Proceedings of the Twentieth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment (AIIDE 2024)

Procedurally Puzzling: On Algorithmic Difficulty and Player Experience in QD-Generated Logic Grid Puzzles

Fiona Shyne, Kaylah Facey, Seth Cooper

Khoury College of Computer Sciences, Northeastern University
shyne.f@northeastern.edu, facey.k@northeastern.edu, se.cooper@northeastern.edu

Abstract

Determining if and how the difficulty of algorithmic puzzle solvers is related to the difficulty and enjoyment for human players is a challenging task. In this work, we explored this relationship using logic grid puzzles. We used an algorithmic solver to estimate the difficulty of the puzzles by capturing the number of "solver loops" through the algorithm. This characteristic was used to generate and evaluate a set of puzzles of varying algorithmic difficulty using constrained MAP-Elites. Then, we ran a user study to gather information on the player experience of these puzzles. We tested the relationship between solver loops and player experience on generated puzzles and found that the number of solver loops is statistically significantly correlated with subjective perception of difficulty and borderline statistically significantly correlated with puzzle correctness.

Introduction

Creating new puzzles is a similarly challenging process to solving the puzzles themselves. Designers must provide enough information to make the puzzle solvable while controlling the difficulty. Often the difficulty of a puzzle is not obvious at the surface level. Further, designers need to create puzzles with a range of difficulties to suit a variety of players. Procedural content generation (PCG) has the potential to aid designers in this task; however, computational means can also struggle to measure difficulty. This work explores algorithmic solving difficulty as a proxy for player experienced difficulty in logic grid puzzles.

Logic grid puzzles (see Figure 1 as an example) are a popular variety of logic puzzles that can be played both on paper and on the computer. Players are given a list of hints written in natural language and must deduce the relationship between different entities. Logic grid puzzles can be created in a wide range of genres, sizes, and difficulties. The designer determines which information is provided in the hints, meaning the same solution can be derived from numerous hint lists.

We expand on our previous work (Shyne, Facey, and Cooper 2024) to develop a constrained quality diversity genetic algorithm that generates solvable and diverse logic grid puzzles. The genetic algorithm takes in a puzzle space, defined by a set of categories each with a set of entities, and returns puzzles with different solutions and algorithmic difficulties. Algorithmic difficulty is measured by the number of iterations through the hint list a solver requires to solve the puzzle. We refer to this as "solver loops". Within the relatively small puzzle space examined in this paper, our algorithm found at least three algorithmic difficulty levels for each possible solution.

We evaluate the relationship between solver loops and player difficulty with a user study. Participants played up to 4 puzzles with varying solutions and solver loops. For this study, we examined three hypotheses.

- **H1**: Algorithmic solver loops and player difficulty (subjective and behavioral) are positively correlated.
- **H2**: Experience with other kinds of logic puzzles is positively associated with performance on a new kind of logic puzzle.
- **H3**: Subjective difficulty and enjoyment have an "inverted U" relationship where enjoyment peaks at medium difficulty. There is still an "inverted U" relationship with different levels of experience, but players with more experience have peaks at higher difficulty.

H1 was partially supported. Specifically, we found that subjective difficulty was statistically significantly correlated with solver loops, and correctness was borderline statistically significantly correlated. H2 and H3 did not have statistically significant results.

This paper provides two major contributions: evidence of a positive relation of algorithmic solver difficulty with player difficulty, and an algorithm that produces solvable puzzles with a diversity of solutions and difficulties. These contributions can guide future work in the procedural generation of puzzles.

Related Work

Automatic Puzzle Generation

A number of techniques have been used to generate puzzles; De Kegel and Haahr (2020) extensively surveys the field. Search-based strategies are the most popular, including evolutionary search (Scirea 2020; Bonomo, Lauf, and Yampolskiy 2015; Ferreira and Toledo 2014; Mantere

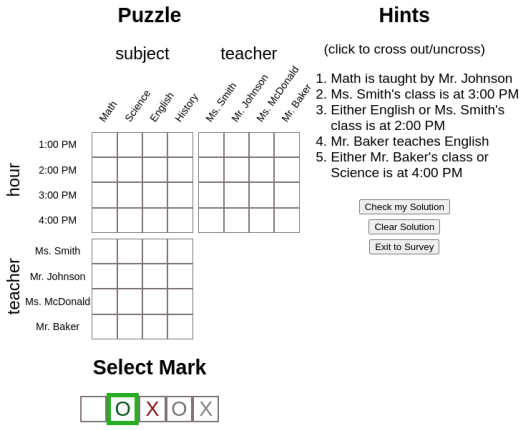


Figure 1: Example logic grid puzzle. This puzzle, along with the steps needed to solve it, was provided to participants as a tutorial.

and Koljonen 2007), tree-based search (Kartal, Sohre, and Guy 2021; Williams-King et al. 2021), constraint satisfaction (Fernández-Vara and Thomson 2012; Dart and Nelson 2012; Oranchak 2010; Batenburg et al. 2009; Hunt, Pong, and Tucker 2007; Chang, Fan, and Sun 2007; Sudha and Kalaiselvi 2020), and hill-climbing (van de Kerkhof et al. 2019). Grammar- and rule-based approaches are also common (Dong and Barnes 2017; Priemer et al. 2016; De Kegel and Haahr 2019; Valls-Vargas, Zhu, and Ontañón 2017; Venco and Lanzi 2021; van Arkel, Karavolos, and Bouwer 2015; Maji, Jana, and Pal 2016), and sometimes multiple techniques are combined (Khalifa and Fayek 2015; Smith et al. 2012). Less commonly, machine learning is used (Hald et al. 2020). Sudoku is among the most popular traditional logic puzzles to generate. Shyne, Facey, and Cooper (2024) is the only previous work known to us to generate grid-based logic puzzles.

Genetic Algorithms For PCG

Genetic algorithms (GAs) or Evolutionary Computation is a popular method of procedural content generation (PCG), having been used to generate many kinds of content, including platformer levels (Moghadam and Rafsanjani 2017; de Pontes and Gomes 2020), tower defense maps (Kraner, Fister, and Brezočnik 2021), and NPC characteristics (Mitsis et al. 2020). Often, in addition to aiming for novelty, these GAs aim to control the difficulty curve of a given game (Moghadam and Rafsanjani 2017; Kraner, Fister, and Brezočnik 2021; Mitsis et al. 2020). GAs are so popular in PCG that Togelius et al. (2011)'s survey of search-based techniques for PCG makes the claim that "evolutionary computation has so far been the method of choice among search-based PCG practitioners". Quality-diversity based approaches have been popular enough to merit their own survey (Gravina et al. 2019). GAs are also often combined with other techniques; for example, Soares de Lima, Feijó, and Furtado (2019) use a GA combined with planning techniques to generate quests, and Beukman, Cleghorn, and James (2022) uses a GA to evolve a neural network capable of generating *Maze* and *Super Mario Bros.* levels.

