Permutation Reconstruction from Differences

Marzio De Biasi

marziodebiasi@gmail.com

Submitted: Feb 5, 2014; Accepted: Sep 21, 2014; Published: Oct 2, 2014
Mathematics Subject Classifications: 68Q17

Abstract

We prove that the problem of reconstructing a permutation $\pi_1, \dots, \pi_n$ of the integers $[1 \dots n]$ given the absolute differences $|\pi_{i+1} - \pi_i|$, $i = 1, \dots, n-1$ is NP–complete. As an intermediate step we first prove the NP–completeness of the decision version of a new puzzle game that we call *Crazy Frog Puzzle*. The permutation reconstruction from differences is one of the simplest combinatorial problems that have been proved to be computationally intractable.

1 Introduction

Permutation reconstruction has been studied as a variation of the graph reconstruction problem that arose from an unsolved conjecture of Ulam [8]. Given an unlabeled graph A, deleting one of its vertex together with its incident edges in each possible way we obtain the minors $A_1, ..., A_n$. Ulam's conjecture says that given two graphs A, B with $n > 3$ vertices, if there exists a bijection $\alpha : \{1, ..., n\} \to \{1, ..., n\}$ such that A_i is isomorphic to $B_{\alpha(i)}$ then A is isomorphic to B. In the permutation reconstruction variant, we consider a permutation p with n entries; we can delete k of the entries in each possible way and renumber them with respect to order and obtain $\binom{n}{k}$ permutations of the numbers from 1 to $n-k$ that are called $(n-k)$-minors. Smith [7] introduced the problem of reconstructing the original permutation p from the multiset $M_k(p)$ of its $(n-k)$-minors and looked at the number N_k defined to be the smallest number such that we can reconstruct permutations of length $n \geqslant N_k$. Raykova [5] proved the existence of N_k for every positive integer k and gave an upper bound of $N_k < \frac{k^2}{4} + 2k + 4$ and a lower bound $N_k > k + \log_2 k$. Monks [4] studied the reconstruction of a permutation p from its set of cycle minors, where each i-th cycle minor is obtained deleting the entry i from the decomposition of p into disjoint cycles and reducing each remaining entry larger than i by 1. He showed that any permutation of $\{1, 2, ..., n\}$ can be reconstructed from its set of cycle minors if and only if $n \geqslant 6$.

In this paper we focus our attention to another variant that has an even simpler formulation: the problem of reconstructing a permutation $\pi_1, \dots, \pi_n$ of the integers $[1 \dots n]$

given the absolute differences $|\pi_{i+1} - \pi_i|$, $i = 1, \ldots, n-1$. We will prove that deciding if such a permutation exists is NP–complete: any given solution can be quickly verified in polynomial time, but there is no known way to build an efficient polynomial time algorithm that find a solution in the first place. And such an efficient algorithm doesn't exist unless $\mathsf{P} = \mathsf{NP}$, which is the major open problem in computer science.

In order to prove our result we first introduce a new puzzle game, the *Crazy Frog Puzzle*, with the following rules: a frog is placed on a square grid board; some cells of the grid are blocked, some are empty. The frog must follow a given sequence of horizontal, vertical or diagonal jumps of varying length; at every jump the frog can only decide to follow the given direction or jump in the opposite direction. For example, when facing an horizontal jump of length two, the frog placed on cell (x, y) can jump left on cell $(x-2, y)$ or right on cell $(x+2, y)$. The frog cannot jump outside the board, on a blocked cell, or on a cell that has already been visited. The aim of the game is to choose the correct directions of the jumps and make the frog visit all the empty cells of the board exactly once. Figure 1 shows an instance of the Crazy Frog Puzzle and its solution.

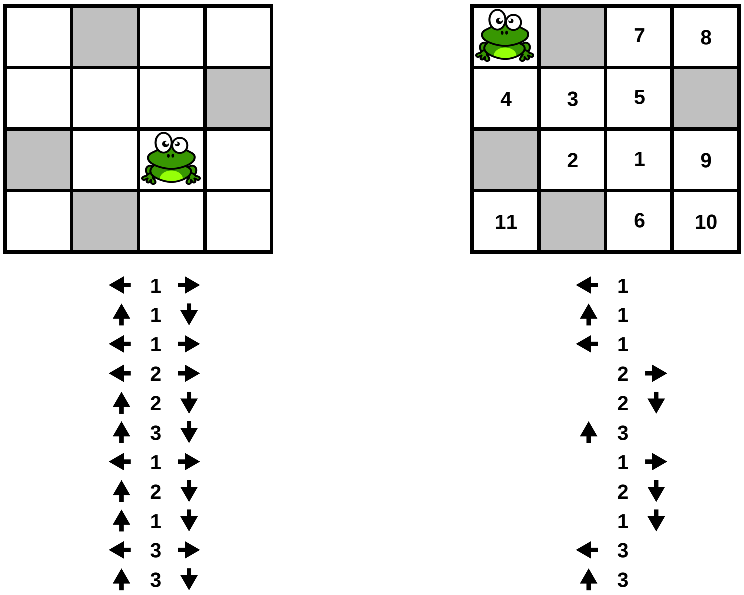


Figure 1: An instance of the Crazy Frog Puzzle on the left and its solution on the right.

In line with the recent interest in the complexity of puzzle games [3] [1], we study how hard it can be to solve a Crazy Frog Puzzle game. In Section 2 we formally define the *decision* version of the Crazy Frog Puzzle (CFP) and give some terminology; in Section 3 we prove that deciding if a given Crazy Frog Puzzle has a solution is NP–complete. In Section 4 we prove that CFP remains NP-complete even if the board is 1-dimensional. Finally in Section 5 we prove that the problem remains NP–complete even if the initial board has no blocked cells and we will show that the 1-dimensional CFP without blocked cells is equivalent to the permutation reconstruction from differences problem.

2 Crazy Frog Puzzle

We define the decision problem [6] that asks whether a given Crazy Frog Puzzle has a solution or not:

Definition 2.1 (Crazy Frog Puzzle).

Input: An $n \times n$ partially filled board: some cells are *empty* some cells are *blocked*, a starting cell $c_0 = (x_0, y_0)$ and a sequence of m *jumps* $\Delta_1, \Delta_2, \ldots, \Delta_m$, $\Delta_i = (dx_i, dy_i), -n \leqslant dx_i, dy_i \leqslant n$ (m equals the number of the initial empty cells).

Question: Does there exist a sequence of integers $(s_1, s_2, \ldots, s_m), s_i \in \{-1, +1\}$ such that if a frog is placed on the starting cell, it can *visit* (we will equivalently use the term *fill*) every empty cell of the board exactly once following the sequence of jumps:

$$(x_i, y_i) = (x_{i-1} + s_i * dx_i, y_{i-1} + s_i * dy_i)$$

(i.e. the frog can choose only the *direction* of the given jumps)? The frog cannot jump outside the board, on a blocked cell or on an already visited cell.

2.1 Terminology

We briefly introduce the terminology used in the next sections.

Horizontal jump : a jump of the form $(\Delta x, 0)$ (*horizontal step* if $\Delta x = 1$);

Vertical jump : a jump of the form $(0, \Delta y)$ (*vertical step* if $\Delta y = 1$);

Diagonal jump : a jump of the form $(\Delta x, \Delta x)$ (*diagonal step* if $\Delta x = 1$);

Sequence of jumps : a (sub)sequence $\Delta_i, \Delta_{i+1}, \ldots, \Delta_m$ of jumps that are part of the input; the frog at each step Δ_i must choose a direction ($s_i \in \{\pm 1\}$) and make the jump ($+\Delta_i$ or $-\Delta_i$).

When there is no ambiguity horizontal/vertical/diagonal jumps are abbreviated with a single letter or a single number; for example if we are describing a horizontal sequence of jumps, we can write $2, 3, 2$ instead of $(2, 0), (3, 0), (2, 0)$, or write $x_1, x_2, \ldots, x_j$ instead of $(x_1, 0), (x_2, 0), \ldots, (x_j, 0)$.

Line : a row of the board; we use a string to represent it using these characters: B=blocked cell, E=empty cell, F=frog, V=visited cell; repeated cells can be represented with power notation (e.g. $B^2FE^3 = BBFEEE$);

Strip : two or more lines;

Configuration : the status of the board cells (B,E or F) and the sequence of the next jumps; a configuration is *valid*, if the frog can complete the sequence of jumps choosing a ±1 direction for each of them;

Gadget : the configuration of a part of the board that have a particular role in the reduction; a gadget can have one or more *entrance cells* and one or more *exit cells*. A *(valid) traversal* of the gadget is a sequence of jump directions that can lead the frog from an entrance cell to an exit cell. We will use capital letters: (e.g. L, C, S) to indicate gadgets, capital letters with a tilde (e.g $\tilde{L}, \tilde{C}, \tilde{S}$) to indicate jump sequences.

In the figures we will use the following notation: gray cells represent blocked cells; a cell with a F represents the frog position; a cell with a t represents the target cell; a cell with a V represents an already visited cell; numbered cells represent a (valid) sequence of jumps that the frog can make. The horizontal coordinates $x = 0, 1, 2, \dots$ are from left to right, the vertical coordinates $y = 0, 1, 2, \dots$ are from top to bottom.

