Contributions:

S. Salcedo-Sanz, L. Carro-Calvo, E.G. Ortiz-Garcia, A.M. Perez-Bellido, and J.A. Portilla-Figueras:
A Nested Two-steps Evolutionary Algorithm for the Light-up Puzzle.

J. Kloetzer, H. Iida, and B. Bouzy:
Playing Amazons Endgames.

Note:
C. Reams:
Playing Perfect Hangman.

S. Pellegrino, A. Hubbard, J. Galbraith, P. Drake, and Y-P. Chen:
Localizing Search in Monte-Carlo Go using Statistical Covariance.

Reports on:
The 2nd UEC Cup Computer-Go Tournament.
The Advances in Computer-Games Conference 2009.
The 13th Annual World Computer-Bridge Championship.
Livingston Chess960 Computer World Championship 2009.
Human vs. Computer-Go Competition.
ICGA 1

A Nested Two-steps Evolutionary Algorithm for the Light-up Puzzle

S. Salcedo-Sanz, 1 L. Carro-Calvo, E. G. Ortiz-García, Á. M. Pérez-Bellido and J. A. Portilla-Figueras


ABSTRACT

The Light-up puzzle is a logic grid-based puzzle really popular as pastime in the Internet. In this paper we present a nested two-steps evolutionary algorithm to solve it. The proposed approach exploits the constraints structure of the puzzle by running two different evolutionary algorithms in a nested fashion: the first evolutionary algorithm looks for feasible individuals in terms of a puzzle constraint, and only these feasible individuals are passed to a second evolutionary algorithm which completes the puzzle. The proposed algorithm is completed by introducing a preprocessing step based on constraint propagation, which improves its performance in the largest Light-up instances. The proposed approach is fully described in the paper, its main properties are detailed and its performance is shown in Light-up instances of different size, obtained from popular web pages devoted to this puzzle.

1. INTRODUCTION

Logic puzzles have been studied for years in Computer Science, Mathematics and Artificial Intelligence fields as important problems for testing different types of algorithms [1]-[12]. In the last few years, research on logic puzzles has increased a lot, since these puzzles are specially well-suited to test heuristic and meta-heuristics approaches [1], which has contributed to the development of this kind of algorithms. Specifically, many meta-heuristic approaches to solve logic puzzles are hybrid approaches, where a global search algorithm is combined with a local search procedure to improve the search. Some examples of this can be found in different works in the literature, focused on different grid-based puzzles [1, 8, 11, 12].

The Light-up puzzle [1, 13] is one of the puzzles that have received attention by the computer science community in the last few years. It is a logic puzzle that has gained popularity in part due to the possibility of on-line playing on the Internet. The rules of the Light-up puzzle are simple: First, it is played on an $N \times M$ rectangular grid. The grid has both black squares and white squares on it. The objective is to place light bulbs on white squares in the grid, so that every white square is illuminated. A square is considered as illuminated by a light bulb if they are in the same row or column, and there are no black squares between them. Also, no light bulb may illuminate another light bulb. Some of the black squares have numbers in them. A number in a black square indicates how many light bulbs share an edge with that square (are beside it). Figure 1 (a) shows an example of a Light-up grid, and Figure 1 (b) a solution for the puzzle. Mathematically, the Light-up puzzle can be defined as a 0-1 Integer Linear Programming problem.

Note that in this version of the puzzle, there is not a specific rule about the number of light bulbs which can be included in the puzzle. This is relevant, because different dispositions of light bulbs may produce a feasible solution to the puzzle (in its classical definition, there may be different feasible solutions for the puzzle). Related puzzles can be then obtained from this definition of the Light-up just by including constraints to the number of light bulbs to be included. One straightforward modification is for example to obtain the Light-up solution including the minimum possible number of light bulbs. Also, another possible related problem would be obtaining the feasible Light-up solution including the maximum possible of light bulbs. The inclusion of constraints in diagonals (not only in row and columns) is another possible modification of the puzzle which will produce a related puzzle.

