

# 2010 年度 大問 5

hari64boli64 (hari64boli64@gmail.com)

2023 年 5 月 3 日

## 1 問題

$$\begin{aligned} & \max \sum_{i=1}^n p_i x_i \\ & \text{subject to } \sum_{i=1}^n s_i x_i \leq S \quad (*) \\ & x_i \in \{0, 1\} \quad (i = 1, \dots, n) \quad (*) \end{aligned}$$

## 2 解答

5.py が解答。

slow とあるのは、正当性の検証目的であり、実際には不要。

### ソースコード 1 answer

```
1 import numpy as np
2 import random
3 from itertools import product
4
5
6 def makeProblem(problem: str):
7     if problem == "P":
8         N = 5
9         S = 5
10        Ps = [2, 3, 2, 1, 3]
11        Ss = [2, 3, 1, 2, 1]
12        return N, S, Ps, Ss
13    elif problem == "Q":
14        N = random.randint(1, 5)
```

```

15         S = random.randint(1, 10)
16         Ps = [random.randint(1, 10) for _ in range(N)]
17         Ss = [random.randint(1, 10) for _ in range(N)]
18         return N, S, Ps, Ss
19     elif problem == "R":
20         N = random.randint(1, 5)
21         S = random.randint(1, 10)
22         W = random.randint(1, 10)
23         Ps = [random.randint(1, 10) for _ in range(N)]
24         Ss = [random.randint(1, 10) for _ in range(N)]
25         Ws = [random.randint(1, 10) for _ in range(N)]
26         return N, S, W, Ps, Ss, Ws
27     else:
28         raise ValueError("problem must be P, Q or R")
29
30
31 def slowP(N, S, Ps, Ss):
32     """
33     bit全探索による(P)の解法
34
35     解法を説明すると、
36     1.  $0 \sim 2^N - 1$ までのbitを生成する これは、各  $x_i$  が1かどうかを表す
37     2. そのbitに対応するxについての制約条件を計算する
38     3. 制約条件を満たすならば、その
39         bitに対応するxについての目的関数の値を計算する
40     4. 3.で計算した値の最大値を答えとする
41     というものである。
42
43     これは、bit全探索の計算量が  $O(2^N)$  であり、指数時間アルゴリズム
44     となっている。
45     """
46     ans = -np.inf
47     for bit in range(1 << N):
48         constraint = sum([Ss[i] * bool(bit & (1 << i)) for i in
49                             range(N)])
50         if constraint <= S:
51             obj = sum([Ps[i] * bool(bit & (1 << i)) for i in
52                             range(N)])
53             ans = max(ans, obj)
54     return ans
55
56
57 def slowQ(N, S, Ps, Ss):
58     """
59     全探索による(Q)の解法
60
61     上とほぼ同様

```

```

58     """
59     ans = -np.inf
60     for Xs in product(range(0, 10 + 1), repeat=N):
61         constraint = sum([Ss[i] * Xs[i] for i in range(N)])
62         if constraint <= S:
63             obj = sum([Ps[i] * Xs[i] for i in range(N)])
64             ans = max(ans, obj)
65     return ans
66
67
68 def slowR(N, S, W, Ps, Ss, Ws):
69     """
70     全探索による(R)の解法
71
72     上とほぼ同様
73     """
74     ans = -np.inf
75     for bit in range(1 << N):
76         constraint1 = sum([Ss[i] * bool(bit & (1 << i)) for i in
77                             range(N)])
78         constraint2 = sum([Ws[i] * bool(bit & (1 << i)) for i in
79                             range(N)])
80         if constraint1 <= S and constraint2 <= W:
81             obj = sum([Ps[i] * bool(bit & (1 << i)) for i in
82                         range(N)])
83             ans = max(ans, obj)
84     return ans
85
86
87 def solveP():
88     """
89     (1),(2)の解答
90     これは  $p_1, \dots, p_n, s_1, \dots, s_n$  について多項式時間,  $s$  について指数
91     時間アルゴリズムである。
92     """
93     N, S, Ps, Ss = makeProblem("P")
94     print(f"{N=},{S=},{Ps=},{Ss=}")
95     print(f"{slowP(N, S, Ps, Ss)=}")
96
97     As = [[None for _ in range(S + 1)] for _ in range(N + 1)]
98     for s in range(1, S + 1):
99         As[0][s] = -np.inf
100     As[0][0] = 0
101
102     for j in range(1, N + 1):
103         for s in range(S + 1):
104             if s < Ss[j - 1]:

```

```

101         As[j][s] = As[j - 1][s]
102     else:
103         As[j][s] = max(As[j - 1][s], Ps[j - 1] + As[j -
104             1][s - Ss[j - 1]])
105
106     print("As=", *As, sep="\n")
107     print("ans=", max(max(a) for a in As))
108
109 def solveQ():
110     """
111     (3)の解答
112     これは  $p_1, \dots, p_n, s_1, \dots, s_n$  について多項式時間,  $s$  について指数
113         時間アルゴリズムである。
114     """
115     N, S, Ps, Ss = makeProblem("Q")
116     print(f"{N=}, {S=}, {Ps=}, {Ss=}")
117     print(f"{slowQ(N, S, Ps, Ss)=}")
118
119     As = [[None for _ in range(S + 1)] for _ in range(N + 1)]
120     for s in range(1, S + 1):
121         As[0][s] = -np.inf
122     As[0][0] = 0
123
124     for j in range(1, N + 1):
125         for s in range(S + 1):
126             # ここが漸化式
127             if s < Ss[j - 1]:
128                 As[j][s] = As[j - 1][s]
129             else:
130                 As[j][s] = max(
131                     As[j - 1][s],
132                     # 以下が増えた これは、 $x_i$ を1~10の範囲内で変
133                         化させている
134                     max(
135                         Ps[j - 1] * x + As[j - 1][s - Ss[j - 1] *
136                             x]
137                         for x in range(1, 10 + 1)
138                         if s - Ss[j - 1] * x >= 0
139                     ),
140                 )
141
142     # print("As=", *As, sep="\n")
143     print("ans=", max(max(a) for a in As))
144
145 def solveR():

```

```

144 """
145 (4)の解答
146 これは  $p_1, \dots, p_n, s_1, \dots, s_n, w_1, \dots, w_n$  について多項式時間,  $S$  に
147 について指数時間アルゴリズムである。
148 """
149 N, S, W, Ps, Ss, Ws = makeProblem("R")
150 print(f"{N=},{S=},{W=},{Ps=},{Ss=},{Ws=}")
151 print(f"{slowR(N, S, W, Ps, Ss, Ws)=}")
152
153 # Aの引数にWを追加すれば良い
154 As = [[[None for _ in range(W + 1)] for _ in range(S + 1)]
155         for _ in range(N + 1)]
156 for s in range(S + 1):
157     for w in range(W + 1):
158         if s != 0 or w != 0:
159             As[0][s][w] = -np.inf
160 As[0][0][0] = 0
161
162 for j in range(1, N + 1):
163     for s in range(S + 1):
164         for w in range(W + 1):
165             if s < Ss[j - 1] or w < Ws[j - 1]:
166                 As[j][s][w] = As[j - 1][s][w]
167             else:
168                 As[j][s][w] = max(
169                     As[j - 1][s][w],
170                     Ps[j - 1] + As[j - 1][s - Ss[j - 1]][w -
171                                     Ws[j - 1]],
172                 )
173
174 # print("As=", *As, sep="\n")
175 print("ans=", max(max(max(a) for a in aa) for aa in As))
176
177 def main():
178     solveP()
179     print("=" * 10)
180     solveQ()
181     print("=" * 10)
182     solveR()
183
184 if __name__ == "__main__":
185     main()

```

## ソースコード 2 output

```

1 N=5,S=5,Ps=[2, 3, 2, 1, 3],Ss=[2, 3, 1, 2, 1]

```

```

2  slowP(N, S, Ps, Ss)=8
3  As=
4  [0, -inf, -inf, -inf, -inf, -inf]
5  [0, -inf, 2, -inf, -inf, -inf]
6  [0, -inf, 2, 3, -inf, 5]
7  [0, 2, 2, 4, 5, 5]
8  [0, 2, 2, 4, 5, 5]
9  [0, 3, 5, 5, 7, 8]
10 ans= 8
11 =====
12 N=3,S=3,Ps=[7, 10, 4],Ss=[1, 4, 1]
13 slowQ(N, S, Ps, Ss)=21
14 ans= 21
15 =====
16 N=4,S=2,W=8,Ps=[5, 7, 7, 10],Ss=[2, 8, 7, 2],Ws=[6, 2, 6, 10]
17 slowR(N, S, W, Ps, Ss, Ws)=5
18 ans= 5

```

### 3 知識

特に無し