1 Полный перебор для 3х недель

In [73]:
```python
import itertools
import numpy as np

# Матрицы вероятностей переходов P[action][state_from][state_to]
transition_probabilities = {
    0: [[0.3, 0.5, 0.2],
        [0.2, 0.6, 0.2],
        [0.1, 0.2, 0.7]],
    1: [[0.2, 0.7, 0.1],
        [0.1, 0.4, 0.5],
        [0.1, 0.2, 0.7]],
    2: [[0.3, 0.4, 0.3],
        [0.2, 0.6, 0.2],
        [0.1, 0.3, 0.6]],
}

# Матрицы вознаграждений R[action][state_from][state_to]
rewards = {
    0: [[110, 100, 70],
        [100,  80, 50],
        [ 80,  60, 40]],
    1: [[120, 100, 70],
        [110, 100, 90],
        [100,  70, 60]],
    2: [[110,  80, 50],
        [100,  60, 40],
        [ 80,  70, 60]],
}

actions = [0, 1, 2]  # Доступные действия (индексы)
states = [0, 1, 2]   # Индексы состояний: 0 - «Отличный», 1 - «Хороший», 2 -
months = 3  # Количество месяцев (длительность стратегии)
state_names = ["Отличный", "Хороший", "Удовлетворительный"]  # Человекочитае
action_names = ["3% скидка", "Бесплатная доставка", "Ничего"]  # Названия де

def calculate_expected_reward(strategy, initial_state, verbose=False):
    """
    Вычисление ожидаемого вознаграждения для заданной стратегии действий.

    Параметры:
    strategy (list[int]): Стратегия действий (порядок действий на каждый мес
    initial_state (int): Начальное состояние системы.
    verbose (bool): Выводить ли промежуточные результаты.

    Возвращает:
    float: Ожидаемое вознаграждение для стратегии.
    """
    # Инициализация вероятностей состояний: на старте мы находимся в начальн
    state_probabilities = [0.0, 0.0, 0.0]
    state_probabilities[initial_state] = 1.0
```

```
    if verbose:
        print(f"\n=== Расчёт для стратегии {strategy} ===")
        print(f"Действия: {[action_names[a] for a in strategy]}")
        print(f"Начальное состояние: {state_names[initial_state]}")
        print(f"Начальное распределение вероятностей: {np.round(state_probab

    total_reward = 0.0  # Общее ожидаемое вознаграждение

    # Процесс для каждого действия в стратегии
    for month, action in enumerate(strategy):
        if verbose:
            print(f"\nМесяц {month+1}: Действие = {action} ({action_names[ac
            print(f"  Текущие вероятности состояний: {np.round(state_probabi

        step_reward = 0.0  # Вознаграждение за текущий месяц
        new_state_probabilities = [0.0, 0.0, 0.0]  # Вероятности для нового

        # Проходим по всем возможным переходам между состояниями
        for i in range(3):
            if state_probabilities[i] > 0:  # Оптимизация: показываем только
                if verbose:
                    print(f"  Из состояния {i} ({state_names[i]}) с вероятно

                for j in range(3):
                    transition_probability = transition_probabilities[actior
                    reward = rewards[action][i][j]
                    contribution = state_probabilities[i] * transition_proba

                    if contribution > 0 and verbose:
                        print(f"    → в состояние {j} ({state_names[j]}) с в
                              f"награда {reward}, вклад: {contribution:.3f}"

                    total_reward += contribution
                    step_reward += contribution
                    new_state_probabilities[j] += state_probabilities[i] * t

        # Обновляем вероятности состояний для следующего шага
        state_probabilities = new_state_probabilities

        if verbose:
            print(f"  Вознаграждение за месяц {month+1}: {step_reward:.3f}")
            print(f"  Новые вероятности состояний: {np.round(state_probabili
            print(f"  Накопленное вознаграждение: {total_reward:.3f}")

    if verbose:
        print(f"\nИтоговое ожидаемое вознаграждение для стратегии {strategy}

    return total_reward

best_strategies = {0: None, 1: None, 2: None}  # Словарь для хранения лучших
best_rewards = {0: -1e9, 1: -1e9, 2: -1e9}  # Словарь для хранения лучших во

# Поиск наилучшей стратегии для каждого начального состояния
for initial_state in range(3):
    best_strategy = None
    best_reward = -1e9  # Начальная очень низкая оценка
```

```
    strategies_evaluated = 0
    improvements = 0

    print(f"\n{'='*80}")
    print(f"Поиск лучшей стратегии для начального состояния «{state_names[ir
    print(f"{'='*80}")

    # Подсчет общего количества стратегий
    total_strategies = len(actions) ** months
    print(f"Всего возможных стратегий: {total_strategies}")

    # Генерация всех возможных стратегий на протяжении заданного количества
    for strategy in itertools.product(actions, repeat=months):
        strategies_evaluated += 1

        # Подробный вывод для первой и последней стратегии
        verbose = strategies_evaluated == 1 or strategies_evaluated == total

        if verbose:
            print(f"\nПроверка стратегии {strategies_evaluated}/{total_strat

        expected_reward = calculate_expected_reward(strategy, initial_state,

        # Обновляем лучшую стратегию, если текущая дает большее вознагражден
        if expected_reward > best_reward:
            improvement = expected_reward - best_reward if best_reward != -1
            improvements += 1
            print(f"Улучшение #{improvements}: стратегия {strategy} = {[acti
                  f"вознаграждение: {expected_reward:.2f} (+{improvement:.2f
            best_strategy = strategy
            best_reward = expected_reward
            best_strategies[initial_state] = best_strategy
            best_rewards[initial_state] = best_reward

    # Выводим результаты для каждого начального состояния
    print(f"\n{'-'*80}")
    print(f"Для начального состояния «{state_names[initial_state]}»:")
    print(f"  Лучшая стратегия действий: {best_strategy} = {[action_names[a]
    print(f"  Ожидаемое вознаграждение: {best_reward:.2f}")
    print(f"  Количество улучшений: {improvements} из {strategies_evaluated}
    print(f"{'-'*80}\n")

print(f"\n{'='*40} Ответ {'='*40}")
for initial_state in range(3):
    print(f"Для начального состояния «{state_names[initial_state]}»:")
    print(f"  Лучшая стратегия действий: {best_strategies[initial_state]} =
          f"{[action_names[a] for a in best_strategies[initial_state]]}")
    print(f"  Ожидаемое вознаграждение: {best_rewards[initial_state]:.2f}")
    print(f"{'-'*80}\n")
```

```
================================================================================
====
Поиск лучшей стратегии для начального состояния «Отличный»
================================================================================
====
Всего возможных стратегий: 27

Проверка стратегии 1/27: (0, 0, 0)

=== Расчёт для стратегии (0, 0, 0) ===
Действия: ['3% скидка', '3% скидка', '3% скидка']
Начальное состояние: Отличный
Начальное распределение вероятностей: [1. 0. 0.]

Месяц 1: Действие = 0 (3% скидка)
  Текущие вероятности состояний: [1. 0. 0.]
  Из состояния 0 (Отличный) с вероятностью 1.000:
    → в состояние 0 (Отличный) с вер. 0.300, награда 110, вклад: 33.000
    → в состояние 1 (Хороший) с вер. 0.500, награда 100, вклад: 50.000
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 70, вклад: 1
4.000
  Вознаграждение за месяц 1: 97.000
  Новые вероятности состояний: [0.3 0.5 0.2]
  Накопленное вознаграждение: 97.000

Месяц 2: Действие = 0 (3% скидка)
  Текущие вероятности состояний: [0.3 0.5 0.2]
  Из состояния 0 (Отличный) с вероятностью 0.300:
    → в состояние 0 (Отличный) с вер. 0.300, награда 110, вклад: 9.900
    → в состояние 1 (Хороший) с вер. 0.500, награда 100, вклад: 15.000
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 70, вклад: 4.
200
  Из состояния 1 (Хороший) с вероятностью 0.500:
    → в состояние 0 (Отличный) с вер. 0.200, награда 100, вклад: 10.000
    → в состояние 1 (Хороший) с вер. 0.600, награда 80, вклад: 24.000
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 50, вклад: 5.
000
  Из состояния 2 (Удовлетворительный) с вероятностью 0.200:
    → в состояние 0 (Отличный) с вер. 0.100, награда 80, вклад: 1.600
    → в состояние 1 (Хороший) с вер. 0.200, награда 60, вклад: 2.400
    → в состояние 2 (Удовлетворительный) с вер. 0.700, награда 40, вклад: 5.
600
  Вознаграждение за месяц 2: 77.700
  Новые вероятности состояний: [0.21 0.49 0.3 ]
  Накопленное вознаграждение: 174.700

Месяц 3: Действие = 0 (3% скидка)
  Текущие вероятности состояний: [0.21 0.49 0.3 ]
  Из состояния 0 (Отличный) с вероятностью 0.210:
    → в состояние 0 (Отличный) с вер. 0.300, награда 110, вклад: 6.930
    → в состояние 1 (Хороший) с вер. 0.500, награда 100, вклад: 10.500
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 70, вклад: 2.
940
  Из состояния 1 (Хороший) с вероятностью 0.490:
    → в состояние 0 (Отличный) с вер. 0.200, награда 100, вклад: 9.800
    → в состояние 1 (Хороший) с вер. 0.600, награда 80, вклад: 23.520
```

```
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 50, вклад: 4.
900
  Из состояния 2 (Удовлетворительный) с вероятностью 0.300:
    → в состояние 0 (Отличный) с вер. 0.100, награда 80, вклад: 2.400
    → в состояние 1 (Хороший) с вер. 0.200, награда 60, вклад: 3.600
    → в состояние 2 (Удовлетворительный) с вер. 0.700, награда 40, вклад: 8.
400
  Вознаграждение за месяц 3: 72.990
  Новые вероятности состояний: [0.191 0.459 0.35 ]
  Накопленное вознаграждение: 247.690

Итоговое ожидаемое вознаграждение для стратегии (0, 0, 0): 247.690
Улучшение #1: стратегия (0, 0, 0) = ['3% скидка', '3% скидка', '3% скидка'],
вознаграждение: 247.69 (+247.69)
Улучшение #2: стратегия (0, 0, 1) = ['3% скидка', '3% скидка', 'Бесплатная д
оставка'], вознаграждение: 262.75 (+15.06)
Улучшение #3: стратегия (0, 1, 1) = ['3% скидка', 'Бесплатная доставка', 'Бе
сплатная доставка'], вознаграждение: 272.55 (+9.80)
Улучшение #4: стратегия (1, 1, 1) = ['Бесплатная доставка', 'Бесплатная дост
авка', 'Бесплатная доставка'], вознаграждение: 278.40 (+5.85)

Проверка стратегии 27/27: (2, 2, 2)

=== Расчёт для стратегии (2, 2, 2) ===
Действия: ['Ничего', 'Ничего', 'Ничего']
Начальное состояние: Отличный
Начальное распределение вероятностей: [1. 0. 0.]

Месяц 1: Действие = 2 (Ничего)
  Текущие вероятности состояний: [1. 0. 0.]
  Из состояния 0 (Отличный) с вероятностью 1.000:
    → в состояние 0 (Отличный) с вер. 0.300, награда 110, вклад: 33.000
    → в состояние 1 (Хороший) с вер. 0.400, награда 80, вклад: 32.000
    → в состояние 2 (Удовлетворительный) с вер. 0.300, награда 50, вклад: 1
5.000
  Вознаграждение за месяц 1: 80.000
  Новые вероятности состояний: [0.3 0.4 0.3]
  Накопленное вознаграждение: 80.000

Месяц 2: Действие = 2 (Ничего)
  Текущие вероятности состояний: [0.3 0.4 0.3]
  Из состояния 0 (Отличный) с вероятностью 0.300:
    → в состояние 0 (Отличный) с вер. 0.300, награда 110, вклад: 9.900
    → в состояние 1 (Хороший) с вер. 0.400, награда 80, вклад: 9.600
    → в состояние 2 (Удовлетворительный) с вер. 0.300, награда 50, вклад: 4.
500
  Из состояния 1 (Хороший) с вероятностью 0.400:
    → в состояние 0 (Отличный) с вер. 0.200, награда 100, вклад: 8.000
    → в состояние 1 (Хороший) с вер. 0.600, награда 60, вклад: 14.400
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 40, вклад: 3.
200
  Из состояния 2 (Удовлетворительный) с вероятностью 0.300:
    → в состояние 0 (Отличный) с вер. 0.100, награда 80, вклад: 2.400
    → в состояние 1 (Хороший) с вер. 0.300, награда 70, вклад: 6.300
    → в состояние 2 (Удовлетворительный) с вер. 0.600, награда 60, вклад: 1
0.800
```

```
  Вознаграждение за месяц 2: 69.100
  Новые вероятности состояний: [0.2  0.45 0.35]
  Накопленное вознаграждение: 149.100

Месяц 3: Действие = 2 (Ничего)
  Текущие вероятности состояний: [0.2  0.45 0.35]
  Из состояния 0 (Отличный) с вероятностью 0.200:
    → в состояние 0 (Отличный) с вер. 0.300, награда 110, вклад: 6.600
    → в состояние 1 (Хороший) с вер. 0.400, награда 80, вклад: 6.400
    → в состояние 2 (Удовлетворительный) с вер. 0.300, награда 50, вклад: 3.
000
  Из состояния 1 (Хороший) с вероятностью 0.450:
    → в состояние 0 (Отличный) с вер. 0.200, награда 100, вклад: 9.000
    → в состояние 1 (Хороший) с вер. 0.600, награда 60, вклад: 16.200
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 40, вклад: 3.
600
  Из состояния 2 (Удовлетворительный) с вероятностью 0.350:
    → в состояние 0 (Отличный) с вер. 0.100, награда 80, вклад: 2.800
    → в состояние 1 (Хороший) с вер. 0.300, награда 70, вклад: 7.350
    → в состояние 2 (Удовлетворительный) с вер. 0.600, награда 60, вклад: 1
2.600
  Вознаграждение за месяц 3: 67.550
  Новые вероятности состояний: [0.185 0.455 0.36 ]
  Накопленное вознаграждение: 216.650

Итоговое ожидаемое вознаграждение для стратегии (2, 2, 2): 216.650

--------------------------------------------------------------------------
----
Для начального состояния «Отличный»:
  Лучшая стратегия действий: (1, 1, 1) = ['Ничего', 'Ничего', 'Ничего']
  Ожидаемое вознаграждение: 278.40
  Количество улучшений: 4 из 27 оцененных стратегий
--------------------------------------------------------------------------
----


==========================================================================
====
Поиск лучшей стратегии для начального состояния «Хороший»
==========================================================================
====
Всего возможных стратегий: 27

Проверка стратегии 1/27: (0, 0, 0)

=== Расчёт для стратегии (0, 0, 0) ===
Действия: ['3% скидка', '3% скидка', '3% скидка']
Начальное состояние: Хороший
Начальное распределение вероятностей: [0. 1. 0.]

Месяц 1: Действие = 0 (3% скидка)
  Текущие вероятности состояний: [0. 1. 0.]
  Из состояния 1 (Хороший) с вероятностью 1.000:
    → в состояние 0 (Отличный) с вер. 0.200, награда 100, вклад: 20.000
    → в состояние 1 (Хороший) с вер. 0.600, награда 80, вклад: 48.000
```

```
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 50, вклад: 1
0.000
  Вознаграждение за месяц 1: 78.000
  Новые вероятности состояний: [0.2 0.6 0.2]
  Накопленное вознаграждение: 78.000

Месяц 2: Действие = 0 (3% скидка)
  Текущие вероятности состояний: [0.2 0.6 0.2]
  Из состояния 0 (Отличный) с вероятностью 0.200:
    → в состояние 0 (Отличный) с вер. 0.300, награда 110, вклад: 6.600
    → в состояние 1 (Хороший) с вер. 0.500, награда 100, вклад: 10.000
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 70, вклад: 2.
800
  Из состояния 1 (Хороший) с вероятностью 0.600:
    → в состояние 0 (Отличный) с вер. 0.200, награда 100, вклад: 12.000
    → в состояние 1 (Хороший) с вер. 0.600, награда 80, вклад: 28.800
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 50, вклад: 6.
000
  Из состояния 2 (Удовлетворительный) с вероятностью 0.200:
    → в состояние 0 (Отличный) с вер. 0.100, награда 80, вклад: 1.600
    → в состояние 1 (Хороший) с вер. 0.200, награда 60, вклад: 2.400
    → в состояние 2 (Удовлетворительный) с вер. 0.700, награда 40, вклад: 5.
600
  Вознаграждение за месяц 2: 75.800
  Новые вероятности состояний: [0.2 0.5 0.3]
  Накопленное вознаграждение: 153.800

Месяц 3: Действие = 0 (3% скидка)
  Текущие вероятности состояний: [0.2 0.5 0.3]
  Из состояния 0 (Отличный) с вероятностью 0.200:
    → в состояние 0 (Отличный) с вер. 0.300, награда 110, вклад: 6.600
    → в состояние 1 (Хороший) с вер. 0.500, награда 100, вклад: 10.000
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 70, вклад: 2.
800
  Из состояния 1 (Хороший) с вероятностью 0.500:
    → в состояние 0 (Отличный) с вер. 0.200, награда 100, вклад: 10.000
    → в состояние 1 (Хороший) с вер. 0.600, награда 80, вклад: 24.000
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 50, вклад: 5.
000
  Из состояния 2 (Удовлетворительный) с вероятностью 0.300:
    → в состояние 0 (Отличный) с вер. 0.100, награда 80, вклад: 2.400
    → в состояние 1 (Хороший) с вер. 0.200, награда 60, вклад: 3.600
    → в состояние 2 (Удовлетворительный) с вер. 0.700, награда 40, вклад: 8.
400
  Вознаграждение за месяц 3: 72.800
  Новые вероятности состояний: [0.19 0.46 0.35]
  Накопленное вознаграждение: 226.600

Итоговое ожидаемое вознаграждение для стратегии (0, 0, 0): 226.600
Улучшение #1: стратегия (0, 0, 0) = ['3% скидка', '3% скидка', '3% скидка'],
вознаграждение: 226.60 (+226.60)
Улучшение #2: стратегия (0, 0, 1) = ['3% скидка', '3% скидка', 'Бесплатная д
оставка'], вознаграждение: 241.80 (+15.20)
Улучшение #3: стратегия (0, 1, 1) = ['3% скидка', 'Бесплатная доставка', 'Бе
сплатная доставка'], вознаграждение: 251.80 (+10.00)
Улучшение #4: стратегия (1, 1, 1) = ['Бесплатная доставка', 'Бесплатная дост
```

```
авка', 'Бесплатная доставка'], вознаграждение: 257.25 (+5.45)

Проверка стратегии 27/27: (2, 2, 2)

=== Расчёт для стратегии (2, 2, 2) ===
Действия: ['Ничего', 'Ничего', 'Ничего']
Начальное состояние: Хороший
Начальное распределение вероятностей: [0. 1. 0.]

Месяц 1: Действие = 2 (Ничего)
  Текущие вероятности состояний: [0. 1. 0.]
  Из состояния 1 (Хороший) с вероятностью 1.000:
    → в состояние 0 (Отличный) с вер. 0.200, награда 100, вклад: 20.000
    → в состояние 1 (Хороший) с вер. 0.600, награда 60, вклад: 36.000
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 40, вклад: 8.
000
  Вознаграждение за месяц 1: 64.000
  Новые вероятности состояний: [0.2 0.6 0.2]
  Накопленное вознаграждение: 64.000

Месяц 2: Действие = 2 (Ничего)
  Текущие вероятности состояний: [0.2 0.6 0.2]
  Из состояния 0 (Отличный) с вероятностью 0.200:
    → в состояние 0 (Отличный) с вер. 0.300, награда 110, вклад: 6.600
    → в состояние 1 (Хороший) с вер. 0.400, награда 80, вклад: 6.400
    → в состояние 2 (Удовлетворительный) с вер. 0.300, награда 50, вклад: 3.
000
  Из состояния 1 (Хороший) с вероятностью 0.600:
    → в состояние 0 (Отличный) с вер. 0.200, награда 100, вклад: 12.000
    → в состояние 1 (Хороший) с вер. 0.600, награда 60, вклад: 21.600
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 40, вклад: 4.
800
  Из состояния 2 (Удовлетворительный) с вероятностью 0.200:
    → в состояние 0 (Отличный) с вер. 0.100, награда 80, вклад: 1.600
    → в состояние 1 (Хороший) с вер. 0.300, награда 70, вклад: 4.200
    → в состояние 2 (Удовлетворительный) с вер. 0.600, награда 60, вклад: 7.
200
  Вознаграждение за месяц 2: 67.400
  Новые вероятности состояний: [0.2 0.5 0.3]
  Накопленное вознаграждение: 131.400

Месяц 3: Действие = 2 (Ничего)
  Текущие вероятности состояний: [0.2 0.5 0.3]
  Из состояния 0 (Отличный) с вероятностью 0.200:
    → в состояние 0 (Отличный) с вер. 0.300, награда 110, вклад: 6.600
    → в состояние 1 (Хороший) с вер. 0.400, награда 80, вклад: 6.400
    → в состояние 2 (Удовлетворительный) с вер. 0.300, награда 50, вклад: 3.
000
  Из состояния 1 (Хороший) с вероятностью 0.500:
    → в состояние 0 (Отличный) с вер. 0.200, награда 100, вклад: 10.000
    → в состояние 1 (Хороший) с вер. 0.600, награда 60, вклад: 18.000
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 40, вклад: 4.
000
  Из состояния 2 (Удовлетворительный) с вероятностью 0.300:
    → в состояние 0 (Отличный) с вер. 0.100, награда 80, вклад: 2.400
    → в состояние 1 (Хороший) с вер. 0.300, награда 70, вклад: 6.300
```

```
    → в состояние 2 (Удовлетворительный) с вер. 0.600, награда 60, вклад: 1
0.800
  Вознаграждение за месяц 3: 67.500
  Новые вероятности состояний: [0.19 0.47 0.34]
  Накопленное вознаграждение: 198.900

Итоговое ожидаемое вознаграждение для стратегии (2, 2, 2): 198.900

--------------------------------------------------------------------------------
----
Для начального состояния «Хороший»:
  Лучшая стратегия действий: (1, 1, 1) = ['Ничего', 'Ничего', 'Ничего']
  Ожидаемое вознаграждение: 257.25
  Количество улучшений: 4 из 27 оцененных стратегий
--------------------------------------------------------------------------------
----


================================================================================
====
Поиск лучшей стратегии для начального состояния «Удовлетворительный»
================================================================================
====
Всего возможных стратегий: 27

Проверка стратегии 1/27: (0, 0, 0)

=== Расчёт для стратегии (0, 0, 0) ===
Действия: ['3% скидка', '3% скидка', '3% скидка']
Начальное состояние: Удовлетворительный
Начальное распределение вероятностей: [0. 0. 1.]

Месяц 1: Действие = 0 (3% скидка)
  Текущие вероятности состояний: [0. 0. 1.]
  Из состояния 2 (Удовлетворительный) с вероятностью 1.000:
    → в состояние 0 (Отличный) с вер. 0.100, награда 80, вклад: 8.000
    → в состояние 1 (Хороший) с вер. 0.200, награда 60, вклад: 12.000
    → в состояние 2 (Удовлетворительный) с вер. 0.700, награда 40, вклад: 2
8.000
  Вознаграждение за месяц 1: 48.000
  Новые вероятности состояний: [0.1 0.2 0.7]
  Накопленное вознаграждение: 48.000

Месяц 2: Действие = 0 (3% скидка)
  Текущие вероятности состояний: [0.1 0.2 0.7]
  Из состояния 0 (Отличный) с вероятностью 0.100:
    → в состояние 0 (Отличный) с вер. 0.300, награда 110, вклад: 3.300
    → в состояние 1 (Хороший) с вер. 0.500, награда 100, вклад: 5.000
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 70, вклад: 1.
400
  Из состояния 1 (Хороший) с вероятностью 0.200:
    → в состояние 0 (Отличный) с вер. 0.200, награда 100, вклад: 4.000
    → в состояние 1 (Хороший) с вер. 0.600, награда 80, вклад: 9.600
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 50, вклад: 2.
000
  Из состояния 2 (Удовлетворительный) с вероятностью 0.700:
```

```
    → в состояние 0 (Отличный) с вер. 0.100, награда 80, вклад: 5.600
    → в состояние 1 (Хороший) с вер. 0.200, награда 60, вклад: 8.400
    → в состояние 2 (Удовлетворительный) с вер. 0.700, награда 40, вклад: 1
9.600
  Вознаграждение за месяц 2: 58.900
  Новые вероятности состояний: [0.14 0.31 0.55]
  Накопленное вознаграждение: 106.900

Месяц 3: Действие = 0 (3% скидка)
  Текущие вероятности состояний: [0.14 0.31 0.55]
  Из состояния 0 (Отличный) с вероятностью 0.140:
    → в состояние 0 (Отличный) с вер. 0.300, награда 110, вклад: 4.620
    → в состояние 1 (Хороший) с вер. 0.500, награда 100, вклад: 7.000
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 70, вклад: 1.
960
  Из состояния 1 (Хороший) с вероятностью 0.310:
    → в состояние 0 (Отличный) с вер. 0.200, награда 100, вклад: 6.200
    → в состояние 1 (Хороший) с вер. 0.600, награда 80, вклад: 14.880
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 50, вклад: 3.
100
  Из состояния 2 (Удовлетворительный) с вероятностью 0.550:
    → в состояние 0 (Отличный) с вер. 0.100, награда 80, вклад: 4.400
    → в состояние 1 (Хороший) с вер. 0.200, награда 60, вклад: 6.600
    → в состояние 2 (Удовлетворительный) с вер. 0.700, награда 40, вклад: 1
5.400
  Вознаграждение за месяц 3: 64.160
  Новые вероятности состояний: [0.159 0.366 0.475]
  Накопленное вознаграждение: 171.060

Итоговое ожидаемое вознаграждение для стратегии (0, 0, 0): 171.060
Улучшение #1: стратегия (0, 0, 0) = ['3% скидка', '3% скидка', '3% скидка'],
вознаграждение: 171.06 (+171.06)
Улучшение #2: стратегия (0, 0, 1) = ['3% скидка', '3% скидка', 'Бесплатная д
оставка'], вознаграждение: 187.10 (+16.04)
Улучшение #3: стратегия (0, 1, 1) = ['3% скидка', 'Бесплатная доставка', 'Бе
сплатная доставка'], вознаграждение: 202.05 (+14.95)
Улучшение #4: стратегия (1, 0, 1) = ['Бесплатная доставка', '3% скидка', 'Бе
сплатная доставка'], вознаграждение: 205.10 (+3.05)
Улучшение #5: стратегия (1, 1, 1) = ['Бесплатная доставка', 'Бесплатная дост
авка', 'Бесплатная доставка'], вознаграждение: 220.05 (+14.95)
Улучшение #6: стратегия (2, 1, 1) = ['Ничего', 'Бесплатная доставка', 'Беспл
атная доставка'], вознаграждение: 222.65 (+2.60)

Проверка стратегии 27/27: (2, 2, 2)

=== Расчёт для стратегии (2, 2, 2) ===
Действия: ['Ничего', 'Ничего', 'Ничего']
Начальное состояние: Удовлетворительный
Начальное распределение вероятностей: [0. 0. 1.]

Месяц 1: Действие = 2 (Ничего)
  Текущие вероятности состояний: [0. 0. 1.]
  Из состояния 2 (Удовлетворительный) с вероятностью 1.000:
    → в состояние 0 (Отличный) с вер. 0.100, награда 80, вклад: 8.000
    → в состояние 1 (Хороший) с вер. 0.300, награда 70, вклад: 21.000
    → в состояние 2 (Удовлетворительный) с вер. 0.600, награда 60, вклад: 3
```

```
6.000
  Вознаграждение за месяц 1: 65.000
  Новые вероятности состояний: [0.1 0.3 0.6]
  Накопленное вознаграждение: 65.000

Месяц 2: Действие = 2 (Ничего)
  Текущие вероятности состояний: [0.1 0.3 0.6]
  Из состояния 0 (Отличный) с вероятностью 0.100:
    → в состояние 0 (Отличный) с вер. 0.300, награда 110, вклад: 3.300
    → в состояние 1 (Хороший) с вер. 0.400, награда 80, вклад: 3.200
    → в состояние 2 (Удовлетворительный) с вер. 0.300, награда 50, вклад: 1.
500
  Из состояния 1 (Хороший) с вероятностью 0.300:
    → в состояние 0 (Отличный) с вер. 0.200, награда 100, вклад: 6.000
    → в состояние 1 (Хороший) с вер. 0.600, награда 60, вклад: 10.800
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 40, вклад: 2.
400
  Из состояния 2 (Удовлетворительный) с вероятностью 0.600:
    → в состояние 0 (Отличный) с вер. 0.100, награда 80, вклад: 4.800
    → в состояние 1 (Хороший) с вер. 0.300, награда 70, вклад: 12.600
    → в состояние 2 (Удовлетворительный) с вер. 0.600, награда 60, вклад: 2
1.600
  Вознаграждение за месяц 2: 66.200
  Новые вероятности состояний: [0.15 0.4  0.45]
  Накопленное вознаграждение: 131.200

Месяц 3: Действие = 2 (Ничего)
  Текущие вероятности состояний: [0.15 0.4  0.45]
  Из состояния 0 (Отличный) с вероятностью 0.150:
    → в состояние 0 (Отличный) с вер. 0.300, награда 110, вклад: 4.950
    → в состояние 1 (Хороший) с вер. 0.400, награда 80, вклад: 4.800
    → в состояние 2 (Удовлетворительный) с вер. 0.300, награда 50, вклад: 2.
250
  Из состояния 1 (Хороший) с вероятностью 0.400:
    → в состояние 0 (Отличный) с вер. 0.200, награда 100, вклад: 8.000
    → в состояние 1 (Хороший) с вер. 0.600, награда 60, вклад: 14.400
    → в состояние 2 (Удовлетворительный) с вер. 0.200, награда 40, вклад: 3.
200
  Из состояния 2 (Удовлетворительный) с вероятностью 0.450:
    → в состояние 0 (Отличный) с вер. 0.100, награда 80, вклад: 3.600
    → в состояние 1 (Хороший) с вер. 0.300, награда 70, вклад: 9.450
    → в состояние 2 (Удовлетворительный) с вер. 0.600, награда 60, вклад: 1
6.200
  Вознаграждение за месяц 3: 66.850
  Новые вероятности состояний: [0.17  0.435 0.395]
  Накопленное вознаграждение: 198.050

Итоговое ожидаемое вознаграждение для стратегии (2, 2, 2): 198.050

--------------------------------------------------------------------------
----
Для начального состояния «Удовлетворительный»:
  Лучшая стратегия действий: (2, 1, 1) = ['Ничего', 'Ничего', 'Ничего']
  Ожидаемое вознаграждение: 222.65
  Количество улучшений: 6 из 27 оцененных стратегий
--------------------------------------------------------------------------
```

```
----


======================================== Ответ ============================
===========
Для начального состояния «Отличный»:
  Лучшая стратегия действий: (1, 1, 1) = ['Бесплатная доставка', 'Бесплатная
доставка', 'Бесплатная доставка']
  Ожидаемое вознаграждение: 278.40
------------------------------------------------------------------------------
----

Для начального состояния «Хороший»:
  Лучшая стратегия действий: (1, 1, 1) = ['Бесплатная доставка', 'Бесплатная
доставка', 'Бесплатная доставка']
  Ожидаемое вознаграждение: 257.25
------------------------------------------------------------------------------
----

Для начального состояния «Удовлетворительный»:
  Лучшая стратегия действий: (2, 1, 1) = ['Ничего', 'Бесплатная доставка',
'Бесплатная доставка']
  Ожидаемое вознаграждение: 222.65
------------------------------------------------------------------------------
----
```

