REFLECTIONS ON REFLEXION—COMPUTATIONAL COMPLEXITY CONSIDERATIONS ON A PUZZLE GAME

Markus Holzer
Institut für Informatik, Technische Universität München,
Boltzmannstraße 3, D-85748 Garching bei München, Germany
holzer@in.tum.de

Stefan Schwoon
Institut für Formale Methoden der Informatik, Universität Stuttgart,
Universitätsstraße 38, D-70569 Stuttgart, Germany
schwoosn@fmi.uni-stuttgart.de

Abstract

We investigate the complexity of solving puzzles in the game REFLEXION. It is shown that the difficulty of the puzzles depends critically on the features used in the puzzles; the most basic variant of the game is SL-complete, and hence can be solved in polynomial time, whereas other variants are NP- or even PSPACE-complete.

1. *Introduction*

REFLEXION is a solitaire game created by Juho Pohjonen.[1] The REFLEXION game takes place on a rectangular finite two-dimensional grid, the board. Every cell on an edge of the board is a wall; inner cells may contain walls, mirrors, or diamonds. Moreover, one cell is designated as the start, and another as the target. Mirrors can assume two diagonal orientations. The basic rules are as follows: The player may release the ball at the start. The ball moves parallel to the x-axis or the y-axis; once released, it keeps moving until the game ends. When the ball hits a diamond, the diamond is removed. When the ball hits a wall, it starts moving in the opposite direction. When the ball hits a mirror, it takes a right or a left turn depending on the orientation of the mirror. The player may change the orientation of mirrors while the ball is moving.

[1] Not to be confused with "Reflections", whose complexity was recently investigated in [6].

The object of the game is to reach the target after removing all the diamonds. In our figures, walls are represented by black squares, free space by white squares, diamonds by octagons, and the start and the target by S and E, respectively. Figure 1(a) shows an instance of a REFLEXION board.

In many levels of the game various extensions of the basic rules are used. Sometimes, for instance, the mirrors must be fixed before the ball is released. Sometimes the mirrors vanish when touched by the ball; other levels feature additional objects like movable walls or hostile creatures. Moreover, some levels contain trigger cells, which may open or close certain walls when the ball crosses them. An implementation of REFLEXION for Windows systems is available on the Internet and can be downloaded from `http://www.cs.uu.nl/people/markov/gmaker/games.html`.

In this paper, we investigate the complexity of solving REFLEXION puzzles for different variants of the rules, namely (i) Basic REFLEXION, (ii) REFLEXION with frozen mirrors, (iii) REFLEXION with vanishing mirrors, (iv) REFLEXION with dynamic walls, and (v) REFLEXION with movable boxes. Formally, we consider (for each variant) the following question: Given a REFLEXION board, can one win the game, i.e., remove all diamonds and then reach the target?

The paper is organized as follows: In Sections 3 through 7 we consider the five variants listed above, and show that the associated problems are SL-complete (for variant 1), NP-complete (for variants 2 and 3), and PSPACE-complete (for variants 4 and 5). Thus, the problem falls into quite different complexity classes depending on the exact set of rules. This distinguishes REFLEXION from a number of other solitaire games as, e.g., Atomix [5], RushHour [3], or Sokoban [2], for which even the most basic variant is already PSPACE-complete. In Section 8 we summarize our results and state an open problem.

2. *Definitions*

We assume familiarity with the basic concepts in complexity theory [7]. In particular we use the well-known chain of inclusions L $\subseteq$ SL $\subseteq$ NL $\subseteq$ P $\subseteq$ NP $\subseteq$ PSPACE.

Here L (NL, resp.) is the set of problems accepted by deterministic (nondeterministic, resp.) logarithmic-space bounded Turing machines, and SL is the set of languages accepted by symmetric-logspace bounded Turing machines. Roughly speaking, a Turing machine is called symmetric if and only if its transition function is reversible. A more detailed explanation of symmetry can be found in [1]. Moreover, P (NP, resp.) is the set of problems accepted by deterministic (nondeterministic, resp.) polynomial-time bounded Turing machines, and PSPACE is $\bigcup_k \mathrm{DSpace}(n^k) = \bigcup_k \mathrm{NSpace}(n^k)$ by Savitch's theorem [8]. Finally, completeness and hardness are always meant with respect to deterministic many-one-log-space reducibilities.

