s)$.

We first observe that, unlike conventional mazes, Rook Jumping Mazes introduce a forward directional bias. It is easier to move forward in the maze than to trace backwards. As with conventional mazes, puzzlers will tend to trace a solution backwards, but such backwards tracing is a greater challenge to visual perception. This has interesting ramifications for the maze experience.

First, since some Rook Jumping Mazes are in a walkable form, laid out with large tiles on the ground, we need to give attention to the forward experience. Since backtracing is more challenging, a black hole tends to encourage the puzzler to restart from the

[7] Also: `http://cs.gettysburg.edu/~tneller/resources/sls/index.html`

[8] The descriptive maze terms "black hole" and "white holes" were coined by Fisher in [7].

beginning. In this brief moment of disengagement, especially if frequently repeated, the puzzler may become demotivated. Rather than walking back to the start, one may simply walk away. We therefore favor elimination of black holes, allowing the possibility of imminent breakthrough to a solution to keep the puzzler engaged. No step is to be feared as a trap; there are only puzzling diversions from the goal.

Even some of those we have observed walking large scale mazes will sometimes stop and visually work backwards from the goal in planning future steps. To provide both an incentive for the forward experience, and a challenge for the backward experience, we do allow white holes. Further, we note that, compared to black holes, white holes are easy to escape by retracing steps, for retracing backward steps is merely proceeding forward.

3.2 Start/Goal State Locations

Our design team had a preference for variety in start/goal locations. However, some of our experiences with mazes generated with such variety yielded the following observations. (1) Traditional mazes, probably influenced by the left-to-right, top-to-bottom (LRTB) scripting system, often have the start and goal in the upper-left and lower-right, respectively. Even considering variations, one rarely starts a maze from within. (2) Non-corner start locations can sometimes yield a forced first move, whereas there is always a choice starting from the corner. (3) Variation of the goal location yielded pleasing diversity of generated mazes. We thus opted to fix the start location in the upper-left corner, and vary the goal location at random.

3.3 Shortest Solution Uniqueness

One should also consider the importance of a unique shortest solution path, i.e., if $p_1, p_2 \in \arg\min_{p \in P_{s_{\text{start}}, s_{\text{goal}}}} |p|$, then $p_1 = p_2$. In mazes featuring many black holes, simply finding a path to the goal may be a sufficiently satisfying accomplishment.

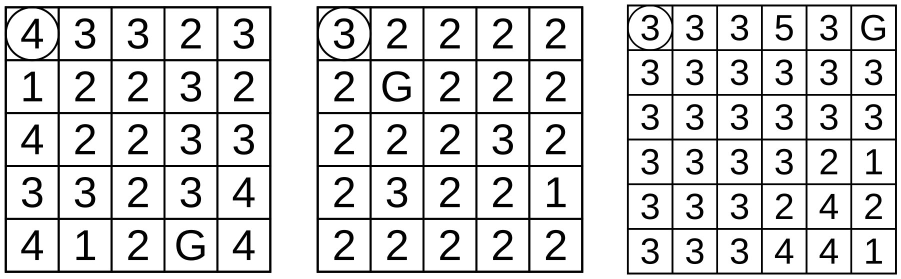
However, in mazes without black holes, where all states are reaching states, there are generally an infinite number of solution paths with repeated states, and many distinct paths without repeated states. In such cases, the experience of finding a path to the goal seems not to provide the same level of satisfaction as knowing that one has found the *best, shortest solution*. In such mazes, random experimentation will eventually yield success, and random experimentation coupled with good memory can yield speedy success. However, this can feel like a “given” or “inevitable” result.

The existence of a unique optimal solution can drive a deeper level of analytical engagement. To find a path and then be informed that a shorter path exists can motivate a more intense exploration of alternative paths, leading to a familiarity with the complete topology for smaller mazes, e.g., 5×5. The satisfaction of having found the unique optimal path is akin to that of forming an elegant mathematical proof. There is a beauty to simplicity.

Since our design team prefers no black holes, the uniqueness of a minimal length solution has great value.

