Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

КУРСОВАЯ РАБОТА

Дисциплина: **Методы оптимизации**
Тема: **Формулировка и решение задачи выбора оптимального решения с использованием различных математических моделей**

Выполнил студент группы 13541/3 _____ Н.С. Бояркин
(подпись)

Руководитель _____ А.Г. Сиднев
(подпись)

Санкт-Петербург
2018 г.

Санкт-Петербургский политехнический университет Петра Великого
Институт компьютерных наук и технологий
Кафедра компьютерных систем и программных технологий

ЗАДАНИЕ
НА ВЫПОЛНЕНИЕ КУРСОВОЙ РАБОТЫ

студенту группы _13541/3_ _____Бояркин Никита Сергеевич_____
(номер группы) *(фамилия, имя, отчество)*

1. Тема работы: Формулировка и решение задачи выбора оптимального решения с использованием различных математических моделей

2. Срок сдачи законченного проекта *29.05.2018*

3. Исходные данные к проекту (работе): Содержательные описания задач поиска оптимальных решений и задачи анализа сетевого графа

4. Содержание пояснительной записки (перечень подлежащих разработке вопросов): введение, формализация и решение многокритериальной оптимизационной задачи, поиск оптимальной стратегии принятия решений с использованием марковских моделей, анализ потокового графа, оптимизация сети систем массового обслуживания, заключение, список использованных источников, приложения.

Дата получения задания: 5 февраля 2018 г.

Руководитель _____ А.Г. Сиднев
(подпись) *(инициалы, фамилия)*

Задание принял к исполнению _____ Н.С. Бояркин_____
(подпись студента) *(инициалы, фамилия)*

5 февраля 2018 г
(дата)

Оглавление

Введение

В данной работе рассматриваются следующие задачи:

1. Формализация многокритериальной оптимизационной задачи, методы сведения к однокритериальной, решение с использованием Optimization Toolbox системы Matlab;
2. Поиск оптимальной стратегии принятия решений с использованием марковских моделей;
3. Оптимизация сетей систем массового обслуживания;
4. Решение задачи анализа потокового графа с использованием методики GERT и алгебры потоковых графов.

Глава 1

Формализация многокритериальной оптимизационной задачи, методы сведения к однокритериальной, решение с использованием Optimization Toolbox системы Matlab.

1.1 Постановка задачи

Вариант 14

Компания Nakia выпускает под своим брендом телефоны трёх ценовых сегментов:

1. LowEnd – розничная цена аппарата 60$, стоимость производства 30$
2. MiddleEnd – розничная цена 300$, стоимость производства 100$
3. HighEnd – цена в розничной сети 2000$, стоимость производства 220$

Мощность фабрик компании такова, что достижимые объёмы выпуска: 70 млн дешевых устройств, 30 млн устройств среднего сегмента, 1 млн дорогих телефонов. Розничная сеть может продать не более 80 млн устройств в год.

Доход фирмы от продажи дополнений, можно заранее оценить по формуле $F = (0.1x_1 + 10x_2 + 70\sqrt{x_3})^{\frac{3}{2}}$.

При выпуске больших партий дешевых телефонов неизбежен антирекламный эффект, обусловленный относительно высоким процентом брака и падением престижа марки. Ущерб от антирекламы можно оценить по формуле $F = ln(20x_1 + 3x_2 + 0.01x_3)$.

По каждому из сегментов компания должна производить не менее половины максимального объема выпуска.

Необходимо найти годовой объём производства телефонов каждого сегмента для достижения

1. Максимизации оборота
2. Минимизации затрат на производство

3. Максимизации средней цены телефона
4. Максимизации выручки от продажи дополнений
5. Минимизации антирекламного эффекта

1.1.1 Решение

1.1.2 Обозначения

Для решения задачи будем использовать следующие обозначения:

- x_1 – количество произведенных дешевых телефонов (в миллионах).
- x_2 – количество произведенных телефонов среднего сегмента (в миллионах).
- x_3 – количество произведенных дорогих телефонов (в миллионах).

1.1.3 Критерии

Введем следующие функции для определения критериев:

$$\begin{cases} F_1 = 60x_1 + 300x_2 + 2000x_3 \\ F_2 = (0.1x_1 + 10x_2 + 70\sqrt{x_3})^{\frac{3}{2}} \\ F_3 = 30x_1 + 100x_2 + 220x_3 \\ F_4 = ln(20x_1 + 3x_2 + 0.01x_3) \\ F_5 = F_1/(x_1 + x_2 + x_3) \\ F_6 = (F_1 + F_2) - (F_3 + F_4) \end{cases}$$

Тогда задача сводится к минимизации или максимизации следующих функций:

1. Максимизация оборота – $F_6 \to max$
2. Минимизация затрат на производство – $F_3 \to min$
3. Максимизация средней цены телефона – $F_5 \to max$
4. Максимизация выручки от продажи дополнений – $F_2 \to max$
5. Минимизация антирекламного эффекта – $F_4 \to min$

Значения взяты в миллионах для повышения точности вычислений в MATLAB, особенно для сверток. Кроме того, такое решение подразумевается исходя из задания.

*** Функция F_1 используется только для вычисления критерия F_6, поэтому не будет впоследствии расчитываться отдельно.**

1.1.4 Ограничения

Формализуем ограничения, приведенные в формулировке задания:

$$\begin{cases} x_1 \leq 70 \\ -x_1 \leq -35 \\ x_2 \leq 30 \\ -x_2 \leq -15 \\ x_3 \leq 1 \\ -x_3 \leq -0.5 \\ x_1 + x_2 + x_3 \leq 80 \end{cases}$$

1.1.5 Поиск оптимумов частных критериев

Разработаем программу для MATLAB, которая решает задачу, в соответствии с ограничениями, для каждого из критериев (Приложение 1).

Результат решения задачи:

```
-- F1 max --
x1 = 49.000, x2 = 30.000, x3 = 1.000, x1 + x2 + x3 = 80.000
!F1 = 13940.000, F2 = 7258.939, F3 = 4690.000, F4 = 6.975, F5 = 174.250, F6 =
    ↪ 16501.963

-- F2 max --
x1 = 49.000, x2 = 30.000, x3 = 1.000, x1 + x2 + x3 = 80.000
F1 = 13939.995, !F2 = 7258.939, F3 = 4689.998, F4 = 6.975, F5 = 174.250, F6 =
    ↪ 16501.961

-- F3 min --
x1 = 35.000, x2 = 15.000, x3 = 0.500, x1 + x2 + x3 = 50.500
F1 = 7600.000, F2 = 2892.251, !F3 = 2660.000, F4 = 6.613, F5 = 150.495, F6 =
    ↪ 7825.638

-- F4 min --
x1 = 35.000, x2 = 15.000, x3 = 0.528, x1 + x2 + x3 = 50.528
F1 = 7656.563, F2 = 2921.812, F3 = 2666.229, !F4 = 6.613, F5 = 151.530, F6 =
    ↪ 7905.533

-- F5 max --
x1 = 35.000, x2 = 30.000, x3 = 1.000, x1 + x2 + x3 = 66.000
F1 = 13100.000, F2 = 7218.316, F3 = 4270.000, F4 = 6.672, !F5 = 198.485, F6 =
    ↪ 16041.644

-- F6 max --
x1 = 49.000, x2 = 30.000, x3 = 1.000, x1 + x2 + x3 = 80.000
F1 = 13940.000, F2 = 7258.939, F3 = 4690.000, F4 = 6.975, F5 = 174.250, !F6 =
    ↪ 16501.964
```

Результирующие значения в формате таблицы:

*** Функция F_1 используется только для вычисления критерия F_6, поэтому не расчитывается отдельно.**

Таблица 1.1 Результирующие значения поиска оптимумов

	$x_1, Mill$	$x_2, Mill$	$x_3, Mill$	$F_2, Mill\$$	$F_3, Mill\$$	$F_4, Mill\$$	$F_5, Mill\$$	$F_6, Mill\$$
F2 max	49	30	1	7258.939	4689.998	6.975	174.250	16501.961
F3 min	35	15	0.5	2892.251	2660.000	6.613	150.495	7825.638
F4 min	35	15	0.528	2921.812	2666.229	6.613	151.530	7905.533
F5 max	35	30	1	7218.316	4270.000	6.672	198.485	16041.644
F6 max	49	30	1	7258.939	4690.000	6.975	174.250	16501.964

1.1.6 Аддитивная свертка критериев

Для использования метода аддитивной свертки необходимо выполнить нормировку критериев, с тем чтобы сделать их значения соизмеримыми, а единицы измерения безразмерными. Нормировка производится делением функции критерия на модуль ее минимума или максимума.

$$f_i(x) = F_i(x)/|F_i^{extr}|$$

Формула аддитивной свертки имеет вид:

$$F_a(x) = \sum_{i=1}^{N} \lambda_i f_i(x), 0 < \lambda_i < 1, \sum_i \lambda_i = 1,$$

где $f_i(x)$ - критерии оптимальности, N – их общее число, а λ_i – коэффициенты важности. Примем коэффициенты важности равными $\lambda_2 = 0.15, \lambda_3 = 0.15, \lambda_4 = 0.05, \lambda_5 = 0.05, \lambda_6 = 0.6$. Коэффициент при F_6 очевидно наибольший, так как итоговый оборот интересует прежде всего.

Решение задачи при помощи аддитивной свертки (Приложение 2).

Результат решения при помощи аддитивной свертки:

```
-- FS --
FS = -0.491
x1 = 35.000, x2 = 30.000, x3 = 1.000, x1 + x2 + x3 = 66.000
F2 = 7218.316, F3 = 4270.000, F4 = 6.672, F5 = 198.485, F6 = 16041.644
F2/rF2 = 99.440%, F3/rF3 = 160.526%, F4/rF4 = 100.893%, F5/rF5 = 100.000%, F6/rF6
    ↪ = 97.211%
```

Стоит отметить, что при больших значениях минимума или максима функций критерии оптимальности становятся очень маленькими, и поэтому MATLAB выдает не совсем корректные результаты. Это еще раз подтверждает правильность взятия значений $x1, x2, x3$ в миллионах.

*** Функция F_1 используется только для вычисления критерия F_6, поэтому не расчитывается отдельно.**

Таблица 1.2 Результирующие значения метода аддитивной свертки

	result, Mill$	proportion, %	difference, %
F2 max	7218.316	99.44	0.56
F3 min	4270.000	160.526	60.526
F4 min	6.672	100.893	0.893
F5 max	174.25	100	0
F6 max	16501.96	97.211	2.789
Mean			**12.95**

1.1.7 Мультипликативная свертка критериев

Формула мультипликативной свертки имеет вид:

$$F_m(x) = \prod_{i=1}^{N} f_i(x)^{\lambda_i}, 0 < \lambda_i < 1, \sum_i \lambda_i = 1,$$

где $f_i(x)$ - критерии оптимальности, N – их общее число, а λ_i – коэффициенты важности. Коэффициенты важности оставим равными $\lambda_2 = 0.15, \lambda_3 = 0.15, \lambda_4 = 0.05, \lambda_5 = 0.05, \lambda_6 = 0.6$.

Решение задачи при помощи аддитивной свертки (Приложение 3).

Результат решения при помощи мультипликативной свертки:

```
-- FS --
FS = 1.093
x1 = 35.000, x2 = 30.000, x3 = 1.000, x1 + x2 + x3 = 66.000
F2 = 7218.315, F3 = 4270.004, F4 = 6.672, F5 = 198.485, F6 = 16041.646
F2/rF2 = 99.440%, F3/rF3 = 160.526%, F4/rF4 = 100.893%, F5/rF5 = 100.000%, F6/rF6
    ↪ = 97.211%
```

По результирующему значению мультипликативной свертки можно заметить, что она лучше справляется в ситуации с очень маленькими критериями оптимальности, чем адаптивная свертка.

Таблица 1.3 Результирующие значения метода мультипликативной свертки

	result, Mill$	proportion, %	difference, %
F2 max	7218.315	99.44	0.56
F3 min	4270.004	160.526	60.526
F4 min	6.672	100.893	0.893
F5 max	198.485	100	0
F6 max	16041.646	97.211	2.789
Mean			**12.95**

Результат аналогичен аддитивной свертке.

*** Функция F_1 используется только для вычисления критерия F_6, поэтому не расчитывается отдельно.**

1.1.8 Максимин или минимакс

Максиминную свертку представим в следующем виде:

$$C_i(a) = minw_iC_i(a)$$

Решение a^* является наилучшим, если для всех a выполняется условие:

$$C(a^*) \geq C(a)$$

или

$$a^* = argmaxC(a) = argmaxminw_iC_i(a)$$

Решение задачи в среде MATLAB (Приложение 4).

