審判長講評

審判長　近山 隆

本年も昨年、一昨年と同様、300 分の競技時間に対して全 11 問を出題した。チームごと、問題ごとの正答数等を以下に示す。

表 1　正解数ごとのチーム数と正答問題のパターン

正解数	チーム数	累計	正答パターン
11	1	1	すべて
10	1	2	K以外
9	1	3	JK以外
8	1	4	HJK以外
7	7	11	A-G: 3; A-EGI: 3; A-EGJ: 1
6	11	22	A-EG: 10; A-DFG: 1
5	6	28	A-E: 3; A-DG: 3
4	8	36	A-D: 7; A-CG: 1
3	9	45	A-C: 9
2	0	45	
1	0	45	
0	0	45	

表 2　問題ごとの正答チーム数・誤答チーム数・提出数

問題	A	B	C	D	E	F	G	H	I	J	K
正答数	45	45	45	35	24	8	26	3	6	4	1
失敗数	0	0	0	5	3	5	2	2	7	3	2
提出数	48	55	81	109	52	51	77	13	55	9	3

失敗数は、解答を提出したが正解にまで到らなかったチームの数である。
提出数は、正誤の如何に限らず、提出されたプログラムの総数である。

作題にあたって目標としたのは、例年と同じく以下の各項目である。

- すべてのチームが少なくとも 1 題は正答する
- すべての問題を少なくとも 1 チームは正答する
- あまり早くに全問正答し終えるチームはない
- 問題セットは解答に多様な領域の知識を必要とするものとする

最初のふたつと三番目を同時に実現するのが難しいのだが、今回はかなりうまく行った。最初のふたつは実現しており、三番目については全問正答したチームの最後の正答は開始後 294 分、すなわち競技終了のわずか 6 分前だった。また、各問題に対する最初の正答は、3 問に対して最初に正答したチームが 1 チームあっただけで、他はすべて別々のチームが最初の正答チームとなった。これは最後の項目についても実現できたことを意味するのかもしれない。

全チームが A, B, C の 3 問に正答した。A と B に全チームが正答するのは予期できたが、C まで全チームが正答したことは、国内予選通過チームのレベルの底上げがうかがわれる。このレベルのチームが集まるのなら、比較的易しい問題の数は減らしてもよいかもしれない。

以下、各問題について解法を中心に解説する。

問題 A. Rearranging a Sequence

Move-to-Front と呼ばれる、種々のデータ圧縮の前処理としてよく用いられる操作を多数回繰り返した結果を問うものである。実際にこの操作を繰り返すのがわかりやすいが、配列によるデータ列の表現では計算量は $O(MN)$であり（M は操作回数、N はデータ要素数）、手間がかかりすぎる。列を双方向リストで表現し、各値に対応するセルへのポインタを別途配列に保持するなどすれば、手間は $O(M+N)$ で済む。しかし、最終的な並び順だけが問われているのだから、実際に操作を行う必要はなく、各データ要素に対する操作の順序を記録して、後の操作対象ほど前に来るように安定ソートする $O(M+N \log N)$ の方法や、操作指示列を逆転して元の要素列の前につけて、前から順に重複を取り除く $O(M+N)$ の方法などもある。

どのチームもこの程度の問題には苦労することがなかったようで、ほとんどのチームが最初の提出で正答となり、開始から 30 分程度で全チームが正答し終えた。

問題 B. Quality of Check Digits

無味乾燥な数字列の誤り検出のために付加した検査用数字（チェックディジット）が、典型的な誤りを検出できる能力を持つかを調べる問題。日本で使われ始めたマイナンバーのチェックディジットには十分な誤り検出力がないことが問題視されている。それはともかく、単純に規則通り計算してみれば解答として十分である。入力した演算表に従って、すべての基本番号（4 桁なので 1 万種類）のひとつひとつに対し、規則通りにチェックディジットを生成して得られる 5 桁の数字列について、そのひとつを間違えた場合（45 通り）と、隣同士の数字を入れ替えた場合（最大 4 通り）の計 49 通りについて、エラーが検出できるか否かを試せばよい。この方法で計算量も十分小さい。

