A Direct Approach to Searching with Lies *

Vladimir B. Balakirsky

The author is currently with the
Electrical Engineering Department,
Eindhoven University of Technology,
P.O.Box 513, 5600 MB Eindhoven, the Netherlands
E-mail addresses :
v.b.balakirsky@ele.tue.nl
vbal@vbal.spb.su

Abstract We introduce a tree structure in such a way that the nodes "contain" integer-valued vectors, and the vector contained in each node is uniquely determined by the arithmetic operation applied to the vectors contained in its sons. Relationships between the distribution of terminal nodes of the tree and the vector contained in the root relates trees to the coverings of the Hamming space by the sets whose cardinalities coincide with the sizes of the Hamming balls. Application of this construction to the searching problem when one of M integers is unknown and has to be found with n questions under the assumption that e lies are allowed in the answers represents many of the known results as corollaries from the properties of the trees and leads to a new upper bound on the number of questions that differs from the known lower bound by $3e$.

Key words :
Search for a pattern, Trees, Coverings

---

*The work was supported by the SFB-343, Universität Bielefeld, Germany

1 Introduction

Let us consider the following game (see Fig.1). Suppose that there is a board containing two rows and $e+1$ columns numbered by integers $0,\dots,e$ and a player who is given $c_0+\dots+c_e$ coins. The player distributes c_j coins among the rows of the j-th column, $j=0,\dots,e$. Then another player, called the devil, puts a pointer either on the first or the second row. After that all coins of the selected row are shifted by 1 position to the left; the coins of the leftmost column are removed from the board. These actions represent the first move of the game. As a result of the move, the player receives new numbers of coins in each column, and the participants make the second move. The game is finished when the board is either empty, or contains only 1 coin, no matter in which column. The goal of the player is to finish the game as fast as possible, while the devil tries to play as long as possible. Some positions that are attained in the game when the player is initially given 4 coins in column 0 and 1 coin in column 1 are shown in Fig.1, where the pointer drawn in the middle of the board means that the devil does not distinguish between the variants. Note that if the player distributes the 4 coins of column 0 into equal halves at the first move, then the devil can make the game consist of 4 moves. However, if these coins are distributed in the proportion $1:3$ and the proportion in column 1 is $1:0$, then the game is finished after at most 3 moves whatever the devil does.

The game described above relates to the problem of the search for an unknown integer m chosen from the set $\{1,\dots,M\}$, where M is given. Suppose that this integer is chosen by the "devil". We ask him a question, i.e., present some subset $\mathcal{M}$ of the set $\{1,\dots,M\}$ and receive the answer "yes" if $m\in\mathcal{M}$ and "no" if $m\notin\mathcal{M}$. After that we form the next question based on this and previous answers. Suppose that the devil is allowed to lie by answering "yes" if $m\notin\mathcal{M}$ and "no" if $m\in\mathcal{M}$, but the number of lies is not greater than a given number e. Furthermore, he is allowed to change m during the searching procedure in such a way that the integer he currently has in mind satisfies the rules of the game, i.e., he lied for this integer not more than e times. The positions presented in Fig.1 correspond to the case when $M=5$, $e=1$, the question at time instant -1 was : is $m\in\{1\}$, and the answer was "yes". As a result, we count 1 and 0 lies if $m\in\{2,3,4,5\}$ and $m=1$, respectively, and generate the vector $\mathbf{c}_0=(4,1)$. The distribution of coins at time instant 0 in the proportion 1 : 3 and 1 : 0 in the 0-th and 1-st columns, respectively, may correspond to the question : is $m \in \{1, 2\}$, and the answer "yes" leads to the conclusion that if $m = 1$, then the devil did not lie in his answers for all $t \leq 0$ and if $m = 2$ then he lied once. The integers 3, 4, 5 can be deleted from the searching domain because if one of them was chosen, then the devil lied twice. The coins can be numbered in advance, and this procedure exactly coincides with the game above. The coin remaining on the board determines the integer.

This problem was considered by Berlekamp [4] in the context of data transmission over a binary symmetric channel with noiseless feedback : one of M messages should be delivered to the receiver by sending a binary sequence over a channel where at most e bits can be changed. Berlekamp presented the problem as a search problem, proved a lower bound on the number of questions, and analyzed searching strategies in the asymptotic case when $M = 2^{\rho n}$, $e = \alpha n$, ρ and α are constants, and $n \to \infty$. The study was continued in many papers, where the authors attribute the statement of the problem to Ulam who published a question concerning the search for an integer from one to one million : "Now suppose one were allowed to lie once, or twice, then how many questions would one need to get the right answer?" ([13], p.281). Solutions to the search problem for arbitrary M, which include specific searching strategies that allow us to get the integer with the minimal number of questions, were obtained by Pelc for $e = 1$ [10], Guzicki for $e = 2$ [8], and Deppe for $e = 3$ [6]. An asymptotic analysis of the case of arbitrary e was presented by Spencer [12]. Searching strategies and possible extensions of the problem were considered by Aigner [1], Czyzowicz, Mundici, and Pelc [5], Lawler and Sarkissian [9], and other authors.

The basic idea of our research is the introduction of a tree structure containing both the topology and the arithmetic in such a way that the results of analysis of searching procedures can be derived as corollaries from the properties of that structure. By the "arithmetic" we mean that each node of a tree "contains" a certain integer-valued vector of length $e + 1$ and the vector contained in the parent of any two nodes can be calculated as a function of the vectors contained in these nodes. This function is denoted by $[+]$ since it is similar to the addition of the integers in decimal when the result at the current position is formed based on the corresponding components of the terms and the carry coming from the neighboring position. It turns out that there is a one-to-one mapping between the vector $\mathbf{c}$ contained in the root of the tree and the vector $\mathbf{k}$ specifying the distribution of terminal nodes, and this mapping is well-known in the theory of partitions in relation to the Simon Newcomb's problem [2]. The new point is the interpretation of this transformation as a possibility to control the addition (operation with the carry symbols) used to construct the vector $\mathbf{c}$ by the merging (component-wise operation) used to construct the vector $\mathbf{k}$, and conclusions of this type are important for a possible increasing of the speed of computer computations [3]. Furthermore, if the root of the tree is located at level n, then both the linear combination of the sizes of the Hamming balls in the space $\{0,1\}^n$ having the radii $0,\ldots,e$ taken with the coefficients $c_j, j = 0,\ldots,e$, and the linear combination of the powers of 2 taken with the coefficients $k_i, i = 0,\ldots,e$, is equal to 2^n; thus, we relate the equation describing a covering of the Hamming space to the necessary condition for existence of a binary tree with the fixed distribution of terminal nodes. The use of this observation for the searching problem leads to the known Hamming–type lower bound on the number of questions derived from the Kraft inequality. Another application of our structure concerns specific searching strategies constructed from the trees by the so-called balanced partitioning. If $e = 1$, then the resulting strategy gives the minimal possible number of questions; for $e > 1$ the strategy is sub-optimal because of some problems arising with the partitions of small numbers (a more detailed discussion of this point is left for another paper). Introduction of some special steps to this procedure allows us to simplify the analysis and present an explicit construction of the searching strategies for which the necessary number of questions differs from the known lower bound by $3e$. Hence, if $M = 2^{n\rho}$ and e is fixed, then we claim that $\rho \to 1$, as $n \to \infty$, while the Schalkwijk's lower bound on the code length for data transmission over a binary channel with feedback [11] guarantees the parameter $\rho < 1$ (more precisely, ρ can be expressed using the Fibonacci numbers). The issue of our considerations, as we think about them, is the point that the structure obtained by crossing the topology and the algebra can serve as a basis for constructing a theory of the search.

The paper is organized as follows. We begin with the statement of the searching problem in Section 2 since this can be helpful for reading the formal considerations of Sections 3,4. Applications of the tree structure to this problem are presented in Section 5. The proofs of all results are given in the Appendix (the main text only indicates the ideas of the proofs) because they are very simple and the reader is assumed to carry them out himself using the Appendix only for some details.

2 Formal statement of the problem

Let $e \geq 0$ be given and let $\mathbf{c} = (c_0, \ldots, c_e)$ be an integer-valued vector with non-negative components. The all-zero vector of length $e+1$ will be denoted by $\mathbf{0}$ and the vector containing 1 in the i-th position and 0's in all other positions will be denoted by $\mathbf{1}_i$, $i = 0, \ldots, e$. Let

$$\mathcal{D}_e(\mathbf{c}) = \left\{ (\delta_0, \ldots, \delta_e) \notin \{-\mathbf{c}, \mathbf{c}\} : \;\; \delta_j \in \mathcal{D}(c_j), \; j = 0, \ldots, e \right\},$$

where $-\mathbf{c} = (-c_0, \ldots, -c_e)$ and

$$\mathcal{D}(c) = \left\{ \; \delta : \;\; \delta \equiv c, \; |\delta| \leq c \; \right\}, \;\; \text{for all} \;\; c \geq 0. \tag{1}$$

Hereafter, the relation $x \equiv y$ is understood as $x \equiv y \pmod 2$ for all integers x and y.

For all $\boldsymbol{\delta} = (\delta_0, \ldots, \delta_e) \in \mathcal{D}_e(\mathbf{c})$, let $\mathbf{a}(\mathbf{c}|\boldsymbol{\delta}) = (a_0, \ldots, a_e)$ and $\mathbf{b}(\mathbf{c}|\boldsymbol{\delta}) = (b_0, \ldots, b_e)$ be the vectors with components

$$a_j = \frac{c_j + \delta_j}{2} + \frac{c_{j+1} - \delta_{j+1}}{2}, \;\; b_j = \frac{c_j - \delta_j}{2} + \frac{c_{j+1} + \delta_{j+1}}{2}, \tag{2}$$

where $j = 0, \ldots, e$ and we assume that $c_{e+1} = \delta_{e+1} = 0$. Let us introduce an integer-valued function N_e as follows :

$$N_e(\mathbf{1}_0) = \ldots = N_e(\mathbf{1}_e) = 0 \tag{3}$$

and

$$N_e(\mathbf{c}) \;\; = \;\; 1 + \min_{\boldsymbol{\delta} \in \mathcal{D}_e(\mathbf{c})} \max \left\{ N_e(\mathbf{a}(\mathbf{c}|\boldsymbol{\delta})), N_e(\mathbf{b}(\mathbf{c}|\boldsymbol{\delta})) \right\}, \quad \text{for all} \;\; \mathbf{c} \notin \{\mathbf{0}, \mathbf{1}_0, \ldots, \mathbf{1}_e\}. \tag{4}$$

Furthermore, for all $n = 0, 1, \ldots,$ let

$$\mathcal{R}_e^{(n)} = \Big\{ \mathbf{c} : \ N_e(\mathbf{c}) = n \Big\} \tag{5}$$

and

$$\mathcal{R}_e^{(n)*} = \Big\{ \mathbf{c} : \ N_e(\mathbf{c}) = n, \ N_e(\mathbf{c} + \mathbf{1}_0) = n + 1 \Big\}. \tag{6}$$

The sets $\mathcal{R}_1^{(0)*}, \ldots, \mathcal{R}_1^{(5)*}$ are presented in Table 1.

The notation above arises from the following searching procedure. Suppose that the questioner has constructed $e + 1$ pairwise disjoint subsets $\mathcal{M}_0, \ldots, \mathcal{M}_e$ of the set $\{1, \ldots, M\}$ (initially $\mathcal{M}_0 = \ldots = \mathcal{M}_{e-1} = \emptyset$ and $\mathcal{M}_e = \{1, \ldots, M\}$) in such a way that $\mathcal{M}_j$ consists of c_j integers having the property : if the unknown integer $m \in \{1, \ldots, M\}$ belongs to $\mathcal{M}_j$, then the responder lied exactly $e - j$ times in his previous answers. The questioner assigns a vector $\boldsymbol{\delta}^0 = (\delta_0^0, \ldots, \delta_e^0) \in \mathcal{D}_e(\mathbf{c})$ and presents a set $\mathcal{M}$ (question) consisting of $(c_0 + \delta_0^0)/2$ elements of $\mathcal{M}_0, \ldots,$ $(c_e + \delta_e^0)/2$ elements of $\mathcal{M}_e$. The answers $m \in \mathcal{M}$ and $m \notin \mathcal{M}$ generate subsets

$$\mathcal{M}_j' = \Big(\mathcal{M}_j \bigcap \mathcal{M}\Big) \bigcup \Big(\mathcal{M}_{j+1} \bigcap \overline{\mathcal{M}}\Big) \tag{7}$$

and

$$\mathcal{M}_j'' = \Big(\mathcal{M}_j \bigcap \overline{\mathcal{M}}\Big) \bigcup \Big(\mathcal{M}_{j+1} \bigcap \mathcal{M}\Big), \tag{8}$$

respectively, where $\overline{\mathcal{M}} = (\mathcal{M}_0 \bigcup \ldots \bigcup \mathcal{M}_e) \backslash \mathcal{M}$ denotes the set of integers under consideration that are not included in the question. The vector $\mathbf{c} = (|\mathcal{M}_0|, \ldots, |\mathcal{M}_e|)$ can be considered as the *state of the search*, which is partitioned into two new states $\mathbf{a}(\mathbf{c}|\boldsymbol{\delta}^0) = (|\mathcal{M}_0'|, \ldots, |\mathcal{M}_e'|)$ and $\mathbf{b}(\mathbf{c}|\boldsymbol{\delta}^0) = (|\mathcal{M}_0''|, \ldots, |\mathcal{M}_e''|)$. The parameter $N_e(\mathbf{c})$ defined in (3), (4) is the minimal number of questions in the minimax sense, i.e., if the current state of the search $\mathbf{c}$ belongs to the set $\mathcal{R}_e^{(n)}$, then the integer can be found after at most n questions when e lies are allowed in the answers.

