Naturalness in Tasks for Olympiads in Informatics

Pavel S. Pankov

(Kyrgyzstan)

Remark. This paper was written independedly of

Burton B.A., Hiron M. Creating informatics olympiad tasks: exploring the black art, 16-36,

but considers the same items in general. Comparison of approaches of this paper and ours demonstrates both similar proposals and differences.

Abstract

There are two main ways to invent tasks for Olympiads of high level: one way is to invent or choose an effective algorithm and compose corresponding subject, and another way is to think out a real situation or task and try to formalize it. The second way is more difficult because it is multi-stage: the author needs to find some effective algorithm for a task obtained; if the best algorithm is obvious or the only algorithm seems to be exponential then we need to rework the formulation, etc. But by our opinion the second way is preferable because it can yield original tasks with natural, short and elegant formulation and give less advantage to experienced participants. We shall consider the second way in detail in this paper.

1.Natural Ways to Generate Tasks

Since we have not found publications on generating of task ideas in informatics, we shall review a way to generate ideas basing on actions in spaces which we call *natural*, and include our own experience.

Choosing a space

Traditionally:

S1) integer numbers (one-dimensional grid);

S2) pairs of integer numbers (plane grid) (it is introduced as "grid of streets in the host town of the olympiad");

S3) triples of integer numbers (space grid);

S4) a graph;

S5) space of solids rolling on a plane;

S6) ring (a segment with glued ends) with finite number of elements.

We propose also:

S7) two connected rings (figure-of-eight); it is a kind of S4 but the general graph demands a vast description since it is self-explanatory.

S8) integer grids on non-Euclidean spaces, e.g. Topological torus, Moebius band.

Choosing actors (moving, changing objects)

Involving more than one actor provides *a game*. But many interesting tasks with more than one actor could be generated without the idea of being games, for instance, by means of cooperation of actors. Something more, the organization connected with proposal of a game-task during an OI is too difficult. But some games can be imitated by means of formulating the aim of task as minimax.

Traditionally we have the following configurations of actors:

A1) a point is used as an actor (a point can move by 1 or "jump" but rules of jumping must be very simple);

A2) some rectangles.

We propose also

A3) two or three points (or many points with very simple conditions);

A4) moving "train" of a length of one edge or one arc within S4;

A5) moving "train" of a given length within S2, S6, S7.

Choosing actions

The main action proposed is natural *moving*. Moving of a train along a graph or a grid is a consecutive passing its vertices by the head of the train, by its intermediate points (if its length is greater than 1) and by its tail. Simple *cutting*, *gluing*, *deleting* and *adding* are also natural actions.

Choosing conditions, restrictions and obstacles

Conditions may be natural, i.e. actors must | cannot

C1) coincide C2) pass C3) overlap C4) touch C5) be seen C6) cross C7) cross itself (for a train).

For A3 natural conditions are also

C8) be near / far each from other.

Another natural kind of restrictions is the following:

C9) an actor can make only a given number of steps.

Traditional type of obstacles as a labyrinth but some points or rectangles can replace it.

Aims and composing of tasks

G1) to reach / build / compose something in a minimal number of steps;

G2) to build / compose the least / greatest object;

G3) to find the shortest / longest way;

G4) to catch / to escape from another actor in a minimal time / with minimal number of steps or expenditures for all possible actions of another actor (i.e. if its behavior is optimal).

2. Grading System

If the author cannot find the best algorithm then the “natural” tasks give the following ways: *Open-Ended Tasks* or

“Evident” Solutions

For “natural” tasks the answer is "seen" by a human for all initial data within given restrictions. The author may compose as many sufficiently different cases as necessary covering all aspects of the task. A good solution (algorithm) by a competitor must solve many of them. In other words, we propose the hypothesis: if the answers for all initial data are evident for a human then there exists a good algorithm solving the task on the modern computer during an appropriate time.

3. Examples of Tasks

Task 1. Given a graph, its vertices are "houses" (less than 7). The Instrument has counted mice under each of houses at different moments. During all this measuring, each mouse could pass to another neighbor house only once. Write a program to find the least possible number of all mice.

Task 2. A graph is given. Firstly, the head *H* and the tail *T* of a train are in two neighbor vertices. Write a program finding one of the shortest ways to be passed by the train (moving forward only) in order to put its head to the primary position of *T* and its tail to one of *H*.