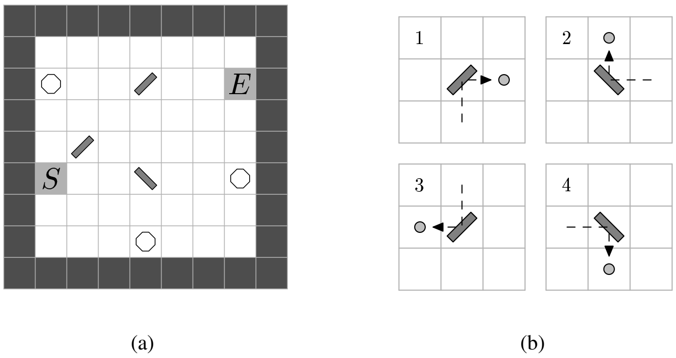


Figure 1: (a) A sample puzzle. (b) In Basic REFLEXION, the ball can travel to all four directions from a mirror.

3. *Basic* REFLEXION *Puzzles*

In the most simple variant of REFLEXION puzzles, which we call Basic REFLEXION, the player may flip the mirrors at any time, both before and after releasing the ball.

THEOREM 1 *Solving Basic* REFLEXION *puzzles is SL-complete (even for puzzles with no diamonds).*

COROLLARY 1 *Basic* REFLEXION *puzzles can be solved in polynomial time.*

Intuitively, the reasons for this are that movements of the ball can be reversed, and that diamonds can be collected in any order. Thus, it suffices to identify the reachable area of the board. This is explained more precisely in the next paragraphs.

First, at any time during a game, it is possible for the player to direct the ball against some wall. As the ball will then reverse its direction, every single move can eventually be reversed. These claims follow from simple induction arguments; we omit the details. Thus, if the ball can reach some mirror, then it can also reach all the cells in a direct line between the mirror and the nearest obstacle (i.e, a mirror or a wall) in all four cardinal directions. Figure 1(b) illustrates this idea. Therefore, we can find the reachable cells by depth-first search from the start cell, branching into all directions whenever we find a mirror. The puzzle has a solution if and only if all diamonds and the target are reachable.

Now we are ready to show containment of Basic REFLEXION within symmetric logspace. We reduce the problem to the graph accessibility problem on undirected graphs, a problem which is known to be SL-complete [1]. From a given Basic REFLEXION puzzle instance we construct an instance of the graph accessibility problem for undirected graphs as follows: First, we define the *basic graph* $G = (V, E)$, where V contains one node for each mirror and a node s for the start. Nodes are connected if they share their x or y coordinate, and there is no obstacle between them (for s, connections are induced by the initial direction of the ball).

Let $m \geq 0$ be the number of diamonds in the puzzle. For each diamond, we construct a new graph G_i with $1 \leq i \leq m$ as follows: G_i is a copy of G with an additional *diamond node* (representing the i-th diamond), connected to the graph as described above. In analogy, we construct the graph G_{m+1} from the basic graph and the target. We are now ready to construct our instance of the graph accessibility problem: Take the graph H as the disjoint union of $G_1, \ldots, G_{m+1}$, and for all $1 \leq i \leq m$ connect the diamond node of G_i with the copy of s in G_{i+1}. Then it is easy to see that the Basic REFLEXION puzzle has a solution if and only if there is an undirected path in H connecting the copy of s in G_1 with the target node in G_{m+1}. This shows containment of the Basic REFLEXION puzzle within the complexity class SL. The H-graph of our running example is depicted in Figure 2.

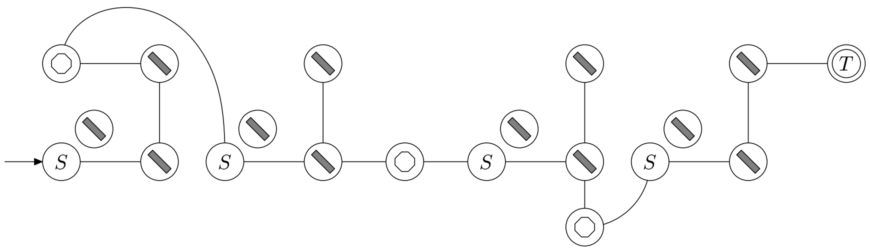


Figure 2: Graph H constructed for the puzzle in Figure 1(a).

For the hardness part we argue as follows: Let $G = (V, E)$ be an undirected graph with $E \subseteq V \times V$. Without loss of generality we may assume that $V = \{1, 2, \ldots, n\}$ and that node n has a loop. Further, let 1 be the source node and n the target node for the graph accessibility problem. We design an instance of the Basic REFLEXION puzzle as follows: First, consider an area of size $2n \times n$, where the lines and columns are numbered from left to right and top to bottom. Then we place mirrors along the diagonal (left half of the area) and moreover, if $(i, j) \in E$, then mirrors are placed at positions $(n + i, j)$ and $(n + j, i)$. It is easy to see, that a ball entering column i can leave the puzzle at column j if (i, j) or $(j, i) \in E$; implicitly we have assumed, that the puzzle under consideration is surrounded by walls. Finally, we place n of these areas on top of each other, shifting each of these areas n columns to the right such that a staircase appears, place the source S on top of the first column, the target E at the bottom of the last column, which is column $(n-1) \cdot n$, and place walls in a rectangular fashion around the whole area. Then one can easily verify that there is a path in G linking 1 and n if and only if the ball released downwards from the source can reach the target. This shows SL-hardness, and together with the above mentioned containment considerations proves Theorem 1.