The set $\mathcal{R}_e^{(n)*}$ defined in (6) will be essentially used in the analysis. Suppose $e = 1$ and we search for one of 4 integers 1, 2, 3, 4. One can easily check that the minimal number of questions which is necessary to find the integer, is equal to 5. Note that 8 is the 0-th component of the vector contained in

$\mathcal{R}_1^{(5)*}$ that contains 4 in the 1-st component (see Table 1), which means that we can solve a more difficult problem with 5 questions : the unknown integer belongs to the set $\{1, \dots, 12\}$ and if it is chosen from the set $\{1, \dots, 4\}$, then 1 lie is allowed in the answers; otherwise no lies are allowed. Hence, we can imagine that we have to solve the last problem and include *dummy* integers $5, \dots, 12$ into the questions. Certainly, these integers cannot reduce the number of questions, but it turns out that such a possibility simplifies the description of the searching strategy and the analysis (in some sense, a similar situation arises in the search without lies : if we search for one of 5 integers $1, \dots, 5$, then we can include integers 6, 7, 8 into the searching domain and always divide the current set into two equal halves).

The search problem under consideration can be formulated as follows : *given an integer M, find the minimal value of n such that the set $\mathcal{R}_e^{(n)}$ includes the vector, called the initial state of the search, whose last component is equal to M and all of whose other components are equal to 0; describe also the corresponding searching strategy, i.e., rules for constructing the vectors $\boldsymbol{\delta}^0$ for every state that can be reached from the initial state.*

3 Trees of class $\mathcal{T}_e^{(n)}$

We will associate integer-valued vectors of length $e+1$ with the nodes of a tree and say that each node "contains" a certain vector.

Definition 1 *Given $n, e > 0$, a rooted regular binary tree belongs to the class $\mathcal{T}_e^{(n)}$ if and only if*

$(T1)$ *the root is located at level n;*

$(T2)$ *all terminal nodes are located at levels $0, \dots, e$;*

$(T3)$ *each terminal node located at level $i \in \{0, \dots, e\}$ contains the vector $\mathbf{1}_i$;*

$(T4)$ *if two internal nodes having the same parent contain $\mathbf{a} = (a_0, \dots, a_e)$ and $\mathbf{b} = (b_0, \dots, b_e)$, then the parent contains the vector $\mathbf{c} = (c_0, \dots, c_e)$ such that*

$$c_j = \sum_{i=j}^{e} (-1)^{i+j}(a_i + b_i) = a_j + b_j - c_{j+1}, \quad j = 0, \dots, e, \qquad (9)$$

where $c_{e+1} = 0$;

$(T5)$ *each node of the tree contains a vector with non-negative components.*

An example of a tree of class $\mathcal{T}_e^{(n)}$ for $n = 4$ and $e = 1$ is given in Fig.2.

Obviously, the vectors contained in all internal nodes are uniquely determined by the topology of the tree. An important ingredient of the topology is the vector $\mathbf{k} = (k_0, \ldots, k_e)$, where k_i is the number of terminal nodes at level i (each of these nodes contains $\mathbf{1}_i$), $i = 0, \ldots, e$. This vector will be referred to as the *distribution of terminal nodes of the tree.*

Proposition 1 *The distribution of terminal nodes of any tree of class* $\mathcal{T}_e^{(n)}$ *belongs to the set*

$$\mathcal{K}_e^{(n)} = \left\{ (k_0, \ldots, k_e): \quad k_0, \ldots, k_e \geq 0, \quad \sum_{i=0}^{e} k_i 2^i = 2^n \right\}. \qquad (10)$$

We will present explicit formulas relating the distribution of terminal nodes of a tree of class $\mathcal{T}_e^{(n)}$ to the vector contained in the root, which are derived based on the following observations. Suppose there are two nodes at level $t-1$ containing the vectors $\mathbf{a}, \mathbf{b}$ and having a common parent at level t. Based on $(T4), (T5)$ we may introduce a graph with $e+1$ parts numbered by $0, \ldots, e$ in such a way that the j-th part has $a_j + b_j$ vertices; $j = 0, \ldots, e$ (Fig.3). Every vertex of the e-th part "absorbs" exactly one vertex of the $(e-1)$-st part, every remaining vertex of the $(e-1)$-st part "absorbs" exactly one vertex of the $(e-2)$-nd part, etc. The *total* number $F_{j,i}^{(t)}$ of vertices of the i-th part that are absorbed by every remaining vertex of the j-th part by level t is calculated as the sum $F_{j,i}^{(t-1)} + F_{j-1,i}^{(t-1)}$, for all $j \in \{1, \ldots, e\}$ and $i \in \{0, \ldots, j-1\}$, where the initial condition of the recursion is defined as $(F_{j,0}^{(j)}, \ldots, F_{j,e}^{(j)}) = \mathbf{1}_j$, $j = 1, \ldots, e$. These considerations allow us to express the distribution of terminal nodes of any tree of class $\mathcal{T}_e^{(n)}$ as a function of the components of the vector contained in the root. On the other hand, we can express the vector contained in the root of the tree as a function of the components of the distribution of terminal nodes.

Notice that the binomial coefficients $\binom{n}{i}$ can be recurrently introduced by the equations

$$\binom{0}{i} = \begin{cases} 0, & \text{if } i \neq 0, \\ 1, & \text{if } i = 0, \end{cases} \qquad \binom{n}{0} = 1, \quad n = 1, 2, \ldots, \tag{11}$$

and, for all $n \geq 1$,

$$\binom{n}{i} = \binom{n-1}{i} + \binom{n-1}{i-1}, \quad i = 1, 2, \ldots \tag{12}$$

We will use another recursion; namely, let us introduce a function $\begin{bmatrix} n \\ i \end{bmatrix}$ in such a way that

$$\begin{bmatrix} 0 \\ i \end{bmatrix} = \begin{cases} 0, & \text{if } i \neq 0, \\ 1, & \text{if } i = 0, \end{cases} \qquad \begin{bmatrix} n \\ 0 \end{bmatrix} = 1, \quad n = 1, 2, \ldots, \tag{13}$$

and, for all $n \geq 1$,

$$\begin{bmatrix} n \\ i \end{bmatrix} = \begin{bmatrix} n \\ i-1 \end{bmatrix} + \begin{bmatrix} n-1 \\ i \end{bmatrix}, \quad i = 1, 2, \ldots \tag{14}$$

Proposition 2 *For all $n, i \geq 0$ such that $n + i \geq 1$,*

$$\begin{bmatrix} n \\ i \end{bmatrix} = \binom{n+i-1}{i}. \tag{15}$$

Furthermore,

$$\sum_{i=0}^{j} \begin{bmatrix} n-j \\ i \end{bmatrix} 2^{j-i} = \sum_{i=0}^{j} \binom{n}{i}, \quad \textit{for all } j \in \{0, \ldots, n\}. \tag{16}$$

Theorem (duality formulas for the trees of class $\mathcal{T}_e^{(n)}$: the distribution of terminal nodes versus the vector contained in the root).

1. *Any tree of class $\mathcal{T}_e^{(n)}$, whose root contains a given vector* $\mathbf{c}$, *has the distribution of terminal nodes* $\mathbf{k} \in \mathcal{K}_e^{(n)}$ *such that*

$$k_i = \sum_{j=i}^{e} \left[\begin{array}{c} n-j \\ j-i \end{array} \right] c_j, \quad i = 0, \dots, e. \tag{17}$$

2. *The root of any tree of class $\mathcal{T}_e^{(n)}$ having a given distribution of terminal nodes* $\mathbf{k} \in \mathcal{K}_e^{(n)}$ *contains the vector* $\mathbf{c}$ *such that*

$$c_j = \sum_{i=j}^{e} (-1)^{i+j} \left[\begin{array}{c} n-i \\ i-j \end{array} \right] k_i, \quad j = 0, \dots, e. \tag{18}$$

Corollary *A tree of class $\mathcal{T}_e^{(n)}$ whose root contains a given integer-valued vector* $\mathbf{c}$ *of length $e+1$ exists only if* $\mathbf{c} \in \mathcal{C}_e^{(n)}$, *where*

$$\mathcal{C}_e^{(n)} = \left\{ (c_0, \dots, c_e) : \quad c_0, \dots, c_e \geq 0, \quad \sum_{j=0}^{e} c_j V_j^{(n)} = 2^n \right\} \tag{19}$$

and

$$V_j^{(n)} = \sum_{i=0}^{j} \binom{n}{i}, \quad j = 0, \dots, e. \tag{20}$$

To prove *1* and *2* we use induction on the pairs of nodes of the tree, which goes from terminal nodes to the root. The corollary follows from (10), (16), and (17).

Note that if we substitute the components $c_0, \dots, c_e$ expressed by (18) into the right hand side of (17), then we obtain the identity

$$k_i = \sum_{j=i}^{e} \left[\begin{array}{c} n-j \\ j-i \end{array} \right] \sum_{i'=j}^{e} (-1)^{i'+j} \left[\begin{array}{c} n-i' \\ i'-j \end{array} \right] k_{i'}.$$

It is easy to check that this identity holds because of the following representation of the indicator function ($\chi\{S\} = 1$ if the statement S is true and $\chi\{S\} = 0$ otherwise) :

$$\chi\{\, i' = i \,\} = \sum_{j=0}^{i'-i} (-1)^j \binom{i'-i}{j}.$$

Example 1 Let $n = 7, e = 3$. The vector $\mathbf{c} = (6, 0, 2, 1)$ is contained in the root of a tree of class $\mathcal{T}_3^{(7)}$ such that the distribution of terminal nodes has components

$$\begin{aligned}
k_0 &= \begin{bmatrix} 7 \\ 0 \end{bmatrix} \cdot 6 + \begin{bmatrix} 6 \\ 1 \end{bmatrix} \cdot 0 + \begin{bmatrix} 5 \\ 2 \end{bmatrix} \cdot 2 + \begin{bmatrix} 4 \\ 3 \end{bmatrix} \cdot 1 \\
&= 1 \cdot 6 + 6 \cdot 0 + 15 \cdot 2 + 20 \cdot 1 = 56, \\
k_1 &= \begin{bmatrix} 6 \\ 0 \end{bmatrix} \cdot 0 + \begin{bmatrix} 5 \\ 1 \end{bmatrix} \cdot 2 + \begin{bmatrix} 4 \\ 2 \end{bmatrix} \cdot 1 = 1 \cdot 0 + 5 \cdot 2 + 10 \cdot 1 = 20, \\
k_2 &= \begin{bmatrix} 5 \\ 0 \end{bmatrix} \cdot 2 + \begin{bmatrix} 4 \\ 1 \end{bmatrix} \cdot 1 = 1 \cdot 2 + 4 \cdot 1 = 6, \\
k_3 &= \begin{bmatrix} 4 \\ 0 \end{bmatrix} \cdot 1 = 1 \cdot 1 = 1.
\end{aligned}$$

