Транспортни проблем и његово решавање

Алекса Шкрбић, IN29/ 2018, aleksa.shrbic@gmail.com
Владимир Трпка, IN41 /2018, vtrpka@gmail.com

I Увод

Извештај се бави анализом метода и алгоритама за решавање транспортног проблема. Транспортни проблем спада у групу проблема за проучавање оптималног транспорта и расподеле ресурса, тачније циљ је пронаћи најбољи (оптималан) начин на који се из више почетних дестинација, достављају ресурси на више одредишних места. Циљ је да се изврши минимизација цене транспорта, шта год она представљала(цену у новчаним јединицама, најкраће растојање транспорта, најмањи утрошак времена). Примена транспортног проблема највише се везује за економију.

II Опис проблема

Што се тиче поступка самог проблема, прво што је неопходно, одабрати методу која ће нам омогућити задавање почетног решења, тј потребно је одредити број ресурса који је потребно провући кроз одређено поље, које представља пресек између капацитета одређеног складишта и потражње која се налази за задату дестинацију. Вредност поља представља број ресурса које је неопходно испоручити од извора до одредишта. Након тога, потребно је пронаћи оптимално решење, а оно представља нову расподелу ресурса тј број ресурса који се испоручују. Ограничење које може да се јави јесте за случај када дати проблем није затвореног типа (балансирани проблем),односно када потражња није еквивалентна понуди, потребно је додатно оптимизовати алгоритам, да буде у стању да ради уколико је понуда мања од потражње или је потражња мања од понуде(и не представља неки проблем јер ће свакако сви потражени ресурси бити намирени).

III Решавање проблема

Балансирање проблема

Пре избора методе за одређивање почетног решења потребно је извршити његово балансирања, тј омогућити алгоритмима да раде са проблемима отвореног типа. Балансирање ће се конкретно примењивати у случајевима када је потражња ресурса већа од понуде. Тада ће алгоритам за балансирање додати вредност за понуде и омогућити креирање проблема затвореног типа (понуда је еквивалентна потраживањима). Уколико се пак деси да је понуда већа од потражње, алгоритам ће и ту креирати затворени проблем тако што ће додати једну вредност за понуду. С обзиром да се ту ради о ресурсима који никада неће бити транспортовани, цена транспорта од извора до те дестинације износиће 0.

За одређивање почетног решења коришћење су три методе: Метод северозападног угла, метод најмањих цена и Вогелов метод.

A. *Метод северозападног угла*

Метод северозападног угла подразумева решавање проблема, кретањем из крајње леве позиције(тј од првог елемента матрице цена). Метода се примењује докле год се капацитет извора не искористи у потпуности или се не испуне сва потраживања. Приликом третирања различитих проблема, ова метода је увек давала највећу вредност цене транспорта, тј показала се доста слабије у односу на поменуте 2 методе, што је и за очекивати, јер је ово једина метода која не узима у обзир цене транспорта него само почетну позицију која се користи за рад алгоритма.

Б. Метод најмањих цена

Метод најмањих цена, подразумева да се у матрици цена пронађе најмања цена и да се кроз то поље (пресек понуде и потражње) провуче што већа вредност ресурса. Поступак се итеративно понавља, односно у сваком следећем кораку тражи се најмања цена кроз коју ће се провући што већа вредност ресурса.

B. Вогелов метод

Последњи и уједно најкомплекснији метод за одабир почетног решења јесте Вогелов Метод. Најмању (најбољу или оптималну) вредност цене транспорта за различите случајеве давао је овај метод. Функционише тако што тражи две најмање вредности цена, за све редове и колоне и одузима их. Потом проналази највећу разлику цена међу редовима и колонама и када њу пронађе, кажњава је тако што тражи и одабира најмању цену у матрици кроз коју покушава да провуче што већу вредност ресурса. Поступак се понавља док се не “намири” свако потраживање или не исцрпе сви ресурси.

```
def severozapadni_ugao(ponuda, potraznja):
    i = 0
    j = 0
    pocetnoResenje_SZ = []
    ponudaCopy = ponuda.copy()
    potraznjaCopy = potraznja.copy()
    while len(pocetnoResenje_SZ) < len(ponuda) + len(potraznja) - 1:
        pon = ponudaCopy[i]
        potr = potraznjaCopy[j]
        v = min(pon, potr)
        ponudaCopy[i] -= v
        potraznjaCopy[j] -= v
        pocetnoResenje_SZ.append(((i, j), v))
        if ponudaCopy[i] == 0 and i < len(ponuda) - 1:
            i += 1
        elif potraznjaCopy[j] == 0 and j < len(potraznja) - 1:
            j += 1
    return pocetnoResenje_SZ
```