題意さえきちんと読み取れれば、解くのは簡単だろう。実際、この問題もすべてのチームが正答できた。正答まで時間がかかったチームもあったが、C 以降の問題に先に正答していてプログラミングの実力は十分ありそうなので、題意を読み取るのに若干苦労したのかもしれない。

問題 C. Distribution Center

工場の生産ラインから倉庫へのコンベヤーレーンという設定だが、要はあみだくじ状の経路をたどる際に、分岐をしてもしなくてもよいとしたとき、行きつける先を数える問題である。レーン数 n やロボットアームの数 m はかなり大きくなるので、コンベヤーレーンにつながる生産ライン番号を個別に記録し、これをロボットアームごとに更新する計算量 $O(mn)$ の方法では、時間がかかりすぎる。製品の移動は隣同士のレーンでしかできないことから、特定の行先に移送できる生産ラインは必ず連続番号になる、ということに気づけば、各レーンに移送できる生産ライン番号を 2 数間の区間として表現できることがわかるだろう。この表現をとれば、ロボットアームごとに両側のレーンの区間をつなげたものに更新するだけなので、計算量は $O(m+n)$ で済む。

この問題も全チームが正答できたが、A, B に比べて正答を得るまでに何度も誤答を提出したチームが少なくなかった。

問題 D. Hidden Anagrams

ふたつの文字列について、互いにアナグラムになっている部分文字列を見つける問題である。4000 文字の文字列の部分文字列は、それだけでも$1 + 2 + 3 + \cdots + 4000$で、約 8 百万ある。ふたつの 4000 文字の文字列の同長の部分文字列の組合わせすべてとなると、$1 + 2^2 + 3^2 + \cdots + 4000^2$で約 200 億組にもなり、これを全部試すのでは時間切れ必至である。

ヒントになるのが、文字列探索の Rabin-Karp アルゴリズムである。部分文字列のハッシュ値を計算して一致の可能性を絞る方法で、部分文字列の開始位置を一文字ずらすたびに必要な計算量が一定で済むハッシュ関数を用い、探索の手間をおおむね線形に抑えるものだ。これは通常の文字列探索用なので文字の順序もハッシュ値に反映させるが、アナグラムに順序は無関係である。文字のマルチセットとして同じなら値が同じになるハッシュ関数で、開始位置を一文字ずらす際の計算の手間が一定のものを使えば良い。こうすれば文字列長 n の文字列の長さ m の部分文字列$n - m + 1$個に対するハッシュ関数の計算が $O(n)$, このハッシュ値をまたハッシュ表に登録すれば一致の検索もおおむね $O(n)$ でできる。これを

$m=n$ から始めて次第に部分文字列を短くして行けばよいので、全体の計算量も $O(n^2)$ で済む。ハッシュ関数としては文字コードの和などでは値の範囲が狭くて衝突が多くなるので、個々の文字に文字コード以外の値を適宜振り、その和を用いるなどが考えられる。乱数を使って値を振るのが Zobrist hashing と呼ばれる手法である。よほど運が悪くない限り衝突しにくいハッシュ関数を得られることが知られており、ボードゲームのコンピュータプレイヤで盤面状態のハッシュによく用いられる。この問題の場合にも適しているだろう。

この問題には 35 チームが正答できたが、かなり苦労したチームも多かったようだ。正答にならなかった解答中では時間制限にひっかかったものが多かった。

問題 E. Infallibly Crack Perplexing Cryptarithm

演算子まで隠れている覆面算の問題である。あやしげな問題名の頭文字は ICPC になっている。文字種は 8 と大きくないので、覆面文字への割当ての全部の組み合わせでも最大 8 の階乗、4 万通り程度しかない。この程度なら実際に作ってみて、文法に合うか、等式の両辺が等しくなるかを全部試しても時間制限にかかることはない。文脈自由文法に対して正しく構文解析できさえすれば、文法も小さいのでさほど難しくなかろう。最大 31 文字しかないので、構文解析の効率は問題にならない。計算量を見通して、高度なアルゴリズムは不要とする判断力も、高度なアルゴリズムの知識と同様に大切である。