3 NP–Completeness

We will prove that the CFP problem is NP–hard giving a polynomial time reduction from the Hamiltonian path problem on grid graphs [2]. First we underline a general property that we will use in the gadget construction.

Lemma 3.1 (Gadget construction)**.** *Given a sequence of jumps $(x_1, y_1), \dots, (x_m, y_m)$ and a $w \times h$ rectangular area R of the board, we can construct a corresponding gadget in which all jumps of the sequence must be made inside it and the frog can exit it only at the end.*

Proof. It is sufficient to extend the area adding a $B^{\max\{x_i\}}$ border of blocked cells on the left/right, a $B^{\max\{y_i\}}$ border of blocked cells on the top/bottom. Then we can make the frog enter the gadget with a vertical jump $y_{In} > \max\{y_i\}$ and leave it with another vertical jump $y_{Out} > \max\{y_i\}$. The extended sequence of jumps for the traversal is:

$$\dots, y_{In}, (x_1, y_1), \dots, (x_n, y_n), y_{Out}, \dots$$

□

Note that *a)* if R contains $m + 1$ empty cells (the number of jumps is equal to the number of empty cells minus one), then a valid traversal of the gadget implies that the frog must visit (fill) all its empty cells; *b)* instead of vertical jumps we can use long enough horizontal or diagonal jumps. Figure 2 shows an example of a 5×5 partially filled region and a sequence of 6 jumps; the region can be embedded in a 7×7 gadget that can be traversed in 4 different ways; all traversals completely fill the original inner region.

3.1 Reduction overview

The reduction is from the NP–complete Hamiltonian path problem on grid graphs (which may also contain holes) [2]. Let G^∞ be the infinite graph whose vertex set consists of all points of the plane with integer coordinates and in which two vertices are connected if and only if the (Euclidean) distance between them is equal to 1. A *grid graph* is a finite, node–induced subgraph of G^∞.

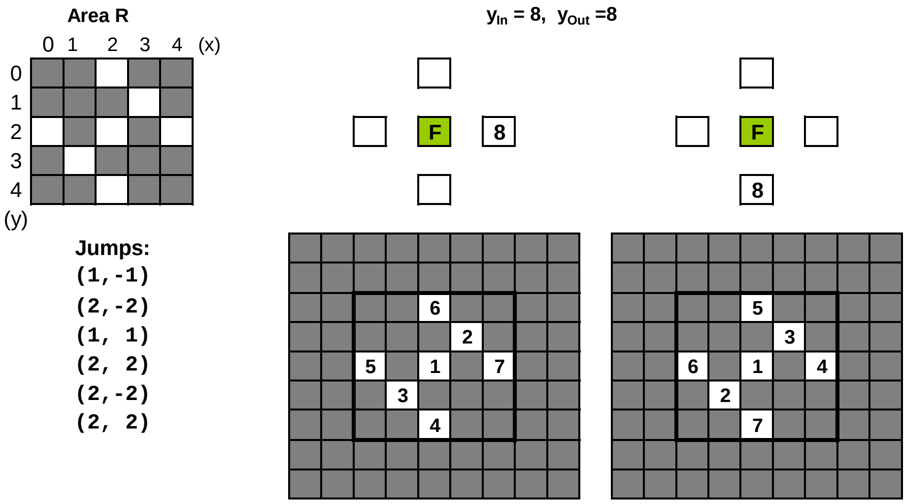


Figure 2: A 5 × 5 partially filled region and a sequence of 6 jumps; the region can be embedded in a 7 × 7 gadget; two of the four possible traversals are showed on the right.

Given an $m \times m$ grid graph G in which $|V| = n$, $s, t \in V$ are the source and target nodes; and the coordinates of node $u_i, 1 \leqslant i \leqslant n$ on the grid are (x_{u_i}, y_{u_i}).

Pick the first $k \geqslant 2$ such that $2^k \geqslant 4m$ and build a $2^k - 1 \times 2^k - 1$ board R with all the cells blocked except the cells at coordinates $(4x_{u_i}, 4y_{u_i})$ corresponding to the *nodes* of G and the empty *target cell* $(4x_t + 1, 4y_t)$ one step aside from the node t. We call this part of the board the *graph area*. The frog is initially positioned on cell $(4x_s, 4y_s)$.

For brevity, throughout the paper we will denote its size with $w = 2^k - 1$ and $v = 2^{k-1} = \lceil w/2 \rceil$.

We extend the board at the bottom with $n - 1$ *edge gadgets* $L_1, L_2, \ldots, L_{n-1}$; each edge gadget L_i has an associated *cleanup gadget* C_i placed on its right. We will generate a fixed sequence of jumps that will force the following logical *phases*:

- $[\tilde{L}_1]$ enter gadget L_1, choose one of the four directions up, down, left, right and return to cell $(x_0 \pm 4, y_0)$ or $(x_0, y_0 \pm 4)$ in the graph area;
- $[\tilde{L}_2]$ enter gadget L_2, choose one of the four directions and return to cell $(x_1 \pm 4, y_1)$ or $(x_1, y_1 \pm 4)$ in the graph area;
- ...
- $[\tilde{L}_{n-1}]$ enter gadget L_{n-1}, choose one of the four directions and return to cell $(x_{n-1} \pm 4, y_{n-1})$ or $(x_{n-1}, y_{n-1} \pm 4)$ in the graph area;
- $[\tilde{T}]$ jump to the target cell (x_t, y_t);
- enter the cleanup gadgets area;
- $[\tilde{C}_1]$ completely fill the lines of the edge gadget L_1 that have an already visited cell (visited during phase $\tilde{L}_1$); then completely fill the lines that are still empty;

- ...
- $[\tilde{C}_{n-1}]$ completely fill the lines of the edge gadget L_{n-1} that have an already visited cell (visited during phase $\tilde{L}_{n-1}$); then completely fill the lines that are still empty.

An outline of the whole board is shown in Figure 3.

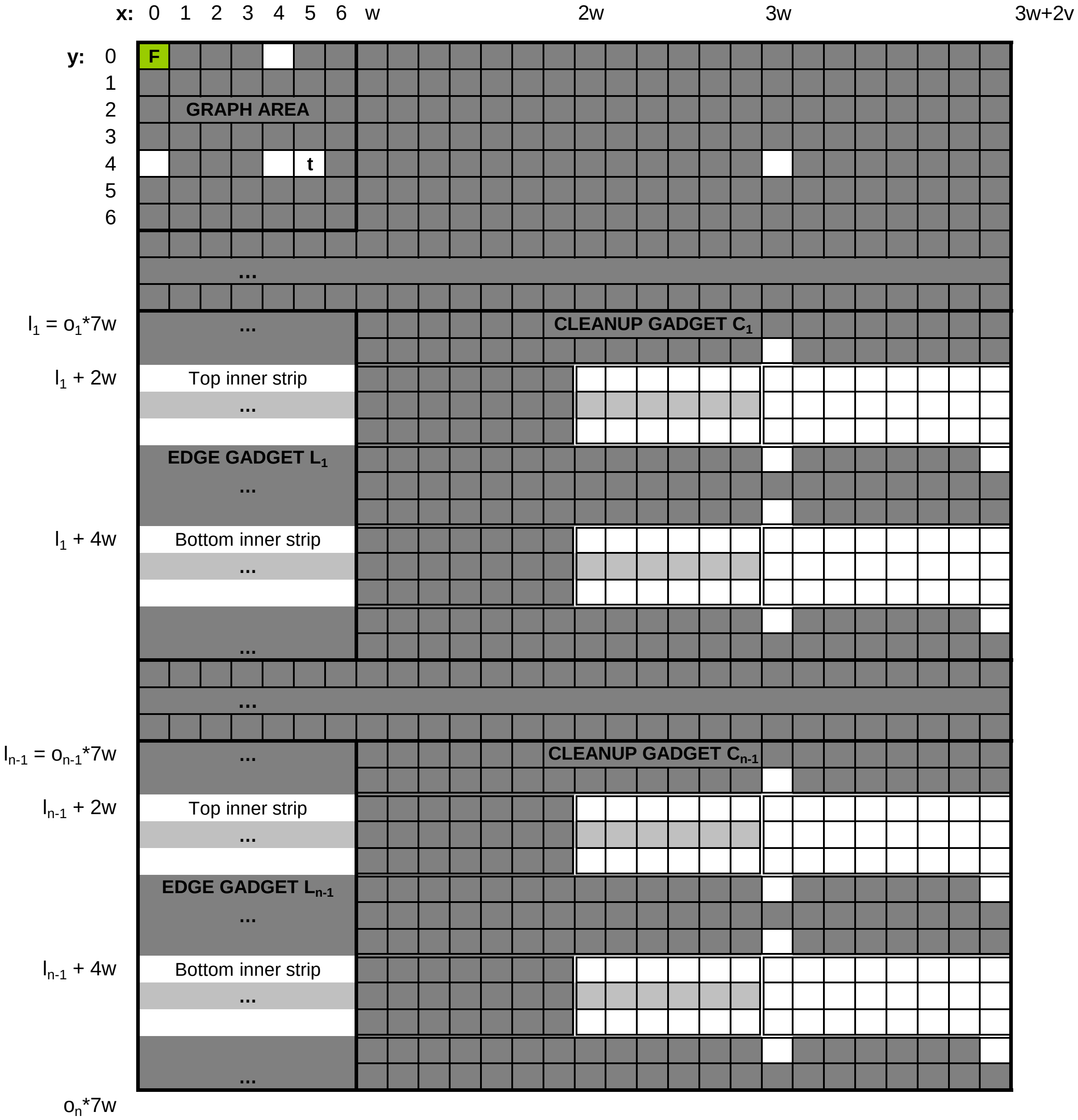


Figure 3: An outline of the board generated by the reduction.

