Wiktor Rzeźniczak

152512

Sprawozdanie z zadania czwartego

int binomial(int n, int k):
Funkcja oblicza dwumian Newtona zadanych parametrami n i k, które odpowiadają górnej i dolnej części wyrażenia**.**

void FileRead(string filename):
Funkcja wczytuje plik z zadanym parametrem nazwy (plik musi być w folderze Instancje) jako strumień, czyli omija białe znaki. Następnie taki strumień, po rzutowaniu na int, jest przekazywany do liniowego, dwuwymiarowego wektora integerów Instance. Drugi wiersz wektora Instance jest wypełniany zerami, co dla programu oznacza, że ten dany element 'kolumnę wyżej' jest nieużyty.

int CheckCutNum(int elem_cnt):
Funkcja zachłannie przyrównuje początkową wartość do dwumianu Newtona zawartego na stronie oblicza liczbę elementów jaką będzie zawierała mapa w oparciu o liczność zbioru instancji (elem_cnt) i zwraca w postaci zmiennej n.

int Sum(vector<int> tmp):
Funkcja oblicza sumę wszystkich elementów wektora zadanego jako argument. Wykorzystanie ogólne do wielu celów w późniejszych etapach.

void GetSetFirstElem(vector<int> tmp):
Tutaj ustawiana jest wartość pierwszego elementu wyjściowej mapy MapOfCuts. Oblicza się ją i ustawia w ciele tej funkcji. Obliczenie to polega na odjęciu największego (MapLen) i drugiego w kolejności największego elementu wektora Instance (first)od siebie. Ustawia się też wtedy drugi wiersz wektora Instance w miejscu wystąpienia first na 1. W konsekwencji MapOfCuts ma już podany jeden element, co w praktyce bardzo nieznacznie wpływa na złożoność obliczeniową.

bool CheckAnswer(vector<int> tmp_answer, int elem):
Druga co do ważności funkcja, która sprawdza zawartość dotychczasowej odpowiedzi, plus element który jest kandydatem na dodanie do rozwiązanie. Począwszy od zbioru wielkości 2, sumuje wszystkie k-elementowe zbiory tmp_answer i sprawdza czy te sumy znajdują się w Instance. Wykorzystana jest tu funkcja Sum(vector<int> *arg). Jeśli cokolwiek się nie znajdzie w Instance, zwracana jest wartość boolowska nieprawdy. Złożoność $O(k \log_k * n)$.

void GiveMeMapOfCutsPlease(vector<vector<int>> tmp, vector<int> answer):
Główna funkcja programu, która za pierwszym wywołaniem przyjmuje oryginalne instancje i rozwiązanie (oczywiście bez rozwiązania – tylko z pierwszym elementem), a potem ich lokalne kopie na których funkcja operuje. W pętli przegląda się cały wektor Instance i sprawdza się, czy suma dotychczasowego rozwiązania i element przeglądany obecnie są mniejsze od wymaganej wielkości MapLen. Następnie sprawdza się funkcją CheckAnswer, czy zadany element może zostać dodany. Na końcu odczytuję się flagę pseudo-boolowską danego elementu, czy został już on użyty czy nie. Jeśli nie został użyty, dodaje się tą wartość do lokalnego rozwiązania i ustawia się flagę na 1.