Puzzle Difficulty

Estimating puzzle difficulty is a popular area of research. Some papers explicitly model human puzzle solving strategies; Jarusek and Pelánek (2010) and Jarusek and Pelánek (2011) propose a "cognitive engineering" model based on human puzzle solving behavior that successfully models the difficulty of transport puzzles like Sokoban. Other papers identify features that are likely to correlate with difficulty and calculate weighted difficulty functions using user data or expert ratings (van Kreveld, Löffler, and Mutser 2015; Wang, Wang, and Sun 2012; Spierewka, Szrajber, and Szajerman 2021; Kartal, Sohre, and Guy 2021). Some papers propose generalized constraint propagation algorithms using assumptions about human problem solving (Hunt, Pong, and Tucker 2007; Pelánek 2011), while others directly rely on known puzzle-specific techniques (Chang, Fan, and Sun 2007; Xu and Xu 2009). It is also common to use computational techniques that do not directly consider human problem solving but are correlated with user data or expert ratings (Chen, White, and Sturtevant 2023; Naji and Salous 2024). The simplest method is to directly estimate difficulty by how many iterations a given algorithm takes to solve the puzzle (Mantere and Koljonen 2007; Batenburg et al. 2009), as in our "solver loops" metric. Pusey, Wong, and Rappa (2021) propose a set of generalized user data based metrics that may be used to evaluate the challenge of puzzles. Kristensen and Burelli (2024) propose methods of estimating personalized and cohort difficulty for new and existing puzzle levels.

The Relationship Between Difficulty and Enjoyment

Previous work has investigated the relationship between difficulty and enjoyment in games. Despite this, there is no clear consensus on the direction or shape of the difficulty-enjoyment curve, or even its existence. Intuitively, most researchers expect an "inverted U" curve; that is, players should prefer games that are not too easy, not too hard. Abuhamdeh and Csikszentmihalyi (2011) first found such a relationship for online chess players. Brändle, Wu, and Schulz (2024) supports the concept of an inverted U curve with respect to video games, though for the easiest levels, they find that increased difficulty is correlated with reduced enjoyment. Ma, Pei, and Meng (2017) and Meng et al. (2016) support the inverted U concept with neurological indicators of engagement. Colwell and Glavin (2018) indirectly supports the concept by showing that dynamic difficulty adjustment improves player experience for both strong and weak players, suggesting that players prefer challenges that match their own abilities. On the other hand, Demediuk et al. (2019) and Lomas et al. (2017) suggest that difficulty and enjoyment are predominately negatively correlated, while Cutting et al. (2022) and Biemer and Cooper (2024) fail to find any significant relationship.

System Overview

Our system can generate logic puzzles with different solutions and varying difficulties for each solution. This extends our previous work (Shyne, Facey, and Cooper 2024) that was able to generate difficult logic puzzles using a constraint-based GA. Our system implements a variation of constrained MAP-Elites (Khalifa et al. 2018), where one population contains only infeasible individuals and another population contains a MAP-Elites grid.

Logic grid puzzles require players to find relationships between entities, using a list of natural language hints. New puzzle spaces are specified by a set of categories, each with a set of entities. Categories can be numerical (e.g. times) or categorical (e.g. names). We can define the size of the puzzle space by M×N where M is the number of categories and N is the number of entities in each category. In this paper, we focus on 3×4 puzzles, specifically in the theme of a school schedule.

Representation

We used the same representation of logic grid puzzles from our previous work. Each puzzle is described by a list of hints generated using a grammar. This grammar contains the major types of hints found in these types of puzzles, and each has its own logical interpretation. Given a list of hints, a solver is able to return a complete or partial solution along with how many iterations through the hint set it took (which we will call "solver loops").

The genetic operators in this paper are also the same as in previous work. The initial population of individuals is made by randomly generating between 3 and 5 hints per puzzle. Mutation randomly removes or adds a hint. Crossover randomly shuffles the hints between the two children.

Solver

Shyne, Facey, and Cooper (2024) provides a rule-based algorithm that takes in a set of hints in the logic grid puzzle grammar and solves the puzzle to the best of its ability. The solver iterates through the hint list in order and applies marks to the grid based on the current state of the puzzle and the logic of the hint. For example, when the solver encounters "science is not 1pm" it will put an "X"on the science and 1pm cell. After examining a hint, it will examine to current game state to see if any other marks can be made; for example, if all cells in a row are "X"except for one it will add a "O"in the empty cell. It iterates through the hint list as many times as necessary until the puzzle is solved or no more marks can be made. One "solver loop" is defined as one complete iteration through the hint list.

Rather than using a standard SAT solver, the algorithm was designed to mimic how a human might approach solving these types of puzzles, based on the authors' own strategies. It only makes changes to the game state when it is confident a mark can be, and therefore no guesswork is required. Hoffmann et al. (2022) looked at puzzle-solving behavior and found that players often believe guesswork should not be required for puzzle-solving. It is important to note that since the solver looks at the hints in order, the same hint list arranged in a different order may result in different solver behavior. This is based on the assumption that humans are likely to approach the problem from top to bottom as well.

Genetic Algorithm

Our GA combines the constraint-based algorithm Feasible-Infeasible Two-Population (FI-2Pop) and the quality-diversity algorithm MAP-Elites. FI-2Pop addresses the problem of evolution within constrained problem spaces. It does this by maintaining two populations: an infeasible population where individuals are selected to approach feasibility and a feasible population that is selected based on optimization criteria. Shyne, Facey, and Cooper (2024) used FI-2Pop to produce solvable puzzles that were optimized to maximize difficulty. However, we want to generate puzzles with a variety of difficulties in this work.

The MAP-Elites algorithm generates high-quality solutions that vary by two diversity measures. The MAP-Elites population is represented by a two-dimensional grid of individuals, where each x and y coordinate is determined by the diversity measures. For each child produced, the appropriate grid cell is calculated. If there is not an individual stored in that cell or if the child has better fitness than the existing individual, it is placed in that cell. Otherwise, the child is discarded. In this paper, the two measures of diversity are difficulty (number of solver loops) and the string representation of the puzzle solution.

Khalifa et al. (2018) proposed a constrained MAP-Elites, where each MAP-Elite cell contains two populations, an infeasible and a feasible population. However, in our work, infeasible (not solvable) puzzles do not have a solution or difficulty and therefore cannot be placed in a MAP-Elite cell. To accommodate this, our MAP-Elite grid contains only feasible individuals, and we separately maintain an infeasible population. This algorithm is described in more detail in Algorithm 2.