Результат решения задачи:

```
-- FS --
FS = (1.353, 1.353, 1.005, 1.061, 1.285)
x1 = 35.000, x2 = 23.294, x3 = 1.000, x1 + x2 + x3 = 59.294
F2 = 5364.414, F3 = 3599.419, F4 = 6.646, F5 = 187.004, F6 = 12846.607
F2/rF2 = 73.901%, F3/rF3 = 135.317%, F4/rF4 = 100.503%, F5/rF5 = 94.216%, F6/rF6 =
    ↪  77.849%
```

Результат решения задачи в виде таблицы:

Таблица 1.4 Результирующие значения метода максимин

	result, Mill$	proportion, %	difference, %
F2 max	5364.414	73.901	26.099
F3 min	3599.419	135.317	35.317
F4 min	6.646	100.503	0.503
F5 max	187.004	94.216	5.784
F6 max	12846.607	77.849	22.151
Mean			**17.97**

*** Функция F_1 используется только для вычисления критерия F_6, поэтому не расчитывается отдельно.**

1.1.9 Метод последовательных уступок

Для метода последовательных уступок данные целевые функции не очень подходят, ввиду их нелинейности. Введем три новых критерия, которые хорошо иллюстрируют метод последовательных уступок:

$$\begin{cases} F_7 = 30x_1 + 200x_2 + 1780x_3 \\ F_8 = x_1 + 10x_3 \\ F_9 = x_1 + 2x_2 \end{cases}$$

Тогда задача сводится к минимизации или максимизации следующих функций:

1. Максимизация оборота (без антирекламного эффекта и продажи дополнений) – $F_7 \rightarrow max$
2. Максимизация доли дешевых и дорогих телефонов – $F_8 \rightarrow max$
3. Минимизация доли средних и дешевых телефонов – $F_9 \rightarrow min$

Расположим критерии в порядке значимости:

$$F_7 > F_8 > F_9$$

Для решения задачи была выбрана уступка равная 15%.

Решение задачи для критерия F_7 (Приложение 5).

Результат решения задачи для критерия F_7:

```
-- F7 max --
x1 = 49.000, x2 = 30.000, x3 = 1.000, x1 + x2 + x3 = 80.000
!F7 = 9250.000, F8 = 59.000, F9 = 109.000
```

Результаты решения для критерия F_7 в виде таблицы:

Таблица 1.5 Результирующие значения метода последовательных уступок для F_7

	result, Mill$	proportion, %	difference, %
F7	9250	100	0
F8	59	-	-
F9	109	-	-

Максимум целевой функции F_7 равен 9250. Для расчета максимума F_8 будет добавлено новое ограничение:

$$F_7 <= 9250 * 0.85$$

$$F_7 <= 7862.5$$

Решение задачи для критерия F_8 (Приложение 5).

Результат решения задачи для критерия F_8:

```
9250.000 >= F7 >= 0.8 * 9250.000
9250.000 >= F7 >= 7862.500

-- F8 max --
x1 = 57.162, x2 = 21.838, x3 = 1.000, x1 + x2 + x3 = 80.000
F7 = 7862.500, !F8 = 67.162, F9 = 100.838
```

Результаты решения для критерия F_8 в виде таблицы:

Максимум целевой функции F_8 равен 67.162. Для расчета максимума F_9 будет добавлено новое ограничение:

Таблица 1.6 Результирующие значения метода последовательных уступок для F_8

	result, Mill$	proportion, %	difference, %
F7	7862.5	85	15
F8	67.162	100	0
F9	100.838	-	-

$$F_8 <= 67.162 * 0.85$$

$$F_8 <= 57.088$$

Решение задачи для критерия F_9 (Приложение 5).

Результат решения задачи для критерия F_9:

```
9250.000 >= F7 >= 0.85 * 9250.000
9250.000 >= F7 >= 7862.500

67.162 >= F8 >= 0.85 * 67.162
67.162 >= F8 >= 57.088

-- F9 max --
x1 = 47.088, x2 = 23.349, x3 = 1.000, x1 + x2 + x3 = 71.437
F7 = 7862.500, F8 = 57.088, !F9 = 93.786
```

Результаты решения для критерия F_9 в виде таблицы:

Таблица 1.7 Результирующие значения метода последовательных уступок для F_9

	result, Mill$	proportion, %	difference, %
F7	7862.5	85	15
F8	57.088	85	15
F9	93.786	100	0

Отличие результатов целевых функций от максимальных или минимальных значений не превышают принятое значение уступки 15%.

1.1.10 Метод достижения цели (fgoalattain)

Функция fgoalattain решает задачу достижения цели, которая является одной из формулировок задач для векторной оптимизации. Аргументы fgoalattain схожи с функцией fgoalattain, за исключением добавления целевых значений и весов. Кроме того, одновременно ищутся оптимальные значения для всех целевых функций, а не для одной.

Решение задачи в среде MATLAB (Приложение 6).

Результат решения задачи:

```
-- FS --
FS = (-5038.049, 3473.833, 6.641, -184.559, -12269.075)
x1 = 35.000, x2 = 22.038, x3 = 1.000, x1 + x2 + x3 = 58.038
F2 = 5038.049, F3 = 3473.833, F4 = 6.641, F5 = 184.559, F6 = 12269.075
F2/rF2 = 69.405%, F3/rF3 = 130.595%, F4/rF4 = 100.429%, F5/rF5 = 92.984%, F6/rF6 =
    ↪ 74.349%
```

Результаты решения для критерия в виде таблицы:

Таблица 1.8 Результирующие значения метода достижения цели

	result, Mill$	proportion, %	difference, %
F2 max	5038.049	69.405	30.595
F3 min	3473.833	130.595	30.595
F4 min	6.641	100.429	0.429
F5 max	184.559	92.984	7.016
F6 max	12269.075	74.349	25.651
Mean			**18.86**

*** Функция F_1 используется только для вычисления критерия F_6, поэтому не расчитывается отдельно.**

1.1.11 Введение метрики в пространстве критериев

Для перехода к однокритериальной задаче оптимизации методом введения метрики в пространстве целевых функций необходимо определить координаты идеальной точки $a_i = (f_1^*, f_2^*, ..., f_1^N)$, где $f_i = min(f_i(x))$. Данные оптимальные значения уже известны из предыдущих пунктов работы и равняются:

$$a = [7258.939, 2660, 6.613, 198.485, 16501.964]$$

Введем в пространстве критериев метрику в виде евклидова расстояния:

$$p(y, a) = (\sum_{i=1}^{N}(a_i - y_i)^2)^{\frac{1}{2}}$$

Тогда за целевую функцию (обобщенный критерий), с учётом необходимости нормировки, можно взять выражение:

$$f = \sum_{i=1}^{N}(\frac{a_i - f_i}{f_i^*})^2 = \sum_{i=1}^{N}(1 - \frac{f_i}{f_i^*})^2$$

Решение задачи в среде MATLAB (Приложение 7).

Результат решения задачи:

```
-- FS --
FS = 0.243
x1 = 35.000, x2 = 24.145, x3 = 1.000, x1 + x2 + x3 = 60.145
F2 = 5589.230, F3 = 3684.450, F4 = 6.650, F5 = 188.602, F6 = 13241.479
F2/rF2 = 76.998%, F3/rF3 = 138.513%, F4/rF4 = 100.553%, F5/rF5 = 95.021%, F6/rF6 =
    ↪  80.242%
```

Результаты решения для критерия в виде таблицы:

Таблица 1.9 Результирующие значения метода введения метрики в пространстве критериев

	result, Mill$	proportion, %	difference, %
F2 max	5589.230	76.998	23.002
F3 min	3684.450	138.513	38.513
F4 min	6.650	100.553	0.553
F5 max	188.602	95.021	4.979
F6 max	13241.479	80.242	19.758
Mean			**17.361**

*** Функция F_1 используется только для вычисления критерия F_6, поэтому не расчитывается отдельно.**

1.1.12 Оценка Парето-оптимальности полученных решений

Выделим результаты решения задачи различными методами в отдельную таблицу. Метод последовательных уступок из-за нелинейности некоторых целевых функций не попадает в таблицу.

*** Функция F_1 используется только для вычисления критерия F_6, поэтому не расчитывается отдельно.**

Парето-оптимальными по соотношению доходов к расходам можно назвать только аддитивную и мультипликативную светку, так как при их использовании получается наибольший оборот (F_6), а оборот как раз таки и характеризует разницу между доходами и расходами. Для этих методов наибольший оборот получается благодаря введению коэффициентов значимости.

1.1.13 Решение задачи стохастического программирования

Рассмотрим задачу стохастического программирования на основе задачи однокритериальной оптимизации, которая была получена из исходной методом введения метрики в пространстве критериев.

Преобразуем последнее ограничение системы:

$$x_1 + x_2 + x_3 \leq 80$$

в вероятностное, тогда:

$$P\{\alpha_1 x_1 + \alpha_2 x_2 + \alpha_3 x_3 \leq 80\} \geq \alpha$$

Таблица 1.10 Оценка Парето-оптимальности полученных решений

Метод	$x_1, Mill$	$x_2, Mill$	$x_3, Mill$	$F_2, Mill\$$	$F_3, Mill\$$	$F_4, Mill\$$	$F_5, Mill\$$	$F_6, Mill\$$	Средняя разница %
Аддитивная свертка	35	30	1	7218.316	4270.000	6.672	198.485	16041.644	**12.95**
Мультипликативная свертка	35	30	1	7218.315	4270.004	6.672	198.485	16041.646	**12.95**
Минимакс	35	23.294	1	5364.414	3599.419	6.646	187.004	12846.607	**17.97**
Метод последовательных уступок	-	-	-	-	-	-	-	-	**-**
Метод достижения цели	35	22.038	1	5038.049	3473.833	6.641	184.559	12269.075	**18.86**
Введение метрики в пространстве критериев	35	24.145	1	5589.230	3684.450	6.650	188.602	13241.479	**17.361**

где все a_i нормально распределены и имеют следующие математические ожидания и дисперсии:

$$M[\alpha_1] = 1, M[\alpha_2] = 1, M[\alpha_3] = 1$$

$$\sigma[\alpha_1] = 0.5, \sigma[\alpha_2] = 0.5, \sigma[\alpha_3] = 0.5$$

где СКО равняется половине математического ожидания. По таблице функции нормального распределения находим коэффициенты K_α:

$$K_{0.5} = 0, K_{0.6} = 0.2533, K_{0.7} = 0.5244, K_{0.8} = 0.8416, K_{0.9} = 1.2816$$

Таким образом, вероятностное ограничение становится эквивалентно детерминированному неравенству:

$$x_1 + x_2 + x_3 + K_\alpha \sqrt{0.5x_1^2 + 0.5x_2^2 + 0.5x_3^2} \leq 80$$

Решение задачи в среде MATLAB (Приложение 8).

Результат решения задачи:

```
-- FS --
FS = 0.243
x1 = 35.000, x2 = 24.145, x3 = 1.000, x1 + x2 + x3 = 60.145
```

```
F2 = 5589.230, F3 = 3684.450, F4 = 6.650, F5 = 188.602, F6 = 13241.479
F2/rF2 = 76.998%, F3/rF3 = 138.513%, F4/rF4 = 100.553%, F5/rF5 = 95.021%, F6/rF6 =
    ↪ 80.242%
Mean = 17.361%

-- a = 0.1, Ka = -1.282 --
x1 = 35.000, x2 = 24.145, x3 = 1.000, x1 + x2 + x3 = 60.145
F2 = 5589.229, F3 = 3684.451, F4 = 6.650, F5 = 188.601, F6 = 13241.473
F2/rF2 = 76.998%, F3/rF3 = 138.513%, F4/rF4 = 100.553%, F5/rF5 = 95.020%, F6/rF6 =
    ↪ 80.242%
Mean = 17.361%

-- a = 0.2, Ka = -0.842 --
x1 = 35.000, x2 = 24.144, x3 = 1.000, x1 + x2 + x3 = 60.145
F2 = 5589.229, F3 = 3684.450, F4 = 6.650, F5 = 188.602, F6 = 13241.477
F2/rF2 = 76.998%, F3/rF3 = 138.513%, F4/rF4 = 100.553%, F5/rF5 = 95.021%, F6/rF6 =
    ↪ 80.242%
Mean = 17.361%

-- a = 0.3, Ka = -0.524 --
x1 = 35.000, x2 = 24.144, x3 = 1.000, x1 + x2 + x3 = 60.145
F2 = 5589.229, F3 = 3684.450, F4 = 6.650, F5 = 188.602, F6 = 13241.477
F2/rF2 = 76.998%, F3/rF3 = 138.513%, F4/rF4 = 100.553%, F5/rF5 = 95.021%, F6/rF6 =
    ↪ 80.242%
Mean = 17.361%

-- a = 0.4, Ka = -0.253 --
x1 = 35.000, x2 = 24.144, x3 = 1.000, x1 + x2 + x3 = 60.145
F2 = 5589.229, F3 = 3684.450, F4 = 6.650, F5 = 188.602, F6 = 13241.477
F2/rF2 = 76.998%, F3/rF3 = 138.513%, F4/rF4 = 100.553%, F5/rF5 = 95.021%, F6/rF6 =
    ↪ 80.242%
Mean = 17.361%

-- a = 0.5, Ka = 0.000 --
x1 = 35.000, x2 = 24.145, x3 = 1.000, x1 + x2 + x3 = 60.145
F2 = 5589.230, F3 = 3684.450, F4 = 6.650, F5 = 188.602, F6 = 13241.479
F2/rF2 = 76.998%, F3/rF3 = 138.513%, F4/rF4 = 100.553%, F5/rF5 = 95.021%, F6/rF6 =
    ↪ 80.242%
Mean = 17.361%

-- a = 0.6, Ka = 0.253 --
x1 = 35.000, x2 = 24.145, x3 = 1.000, x1 + x2 + x3 = 60.145
F2 = 5589.230, F3 = 3684.450, F4 = 6.650, F5 = 188.602, F6 = 13241.479
F2/rF2 = 76.998%, F3/rF3 = 138.513%, F4/rF4 = 100.553%, F5/rF5 = 95.021%, F6/rF6 =
    ↪ 80.242%
Mean = 17.361%

-- a = 0.7, Ka = 0.524 --
x1 = 35.000, x2 = 24.145, x3 = 1.000, x1 + x2 + x3 = 60.145
F2 = 5589.230, F3 = 3684.450, F4 = 6.650, F5 = 188.602, F6 = 13241.479
F2/rF2 = 76.998%, F3/rF3 = 138.513%, F4/rF4 = 100.553%, F5/rF5 = 95.021%, F6/rF6 =
    ↪ 80.242%
Mean = 17.361%

-- a = 0.8, Ka = 0.842 --
x1 = 35.000, x2 = 20.002, x3 = 1.000, x1 + x2 + x3 = 56.002
F2 = 4523.611, F3 = 3270.208, F4 = 6.633, F5 = 180.361, F6 = 11347.389
F2/rF2 = 62.318%, F3/rF3 = 122.940%, F4/rF4 = 100.308%, F5/rF5 = 90.869%, F6/rF6 =
    ↪ 68.764%
Mean = 20.259%
```

```
-- a = 0.9, Ka = 1.282 --
x1 = 34.155, x2 = 14.355, x3 = 0.000, x1 + x2 + x3 = 48.510
F2 = 1781.719, F3 = 2460.145, F4 = 6.588, F5 = 131.020, F6 = 5670.774
F2/rF2 = 24.545%, F3/rF3 = 92.487%, F4/rF4 = 99.619%, F5/rF5 = 66.010%, F6/rF6 =
    ↪ 34.364%
Mean = 36.595%
```