Слика 1: Метод северозападног угла

```python
#%%
#Balansiranje problema
def balansiranje_problema(ponuda, potraznja, cene, penali = None):
    ukupna_ponuda = sum(ponuda)
    ukupna_potraznja = sum(potraznja)

    if ukupna_ponuda < ukupna_potraznja:
        if penali is None:
            raise Exception('Supply less than demand, penalties required')
        nova_ponuda = ponuda + [ukupna_potraznja - ukupna_ponuda]
        nove_cene = cene + [penali]
        return nova_ponuda, potraznja, nove_cene
    if ukupna_ponuda > ukupna_potraznja:
        nova_potraznja = potraznja + [ukupna_ponuda - ukupna_potraznja]
        nove_cene = cene + [[0 for _ in potraznja]]
        return ponuda, nova_potraznja, nove_cene
    return ponuda, potraznja, cene
```

Слика 2: Функција Балансирање Проблема

Г. Проналазак оптималног решења

После одређивања почетног решења потребно је извршити његову оптимизацију, односно спровести одговарајуће кораке који ће омогућити додатну минимизацију цене транспорта. У случају транспортног проблема користи се итеративни поступак за проналазак оптималног решења.

Алгоритам се спроводи у неколико корака. Први корак је израчунавање потенцијала (u_i и v_j). Сваком реду и свакој колони потребно је додати још једну вредност, потенцијал(u_i за редове, v_j за колоне). Потенцијали се бирају и рачунају тако да је њихова сума једнака одговарајућој цени, односно $c_{ij} - u_i - v_j = 0$, где индекси (i и j) представљају индексе оптерећења у матрици трошкова. Први потенцијал креира се тако што се дода нула у ред/колону која има највише распоређених ресурса. Наравно почетно решење, сваке од одабраних метода имаће утицај на формирање оптималног решења итеративним поступком.Услов који мора да постоји да би се спровео итеративни поступак, јесте ограничење m + n – 1, тј мора да постоји бар оволико попуњених поља (где су m врсте, а n колоне).

Након одређивања потенцијала u_i и v_j потребно је одредити тежине, тј вредности нових цена за непопуњена поља у матрици цена, по формули $c'_{ij} = c_{ij} – u_i - v_j$. Уколико су све вредности за нове цене позитивне, алгоритам ту стаје и пронађено је оптимално решење тј почетно решење је и оптимално. Следеће је само одредити цену транспорта, а то се ради тако што се број ресурса који су распоређени у сваком пољу, помножи са ценом транспорта.

У супротном, проналази најнегативније решење и од њега креће да формира контуру (многоугао) који ће на својим крајевима садржати само оптерећена поља. Темена контуре се означе са + или – наизменично, с тим да ће почетно поље, тј поље од којег се креће почињати са +. Од поља која су означена ознаком -, бира се оно поље које има најмање оптерећење и ту вредност одузимамо од свих поља означених ознаком - , односно сабирамо са свим пољима која имају ознаку +.