On the other hand, if $\mathbf{k} = (56, 20, 6, 1)$ is the distribution of terminal nodes of a tree of class $\mathcal{T}_3^{(7)}$, then the vector contained in the root has components

$$\begin{aligned}
c_0 &= \begin{bmatrix} 7 \\ 0 \end{bmatrix} \cdot 56 - \begin{bmatrix} 6 \\ 1 \end{bmatrix} \cdot 20 + \begin{bmatrix} 5 \\ 2 \end{bmatrix} \cdot 6 - \begin{bmatrix} 4 \\ 3 \end{bmatrix} \cdot 1 \\
&= 1 \cdot 56 - 6 \cdot 20 + 15 \cdot 6 - 20 \cdot 1 = 6, \\
c_1 &= \begin{bmatrix} 6 \\ 0 \end{bmatrix} \cdot 20 - \begin{bmatrix} 5 \\ 1 \end{bmatrix} \cdot 6 + \begin{bmatrix} 4 \\ 2 \end{bmatrix} \cdot 1 = 1 \cdot 20 - 5 \cdot 6 + 10 \cdot 1 = 0, \\
c_2 &= \begin{bmatrix} 5 \\ 0 \end{bmatrix} \cdot 6 - \begin{bmatrix} 4 \\ 1 \end{bmatrix} \cdot 1 = 1 \cdot 6 - 4 \cdot 1 = 2, \\
c_3 &= \begin{bmatrix} 4 \\ 0 \end{bmatrix} \cdot 1 = 1 \cdot 1 = 1.
\end{aligned}$$

Since $(V_0^{(7)}, \ldots, V_3^{(7)}) = (1, 8, 29, 64)$,

$$\sum_{j=0}^{3} c_j V_j^{(7)} = 6 \cdot 1 + 0 \cdot 8 + 2 \cdot 29 + 1 \cdot 64 = 2^7.$$

Furthermore,

$$\sum_{i=0}^{3} k_i 2^i = 56 \cdot 1 + 20 \cdot 2 + 6 \cdot 4 + 1 \cdot 8 = 2^7.$$

The *only if* statement of the corollary cannot be replaced by an *if and only if* statement because, for some vectors $\mathbf{c}$ satisfying (19), trees of class $\mathcal{T}_e^{(n)}$ containing $\mathbf{c}$ in the root do not exist. For example, if $e = 2$, then the vector $\mathbf{c} = (4, 1, 4)$ satisfies (19) for $n = 7$ since $4 \cdot 1 + 1 \cdot 8 + 4 \cdot 29 = 2^7$. Nevertheless, $(V_0^{(6)}, V_1^{(6)}, V_2^{(6)}) = (1, 7, 22)$ and

$$\max\{\ a_0 + 7a_1 + 22a_2,\ b_0 + 7b_1 + 22b_2\ \} > 2^6$$

for all vectors (a_0, a_1, a_2) and (b_0, b_1, b_2) with non-negative components such that $a_0 + b_0 = a_1 + b_1 = 5$, $a_2 + b_2 = 4$. Note that if we increase n by 1 and the 0-th component of the vector $\mathbf{c}$ by 95, i.e., consider the vector $(99, 1, 4)$ for $n = 8$, then such a tree exists.

The following interpretation of the results presented above is also possible. The operation introduced in $(T4)$ can be viewed as "addition" $\mathbf{c} = \mathbf{a}\,[+]\,\mathbf{b}$, where the "result" $a_j + b_j$ and "carry" c_{j+1} coming from the neighboring position are used to calculate the j-th component of the vector $\mathbf{c}$. The presence of carries makes it possible to construct an object with new properties. This situation really takes place since the vector $\mathbf{c}$ describes a covering of the Hamming space of dimension n by subsets whose cardinalities are equal to the sizes of the Hamming balls $V_0^{(n)}, \ldots, V_e^{(n)}$, while the terms describe similar coverings of the space of dimension $n-1$. The operation $[+]$ is controlled by a component-wise addition $\mathbf{k}' + \mathbf{k}''$, where the vectors $\mathbf{k}', \mathbf{k}'' \in \mathcal{K}_e^{(n-1)}$ are the distributions of terminal nodes of the trees whose roots contain $\mathbf{a}$ and $\mathbf{b}$, respectively. The carry symbols are absent in the last operation and it can be viewed as merging.

We conclude the section with the proposition which will be used in the analysis.

Proposition 3 *If* $\mathbf{c} \in \mathcal{C}_e^{(t)}$, $\mathbf{a} \in \mathcal{C}_e^{(t-1)}$, *and the vectors* $\mathbf{a}, \mathbf{c}$ *are expressed* *by (2), then*

$$\sum_{j=0}^{e} \binom{t-1}{j} \delta_j = 0. \tag{21}$$

Furthermore, if (21) holds, $\mathbf{c} \in \mathcal{C}_e^{(t)}$*, and* $\mathbf{a}$ *is such that (2) is satisfied, then* $\mathbf{a} \in \mathcal{C}_e^{(t-1)}$.

4 Trees of class $\mathcal{T}_e^{(n)*}$

If the root of some tree of class $\mathcal{T}_e^{(n)}$ contains a given vector $\mathbf{c}$, then we cannot conclude that $\mathbf{c}$ belongs to the set $\mathcal{R}_e^{(n)}$ defined in (5). For example, there exists a tree of class $\mathcal{T}_1^{(4)}$ whose root contains the vector $(1,3)$ (see Fig.2), but $(1,3) \notin \mathcal{R}_1^{(4)}$.

Definition 2 *A tree belongs to the class* $\mathcal{T}_e^{(n)*}$ *if and only if*

$(T1^*)$ *it belongs to the class* $\mathcal{T}_e^{(n)}$;

$(T2^*)$ *for any pair of internal nodes containing the vectors* $\mathbf{a} = (a_0, \ldots, a_e)$, $\mathbf{b} = (b_0, \ldots, b_e)$ *and having the same parent,*

$$\delta_j \in \mathcal{D}(c_j), \text{ for all } j = 0, \ldots, e,$$

where $c_0, \ldots, c_e$ *are calculated in accordance with (9), the sets* $\mathcal{D}(c_0), \ldots, \mathcal{D}(c_e)$ *are defined in (1), and* $\delta_e, \ldots, \delta_0$ *are recurrently defined by the equations*

$$\delta_j = \delta_{j+1} + a_j - b_j, \ \ j = e, \ldots, 0 \tag{22}$$

with $\delta_{e+1} = 0$.

Note that the tree shown in Fig.2 does not belong to the class $\mathcal{T}_e^{(n)*}$ since the restriction $(T2^*)$ is violated for the nodes at level 3 containing the vectors $\mathbf{a} = (0,2)$ and $\mathbf{b} = (4,1)$: using (22) we obtain $(\delta_0, \delta_1) = (-3, +1)$, but $\mathbf{c} = (1,3)$, and the inequality $|\delta_0| \le c_0$ included in the definition of the set $\mathcal{D}(c_0)$ does not hold. The structure of a tree of class $\mathcal{T}_2^{(8)*}$ is illustrated in Table 2, where we show both the vectors contained in the nodes and the distributions of terminal nodes of corresponding subtrees.

Lemma 1 *A tree of class $\mathcal{T}_e^{(n)*}$ whose root contains a given vector $\mathbf{c} \notin \{\mathbf{1}_0, \ldots, \mathbf{1}_e\}$ exists if and only if $\mathbf{c}$ belongs to the set $\mathcal{R}_e^{(n)*}$ defined in (6). Moreover,*

$$\mathbf{a}(\mathbf{c}|\boldsymbol{\delta}^0), \mathbf{b}(\mathbf{c}|\boldsymbol{\delta}^0) \in \mathcal{R}_e^{(n-1)*} \bigcup \{\mathbf{1}_0, \ldots, \mathbf{1}_e\} \tag{23}$$

for any vector $\boldsymbol{\delta}^0 \in \mathcal{D}_e(\mathbf{c})$ minimizing the expression at the right hand side of (4).

The statement of Lemma 1 follows from the equivalence of the restrictions on components of the vectors included in the set $\mathcal{D}_e(\mathbf{c})$ to the restrictions on components $\delta_0, \ldots, \delta_e$ in $(T1^*)$, $(T2^*)$ (if $\boldsymbol{\delta} \in \{-\mathbf{c}, \mathbf{c}\}$, then $(T1^*)$ is violated). This statement says that if we want to construct a tree of class $\mathcal{T}_e^{(n)*}$ whose root contains a certain vector $\mathbf{c}$ satisfying (19), then we have to assign the first question in such a way that the sons contain vectors $\mathbf{a}$ and $\mathbf{b}$ satisfying the same equation with $n-1$ substituted for n. A similar requirement holds for all other internal nodes of the tree. Therefore, if a tree of class $\mathcal{T}_e^{(n)*}$ has been constructed, then the linear combination of the sizes of the Hamming balls $V_0^{(t)}, \ldots, V_e^{(t)}$ where the coefficients are the components of the vector contained in any node at level $t \in \{0, \ldots, n\}$ is equal to 2^t. This observation will be developed in the next section.

5 Searching strategies

5.1 Description of searching strategies based on trees of class $\mathcal{T}_e^{(n)*}$

A sequential searching strategy is usually understood as an algorithm which defines a subset of the searching domain depending on the previous "yes-no" answers. Suppose that the domain consists of $c_0 + \ldots + c_e$ integers, where the vector $\mathbf{c} = (c_0, \ldots, c_e)$ is given in such a way that $\sum_{j=0}^{e} c_j V_j^{(n)} = 2^n$. This means that $\mathbf{c} \in \mathcal{C}_e^{(n)}$, so there may exist a tree of class $\mathcal{T}_e^{(n)*}$ whose root contains the vector $\mathbf{c}$ as the condition of the corollary is satisfied. Suppose that such a tree really exists; for example, this is the case for $e = 2$, $\mathbf{c} = (108, 0, 4)$, and $n = 8$ (see Table 2). If the topology of the tree is known, then the vectors contained in every node and the vectors $\boldsymbol{\delta}$ can be recurrently determined by calculating (9) and (22) from the terminal nodes (all terminal nodes contain the unit vectors) to the root. These calculations can be also translated into a searching strategy under the following assumption : if the responder has chosen an integer belonging to the set $\{C_{j+1}+1, \ldots, C_j\}$, then j lies are allowed in the answers (we denote $C_{e+1} = 0$, $C_i = c_i + \ldots + c_e$, $i = 0, \ldots, e$, and we assume that the set above is empty for $C_{j+1} = C_j$),- the procedure was described in Section 2 (see (7), (8)) and we add that the questioner can imagine that he is climbing down the tree from the root, located at the highest level, in such a way that he knows the vector $\boldsymbol{\delta}$ for each current position and presents a question having the structure determined by this vector. However, the original problem can be different in the sense that the vector $\mathbf{c}$ which describes the searching domain does not have to belong to the set $\mathcal{C}_e^{(n)}$ for any n. For example, the problem can be formulated as finding of one of 4 integers chosen from the set $\mathcal{M}_2 = \{1, \ldots, 4\}$, where $e = 2$ lies are allowed in the answers. If the questioner knows how to deal with a more difficult problem when the searching domain consists of 112 integers and 2 lies are allowed in the answers under the assumption that the integer belongs to $\mathcal{M}_2$, none otherwise, then he can also deal with the original problem. In this case, the integers belonging to the set $\{5, \ldots, 112\}$ included in the questions can be viewed as *dummy* integers since the responder does not pay attention to them. A non-trivial conclusion that comes from introducing the trees is that the original problem can be solved with n questions *only if* a more difficult problem can be solved with the same number of questions for a properly chosen number of dummy integers, namely, this choice should be such that the initial state of the search belongs to the set $\mathcal{C}_e^{(n)}$. A formalization of this approach is given below.

For a fixed triple (M, n, e), let

$$\mathbf{c}_e^{(n)}(M) = (c_0^{(n)}(M), \ldots, c_e^{(n)}(M)), \;\; \mathbf{k}_e^{(n)}(M) = (k_0^{(n)}(M), \ldots, k_e^{(n)}(M)),$$

where

$$c_j^{(n)}(M) \;\; = \;\; \begin{cases} 2^n - MV_e^{(n)}, & \text{if } j = 0, \\ 0, & \text{if } j \in \{1, \ldots, e-1\}, \\ M, & \text{if } j = e, \end{cases} \tag{24}$$

$$k_i^{(n)}(M) \;=\; \begin{cases} 2^n - MV_e^{(n)} + \left[\begin{array}{c} n-e \\ e \end{array}\right] M, & \text{if } i=0, \\ \left[\begin{array}{c} n-e \\ e-i \end{array}\right] M, & \text{if } i \in \{1,\ldots,e\}. \end{cases}$$

In particular,

$$\mathbf{c}_1^{(n)}(M) = (2^n - (n+1)M, M), \;\; \mathbf{k}_1^{(n)}(M) = (2^n - 2M, M). \tag{25}$$

Lemma 2 *Any integer belonging to the set $\{1,\ldots,M\}$ can be found with at most n questions when e lies are allowed in the answers if and only if there exists a tree of class $\mathcal{T}_e^{(n)*}$ having the distribution of terminal nodes $\mathbf{k}_e^{(n)}(M) \in \mathcal{K}_e^{(n)}$ whose root contains the vector $\mathbf{c}_e^{(n)}(M) \in \mathcal{C}_e^{(n)}$.*

5.2 Partition strategies

If $\mathbf{c}$ is the current state of the search, then we have to assign a certain vector $\boldsymbol{\delta}$ for that state. We will denote this vector by $\boldsymbol{\delta}(\mathbf{c})$ and say that it *determines a partition of* $\mathbf{c}$.