Результаты решения для критерия в виде таблицы:

Таблица 1.11 Решение задачи стохастического программирования

α	K_α	$x_1, Mill$	$x_2, Mill$	$x_3, Mill$	$F_2, Mill\$$	$F_3, Mill\$$	$F_4, Mill\$$	$F_5, Mill\$$	$F_6, Mill\$$	Средняя разница %
det	det	35	24.145	1	5589.230	3684.450	6.650	188.602	13241.479	**17.361**
0.1	-1.282	35	24.145	1	5589.230	3684.450	6.650	188.602	13241.479	**17.361**
0.2	-0.842	35	24.145	1	5589.230	3684.450	6.650	188.602	13241.479	**17.361**
0.3	-0.524	35	24.145	1	5589.230	3684.450	6.650	188.602	13241.479	**17.361**
0.4	-0.253	35	24.145	1	5589.230	3684.450	6.650	188.602	13241.479	**17.361**
0.5	0	35	24.145	1	5589.230	3684.450	6.650	188.602	13241.479	**17.361**
0.6	0.253	35	24.145	1	5589.230	3684.450	6.650	188.602	13241.479	**17.361**
0.7	0.524	35	24.145	1	5589.230	3684.450	6.650	188.602	13241.479	**17.361**
0.8	0.842	35	20.002	1	4523.611	3270.208	6.633	180.361	11347.389	**20.259**
0.9	1.282	34.155	14.355	0	1781.719	2460.145	6.588	131.020	5670.774	**36.595**

*** Функция F_1 используется только для вычисления критерия F_6, поэтому не расчитывается отдельно.**

Увеличение доверительной вероятности α приводит к ухудшению результатов решения. Неизменность результатов на промежутке $\alpha = [0.1, 0.7]$ объясняется выбором ограничения.

1.2 Вывод

Можно заметить, что аддитивная и мультипликативная свертка выдают одинаковый, наиболее оптимальный результат. Наилучший результат этих методов обусловлен наличием коэффициентов значимости.

Методы, не подразумевающие введение весовых коэффициентов показывают похожий результат, который в целом хуже, чем у аддитивной и мультипликативной свертки.

Глава 2

Поиск оптимальной стратегии принятия решений с использованием марковских моделей

2.1 Постановка задачи

Вариант 16

Ежедневно утром производится проверка дорогостоящей машины с целью выявления, находится ли она в исправном состоянии, требует мелкого ремонта или нуждается в серьезном ремонте. Обозначим эти состояния 0, 1, 2 соответственно. Если машина находится в совершенно исправном состоянии, то вероятность того, что она останется в таком же состоянии на начало следующего дня, равна p(0|0), вероятность того, что потребуется мелкий ремонт, равна p(1|0) и вероятность того, что возникает необходимость серьезного ремонта, равна p(2|0). В случае когда машина требует ремонта, фирма может прибегнуть к услугам двух ремонтных фирм, одна из которых (фирма F, гарантирующая качество ремонта) взимает плату M за мелкий ремонт и плату R за крупный. Вторая (фирма T, не гарантирующая качества ремонта) взымает соответственно плату m и r, где m<M и r<R. Легко себе представить, что качество работ, производимых фирмой F, выше, чем у фирмы T, что отражается значением вероятности полностью исправного состояния машины на начало следующего за ремонтом дня. Пусть решение d=1 определяет выбор фирмы F и решение d=2 — выбор фирмы T. Обозначим через p(j | i, d) вероятность перехода машины в состояние j на следующем отрезке (j=0,1,2) при условии, что она находится в состоянии I на текущем отрезке (i=1,2) и принимается решение d(d=1,2).

p (0 | 0) = 0.6 (машина осталась исправной)
p (1 | 0) = 0.3 (машина требует мелкого ремонта)
p (2 | 0) = 0.1 (машина требует крупного ремонта)

p (0 | 1, 1) = 0.9 [M = 14] (фирма F выполнила мелкий ремонт)
p (1 | 1, 1) = 0.1 (фирма F в процессе выполнения мелкого ремонта)
p (2 | 1, 1) = 0 (невозможное событие – если выполнен крупный ремонт, то мелкий не нужен)
p (0 | 2, 1) = 0.6 [R = 21] (фирма F выполнила крупный ремонт)
p (1 | 2, 1) = 0.3 [R – M = 7] (фирма F выполнила мелкий ремонт, но надо доделать до крупного)

p (2 | 2, 1) = 0.1 (фирма F в процессе выполнения крупного ремонта)

p (0 | 1, 2) = 0.7 [m = 12] (фирма T выполнила мелкий ремонт)
p (1 | 1, 2) = 0.2 (фирма T в процессе выполнения мелкого ремонта)
p (2 | 1, 2) = 0 (невозможное событие – если выполнен крупный ремонт, то мелкий не нужен)
p (0 | 2, 2) = 0.5 [r = 19] (фирма T выполнила крупный ремонт)
p (1 | 2, 2) = 0.4 [r – m = 7] (фирма T выполнила мелкий ремонт, но надо доделать до крупного)
p (2 | 2, 2) = 0.1 (фирма T в процессе выполнения крупного ремонта)

Найдите оптимальную стратегию и минимальные затраты на отрезке N=∞.

Доп задание

Предположите, что фирме F для выполнения крупного ремонта требуется 1 полный рабочий день, а фирме T — 2 полных рабочих дня. Считайте далее, что фирма — владелец машины несет потери в размере c единиц за каждый день ее простоя. Покажите, как при этих условиях нужно изменить уравнения.

2.2 Решение

2.2.1 Решения и стратегии

Имеется три состояния машины:

- 1 – исправна;
- 2 – легкая поломка (необходим мелкий ремонт);
- 3 – серьезная поломка (необходим крупный ремонт);

Для данных состояний имеется три решения:

- X_1 – ничего не делать;
- X_2 – выбрать фирму F;
- X_3 – выбрать фирму T;

Таким образом, возможны следующие стратегии:

Таблица 2.1 Возможные стратегии

№	Исправна	Легкая поломка	Серьезная поломка
1	$X_1, (-)$	$X_2, (F)$	$X_3, (T)$
2	$X_1, (-)$	$X_2, (F)$	$X_2, (F)$
3	$X_1, (-)$	$X_3, (T)$	$X_3, (T)$
4	$X_1, (-)$	$X_3, (T)$	$X_1, (F)$

2.2.2 Построение матриц переходных вероятностей и матриц расходов

P_1 и P_2 – матрицы переходных вероятностей для компаний F и T соответственно:

$$P_1 = \begin{bmatrix} 0.6 & 0.3 & 0.1 \\ - & - & - \\ - & - & - \end{bmatrix} \; P_2 = \begin{bmatrix} - & - & - \\ 0.9 & 0.1 & 0 \\ 0.6 & 0.3 & 0.1 \end{bmatrix} \; P_3 = \begin{bmatrix} - & - & - \\ 0.7 & 0.2 & 0.1 \\ 0.5 & 0.4 & 0.1 \end{bmatrix}$$

R_1 и R_2 – матрицы расходов для компаний F и T соответственно:

$$R_1 = \begin{bmatrix} 0 & 0 & 0 \\ - & - & - \\ - & - & - \end{bmatrix} \; R_2 = \begin{bmatrix} - & - & - \\ 14 & 0 & 0 \\ 21 & 7 & 0 \end{bmatrix} \; R_3 = \begin{bmatrix} - & - & - \\ 12 & 0 & 0 \\ 19 & 7 & 0 \end{bmatrix}$$

2.2.3 Нахождение величин ожидаемого дохода

Ожидаемый доход вычисляется по формуле:

$\nu_i(X_k) = \sum_{j=1}^{m} p_{i,j}(X_k) r_{i,j}(X_k)$

Тогда для первого решения (ничего не делать):

$\nu_1(X_1) = 0$

$\nu_2(X_1) = -$

$\nu_3(X_1) = -$

Для второго решения (выбрать фирму F):

$\nu_1(X_2) = -$

$\nu_2(X_2) = 0.9 \cdot 14 = 12.6$

$\nu_3(X_2) = 0.6 \cdot 21 + 0.3 \cdot 7 = 14.7$

Тогда для третьего решения (выбрать фирму T):

$\nu_1(X_3) = -$

$\nu_2(X_3) = 0.7 \cdot 12 = 8.4$

$\nu_3(X_3) = 0.5 \cdot 19 + 0.4 \cdot 7 = 12.3$

2.2.4 Формулировка задачи в виде задачи линейного программирования

Приведение к задаче линейного программирования производится следующим образом:

$$\begin{cases} \sum\limits_{j=1}^{m}\sum\limits_{i=1}^{M}\nu_j(X_i)\omega_{ji} \to \max; \\ \sum\limits_{i=1}^{M}\omega_{ji} - \sum\limits_{k=1}^{m}\sum\limits_{i=1}^{M}p_{kj}(X_i)\omega_{ki} = 0, \quad j = \overline{1,m}, \\ \sum\limits_{j=1}^{m}\sum\limits_{i=1}^{M}\omega_{ji} = 1, \\ \omega_{ji} \geqslant 0, j = \overline{1,m},\ i = \overline{1,M}. \end{cases}$$

Для данной задачи:

$$\begin{cases} 0 \cdot w_{11} + 12.6 \cdot w_{22} + 8.4 \cdot w_{23} + 14.7 \cdot w_{32} + 12.3 \cdot w_{33} \to min \\ (1 - 0.6) \cdot w_{11} - 0.9 \cdot w_{22} - 0.7 \cdot w_{23} - 0.6 \cdot w_{32} - 0.5 \cdot w_{33} = 0 \\ -0.3 \cdot w_{11} + (1 - 0.1) \cdot w_{22} + (1 - 0.2) \cdot w_{23} - 0.3 \cdot w_{32} - 0.4 \cdot w_{33} = 0 \\ -0.1 \cdot w_{11} - 0 \cdot w_{22} - 0.1 \cdot w_{23} + (1 - 0.1) \cdot w_{32} + (1 - 0.1) \cdot w_{33} = 0 \\ w_{11} + w_{22} + w_{23} + w_{32} + w_{33} = 1 w_{ij} \geq 0, i = \overline{1,2,3}, j = \overline{1,2,3} \end{cases}$$

2.2.5 Решение задачи

Разработаем скрипт для расчета вероятностей w_{ij} в среде MATLAB (Приложение 9).

Результат расчета вероятностей:

```
w =

    0.618181818534862
   1.64185972428138e-10
    0.28181818131737
   3.52227535171652e-09
    0.0999999964613061

opt =

    3.59727273338617
```

Таким образом машина исправна с вероятностью 61.82%. Для мелкого ремонта следует обращаться в фирму T (28.18%). Для крупного ремонта следует обращаться в фирму F (10%).

2.2.6 Дополнительное задание

Предположите, что фирме F для выполнения крупного ремонта требуется 1 полный рабочий день, а фирме T — 2 полных рабочих дня. Считайте далее, что фирма — владелец машины несет потери в размере c единиц за каждый день ее простоя. Покажите, как при этих условиях нужно изменить уравнения.

Для решения задачи нет смысла определять новые состояния, как это указано в методических указаниях. Достаточно просто изменить матрицы расходов для компаний F и T следующим образом:

$$R_1 = \begin{bmatrix} 0 & 0 & 0 \\ - & - & - \\ - & - & - \end{bmatrix} \; R_2 = \begin{bmatrix} - & - & - \\ 14 & 0 & 0 \\ 21 + c & 7 + c & 0 \end{bmatrix} \; R_3 = \begin{bmatrix} - & - & - \\ 12 & 0 & 0 \\ 19 + 2 \cdot c & 7 + 2 \cdot c & 0 \end{bmatrix}$$

Дальше задача решается аналогичным образом.