Поступак се понавља док не дођемо до матрице цена која ни у једном свом пољу неће садржати негативну вредност за c'_{ij}

```python
# %%
def potencijali(pocetnoResenje, cene):
    u = [None] * len(cene)
    v = [None] * len(cene[0])
    u[0] = 0
    cena = 0
    pocetnoResenjeCopy = pocetnoResenje.copy()
    while len(pocetnoResenjeCopy) > 0:
        for indeksi, resenje in enumerate(pocetnoResenjeCopy):
            i, j = resenje[0]
            if u[i] is None and v[j] is None: continue

            cena = cene[i][j]
            if u[i] is None:
                u[i] = cena - v[j]
            else:
                v[j] = cena - u[i]
            pocetnoResenjeCopy.pop(indeksi)
            break
    print(u)
    print(v)
    return u, v
```

Слика 3: Ф-ја за рачунање потенцијала у итеративном поступку

```python
#Ovde izgleda trazi gde treba provuci tj kakvu konturu je neophodno napraviti

#Ovde nam trebaju cvorovi kroz koje ce proci petlja
def potencijalniCvorovi(petlja, nijePosecen):
    poslednjiCvor = petlja[-1]
    cvoroviURedu = [n for n in nijePosecen if n[0] == poslednjiCvor[0]]
    cvoroviUKoloni = [n for n in nijePosecen if n[1] == poslednjiCvor[1]]
    if len(petlja) < 2:
        return cvoroviURedu + cvoroviUKoloni
    else:
        prepre_cvor = petlja[-2]
        potez_red = prepre_cvor[0] == poslednjiCvor[0]
        if potez_red:
            return cvoroviUKoloni
        return cvoroviURedu



def pronadjiPetlju(pocetniIndexi, pocetnoPolje, petlja):
    if len(petlja) > 3:
        kraj = len(potencijalniCvorovi(petlja, [pocetnoPolje])) == 1
        if kraj:
            return petlja

    nisuPoseceni = list(set(pocetniIndexi) - set(petlja))
    moguciCvorovi = potencijalniCvorovi(petlja, nisuPoseceni)
    for sledeci_cvor in moguciCvorovi:
        sledecaPetlja = pronadjiPetlju(pocetniIndexi, pocetnoPolje, petlja + [sledeci_cvor])
        if sledecaPetlja:
            return sledecaPetlja
```

Слика 4: Ф-је за избор потенцијалних чворова и проналазак контуре која ће проћи кроз њих

IV Анализа и коментарисање решења

За анализу и коментарисање решења посматраћемо проблем са следећим параметрима:

$$\text{Потраживања} = [3345] \qquad \text{Понуда} = [267]$$

$$\text{Цене} = \begin{bmatrix} 20111513 \\ 17141213 \\ 15121818 \end{bmatrix}$$

Прво што је речено јесте да је потребно урадити балансирање проблема, међутим то овде није случај, из разлога што је ово прпблем затвореног типа, односно сума свих потраживања екивавелентна је суми свих понуда.

А. Метод северозападног угла и итеративни поступак

Применом методе северозападног угла за решавање транспортног проблема добијено је следеће почетно решење:

$$\text{ПочетноРешење}_{C3} = \begin{bmatrix} 2^{20} & 0^{11} & 0^{15} & 0^{13} \\ 1^{17} & 3^{14} & 2^{12} & 0^{13} \\ 0^{15} & 0^{12} & 2^{18} & 5^{18} \end{bmatrix} \quad Z = 249$$

где почетно решење представља нову матрицу расподеле ресурса, а Z представља цену транспорта (у новчаним јединацама, умножак ресурса са ценом транспорта).

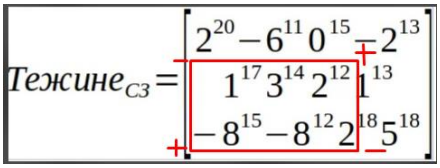
Што се тиче итеративног поступка за методу северозападног угла добијено је следеће: раније је наведено да се прво одређују потенцијали u_i и v_j. добијене су следеће вредности за потенцијале:

$$vj = [20171515] \qquad ui = [0 - 33]$$

Следеће су биле нове цене(тежине) за небазне промењљиве:

$$\text{Тежине}_{\text{СЗ}} = \begin{bmatrix} 2^{20} & -6^{11} & 0^{15} & -2^{13} \\ 1^{17} & 3^{14} & 2^{12} & 1^{13} \\ -8^{15} & -8^{12} & 2^{18} & 5^{18} \end{bmatrix}$$

Видимо да нису добијене све позитивне вредности за тежине, стога алгоритам наставља са радом. Позиција најнегативнијег елемента јесте (2,0), а вредност тежине у том пољу јесте -8. Од те вредности, тј са те позиције потребно је формирати контуру која треба да пролази кроз попуњена поља, а на својим крајевима садржи сва оптерећена поља.



