MedCrave
Step into the World of Research
International Robotics & Automation Journal

Research Article

Optimal solutions for lightbot code hour

Abstract

Lightbot is an interesting game app that has strong educational purpose to guide kids at young age to learn programming skills. It has a free version named Lightbot Code Hour consisting of 20 levels. Although it is not too hard to solve the puzzles, finding the optimal solutions with the smallest number of blocks is non-trivial for some levels. In this paper, we provide a complete list of the block numbers for optimal solutions adding up to 174 blocks total as the smallest number, as well as the detailed analysis of all the non-trivial levels.

Keywords: lightbot, optimal solution, puzzles, spiral move

Volume 11 Issue 1 - 2025

Songming Hou, Yan Huang
[1]Program of Mathematics and Statistics and Center of Applied Physics Louisiana Tech University
USA
[2]Beijing and Engine Software Shuiqingmuhua Yuan Zhongguancun, China

Correspondence: Songming Hou, Program of Mathematics and Statistics and Center of Applied Physics Louisiana Tech University, Ruston, Louisiana, 71272, USA

Received: February 10, 2025 | **Published:** February 24, 2025

Introduction

Lightbot is an interesting game app that has strong educational purpose to introduce programming concepts like loops and subroutines to kids at young age. It has a free version named Lightbot Code Hour consisting of 20 levels. The goal is to light up all the lights using a robot by giving it fixed instructions. It has been studied in.[1] However, the optimal solutions are not discussed.

When looking for the optimal solutions for Lightbot levels, the idea of periodicity comes into play which is shared by another game named the Rubik's Snake invented over 40 years ago.[2] The applications of the Rubik's Snake include the study of protein folding[3] and for the construction of reconfigurable modular robots.[4–6] Some mathematical problems including periodicity concerning a Rubik's Snake have been studied.[7] In,[8] theorems about Möbius, palindromic and periodic Rubik's snakes were proved. Further, the study of shortest path of Rubik's Snake knots[9–13] also utilized periodicity and share the same scheme of optimization with this work. We borrow some ideas from the study of Rubik's snake to carefully study the optimal solutions for Lightbot Code Hours and provide a complete list of the block numbers for optimal solutions adding up to 174 blocks total as the smallest number, as well as the detailed analysis of all the non-trivial levels. The rest of the paper is organized as follows: in Section 2, we give a simple tutorial how to play Lightbot Code Hour. In Section 3, we give a complete list of number of blocks needed for the construction. In Section 4, we explain level 3-5 in details, which is the easiest non-trivial level. In Section 5,6 and 7, we explain level 2-3, 2-4 and 2-5. In Section 8, we explain levels 2-6, 3-3 and 3-6 together, as they share the same mathematical ideas. We conclude in Section 9.

A simple tutorial to how to play Lightbot Code Hour

Lightbot Code Hour can be downloaded for free from app stores on one's smart phone. After that, simply open the app and start with the very first level. From level 1-1 to 1-8, one only need to add icons representing forward, left turn, right turn, jump, and light to the main program. The blue spots are lights that are off initially. A robot is initially at certain position with certain orientation. It cannot go out of bound. It cannot jump to more than 1 level above or jump forward to the same level. It cannot go forward if the forward direction has uneven surface. If the robot is on a blue spot and the light command is used, then the light will be switched from off to on or from on to off. Yellow color rep- resents that a light is on. When all lights are on, the puzzle is solved and the robot stops even if there are more commands not processed. Since we only have a main program and the only way is to step-by-step follow the commands, the solution is obvious and can be easily optimized to have the smallest number of blocks.

From level 2-1 to 2-6, procedures (subroutines) are introduced. They are labeled as P1 and P2. The main program can call them and they can call each other or even oneself. The name for level 3-1 to 3-6 is "loops". However, there is no clear-cut compared with level 2-1 to 2-6 except that the developer only allowed one block in the main program for Stage 3 levels. We assume the original idea of the developer might be that part of Stage 2 levels do not need a loop. After we found the optimal solutions, they indeed all need a loop. There could be more than one block needed in the main program though for Stage 2 levels.