---

2 Полный перебор для бесконечного горизонта планирования

In [74]:

```python
import itertools
import numpy as np

# Матрицы переходов и доходов для каждого действия
transition_matrix = {
    0: [[0.3, 0.5, 0.2],
        [0.2, 0.6, 0.2],
        [0.1, 0.2, 0.7]],
    1: [[0.2, 0.7, 0.1],
        [0.1, 0.4, 0.5],
        [0.1, 0.2, 0.7]],
    2: [[0.3, 0.4, 0.3],
        [0.2, 0.6, 0.2],
        [0.1, 0.3, 0.6]],
}

reward_matrix = {
    0: [[110, 100, 70],
        [100,  80, 50],
        [ 80,  60, 40]],
    1: [[120, 100, 70],
        [110, 100, 90],
```

```
        [100,  70, 60]],
    2: [[110,  80, 50],
        [100,  60, 40],
        [ 80,  70, 60]],
}

actions = [0, 1, 2]  # Индексы доступных действий
states = [0, 1, 2]   # Состояния: 0 - «Отличный», 1 - «Хороший», 2 - «Удовле
state_names = ["Отличный", "Хороший", "Удовлетворительный"]
action_names = ["3% скидка", "Бесплатная доставка", "Ничего"]

def evaluate_policy(policy, verbose=True):
    """
    Оценка политики: вычисление среднего дохода и стационарного распределени

    Параметры:
    policy (list[int]): Стратегия действий для каждого состояния.
    verbose (bool): Выводить ли детальные промежуточные результаты.

    Возвращает:
    float: Средний доход от применения политики.
    numpy.ndarray: Стационарное распределение состояний.
    """
    # Инициализация матрицы переходов и вектора вознаграждений для данной по
    transition_matrix_policy = np.zeros((3, 3))
    reward_vector_policy = np.zeros(3)

    if verbose:
        print(f"\n{'-'*80}")
        policy_str = ", ".join([f"s{i}→a{a}({action_names[a]})" for i, a in
        print(f"Оценка политики: {policy} [{policy_str}]")
        print(f"{'-'*80}")

    # Для каждого состояния, вычисляем ожидаемое вознаграждение и вероятност
    for state in states:
        action = policy[state]
        transition_matrix_policy[state, :] = transition_matrix[action][state

        if verbose:
            print(f"\nСостояние {state} ({state_names[state]}), выбрано дейс
            print(f"  Вероятности переходов P[{state},:] = {transition_matri

        state_reward = 0
        for next_state in states:
            transition_prob = transition_matrix[action][state][next_state]
            reward = reward_matrix[action][state][next_state]
            contrib = transition_prob * reward
            state_reward += contrib

            if verbose:
                print(f"  Переход в {next_state} ({state_names[next_state]})
                      f"награда = {reward}, вклад = {contrib:.3f}")

        reward_vector_policy[state] = state_reward

        if verbose:
```

```
            print(f"  Суммарное ожидаемое вознаграждение r[{state}] = {state

    if verbose:
        print("\nМатрица переходов для политики:")
        for i in range(3):
            print(f"  {transition_matrix_policy[i, :]}")
        print("\nВектор вознаграждений для политики:")
        print(f"  {reward_vector_policy}")

    # Решение задачи с собственными значениями для нахождения стационарного
    eigenvalues, eigenvectors = np.linalg.eig(transition_matrix_policy.T)

    if verbose:
        print("\nСобственные значения матрицы переходов:")
        for i, ev in enumerate(eigenvalues):
            print(f"  λ{i} = {ev:.6f} (|λ{i} - 1| = {abs(ev - 1.0):.6f})")

    stationary_state_idx = np.argmin(np.abs(eigenvalues - 1.0))

    if verbose:
        print(f"\nИндекс собственного значения, ближайшего к 1: {stationary_
        print(f"Соответствующий собственный вектор (ненормированный):")
        print(f"  {np.real(eigenvectors[:, stationary_state_idx])}")

    stationary_distribution = np.real(eigenvectors[:, stationary_state_idx])

    # Нормировка стационарного распределения
    stationary_distribution /= stationary_distribution.sum()

    # Рассчитываем средний доход
    average_reward = float(np.dot(stationary_distribution, reward_vector_pol

    if verbose:
        print("\nСтационарное распределение состояний (нормированное):")
        for i, prob in enumerate(stationary_distribution):
            print(f"  μ({i}) = {prob:.6f} ({state_names[i]})")

        print("\nРасчет среднего дохода:")
        for i, (prob, reward) in enumerate(zip(stationary_distribution, rewa
            print(f"  Состояние {i}: μ({i}) * r({i}) = {prob:.6f} * {reward:

        print(f"\nСредний доход g = μ·r = {average_reward:.6f}")

    return average_reward, stationary_distribution

# Инициализация переменных для поиска лучшей политики
best_policy = None
best_gain = -1e9  # Начальное значение для максимального дохода
best_stationary_distribution = None

# Общее количество политик
total_policies = len(actions) ** len(states)
print(f"Всего возможных политик: {total_policies}")

# Перебор всех возможных политик и выбор лучшей
policy_count = 0
```

```
improvements = 0

for policy in itertools.product(actions, repeat=3):
    policy_count += 1
    policy = list(policy)  # Преобразование кортежа в список для удобства

    print(f"\n{'='*80}")
    print(f"Политика {policy_count}/{total_policies}: {policy} " +
          f"[{', '.join([f's{i}→{a}({action_names[a]})' for i, a in enumerat

    # Подробный вывод только для первой, лучшей и последней политики
    verbose = (policy_count == 1) or (policy_count == total_policies)
    gain, stationary_distribution = evaluate_policy(policy, verbose=verbose)

    print(f"\nПолитика {policy}: средний доход g = {gain:.6f}")

    if gain > best_gain:
        improvements += 1
        improvement = gain - best_gain
        print(f"УЛУЧШЕНИЕ #{improvements}: +{improvement:.6f} (было {best_ga
        best_gain = gain
        best_policy = policy
        best_stationary_distribution = stationary_distribution

print(f"\n\n\n{'='*40} Ответ {'='*40}")
print("Лучшая стационарная стратегия для состояний (0-Отличный, 1-Хороший, 2
policy_str = ", ".join([f"s{i}→a{a}({action_names[a]})" for i, a in enumerat
print(f"  {best_policy} [{policy_str}]")
print("\nСтационарное распределение состояний μ:")
for i, prob in enumerate(best_stationary_distribution):
    print(f"  μ({i}) = {prob:.6f} ({state_names[i]})")
print(f"\nСредний доход g при лучшей стратегии: {best_gain:.6f}")
print(f"Количество улучшений: {improvements} из {policy_count} проверенных г

print(f"\n\n\n{'='*40} Лучшая политика {'='*40}")
evaluate_policy(best_policy, verbose=True)
```

```
Всего возможных политик: 27

================================================================================
====
Политика 1/27: [0, 0, 0] [s0→0(3% скидка), s1→0(3% скидка), s2→0(3% скидка)]

--------------------------------------------------------------------------------
----
Оценка политики: [0, 0, 0] [s0→a0(3% скидка), s1→a0(3% скидка), s2→a0(3% ски
дка)]
--------------------------------------------------------------------------------
----

Состояние 0 (Отличный), выбрано действие 0 (3% скидка):
  Вероятности переходов P[0,:] = [0.3 0.5 0.2]
  Переход в 0 (Отличный): вероятность = 0.300, награда = 110, вклад = 33.000
  Переход в 1 (Хороший): вероятность = 0.500, награда = 100, вклад = 50.000
  Переход в 2 (Удовлетворительный): вероятность = 0.200, награда = 70, вклад
= 14.000
  Суммарное ожидаемое вознаграждение r[0] = 97.000

Состояние 1 (Хороший), выбрано действие 0 (3% скидка):
  Вероятности переходов P[1,:] = [0.2 0.6 0.2]
  Переход в 0 (Отличный): вероятность = 0.200, награда = 100, вклад = 20.000
  Переход в 1 (Хороший): вероятность = 0.600, награда = 80, вклад = 48.000
  Переход в 2 (Удовлетворительный): вероятность = 0.200, награда = 50, вклад
= 10.000
  Суммарное ожидаемое вознаграждение r[1] = 78.000

Состояние 2 (Удовлетворительный), выбрано действие 0 (3% скидка):
  Вероятности переходов P[2,:] = [0.1 0.2 0.7]
  Переход в 0 (Отличный): вероятность = 0.100, награда = 80, вклад = 8.000
  Переход в 1 (Хороший): вероятность = 0.200, награда = 60, вклад = 12.000
  Переход в 2 (Удовлетворительный): вероятность = 0.700, награда = 40, вклад
= 28.000
  Суммарное ожидаемое вознаграждение r[2] = 48.000

Матрица переходов для политики:
  [0.3 0.5 0.2]
  [0.2 0.6 0.2]
  [0.1 0.2 0.7]

Вектор вознаграждений для политики:
  [97. 78. 48.]

Собственные значения матрицы переходов:
  λ0 = 1.000000 (|λ0 - 1| = 0.000000)
  λ1 = 0.100000 (|λ1 - 1| = 0.900000)
  λ2 = 0.500000 (|λ2 - 1| = 0.500000)

Индекс собственного значения, ближайшего к 1: 0
Соответствующий собственный вектор (ненормированный):
  [-0.29231364 -0.69424489 -0.65770569]

Стационарное распределение состояний (нормированное):
  μ(0) = 0.177778 (Отличный)
```

```
  μ(1) = 0.422222 (Хороший)
  μ(2) = 0.400000 (Удовлетворительный)

Расчет среднего дохода:
  Состояние 0: μ(0) * r(0) = 0.177778 * 97.000 = 17.244444
  Состояние 1: μ(1) * r(1) = 0.422222 * 78.000 = 32.933333
  Состояние 2: μ(2) * r(2) = 0.400000 * 48.000 = 19.200000

Средний доход g = μ·r = 69.377778

Политика [0, 0, 0]: средний доход g = 69.377778
УЛУЧШЕНИЕ #1: +1000000069.377778 (было -1000000000.000000, стало 69.377778)

================================================================================
====
Политика 2/27: [0, 0, 1] [s0→0(3% скидка), s1→0(3% скидка), s2→1(Бесплатная
доставка)]

Политика [0, 0, 1]: средний доход g = 76.577778
УЛУЧШЕНИЕ #2: +7.200000 (было 69.377778, стало 76.577778)

================================================================================
====
Политика 3/27: [0, 0, 2] [s0→0(3% скидка), s1→0(3% скидка), s2→2(Ничего)]

Политика [0, 0, 2]: средний доход g = 77.185185
УЛУЧШЕНИЕ #3: +0.607407 (было 76.577778, стало 77.185185)

================================================================================
====
Политика 4/27: [0, 1, 0] [s0→0(3% скидка), s1→1(Бесплатная доставка), s2→0
(3% скидка)]

Политика [0, 1, 0]: средний доход g = 68.375000

================================================================================
====
Политика 5/27: [0, 1, 1] [s0→0(3% скидка), s1→1(Бесплатная доставка), s2→1(Б
есплатная доставка)]

Политика [0, 1, 1]: средний доход g = 78.781250
УЛУЧШЕНИЕ #4: +1.596065 (было 77.185185, стало 78.781250)

================================================================================
====
Политика 6/27: [0, 1, 2] [s0→0(3% скидка), s1→1(Бесплатная доставка), s2→2(Н
ичего)]

Политика [0, 1, 2]: средний доход g = 80.194444
УЛУЧШЕНИЕ #5: +1.413194 (было 78.781250, стало 80.194444)

================================================================================
====
Политика 7/27: [0, 2, 0] [s0→0(3% скидка), s1→2(Ничего), s2→0(3% скидка)]

Политика [0, 2, 0]: средний доход g = 63.466667
```

```
================================================================================
====
Политика 8/27: [0, 2, 1] [s0→0(3% скидка), s1→2(Ничего), s2→1(Бесплатная дос
тавка)]

Политика [0, 2, 1]: средний доход g = 70.666667

================================================================================
====
Политика 9/27: [0, 2, 2] [s0→0(3% скидка), s1→2(Ничего), s2→2(Ничего)]

Политика [0, 2, 2]: средний доход g = 70.444444

================================================================================
====
Политика 10/27: [1, 0, 0] [s0→1(Бесплатная доставка), s1→0(3% скидка), s2→0
(3% скидка)]

Политика [1, 0, 0]: средний доход g = 70.734694

================================================================================
====
Политика 11/27: [1, 0, 1] [s0→1(Бесплатная доставка), s1→0(3% скидка), s2→1
(Бесплатная доставка)]

Политика [1, 0, 1]: средний доход g = 77.346939

================================================================================
====
Политика 12/27: [1, 0, 2] [s0→1(Бесплатная доставка), s1→0(3% скидка), s2→2
(Ничего)]

Политика [1, 0, 2]: средний доход g = 77.932203

================================================================================
====
Политика 13/27: [1, 1, 0] [s0→1(Бесплатная доставка), s1→1(Бесплатная достав
ка), s2→0(3% скидка)]

Политика [1, 1, 0]: средний доход g = 69.222222

================================================================================
====
Политика 14/27: [1, 1, 1] [s0→1(Бесплатная доставка), s1→1(Бесплатная достав
ка), s2→1(Бесплатная доставка)]

Политика [1, 1, 1]: средний доход g = 79.472222

================================================================================
====
Политика 15/27: [1, 1, 2] [s0→1(Бесплатная доставка), s1→1(Бесплатная достав
ка), s2→2(Ничего)]

Политика [1, 1, 2]: средний доход g = 80.864198
УЛУЧШЕНИЕ #6: +0.669753 (было 80.194444, стало 80.864198)
```

```
================================================================================
====
Политика 16/27: [1, 2, 0] [s0→1(Бесплатная доставка), s1→2(Ничего), s2→0(3%
скидка)]

Политика [1, 2, 0]: средний доход g = 64.163265

================================================================================
====
Политика 17/27: [1, 2, 1] [s0→1(Бесплатная доставка), s1→2(Ничего), s2→1(Бес
платная доставка)]

Политика [1, 2, 1]: средний доход g = 70.775510

================================================================================
====
Политика 18/27: [1, 2, 2] [s0→1(Бесплатная доставка), s1→2(Ничего), s2→2(Нич
его)]

Политика [1, 2, 2]: средний доход g = 70.576271

================================================================================
====
Политика 19/27: [2, 0, 0] [s0→2(Ничего), s1→0(3% скидка), s2→0(3% скидка)]

Политика [2, 0, 0]: средний доход g = 65.304348

================================================================================
====
Политика 20/27: [2, 0, 1] [s0→2(Ничего), s1→0(3% скидка), s2→1(Бесплатная до
ставка)]

Политика [2, 0, 1]: средний доход g = 73.130435

================================================================================
====
Политика 21/27: [2, 0, 2] [s0→2(Ничего), s1→0(3% скидка), s2→2(Ничего)]

Политика [2, 0, 2]: средний доход g = 73.636364

================================================================================
====
Политика 22/27: [2, 1, 0] [s0→2(Ничего), s1→1(Бесплатная доставка), s2→0(3%
скидка)]

Политика [2, 1, 0]: средний доход g = 65.500000

================================================================================
====
Политика 23/27: [2, 1, 1] [s0→2(Ничего), s1→1(Бесплатная доставка), s2→1(Бес
платная доставка)]

Политика [2, 1, 1]: средний доход g = 76.187500

================================================================================
```

```
====
Политика 24/27: [2, 1, 2] [s0→2(Ничего), s1→1(Бесплатная доставка), s2→2(Нич
его)]

Политика [2, 1, 2]: средний доход g = 77.638889

================================================================================
====
Политика 25/27: [2, 2, 0] [s0→2(Ничего), s1→2(Ничего), s2→0(3% скидка)]

Политика [2, 2, 0]: средний доход g = 59.826087

================================================================================
====
Политика 26/27: [2, 2, 1] [s0→2(Ничего), s1→2(Ничего), s2→1(Бесплатная доста
вка)]

Политика [2, 2, 1]: средний доход g = 67.652174

================================================================================
====
Политика 27/27: [2, 2, 2] [s0→2(Ничего), s1→2(Ничего), s2→2(Ничего)]

--------------------------------------------------------------------------------
----
Оценка политики: [2, 2, 2] [s0→a2(Ничего), s1→a2(Ничего), s2→a2(Ничего)]
--------------------------------------------------------------------------------
----

Состояние 0 (Отличный), выбрано действие 2 (Ничего):
  Вероятности переходов P[0,:] = [0.3 0.4 0.3]
  Переход в 0 (Отличный): вероятность = 0.300, награда = 110, вклад = 33.000
  Переход в 1 (Хороший): вероятность = 0.400, награда = 80, вклад = 32.000
  Переход в 2 (Удовлетворительный): вероятность = 0.300, награда = 50, вклад
= 15.000
  Суммарное ожидаемое вознаграждение r[0] = 80.000

Состояние 1 (Хороший), выбрано действие 2 (Ничего):
  Вероятности переходов P[1,:] = [0.2 0.6 0.2]
  Переход в 0 (Отличный): вероятность = 0.200, награда = 100, вклад = 20.000
  Переход в 1 (Хороший): вероятность = 0.600, награда = 60, вклад = 36.000
  Переход в 2 (Удовлетворительный): вероятность = 0.200, награда = 40, вклад
= 8.000
  Суммарное ожидаемое вознаграждение r[1] = 64.000

Состояние 2 (Удовлетворительный), выбрано действие 2 (Ничего):
  Вероятности переходов P[2,:] = [0.1 0.3 0.6]
  Переход в 0 (Отличный): вероятность = 0.100, награда = 80, вклад = 8.000
  Переход в 1 (Хороший): вероятность = 0.300, награда = 70, вклад = 21.000
  Переход в 2 (Удовлетворительный): вероятность = 0.600, награда = 60, вклад
= 36.000
  Суммарное ожидаемое вознаграждение r[2] = 65.000

Матрица переходов для политики:
  [0.3 0.4 0.3]
  [0.2 0.6 0.2]
```

```
  [0.1 0.3 0.6]

Вектор вознаграждений для политики:
  [80. 64. 65.]

Собственные значения матрицы переходов:
  λ0 = 1.000000 (|λ0 - 1| = 0.000000)
  λ1 = 0.138197 (|λ1 - 1| = 0.861803)
  λ2 = 0.361803 (|λ2 - 1| = 0.638197)

Индекс собственного значения, ближайшего к 1: 0
Соответствующий собственный вектор (ненормированный):
  [0.2981424  0.74535599 0.59628479]

Стационарное распределение состояний (нормированное):
  μ(0) = 0.181818 (Отличный)
  μ(1) = 0.454545 (Хороший)
  μ(2) = 0.363636 (Удовлетворительный)

Расчет среднего дохода:
  Состояние 0: μ(0) * r(0) = 0.181818 * 80.000 = 14.545455
  Состояние 1: μ(1) * r(1) = 0.454545 * 64.000 = 29.090909
  Состояние 2: μ(2) * r(2) = 0.363636 * 65.000 = 23.636364

Средний доход g = μ·r = 67.272727

Политика [2, 2, 2]: средний доход g = 67.272727



======================================== Ответ ==============================
===========
Лучшая стационарная стратегия для состояний (0-Отличный, 1-Хороший, 2-Удовле
творительный):
  [1, 1, 2] [s0→a1(Бесплатная доставка), s1→a1(Бесплатная доставка), s2→a2(Н
ичего)]

Стационарное распределение состояний μ:
  μ(0) = 0.111111 (Отличный)
  μ(1) = 0.382716 (Хороший)
  μ(2) = 0.506173 (Удовлетворительный)

Средний доход g при лучшей стратегии: 80.864198
Количество улучшений: 6 из 27 проверенных политик



======================================== Лучшая политика ====================
====================

------------------------------------------------------------------------------
----
Оценка политики: [1, 1, 2] [s0→a1(Бесплатная доставка), s1→a1(Бесплатная дос
тавка), s2→a2(Ничего)]
------------------------------------------------------------------------------
----
```

```
Состояние 0 (Отличный), выбрано действие 1 (Бесплатная доставка):
  Вероятности переходов P[0,:] = [0.2 0.7 0.1]
  Переход в 0 (Отличный): вероятность = 0.200, награда = 120, вклад = 24.000
  Переход в 1 (Хороший): вероятность = 0.700, награда = 100, вклад = 70.000
  Переход в 2 (Удовлетворительный): вероятность = 0.100, награда = 70, вклад
= 7.000
  Суммарное ожидаемое вознаграждение r[0] = 101.000

Состояние 1 (Хороший), выбрано действие 1 (Бесплатная доставка):
  Вероятности переходов P[1,:] = [0.1 0.4 0.5]
  Переход в 0 (Отличный): вероятность = 0.100, награда = 110, вклад = 11.000
  Переход в 1 (Хороший): вероятность = 0.400, награда = 100, вклад = 40.000
  Переход в 2 (Удовлетворительный): вероятность = 0.500, награда = 90, вклад
= 45.000
  Суммарное ожидаемое вознаграждение r[1] = 96.000

Состояние 2 (Удовлетворительный), выбрано действие 2 (Ничего):
  Вероятности переходов P[2,:] = [0.1 0.3 0.6]
  Переход в 0 (Отличный): вероятность = 0.100, награда = 80, вклад = 8.000
  Переход в 1 (Хороший): вероятность = 0.300, награда = 70, вклад = 21.000
  Переход в 2 (Удовлетворительный): вероятность = 0.600, награда = 60, вклад
= 36.000
  Суммарное ожидаемое вознаграждение r[2] = 65.000

Матрица переходов для политики:
  [0.2 0.7 0.1]
  [0.1 0.4 0.5]
  [0.1 0.3 0.6]

Вектор вознаграждений для политики:
  [101.  96.  65.]

Собственные значения матрицы переходов:
  λ0 = 1.000000 (|λ0 - 1| = 0.000000)
  λ1 = 0.100000 (|λ1 - 1| = 0.900000)
  λ2 = 0.100000 (|λ2 - 1| = 0.900000)

Индекс собственного значения, ближайшего к 1: 0
Соответствующий собственный вектор (ненормированный):
  [0.17247204 0.59407034 0.78570594]

Стационарное распределение состояний (нормированное):
  μ(0) = 0.111111 (Отличный)
  μ(1) = 0.382716 (Хороший)
  μ(2) = 0.506173 (Удовлетворительный)

Расчет среднего дохода:
  Состояние 0: μ(0) * r(0) = 0.111111 * 101.000 = 11.222222
  Состояние 1: μ(1) * r(1) = 0.382716 * 96.000 = 36.740741
  Состояние 2: μ(2) * r(2) = 0.506173 * 65.000 = 32.901235

Средний доход g = μ·r = 80.864198
```