Definition 3 *A partition strategy for a given triple (M,n,e) is an algorithm for constructing a vector $\boldsymbol{\delta}(\mathbf{c}) \in \mathcal{D}_e(\mathbf{c})$ for all $\mathbf{c} \in \mathcal{S}_e^{(t)} \backslash \{\mathbf{1}_0,\ldots,\mathbf{1}_e\} \subseteq \mathcal{C}_e^{(t)}$, $t=n,\ldots,1$, where*

$$\begin{array}{rcl} \mathcal{S}_e^{(n)} &=& \{\, \mathbf{c}_e^{(n)}(M) \,\}, \\ \mathcal{S}_e^{(t-1)} &=& \displaystyle\bigcup_{\mathbf{c} \in \mathcal{S}_e^{(t)} \backslash \{\mathbf{1}_0,\ldots,\mathbf{1}_e\}} \{\, \mathbf{a}(\mathbf{c}|\boldsymbol{\delta}(\mathbf{c})), \mathbf{b}(\mathbf{c}|\boldsymbol{\delta}(\mathbf{c})) \,\}, \quad t=n,\ldots,2, \end{array}$$

and $\mathbf{c}_e^{(n)}(M)$ is defined in (24).

Lemma 3 *A vector $\boldsymbol{\delta} = (\delta_0,\ldots,\delta_e)$ can determine a partition of a given state of the search $\mathbf{c} \in \mathcal{C}_e^{(t)}$ if and only if the following restrictions are satisfied:*

(Δ0) $\delta_0 = -\sum_{j=1}^{e} \binom{t-1}{j} \delta_j$;

(Δ1) $\delta_j \equiv c_j$ *and* $|\,\delta_j\,| \le c_j$ *for all* $j=1,\ldots,e$;

(Δ2) $\left|\sum_{j=1}^{e} \binom{t-1}{j} \delta_j\right| \le c_0$.

Obviously, any partition strategy can be translated to a searching strategy using dummy integers in the same way as was considered in Section 2. Note however that vectors $(\delta_0, \ldots, \delta_e)$ satisfying $(\Delta 0)$, $(\Delta 1)$ exist for any vector $\mathbf{c}$ with non-negative components, but it may be that none of them satisfies $(\Delta 2)$.

Lemma 4 *Any integer belonging to the set $\{1, \ldots, M\}$ can be found with at most n questions when e lies are allowed in the answers if and only if there exists a partition strategy for the triple (M, n, e).*

Example 2 Suppose $e = 1$, $t = n$, and the statement :

$$\begin{cases} M \text{ is even} \implies (n+1)M \leq 2^n \\ M \text{ is odd} \implies (n+1)M \leq 2^n - n + 1 \end{cases} \tag{26}$$

is true. Then the vector $\mathbf{c}_1^{(n)}(M)$ defined in (25) has non-negative components, and the restrictions $(\Delta 0)$–$(\Delta 2)$ become

$$\begin{array}{ll} (\Delta 0): & \delta_0 = -(n-1)\delta_1; \\ (\Delta 1): & \delta_1 \equiv M, \ \ |\delta_1| \leq M; \\ (\Delta 2): & |(n-1)\delta_1| \leq 2^n - (n+1)M. \end{array}$$

It can be easily seen that these restrictions are satisfied for the vector

$$(\delta_0, \delta_1) = \begin{cases} (0,0), & \text{if } M \text{ is even,} \\ (-n+1, +1), & \text{if } M \text{ is odd.} \end{cases} \tag{27}$$

On the other hand, if $(\Delta 0)$–$(\Delta 2)$ are satisfied for some vector $\boldsymbol{\delta}$, then they are also satisfied for the vector defined in (27). Thus, (26) gives a necessary and sufficient condition for it to be possible to partition the state $\mathbf{c}_1^{(n)}(M)$. We will show in subsection 5.5 that (26) is also a necessary and sufficient condition for it to be possible to partition all states contained in the nodes of some tree of class $\mathcal{T}_1^{(n)*}$ whose root contains $\mathbf{c}_1^{(n)}(M)$.

The following property of the vectors that can determine a partition of the states of the search will be used in the subsequent considerations.

Proposition 4 *Let* $\mathbf{c} \in \mathcal{C}_e^{(t)}$ *be given and let* η *be the index of the last positive component of* $\mathbf{c}$*. Let*

$$\begin{aligned}
\delta_j^* &= \arg\min_{\delta \in \mathcal{D}(c_j)} \left| \binom{t-1}{j}\delta + \sum_{i=j+1}^{e} \binom{t-1}{i}\delta_i^* \right|, \quad j = e, \ldots, 1, \qquad (28)\\
\delta_0^* &= -\sum_{j=1}^{e} \binom{t-1}{j}\delta_j^*.
\end{aligned}$$

If

$$c_0 \geq \binom{t-1}{\eta}, \qquad (29)$$

then the vector $(\delta_0^*, \ldots, \delta_e^*)$ *satisfies the restrictions* $(\Delta 0)$–$(\Delta 2)$ *and can determine a partition of* $\mathbf{c}$*.*

If $t-1 \geq 2\eta$, then the inequality $|\delta_0^*| \leq \binom{t-1}{\eta}$ holds for any vector $\mathbf{c}$ with non-negative components, and the condition $\mathbf{c} \in \mathcal{C}_e^{(t)}$ is used in the proof only for the case $t-1 < 2\eta$. Note that the statement is also true with the index of the last odd component of $\mathbf{c}$ substituted for η, but this possible improvement is not used in our analysis.

5.3 Transfer of the states of the search

For a fixed partition strategy, there may exist $t \in \{1, \ldots, n\}$ and states $\mathbf{c} \in \mathcal{C}_e^{(t)}$ that can be reached from the initial state such that the restriction $(\Delta 2)$ is not satisfied, which means that the responder can make the searching procedure starting with $\mathbf{c}$ consist of $t+1$ questions. We introduce an operation called *transfer of the states of the search to the level* $t+1$ and use it as a tool for the analysis of the necessary number of questions. This operation can be also of interest for various extensions of the problem when the questions should be formed in a real time mode; in this case the transfer can be viewed as some *Gedankenexperiment*, i.e., it exists only in the imagination of the questioner, who adds some extra dummy integers into the set $\mathcal{M}_0$ while constructing a partition of the current state of the search.

Definition 4 *Let* $\mathbf{c} = (c_0, \ldots, c_e) \in \mathcal{C}_e^{(t)}$ *be some state of the search that can be reached from the initial state using a fixed partition strategy. Transfer* *of* $\mathbf{c}$ *to the level* $\hat{t} = t+1$ *is understood as changing the 0-th component of the vector* $\mathbf{c}$ *in such a way that the new vector* $\hat{\mathbf{c}} = (\hat{c}_0, \ldots, \hat{c}_e)$ *has components*

$$\hat{c}_0 = 2^{\hat{t}} - \sum_{j=1}^{e} c_j V_j^{(\hat{t})} = c_0 + \sum_{j=0}^{e} \binom{\hat{t}-1}{j} c_j, \tag{30}$$
$$(\hat{c}_1, \ldots, \hat{c}_e) = (c_1, \ldots, c_e).$$

Example 3 Let $e = 3$, $t = 7$, and $\mathbf{c} = (6, 0, 2, 1)$ (see Table 3). Then $\mathbf{c} \in \mathcal{C}_3^{(7)}$, but $c_0 = 6$ is less than the sum in the restriction $(\Delta 2)$ for any vector $\boldsymbol{\delta}$ satisfying the restrictions $(\Delta 0)$, $(\Delta 1)$. We transfer $\mathbf{c}$ to level $\hat{t} = 8$ and form the vector $\hat{\mathbf{c}} = (89, 0, 2, 1)$, where

$$89 = 2^8 - (0 \cdot 9 + 2 \cdot 37 + 1 \cdot 93) = 6 + (6 \cdot 1 + 0 \cdot 7 + 2 \cdot 21 + 1 \cdot 35).$$

If the partition strategy is fixed in such a way that we use (28) for all states of the search, then we reach the state $(1, 3, 0, 0)$ at level 4. The restriction $(\Delta 2)$ is again not satisfied and we transfer this state to level 5 by constructing a new state $(14, 3, 0, 0)$.

5.4 Bounds on the number of questions

The known lower bound on the number of questions [4] can be derived from the observation that the 0-th component of the vector $\mathbf{c}_e^{(n)}(M)$ defined in (24) must be non-negative.

Lemma 5 *Any integer belonging to the set* $\{1, \ldots, M\}$ *can be found after at most* n *questions when* e *lies are allowed in the answers only if*

$$MV_e^{(n)} \leq 2^n, \tag{31}$$

where $V_e^{(n)}$ *is defined in (20).*

Note that inequality (31) can be introduced as

$$M \sum_{i=0}^{e} \left[\begin{array}{c} n-e \\ e-i \end{array} \right] 2^{-(n-i)} \leq 1,$$

which makes it closer to the well-known Kraft inequality [7].

To derive an upper bound on the number of questions we fix a specific partition strategy by assigning a vector $\boldsymbol{\delta}(\mathbf{c})$ for all states of the search $\mathbf{c}$ and run the experiment presented below.

(E_1) Set $\mathcal{I} = \emptyset$, $i = 0$, $t = n^*$, and $\mathbf{c} = \mathbf{c}_e^{(n^*)}(M)$, where n^* is the minimal integer satisfying (31).

(E_2) Increase i by 1 and set $\mathbf{c}_i = \mathbf{c}$.

(E_3) If the restriction ($\Delta 2$) is not satisfied, then replace $\mathbf{c}$ with the vector $\hat{\mathbf{c}}$ defined in (30). Include i into the set $\mathcal{I}$, increase t by 1, and go to (E_2).

(E_4) Calculate the components of the vectors $\mathbf{a}, \mathbf{b}$ by (2) and replace $\mathbf{c}$ with one of these vectors. Decrease t by 1 and if $\mathbf{c} \notin \{\mathbf{1}_0, \ldots, \mathbf{1}_e\}$, then go to ($E_2$); stop otherwise.

We do not specify the selection rule at step (E_4) since our goal is to prove that the experiment (E_1)–(E_4) always stops with the number of transfers $|\mathcal{I}| \leq 3e$ for a certain partition strategy. This statement simultaneously gives an upper bound on the additional number of questions as compared to n^* which is required by an optimal partition strategy.

Let us introduce the following notation : for all $\xi, \eta \in \{0, \ldots, e\}$, $\xi < \eta$, let

$$\mathcal{G}_{\xi,\eta} = \left\{ \mathbf{c} : \;\; c_\xi \geq c_\eta = 1; \;\; c_j = 0, \; j \in \{\xi + 1, \ldots, e\} \backslash \{\eta\} \right\} \tag{32}$$

and let

$$\mathcal{G}_{\eta,\eta} = \left\{ \mathbf{c} : \;\; c_\eta > 1; \;\; c_j = 0, \; j \in \{\eta + 1, \ldots, e\} \right\}. \tag{33}$$

Obviously we can uniquely determine η and $\xi \leq \eta$ such that $\mathbf{c} \in \mathcal{G}_{\xi,\eta}$ for any vector $\mathbf{c} \notin \{\mathbf{0}, \mathbf{1}_0, \ldots, \mathbf{1}_e\}$ with non-negative components using the following rules : η is the index of the last positive component of $\mathbf{c}$ and $\xi = \eta$ if $c_\eta > 1$; otherwise ξ is the maximal index of a positive component which is less than η. Let $(\xi_1, \eta_1), (\xi_2, \eta_2), \ldots$ be the sequence of pairs corresponding to the sequence $\mathbf{c}_1, \mathbf{c}_2, \ldots$ constructed in the experiment (E_1)–(E_4), where

$$(\xi_i, \eta_i) = (\xi, \eta) \iff \mathbf{c}_i \in \mathcal{G}_{\xi,\eta}. \tag{34}$$

Proposition 5 *For all* $\mathbf{c} \in \mathcal{C}_e^{(t)} \cap \mathcal{G}_{\xi,\eta}$*, let the components of the vector* $\boldsymbol{\delta}(\mathbf{c}) = (\delta_0, \ldots, \delta_e)$ *be defined as follows :*

$$\begin{aligned} \delta_j &= \begin{cases} \delta_j^*, & \text{if } \xi = \eta, \\ +1, & \text{if } j = \eta \text{ and } \xi < \eta, \\ -c_j, & \text{if } j \neq \eta \text{ and } \xi < \eta \end{cases} \quad j = 1, \ldots, e, \\ \delta_0 &= -\sum_{j=1}^{e} \binom{t-1}{j} \delta_j, \end{aligned} \tag{35}$$

where $\delta_e^*, \ldots, \delta_1^*$ *are defined in (28), and let the components of the vectors* $\mathbf{a}, \mathbf{b}$ *be defined by (2).*

