Olympiads in Informatics, 2021, Vol. 15, 75–82
© 2021 *IOI, Vilnius University*
DOI: 10.15388/ioi.2021.06

Automatic Makers as a Source for Olympiad Tasks

Pavel S. PANKOV, Taalaibek M. IMANALIEV,
Azret A. KENZHALIEV

Institute of Mathematics, Kyrgyzstan
American University in Central Asia, Kyrgyzstan
Korea Advanced Institute of Science and Technology (KAIST).
e-mail: pps5050@mail.ru, imanaliev_t@auca.kg, azret.kenzhaliev@gmail.com

Abstract. Automatic maker is meant as a device to produce (or compose) things by itself, by a model, customization or a program. Nowadays the most advanced automatic makers in mass use are 3D-printers.The purpose of the paper is to present methods of generating various Olympiad tasks by using evident images of virtual automatic makers. As a perspective, production (performing) of processes is also considered (as automatic 4D-makers). Besides of the main operation: putting a pixel (voxel), the following primitives with virtual things can be involved: cutting; gluing; putting a building block; erasing a building block; copying a fragment, (for 4D-) shifting a building block. Tasks are generated naturally: to make a given thing (perform a given simple process) optimally in any sense (with respect to time; to number of primitives; to number of building blocks etc.). Such tasks are well-understood, have short formulations and are difficult to be solved even with initial data of small volume; a "brute force" method is either inapplicable or gives too overestimate of complexity.

Keywords: Olympiad, informatics, automatic maker, optimization tasks, 3D-printer.

Introduction

Automatic maker is meant as a device to produce (or compose) things by itself, by a model, customization or a program. Since ancient times, various copying and duplicating equipment may be considered as predecessors of such makers. Since computers appeared, they could produce 2D-things by means of plotters and (2D-)printers. Nowadays the most advanced in mass use are 3D-printers. Industrial robots can produce almost all of these.

Remark. It is interesting that "3D-printer" is called so only by analogy. Its principle is quite different.

The purpose of the paper is to present methods of generating various Olympiad tasks by using evident images of virtual automatic makers.

As a perspective and as a source for tasks, we consider production (performing) of processes also. Such devices may be considered as 3D(space)+1D(time)=4D-makers, we also consider 1D(line)+1D(time)-makers; 2D(plane)+1D(time)-makers.

Considered primitives: besides of the main operation: putting a pixel (voxel), cutting ("by a lazer beam"), putting a building block; erasing a building block; copying a fragment, (for 4D-) shifting etc.

Simplified schemes of such makers generate Olympiad tasks naturally: to make a given thing (perform a given simple process) optimally in any sense (with respect to time; to number of primitives; to number of building blocks etc.).

For solving an Olympiad task by the contestant, in addition to the common limits in the CPU time (traditionally 1 second) and in memory (traditionally 256 megabyte) there exists an actual limit (*) on average time to write and debug a program even if the contestant has necessary skills and vision of the algorithm (about 1–1.5 hours). Sometimes such limit is achieved by means of a long-winded and complicated text of the task, with many "permissions" and "bans" (**). In proposed tasks this limit is achieved naturally, because of their geometrical content.

We hope that proposed tasks on automatic makers can be done well-understood, to have "short and elegant formulations" (Dagienė *et al.*, 2007) and are difficult to be solved even with initial data of small volume; a "brute force" method is either inapplicable or gives too overestimate of complexity.

On the other hand, such tasks are "natural" in sense of (Pankov, 2008): a human can "see" the answer for corresponding task image with initial data of small volume without calculations.

Methods to generate and examples of such tasks are considered in the paper.

We will write "input" in examples in abbreviated form: lines are separated with the sign \ .

Three examples:

Task 1. Spear. (Pankov *et al.*, 2018)

It is known that Japan appeared as Drops into Ocean from Spear (see the picture). Let us formalize and optimize this process to create an archipelago. Divide Ocean into equal squares. Let each Drop be a square of Land.