Out[74]: (80.8641975308642, array([0.11111111, 0.38271605, 0.50617284]))

---

3 Метод итерации по стратегиям без дисконтирования

In [75]:
```python
import numpy as np

# 1) Задаём P и R по условию задачи
P = {
    0: [[0.3, 0.5, 0.2],
        [0.2, 0.6, 0.2],
        [0.1, 0.2, 0.7]],
    1: [[0.2, 0.7, 0.1],
        [0.1, 0.4, 0.5],
        [0.1, 0.2, 0.7]],
    2: [[0.3, 0.4, 0.3],
        [0.2, 0.6, 0.2],
        [0.1, 0.3, 0.6]],
}

R = {
    0: [[110, 100, 70],
        [100,  80, 50],
        [ 80,  60, 40]],
    1: [[120, 100, 70],
        [110, 100, 90],
        [100,  70, 60]],
    2: [[110,  80, 50],
        [100,  60, 40],
        [ 80,  70, 60]],
}

num_states = 3
actions = [0, 1, 2]  # 0=скидка, 1=доставка, 2=ничего
state_names = ["Отличный", "Хороший", "Удовлетворительный"]
action_names = ["3% скидка", "Бесплатная доставка", "Ничего"]

def evaluate_policy(policy, verbose=True):
    """Policy evaluation (gain g и bias h)"""
    m = num_states
    # 1) Собираем P_pi и r_pi
    P_pi = np.zeros((m, m))
    r_pi = np.zeros(m)

    if verbose:
        print("\n" + "="*70)
        print(f"ОЦЕНКА ПОЛИТИКИ: {policy} ({[action_names[a] for a in policy
        print("="*70)
        print("\n1) Собираем P_π и r_π на основе текущей политики:")

    for i in range(m):
        a = policy[i]
        if verbose:
            print(f"\n  Состояние {i} ({state_names[i]}) → действие {a} ({ac
```

```
        # Заполнение строк матрицы переходов P_pi
        for j in range(m):
            P_pi[i, j] = P[a][i][j]
            r_pi[i] += P[a][i][j] * R[a][i][j]

            if verbose:
                print(f"    Переход в состояние {j} ({state_names[j]}): P[{a
                      f"R[{a}][{i}][{j}] = {R[a][i][j]}, вклад в r_π[{i}] +=

    if verbose:
        print("\n  Итоговая матрица переходов P_π:")
        for i in range(m):
            print(f"    {P_pi[i, :]}")
        print("\n  Итоговый вектор вознаграждений r_π:")
        print(f"    {r_pi}")

    # 2) Строим и решаем систему (m+1)x(m+1) на [h0..h_{m-1}, g]
    A = np.zeros((m + 1, m + 1))
    b = np.zeros(m + 1)

    if verbose:
        print("\n2) Строим систему уравнений (m+1)×(m+1) для нахождения [h₀,
        print("   (I - P_π)h + g·1 = r_π, с дополнительным условием h[m-1] =

    for i in range(m):
        A[i, i] = 1.0
        A[i, :m] -= P_pi[i, :]
        A[i, m] = 1.0
        b[i] = r_pi[i]

        if verbose:
            eq_parts = []
            for j in range(m):
                if j == i:
                    coef = 1.0 - P_pi[i][j]
                else:
                    coef = -P_pi[i][j]

                if abs(coef) > 1e-6:  # Если коэффициент не слишком мал
                    sign = '+' if coef >= 0 and j > 0 else ''
                    eq_parts.append(f"{sign}{coef:.3f}·h[{j}]")

            eq_parts.append("+ g")
            eq = " ".join(eq_parts)
            print(f"    Уравнение {i+1}: {eq} = {r_pi[i]:.3f}")

    # фиксация h[m-1] = 0
    A[m, m - 1] = 1.0
    b[m] = 0.0

    if verbose:
        print(f"    Дополнительное условие: h[{m-1}] = 0")
        print("\n  Матрица системы A:")
        for i in range(m+1):
            print(f"    {A[i, :]}")
        print("\n  Вектор правых частей b:")
```

```
        print(f"    {b}")

    # Решаем систему
    x = np.linalg.solve(A, b)
    h = x[:m]
    g = x[m]

    if verbose:
        print("\n3) Решение системы уравнений:")
        print(f"  Вектор смещений h = {h}")
        print(f"  Средний доход g = {g:.6f}")

        # Проверка решения
        print("\n4) Проверка решения (должно быть приблизительно равно r_π):
        for i in range(m):
            check_sum = 0
            for j in range(m):
                check_sum += P_pi[i, j] * h[j]
            check_val = h[i] - check_sum + g
            error = abs(check_val - r_pi[i])
            print(f"  Уравнение {i+1}: h[{i}] - ∑_j P_π[{i},{j}]·h[j] + g =

    return g, h

def improve_policy(policy, h, verbose=True):
    """Policy improvement"""
    m = num_states
    new_pol = policy.copy()

    if verbose:
        print(f"\n" + "="*40 + "Улучшение политики" + "="*40)
        print(f"\nТекущая политика: {policy} ({[action_names[a] for a in pol
        print(f"Вектор смещений h = {h}")

    for i in range(m):
        if verbose:
            print(f"\nДля состояния {i} ({state_names[i]}) ищем оптимальное

        best_q, best_a = -1e9, None
        for a in actions:
            # считаем Q(i,a) = r(i,a) + sum_j P[a][i][j]·h[j]
            r_ia = 0.0
            bias_term = 0.0

            for j in range(m):
                r_ia += P[a][i][j] * R[a][i][j]
                bias_term += P[a][i][j] * h[j]

            q = r_ia + bias_term

            if verbose:
                print(f"  Действие {a} ({action_names[a]}):")
                # Детально показываем расчет r(i,a)
                print(f"    r({i},{a}) = ", end="")
                for j in range(m):
                    print(f"P[{a}][{i}][{j}]·R[{a}][{i}][{j}] = {P[a][i][j]:
```

```
                    if j < m-1:
                        print(" + ", end="")
                print(f" = {r_ia:.3f}")

                # Детально показываем расчет bias-терма
                print(f"    Bias term = ", end="")
                for j in range(m):
                    print(f"P[{a}][{i}][{j}]·h[{j}] = {P[a][i][j]:.3f}·{h[j]
                    if j < m-1:
                        print(" + ", end="")
                print(f" = {bias_term:.3f}")

                # Итоговый Q-value
                print(f"    Q({i},{a}) = {r_ia:.3f} + {bias_term:.3f} = {q:.

            if q > best_q:
                if verbose and best_a is not None:
                    print(f"    Лучше предыдущего действия {best_a} с Q = {b
                best_q, best_a = q, a
            elif verbose:
                print(f"    Хуже текущего лучшего действия {best_a} с Q = {b

        new_pol[i] = best_a

        if verbose:
            action_changed = policy[i] != new_pol[i]
            print(f"  Лучшее действие для состояния {i}: {best_a} ({action_r
            if action_changed:
                print(f"  >>> Политика ИЗМЕНЕНА для состояния {i}: {policy[i
            else:
                print(f"  >>> Политика НЕ ИЗМЕНЕНА для состояния {i}: остает

    if verbose:
        print("\nИтог улучшения политики:")
        print(f"  Было: {policy} ({[action_names[a] for a in policy]})")
        print(f"  Стало: {new_pol} ({[action_names[a] for a in new_pol]})")
        if policy == new_pol:
            print("  Политика НЕ ИЗМЕНИЛАСЬ - достигнута оптимальная политик
        else:
            print("  Политика ИЗМЕНИЛАСЬ - требуется продолжить итерации.")

    return new_pol

def policy_iteration():
    policy = [0] * num_states  # стартуем, например, всегда со "скидки"
    iteration = 0

    print("\nНачальная политика:")
    for i, a in enumerate(policy):
        print(f"  Состояние {i} ({state_names[i]}) → Действие {a} ({action_r

    while True:
        print(f"\n{'*'*40} Итерация {iteration+1} {'*'*40}")

        print(f"\nТекущая политика:")
        for i, a in enumerate(policy):
```

```
            print(f"  {i} ({state_names[i]}) → {a} ({action_names[a]})")

        # Оценка этой политики
        g, h = evaluate_policy(policy)
        print(f"\nОценка текущей политики:")
        print(f"  Средний доход g = {g:.4f}")

        # Улучшаем политику
        new_pol = improve_policy(policy, h)

        # Если не изменилось — готово
        if new_pol == policy:
            print("\n" + "="*80)
            print("Политика не изменилась. Алгоритм завершён.")
            print("="*80)
            break

        policy = new_pol
        iteration += 1

    return policy, g, h

if __name__ == "__main__":
    opt_policy, opt_gain, opt_h = policy_iteration()
    print(f"\n\n\n{'='*40} Ответ {'='*40}")

    for i, a in enumerate(opt_policy):
        print(f"  Состояние {i} ({state_names[i]}) → Действие {a} ({action_r
    print(f"\nОптимальный средний доход g* = {opt_gain:.4f}")
    print("="*80)
```

```
Начальная политика:
  Состояние 0 (Отличный) → Действие 0 (3% скидка)
  Состояние 1 (Хороший) → Действие 0 (3% скидка)
  Состояние 2 (Удовлетворительный) → Действие 0 (3% скидка)

************************************* Итерация 1 ************************
***************

Текущая политика:
  0 (Отличный) → 0 (3% скидка)
  1 (Хороший) → 0 (3% скидка)
  2 (Удовлетворительный) → 0 (3% скидка)

========================================================================
ОЦЕНКА ПОЛИТИКИ: [0, 0, 0] (['3% скидка', '3% скидка', '3% скидка'])
========================================================================

1) Собираем P_π и r_π на основе текущей политики:

  Состояние 0 (Отличный) → действие 0 (3% скидка):
    Переход в состояние 0 (Отличный): P[0][0][0] = 0.300, R[0][0][0] = 110,
вклад в r_π[0] += 33.000
    Переход в состояние 1 (Хороший): P[0][0][1] = 0.500, R[0][0][1] = 100, в
клад в r_π[0] += 50.000
    Переход в состояние 2 (Удовлетворительный): P[0][0][2] = 0.200, R[0][0]
[2] = 70, вклад в r_π[0] += 14.000

  Состояние 1 (Хороший) → действие 0 (3% скидка):
    Переход в состояние 0 (Отличный): P[0][1][0] = 0.200, R[0][1][0] = 100,
вклад в r_π[1] += 20.000
    Переход в состояние 1 (Хороший): P[0][1][1] = 0.600, R[0][1][1] = 80, вк
лад в r_π[1] += 48.000
    Переход в состояние 2 (Удовлетворительный): P[0][1][2] = 0.200, R[0][1]
[2] = 50, вклад в r_π[1] += 10.000

  Состояние 2 (Удовлетворительный) → действие 0 (3% скидка):
    Переход в состояние 0 (Отличный): P[0][2][0] = 0.100, R[0][2][0] = 80, в
клад в r_π[2] += 8.000
    Переход в состояние 1 (Хороший): P[0][2][1] = 0.200, R[0][2][1] = 60, вк
лад в r_π[2] += 12.000
    Переход в состояние 2 (Удовлетворительный): P[0][2][2] = 0.700, R[0][2]
[2] = 40, вклад в r_π[2] += 28.000

  Итоговая матрица переходов P_π:
    [0.3 0.5 0.2]
    [0.2 0.6 0.2]
    [0.1 0.2 0.7]

  Итоговый вектор вознаграждений r_π:
    [97. 78. 48.]

2) Строим систему уравнений (m+1)×(m+1) для нахождения [h₀, h₁, h₂, g]:
   (I - P_π)h + g·1 = r_π, с дополнительным условием h[m-1] = 0
    Уравнение 1: 0.700·h[0] -0.500·h[1] -0.200·h[2] + g = 97.000
    Уравнение 2: -0.200·h[0] +0.400·h[1] -0.200·h[2] + g = 78.000
    Уравнение 3: -0.100·h[0] -0.200·h[1] +0.300·h[2] + g = 48.000
```

```
    Дополнительное условие: h[2] = 0

  Матрица системы A:
    [ 0.7 -0.5 -0.2  1. ]
    [-0.2  0.4 -0.2  1. ]
    [-0.1 -0.2  0.3  1. ]
    [0. 0. 1. 0.]

  Вектор правых частей b:
    [97. 78. 48.  0.]

3) Решение системы уравнений:
  Вектор смещений h = [85.33333333 64.22222222  0.        ]
  Средний доход g = 69.377778

4) Проверка решения (должно быть приблизительно равно r_π):
  Уравнение 1: h[0] - ∑_j P_π[0,2]·h[j] + g = 97.000000, r_π[0] = 97.000000,
ошибка = 0.000000e+00
  Уравнение 2: h[1] - ∑_j P_π[1,2]·h[j] + g = 78.000000, r_π[1] = 78.000000,
ошибка = 0.000000e+00
  Уравнение 3: h[2] - ∑_j P_π[2,2]·h[j] + g = 48.000000, r_π[2] = 48.000000,
ошибка = 0.000000e+00

Оценка текущей политики:
  Средний доход g = 69.3778

========================================Улучшение политики=================
======================

Текущая политика: [0, 0, 0] (['3% скидка', '3% скидка', '3% скидка'])
Вектор смещений h = [85.33333333 64.22222222  0.        ]

Для состояния 0 (Отличный) ищем оптимальное действие:
  Действие 0 (3% скидка):
    r(0,0) = P[0][0][0]·R[0][0][0] = 0.300·110 = 33.000 + P[0][0][1]·R[0][0]
[1] = 0.500·100 = 50.000 + P[0][0][2]·R[0][0][2] = 0.200·70 = 14.000 = 97.00
0
    Bias term = P[0][0][0]·h[0] = 0.300·85.333 = 25.600 + P[0][0][1]·h[1] =
0.500·64.222 = 32.111 + P[0][0][2]·h[2] = 0.200·0.000 = 0.000 = 57.711
    Q(0,0) = 97.000 + 57.711 = 154.711
  Действие 1 (Бесплатная доставка):
    r(0,1) = P[1][0][0]·R[1][0][0] = 0.200·120 = 24.000 + P[1][0][1]·R[1][0]
[1] = 0.700·100 = 70.000 + P[1][0][2]·R[1][0][2] = 0.100·70 = 7.000 = 101.00
0
    Bias term = P[1][0][0]·h[0] = 0.200·85.333 = 17.067 + P[1][0][1]·h[1] =
0.700·64.222 = 44.956 + P[1][0][2]·h[2] = 0.100·0.000 = 0.000 = 62.022
    Q(0,1) = 101.000 + 62.022 = 163.022
    Лучше предыдущего действия 0 с Q = 154.711, обновляем.
  Действие 2 (Ничего):
    r(0,2) = P[2][0][0]·R[2][0][0] = 0.300·110 = 33.000 + P[2][0][1]·R[2][0]
[1] = 0.400·80 = 32.000 + P[2][0][2]·R[2][0][2] = 0.300·50 = 15.000 = 80.000
    Bias term = P[2][0][0]·h[0] = 0.300·85.333 = 25.600 + P[2][0][1]·h[1] =
0.400·64.222 = 25.689 + P[2][0][2]·h[2] = 0.300·0.000 = 0.000 = 51.289
    Q(0,2) = 80.000 + 51.289 = 131.289
    Хуже текущего лучшего действия 1 с Q = 163.022, пропускаем.
  Лучшее действие для состояния 0: 1 (Бесплатная доставка), Q = 163.022
```

```
  >>> Политика ИЗМЕНЕНА для состояния 0: 0 → 1

Для состояния 1 (Хороший) ищем оптимальное действие:
  Действие 0 (3% скидка):
    r(1,0) = P[0][1][0]·R[0][1][0] = 0.200·100 = 20.000 + P[0][1][1]·R[0][1]
[1] = 0.600·80 = 48.000 + P[0][1][2]·R[0][1][2] = 0.200·50 = 10.000 = 78.000
    Bias term = P[0][1][0]·h[0] = 0.200·85.333 = 17.067 + P[0][1][1]·h[1] =
0.600·64.222 = 38.533 + P[0][1][2]·h[2] = 0.200·0.000 = 0.000 = 55.600
    Q(1,0) = 78.000 + 55.600 = 133.600
  Действие 1 (Бесплатная доставка):
    r(1,1) = P[1][1][0]·R[1][1][0] = 0.100·110 = 11.000 + P[1][1][1]·R[1][1]
[1] = 0.400·100 = 40.000 + P[1][1][2]·R[1][1][2] = 0.500·90 = 45.000 = 96.00
0
    Bias term = P[1][1][0]·h[0] = 0.100·85.333 = 8.533 + P[1][1][1]·h[1] =
0.400·64.222 = 25.689 + P[1][1][2]·h[2] = 0.500·0.000 = 0.000 = 34.222
    Q(1,1) = 96.000 + 34.222 = 130.222
    Хуже текущего лучшего действия 0 с Q = 133.600, пропускаем.
  Действие 2 (Ничего):
    r(1,2) = P[2][1][0]·R[2][1][0] = 0.200·100 = 20.000 + P[2][1][1]·R[2][1]
[1] = 0.600·60 = 36.000 + P[2][1][2]·R[2][1][2] = 0.200·40 = 8.000 = 64.000
    Bias term = P[2][1][0]·h[0] = 0.200·85.333 = 17.067 + P[2][1][1]·h[1] =
0.600·64.222 = 38.533 + P[2][1][2]·h[2] = 0.200·0.000 = 0.000 = 55.600
    Q(1,2) = 64.000 + 55.600 = 119.600
    Хуже текущего лучшего действия 0 с Q = 133.600, пропускаем.
  Лучшее действие для состояния 1: 0 (3% скидка), Q = 133.600
  >>> Политика НЕ ИЗМЕНЕНА для состояния 1: остается 0

Для состояния 2 (Удовлетворительный) ищем оптимальное действие:
  Действие 0 (3% скидка):
    r(2,0) = P[0][2][0]·R[0][2][0] = 0.100·80 = 8.000 + P[0][2][1]·R[0][2]
[1] = 0.200·60 = 12.000 + P[0][2][2]·R[0][2][2] = 0.700·40 = 28.000 = 48.000
    Bias term = P[0][2][0]·h[0] = 0.100·85.333 = 8.533 + P[0][2][1]·h[1] =
0.200·64.222 = 12.844 + P[0][2][2]·h[2] = 0.700·0.000 = 0.000 = 21.378
    Q(2,0) = 48.000 + 21.378 = 69.378
  Действие 1 (Бесплатная доставка):
    r(2,1) = P[1][2][0]·R[1][2][0] = 0.100·100 = 10.000 + P[1][2][1]·R[1][2]
[1] = 0.200·70 = 14.000 + P[1][2][2]·R[1][2][2] = 0.700·60 = 42.000 = 66.000
    Bias term = P[1][2][0]·h[0] = 0.100·85.333 = 8.533 + P[1][2][1]·h[1] =
0.200·64.222 = 12.844 + P[1][2][2]·h[2] = 0.700·0.000 = 0.000 = 21.378
    Q(2,1) = 66.000 + 21.378 = 87.378
    Лучше предыдущего действия 0 с Q = 69.378, обновляем.
  Действие 2 (Ничего):
    r(2,2) = P[2][2][0]·R[2][2][0] = 0.100·80 = 8.000 + P[2][2][1]·R[2][2]
[1] = 0.300·70 = 21.000 + P[2][2][2]·R[2][2][2] = 0.600·60 = 36.000 = 65.000
    Bias term = P[2][2][0]·h[0] = 0.100·85.333 = 8.533 + P[2][2][1]·h[1] =
0.300·64.222 = 19.267 + P[2][2][2]·h[2] = 0.600·0.000 = 0.000 = 27.800
    Q(2,2) = 65.000 + 27.800 = 92.800
    Лучше предыдущего действия 1 с Q = 87.378, обновляем.
  Лучшее действие для состояния 2: 2 (Ничего), Q = 92.800
  >>> Политика ИЗМЕНЕНА для состояния 2: 0 → 2

Итог улучшения политики:
  Было: [0, 0, 0] (['3% скидка', '3% скидка', '3% скидка'])
  Стало: [1, 0, 2] (['Бесплатная доставка', '3% скидка', 'Ничего'])
  Политика ИЗМЕНИЛАСЬ - требуется продолжить итерации.
```

```
