Olympiads in Informatics, 2020, Vol. 14, 143–150
© 2020 *IOI, Vilnius University*
DOI: 10.15388/ioi.2020.11

Pattern Recognition and Related Topics of Olympiad Tasks

Pavel S. PANKOV, Azret A. KENZHALIEV
Institute of Mathematics, Kyrgyzstan
Korea Advanced Institute of Science and Technology (KAIST).
e-mail: pps5050@mail.ru, azret.kenzhaliev@gmail.com

Abstract. In most of tasks proposed for olympiads in informatics inital data are taken arbitrary from any ranges such that a "brute force" solution (close to immediate translation of condition of the task into an algorithmical one) can pass tests with initial data from narrow ranges, and the task itself is to improve that solution. We make a survey of types of tasks where inital data are too vast to use a "brute force" but they are announced to be connected and the task itself is to extract necessary information in an optimal way. We consider tasks on recognition and on restoration of data from a unified point of view. We also make an attempt to describe various types of tasks formally.

Keywords: Olympiad, informatics, tasks on recognition, reactive tasks.

Introduction

In most of tasks proposed for olympiads in informatics inital data are taken arbitrary from any ranges such that a "brute force" solution (close to immediate translation of condition of the task into an algorithmical one) can pass tests with initial data from narrow ranges, and the task itself is to improve that solution. We make a survey of types of tasks where inital data are too vast to use a "brute force" but they are announced to be connected because they are images of certain virtual objects. The task itself is to extract necessary information about these objects in an relatively optimal way. We consider tasks on recognition and on restoration of data from a unified point of view. We also make an attempt to describe various types of tasks formally.

1. Definitions

A common reactive task can be presented as follows. All sets are meant to be finite.

Definition 1. Two computer-presentable sets (large) X and Y and a computable function $F : X \to Y$ are described (preferably, verbally). Write a program $P : X \to Y$ which implements the function F during a prescribed time (traditionally 1 second), also with restriction on memory used. Total amount of points is announced. Usually, some subtasks are given too in the following form: some sets $X_1 \subset X_2 \subset \dots \subset X_k \subset X$ are described and amounts of points (in increasing order) for corresponding programs $P_j : X_j \to Y$ are also announced.

If the function F is defined by means of any virtual objects of a set U not belonging to X and X itself contains images of these objects we propose the following construction for some types of tasks:

Definition 2. Three computer-presentable sets (large) U, (large) X and Y, a computable function $G : U \to Y$ ("extraction of information") and a computable dyadic predicate $Q : U \times X \to \{true, false\}$ are described.

Conditions of correctness:

(*) Either $(\forall x \in X)(\exists u \in U)Q(u, x)$ or output "*No solution*" is also permitted.

(**) $(Q(u_1, x_1) \wedge Q(u_2, x_2) \wedge (G(u_1) \neq G(u_2))) \Rightarrow (x_1 \neq x_2)$.

Write a program $P : X \to Y$ such that $Q(u, x) \Rightarrow (G(u) = P(x))$, working during a prescribed time, also with restriction on memory used. Total amount of points is announced.

Some subtasks are given too in the following form: some sets $U_1 \subset U_2 \subset \dots \subset U_k \subset U$ (also, $Y_1 \subset Y_2 \subset \dots \subset Y_k \subset Y$) are described and amounts of points (in increasing order) for corresponding programs $P_j : X \to Y$ (or $P_j : X \to Y_j$) are also announced.

The sense of such tasks is that the evident algorithm

$$P_0(x) := (\textit{for all } u \in U)\ (\textit{if } Q(u, x) \textit{ then output } G(u))$$

is obviously too slow.

Remark 1. Restoration of u by x is impossible sometimes while value of some function (G) of u can be found by x.

The simplest example is the following:

Task 1. Let U be the set of binary 2×2 -matrices $u = \{u_{ij} : i, j = 1, 2\}$. Given four sums $S_4(u) := \{u_{11} + u_{12}; u_{21} + u_{22}; u_{11} + u_{21}; u_{12} + u_{22}\}$, found $G(u) := abs(det(u))$.