Task: given a $K \times K$-binary matrix ('0's mean Ocean, '1's do Land) and the set of possible steps of Spear. Initially Spear is over the North-East corner of the matrix.

How many steps of Spear are necessary to create all Lands (to pass all '1's)?

The simplest sufficient set of possible steps of Spear is {S, W, E}.

Example for K=11: Input: 11. \

00000000000000
00000000000100
00000000001110
00000000000000
00000000011000
00000000110000
00000011110000

```
00011111000000
00000000000000
00010110000000
00010000000000
```

Output: 32.

[Possible beginnings of the optimal ways: WWSES… or SWWSE…]

Task 2. 3D-printer (Kyrgyzstan Regional Olympiads, February 2020)

The *X*- and *Y*-axes are horizontal, the *Z*-axis is down. Sides of all cubes are equal 1 and parallel to the axes, coordinates of their centers are integer numbers. The lower semi-space "$Z \le 1$" is filled with cubes. The initial coordinates of Cube-printer are (0, 0, 0). Given one, two or three cubes with coordinates in $[-N, N] \times [-N, N] \times [1, N]$. At each step Cube-printer moves by one along one of axes and erases a met cube. How many steps of Cube-printer are necessary to erase the given cube(s) (with cube(s) over them only)?

Example for two cubes: Input: 2 \ 8 7 1 \ 9 7 5. Output: 21.

It is seen that the complexity of this task does not depend on $N > 10$.

Task 3. Robot (Kyrgyzstan Regional Olympiads, March 2021, improved). Cubic Robot of volume 1 moves in continuous media (for instance, the warm iron cube moves in dense snow). Its edges are parallel to *X-, Y-, Z*-axes. A "shift" of Robot is its motion along or against one of the axes by an integer number J ($|J| \leq 10^{12}$). Given a sequence of 2..6 shifts, find the volume of Robot's "trace" (empty space in media made by Robot's motions including Robot itself).

Example for three shifts. Input: 3 \ Z 5 \ X –6 \ X 4. Output: 12.

We present examples of tasks but we do not propose any general algorithms to solve such tasks. On the contrary, we suppose that such algorithms do not exist and preferences of such tasks are that each task demands its own algorithm, with little discoveries, to smooth out the effect of training contestants.

1. Media and Operations

We will use the following spatial primitives: segments of length 1 on 1D; squares in 2D (pixels) and cubes in 3D (voxels), their sides are equal 1 and are parallel to the axes; coordinates of their centers are integer numbers. We will name all them "spexels".

We will consider 3D-printer in general sense: arranging arbitrary set of spexels in space. It is convenient to do it in zero gravity otherwise certain supports are necessary. We will mention "transparent spexels" for this purpose. (For generation of tasks, "transparent pixels" will be used too).

Temporal step is equal 1.

To present processes, discretizing the idea of Eulerian coordinates in continuum mechanics we propose time-space primitives: space primitives existing during one temporal step. Their indices are space coordinates of their centers and temporal one (natural number). We will name them "timexels".

An automatic time-maker works as follows. It arranges spexels and waits one temporal step. Then it (very quickly, by the idea of cinema) re-arranges (in any way) next spexels for the next step.

There can be homogeneous and non-homogeneous spexels and timexels, for instance, of different colors.

Remark. By our opinion, the terms "spexels" and "timexels" can be applied to other objects which are in use, for instance, regular triangles and regular hexagons, triangles in splines.

We will use the following operations:

Usual *putting* of spexel or timexel to the given spot (with given coordinates).

Cutting along sides (endpoints) of spexels ("by a laser beam"). There are two kinds of cutting in 2D and 3D cases: cutting along all straight line or plane; cutting along one common side of two spexels (1-*cutting*).

By custom, adjacent spexels are suggested to be glued. Hence, the operation of *gluing* can be applied either to previously cut spexels or to a "new" one.

Erasing a spexel.

Moving a spexel. We will consider moving along the coordinate axes.

Moving some glued spexels.

Repainting spexels.