************************************* Итерация 2 ************************
***************

Текущая политика:
  0 (Отличный) → 1 (Бесплатная доставка)
  1 (Хороший) → 0 (3% скидка)
  2 (Удовлетворительный) → 2 (Ничего)

======================================================================
ОЦЕНКА ПОЛИТИКИ: [1, 0, 2] (['Бесплатная доставка', '3% скидка', 'Ничего'])
======================================================================

1) Собираем P_π и r_π на основе текущей политики:

  Состояние 0 (Отличный) → действие 1 (Бесплатная доставка):
    Переход в состояние 0 (Отличный): P[1][0][0] = 0.200, R[1][0][0] = 120,
вклад в r_π[0] += 24.000
    Переход в состояние 1 (Хороший): P[1][0][1] = 0.700, R[1][0][1] = 100, в
клад в r_π[0] += 70.000
    Переход в состояние 2 (Удовлетворительный): P[1][0][2] = 0.100, R[1][0]
[2] = 70, вклад в r_π[0] += 7.000

  Состояние 1 (Хороший) → действие 0 (3% скидка):
    Переход в состояние 0 (Отличный): P[0][1][0] = 0.200, R[0][1][0] = 100,
вклад в r_π[1] += 20.000
    Переход в состояние 1 (Хороший): P[0][1][1] = 0.600, R[0][1][1] = 80, вк
лад в r_π[1] += 48.000
    Переход в состояние 2 (Удовлетворительный): P[0][1][2] = 0.200, R[0][1]
[2] = 50, вклад в r_π[1] += 10.000

  Состояние 2 (Удовлетворительный) → действие 2 (Ничего):
    Переход в состояние 0 (Отличный): P[2][2][0] = 0.100, R[2][2][0] = 80, в
клад в r_π[2] += 8.000
    Переход в состояние 1 (Хороший): P[2][2][1] = 0.300, R[2][2][1] = 70, вк
лад в r_π[2] += 21.000
    Переход в состояние 2 (Удовлетворительный): P[2][2][2] = 0.600, R[2][2]
[2] = 60, вклад в r_π[2] += 36.000

  Итоговая матрица переходов P_π:
    [0.2 0.7 0.1]
    [0.2 0.6 0.2]
    [0.1 0.3 0.6]

  Итоговый вектор вознаграждений r_π:
    [101.  78.  65.]

2) Строим систему уравнений (m+1)×(m+1) для нахождения [h₀, h₁, h₂, g]:
   (I - P_π)h + g·1 = r_π, с дополнительным условием h[m-1] = 0
    Уравнение 1: 0.800·h[0] -0.700·h[1] -0.100·h[2] + g = 101.000
    Уравнение 2: -0.200·h[0] +0.400·h[1] -0.200·h[2] + g = 78.000
    Уравнение 3: -0.100·h[0] -0.300·h[1] +0.400·h[2] + g = 65.000
    Дополнительное условие: h[2] = 0

  Матрица системы A:
    [ 0.8 -0.7 -0.1  1. ]
    [-0.2  0.4 -0.2  1. ]
```

```
    [-0.1 -0.3  0.4  1. ]
    [0. 0. 1. 0.]

  Вектор правых частей b:
    [101.  78.  65.   0.]

3) Решение системы уравнений:
  Вектор смещений h = [51.52542373 25.93220339  0.        ]
  Средний доход g = 77.932203

4) Проверка решения (должно быть приблизительно равно r_π):
  Уравнение 1: h[0] - ∑_j P_π[0,2]·h[j] + g = 101.000000, r_π[0] = 101.00000
0, ошибка = 0.000000e+00
  Уравнение 2: h[1] - ∑_j P_π[1,2]·h[j] + g = 78.000000, r_π[1] = 78.000000,
ошибка = 2.842171e-14
  Уравнение 3: h[2] - ∑_j P_π[2,2]·h[j] + g = 65.000000, r_π[2] = 65.000000,
ошибка = 0.000000e+00

Оценка текущей политики:
  Средний доход g = 77.9322

========================================Улучшение политики=================
======================

Текущая политика: [1, 0, 2] (['Бесплатная доставка', '3% скидка', 'Ничего'])
Вектор смещений h = [51.52542373 25.93220339  0.        ]

Для состояния 0 (Отличный) ищем оптимальное действие:
  Действие 0 (3% скидка):
    r(0,0) = P[0][0][0]·R[0][0][0] = 0.300·110 = 33.000 + P[0][0][1]·R[0][0]
[1] = 0.500·100 = 50.000 + P[0][0][2]·R[0][0][2] = 0.200·70 = 14.000 = 97.00
0
    Bias term = P[0][0][0]·h[0] = 0.300·51.525 = 15.458 + P[0][0][1]·h[1] =
0.500·25.932 = 12.966 + P[0][0][2]·h[2] = 0.200·0.000 = 0.000 = 28.424
    Q(0,0) = 97.000 + 28.424 = 125.424
  Действие 1 (Бесплатная доставка):
    r(0,1) = P[1][0][0]·R[1][0][0] = 0.200·120 = 24.000 + P[1][0][1]·R[1][0]
[1] = 0.700·100 = 70.000 + P[1][0][2]·R[1][0][2] = 0.100·70 = 7.000 = 101.00
0
    Bias term = P[1][0][0]·h[0] = 0.200·51.525 = 10.305 + P[1][0][1]·h[1] =
0.700·25.932 = 18.153 + P[1][0][2]·h[2] = 0.100·0.000 = 0.000 = 28.458
    Q(0,1) = 101.000 + 28.458 = 129.458
    Лучше предыдущего действия 0 с Q = 125.424, обновляем.
  Действие 2 (Ничего):
    r(0,2) = P[2][0][0]·R[2][0][0] = 0.300·110 = 33.000 + P[2][0][1]·R[2][0]
[1] = 0.400·80 = 32.000 + P[2][0][2]·R[2][0][2] = 0.300·50 = 15.000 = 80.000
    Bias term = P[2][0][0]·h[0] = 0.300·51.525 = 15.458 + P[2][0][1]·h[1] =
0.400·25.932 = 10.373 + P[2][0][2]·h[2] = 0.300·0.000 = 0.000 = 25.831
    Q(0,2) = 80.000 + 25.831 = 105.831
    Хуже текущего лучшего действия 1 с Q = 129.458, пропускаем.
  Лучшее действие для состояния 0: 1 (Бесплатная доставка), Q = 129.458
  >>> Политика НЕ ИЗМЕНЕНА для состояния 0: остается 1

Для состояния 1 (Хороший) ищем оптимальное действие:
  Действие 0 (3% скидка):
    r(1,0) = P[0][1][0]·R[0][1][0] = 0.200·100 = 20.000 + P[0][1][1]·R[0][1]
```

```
[1] = 0.600·80 = 48.000 + P[0][1][2]·R[0][1][2] = 0.200·50 = 10.000 = 78.000
    Bias term = P[0][1][0]·h[0] = 0.200·51.525 = 10.305 + P[0][1][1]·h[1] =
0.600·25.932 = 15.559 + P[0][1][2]·h[2] = 0.200·0.000 = 0.000 = 25.864
    Q(1,0) = 78.000 + 25.864 = 103.864
  Действие 1 (Бесплатная доставка):
    r(1,1) = P[1][1][0]·R[1][1][0] = 0.100·110 = 11.000 + P[1][1][1]·R[1][1]
[1] = 0.400·100 = 40.000 + P[1][1][2]·R[1][1][2] = 0.500·90 = 45.000 = 96.00
0
    Bias term = P[1][1][0]·h[0] = 0.100·51.525 = 5.153 + P[1][1][1]·h[1] =
0.400·25.932 = 10.373 + P[1][1][2]·h[2] = 0.500·0.000 = 0.000 = 15.525
    Q(1,1) = 96.000 + 15.525 = 111.525
    Лучше предыдущего действия 0 с Q = 103.864, обновляем.
  Действие 2 (Ничего):
    r(1,2) = P[2][1][0]·R[2][1][0] = 0.200·100 = 20.000 + P[2][1][1]·R[2][1]
[1] = 0.600·60 = 36.000 + P[2][1][2]·R[2][1][2] = 0.200·40 = 8.000 = 64.000
    Bias term = P[2][1][0]·h[0] = 0.200·51.525 = 10.305 + P[2][1][1]·h[1] =
0.600·25.932 = 15.559 + P[2][1][2]·h[2] = 0.200·0.000 = 0.000 = 25.864
    Q(1,2) = 64.000 + 25.864 = 89.864
    Хуже текущего лучшего действия 1 с Q = 111.525, пропускаем.
  Лучшее действие для состояния 1: 1 (Бесплатная доставка), Q = 111.525
  >>> Политика ИЗМЕНЕНА для состояния 1: 0 → 1

Для состояния 2 (Удовлетворительный) ищем оптимальное действие:
  Действие 0 (3% скидка):
    r(2,0) = P[0][2][0]·R[0][2][0] = 0.100·80 = 8.000 + P[0][2][1]·R[0][2]
[1] = 0.200·60 = 12.000 + P[0][2][2]·R[0][2][2] = 0.700·40 = 28.000 = 48.000
    Bias term = P[0][2][0]·h[0] = 0.100·51.525 = 5.153 + P[0][2][1]·h[1] =
0.200·25.932 = 5.186 + P[0][2][2]·h[2] = 0.700·0.000 = 0.000 = 10.339
    Q(2,0) = 48.000 + 10.339 = 58.339
  Действие 1 (Бесплатная доставка):
    r(2,1) = P[1][2][0]·R[1][2][0] = 0.100·100 = 10.000 + P[1][2][1]·R[1][2]
[1] = 0.200·70 = 14.000 + P[1][2][2]·R[1][2][2] = 0.700·60 = 42.000 = 66.000
    Bias term = P[1][2][0]·h[0] = 0.100·51.525 = 5.153 + P[1][2][1]·h[1] =
0.200·25.932 = 5.186 + P[1][2][2]·h[2] = 0.700·0.000 = 0.000 = 10.339
    Q(2,1) = 66.000 + 10.339 = 76.339
    Лучше предыдущего действия 0 с Q = 58.339, обновляем.
  Действие 2 (Ничего):
    r(2,2) = P[2][2][0]·R[2][2][0] = 0.100·80 = 8.000 + P[2][2][1]·R[2][2]
[1] = 0.300·70 = 21.000 + P[2][2][2]·R[2][2][2] = 0.600·60 = 36.000 = 65.000
    Bias term = P[2][2][0]·h[0] = 0.100·51.525 = 5.153 + P[2][2][1]·h[1] =
0.300·25.932 = 7.780 + P[2][2][2]·h[2] = 0.600·0.000 = 0.000 = 12.932
    Q(2,2) = 65.000 + 12.932 = 77.932
    Лучше предыдущего действия 1 с Q = 76.339, обновляем.
  Лучшее действие для состояния 2: 2 (Ничего), Q = 77.932
  >>> Политика НЕ ИЗМЕНЕНА для состояния 2: остается 2

Итог улучшения политики:
  Было: [1, 0, 2] (['Бесплатная доставка', '3% скидка', 'Ничего'])
  Стало: [1, 1, 2] (['Бесплатная доставка', 'Бесплатная доставка', 'Ничег
o'])
  Политика ИЗМЕНИЛАСЬ - требуется продолжить итерации.

************************************* Итерация 3 ************************
***************

Текущая политика:
```

```
  0 (Отличный) → 1 (Бесплатная доставка)
  1 (Хороший) → 1 (Бесплатная доставка)
  2 (Удовлетворительный) → 2 (Ничего)

======================================================================
ОЦЕНКА ПОЛИТИКИ: [1, 1, 2] (['Бесплатная доставка', 'Бесплатная доставка',
'Ничего'])
======================================================================

1) Собираем P_π и r_π на основе текущей политики:

  Состояние 0 (Отличный) → действие 1 (Бесплатная доставка):
    Переход в состояние 0 (Отличный): P[1][0][0] = 0.200, R[1][0][0] = 120,
вклад в r_π[0] += 24.000
    Переход в состояние 1 (Хороший): P[1][0][1] = 0.700, R[1][0][1] = 100, в
клад в r_π[0] += 70.000
    Переход в состояние 2 (Удовлетворительный): P[1][0][2] = 0.100, R[1][0]
[2] = 70, вклад в r_π[0] += 7.000

  Состояние 1 (Хороший) → действие 1 (Бесплатная доставка):
    Переход в состояние 0 (Отличный): P[1][1][0] = 0.100, R[1][1][0] = 110,
вклад в r_π[1] += 11.000
    Переход в состояние 1 (Хороший): P[1][1][1] = 0.400, R[1][1][1] = 100, в
клад в r_π[1] += 40.000
    Переход в состояние 2 (Удовлетворительный): P[1][1][2] = 0.500, R[1][1]
[2] = 90, вклад в r_π[1] += 45.000

  Состояние 2 (Удовлетворительный) → действие 2 (Ничего):
    Переход в состояние 0 (Отличный): P[2][2][0] = 0.100, R[2][2][0] = 80, в
клад в r_π[2] += 8.000
    Переход в состояние 1 (Хороший): P[2][2][1] = 0.300, R[2][2][1] = 70, вк
лад в r_π[2] += 21.000
    Переход в состояние 2 (Удовлетворительный): P[2][2][2] = 0.600, R[2][2]
[2] = 60, вклад в r_π[2] += 36.000

  Итоговая матрица переходов P_π:
    [0.2 0.7 0.1]
    [0.1 0.4 0.5]
    [0.1 0.3 0.6]

  Итоговый вектор вознаграждений r_π:
    [101.  96.  65.]

2) Строим систему уравнений (m+1)×(m+1) для нахождения [h₀, h₁, h₂, g]:
   (I - P_π)h + g·1 = r_π, с дополнительным условием h[m-1] = 0
    Уравнение 1: 0.800·h[0] -0.700·h[1] -0.100·h[2] + g = 101.000
    Уравнение 2: -0.100·h[0] +0.600·h[1] -0.500·h[2] + g = 96.000
    Уравнение 3: -0.100·h[0] -0.300·h[1] +0.400·h[2] + g = 65.000
    Дополнительное условие: h[2] = 0

  Матрица системы A:
    [ 0.8 -0.7 -0.1  1. ]
    [-0.1  0.6 -0.5  1. ]
    [-0.1 -0.3  0.4  1. ]
    [0. 0. 1. 0.]
```

```
  Вектор правых частей b:
    [101.  96.  65.   0.]

3) Решение системы уравнений:
  Вектор смещений h = [55.30864198 34.44444444  0.        ]
  Средний доход g = 80.864198

4) Проверка решения (должно быть приблизительно равно r_π):
  Уравнение 1: h[0] - ∑_j P_π[0,2]·h[j] + g = 101.000000, r_π[0] = 101.00000
0, ошибка = 0.000000e+00
  Уравнение 2: h[1] - ∑_j P_π[1,2]·h[j] + g = 96.000000, r_π[1] = 96.000000,
ошибка = 0.000000e+00
  Уравнение 3: h[2] - ∑_j P_π[2,2]·h[j] + g = 65.000000, r_π[2] = 65.000000,
ошибка = 1.421085e-14

Оценка текущей политики:
  Средний доход g = 80.8642

========================================Улучшение политики==================
======================

Текущая политика: [1, 1, 2] (['Бесплатная доставка', 'Бесплатная доставка',
'Ничего'])
Вектор смещений h = [55.30864198 34.44444444  0.        ]

Для состояния 0 (Отличный) ищем оптимальное действие:
  Действие 0 (3% скидка):
    r(0,0) = P[0][0][0]·R[0][0][0] = 0.300·110 = 33.000 + P[0][0][1]·R[0][0]
[1] = 0.500·100 = 50.000 + P[0][0][2]·R[0][0][2] = 0.200·70 = 14.000 = 97.00
0
    Bias term = P[0][0][0]·h[0] = 0.300·55.309 = 16.593 + P[0][0][1]·h[1] =
0.500·34.444 = 17.222 + P[0][0][2]·h[2] = 0.200·0.000 = 0.000 = 33.815
    Q(0,0) = 97.000 + 33.815 = 130.815
  Действие 1 (Бесплатная доставка):
    r(0,1) = P[1][0][0]·R[1][0][0] = 0.200·120 = 24.000 + P[1][0][1]·R[1][0]
[1] = 0.700·100 = 70.000 + P[1][0][2]·R[1][0][2] = 0.100·70 = 7.000 = 101.00
0
    Bias term = P[1][0][0]·h[0] = 0.200·55.309 = 11.062 + P[1][0][1]·h[1] =
0.700·34.444 = 24.111 + P[1][0][2]·h[2] = 0.100·0.000 = 0.000 = 35.173
    Q(0,1) = 101.000 + 35.173 = 136.173
    Лучше предыдущего действия 0 с Q = 130.815, обновляем.
  Действие 2 (Ничего):
    r(0,2) = P[2][0][0]·R[2][0][0] = 0.300·110 = 33.000 + P[2][0][1]·R[2][0]
[1] = 0.400·80 = 32.000 + P[2][0][2]·R[2][0][2] = 0.300·50 = 15.000 = 80.000
    Bias term = P[2][0][0]·h[0] = 0.300·55.309 = 16.593 + P[2][0][1]·h[1] =
0.400·34.444 = 13.778 + P[2][0][2]·h[2] = 0.300·0.000 = 0.000 = 30.370
    Q(0,2) = 80.000 + 30.370 = 110.370
    Хуже текущего лучшего действия 1 с Q = 136.173, пропускаем.
  Лучшее действие для состояния 0: 1 (Бесплатная доставка), Q = 136.173
  >>> Политика НЕ ИЗМЕНЕНА для состояния 0: остается 1

Для состояния 1 (Хороший) ищем оптимальное действие:
  Действие 0 (3% скидка):
    r(1,0) = P[0][1][0]·R[0][1][0] = 0.200·100 = 20.000 + P[0][1][1]·R[0][1]
[1] = 0.600·80 = 48.000 + P[0][1][2]·R[0][1][2] = 0.200·50 = 10.000 = 78.000
    Bias term = P[0][1][0]·h[0] = 0.200·55.309 = 11.062 + P[0][1][1]·h[1] =
```

```
0.600·34.444 = 20.667 + P[0][1][2]·h[2] = 0.200·0.000 = 0.000 = 31.728
    Q(1,0) = 78.000 + 31.728 = 109.728
  Действие 1 (Бесплатная доставка):
    r(1,1) = P[1][1][0]·R[1][1][0] = 0.100·110 = 11.000 + P[1][1][1]·R[1][1]
[1] = 0.400·100 = 40.000 + P[1][1][2]·R[1][1][2] = 0.500·90 = 45.000 = 96.00
0
    Bias term = P[1][1][0]·h[0] = 0.100·55.309 = 5.531 + P[1][1][1]·h[1] =
0.400·34.444 = 13.778 + P[1][1][2]·h[2] = 0.500·0.000 = 0.000 = 19.309
    Q(1,1) = 96.000 + 19.309 = 115.309
    Лучше предыдущего действия 0 с Q = 109.728, обновляем.
  Действие 2 (Ничего):
    r(1,2) = P[2][1][0]·R[2][1][0] = 0.200·100 = 20.000 + P[2][1][1]·R[2][1]
[1] = 0.600·60 = 36.000 + P[2][1][2]·R[2][1][2] = 0.200·40 = 8.000 = 64.000
    Bias term = P[2][1][0]·h[0] = 0.200·55.309 = 11.062 + P[2][1][1]·h[1] =
0.600·34.444 = 20.667 + P[2][1][2]·h[2] = 0.200·0.000 = 0.000 = 31.728
    Q(1,2) = 64.000 + 31.728 = 95.728
    Хуже текущего лучшего действия 1 с Q = 115.309, пропускаем.
  Лучшее действие для состояния 1: 1 (Бесплатная доставка), Q = 115.309
  >>> Политика НЕ ИЗМЕНЕНА для состояния 1: остается 1

Для состояния 2 (Удовлетворительный) ищем оптимальное действие:
  Действие 0 (3% скидка):
    r(2,0) = P[0][2][0]·R[0][2][0] = 0.100·80 = 8.000 + P[0][2][1]·R[0][2]
[1] = 0.200·60 = 12.000 + P[0][2][2]·R[0][2][2] = 0.700·40 = 28.000 = 48.000
    Bias term = P[0][2][0]·h[0] = 0.100·55.309 = 5.531 + P[0][2][1]·h[1] =
0.200·34.444 = 6.889 + P[0][2][2]·h[2] = 0.700·0.000 = 0.000 = 12.420
    Q(2,0) = 48.000 + 12.420 = 60.420
  Действие 1 (Бесплатная доставка):
    r(2,1) = P[1][2][0]·R[1][2][0] = 0.100·100 = 10.000 + P[1][2][1]·R[1][2]
[1] = 0.200·70 = 14.000 + P[1][2][2]·R[1][2][2] = 0.700·60 = 42.000 = 66.000
    Bias term = P[1][2][0]·h[0] = 0.100·55.309 = 5.531 + P[1][2][1]·h[1] =
0.200·34.444 = 6.889 + P[1][2][2]·h[2] = 0.700·0.000 = 0.000 = 12.420
    Q(2,1) = 66.000 + 12.420 = 78.420
    Лучше предыдущего действия 0 с Q = 60.420, обновляем.
  Действие 2 (Ничего):
    r(2,2) = P[2][2][0]·R[2][2][0] = 0.100·80 = 8.000 + P[2][2][1]·R[2][2]
[1] = 0.300·70 = 21.000 + P[2][2][2]·R[2][2][2] = 0.600·60 = 36.000 = 65.000
    Bias term = P[2][2][0]·h[0] = 0.100·55.309 = 5.531 + P[2][2][1]·h[1] =
0.300·34.444 = 10.333 + P[2][2][2]·h[2] = 0.600·0.000 = 0.000 = 15.864
    Q(2,2) = 65.000 + 15.864 = 80.864
    Лучше предыдущего действия 1 с Q = 78.420, обновляем.
  Лучшее действие для состояния 2: 2 (Ничего), Q = 80.864
  >>> Политика НЕ ИЗМЕНЕНА для состояния 2: остается 2

Итог улучшения политики:
  Было: [1, 1, 2] (['Бесплатная доставка', 'Бесплатная доставка', 'Ничего'])
  Стало: [1, 1, 2] (['Бесплатная доставка', 'Бесплатная доставка', 'Ничег
о'])
  Политика НЕ ИЗМЕНИЛАСЬ - достигнута оптимальная политика.

================================================================================
====
Политика не изменилась. Алгоритм завершён.
================================================================================
====
```

```
======================================== Ответ ============================
===========
  Состояние 0 (Отличный) → Действие 1 (Бесплатная доставка)
  Состояние 1 (Хороший) → Действие 1 (Бесплатная доставка)
  Состояние 2 (Удовлетворительный) → Действие 2 (Ничего)

Оптимальный средний доход g* = 80.8642
================================================================================
====
```