Infeasible Population The infeasible population behaves similarly to the base FI-2Pop algorithm. Any individual that is infeasible (unsolvable) is placed in the infeasible population. Selection uses rank selection based on infeasible fitness. The infeasible fitness is calculated using the same formula as defined in Shyne, Facey, and Cooper (2024) and determines how close to becoming solvable the puzzle is.

MAP-Elites Grid Feasible individuals are selected from and placed in a MAP-Elites grid. The X dimension represents the number of solver loops, where the xth column contains individuals requiring $x + 1$ solver loops to complete. However, all puzzles with 10 or more solver loops are placed in the last column. The Y dimension represents the solution as a string. Since there are many solutions possible for each puzzle space (576 for 3×4 puzzles), the number of rows in the MAP-Elites grid grows throughout evolution. Every time the grid sees a new solution, it adds a row to the grid and assigns it a row index. This index can later be looked up in a hashtable.

To place a new child in the map, the row and column are used to determine a cell location. If that cell location

Algorithm 1: PlaceChild

Input : H a hint list, I the infeasible population, M a MAP-Elites grid

if H *is infeasible* **then**
  place H in I
else
  set l as the number of solver loops
  set i as the index of the solution of H in M
  set c as $M[i, l]$
  if c *is empty or* $|c| >= |h|$ **then**
    set $M[i, l]$ to H
  end if

is empty, it is automatically placed there. Otherwise, it replaces the existing elite in that cell only if the size of the hint list is at least as small as that of the existing elite. The number of hints is used as the MAP-Elites fitness indicator in order to reduce redundancies in puzzles and make them more comprehensible. When selecting an individual for mutation or crossover, a random elite from the grid is returned.

Genetic Algorithm Results

To test the capabilities of the GA, we generated 3×4 puzzles with a school schedule theme. These puzzles have three categories: Subject, Teacher, and Hour. The puzzle is complete when the solver can determine the subject and teacher for each hour.

We ran our GA for 10 trials, each for 3000 generations. The population size was set at 300, the mutation rate at 0.8, and the crossover rate at 0.6, with 10 infeasible elites being maintained.

The infeasible population very quickly converges to feasibility with maximum infeasible fitness being reached by generation two. The average hint size of the feasible individual in the MAP-Elites grid is shown in Figure 2. Note that a QD-score was not measured here, as hint size is a minimization problem (the QD-score is calculated using the sum of fitnesses, so it would increase as the grid was filled and decrease as members improved in fitness). The average converges to a little over 4 hints after 3000 generations. This average is fairly consistent across the number of solver loops, as shown in Figure 4, apart from the case of 1 solver loop where the average is higher. Looking at the total number of children produced for each solver loop count (see Figure 4), 2 and 3 solver loops were the most common. The GA was able to find at least one puzzle, by generation 1000, for all 576 possible solutions (as shown in Figure 3). Every solution found had at least three levels of difficulty (solver loops) by the last generation (as shown in Figure 5). Most solutions had 4 to 5 levels of difficulty. The maximum number of solver loops found was 6.

User Study

We conducted a user study to validate the correlation of solver loops with difficulty, as well as to gather player data

Algorithm 2: Two Population MAP-Elites

Input : N population size, E number of infeasible elites

set I to empty list
set M to empty MAP-Elites
for i *in* N **do**
  generate random hint list h
  PlaceChild(h, I, M)
end for
while *Stopping criteria not met* **do**
  create an empty list I_{temp}
  copy over top E individuals from I to I_{temp}
  for i *in (N / 2)* **do**
    $c_{feasible}$ = True at rate of $|M|$ / ($|M|$ + $|I|$)
    if $c_{feasible}$ **then**
      P_1, P_2 = select random from elites in M
    else
      P_1, P_2 = select from I based on infeasible fitness
    C_1, C_2 = cross over P_1 and P_2
    C_1, C_2 = mutate C_1 and C_2
    PlaceChild(c_1, I_{temp}, M)
    PlaceChild(c_2, I_{temp}, M)
  end for
  set I to I_{temp}
end while

that may be useful to the future generation and use of logic grid puzzles.

Puzzles

We selected 16 puzzles from one of the GA runs described in the previous section. The puzzles were chosen by taking the 1, 2, 4, and 6 solver loop puzzles from the 4 solutions that had all of those difficulties. This results in 4 distinct solutions, each with 4 levels of algorithmic difficulty (solver loops). This setup allowed us to control for both puzzle solutions and individual puzzles. We have provided all difficulties for one of the solutions used in the **Appendix**.

The hint grammar produces human-readable, but not necessarily grammatically correct hints. For the user study, hints were modified slightly to be grammatically correct. For example "The teacher Ms. Smith is the hour 2:00 PM" was changed to "Ms. Smith teaches at 2:00 PM."

Participants solved puzzles on a custom user interface designed for this study. Participants could select which mark to use for each grid cell. "O"represents two entities that are connected, and "X"represents two entities that are not connected. Participants could also remove marks and make "unsure" marks (for both "O"and "X"). Participants could also click on the hints to cross them out, to aid them in keeping track of which hints they were done with. When participants thought they had finished the puzzle, they could check if their answers were correct. Their answer was labeled as correct if all "O"(excluding "unsure" marks) were placed in the correct spots, but participants did not need to place "X"marks. At any point, participants could clear the puzzle

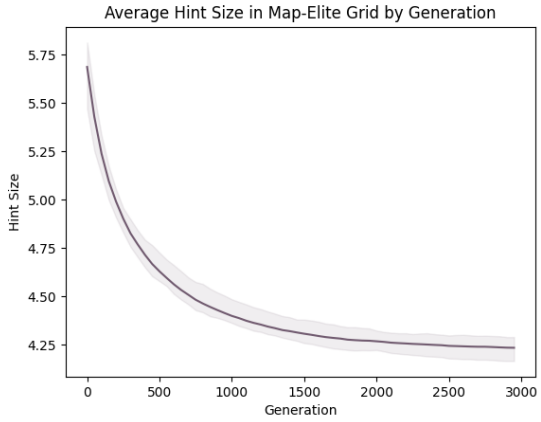


Figure 2: Average hint size in MAP-Elites Grid by generation. The solid line represents the average and the shaded area represents the min and max values between GA runs.

grid or exit the puzzle and continue to the survey.

Procedure

Participants were recruited on Prolific, where they were redirected to a link to our interface. After accepting the informed consent, they were given a brief survey. This survey asked about their experience with logic puzzles generally (such as Sodoku or Nonograms), and with logic grid puzzles specifically. After the survey, they proceeded to the tutorial. The tutorial first introduced the user interface of our application and then provided a step-by-step example on how to solve a logic grid puzzle. The UI and example puzzle are shown in Figure 1. However, participants could skip directly to the application at any point after the UI introduction.

