# Metrics for Better Puzzles

Cameron Browne
Computational Creativity Group
Imperial College London
camb@doc.ic.ac.uk

**Abstract.** While most chapters of this book deal with run-time telemetry observed from user data, we turn now to a different type of metric, namely design-time metrics for level generation. We describe Hour Maze, a new type of pure deduction puzzle, and outline methods for the automated generation and solution of levels. Solution involves a deductive search that estimates a level's iterative and strategic depth. This information, together with symmetry analysis of wall and hint distributions, provides useful metrics with which levels may be classified and described. Results from a user survey indicate that players' enjoyment of computer-designed levels may be affected by their perception of whether those levels are indeed computer-designed or handcrafted by humans. We suggest ways in which puzzle metrics may be used to increase the perception of intelligence and personality behind level designs, to make them more interesting for players.

**Keywords.** Logic puzzle; Hour Maze; Nikoli; deductive search; quality metrics; procedural content generation.

## 1 Introduction

The current crop of smart phones and handheld game devices are the ideal platform for logic puzzles, which are enjoying a surge in popularity due to the mainstream success of titles such as Sudoku and Kakuro. Such puzzles can be played easily on small screens without losing any of their appeal, and can provide a deep, engaging playing experience while being conveniently short and self-contained.

Fig. 1 shows a new logic puzzle called Hour Maze currently under development for iOS devices. We have developed tools for automatically generating and solving Hour Maze levels and measuring key metrics; however, the ability to quickly generate large numbers of levels raises certain questions:

- *How do we know which of these levels will interest players?*
- *How do we make levels more attractive and engaging?*
- *How do we fine-tune the search to focus on fewer, better designs?*


**Fig. 1.** Full size 12x12 Hour Maze in the iPad prototype.

There has long been debate over the merits of computer-generated puzzles compared to those handcrafted by human experts. Japanese publisher Nikoli is understandably derisive of automatically generated content and its potential to

flood the market with inferior mass product. For example, Nikoli's chief editor Nobuhiko Kanamoto (2001) observes that:

> *Computer-generated Sudoku puzzles are lacking a vital ingredient that makes puzzles enjoyable - the sense of communication between solver and author*.

The aim of this chapter is to investigate how metrics may be used to classify puzzles and help address perceived biases against computer-generated content, using Hour Maze as a test case. We stress that the term *metrics* in this chapter refers to design-time metrics for puzzle design, rather than the more usual (tele)metrics obtained from players at run-time, as described in the other chapters of this book.

## 1.1 Approach

After outlining some basic principles of puzzle design and introducing Hour Maze more fully, we describe strategies for solving Hour Maze levels and a deductive search method that can quickly finds a puzzle's unique solution based on these strategies.

Metrics for measuring key aspects of a given level are described, namely symmetry and depth, and a method for automatically generating levels using these metrics is outlined. The final sections describe a puzzle design experiment which explores player perceptions of computer- and human-made levels, and how puzzle metrics might be harnessed to produce more engaging puzzles.

# 2 Deduction Puzzles

Puzzles are distinct from games in that they typically involve a single player; the solver. Thompson (2000) makes the following observation about the intimate relationship between games and puzzles:

> *Every board position presents the player with the puzzle, 'What is the best move?', which in theory could be solved by logic alone. A good abstract game can therefore be thought of as a "family" of potentially interesting logic puzzles, and the play consists of each player posing such a puzzle to the other.* (p1)

Salen and Zimmerman classify puzzles as a subset of games according to their excellent (2004) definition:

*A game is a system in which players engage in an artificial conflict, defined by rules, that results in a quantifiable outcome.* (p81)

However, solitaire puzzles can also be viewed as two-player games that provide a contest between the *designer* who creates the puzzle and the *player* who tries to solve it. The designer constitutes a "null" player who may not be physically present for the contest, but whose wit and personality can be evident in the challenge that they set to the solver (if the level is well designed). It is this feeling of intelligence behind each level design that we are trying to capture.

## 2.1 Japanese Logic Puzzles

Hour Maze belongs to a class of puzzles known as *Japanese logic puzzles*, due to their popularity in that country. Nikoli, the foremost publisher of such puzzles, provides the definitive catalogue including Sudoku, Kakuro, Slitherlink, and so on. These puzzles are characterised by the following qualities:

- Single player.
- Simple rules.
- Unique solution.
- Can be solved by deduction.
- Context free, i.e. universal symbols such as numbers, not letters or words.

Logic puzzles may be described as *pure deduction puzzles* as each unique solution may be determined by logic and deduction alone without the need for guesswork. Some lookahead may be required to solve particularly difficult situations, but such lookahead should take the form of existence proofs that prove or eliminate choices rather than simply finding possible candidates and "trying them out".

Nikoli designer Hiroshi Higashida describes puzzles as questions that challenge the player, require their deduction according to the rules, and don't depend on chance or actions from other players (2010). By the same token, he points out that designers are also bound by the rules, and that these dictate the ultimate designs.

## 2.2 Existing Approaches

There has been considerable research interest into Japanese logic puzzles. Proposed methods for their solution include: evolutionary approaches (Sato, 2010), SAT (satisfyability) problem approaches (Herting, 2004), Binary Decision Trees (Knuth, 2011), Monte Carlo methods (Cazenave, 2009), and "branch and bound" DFS (Jing et al, 2007). Yato (2003) demonstrates that it is hard to determine whether a given solution is unique, for at least one type of Japanese logic puzzle.

By contrast, there has been relatively little work on *procedural content generation* (PCG) approaches for logic puzzle levels. Methods have been proposed for the generation of *nonograms* or Picture-logic puzzles (Ortiz-Garcia et al, 2007) and general logic puzzle levels using evolutionary techniques (Ashlock, 2010). These approaches produce levels of desired difficulty but do not otherwise address issues of aesthetics or player preference (apart from difficulty) in level design.

While some computer-generated content does make it to print, these PCG methods typically have a commercial motivation and are therefore not publicly released, and it is probably safe to say that no current PCG approach can produce puzzle designs as good or varied as those produced by the best human designers. However, we are starting to see board games of publishable quality being produced by automated systems based on aesthetic measurements (Browne and Maire, 2010) and similar approaches may be applied to puzzle and level design.

# 3 Hour Maze

We now describe Hour Maze in more detail. Fig. 2 shows an example 7x7 Hour Maze level (left) and its solution (right). Note that this simple example is a miniature size with only four colour sets for the sake of clarity; the full puzzle size is 12x12 and includes twelve colour sets.

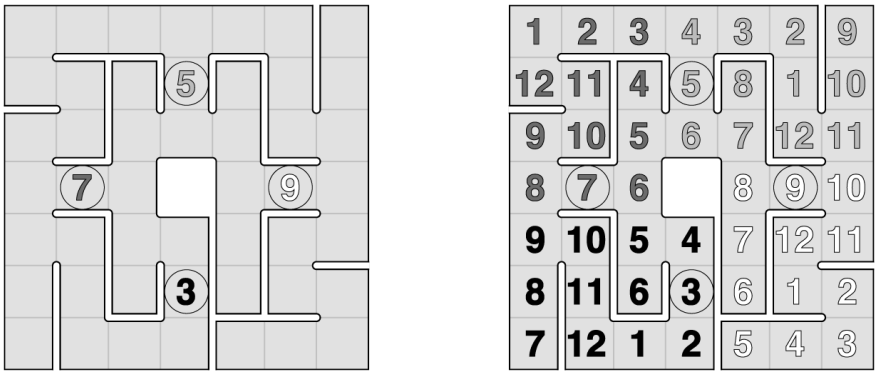


**Fig. 2.** Example level with solution.

## 3.1 Rules

The player must fill the grid with hour values (numbers ranging from 1 to 12) and assign each one a colour such that:

- There is exactly one number of each colour.
- Numbers adjacent in the grid (i.e. not separated by a wall) must be sequential on the clockface. For example, 12 can only be adjacent to 1 and/or 11.
- Each colour group forms a single connected set.

One number of each colour is revealed as a *hint* to start the player off (circled). The numbers within each colour need not start at 1, as long as they are sequential.

## 3.2 Definitions

We distinguish between the *puzzle* (the unique combination of rules that defines Hour Maze) and its *levels* (each specific instance of the puzzle). Each level consists of a:

- *Grid*: Square grid, typically square in footprint.
- *Maze*: The wall pattern within the grid.
- *Hint set*: The number of each colour revealed to the player (circled).
- *Solution*: The unique {*number*, *colour*} pairs realised for each cell.

The square grid enforces a parity on each puzzle (Fig. 3). The player can deduce from a given hint set whether each cell will contain an even (*e*) or odd (*o*) number.

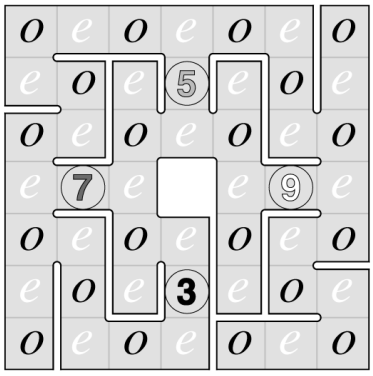


**Fig. 3.** Hints enforce a parity on each level.

Each coloured group of numbers 1..12 is described as the *hour set* or *colour set* of colour *c*.

## 3.3 Walkthrough

To give a taste of the puzzle, we now describe the steps that a player might take to solve the 7x7 level shown in Fig. 2. This small example can be solved using relatively few strategies, while full size 12x12 levels with twelve hour sets are typically harder and require more sophisticated strategies for solution.

Firstly, observe that white is the only colour that can reach dead end ***a*** (Fig. 4, left) within 12 steps. This entire corridor must therefore be white (right).