2.3 Вывод

Были определены оптимальные стратегии для конкретной задачи. Машина исправна с вероятностью 61.82%. Для мелкого ремонта следует обращаться в фирму T (28.18%). Для крупного ремонта следует обращаться в фирму F (10%).

Линейное программирование позволяет достаточно легко и быстро решать подобные задачи, однако требует некоторых предварительных преобразований перед использованием.

Глава 3

Оптимизация сетей систем массового обслуживания

3.1 Постановка задачи

Вариант 030

Задача №2.

Найти

$$min\ S = \sum_{i=1}^{M} c_i \mu_i^{\alpha_i}$$

при ограничении

$$\lambda = e_1 G_M(N-1)/G_M(N) = \lambda^*.$$

Дано:

$$\left\{ \boldsymbol{\lambda}^*, \boldsymbol{M}, N, \quad \boldsymbol{\pi} = \left\{ p_{ij} \right\}_{\substack{i=\overline{1,M} \\ j=1,M}}, \vec{c} = (c_1, c_2, \ldots, c_M), \vec{\boldsymbol{\alpha}} = (\alpha_1, \alpha_2, \ldots, \alpha_M) \right\}$$

где

$\boldsymbol{M}$ — число узлов;

$\boldsymbol{N}$ — число заявок в сети;

$\vec{c} = (c_1, c_2, \ldots, c_M)$ — вектор, определяющий число каналов в узле;

$\vec{\boldsymbol{\alpha}} = (\alpha_1, \alpha_2, \ldots, \alpha_M)$ — вектор, определяющий коэффициенты важности узлов. Коэффициенты важности α_i узлов ССМО входят в формулу расчета её стоимости.

λ^* — заданная интенсивность потока в 1-м узле: $\lambda_1 = \lambda^*$

$$I_k = \left(I_1, I_2, \ldots, I_k \right)$$

Примечание. Используется следующее обозначение:

030		0	0,1	06	0,3		5	-	3	2_4	I_4
		0,7	0	0,2	0,1						
		0,9	0	0	0,1						
		0,3	0,3	0,4	0						

3.2 Решение

3.2.1 Методика решения простыми итерациями

Задача оптимизации замкнутой однородной сети сводится к решению системы системы нелинейных уравнений:

$$\mu_1 = \lambda^* / U_1(N);$$

$$\mu_1^{a_i} = \frac{c_1 a_1}{c_i a_i} \mu_1^{a_1} \frac{L_i(N) - L_i(N-1)}{L_1(N) - L_1(N-1)}, \quad i = \overline{2, M}.$$

Для многоканальной сети МО используется итерационная процедура расчета среднего числа заявок в очереди. Процедура инициализируется следующими начальными условиями:

$$P_i(0,0) = 1, \quad i = \overline{1, M} \; ; \; r = 1$$

Первый шаг процедуры:

$$\overline{t}_i(r) = \sum_{n=1}^{r} \frac{n}{\mu_i(n)} P_i(n-1, r-1), \quad i = \overline{1, M}$$

$$\mu_i(n) = \begin{cases} n\mu_i, & n < m_i \\ m_i \mu_i, & n \geq m_i \end{cases}$$ где m_i — число каналов в i-м узле.

Второй шаг процедуры:

$$\lambda_1(r) = \frac{r}{\sum_{j=1}^{M} \frac{\omega_j}{\omega_1} \overline{t}_j(r)}$$

Третий шаг процедуры:

$$\begin{cases} P_i(n,r) = \dfrac{\omega_i \lambda_1(r)}{\omega_1 \mu_i(n)} P_i(n-1,\ r-1), \ n = \overline{1, r}, \\ P_i(0,r) = 1 - \sum_{n=1}^{r} P_i(n,r), \end{cases}$$

После этого значение r увеличивается на единицу до достижения N. В результате работы процедуры формируются значения следующих характеристик:

$$\bar{t}_i(N),\ i=\overline{1,M};\quad \{P_i(n,N)\},\ i=\overline{1,M};\ n=\overline{0,N}.$$

3.2.2 Решение задачи методом простых итераций

Разработаем скрипт для расчета результирующего вектора μ в среде MATLAB (Приложение 10).

Результат расчета вектора μ:

```
Iteration #1
Summ u(i) = 26.030301
u(i) = [2.812500, 12.738971, 3.492944, 6.985887]

Iteration #2
Summ u(i) = 2.131577
u(i) = [1.651439, 0.000621, 0.466883, 0.012634]

Iteration #3
Summ u(i) = 4677058315956.762700
u(i) = [881.151124, 4676727098598.208000, 25284.797051, 331191192.606743]

Iteration #4
Summ u(i) = 1.500003
u(i) = [1.500001, 0.000000, 0.000002, 0.000000]

Iteration #5
Summ u(i) = NaN
u(i) = [NaN, NaN, NaN, NaN]
```

Алгоритм ожидаемо не сошелся, так как метод простых итераций не рекомендуется к использованию из-за ненадежности в плане сходимости.

3.2.3 Методика решения через через нормирующую константу методом Ньютона

Задача оптимизации замкнутой однородной сети сводится к решению системы нелинейных уравнений:

$$\mu_1 = \lambda^*/U_1(N);$$
$$\mu_1^{a_i} = \frac{c_1 a_1}{c_i a_i}\mu_1^{a_1}\frac{L_i(N)-L_i(N-1)}{L_1(N)-L_1(N-1)},\quad i=\overline{2,M}.$$

Тогда интенсивность на выходе i -го узла однородной замкнутой сети СМО:

$$\lambda_i(N)=\omega_i G_M(N-1)/G_M(N)$$

Среднее число заявок в граничном узле однородной замкнутой сети СМО:

$$\overline{n_M}(N) = \frac{\sum_{n=1}^{N} n Z_M(n) G_{M-1}(N-n)}{G_M(N)}$$

Методика расчета нормирующей константы:

$$G_1(k) = Z_1(k) = \frac{\omega_i^k}{\prod_{j=1}^{k} \mu_1(j)}, \quad k = \overline{0, N};$$

$$G_r(0) = 1, \quad r = \overline{1, M};$$

$$G_r(k) = \sum_{l=0}^{k} Z_r(l) G_{r-1}(k-l)$$

Алгоритм Ньютона для решения системы нелинейных уравнений (W – матрица якоби для системы уравнений F):

$$x^{(k+1)} = x^{(k)} - W^{-1}(x^{(k)}) \cdot F(x^{(k)}), k = 0, 1, 2, \ldots$$

3.2.4 Решение задачи через нормирующую константу методом Ньютона

Решение задачи через нормирующую константу методом Ньютона с начальным приближением (Приложение 11).

Результат решения задачи методом Ньютона с начальным приближением $u = [10, 10, 10, 10]$ и значением ошибки $e = 1e^{-6}$:

```
Start Conditions
e = 0.000001
w = [1.000000 0.220779 0.805195 0.402597]
u0 = [1.500000 0.331169 1.207792 0.603896]

0 | u = [10.000000 10.000000 10.000000 10.000000] | F(u) = [-8.115572 -9.883869
    ↪ -5.310920 -9.452506] | S = 80.000000
1 | u = [1.950351 0.066599 1.553885 0.345572] | F(u) = [8.060789 3195.369625
    ↪ 0.505314 52.660296] | S = 7.832812
2 | u = [6.139607 0.235956 5.078620 1.243394] | F(u) = [2.780702 4560.949367
    ↪ 0.429830 75.412690] | S = 25.395155
3 | u = [7.236115 0.323986 6.115423 1.745173] | F(u) = [0.461988 2370.124917
    ↪ 0.023511 39.297704] | S = 30.841395
4 | u = [6.386103 0.343664 5.434348 1.897411] | F(u) = [0.083022 901.425404
    ↪ -0.046716 14.756548] | S = 28.123053
5 | u = [5.297395 0.352083 4.512774 1.967574] | F(u) = [0.039631 316.907201
    ↪ -0.023056 4.768651] | S = 24.259653
6 | u = [4.327837 0.363774 3.684367 1.950805] | F(u) = [0.048224 108.424858
    ↪ -0.004624 1.171350] | S = 20.653566
```

```
7 | u = [3.567383 0.386452 3.032279 1.777383] | F(u) = [0.071425 37.202756
    ↪ 0.002715 0.136602] | S = 17.526996
8 | u = [3.045719 0.428871 2.583950 1.532460] | F(u) = [0.090938 13.122322
    ↪ 0.003303 0.007226] | S = 15.182001
9 | u = [2.735042 0.497981 2.316126 1.364156] | F(u) = [0.082449 4.723376 0.003070
    ↪  0.008377] | S = 13.826611
10 | u = [2.569108 0.591590 2.172423 1.272798] | F(u) = [0.055265 1.634580
    ↪ 0.002543 0.006852] | S = 13.211837
11 | u = [2.493841 0.688836 2.106835 1.230292] | F(u) = [0.025161 0.460804
    ↪ 0.001435 0.003851] | S = 13.039607
12 | u = [2.470983 0.748757 2.086766 1.216992] | F(u) = [0.005178 0.069517
    ↪ 0.000342 0.000914] | S = 13.046996
13 | u = [2.468085 0.761909 2.084199 1.215249] | F(u) = [0.000187 0.002179
    ↪ 0.000013 0.000035] | S = 13.058885
14 | u = [2.468013 0.762357 2.084136 1.215205] | F(u) = [0.000000 0.000002
    ↪ 0.000000 0.000000] | S = 13.059421
15 | u = [2.468013 0.762357 2.084136 1.215205] | F(u) = [0.000000 0.000002
    ↪ 0.000000 0.000000] | S = 13.059421

Result
u = [2.468013 0.762357 2.084136 1.215205]
F(u) = [0.000000 0.000000 0.000000 0.000000]
L(N) = [1.794934 0.865945 1.606536 1.142912]
lam(N) = [3.000000 0.662338 2.415584 1.207792]
S = 13.059421
```

Алгоритм успешно сходится за 16 итераций. С каждой итерацией алгоритма значения целевых функций стемятся к нулю, результирующая цена также постепенно уменьшается.

3.3 Вывод

Как и ожидалось, алгоритм простых итераций не сходится, поэтому пришлось использовать более надежный алгоритм Ньютона. Модифицированный алгоритм Ньютона чрезвычайно чувствителен к выбору начального приближения, поэтому был использован вариант алгоритма с расчетом матрицы Якоби. Для увеличения производительности данного алгоритма можно подключить пакет MATLAB Parallel Computing Toolbox, который использует MPI для эффективного распараллеливания.

Глава 4

Решение задачи анализа потокового графа с использованием методики GERT и алгебры потоковых графов

4.1 Постановка задачи

Вариант 25

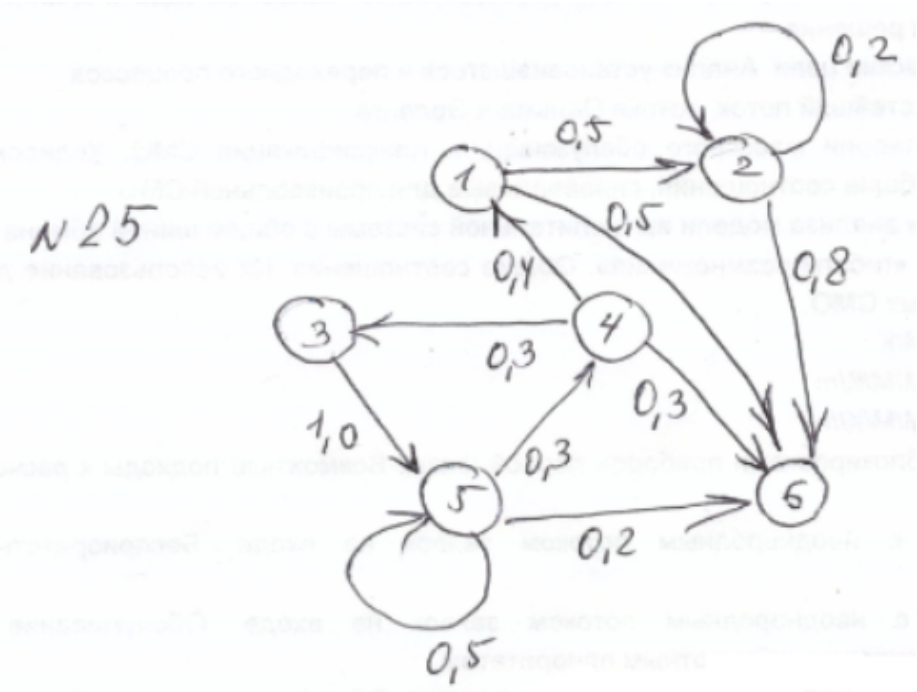

Рис. 4.1: Исходный граф системы

Каждой дуге (ij) поставлены в соответствие следующие данные:

- Закон распределения времени выполнения работы. Будем считать его нормальным.
- Параметры закона распределения (математическое ожидание M и дисперсия D).
- Вероятность P_{ij} выполнения работы, показанная на графе.

Необходимо найти:

- Вероятность выхода в завершающий узел графа (для всех вариантов узел 6).
- Математическое ожидание.
- Дисперсию времени выхода процесса в завершающий узел графа.

В отчете перечислить все петли всех порядков, обнаруженные на графе, выписать уравнение Мейсона, получить решение для $W_E(s)$ и найти требуемые параметры.