この問題は 24 チームが正答した。誤解答のほとんどは出力結果が正しくないものだった。

問題 F. Three Kingdoms of Bourdelot

それぞれ先祖－子孫関係を羅列した文書群が与えられたとき、「特定のふたりが先祖－子孫関係にある」という仮説が文書群の記述と矛盾しないか、を判定する問題である。ちょっと面倒なのは、先祖－子孫関係に「ある」二者を列挙した「正文書」と、「ない」二者を列挙した「負文書」があり、各々の文書がそのどちらかわかっていない、ということである。各文書についてどのように正負を仮定しても矛盾が生じるようなら、仮説が成り立つ余地はない。

解法はさほど複雑ではない。与えられた仮説から始めて、先祖－子孫関係にあることになる二人の組を蓄えていく。ある文書中の関係が既知のものと一致するなら、それは正文書と考えるしかなく、その文書中にある関係はすべて成立することになる。このとき、「子孫の子孫は子孫」という推移律からわかる関係もすべて既知ということになる。こうすることでふたりの人物が互いの子孫ということになってしまうようなら、仮説と文書群は矛盾することがわかる。これを新たな関係が現れなくなるまで繰り返す。この過程で正文書と決められなかった文書には、既知の関係と一致するふたりの組についての言及はないはずである。したがって、そうした文書はすべて負文書であるとして矛盾は生じない。

この問題は、計算自体はさほど難しくないのだが、問題内容を理解して解法を整理するのが簡単ではなかったのだろう。正答に至らなかったチームも含めてこの問題に手を付けたのは 13 チーム、正答できたのは 8 チームだけと、審判団の想定よりもかなり少なかった。論理的に問題を整理して理解する能力も、プログラミング力の重要な一要素である。

問題 G. Placing Medals on a Binary Tree

完全二分木の指定深さのノードをひとつずつマーキングしていく、という問題。制約条件に合致するマーキングが可能か否かを、指定深さ d に対応する数値 $1/2^d$ の和が 1 を超えるか否かで判定できる、ということに気づければ、後はこの和の計算に必要な計算量をどうやって抑えるかの問題になる。

2 進小数に対する多倍長計算なので、繰り上がり処理の計算量が問題になる。足しこんでいくだけなら普通に繰り上がりを処理しても、繰り上がり処理が 1 の桁を減らして次の繰り上がりを起きにくくするので、1 回あたりの償却計算量は $O(1)$、全体としてマーキングの回数 n に対して $O(n)$ で済む。しかしこの問題では制約を満たすマーキングができない場合、マーキングしないで次に進む。だからせっかく処理した繰り上がりをキャンセルせねばならない。これでは計算量が $O(n^2)$ になり、制限時間内に解けないだろう。

繰り上がり処理を高速化する方法はいくつか考えられる。2 進小数の中の最初の 0 の桁を覚えておけ

ば、実際に計算しなくても、その桁あるいはそれより上の桁に 1 を足すと小数点の上まで繰り上がりが生じることがわかる。ちょうど 1 になる場合はマーキングできるのだが、最初の 0 の桁より下の 1 の桁の個数も覚えておけばよい。マーキングできる場合には繰り上がり処理や最初の 0 の桁の更新が必要になるが、これはキャンセルの必要がないので償却計算量としては $O(1)$ であり、全体として $O(n)$ で済む。

2 進小数の表現を工夫することで、繰り上がり処理を高速化することも考えられる。連続する 1 の桁を始まりと終わりの区間として表現すれば、ひとつの 1 の連続に対する繰り上がり処理は計算量 $O(1)$で済む。この区間を木構造の表に登録すれば更新 1 回あたり $O(\log n)$ の手間で、全体は $O(n \log n)$ である。これぐらいの計算量なら制限時間を超えないだろう。