4. REFLEXION *with Frozen Mirrors*

In some levels of REFLEXION the player may change the orientation of the mirrors only before releasing the ball; once the ball is released, the mirrors are "frozen." Intuitively, this makes the game harder than Basic REFLEXION because the ball can no longer visit every reachable cell in arbitrary order. Theorem 2, which we shall prove in the following, states that this intuition is justified.

THEOREM 2 *Solving* REFLEXION *puzzles with frozen mirrors is NP-complete.*

Membership of the problem in NP is obvious: A nondeterministic Turing machine can solve puzzles by choosing an orientation for each mirror, then simulating the movement of the ball. The simulation can terminate when the ball hits a wall for the second time (because from then on it will travel between the first and the second wall it has hit), and the player wins if the path contains all the diamonds and the target cell. A path between two walls can only be of polynomial length because no mirror can be used more than twice in it.

We shall prove NP-hardness by reducing the problem 3-SAT [4] to REFLEXION with frozen mirrors. 3-SAT can be stated as follows: Given n Boolean variables $x_1, \ldots, x_n$ and a Boolean formula of the form $\bigwedge_{i=1}^{k} c_i$ where each *clause* c_i is a disjunction of three *literals* $\ell_{i1} \vee \ell_{i2} \vee \ell_{i3}$ such that each literal is one of the n Boolean variables or its negation, is the formula satisfiable, i.e., is there a valuation of $x_1, \ldots, x_n$ so that the formula evaluates to true?

The rest of this section describes how one can construct a REFLEXION puzzle from a 3-SAT formula such that the REFLEXION puzzle can be solved if and only if the formula is satisfiable. The construction relies on a number of basic devices with special properties described in Subsection 4.1. Subsection 4.2 describes how to combine these devices to complete the reduction.

In the following, we use two colours for the mirrors, green and red (or light and dark gray). This distinction is merely cosmetic and should help the reader understand the solution more quickly. Both types of mirrors are in principle equal in rights; however, we use red when a mirror faces walls on two sides. Since directing the ball against walls will not be useful, there is only one useful orientation for red mirrors, and that is the orientation we shall show. Green mirrors can be useful in both orientations.

4.1. *Basic Devices*

The start cell of our construction will be placed in a small cul-de-sac with an opening below (see Figure 3(a)) such that the ball travels downwards when released. Now, if the ball hits a wall, it will travel back to the start cell, hit the wall above it, and thus enter a loop. Therefore, we can from now on rule out all configurations of mirrors which cause the ball to hit a wall (before reaching the target).

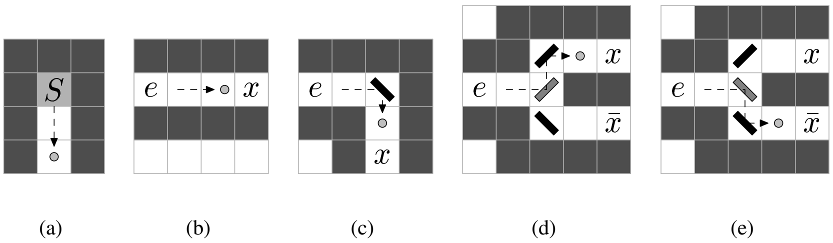


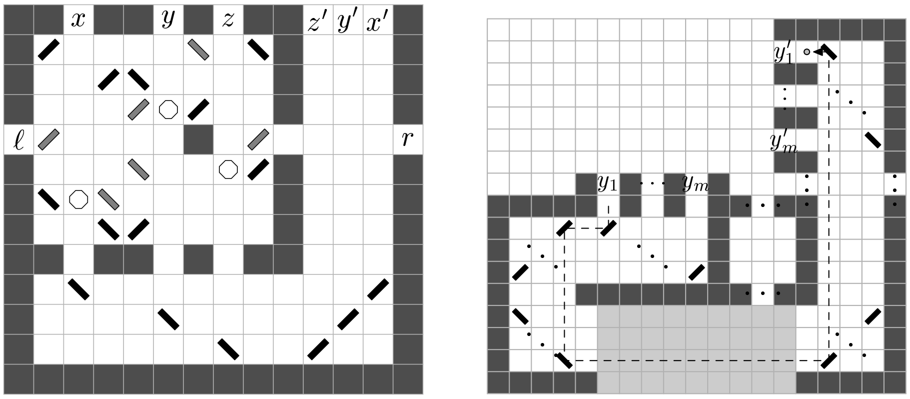
Figure 3: Basic devices

Wires To connect various devices we use "wires", i.e. lines of free cells with walls on both sides. Bent wires are also possible, see Figures 3 (b) and (c).