1email: sancho.salcedo@uah.es
Figure 1: Example of a Light-up puzzle; (a) Light-up puzzle; (b) Light-up solution, grey squares stand for illuminated ones.

The Light-up puzzle (in its classical version) is challenging and addictive because of its unique constraint structure. This problem is related to other problems, such as the \textit{crossbar packet-switch} problem \cite{19}, which is equivalent to a Light-up puzzle without any black square constraints. In fact, it has been shown that the Light-up puzzle is NP-hard \cite{13}. The proof is based on reducing the Light-up puzzle to an equivalent Circuit-SAT problem, which is known to be NP-hard. As many other NP-hard grid puzzles, the Light-up puzzle is a very good test bed for the developing of efficient computational intelligence algorithms \cite{1, 12, 17}.

In this paper we propose a novel evolutionary-based algorithm for solving the Light-up puzzle. Evolutionary algorithms (EAs) \cite{14} are robust problem solving techniques based on natural evolution processes. Evolutionary algorithms are population-based techniques, where each \textit{individual} of the population is usually a string of numbers representing a potential solution to the problem by means of a given encoding (\textit{genome}). There exist many different encodings used in evolutionary algorithms. Binary (genetic algorithms), integer (evolutionary algorithms) or real encodings (evolution strategies) are the most common implementations, but many other possibilities exist, such as trees (Genetic Programming), strings of permutations, grouping representation etc. Specifically, in this paper we propose a nested two-steps evolutionary algorithm, where the first step is formed by an evolutionary algorithm devoted to finding partial solutions to the puzzle, feasible in terms of the second constraint of the puzzle (conditions in black squares of the puzzle). Those individuals in the evolutionary algorithm which are feasible in terms of this puzzle constraint are then passed to a second evolutionary approach, which looks for the final solution to the puzzle. This second evolutionary algorithm passes the control back to the first one in the case that no complete feasible solution for the puzzle is found, so a nested procedure is considered. In order to implement this evolutionary two-steps approach we have developed a novel encoding strategy for the puzzle, that we have shown to be very effective for solving the puzzle. We have tested the performance of the proposed approach in several Light-up puzzles which can be freely downloaded from the Internet, comparing the proposed approach with existing algorithms for the Light-up puzzle.

The rest of this paper is structured as follows: the next section presents the nested two-steps evolutionary-based algorithm proposed in this paper to solve the Light-up puzzle. Section 3 shows the performance of the proposed algorithm in several Light-up instances from the Internet. Section 4 concludes the paper with some final remarks.

2. THE PROPOSED NESTED TWO-STEPS EVOLUTIONARY ALGORITHM

This section describes the evolutionary algorithm for the Light-up puzzle proposed in this paper. First, an initial preprocessing step based on constraint propagation, which reduces the search space, is detailed. This preprocessing will be also used in the second step of the algorithm to improve the evolutionary algorithm’s performance. The encoding used in our approach and its structure are described next.

2.1 Preprocessing step: constraint propagation

Irrespective of the encoding and algorithms used to solve the puzzle, a preprocessing step can be used to fix light bulbs or white squares, reducing in this way the search space of the puzzle. For example, if there is a black square with a 0 condition in it, it means that the squares beside this condition must be white squares, without a light bulb in them. In a similar way, black squares with a 4 condition fix 4 light bulbs in the puzzle. In addition, the fact
that a given square has (or has not) a light bulb can affect other squares in the puzzle (constraint propagation).

This preprocessing is applied at the beginning of the algorithm, and then the squares which have not been set
in this process are encoded. As an example, Figure 2 shows an example of the preprocessing procedure and the
final encoding of the puzzle: Figure 2 (a) shows the puzzle, Figure 2 (b) shown the first step of the preprocessing
algorithm, where three white squares are fixed (“X” in the figure). Figure 2 (c) shows the second step of the
preprocessing, where two light bulbs are set in the grid, and the corresponding illuminated squares (in grey in the
figure) are also fixed as a consequence of the previous step (constraint propagation). Finally Figure 2 (d) shows
the unknown squares after the preprocessing procedure.