3.2 Edge gadgets

Each edge gadget L_i is a rectangular area that has the same width w of the graph area, and $7w$ lines: $2w$ blocked lines; a first *top inner strip* of height w in which even lines are empty and odd lines are blocked; w blocked lines; a second *bottom inner strip* of height w in which even lines are empty and odd lines are blocked; and finally $2w$ blocked lines. If the gadget is positioned at row l_i its structure is:

row #	cells	repetitions
l_i:	B^w	$\times 2w$
$l_i + 2w$:	E^w B^w	$\times \lfloor \frac{w}{2} \rfloor$
	E^w	$\times 1$
$l_i + 3w$:	B^w	$\times w$
$l_i + 4w$:	E^w B^w	$\times \lfloor \frac{w}{2} \rfloor$
	E^w	$\times 1$
$l_i + 5w$:	B^w	$\times 2w$

We make the frog enter an edge gadget from the graph area with a vertical jump J_{L_i} on an empty cell of the upper inner strip, and leave it from the bottom inner strip with a vertical jump $J'_{L_i} = J_{L_i} + 2w$ that make it return to the graph area.

The vertical positions l_i of the edge gadgets must be chosen in such a way that the frog cannot leave one of them and directly jump to another edge gadget, but is forced to return to the graph area. This is achieved using the vertical positions $l_i = o_i * 7w$, for odd $o_i \geqslant 1$, and setting $J_{L_i} = l_i + 2w$ and $J'_{L_i} = J_{L_i} + 2w = l_i + 4w$. Indeed if the frog begins on $(4x, 4y)$ in the graph area for some k, it'll end up at vertical coordinate $l_i + 4w + 4y + 4z$ for z in $\{-1, 0, 1\}$ at the end of the traversal of gadget L_i. At this point the next vertical jump of $(0, l_i + 4w)$ shouldn't allow the frog to jump further down to another edge gadget; i.e. $l_i + 4w + 4y + 4z + l_i + 4w = 2l_i + 8w + 4y + 4z$ should be different from $l_j + 2w + 2a$ and $l_j + 4w + 2b$ for all $i \neq j$ and $0 \leqslant a, b < w/2$; but the inequalities $2l_i + 8w + 4y + 4z \neq l_j + 2w + 2a$ and $2l_i + 8w + 4y + 4z \neq l_j + 4w + 2b$ hold because the left-hand side is even, and the right-hand side is an odd number ($l_j = o_j * 7w$ is odd because both o_j and w are odd) plus an even number.

The sequence of jumps inside each edge gadget is:

$$\tilde{L}_{seq} = (2, 2), (0, 2w), (2, -2)$$

The first jump is a diagonal jump that must be made in the top inner strip, the second jump forces the frog to jump to the bottom inner strip, the third jump is a diagonal jump that must be made in the bottom inner strip. The traversal of gadget L_i is forced with the sequence of jumps:

$$\tilde{L}_i = (0, J_{L_i}), \tilde{L}_{seq}, (0, J'_{L_i})$$

Suppose that the frog is on cell (x, y) and enters the gadget L_i from the top with the vertical jump $(0, J_{L_i})$: after the $\tilde{L}_{seq}$ jumps it can only use the final vertical jump $(0, J'_{L_i})$ to return to the graph area, and its final position must be one of the cells: $(x+4, y), (x-4, y), (x, y-4), (x, y+4)$. After each traversal only four cells of the inner strips are visited. Figure 4 shows an example of a 15×60 edge gadget ($w = 15$) and its possible traversals.

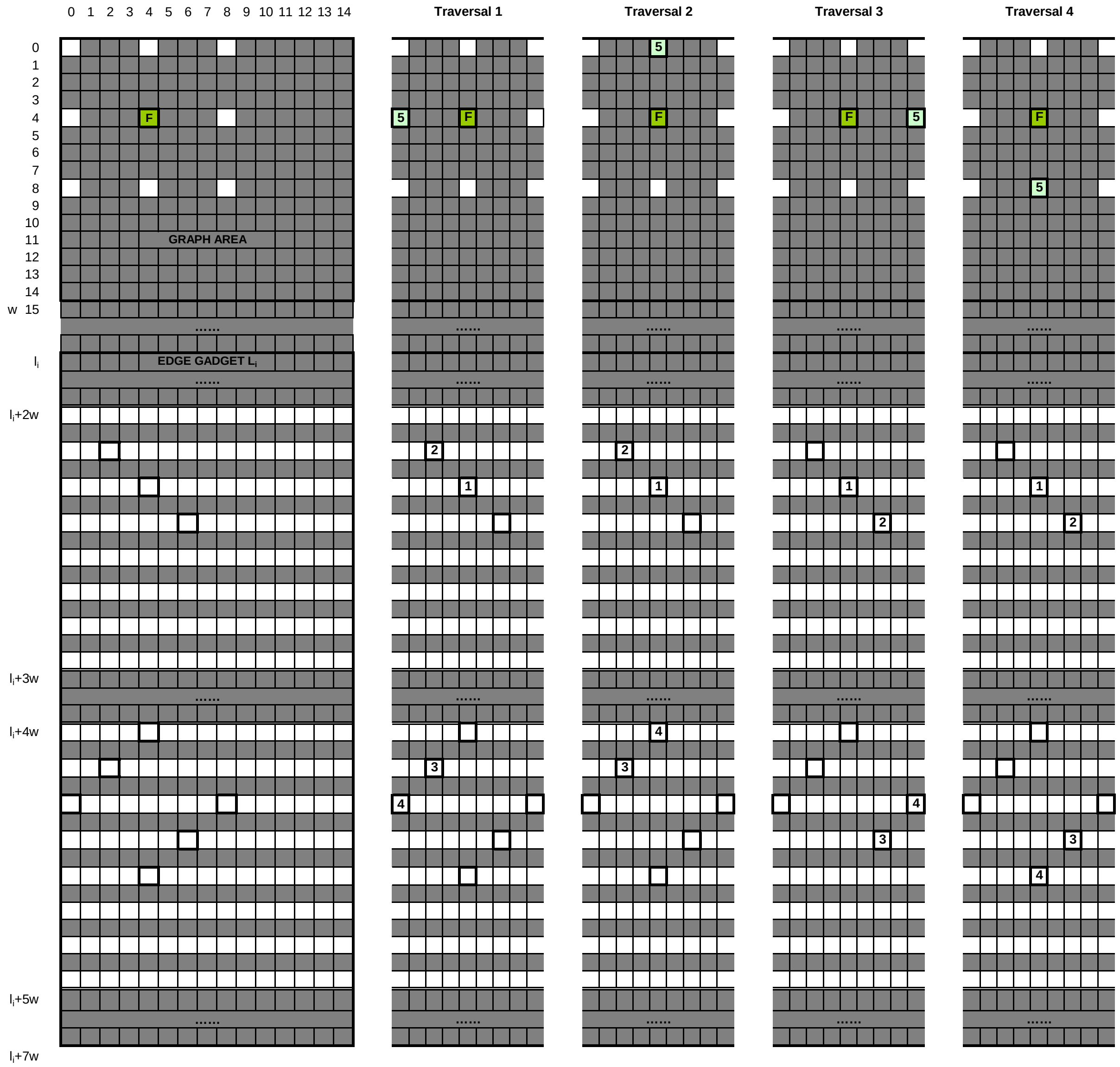


Figure 4: The edge gadget and its four possible traversals.

We can define the first part of the jump sequence of the CFP in this way:

Phase	Jumps	
$\tilde{L}_1$	$(0, J_{L_1}), \tilde{L}_{seq}, (0, J'_{L_1})$	enter edge gadget L_1, traverse it and return to graph area;
$\tilde{L}_2$	$(0, J_{L_2}), \tilde{L}_{seq}, (0, J'_{L_2})$	enter edge gadget L_2, traverse it and return to graph area;
...	...	...
$\tilde{L}_{n-1}$	$(0, J_{L_{n-1}}), \tilde{L}_{seq}, (0, J'_{L_{n-1}})$	enter edge gadget L_{n-1}, traverse it and return to graph area;
$\tilde{T}$	(1,0)	jump to target cell t.

Note that the final odd horizontal step (the only odd horizontal step), forces the frog to choose a path in the graph area in which the final cell is the one corresponding to node t. Furthermore if the frog is on cell (x_i, y_i) and traverses the edge gadget L_i it must, by construction, be in one of the four adjacent cells $(x_i \pm 4, y_i), (x_i, y_i \pm 4)$ and that cell must be empty (i.e. correspond to an unvisited node), so the sequence of cells visited in the graph area must correspond to an Hamiltonian path from s to t on the original graph G. So the following lemma holds:

Lemma 3.2. *The frog can reach cell (x_t, y_t) from its initial position (x_s, y_s) if and only if there is an Hamiltonian path from s to t in the original graph G.*

At the end of the graph area traversal, most cells of the edge gadgets are still empty, so we must extend the jump sequence to let the frog visit all of them and completely fill the board.