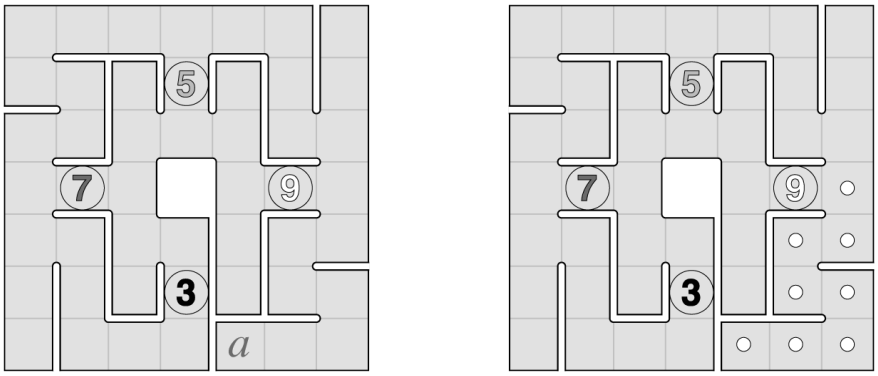


**Fig. 4.** Only white can reach cell ***a***.

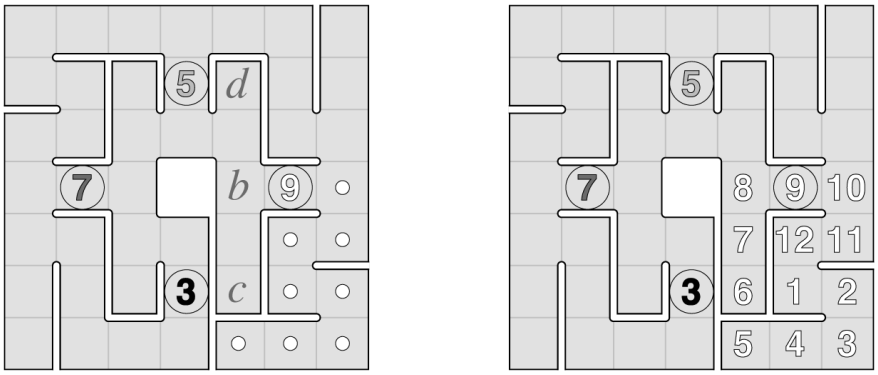


**Fig. 5.** White extends to cell ***c*** with known orientation.

Colour sets cannot branch (explained shortly), so the white set must extend to the left of the hint 9 through cell ***b*** (Fig. 5, left). It cannot extend upwards from ***b*** as this would isolate corridor ***c***, so the white set must extend to ***c***. Cell ***b*** must be numbered either 8 or 10 due to the 9 hint, and is three steps away from the hint 5 so cell ***b*** must be 8; the ordering of the white group is therefore known (right).

Using similar logic, light grey is the only colour that can reach dead end ***e*** and this set must extend to dead end ***d***, as any other colouring would isolate at least one corridor (Fig. 6, left). The position of all light grey cells is therefore known and their number order determined by the nearby white hour set (right).

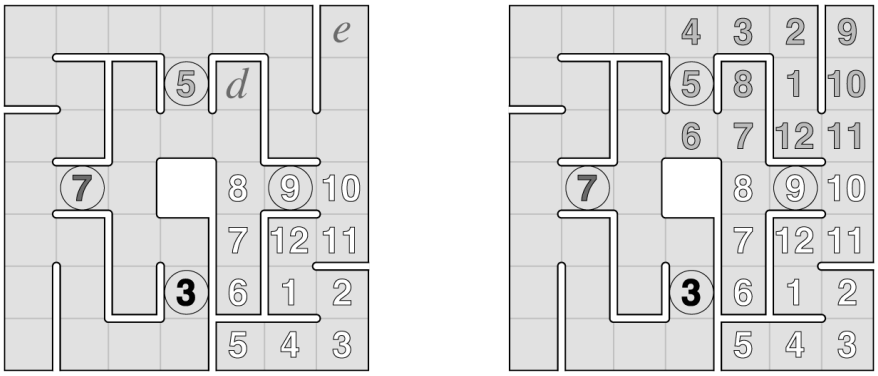


**Fig. 6.** Light grey spans cells ***d*** and ***e*** with known orientation.

Fig. 7 (left) shows a tentative black extension to a nearby dead end that would isolate the cell marked ***f***; black cannot make this extension so the dark grey set must visit this dead end instead. Dark grey is also the only remaining colour that can visit dead end ***g***, so the dark grey set must extend between ***f*** and ***g*** with number order determined by the light grey set (right).

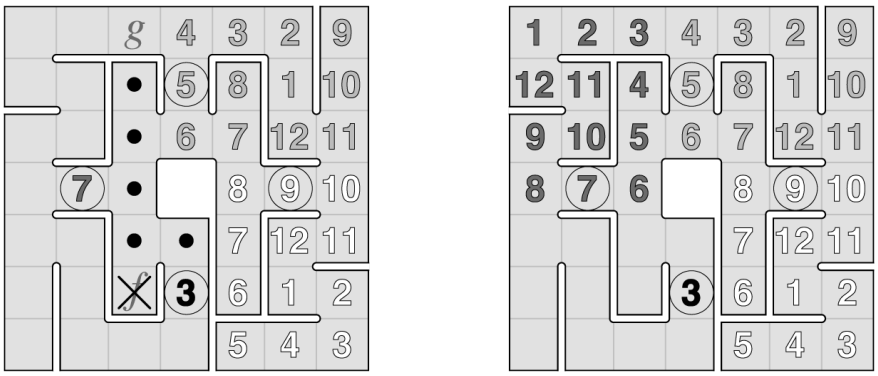

**Fig. 7.** An invalid black placement defines the dark grey span.

All remaining cells must be black (Fig. 8, left) with number order determined by the dark grey set (right). This level has now been solved using pure deduction.

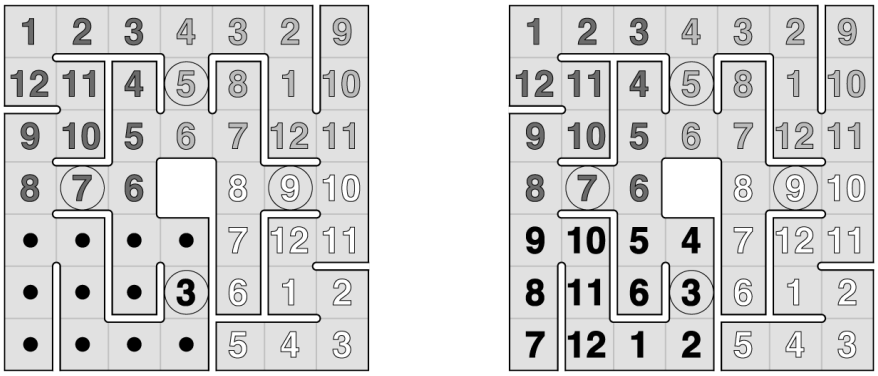

**Fig. 8.** Black fills the remaining cells with known orientation.

We emphasise again that small 7x7 levels such as the example shown can generally be solved easily with few strategies, whereas full size 12x12 levels can be significantly deeper and require more complex and varied strategies to solve.

## 3.4 History

The idea for Hour Maze was devised by puzzle designer Mike Reilly in 2007. The original incarnation of the puzzle used colour to demarcate hour sets without re-

gard for connectivity; colour provided decoration and clarity but was not at that stage an integral part of the puzzle. Colour was removed altogether for a black and white print run that presented levels as monochrome collections of *H* hour sets.

At this stage Hour Maze was not a pure deduction puzzle as each level supported multiple solutions and was typically solved by intuition, guesswork and backtracking. However, the notion of group colour provided an obvious way to promote the puzzle to the pure deduction class with one simple rule; that each colour set form a single connected group. This additional rule was found to add strategic depth and forms the current rule set as implemented for the iOS prototype.

We considered also applying an additional Sudoku-style rule forbidding the same number to occur in any row or column. However, this was found to be unnecessary (the connected colour rule adds sufficient depth) and would have had the unfortunate side-effect of placing Hour Maze as yet another Sudoku variant. As Higashida (2010) says, '*the simpler the rule is, the better*' (p218).

Higashida points out that the usual deductive rules were relaxed for the Nintendo DS implementation of his popular Number Link puzzle (2010) by allowing multiple solutions for each level and imposing a time limit. Players were encouraged to replay each level in an effort to improve their best time, which introduces a new competitive element but means that Number Link DS is no longer a pure deduction puzzle. We decided not to follow this route for the iOS implementation of Hour Maze but to keep it as a pure deduction puzzle, for the sake of elegance and the added challenge to players.

# 4 Automated Solution

The first step in automated level design is to define an automated solver that can prove whether a given level is deducible or not. This section introduces a number of strategies for solving Hour Maze puzzles, and methods for their automation in order to perform deductive searches.

## 4.1 Strategies

The strategies employed when solving Hour Maze puzzles typically break down into *number-based* and *colour-based* strategies. These are local in nature, and the aim of each is to successively eliminate invalid number or colour choices, respectively, until only the solution remains.

Table 1 lists the main Hour Maze strategies which were implemented for the automated solver. These are explained in further detail, with examples, in Appendix A.

| Number-Based Strategies | Colour-Based Strategies |
|---|---|
| 1. Number Count. | 5. Colour Count. |
| 2. Number Availability. | 6. Colour Availability. |
| 3. Neighbour Conflict. | 7. Colour Reach. |
| 4. Potential Neighbour Conflict. | 8. Colour Fill. |
| | 9. Colour Exclusion. |
| | 10. Colour Bounds. |
| | 11. Colour Isolation. |

**Table 1.** Number-based and colour-based strategies for Hour Maze.

In the following discussion, *H* denotes the number of hour (colour) sets. The number and/or colour of a cell are *realised* if reduced to a single valid choice.

## 4.2 Deductive Search

These strategies are used to form a *deductive search* method. The algorithm works by initially setting all valid colours and numbers as potential choices for each cell (except for hint cells) then iteratively eliminating these choices until only one possible colour and number remains for each cell.