Each participant could play up to 4 puzzles, one from each solution and one from each solver loop count. The combination of the solver loop count and the solution was randomly selected for each participant. While playing the puzzle we recorded every action they took along with their progress in the puzzle. The list of all measures recorded is given in Table 1.

After finishing each puzzle, participants filled out a survey about it. To measure difficulty, we used the cognitive section of the Video Game Demand Scale (Bowman, Wasserman, and Banks 2018), and to measure enjoyment, we used the engagement section of the Game User Satisfaction Scale (GUESS) (Phan, Keebler, and Chaparro 2016). The order of the questions was randomized, and all questions were measured on a Likert scale from 1 to 7. To aggregate the data, Likert values were averaged for all questions within each scale.

Participants were paid $2.50 for participation in this study, regardless of the number of puzzles they completed. The protocol of this study was approved by the IRB of the authors' host university.

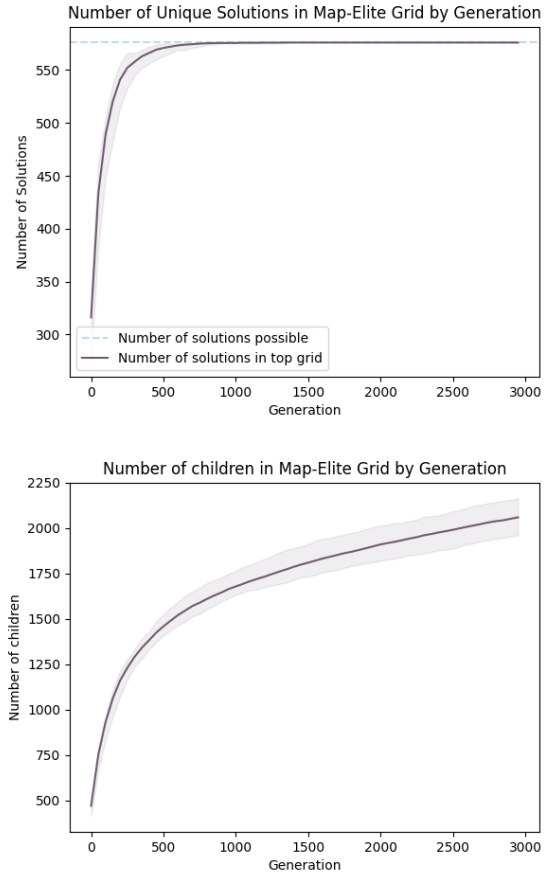


Figure 3: Total number of feasible puzzles in grid (top) and number of unique solutions (bottom) by generation. Solid line is the average across trials and the shaded area is the min and max values.

Analysis

The statistical tests performed for each hypothesis are given below.

Hypothesis 1: Solver Loops and Difficulty To examine the relationship between algorithmic solver time and difficulty, we perform a correlation analysis of the solver loops against all five behavioral difficulty measures and subjective difficulty (as described in Table 1). We do not assume a linear relationship between solver loops and difficulty, so we employ Spearman correlation which is based on ranks.

Hypothesis 2: Player Experience

To examine how experience with different logic puzzles affects experience with a new puzzle type, we grouped participants into three groups based on the initial survey: novices who had no experience in logic puzzles, intermediate users who had experience in logic puzzles but not logic grid puzzles, and experts who had experience in logic grid puzzles. We then performed a two-way ANOVA with the participant group and solver loop of the puzzle as the independent

	Measure	Description
Behavioral Difficulty	Time spent	Total time spent on puzzle recorded in milliseconds and presented in minutes
	Number of attempts	The number of times the participant clicked the check solution button
	Percent of Error	The total number of errors in puzzles divided by the total number of marks on the puzzle grid at the end of the play session
	Percent incomplete	The number of empty or incorrect marks divided by the total number of marks on the puzzle grid at the end of the play session
	Correctness	1 if the solution was correct, 0 otherwise (negatively associated with difficulty)
Subjective Measures	Subjective Difficulty	Average of responses (in 1-7 Likert scale) to the cognitive section of the video game demand scale (with some questions being inverted)
	Subjective Enjoyment	Average of responses (in 1-7 Likert scale) to the engagement section of the Game User Satisfaction Scale (with some questions being inverted)

Table 1: A description of the data points collected from each playthrough of a puzzle.

factors and subjective difficulty as the dependent variable. Tukey tests were used as the post-hoc tests.

Hypothesis 3: Relationship between Difficulty and Enjoyment

We predicted that there is a quadratic relationship between difficulty and enjoyment, where enjoyment peaks at medium difficulty (an "inverted U" shape), as has been demonstrated in some previous work (discussed in **Related Work**). To test this, we perform quadratic regression between subjective enjoyment and subjective difficulty. We use the r-squared value as a measure of goodness of fit. The r-squared value ranges between 0 and 1 and measures how much variation in the Y (enjoyment) variable can be explained by the X variable (subjective difficulty), with higher scores being better.

User Study Results

Participants There were 65 participants who attempted the study, with 1 participant returning the survey without completion. Out of the 64 participants we recorded data from, we removed 1 from analysis due to suspicious activity (large number of puzzles played with identical answers to survey questions). Out of the remaining 63 participants, 19 (29%) had experience with logic grid puzzles ("experts"), 27 (42%) had experience with logic puzzles but not logic grid puzzles ("intermediates"), and 17 (26%) had no experience with logic puzzles ("novices"). The majority (50) of participants played only one puzzle, with 8 participants playing 2 puzzles, 3 participants playing 3 puzzles, and 2 participants playing the max number of 4 puzzles. They spent a median time of 5.6 minutes on each puzzle, and Prolific reported a median time of 10 minutes and 8 seconds in total. This resulted in a median hourly rate of $14.79/hr. All data collected, along with the puzzles, is available on Open Science Framework[1].

[1]https://osf.io/kg4qs/

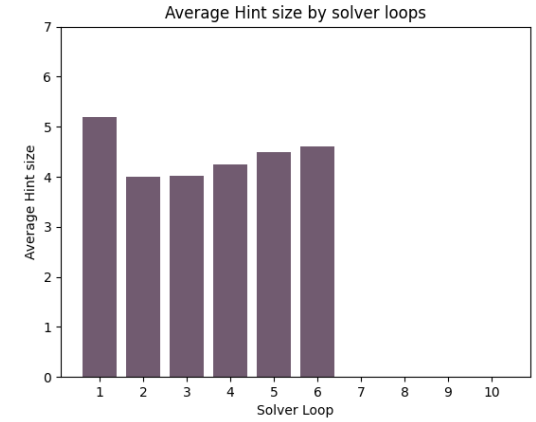
Measure	Correlation	p-value
Time Spent	0.174	0.139
Number of Attempts	-0.041	0.715
Percent Error	0.126	0.283
Percent Incomplete	0.162	0.167
Correctness *	-0.217	0.063
Subjective Difficulty **	0.301	0.015

Table 2: Correlation between solver loops and difficulty measures. The '*' mark denotes p-values under 0.1 and the '**' mark denotes p-values under 0.05.