![Figure 2](image_url)

Figure 2: Example of the preprocessing carried out in a Light-up puzzle and the corresponding puzzle encoding after it;
(a) Initial Light-up puzzle; (b) First step of the preprocessing procedure; (c) Second step of the preprocessing procedure; (d) Squares after the preprocessing procedure; (e) Encoding of the not fixed squares in the puzzle; (f) Final solution of the puzzle.

2.2 Puzzle encoding

One of the problems of the binary encoding for the Light-up puzzle, just like the one used in [17], is that the
search space of the puzzle becomes huge. Consider the Light-up puzzle in Figure 3 (a). After applying the
preprocessing step, only 4 squares are fixed (the ones beside the black square with a 0 on it, Figure 3 (b)).

![Figure 3](image_url)

Figure 3: Example of the space size that a binary encoding generates; (a) Light-up puzzle; (b) Light-up after applying the
preprocessing step. A binary encoding in this puzzle involves a search space of $2^{43}$ possible individuals; (c) Encoding of the
reduction in search space size by only encoding the squares beside numbered black squares in the grid.

Thus, the binary encoding of this puzzle involves 43 squares, i.e., a search space of size $2^{43}$. The question
is whether an alternative encoding could reduce the search space size, and thus improve the search abilities of
an evolutionary algorithm. One idea for efficiently encoding the Light-up puzzle is to analyze the constraints’
structure of the problem. Note the following: there are specific numbers of possible light bulbs combinations
beside numbered black squares. There is just one possible combination for black squares with a 0, there are four
for black squares with a 1 on them, six for black squares with a 2, four for black squares with a 3 and finally one
for black squares with a 4. The proposal is then to separate the Light-up puzzle encoding in two parts, a first part
for squares beside numbered black squares, and a second part for the rest of the squares in the puzzle, as Figure
3 (c) shows. The first part (the one corresponding to squares beside numbered black squares) has the following structure:

$$[N_1, N_2, \ldots, N_{nbs}]$$  \quad (1)

where $N_j$ stands for the specific combination of light bulbs in the $j$-th numbered black square, and $nbs$ stands for the number of numbered black squares in the puzzle. For example, in Figure 3 (c), $N_1$ is a number between 1 and 4, since the first black square (ordered starting from the leftmost at the top of the grid) has a 3 on it, and so there are 4 possibilities to set 3 light bulbs in the squares beside it. On the other hand, $N_4$ is a number between 1 and 6, since the associated black square has a number 2 on it, and then there are 6 possibilities of setting light bulbs in the squares beside it.

The second part of the encoding, the one related to the squares that are not beside numbered black squares, is considered binary:

$$[b_1, b_2, \ldots, b_{os}]$$  \quad (2)

where an element $b_j$ is 1 if a light bulb is set in a square $j$, 0 if not. Here $os$ stands for the number of squares that are not beside numbered black squares.

### 2.3 Description of the proposed two-steps evolutionary algorithm

The idea of the nested two-steps evolutionary approach is the following: after a first application of the preprocessing step, a first evolutionary algorithm looks for light bulb configurations beside numbered black squares that are feasible (fulfill all black square constraints of the puzzle, and do not break any puzzle rules). Solutions that are feasible in this first part of the algorithm pass to a second step. Then the preprocessing step is applied again in order to further reduce the search space from the light bulbs set in the first step of the algorithm, and then a different evolution phase is carried out to set light bulbs in the remaining unknown squares (required to illuminate all the grid, and thus obtain the solution to the puzzle). Figure 4 shows an outline of the proposed evolutionary algorithm for the Light-up puzzle.

Figure 4: Scheme of the two-steps EA for the Light-up proposed in this paper. CP stands for Constraint Propagation.

Several points must be taken into account in order to implement this two-steps evolutionary algorithm: first, depending on the specific solution or puzzle, not all configurations of light bulbs in numbered black squares are valid. For example, consider Figure 5. The situations in Figure 5 (a) and (b) are valid configurations, but Figure 5 (c) and (d) show non-valid configurations. In Figure 5 (c) two light bulbs illuminate each other, and in Figure 5 (d) the black square at the bottom has two light bulbs beside, whereas only one is permitted due to the black square condition.
Figure 5: Example of a situation in which not all combinations beside a numbered black square are valid; (a) valid combination; (b) valid combination; (c) non-valid combination, since two light bulbs illuminate each other; (d) non-valid combination since there are two light bulbs beside a black square with a 1 as constraint.