The cleanup gadgets are more complicated because they must allow the frog to fill both the lines of the edge gadgets that has a single blocked cell, and the lines of the edge gadgets that have been left empty.

3.3 Cleanup gadgets

Every cleanup gadget C_i has two similar *strip cleanup gadgets* S_i^a,S_i^b one for the top and one for the bottom inner strip of the corresponding edge gadget L_i. The strip cleanup gadget is placed on the right of the corresponding inner strip, at coordinate $(w, l_i + 2w)$ for top inner strips ($(w, l_i + 4w)$ for bottom inner strips), and is a $3w + 2v \times w$ rectangular area of the board (note that $v = \lceil w/2 \rceil$ is simply the number of even rows in the inner strip) with the following structure:

row #	cells
even rows:	$B^w E^w E^{2v}$
odd rows:	$B^w B^w E^{2v}$

Furthermore, outside the gadget at coordinates $(3w, l_i + 2w - 1), (3w, l_i + 3w), (3w + 2v - 1, l_i + 3w)$ there are three empty cells that are the entrance and exit cells of the gadget S_i. Figure 5 shows the outline of a strip cleanup gadget associated to a 7×7 $(w = 7)$ inner strip.

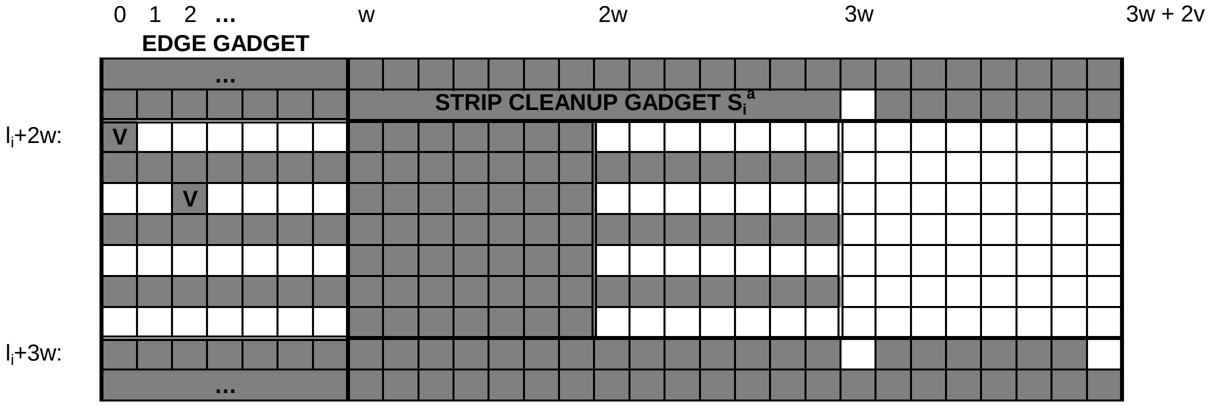


Figure 5: A strip cleanup gadget S_i associated to the top inner strip of the edge gadget L_i (assuming $w = 7$).

The jump sequence $\tilde{C}_i$ using for traversing the cleanup gadget C_i is:

$$\tilde{C}_i = \tilde{S}_i, (-2v + 1,), (0, w - 1), \tilde{S}_i$$

where $\tilde{S}_i$ is the jump sequence for traversing a strip cleanup gadget and $(-2v+1,), (0, w-1)$ are the jumps from the exit of the top strip cleanup gadget to the entrance of the bottom strip cleanup gadget.

The jump sequence $\tilde{S}_i$ that allows the frog to traverse the strip cleanup gadget S_i has the following components:

- v *vertical selector sequences*; that allow the frog to choose an even row of the inner strip;
- 2 *horizontal hole sequences* that allow the frog to completely fill the two even rows of the inner strip that have an already visited cell;
- $v - 2$ *horizontal fill sequences* that allow the frog to completely fill the remaining $v - 2$ even empty rows of the inner strip.

The horizontal hole and fill sequences are embedded in the vertical selector sequences.

3.4 Horizontal sequences

We see how to build the two horizontal hole sequences that can be used to fill the lines of the edge gadgets that are unvisited or that contain an already visited cell.

Lemma 3.3 (Horizontal fill sequence)**.** *If $w = 2^k - 1$, starting from the line:*

$$E^w B^w E^w F E$$

we can force the frog to correctly visit and fill all the empty cells and finally jump on the rightmost cell; i.e. the final configuration is:

$$V^w B^w V^w V F$$

Proof. We can use the the following horizontal jump sequence:

$$\tilde{Z} = 3w, \overbrace{1, 1, \ldots, 1}^{w-1 \text{ times}}, w+1, \overbrace{1, 1, \ldots, 1}^{w-1 \text{ times}}, 2$$

It forces the frog to jump to the leftmost empty cell, fill the w empty cells of the first E^w block, jump to the second E^w block, fill it and finally jump on the rightmost cell. □

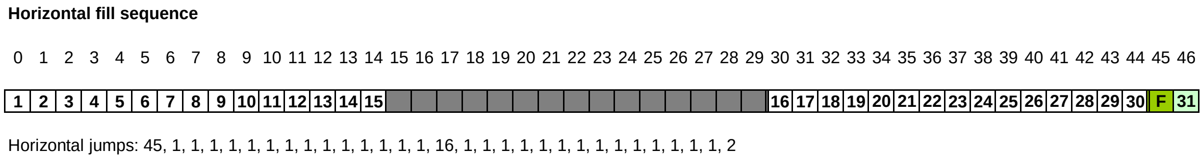


Figure 6: An horizontal fill sequence on an inner strip of width $w = 15$ ($k = 4$).

Figure 6 shows an horizontal fill sequence on an inner strip of width $w = 15$ ($k = 4, v = 8$). We can also build an horizontal sequence that completely fills an inner strip row that contains an already visited cell (i.e. leaves a "hole"). First we prove the following:

Lemma 3.4 (Binary jump sequence)**.** *If $w = 2^k - 1, v = 2^{k-1}$, $k \geqslant 2$, starting from the line:*

$$E^{v-1} F E^{v-1}$$

i.e. the frog is placed in the middle of an empty block of length w, we can force the frog to correctly fill the line and end in an arbitrary even position; (i.e. the final configuration is:

$$V^{a-1} F V^{w-a}$$

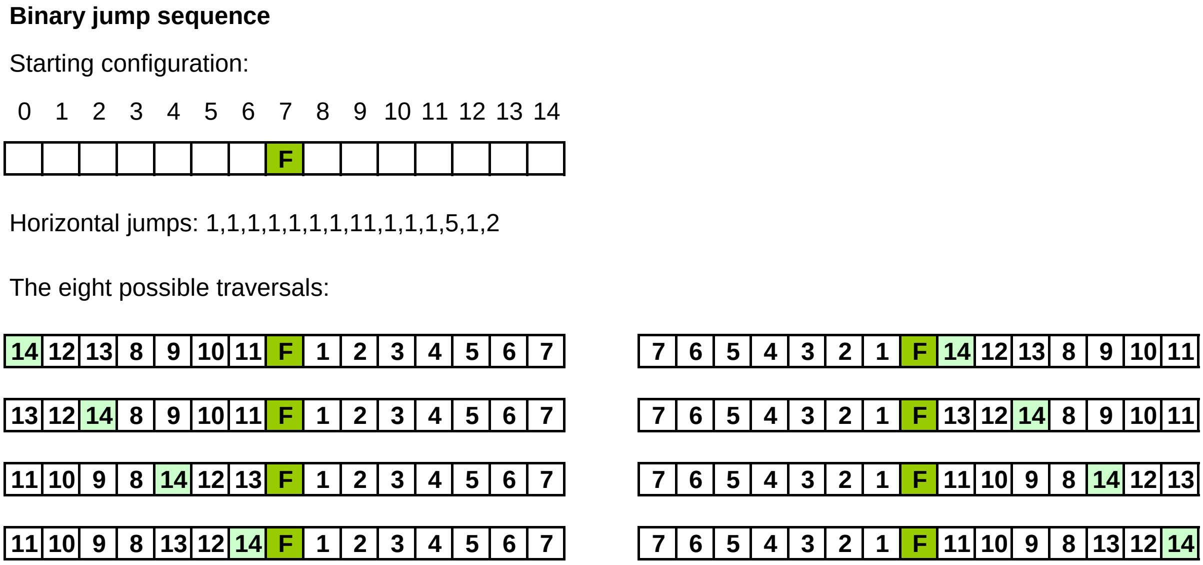
(where a is even) using a fixed binary jump sequence:

$$\tilde{U}_{k-1}, \tilde{U}_{k-2}, \ldots, \tilde{U}_2, \tilde{U}_1$$

where $\tilde{U}_j$ is the sequence of horizontal jumps: $\overbrace{1, 1, \ldots, 1}^{2^j-1 \textit{ times}}, 2^j + 2^{j-1} - 1.$