Hypothesis 1: Solver Loops and Difficulty Hypothesis 1 is partially supported. The results of the Spearman correlation are shown in Table 2. Only the correlation between solver loops and subjective difficulty was statistically significant ($c = 0.30$, $p = 0.015$); however, the correlation between solver loops and correctness was borderline statistically significant ($c = -0.22$, $p = 0.063$).

A violin plot of correctness and subjective difficulty by solver loops is shown in Figure 6. Across all solver loops subjective difficulty was high ($m = 5.41$, $sd = 1.04$), with a little under half of the puzzle solutions being correct ($m = 0.43$, $sd = 0.50$). The mean subjective difficulty for 1 solver loop was 5.03 ($sd = 1.02$), and the mean subjective difficulty for 6 solver loops was 5.82 ($sd = 1.04$). The majority (60%) of puzzles with 1 solver loop were correct, whereas only 20% of puzzles with 6 solver loops were correct.

Hypotheses 2 and 3 The statistical tests for both hypotheses 2 (see Table 3) and 3 (see Table 4) did not show any significance. Therefore these hypotheses were not supported.



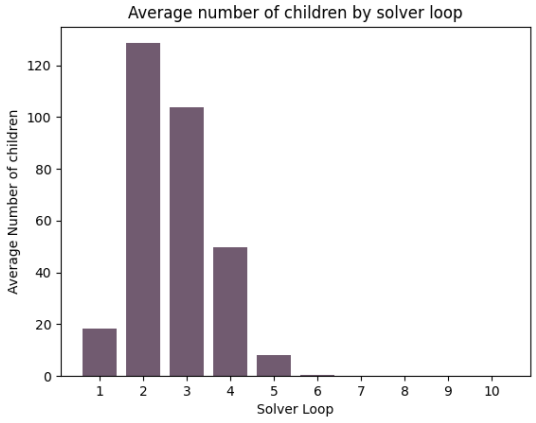


Figure 4: Average hint size of puzzles (top) and the total number of children produced throughout evolution (bottom) by the number of solver loops.

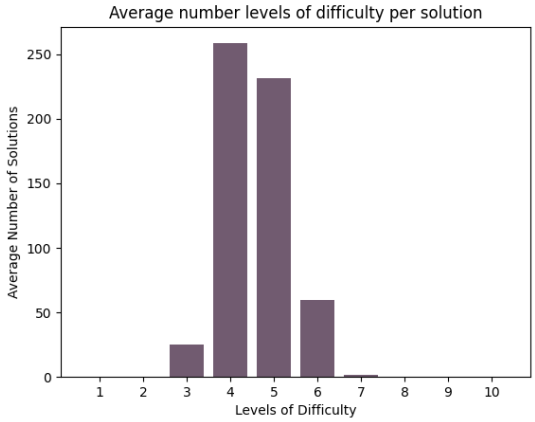


Figure 5: The average number of solutions in Map-Elites grid for each possible number of difficult levels (solver loops). Across all GA runs, each solution had at least three different puzzles at different solver loops sizes, with most having 4 or more.

Group	R-score
All participants	0.021
Novices	0.071
Intermediates	0.092
Experts	0.031

Table 3: R-squared for quadratic regression between subjective difficulty and enjoyment, by subject group. R-squared ranges between 0 and 1, with 1 meaning variation can be completely explained by the model. All R-squared values are too small to signify a quadratic relationship.

Discussion

Relationship of Algorithmic Solving Time to Player Experience

We found partial support for the relationship between solver loops and puzzle difficulty. However, the subjective difficulty was high across levels of solver loops. This could be indicative of the inherent difficulty of these types of puzzles or could signal that more fine-tuned measures might be needed for lower difficulties. There is also still a lot of variation of subjective difficulty within each solver loop level. This could be related to the variation in perceived difficulty between participants. Given that most participants played only one puzzle, we can not compare how one user might perceive puzzles with varying solver loops.

Solver loops were less related to behavioral measures of difficulty. While correctness was borderline statistically significant, no other measures were significantly correlated with solver loops. This could be because these measures are not a reliable predictor of puzzle difficulty. For example, a short playtime could mean the user found the puzzle to be simple, or that they found the puzzle too hard and gave up quickly. It is possible that combining these behavioral measures, such as time spent and correctness, could result in a more accurate measurement of difficulty. This could be examined in future work.

Future work could also explore more nuanced proxies of difficulty. For example, algorithmic difficulty could be used to build a predictive model for subjective or behavioral difficulty. Future work could also examine predictions for enjoyment along with difficulty. We did not find a relationship between enjoyment and difficulty, so perhaps other predictors could be used.

Generating Diverse Logic Puzzles

We created a modified version of constrained Map-Elites that was able to find a large number of diverse puzzles. Our algorithm was able to find a puzzle for every possible solution, with at least three levels of algorithmic difficulty. The number of puzzles in the Map-Elites grid had not yet converged at 3000 generations, suggesting that even more levels of algorithmic difficulty could be found given more time. Further, the average hint size was small (4 hints) which leads to more readable puzzles. This tool can allow designers to generate puzzles with varying difficulties while assuring that

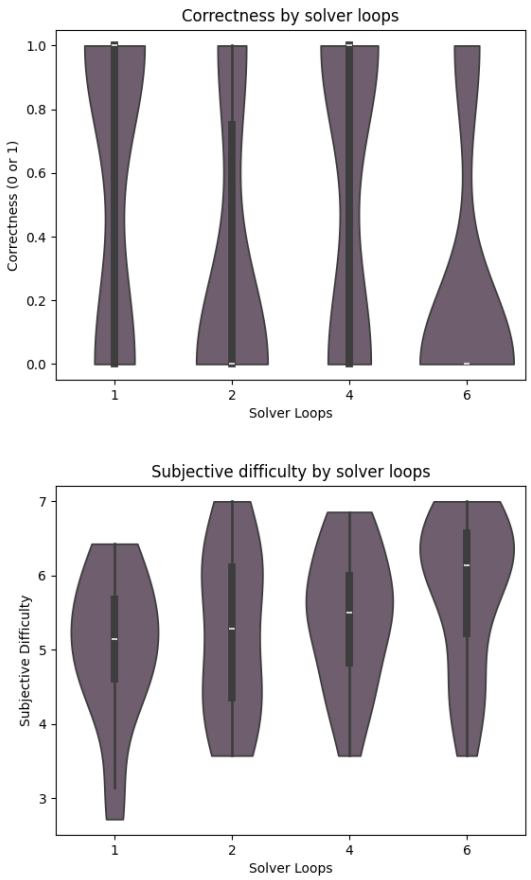


Figure 6: Violin plots of correctness and subjective difficulty by solver loops. The shaded areas show a kernel density estimate (KDE), and the black marks show a box and whisker plot.

players never see identical solutions.