この問題は 26 チームが正答した。それなりに難しい問題と考えられるが、二分木や 2 進表現には習熟したチームが多く、理解しやすかったのかも知れない。誤答の提出も多かったが、制限時間超過よりも出力の誤りが多かった。問題に取り組んだが正答に至らなかったのは 2 チームだけだが、何度も失敗した上でやっと正答できたチームも多く、かなり時間を取られてしまったであろう。

問題 H. Animal Companion in Maze

有向グラフで、閉路の有無を調べ、閉路がない場合は、出次数 0 のノードまでの最遠点の距離を求める問題である。ただし、入ってきたドアからすぐ出ることはできない。つまり、直前のノードに戻るエッジは辿れない、という点がちょっと面倒になっている。

この制約はノードを分割することで取り除くことが考えられる。つまり、双方向のドアにあたるエッジについては、当該エッジから入ってきた場合を表すノードと、他のエッジから入ってきた場合のノードとを別のものとして扱い、前者には同じドアから戻るエッジをなくす。このようなグラフに対して解いても答えは変わらない。しかしこれでは分割してできるノードの持つエッジの総数が大きいとき、総エッジ数が元のグラフの大きさの二乗オーダになってしまう。

そこでまずグラフを強連結成分に分解する。行ってすぐ戻る、というのを除いては閉路がないとしたら、個々の強連結成分は双方向のエッジだけからなる木になっているはずだ。木ならばエッジ数はノード数よりひとつ小さい。強連結成分が木でなければ、双方向エッジをすぐ戻るもの以外の閉路がある。さて、強連結成分をひとまとまりのノードと考えると、強連結成分同士は単方向エッジで結ばれていて、グラフ全体は有向非巡回グラフ (DAG) になる。普通の DAG の最大経路長を知るには、エッジの方向に従ってノードをトポロジカルソートし、極大ノードから順に最大経路長を決めていけばよい。しかし、この問題の場合は強連結成分の中を通る経路長が入口・出口のノードによって異なる。極大ノードから強連結成分の各入口までの最大経路長がわかったとき、各出口までの最大経路長を知りたいが、個別に調べるのでは強連結成分のノード数の 2 乗オーダの手間になってしまう。強連結成分のノードひとつを選び、これを根とする木を考える。この木を深さ優先で走査すれば、各ノードに子孫ノード中の入口経由で達する最大経路長を知ることができる。もう一度走査することで、親ノード経由のものも含めた最大経路長を知ることができる。この方法なら強連結成分サイズに比例する計算量で済む。

この問題の正答は 3 チームしかなかった。問題記述がやや複雑で、理解に手間取ったために避けたチームも多かったのかもしれない。誤答の提出も多かったが、正答したチームはあまり苦労しなかったようだ。

問題 I. Skinny Polygon

最小バウンディングボックス $(0,0):(a,b)$ を持ち、整数座標頂点を持つ面積 25,000 以内の三角形または四角形を求めよ、という問題。a,bは10^9にまでなるので、面積 25,000 以内にするには、問題名に言うようにかなりスリムでなくてはならない。一見計算幾何の探索問題のように見えるが、実は数論などの知識を用いればあまり面倒な計算はせず解ける問題になっている。

a,bの最大公約数をgとする。Bézout の等式から$|ad-bc|=g$となるc,dが存在することがわかる。$(0,0),(a,b),(c,d)$ の三頂点を持つ三角形の面積は$|ad-bc|/2=g/2$ であり、このようなc,dは拡張ユークリッド互除法で見つけることができる。一方、$(0,0),(a-1,b),(a/g,b/g),(a,b-1)$ の四頂点からなる四角形の面積は、$(a+b)/2g$になる。これらふたつの図形の面積の相乗平均をとると$\sqrt{a+b}/2$となり、$a\le 10^9, b\le 10^9$なのだから、この値は$\sqrt{(2\times 10^9)}/2=22360.679\cdots$ を上回らない。相乗平均が 25,000 を超えないのだから、どちらか一方は 25,000 以内であるはずで、それを解答とすれば良い。

