

|  |
| --- |
| МИНОБРНАУКИ РОССИИ |
| Федеральное государственное бюджетное образовательное учреждениевысшего образования"МИРЭА - Российский технологический университет"РТУ МИРЭА |

**Институт** Информационных Технологий

**Кафедра** Вычислительной Техники

**ПРАКТИЧЕСКАЯ РАБОТА №1**

**Метод Парето**

**по дисциплине**

**«Теория принятия решений»**

Студент группы:ИКБО-15-19 Ратников Владислав Витальевич *(Фамилия студента)*

Руководитель работы Железняк Л.М.\_

*(Фамилия преподавателя)*

|  |  |  |
| --- | --- | --- |
|  |  |  |

Москва 2021

Оглавление

**Цель работы3**

**Ход выполнения работы3**

Поиск Парето-оптимального множества4

Указание верхних/нижних границ критериев5

Субоптимизация5

Лексикографическая оптимизация6

Геомерический метод7

Метод Парето

Предметная область : Выбор телефона

Альтернативы: A.2 , A.3, Mi 10 , Mi 10 lite , Mi 10T lite , Mi 10T , Mi 10T pro , Mi 8 , Mi play , mi 9.

Критерии: Толщина(-) , вес(-), опер память(+) , память (+), аккумулятор (+), к.во камер(+).

(-) чем меньше тем лучше

(+) чем больше тем лучше

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Альтернативы | Толщина(мм)- | Вес(г)- | Опер память(Гб)+ | Память (Гб)+ | Аккумулятор(Mah)+ | К.во камер+ |
| А1 | A.2 | 10 | 260 | 4 | 64 | 3010 | 2 |
| А2 | A.3 | 10 | 200 | 4 | 128 | 4030 | 3 |
| А3 | Mi10 | 8 | 200 | 6 | 64 | 4160 | 2 |
| А4 | Mi 10 lite | 10 | 190 | 6 | 64 | 4820 | 3 |
| А5 | Mi 10T lite | 9 | 200 | 6 | 128 | 5000 | 2 |
| А6 | Mi 10T | 8 | 200 | 8 | 128 | 5000 | 3 |
| А7 | Mi 10T pro | 9 | 260 | 8 | 128 | 3400 | 3 |
| А8 | Mi 8 | 8 | 200 | 6 | 64 | 3000 | 2 |
| А9 | Mi Play | 9 | 200 | 4 | 64 | 3300 | 2 |
| А10 | Mi 9 | 8 | 220 | 6 | 64 | 4780 | 2 |

Знаком (-) указывается отрицательное стремление

критерия (чем меньше, тем лучше), а знаком (+) – положительное (чем  больше, тем лучше).

Было определено, что оптимизация по Парето использует отношение  Парето-доминирования, которое отдаёт предпочтение одному объекту перед  другим только» том случае, когда первый объект по всем критериям не хуже  второго и хотя бы но одному из них лучше. При истинности этого условия  первый объект считается доминирующим, а второй - доминируемым. Два  объекта, для которых предпочтение хотя бы, по одному критерию  расходится, считаются несравнимыми

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| А | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | x | x | x | x | x | x | x | x | x | x |
| 2 | A2 | x | x | x | x | x | x | x | x | x |
| 3 | A3 | H | x | x | x | x | x | x | x | x |
| 4 | A4 | H | H | x | x | x | x | x | x | x |
| 5 | A5 | H | H | H | x | x | x | x | x | x |
| 6 | A6 | A6 | A6 | H | A6 | x | x | x | x | x |
| 7 | A7 | H | H | H | H | H | x | x | x | x |
| 8 | H | H | H | H | H | H | H | x | x | x |
| 9 | A8 | H | H | H | H | H | H | H | x | x |
| 10 | A10 | H | H | H | H | H | H | A10 | A10 | x |

Вывод:

Данный подход является не эффективным в случае наличия большого количества неплохих альтернатив и требует дополнительной информации.

**Указание верхних/нижних границ критериев.**

Вес не более 200г. и Оперативной памяти не менее 8 гб.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Альтернативы | Критерии | | | | | |
| Толщина  (мм)- | Вес(г)- | Опер память(Гб)+ | Память (Гб)+ | Аккумулятор  (Mah)+ | К.во камер+ |
| 6 | Mi 10T | 9 | 200 | 8 | 128 | 5000 | 2 |
| 7 | Mi 10T pro | 8 | 200 | 8 | 128 | 3400 | 3 |

Поиск оптимального решения среди ограниченного количества альтернатив.

Вывод:

Решение, найденное с помощью данного метода, является крайне субъективным.