Proof. We can prove it by induction on k: when $k = 2$ the line is EFE, and the binary jump sequence is $\tilde{U}_1 = 1, 2$; so the frog can make the first jump left or right and then reach one of the two possible final even cells (VVF or FVV). Now, suppose that the statement holds for k; and the frog is placed in the middle of a $w' = 2^{k+1} - 1$ empty block ($E^{2^k} F E^{2^k}$). The first jump sequence is

$\tilde{U}_k = \overbrace{1, 1, \dots, 1}^{2^k-1 \text{ times}}, 2^k + 2^{k-1} - 1$; suppose that the target even cell is on the right half of the line, the frog can make $2^k - 1$ left horizontal jumps and completely fill the left half of the line ($FV^{2^k} E^{2^k-1}$); then the final (right) jump $2^k + 2^{k-1} - 1$ can lead it in the middle of the right half of the line that contains the target even cell ($V^{2^k} E^{2^k-1} F E^{2^k-1}$); and by induction hypothesis it can fill the right half and end in the target even cell. If the target even cell was on the left half of the line, the frog can simply invert the jump directions. Informally the line can be seen as a binary tree with the leaves corresponding to even cells; the sequence of jumps allows the frog to choose a half of the tree, fill it, jump to the other half, fill it and so on until the tree is completely visited. □



Binary jump sequence

Starting configuration:

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
							F							

Horizontal jumps: 1,1,1,1,1,1,1,11,1,1,1,5,1,2

The eight possible traversals:

14	12	13	8	9	10	11	F	1	2	3	4	5	6	7
13	12	14	8	9	10	11	F	1	2	3	4	5	6	7
11	10	9	8	14	12	13	F	1	2	3	4	5	6	7
11	10	9	8	13	12	14	F	1	2	3	4	5	6	7

7	6	5	4	3	2	1	F	14	12	13	8	9	10	11
7	6	5	4	3	2	1	F	13	12	14	8	9	10	11
7	6	5	4	3	2	1	F	11	10	9	8	14	12	13
7	6	5	4	3	2	1	F	11	10	9	8	13	12	14

Figure 7: A binary jump sequence on a line of width $w = 15$ ($k = 4$) and the jumps that place the frog on cells 0,2,4,6,8,10,12,14.

Figure 7 shows some examples of a binary jump sequence on a line of width $w = 15$ ($k = 4$). If we reverse all the sequences of jumps we can go from the final configuration of Lemma 3.4 to the initial configuration, so the following also holds:

Lemma 3.5 (Reverse binary jump sequence)**.** *If $w = 2^k - 1, v = 2^{k-1}$, $k \geqslant 2$ and the frog is placed on an arbitrary even cell of an empty line of length w ($E^{a-1} F E^{w-a}$), then the following fixed sequence of jumps that allows the frog to completely fill the line and end in the center cell ($V^{v-1} F V^{v-1}$):*

$$\tilde{U}_1^R, \tilde{U}_2^R, \dots, \tilde{U}_{k-2}^R, \tilde{U}_{k-1}^R$$

where $\tilde{U}_j^R$ is the reverse of the sequence $\tilde{U}_j$ of horizontal jumps:
$\tilde{U}_j^R = 2^j + 2^{j-1} - 1, \overbrace{1, 1, \ldots, 1}^{2^j-1 \text{ times}}$

Combining the sequences used in the previous two lemmas we can build the horizontal hole sequence:

Lemma 3.6 (Horizontal hole sequence)**.** *If $w = 2^k - 1, v = 2^{k-1}, k \geqslant 2$, starting from the line:*

$$E^w B^w E^w F E$$

we can force the frog to correctly visit and fill the line except one empty (or visited) cell in an arbitrary even position of the leftmost empty block and finally jump on the rightmost cell; i.e. the final configuration is:

$$(V^{a-1} E V^{w-a}) B^w V^w V F$$

where a is even.

Proof. We use the following sequence of horizontal jumps:

$$\tilde{H} = 2w + v, \tilde{U}_{k-1}, \tilde{U}_{k-2}, \ldots, \tilde{U}_2, 1, 2w, \tilde{U}_1^R, \tilde{U}_2^R, \tilde{U}_3^R, \ldots, \tilde{U}_{k-1}^R, v + 1$$

Where $\tilde{U}_j$ is the sequence of horizontal jumps: $\overbrace{1, 1, \ldots, 1}^{2^j-1 \text{ times}}, 2^j + 2^{j-1} - 1$; and $\tilde{U}_j^R$ is the reverse of $\tilde{U}_j$.

The first jump $(2w + v)$ forces the frog in the middle of the first E^w block, then it can follow the fixed sequence of jumps of Lemma 3.4 except for the very last jump; note that the sequence of Lemma 3.4 ends with $\ldots, \tilde{U}_2, \tilde{U}_1$ while the sequence here is $\ldots, \tilde{U}_2, 1, \ldots,$ i.e. it misses the final horizontal jump of $\tilde{U}_1$ that has length 2 ($\Delta x = 2$). In this way we can reach a configuration in which an arbitrary even cell $(2x, \cdot)$ is left empty, and the frog is positioned at distance 2 from that cell, i.e. it is on the even cell $(2x+2, \cdot)$ or $(2x-2, \cdot)$. So it can jump on the second E^w block with the horizontal jump $2w$ and land on the even cell $(2x + 2 + 2w, \cdot)$ or $(2x - 2 + 2w, \cdot)$; and then it can completely fill the block reaching its center cell by using the fixed sequence of jumps of Lemma 3.5. Finally it can make a final $v + 1$ jump to reach the rightmost empty cell. □

Figure 8 show a possible traversal of the line $E^7 B^7 E^7 F E, (k = 3, w = 7, v = 4)$ that leaves a hole (corresponding to an already visited cell) in even position using the sequence of jumps defined in Lemma 3.6.

Figure 9 shows three horizontal hole sequences with the already visited cell in different positions on an inner strip of width $w = 15$ $(k = 4, v = 8)$.

Note that in both types of horizontal sequences (fill and hole), we can extend the first and last steps so that the frog can be positioned farther from the inner strips.

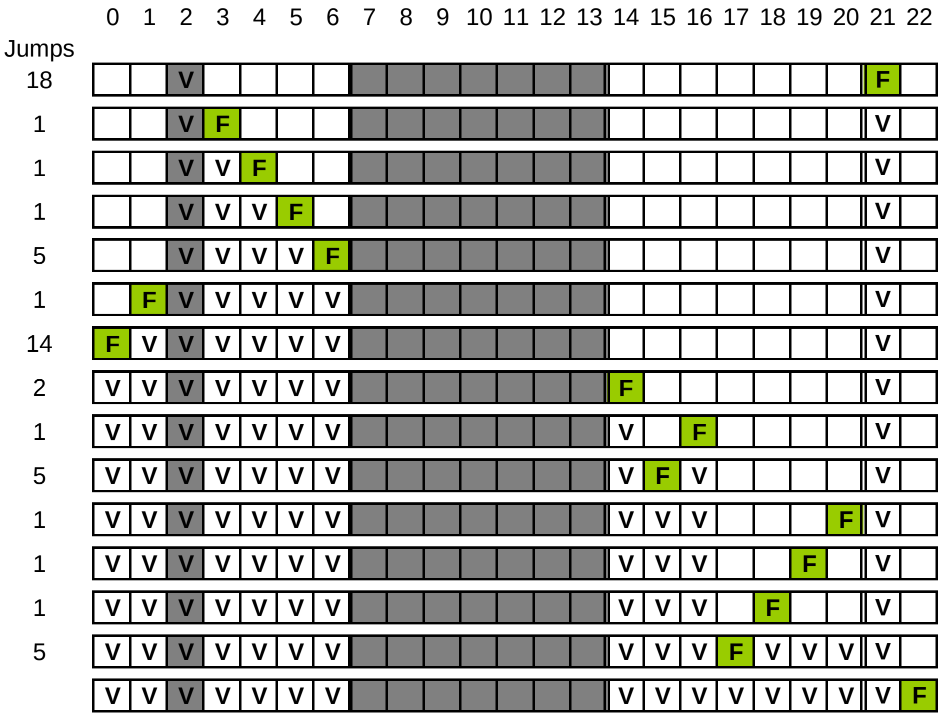


Figure 8: A possible horizontal hole sequence traversal that fills an inner strip row with an already visited cell.

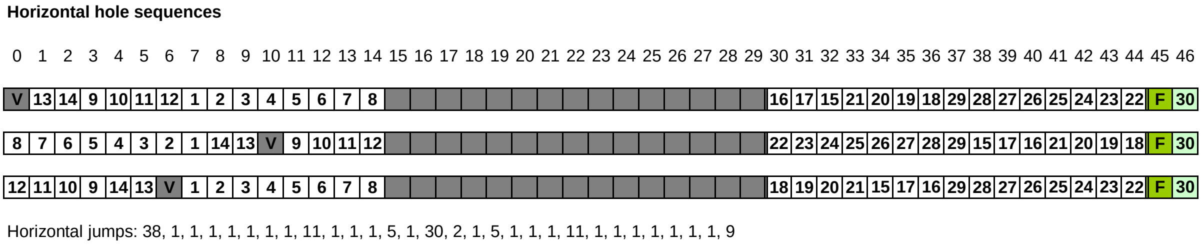

Horizontal hole sequences
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46
V 13 14 9 10 11 12 1 2 3 4 5 6 7 8 16 17 15 21 20 19 18 29 28 27 26 25 24 23 22 F 30
8 7 6 5 4 3 2 1 14 13 V 9 10 11 12 22 23 24 25 26 27 28 29 15 17 16 21 20 19 18 F 30
12 11 10 9 14 13 V 1 2 3 4 5 6 7 8 18 19 20 21 15 17 16 29 28 27 26 25 24 23 22 F 30
Horizontal jumps: 38, 1, 1, 1, 1, 1, 1, 1, 11, 1, 1, 1, 5, 1, 30, 2, 1, 5, 1, 1, 1, 11, 1, 1, 1, 1, 1, 1, 1, 9

Figure 9: Three horizontal hole sequences with the already visited cell in different positions and a horizontal fill sequence on an inner strip of width $w = 15$ $(k = 4, v = 8)$.