Future work could expand this methodology to different puzzle types. Our system requires three properties of a puzzle game: a way to represent puzzles, the existence of multiple puzzles for the same solution, and a measurement of algorithmic solving time. This could be accomplished for a variety of puzzle games such as Sudoku or Sokoban.

Future work could also explore new mediums of gameplay using these logic grid puzzles. Logic grid puzzles are a flexible medium that could be expanded in new directions. They often already incorporate story elements, but this could be elaborated on. Perhaps murder mysteries or tabletop top roleplaying sessions could be designed using a logic grid puzzle as a basis. A generative system, such as ours, could be used to create narrative games with quality gameplay elements.

Conclusion

We developed an algorithm that was able to produce a diverse set of logic grid puzzles with a variety of difficulties. We generated puzzles for every possible solution, with at least three levels of difficulty per solution. Further, we tested our proxy for difficulty against the experience of human players and found that algorithmic difficulty (solver loops) significantly correlated with perceived difficulty. We hope this work can guide future efforts in puzzle generation.

Source of Variation	F value	p-value
User Group	1.61	0.21
Solver Loops	1.60	0.20
Interaction	1.00	0.43

Table 4: Two-way ANOVA results from user group by solver loops for subjective difficulty, including F-scores and p-values. The ANOVA did not show significant results.

Appendix

Example Puzzles

We provide four puzzles used in the user study. These puzzles all have the same solution but different numbers of solver loops: 1 loop in Figure 7, 2 loops in Figure 8, 4 loops in Figure 9, and 6 loops in Figure 10. We have included the version of each puzzle given to participants, including the light grammar editing described in **User Study**.

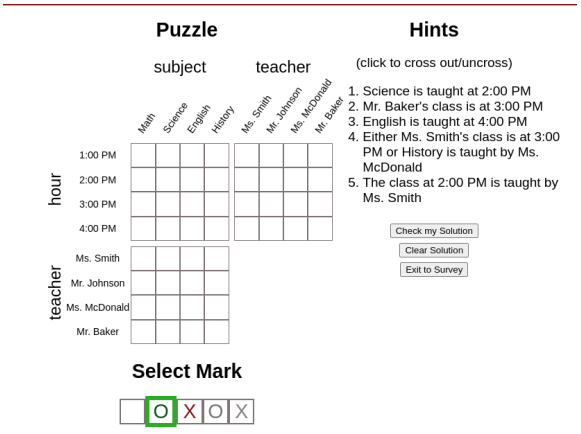


Figure 7: Puzzle from the user study with 1 solver loop. The solution is the same as in Figure 8, 9, and 10.

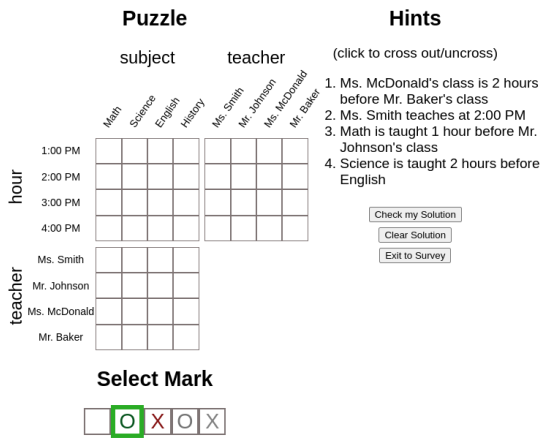


Figure 8: Puzzle from the user study with 2 solver loops. The solution is the same as in Figure 7, 9, and 10.

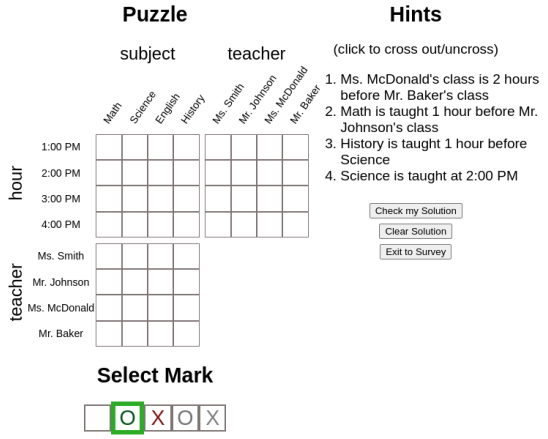


Figure 9: Puzzle from the user study with 4 solver loops.The solution is the same as in Figure 7, 8, and 10.

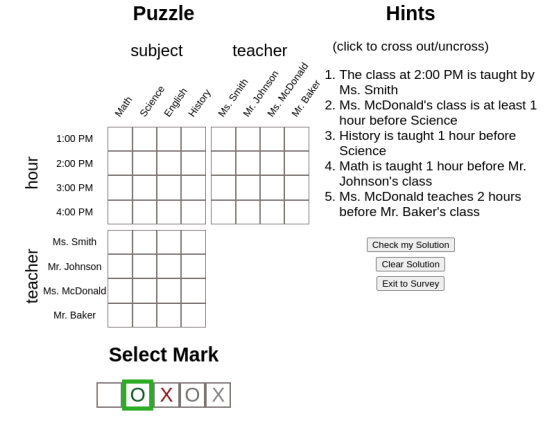


Figure 10: Puzzle from the user study with 6 solver loops. The solution is the same as in Figure 7, 8, and 9.

References

Abuhamdeh, S.; and Csikszentmihalyi, M. 2011. The Importance of Challenge for the Enjoyment of Intrinsically Motivated, Goal-Directed Activities. *Personality & social psychology bulletin*, 38: 317–30.

Batenburg, K. J.; Henstra, S.; Kosters, W. A.; and Palenstijn, W. J. 2009. Constructing simple nonograms of varying difficulty. *Pure Mathematics and Applications (Pu. MA)*, 20: 1–15.

Beukman, M.; Cleghorn, C. W.; and James, S. 2022. Procedural content generation using neuroevolution and novelty search for diverse video game levels. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '22. ACM.

Biemer, C.; and Cooper, S. 2024. Solution Path Heuristics for Predicting Difficulty and Enjoyment Ratings of Roguelike Level Segments. In *Proceedings of the 19th International Conference on the Foundations of Digital Games*, FDG '24. New York, NY, USA: Association for Computing Machinery.