($-$) *If* $0 = \xi = \eta$*, then* $|\delta_0| \leq c_0$*,* $\mathbf{a}, \mathbf{b} \in \mathcal{G}_{0,0} \cup \{\mathbf{1}_0\}$*.*

($-$) *If* $0 = \xi < \eta$*, then* $|\delta_0| \leq c_0$*,* $\mathbf{a} \in \mathcal{G}_{0,\eta} \cup \{\mathbf{1}_\eta\}$*, and* $\mathbf{b} \in \mathcal{G}_{0,\eta-1}$*.*

($-$) *If* $0 < \xi < \eta$ *and* $c_0 \geq \sum_{j=1}^{e} \binom{t-1}{j} c_j$*, then* $|\delta_0| \leq c_0$*,* $\mathbf{a} \in \mathcal{G}_{\xi-1,\eta}$*, and* $\mathbf{b} \in \mathcal{G}_{\xi,\eta-1}$*.*

($-$) *If* $0 < \xi = \eta$*,* $c_\eta = 2$*, and* $c_0 \geq \sum_{j=1}^{e} \binom{t-1}{j} c_j$*, then* $|\delta_0| \leq c_0$*,* $\mathbf{a}, \mathbf{b} \in \mathcal{G}_{\eta-1,\eta}$*.*

($-$) *If* $0 < \xi = \eta$*,* $c_\eta \geq 3$*, and* $c_0 \geq \binom{t-1}{\eta}(c_\eta - 2)$*, then* $|\delta_0| \leq c_0$*,* $\mathbf{a}, \mathbf{b} \in \mathcal{G}_{\eta,\eta}$*, and* $a_0 \geq \binom{t-2}{\eta}(a_\eta - 2)$*,* $b_0 \geq \binom{t-2}{\eta}(b_\eta - 2)$*.*

The statements of Proposition 5 are of interest because the inequality $|\delta_0| \leq c_0$ means that the vector $\boldsymbol{\delta}(\mathbf{c})$ can determine a partition of $\mathbf{c}$ and our experiment goes through (E_4). The first two statements claim that all states of the search belonging to the set $\mathcal{G}_{0,\eta}$ can be partitioned without transfers for all $\eta \in \{0, \ldots, e\}$. The third and fourth statements claim that the state $\mathbf{c}$ obtained after the transfer of the previous state to the level t can be partitioned using the vector $\boldsymbol{\delta}(\mathbf{c})$. The lower bounds on c_0 and a_0, b_0 in the last statement show that the partition of $\mathbf{c}$ can be continued without transfers at least until the η-th component of the new state becomes equal to 2.

Lemma 6 *Let* n^* *be the minimal integer satisfying (31). Then there exist searching strategies that require at most*

$$n \leq n^* + 3e \tag{36}$$

questions to find any integer belonging to the set $\{1,\ldots,M\}$ when e lies are allowed in the answers.

The proof is based on the analysis of the results of the experiment (E_1)–(E_4) with the partition strategy defined by (35). Let $i_1 < \ldots < i_{|\mathcal{I}|}$ denote the elements included into the set $\mathcal{I}$ and let $\hat{\xi}_k = \xi_{i_k}$, $\hat{\eta}_k = \eta_{i_k}$ for all $k = 1,\ldots,|\mathcal{I}|$. Then using Proposition 5 we prove the following statements :

(a) the experiment (E_1)–(E_4) stops with some $i < \infty$ and a finite set $\mathcal{I}$ for any selection rule in (E_4);

(b) $2e \geq \hat{\xi}_1 + \hat{\eta}_1 \geq \ldots \geq \hat{\xi}_{|\mathcal{I}|} + \hat{\eta}_{|\mathcal{I}|} > 0$;

(c) every odd integer belonging to the set $\{1,\ldots,2e\}$ occurs in the sequence $\hat{\xi}_1 + \hat{\eta}_1,\ldots,\hat{\xi}_{|\mathcal{I}|} + \hat{\eta}_{|\mathcal{I}|}$ at most once;

(d) every even integer belonging to the set $\{1,\ldots,2e\}$ occurs in the sequence $\hat{\xi}_1 + \hat{\eta}_1,\ldots,\hat{\xi}_{|\mathcal{I}|} + \hat{\eta}_{|\mathcal{I}|}$ at most twice.

Hence $|\mathcal{I}| \leq 3e$, and (36) follows.

Lemmas 5,6 show that the difference between the values of the upper and lower bounds on the minimal number of questions needed to find any of M integers is at most $3e$ and does not depend on M. Furthermore, the upper bound can be attained with the explicit partition strategy. However, these results can be improved for specific data, and we consider some of these data in the next subsections.

5.5 Solution to the search problem for $e = 1$

The introduction of trees of class $\mathcal{T}_e^{(n)*}$ allows us to present an optimal solution to the search problem for $e = 1$ as a direct generalization of the solution to the problem for $e = 0$ when $\log M$ is an integer and we always divide the current searching domain into two halves. The result given in the corollary below is known [10], but it was proved using a much more complex searching strategy. Note also that if we are only interested in the minimal number of questions needed to solve the problem for $e = 1$, then many of the previous considerations can be omitted.

Lemma 7 *Any integer belonging to the set* $\{1,\ldots,M\}$ *can be found with* n *questions when 1 lie is allowed in the answers if and only if there exists a tree of class* $\mathcal{T}_1^{(n)*}$ *such that the root contains* $\mathbf{c}_1^{(n)}(M)$ *and, for all* $t \in \{1,\ldots,n\}$ *and* $\tilde{M} \le M$,

(H$_c$) *if some internal node located at level* t *contains* $\mathbf{c}_1^{(t)}(\tilde{M})$*, then its sons contain* $\mathbf{c}_1^{(t-1)}(\lceil \tilde{M}/2\rceil), \mathbf{c}_1^{(t-1)}(\lfloor \tilde{M}/2\rfloor)$ *and the corresponding subtrees have distributions of terminal nodes* $\mathbf{k}_1^{(t-1)}(\lceil \tilde{M}/2\rceil), \mathbf{k}_1^{(t-1)}(\lfloor \tilde{M}/2\rfloor)$.

Equivalently, the problem under consideration can be solved with n *questions if and only if it can be solved by constructing a partition strategy for the triple* (M,n,e) *in such a way that*

(H$_\delta$) *the vector* (δ_0,δ_1)*, which determines the partition of any vector* (c_0,c_1) *with* $c_0 + (t+1)c_1 = 2^t$*, is equal to* $(0,0)$ *for* c_1 *even and equal to* $(-t+1,+1)$ *for* c_1 *odd.*

Corollary *The minimal number of questions which is necessary to find any integer belonging to the set* $\{1,\ldots,M\}$ *when 1 lie is allowed in the answers is equal to the minimal integer* n *satisfying (26).*

Obviously, (H$_c$) does not imply dividing of the 0-th component of the current state of the search into two integers whose difference is at most 1. For example, there exists a tree of class $\mathcal{T}_1^{(3)*}$ containing the vector $(4,1)$ in the root and having the distribution of terminal nodes $(6,1)$. Using (H$_c$) we obtain that the sons contain $(1,1),(4,0)$ and that the distributions of terminal nodes of the corresponding subtrees are equal to $(2,1),(4,0)$. Thus, the vector $(4,1)$ is partitioned in such a way that we enter the "optimal strategy" part of Fig.1.

5.6 Balanced partitioning of the states of the search

To construct a partition strategy in the sense of Definition 3 that allows us to find a searching strategy for solving the problem with n questions we have to assign a vector $\boldsymbol{\delta}(\mathbf{c})$ to every state of the search $\mathbf{c} \in \mathcal{S}_e^{(t)} \subseteq \mathcal{C}_e^{(t)}$ that can be reached from the initial state $\mathbf{c}_e^{(n)}(M)$ after $n-t$ partitions, $t = n,\ldots,1$. We have some freedom in the choice of the e-th component of the vector $\boldsymbol{\delta}(\mathbf{c})$, while the 0-th component is calculated using the deterministic function defined in ($\Delta 0$), and there can be some number of vectors $\boldsymbol{\delta}(\mathbf{c})$ satisfying the restrictions ($\Delta 1$), ($\Delta 2$). The difficulty of the analysis of the partition strategies comes from the fact that a particular assignment of the vector $\boldsymbol{\delta}(\mathbf{c})$ determines the states of the search that should be included into the set $\mathcal{S}_e^{(t-1)}$ and we cannot separately deal with the partitions of every state. If $e = 1$, then this difficulty can be escaped as shown in the previous subsection, but in the general case the procedures based on sequential assignment of the vectors $\boldsymbol{\delta}(\mathbf{c})$ for all $\mathbf{c}$ are sub-optimal. The trees of class $\mathcal{T}_e^{(n)*}$ lead to an essential simplification of the natural approach to the partitioning developed in [10] and other papers where the authors "balance the volume", i.e., in our terminology, assign a vector $\boldsymbol{\delta}(\mathbf{c})$ minimizing the absolute value of the difference of $\sum_{j=0}^{e} a_j V_j^{(t-1)}$ and $\sum_{j=0}^{e} b_j V_j^{(t-1)}$ ($a_0, \ldots, a_e$ and $b_0, \ldots, b_e$ are the components of the vectors contained in the sons of the node that contains $\mathbf{c}$). This simplification arises from the restriction on the set of possible states that can be contained in the nodes of a tree of class $\mathcal{T}_e^{(n)*}$: we require that $\mathbf{a}, \mathbf{b} \in \mathcal{C}_e^{(t-1)}$ and hence the sums must be exactly equal to each other. We call our algorithm the *balanced partitioning of a given state of the search* $\mathbf{c} \in \mathcal{C}_e^{(t)}$, and present it below.

$\odot^*$ *For* $j = e$ *step* -1 *until 0 do*

$$\delta_j^* = \arg \min_{\substack{\delta \equiv c_j \\ |\delta| \le c_j}} \left| \binom{t-1}{j} \delta + \binom{t-1}{j+1} \delta_{j+1}^* + \ldots + \binom{t-1}{e} \delta_e^* \right|.$$

$\ominus^*$ *If* $\sum_{j=0}^{e} \binom{t-1}{j} \delta_j^* = 0$, *then output* **OK**.

We fix an n (initially n is chosen from the upper bound (31)), construct the set $\mathcal{S}_e^{(n)} = \{\mathbf{c}_e^{(n)}(M)\}$, and assign $t = n$. For all $\mathbf{c} \in \mathcal{S}_e^{(t)} \backslash \{\mathbf{1}_0, \ldots, \mathbf{1}_e\}$, we run the balanced partitioning and, having received **OK**, form the vectors $\mathbf{a}, \mathbf{b}$ with components defined in (2) and include these vectors into the set $\mathcal{S}_e^{(t-1)}$. Then we decrease t by 1 and repeat the operations. An example of constructing the tree by this algorithm is given in Table 2. If the **OK** signal is received for all states under consideration, we say that the partition strategy is successful and any of M integers can be found with n questions. If this is not the case for some $\mathbf{c}$, then we increase n by 1 and run the procedure again from the very beginning. Note that this approach does match the ideas of transferring the states of the search developed to prove the upper bound on the necessary number of questions. However, after some preliminary work we find a value of n for which we can construct the whole tree by balanced partitioning.

Obviously, the algorithm of balanced partitioning is optimal for $e = 1$ and constructs the partition strategy for the value of n specified in Lemma 7. In the general case the algorithm is not optimal, but it is very simple and the results are rather good. In particular, the only problem which we noticed for $e = 2$ concerns the partitioning of the state $\mathbf{c} = (40, 15, 7)$ for $t = 9$: in this case, our algorithm generates $\mathbf{a} = (27, 9, 4)$ and $\mathbf{b} = (28, 13, 3)$; however $\mathbf{a} \notin \mathcal{R}_2^{(8)}$, and we have to increase n. An optimal scheme developed by Guzicki [8] partitions this state into $\mathbf{a} = (36, 8, 4)$ and $\mathbf{b} = (19, 14, 3)$; as a result, the partition strategy is successful. After including this special partition into the program, we could not find any difference in the results for all integers up to 10 million. We also inspected the algorithm for $\log M = 0, \ldots, 16$ and $e = 1, \ldots, 8$, as in the paper by Lawler and Sarkissian [9]. Balanced partitioning leads to the same number of questions as in [9], except for one pair of parameters : if $\log M = 8$ and $e = 5$, then we solve the searching problem with 24 questions instead of 25. If $\log M$ is not an integer, then the advantage becomes more essential.