An attractive aspect of deductive search is that any solution found is guaranteed to be unique for that level. The basic algorithm is outlined below.

### 4.2.1 Algorithm

The search state is initialised as follows:

```
for (each cell c)
    if (hintNumber[c] != 0)
    {
        // Known hint
        number[c] = hintNumber[c];
        colour[c] = hintColour[c];
    }
    else
    {
```

```
            // Unknown number and colour
            number[c] = 0;
            colour[c] = 0;
            numberChoices[c] = allNumbers[parity];
            colourChoices[c] = allColours;
        }
```

For each cell, if it is a hint then the known {*number*, *colour*} pair is stored for that cell, otherwise the choices for that cell are initialised to include all possible numbers and colours. `allNumbers[parity]` is a bitset that includes a flag for either all even or all odd hour values in the range 1..12, depending on the parity defined by the hint set (this step immediately halves the number of potential number choices). `allColours` is a bitset that includes a flag for all possible colour values in the range 1..*H*.

The following steps are then iteratively applied until either a solution is found or the deductive process fails, in which case the level does not have a deducible solution. Failure will occur either when all possible choices for a cell have been eliminated or an iteration passes without a number or colour choice being eliminated:

```
iterations = 0;
while (!isSolved())
{
    iterations++;
    if
    (
        badCell()
        ||
        !resolveNumberChoices()
        &&
        !resolveColourChoices()
        &&
        !resolveForcedColours()
        &&
        !eliminateColoursByNumber()
    )
        return 0;  // not deducible
}
return iterations;
```

`badCell()` returns `true` if `numberChoices[c]` or `numberChoices[c]` is 0 for any cell, in which case that cell would have no remaining valid number or colour choices.

`resolveNumberChoices()` visits each cell and eliminates any number choice that violates any of the number-based strategies described above. `resolveColourChoices()`, `resolveForcedChoices()` and `resolveColoursByNumber()` visit each cell and eliminate colour choices that violate any of the colour-based strategies described above (these are split into multiple methods due to their added complexity). Each of these methods return `true` if they result in any number or colour choice elimination being performed that iteration.

`resolveNumberChoices()` also marks the number for a given cell as "realised" if it has only valid number choice remaining by setting `number[c]` to that (non-zero) value, and `resolveColourChoices()` marks the colour for a given cell as "realised" if it has only valid colour choice remaining by setting `colour[c]` to that (non-zero) value. `isSolved()` returns `true` if all cells have a realised colour and number.

Note that this deductive approach is strictly strategy-based and does not involve any form of lookahead. Lookahead is not typically required due to the localised nature of the puzzle, and because several of the strategies chunk useful knowledge that might otherwise require combinatorial search.

A search result of 0 due to the complete elimination of colour or number choices for a given cell, making it unrealisable, indicates that the level has no possible solution. On the other hand, a search result of 0 due to the deduction process stalling (i.e. failing to eliminate any colour or number choices for a given iteration) only indicates that the level is not deducible using the given strategies. A unique solution could conceivably be found using additional strategies or some form of lookahead search for proving/disproving colour and number values; we are following this point up in related work.

### 4.2.2 Performance

This deductive search approach is reasonably efficient at finding (or disproving) solutions. Table 2 lists the mean solution times using deductive search for maze sizes 5x5 to 12x12, and their standard deviations (σ). Timings were averaged over 3,000 randomly generated levels (10 repetitions each) using a single dedicated thread on a MacBook Pro machine with *i*5 processor. Solution speeds for smaller levels are sufficient for the current application, but speed can become a consideration for full size 12x12 levels which can involve considerable combinatorial complexity (discussed shortly).

| Maze Size | Hour Sets | Solution Time | σ |
|---|---|---|---|
| 5x5 | 2 | 0.0047*s* | ±0.00077 |
| 6x6 | 3 | 0.010*s* | ±0.0029*s* |
| 7x7 | 4 | 0.017*s* | ±0.0035*s* |
| 8x8 | 5 | 0.026*s* | ±0.0064*s* |
| 9x9 | 6 | 0.036*s* | ±0.0089*s* |
| 10x10 | 8 | 0.060*s* | ±0.018*s* |
| 11x11 | 10 | 0.089*s* | ±0.025*s* |
| 12x12 | 12 | 0.11*s* | ±0.028*s* |

**Table 2.** Mean solution time by maze size.

An additional benefit of deductive search is that information about the number of deductive iterations and strategies required to solve a level is produced as a by-product of the search. This information can be harnessed to estimate the depth or difficulty of a given level, as described in the following section.

# 5 Puzzle Metrics

This section describes metrics used to describe the most important aspects of a given level, namely its visual symmetry and solution depth. Visual symmetry appeals to the aesthetic sense of the player and can be used to convey an impression of handcrafted rather than computer-generated design (explored further in Section 7). Deeper puzzles provide challenges that appeal to the intellect of the player, which can increase the addictiveness and replay value of a puzzle.

## 5.1 Symmetry Metrics

Symmetry metrics describe the degree of symmetry visible in each level design. There are two types of measurable symmetry: *wall symmetry* and *hint symmetry*.

Seven symmetry types are supported, as shown in Fig. 9. These include no symmetry (left) and translation, reflection and rotation x2 and x4. *Sym* denotes the total number of symmetries (7).

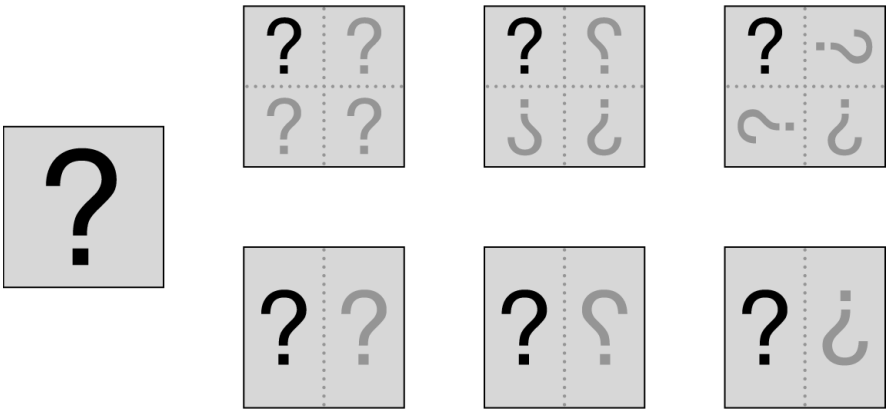

**Fig. 9.** Supported symmetries.

***Wall Symmetry* ($S_w$)**: Wall symmetry is a measure of the degree of symmetry in a maze's wall placement.

$$S_w = \max_{[0 \leq s < Sym]} \left( \frac{agree_w(s)}{T_w} \right) \tag{1}$$

where $agree_w(s)$ returns the number of walls that agree with symmetry type $s$, i.e. the number of walls which have the full complement of translated, reflected or rotated counterparts, as appropriate for that symmetry type. $T_w$ is the total number of walls. The final value is the maximum symmetry for any symmetry type $s$.

***Hint Symmetry* ($S_w$)**: Hint symmetry is a measure of the degree of symmetry in a level's hint placement.

$$S_h = \max_{[0 \leq s < Sym]} \left( \frac{agree_h(s)}{H} \right) \tag{2}$$

where $agree_h(s)$ returns the number of hints that agree with symmetry type $s$, i.e. the number of hints which have the full complement of translated, reflected or rotated counterparts, as appropriate for that symmetry type. $H$ is the number of hour sets. The final value is the maximum symmetry for any symmetry type $s$.

***Symmetry* ($S$)**: The overall symmetry value for a level $S$ is the average of its wall symmetry $S_w$ and hint symmetry $S_h$ values:

$$S = \frac{S_w + S_h}{2} \tag{3}$$

## 5.2 Depth Metrics

Depth metrics indicate the depth or difficulty of a given level. This may not be immediately apparent from a visual inspection of a puzzle, but can be estimated as a by-product of the deductive search process. We distinguish between *search depth* and *strategic depth*.

***Search Depth* ($D_i$)**: Search depth refers to the number of deductive iterations required to solve a level. If the search depth is small, this indicates that multiple deductions can be made across the level with each iteration, hence the puzzle has more readily available clues at each point and is more easily solvable. If the search depth is large, on the other hand, this indicates that there are relatively few deductions possible per iteration, and the player must search for each opening which makes the level harder to solve.

$$D_i = max(1, log_{10}(iterations{+}1)/2) \tag{4}$$

where *iterations* is the number of deductive search iterations require to solve the puzzle. The $\log_{10}$ of this number is taken and divided by 2 then capped at 1. This gives a continuous mapping of iteration counts from 0 to 100 to the range [0..1] and a value of 1 for counts above 100. Full size 12x12 levels can require over 200 deductive iterations for solution, but for the purposes of the experiment (described in Section 7) it was beneficial to focus on smaller iteration counts typical of smaller puzzles.

***Strategic Depth* ($D_s$)**: Strategic depth refers to the number of different strategies required to solve a level. This is estimated by disabling each strategy in turn then testing whether the level is still solvable; the final value is the ratio of strategies required to solve the level:

$$D_s = \frac{\sum_{s=0}^{Str} required(s)}{Str} \tag{5}$$

where *Str* denotes the number of available strategies and *required*(*s*) indicates whether the level becomes non-deducible if strategy *s* is removed from the deductive search.

***Depth* (*D*)**: The overall depth value for a level $D$ is the average of its search depth $D_i$ and strategic depth $D_s$ values:

$$D = \frac{D_i + D_s}{2} \tag{6}$$

# 6 Level Design

This section describes the process of automated level design. This includes the requirements for good level design, approaches that a human designer might take to manually handcraft levels, and a method for automatically generating levels of specified difficulty that attempt to incorporate at least some handcrafted touches.