Bonomo, D.; Lauf, A. P.; and Yampolskiy, R. 2015. A crossword puzzle generator using genetic algorithms with Wisdom of Artificial Crowds. In *2015 Computer Games: AI, Animation, Mobile, Multimedia, Educational and Serious Games (CGAMES)*, 44–49.

Bowman, N. D.; Wasserman, J.; and Banks, J. 2018. Development of the video game demand scale. In *Video games*, 208–233. Routledge.

Brändle, F.; Wu, C. M.; and Schulz, E. 2024. Leveling up fun: Learning progress, expectations, and success influence enjoyment in video games.

Chang, C.; Fan, Z.; and Sun, Y. 2007. A Difficulty Metric and Puzzle Generator for Sudoku. *UMAPJournal*, 305.

Chen, E. Y. C.; White, A.; and Sturtevant, N. R. 2023. Entropy as a Measure of Puzzle Difficulty. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 19(1): 34–42.

Colwell, A. M.; and Glavin, F. G. 2018. Colwell's Castle Defence: A Custom Game Using Dynamic Difficulty Adjustment to Increase Player Enjoyment. *arXiv preprint arXiv:1806.04471*.

Cutting, J.; Deterding, S.; Demediuk, S.; and Sephton, N. 2022. Difficulty-skill balance does not affect engagement and enjoyment: A pre-registered study using AI-controlled difficulty.

Dart, I.; and Nelson, M. J. 2012. Smart terrain causality chains for adventure-game puzzle generation. In *2012 IEEE Conference on Computational Intelligence and Games (CIG)*, 328–334.

De Kegel, B.; and Haahr, M. 2019. Towards Procedural Generation of Narrative Puzzles for Adventure Games. In Cardona-Rivera, R. E.; Sullivan, A.; and Young, R. M., eds., *Interactive Storytelling*, 241–249. Cham: Springer International Publishing. ISBN 978-3-030-33894-7.

De Kegel, B.; and Haahr, M. 2020. Procedural Puzzle Generation: A Survey. *IEEE Transactions on Games*, 12(1): 21–40.

de Pontes, R. G.; and Gomes, H. M. 2020. Evolutionary Procedural Content Generation for an Endless Platform Game. In *2020 19th Brazilian Symposium on Computer Games and Digital Entertainment (SBGames)*, 80–89.

Demediuk, S.; Tamassia, M.; Li, X.; and Raffe, W. L. 2019. Challenging ai: Evaluating the effect of mcts-driven dynamic difficulty adjustment on player enjoyment. In *Proceedings of the Australasian Computer Science Week Multiconference*, 1–7.

Dong, Y.; and Barnes, T. 2017. Evaluation of a template-based puzzle generator for an educational programming game. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*, 1–4.

Fernández-Vara, C.; and Thomson, A. 2012. Procedural Generation of Narrative Puzzles in Adventure Games: The Puzzle-Dice System. In *Proceedings of the The Third Workshop on Procedural Content Generation in Games*, PCG'12, 1–6. New York, NY, USA: Association for Computing Machinery. ISBN 9781450314473.

Ferreira, L.; and Toledo, C. 2014. Generating levels for physics-based puzzle games with estimation of distribution algorithms. In *Proceedings of the 11th Conference on Advances in Computer Entertainment Technology*, ACE '14. New York, NY, USA: Association for Computing Machinery. ISBN 9781450329453.

Gravina, D.; Khalifa, A.; Liapis, A.; Togelius, J.; and Yannakakis, G. N. 2019. Procedural Content Generation through Quality Diversity. In *2019 IEEE Conference on Games (CoG)*, 1–8.

Hald, A.; Hansen, J. S.; Kristensen, J.; and Burelli, P. 2020. Procedural Content Generation of Puzzle Games using Conditional Generative Adversarial Networks. In *Proceedings of the 15th International Conference on the Foundations of Digital Games*, FDG '20. New York, NY, USA: Association for Computing Machinery. ISBN 9781450388078.

Hoffmann, R.; Zhu, X.; Akgün, Ö.; and Nacenta, M. A. 2022. Understanding How People Approach Constraint Modelling and Solving. In *28th International Conference on Principles and Practice of Constraint Programming (CP 2022)*, 28:1–28:18. Association for Constraint Programming.

Hunt, M.; Pong, C.; and Tucker, G. 2007. Difficulty-driven sudoku puzzle generation. *The UMAP Journal*, 29(3): 343–362.

Jarusek, P.; and Pelánek, R. 2010. Difficulty Rating of Sokoban Puzzle. volume 222, 140–150.

Jarusek, P.; and Pelánek, R. 2011. What Determines Difficulty of Transport Puzzles?

Kartal, B.; Sohre, N.; and Guy, S. 2021. Data Driven Sokoban Puzzle Generation with Monte Carlo Tree Search. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 12(1): 58–64.

Khalifa, A.; and Fayek, M. 2015. Automatic puzzle level generation: A general approach using a description language. Institute of Electrical and Electronics Engineers.

Khalifa, A.; Lee, S.; Nealen, A.; and Togelius, J. 2018. Talakat: Bullet hell generation through constrained map-elites. In *Proceedings of The Genetic and Evolutionary Computation Conference*, 1047–1054.

Kraner, V.; Fister, I.; and Brezočnik, L. 2021. Procedural Content Generation of Custom Tower Defense Game Using Genetic Algorithms. In Karabegović, I., ed., *New Technologies, Development and Application IV*, 493–503. Cham: Springer International Publishing. ISBN 978-3-030-75275-0.

Kristensen, J. T.; and Burelli, P. 2024. Difficulty Modelling in Mobile Puzzle Games: An Empirical Study on Different Methods to Combine Player Analytics and Simulated Data. arXiv:2401.17436.

Lomas, J. D.; Koedinger, K.; Patel, N.; Shodhan, S.; Poonwala, N.; and Forlizzi, J. L. 2017. Is difficulty overrated? The effects of choice, novelty and suspense on intrinsic motivation in educational games. In *Proceedings of the 2017 CHI conference on human factors in computing systems*, 1028–1039.

Ma, Q.; Pei, G.; and Meng, L. 2017. Inverted U-Shaped Curvilinear Relationship between Challenge and One's Intrinsic Motivation: Evidence from Event-Related Potentials. *Frontiers in Neuroscience*, 11.

Maji, A. K.; Jana, S.; and Pal, R. K. 2016. *A Comprehensive Sudoku Instance Generator*, 215–233. New Delhi: Springer India. ISBN 978-81-322-2653-6.

