A Simple and Rapid Lights-up Solver

Article · November 2010

DOI: 10.1109/TAAI.2010.75

Shi-Jim Yen
National Dong Hwa University

Cheng-Wei Chou
NCSIST

A simple and rapid Lights-up solver

Shi-Jim Yen[1], Shih-Yuan Chiu[2]

[1P]Department of Computer Science and Information Engineering, National Dong Hwa University, Hualien, Taiwan. sjyen@mail.ndhu.edu.tw
[2P]Department of Computer Science and Information Engineering, National Dong Hwa University, Hualien, Taiwan. d9621005@ems.ndhu.edu.tw

Abstract

This paper proposed a solver mixing pattern matching method with local search to solve lights-up puzzle games. First, use the rule of thumb to deal with some part of the region in a very short time. Then, use the local search method to complement the part in which the former method is not able to solve. After authenticating our method by solving the problems collected from Internet and comparing with other papers, the conclusion demonstrates that our approach is fast and efficient.

Keywords: Puzzle Game, Lights-up, Local search

1. Introduction

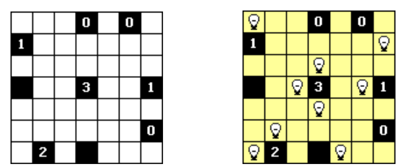
Lights-up is a puzzle game of logical-type which consists of problems in the form of squares in black and white of *N* by *M*. The objective is to place light bulbs on some white squares and light up all the white squares. There are three rules in this game:

1. The light bulbs can only be placed on the white squares to light up all the white squares in the same column and row. However, if there are any black squares blocking up the track of the light, there will be no light passing through the black squares.

2. The number in a black square means the number of bulbs should surround (in the positions of up, down, left, and right) the black square. It is wrong to have a bigger or smaller number than the one in the black square. If there is no number in the black square, there will be no implication for the number of bulbs surrounding the black square.

3. There shouldn't be any light bulbs placed on the squares lightened by other light bulbs.

Figure 1(a) is an example of a lights-up problem, and Figure 1(b) is the answer to this problem:



(a) (b)

Figure 1: An example of a lights-up problem (a) and its answer (b)

Emilio G once suggested an evolutionary computation by mixing Genetic Algorithm and Hopfield Network to produce a lights-up solver [1].

Although the method suggested by Emilio G can really solve problems of lights-up, we believe the rule of thumb can solve problems of certain region faster than the evolutionary computation. Though simply using the rule of thumb cannot solve the whole problem, with the method of local search to solve the rest of the problems, the whole problem can completely be solved. In the result of the following experiment, we can show that our approach is more efficient.

2. Our Solver

Our solver is divided into two parts by using four types of pattern matching to firstly deal with parts of the region quickly. Then, use local search to solve the parts, divided into several sub-goals, that matching pattern cannot deal with.

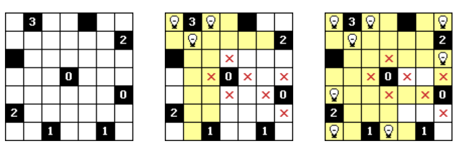
A. pattern match

Matching Patterns can be divided into four types of patterns:

Pattern I：Complete the black squares which are able to be solved. Make sure if the $Space_i$+$Have_i$ of the black square i($i \in \{1,\ldots,N_b\}$, N_b means the total number of the black squares with numbers) equals to the $Need_i$. If it does, SpP_i will all be placed with bulbs

$Space_i$ refers to the number of white squares, which is able to be placed with bulbs, surrounding the black square *i*. $Have_i$ and $Need_i$ means the number of bulbs that surround black square *i* now, and the total number of bulbs that should surround black square *i*. SpP_i refers to all the positions, which are able to be placed with light bulbs, surrounding black square *i*.

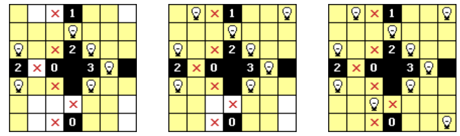
If $Have_i$ and $Need_i$ are the same, all the positions in SpP_i will be marked with x. Those squares marked with x cannot be placed with light bulbs. In figure 2(a), 2(b), and 2(c), a result of every step tested by pattern I two times is presented.



(a) (b) (c)

Figure 2: The first test's result by pattern I is from (a) to (b); the second one is from (b) to (c).

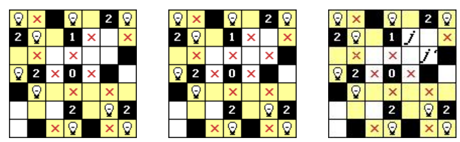
Pattern II：If $Source_j$ of some white square *j* ($j \in \{1,\ldots,N_w\}$, N_w refers to the amount of white squares) without being lightened (whether it is marked with x or not) equals to 1, the position of SoP_j can be placed with light bulbs. $Source_j$ and SoP_j individually refer to the source number and the position of which can light up the white square *j*. In figure 3(a), 3(b), and 3(c) show the result of each process tested twice by pattern II



(a) (b) (c)

Figure 3: A result of first test by pattern II is from (a) to (b); the second one is from (b) to (c).