---

4 Метод итерации по стратегии с дисконтированием

In [76]:

```python
import numpy as np

# 1) Задаём P и R по условию
P = {
    0: [[0.3, 0.5, 0.2],
        [0.2, 0.6, 0.2],
        [0.1, 0.2, 0.7]],
    1: [[0.2, 0.7, 0.1],
        [0.1, 0.4, 0.5],
        [0.1, 0.2, 0.7]],
    2: [[0.3, 0.4, 0.3],
        [0.2, 0.6, 0.2],
        [0.1, 0.3, 0.6]],
}

R = {
    0: [[110, 100, 70],
        [100,  80, 50],
        [ 80,  60, 40]],
    1: [[120, 100, 70],
        [110, 100, 90],
        [100,  70, 60]],
    2: [[110,  80, 50],
        [100,  60, 40],
        [ 80,  70, 60]],
}

num_states = 3
actions = [0, 1, 2]  # 0=скидка, 1=доставка, 2=ничего
state_names = ["Отл.", "Хор.", "Уд."]
action_names = ["3% скидка", "доставка", "ничего"]

γ = 0.7  # коэффициент дисконтирования

def evaluate_policy_discount(policy, gamma=γ, verbose=True):
    """
    Решаем (I - γ Pπ) V = rπ
```

```
    возвращаем вектор V размера m.
    """
    m = num_states
    Pπ = np.zeros((m, m))
    rπ = np.zeros(m)

    if verbose:
        print("\n" + "="*40 + "ОЦЕНКА ПОЛИТИКИ" + "="*40)
        print(f"Действия политики: {[action_names[a] for a in policy]}")
        print("\n1) Создаём матрицу переходов Pπ и вектор вознаграждений rπ

    for i in range(m):
        a = policy[i]
        if verbose:
            print(f"\n  Состояние {i} ({state_names[i]}) → действие {a} ({ac

        for j in range(m):
            Pπ[i, j] = P[a][i][j]
            rπ[i] += P[a][i][j] * R[a][i][j]

            if verbose:
                print(f"    Переход в {j} ({state_names[j]}): P[{a}][{i}][{j
                      f"R[{a}][{i}][{j}] = {R[a][i][j]}, вклад в rπ[{i}]: {P

    if verbose:
        print("\n  Итоговая матрица переходов Pπ:")
        for i in range(m):
            print(f"    {Pπ[i, :]}")

        print("\n  Итоговый вектор вознаграждений rπ:")
        print(f"    {rπ}")

        print("\n2) Формируем систему уравнений (I - γ·Pπ)·V = rπ:")

    # матрица I - γ·Pπ
    A = np.eye(m) - gamma * Pπ

    if verbose:
        print(f"\n  Матрица γ·Pπ (γ = {gamma}):")
        for i in range(m):
            print(f"    {gamma * Pπ[i, :]}")

        print("\n  Матрица системы A = I - γ·Pπ:")
        for i in range(m):
            print(f"    {A[i, :]}")

        print("\n  Система уравнений:")
        for i in range(m):
            eq_parts = []
            for j in range(m):
                if abs(A[i, j]) > 1e-10:
                    sign = "+" if A[i, j] > 0 and j > 0 else ""
                    eq_parts.append(f"{sign}{A[i, j]:.3f}·V[{j}]")

            eq = " ".join(eq_parts)
            print(f"    {eq} = {rπ[i]:.3f}")
```

```
    # Решаем систему
    V = np.linalg.solve(A, rπ)

    if verbose:
        print("\n3) Решение системы дает значения функции ценности V:")
        for i in range(m):
            print(f"    V[{i}] ({state_names[i]}) = {V[i]:.6f}")

        # Проверка решения
        print("\n4) Проверка решения (A·V должно быть ≈ rπ):")
        for i in range(m):
            check_val = 0
            for j in range(m):
                check_val += A[i, j] * V[j]
            error = abs(check_val - rπ[i])
            print(f"    Уравнение {i+1}: {check_val:.6f} ≈ {rπ[i]:.6f}, ошиб

    return V

def improve_policy_discount(policy, V, gamma=γ, verbose=True):
    """
    Для каждого состояния i находим действие a, максимизирующее
    Q(i,a) = r(i,a) + γ ∑_j P[a][i][j]·V[j]
    """
    m = num_states
    new_pol = policy.copy()

    if verbose:
        print("\n" + "="*40 + "Улучшение политики" + "="*40)
        print(f"\nТекущая политика: {policy} ({[action_names[a] for a in pol
        print(f"Текущие значения V: {np.round(V, 4)}")

    for i in range(m):
        if verbose:
            print(f"\nДля состояния {i} ({state_names[i]}) ищем оптимальное

        best_q, best_a = -float('inf'), None
        q_values = []

        for a in actions:
            # Вычисляем Q(i,a) = r(i,a) + γ·∑_j P[a][i][j]·V[j]
            # 1. Мгновенное вознаграждение r(i,a)
            r_ia = 0.0
            for j in range(m):
                r_ia += P[a][i][j] * R[a][i][j]

            # 2. Дисконтированное будущее вознаграждение
            future_val = 0.0
            for j in range(m):
                future_val += gamma * P[a][i][j] * V[j]

            # 3. Суммарное Q-значение
            q = r_ia + future_val
            q_values.append(q)
```

```
            if verbose:
                print(f"  Действие {a} ({action_names[a]}):")

                # Детально показываем вычисление r(i,a)
                print(f"    r({i},{a}) = ", end="")
                for j in range(m):
                    print(f"P[{a}][{i}][{j}]·R[{a}][{i}][{j}] = {P[a][i][j]:
                    if j < m-1:
                        print(" + ", end="")
                print(f" = {r_ia:.3f}")

                # Детально показываем вычисление дисконтированного будущего
                print(f"    γ·∑_j P[{a}][{i}][j]·V[j] = ", end="")
                for j in range(m):
                    print(f"γ·P[{a}][{i}][{j}]·V[{j}] = {gamma:.1f}·{P[a][i]
                    if j < m-1:
                        print(" + ", end="")
                print(f" = {future_val:.3f}")

                # Итоговое Q-значение
                print(f"    Q({i},{a}) = {r_ia:.3f} + {future_val:.3f} = {q:

            if q > best_q:
                if verbose and best_a is not None:
                    print(f"    Лучше предыдущего действия {best_a} с Q={bes
                best_q, best_a = q, a
            elif verbose:
                print(f"    Хуже текущего лучшего действия {best_a} с Q={bes

        # Определяем лучшее действие для этого состояния
        new_pol[i] = best_a

        if verbose:
            action_changed = new_pol[i] != policy[i]
            print(f"  Лучшее действие для состояния {i}: {best_a} ({action_r
            print(f"  Q-значения всех действий: {np.round(q_values, 3)}")

            if action_changed:
                print(f"  >>> Политика ИЗМЕНЕНА для состояния {i}: {policy[i
            else:
                print(f"  >>> Политика НЕ ИЗМЕНЕНА для состояния {i}: остает

    if verbose:
        print("\nИтог улучшения политики:")
        print(f"  Было: {policy} ({[action_names[a] for a in policy]})")
        print(f"  Стало: {new_pol} ({[action_names[a] for a in new_pol]})")
        if policy == new_pol:
            print("  Политика НЕ ИЗМЕНИЛАСЬ - достигнута оптимальная политик
        else:
            print("  Политика ИЗМЕНИЛАСЬ - требуется продолжить итерации.")

    return new_pol

def policy_iteration_discount():
    # стартуем, например, всегда «3% скидка»
    policy = [0] * num_states
```

```
    it = 0

    print(f"\nКоэффициент дисконтирования γ = {γ}")
    print("\nНачальная политика:")
    for i, a in enumerate(policy):
        print(f"  Состояние {i} ({state_names[i]}) → Действие {a} ({action_r

    while True:
        print(f"\n{'*'*40 } Итерация {it+1} {'*'*40}")

        print(f"\nТекущая политика:")
        for i, a in enumerate(policy):
            print(f"  {i} ({state_names[i]}) → {a} ({action_names[a]})")

        # оценка
        V = evaluate_policy_discount(policy)
        print(f"\nРезультат оценки политики:")
        print(f"  V = {np.round(V, 3)}")

        # улучшение
        new_pol = improve_policy_discount(policy, V)

        # Проверка на сходимость
        if new_pol == policy:
            print("\n" + "="*80)
            print("Политика не изменилась. Алгоритм завершён.")
            print("="*80)
            break

        # Информация об изменении политики
        print("\nИзменения в политике:")
        for i in range(num_states):
            if new_pol[i] != policy[i]:
                print(f"  Состояние {i} ({state_names[i]}): {policy[i]} ({ac

        policy = new_pol
        it += 1

    return policy, V

if __name__ == "__main__":
    opt_pol, opt_V = policy_iteration_discount()

    print("\n" + "="*40 + " Ответ " + "="*40)

    print("\nОптимальная политика:")
    for i, a in enumerate(opt_pol):
        print(f"  Состояние {i} ({state_names[i]}) → Действие {a} ({action_r

    print("\nОптимальная функция ценности:")
    for i, v in enumerate(opt_V):
        print(f"  V*[{i}] ({state_names[i]}) = {v:.6f}")

    print("\nСтоимость состояний V* в округлённом виде:")
    print(f"  {np.round(opt_V, 3)}")
```

```
Коэффициент дисконтирования γ = 0.7

Начальная политика:
  Состояние 0 (Отл.) → Действие 0 (3% скидка)
  Состояние 1 (Хор.) → Действие 0 (3% скидка)
  Состояние 2 (Уд.) → Действие 0 (3% скидка)

************************************ Итерация 1 ************************
***************

Текущая политика:
  0 (Отл.) → 0 (3% скидка)
  1 (Хор.) → 0 (3% скидка)
  2 (Уд.) → 0 (3% скидка)

=======================================ОЦЕНКА ПОЛИТИКИ====================
==================
Действия политики: ['3% скидка', '3% скидка', '3% скидка']

1) Создаём матрицу переходов Pπ и вектор вознаграждений rπ для текущей полит
ики:

  Состояние 0 (Отл.) → действие 0 (3% скидка):
    Переход в 0 (Отл.): P[0][0][0] = 0.300, R[0][0][0] = 110, вклад в rπ[0]:
33.000
    Переход в 1 (Хор.): P[0][0][1] = 0.500, R[0][0][1] = 100, вклад в rπ[0]:
50.000
    Переход в 2 (Уд.): P[0][0][2] = 0.200, R[0][0][2] = 70, вклад в rπ[0]: 1
4.000

  Состояние 1 (Хор.) → действие 0 (3% скидка):
    Переход в 0 (Отл.): P[0][1][0] = 0.200, R[0][1][0] = 100, вклад в rπ[1]:
20.000
    Переход в 1 (Хор.): P[0][1][1] = 0.600, R[0][1][1] = 80, вклад в rπ[1]:
48.000
    Переход в 2 (Уд.): P[0][1][2] = 0.200, R[0][1][2] = 50, вклад в rπ[1]: 1
0.000

  Состояние 2 (Уд.) → действие 0 (3% скидка):
    Переход в 0 (Отл.): P[0][2][0] = 0.100, R[0][2][0] = 80, вклад в rπ[2]:
8.000
    Переход в 1 (Хор.): P[0][2][1] = 0.200, R[0][2][1] = 60, вклад в rπ[2]:
12.000
    Переход в 2 (Уд.): P[0][2][2] = 0.700, R[0][2][2] = 40, вклад в rπ[2]: 2
8.000

  Итоговая матрица переходов Pπ:
    [0.3 0.5 0.2]
    [0.2 0.6 0.2]
    [0.1 0.2 0.7]

  Итоговый вектор вознаграждений rπ:
    [97. 78. 48.]

2) Формируем систему уравнений (I - γ·Pπ)·V = rπ:
```

```
  Матрица γ·Pπ (γ = 0.7):
    [0.21 0.35 0.14]
    [0.14 0.42 0.14]
    [0.07 0.14 0.49]

  Матрица системы A = I - γ·Pπ:
    [ 0.79 -0.35 -0.14]
    [-0.14  0.58 -0.14]
    [-0.07 -0.14  0.51]

  Система уравнений:
    0.790·V[0] -0.350·V[1] -0.140·V[2] = 97.000
    -0.140·V[0] +0.580·V[1] -0.140·V[2] = 78.000
    -0.070·V[0] -0.140·V[1] +0.510·V[2] = 48.000

3) Решение системы дает значения функции ценности V:
    V[0] (Отл.) = 267.398952
    V[1] (Хор.) = 246.968845
    V[2] (Уд.) = 198.614833

4) Проверка решения (A·V должно быть ≈ rπ):
    Уравнение 1: 97.000000 ≈ 97.000000, ошибка = 1.421085e-14
    Уравнение 2: 78.000000 ≈ 78.000000, ошибка = 1.421085e-14
    Уравнение 3: 48.000000 ≈ 48.000000, ошибка = 0.000000e+00

Результат оценки политики:
  V = [267.399 246.969 198.615]

========================================Улучшение политики==================
======================

Текущая политика: [0, 0, 0] (['3% скидка', '3% скидка', '3% скидка'])
Текущие значения V: [267.399  246.9688 198.6148]

Для состояния 0 (Отл.) ищем оптимальное действие:
  Действие 0 (3% скидка):
    r(0,0) = P[0][0][0]·R[0][0][0] = 0.300·110 = 33.000 + P[0][0][1]·R[0][0]
[1] = 0.500·100 = 50.000 + P[0][0][2]·R[0][0][2] = 0.200·70 = 14.000 = 97.00
0
    γ·∑_j P[0][0][j]·V[j] = γ·P[0][0][0]·V[0] = 0.7·0.300·267.399 = 56.154 +
γ·P[0][0][1]·V[1] = 0.7·0.500·246.969 = 86.439 + γ·P[0][0][2]·V[2] = 0.7·0.2
00·198.615 = 27.806 = 170.399
    Q(0,0) = 97.000 + 170.399 = 267.399
  Действие 1 (доставка):
    r(0,1) = P[1][0][0]·R[1][0][0] = 0.200·120 = 24.000 + P[1][0][1]·R[1][0]
[1] = 0.700·100 = 70.000 + P[1][0][2]·R[1][0][2] = 0.100·70 = 7.000 = 101.00
0
    γ·∑_j P[1][0][j]·V[j] = γ·P[1][0][0]·V[0] = 0.7·0.200·267.399 = 37.436 +
γ·P[1][0][1]·V[1] = 0.7·0.700·246.969 = 121.015 + γ·P[1][0][2]·V[2] = 0.7·0.
100·198.615 = 13.903 = 172.354
    Q(0,1) = 101.000 + 172.354 = 273.354
    Лучше предыдущего действия 0 с Q=267.399, обновляем.
  Действие 2 (ничего):
    r(0,2) = P[2][0][0]·R[2][0][0] = 0.300·110 = 33.000 + P[2][0][1]·R[2][0]