Here X is the set of tuples of four integer numbers $x = \{x_1; x_2; x_3; x_4\}$ less than 3;

$$Q(u, x) := (u_{11} + u_{12} = x_1) \wedge (u_{21} + u_{22} = x_2) \wedge (u_{11} + u_{21} = x_3) \wedge (\, u_{12} + u_{22} = x_4).$$

Consider two elements of U : $u_1 = \{\delta_{ij} : i, j = 1, 2\}$ and $u_2 = \{1 - \delta_{ij} : i, j = 1, 2\}$ where δ_{ij} is the Kronecker symbol. As $S_4(u_1) = S_4(u_2)$, u cannot be found. Nevertheless $G(u) = 1$ can be found uniquely.

Remark 2. From the standpoint of Definitions 1 and 2 all tasks are divided into "common" (Y is as large as X), "classification" (Y is small) and "alternative" ($Y = \{YES, NO\}$).

2. Examples of Tasks on Extraction of Information

The following task is the utterly simplified Task Character Recognition, IOI-1997.

Task 2. The file FONT.DAT contains of 26 ideal character images (abcdefghijklm nopqrstuvwxyz) as 20 × 20 binary arrays. "Corruption" is an inverting of no more than 50 of signs in an image.

Write a program that restores corrupted images.

Input: a 20 × 20 binary array being one of 26 images corrupted.

Output: one of abcdefghijklmnopqrstuvwxyz.

The set U contains 26 elements $\{u_1, ..., u_{26}\}$.

Condition (**): the Hamming distance *H-dist* between each two different ideal character images is greater than 100.

(The mentioned task permits some kinds of "Corruption" which seem difficult to be expressed by any "distance").

Develop a task due to Definition 2:

Task 2a. … Input: a 40 × 40 binary array being one of 26 images corrupted and shifted …

(easy)

Task 2b. … Input: a 40 × 40 binary array being one of 26 images shifted and corrupted …

(difficult)

Condition (**): the *H-dist* between each shifts into 40 × 40 binary array of two different ideal character images is greater than 100.

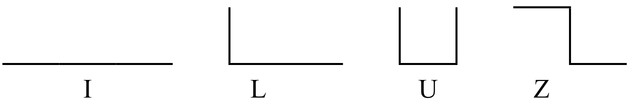
By Definition 2: The set U consists of all shifts of 26 elements $\{u_1, ... , u_{26}\}$ into 40 × 40 binary array. The set X contains 40 × 40 binary arrays;

$X := \{x : (\exists u \in U)(H\text{-}dist(u, x) \leq 50)\}$.

$Q(u, x)$:= "$H\text{-}dist(u, x) \leq 50$". The function G transforms an image to a letter.

The following simple task (the Kyrgyzstan quarterfinal of the ICPC administered by the ACM, Bishkek, November 2015):

Task 3. Given a sequence of four triples of natural numbers less than 100 (four distinct integer points) in space presenting a broken line of three unit links and the following flat broken lines denoted with letters:



If a given broken line can be superposed with one of these then output one the corresponding letter, else output Unknown.

This task has many solutions (calculate the square of distance between the first and the last points …; calculate overall dimensions of four points …; compare the second vector-link with the first one …) nevertheless all they demand any resourcefulness and cannot be reduced to "brute force".

This task demonstrates the following features.

Given points are connected (the differences of the second and the first etc. have the length one) a proiri.

The estimation "less than 100" is not sufficient in order to search for a solution in contrast to other tasks where such estimations define their difficulty.

The set of such sequences is subdivided into five subsets (five elements of *Y*) defined non-formally.

We found an only mention of tomography in Olympiad tasks:

Task 4. Problem Giza (FARIO 2007), a brief description after (Burton *et al.*, 2008).

"This task is a discrete 2-dimensional version of the general tomography problem." To recover a secret image formed from grid cells, given only the cell counts in each of 10 rows, 10 columns and 19 diagonals parallel to the main one.