With this in mind, the two-steps evolutionary algorithm for the Light-up puzzle follows the structure given in Figure 4. First, a population of \( \xi_1 \) individuals, encoded following Equation (1), is randomly obtained. This first evolutionary algorithm has a structure similar to a standard evolutionary algorithm [14], with operators of selection, crossover and mutation. The selection procedure is based on a roulette wheel procedure, as the standard implementation. The crossover operator is a standard two-points crossover. The mutation operator is in this case different from the standard one based on flip of bits: in this case each element \( N_j \) in the encoding is randomly changed by a different feasible value. Each individual is then analyzed looking for errors which make it unfeasible. Each error (light bulb illuminating another light bulb, or violation of a numbered black squares constraint) affect the objective function of the algorithm (which must be minimized). In fact, each error is penalized with a penalty value of \( P_v \) units (100 units in our implementation). Feasible individuals in terms of black squares constraints (no errors, 0 penalty values) are then passed to the second step, and note that light bulbs from the first step are not touched anymore in this second one. The preprocessing step is applied again at this point, in order to reduce the search space size for the second evolution. In this case, the starting point are the light bulbs set by the first evolutionary algorithm. After this, a new population of \( \xi_2 \) individuals with the encoding given by Equation (2) is randomly generated. In this second step, the evolutionary algorithm looks for setting light bulbs in such a way that all the squares in the grid are illuminated, without violating any puzzle constraint. The objective function to be minimized by the algorithm is defined in the following way: each square that is not illuminated is penalized with a penalty value of \( P_w \) (10 units in our implementation). Each violated puzzle constraint is penalized with \( P_x \) units in this second step (40 units in our implementation). If a solution for the puzzle is found in this second step, the algorithm is stopped. On the contrary, the objective value of the individual in this second step is passed back to the first step, assigned to the specific initial individual there, and the search is continued. As an example, Figures 6 (a) and (b) show two different configurations with 0 errors in the first step, so both solutions would be passed to the second step of the algorithm. Figure 6 shows the two best possible solutions found in this second step. Note that the first configuration coming from the first step (Figure 6 (a)), produces the final solution given in Figure 6 (c), with a fitness function of 20 units (10 for each not illuminated square). On the other hand, the second configuration from the first step (Figure 6 (b)), is able to produce a solution for the puzzle, given in Figure 6 (d).

Note that the proposed two-steps algorithm’s structure is similar to the constraint structure of the Light-up puzzle. In this case, a first evolution is used to obtain a feasible solution in terms of the second constraint of the puzzle, and only those individuals which are feasible in this sense, are passed to a different second evolutionary algorithm, which will look for setting more light bulbs to illuminate all the grid, without violating the first constraint.

3. EXPERIMENTS AND RESULTS

In order to show the performance of the proposed algorithm for the Light-up puzzle, we have solved a number of puzzles, of different sizes, obtained from [16] and [15]. The first 10 puzzles considered are small size ones (7 × 7), easy to solve. The hardest instances are 3 puzzles of size 25 × 25. We have also solved medium-size puzzles (10 × 10) and medium-large instances (14 × 14). The two-steps evolutionary-based algorithm proposed in this paper has been applied to solve all the instances puzzles. In all the instances, 200 runs of each algorithm have been launched, computing the percentage of convergence to a solution. The maximum number of function evaluations allowed is different in each puzzle, depending on its size. Note that this is only an indication of the maximum number of function evaluations, since the algorithm is stopped as soon as it obtains a feasible solution for the puzzle. A comparison with the best existing algorithm for the puzzle (in [17]) is carried out.