Дополнительное задание

Решить задачу используя методику анализа потокового графа, основанную на обработке матрицы передач (Branch Transmittance Matrix).

4.2 Решение

4.2.1 Построение замкнутой GERT-сети

Чтобы определить эквивалентную W-функцию для анализируемой GERT-сети, необходимо замкнуть сеть дугой, исходящей из узла 6 в узел 1:

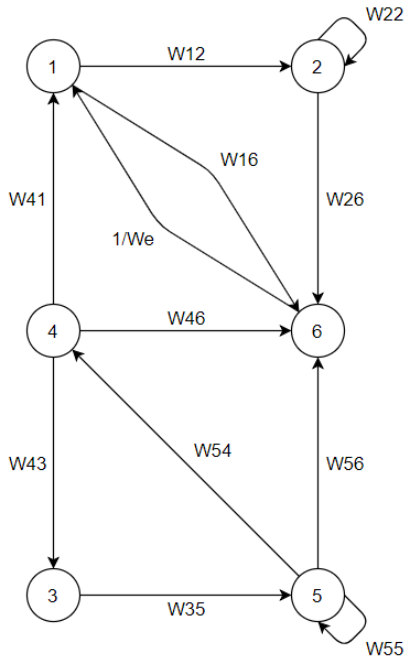

Рис. 4.2: Замкнутая GERT-сеть

4.2.2 Построение W-функции

Найдем W-функции для дуг GERT-сети:

Таблица 4.1 W-функции для дуг GERT-сети

Начало	Конец	Вероятность	Мат. ожидание	Дисперсия	W-функция
1	2	0.5	10	4	$0.5 \cdot e^{10t+8t^2}$
1	6	0.5	23	25	$0.5 \cdot e^{23t+312.5t^2}$
2	2	0.2	13	16	$0.2 \cdot e^{13t+128t^2}$
2	6	0.8	11	16	$0.8 \cdot e^{11t+128t^2}$
3	5	1	10	9	$1 \cdot e^{10t+40.5t^2}$
4	1	0.4	37	16	$0.4 \cdot e^{37t+128t^2}$
4	3	0.3	12	16	$0.3 \cdot e^{12t+128t^2}$
4	6	0.3	12	49	$0.3 \cdot e^{12t+1200.5t^2}$
5	4	0.3	15	25	$0.3 \cdot e^{15t+312.5t^2}$
5	5	0.5	19	4	$0.5 \cdot e^{19t+8t^2}$
5	6	0.2	42	9	$0.2 \cdot e^{42t+40.5t^2}$

4.2.3 Построение уравнения Мейсона

Найдем все петли первого порядка:

$W_{12} \cdot W_{26} \cdot \frac{1}{W_E}$

$W_{16} \cdot \frac{1}{W_E}$

$W_{35} \cdot W_{54} \cdot W_{43}$

W_{22}

W_{55}

Найдем все петли второго порядка:

$W_{16} \cdot W_{22} \cdot \frac{1}{W_E}$

$W_{16} \cdot W_{35} \cdot W_{54} \cdot W_{43} \cdot \frac{1}{W_E}$

$W_{16} \cdot W_{55} \cdot \frac{1}{W_E}$

$W_{12} \cdot W_{26} \cdot W_{55} \cdot \frac{1}{W_E}$

$W_{12} \cdot W_{26} \cdot W_{35} \cdot W_{54} \cdot W_{43} \cdot \frac{1}{W_E}$

$W_{22} \cdot W_{55}$

$W_{22} \cdot W_{35} \cdot W_{54} \cdot W_{43}$

Найдем все петли третьего порядка:

$W_{16} \cdot W_{22} \cdot W_{35} \cdot W_{54} \cdot W_{43} \cdot \frac{1}{W_E}$

$W_{16} \cdot W_{22} \cdot W_{55} \cdot \frac{1}{W_E}$

Таким образом уравнение Мейсона будет иметь следующий вид:

$H = 1 - (W_{12}W_{26}\frac{1}{W_E} + W_{16}\frac{1}{W_E} + W_{35}W_{54}W_{43} + W_{22} + W_{55}) + (W_{16}W_{22}\frac{1}{W_E} + W_{16}W_{35}W_{54}W_{43}\frac{1}{W_E} + W_{16}W_{55}\frac{1}{W_E} + W_{12}W_{26}W_{55}\frac{1}{W_E} + W_{12}W_{26}W_{35}W_{54}W_{43}\frac{1}{W_E} + W_{22}W_{55} + W_{22}W_{35}W_{54}W_{43}) - (W_{16}W_{22}W_{35}W_{54}W_{43}\frac{1}{W_E} + W_{16}W_{22}W_{55}\frac{1}{W_E})$

В результате эквивалентная W-функция равняется:

$W_E(s) = \frac{W_{12}W_{26} + W_{16} - W_{16}W_{22} - W_{16}W_{35}W_{54}W_{43} - W_{16}W_{55} - W_{12}W_{26}W_{55} - W_{12}W_{26}W_{35}W_{54}W_{43} + W_{16}W_{22}W_{35}W_{54}W_{43} + W_{16}W_{22}W_{55}}{1 - W_{35}W_{54}W_{43} - W_{22} - W_{55} + W_{22}W_{55} + W_{22}W_{35}W_{54}W_{43}}$

4.2.4 Расчет статистических значений

Расчет математического ожидания (μ_{1E}) и дисперсии (σ_E) производится по следующим образом:

$W_E(s) = p_E \cdot M_E(s), p_E = W_E(0) \Longrightarrow M_E(s) = \frac{W_E(s)}{W_E(0)}$

$\mu_{1E} = \frac{dM_E(s)}{ds}|s = 0$

$\mu_{2E} = \frac{d^2M_E(s)}{ds^2}|s = 0$

$\sigma_E = \mu_{2E} - \mu_{1E}^2$

Разработаем скрипт для расчета статистических значений в среде MATLAB (Приложение 12).

Результат вычисления статистических значений:

```
We(0) = 1.000
me1 = 23.625
me2 = 1065.438
de = 507.297
```

4.2.5 Дополнительное задание

Для данного метода W-функция W_e расчитывается по формуле:

$M_{1n} = \frac{1}{w_{n1}} = -\frac{\frac{\delta det(\tilde{A})}{\delta w_{n1}}}{det(\tilde{A}|_{w_{n1}=0})}$, где

$n = 6, \tilde{A} = I_n - \tilde{Q}^T, A = I - Q^T, q_{ij}(s) = p_{ij}m_{ij}(s)$

Разработаем скрипт для расчета статистических значений при помощи анализа потокового графа в среде MATLAB (Приложение 12).

Результат вычисления статистических значений:

```
We(0) = -1.000
me1 = 23.625
me2 = 1065.438
de = 507.297
```

Результаты идентичны предыдущему решению, за исключением $W_e(0)$, что объясняется результирующей функцией взятой с обратным знаком.

4.3 Вывод

В ходе данной лабораторной работы были получены навыки работы с вероятностными графами и их обработка с помощью методики GERT. При заданных значениях вероятности, мат. ожидания и дисперсии для каждой дуги исходного графа достаточно легко расчитываются W-функции, которые необходимы для построения формулы Мейсона. После этого из формулы Мейсона по формулам математической статистики достаточно легко расчитывается результирующее мат. ожидание и дисперсия.

Решение путем анализа потокового графа показало аналогичные результаты, что подтверждает корректность решения. Однако, метод анализа потокового графа выполняется заметно медленнее, даже на небольшом графе.

Заключение

В работе были рассмотрены различные математические модели для решения задачи выбора оптимального решения.

При анализе результатов решения многокритериальной задачи можно заметить, что аддитивная и мультипликативная свертка выдают одинаковый, наиболее оптимальный результат. Наилучший результат этих методов обусловлен наличием коэффициентов значимости. Методы, не подразумевающие введение весовых коэффициентов показывают похожий результат, который в целом хуже, чем у аддитивной и мультипликативной свертки.

В процессе поиска оптимальной стратегии принятия решений с использованием марковских моделей были определены оптимальные стратегии для конкретной задачи. Линейное программирование позволяет достаточно легко и быстро находить оптимальную стратегию, однако требует некоторых предварительных преобразований и формализаций перед использованием.

Как и ожидалось, при оптимизации многоканальной замкнутой CCMO алгоритм простых итераций не сходится, поэтому пришлось использовать более надежный алгоритм Ньютона. Модифицированный алгоритм Ньютона чрезвычайно чувствителен к выбору начального приближения, поэтому был использован вариант алгоритма с расчетом матрицы Якоби. Для увеличения производительности данного алгоритма можно подключить пакет MATLAB Parallel Computing Toolbox, который использует MPI для эффективного распараллеливания.

В результате решения задачи анализа потокового графа можно сделать вывод, что при заданных значениях вероятности, мат. ожидания и дисперсии для каждой дуги исходного графа достаточно легко расчитываются W-функции, которые необходимы для построения формулы Мейсона. После этого из формулы Мейсона по формулам математической статистики достаточно легко расчитывается результирующее мат. ожидание и дисперсия. Решение путем анализа потокового графа показало аналогичные результаты, что подтверждает корректность решения. Однако, метод анализа потокового графа выполняется заметно медленнее, даже на небольшом графе.

Среда MATLAB действительно хорошо подходит для решения оптимизационных задач засчет множества специфичных математических библиотек и пакетов.

Список использованных источников

1. Д. Филлипс, А. Гарсиа-Диас. Методы анализа сетей: Пер. с англ. — М.: Мир, 1984.—496 с, ил.
2. Ren, Yu. The methodology of flowgraph models. PhD thesis, The London School of Economics and Political Science (LSE). — 2011.
3. Сиднев А. Г. Системный анализ. Часть 1 [Электронный ресурс] // Интранет- портал ИИТУ СПбПУ. 2018. URL: http://intranet.ftk.spbstu.ru/docinfo.php?InfoFtkDocumentID=1386947 (дата обращения: 22.04.2018)
4. Горбунов В.М. Теория принятия решений: учебное пособие. Томск: Изд-во Томск. политех. ун-та, 2010.
5. Системный анализ и принятие решений: учебное пособие / Е.Н. Бендерская, Д.Н. Колесников, В.И. Пахомова и др.; Под ред. Д.Н. Колесникова. 2-е изд. СПб.: Изд-во СПбГПУ, 2001.
6. Optimization Toolbox: описание функции FGOALATTAIN [Электронный ре- сурс] // MATLAB.Exponenta, центр компетенций MathWorks. URL: http://matlab.exponenta.ru/optimiz/book_4/2/fgoalattain.php (дата обращения: 17.02.2018).
7. Таха Х. А. Введение в исследование операций: Пер. с англ. 7-е изд. М.: Изда-тельский дом «Вильямс», 2005.

Приложение 1

Решение задачи методом линейного программирования

```
clear all;
close all;
clc;
format long g;

% -F1 -> min
mF1 = @(X) -(60 * X(1) + 300 * X(2) + 2000 * X(3));
% -F2 -> min
mF2 = @(X) -((0.1 * X(1) + 10 * X(2) + 70 * sqrt(X(3))) ^ 1.5);
% F3 -> min
pF3 = @(X) 30 * X(1) + 100 * X(2) + 220 * X(3);
% F4 -> min
pF4 = @(X) log(20 * X(1) + 3 * X(2) + 0.01 * X(3));
% -F5 -> min
mF5 = @(X) -((-mF1(X)) / (X(1) + X(2) + X(3)));
% -F6 -> min
mF6 = @(X) -(((-mF1(X)) + (-mF2(X))) - (pF3(X) + pF4(X)));

A = [1, 0, 0;
-1, 0, 0;
0, 1, 0;
0, -1, 0;
0, 0, 1;
0, 0, -1;
1, 1, 1];

B = [70;
-35;
30;
-15;
1;
-0.5;
80];

Aeq = [];
Beq = [];

S = [35; 15; 0.5];

[x1, result1] = fmincon(mF1, S, A, B, Aeq, Beq);
[x2, result2] = fmincon(mF2, S, A, B, Aeq, Beq);
[x3, result3] = fmincon(pF3, S, A, B, Aeq, Beq);
[x4, result4] = fmincon(pF4, S, A, B, Aeq, Beq);
[x5, result5] = fmincon(mF5, S, A, B, Aeq, Beq);
[x6, result6] = fmincon(mF6, S, A, B, Aeq, Beq);

formatter = '-- F1 max --\n%s\n%s\n\n-- F2 max --\n%s\n%s\n\n-- F3 min --\n%s\
    ↪ n%s\n\n-- F4 min --\n%s\n%s\n\n-- F5 max --\n%s\n%s\n\n-- F6 max --\n%s\
    ↪ n%s\n\n';
s1 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', x1, sum(
    ↪ x1));
s2 = sprintf('!F1 = %.3f, F2 = %.3f, F3 = %.3f, F4 = %.3f, F5 = %.3f, F6 = %.3
    ↪ f', -result1, -mF2(x1), pF3(x1), pF4(x1), -mF5(x1), -mF6(x1));
s3 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', x2, sum(
    ↪ x2));
s4 = sprintf('F1 = %.3f, !F2 = %.3f, F3 = %.3f, F4 = %.3f, F5 = %.3f, F6 = %.3
```

```
    ↪ f', -mF1(x2), -result2, pF3(x2), pF4(x2), -mF5(x2), -mF6(x2));
s5 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', x3, sum(
    ↪ x3));
s6 = sprintf('F1 = %.3f, F2 = %.3f, !F3 = %.3f, F4 = %.3f, F5 = %.3f, F6 = %.3
    ↪ f', -mF1(x3), -mF2(x3), result3, pF4(x3), -mF5(x3), -mF6(x3));
s7 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', x4, sum(
    ↪ x4));
s8 = sprintf('F1 = %.3f, F2 = %.3f, F3 = %.3f, !F4 = %.3f, F5 = %.3f, F6 = %.3
    ↪ f', -mF1(x4), -mF2(x4), pF3(x4), result4, -mF5(x4), -mF6(x4));
s9 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', x5, sum(
    ↪ x5));
s10 = sprintf('F1 = %.3f, F2 = %.3f, F3 = %.3f, F4 = %.3f, !F5 = %.3f, F6 =
    ↪ %.3f', -mF1(x5), -mF2(x5), pF3(x5), pF4(x5), -result5, -mF6(x5));
s11 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', x6, sum(
    ↪ x6));
s12 = sprintf('F1 = %.3f, F2 = %.3f, F3 = %.3f, F4 = %.3f, F5 = %.3f, !F6 =
    ↪ %.3f', -mF1(x6), -mF2(x6), pF3(x6), pF4(x6), -mF5(x6), -result6);
fprintf(formatter, s1, s2, s3, s4, s5, s6, s7, s8, s9, s10, s11, s12);
```