6 Concluding remarks

Trees of class $\mathcal{T}_e^{(n)}$ can be considered as tools for specifying a partial one-to-one mapping between vectors of length $e + 1$, called $\mathbf{c}$ and $\mathbf{k}$, having non-negative components. These vectors are interpreted as the state of the search contained in the root of the tree and the distribution of terminal nodes of that tree. This interpretation makes sense because we define the mapping in such a way that the linear combination of the sizes of the Hamming balls $V_0^{(n)}, \ldots, V_e^{(n)}$ taken with the coefficients defined by $\mathbf{c}$ is equal to the linear combination of powers of 2 taken with the coefficients defined by $\mathbf{k}$ and equal to 2^n. The first linear combination describes a covering of the Hamming space of dimension n, while the second one describes the structure of a rooted regular binary tree. It seems to be that the invertible mapping of coverings to trees, which includes both the topology and arithmetic is useful for different purposes.

The trees developed in the paper were introduced independently of the search problem in the following context. Let us consider the question : How to write a program for a machine in such a way that we can imagine that the machine “lives” ? Let us accept the following definition for a model : Life is some game where every participant must die. Then we arrive at the points : (1) The life time can be controlled by amount of food given to a participant. (2) ’Food’ belongs to the game space (a preliminary version of the paper contained a subtitle “Walks of cannibals”). (3) If $F^{(n-1)}$ and $F^{(n)}$ are the amounts of food needed for a participant to live $n-1$ and n time instants, respectively, then at the n-th time instant he has to eat another participant, who has eaten the difference of the food $F^{(n)} - F^{(n-1)}$. (4) The death of participants should be introduced as a necessary condition for existence of the system. It turns out that the trees of class $\mathcal{T}_e^{(n)}$ can be viewed as some mathematical formalization of these rules.

7 Acknowledgement

The author is grateful to Christian Deppe for introducing the Ulam’s problem to him. Interesting discussions with Christian Deppe and Oliver Riordan are highly appreciated. Due to Oliver, a number of small errors contained in the preliminary version both in mathematics and English are corrected. Many thanks to Professor Rudolf Ahlswede for the possibility to do this research in the University of Bielefeld.

Appendix T

T.1 Proof of Theorem and Corollary

Proof of *1.* Obviously, (17) is true for any terminal node : if $\mathbf{c} = \mathbf{1}_j$, then $\mathbf{k} = \mathbf{1}_j$ for all $j = 0, \dots, e$. Suppose that (17) is true for some pair of nodes located at level $n-1$ and containing the vectors $\mathbf{a}, \mathbf{b}$, i.e., if $\mathbf{k}', \mathbf{k}'' \in \mathcal{K}_e^{(n-1)}$ are the distributions of terminal nodes of the corresponding subtrees of class $\mathcal{T}_e^{(n-1)}$, then

$$k_i' = \sum_{j=i}^{e} \left[\begin{array}{c} n-1-j \\ j-i \end{array} \right] a_j, \quad k_i'' = \sum_{j=i}^{e} \left[\begin{array}{c} n-1-j \\ j-i \end{array} \right] b_j, \tag{T.1}$$

for all $i = 0, \dots, e$. If these nodes are sons of a node located at level n, then this node contains the vector $\mathbf{c}$ constructed by (9), and the tree has the distribution of terminal nodes $\mathbf{k} = \mathbf{k}' + \mathbf{k}'' \in \mathcal{K}_e^{(n)}$. Thus, using (9), (14), and (T.1) we obtain

$$\begin{aligned} k_i &= k_i' + k_i'' \\ &= \sum_{j=i}^{e} \left[\begin{array}{c} n-1-j \\ j-i \end{array} \right] (a_j + b_j) \\ &= \sum_{j=i}^{e} \left[\begin{array}{c} n-1-j \\ j-i \end{array} \right] (c_j + c_{j+1}) \\ &= c_i + \sum_{j=i+1}^{e} \left\{ \left[\begin{array}{c} n-1-j \\ j-i \end{array} \right] + \left[\begin{array}{c} n-j \\ j-1-i \end{array} \right] \right\} c_j \\ &= c_i + \sum_{j=i+1}^{e} \left[\begin{array}{c} n-j \\ j-i \end{array} \right] c_j \\ &= \sum_{j=i}^{e} \left[\begin{array}{c} n-j \\ j-i \end{array} \right] c_j, \end{aligned}$$

for all $i = 0, \dots, e$.

Proof of *2.* Obviously, (18) is true for any terminal node : if $\mathbf{k} = \mathbf{1}_i$, then $\mathbf{c} = \mathbf{1}_i$ for all $i = 0, \dots, e$. Suppose that (18) is true for some pair of nodes located at level $n-1$ such that $\mathbf{k}', \mathbf{k}'' \in \mathcal{K}_e^{(n-1)}$ are the distributions of terminal nodes of the corresponding subtrees of class $\mathcal{T}_e^{(n-1)}$ and these nodes contain the vectors

$$a_j = \sum_{i=j}^{e}(-1)^{i+j}\left[\begin{array}{c} n-1-i \\ i-j \end{array}\right] k_i', \quad b_j = \sum_{i=j}^{e}(-1)^{i+j}\left[\begin{array}{c} n-1-i \\ i-j \end{array}\right] k_i'', \quad \text{(T.2)}$$

for all $j = 0, \ldots, e$. If these nodes are sons of a node located at level n, then this node contains the vector $\mathbf{c}$ constructed by (9), and the tree has the distribution of terminal nodes $\mathbf{k} = \mathbf{k}' + \mathbf{k}'' \in \mathcal{K}_e^{(n)}$. Using (9) and (T.2) we have

$$c_e = a_e + b_e = \left[\begin{array}{c} n-1-e \\ 0 \end{array}\right](k_e' + k_e'') = \left[\begin{array}{c} n-e \\ 0 \end{array}\right] k_e.$$

Suppose that (18) is proved for some value $j+1 \in \{1, \ldots, e\}$, i.e.,

$$c_{j+1} = \sum_{i=j+1}^{e}(-1)^{i+j+1}\left[\begin{array}{c} n-i \\ i-j-1 \end{array}\right] k_i.$$

Then using (9), (14), and (T.2) we obtain

$$\begin{aligned}
c_j &= a_j + b_j - c_{j+1} \\
&= \sum_{i=j}^{e}(-1)^{i+j}\left[\begin{array}{c} n-1-i \\ i-j \end{array}\right] k_i - \sum_{i=j+1}^{e}(-1)^{i+j+1}\left[\begin{array}{c} n-i \\ i-j-1 \end{array}\right] k_i \\
&= k_j + \sum_{i=j+1}^{e}(-1)^{i+j}\left\{\left[\begin{array}{c} n-1-i \\ i-j \end{array}\right] + \left[\begin{array}{c} n-i \\ i-j-1 \end{array}\right]\right\} k_i \\
&= k_j + \sum_{i=j+1}^{e}(-1)^{i+j}\left[\begin{array}{c} n-i \\ i-j \end{array}\right] k_i \\
&= \sum_{i=j}^{e}(-1)^{i+j}\left[\begin{array}{c} n-i \\ i-j \end{array}\right] k_i,
\end{aligned}$$

i.e., (18) is proved by induction on $j = e, \ldots, 0$.

Proof of Corollary. If $\mathbf{k} \in \mathcal{K}_e^{(n)}$ is the distribution of terminal nodes of a tree of class $\mathcal{T}_e^{(n)}$ whose root contains the vector $\mathbf{c}$, then using (17) we obtain

$$\sum_{i=0}^{e} k_i 2^i = \sum_{i=0}^{e} 2^i \sum_{j=i}^{e}\left[\begin{array}{c} n-j \\ j-i \end{array}\right] c_j = \sum_{j=0}^{e} c_j \sum_{i=0}^{j}\left[\begin{array}{c} n-j \\ j-i \end{array}\right] 2^i.$$

By Proposition 2, this sum coincides with the sum in the definition of the set $\mathcal{C}_e^{(n)}$ (see (19)), and it is equal to 2^n by Proposition 1.

Appendix L

L.1 Proof of Lemma 1

The *if and only if* statement follows directly from the definitions. To prove (23) let us denote $\mathbf{a} = \mathbf{a}(\mathbf{c}|\boldsymbol{\delta}^0)$ and $\mathbf{b} = \mathbf{b}(\mathbf{c}|\boldsymbol{\delta}^0)$. Since $\mathbf{c} \in \mathcal{R}_e^{(n)*}$, the vector $\mathbf{c}$ also belongs to the set $\mathcal{R}_e^{(n)}$ and we conclude that $N_e(\mathbf{a})$, $N_e(\mathbf{b}) \leq n-1$. If $\mathbf{a} \notin \mathcal{R}_e^{(n-1)*}$, then $N_e(\mathbf{a}+\mathbf{1}_0) \leq n-1$, but the pair $(\mathbf{a}+\mathbf{1}_0, \mathbf{b})$ and the vector $\boldsymbol{\delta}^0 + 2 \cdot \mathbf{1}_0$ lead to the vector $\mathbf{c} + \mathbf{1}_0 \in \mathcal{R}_e^{(n)}$ in accordance with (9). Such a conclusion contradicts the condition $\mathbf{c} \in \mathcal{R}_e^{(n)*}$.

L.2 Proof of Lemma 2

Proof of the *if* statement. The existence of a tree of class $\mathcal{T}_e^{(n)*}$ containing the vector $\mathbf{c}_e^{(n)}(M)$ in the root means that the conditions of Lemma 1 are satisfied for a vector whose e-th component is equal to M. Reference to Lemma 1 completes the proof.

Proof of the *only if* statement. If any integer $m \in \{1, \ldots, M\}$ can be found after n questions, then $M \cdot \mathbf{1}_e \in \mathcal{R}_e^{(n)}$. Thus, $\mathbf{c}^* = c_0^* \cdot \mathbf{1}_0 + M \cdot \mathbf{1}_e \in \mathcal{R}_e^{(n)*}$ for some $c_0^* \geq 0$ and, by Lemma 1, there is a tree of class $\mathcal{T}_e^{(n)*}$ whose root contains $\mathbf{c}^*$. By the corollary from the theorem, $\mathbf{c}^* \in \mathcal{C}_e^{(n)}$, which means that $\mathbf{c}^*$ coincides with $\mathbf{c}_e^{(n)}(M)$. That the distribution of the terminal nodes of the tree is given by $\mathbf{k}_e^{(n)}(M)$ then follows from the theorem.

L.3 Proof of Lemma 3

The restriction $(\Delta 0)$ follows from Proposition 3, the restriction $(\Delta 1)$ coincides with $\delta_j \in \mathcal{D}(c_j)$, $j = 1, \ldots, e$, and $(\Delta 2)$ follows from $\delta_0 \in \mathcal{D}(c_0)$. Therefore, it remains to prove that the constraint $\delta_0 \equiv c_0$ can be omitted. We write

$$c_0 \equiv \sum_{j=1}^{e} c_j V_j^{(t)} \equiv \sum_{j=1}^{e} \binom{t-1}{j} c_j \equiv \sum_{j=1}^{e} \binom{t-1}{j} \delta_j \equiv \delta_0. \tag{L.1}$$

The first relation in (L.1) follows from $\sum_{j=0}^{e} c_j V_j^{(t)} = 2^t$, the second relation follows from the identity $V_j^{(t)} = 2V_j^{(t-1)} - \binom{t-1}{j}$, $j = 1, \ldots, e$, and the third one follows from $\delta_j \equiv c_j$, $j = 1, \ldots, e$.

L.4 Proof of Lemma 4

The proof directly follows from Definitions 2,3, Lemmas 1,2, and identity (21).

L.5 Proof of Lemma 5

Since the last component of the vector $\mathbf{c}$ contained in the root of a tree of class $\mathcal{T}_e^{(n)}$ is at least M, we obtain (31) from the condition $\mathbf{c} \in \mathcal{C}_e^{(n)}$.

L.6 Proof of Lemma 6

The proof is complete if we prove statements (a)–(d) of subsection 5.4.