## 6.1 Requirements

Hour Maze levels must satisfy the following mandatory requirements:

- Square grid.
- Number of cells $N$ a multiple of 12.
- Fully connected, i.e. all cells reachable from all others by adjacent steps.
- Each cell assigned exactly one colour and exactly one number.
- Non-sequential number pairs separated by walls.
- Same-coloured cells form a single connected set.
- No 2x2 gaps of empty cells.
- Deducible solution.

2x2 gaps are forbidden as they tend to reduce the maze-like nature of the puzzle. Hour Maze levels should ideally satisfy the following optional requirements:

- Square footprint.
- Wall symmetry (where possible).
- Hint symmetry (where possible).

Levels with square footprints are preferred for aesthetic reasons, although this is not strictly a requirement of the puzzle. Wall symmetry and hint symmetry are mainly for aesthetic reasons but also serve a more subtle purpose; they are more likely to give the player the impression that the level was carefully handcrafted by a human designer, which is more likely to increase their appreciation of the puzzle. The importance of this impression is discussed in more detail in Section 7.

## 6.2 Manual Level Design

Initial tests quickly revealed the difficulty of manually designing Hour Maze levels, especially for larger sizes. For example, Fig. 10 shows two degenerate wall constructions that would make level generation impossible.

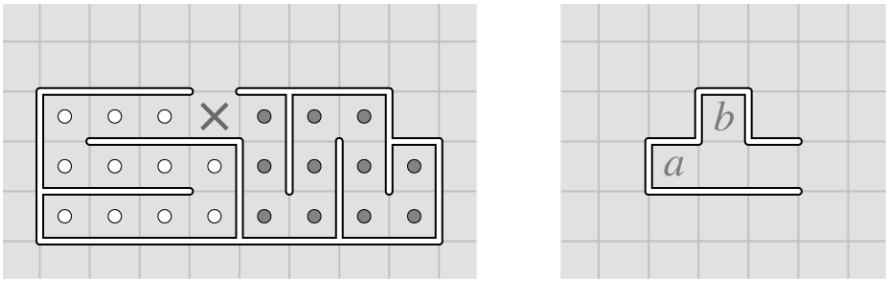


**Fig. 10.** Degenerate wall constructions.

The example on the left shows two corridors of 11 coloured cells sharing a common exit marked X. This common cell can only be assigned one colour, hence it is not possible to extend the other colour to the required 12th cell to complete its set.

The example on the right shows a corridor with two dead ends marked ***a*** and ***b***. Any colour set that visits both ***a*** and ***b*** will be bounded at both ends to give a maximum length of 3, while any colour set that visits only one of these dead ends will isolate the other to give an uncolourable cell. Again, any it would not be possible to create a complete solution for any level that contained this (or a similar) wall pattern.

While it is possible to automatically test for such degenerate cases, these are only indicative of the difficulties facing the Hour Maze level designer. In order to avoid the need for educating the level designer in such subtleties and to allow the generation of a range of high quality content in a reasonable amount of time, it would be desirable to completely automate the design process.

## 6.3 Automated Level Design

The following section describes an automated level design process that has proven to produce acceptable results. We initially tried a process that mimicked how a human designer would approach level design, that is to create an attractive wall pattern first then populate it with colour and number. However, this proved difficult to implement and produced poor results for smaller levels and no results at all

for larger levels within acceptable times, due to the problems such as the degenerate wall patterns described above and the sheer complexity of the problem at larger sizes.

Instead, a simple paradigm shift provided a breakthrough: place colour and number first then place the walls to suit. This allowed the fast generation of deducible levels (several per second) even for larger sizes, although the results tended to be rather random and without apparent purpose in design. A typical overnight run would produce 30,000+ levels which would prove something of a headache to sort through looking for gems, which were not guaranteed. It proved beneficial to find a compromise in the automated design process and introduce symmetrical constraints at various points, which greatly improved the average quality of the resulting content.

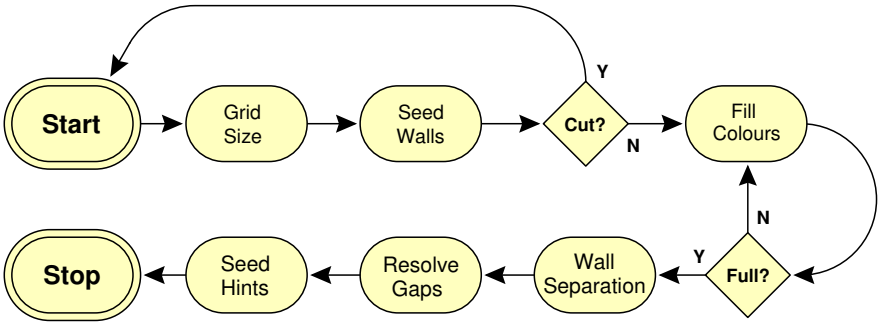


**Fig. 11.** The automated level design process.

Figure 11 shows a flow chart of the automated level design process. Firstly, a grid size is determined and seed walls placed until a wall set is found that does not cut the level into disconnected areas. The cells are then semi-randomly filled with colour sets, with each invalid colour being repeated, until all cells are coloured. Walls are then placed to separate conflicting numbers, 2x2 gaps of empty cells are resolved, and initial hints are selected. These steps are now described in detail.

### 6.3.1 Grid Size

The level size is defined by the desired number of hour sets $H$, which must in the range 2..12. The design process is illustrated using a small level size of $H$=4 for clarity.

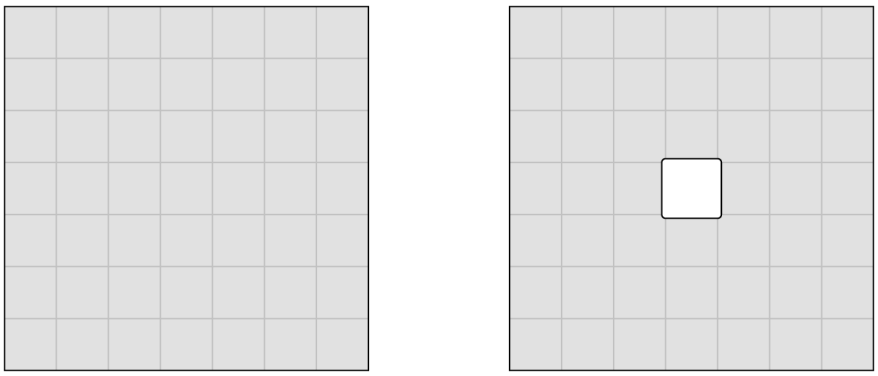

**Fig. 12.** 7x7 template symmetrically reduced from 49 to 48 playable cells.

Four hour sets will require 4 x 12 = 48 cells, so the smallest square grid that supports this number (7 x 7 = 49 cells) is chosen. The superfluous cell is flagged as `unused` and treated as an inverted "hole" or out-of-bounds region that is not assigned a colour or number. The central cell is chosen to maximise symmetry in the design (Fig. 12).

### 6.3.2 Seed Walls

To establish a symmetrical basis, a number of seed walls are placed in the grid and translated, reflected or rotated. For example, Fig. 13 shows four seed walls placed in the top left quadrant (left) that are reflected horizontally and vertically (right).

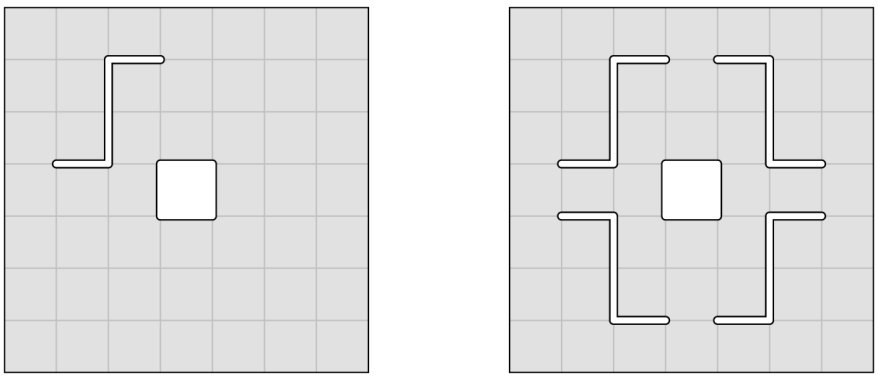

**Fig. 13.** Seed walls in top left quadrant, reflected horizontally and vertically.

One of the seven symmetry types is chosen at random and the initial seed walls placed in the top left quadrant or left half as appropriate (no seed walls are placed for the "no symmetry" case). The placement of seed walls and their symmetrical repetition may result in subregions walled off from the remainder of the grid, which violates requirement 3 that the maze remains fully connected. Such placements are discarded and regenerated.

### 6.3.3 Random Colour Walks

Once the seed walls are placed and repeated symmetrically, the *H* hour sets are added to the design one by one. The first set is added by selecting a random cell and placing a random walk of twelve adjacent cells (Fig. 14, left).