Листинг 4.1: Решение задачи методом линейного программирования

Приложение 2

Решение задачи при помощи аддитивной свертки

```
clear all;
close all;
clc;
format long g;

% -F1 -> min
mF1 = @(X) -(60 * X(1) + 300 * X(2) + 2000 * X(3));
% -F2 -> min
mF2 = @(X) -((0.1 * X(1) + 10 * X(2) + 70 * sqrt(X(3))) ^ 1.5);
% F3 -> min
pF3 = @(X) 30 * X(1) + 100 * X(2) + 220 * X(3);
% F4 -> min
pF4 = @(X) log(20 * X(1) + 3 * X(2) + 0.01 * X(3));
% -F5 -> min
mF5 = @(X) -((-mF1(X)) / (X(1) + X(2) + X(3)));
% -F6 -> min
mF6 = @(X) -(((-mF1(X)) + (-mF2(X))) - (pF3(X) + pF4(X)));

cF2 = 0.15;
rF2 = 7258.939;
cF3 = 0.15;
rF3 = 2660;
cF4 = 0.05;
rF4 = 6.613;
cF5 = 0.05;
rF5 = 198.485;
cF6 = 0.6;
rF6 = 16501.964;

pFS = @(X) cF2 * (mF2(X) / rF2) + cF3 * (pF3(X) / rF3) + cF4 * (pF4(X) / rF4)
    ↪ + cF5 * (mF5(X) / rF5) + cF6 * (mF6(X) / rF6);

A = [1, 0, 0;
```

```
-1, 0, 0;
0, 1, 0;
0, -1, 0;
0, 0, 1;
0, 0, -1;
1, 1, 1];

B = [70;
-35;
30;
-15;
1;
-0.5;
80];

Aeq = [];
Beq = [];

S = [35; 15; 0.5];

[xS, resultS] = fmincon(pFS, S, A, B, Aeq, Beq);

formatter = '-- FS --\n%s\n%s\n%s\n%s\n\n';
s1 = sprintf('FS = %.3f', resultS);
s2 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', xS, sum(
    ↪ xS));
s3 = sprintf('F2 = %.3f, F3 = %.3f, F4 = %.3f, F5 = %.3f, F6 = %.3f', -mF2(xS)
    ↪ , pF3(xS), pF4(xS), -mF5(xS), -mF6(xS));
s4 = sprintf('F2/rF2 = %.3f%%, F3/rF3 = %.3f%%, F4/rF4 = %.3f%%, F5/rF5 = %.3f
    ↪ %%, F6/rF6 = %.3f%%', -mF2(xS) / rF2 * 100, pF3(xS) / rF3 * 100, pF4(xS)
    ↪  / rF4 * 100, -mF5(xS) / rF5 * 100, -mF6(xS) / rF6 * 100);
fprintf(formatter, s1, s2, s3, s4);
```

Листинг 4.2: Решение задачи при помощи аддитивной свертки

Приложение 3

Решение задачи при помощи мультипликативной свертки

```
clear all;
close all;
clc;
format long g;

% 1 / F1 -> min
mF1 = @(X) 1 / (60 * X(1) + 300 * X(2) + 2000 * X(3));
% 1 / F2 -> min
mF2 = @(X) 1 / ((0.1 * X(1) + 10 * X(2) + 70 * sqrt(X(3))) ^ 1.5);
% F3 -> min
pF3 = @(X) 30 * X(1) + 100 * X(2) + 220 * X(3);
% F4 -> min
pF4 = @(X) log(20 * X(1) + 3 * X(2) + 0.01 * X(3));
% 1 / F5 -> min
mF5 = @(X) 1 / ((1 / mF1(X)) / (X(1) + X(2) + X(3)));
% 1 / F6 -> min
mF6 = @(X) 1 / (((1 / mF1(X)) + (1 / mF2(X))) - (pF3(X) + pF4(X)));

cF2 = 0.15;
```

```
rF2 = 7258.939;
cF3 = 0.15;
rF3 = 2660;
cF4 = 0.05;
rF4 = 6.613;
cF5 = 0.05;
rF5 = 198.485;
cF6 = 0.6;
rF6 = 16501.964;

pFS = @(X) nthroot(mF2(X) * rF2, 1 / cF2) * nthroot(pF3(X) / rF3, 1 / cF3) *
   ↪ nthroot(pF4(X) / rF4, 1 / cF4) * nthroot(mF5(X) * rF5, 1 / cF5) *
   ↪ nthroot(mF6(X) * rF6, 1 / cF6);

A = [1, 0, 0;
-1, 0, 0;
0, 1, 0;
0, -1, 0;
0, 0, 1;
0, 0, -1;
1, 1, 1];

B = [70;
-35;
30;
-15;
1;
-0.5;
80];

Aeq = [];
Beq = [];

S = [35; 15; 0.5];

[xS, resultS] = fmincon(pFS, S, A, B, Aeq, Beq);

formatter = '-- FS --\n%s\n%s\n%s\n%s\n\n';
s1 = sprintf('FS = %.3f', resultS);
s2 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', xS, sum(
   ↪ xS));
s3 = sprintf('F2 = %.3f, F3 = %.3f, F4 = %.3f, F5 = %.3f, F6 = %.3f', 1 / mF2(
   ↪ xS), pF3(xS), pF4(xS), 1 / mF5(xS), 1 / mF6(xS));
s4 = sprintf('F2/rF2 = %.3f%%, F3/rF3 = %.3f%%, F4/rF4 = %.3f%%, F5/rF5 = %.3f
   ↪ %%, F6/rF6 = %.3f%%', (1 / mF2(xS)) / rF2 * 100, pF3(xS) / rF3 * 100,
   ↪ pF4(xS) / rF4 * 100, (1 / mF5(xS)) / rF5 * 100, (1 / mF6(xS)) / rF6 *
   ↪ 100);
fprintf(formatter, s1, s2, s3, s4);
```

Листинг 4.3: Решение задачи при помощи мультипликативной свертки

Приложение 4

Решение задачи при помощи функции fminimax

```
clear all;
close all;
clc;
```

```
format long g;

% 1 / F1 -> min
mF1 = @(X) 1 / (60 * X(1) + 300 * X(2) + 2000 * X(3));
% 1 / F2 -> min
mF2 = @(X) 1 / ((0.1 * X(1) + 10 * X(2) + 70 * sqrt(X(3))) ^ 1.5);
% F3 -> min
pF3 = @(X) 30 * X(1) + 100 * X(2) + 220 * X(3);
% F4 -> min
pF4 = @(X) log(20 * X(1) + 3 * X(2) + 0.01 * X(3));
% 1 / F5 -> min
mF5 = @(X) 1 / ((1 / mF1(X)) / (X(1) + X(2) + X(3)));
% 1 / F6 -> min
mF6 = @(X) 1 / (((1 / mF1(X)) + (1 / mF2(X))) - (pF3(X) + pF4(X)));

rF1 = 13940;
rF2 = 7258.939;
rF3 = 2660;
rF4 = 6.613;
rF5 = 198.485;
rF6 = 16501.964;

A = [1, 0, 0;
-1, 0, 0;
0, 1, 0;
0, -1, 0;
0, 0, 1;
0, 0, -1;
1, 1, 1];

B = [70;
-35;
30;
-15;
1;
-0.5;
80];

Aeq = [];
Beq = [];

S = [35; 15; 0.5];

[xS, resultS] = fminimax(@m4f, S, A, B, Aeq, Beq);

formatter = '-- FS --\n%s\n%s\n%s\n%s\n\n';
s1 = sprintf('FS = (%.3f, %.3f, %.3f, %.3f, %.3f)', resultS(2:6));
s2 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', xS, sum(
    ↪ xS));
s3 = sprintf('F2 = %.3f, F3 = %.3f, F4 = %.3f, F5 = %.3f, F6 = %.3f', 1 / mF2(
    ↪ xS), pF3(xS), pF4(xS), 1 / mF5(xS), 1 / mF6(xS));
s4 = sprintf('F2/rF2 = %.3f%%, F3/rF3 = %.3f%%, F4/rF4 = %.3f%%, F5/rF5 = %.3f
    ↪ %%, F6/rF6 = %.3f%%', (1 / mF2(xS)) / rF2 * 100, pF3(xS) / rF3 * 100,
    ↪ pF4(xS) / rF4 * 100, (1 / mF5(xS)) / rF5 * 100, (1 / mF6(xS)) / rF6 *
    ↪ 100);
fprintf(formatter, s1, s2, s3, s4);
```

Листинг 4.4: Решение задачи при помощи функции fminimax

```
function result = m4f(X)
    % 1 / F1 -> min
```

```
    mF1 = 1 / (60 * X(1) + 300 * X(2) + 2000 * X(3));
    % 1 / F2 -> min
    mF2 = 1 / ((0.1 * X(1) + 10 * X(2) + 70 * sqrt(X(3))) ^ 1.5);
    % F3 -> min
    pF3 = 30 * X(1) + 100 * X(2) + 220 * X(3);
    % F4 -> min
    pF4 = log(20 * X(1) + 3 * X(2) + 0.01 * X(3));
    % 1 / F5 -> min
    mF5 = 1 / ((1 / mF1) / (X(1) + X(2) + X(3)));
    % 1 / F6 -> min
    mF6 = 1 / (((1 / mF1) + (1 / mF2)) - (pF3 + pF4));

    rF1 = 13940;
    rF2 = 7258.939;
    rF3 = 2660;
    rF4 = 6.613;
    rF5 = 198.485;
    rF6 = 16501.964;

    result(1) = mF1 * rF1;
    result(2) = mF2 * rF2;
    result(3) = pF3 / rF3;
    result(4) = pF4 / rF4;
    result(5) = mF5 * rF5;
    result(6) = mF6 * rF6;
end
```

Листинг 4.5: Внутренняя функция для fminimax

Приложение 5

Решение задачи при помощи метода последовательных уступок

```
clear all;
close all;
clc;
format long g;

% -F7 -> min
mF7 = @(X) -(30 * X(1) + 200 * X(2) + 1780 * X(3));
% -F8 -> min
mF8 = @(X) -(X(1) + 10 * X(3));
% F9 -> min
pF9 = @(X) X(1) + 2 * X(2);

A = [1, 0, 0;
-1, 0, 0;
0, 1, 0;
0, -1, 0;
0, 0, 1;
0, 0, -1;
1, 1, 1];

B = [70;
-35;
30;
-15;
1;
```

```
-0.5;
80];

Aeq = [];
Beq = [];

S = [35; 15; 0.5];

[x7, result7] = fmincon(mF7, S, A, B, Aeq, Beq);

formatter = '-- F7 max --\n%s\n%s\n\n';
s1 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', x7, sum(
    ↪ x7));
s2 = sprintf('!F7 = %.3f, F8 = %.3f, F9 = %.3f', -result7, -mF8(x7), pF9(x7));
fprintf(formatter, s1, s2);
```

Листинг 4.6: Решение задачи для критерия F7

```
clear all;
close all;
clc;
format long g;

% -F7 -> min
mF7 = @(X) -(30 * X(1) + 200 * X(2) + 1780 * X(3));
% -F8 -> min
mF8 = @(X) -(X(1) + 10 * X(3));
% F9 -> min
pF9 = @(X) X(1) + 2 * X(2);

K = 0.85

rF7 = 9250;

A = [1, 0, 0;
-1, 0, 0;
0, 1, 0;
0, -1, 0;
0, 0, 1;
0, 0, -1;
1, 1, 1;
-30, -200, -1780];

B = [70;
-35;
30;
-15;
1;
-0.5;
80;
-rF7 * K];

Aeq = [];
Beq = [];

S = [35; 15; 0.5];

[x8, result8] = fmincon(mF8, S, A, B, Aeq, Beq);

fprintf('%.3f >= F7 >= %.1f * %.3f\n', rF7, K, rF7);
fprintf('%.3f >= F7 >= %.3f\n\n', rF7, rF7 * K);
```

```
formatter = '-- F8 max --\n%s\n%s\n\n';
s1 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', x8, sum(
    ↪ x8));
s2 = sprintf('F7 = %.3f, !F8 = %.3f, F9 = %.3f', -mF7(x8), -result8, pF9(x8));
fprintf(formatter, s1, s2);
```