この問題には 6 チームが正答したが、正答したチームのほとんどは最初の提出で正答、他のチームも

3 回以内の誤答の後に正答に至っている。挑戦して失敗した 7 チームの誤答には時間制限超過も多かったが、効率的でない探索を用いた結果だろう。

問題 J. Cover the Polygon with Your Disk

凸多角形と円盤の共通部分の面積を最大化せよ、という問題である。片方が円盤なので、共通部分の面積は回転しても変化がなく、円盤の位置だけに依存する。

円盤の位置を決めたときに共通部分面積を求めるには、まず凸多角形と円盤の周が交わる点と、円盤に隠れる多角形の頂点を数え上げる。共通部分は円盤の中心とこれらの点がなす扇形や三角形を併せたものである。円盤の中心が多角形の外にある場合には負の面積を考えることになるが、特別扱いは不必要である。

共通部分の面積を最大化するには、二次元空間での探索が必要になる。これにはさまざまなよく知られた手法がある。最急降下法などの勾配法、Nelder-Mead 法、準 Newton 法、CMA-ES（共分散行列適応進化戦略）など。この問題では凸図形同士の共通部分面積なので、極大点がひとつしかない単峰関数の最大値探索であり、時間の制約も厳しくないので、どの手法でもよいだろう。

この問題に正答したのは 4 チームで、いずれも誤答を経ずに正答に至っている。上記のように考えればそれほど難しい問題でもないのだが、誤答を含めた提出数も少なく、この問題に取り組む前に時間切れになったチームが多かったのだろう。

問題 K. Black and White Boxes

Nim（三山崩し）に似た、箱の山から黒白の箱を取り去っていくゲームが対象である。零和有限確定完全情報ゲームなので先手後手いずれかの必勝だが、ふたりのプレイヤが取り去ることができる箱の色が異なるというルールなので、初期状態によっては一方が先後にかかわらず必勝になることもある。与えられた候補の中から山を選んでできる「公平」な（先後によって必勝となる側が変わる）初期状態の内で、箱の数が最大のものを求めるのが問題である。

初期状態が「公平」か否かは、ゲーム規則に従ってゲーム木探索をすればわかるが、それでは時間がかかりすぎる。公平性を軽く計算できる状態特徴量を決められると良い。箱の数が少ない場合から帰納的に考えていくと、以下のような特徴量が考えられる。まず、山の底が黒なら底から連続する黒箱の数を、底が白なら連続する白箱の数をマイナスにしたものを考える。それより上の箱については黒白に応じて $\pm 1/2, \pm 1/4, \pm 1/8, \pm 1/16, \cdots$ を加えていき、それらの和を山の特徴量とする。こうした山の特徴量をすべての山について和を取ると、その正負で黒白どちらを取り去る側の必勝かが決まる。これが 0 になる場合に後手必勝である。実はどちらにとっても先手必勝になるような初期状態は存在しない。

山の高さは最大 40 なので、このような特徴量は 64 ビットの整数、あるいは倍精度浮動小数点数で表現できる。候補となる山の特徴量を計算しておけば、その和を 0 にするような組み合わせを見つけるのは部分和問題で NP 完全だが、この問題の入力データ範囲程度ならなんとかなる。まず候補の山を半数ずつに分ける。候補は最大40山だから半分なら20で、その片方の部分和すべての個数は最大2^{20}である。この程度までなら十分メモリに収まるし計算時間も許容範囲である。このふたつを昇順と降順にソートしてから突き合わせていけば、和が 0 になる組すべてをみつけられる。もっとも手間がかかるのはソーティングで、計算量は候補の山の個数 n に対して $O\left(n \times 2^{n/2}\right)$ である。

この問題は解析がそう簡単ではなく、場合によってはどのチームも解けないかもしれないと考えていたが、1 チームだけが正答に達した。結局、この問題を解けたか否か優勝と準優勝を分けた。

以上