אלגוריתמים כלכליים

מיכאל טרושקין

---

* שאלה 2: חלוקה מסודרת

כתבו אלגוריתם המקבל שני וקטורי-ערכים, וחלוקה בין שני שחקנים, ובודק אם החלוקה מסודרת.

- אם החלוקה מסודרת, האלגוריתם מחזיר "כן".
- אם החלוקה לא מסודרת, האלגוריתם מחזיר שיפור פארטו שלה.

זה מימוש האלגוריתם שיש בקובץ של ה PDF, בעצם עם החלוקה לא מסודרת קיימים 2 חפצים,
עם ערכים a_1, a_2, b_1, b_2 כאשר b_i זה הערך של חפץ i לשחקן ב, ו a_i זה הערך של חפץ i לשחקן א.
כך שמתקיים:

$$\frac{a_1}{a_2} < \frac{b_1}{b_2}$$
$$and$$
$$\frac{a_1}{b_1} < \frac{a_2}{b_2}$$

אם שניהם לא מתקיימים אז כפי שנאמר בקובץ מספיק למצוא ערכים z, y כך ש:

$$\frac{a_1}{a_2} < \frac{z}{y} < \frac{b_1}{b_2}$$

ערכים כאלו ניתן למצוא בשיטה הבאה, נסמן:

$$\frac{a_1}{b_1} = l,\ \frac{a_2}{b_2} = m$$

נקבע:

$$z = l + m,\ y = 2$$

נשים לב כי z/y זה האמצע בין m, l אבל הם לא ערכים קטנים מ 1, ולכן נחלק את שניהם במספר כלשהו גדול מהגדול מביניהם כלומר:

$$f = \max\{z, y\}$$

ואז:

$$z = z/f,\ y = z/f$$

משום שחילקנו את שניהם אז ה f מצתמצם ונשאר עם הערך האמצע.

כעת נעביר לשחק א, z מהחפץ השני ולשחקן ב y מהחפץ הראשון.
שחקן א, עם כך מפסיק ya_1 אבל מרוויח za_2 נשים לב כי:

$$\frac{a_1}{a_2} < \frac{z}{y} \Longrightarrow za_2 > ya_1$$

ולכן סך הכל הרוויח כנ"ל עבור שחקן ב.

```python
def fixAlgo(v1, v2, partition):
    badI = -1
    badJ = -1
    print(v1,v2)
    for i in range(len(v1)):
        for j in range(len(v1)):
            if i == j:
                continue
            #print(i,j)
            if partition[i] > 0 and partition[j] < 1:
                # Player 1, got item[i] and player 2, got item[j].
                if v1[i] / v2[i] < v1[j] / v2[j] and v1[i] / v1[j] < v2[i] / v2[j]:
                    # print(i,j)
                    # print("item 1 - p1 :",v1[i],"| p2 :", v2[i])
                    # print("item 1 - p1 :",v1[j],"| p2 :", v2[j])
                    badI = i
                    badJ = j
                    break
        if badJ >= 0:
            break
    if(badJ == -1):
        return "yes"

    a = v1[badI]/ v1[badJ]
    b = v2[badI] / v2[badJ]

    z,y = frac(a,b)
    # print("need to find z,y such that", a, "< z/y <", b, "z := ", z, "y := ", y, "| z/y :=", z/y)
    # print(badI, badJ)

    pn = np.array(partition)*1.0
    pn[badI] -= z
    pn[badJ] += y
    print(partition)
    return list(pn)
```