A complete list of the number of blocks needed for optimal solutions

In this section, we provide a complete list of the number of blocks needed for optimal solutions. The might be helpful for readers who do not know if there is room to improve their own solutions.

For Stage 1 levels, the number of blocks are 3+9+6+8+11+12+12+12=73.

For Stage 2 levels, the number of blocks are 7+7+10+14+10+11=59.

For Stage 3 levels, the number of blocks are 4+4+8+7+10+9=42.

Overall, 73+59+42=174 is the optimal solution. Figure 1 shows a screen capture proving that we have reached this solution. To our knowledge, this was not known in the literature. Beginners might find it very difficult to get within 200 blocks.

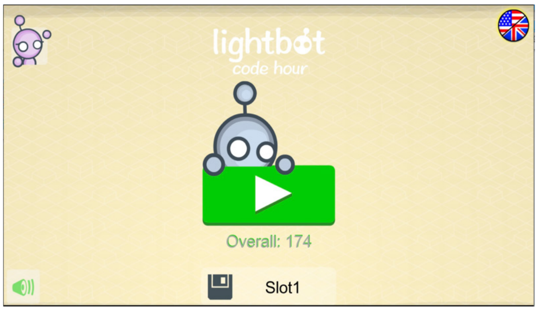

Figure 1 The overall optimal solution.

Level 3-5

Level 1-1 to 1-8 are all trivial to get optimal solutions. There are several other trivial levels as well. The reason why we start with a detailed explanation for level 3-5 is that it is the easiest among the non-trivial levels. Figure 2 is a screen shot of level 3-5 and its optimal solution with 10 blocks. It is quite obvious that to light up all lights, we could utilize a period 4 pattern here. Periodicity has been studied in the past in another game the Rubik's Snake.[8,14] A first attempt is to make a string of instructions "light, jump, light, forward, light, forward, light, jump, left" and make them in procedure 1 and then let procedure 1 call itself to form a loop. However, this would cost too many blocks. We try to find a way to use P1 to call P2 a few times in which a common pattern is held. We borrow the idea of smallest common multiple in number theory. The "smallest common sequence" is "forward, light, jump". We make this P2. Note that the robot would do nothing in a flat surface when a jump is the current command and would do nothing with an un- even surface in the current orientation when forward is the command.

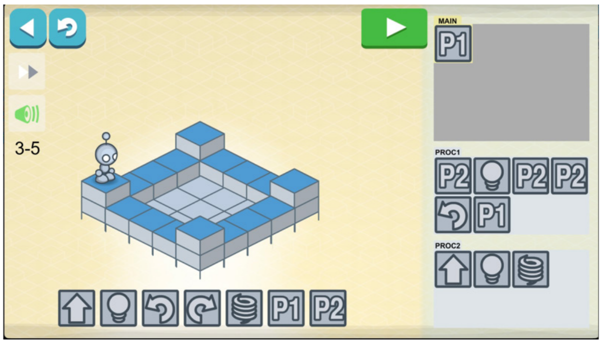

Figure 2 Level 3-5.

If we only repeat the above pattern, hoping to cover one edge then make a left turn, there is a problem of missing one light. To fix the problem, one must add a light command, making it "P2, light, P2, P2, left, P1" in procedure 1 as one period. Repeating 4 times we have it solved. We used a computer code to verify that it cannot be improved.

Level 2-3

An initial idea is to directly circle around while lighting up all the 3 lights. However, this is not feasible as there are limited number of blocks in each period. We used a computer code to exhaust all the solutions with 7 or less blocks in a period starting with the initial position and orientation with no result. The trick is to first make a left turn, then enter a loop. By making a spiral move, we solved the problem with only 10 blocks. We verified using a computer code that this cannot be improved. Figure 3 shows the optimal solution.