Proof of (a). Let us fix some sequence $\overline{\mathbf{c}} = (\mathbf{c}_1, \mathbf{c}_2, \ldots)$ constructed in the experiment (E_1)–(E_4) when the partition strategy is defined by (35). Using notations (32)–(34) and the obvious inequalities

$$\xi_{i+1} \le \xi_i, \ \eta_{i+1} \le \eta_i, \ \text{ for all } \ i \ge 1, \tag{L.2}$$

we conclude that $\overline{\mathbf{c}}$ generates a non-increasing sequence $\overline{\mathbf{s}} = (\xi_1 + \eta_1, \xi_2 + \eta_2, \ldots)$. If the current state of the search belongs to the set $\mathcal{G}_{0,\eta}$ for some η, then the experiment goes only through (E_4) and stops after at most t steps. Otherwise, denoting the η-th component of the vector $\mathbf{c}_i$ by $c_{i,\eta}$ for all $i = 1, 2, \ldots$ and $\eta \in \{0, \ldots, e\}$, we obtain

$$\left.\begin{array}{l} \xi_{i+1}, \xi_i > 0 \\ \xi_{i+1} + \eta_{i+1} = \xi_i + \eta_i \\ c_{i+1,\eta_{i+1}} = c_{i,\eta_i} \end{array}\right\} \implies i \in \mathcal{I} \implies i+1 \notin \mathcal{I} \tag{L.3}$$

and

$$\left.\begin{array}{l} \xi_{i+1} > 0 \\ i+1 \notin \mathcal{I} \end{array}\right\} \implies \left[\begin{array}{l} \xi_{i+2} + \eta_{i+2} < \xi_{i+1} + \eta_{i+1} \\ c_{i+2,\eta_{i+2}} < c_{i+1,\eta_{i+1}} \end{array}\right. \tag{L.4}$$

(both equations at the left hand side and at least one inequality at the right hand side should be valid). Really, using (L.2) we have

$$\xi_{i+1} + \eta_{i+1} = \xi_i + \eta_i \implies \xi_{i+1} = \xi_i,\ \eta_{i+1} = \eta_i$$

and if $\xi_i, \xi_{i+1} > 0$, $c_{i+1,\eta_{i+1}} = c_{i,\eta_i}$, then the experiment goes through (E_3) for a given i. Thus $i \in \mathcal{I}$ and, because of the rules of the transfer (30), the restrictions on the 0-th component of the current state of the search $\mathbf{c}$ given in Proposition 5 are satisfied. Hence $i+1 \notin \mathcal{I}$, the experiment goes through (E_4) for the value $i+1$ and either the sum $\xi_{i+1} + \eta_{i+1}$ is decreased or the last component of the state of the search is changed. The statements (L.3), (L.4) show that the sequence $\bar{\mathbf{s}}$ has finite length, and (a) is proved.

Proof of (b). The first $|\mathcal{I}|$ inequalities follow from (L.2) and the observation that $\hat{\bar{\mathbf{s}}} = (\hat{\xi}_1 + \hat{\eta}_1, \ldots, \hat{\xi}_{|\mathcal{I}|} + \hat{\eta}_{|\mathcal{I}|})$ is a subsequence of $\bar{\mathbf{s}}$. The last inequality follows from the first statement of Proposition 5.

Proof of (c). If $\hat{\xi}_k + \hat{\eta}_k$ is odd, then $\hat{\xi}_k < \hat{\eta}_k$, the experiment goes through (E_4) with $i = i_k + 1$ and $\xi_{i+1} + \eta_{i+1} < \hat{\xi}_k + \hat{\eta}_k$. Hence, all other members of $\hat{\bar{\mathbf{s}}}$ having indices $k' > k$ are also less than $\hat{\xi}_k + \hat{\eta}_k$.

Proof of (d). If $\hat{\xi}_k + \hat{\eta}_k$ is even and $\hat{\xi}_k < \hat{\eta}_k$, then using the previous arguments we obtain that $\hat{\xi}_k + \hat{\eta}_k$ occurs in the sequence $\hat{\bar{\mathbf{s}}}$ only once. If $\hat{\xi}_k = \hat{\eta}_k = \eta$, then the experiment can go through (E_4) at most twice, as shown by Proposition 5 : when the current state of the search reaches the set $\mathcal{G}_{\eta,\eta}$ for the first time and when the 0-th component is equal to 2.

L.7 Proof of Lemma 7

The equivalence of the statement that one can solve the searching problem with n questions to the statement that there exists a tree of class $\mathcal{T}_1^{(n)*}$ whose root contains the vector $\mathbf{c}_1^{(n)}(M)$ was established in Lemma 2. Let us suppose that such a tree is given and select some node, which is located at level $t \in \{1, \ldots, n\}$, contains $\mathbf{c}_1^{(t)}(\tilde{M})$, and has sons containing the vectors $\mathbf{c}_1^{(t-1)}(\tilde{M}'), \mathbf{c}_1^{(t-1)}(\tilde{M}'')$ with $\tilde{M}' \geq \tilde{M}''$. Let $\delta = \tilde{M}' - \tilde{M}''$. Then the vector $\boldsymbol{\delta} = (-(t-1)\delta, \delta)$, which determines the partition of $\mathbf{c}_1^{(t)}(\tilde{M})$, satisfies $(\Delta 0)$–$(\Delta 2)$, i.e.,

$$\delta \leq \tilde{M},\ \ (t-1)\delta \leq 2^t - (t+1)\tilde{M}.$$

Thus, similar inequalities with the parameter $\hat{\delta} = \lceil \tilde{M}/2 \rceil - \lfloor \tilde{M}/2 \rfloor$ substituted for δ also hold, and we conclude that if there are trees of class $\mathcal{T}_1^{(t-1)*}$ whose roots contain $\mathbf{c}_1^{(t-1)}(\lceil\tilde{M}/2\rceil)$ and $\mathbf{c}_1^{(t-1)}(\lfloor\tilde{M}/2\rfloor)$, then they can be combined into a tree of class $\mathcal{T}_1^{(t)*}$ whose root contains $\mathbf{c}_1^{(t)}(\tilde{M})$. Furthermore, using the arguments included in the proof of Lemma 2 we obtain that the existence of a tree of class $\mathcal{T}_1^{(t-1)*}$ whose root contains $\mathbf{c}_1^{(t-1)}(\tilde{M}')$ implies the existence of these trees. Hence, we can modify the tree in such a way that the new tree satisfies (H$_c$) for a certain node. Running this procedure for all nodes, we construct a tree with the desired property.

The proof of Corollary is based on the following observation : the statement

$$\begin{cases} \tilde{M} \ \text{ is even} & \Longrightarrow \quad (t+1)\tilde{M} \le 2^t \\ \tilde{M} \ \text{ is odd} & \Longrightarrow \quad (t+1)\tilde{M} \le 2^t - t + 1 \end{cases} \tag{L.5}$$

implies

$$\begin{cases} \lceil\tilde{M}/2\rceil \ \text{ is even} & \Longrightarrow \quad t\cdot\lceil\tilde{M}/2\rceil \le 2^{t-1} \\ \lceil\tilde{M}/2\rceil \ \text{ is odd} & \Longrightarrow \quad t\cdot\lceil\tilde{M}/2\rceil \le 2^{t-1} - t + 2. \end{cases} \tag{L.6}$$

Really, using (L.5) we obtain $\lceil\tilde{M}/2\rceil \le 2^{t-1}/(t+1) + 1$ and check that

$$\frac{2^{t-1}}{t+1} + 1 \le \frac{2^{t-1}}{t} - \frac{t-2}{t}, \quad \text{for all } \ t \ge 8.$$

Therefore the observation is true for $t \ge 8$. If $t = 3, \ldots, 7$, then inequalities (L.5) give the values $\tilde{M} = 2, 2, 4, 8, 16$, i.e., $\log\tilde{M}$ is an integer and (L.5) also implies (L.6). Hence, (26) implies the necessary and sufficient condition for it to be possible to partition the vector $\mathbf{c}_1^{(n-1)}(\lceil M/2\rceil)$. Clearly, a similar conclusion is also true for the vector $\mathbf{c}_1^{(n-1)}(\lfloor M/2\rfloor)$. Repeating these considerations for all $t = n-1, \ldots, 1$, we prove the statement.

Appendix P

P.1 Proof of Proposition 1

Any tree of class $\mathcal{T}_e^{(n)}$ having the distribution of terminal nodes $(k_0, \ldots, k_e)$ can be constructed by a modification of the complete n-level binary tree in such a way that $k_i 2^i$ nodes of level 0 are replaced with k_i terminal nodes located at level i. Using this argument for all $i \in \{1, \ldots, e\}$ we can express the total number of terminal nodes of the complete tree, which is equal to 2^n, as the sum $\sum_{i=0}^{e} k_i 2^i$.

P.2 Proof of Proposition 2

Proof of (15). We use induction on $s = n+i$. Obviously, (15) is true when $s = 1$. Suppose that (15) is also true for all pairs (n', i') with $n'+i' = s-1 \geq 1$ and fix a particular pair (n, i) with $n + i = s$. Then applying the induction hypothesis to the pairs $(n-1, i)$ and $(n, i-1)$ we have

$$\begin{bmatrix} n-1 \\ i \end{bmatrix} = \binom{n+i-2}{i}, \quad \begin{bmatrix} n \\ i-1 \end{bmatrix} = \binom{n+i-2}{i-1}.$$

Using these equations in (14) and the binomial recursion (12) we obtain (15).

Proof of (16). We use (11), (12) and write

$$\begin{aligned} \sum_{i=0}^{j} \binom{n}{i} &= \sum_{i=0}^{j} \binom{n-1}{i} + \sum_{i=1}^{j} \binom{n-1}{i-1} \\ &= 2\sum_{i=0}^{j} \binom{n-1}{i} - \binom{n-1}{j}. \end{aligned} \tag{P.1}$$

Similar transformations lead to the equation

$$\sum_{i=0}^{j} \binom{n-j+i-1}{i} 2^{j-i} = 2\sum_{i=0}^{j} \binom{n-j+i-2}{i} 2^{j-i} - \binom{n-1}{j}.$$

Hence, using (15) we obtain

$$\sum_{i=0}^{j} \begin{bmatrix} n-j \\ i \end{bmatrix} 2^{j-i} = 2\sum_{i=0}^{j} \begin{bmatrix} n-1-j \\ i \end{bmatrix} 2^{j-i} - \binom{n-1}{j}, \tag{P.2}$$

and (16) follows from (P.1), (P.2) by induction on n.

P.3 Proof of Proposition 3

To prove the statement we use the following identity : for all $\mathbf{c}$,

$$\sum_{j=0}^{e} (c_j + c_{j+1}) V_j^{(t-1)} = \sum_{j=0}^{e} c_j V_j^{(t)}.$$

Therefore, using (2) and the condition $\mathbf{c} \in \mathcal{C}_e^{(t)}$ we write

$$\begin{aligned} 2\sum_{j=0}^{e} a_j V_j^{(t-1)} &= \sum_{j=0}^{e} c_j V_j^{(t)} + \delta_0 + \sum_{j=1}^{e} \delta_j (V_j^{(t-1)} - V_{j-1}^{(t-1)}) \\ &= 2^t + \sum_{j=0}^{e} \binom{t-1}{j} \delta_j. \end{aligned}$$

P.4 Proof of Proposition 4

Suppose that a vector $\mathbf{c}$ satisfying the conditions of the proposition has s positive components at positions $1, \ldots, e$ and denote these positions by $j_1 < \ldots < j_s$. Then $j_s = \eta$ and

$$\binom{t-1}{j_i} \geq \binom{t-1}{j_{i-1}}, \quad i = 2, \ldots, s. \tag{P.3}$$

Inequality (P.3) is obviously true for $t - 1 \geq 2\eta$ since $\binom{t-1}{\eta}, \ldots, \binom{t-1}{1}$ is a decreasing sequence in this case. If $\eta > (t-1)/2$, then $c_\eta = 1$ because $\mathbf{c} \in \mathcal{C}_e^{(t)}$ implies $c_\eta V_\eta^{(t)} \leq 2^t$ and $\eta > (t-1)/2$ implies $\eta \geq t/2$. A similar argument leads to the equations $c_j = 0$ for all j such that $t - 1 - \eta < j < \eta$.

Using (P.3) and induction on j decreasing from e to 1 one can easily check that

$$\left| \sum_{i=j}^{e} \binom{t-1}{j} \delta_i^* \right| \leq \binom{t-1}{\eta}$$

for all $j = 1, \ldots, e$.