Choice and Merge Device Figures 3(d) and (e) show a so-called *choice device*. If the ball enters at e, it can exit at either x or $\bar{x}$, depending on the green mirror. The same idea can be used to merge two paths: If the ball enters the device at either x or $\bar{x}$, it can leave at e, provided that the green mirror has the right orientation. Choice and merge devices can easily be generalized to more than two exits or entries, respectively.

The Diamond Case The most complicated of our devices is called the diamond case, shown in Figure 4(a), with two important properties: First, if the ball enters at x, y, or z, it can exit only at x', y', or z', respectively, removing one of the diamonds along the way. Secondly, if the ball enters at r, it can leave at ℓ; moreover, the green mirrors can be configured in such a way that any two of the three diamonds may be removed (but never all three of them). For lack of space, we omit a detailed proof;

however, the reader can easily find configurations for each pair of diamonds, and convince himself/herself that all other configurations of the mirrors lead the ball against some wall.



(a) (b)

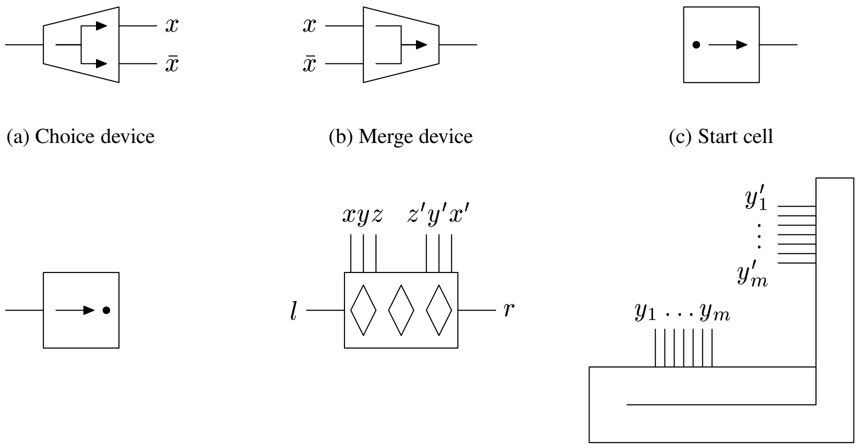
Figure 4: (a) A diamond case. (b) The bus.

The Bus A bus of size m essentially fulfils the function of m wires running in parallel, however, we allow for devices to "tap into" the wires. For the main construction we will need wires of a particular shape. Figure 4(b) shows a schematic view of the bus. The entries are $y_1, \ldots, y_m$ and the exits are $y'_1, \ldots, y'_m$; they are connected by m wires leading from y_i to y'_i, for $1 \leq i \leq m$. The bus is open at the bottom; in the main construction we will connect other devices to the bottom and place additional mirrors in the shaded area of the bus to divert the horizontal movement on the wires. The size of the open area can be enlarged to an arbitrary size.

4.2. *The Reduction*

To simplify the presentation we use diagrams for the basic devices presented in Subsection 4.1. The diagrams are shown in Figure 5; for wires, we use simple lines.

Consider a 3-SAT formula $\bigwedge_{i=1}^{k} c_i$, where $c_i = \ell_{i1} \vee \ell_{i2} \vee \ell_{i3}$, with variables $x_1, \ldots, x_n$. An overview of the corresponding REFLEXION puzzle is given in Figure 6. The puzzle uses a bus with $2n$ wires for all variables and their negations, which is connected to k diamond cases, one for each clause. The three entry/exit pairs at the



(a) Choice device (b) Merge device (c) Start cell

(d) Target cell (e) Diamond case (f) Bus

Figure 5: Basic diagrams

top of the case for c_i, with $1 \leq i \leq k$, shall represent the literals ℓ_{i1}, ℓ_{i2}, and ℓ_{i3}, resp. Thus, we modify the lower $2n$ rows of the bus as follows: For every literal ℓ, the wire associated with ℓ in the bus contains a mirror in those columns that are above an entry or exit of some diamond case that is also associated with ℓ. (There is only one useful orientation for these mirrors.)