Листинг 4.7: Решение задачи для критерия F8

```
clear all;
close all;
clc;
format long g;

% -F7 -> min
mF7 = @(X) -(30 * X(1) + 200 * X(2) + 1780 * X(3));
% -F8 -> min
mF8 = @(X) -(X(1) + 10 * X(3));
% F9 -> min
pF9 = @(X) X(1) + 2 * X(2);

K = 0.85

rF7 = 9250;
rF8 = 67.162;

A = [1, 0, 0;
-1, 0, 0;
0, 1, 0;
0, -1, 0;
0, 0, 1;
0, 0, -1;
1, 1, 1;
-30, -200, -1780;
-1, 0, -10];

B = [70;
-35;
30;
-15;
1;
-0.5;
80;
-rF7 * K;
-rF8 * K];

Aeq = [];
Beq = [];

S = [35; 15; 0.5];

[x9, result9] = fmincon(pF9, S, A, B, Aeq, Beq);

fprintf('%.3f >= F7 >= %.2f * %.3f\n', rF7, K, rF7);
fprintf('%.3f >= F7 >= %.3f\n\n', rF7, rF7 * K);

fprintf('%.3f >= F8 >= %.2f * %.3f\n', rF8, K, rF8);
fprintf('%.3f >= F8 >= %.3f\n\n', rF8, rF8 * K);

formatter = '-- F9 max --\n%s\n%s\n\n';
s1 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', x9, sum(
```

```
   ↪ x9));
s2 = sprintf('F7 = %.3f, F8 = %.3f, !F9 = %.3f', -mF7(x9), -mF8(x9), result9);
fprintf(formatter, s1, s2);
```

Листинг 4.8: Решение задачи для критерия F9

Приложение 6

Решение задачи при помощи метода достижения цели (fgoalattain)

```
clear all;
close all;
clc;
format long g;

% -F1 -> min
mF1 = @(X) -(60 * X(1) + 300 * X(2) + 2000 * X(3));
% -F2 -> min
mF2 = @(X) -((0.1 * X(1) + 10 * X(2) + 70 * sqrt(X(3))) ^ 1.5);
% F3 -> min
pF3 = @(X) 30 * X(1) + 100 * X(2) + 220 * X(3);
% F4 -> min
pF4 = @(X) log(20 * X(1) + 3 * X(2) + 0.01 * X(3));
% -F5 -> min
mF5 = @(X) -((-mF1(X)) / (X(1) + X(2) + X(3)));
% -F6 -> min
mF6 = @(X) -(((-mF1(X)) + (-mF2(X))) - (pF3(X) + pF4(X)));

rF1 = -13940;
rF2 = -7258.939;
rF3 = 2660;
rF4 = 6.613;
rF5 = -198.485;
rF6 = -16501.964;

pFS = @(X) [mF2(X), pF3(X), pF4(X), mF5(X), mF6(X)];

G = [rF2, rF3, rF4, rF5, rF6];

W = abs(G);

A = [1, 0, 0;
-1, 0, 0;
0, 1, 0;
0, -1, 0;
0, 0, 1;
0, 0, -1;
1, 1, 1];

B = [70;
-35;
30;
-15;
1;
-0.5;
80];

Aeq = [];
```

```
Beq = [];

S = [35; 15; 0.5];

[xS, resultS] = fgoalattain(pFS, S, G, W, A, B, Aeq, Beq);

formatter = '-- FS --\n%s\n%s\n%s\n%s\n\n';
s1 = sprintf('FS = (%.3f, %.3f, %.3f, %.3f, %.3f)', resultS);
s2 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', xS, sum(
    ↪ xS));
s3 = sprintf('F2 = %.3f, F3 = %.3f, F4 = %.3f, F5 = %.3f, F6 = %.3f', -mF2(xS)
    ↪ , pF3(xS), pF4(xS), -mF5(xS), -mF6(xS));
s4 = sprintf('F2/rF2 = %.3f%%, F3/rF3 = %.3f%%, F4/rF4 = %.3f%%, F5/rF5 = %.3f
    ↪ %%, F6/rF6 = %.3f%%', mF2(xS) / rF2 * 100, pF3(xS) / rF3 * 100, pF4(xS)
    ↪ / rF4 * 100, mF5(xS) / rF5 * 100, mF6(xS) / rF6 * 100);
fprintf(formatter, s1, s2, s3, s4);
```

Листинг 4.9: Решение задачи при помощи метода достижения цели (fgoalattain)

Приложение 7

Решение задачи при помощи введения метрики в пространстве критериев

```
clear all;
close all;
clc;
format long g;

% -F1 -> min
mF1 = @(X) -(60 * X(1) + 300 * X(2) + 2000 * X(3));
% -F2 -> min
mF2 = @(X) -((0.1 * X(1) + 10 * X(2) + 70 * sqrt(X(3))) ^ 1.5);
% F3 -> min
pF3 = @(X) 30 * X(1) + 100 * X(2) + 220 * X(3);
% F4 -> min
pF4 = @(X) log(20 * X(1) + 3 * X(2) + 0.01 * X(3));
% -F5 -> min
mF5 = @(X) -((-mF1(X)) / (X(1) + X(2) + X(3)));
% -F6 -> min
mF6 = @(X) -(((-mF1(X)) + (-mF2(X))) - (pF3(X) + pF4(X)));

rF1 = -13940;
rF2 = -7258.939;
rF3 = 2660;
rF4 = 6.613;
rF5 = -198.485;
rF6 = -16501.964;

pFS = @(X) (1 - mF2(X) / rF2) ^ 2 + (1 - pF3(X) / rF3) ^ 2 + (1 - pF4(X) / rF4
    ↪ ) ^ 2 + (1 - mF5(X) / rF5) ^ 2 + (1 - mF6(X) / rF6) ^ 2;

A = [1, 0, 0;
-1, 0, 0;
0, 1, 0;
0, -1, 0;
0, 0, 1;
0, 0, -1;
1, 1, 1];
```

```
B = [70;
-35;
30;
-15;
1;
-0.5;
80];

Aeq = [];
Beq = [];

S = [35; 15; 0.5];

[xS, resultS] = fmincon(pFS, S, A, B, Aeq, Beq);

formatter = '-- FS --\n%s\n%s\n%s\n%s\n\n';
s1 = sprintf('FS = %.3f', resultS);
s2 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', xS, sum(
    ↪ xS));
s3 = sprintf('F2 = %.3f, F3 = %.3f, F4 = %.3f, F5 = %.3f, F6 = %.3f', -mF2(xS)
    ↪ , pF3(xS), pF4(xS), -mF5(xS), -mF6(xS));
s4 = sprintf('F2/rF2 = %.3f%%, F3/rF3 = %.3f%%, F4/rF4 = %.3f%%, F5/rF5 = %.3f
    ↪ %%, F6/rF6 = %.3f%%', mF2(xS) / rF2 * 100, pF3(xS) / rF3 * 100, pF4(xS)
    ↪ / rF4 * 100, mF5(xS) / rF5 * 100, mF6(xS) / rF6 * 100);
fprintf(formatter, s1, s2, s3, s4);
```

Листинг 4.10: Решение задачи при помощи введения метрики в пространстве критериев

Приложение 8

Решение задачи стохастического программирования

```
clear all;
close all;
clc;
format long g;

% -F1 -> min
mF1 = @(X) -(60 * X(1) + 300 * X(2) + 2000 * X(3));
% -F2 -> min
mF2 = @(X) -((0.1 * X(1) + 10 * X(2) + 70 * sqrt(X(3))) ^ 1.5);
% F3 -> min
pF3 = @(X) 30 * X(1) + 100 * X(2) + 220 * X(3);
% F4 -> min
pF4 = @(X) log(20 * X(1) + 3 * X(2) + 0.01 * X(3));
% -F5 -> min
mF5 = @(X) -((-mF1(X)) / (X(1) + X(2) + X(3)));
% -F6 -> min
mF6 = @(X) -(((-mF1(X)) + (-mF2(X))) - (pF3(X) + pF4(X)));

rF1 = -13940;
rF2 = -7258.939;
rF3 = 2660;
rF4 = 6.613;
rF5 = -198.485;
rF6 = -16501.964;
```

```
pFS = @(X) (1 - mF2(X) / rF2) ^ 2 + (1 - pF3(X) / rF3) ^ 2 + (1 - pF4(X) / rF4
    ↪ ) ^ 2 + (1 - mF5(X) / rF5) ^ 2 + (1 - mF6(X) / rF6) ^ 2;

A = [1, 0, 0;
-1, 0, 0;
0, 1, 0;
0, -1, 0;
0, 0, 1;
0, 0, -1;
1, 1, 1];

B = [70;
-35;
30;
-15;
1;
-0.5;
80];

Aeq = [];
Beq = [];

lb = [];
ub = [];

S = [35; 15; 0.5];

O = optimoptions('fmincon', 'Display', 'none');

[xS, resultS] = fmincon(pFS, S, A, B, Aeq, Beq, lb, ub, [], O);

global K;

Ka = icdf('Normal', 0.1 : 0.1 : 0.9, 0, 1);
sizeK = size(Ka, 2);
xV = zeros(3, sizeK);
rV = zeros(3, sizeK);
for i = 1 : 1 : sizeK
K = Ka(i);
fmincon(pFS, S, A, B, Aeq, Beq);

[xV(:, i), rV(:, i)] = fmincon(pFS, S, A(1:6, :), B(1:6, :), Aeq, Beq, lb, ub,
    ↪  @m8f, O);
end

mean = (abs(mF2(xS) / rF2 - 1) + abs(pF3(xS) / rF3 - 1) + abs(pF4(xS) / rF4 -
    ↪ 1) + abs(mF5(xS) / rF5 - 1) + abs(mF6(xS) / rF6 - 1)) / 5 * 100;

formatter = '-- FS --\n%s\n%s\n%s\n%s\n%s\n\n';
s1 = sprintf('FS = %.3f', resultS);
s2 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', xS, sum(
    ↪ xS));
s3 = sprintf('F2 = %.3f, F3 = %.3f, F4 = %.3f, F5 = %.3f, F6 = %.3f', -mF2(xS)
    ↪ , pF3(xS), pF4(xS), -mF5(xS), -mF6(xS));
s4 = sprintf('F2/rF2 = %.3f%%, F3/rF3 = %.3f%%, F4/rF4 = %.3f%%, F5/rF5 = %.3f
    ↪ %%, F6/rF6 = %.3f%%', mF2(xS) / rF2 * 100, pF3(xS) / rF3 * 100, pF4(xS)
    ↪ / rF4 * 100, mF5(xS) / rF5 * 100, mF6(xS) / rF6 * 100);
s5 = sprintf('Mean = %.3f%%', mean);
fprintf(formatter, s1, s2, s3, s4, s5);
```

```
for i = 1 : 1 : sizeK
mean = (abs(mF2(xV(:, i)) / rF2 - 1) + abs(pF3(xV(:, i)) / rF3 - 1) + abs(pF4(
    ↪ xV(:, i)) / rF4 - 1) + abs(mF5(xV(:, i)) / rF5 - 1) + abs(mF6(xV(:, i))
    ↪ / rF6 - 1)) / 5 * 100;

formatter = '-- %s --\n%s\n%s\n%s\n%s\n\n';
s1 = sprintf('a = 0.%d, Ka = %.3f', i, Ka(i));
s2 = sprintf('x1 = %.3f, x2 = %.3f, x3 = %.3f, x1 + x2 + x3 = %.3f', xV(:, i),
    ↪  sum(xV(:, i)));
s3 = sprintf('F2 = %.3f, F3 = %.3f, F4 = %.3f, F5 = %.3f, F6 = %.3f', -mF2(xV
    ↪ (:, i)), pF3(xV(:, i)), pF4(xV(:, i)), -mF5(xV(:, i)), -mF6(xV(:, i)));
s4 = sprintf('F2/rF2 = %.3f%%, F3/rF3 = %.3f%%, F4/rF4 = %.3f%%, F5/rF5 = %.3f
    ↪ %%, F6/rF6 = %.3f%%', mF2(xV(:, i)) / rF2 * 100, pF3(xV(:, i)) / rF3 *
    ↪ 100, pF4(xV(:, i)) / rF4 * 100, mF5(xV(:, i)) / rF5 * 100, mF6(xV(:, i))
    ↪  / rF6 * 100);
s5 = sprintf('Mean = %.3f%%', mean);
fprintf(formatter, s1, s2, s3, s4, s5);
end
```

Листинг 4.11: Решение задачи стохастического программирования

```
function [c, ceq] = m8f(X)
    global K;

    M1 = 1;
    M2 = 1;
    M3 = 1;
    D1 = M1 / 2;
    D2 = M2 / 2;
    D3 = M3 / 2;

    c = M1 * X(1) + M2 * X(2) + M3 * X(3) + K * sqrt(D1 * (X(1) ^ 2) + D2 * (X
    ↪ (2) ^ 2) + D3 * (X(3) ^ 2)) - 80;
    ceq = [];
end
```

Листинг 4.12: Внутренняя функция для решения задачи стохастического программирования