Task 5 (National Olympiad in Kyrgyzstan, March 2019).

Let us call the letter T a symmetrical figure consisting of two contiguous narrow strips (width is not less than 3, height is not less than 2).

In a 8 × 9 binary array one, two or three (non-touching) letters T were depicted. The numbers of '1' bits in each of the 8 vertical columns [v-sums] and in each of the 9 horizontal lines (bottom to top) [h-sums] were detected using X-rays. For given data, find the number N of letters T and the width and height of each letter. If this is not possible, output zero.

Input: two lines of words of decimal digits; the length of the first is 8, the length of the second is 9.

Output: In the first line: the number N in 0 … 3.

If N > 0, then in each of the next N lines: two natural numbers (width and height) of the letter T, separated by a space. If N > 1 then output the data in lexicographical order: by greater width; by greater height (if there are the same, then output each one).

Example (pulled in line). Input: 01141261; 011112415 Output: 2; 5 4; 3 6

The peculiarity of this task (as well as Task 1) is that two different arrangements can yield a same input, for instance: 01411410; 111301113. Nevertheless, the answer is unique: 2; 3 4; 3 4

By Definition 2: The set *U* consists of all possible positions of 1 ... 3 "T"s on 8×9 binary array. The set *X* contains tuples of 8- and 9- words of digits; $Q(u, x)$:= *"x contains v-sums and h-sums of u"*. $X := \{x: Q(u, x)\}$.

The function *G* transforms an image to the list of "T"s that generated it.

Beginning of a possible algorithm for Task 5.

Subdivide (mentally) each "T" into the horizontal *cap* (its width is odd and not less than 3) and the vertical *leg* (height is not less than 1).

Lemma. There are no traces of three T in an h-sum (because “8” is too narrow).

Applying Algorithm-5 “If (h-sum = 5) or ((h-sum = 6) and (next h-sum > 0)) then there is a 5-cap in this line otherwise there is not” to all lines we distinguish all 5-caps.

Applying Algorithm-3 “If (h-sum = 6) then there are two 3-caps; if ((h-sum = 3 or 4) then there is one 3-cap in this line otherwise there is not” to the rest of lines we distinguish all 3-caps.

Sketch of proof of (**). Within the initial data, there are the numbers of 3-caps and 5-caps and at most three lines for them. Changing lengths of legs causes changings in v-sums necessarily.

(The jury is to write down such a proof. The contestant thinks it swiftly, together with writing a program).

Ending of the algorithm for Task 5.

As there are only a few possible positions for caps, a “brute force” search for lengths of legs from each of such positions ends the solution.

Task 6 (National Olympiad in Kyrgyzstan, March 2019, improved).

By integer numbers A, B, C, D in 0 ... 9, for the function $F(X) = ((AX + B)X + C)X + D$ the numbers F(1), F(2), F(3), F(4) were calculated and one of them was changed by less than 3. By these four numbers find F(5).

3. Types of Metrical and Kinematic Spaces to Be Used in Tasks

Many of Olympiad tasks involve metrics (or distances between objects as its equivalent) or motion, evidently or latently. By our experience, changing Euclidean space to any other type of space (Borubaev *et al.*, 2003) can be made in many tasks, transforms a simple task into a complicated one (although its formal complexity does not increase) and is interesting and difficult for contestants.

Usually in tasks, 2D-Euclidean space (plane) is modified by means of “teleportation” (some given pairs of points are glued). We survey more regular constructions below.

By the way, most of spaces mentioned below were patented as variations of chess.

We will consider only integer points and metrics and motion of Manhattan type.

As a base, we take Square(s) $0 \le X \le 19$, $0 \le Y \le 19$ and Cube(s) $0 \le X \le 19$, $0 \le Y \le 19$, $0 \le Z \le 19$.

2D-cylinder: glue the side X = 0 of Square with the side X = 19 of Square. For example, the points (0; 5) and (19; 5) coincide; *dist*((17; 6), (2; 5)) = 5.