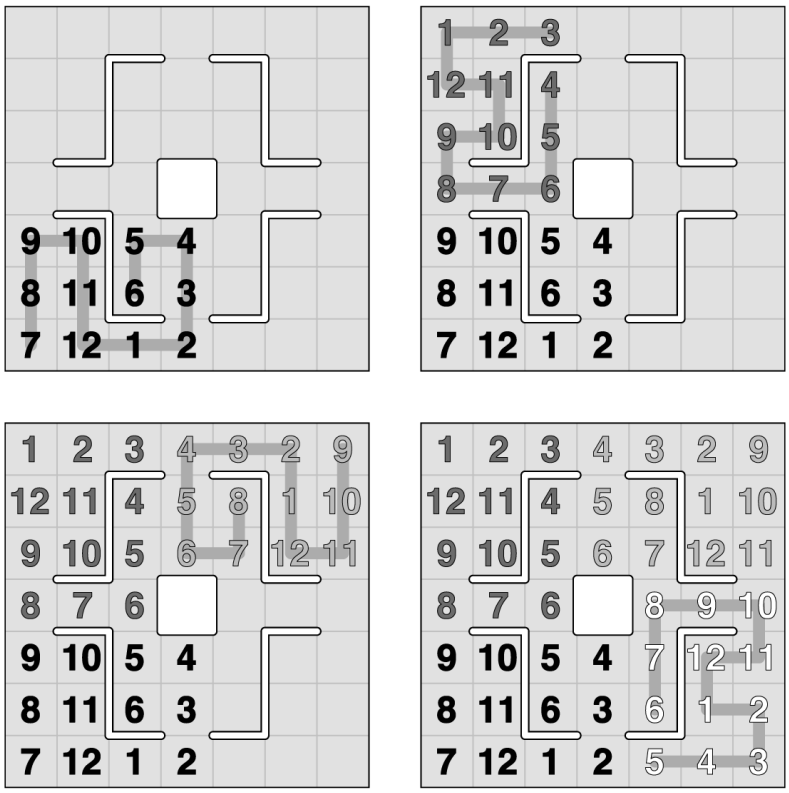


**Fig. 14.** Semi-random walk for each hour set.

Subsequent hour sets are placed by extending a known cell value to an empty adjacent neighbour and completing a random walk for it, extending in all available directions until all 12 cells are assigned for that set.

The placement of hour sets may result in uncoloured subregions of connected empty cells whose number is not divisible by 12. Such subregions will not allow the full placement of all remaining hour sets, so such degenerate sets are discarded and new colourings regenerated for them. Backtracking to regenerate the previously placed hour set occurs if this occurs an excessive number of times ($T$=200).

### 6.3.4 Mandatory Separation

Each cell has now been assigned a colour a number, but with no guarantee of sequentiality between hour sets or where hour sets double back on themselves. The next step is to place mandatory walls to separate adjacent non-sequential hour pairs. For example, Fig. 15 shows number conflicts marked ♢ (left) and walls placed to resolve these conflicts (right).

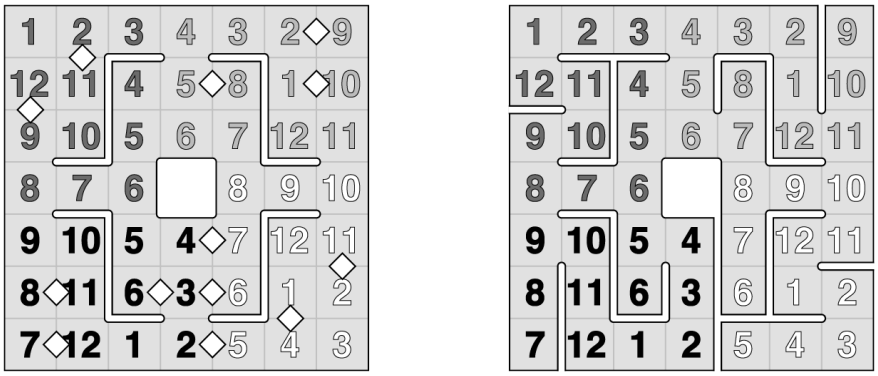

**Fig. 15.** Walls are placed between conflicting neighbours.

This placement of mandatory separating walls will typically reduce the symmetry of the resulting design, and is the step of the generation process most limits the impression of a handcrafted appearance. This can be seen in the example, in which the wall pattern is noticeably less symmetric following mandatory separation.

### 6.3.5 Gap Resolution

Wall placement is completed by adding walls to split any 2x2 gaps of empty cells within the design. For example, Fig. 16 shows the design so far with a single 2x2 gap marked, and two of the four possible walls that may split this gap dotted. The other two possible walls that would split this gap are not shown, as their placement would split a number set into two disjoint sets, hence they are illegal.

The topmost of the two possible walls is chosen (right) as it agrees in reflective symmetry with three other walls while the other wall choice only agrees in reflective symmetry with one other wall. Gap resolution is another way in which symmetry may be encouraged in the final design.

In some cases 2x2 gaps may overlap, in which case it is sufficient to select the single wall that resolves both gaps. It is generally best to place as few walls as possible as each wall placement applies one more constraint to the level which makes it that much easier to solve. Additional walls should only be placed if they significantly improve the symmetry or balance of the final design.

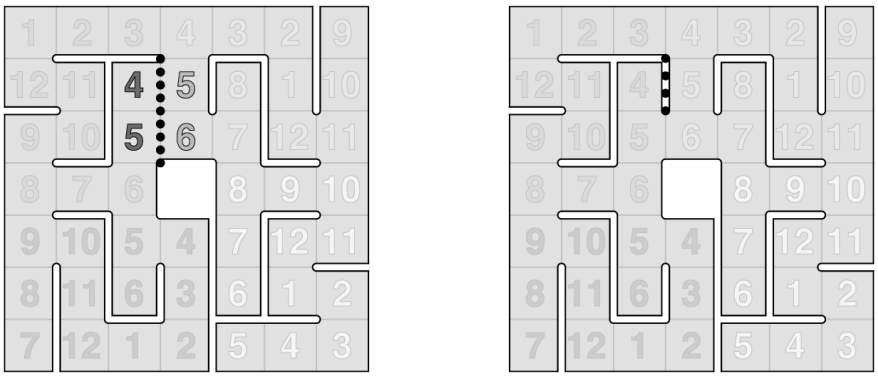

**Fig. 16.** A symmetrical wall is chosen to break the 2x2 gap.

### 6.3.6 Hint Selection

Having now completed the wall, colour and number placements for this level, the final stage is to choose its optimal hint set. Deducibility – which has conveniently been ignored until now – becomes an important factor in this final stage, as some hint sets will allow deducible solutions and some will not.

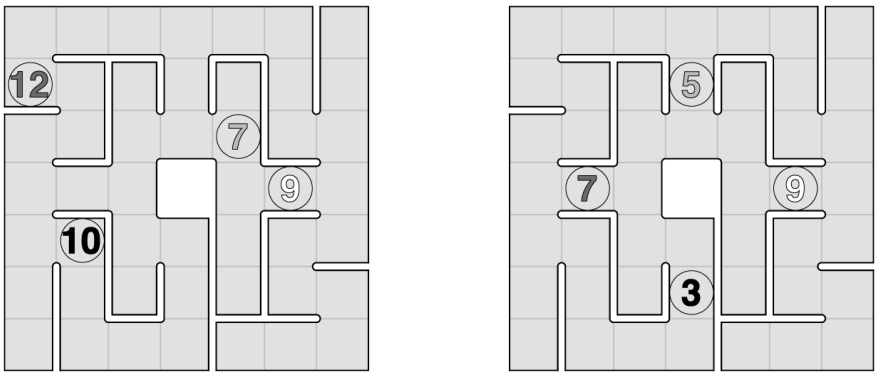


**Fig. 17.** Asymmetrical and symmetrical hint sets.

The choice of hint set will depend on the importance that the designer places on each of the two most important design considerations: depth and symmetry. Fig. 17 shows two deducible (hence valid) hint sets for the same level, with the set on the left favouring depth and the set on the right favouring symmetry. Rarely will a hint set favour both; these are the gems that the designer seeks.

Note that hint sets may exhibit two types of symmetry:

- *Geometrical symmetry*: Location within the grid.
- *Algebraic symmetry*: Numerical relationships between the hint values, e.g. all the same number, all different numbers (only really applicable to full size levels with the full complement of 12 hour sets), consecutive numbers, etc.

For example, Fig. 17 (right) shows strong geometrical symmetry and a weak algebraic symmetry in the odd sequence {3, 5, 7, 9}. The current application only measures geometric symmetry, but could benefit in future from also measuring algebraic symmetry.

Random hint sets are easily generated by selecting a random number in the range 1..12 to represent the hint for each hour set. Hint sets may be conveniently stored in a single 64 bit integer. However, a more systematic approach is required if hint sets optimising depth and/or symmetry are to be found.

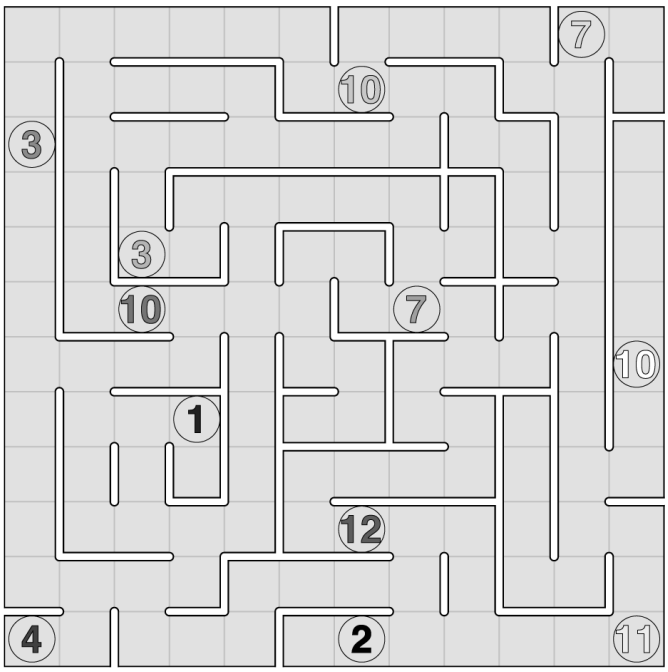


**Fig. 18.** Full size 12x12 level with twelve hour sets.

The number of possible hint sets $S_H$ for $H$ hour sets will be:

$$S_H = 12^H \tag{7}$$

For the small 7x7 size with 4 hour sets, there will therefore be $12^4 = 20{,}736$ possible hint set combinations, a manageable number that can be tested exhaustively in less than a minute on a standard laptop machine. Of these 20,736 we can expect around 2,404 (11.6%) to be deducible on average, but only one or two (0.0075%) to be both deducible and symmetric on average.