Приложение 9

Поиск оптимальной стратегии при помощи линейного программирования

```
clear all;
close all;
clc;
format long g;

f = [0 12.6 8.4 14.7 12.3];

A=[];
b=[];

Aeq = [0.4 -0.9 -0.7 -0.6 -0.5;
-0.3 0.9 0.8 -0.3 -0.4;
-0.1 0 -0.1 0.9 0.9;
1 1 1 1 1];
Beq = [0; 0; 0; 1];
```

```
lb = zeros(5, 1);
ub = ones(5, 1);

[w, opt] = linprog(f, A, b, Aeq, Beq, lb, ub)
```

Листинг 4.13: Поиск оптимальной стратегии при помощи линейного программирования

Приложение 10

Решение СМО методом простых итераций

```
clear all;
close all;
clc;
format long g;

p =   [0 0.1 0.6 0.3;
0.7 0 0.2 0.1;
0.9 0 0 0.1;
0.3 0.3 0.4 0];

M = 4;
N = 5;

lam_1_ = 3;

c = [2 2 2 2];
a = [1 1 1 1];

u = [1 1 1 1];

%% Initialize w

A = p' - diag(ones(1, M));
A(4, :) = [1; 1; 1; 1];
b = [0; 0; 0; 1];
w = inv(A) * b;
w = w';


%% Initialize u

for i = 1 : M
u(i) = w(i) / (c(i) * w(1)) * lam_1_;
end

u_last = u;

for iteration = 1 : 5
    %% Find L(N)
    [L_N, lam_1_N] = fl(u, w, c, M, N);
    L_N_1 = fl(u, w, c, M, N - 1);

    %% Find u1
    u(1) = lam_1_ / lam_1_N * u(1);

    %% Find u
    for i = 2 : M
```

```
    u(i) = u(1)  * (L_N(i) - L_N_1(i)) / (L_N(1) - L_N_1(1)) * (c(1) * a(1)) /
   ↪  (c(i) * a(i));
    end

    %% Finish iteration

    % min_delta = min(abs(u - u_last));

    u_last = u;

    fprintf('Iteration #%d\nSumm u(i) = %f\nu(i) = [%f, %f, %f, %f]\n\n',
   ↪ iteration, sum(u), u(1), u(2), u(3), u(4));
end
```

Листинг 4.14: Решение СМО методом простых итераций

```
function [L, lumbda1] = fl(u, w, c, M, N)
    P = zeros(M, N + 1, N + 1);
    t = zeros(1, M);

    P(:, 1, 1) = 1;
    for r = 1 : N
        %% Step 1

        for i = 1 : M
            t(i) = 0;
            for n = 1 : r
                t(i) = t(i) + n / (min(n, c(i)) * u(i)) * P(i, n, r);
            end
        end

        %% Step 2

        temp = 0;
        for i = 1 : M
        temp = temp + w(i) * t(i) / w(1);
        end

        lumbda1 = r / temp;

        %% Step 3

        for i = 1 : M
            for n = 1 : r
                P(i, n + 1, r + 1) = w(i) / w(1) * lumbda1 / (min(n, c(i)) * u
   ↪ (i)) * P(i, n, r);
            end

            P(i, 1, r + 1) = 1;
            for n = 1 : r
                P(i, 1, r + 1) = P(i, 1, r + 1) - P(i, n + 1, r + 1);
            end
        end
    end

    L = t;
end
```

Листинг 4.15: Итеративная функция решения для ССМО

Приложение 11

Решение СМО методом Ньютона

```
clear all;
close all;
clc;
format long g;

% delete(gcp);
% distcomp.feature('LocalUseMpiexec', false);
% parpool();

p =   [0 0.1 0.6 0.3;
0.7 0 0.2 0.1;
0.9 0 0 0.1;
0.3 0.3 0.4 0];

M = 4;
N = 5;

lam_1_ = 3;

c = [2 2 2 2];
a = [1 1 1 1];

%% Initialize w

A = p' - diag(ones(1, M));
A(4, :) = [1; 1; 1; 1];
b = [0; 0; 0; 1];
w = inv(A) * b;
w = (1 / w(1)) * w';

%% Error

e = 1e-06;

%% First approximation

u0 = zeros(1, M);
for i = 1 : M
u0(i) = w(i) / (c(i) * w(1)) * lam_1_;
end

fprintf('Start Conditions\ne = %f\nw = [%.6f %.6f %.6f %.6f]\nu0 = [%.6f %.6f
   ↪ %.6f %.6f]\n\n', e, w, u0);

%% Syms u

u = sym('u', [1 M]);

%% Find characteristics

G = gl3(u, w, c, M, N);
[L, Lam] = pl3(u, w, c, M, N, G);
[pL, pLam] = pl3(u, w, c, M, N - 1, G);

% [L, Lam] = fl3(u, w, c, M, N);
% [pL, pLam] = fl3(u, w, c, M, N - 1);
```

```
Function = sym(zeros(1, M));
Function(1) = simplify(lam_1_ / Lam(1) * u(1));

for i = 2 : M
% Function(i) = simplify(Function(1) * (L(i) - pL(i)) / (L(1) - pL(1)) * (c(1)
   ↪  * a(1)) / (c(i) * a(i)));
Function(i) = simplify(u(1) * (L(i) - pL(i)) / (L(1) - pL(1)) * (c(1) * a(1))
   ↪ / (c(i) * a(i)));
end

%% Newton method with jacobian matrix

% uCurrent = u0;
uCurrent = [10 10 10 10];
uPrevious = zeros(1, M);

jaco = jacobian(Function - u);

index = 0;
condition = true;
while condition
    resf = zeros(M, 1);
    % parfor i = 1 : M
    for i = 1 : M
        resf(i) = subs(Function(i) - u(i), u, uCurrent);
    end

    rU = double(uCurrent);
    fprintf('%d | u = [%.6f %.6f %.6f %.6f] | F(u) = [%.6f %.6f %.6f %.6f] | S
   ↪  = %.6f\n', index, double(uCurrent), double(resf), c * rU');

    uPrevious = uCurrent;

    resjaco = zeros(M, M);
    % parfor i = 1 : M
    for i = 1 : M
        for j = 1 : M
            resjaco(i, j) = subs(jaco(i, j), u, uPrevious);
        end
    end

    resf = zeros(M, 1);
    % parfor i = 1 : M
    for i = 1 : M
        resf(i) = subs(Function(i) - u(i), u, uPrevious);
    end

    uCurrent = uPrevious - (resjaco \ resf)';

    condition = false;
    for i = 1 : M
        condition = condition || abs(uCurrent(i) - uPrevious(i)) > e;
    end

    index = index + 1;

    if (~condition)
    rU = double(uCurrent);
    fprintf('%d | u = [%.6f %.6f %.6f %.6f] | F(u) = [%.6f %.6f %.6f %.6f] | S
```

```
    ↪  = %.6f\n', index, double(uCurrent), double(resf), c * rU');
    end
end

resf = zeros(M, 1);
% parfor i = 1 : M
for i = 1 : M
    resf(i) = subs(Function(i) - u(i), u, uCurrent);
end

rU = double(uCurrent);
rL = double(subs(L, u, uCurrent));
rLam = double(subs(Lam, u, uCurrent));
rS = c * rU';
fprintf('\nResult\nu = [%.6f %.6f %.6f %.6f]\nF(u) = [%.6f %.6f %.6f %.6f]\nL(
    ↪ N) = [%.6f %.6f %.6f %.6f]\nlam(N) = [%.6f %.6f %.6f %.6f]\nS = %.6f\n',
    ↪  rU, double(resf), rL, rLam, rS);
```

Листинг 4.16: Решение СМО методом Ньютона

```
function [G] = gl3(u, w, c, M, N)
    G = sym(zeros(M, N + 1));
    G(:, 1) = 1;

    for k = 0 : N
        %% Find G(1, k)

        tempMul = 1;
        for j = 1 : k
            tempMul = tempMul * min(j, c(1)) * u(1);
        end

        G(1, k + 1) = (w(1) ^ k) / tempMul;
    end

    for r = 2 : M
        for k = 1 : N
            %% Find G(r, k)

            tempSum = 0;
            for h = 0 : k
                Z = 0;
                if (h == 0)
                    Z = 1;
                else
                    %% Find Z(r, h)
                    tempMul = 1;
                    for j = 1 : h
                        tempMul = tempMul * min(j, c(r)) * u(r);
                    end

                    Z = (w(r) ^ h) / tempMul;
                end

                tempSum = tempSum + Z * G(r - 1, k - h + 1);
            end

            G(r, k + 1) = tempSum;
        end
    end
```

```
    G = simplify(G);
end
```

Листинг 4.17: Расчет нормирующей константы

```
function [L, Lam] = pl3(u, w, c, M, N, G)
    L = sym(zeros(1, M));
    Lam = sym(zeros(1, M));

    for i = 1 : M
        tempSum = 0;
        for n = 1 : N
            Z = 0;
            if (n == 0)
                Z = 1;
            else
                %% Find Z(i, n)
                tempMul = 1;
                for j = 1 : n
                    tempMul = tempMul * min(j, c(i)) * u(i);
                end

                Z = (w(i) ^ n) / tempMul;
            end

            tempSum = tempSum + n * Z * G(M - 1, N - n + 1);
        end

        L(i) = tempSum / G(M, N + 1);
        Lam(i) = w(i) * G(M, N - 1 + 1) / G(M, N + 1);
    end

    L = simplify(L);
    Lam = simplify(Lam);
end
```

Листинг 4.18: Расчет основных характеристик СМО

Приложение 12

Расчет статистических значений для GERT сети

```
clear all;
close all;
clc;
format long g;

syms s;

% W-functions
W12 = 0.5 * exp(10*s + 8 * s^2);
W16 = 0.5 * exp(23 * s + 312.5 * s^2);
W22 = 0.2 * exp(13 * s + 128 * s^2);
W26 = 0.8 * exp(11 * s + 128 * s^2);
W35 = 1 * exp(10 * s + 40.5 * s^2);
W41 = 0.4 * exp(37 * s + 128 * s^2);
W43 = 0.3 * exp(12 * s + 128 * s^2);
W46 = 0.3 * exp(12 * s + 1200.5 * s^2);
```

```
W54 = 0.3 * exp(15 * s + 312.5  * s^2);
W55 = 0.5 * exp(19 * s + 8 * s^2);
W56 = 0.2 * exp(42 * s + 40.5 * s^2);

% We(s)
We = (W12 * W26 + W16 - W16 * W22 - W16 * W35 * W54 * W43 - W16 * W55 - W12 *
   ↪ W26 * W55 - W12 * W26 * W35 * W54 * W43 + W16 * W22 * W35 * W54 * W43 +
   ↪ W16 * W22 * W55) / (1 - W35 * W54 * W43 - W22 - W55 + W22 * W55 + W22 *
   ↪ W35 * W54 * W43);
We = simplify(We);

% We(0)
We0 = subs(We, 's', 0);
fprintf('We(0) = %.3f\n', double(We0));

% Me(s)
Me = We / We0;

% me1
me1 = diff(Me, 's', 1);
me1 = subs(me1, 's', 0);
fprintf('me1 = %.3f\n', double(me1));

% me2
me2 = diff(Me, 's', 2);
me2 = subs(me2, 's', 0);
fprintf('me2 = %.3f\n', double(me2));

% de
de = me2 - me1 ^ 2;
fprintf('de = %.3f\n', double(de));
```

Листинг 4.19: Расчет статистических значений для GERT сети

Приложение 13

Расчет статистических значений при помощи анализа потокового графа

```
clear all;
close all;
clc;
format long g;

syms s;
syms W61;

% W-functions
W12 = 0.5 * exp(10*s + 8 * s^2);
W16 = 0.5 * exp(23 * s + 312.5 * s^2);
W22 = 0.2 * exp(13 * s + 128 * s^2);
W26 = 0.8 * exp(11 * s + 128 * s^2);
W35 = 1 * exp(10 * s + 40.5 * s^2);
W41 = 0.4 * exp(37 * s + 128 * s^2);
W43 = 0.3 * exp(12 * s + 128 * s^2);
W46 = 0.3 * exp(12 * s + 1200.5 * s^2);
W54 = 0.3 * exp(15 * s + 312.5  * s^2);
```

```
W55 = 0.5 * exp(19 * s + 8 * s^2);
W56 = 0.2 * exp(42 * s + 40.5 * s^2);

Q = [0 W12 0 0 0 W16;
     0 W22 0 0 0 W26;
     0 0 0 0 W35 0;
     W41 0 W43 0 0 W46;
     0 0 0 W54 W55 W56;
     W61 0 0 0 0 0];

% Determinate A
A = eye(size(Q, 1)) - Q';
detA = det(A);
dDetA = diff(detA, W61, 1);
detA0 = subs(detA, W61, 0);

% We(s)
We = dDetA / detA0;
We = simplify(We);

% We(0)
We0 = subs(We, 's', 0);
fprintf('We(0) = %.3f\n', double(We0));

% Me(s)
Me = We / We0;

% me1
me1 = diff(Me, 's', 1);
me1 = subs(me1, 's', 0);
fprintf('me1 = %.3f\n', double(me1));

% me2
me2 = diff(Me, 's', 2);
me2 = subs(me2, 's', 0);
fprintf('me2 = %.3f\n', double(me2));

% de
de = me2 - me1 ^ 2;
fprintf('de = %.3f\n', double(de));
```

Листинг 4.20: Расчет статистических значений при помощи анализа потокового графа