3.5 Vertical selector sequence

Using a similar sequence of the horizontal hole sequence, we can build a sequence of jumps that allows the frog, placed on the top of a $2v \times w$ empty area to vertically select a different even row, v times and finally exit the area on the cell at the bottom-left.

Lemma 3.7 (Vertical selector sequence)**.** *If* $w = 2^k - 1, v = 2^{k-1}, k \geqslant 2$*, starting from the* $2v \times w + 2$ *configuration:*

row #	*cells*
$l_i + 2w - 1:$	$FB^{2v-2}B$
$l_i + 2w$*:*	E^{2v}
	$\dots$
	E^{2v}
$l_i + 3w:$	$EB^{2v-2}E$

we can allow the frog to choose a different even row v *times, and reach the configuration:*

row #	*cells*
$l_i + 2w - 1:$	$VB^{2v-2}B$
$l_i + 2w$*:*	V^{2v}
	$\dots$
	V^{2v}
$l_i + 3w:$	$FB^{2v-2}V$

Proof. The sequence is:

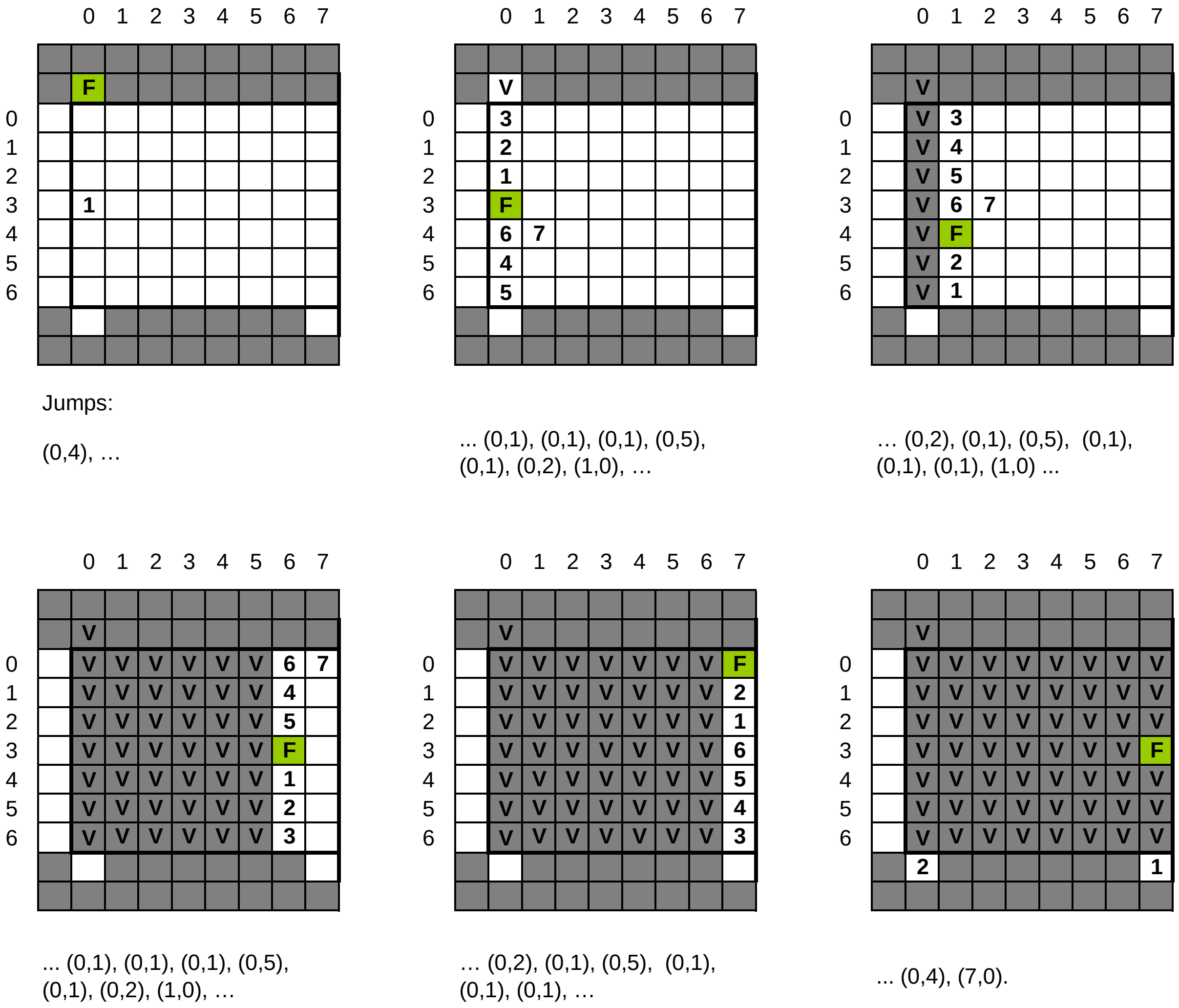
$$\tilde{V} = (0, v), \tilde{V'}_1, \overline{(1, 0)}, \tilde{V}''_1, (1, 0), \dots, (1, 0), \tilde{V'}_v, \overline{(1, 0)}, \tilde{V}''_v, (0, v), (2v - 1, 0)$$

Where:

$$\tilde{V'}_i = \tilde{U}_{k-1}, \tilde{U}_{k-2}, \dots, \tilde{U}_2, \tilde{U}_1$$
$$\tilde{V''}_i = \tilde{U}^R_1, \tilde{U}^R_2, \tilde{U}^R_3, \dots, \tilde{U}^R_{k-1}$$

$\tilde{U}_j$ is the sequence of vertical jumps: $\overbrace{1, 1, \dots, 1}^{2^j-1 \text{ times}}, 2^j + 2^{j-1} - 1$; and $\tilde{U}^R_j$ is the reverse of $\tilde{U}_j$. In other words we are using the sequence of jumps of Lemma 3.4 and Lemma 3.5, but here the jumps are vertical. The (vertical) sequence $\tilde{U}_j$ selects a cell on an even row, then, after an horizontal right jump $(1, 0)$, the reverse $\tilde{U}^R_j$ allows the frog to reach the center row again. □

Figure 10 shows the possible traversal of a 7×8 area using the vertical selector sequence.



Jumps:

(0,4), ...

... (0,1), (0,1), (0,1), (0,5), (0,1), (0,2), (1,0), ...

... (0,2), (0,1), (0,5), (0,1), (0,1), (0,1), (1,0) ...

... (0,1), (0,1), (0,1), (0,5), (0,1), (0,2), (1,0), ...

... (0,2), (0,1), (0,5), (0,1), (0,1), (0,1), ...

... (0,4), (7,0).

Figure 10: An example of the vertical selector sequence on a 7 × 8 area ($w = 7, v = 4$).

3.6 Linking the gadgets

The horizontal inner jumps $\overline{(1,0)}$ in the vertical selector sequences $\tilde{V}$ can be replaced by an horizontal hole sequence or by an horizontal fill sequence (extending their first and last jump). So we can build a sequence of jumps that allows the frog to: *a)* select 2 even rows of the inner strip with a visited cell and fill them; *b)* select the remaining $v - 2$ empty even rows and fill them.

The complete inner strip sequence S_i is:

$$\begin{aligned}\tilde{S}_i = \quad & (0,v), \tilde{V'}_1, \tilde{H}, \tilde{V}_1'', (1,0), \tilde{V'}_2, \tilde{H}, \tilde{V}_2'', (1,0), \\ & \tilde{V'}_3, \tilde{Z}, \tilde{V}_3'', (1,0), \ldots, \tilde{V'}_v, \tilde{Z}, \tilde{V}_v'', (0,v), (2v-1,0)\end{aligned}$$

Figure 11 shows a possible traversal of the strip cleanup gadget of Figure 5.