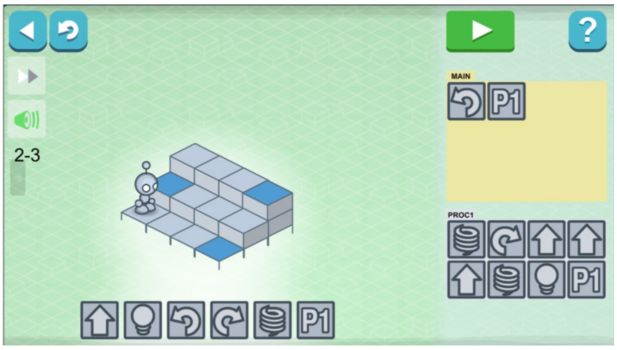

Figure 3 Level 2-3.

Level 2-4

The challenge here is changing directions. In a loop, the robot can only keep being clockwise or counterclockwise. The trick is to first finish the initial part, then enter a loop. The optimal solution has 14 blocks. Due to the map structure, there is no room to further improve, as jumping is meaningless and no alternative path is available unless making meaningless moves backward. Figure 4 shows the optimal solution.

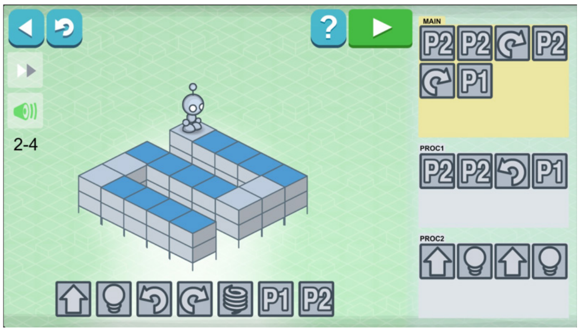

Figure 4 Level 2-4.

Level 2-5

This is a tough level to optimize. We look for a pattern so that after several periods we are at the next stair. It turns out that by adding some unnecessary turns and forward commands, this can be achieved. We came up with a beautiful optimal solution with only 10 blocks. This might be the biggest surprise in the entire game. In the first attempt people might not even be able to solve the problem with less than 20 blocks, and might get stuck with 19 blocks for a long time. Figure 5 shows the optimal solution.

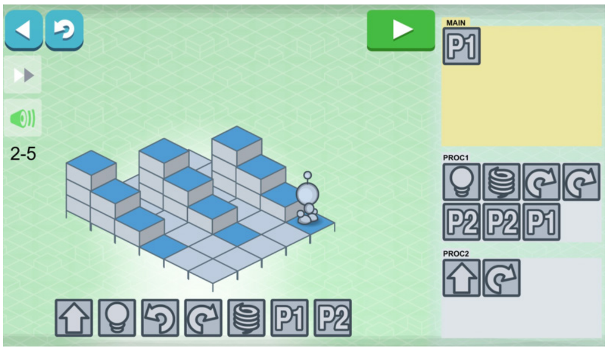

Figure 5 Level 2-5.

Level 2-6, 3-3 and 3-6

The beauty of symmetry in mathematics comes into play in these three levels. We are motivated by Figure 10 in[15] that a square setup can have a complicated periodic 4 path.

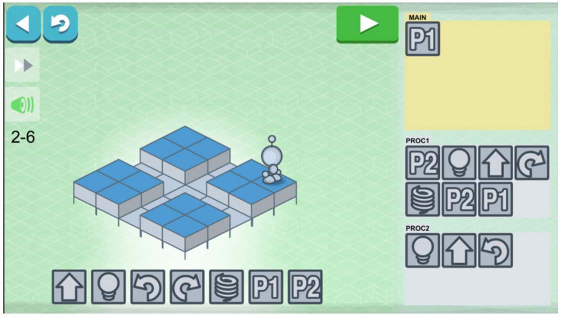

Figure 6 Level 2-6.

Each turn rotates 90 degrees. If the net gain in each period is 90 degrees, we can imagine that with period 4 there is a chance that we are back to where we started and exhausted the light positions. Also, if the period is 3*4 or 5*4, after 3 or 5 periods the net gain is plus or minus 90 degrees, and we could be back to the original place after 12 or 20 periods. Note that rotation cannot be blocked but jumping and going forward might not really get executed: it cannot go out of bound and it cannot jump to more than 1 level above, cannot jump to the same level, cannot go forward if the level is not flat in front. These limitations will make the path for each period have different shapes. The only invariance is the net rotation angle of each period must be the same. This is the math behind the design. Below we use 12 periods for 2-6 and 3-6, and 20 periods for 3-3. However, the last period does not need to finish as the goal is to light up all the lights.