Поља кроз која пролази контура су на следећим позицијама [(2, 0), (2, 2), (1, 2), (1, 0)]. Нове вредности матрице цена, након проласка контуре кроз означене чворове су:

$$\text{Расподела}_{\text{СЗ}} = \begin{bmatrix} 2^{20} & 0^{11} & 0^{15} & 0^{13} \\ 0^{17} & 3^{14} & 3^{12} & 0^{13} \\ 1^{15} & 0^{12} & 1^{18} & 5^{18} \end{bmatrix} \qquad Z = 241$$

Након прве итерације, може се уочити да је добијена мања вредност цене транспорта (Z). По истом поступку алгоритам је извршио још 5 итерација (укупно 6 са првом итерацијом). Финална матрица цена (након поступка оптимизације) изгледа овако:

$$\text{Финалнаматрица}_{\text{СЗ}} = \begin{bmatrix} 0^{20} & 0^{11} & 0^{15} & 2^{13} \\ 0^{17} & 0^{14} & 4^{12} & 2^{13} \\ 3^{15} & 3^{12} & 0^{18} & 1^{18} \end{bmatrix} \qquad Z = 199$$

Б. Метод најмањих цена и итеративни поступак

Применом методе најмањих цена за решавање транспортног проблема добијено је следеће почетно решење:

$$\text{ПочетноРешење}_{\text{НЦ}} = \begin{bmatrix} 0^{20} & 2^{11} & 0^{15} & 0^{13} \\ 0^{17} & 0^{14} & 4^{12} & 2^{13} \\ 3^{15} & 1^{12} & 0^{18} & 3^{18} \end{bmatrix} \qquad Z = 207$$

Уочљиво је да је ова метода дала боље почетно решење у односу на метод северозападног угла, са знатно нижом ценом транспорта.

Применом итеративног поступка добијене су следеће вредности за потенцијале u и v:

$$vj = [14111617] \qquad ui = [0 - 41]$$

Тежине су следеће:

$$\text{Тежине}_{\text{НЦ}} = \begin{bmatrix} 6^{20} & 2^{11} & -1^{15} & -4^{13} \\ 7^{17} & 7^{14} & 4^{12} & 2^{13} \\ 3^{15} & 1^{12} & 1^{18} & 3^{18} \end{bmatrix}$$

Као и код претходне методе, види се да је потребно наставити са радом алгоритма, јер нису добијене све позитивне вредности. Укупно је извршено 3 итерације(за три мање од северозападног угла и добијена је иста финална матрица као и за метод северозападног угла (мада ипак је и поступак потпуно исти)).

$$\text{Финалнаматрица}_{\text{СЗ}} = \begin{bmatrix} 0^{20} & 0^{11} & 0^{15} & 2^{13} \\ 0^{17} & 0^{14} & 4^{12} & 2^{13} \\ 3^{15} & 3^{12} & 0^{18} & 1^{18} \end{bmatrix} \qquad Z = 199$$

Б. Вогелов метод и итеративни поступак

Вогелов метод дао је следеће:

$$\text{Почетнорешење}_{\text{Вогел}} = \begin{bmatrix} 0^{20} & 0^{11} & 0^{15} & 2^{13} \\ 0^{17} & 0^{14} & 4^{12} & 2^{13} \\ 3^{15} & 3^{12} & 0^{18} & 1^{18} \end{bmatrix} \qquad Z = 199$$

Анализом почетног решења и цене транспорта уочљиво је да је овај метод дао најбоље вредности за почетно решење и најнижу цену транспорта Z. Такође се да и приметити и да је нова матрица расподеле ресурса као и сама цена транспорта еквивалентна финалним матрицама расподеле ресурса, које су добијене итеративним поступцима за методе северозападног угла, и методе најмањих цена.
Мада у сваком случају, и ово почетно решење биће провучено кроз итеративни поступак, да би се уочило понашање алгоритма при датитим условима.