Finally we can link together the cleanup sequences $\tilde{C}_i$; first we add an empty cell at coordinates $(3w, y_t)$ that allows the frog to jump from the target cell (x_t, y_t) to the

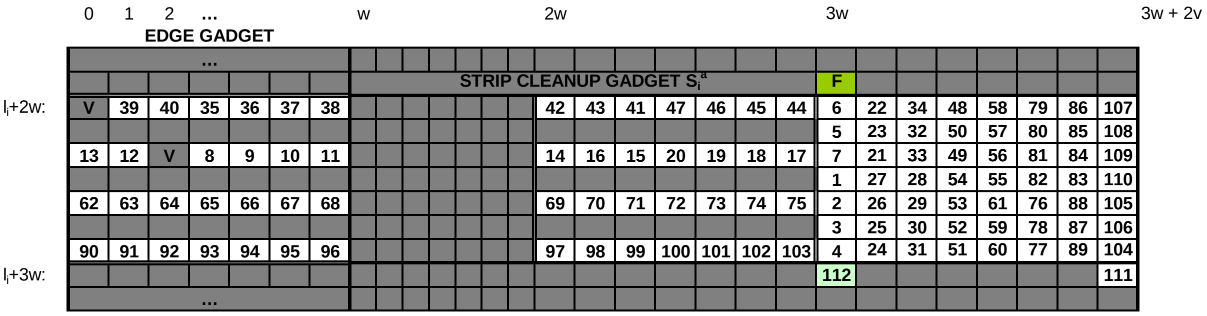


Figure 11: A possible traversal of the strip cleanup gadget of Figure 5, that combines the vertical selector sequence with the horizontal sequences.

entrance cell of the top inner strip gadget S_i^a; then we add a vertical jump from gadget C_i to gadget C_{i+1}, $i = 1, \ldots, n-2$:

$$\begin{aligned}\tilde{C} = \ & (3w - x_t, 0), (0, l_1 + 2w - 1 - y_t), \tilde{C}_1, (0, l_2 + 2w - 1 - l_1 + 5w), \ldots \\ & \ldots, (0, l_{n-1} + 2w - 1 - l_{n-2} + 5w), \tilde{C}_{n-1}\end{aligned}$$

Theorem 3.8. *The* Crazy Frog Puzzle *is* NP*–complete.*

Proof. The problem is NP–hard: by construction if the original grid graph problem has a solution, then there is a valid traversal of the graph area and the frog can complete the board using the cleanup gadget. If the board has a valid traversal, as seen above the sequences of nodes traversed in the graph area corresponds to a Hamiltonian path in G from s to t. The instance of the CFP can be constructed in polynomial time because the size of the whole board is $3w + 2v \times 7o_n w$, where $w = 2^k - 1, v = 2^{k-1}, o_n = 2n - 1$, and $2^k - 1 < 8n$; so the board can be constructed in time $O(n^2)$.

The problem is in NP because a solution can easily be checked in polynomial time. □

4 One dimensional variant

It is easy to see that even if we restrict the board to be one dimensional the problem remains NP–complete.

Theorem 4.1. *The Crazy Frog Puzzle remains* NP*–complete even if restricted to 1-Dimensional boards (*1-D Crazy Frog Puzzle*), i.e. boards of size $w \times 1$.*

Proof. The immediate reduction is from the Crazy Frog Puzzle: given an instance of the CFP, i.e. a $n \times n$ partially filled board and a sequence of jumps: $\Delta_1, \Delta_2, \ldots, \Delta_m$ check the sequence and if there is a jump $(\Delta x_i, \Delta y_i)$ such that $dx_i \geqslant n$ or $dy_i \geqslant n$ reject (the jump brings the frog outside of the board). Otherwise expand it to size $3n \times 3n$ adding a border of blocked cells of width n in all the four directions. Then build an equivalent one dimensional crazy frog puzzle of size $(3n)^2$ putting the lines of the expanded board side by side (cells (x, y) is mapped to cell $x + 3ny$) and converting every bidimensional jump $(\Delta x_i, \Delta y_i)$ to the one dimensional jump: $\Delta x_i + 3n * \Delta y_i$. By construction every one dimensional $\Delta x_i + 3n * \Delta y_i$ jump will lead the frog from $x_0 + 3ny_0$ to the cell $(x_0 \pm \Delta x_i) + 3n * (y_0 \pm \Delta y)$ that corresponds to the original bidimensional cell $(x_0 \pm \Delta x_i, y_0 \pm \Delta y_i)$. Borders prevent the frog to make moves that are invalid in the corresponding bidimensional configuration; for example using a left jump $x + 3n * 0$ from cell $0 + 3n * 1$ (on the left border in the bidimensional board) to reach another part of the one dimensional board. □

Figure 12 shows a simple example of a 3 × 3 CFP instance transformed to a 1-Dimensional CFP board of length 81.

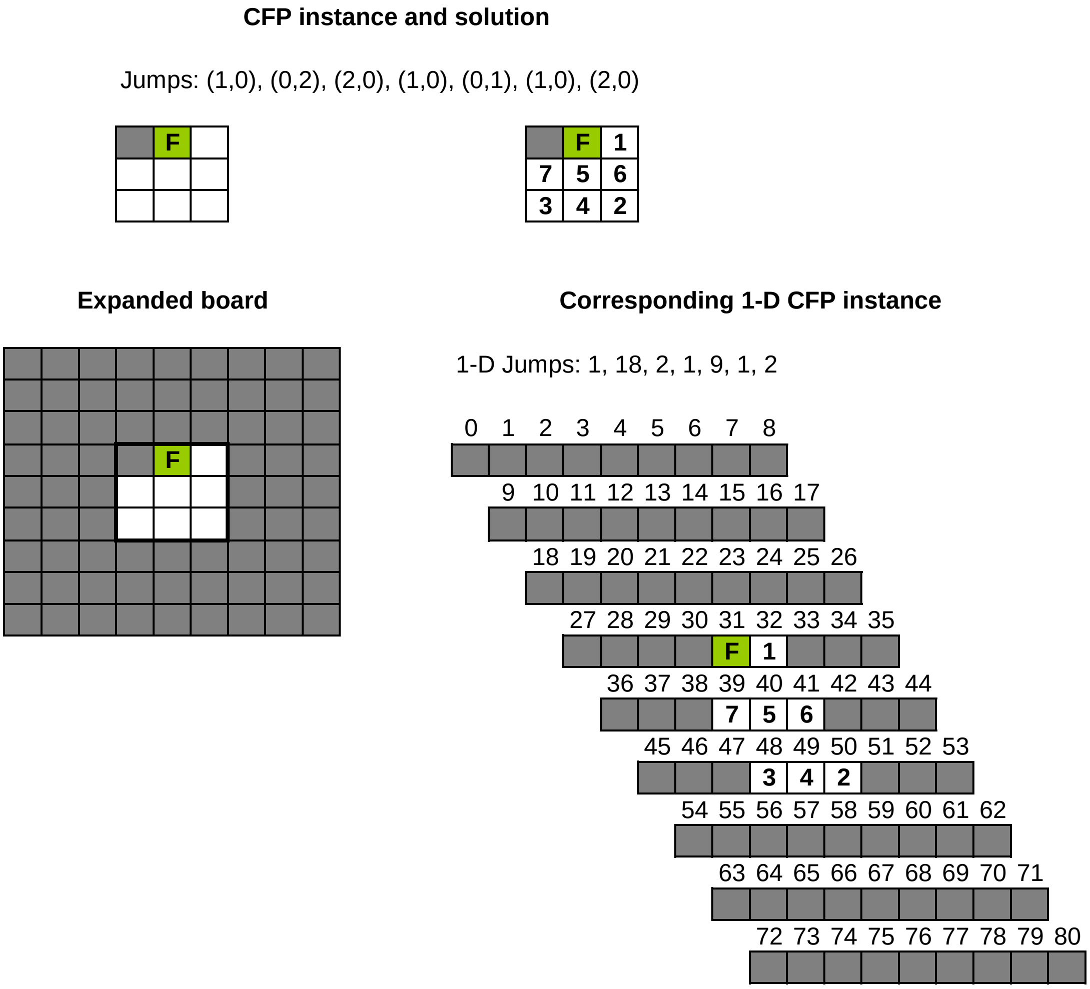


Figure 12: A simple example of a 3 × 3 CFP instance transformed to a 1-Dimensional CFP board of length 81.

We can also fix the starting position of the frog:

Lemma 4.2. *Without loss of generality we can assume that in the 1-D CFP instance the frog is placed on the leftmost cell.*

Proof. Suppose that the 1-D board is $R = (E|B)^a F(E|B)^b$ and the sequence of jumps is: $J_1, J_2, \ldots, J_m$. We can extend the board with a cell on the left that will be the new starting position of the frog: $S' = FR$ and add a jump to the sequence: $(a+1, 0), J_1, J_2, \ldots, J_m$. The first jump, that must be towards the right, places the frog on the original position. □

5 Permutation reconstruction from differences

We first prove that 1-D CFP is hard even if the initial board is empty.

Lemma 5.1. *1-D CFP remains* NP*-complete even if the initial board is empty.*

Proof. Given an instance of the 1-D CFP, i.e. a configuration $R = F\{B, E\}^{n-1}$ and a sequence $\tilde{J} = J_1, J_2, \ldots, J_m$ of m jumps; suppose that R contains $p = n - m - 1$ blocked cells at coordinates $x_1, x_2, \ldots, x_p$; let $d_1 = x_1$, $d_i = x_i - x_{i-1}, i = 2, 3, \ldots, p$, $d_{p+1} = n - x_p$. We start with an empty line of length $2n+1$: $R' = FE^nE^n$ and extend the jump sequence in this way:

$$\tilde{J}' = d_1 + 1, d_2, \ldots, d_p, d_{p+1}, \overbrace{1, 1, \ldots, 1}^{n-1 \text{ times}}, 2n - 1, \tilde{J}$$

(note that $|\tilde{J}'| = 2n$). The $n-1$ steps forces a sequence of n contiguous visited cells, and it must be aligned with the rightmost part of the board otherwise the frog will never be able to reach that cell during jumps J_i, because $J_i < n$ (otherwise the original instance doesn't have a solution). But, by construction, the only way to align it to the right is to make the d_i jumps towards the right, and they recreate exactly the p blocked cells of R. The jump $2n - 1$ forces the frog to the second cell, which is also the starting cell of the original configuration R. The modified instance with the empty board has a solution if and only if the original instance has a solution. □

Figure 13 shows a reduction from 1-D CFP to 1-D CFP with initial empty board.