However, full size 12x12 levels (e.g. Fig. 18) are another story. The search space of $12^{12} = 8{,}916{,}100{,}448{,}256$ hint set combinations could require centuries of computation to explore exhaustively, and we are currently investigating more feasible sampling approaches based on Monte Carlo methods. In the meantime, our prototype Hour Maze app selects hint sets for larger levels by generating a random sample over a given time period and choosing the best, which produces acceptable if not ideal results.

## 7 User Survey

Given the ability to automatically generate levels parameterised for depth and symmetry, the question becomes these can potentially be of the same quality that a human designer would produce, in order to entertain players to the same degree. A user survey was conducted to explore these questions.

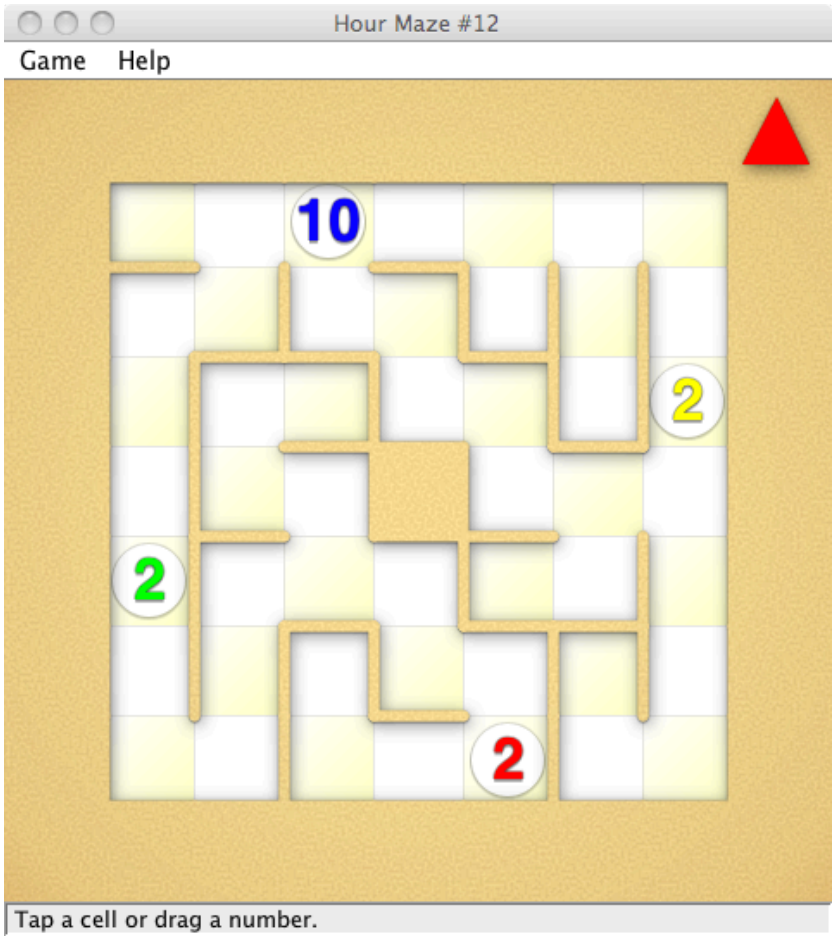


**Fig. 19.** Survey app showing introductory level #12.

## 7.1 Design

While this chapter focusses on design-time metrics, the user survey was designed more along the lines of standard game telemetry practices, which are well-represented in the other chapters of this book. Participants were asked to download and run a Java app that presented them with a number of Hour Maze levels then asked them questions about each level. The survey app, shown in Fig. 19, is based on the Hour Maze iOS prototype.

Upon launching the app, users were shown a welcome screen stating that:

> *You will be shown a number of puzzle levels, each of which may be designed by either human or computer.*

The phrasing of this sentence was critical as in fact *all* of the puzzles presented to the user were computer generated. The intention was to plant the idea that some of the puzzles would be handcrafted by human designers, and to investigate how this would affect the player's perception and enjoyment of the subsequent levels.

Users were presented with a series of Hour Maze levels randomly selected from a predefined data set of 80 designs. After solving each level, subjects were asked the following questions and their answers anonymously emailed to a dedicated gmail account (hour.maze.1@gmail.com, password available on request):

1. *Do you think that puzzle was designed by human or automatically generated by a computer?*

2. *Was that puzzle as interesting as the others you've done during the survey?*

An exception to this process was when a subject ran the survey for the first time, in which case they were shown the Hour Maze rules, started on the same introductory level (#12) and question 2 was rephrased as '*Did you find that puzzle interesting?'*

Since Hour Maze would be new to all participants, an initial learning curve could be expected and the first answer set from each subject was likely to be unreliable. Making level #12 the first to be shown when the survey app is launched meant that each subject's first attempt could be identified among the anonymous user data and culled.

## 7.2 Data

The data set of 80 predefined designs used for the survey was based on the 10 symmetrical mazes shown in Fig. 20 and 10 asymmetrical mazes shown in Fig. 21. Note that not all members of the "symmetrical" set are perfectly symmetrical –

90% symmetry was deemed sufficient for inclusion – and not all of the members of the "asymmetrical" set are perfectly asymmetrical.

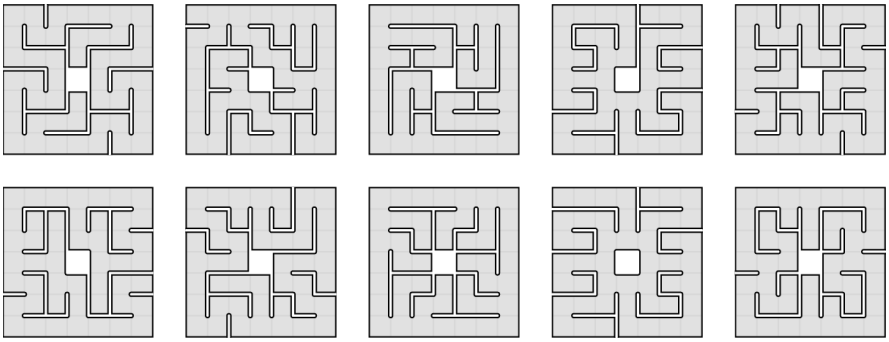

**Fig. 20.** Symmetrical survey mazes.

For example, the maze shown on the bottom right of Fig. 21 is a borderline case with a relatively high symmetry value (0.457) due to some walls being strongly symmetrical in the reflection (x4) group and some being strongly symmetrical in the rotation (x2) group, but the majority of walls are both strongly symmetric in any single group. However, this maze looks more symmetrical to the naked eye than others in this set, so future implementations might benefit from a symmetry metric that incorporates symmetries from more than one group at once.

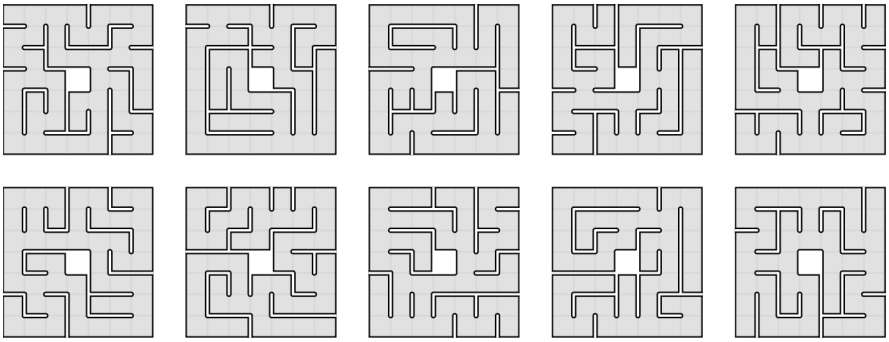

**Fig. 21.** Asymmetrical survey mazes.

For each of these mazes, four hint sets were selected based on their symmetry and the depth of the resulting level, to give the following versions:

1. Symmetrical, easy.

2. Symmetrical, hard.
3. Asymmetrical, easy.
4. Asymmetrical, hard.

All hint set combinations were exhaustively tested for each level to find the easiest (shallowest) and hardest (deepest) possible versions of the least and most symmetrical hint sets for each maze. The final data set of 80 levels consisted of these four versions of each of the 20 mazes.

### 7.2.1 Data Set Metrics

The smaller 7x7 maze with four hour sets was chosen as the test size for reasons of practicality. Initial tests indicated that larger mazes were more daunting and exhausting for new players, while smaller mazes meant an easier learning curve and shorter playing times, hence players were more likely to keep replaying the survey app to generate larger amounts of data.

Wall symmetry values within the data set ranged from .333 to 1.0 and hint symmetry values ranged from 0.0 to 1.0. Search depth ranged from 0.500 to 0.766 while strategic depth ranged from 1/9 = 0.111 to 5/9 = 0.556 (only the first nine strategies were implemented for the survey app). This narrow range in depth is the downside of choosing a smaller maze size as the resulting levels are necessarily limited in scope. This is not a major concern, however, as questions of symmetry and depth are secondary to the research aims.

## 7.3 Subjects

Subjects were recruited through online board game sites and newsgroups, and asked to confirm that they were at least 17 years of age. Approximately 50 subjects anonymously ran the survey over a one-week period, based on the number of results involving introductory puzzle #12. On average, each subject completed around 11 puzzle-and-answer sets, of which the first set was culled.

## 7.4 Results

553 answer sets were anonymously submitted, of which 54 involved introductory puzzle #12 and were culled, leaving a total of $R$=499 usable results.

Each answer set was of the form [*id hc ui*] where:

- *id* = Index of the level (1..80).
- *hc* = Subject thought the level was human (0) or computer (1) generated.
- *ui* = Subject found the level uninteresting (0) or interesting (1).