A play of the puzzle consists of two phases: In the first phase, the ball moves from the start to a choice gate. The player may configure the choice gate to lead to exit x_1 or $\bar{x}_1$, which corresponds to choosing the value of variable x_1. If x_1 (or $\bar{x}_1$) is chosen, the subsequent passage through the bus removes one diamond for each occurrence of x_1 (or $\bar{x}_1$) in the formula. The ball then passes through a merge gate (whose configuration is dictated by the earlier choice). Next, a truth value for x_2 is chosen, etc., the process is repeated up to x_n, after which the first phase ends. In the second phase, the ball moves to the rightmost diamond case. From there, it traverses the diamond cases (removing up to two diamonds from each case) and reaches the target. At this point, the play is won if and only if all the diamonds have been removed.

Now, the formula is satisfiable if and only if there exists a valuation that makes at least one literal true in each clause. In that case, there is a play of the puzzle in which the first phase removes at least one diamond from each case. The diamond

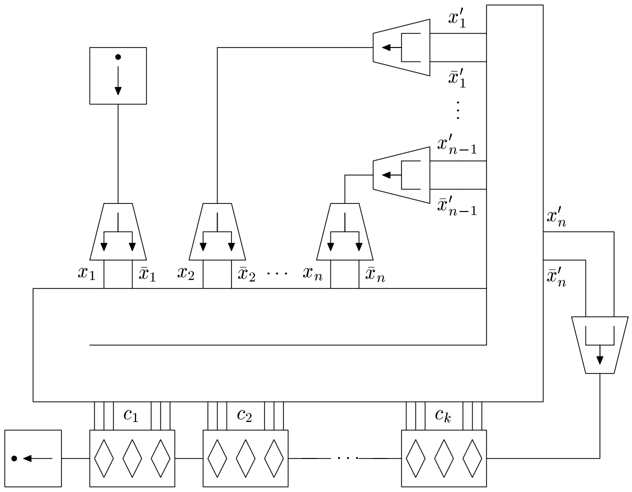


Figure 6: Overview of the construction.

cases can then be set up in such a way that the second phase removes the two or less remaining diamonds. Thus, the puzzle has a solution if and only if the formula is satisfiable. Moreover, the size of the puzzle is clearly polynomial in the size of the formula, which completes the proof of Theorem 2.

5. REFLEXION *with Vanishing Mirrors*

Another variation of the Basic REFLEXION rules used in the game is that the mirrors vanish immediately after their first contact with the ball.

THEOREM 3 *Solving* REFLEXION *puzzles with vanishing mirrors is NP-complete.*

Membership in NP follows from a similar argument than the one for frozen mirrors: A nondeterministic Turing machine can guess orientations of the mirrors and simulate the movement of the ball until it enters a loop, i.e., travels between two walls (the mirrors in between having been removed). A solution exists if the path including one full iteration of the loop reaches the target after passing all the diamonds.

To show NP-hardness, one can re-use the reduction from Section 4. Notice that a solution for the puzzles constructed in Subsection 4.2 does not touch any mirror twice; hence, if such a puzzle is solvable with frozen mirrors, it is also solvable with vanishing mirrors. For the converse, it suffices to realize that changing any of the red mirrors makes almost no difference—all that happens is that the ball hits a wall and then travels into the other direction unopposed—and that changing the green mirrors in an "unexpected" way either has the same effect or results in a premature loop.

6. REFLEXION *with Dynamic Walls*

In this variant, cells in the puzzles may act as triggers or as doors. Trigger cells are associated with one or more door cells. When the ball crosses a trigger, its associated door will change from being a wall to being empty, and *vice versa*. We say that a door is open or closed if it is currently an empty cell or a wall, resp. W.l.o.g., we assume that all doors are open initially. Triggers will be marked with a disc bearing some label, whereas its associated doors appear as shaded cells with the same label.

Like in Basic REFLEXION, every movement of the ball can be reversed in this variant. The argument is the same as for Basic REFLEXION with the addition that walking a path back and forth touches every trigger cell an even number of times. Therefore, a puzzle with dynamic walls has a solution if and only if all diamonds and the target are reachable from the start. In fact, we can reduce the problem to the question whether a single cell is reachable.