Task 3. Let the streets in the city form a rectangular grid …. The firm Logic [sponsor] is situated at a given crossing (*X*,*Y*). Two friends wish to come to Logic. Now the first is at the crossing (*X1*, *Y1*), the second is at the crossing (*X2*, *Y2*). Because of plentiful snowing they wish to minimize the trampled path (the sum of paths trampled by the first, by the second and by the both going together). Write a program calculating the minimal length of path.

Task 4. At night, a mouse is anywhere within a long ditch of "figure-of-eight" of length 2008 meters. The mouse can run quickly but cannot climb out. Two men with sacks stand at *X1* and *X2* meters. The men's velocity is 1 meter/second. Write a program calculating the minimal time to catch the mouse in any case.

Task 5. A piece of the upper half-plane is cut by broken (not self-touching) line connecting N points: ($X[1]$, $Y[1] = 0$), ($X[2]$, $Y[2] > 0$), …, ($X[N-1]$, $Y[N-1] > 0$), ($X[N]$, $Y[N] = 0$). Given $(2N-2)$ integer numbers … write an algorithm detecting whether this piece can be extracted from the half-plane by means of motion within the plane (1990, to solve by pen-and-paper).

Generation idea: what surfaces can be punched?

During the Olympiad, all organizers and contestants also thought that the only possible motion was a parallel shift and all algorithms investigated possibility of such a motion only. But just after the closing ceremony one of contestants found an example of "upper half of a (narrow) crescent" which can be extracted by rotation!

This task demonstrates both dangers and interest arising while implementing the proposed approach.

4. Proposals and Conclusion

Task 6. Given a graph (with less than 10 vertices) and two sets B (initial positions of wanderers) and E of its vertices, $|E| \geq |B|$. Write a program finding the least number of steps to move all wanderers from B to E (they do not meet each other).

Task 7. Consider a rectangular grid $0 \leq X \leq M$, $0 \leq Y \leq N$.

A) For all $Y = 0, 1, \ldots, N$, points $(0, Y)$ and $(M, N - Y)$ are the same; or

B) For all $Y = 0, 1, \ldots, N$, points $(0, Y)$ and (M, Y) are the same; for all $X = 0, 1, \ldots, M$, points $(X, 0)$ and (X, N) are the same.

Write a program calculating

C) the shortest way along the grid between two given points; or

D) the shortest cycle connecting three given points along the grid.

Task 8. The head H of a train of length N is points at the point $(0, N)$, its tail T is at the point $(0, 0)$. The train can move (forward only) along the rectangular grid not self-touching and cannot pass given points …. Write a program finding one of the shortest ways to be passed by the train (moving forward only) in order to put H to the point (XH, YH) and at the same time to put T to the point (XT, YT).

Main restriction: $|XH - XT| + |YH - YT| \leq N$.

Task 9. Cut a given rectangle with integer sides by two segments parallel to its sides (to three or four rectangles with integer sides) and

A) compose of these rectangles (without overlapping) a polygon (with all angles right) of the least possible perimeter; or

B) shift and overlap them to compose a polygon of the least possible area. (Only parallel shift is permitted).

Demonstrate composing a task on a given theme.

There is Fig. 1. *Regions that hosted finals of the NOI* (15 towns) in the paper Dagiene (2007). *Idea:* "Two friends with bicycles decided to make photos of these towns for the illustrated history of NOIs". Choose the endpoints: "Now (in the morning) they are in Vilnius (the capital; 16th town) and must return here". Further, the array of distances (may be, in hours rather than in kilometers) between some pairs of these 16 points must be given. Also, choose the time necessary for every town (for instance, 3 hours). To make the task more realistic, add: "one can ride or make photos not more 12 hours a day." Thus, we obtain the *Task 10*: "Write a program calculating the minimal number of days for such enterprise under given conditions".

We hope that tasks built in such a way would yield short and elegant formulation (Dagiene, 2007), would be interesting for young people and attractive for prospective sponsors. Also, such tasks give less advantage to experienced participants because they would not be able to use known algorithms immediately.

Analysis of programs written by contestants within conditions of Subsection 2.1 “Evident Solutions” as it was proposed by (Verhoeff, 2006) and is seen from Task 5 would yield interesting and unexpected results.

Thank you for attention!