Table 1 shows the results obtained by the two compared algorithms. In this table we include the comparison in
terms of percentage of convergence in 200 runs of the algorithms, considering the compared approaches without
the preprocessing step described in Section 1, with the preprocessing step applied once, and also the results of
the proposed approach when the preprocessing step is applied twice (before the application of each evolutionary
algorithm).

First, note that both approaches can easily solve the smallest Light-up instances considered (7 × 7), obtaining a
solution in all the runs of the algorithms (we have not displayed the results of these instances in the table, since
all the algorithms tested always obtained a solution for these instances). In the 10 × 10 instances, the two-steps
genetic algorithm is able to solve all the puzzles (100% of convergence in the 200 runs) when it is run applying
the preprocessing step once. However, the HNN-GA in [17] obtains 100% of convergence in two instances, but
71% and 92% in the other two. The percentage of convergence in the case of not including the preprocessing
step is smaller in both algorithms. In the medium-large (14 × 14) and very large instances (25 × 25), the results
of both algorithms without including the preprocessing step, and including it once, are quite similar. Both are
able to solve the Light-up puzzle instances in this last case, though the percentage of convergence in the 200
runs is low in the largest Light-up instances, around the 10% of the runs found the correct solution to the puzzle.
The last column of Table 1 shows the performance of the proposed two-steps evolutionary algorithm when the
preprocessing step is applied twice (one at the beginning and one before the second step of the algorithm). Note
that the performance of the proposed approach is improved by the second application of the preprocessing step.
The improvement in the 14 × 14 puzzles is good, but in the largest puzzles 25 × 25 is even more significant as can
be seen in Table 1. Note that in the largest Light-up instances considered the inclusion of the preprocessing step
is crucial in order to improve the performance of both algorithms. The preprocessing step is a way of reducing
the search space size, so its inclusion makes that the algorithm must search in a reduced search space and then its
performance is much better than when the preprocessing approach is not included.

A comparison with an alternative solution method is also carried out in this experimental section. Specifically, the
Light-up puzzle has been previously solved using a translator from constraint satisfaction problems (CSP) to SAT
problems (SUGAR [20]), and solving the corresponding SAT using the Minisat solver [21]. This approach can be
applied to linear CSP, as the Light-up presented in this paper. The SUGAR+Minisat approach has been applied to
the Light-up puzzle instances considered in this paper by downloading the code of the algorithm, available in [22].
The SUGAR+Minisat approach is a robust approach for this linear version of the puzzle, and it is able to solve all the instances tackled in this section. Thus, it can be used as comparison algorithm. Note that this approach is deterministic, so no percentage of solution can be given using SUGAR+Minisat. The computation time of the proposed two-steps GA and the SUGAR+Minisat approach is shown in Table 2. This table shows the average computation time in all the instances of similar size. In the case of the SUGAR+Minisat, the computation time includes the translation from the puzzle to the SUGAR encoding (using a script of Matlab), the conversion from CSP to SAT using SUGAR, the resolution of the SAT problem with the Minisat code and finally the decoding of the problem from Minisat to SUGAR again. In the case of the two-steps genetic algorithm, the computation time is calculated including the first pre-processing step. Note that the proposed approach obtains the puzzle solution in times comparable to that of the SUGAR+Minisat approach, and it seems that the computation time is slightly smaller using the two-steps approach in the smallest Light-up instances. Anyway, the conclusion is that the proposed two-steps genetic algorithm is also a robust approach to solve the Light-up puzzle, comparable to very good existing solutions as the one in [22].

Table 1: Comparison of the HNN-GA and two-steps GA performance in the Light-up puzzles used as benchmark in this paper. Results on the algorithms without including the preprocessing step are also displayed. The results are given as percentage of convergence over 200 runs of the algorithm.