[1] = 0.400·80 = 32.000 + P[2][0][2]·R[2][0][2] = 0.300·50 = 15.000 = 80.000
    γ·∑_j P[2][0][j]·V[j] = γ·P[2][0][0]·V[0] = 0.7·0.300·267.399 = 56.154 +
```

```
γ·P[2][0][1]·V[1] = 0.7·0.400·246.969 = 69.151 + γ·P[2][0][2]·V[2] = 0.7·0.3
00·198.615 = 41.709 = 167.014
    Q(0,2) = 80.000 + 167.014 = 247.014
    Хуже текущего лучшего действия 1 с Q=273.354, пропускаем.
  Лучшее действие для состояния 0: 1 (доставка), Q=273.354
  Q-значения всех действий: [267.399 273.354 247.014]
  >>> Политика ИЗМЕНЕНА для состояния 0: 0 → 1

Для состояния 1 (Хор.) ищем оптимальное действие:
  Действие 0 (3% скидка):
    r(1,0) = P[0][1][0]·R[0][1][0] = 0.200·100 = 20.000 + P[0][1][1]·R[0][1]
[1] = 0.600·80 = 48.000 + P[0][1][2]·R[0][1][2] = 0.200·50 = 10.000 = 78.000
    γ·∑_j P[0][1][j]·V[j] = γ·P[0][1][0]·V[0] = 0.7·0.200·267.399 = 37.436 +
γ·P[0][1][1]·V[1] = 0.7·0.600·246.969 = 103.727 + γ·P[0][1][2]·V[2] = 0.7·0.
200·198.615 = 27.806 = 168.969
    Q(1,0) = 78.000 + 168.969 = 246.969
  Действие 1 (доставка):
    r(1,1) = P[1][1][0]·R[1][1][0] = 0.100·110 = 11.000 + P[1][1][1]·R[1][1]
[1] = 0.400·100 = 40.000 + P[1][1][2]·R[1][1][2] = 0.500·90 = 45.000 = 96.00
0
    γ·∑_j P[1][1][j]·V[j] = γ·P[1][1][0]·V[0] = 0.7·0.100·267.399 = 18.718 +
γ·P[1][1][1]·V[1] = 0.7·0.400·246.969 = 69.151 + γ·P[1][1][2]·V[2] = 0.7·0.5
00·198.615 = 69.515 = 157.384
    Q(1,1) = 96.000 + 157.384 = 253.384
    Лучше предыдущего действия 0 с Q=246.969, обновляем.
  Действие 2 (ничего):
    r(1,2) = P[2][1][0]·R[2][1][0] = 0.200·100 = 20.000 + P[2][1][1]·R[2][1]
[1] = 0.600·60 = 36.000 + P[2][1][2]·R[2][1][2] = 0.200·40 = 8.000 = 64.000
    γ·∑_j P[2][1][j]·V[j] = γ·P[2][1][0]·V[0] = 0.7·0.200·267.399 = 37.436 +
γ·P[2][1][1]·V[1] = 0.7·0.600·246.969 = 103.727 + γ·P[2][1][2]·V[2] = 0.7·0.
200·198.615 = 27.806 = 168.969
    Q(1,2) = 64.000 + 168.969 = 232.969
    Хуже текущего лучшего действия 1 с Q=253.384, пропускаем.
  Лучшее действие для состояния 1: 1 (доставка), Q=253.384
  Q-значения всех действий: [246.969 253.384 232.969]
  >>> Политика ИЗМЕНЕНА для состояния 1: 0 → 1

Для состояния 2 (Уд.) ищем оптимальное действие:
  Действие 0 (3% скидка):
    r(2,0) = P[0][2][0]·R[0][2][0] = 0.100·80 = 8.000 + P[0][2][1]·R[0][2]
[1] = 0.200·60 = 12.000 + P[0][2][2]·R[0][2][2] = 0.700·40 = 28.000 = 48.000
    γ·∑_j P[0][2][j]·V[j] = γ·P[0][2][0]·V[0] = 0.7·0.100·267.399 = 18.718 +
γ·P[0][2][1]·V[1] = 0.7·0.200·246.969 = 34.576 + γ·P[0][2][2]·V[2] = 0.7·0.7
00·198.615 = 97.321 = 150.615
    Q(2,0) = 48.000 + 150.615 = 198.615
  Действие 1 (доставка):
    r(2,1) = P[1][2][0]·R[1][2][0] = 0.100·100 = 10.000 + P[1][2][1]·R[1][2]
[1] = 0.200·70 = 14.000 + P[1][2][2]·R[1][2][2] = 0.700·60 = 42.000 = 66.000
    γ·∑_j P[1][2][j]·V[j] = γ·P[1][2][0]·V[0] = 0.7·0.100·267.399 = 18.718 +
γ·P[1][2][1]·V[1] = 0.7·0.200·246.969 = 34.576 + γ·P[1][2][2]·V[2] = 0.7·0.7
00·198.615 = 97.321 = 150.615
    Q(2,1) = 66.000 + 150.615 = 216.615
    Лучше предыдущего действия 0 с Q=198.615, обновляем.
  Действие 2 (ничего):
    r(2,2) = P[2][2][0]·R[2][2][0] = 0.100·80 = 8.000 + P[2][2][1]·R[2][2]
[1] = 0.300·70 = 21.000 + P[2][2][2]·R[2][2][2] = 0.600·60 = 36.000 = 65.000
```

```
    γ·∑_j P[2][2][j]·V[j] = γ·P[2][2][0]·V[0] = 0.7·0.100·267.399 = 18.718 +
γ·P[2][2][1]·V[1] = 0.7·0.300·246.969 = 51.863 + γ·P[2][2][2]·V[2] = 0.7·0.6
00·198.615 = 83.418 = 154.000
    Q(2,2) = 65.000 + 154.000 = 219.000
    Лучше предыдущего действия 1 с Q=216.615, обновляем.
  Лучшее действие для состояния 2: 2 (ничего), Q=219.000
  Q-значения всех действий: [198.615 216.615 219.   ]
  >>> Политика ИЗМЕНЕНА для состояния 2: 0 → 2

Итог улучшения политики:
  Было: [0, 0, 0] (['3% скидка', '3% скидка', '3% скидка'])
  Стало: [1, 1, 2] (['доставка', 'доставка', 'ничего'])
  Политика ИЗМЕНИЛАСЬ - требуется продолжить итерации.

Изменения в политике:
  Состояние 0 (Отл.): 0 (3% скидка) → 1 (доставка)
  Состояние 1 (Хор.): 0 (3% скидка) → 1 (доставка)
  Состояние 2 (Уд.): 0 (3% скидка) → 2 (ничего)

************************************* Итерация 2 ************************
***************

Текущая политика:
  0 (Отл.) → 1 (доставка)
  1 (Хор.) → 1 (доставка)
  2 (Уд.) → 2 (ничего)

=========================================ОЦЕНКА ПОЛИТИКИ====================
==================
Действия политики: ['доставка', 'доставка', 'ничего']

1) Создаём матрицу переходов Pπ и вектор вознаграждений rπ для текущей полит
ики:

  Состояние 0 (Отл.) → действие 1 (доставка):
    Переход в 0 (Отл.): P[1][0][0] = 0.200, R[1][0][0] = 120, вклад в rπ[0]:
24.000
    Переход в 1 (Хор.): P[1][0][1] = 0.700, R[1][0][1] = 100, вклад в rπ[0]:
70.000
    Переход в 2 (Уд.): P[1][0][2] = 0.100, R[1][0][2] = 70, вклад в rπ[0]:
7.000

  Состояние 1 (Хор.) → действие 1 (доставка):
    Переход в 0 (Отл.): P[1][1][0] = 0.100, R[1][1][0] = 110, вклад в rπ[1]:
11.000
    Переход в 1 (Хор.): P[1][1][1] = 0.400, R[1][1][1] = 100, вклад в rπ[1]:
40.000
    Переход в 2 (Уд.): P[1][1][2] = 0.500, R[1][1][2] = 90, вклад в rπ[1]: 4
5.000

  Состояние 2 (Уд.) → действие 2 (ничего):
    Переход в 0 (Отл.): P[2][2][0] = 0.100, R[2][2][0] = 80, вклад в rπ[2]:
8.000
    Переход в 1 (Хор.): P[2][2][1] = 0.300, R[2][2][1] = 70, вклад в rπ[2]:
21.000
    Переход в 2 (Уд.): P[2][2][2] = 0.600, R[2][2][2] = 60, вклад в rπ[2]: 3
```

```
6.000

  Итоговая матрица переходов Pπ:
    [0.2 0.7 0.1]
    [0.1 0.4 0.5]
    [0.1 0.3 0.6]

  Итоговый вектор вознаграждений rπ:
    [101.  96.  65.]

2) Формируем систему уравнений (I - γ·Pπ)·V = rπ:

  Матрица γ·Pπ (γ = 0.7):
    [0.14 0.49 0.07]
    [0.07 0.28 0.35]
    [0.07 0.21 0.42]

  Матрица системы A = I - γ·Pπ:
    [ 0.86 -0.49 -0.07]
    [-0.07  0.72 -0.35]
    [-0.07 -0.21  0.58]

  Система уравнений:
    0.860·V[0] -0.490·V[1] -0.070·V[2] = 101.000
    -0.070·V[0] +0.720·V[1] -0.350·V[2] = 96.000
    -0.070·V[0] -0.210·V[1] +0.580·V[2] = 65.000

3) Решение системы дает значения функции ценности V:
    V[0] (Отл.) = 300.119474
    V[1] (Хор.) = 284.707288
    V[2] (Уд.) = 251.373955

4) Проверка решения (A·V должно быть ≈ rπ):
    Уравнение 1: 101.000000 ≈ 101.000000, ошибка = 1.421085e-14
    Уравнение 2: 96.000000 ≈ 96.000000, ошибка = 1.421085e-14
    Уравнение 3: 65.000000 ≈ 65.000000, ошибка = 0.000000e+00

Результат оценки политики:
  V = [300.119 284.707 251.374]

========================================Улучшение политики==================
======================

Текущая политика: [1, 1, 2] (['доставка', 'доставка', 'ничего'])
Текущие значения V: [300.1195 284.7073 251.374 ]

Для состояния 0 (Отл.) ищем оптимальное действие:
  Действие 0 (3% скидка):
    r(0,0) = P[0][0][0]·R[0][0][0] = 0.300·110 = 33.000 + P[0][0][1]·R[0][0]
[1] = 0.500·100 = 50.000 + P[0][0][2]·R[0][0][2] = 0.200·70 = 14.000 = 97.00
0
    γ·∑_j P[0][0][j]·V[j] = γ·P[0][0][0]·V[0] = 0.7·0.300·300.119 = 63.025 +
γ·P[0][0][1]·V[1] = 0.7·0.500·284.707 = 99.648 + γ·P[0][0][2]·V[2] = 0.7·0.2
00·251.374 = 35.192 = 197.865
    Q(0,0) = 97.000 + 197.865 = 294.865
  Действие 1 (доставка):
```

```
    r(0,1) = P[1][0][0]·R[1][0][0] = 0.200·120 = 24.000 + P[1][0][1]·R[1][0]
[1] = 0.700·100 = 70.000 + P[1][0][2]·R[1][0][2] = 0.100·70 = 7.000 = 101.00
0
    γ·∑_j P[1][0][j]·V[j] = γ·P[1][0][0]·V[0] = 0.7·0.200·300.119 = 42.017 +
γ·P[1][0][1]·V[1] = 0.7·0.700·284.707 = 139.507 + γ·P[1][0][2]·V[2] = 0.7·0.
100·251.374 = 17.596 = 199.119
    Q(0,1) = 101.000 + 199.119 = 300.119
    Лучше предыдущего действия 0 с Q=294.865, обновляем.
  Действие 2 (ничего):
    r(0,2) = P[2][0][0]·R[2][0][0] = 0.300·110 = 33.000 + P[2][0][1]·R[2][0]
[1] = 0.400·80 = 32.000 + P[2][0][2]·R[2][0][2] = 0.300·50 = 15.000 = 80.000
    γ·∑_j P[2][0][j]·V[j] = γ·P[2][0][0]·V[0] = 0.7·0.300·300.119 = 63.025 +
γ·P[2][0][1]·V[1] = 0.7·0.400·284.707 = 79.718 + γ·P[2][0][2]·V[2] = 0.7·0.3
00·251.374 = 52.789 = 195.532
    Q(0,2) = 80.000 + 195.532 = 275.532
    Хуже текущего лучшего действия 1 с Q=300.119, пропускаем.
  Лучшее действие для состояния 0: 1 (доставка), Q=300.119
  Q-значения всех действий: [294.865 300.119 275.532]
  >>> Политика НЕ ИЗМЕНЕНА для состояния 0: остается 1

Для состояния 1 (Хор.) ищем оптимальное действие:
  Действие 0 (3% скидка):
    r(1,0) = P[0][1][0]·R[0][1][0] = 0.200·100 = 20.000 + P[0][1][1]·R[0][1]
[1] = 0.600·80 = 48.000 + P[0][1][2]·R[0][1][2] = 0.200·50 = 10.000 = 78.000
    γ·∑_j P[0][1][j]·V[j] = γ·P[0][1][0]·V[0] = 0.7·0.200·300.119 = 42.017 +
γ·P[0][1][1]·V[1] = 0.7·0.600·284.707 = 119.577 + γ·P[0][1][2]·V[2] = 0.7·0.
200·251.374 = 35.192 = 196.786
    Q(1,0) = 78.000 + 196.786 = 274.786
  Действие 1 (доставка):
    r(1,1) = P[1][1][0]·R[1][1][0] = 0.100·110 = 11.000 + P[1][1][1]·R[1][1]
[1] = 0.400·100 = 40.000 + P[1][1][2]·R[1][1][2] = 0.500·90 = 45.000 = 96.00
0
    γ·∑_j P[1][1][j]·V[j] = γ·P[1][1][0]·V[0] = 0.7·0.100·300.119 = 21.008 +
γ·P[1][1][1]·V[1] = 0.7·0.400·284.707 = 79.718 + γ·P[1][1][2]·V[2] = 0.7·0.5
00·251.374 = 87.981 = 188.707
    Q(1,1) = 96.000 + 188.707 = 284.707
    Лучше предыдущего действия 0 с Q=274.786, обновляем.
  Действие 2 (ничего):
    r(1,2) = P[2][1][0]·R[2][1][0] = 0.200·100 = 20.000 + P[2][1][1]·R[2][1]
[1] = 0.600·60 = 36.000 + P[2][1][2]·R[2][1][2] = 0.200·40 = 8.000 = 64.000
    γ·∑_j P[2][1][j]·V[j] = γ·P[2][1][0]·V[0] = 0.7·0.200·300.119 = 42.017 +
γ·P[2][1][1]·V[1] = 0.7·0.600·284.707 = 119.577 + γ·P[2][1][2]·V[2] = 0.7·0.
200·251.374 = 35.192 = 196.786
    Q(1,2) = 64.000 + 196.786 = 260.786
    Хуже текущего лучшего действия 1 с Q=284.707, пропускаем.
  Лучшее действие для состояния 1: 1 (доставка), Q=284.707
  Q-значения всех действий: [274.786 284.707 260.786]
  >>> Политика НЕ ИЗМЕНЕНА для состояния 1: остается 1

Для состояния 2 (Уд.) ищем оптимальное действие:
  Действие 0 (3% скидка):
    r(2,0) = P[0][2][0]·R[0][2][0] = 0.100·80 = 8.000 + P[0][2][1]·R[0][2]
[1] = 0.200·60 = 12.000 + P[0][2][2]·R[0][2][2] = 0.700·40 = 28.000 = 48.000
    γ·∑_j P[0][2][j]·V[j] = γ·P[0][2][0]·V[0] = 0.7·0.100·300.119 = 21.008 +
γ·P[0][2][1]·V[1] = 0.7·0.200·284.707 = 39.859 + γ·P[0][2][2]·V[2] = 0.7·0.7
00·251.374 = 123.173 = 184.041
```

```
    Q(2,0) = 48.000 + 184.041 = 232.041
  Действие 1 (доставка):
    r(2,1) = P[1][2][0]·R[1][2][0] = 0.100·100 = 10.000 + P[1][2][1]·R[1][2]
[1] = 0.200·70 = 14.000 + P[1][2][2]·R[1][2][2] = 0.700·60 = 42.000 = 66.000
    γ·∑_j P[1][2][j]·V[j] = γ·P[1][2][0]·V[0] = 0.7·0.100·300.119 = 21.008 +
γ·P[1][2][1]·V[1] = 0.7·0.200·284.707 = 39.859 + γ·P[1][2][2]·V[2] = 0.7·0.7
00·251.374 = 123.173 = 184.041
    Q(2,1) = 66.000 + 184.041 = 250.041
    Лучше предыдущего действия 0 с Q=232.041, обновляем.
  Действие 2 (ничего):
    r(2,2) = P[2][2][0]·R[2][2][0] = 0.100·80 = 8.000 + P[2][2][1]·R[2][2]
[1] = 0.300·70 = 21.000 + P[2][2][2]·R[2][2][2] = 0.600·60 = 36.000 = 65.000
    γ·∑_j P[2][2][j]·V[j] = γ·P[2][2][0]·V[0] = 0.7·0.100·300.119 = 21.008 +
γ·P[2][2][1]·V[1] = 0.7·0.300·284.707 = 59.789 + γ·P[2][2][2]·V[2] = 0.7·0.6
00·251.374 = 105.577 = 186.374
    Q(2,2) = 65.000 + 186.374 = 251.374
    Лучше предыдущего действия 1 с Q=250.041, обновляем.
  Лучшее действие для состояния 2: 2 (ничего), Q=251.374
  Q-значения всех действий: [232.041 250.041 251.374]
  >>> Политика НЕ ИЗМЕНЕНА для состояния 2: остается 2

Итог улучшения политики:
  Было: [1, 1, 2] (['доставка', 'доставка', 'ничего'])
  Стало: [1, 1, 2] (['доставка', 'доставка', 'ничего'])
  Политика НЕ ИЗМЕНИЛАСЬ - достигнута оптимальная политика.

================================================================================
====
Политика не изменилась. Алгоритм завершён.
================================================================================
====

======================================== Ответ ==============================
===========

Оптимальная политика:
  Состояние 0 (Отл.) → Действие 1 (доставка)
  Состояние 1 (Хор.) → Действие 1 (доставка)
  Состояние 2 (Уд.) → Действие 2 (ничего)

Оптимальная функция ценности:
  V*[0] (Отл.) = 300.119474
  V*[1] (Хор.) = 284.707288
  V*[2] (Уд.) = 251.373955

Стоимость состояний V* в округлённом виде:
  [300.119 284.707 251.374]
```

---

5 решение методами линейного программирования (Без дисконтирования)