Jeśli można coś jeszcze dodać (czyli liczność zbioru rozwiązania jest mniejsza od n), wywołuje się rekurencyjnie tę funkcję z lokalnymi argumentami. Rekurencja się powtarza tak długo, aż nie osiągnie się n elementów w rozwiązaniu i suma elementów nie osiągnie MapLen. Złożoność $O(n\text{-}1^{n-1})$.

```
Instancja ‘ins-PDP-11a-asc’:
2 3 4 4 5 5 6 6 6 7 7 8 8 9 10 11 11 11 11 12 13 13 14 14 15 15 16 16 17 18 19 19
19 20 20 21 21 22 23 23 25 25 25 26 26 27 27 27 30 30 31 31 32 32 35 35 36 36 38 38
38 40 41 42 42 43 44 46 46 49 51 52 53 57 57 61 63 67
Dlugosc mapy = 67
Liczba elementow: 12
Wynik:
4 11 6 2 3 5 5 6 4 7 8 6
Czas: 1144.27

Instancja 'ins-PDP-12a-asc':
2 3 4 4 5 5 6 6 6 6 7 7 8 8 9 10 11 11 11 11 12 12 13 13 14 14 15 15 16 16 17 18 19
19 19 20 20 20 21 21 22 23 23 25 25 25 26 26 27 27 27 27 30 30 31 31 31 32 32 33 35
35 36 36 38 38 38 38 40 41 42 42 43 44 46 46 47 49 50 51 52 53 57 57 58 61 63 63 67
69 73
Dlugosc mapy = 73
Liczba elementow: 13
Trwało ponad godzinę, przerwano

Instancja 'ins-PDP-13a-asc':
2 3 3 4 4 5 5 6 6 6 6 7 7 8 8 9 9 10 11 11 11 11 12 12 13 13 14 14 15 15 15 16 16
17 18 19 19 19 20 20 20 21 21 22 23 23 23 25 25 25 26 26 27 27 27 27 30 30 30 31 31
31 32 32 33 34 35 35 36 36 36 38 38 38 38 40 41 41 42 42 43 44 46 46 47 49 50 50 51
52 53 53 57 57 58 61 61 63 63 66 67 69 72 73 76
Dlugosc mapy = 76
Liczba elementow: 14
Trwało ponad godzinę, przerwano

Instancja 'ins-PDP-14a-asc':
2 3 3 4 4 5 5 5 6 6 6 6 7 7 8 8 8 9 9 10 11 11 11 11 12 12 13 13 14 14 14 15 15 15
16 16 17 18 19 19 19 20 20 20 20 21 21 22 23 23 23 25 25 25 26 26 27 27 27 27 28 30
30 30 31 31 31 32 32 33 34 35 35 35 36 36 36 38 38 38 38 39 40 41 41 41 42 42 43 44
46 46 46 47 49 50 50 51 52 53 53 55 57 57 58 58 61 61 63 63 66 66 67 69 71 72 73 76
77 81
Dlugosc mapy = 81
Liczba elementow: 15
Trwało ponad godzinę, przerwano

Instancja ‘ins-PDP-11b-asc:
12 13 15 25 27 35 38 42 47 48 54 57 60 66 66 74 79 82 89 93 95 101 104 107 108 112
114 123 135 136 139 146 150 151 161 167 171 184 186 193 196 199 205 209 211 212 224
228 239 241 247 250 253 259 262 265 286 294 307 307 313 319 321 334 346 351 360 373
395 398 400 408 420 433 446 458 474 512
Dlugosc mapy = 512
Liczba elementow: 12
Wynik:
38 74 27 66 42 15 89 47 35 13 12 54
Czas: 0.453

Instancja ‘ins-PDP-11b-desc’:
512 474 458 446 433 420 408 400 398 395 373 360 351 346 334 321 319 313 307 307 294
286 265 262 259 253 250 247 241 239 228 224 212 211 209 205 199 196 193 186 184 171
167 161 151 150 146 139 136 135 123 114 112 108 107 104 101 95 93 89 82 79 74 66 66
60 57 54 48 47 42 38 35 27 25 15 13 12
Dlugosc mapy = 512
Liczba elementow: 12
Wynik:
38 74 27 66 42 15 89 47 35 13 12 54
Czas: 0.437
```

```
Instancja ‘ins-PDP-14b-asc’:
12 13 15 25 25 27 35 38 42 47 48 54 57 57 60 63 66 66 74 79 79 82 82 89 93 95 101
104 107 108 112 114 120 123 135 136 136 137 139 146 150 151 161 161 164 167 171 184
186 193 194 196 199 199 205 209 211 212 221 224 228 230 239 241 247 250 253 259 262
265 272 273 286 287 287 294 300 307 307 313 319 321 329 334 344 346 351 360 366 373
376 395 398 400 408 408 420 423 423 433 433 446 458 458 471 474 480 483 512 512 515
528 537 540 559 594 594 607 619 673
Dlugosc mapy = 673
Liczba elementow: 15
Wynik:
54 12 13 35 47 89 15 42 66 27 74 38 25 57 79
Czas: 21.174

Instancja 'ins-PDP-14b-desc':
673 619 607 594 594 559 540 537 528 515 512 512 483 480 474 471 458 458 446 433 433
423 423 420 408 408 400 398 395 376 373 366 360 351 346 344 334 329 321 319 313 307
307 300 294 287 287 286 273 272 265 262 259 253 250 247 241 239 230 228 224 221 212
211 209 205 199 199 196 194 193 186 184 171 167 164 161 161 151 150 146 139 137 136
136 135 123 120 114 112 108 107 104 101 95 93 89 82 82 79 79 74 66 66 63 60 57 57
54 48 47 42 38 35 27 25 25 15 13 12
Dlugosc mapy = 673
Liczba elementow: 15
Wynik:
54 12 13 35 47 89 15 42 66 27 74 38 25 57 79
Czas: 22.393
```