Many programmers invented the following space independently but we could not find tasks in informatics with it:

2D-torus: glue the side X = 0 of Square with the side X = 19 of Square and the side Y = 0 of Square with the side Y = 19 of Square. For example, the points (5; 19) and (5; 0) coincide; *dist*((17; 16), (2; 3)) = 10.

2D-Moebius band: glue the side X = 0 of Square with the reversed side X = 19 of Square as follows: (0; 0) coincides with (19; 19); (0; 1) coincides with (19; 18) etc.

2D-Riemann surface of square root:

Task 7 (National Olympiad in Kyrgyzstan, March 2019).

Wizard put Square$_1$ and Square$_2$ together, cut section on both ones from the center till the point (9.5; 0) and glued the left side of the section of Square$_1$ with the right side of the section of Square$_2$ and the right side of the section of Square$_1$ with the left side of the section of Square$_2$. For example, $dist((9; 5) - 1, (10; 5) - 2) = 1$…. Find the distance between given two points on these Squares.

3D-torus is made of Cube by means of gluing all opposite facets.

There also exists the 3D-space "Multistory" of some floors ($Cube_1$, $Cube_2$, ..., $Cube_k$) connected by a (prompt) elevator ($Cube_0$) only with $Cube\text{-}Door_0$ (X = 0) matching with all $Cube\text{-}Door_j$ $j = 1 \ldots k$ (X = 19).

Metrics in this space is presented as follows. If two points P_1 and P_2 belong to a same $Cube_j$ then $dist(P_1, P_2)$ is defined as usually; if $P_1 \in Cube_0$ and $P_2 \in Cube_j$,$j > 0$, then $dist(P_1, P_2)$ is defined in the gluing them; if $P_1 \in Cube_i$ and $P_2 \in Cube_j$ $(0 < i < j \le k)$ then

$$dist(P_1, P_2) := min\{dist(P_1, P) + dist(P, P_2) : P \in Door\}.$$

3. Ways to Generate Tasks on Extraction of Information

In this section we shall not consider the full procedure of creating a task. It was considered in details in Kemkes *et al.* (2007), Diks *et al.* (2008), Burton *et al.* (2008) and other publications. We propose some schemas being generations of examples in Section 2.

Remark 3. Each such task can be formulated in two ways: with "structured" and "non-structured" initial data. For instance, a complicated version of Task 3:

Task 8. Given a set of four triples of natural numbers less than 100 (four distinct integer points) in space such that after some permutation they form a broken line of three unit links …

(A proper solution is not necessarily to find such permutation: the method "calculate overall dimensions of four points …" can be applied without permutation).

Remark 4. Relatively simple tasks in multidimensional spaces are preferred because they demand imagination and practically exclude "exhaustive search".

The following task generates Task 2 and Task 6.

General task 9. There are defined a metric *dist* and a family of k connected subsets $V_1, ..., V_k$ of the set X. It is guaranteed that $dist(V_i, V_j) > 2M\delta_{ij}$, $M > 0$. Given $x \in X$, find $i \in 1 \ldots k$: $dist(V_i, x) \le M$.

Idea of solution: using connectivity, organize effective search of such $v_j \in V_j$ that $dist(v_i, x)$ is as small as possible.

The idea of detecting compact objects by their integral indexes was proposed by A.N. Tikhonov in 1943 for detecting ore bodies underground and further was developed for medicine (tomography). We adapt it to "integer data" and, due to Remark 1, change detection of object itself to detection of some its indexes, as in Task 5.

General task 10. Withih frames of Definition 2, elements $u \in U$ are presented as sets of integer numbers. Elements $x \in X$ consist of certian sums of these numbers for all $u \in U$. Conditions on U are such that (**) takes place.

Example of task of Yes/No type where given information yields an effective solution.