3.4 Minimum Solution Path Length

At first, one might think that a long minimum solution path would be of prime importance in maze design. We might associate solution length with maze difficulty. However, this is not necessarily the case. Mazes with longer minimum solution paths may involve many forced moves, or stereotyped patterns such as sequences of U-turn moves within a row or column.



4	3	3	2	3
1	2	2	3	2
4	2	2	3	3
3	3	2	3	4
4	1	2	G	4

(a) Maze with length 19 solution

3	2	2	2	2
2	G	2	2	2
2	2	2	3	2
2	3	2	2	1
2	2	2	2	2

(b) Maze with 2-jump clusters

3	3	3	5	3	G
3	3	3	3	3	3
3	3	3	3	3	3
3	3	3	3	2	1
3	3	3	2	4	2
3	3	3	4	4	1

(c) Maze with 3-jump clusters

Fig. 3. Illustrative design examples

Consider the illustrative example of Figure 3(a). In seeking to maximize the minimum solution path length, we generate puzzles which, at first, are interesting. However, many of these are easily backtraceable and have stereotyped sequences of successive row/column as in column 3 and then row 2 in the end of the minimum length solution of 19 moves.[9]

An analogy to standard walled mazes may help. A depth-first generation algorithm will generally yield a long solution path, but with relatively few and obvious decisions along the winding solution. In the same way, minimum solution length may not matter so much as the frequency and complexity of decisions along the solution path.

3.5 Forward/Backward Decisions and Initial Forced Moves

Let us assume a unique shortest solution path p^*.[10] We define the number of forward decisions as the number of non-goal states along p^* with more than one successor, i.e., $|S'|$ where $S' \subset S$ and $\{s \in S' | s \in p^* \text{ and } |\sigma(s)| > 1\}$. Similarly, we define the number of backward decisions as the number of non-start states along p^* with more than one predecessor, i.e., $|S'|$ where $S' \subset S$ and $\{s \in S' | s \in p^* \text{ and } |\pi(s)| > 1\}$.

Forced moves serve mainly to lengthen a solution and cause previously visited states to pass out of memory. However, a solution with many forced moves is relatively uninteresting. Forced initial moves are particularly uninteresting, because the puzzle might

[9] Solution to Figure 3(a): right, down, left, right, up, down, left, right, left, down, right, up, up, down, up, left, right, right, down

[10] When there exist multiple shortest solution paths, we may average this measure for such paths.

as well have a start position at the first state providing a decision. Similarly, states with only one possible predecessor are relatively uninteresting, as they make backtracing too easy.

3.6 Same Jump Clusters

A further interesting feature of Rook Jumping Mazes is what we will call a *same jump cluster*. Same jump clusters are maximal strongly connected subgraphs of the maze with all states sharing a common jump number. The same jump cluster $J(s) \subset S$ is defined recursively as follows: $s \in J(s)$. If $s_1 \in J(s)$ and $s_2 \in \sigma(s_1)$ and $j(s_1) = j(s_2)$, then $s_2 \in J(s)$.

For those with experience of Rook Jumping Mazes, same jump clusters stand out as partial sub-lattices of jump number j, with cells in the sublattice being j cells apart. Whereas 1-jump clusters are too easily traversed, 2- and 3-jump clusters yield mazes of greater interest, as shown in Figures 3(b) and 3(c). However, most of our design team preferred a diversity of jump numbers and fewer same jump clusters in general.

3.7 Example Energy Function

Not every feature has the same priority. Suppose one has integer-valued feature measures f_1 and f_2. We can combine the two, prioritizing f_1 over f_2 in the energy function, with the expression $(r_2 + 1)f_1 + f_2$, where r_2 is the range of possible values for f_2. In this way, the smallest change to f_1 matters more than the greatest possible change to f_2.