In [77]:

```
import numpy as np
from scipy.optimize import linprog


# 1. Определяем матрицы вероятностей P и вознаграждений R по условию задачи

# P[a][s][s'] - вероятность перехода из состояния s в s' при действии a
P = {
    0: [[0.3, 0.5, 0.2],
        [0.2, 0.6, 0.2],
        [0.1, 0.2, 0.7]],
    1: [[0.2, 0.7, 0.1],
        [0.1, 0.4, 0.5],
        [0.1, 0.2, 0.7]],
    2: [[0.3, 0.4, 0.3],
        [0.2, 0.6, 0.2],
        [0.1, 0.3, 0.6]],
}

# R[a][s][s'] - вознаграждение за переход из состояния s в s' при действии a
R = {
    0: [[110, 100, 70],
        [100,  80, 50],
        [ 80,  60, 40]],
    1: [[120, 100, 70],
        [110, 100, 90],
        [100,  70, 60]],
    2: [[110,  80, 50],
        [100,  60, 40],
        [ 80,  70, 60]],
}

# Состояния и действия
state_names = ["Отличный", "Хороший", "Удовлетворительный"]
action_names = ["3% скидка", "Бесплатная доставка", "Ничего"]

# Параметры задачи
num_states = 3  # Количество состояний
num_actions = 3  # Количество действий
num_variables = num_states * num_actions  # Общее количество переменных (9)


# 2. Формируем вектор вознаграждений r для каждого состояния и действия
print("\n1. ФОРМИРОВАНИЕ ВЕКТОРА ВОЗНАГРАЖДЕНИЙ")
print("-"*60)

reward_vector = np.zeros(num_variables)
print("\nРасчет ожидаемых моментальных вознаграждений r(s,a):")

for state in range(num_states):
    print(f"\nСостояние {state} ({state_names[state]}):")

    for action in range(num_actions):
        idx = state * num_actions + action
        # Вычисляем детальное ожидаемое вознаграждение
```

```
        reward = 0
        print(f"  Действие {action} ({action_names[action]}):")

        for next_state in range(num_states):
            contribution = P[action][state][next_state] * R[action][state][r
            reward += contribution
            print(f"    Переход в {next_state} ({state_names[next_state]}):
                  f"P={P[action][state][next_state]:.3f}, R={R[action][state
                  f"вклад = {contribution:.3f}")

        reward_vector[idx] = reward
        print(f"  Итоговое r({state},{action}) = {reward:.3f} (индекс в вект

print("\nИтоговый вектор вознаграждений reward_vector:")
for i in range(num_variables):
    state = i // num_actions
    action = i % num_actions
    print(f"  reward_vector[{i}] = r({state},{action}) = {reward_vector[i]:.

# 3. Формируем матрицу A_eq и вектор b_eq для решения задачи линейного прогр
print("\n2. ФОРМИРОВАНИЕ СИСТЕМЫ ОГРАНИЧЕНИЙ")
print("-"*60)

# A_eq и b_eq для условия балансировки потоков и нормировки
A_eq = np.zeros((num_states + 1, num_variables))
b_eq = np.zeros(num_states + 1)

print("\nФормирование условий балансировки потоков:")
# а) Баланс потоков: для каждого состояния j, учитываем все действия a
for next_state in range(num_states):
    print(f"\nДля состояния {next_state} ({state_names[next_state]}):")
    equation_terms = []

    for state in range(num_states):
        for action in range(num_actions):
            idx = state * num_actions + action
            coef = 0

            # Если это состояние j, добавляем +1*x(j,a)
            if state == next_state:
                A_eq[next_state, idx] += 1.0
                coef += 1.0
                equation_terms.append(f"+1.0·x({state},{action})")

            # Вычитаем вероятность перехода в j из любого состояния
            transition_prob = P[action][state][next_state]
            A_eq[next_state, idx] -= transition_prob
            coef -= transition_prob

            if abs(coef) > 1e-10:  # Показываем только значимые коэффициенты
                print(f"  x({state},{action}) с коэффициентом {coef:.3f}")
                if state == next_state:
                    print(f"    = 1.0 (потому что это текущее состояние) - {
                else:
                    print(f"    = 0.0 (не текущее состояние) - {transition_p
```

```
    print(f"  Полное уравнение баланса (в матричном виде): ∑ коэффициентов ·

print("\nУсловие нормировки:")
# б) Нормировка: сумма всех переменных x_{s,a} равна 1
A_eq[num_states, :] = 1.0
b_eq[num_states] = 1.0

print("  Сумма всех x(s,a) = 1.0:")
for state in range(num_states):
    for action in range(num_actions):
        print(f"    + x({state},{action})")

print("\nИтоговая матрица коэффициентов A_eq:")
for i in range(A_eq.shape[0]):
    if i < num_states:
        row_desc = f"Уравнение баланса для состояния {i} ({state_names[i]})"
    else:
        row_desc = "Условие нормировки"
    print(f"  Строка {i} ({row_desc}):")

    # Выводим значения по частям для лучшей читаемости
    for j in range(num_variables):
        state_j = j // num_actions
        action_j = j % num_actions
        if abs(A_eq[i, j]) > 1e-10:  # Только значимые коэффициенты
            print(f"    A_eq[{i},{j}] (для x({state_j},{action_j})) = {A_eq[

print("\nИтоговый вектор правых частей b_eq:")
for i in range(len(b_eq)):
    if i < num_states:
        row_desc = f"Уравнение баланса для состояния {i} ({state_names[i]})"
    else:
        row_desc = "Условие нормировки"
    print(f"  b_eq[{i}] ({row_desc}) = {b_eq[i]}")

print("\nСистема уравнений в развернутом виде:")
for i in range(num_states):
    equation_parts = []
    for j in range(num_states * num_actions):
        if abs(A_eq[i, j]) > 1e-10:
            state_j = j // num_actions
            action_j = j % num_actions
            sign = "+" if A_eq[i, j] > 0 and equation_parts else ""
            equation_parts.append(f"{sign}{A_eq[i, j]:.4f}·y({state_j},{acti

    equation = " ".join(equation_parts)
    print(f"  {equation} = {b_eq[i]}")


# 4. Решаем задачу линейного программирования для максимизации r^T * x
print("\n3. РЕШЕНИЕ ЗАДАЧИ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ")
print("-"*60)

print("\nЗадача: максимизировать r^T·x при ограничениях A_eq·x = b_eq, x ≥ 0
print("  где x - вектор переменных x(s,a), r - вектор моментальных вознагра
print("  При этом для решения задачи максимизации через linprog (который мин
```

```
print("  мы минимизируем -r^T·x")

print("\nВектор коэффициентов целевой функции (-r):")
for i in range(num_variables):
    state = i // num_actions
    action = i % num_actions
    print(f"  c[{i}] для x({state},{action}) = {-reward_vector[i]:.4f}")

# Задача сводится к минимизации -r^T * x
result = linprog(
    c=-reward_vector,  # Минимизируем -r^T * x
    A_eq=A_eq,
    b_eq=b_eq,
    bounds=[(0, None)] * num_variables,
    method='highs'  # Используем метод 'highs', возможен также 'revised simp
)

# Проверка успешности решения задачи
if not result.success:
    print(f"\nОшибка в решении LP задачи: {result.message}")
else:
    print(f"\nЗадача успешно решена за {result.nit} итераций.")
    print(f"Статус: {result.message}")

# Оптимальное решение
optimal_x = result.x
optimal_reward = reward_vector.dot(optimal_x)

print("\nОптимальные значения переменных x(s,a):")
for i in range(num_variables):
    state = i // num_actions
    action = i % num_actions
    print(f"  x({state},{action}) = {optimal_x[i]:.6f}")

print(f"\nОптимальное значение целевой функции (средний доход): {optimal_rew

# 5. Восстанавливаем политику для каждого состояния
print("\n4. ВОССТАНОВЛЕНИЕ ОПТИМАЛЬНОЙ ПОЛИТИКИ")
print("-"*60)

print("\nДля каждого состояния выбираем действие с наибольшим значением x(s,
optimal_policy = []

for state in range(num_states):
    print(f"\nСостояние {state} ({state_names[state]}):")
    values = [optimal_x[state * num_actions + action] for action in range(nu

    print("  Значения x(s,a) для разных действий:")
    for action in range(num_actions):
        print(f"    x({state},{action}) = {values[action]:.6f}")

    best_action = int(np.argmax(values))  # Действие с максимальной вероятно
    optimal_policy.append(best_action)

    print(f"  Максимальное значение: x({state},{best_action}) = {values[best
    print(f"  Выбранное оптимальное действие: {best_action} ({action_names[b
```

```python
# 6. Выводим итоговые результаты
print("\n5. ИТОГОВЫЕ РЕЗУЛЬТАТЫ")
print("-"*60)

print(f"\nОптимальное среднее вознаграждение g* = {optimal_reward:.4f}")
print("\nОптимальная стационарная детерминированная политика:")
for state in range(num_states):
    print(f"  Состояние {state} ({state_names[state]}) → Действие {optimal_p

print("\nСтационарное распределение по состояниям:")
for state in range(num_states):
    state_sum = sum(optimal_x[state * num_actions + action] for action in ra
    print(f"  Состояние {state} ({state_names[state]}): {state_sum:.6f}")
```

```
1. ФОРМИРОВАНИЕ ВЕКТОРА ВОЗНАГРАЖДЕНИЙ
------------------------------------------------------------

Расчет ожидаемых моментальных вознаграждений r(s,a):

Состояние 0 (Отличный):
  Действие 0 (3% скидка):
    Переход в 0 (Отличный): P=0.300, R=110, вклад = 33.000
    Переход в 1 (Хороший): P=0.500, R=100, вклад = 50.000
    Переход в 2 (Удовлетворительный): P=0.200, R=70, вклад = 14.000
  Итоговое r(0,0) = 97.000 (индекс в векторе reward_vector: 0)
  Действие 1 (Бесплатная доставка):
    Переход в 0 (Отличный): P=0.200, R=120, вклад = 24.000
    Переход в 1 (Хороший): P=0.700, R=100, вклад = 70.000
    Переход в 2 (Удовлетворительный): P=0.100, R=70, вклад = 7.000
  Итоговое r(0,1) = 101.000 (индекс в векторе reward_vector: 1)
  Действие 2 (Ничего):
    Переход в 0 (Отличный): P=0.300, R=110, вклад = 33.000
    Переход в 1 (Хороший): P=0.400, R=80, вклад = 32.000
    Переход в 2 (Удовлетворительный): P=0.300, R=50, вклад = 15.000
  Итоговое r(0,2) = 80.000 (индекс в векторе reward_vector: 2)

Состояние 1 (Хороший):
  Действие 0 (3% скидка):
    Переход в 0 (Отличный): P=0.200, R=100, вклад = 20.000
    Переход в 1 (Хороший): P=0.600, R=80, вклад = 48.000
    Переход в 2 (Удовлетворительный): P=0.200, R=50, вклад = 10.000
  Итоговое r(1,0) = 78.000 (индекс в векторе reward_vector: 3)
  Действие 1 (Бесплатная доставка):
    Переход в 0 (Отличный): P=0.100, R=110, вклад = 11.000
    Переход в 1 (Хороший): P=0.400, R=100, вклад = 40.000
    Переход в 2 (Удовлетворительный): P=0.500, R=90, вклад = 45.000
  Итоговое r(1,1) = 96.000 (индекс в векторе reward_vector: 4)
  Действие 2 (Ничего):
    Переход в 0 (Отличный): P=0.200, R=100, вклад = 20.000
    Переход в 1 (Хороший): P=0.600, R=60, вклад = 36.000
    Переход в 2 (Удовлетворительный): P=0.200, R=40, вклад = 8.000
  Итоговое r(1,2) = 64.000 (индекс в векторе reward_vector: 5)

Состояние 2 (Удовлетворительный):
  Действие 0 (3% скидка):
    Переход в 0 (Отличный): P=0.100, R=80, вклад = 8.000
    Переход в 1 (Хороший): P=0.200, R=60, вклад = 12.000
    Переход в 2 (Удовлетворительный): P=0.700, R=40, вклад = 28.000
  Итоговое r(2,0) = 48.000 (индекс в векторе reward_vector: 6)
  Действие 1 (Бесплатная доставка):
    Переход в 0 (Отличный): P=0.100, R=100, вклад = 10.000
    Переход в 1 (Хороший): P=0.200, R=70, вклад = 14.000
    Переход в 2 (Удовлетворительный): P=0.700, R=60, вклад = 42.000
  Итоговое r(2,1) = 66.000 (индекс в векторе reward_vector: 7)
  Действие 2 (Ничего):
    Переход в 0 (Отличный): P=0.100, R=80, вклад = 8.000
    Переход в 1 (Хороший): P=0.300, R=70, вклад = 21.000
    Переход в 2 (Удовлетворительный): P=0.600, R=60, вклад = 36.000
  Итоговое r(2,2) = 65.000 (индекс в векторе reward_vector: 8)
```

```
Итоговый вектор вознаграждений reward_vector:
  reward_vector[0] = r(0,0) = 97.000
  reward_vector[1] = r(0,1) = 101.000
  reward_vector[2] = r(0,2) = 80.000
  reward_vector[3] = r(1,0) = 78.000
  reward_vector[4] = r(1,1) = 96.000
  reward_vector[5] = r(1,2) = 64.000
  reward_vector[6] = r(2,0) = 48.000
  reward_vector[7] = r(2,1) = 66.000
  reward_vector[8] = r(2,2) = 65.000

2. ФОРМИРОВАНИЕ СИСТЕМЫ ОГРАНИЧЕНИЙ
------------------------------------------------------------

Формирование условий балансировки потоков:

Для состояния 0 (Отличный):
  x(0,0) с коэффициентом 0.700
    = 1.0 (потому что это текущее состояние) - 0.300 (P[0][0][0])
  x(0,1) с коэффициентом 0.800
    = 1.0 (потому что это текущее состояние) - 0.200 (P[1][0][0])
  x(0,2) с коэффициентом 0.700
    = 1.0 (потому что это текущее состояние) - 0.300 (P[2][0][0])
  x(1,0) с коэффициентом -0.200
    = 0.0 (не текущее состояние) - 0.200 (P[0][1][0])
  x(1,1) с коэффициентом -0.100
    = 0.0 (не текущее состояние) - 0.100 (P[1][1][0])
  x(1,2) с коэффициентом -0.200
    = 0.0 (не текущее состояние) - 0.200 (P[2][1][0])
  x(2,0) с коэффициентом -0.100
    = 0.0 (не текущее состояние) - 0.100 (P[0][2][0])
  x(2,1) с коэффициентом -0.100
    = 0.0 (не текущее состояние) - 0.100 (P[1][2][0])
  x(2,2) с коэффициентом -0.100
    = 0.0 (не текущее состояние) - 0.100 (P[2][2][0])
  Полное уравнение баланса (в матричном виде): ∑ коэффициентов · x(s,a) = 0

Для состояния 1 (Хороший):
  x(0,0) с коэффициентом -0.500
    = 0.0 (не текущее состояние) - 0.500 (P[0][0][1])
  x(0,1) с коэффициентом -0.700
    = 0.0 (не текущее состояние) - 0.700 (P[1][0][1])
  x(0,2) с коэффициентом -0.400
    = 0.0 (не текущее состояние) - 0.400 (P[2][0][1])
  x(1,0) с коэффициентом 0.400
    = 1.0 (потому что это текущее состояние) - 0.600 (P[0][1][1])
  x(1,1) с коэффициентом 0.600
    = 1.0 (потому что это текущее состояние) - 0.400 (P[1][1][1])
  x(1,2) с коэффициентом 0.400
    = 1.0 (потому что это текущее состояние) - 0.600 (P[2][1][1])
  x(2,0) с коэффициентом -0.200
    = 0.0 (не текущее состояние) - 0.200 (P[0][2][1])
  x(2,1) с коэффициентом -0.200
    = 0.0 (не текущее состояние) - 0.200 (P[1][2][1])
  x(2,2) с коэффициентом -0.300
    = 0.0 (не текущее состояние) - 0.300 (P[2][2][1])
```

```
  Полное уравнение баланса (в матричном виде): ∑ коэффициентов · x(s,a) = 0

Для состояния 2 (Удовлетворительный):
  x(0,0) с коэффициентом -0.200
    = 0.0 (не текущее состояние) - 0.200 (P[0][0][2])
  x(0,1) с коэффициентом -0.100
    = 0.0 (не текущее состояние) - 0.100 (P[1][0][2])
  x(0,2) с коэффициентом -0.300
    = 0.0 (не текущее состояние) - 0.300 (P[2][0][2])
  x(1,0) с коэффициентом -0.200
    = 0.0 (не текущее состояние) - 0.200 (P[0][1][2])
  x(1,1) с коэффициентом -0.500
    = 0.0 (не текущее состояние) - 0.500 (P[1][1][2])
  x(1,2) с коэффициентом -0.200
    = 0.0 (не текущее состояние) - 0.200 (P[2][1][2])
  x(2,0) с коэффициентом 0.300
    = 1.0 (потому что это текущее состояние) - 0.700 (P[0][2][2])
  x(2,1) с коэффициентом 0.300
    = 1.0 (потому что это текущее состояние) - 0.700 (P[1][2][2])
  x(2,2) с коэффициентом 0.400
    = 1.0 (потому что это текущее состояние) - 0.600 (P[2][2][2])
  Полное уравнение баланса (в матричном виде): ∑ коэффициентов · x(s,a) = 0

Условие нормировки:
  Сумма всех x(s,a) = 1.0:
    + x(0,0)
    + x(0,1)
    + x(0,2)
    + x(1,0)
    + x(1,1)
    + x(1,2)
    + x(2,0)
    + x(2,1)
    + x(2,2)

Итоговая матрица коэффициентов A_eq:
  Строка 0 (Уравнение баланса для состояния 0 (Отличный)):
    A_eq[0,0] (для x(0,0)) = 0.7000
    A_eq[0,1] (для x(0,1)) = 0.8000
    A_eq[0,2] (для x(0,2)) = 0.7000
    A_eq[0,3] (для x(1,0)) = -0.2000
    A_eq[0,4] (для x(1,1)) = -0.1000
    A_eq[0,5] (для x(1,2)) = -0.2000
    A_eq[0,6] (для x(2,0)) = -0.1000
    A_eq[0,7] (для x(2,1)) = -0.1000
    A_eq[0,8] (для x(2,2)) = -0.1000
  Строка 1 (Уравнение баланса для состояния 1 (Хороший)):
    A_eq[1,0] (для x(0,0)) = -0.5000
    A_eq[1,1] (для x(0,1)) = -0.7000
    A_eq[1,2] (для x(0,2)) = -0.4000
    A_eq[1,3] (для x(1,0)) = 0.4000
    A_eq[1,4] (для x(1,1)) = 0.6000
    A_eq[1,5] (для x(1,2)) = 0.4000
    A_eq[1,6] (для x(2,0)) = -0.2000
    A_eq[1,7] (для x(2,1)) = -0.2000
    A_eq[1,8] (для x(2,2)) = -0.3000
```

```
  Строка 2 (Уравнение баланса для состояния 2 (Удовлетворительный)):
    A_eq[2,0] (для x(0,0)) = -0.2000
    A_eq[2,1] (для x(0,1)) = -0.1000
    A_eq[2,2] (для x(0,2)) = -0.3000
    A_eq[2,3] (для x(1,0)) = -0.2000
    A_eq[2,4] (для x(1,1)) = -0.5000
    A_eq[2,5] (для x(1,2)) = -0.2000
    A_eq[2,6] (для x(2,0)) = 0.3000
    A_eq[2,7] (для x(2,1)) = 0.3000
    A_eq[2,8] (для x(2,2)) = 0.4000
  Строка 3 (Условие нормировки):
    A_eq[3,0] (для x(0,0)) = 1.0000
    A_eq[3,1] (для x(0,1)) = 1.0000
    A_eq[3,2] (для x(0,2)) = 1.0000
    A_eq[3,3] (для x(1,0)) = 1.0000
    A_eq[3,4] (для x(1,1)) = 1.0000
    A_eq[3,5] (для x(1,2)) = 1.0000
    A_eq[3,6] (для x(2,0)) = 1.0000
    A_eq[3,7] (для x(2,1)) = 1.0000
    A_eq[3,8] (для x(2,2)) = 1.0000

Итоговый вектор правых частей b_eq:
  b_eq[0] (Уравнение баланса для состояния 0 (Отличный)) = 0.0
  b_eq[1] (Уравнение баланса для состояния 1 (Хороший)) = 0.0
  b_eq[2] (Уравнение баланса для состояния 2 (Удовлетворительный)) = 0.0
  b_eq[3] (Условие нормировки) = 1.0

Система уравнений в развернутом виде:
  0.7000·y(0,0) +0.8000·y(0,1) +0.7000·y(0,2) -0.2000·y(1,0) -0.1000·y(1,1)
-0.2000·y(1,2) -0.1000·y(2,0) -0.1000·y(2,1) -0.1000·y(2,2) = 0.0
  -0.5000·y(0,0) -0.7000·y(0,1) -0.4000·y(0,2) +0.4000·y(1,0) +0.6000·y(1,1)
+0.4000·y(1,2) -0.2000·y(2,0) -0.2000·y(2,1) -0.3000·y(2,2) = 0.0
  -0.2000·y(0,0) -0.1000·y(0,1) -0.3000·y(0,2) -0.2000·y(1,0) -0.5000·y(1,1)
-0.2000·y(1,2) +0.3000·y(2,0) +0.3000·y(2,1) +0.4000·y(2,2) = 0.0

3. РЕШЕНИЕ ЗАДАЧИ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ
------------------------------------------------------------

Задача: максимизировать r^T·x при ограничениях A_eq·x = b_eq, x ≥ 0
  где x - вектор переменных x(s,a), r - вектор моментальных вознаграждений
  При этом для решения задачи максимизации через linprog (который минимизиру
ет),
  мы минимизируем -r^T·x

Вектор коэффициентов целевой функции (-r):
  c[0] для x(0,0) = -97.0000
  c[1] для x(0,1) = -101.0000
  c[2] для x(0,2) = -80.0000
  c[3] для x(1,0) = -78.0000
  c[4] для x(1,1) = -96.0000
  c[5] для x(1,2) = -64.0000
  c[6] для x(2,0) = -48.0000
  c[7] для x(2,1) = -66.0000
  c[8] для x(2,2) = -65.0000

Задача успешно решена за 3 итераций.
```

```
Статус: Optimization terminated successfully. (HiGHS Status 7: Optimal)

Оптимальные значения переменных x(s,a):
  x(0,0) = 0.000000
  x(0,1) = 0.111111
  x(0,2) = 0.000000
  x(1,0) = 0.000000
  x(1,1) = 0.382716
  x(1,2) = 0.000000
  x(2,0) = 0.000000
  x(2,1) = 0.000000
  x(2,2) = 0.506173

Оптимальное значение целевой функции (средний доход): 80.864198

4. ВОССТАНОВЛЕНИЕ ОПТИМАЛЬНОЙ ПОЛИТИКИ
------------------------------------------------------------

Для каждого состояния выбираем действие с наибольшим значением x(s,a):

Состояние 0 (Отличный):
  Значения x(s,a) для разных действий:
    x(0,0) = 0.000000
    x(0,1) = 0.111111
    x(0,2) = 0.000000
  Максимальное значение: x(0,1) = 0.111111
  Выбранное оптимальное действие: 1 (Бесплатная доставка)

Состояние 1 (Хороший):
  Значения x(s,a) для разных действий:
    x(1,0) = 0.000000
    x(1,1) = 0.382716
    x(1,2) = 0.000000
  Максимальное значение: x(1,1) = 0.382716
  Выбранное оптимальное действие: 1 (Бесплатная доставка)

Состояние 2 (Удовлетворительный):
  Значения x(s,a) для разных действий:
    x(2,0) = 0.000000
    x(2,1) = 0.000000
    x(2,2) = 0.506173
  Максимальное значение: x(2,2) = 0.506173
  Выбранное оптимальное действие: 2 (Ничего)

5. ИТОГОВЫЕ РЕЗУЛЬТАТЫ
------------------------------------------------------------

Оптимальное среднее вознаграждение g* = 80.8642

Оптимальная стационарная детерминированная политика:
  Состояние 0 (Отличный) → Действие 1 (Бесплатная доставка)
  Состояние 1 (Хороший) → Действие 1 (Бесплатная доставка)
  Состояние 2 (Удовлетворительный) → Действие 2 (Ничего)

Стационарное распределение по состояниям:
  Состояние 0 (Отличный): 0.111111
```

```
Состояние 1 (Хороший): 0.382716
Состояние 2 (Удовлетворительный): 0.506173
```

---

5 решение методами линейного программирования (С дисконтированием)