Also, in 2-6 and 3-6, we used a trick that in procedure 1 (the loop), we call procedure 2 multiple times to save blocks. In other words, we try to have one period having some repeating patterns in it that we can utilize procedure 2 (although for 3-6, we could also simply call procedure 1 and perform a light command, jump 3 times, forward, left, forward, then call procedure 1 for a loop without using procedure 2, which travels the same path and ties for the optimal solution with 9 blocks).

It is also worth mentioning that we should not only exhaust the light positions but also make sure for each light, we apply the light command on it odd number of times, as it is turned off after even number of times. For easier levels, we could simply apply the light command once for each light. In general, as the goal is to get optimal number of blocks, we do not limit our choice to once per light. In the optimal solution for level 2-6, there are three lights that were on and off then on again. Intuition seems to tell us the number of such special lights should be a multiple of 4 for this level due to symmetry. However, keep in mind that the game is over when all the lights are on and we are not finishing the 12th period, making it 3 instead 4 such special lights. They are labeled as "3,6,9" in Figure 7.

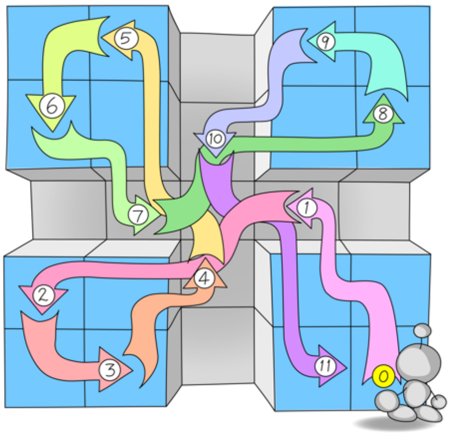

Figure 7 Level 2-6 explained.

By using a computer code, we verified that these are indeed the optimal solutions. Figures 6,8,10 show the optimal solutions. Figures 7,9,11 give detailed explanation of the path using symmetry. Each period is clearly labeled, helping readers to keep track.

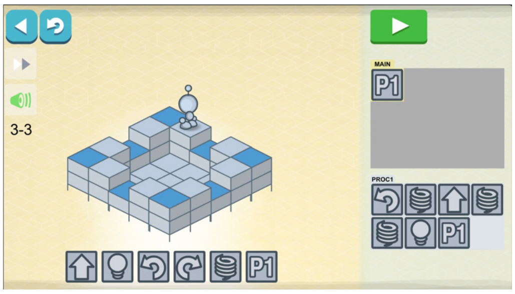

Figure 8 Level 3-3.

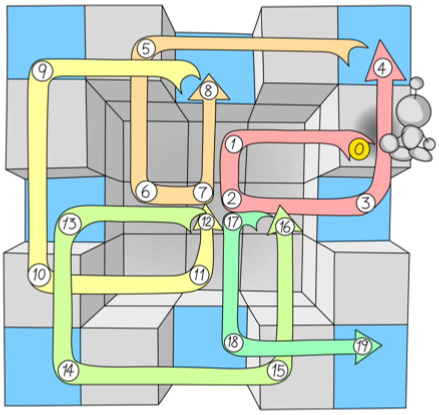

Figure 9 Level 3-3 explained.

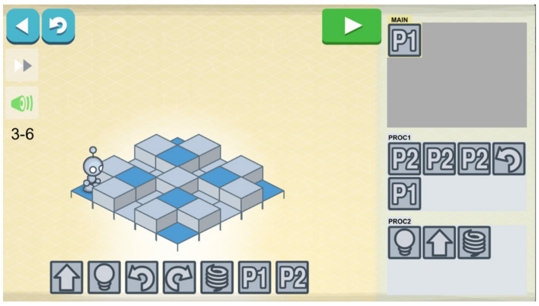

Figure 10 Level 3-6.