We compute our energy function e for a given maze configuration c as follows. Start with an energy, i.e., maze badness score, of 0. If there is no unique optimal solution, penalize the design by adding $|S|^3$ to the score. Next, we add $n|S|^2$ to the score, where n is the number of non-reaching states. Assuming there is a unique optimal solution, let d_f and d_b be the number of forward and backward decisions, respectively, along the optimal path. Reward more decisions by *subtracting* $\min(d_\mathrm{f}, d_\mathrm{b})$ from the score. Partition all states into a set of jump clusters $\mathcal{J}$. For each jump cluster $J \in \mathcal{J}$, add $(|J| - 1)^2$ to the score. Finally, add m^2 to the score, where m is the number of initial forced moves.[11] Thus, our energy function is

$$e(c) = \begin{cases} n\,|S|^2 - \min(d_\mathrm{f}, d_\mathrm{b}) + \sum_{J \in \mathcal{J}} (|J| - 1)^2 + m^2 & \text{if there exists a unique optimal solution, or} \\ |S|^3 + n\,|S|^2 + \sum_{J \in \mathcal{J}} (|J| - 1)^2 + m^2 & \text{otherwise} \end{cases}$$

4 Observations

4.1 Algorithm Observations

Using the sample energy function of the previous section with hill descent for $20,000$ iterations with an uphill step probability of $.005$, we are pleased with the quality of

[11] This step may be omitted if the start state is located in a corner where forced forward moves are not possible.

the results. This is by no means the only way to generate high-quality Rook Jumping Mazes, yet this approach may be replicated and improved upon by others with relative ease.

The generated maze topologies tend to follow an interesting general pattern. Roughly half of maze states lead to one another in a tangled mass, from which the other half of states form a single branch leading to the goal, yet allowing many opportunities to misstep back into the tangled mass. This topological tendency has been observed in other similar maze generations, e.g., Oskar van Deventer's four-bit mazes.[12] We expect that this is a common topological feature of interesting directed graph maze designs.

4.2 Diversity of Design

Leading maze puzzle designers[13] tend to be fascinated by creating new puzzle formats, each with its own new rule or rules. Thus many of their designs are created manually, which reveals their own particular design approach and style. Quite apart from its graphical appearance, it is possible to recognise a particular designer's style in a new form of puzzle, from the way they have approached the design challenge.

Occasionally, publishing requirements to provide the same type of puzzle on a daily or monthly basis motivates writing a computer program that makes the process automatic or semi-automatic. Such programs then become an expression of their designer's style. In the approach described above, a design style may be diversified by making use of multiple, independently-developed maze-rating functions.

Some puzzle researchers write computer programs to explore the potential of a particular puzzle notion, so as to generate multiple solutions, and then use tests to rank the complexity of each solution. For example, sliding block puzzles can be easily devised with a few pieces, and with a relatively simple yet distinctive playing area; nevertheless one or two starting positions can have a much longer minimum solution path length, sometimes involving more than 150 moves.[14] If you consider this to be the only valid factor, then this would be a way to rank the various solutions found by computer. In practice, market research might find that another starting position provided a more entertaining or satisfying puzzle. Or, if there are multiple ways of solving the same puzzle (e.g., the Rubik's Cube), all quite difficult, players might find the puzzle had greater *repeat appeal*.

[12] Stochastic local search algorithm:
`http://cs.gettysburg.edu/~tneller/mazes/oskar4bit/index.html`

[13] e.g., Robert Abbott (`http://www.logicmazes.com/`),
Adrian Fisher (`http://www.adrianfisherdesign.com/`),
Andrea Gilbert (`http://www.clickmazes.com/`),
Scott Kim (`http://www.scottkim.com/`),
Ed Pegg Jr. (`http://www.mathpuzzle.com/`),
Steve Ryan (`http://steveryangames.com/`),
James Stephens (`http://www.puzzlebeast.com/`),
and Oskar van Deventer (`http://oskarvandeventer.nl/`), to name a few

[14] See, for example, "A Dozen Irritating Sliding Block Puzzles":
`http://www.puzzlebeast.com/slidingblock/sliding_irritating.html`

Fisher considers that, when it comes to the general public including families, whose reading age is about 10 years, and puzzle concentration and aptitude much the same, typically 12 to 15 moves is quite sufficiently entertaining to be fulfilling.