Task 11. Given a sequence of $K \in 4 \ldots 10^4$ pairs of even integer numbers presenting a non-self-crossing circular broken line of links of length 2 and a pair of odd integer numbers presenting a point on a plane. Is this point within the domain bounded with this broken line? [It cannot be on the line].

The first step of the solution is detecting overall dimensions of the domain ($O(K)$ operations).

The main constituent of any algorithm of the solution is detecting whether given "odd" segment of length 2 crosses the broken line ($O(K)$ operations).

Solution 1 is the common search in depth for the growing graph of "odd" segments of length 2 beginning from the given point until either the growth stops or the graph reaches one of the overall dimensions ($O(K^4)$ operations).

But Solution 2 is the following: move along a straight line to one of the overall dimensions and count a crossing of the broken line ($O(K^2)$ operations). If the total number of crossings is odd then the point is within the domain else it is without.

General interactive task 12. A class of images is described and a procedure for queries is accessible. There is a hidden image in this class. By sequence of queries (or of limited number of queries) detect the image.

This general task also splits into two: all images are transformations of a "basic" one; images are sufficiently different.

Task 13. Given a finite set S of integer points on a plane (in a space, in 4D-space…).

Do they belong to one straight line? Do they belong to two straight lines?

Possible solution.

Denote $D[I, J] := \{Z \in S\textit{: points } Z, ZI, ZJ \textit{ are collinear}\}$.

```
1) Take two different points Z1 and Z2 in S.
2) If $D[1, 2] = S$ then output "one straight line"; end
   else
3) Let $Z3$ be no collinear with Z1 and Z2.
4) If $D[1, 2] \cup D[1, 3] = S$ or $D[1, 2] \cup D[2, 3] = S$ or $D[1, 3] \cup D[2, 3] = S$ then
   output "two straight lines"; end
   else output "NO"; end.
```

Mention two popular tasks:

General task 14. Given a word of brackets (or: brackets and square brackets). Is such arrangement of brackets correct (can be obtained from any correct arithmetical expression)?

General Task 15. Given an expression containing digits, signs "+", "–", brackets. Is it written correctly? If it is written correctly then what is its value?

4. Conclusion

We hope that successful application of mathematical schemes and “general tasks” proposed above would clarify distinctions between warious types of tasks yield new tasks with “short and elegant formulation” (Dagienė *et al.*, 2007), and being interesting to solve. This would enlarge the scope of tasks involved into olympiads in informatics.

References

Borubaev, A.A., Pankov, P.S., Chekeev, A.A. (2003). *Spaces Uniformed by Coverings*. Hungarian-Kyrgyz Friendship Society, Budapest (*Chapter 4. Constructive and Computer Presentations of Uniform Spaces*).

Burton, B.A. Heron, M. (2008). Creating Informatics Olympiad Tasks: Exploring the Black Art. *Olympiads in Informatics: Country Experiences and Developments*, 2, 16–36.

Dagienė, V., Skupienė, J. (2007). Contests in programming: quarter century of Lithuanian experience. *Olympiads in Informatics: Country Experiences and Developments*, 1, 37–49.

Diks, K., Kubica, M., Radoszewski, J., Stencel, K. (2008). A proposal for a task preparation process. *Olympiads in Informatics: Tasks and Training*, 2, 64–74.

Kemkes, G., Cormack, G., Munro, I., Vasiga, T. (2007). New task types at the Canadian computing competition. *Olympiads in Informatics: Country Experiences and Developments*, 1, 79–89.

P.S. Pankov (1950), doctor of physical-math. sciences, prof., corr. member of Kyrgyzstani National Academy of Sciences (KR NAS), was the chairman of jury of Bishkek City OIs, 1985–2013, of Republican OIs, 1987–2012, the leader of Kyrgyzstani teams at IOIs, 2002–2013, 2018–2019. Graduated from the Kyrgyz State University in 1969, is a head of laboratory of Institute of mathematics of KR NAS.

A.A. Kenzhaliev (1999). Bronze medal at IOI’2016. Student of Korea Advanced Institute of Science and Technology (KAIST).