2. Tasks on Spatial Arrangements

Task 1 – Solution. If permitted steps are within “a cone” ({S, W, E} or {S, SW, SE, W, E}) then dynamic programming can be applied, the complexity is $O(K^3)$.

Otherwise, for instance for {S, N, W, E}, complexity increases to $O(L^2 * 2^L)$, where L is the number of ‘1’s in the matrix. A dynamic programming on bitmasks where we enumerate all ‘1’s and compute the states *dp[mask][latest_visited_bit]* can be applied – *mask* denotes which ‘1’s were visited by the spear and *latest_visited_bit* denotes which of those ‘1’s marked by the mask was visited last. We iterate over possible next unvisited ‘1’s (“off” bits in mask) and update transition states.

`https://ideone.com/cMkaMZ`

Task 2 – Solution. The limit (*) is vast: there are many different cases. To pass the following tests a “discovery” of zigzag path is to be made:

Test 2.1. Input: N=2; C1=(0,0,20); C2=(0,1,10). Output: 30.

Test 2.2. Input: N=3; C1=(0,0,30); C2=(0,1,20); C3=(2,0,10). Output 60.

Task 3 – Solution. If the limit for length of shifts $|J|$ is not “large” ($|J| \leq 10^6$) then creation of array of all cubes in Robot’s trace solves the task

`https://ideone.com/app8e4`

For “large” $|J|$ we offer the following algorithm.

Let the initial position of Robot be (0 0 0).

Define a “segment” by six integer numbers. For instance, the first shift in Example (Z 5) generates the segment X = 0, Y = 0, Z = 0..5 or (0 0 0 0 0 5). The second shift (X -6) does the segment (-1 -6 0 0 5 5).

Write a procedure Cross to elaborate two segments into one, two or three segments of the same union but with all intersections being empty.

Applying this procedure with the new segment and all preceding segments after each shift we obtain a presentation of Robot’s trace as union of non-intersecting segments. The sum of their volumes yields the answer.

Task 4. Given a natural number $N \geq 2$. Let you to make N separate spexels. Mark N spexels in space by your will and apply the cutting operations (or 1-cutting ones). Find the minimal number of operations.

Example for 3*D*. Input: 3. Output: 8 cuttings (or 16 1-cuttings).

Solution. For cuttings, the answer is evident:

2D: $\min\{U + V + 2: U*V \geq N\}$; *3D:* $\min\{U + V + W + 3: U*V*W \geq N\}$.

For 1-cuttings the only way to solve is to examine all “compact” and “almost round” arrangements. It is difficult to prove that all arrangements that can be optimal are examined and to estimate complexity of solution. The jury is to prove it as a theorem (Pankov *et al.*, 2018) but the contestant can guess.

“Naturalness” means also the following. It is not necessary to write a program for generation of tests for a task. A human can compose sufficiently complex tests where the answer is “seen” but it is difficult to find it by means of any program.

Task 5. Given a natural number $N \geq 10$ and an arrangement of N spexels (in empty space), some of them are marked. How many, at least, other spexels are to be erased (or how many, at least, 1-cuttings are to be made) to pull out all marked spexels?

Example for 3D. Input: 175 voxels forming a 5×5×7-parallelepiped, one central voxel is marked. Output: 2 voxels are to be erased (or 13 cuttings are to be made).

For 2D and $N < 200$ the answer in the test is seen by a human. For 3D and $N < 50$ observing of 3D-presentation of the test with semi-transparent cubes is seen. But a program would be sufficiently complex and to write it during bounded time is difficult (*).

Task 6. Given a natural number $N \geq 10$ and an arrangement of N spexels (adjacent spexels are supposed to be glued). Fix this arrangement. How many, at least, transparent spexels are to be added and glued to connect all given spexels?

Example for 2D (two pixels, the number N is less than 10).

Input: 2 \ 10 8 \ 11 2021. Output: 2013.

Solution would include the following items:

1) Unite given spexels into clusters.
2) Write a procedure finding the Manhattan distance between two clusters.