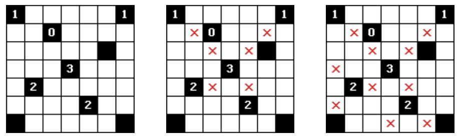
Pattern III：*j'* should be reckoned as x if $Source_j$ of white square *j* marked with x is 2, and corresponds to that (a): two positions in SoP_j are not on the same column and row, and (b) there are no black squares between *j'* and SoP_j , and the position of *j'* is a white square. *j'* can form a coordinates in the shape of a square with two positions in SoP_j and *j*. In figure 4, what (a) shows is a result before being tested by pattern III, while (b) is the one after being tested by it; (c) refers to the positions *j* and *j'* represent.



(a) (b) (c)

Figure 4: A result tested once by pattern III is from (a) to (b), and (c) refers to the positions of *j* and *j'*.

Pattern IV：The position *i'* connecting SpP_{ia} and SpP_{ib} can be reckoned as x if the $Space_i$ of some black square *i* with a number equals to $Need_i$ - $Have_i$+1. SpP_{ia} and SpP_{ib} of SpP_i refer to any two positions not on the same column and row. *i'* does not equal to *i*, and can form a square with SpP_{ia}, SpP_{ib}, and *i*. In figure 5, results tested twice by pattern IV of each process is clear.



(a) (b) (c)

Figure 5: A result tested once by pattern IV is from (a) to (b); the second result is from (b) to (c).

B. local search

These four matching patterns mentioned above can merely deal with parts of the region, and we cannot always obtain more answers in most of the region through the four patterns, as in the example of figure 6. In order to obtain a complete answer, we propose a method of local search to solve the remaining unknown region. We reckon every white square, without being neither lightened nor marked with x, as a small problem. Every goal of each small problem is to prove that the white square cannot be placed with light bulbs. If we can prove that some white square cannot be placed with light bulbs, we can reversely prove that the square should be marked with x. Here are some advantages of using local search:

(a) No need to deeply do a whole search.

(b) No need to check whether it follows the rule or not in every searching step. Except for the hypothesis in the first step, the result of the rest of the step is attributed to pattern match. Thus, examining the correctness after the last step can suppose whether the hypothesis of the first step is right or not.

(c) No need to do backtracking

Not every test of each white square can be proved. But, the region can be solved will become more and more until the problem is complete solved if some region can be proved reversely and then we can scan the whole problem by using the matching pattern.

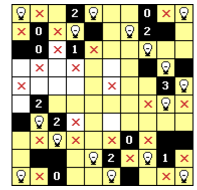


Figure 6: The result in which these four patterns can no longer offer more answers

Local search approach as follows：

I. Doing the following test on each white squares w_i ($i \in \{1, \ldots, k\}$ k means the total number of the white squares that are neither illuminated by light bulbs nor marked with x) on the board (the partly solved problems):

   a. w_i will be the assumed position with which the bulb is placed. Then, after placing the bulb, four matching pattern methods will be used to give further answers until these patterns cannot be able to solve any more regions.

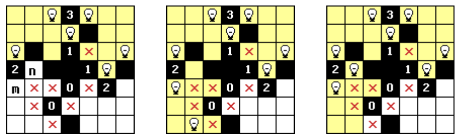
   b. Examine whether there are any errors in the result of a, and then revert the board to the state before step a.

   c. If there are any errors in b, w_i should be set as x.

II. Four matching pattern methods will be reused to update the board in step I.

III. Repeat step I and II until all the problems are solved.

Figure 7 is an example of how local search works. Assume m or n to be lights, and the result of using matching pattern to test is shown in figure 7(b) and 7(c). There are errors in the answer of figure 7(c), so the assumption of which n can be a light is not correct. Thus, we can make sure that the position of n should be marked with x. However, there is no error in figure 7(b), so we cannot have any reverse proof. The position m in this way cannot be reckoned as a light or x.



(a) (b) (c)

Figure 7: The example of assuming m and n to be bulbs in (a) can develop the result of (b) and (c).

3. Experiments

We choose 100 problems about lights-up whose squares are 25 by 25 with hard level from the Internet [2] to demonstrate the accuracy of our proposed approach. Our program is based on Matlab2009, AMD E8400 CPU, 8G RAM, and the operation system is VISTA64. Solving each problem costs 0.1377 seconds in average. Because Emilio G's paper can only solve problems of 14 by 14 without showing the time of calculation, we can just present our data as follows:

(a) The region which pattern match can deal with is 30.93% in average for each problem, and takes 0.00015 seconds in average. It takes 4.8497 e^{-5} seconds to solve each region as 1% in average.

(b) Each region which Local search is responsible for is 69.07% in average in every problem, and it takes 0.1356 seconds in average. The average time to solve each region as 1% takes $2e^{-3}$ seconds.

(c) The time to solve each problem is 0.1371 seconds in average.

4. Conclusion

In Chapter *2.B*, we have already articulated how local search is better than DFS and backtracking. In the data of Chapter 3, we can obviously observe that the speed of pattern match is 40 times faster than local search. Because, even though local search is a faster way, the speed of calculation is not faster than the rule of thumb, it proves that our policy is right to use the rule of thumb instead of using local search for all problems.

Reference

[1] Emilio G. Ortiz-Garcia, "A Hybrid Hopfield Network-Genetic Algorithm Approach for the Lights-up Puzzle," in *2007 IEEE Congress on Evolutionary Computation*, pp. 1403-1407, 2007.

[2] http://www.puzzle-light-up.com/