THEOREM 4 *Solving* REFLEXION *puzzles with dynamic walls is PSPACE-complete (even for puzzles with no diamonds).*

Membership in PSPACE is obvious: A Turing machine can simulate plays in polynomial space, choosing the orientation of a mirror nondeterministically when a ball touches it, and stopping when the number of simulated moves has exceeded the number of potential configurations of the puzzle. PSPACE-hardness will be shown by reduction from QBF, i.e., the problem of deciding whether a quantified Boolean formula holds, which is known to be PSPACE-complete even for formulas of the form $Q_1x_1 \ldots Q_nx_n \bigwedge_{i=1}^{k} c_i$, where $Q_1, \ldots, Q_n \in \{\exists, \forall\}$ and c_i, for $1 \leq i \leq k$, is a disjunction of literals taken from $x_1, \ldots, x_n$ or their negations. Given such a formula, we construct a REFLEXION puzzle that has a solution if and only if the formula holds, which proves the PSPACE-hardness claim. As in Section 4, we introduce some devices first, and then show how to assemble them to achieve the construction.

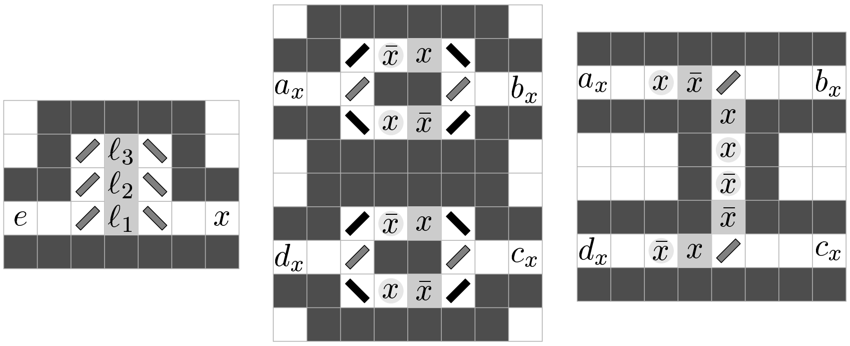
6.1. *Logic Devices*

Apart from the devices from Section 4, we need three additional ones: A gate for representing disjunctive clauses, and gates for existential and universal quantifiers.

Disjunction Gate Consider the simple device shown in Figure 7 (a). Quite obviously, passage from e to x (or *vice versa*) is possible if and only if one of the doors labelled ℓ_1, ℓ_2, or ℓ_3 is open. Thus, this gate can represent a disjunctive clause of the form $\ell_1 \vee \ell_2 \vee \ell_3$. The gate can be generalized to an arbitrary number of literals.

Existential gate Consider the device shown in Figure 7(b). There are two passages from a_x to b_x, and likewise between c_x and d_x. Passage of the lower (upper) route from a_x to b_x restricts the passage from c_x and d_x to the lower (upper) route. Thus, any passage from a_x to d_x *via* b_x and c_x triggers x and $\bar{x}$ an even number of times, leaving the respective doors in the same state as before.

In the main construction, choosing the upper route will correspond to making x true, since doing so will close all $\bar{x}$-labelled doors (including doors elsewhere, e.g. in disjunction gates). Likewise, choosing the lower route makes x false. The part of the puzzle between b_x and c_x will correspond to a QBF formula φ in which x is a free variable, and passage from b_x to c_x will be possible if and only if φ is true under the choice of x. Thus, passage from a_x to d_x will be possible if and only if $\exists x\ \varphi$ is true.



(a) (b) (c)

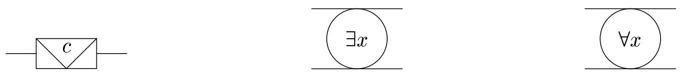
Figure 7: (a) Disjunction gate. (b) Existential gate. (c) Universal gate.

Universal gate The universal gate for a variable x is shown in Figure 7(c). Again, the entry is at a_x, and the exit at d_x. A passage from a_x to d_x consists of the following steps: Upon entry at a_x, the ball triggers x, sealing the central passage from above. Thus, the ball can only go to b_x. Suppose the ball can go from there to c_x. Then, the passage to d_x is blocked, and the ball can only travel upwards in the central wire. The effect of that is to block the passage to a_x and to the lower end of the central passage, and to unblock the passage to d_x and the upper end of the central passage. The only route, then, is to go through b_x and c_x once more, and finally to exit at d_x.

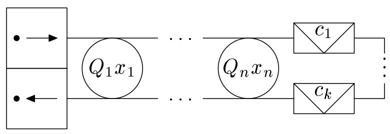
In the final construction the part between b_x and c_x will correspond to a formula φ with x as a free variable. The ball must go through that part twice: Once for $\bar{x}$, and once for x. Thus, passage from a_x to d_x is possible if and only if $\forall x\ \varphi$ is true.

6.2. *The Reduction*

For the reduction, we represent the devices from Subsection 6.1 with the diagrams in Figure 8, (a) to (c). In the disjunction gate, the label c is a disjunction of the form $\bigvee_{j=1}^{m} \ell_j$ for some m and literals $\ell_1, \ldots, \ell_m$.