Mantere, T.; and Koljonen, J. 2007. Solving, rating and generating Sudoku puzzles with GA. In *2007 IEEE Congress on Evolutionary Computation*, 1382–1389.

Meng, L.; Pei, G.; Zheng, J.; and Ma, Q. 2016. Close games versus blowouts: Optimal challenge reinforces one's intrinsic motivation to win. *International Journal of Psychophysiology*, 110: 102–108.

Mitsis, K.; Kalafatis, E.; Zarkogianni, K.; Mourkousis, G.; and Nikita, K. S. 2020. Procedural content generation based on a genetic algorithm in a serious game for obstructive sleep apnea. In *2020 IEEE Conference on Games (CoG)*, 694–697.

Moghadam, A. B.; and Rafsanjani, M. K. 2017. A genetic approach in procedural content generation for platformer games level creation. In *2017 2nd Conference on Swarm Intelligence and Evolutionary Computation (CSIEC)*, 141–146.

Naji, A.; and Salous, S. 2024. Sudoku Puzzle Difficulty Rating based on Fuzzy Logic. *Journal of Computer Science and Technology Studies*, 6: 86–91.

Oranchak, D. 2010. Evolutionary Algorithm for Generation of Entertaining Shinro Logic Puzzles. In Di Chio, C.; Cagnoni, S.; Cotta, C.; Ebner, M.; Ekárt, A.; Esparcia-Alcazar, A. I.; Goh, C.-K.; Merelo, J. J.; Neri, F.; Preuß, M.; Togelius, J.; and Yannakakis, G. N., eds., *Applications of Evolutionary Computation*, 181–190. Berlin, Heidelberg: Springer Berlin Heidelberg. ISBN 978-3-642-12239-2.

Pelánek, R. 2011. Difficulty rating of sudoku puzzles by a computational model. In *Twenty-Fourth International FLAIRS Conference*.

Phan, M. H.; Keebler, J. R.; and Chaparro, B. S. 2016. The development and validation of the game user experience satisfaction scale (GUESS). *Human factors*, 58(8): 1217–1247.

Priemer, D.; George, T.; Hahn, M.; Raesch, L.; and Zündorf, A. 2016. Using Graph Transformation for Puzzle Game Level Generation and Validation. In Echahed, R.; and Minas, M., eds., *Graph Transformation*, 223–235. Cham: Springer International Publishing. ISBN 978-3-319-40530-8.

Pusey, M.; Wong, K. W.; and Rappa, N. A. 2021. The Puzzle Challenge Analysis Tool. A Tool for Analysing the Cognitive Challenge Level of Puzzles in Video Games. *Proc. ACM Hum.-Comput. Interact.*, 5(CHI PLAY).

Scirea, M. 2020. Adaptive Puzzle Generation for Computational Thinking. In Fang, X., ed., *HCI in Games*, 471–485. Cham: Springer International Publishing. ISBN 978-3-030-50164-8.

Shyne, F.; Facey, K.; and Cooper, S. 2024. Poster: Generating Solvable and Difficult Logic Grid Puzzles. In *Genetic and Evolutionary Computation Conference (GECCO '24 Companion), July 14–18, 2024, Melbourne, VIC, Australia*. Elsevier.

Smith, A. M.; Andersen, E.; Mateas, M.; and Popović, Z. 2012. A case study of expressively constrainable level design automation tools for a puzzle game. In *Proceedings of the International Conference on the Foundations of Digital Games*, FDG '12, 156–163. New York, NY, USA: Association for Computing Machinery. ISBN 9781450313339.

Soares de Lima, E.; Feijó, B.; and Furtado, A. 2019. Procedural Generation of Quests for Games Using Genetic Algorithms and Automated Planning.

Spierewka, Ł.; Szrajber, R.; and Szajerman, D. 2021. Procedural Level Generation with Difficulty Level Estimation for Puzzle Games. In Paszynski, M.; Kranzlmüller, D.; Krzhizhanovskaya, V. V.; Dongarra, J. J.; and Sloot, P. M., eds., *Computational Science – ICCS 2021*, 106–119. Cham: Springer International Publishing. ISBN 978-3-030-77977-1.

Sudha, V.; and Kalaiselvi, R. 2020. The Sudoku Puzzle Generator Using DNA Computing. In Kar, N.; Saha, A.; and Deb, S., eds., *Trends in Computational Intelligence, Security and Internet of Things*, 3–14. Cham: Springer International Publishing. ISBN 978-3-030-66763-4.

Togelius, J.; Yannakakis, G. N.; Stanley, K. O.; and Browne, C. 2011. Search-Based Procedural Content Generation: A Taxonomy and Survey. *IEEE Transactions on Computational Intelligence and AI in Games*, 3(3): 172–186.

Valls-Vargas, J.; Zhu, J.; and Ontañón, S. 2017. Graph grammar-based controllable generation of puzzles for a learning game about parallel programming. In *Proceedings of the 12th International Conference on the Foundations of Digital Games*, FDG '17. New York, NY, USA: Association for Computing Machinery. ISBN 9781450353199.

van Arkel, B.; Karavolos, D.; and Bouwer, A. 2015. Procedural generation of collaborative puzzle-platform game levels. In Bakkes, S.; and Nack, F., eds., *GAME-ON' 2015*, 87–93. Eurosis. ISBN 9789077381915. GAME ON' 2015 : 16th International Conference on Intelligent Games and Simulation, GAME ON' 2015 ; Conference date: 02-12-2015 Through 04-12-2015.

van de Kerkhof, M.; de Jong, T.; Parment, R.; Löffler, M.; Vaxman, A.; and van Kreveld, M. 2019. Design and automated generation of Japanese picture puzzles. In *Computer Graphics Forum*, volume 38, 343–353. Wiley Online Library.

van Kreveld, M.; Löffler, M.; and Mutser, P. 2015. Automated puzzle difficulty estimation. In *2015 IEEE Conference on Computational Intelligence and Games (CIG)*, 415–422.

Venco, F.; and Lanzi, P. L. 2021. An Agent-Based Approach for Procedural Puzzle Generation in Graph-Based Maps. In *2021 IEEE Conference on Games (CoG)*, 01–08.

Wang, H.; Wang, Y.-W.; and Sun, C.-T. 2012. Rating Logic Puzzle Difficulty Automatically in a Human Perspective. In *Proceedings of Nordic DiGRA 2012 Conference*. Tampere: DiGRA.

Williams-King, D.; Denzinger, J.; Aycock, J.; and Stephenson, B. 2021. The Gold Standard: Automatically Generating Puzzle Game Levels. *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, 8(1): 191–196.

Xu, C.; and Xu, W. 2009. The model and algorithm to estimate the difficulty levels of Sudoku puzzles. *J. Math. Res*, 11(2): 43–46.