**Субоптимизация.**

Главным критерием является оперативная память, нижняя граница по памяти – 128гб, нижняя граница аккумулятору – 3400 Mah , оптимальным является первое решение.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Альтернативы | Критерии | | | | | |
| Толщина  (мм)- | Вес(г)- | Опер память(Гб)+ | Память (Гб)+ | Аккумулятор  (Mah)+ | К.во камер |
| 6 | Mi 10T | 8 | 200 | 8 | 128 | 5000 | 2 |
| 7 | Mi 10T pro | 9 | 260 | 8 | 128 | 3400 | 3 |
| 5 | Mi 10T lite | 9 | 200 | 6 | 128 | 5000 | 2 |

Поиск оптимального варианта при помощи субоптимизации.

Вывод: Так как в данном методе также, как и в предыдущем выставляются ограничения, то решение тоже является субъективным.

**Лексикографическая оптимизация.**

Порядок критериев: [6, 4, 1, 2, 3, 5]

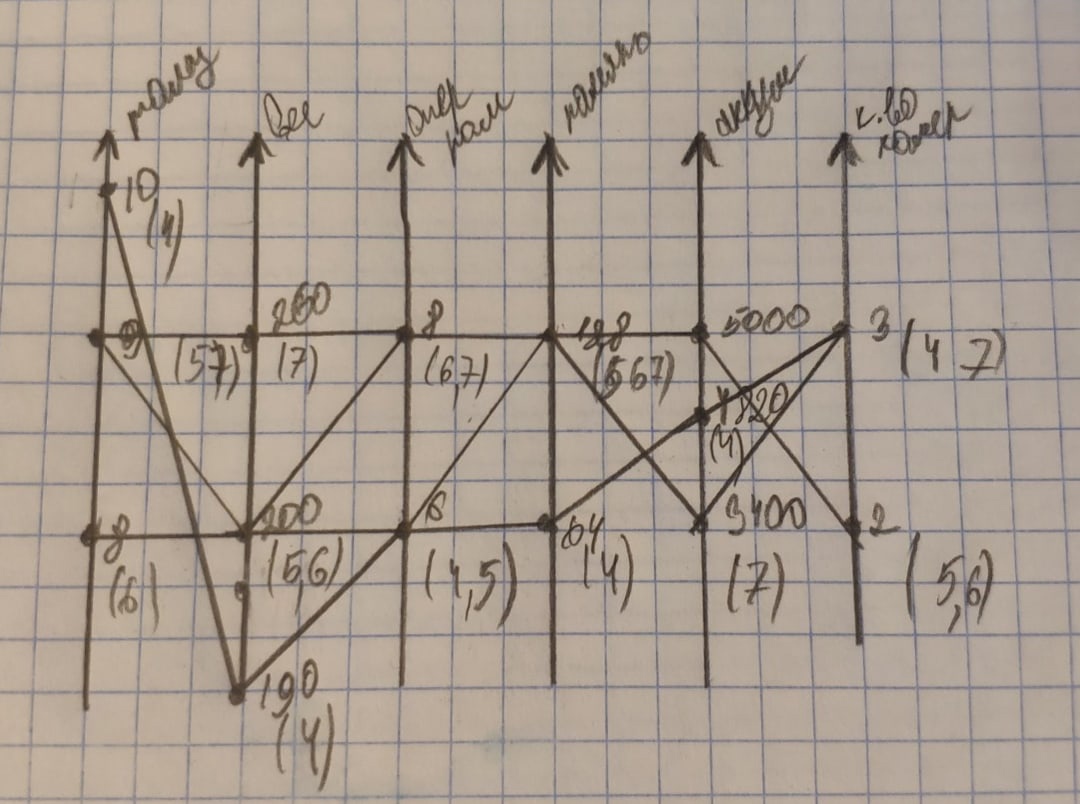
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| № | Альтернативы | Критерии | | | | | |
| Толщина  (мм)- | Вес(г)- | Опер память(Гб)+ | Память (Гб)+ | Аккумулятор  (Mah)+ | К.во камер |
| 6 | Mi 10T | 8 | 200 | 8 | 128 | 5000 | 2 |

Применение метода лексикографической оптимизации.

Вывод:

Главным недостатком данного метода является наличие(учет) при принятии решения главного критерия (оценка состояния).

Геометрический метод.