Добијени су потенцијали:

$$vj = [15121718] \qquad ui = [-5 - 50]$$

Матрица тежина:

$$\text{МатрицаТежина} = \begin{bmatrix} 10^{20} & 4^{11} & 3^{15} & 2^{13} \\ 7^{17} & 9^{14} & 4^{12} & 2^{13} \\ 3^{15} & 3^{12} & 1^{18} & 1^{18} \end{bmatrix}$$

Одавде је уочљиво да су све вредности у матрици тежина позитивне тако да је ово крај нашег алгоритма, што значи да је Вогелов метод дао почетно решење које је заправо и било оптимално.

VI Коначан алгоритам

Све описане функције спојене су у један, коначни алгоритам који извршава транспортни проблем. Прво што се одради јесте балансирање проблема, тј креирање проблема затвореног типа. Одабира се метода за добијање почетног решења(нека од 3 горе поменуте методе). Потом се то почетно решење прослеђује ф-ји “optimalno”, која ће одредити потенцијале и тежине. Извршава се провера да ли су све тежине позитивне и да ли је то крај алгоритма. Ако јесу прослеђено почетно решење јесте и оптимално, у супротном, проналази се петља и креира ново решење, за које се поново проверава је ли оно оптимално или не.

Проналаском оптималног решења, одређује се и цена транспорта

```
def transportniProblem(ponuda, potraznja, cene, penali):
    balansiranaPonuda, balansiranaPotraznja, balansiraneCene = (balansiranje_problema(ponuda, potraznja, cene, penali))
    #pocetnoResenje = severozapadni_ugao(balansiranaPonuda, balansiranaPotraznja)
    pocetnoResenje = najmanja_cena(balansiranaPonuda,balansiranaPonuda,balansiraneCene)
    optimalnoResenje = optimalno(pocetnoResenje, balansiraneCene)

    matrica = np.zeros((len(balansiraneCene), len(balansiraneCene[0])))
    for (i, j), v in optimalnoResenje:
        matrica[i][j] = v
    trosakTransporta = odrediTrosak(balansiraneCene, matrica)
    return matrica, trosakTransporta

def optimalno(pocetnoResenje, cene):
    u,v = potencijali(pocetnoResenje, cene)
    cij = nove_cene_nebaznih_promenljivih(pocetnoResenje, cene, u, v)

    if(provera(cij)):
        najnegativnije = najmanja_nebazna_cena(cij)
        petlja = pronadjiPetlju([p for p,v in pocetnoResenje], najnegativnije, [najnegativnije])
        return optimalno(novoResenje(pocetnoResenje, petlja), cene)
    return pocetnoResenje
```

Слика 5: Финални алгоритам за одређивање транспортног проблема.

VI Закључак

Из свега приложеног и образложеног, уочљиве су све предности и недостаци сваког од алгоритама. Наиме, уколико је циљ добити што боље почетно решење и добити што минималнију цену транспорта, препоручљиво је користити метод најмањих цена или Вогелов метод(мада је Вогелова метода давала боље почетно решење). Обе од ових метода узимају у обзир цену транспорта од извора до одредишта, за разлику од методе северозападног угла(што је и речено на почетку). Још једна предност бољег почетног решења лежи у томе што ће знантно бити олакшан итеративни поступак (поменута је процедура горе и није баш једноставна). Међутим уколико корисник нема могућност да ради на рачунару, онда ове две методе могу бити компликоване, посебно Вогелов метод, док је метод северозападног угла, заиста најједноставнији за рад.

Референце:
1.https://www.imsl.com/blog/solving-transportation-problem

2.https://scipbook.readthedocs.io/en/latest/intro.html

3. https://radzion.com/blog/operations/corner