Министерство образования и науки Российской Федерации

Севастопольский государственный университет

Институт информационных технологий

Кафедра ИС

# ОТЧЕТ

по лабораторной работе №4

ИССЛЕДОВАНИЕ МЕТОДОВ РЕШЕНИЯ МНОГОКРИТЕРИАЛЬНЫХ ЗАДАЧ

ПРИНЯТИЯ РЕШЕНИЙ НА ОСНОВЕ ПОСТРОЕНИЯ МНОЖЕСТВА ПАРЕТО

Выполнил:

ст. гр. ИС/б-21-2-о

Мовенко К. М.

Проверил:

Кротов К.В.

Севастополь

2024

# ЦЕЛЬ РАБОТЫ

Исследовать способы формирования множества Парето-оптимальных решений и определения эффективных решений в этом множестве.

# ЗАДАНИЕ

Вариант 3. Требуется для задаваемого множества в виде: выполнить определение эффективных решений трёхкритериальной задачи выбора с использованием метода идеальной точки. Значения критериев , и для соответствующих решений сведены в матрицу, представленную ниже.

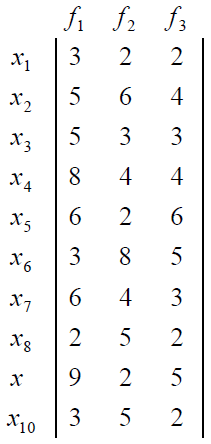


Рисунок 1 – Решения и значения их критериев

# ХОД РАБОТЫ

Из заданного множества решений было получено множество Парето-оптимальных решений. Для этого решения были сравнены отношением строго предпочтения через условие доминирования для их векторных оценок.

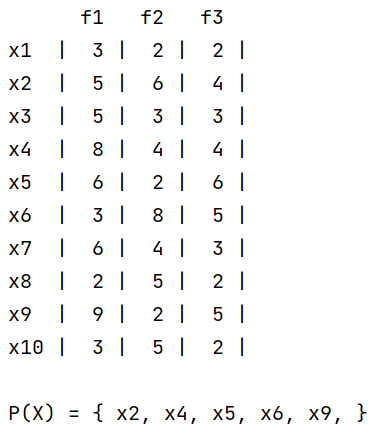


Рисунок 2 – Множество векторных оценок решений и граница Парето

По максимальным значениям скалярных критериев были высчитаны координаты точки утопии в критериальном пространстве. Далее для точки каждого решения в Парето-границе была вычислена метрика до этой точки. По минимальному значению метрики было определено наиболее эффективное решение.

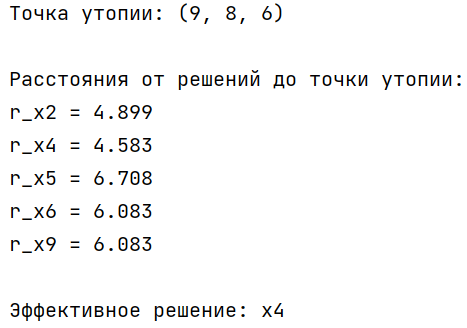


Рисунок 3 – Метод точки утопии для границы Парето

# ТЕКСТ ПРОГРАММЫ

from math import sqrt  
  
  
*# проверка доминирования x1 над x2*def is\_preferable(x1, x2):  
 *# проверка что все fi(x1) >= fi(x2)* all\_greater\_or\_equal = all(fx1 >= fx2 for fx1, fx2 in zip(x1, x2))  
  
 *# проверка что хотя бы один fj(x1) != fj(x2)* at\_least\_one\_not\_equal = any(fx1 != fx2 for fx1, fx2 in zip(x1, x2))  
  
 return all\_greater\_or\_equal and at\_least\_one\_not\_equal  
  
  
*# ВЫЧИСЛЕНИЯ  
  
# множество решений*x = [(3, 2, 2),  
 (5, 6, 4),  
 (5, 3, 3),  
 (8, 4, 4),  
 (6, 2, 6),  
 (3, 8, 5),  
 (6, 4, 3),  
 (2, 5, 2),  
 (9, 2, 5),  
 (3, 5, 2)]  
  
n = len(x) *# число решений*m = len(x[0]) *# число частных критериев  
  
# векторный критерий*f = [[x[i][j] for i in range(n)] for j in range(m)]  
  
y = [] *# значения векторного критерия*P = [] *# множество Парето  
  
# формирование множества Парето-оптимальных решений*for j in range(n):  
 if len(P) == 0:  
 P.append(j)  
 continue  
  
 on\_delete = []  
  
 for i in P:  
 if is\_preferable(x[i], x[j]):  
 break  
  
 if is\_preferable(x[j], x[i]):  
 on\_delete += [i]  
 else:  
 P.append(j)  
 for index in on\_delete:  
 P.remove(index)  
  
*# точка утопии*f\_max = []  
for i in range(m):  
 f\_max.append(max(f[i]))  
  
*# метрики от Парето-оптимальных решений до точки утопии*R = {}  
for i in P:  
 metric = 0  
 for j in range(m):  
 metric += (f\_max[j] - x[i][j]) \*\* 2  
  
 R[i] = sqrt(metric)  
  
*# эффективное решение*x\_opt = min(R, key=R.get) + 1  
  
  
*# ВЫВОД ДАННЫХ  
  
# входные данные: решения и частные критерии*print(' f1 f2 f3 ')  
for i in range(n):  
 print('x%-2i | %2i | %2i | %2i |' % (i+1, f[0][i], f[1][i], f[2][i]))  
print()  
  
*# множество Парето*print('P(X) = { ', end='')  
for i in P:  
 print('x%i' % (i+1), end=', ')  
print('}\n')  
  
print('Точка утопии: (', end='')  
for i in range(m):  
 print(f\_max[i], end=', ' if i < (m-1) else ')\n')  
print()  
  
print('Расстояния от решений до точки утопии:')  
for i, r in R.items():  
 print('r\_x%i = %.3f' % (i+1, r))  
print()  
  
print('Эффективное решение: x%i' % x\_opt)

# ВЫВОД

В ходе работы были исследованы способы формирования множества Парето-оптимальных решений и определения эффективных решений в этом множестве