Definition 5.2 (Permutation Reconstruction from Differences).

Input: a set of $n-1$ distances $a_1, a_2, \ldots, a_{n-1}$ with $a_i > 0$
Question: does there exist a permutation $\pi_1, \ldots, \pi_n$ of the integers $[1..n]$ such that $|\pi_{i+1} - \pi_i| = a_i$, $i = 1, \ldots, n-1$?

Note that if $\pi_1, \ldots, \pi_n$ is a valid solution, then the *mirrored* sequence $n - \pi_1 + 1, n - \pi_2 + 1, \ldots, n - \pi_n + 1$ is also a valid solution. Figure 14 shows an example of a permutation reconstruction from differences problem. The reduction from the 1-D CFP with initial empty board to the permutation reconstruction from differences problem is straightforward.

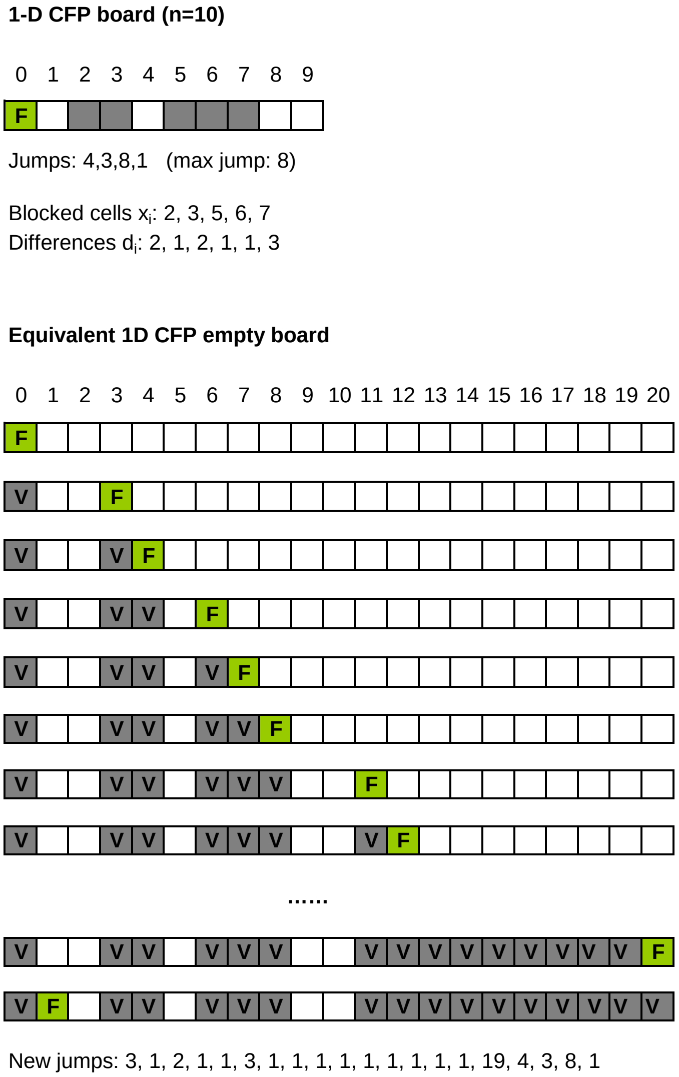

1-D CFP board (n=10)
0 1 2 3 4 5 6 7 8 9
F
Jumps: 4,3,8,1 (max jump: 8)
Blocked cells x_i: 2, 3, 5, 6, 7
Differences d_i: 2, 1, 2, 1, 1, 3
Equivalent 1D CFP empty board
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
F
V
......
New jumps: 3, 1, 2, 1, 1, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 19, 4, 3, 8, 1

Figure 13: An example of reduction from 1-D CFP to 1-D CFP with initial empty board.

Theorem 5.3. Permutation Reconstruction from Differences *(PRD) is* NP*–complete.*

Proof. Given an instance of the 1-D CFP with initial empty board of length n and jumps $J_1, J_2, \dots, J_{n-1}$ it has a solution if and only if a valid permutation of $[1..n+1]$ can be reconstructed from differences $a_1 = n$ and $a_i = J_{i-1}, i = 2, \dots, n$.

($\Rightarrow$) The frog visits all the cells of the board exactly once, so its positions $x_i,\ i = 1, \dots, n$ during the traversal (where $x_1 = 0$ is its starting position) is a permutation of $[0..n-1]$ and it can be transformed to a permutation of $[1..n+1]$ setting $\pi_1 = n+1$ and $\pi_i = x_{i-1} + 1,\ i = 2, \dots, n+1$.

($\Leftarrow$) Suppose that $\pi_1, \dots, \pi_{n+1}$ is a valid permutation that satisfy the difference constraints; we have that π_1 must be 1 or $n+1$ because the first difference $a_1 = n$. Suppose that $\pi_1 = n+1$, then $\pi_2 = 1$ and $\pi_2 - 1, \dots, \pi_{n+1} - 1$ are a valid sequence of positions

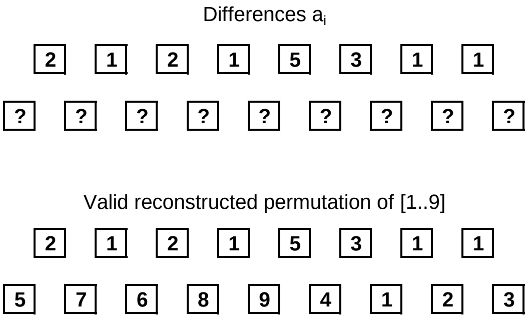


Figure 14: An instance of the Permutation Reconstruction from Differences problem: the differences a_i are $(2,1,2,1,5,3,1,1), n = 9$ and the reconstructed permutation is $(5,7,6,8,9,4,1,2,3)$; the mirrored valid permutation is $(5,3,4,2,1,6,9,8,7)$.

of the frog because $|(\pi_3 - 1) - (\pi_2 - 1)| = J_1, |(\pi_4 - 1) - (\pi_3 - 1)| = J_2, \ldots$, and they represent a valid solution to the 1-D CFP instance, too: the sign of jump J_i is positive if $\pi_{i+2} > \pi_{i+1}$, negative otherwise. If $\pi_1 = 1$ we can simply mirror the values replacing every π_i with $\pi'_i = (n+1) - \pi_i + 1$ because their absolute differences don't change.

□

6 Conclusion

We proved the hardness of a simple problem on permutations that could shed light on other combinatorial or arithmetic open problems. For example there could be a correlation with the graceful labeling problem, indeed if the a_i are themselves a permutation of $[1..n]$ (all values are distinct) then the permutation reconstruction from differences (PRD) problem is equivalent to verify that the sequence is a graceful labeling of the line of $n+1$. So it would be interesting to study some restricted versions of the PRD problem; for example what is its complexity if the differences are from a finite set of size k. As an intermediate step we introduced a new addictive puzzle game that we hope will be soon playable online or as a smartphone application.

References

[1] Erik D. Demaine and Robert A. Hearn. Playing games with algorithms: Algorithmic combinatorial game theory. In Michael H. Albert and Richard J. Nowakowski, editors, *Games of No Chance 3*, volume 56 of *Mathematical Sciences Research Institute Publications*, pages 3–56. Cambridge University Press, 2009.

[2] Alon Itai, Christos H Papadimitriou, and Jayme Luiz Szwarcfiter. Hamilton paths in grid graphs. *SIAM Journal on Computing*, 11(4):676–686, 1982.

[3] Graham Kendall, Andrew J. Parkes, and Kristian Spoerer. A survey of np-complete puzzles. *ICGA Journal*, 31(1):13–34, 2008.

[4] Maria Monks. Reconstructing permutations from cycle minors. *Electr. J. Comb.*, 16(1), 2009.

[5] Mariana Raykova. Permutation reconstruction from minors. *Electr. J. Comb.*, 13(1), 2006.

[6] Michael Sipser. *Introduction to the theory of computation.* PWS Publishing Company, 1997.

[7] Rebecca Smith. Permutation reconstruction. *Electr. J. Comb.*, 13(1), 2006.

[8] S.M. Ulam. *A Collection of Mathematical Problems.* Interscience, New York, NY, USA, 1960.

Addendum – added 9th December 2015

Please note that to the best of our knowledge the problem was first proposed by Mohammad Al-Turkistany in a question posted on MathOverflow.net, a question and answer site for professional mathematicians.

[9] Mohammad Al-Turkistany. How hard is reconstructing a permutation from its differences sequence? http://mathoverflow.net/q/135968 (version: 2013-07-07).