(a) Disjunction gate. (b) Existential gate. (c) Universal gate.



(d) QBF reduction.

Figure 8: Diagrams for logic devices, and QBF reduction.

Let φ be a quantified Boolean formula of the form $Q_1 x_1 \ldots Q_n x_n \bigwedge_{i=1}^{k} c_i$, where $Q_1, \ldots, Q_n \in \{\exists, \forall\}$, and $c_i = \bigvee_{j=1}^{m_i} \ell_{ij}$, where the ℓ_{ij} are variables from x_1 to x_n or their negations. Figure 8 (d) shows a REFLEXION puzzle in which the ball can reach the target if and only if φ is true. The correctness of the construction should be easy to see: In the upper part, traversing the quantifier gates from left to right chooses an assignment for the variables. After that, the doors for all true literals are still open, and the doors for all false literals are closed. Thus, the ball can traverse all disjunction gates if and only if all corresponding clauses are true. Traversing the quantifier gates from right to left (in the lower part) resets the doors and allows the clauses to be evaluated repeatedly. Thus, the target can be reached if and only if φ is true.

One-to-One Relations For the construction above, we used many-to-many relations between triggers and doors, i.e. we assumed that (a) triggers can be associated with multiple doors, i.e., multiple doors can have the same label, and (b) doors can be changed by multiple triggers, i.e., multiple triggers can have the same label. However, our constructions can be modified to hold even for one-to-one relations. The (sketch of the) modifications is as follows: Assumption (a) can be eliminated by using a distinctive label for each occurrence of a literal. Then, in the quantifier gates, each trigger is replaced by a sequence of triggers, one for each distinctive label. For assumption (b), notice that the construction has exactly two triggers for each label, one for setting an assignment, the other for resetting it. Instead, we could use just one trigger and traverse it horizontally to set assignments, and vertically to reset them.

7. REFLEXION *with Movable Boxes*

In this section, we examine an addition to the rules where cells may contain a box. Each box has an orientation, horizontal or vertical. (We represent movable boxes by shaded boxes with arrows ($\updownarrow, \leftrightarrow$) indicating their orientation.) When the ball hits a box, the ball is reflected, and moreover the box moves into the next cell along the (previous) direction of the ball if such a move is compatible with the orientation of the box, i.e., a horizontally oriented box can move only horizontally,[2] and if the endpoint of the move is not occupied by another object (mirror, diamond, or another box).

This variant differs from the variant with dynamic walls in the sense that not all moves are reversible. Nonetheless, it falls into the same complexity class, and we can even adapt the QBF construction from Section 6 for the proof.

THEOREM 5 *Solving* REFLEXION *puzzles with movable boxes is PSPACE-complete (even for puzzles with no diamonds).*

Membership in PSPACE follows from the same arguments as in Theorem 4. For the hardness proof we begin by introducing two basic devices (Subsection 7.1). These will enable us to emulate the logic devices from Subsection 6.1. The final construction is then analogous to the previous QBF reduction, using the adapted logic devices.

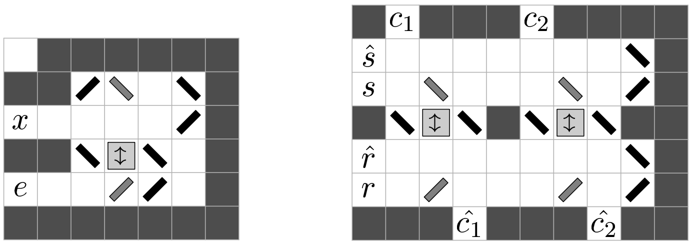
For the rest of the section, fix φ as a QBF formula $Q_1 x_1 \ldots Q_n x_n \bigwedge_{i=1}^{k} c_i$, where $Q_1, \ldots, Q_n \in \{\exists, \forall\}$ and c_i, for $1 \leq i \leq k$, is a disjunction of literals taken from $x_1, \ldots, x_n$ or their negations. We shall construct a REFLEXION puzzle with movable boxes such that the puzzle has a solution if and only if φ is true.

[2] This condition is used in the game, but our construction is actually independent of it.

7.1. *Basic Devices*

We introduce two devices: A one-way gate and a device for representing a literal.

One-Way Gate The device shown in Figure 9(a) has the following property: Entry at e enables exit at x, but not *vice versa*. Note that this property holds independently of the initial position of the movable box.



(a) One-way device.

(b) Literal device.

Figure 9: Basic devices for REFLEXION with movable boxes.