Level indices were matched with the corresponding metrics for each level, giving six more variables of interest for each answer set:

- search depth (*iterations*), strategic depth (*strategies*) and overall *depth*, and
- wall symmetry (*walls*), hint symmetry (*hints*), and overall *symmetry*.

The first result of note is that 47.5% (±8.76% at the 95% confidence level) of survey levels were deemed to be human-designed by participants, when in fact all were computer-designed. This is not surprising, but indicates that the deliberate ascertainment bias caused by the suggestion that some of the levels may be human-designed had the desired effect. Almost half of the survey levels (48.5% ±8.77%) were deemed to be "relatively interesting" by participants. A weak correlation ($r$ = -.209) was found between these two variables *hc* and *ui*, indicating that levels deemed to be human-designed were deemed to be interesting more often than not, but not with any great degree of certainty.

Similarly weak correlations were found between *hc*, *ui* and the other six variables regarding depth and symmetry. However, these still suggested some weak trends which were supported by feedback from survey participants:

- Deeper puzzles appear to be more interesting for users (many participants stated that the small 7x7 levels were in general too easy).
- Strategic depth appeared to be more important than iterative depth.
- Hint symmetry appeared to have a positive effect on player interest.
- Wall symmetry appeared to have a negative effect on player interest.

## 7.5 Discussion

The key findings are that almost 50% of the survey levels were perceived by players to be human-designed, and – more importantly – that there appears to be some relationship between a player's perception of a level as being human-designed and their interest in it. Within the limited range of the 7x7 levels, players appeared to enjoy those of greater strategic depth. It makes sense that strategic depth might be more important that search depth, as strategic depth more likely to capture the way in which players would approach a level themselves (i.e. by applying strategies) rather than the iterative and exhaustive elimination of possible choices; players will find short-cuts and creative ways to chunk repeated patterns into new strategies.

The fact that hint symmetry appeared to have a positive effect on player interest while wall symmetry appeared to have a negative effect is interesting. This may be because walls form the substrate or background of a maze, whereas hints are more immediately a part of the solution and their placement more attributable to an authorial intelligence. The unexpected negative effect of wall symmetry may also be due to the fact that symmetric wall sets are more likely to cause strategy repeti-

tion during solution, since a strategy that works in one part of the puzzle is also likely to work in other parts of the puzzle, translated, reflected or rotated as appropriate, which is boring for the player. This effect will be emphasised for smaller maze sizes in which the limited area means that any symmetry will imply significant repetition, whereas larger maze sizes will allow both symmetry and variety to coexist in the same design.

What do these results suggest for future level designs? Firstly, that levels should be designed with as much strategic depth as possible. Secondly, that hint symmetry is to be encouraged but wall symmetry is to be discouraged (especially where it limits depth).

# 8 Conclusion

This chapter introduces Hour Maze, a new type of pure deduction puzzle, and describes automated methods for the design and solution of levels. A number of strategies for approaching the puzzle are described, including their use in a deductive search method that can quickly solve levels and yield information about the iterative and strategic depth of each level. These metrics, in addition to symmetry information about wall placement within the maze and hint placement within the solution, provide useful information about each level.

There is a perception in similar puzzles, such as Sudoku, that computer-generated levels are necessarily inferior to those handcrafted by human experts. This becomes an important question for Hour Maze, for which it is possible to automatically generate large numbers of levels relatively quickly.

A user survey indicated that players perceived almost 50% of computer-generated levels to be human-generated if the suggestion is made that some of the levels are human-generated, and that these were found by players to be more interesting than those levels deemed to be computer-generated more often than not. Players appear to appreciate strategic depth in levels and hint symmetry in their design, but not necessarily wall symmetry; it appears that wall symmetry can actually harm level design if it reduces the scope for strategic depth.

## 8.1 Next Steps

Next steps will include the application of the knowledge suggested by the survey results in the future development of Hour Maze levels; we will endeavour to generate levels with stronger hint symmetry and greater strategic depth. This will not be a trivial exercise for larger maze sizes due to the combinatorial complexity involved, so we will investigate alternative methods for efficiently sampling the design space based on Monte Carlo methods.

The deductive search method could also be improved with the addition of further, more complex strategies, and the inclusion of a deductive lookahead mechanism to continue the iterative search when direct elimination stalls. Additional techniques for incorporating a sense of intelligence and personality behind the solution of automatically generated levels will also be investigated.

The approaches described in this chapter are not specific to Hour Maze, and have broader applicability to puzzles in general. We demonstrate how simple metrics may be measured for a given puzzle and provide useful insights into the design of its levels, whether automatically or manually generated, and how this information may lead to more interesting content in future.

## Acknowledgements

Thanks to Stephen Tavener for his insights into Hour Maze (and other puzzles) and interpretation of the results, the participants of the puzzle design survey, and Mike Reilly for his patience. I would also like to thank the anonymous reviewers for their excellent suggestions. This research was partly funded by EPSRC grant EP/IOO1964/1 as part of the *UCT for Games and Beyond* project.

## References

Ashlock D (2010) Automatic Generation of Game Elements via Evolution. In: Proc CIG'10, 289-296

Browne C, Maire F (2010) Evolutionary Game Design. IEEE T-CIAIG, 2(1):1-16

Cazenave T (2009) Nested Monte-Carlo Search. IJCAI 2009, 456-461

Herting S (2004) A rule-based approach to the puzzle of Slither Link. CO620 Research Project Report, University of Kent

Higashida H (2010) Machine-Made Puzzles and Hand-Made Puzzles. Nakatsu R et al (eds) ECS 2010, IFIP AICT 333, 214–222

Jing M-Q, Yu, C-H, Lee H-L, C, L-H (2009) Solving Japanese Puzzles with Logical Rules and Depth First Search Algorithm. In: Machine Learning and Cybernetics 2009, 2962-2967

Kanamoto N (2001) A well-made Sudoku is a pleasure to solve.
http://www.nikoli.co.jp/en/puzzles/sudoku/hand_made_sudoku.htm

Kendall G, Parkes A, Spoerer K (2008) A Survey of NP-Complete Puzzles. ICGA Journal, 31(1):13-34

Knuth DE (2011) Nikoli Puzzle Favors. In: Knuth DE (ed) Selected Papers on Fun & Games. CSLI Publications, Stanford

Nikoli (2001) Nikoli.com. http://www.nikoli.com/

Ortiz-Garcia, EG, Salcedo-Sanz, S, Leiva-Murillo, JM, Perez-Bellido, AM, Portilla-Figueras, JA (2007) Automated generation and visualization of picture-logic puzzles. Computers & Graphics, 31:750-760

Reilly M (2007) Hour Maze. http://hourmaze.com/

Salen K, Zimmerman E (2004) Rules of Play: Game Design Fundamentals.MIT Press, Cambridge

Sato Y (2010) Solving Sudoku with Genetic Operations that Preserve Building Blocks. In Proc: CIG'10, 23-29

Thompson M (2000) Defining the Abstract. The Games Journal. http://www.thegamesjournal.com/articles/DefiningtheAbstract.shtml

Yato T (2003) Complexity and Completeness of Finding Another Solution and Its Application to Puzzles. Masters Thesis, University of Tokyo

# Appendix A

# Hour Maze Strategies

This appendix describes the key Hour Maze strategies implemented for the automated solver (Section 4), with examples where appropriate. These generally break down into number-based and colour-based strategies, and tend to be local in nature. Most of these strategies aim to eliminate illegal number or colour choices until the solution is deduced.

In all cases $H$ denotes the number of hour (colour) sets. The number and/or colour of a cell are *realised* if reduced to a single valid choice. The *area* of a branch within the maze is the number of cells that it contains. A *dead end* is a cell surrounded by three walls.

## A.1 Number-Based Strategies

Number-based strategies focus on the elimination of illegal number choices.

### A.1.1 Strategy #1: Number Count

For each hour value $h$, if $H$ occurrences have already been realised then all further potential occurrences are eliminated; $h$ has been fully accounted for.

### A.1.2 Strategy #2: Number Availability

For each potential (or realised) colour $c$ for the given cell, hour values $h$ for which the pairing $\{h, c\}$ has already been used elsewhere are eliminated. There can only be one number of each colour.

### A.1.3 Strategy #3: Neighbour Conflict

Potential neighbours of hints and other realised numbers that do not differ by +1 or -1 are eliminated, as adjacent neighbours must be sequential. For example, Fig. 22 shows a realised number 7 and its potential neighbours 6 and 8. All potential neighbours must be recalculated as each potential number choice is eliminated.

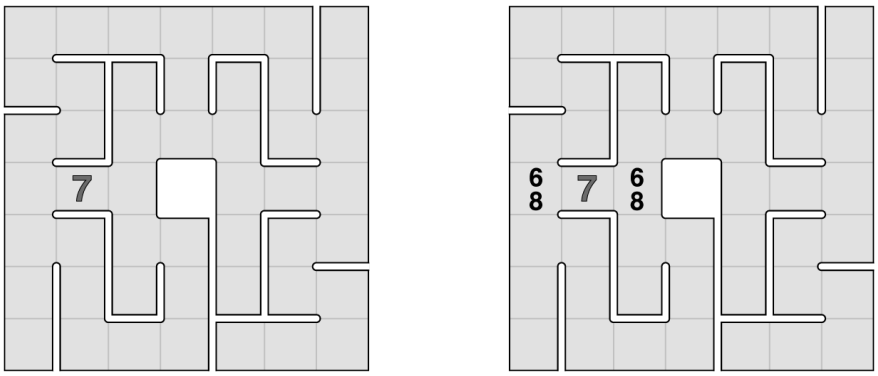


**Fig. 22.** Known numbers constrain potential neighbours.

### A.1.4 Strategy #4: Potential Neighbour Conflict

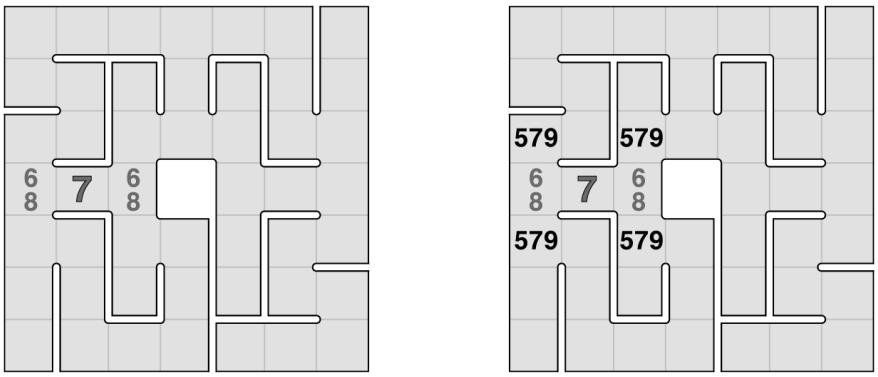


**Fig. 23.** Potential numbers also constraint potential neighbours.

Similarly, potential neighbours that do not differ by +1 or -1 from at least one realised or potential neighbour are eliminated. This strategy will have a trickle-on effect across the level. For example, Fig. 23 shows potential numbers 6 and 8, and their potential neighbours 5, 7 and 9. Again, all potential neighbours must be recalculated as each potential number choice is eliminated.