Код программы:

def output(x):  
 for i in range(len(x)):  
 print(x[i])  
  
  
def inf(x):  
 tab0 = []  
 for i in range(1, len(x)):  
 tab0.append([])  
 for j in range(len(x[i])):  
 tab0[i - 1].append(x[i][j] \* x[0][j])  
 return tab0  
  
  
def show(x):  
 tab2 = []  
 for i in range(len(x)):  
 tab2.append([])  
 for j in range(len(x[i])):  
 if i == 0:  
 if x[i][j] == 1:  
 tab2[i].append('+')  
 else:  
 tab2[i].append('-')  
 else:  
 tab2[i].append(x[i][j])  
 tab2[i].insert(0, str(i))  
 tab2[0][0] = 'K'  
 return tab2  
  
  
def form(x):  
 for j in range(len(x[0])):  
 if x[0][j] == '(+)':  
 x[0][j] = 1  
 else:  
 x[0][j] = -1  
  
  
def alt(x):  
 tab0 = inf(x)  
 tab1 = []  
 for i in range(len(tab0) + 1):  
 tab1.append([])  
 for j in range(len(tab0) + 1):  
 if j == 0:  
 tab1[i].append(str(i))  
 elif i == 0:  
 tab1[i].append(str(j) + ' ')  
 elif j >= i:  
 tab1[i].append('x ')  
 else:  
 tab1[i].append('н ')  
 tab1[0][0] = 'A'  
 for m in range(len(tab0)):  
 for n in range(m):  
 add = True  
 for j in range(len(tab0[m])):  
 if tab0[m][j] < tab0[n][j] or tab0[m] == tab0[n]:  
 add = False  
 if add:  
 tab1[m + 1][n + 1] = 'A' + str(m + 1)  
 return tab1  
  
  
def bord(x, b):  
 tab0 = inf(x)  
 tab2 = show(x)  
 ext = []  
 for i in range(len(tab0)):  
 tab0[i].insert(0, i + 1)  
 for t in range(len(b)):  
 b[t][1] = b[t][1] \* x[0][b[t][0] - 1]  
 for i in range(len(tab0)):  
 if tab0[i][b[t][0]] <= b[t][1]:  
 tab0[i] = [None]  
 for i in range(0, tab0.count([None])):  
 tab0.remove([None])  
 ext.append(tab2[0])  
 for i in range(len(tab0)):  
 ext.append(tab2[tab0[i][0]])  
 return ext  
  
  
def opt(x, o):  
 tab2 = show(x)  
 tab3 = []  
 tab0 = bord(x, o[1])  
 del tab0[0]  
 if x[0][o[0] - 1] == -1:  
 tab0.sort(key=lambda n: n[o[0]])  
 else:  
 tab0.sort(key=lambda n: n[o[0]], reverse=True)  
 tab3.append(tab2[0])  
 for i in range(len(tab0)):  
 tab3.append(tab2[int(tab0[i][0])])  
 return tab3  
  
  
def lex(x, c):  
 tab0 = inf(x)  
 tab2 = show(x)  
 tab4 = []  
 for i in range(len(tab0)):  
 tab0[i].insert(0, i + 1)  
 go = True  
 if go:  
 for i in range(len(c)):  
 tab0.sort(key=lambda n: n[c[i]], reverse=True)  
 for j in range(1, len(tab0)):  
 if tab0[j][c[i]] < tab0[0][c[i]]:  
 tab0[j] = [None]  
 for k in range(0, tab0.count([None])):  
 tab0.remove([None])  
 if len(tab0) == 1:  
 go = False  
 tab4.append(tab2[0])  
 for i in range(len(tab0)):  
 tab4.append(tab2[tab0[i][0]])  
 return tab4  
  
  
def main():  
 form(data)  
 print('\nИсходная таблица:')  
 output(show(data))  
 print('\nТаблица альтернатив:')  
 output(alt(data))  
 print('\nТаблица с установленными верхними и нижними границами:')  
 output(bord(data, border))  
 print('\nМетод субоптимизации:')  
 output(opt(data, optimal))  
 print('\nЛексикографическая оптимизация:')  
 output(lex(data, cri))  
  
  
data = [['(-)', '(-)', '(+)', '(+)', '(+)', '(-)'],  
 [10, 260, 4, 64, 3010, 2],  
 [10, 200, 4, 128, 4030, 3],  
 [8, 200, 6, 64, 4160, 2],  
 [10, 190, 6, 64, 4820, 3],  
 [9, 200, 6, 128, 5000, 2],  
 [8, 200, 8, 128, 5000, 3],  
 [9, 260, 8, 128, 3400, 3],  
 [8, 200, 6, 64, 3000, 2],  
 [9, 200, 4, 64, 3300, 2],  
 [8, 220, 6, 64, 4780, 2]]  
#Верхняя граница  
border = [[1, 9], [5, 5000]]  
#Нижняя граница  
optimal = [1, [[5, 3300], [6, 14]]]  
#Критерии по важности  
cri = [6, 4, 1, 2, 3, 5]  
main()