Literal Device The *literal device* is shown in Figure 9(b). We shall need $2n$ such devices, one for each variable and its negation. In the device for literal ℓ, the position of the movable boxes records the truth value of ℓ; if the boxes are in their upper position (as shown in the figure), ℓ is taken to be false, and if the boxes are shifted down by one cell, ℓ is taken to be true.

The box has several entry/exit pairs. Entry at label e (where $e \in \{s, r, c_i\}$) leads to the exit $\hat{e}$; we assume that entries and exits are protected by appropriate one-way gates. Passage from s to $\hat{s}$ sets ℓ to "true". Note that the player can choose to leave some or all of the boxes in their "false" position; however, as we shall see, doing so can only reduce the chances of completing the puzzle (and since we are just interested in the existence of a solution, we can ignore such "stupid" moves). On the other hand, entry at r and exit at $\hat{r}$ resets ℓ back to false; the player is forced to move all boxes back up to get to $\hat{r}$. Passage from c_1 (or c_2) to $\hat{c}_1$ (or $\hat{c}_2$) serves to check the value of ℓ: The passage is enabled if and only if the player has previously set ℓ to "true" (and has pushed the boxes down). The device can be extended to an arbitrary number of such entry/exit pairs. In our construction, for literal ℓ the number of such pairs will be exactly the number of occurrences of ℓ in φ plus one.

7.2. *Logic Devices*

We can adapt the devices from Subsection 6.1. To simplify the diagrams, we make use of the *access symbols* in Figure 10(a). The symbol labelled "set ℓ" represents a path leading to the s entry of the literal device for ℓ and back from its $\hat{s}$ exit, in the direction indicated by the point of the box. Analogously, the symbol labelled "reset ℓ" represents a path to the r entry and back from the $\hat{r}$ exit, and "check ℓ" represents a path to some unique "checking" entry of the corresponding literal device (i.e., no two occurrences of a "check ℓ" symbol should connect to the same "checking" entry).

Using these access symbols, we can now build the components of our solution: Figure 10, parts (b), (c), and (d) show a disjunction gate, an existential gate, and a universal gate, resp. The QBF reduction for movable boxes is analogous to Figure 8 (d), using the gates presented in this section. The paths leading to the literal devices and back can be implemented by something like the bus device from Section 4 (we omit the details). Thus, the size of the construction is polynomial in the size of φ.

8. *Conclusions*

In this paper we have analyzed the computational complexity of solving REFLEXION puzzle variants. It turned out that the large variety of different features in the game leads to completeness results for SL, NP, and PSPACE. For the SL- and PSPACE-completeness results the presence of diamonds within the puzzle is not necessary. Whether a similar statement is also valid for the presented NP-completeness results for frozen and vanishing mirrors or combinations thereof, remains an open question.

Acknowledgements

Thanks to Klaus Stein for discussing some issues concerning Basic REFLEXION.

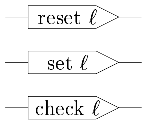
References

[1] C. Àlvarez and R. Greenlaw. A compendium of problems complete for symmetric logarithmic space. Report TR96-039, ECCC, September 1996.

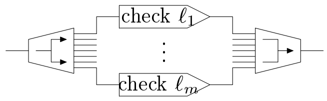
[2] J. Culberson. Sokoban is PSPACE-complete. In *Proceedings of the* 1*st Conference on Fun with Algorithms*, pages 65–76, Island of Elba, Italy, 1999. Carleton Scientific.

[3] G. W. Flake and E. B. Baum. RushHour is PSPACE-complete, or "Why you should generously tip parking lot attendants". *Theoretical Computer Science*, 270(1–2):895–911, 2002.

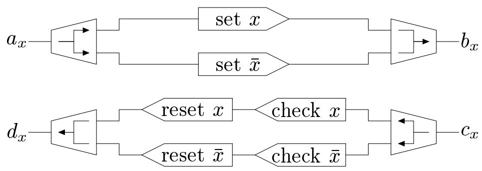
[4] M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. Freeman, 1979.



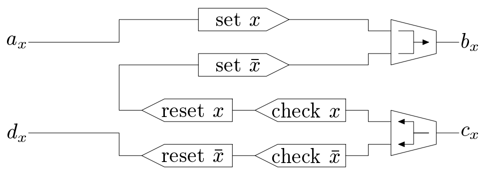
(a) Access symbols



(b) Disjunction gate.



(c) Existential gate.



(d) Universal gate.

Figure 10: Logic devices for REFLEXION with movable boxes.

[5] M. Holzer and S. Schwoon. Assembling molecules in ATOMIX is hard. *Theoretical Computer Science*, 313(3):447–462, February 2004.

[6] D. Kempe. On the complexity of the “Reflections” game. Unpublished Manuscript, 2003.

[7] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.

[8] W. J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, April 1970.