<table>
<thead>
<tr>
<th>Instance size</th>
<th>Max Fitness Evaluations</th>
<th>HNN-GA [17] (no preproc.)</th>
<th>two-steps EA (no preproc.)</th>
<th>HNN-GA [17] (1 preproc.)</th>
<th>two-steps EA (1 preproc.)</th>
<th>two-steps (2 preproc.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10 × 10</td>
<td>10000</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>10 × 10</td>
<td>10000</td>
<td>90</td>
<td>85</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>10 × 10</td>
<td>10000</td>
<td>60</td>
<td>80</td>
<td>71</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>10 × 10</td>
<td>10000</td>
<td>70</td>
<td>90</td>
<td>92</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>14 × 14</td>
<td>40000</td>
<td>40</td>
<td>35</td>
<td>96.5</td>
<td>97.5</td>
<td></td>
</tr>
<tr>
<td>14 × 14</td>
<td>40000</td>
<td>75</td>
<td>84</td>
<td>92</td>
<td>96.5</td>
<td></td>
</tr>
<tr>
<td>14 × 14</td>
<td>40000</td>
<td>80</td>
<td>65</td>
<td>99.5</td>
<td>96.5</td>
<td></td>
</tr>
<tr>
<td>14 × 14</td>
<td>40000</td>
<td>90</td>
<td>90</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>14 × 14</td>
<td>40000</td>
<td>3</td>
<td>5</td>
<td>14.5</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>14 × 14</td>
<td>40000</td>
<td>70</td>
<td>75</td>
<td>100</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>14 × 14</td>
<td>40000</td>
<td>55</td>
<td>60</td>
<td>90</td>
<td>95</td>
<td></td>
</tr>
<tr>
<td>25 × 25</td>
<td>60000</td>
<td>0</td>
<td>0</td>
<td>6</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>25 × 25</td>
<td>60000</td>
<td>2</td>
<td>3</td>
<td>19</td>
<td>20</td>
<td></td>
</tr>
<tr>
<td>25 × 25</td>
<td>60000</td>
<td>2</td>
<td>3</td>
<td>19</td>
<td>20</td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Average computation time (in all instances considered) of two-steps GA and its comparison with SUGAR+minisat approach [22].

<table>
<thead>
<tr>
<th>Instance size</th>
<th>two-steps GA</th>
<th>SUGAR+Minisat [22]</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 × 7</td>
<td>0.042</td>
<td>0.478</td>
</tr>
<tr>
<td>10 × 10</td>
<td>0.122</td>
<td>0.483</td>
</tr>
<tr>
<td>14 × 14</td>
<td>0.702</td>
<td>0.804</td>
</tr>
<tr>
<td>25 × 25</td>
<td>2.250</td>
<td>1.974</td>
</tr>
</tbody>
</table>

4. CONCLUSIONS

In this paper we have proposed a nested two-steps evolutionary-based algorithm for the Light-up puzzle (two-steps EA). The algorithm is based on dividing the search to obtain feasible solutions in terms of the different constraints of the puzzle. In the two-steps EA, a first evolutionary algorithm looks for feasible solutions in terms of the second constraint of the Light-up puzzle (number of light bulbs beside numbered black squares of the puzzle). The individuals that are feasible, are passed to a second evolutionary algorithm to obtain the final solution to the puzzle. We have discussed the effect of including a preprocessing step based on constraint
propagation before both evolutionary approaches. We have shown that this preprocessing step is able to improve the performance of the proposed approach because it reduces the search space of the evolutionary algorithm in both steps of the algorithm. The approach presented in this paper, including the preprocessing step, has shown to be very effective in solving the Light-up puzzle. Future work on this puzzle is possible. Alternative encoding strategies are one of the most interesting issues which can be studied using this puzzle. The application of new soft-computing algorithms for very large puzzles is another possible research line. Also related to this possibility, the study of specially difficult puzzles, where constraint propagation may fail, and the study of novel techniques for reducing the search space could be very interesting for researchers in evolutionary computation algorithms. Finally, this puzzle can be useful for education purposes, since it can be solved using very different methods.

ACKNOWLEDGEMENT

The authors wish to thank the anonymous reviewers of this paper for their hard work, excellent comments and suggestions, which have definitely improved the article. In this case, one of these suggestions by a reviewer resulted in a major improvement in the algorithm’s performance. We also appreciate a lot the excellent work and suggestions of the Editor-in-Chief, Professor H. J. van den Herik.

5. REFERENCES