P.5 Proof of Proposition 5

All statements directly follow from (2) and (32)–(35). To check the inequality for a_0 and a_η in the last statement we note that as $|\delta_\eta| = |\delta_\eta^*| \leq 1$,

$$2a_\eta = 2\lceil c_\eta/2 \rceil \leq c_\eta + 1$$

and

$$2a_0 \geq c_0 + \delta_0 \geq c_0 - \binom{t-1}{\eta} \geq \binom{t-1}{\eta}(c_\eta - 3),$$

where the second inequality is based on the arguments used in the proof of Proposition 4. Hence

$$2a_0 \geq \binom{t-1}{\eta}(2a_\eta - 4) \geq \binom{t-2}{\eta}(2a_\eta - 4).$$

Table 1: Sets $\mathcal{R}_1^{(0)*}, \ldots, \mathcal{R}_1^{(5)*}$.

n	$\mathcal{R}_1^{(n)*}$
0	$\{(1,0),(0,1)\}$
1	$\{(2,0)\}$
2	$\{(4,0),(1,1)\}$
3	$\{(8,0),(4,1),(0,2)\}$
4	$\{(16,0),(11,1),(6,2)\}$
5	$\{(32,0),(26,1),(20,2),(14,3),(8,4)\}$

Table 2: Structure of a tree of class $\mathcal{T}_2^{(8)*}$ whose root contains the vector $\mathbf{c}_2^{(8)}(4) = (108, 0, 4)$.

Level	Label	Parameters of the node						# nodes	Reference	
t	$\ell(\mathbf{c})$	c_0	c_1	c_2	k_0	k_1	k_2		$\ell(\mathbf{a})$	$\ell(\mathbf{b})$
8	0	108	0	4	**192**	**24**	**4**	1	1	1
7	1	54	2	2	96	12	2	2	2	2
6	2	28	2	1	48	6	1	4	3	4
5	3	16	0	1	22	3	1	4	5	6
5	4	14	3	0	26	3	0	4	7	6
4	5	5	0	1	8	2	1	4	8	9
4	6	11	1	0	14	1	0	8	9	10
4	7	6	2	0	12	2	0	4	9	9
3	8	1	0	1	2	1	1	4	11	12
3	9	4	1	0	6	1	0	20	12	13
3	10	8	0	0	8	0	0	8	13	13
2	11	**0**	**0**	**1**	0	0	1	**4**		
2	12	1	1	0	2	1	0	24	14	15
2	13	4	0	0	4	0	0	36	15	15
1	14	**0**	**1**	**0**	0	1	0	**24**		
1	15	2	0	0	2	0	0	96	16	16
0	16	**1**	**0**	**0**	1	0	0	**192**		

Table 3: Partition of some states of the search with the transfer of the state $(6,0,2,1)$ to level 8 and the state $(1,3,0,0)$ to level 5.

t	c_0	c_1	c_2	c_3	δ_0	δ_1	δ_2	δ_3	a_0	a_1	a_2	a_3
7	**6**	0	2	1	$(-\mathbf{10})$	0	+2	−1	–	–	–	–
8	89	0	2	1	−7	0	+2	−1	41	0	3	0
7	41	0	3	0	−15	0	+1	0	13	1	2	0
6	13	1	2	0	−5	+1	0	0	4	2	1	0
5	4	2	1	0	−2	+2	−1	0	1	3	0	0
4	**1**	3	0	0	$(-\mathbf{3})$	+1	0	0	–	–	–	–
5	14	3	0	0	−4	+1	0	0	6	2	0	0

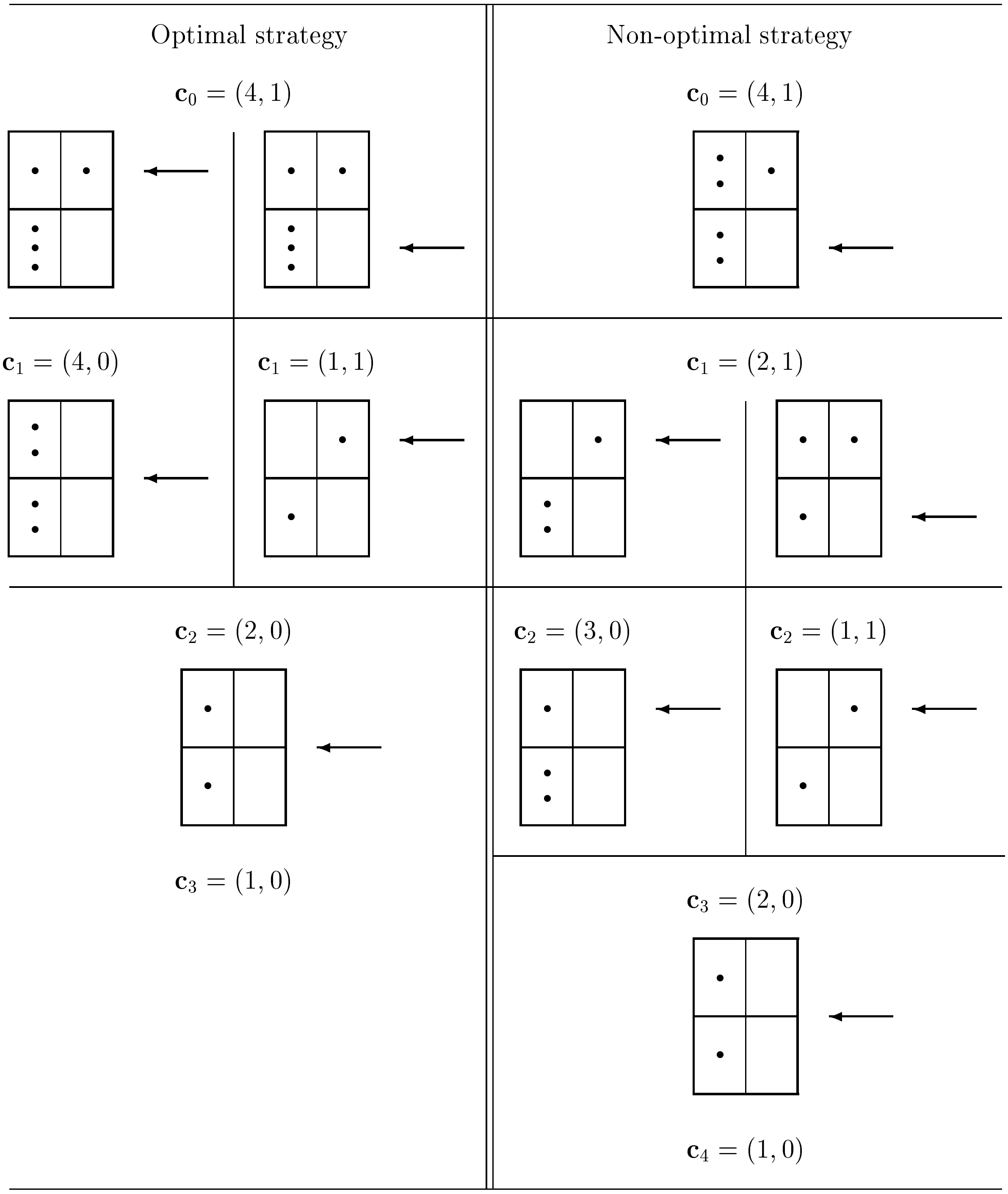

Figure 1: Possible moves of the game for $e = 1$ when the initial position is specified by the vector $\mathbf{c}_0 = (4, 1)$.

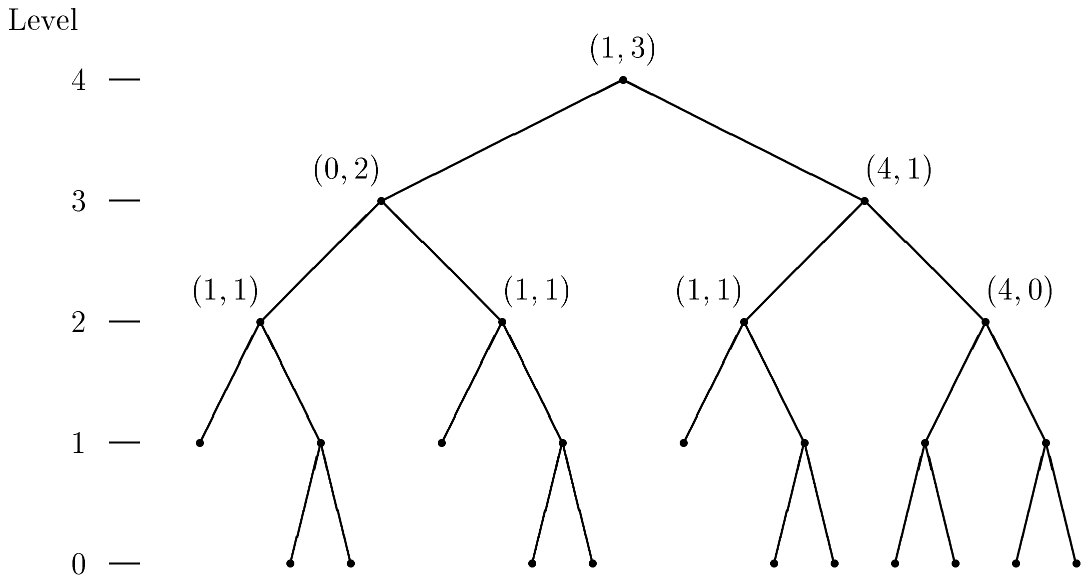

Figure 2: A tree of class $\mathcal{T}_1^{(4)}$ whose root contains $\mathbf{c} = (1, 3) \in \mathcal{C}_1^{(4)}$, the terminal nodes at levels 0 and 1 contain the vectors $(1, 0)$ and $(0, 1)$, respectively, and $\mathbf{k} = (10, 3) \in \mathcal{K}_1^{(4)}$ is the distribution of terminal nodes.

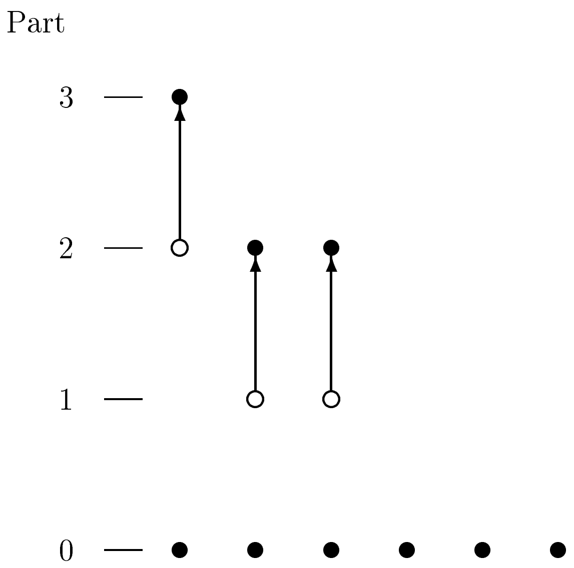

Figure 3: Constructing the vector $\mathbf{c} = (6, 0, 2, 1) \in \mathcal{C}_3^{(7)}$ from $(a_0+b_0, \ldots, a_3+b_3) = (6, 2, 3, 1)$.

References

[1] M. Aigner, "Searching with lies," *J. Combin. Theory Ser. A,* vol.74, pp.43-56, 1996.

[2] G.E. Andrews, *The Theory of Partitions.* (Encyclopedia of Mathematics and its Applications). New York : Addison-Wesley, 1976.

[3] V.B. Balakirsky, "Coding algorithms realizing a "component–wise" computer arithmetic," submitted to *Inform. Comput.,* Feb. 1998.

[4] E.R. Berlekamp, "Block coding for the binary symmetric channel with noiseless, delayless feedback," in : *Error-Correcting Codes,* pp.61-85, New York : Wiley, 1968.

[5] J. Czyzowicz, D. Mundici, and A. Pelc, "Ulam's searching game with lies," *J. Combin. Theory Ser. A,* vol.52, pp.62-76, 1989.

[6] C. Deppe, "Solution of Ulam's searching game with three lies or an optimal adaptive strategy for binary three-error-correcting-codes," *SFB-343 "Diskrete Strukturen in der Mathematik",* Preprint 98-036, Bielefeld, 1998.

[7] R.G. Gallager, *Information Transmission and Reliable Communication.* New York : Wiley, 1968.

[8] W. Guzicki, "Ulam's searching game with two lies," *J. Combin. Theory Ser. A,* vol.54, pp.1-19, 1990.

[9] E.L. Lawler and S. Sarkissian, "An algorithm for "Ulam's Game" and its application to error correcting codes," *Inform. Proc. Let.,* vol.56, pp.89-93, 1995.

[10] A. Pelc, "Solution of Ulam's problem on searching with a lie," *J. Combin. Theory Ser. A,* vol.44, pp.129-140, 1987.

[11] J.P.M. Schalkwijk, "A class of simple and optimal strategies for block coding on the binary symmetric channel with noiseless feedback," *IEEE Trans. Inform. Theory,* vol.17, no.3, pp.283-287, 1971.

[12] J. Spencer, "Ulam's searching game with a fixed number of lies," *Theor. Comput. Sci.,* vol.95, pp.307-321, 1992.

[13] S.M. Ulam, *Adventures of a Mathematician.* New York : Scribner, 1976.