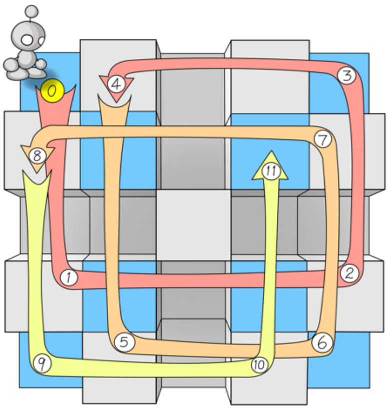

Figure 11 Level 3-6 explained.

Conclusion

Although Lightbot Code Hours is a game for kids to learn programming skills, finding optimal solutions is non-trivial because the goal of using the smallest number of blocks is different from the goal of finding the shortest path. The path could be complicated. We gave the detailed optimal solutions to the non-trivial levels of Lightbot Code Hours. Some mathematical explanations were provided to reveal the beauty of symmetry. We also included the complete breakdown of how to get the overall optimal result of 174 blocks.

Acknowledgment

We thank Ren Liu's group and Sihan Yang for some interesting discussions. S. Hou's research is partially supported by the Walter Koss Endowed Professorship. The title of the professorship is made available through the State of Louisiana Board of Regents Support Funds.

Conflicts of interest

Authors declare that there is no conflict of interest.

References

1. Gil A, Figueira T, Netto JFM. "Extracting learning analytics from lightbot gameplay sessions". XXIII Simpo´sio Brasileiro de Jogos e Entretenimento Digital (SBGames 2024) - Manaus/AM. 2024.
2. Fenyvesi C. "Rubik's snake of 'infinite possibilities"'. The Washington Post. 1981.
3. Iguchi K. A toy model for understanding the conceptual framework of protein folding: Rubik's magic snake model". *Mod Phys Lett B.* 1998;12(13):499–506.
4. Ding X, Lu S, Yang Y. "Configuration transformation theory from a chain-type reconfigurable modular mechanism-rubik's snake". The 13th World Congress in Mechanism and Machine Science. 2011.
5. Zhang X, Liu J. Prototype design of a rubik snake robot. *Mechanisms and Machine Science*. 2016;36.
6. Liu J, Zhang X, Zhang K, et al. "Configuration analysis of a reconfigurable rubik's snake robot*". Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science.* 2019;233(9):3137–3154.
7. Hou S, Chen Y, Li Z. "Some mathematical problems related to the rubik's snake". *J Mech Rob*. 2021;13(1):014502.
8. Hou S, Su J, Chen Y. "Palindromic, periodic and Möbius Rubik's snakes". *International Robotics and Automation Journal.* 2021;7(3):84–88.
9. Hou S, Su J. Shortest paths of trefoil knot designs using rubik's snakes. *International Robotics and Automation Journal*. 2022;8(1):18–20.
10. Hou S, Su J. "Shortest paths of rubik's snake prime knots up to 5 crossings". *International Robotics and Automation Journal.* 2022;8(2):47–50.
11. Hou S, Su J, Mufutau R. "Shortest paths of rubik's snake prime knots with up to 6 crossings and application to roller coaster design". *International Robotics and Automation Journal*. 2023;9(1):30–33.
12. Hou S, Su J, Mufutau R. Shortest paths of rubik's snake composite knots up to 8 crossings". *International Robotics and Automation Journal.* 2023;9(3):110–113.
13. Hou S, Su J, Mufutau R. "Shortest paths of rubik's snake composite knots with 9 crossings*". International Robotics and Automation Journal.* 2024;10(1):25–30.
14. Hou S. "Designing paths for box shapes using a rubik's snake". *International Robotics and Automation Journal*. 2022;8(2):66–68.
15. Hou S, Su J. "Torus knot designs using a rubik's snake". *International Robotics and Automation Journal*. 2024;10(2):43–46.

Citation: Hou S, Huang Y. Optimal solutions for lightbot code hour. *Int Rob Auto J*. 2025;11(1):8–11. DOI: 10.15406/iratj.2025.11.00293