4. Tasks that Presenting Processes

The timexel is denoted as: *<spatial coordinate(s)>, <time>,* optionally *<color>.*

For example, the time step for the spectator is 1/60 second; one primary operation by the maker takes 10 microsecond = *tms*.

Task 7. Given $2N$ timexels: N timexels at time=1 and N timexels at time=2. All spatial coordinates are in 1..100, N in 2..10. A) all timexels are homogeneous or B) timexels at each time are colored equally; timexels cannot pass one through other.

How many operations (shifts by 1) are necessary to make this pass?

Examples for $N = 2$:

Example 7.1. A) 2D: Input: 2 \ 23 25 1 \ 28 22 1\ 25 23 2 \ 26 25 2. Output: 7.

Example 7.2. B) 3D:

Input: 2\ 8 23 25 1 blue \ 8 28 22 1 red \ 8 25 23 2 red \ 8 26 25 2 blue. Output: 9.

An example of task with (**).

Task 8. Given $N1$ colored timexels at time=1 and $N2$ colored timexels at time=2; timexels cannot pass one through other. A shift by 1 takes 1 *tms;* a repainting takes 2 *tms;* an erasing takes 3 *tms*; a creation of a colored timexel in any position takes 10 *tms.* ...

How many *tms* are necessary to make this transformation?

(If $N1 > N2$ then ($N1 - N2$) erasings are necessary; If $N1 < N2$ then ($N2 - N1$) creations are necessary; but sometimes one erasing and one creation are more profitable than many shifts; a similar note is right for repaintings).

A task on a sequence of events:

Task 9. Given natural numbers *M, N* and *K* and by *N* shining timexels for time = 1.. time = *K; N < M < K*N*. Timexels can turn on and turn off at each time. Find the minimum total number of shifts of *M* timexels to provide these events from their optimum initial arrangement.

Example for 1D:

Input: M = 3, N = 2, K = 3 \ 20 1 on \ 30 1 on \ 15 2 on \ 50 2 on \ 20 3 on \ 29 3 on Output: 11.

(the optimum initial arrangement is: M = 3 \ 20 1 on \ 30 1 on \ 50 1 off).

These tasks did not take into account the need to fix timexels in 3D.

Task 10. 2D (3D). Given *N* timexels at time = 1 and *N* timexels at time = 2. All spatial coordinates are in 1..100, *N* in 2..10. At time=1 all given timexels lean on the line *Y* = 0 (on the plane *Z* = 0) by means of pillars made of transparent timexels.

How many transparent timexels are necessary to be created to make this pass? (transparent timexels cannot move; some transparent timexels can be erased).

Example. 2D: Input: N = 3\ 4 2 1 \ 8 1 1 \ 4 7 1\ 5 2 2 \ 6 9 2 \ 9 4 2. Output: 8.

5. Conclusion

There are various publications on creation of new types of Olympiad tasks (Kemkes *et al.*, 2007), (Burton *et al.*, 2008), (Diks *et al.*, 2008), ours (Pankov *et al.*, 2018), (Pankov *et al.*, 2020). We hope that the proposed approach is new in general. We also hope that such tasks would enlarge the scope of tasks involved into Olympiads in Informatics and give ideas for young people to implement in hardware.

References

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

Pankov, P.S. (2008). Naturalness in Tasks for Olympiads in Informatics. *Olympiads in Informatics: Tasks and Training,* **2**, 16–23.

Pankov, P.S., Kenzhaliev, A.A. (2018) Combinatorial property of sets of boxes in Euclidean spaces and theorems in Olympiad tasks. *Olympiads in Informatics,* 12, 111–117.

Pankov, P.S., Kenzhaliev, A.A. (2020). Pattern Recognition and Related Topics of Olympiad tasks. *Olympiads in Informatics,* 14, 143–150.

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

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 physics-mathematics 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, 2018–2020. Graduated from the Kyrgyz State University in 1969, is a head of laboratory of Institute of mathematics of KR NAS.

T.M. Imanaliev (1965), doctor of physics-mathematics sciences, professor of Applied Mathematics and Informatics Department at American University of Central Asia.

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