4.3 The "Aha!" Moment and the Fulfillment of Rule Mastery

Quick mazes (a.k.a. logic mazes) like the Rook Jumping Maze bring many principles of good maze design into sharp focus. Fisher pioneered the name and concept of "Six Minute Mazes" [2, pp. 223-257], whereby each puzzle takes about 6 minutes to solve, so that a player can have the fulfillment of solving ten different puzzles each hour. The player typically spends two minutes assimilating and practicing a new and unfamiliar rule, two minutes exploring the network in earnest (sometimes "methodically" at first, and increasingly playfully and experimentally), reaches an "Aha!" moment after noticing a repeated pattern emerging, forms a hypothesis, tries it out, and in the final two minutes solves the puzzle, feeling that he/she has actively contributed to his/her achievement (rather than just reaching the goal by turning the final corner by accident). It is the feeling of mastering a new rule (such as "jump like a Rook in Chess") which gives so much personal fulfillment; much more so than sticking to the same rule with ever more complex manifestations of that same rule.

For Rook Jumping Mazes, one that has mastered the basic rook jumping rule begins to notice puzzle structures that lead to heuristics that aid the puzzler in solving. For example, in Figure 1, consider cells $(1,1)$, $(1,4)$, $(4,4)$, and $(4,1)$. The first three of these cells form a same jump cluster. The only escape from these cells is through the fourth. Thus, the puzzler should directly jump from $(1,1)$ to $(4,1)$. Initially, puzzlers tend not to form immediately the abstract concept of a same jump cluster (or parity generalizations involving integer multiples of jump numbers). However, the independent formation of such an abstraction, and observation of how this enables the puzzler to avoid suboptimal paths, provides a satisfying sense of learned competence.

5 Maze Variations

One may create variations of the simple Rook Jumping Maze in many ways.

One may vary *tiling* of the maze, using a different regular tilings, e.g., triangular or hexagonal. Semiregular and other tilings present different interesting possibilities at the risk of yielding movement instructions that are difficult for many to grasp.

Additional *topological* constraints may be added or removed, such as allowing toroidal wrap-around grid boundaries, or creating additional graph connectivity as in the abstract strategy board game Surakarta.[15] Simple means of adding constraints include the addition of impassable walls between tiles, impassable tiles, or tiles which may be passed over but cannot be a move destination.

Movement constraints may be varied as well. With the addition of diagonal moves the Rook Jumping Maze becomes a Queen Jumping Maze. Abbott's "no-U-turn" rule increases state complexity so that the current state must be described as the product of the row, the column, and the previous move direction.

[15] `http://en.wikipedia.org/wiki/Surakarta_(game)`

Many rich possibilities for creative variants exist, yet most of the design considerations we outlined remain relevant through such variation.

6 Conclusion

We have described the Rook Jumping Maze, its history, and noted algorithms suitable for generation of such mazes. The core creative work lies in observing features of Rook Jumping Mazes, and expressing one's subjective judgments about maze quality in an objective function. To aid others in this endeavor, we have discussed several features which are important design considerations, and provided an example of how these may be used in concert to yield high-quality results. The interested reader may enjoy viewing these results.[16]

References

1. Loyd, S.: Sam Loyd's Cyclopedia of 5000 Puzzles, Tricks, and Conundrums with Answers. (1914)
2. Fisher, A.: The Amazing Book of Mazes. Harry N. Abrams, Inc., New York, New York, USA (2006)
3. Abbott, R.: Mad Mazes: Intriguing Mind Twisters for Puzzle Buffs, Game Nuts and Other Smart People. Adams Media Corporation (1990)
4. Abbott, R.: Supermazes: Mind Twisters for Puzzle Buffs, Game Nuts, and Other Smart People. Prima Publishing, Rocklin, CA (1997)
5. Hoos, H.H., Stützle, T.: Stochastic Local Search: foundations and applications. Morgan Kaufmann, San Francisco (2005)
6. Neller, T.W.: Teaching stochastic local search. In: Proceedings of the 18th International FLAIRS Conference (FLAIRS-2005), Clearwater Beach, Florida, May 15-17, 2005, Menlo Park, CA, USA, AAAI Press (2005)
7. Fisher, A., Gerster, G.: The Art of the Maze. Weidenfeld & Nicolson, London, England (1990)

[16] Rook Jumping Maze of the Day: `http://cs.gettysburg.edu:8080/jumpmaze/`