Własne instancje:

```
Instancja 'k5inst1':
3 3 5 8 8 6 3 2 9 3 1 5 2 6 1 2 10 5 11 7 8
Dlugosc mapy = 11
Liczba elementow: 6
Wynik:
1 2 3 2 1 2
Czas: 0.015

Instancja 'k5inst2':
79 33 57 21 45 24 55 13 42 62 22 8 25 20 34 12 38 17 5 17 37
Dlugosc mapy = 79
Liczba elementow: 6
Wynik:
17 5 12 8 13 24
Czas: 0.004

Instancja 'k8inst1':
5 6 8 12 13 14 16 17 19 20 20 21 22 24 25 25 31 33 33 36 37 37 38 39 41 45 45 49 52
53 55 57 58 61 62 69 74 74 76 82 86 94 98 107 131
Dlugosc mapy = 131
Liczba elementow: 9
Wynik:
24 13 8 12 5 14 6 16 33
Czas: 0.162

Instancja 'k8inst2':
17 25 29 31 38 47 55 58 69 8 12 14 21 30 38 41 52 4 6 13 22 30 33 44 2 9 18 26 29
40 7 16 24 27 38 9 17 20 31 8 11 22 3 14 11
Dlugosc mapy = 69
Liczba elementow: 9
Wynik:
11 3 8 9 7 2 4 8 17
Czas: 0.906
```

```
Instancja 'k11inst1':
12 7 34 35 82 75 89 41 51 58 135 70 121 161 26 119 37 5 152 82 16 21 40 96 128 126
94 14 88 130 114 103 55 14 25 32 2 80 98 13 27 6 72 8 90 66 29 105 84 9 9 154 95 61
46 101 127 14 86 28 74 69 4 60 140 31 20 126 68 156 86 66 23 68 2 117 54 92
Dlugosc mapy = 161
Liczba elementow: 12
Wynik:
5 2 2 12 13 41 14 6 8 23 9 26
Czas: 8.454

Instancja 'k11inst2':
40 46 420 63 153 35 373 232 434 180 385 359 14 93 206 86 26 186 88 266 348 23 285
313 238 72 139 227 325 134 174 138 192 28 146 60 262 12 322 14 185 47 245 199 123
169 215 406 37 46 42 77 252 278 287 160 299 46 92 140 100 58 98 214 121 61 306 260
345 114 383 51 92 107 168 220 446 128
Dlugosc mapy = 446
Liczba elementow: 12
Wynik:
12 14 14 23 35 42 46 46 46 47 60 61
Czas: 2.735

Instancja 'k14inst1':
13 27 108 471 66 79 42 35 112 57 193 321 265 607 209 515 594 346 74 95 239 194 161
224 433 594 82 136 199 164 559 57 446 344 408 373 47 273 104 329 307 474 287 54 171
376 423 186 673 48 38 351 136 25 241 300 89 151 287 512 93 101 25 123 79 540 228
294 139 107 60 259 400 398 262 272 480 12 360 199 458 63 458 205 146 528 408 619
253 184 433 286 196 114 120 537 395 221 319 313 247 135 167 230 150 420 334 512 366
483 250 82 212 423 137 66 161 307 211 15
Dlugosc mapy = 673
Liczba elementow: 15
Wynik:
54 12 13 35 47 89 15 42 66 27 74 38 25 57 79
Czas: 21.721

Instancja 'k14inst2':
26 54 216 942 132 158 84 70 224 114 386 642 530 1214 418 1030 1188 692 148 190 478
388 322 448 866 1188 164 272 398 328 1118 114 892 688 816 746 94 546 208 658 614
948 574 108 342 752 846 372 1346 96 76 702 272 50 482 600 178 302 574 1024 186 202
50 246 158 1080 456 588 278 214 120 518 800 796 524 544 960 24 720 398 916 126 916
410 292 1056 816 1238 506 368 866 572 392 228 240 1074 790 442 638 626 494 270 334
460 300 840 668 1024 732 966 500 164 424 846 274 132 322 614 422 30
Dlugosc mapy = 1346
Liczba elementow mapy: 15
Wynik:
108 24 26 70 94 178 30 84 132 54 148 76 50 114 158
Czas: 27.644
```

Wnioski:

Problem jest obliczeniowo trudny, ponieważ jego złożoność przyrasta wykładniczo. Jednak pozytywnie na szybkość rozwiązania mają czynniki takie jak losowe uporządkowanie instancji, rozpiętość liczbowa pomiędzy największym a najmniejszym elementem (im większa tym lepiej) i pomiędzy poszczególnymi elementami, oczywiście liczność instancji i obecność powtórzeń (im mniej tym lepiej).