審判長講評

審判長　増原 英彦

本年度は昨年同様に，5 時間で 10 問を出題した．正解数ごとのチーム数および，問題ごとの提出数と正解チーム数は，表 1 および表 2 に示す通りである．今年は，1 チームを除いて1問以上正解している．全問を正解したのは 1 チームだけであった．このチームは 10 問目を正解した時点でコンテスト時間を約 1 時間残していた．

正解数ごとのチーム数の分布を見ると，0 問正解から 10 問正解まで比較的均等に分布している．また，問題ごとの正解チーム数も 1 チームから 33 チームまでほぼ連続している．これは，今回の問題群の難易度が低いものから高いものまでバランス良く作ることができていたためであろうと思われる．

表 1: 正解数ごとのチーム数

正解数	10	9	8	7	6	5	4	3	2	1	0
チーム数	1	1	4	3	5	5	1	4	6	3	1

表 2: 問題ごとの結果(全 34 チーム)

問	正解チーム数	提出数	問題名
A	33	46	Ginkgo Numbers
B	20	24	Stylish
C	28	31	One-Dimensional Cellular Automaton
D	24	51	Find the Outlier
E	8	22	Sliding Block Puzzle
F	19	52	Never Wait for Weights
G	16	32	Let There Be Light
H	2	17	Company Organization
I	7	38	Beautiful Spacing
J	1	7	Cubic Colonies

問題 A. Ginkgo Numbers

ガウス整数(複素整数)の素数判定を行う問題である．対象となる値の存在範囲が狭いため，約数が存在する可能性のある範囲のガウス整数全てについて検査をすればよい．例年の問題 A と比べ，数学的性質の理解が必要かつ長めの問題記述となったが，最終的には全 34 チーム中 33 チームが正解していた．

問題 B. Stylish

例として与えられたプログラムのインデント幅に関する「流儀」を解析し，もう 1 つ与えられたプログラムのインデント量を計算するという問題である．「流儀」は 3 種類の括弧に対するインデント幅のみで決まるうえに幅の上限が小さいので，全てのインデント幅の組み合わせについて前者のプログラムとの整合性を調べればよい．ただし，一部の括弧についてしか情報が得られない場合，後者のプログラムで使われている括弧の種類によってはインデント幅を決定できない行が発生する点に注意が必要である．正解チーム数は20，提出解答数も24とあまり多くなかった．多少の文字列処理が必要なことと，設定がやや複雑な点が敬遠されたのかも知れない．

問題 C. One-Dimensional Cellular Automaton

時間発展する 1 次元セルオートマトンの時刻 T における状態を計算するだけであるが，T の上限が 10 億であることに注意が必要な問題である．状態をとベクトルで，その更新を行列とベクトルの乗算で表わし，行列の T 乗を対数時間で求める解法を使って解くことができる．正解チームは 28 と 2 番目に多く解かれた問題であった．

問題 D. Find the Outlier

未知の一変数 N 次多項式の変数値と式の値の組のうち，誤っているものを発見する問題である．入出力は $N+3$ 組与えられるので，そのうちの 1 組が誤りであることは，残りの $N+2$ 組のうちの $N+1$ 組から推定された多項式に最後の 1 組が整合していることで確められる．$N+1$ 組の入出力から多項式を推定するには，係数についての連立一次方程式を解くか，ラグランジュ多項式を作ればよい．24 チームが正解している．

問題 E. Sliding Block Puzzle

15 パズルや「箱入娘」と同様にタイルを空き地にスライドさせて目的の配置を作るパズルの問題である．障害物も置かれた盤上で 2×2 のブロックを目的地まで移動させるので，迷路探索といった性格も持っている．素朴に考えると，大小全てのブロックの位置の組み合わせが 1 盤面の状態となるが，最大の盤面は 50×50 の大きさであるためにそのまま幅優先探索を行うには適さない．また A*探索等のためのヒューリスティクス関数も自明ではない．実際には 2×2 のブロックの移動だけに注目すればよいので，2×2 ブロックの位置と移動方向の組を 1 状態として表すことで解くことができる．計算量の見積りが難しく，プログラミング量も要求される問題だったためか，正解チーム数は 8 と少なかった．

問題 F. Never Wait for Weights

2 物体の重量差情報が大量に与えられたとき，指定された 2 物体間の重量差を計算する問題である．与えられる重量差情報の個数 M が最大で 10 万となるので，効率的なデータ構造を考える点が主眼となる．例えば，単純な木構造を構築していくだけでは，データが与えられる順によっては $O(M)$ の深さの木ができることがあるため適さない．Union-Find アルゴリズムの高速化手法としても知られているような，木の深さのバランスを取ったり，検索のたびに各ノードから根を直接指すような工夫が必要となる．大量のデータが与えられることを想定した解法を考えるのが難しかったためか，最も提出数が多かった(それだけ不正解が多かった)問題となった．

問題 G. Let There Be Light

複数の光源と，それらをさえぎっている多数の球状の風船(不思議なことに重なりあうことができる)の位置が与えられたとき，定められた個数以下の風船を割って，ある点の明るさを最大化する問題である．基本的には組み合わせ最適化問題であるが，光源の個数が最大で 15 と少ないことから，「見えるようにした

い」光源の組み合わせ全てについて調べればよい. 光源, 目的の点, 風船の関係は幾何的性質によって処理しなければいけなかったためか, 正解チーム数は 16 にとどまった.

問題 H. Company Organization

多数の集合に関して, 包含関係や 2 集合間の共通部分集合が空かどうかといった制約が与えられ, それらの間に矛盾がないかを調べる問題である. ここでは最大 1 万個の制約が与えられ, その中で最初に矛盾が生じる制約を答える必要がある. 解法の一つは, 制約の集合に 1 つでも矛盾があるかどうかを調べる手続きを作り, それを用いて最初に矛盾する制約を 2 分探索によって見つけるものである. 制約の種類は 5 種類だけであるが, それらの組み合わせから正しく矛盾を発見するためには, 制約間の関係の慎重な考察と正確なプログラミングが必要となる. そのためか, 提出数・正解チーム数ともに 2 番目に少ない問題となった.

問題 I. Beautiful Spacing

いわゆる「両端揃え」で英文を整形する際に, 単語間の最大空白量が最も小さくなる場合を求める問題である. 単純には文の途中の単語以降だけからなる英文についての解が, 先頭行に置いた単語列と, それ以降の単語からなる英文についての解で決定されるので, 普通の設であれば動的計画法で解ける問題である. ただし本問では, 1 行の長さが非常に長い場合があるため, ある単語から 1 行に置ける最大・最小単語数といった性質を効率的に求める等の工夫も必要となる. 7 チームが正解する一方で, 不正解チーム(解答を 1 回以上提出したが正解できなかったチーム)数も全問題中で最も多い 7 であった.

問題 J. Cubic Colonies

小立方体を複数結合して作られた物体において, 2 点間を結ぶ物体表面上の最短距離を求める問題である. 一つの方法は, 立体の展開図を作成し, 展開図上での最短距離を求めるものであろう. この場合, 展開図の面が重なる等の点に注意が必要となる. 別の解法としては, 小立方体面の各辺を整数比で分割した点を求めておき, それらの点をノード, 点間を(小立方体面)上で結ぶ線分をエッジとしたグラフを構成するものがある. どちらの方法でも, 相当量のプログラミングが必要となることが予想されたが, 1 チームが後者の方法で正解している.