## A.2 Colour-Based Strategies

Colour-based strategies focus on the elimination of illegal colour choices.

### A.2.1 Strategy #5: Colour Count

As soon as 12 occurrences of any colour *c* have been realised, remove all other potential occurrences of c. There can only be 12 occurrences of any colour; one for each hour value.

### A.2.2 Strategy #6: Colour Availability

For each potential colour *c* for a given cell, if all potential hour values *h* for that cell have previously been realised in other {*h*, *s*} pairings, then eliminate *c* from that cell.

### A.2.3 Strategy #7: Colour Reach

For each colour *c*, eliminate *c* from any cell that cannot possibly be reached by that colour. If the number of cells for which colour *c* has been realised is $R_c$, then a cell will be reachable by colour *c* if it is not more than $R_c$ steps away from the nearest realisation of *c*.

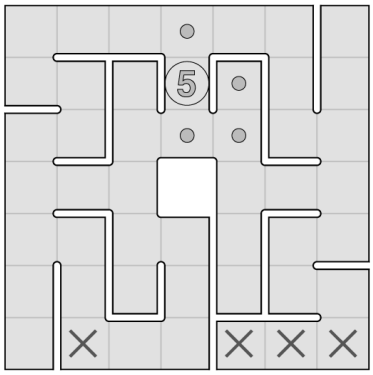


**Fig. 24.** Potential colour eliminated from unreachable cells.

For example, Fig. 24 shows a level with five cells known to be light grey. The number of unrealised members of this set is 12 – 5 = 7, so light grey is eliminated

from any cell that is more than seven steps away from this set (marked X). There will always be at least one realised number of each colour due to the hints.

### A.2.4 Strategy #8: Colour Fill

For each cell adjacent to a cell with realised colour *c*, if that cell belongs to an isloated branch of area less than 12 then all cells along that branch must also be colour *c*. This is because such branches cannot possibly contain any other colour.

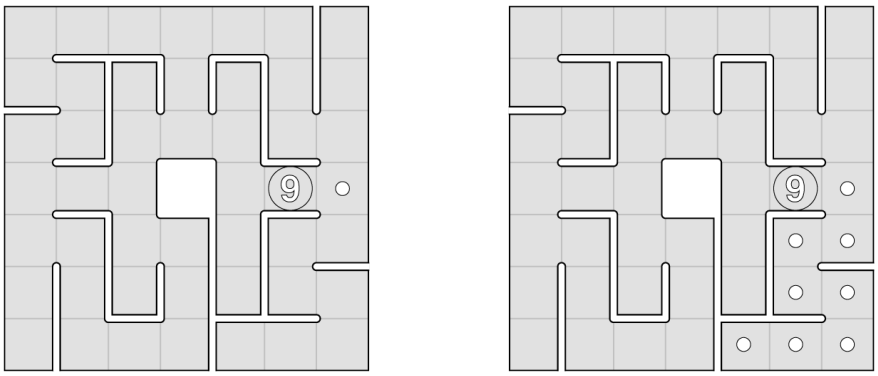


**Fig. 25.** Forced colour fill of an isolated corridor.

For example, Fig. 25 shows a level with two realised white cells that isolate a branch of seven cells in the lower right corner (left). No other colour can possibly reach any of these cells, hence they must be white (right).

### A.2.5 Strategy #9: Colour Exclusion

If any cell has a realised colour *c* and two of its neighbours are also realised as *c*, then *c* can be eliminated from any further neighbours. The three realised cells form a sequential path through the central cell and this path cannot branch.

Fig. 26 shows a white cell with two white neighbours and two unknown neighbours from which white can safely be eliminated (left). Imagine that the white cells are numbered {5, 6, 7} as shown on the right. The cells marked X would have no possible number choice as 5 and 7 are already taken; paths followed by hour sets must be sequential.

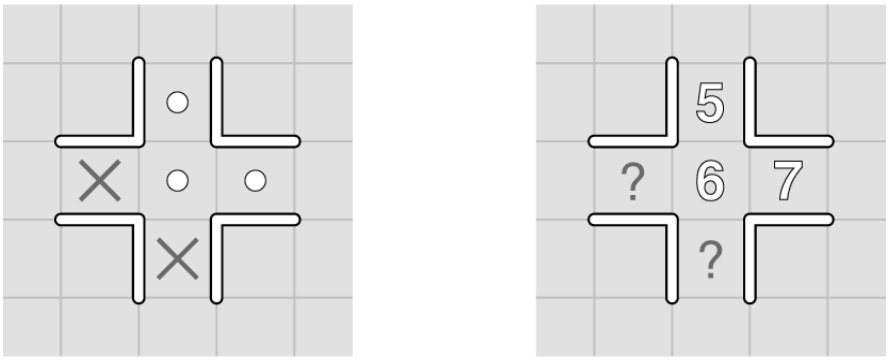


**Fig. 26.** Two neighbours of the same colour exclude other neighbours.

### A.2.6 Strategy #10: Colour Bounds

Any dead end cell with realised colour *c* bounds that colour set at one end. For example, the grey set shown in Fig. 27 is bounded at its topmost cell. This allows deductions to be made about possible extensions of set *c*.

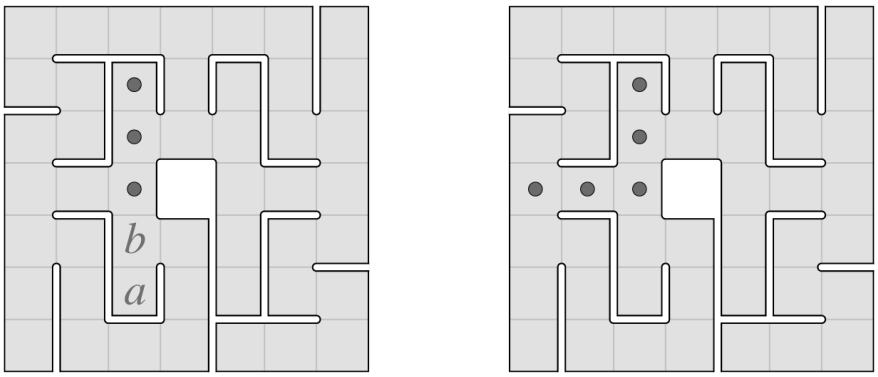


**Fig. 27.** The bounded colour shown cannot enter ***b***.

For example, set *c* cannot extend to cell ***a*** as that would bound it at both ends for a maximum size of 5, as hour set paths must be sequential and cannot branch; grey can be eliminated from cell ***a***.

### A.2.7 Strategy #11: Colour Isolation

Colour *c* can be eliminated from any cell for which such a realisation would isolate branches with area not divisible by 12. Such areas would not support complete hour sets and are hence invalid.

For example, cell ***b*** in Fig. 27 above cannot be coloured grey as that would isolate a branch of area 1 that contains cell ***a***. Grey can safely be eliminated from cell ***b***, hence the grey set must extend to the left as shown (right).

Fig. 28 shows another example with three cells realised as black and possible extensions to cells *j* and *k* (left). However, extension to cell *j* would isolate the branch containing cell *k* to a single cell so black can be eliminated from cell *j*.

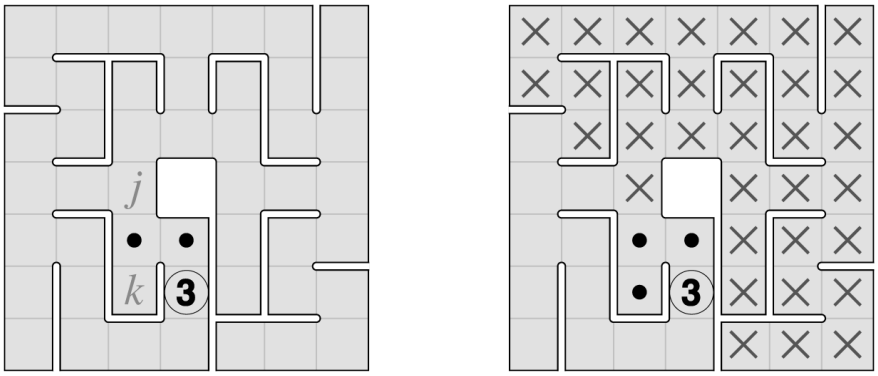


**Fig. 28.** Colouring *j* would isolate corridor *k*.

Note that the same deduction can be reached using different strategies. Cell *k* must be coloured black due to the Fill strategy (#8) as shown on the right, hence black can then be eliminated from cell *j* due to the Exclusion strategy (#9). Each level may contain such cases of strategic redundancy, which complicates the estimation of strategic depth (described in the Puzzle Metrics section 5).