In [ ]:

```python
import numpy as np
from scipy.optimize import linprog


# P[a][s][s'] - вероятность перехода из состояния s в состояние s' при дейст
transition_probabilities = {
    0: [[0.3, 0.5, 0.2],
        [0.2, 0.6, 0.2],
        [0.1, 0.2, 0.7]],
    1: [[0.2, 0.7, 0.1],
        [0.1, 0.4, 0.5],
        [0.1, 0.2, 0.7]],
    2: [[0.3, 0.4, 0.3],
        [0.2, 0.6, 0.2],
        [0.1, 0.3, 0.6]],
}

# R[a][s][s'] - вознаграждение за переход из состояния s в s' при действии a
rewards = {
    0: [[110, 100, 70],
        [100,  80, 50],
        [ 80,  60, 40]],
    1: [[120, 100, 70],
        [110, 100, 90],
        [100,  70, 60]],
    2: [[110,  80, 50],
        [100,  60, 40],
        [ 80,  70, 60]],
}

# Параметры задачи
num_states = 3  # Количество состояний
num_actions = 3  # Количество действий
discount_factor = 0.7  # Дисконт-фактор
state_names = ["Отличный", "Хороший", "Удовлетворительный"]
action_names = ["3% скидка", "Бесплатная доставка", "Ничего"]

# Начальное распределение: стартуем из состояния "Отличный"
initial_distribution = np.array([1.0, 0.0, 0.0])


# 1. Формируем вектор вознаграждений r размером num_states * num_actions
print("\n1. ФОРМИРОВАНИЕ ВЕКТОРА ВОЗНАГРАЖДЕНИЙ")
print("-"*60)

reward_vector = np.zeros(num_states * num_actions)
```

```
print("\nРасчет ожидаемых моментальных вознаграждений r(s,a):")

for state in range(num_states):
    print(f"\nСостояние {state} ({state_names[state]}):")

    for action in range(num_actions):
        idx = state * num_actions + action
        # Вычисляем детальное ожидаемое вознаграждение
        reward = 0
        print(f"  Действие {action} ({action_names[action]}):")

        for next_state in range(num_states):
            contribution = transition_probabilities[action][state][next_stat
            reward += contribution
            print(f"    Переход в {next_state} ({state_names[next_state]}):
                  f"P={transition_probabilities[action][state][next_state]:.
                  f"вклад = {contribution:.3f}")

        reward_vector[idx] = reward
        print(f"  Итоговое r({state},{action}) = {reward:.3f} (индекс в вект

print("\nИтоговый вектор вознаграждений reward_vector:")
for i in range(num_states * num_actions):
    state = i // num_actions
    action = i % num_actions
    print(f"  reward_vector[{i}] = r({state},{action}) = {reward_vector[i]:.

# 3. Формируем матрицу равенств A_eq и вектор b_eq для линейного программиро
print("\n2. ФОРМИРОВАНИЕ СИСТЕМЫ УРАВНЕНИЙ ДЛЯ ДИСКОНТИРОВАННОГО СЛУЧАЯ")
print("-"*60)

# A_eq - матрица коэффициентов, b_eq - вектор правых частей
A_eq = np.zeros((num_states, num_states * num_actions))
b_eq = initial_distribution.copy()

print("\nФормирование системы уравнений для дисконтированного MDP:")
print(f"  Для каждого состояния j: ∑_a y(j,a) - γ·∑_s,a P[a][s][j]·y(s,a) =
print(f"  где γ = {discount_factor}, d₀ = {initial_distribution}")

# Формируем систему уравнений для каждого состояния
for next_state in range(num_states):
    print(f"\nДля состояния {next_state} ({state_names[next_state]}):")
    print(f"  d₀[{next_state}] = {initial_distribution[next_state]}")

    for state in range(num_states):
        for action in range(num_actions):
            idx = state * num_actions + action

            # Коэффициент при y(s,a) в уравнении для состояния next_state
            coef = 0

            # Добавляем +1 для y(j,a) в левой части, если state == next_stat
            if state == next_state:
                A_eq[next_state, idx] += 1.0
                coef += 1.0
                print(f"  Для y({state},{action}): +1.0 (т.к. это текущее со
```

```
            # Вычитаем γ·P[a][s][j] для всех y(s,a)
            transition_prob = transition_probabilities[action][state][next_s
            discounted_prob = discount_factor * transition_prob
            A_eq[next_state, idx] -= discounted_prob
            coef -= discounted_prob

            if abs(discounted_prob) > 1e-10:
                print(f"  Для y({state},{action}): -{discount_factor:.1f}·{t
                      f"(переход в состояние {next_state})")

            if abs(coef) > 1e-10:
                print(f"  Итоговый коэффициент для y({state},{action}): {coe

print("\nИтоговая матрица коэффициентов A_eq:")
for i in range(A_eq.shape[0]):
    print(f"  Строка {i} (уравнение для состояния {i} - {state_names[i]}):")

    # Выводим значения по группам для лучшей читаемости
    for state in range(num_states):
        print(f"    Для переменных состояния {state} ({state_names[state]}):
        for action in range(num_actions):
            idx = state * num_actions + action
            print(f"{A_eq[i, idx]:+.4f}", end=" ")
        print()

print("\nИтоговый вектор правых частей b_eq:")
for i in range(len(b_eq)):
    print(f"  b_eq[{i}] (для состояния {i} - {state_names[i]}) = {b_eq[i]}")

print("\nСистема уравнений в развернутом виде:")
for i in range(num_states):
    equation_parts = []
    for j in range(num_states * num_actions):
        if abs(A_eq[i, j]) > 1e-10:
            state_j = j // num_actions
            action_j = j % num_actions
            sign = "+" if A_eq[i, j] > 0 and equation_parts else ""
            equation_parts.append(f"{sign}{A_eq[i, j]:.4f}·y({state_j},{acti

    equation = " ".join(equation_parts)
    print(f"  {equation} = {b_eq[i]}")

# 4. Решаем задачу линейного программирования (LP): maximize r^T * y <=> mi
print("\n3. РЕШЕНИЕ ЗАДАЧИ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ")
print("-"*60)

print("\nЗадача: максимизировать r^T·y при ограничениях A_eq·y = b_eq, y ≥ 0
print("  где y - вектор переменных y(s,a), r - вектор моментальных вознагра
print("  При решении через linprog (который минимизирует) используем -r^T·y"

print("\nВектор коэффициентов целевой функции (-r):")
for i in range(num_states * num_actions):
    state = i // num_actions
    action = i % num_actions
    print(f"  c[{i}] для y({state},{action}) = {-reward_vector[i]:.4f}")
```

```
# Задача сводится к минимизации -r^T * y
result = linprog(
    c=-reward_vector,  # Минимизируем -r^T * y
    A_eq=A_eq,
    b_eq=b_eq,
    bounds=[(0, None)] * (num_states * num_actions),
    method='highs'  # Используем метод 'highs', можно также использовать 're
)

# Проверка успешности решения задачи
if not result.success:
    print(f"\nОшибка в решении LP задачи: {result.message}")
else:
    print(f"\nЗадача успешно решена за {result.nit} итераций.")
    print(f"Статус: {result.message}")

# Оптимальные значения переменных y
optimal_y = result.x
optimal_value = reward_vector.dot(optimal_y)  # Оптимальный дисконтированный

print("\nОптимальные значения переменных y(s,a):")
for i in range(num_states * num_actions):
    state = i // num_actions
    action = i % num_actions
    print(f"  y({state},{action}) = {optimal_y[i]:.6f}")

print(f"\nОптимальное значение целевой функции (дисконтированный доход): {op

# 5. Восстанавливаем стратегию (политику)
print("\n4. ВОССТАНОВЛЕНИЕ ОПТИМАЛЬНОЙ ПОЛИТИКИ")
print("-"*60)

print("\nДля каждого состояния выбираем действие с наибольшим значением y(s,
optimal_policy = []

for state in range(num_states):
    print(f"\nСостояние {state} ({state_names[state]}):")
    action_values = [optimal_y[state * num_actions + action] for action in r

    print("  Значения y(s,a) для разных действий:")
    for action in range(num_actions):
        print(f"    y({state},{action}) = {action_values[action]:.6f}")

    optimal_action = int(np.argmax(action_values))  # Действие с максимальнс
    optimal_policy.append(optimal_action)

    print(f"  Максимальное значение: y({state},{optimal_action}) = {action_v
    print(f"  Выбранное оптимальное действие: {optimal_action} ({action_name

# 6. Проверка и интерпретация результатов
print("\n5. ИТОГОВЫЕ РЕЗУЛЬТАТЫ")
print("-"*60)

print(f"\nОптимальный дисконтированный доход V* = {optimal_value:.4f}")
print("\nОптимальная детерминированная политика:")
```

```
for state in range(num_states):
    print(f"  Состояние {state} ({state_names[state]}) → Действие {optimal_p
```

```
1. ФОРМИРОВАНИЕ ВЕКТОРА ВОЗНАГРАЖДЕНИЙ
------------------------------------------------------------

Расчет ожидаемых моментальных вознаграждений r(s,a):

Состояние 0 (Отличный):
  Действие 0 (3% скидка):
    Переход в 0 (Отличный): P=0.300, R=110, вклад = 33.000
    Переход в 1 (Хороший): P=0.500, R=100, вклад = 50.000
    Переход в 2 (Удовлетворительный): P=0.200, R=70, вклад = 14.000
  Итоговое r(0,0) = 97.000 (индекс в векторе reward_vector: 0)
  Действие 1 (Бесплатная доставка):
    Переход в 0 (Отличный): P=0.200, R=120, вклад = 24.000
    Переход в 1 (Хороший): P=0.700, R=100, вклад = 70.000
    Переход в 2 (Удовлетворительный): P=0.100, R=70, вклад = 7.000
  Итоговое r(0,1) = 101.000 (индекс в векторе reward_vector: 1)
  Действие 2 (Ничего):
    Переход в 0 (Отличный): P=0.300, R=110, вклад = 33.000
    Переход в 1 (Хороший): P=0.400, R=80, вклад = 32.000
    Переход в 2 (Удовлетворительный): P=0.300, R=50, вклад = 15.000
  Итоговое r(0,2) = 80.000 (индекс в векторе reward_vector: 2)

Состояние 1 (Хороший):
  Действие 0 (3% скидка):
    Переход в 0 (Отличный): P=0.200, R=100, вклад = 20.000
    Переход в 1 (Хороший): P=0.600, R=80, вклад = 48.000
    Переход в 2 (Удовлетворительный): P=0.200, R=50, вклад = 10.000
  Итоговое r(1,0) = 78.000 (индекс в векторе reward_vector: 3)
  Действие 1 (Бесплатная доставка):
    Переход в 0 (Отличный): P=0.100, R=110, вклад = 11.000
    Переход в 1 (Хороший): P=0.400, R=100, вклад = 40.000
    Переход в 2 (Удовлетворительный): P=0.500, R=90, вклад = 45.000
  Итоговое r(1,1) = 96.000 (индекс в векторе reward_vector: 4)
  Действие 2 (Ничего):
    Переход в 0 (Отличный): P=0.200, R=100, вклад = 20.000
    Переход в 1 (Хороший): P=0.600, R=60, вклад = 36.000
    Переход в 2 (Удовлетворительный): P=0.200, R=40, вклад = 8.000
  Итоговое r(1,2) = 64.000 (индекс в векторе reward_vector: 5)

Состояние 2 (Удовлетворительный):
  Действие 0 (3% скидка):
    Переход в 0 (Отличный): P=0.100, R=80, вклад = 8.000
    Переход в 1 (Хороший): P=0.200, R=60, вклад = 12.000
    Переход в 2 (Удовлетворительный): P=0.700, R=40, вклад = 28.000
  Итоговое r(2,0) = 48.000 (индекс в векторе reward_vector: 6)
  Действие 1 (Бесплатная доставка):
    Переход в 0 (Отличный): P=0.100, R=100, вклад = 10.000
    Переход в 1 (Хороший): P=0.200, R=70, вклад = 14.000
    Переход в 2 (Удовлетворительный): P=0.700, R=60, вклад = 42.000
  Итоговое r(2,1) = 66.000 (индекс в векторе reward_vector: 7)
  Действие 2 (Ничего):
    Переход в 0 (Отличный): P=0.100, R=80, вклад = 8.000
    Переход в 1 (Хороший): P=0.300, R=70, вклад = 21.000
    Переход в 2 (Удовлетворительный): P=0.600, R=60, вклад = 36.000
  Итоговое r(2,2) = 65.000 (индекс в векторе reward_vector: 8)
```

```
Итоговый вектор вознаграждений reward_vector:
  reward_vector[0] = r(0,0) = 97.000
  reward_vector[1] = r(0,1) = 101.000
  reward_vector[2] = r(0,2) = 80.000
  reward_vector[3] = r(1,0) = 78.000
  reward_vector[4] = r(1,1) = 96.000
  reward_vector[5] = r(1,2) = 64.000
  reward_vector[6] = r(2,0) = 48.000
  reward_vector[7] = r(2,1) = 66.000
  reward_vector[8] = r(2,2) = 65.000

2. ФОРМИРОВАНИЕ СИСТЕМЫ УРАВНЕНИЙ ДЛЯ ДИСКОНТИРОВАННОГО СЛУЧАЯ
------------------------------------------------------------

Формирование системы уравнений для дисконтированного MDP:
  Для каждого состояния j: ∑_a y(j,a) - γ·∑_s,a P[a][s][j]·y(s,a) = d₀[j]
  где γ = 0.7, d₀ = [1. 0. 0.]

Для состояния 0 (Отличный):
  d₀[0] = 1.0
  Для y(0,0): +1.0 (т.к. это текущее состояние 0)
  Для y(0,0): -0.7·0.300 = -0.210 (переход в состояние 0)
  Итоговый коэффициент для y(0,0): 0.7900
  Для y(0,1): +1.0 (т.к. это текущее состояние 0)
  Для y(0,1): -0.7·0.200 = -0.140 (переход в состояние 0)
  Итоговый коэффициент для y(0,1): 0.8600
  Для y(0,2): +1.0 (т.к. это текущее состояние 0)
  Для y(0,2): -0.7·0.300 = -0.210 (переход в состояние 0)
  Итоговый коэффициент для y(0,2): 0.7900
  Для y(1,0): -0.7·0.200 = -0.140 (переход в состояние 0)
  Итоговый коэффициент для y(1,0): -0.1400
  Для y(1,1): -0.7·0.100 = -0.070 (переход в состояние 0)
  Итоговый коэффициент для y(1,1): -0.0700
  Для y(1,2): -0.7·0.200 = -0.140 (переход в состояние 0)
  Итоговый коэффициент для y(1,2): -0.1400
  Для y(2,0): -0.7·0.100 = -0.070 (переход в состояние 0)
  Итоговый коэффициент для y(2,0): -0.0700
  Для y(2,1): -0.7·0.100 = -0.070 (переход в состояние 0)
  Итоговый коэффициент для y(2,1): -0.0700
  Для y(2,2): -0.7·0.100 = -0.070 (переход в состояние 0)
  Итоговый коэффициент для y(2,2): -0.0700

Для состояния 1 (Хороший):
  d₀[1] = 0.0
  Для y(0,0): -0.7·0.500 = -0.350 (переход в состояние 1)
  Итоговый коэффициент для y(0,0): -0.3500
  Для y(0,1): -0.7·0.700 = -0.490 (переход в состояние 1)
  Итоговый коэффициент для y(0,1): -0.4900
  Для y(0,2): -0.7·0.400 = -0.280 (переход в состояние 1)
  Итоговый коэффициент для y(0,2): -0.2800
  Для y(1,0): +1.0 (т.к. это текущее состояние 1)
  Для y(1,0): -0.7·0.600 = -0.420 (переход в состояние 1)
  Итоговый коэффициент для y(1,0): 0.5800
  Для y(1,1): +1.0 (т.к. это текущее состояние 1)
  Для y(1,1): -0.7·0.400 = -0.280 (переход в состояние 1)
  Итоговый коэффициент для y(1,1): 0.7200
```

```
  Для y(1,2): +1.0 (т.к. это текущее состояние 1)
  Для y(1,2): -0.7·0.600 = -0.420 (переход в состояние 1)
  Итоговый коэффициент для y(1,2): 0.5800
  Для y(2,0): -0.7·0.200 = -0.140 (переход в состояние 1)
  Итоговый коэффициент для y(2,0): -0.1400
  Для y(2,1): -0.7·0.200 = -0.140 (переход в состояние 1)
  Итоговый коэффициент для y(2,1): -0.1400
  Для y(2,2): -0.7·0.300 = -0.210 (переход в состояние 1)
  Итоговый коэффициент для y(2,2): -0.2100

Для состояния 2 (Удовлетворительный):
  d₀[2] = 0.0
  Для y(0,0): -0.7·0.200 = -0.140 (переход в состояние 2)
  Итоговый коэффициент для y(0,0): -0.1400
  Для y(0,1): -0.7·0.100 = -0.070 (переход в состояние 2)
  Итоговый коэффициент для y(0,1): -0.0700
  Для y(0,2): -0.7·0.300 = -0.210 (переход в состояние 2)
  Итоговый коэффициент для y(0,2): -0.2100
  Для y(1,0): -0.7·0.200 = -0.140 (переход в состояние 2)
  Итоговый коэффициент для y(1,0): -0.1400
  Для y(1,1): -0.7·0.500 = -0.350 (переход в состояние 2)
  Итоговый коэффициент для y(1,1): -0.3500
  Для y(1,2): -0.7·0.200 = -0.140 (переход в состояние 2)
  Итоговый коэффициент для y(1,2): -0.1400
  Для y(2,0): +1.0 (т.к. это текущее состояние 2)
  Для y(2,0): -0.7·0.700 = -0.490 (переход в состояние 2)
  Итоговый коэффициент для y(2,0): 0.5100
  Для y(2,1): +1.0 (т.к. это текущее состояние 2)
  Для y(2,1): -0.7·0.700 = -0.490 (переход в состояние 2)
  Итоговый коэффициент для y(2,1): 0.5100
  Для y(2,2): +1.0 (т.к. это текущее состояние 2)
  Для y(2,2): -0.7·0.600 = -0.420 (переход в состояние 2)
  Итоговый коэффициент для y(2,2): 0.5800

Итоговая матрица коэффициентов A_eq:
  Строка 0 (уравнение для состояния 0 - Отличный):
    Для переменных состояния 0 (Отличный): +0.7900 +0.8600 +0.7900
    Для переменных состояния 1 (Хороший): -0.1400 -0.0700 -0.1400
    Для переменных состояния 2 (Удовлетворительный): -0.0700 -0.0700 -0.0700
  Строка 1 (уравнение для состояния 1 - Хороший):
    Для переменных состояния 0 (Отличный): -0.3500 -0.4900 -0.2800
    Для переменных состояния 1 (Хороший): +0.5800 +0.7200 +0.5800
    Для переменных состояния 2 (Удовлетворительный): -0.1400 -0.1400 -0.2100
  Строка 2 (уравнение для состояния 2 - Удовлетворительный):
    Для переменных состояния 0 (Отличный): -0.1400 -0.0700 -0.2100
    Для переменных состояния 1 (Хороший): -0.1400 -0.3500 -0.1400
    Для переменных состояния 2 (Удовлетворительный): +0.5100 +0.5100 +0.5800

Итоговый вектор правых частей b_eq:
  b_eq[0] (для состояния 0 - Отличный) = 1.0
  b_eq[1] (для состояния 1 - Хороший) = 0.0
  b_eq[2] (для состояния 2 - Удовлетворительный) = 0.0

Система уравнений в развернутом виде:
  0.7900·y(0,0) +0.8600·y(0,1) +0.7900·y(0,2) -0.1400·y(1,0) -0.0700·y(1,1)
-0.1400·y(1,2) -0.0700·y(2,0) -0.0700·y(2,1) -0.0700·y(2,2) = 1.0
```

```
  -0.3500·y(0,0) -0.4900·y(0,1) -0.2800·y(0,2) +0.5800·y(1,0) +0.7200·y(1,1)
+0.5800·y(1,2) -0.1400·y(2,0) -0.1400·y(2,1) -0.2100·y(2,2) = 0.0
  -0.1400·y(0,0) -0.0700·y(0,1) -0.2100·y(0,2) -0.1400·y(1,0) -0.3500·y(1,1)
-0.1400·y(1,2) +0.5100·y(2,0) +0.5100·y(2,1) +0.5800·y(2,2) = 0.0

3. РЕШЕНИЕ ЗАДАЧИ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ
------------------------------------------------------------

Задача: максимизировать r^T·y при ограничениях A_eq·y = b_eq, y ≥ 0
  где y - вектор переменных y(s,a), r - вектор моментальных вознаграждений
  При решении через linprog (который минимизирует) используем -r^T·y

Вектор коэффициентов целевой функции (-r):
  c[0] для y(0,0) = -97.0000
  c[1] для y(0,1) = -101.0000
  c[2] для y(0,2) = -80.0000
  c[3] для y(1,0) = -78.0000
  c[4] для y(1,1) = -96.0000
  c[5] для y(1,2) = -64.0000
  c[6] для y(2,0) = -48.0000
  c[7] для y(2,1) = -66.0000
  c[8] для y(2,2) = -65.0000

Задача успешно решена за 4 итераций.
Статус: Optimization terminated successfully. (HiGHS Status 7: Optimal)

Оптимальные значения переменных y(s,a):
  y(0,0) = 0.000000
  y(0,1) = 1.326165
  y(0,2) = 0.000000
  y(1,0) = 0.000000
  y(1,1) = 1.151964
  y(1,2) = 0.000000
  y(2,0) = 0.000000
  y(2,1) = 0.000000
  y(2,2) = 0.855205

Оптимальное значение целевой функции (дисконтированный доход): 300.119474

4. ВОССТАНОВЛЕНИЕ ОПТИМАЛЬНОЙ ПОЛИТИКИ
------------------------------------------------------------

Для каждого состояния выбираем действие с наибольшим значением y(s,a):

Состояние 0 (Отличный):
  Значения y(s,a) для разных действий:
    y(0,0) = 0.000000
    y(0,1) = 1.326165
    y(0,2) = 0.000000
  Максимальное значение: y(0,1) = 1.326165
  Выбранное оптимальное действие: 1 (Бесплатная доставка)

Состояние 1 (Хороший):
  Значения y(s,a) для разных действий:
    y(1,0) = 0.000000
    y(1,1) = 1.151964
```

```
    y(1,2) = 0.000000
  Максимальное значение: y(1,1) = 1.151964
  Выбранное оптимальное действие: 1 (Бесплатная доставка)

Состояние 2 (Удовлетворительный):
  Значения y(s,a) для разных действий:
    y(2,0) = 0.000000
    y(2,1) = 0.000000
    y(2,2) = 0.855205
  Максимальное значение: y(2,2) = 0.855205
  Выбранное оптимальное действие: 2 (Ничего)

5. ПРОВЕРКА И ИНТЕРПРЕТАЦИЯ РЕЗУЛЬТАТОВ
------------------------------------------------------------

Оптимальный дисконтированный доход V* = 300.1195

Оптимальная детерминированная политика:
  Состояние 0 (Отличный) → Действие 1 (Бесплатная доставка)
  Состояние 1 (Хороший) → Действие 1 (Бесплатная доставка)
  Состояние 2 (Удовлетворительный) → Действие 2 (Ничего)
```

6

In [79]:

```python
import numpy as np
from scipy.optimize import linprog


# 1. Определяем матрицы вероятностей P и вознаграждений R по условию задачи

# P[a][s][s'] - вероятность перехода из состояния s в s' при действии a
P = {
    0: [[0.3, 0.5, 0.2],
        [0.2, 0.6, 0.2],
        [0.1, 0.2, 0.7]],
    1: [[0.2, 0.7, 0.1],
        [0.1, 0.4, 0.5],
        [0.1, 0.2, 0.7]],
    2: [[0.3, 0.4, 0.3],
        [0.2, 0.6, 0.2],
        [0.1, 0.3, 0.6]],
}

# R[a][s][s'] - вознаграждение за переход из состояния s в s' при действии a
R = {
    0: [[110, 100, 70],
        [100,  80, 50],
        [ 80,  60, 40]],
    1: [[120, 100, 70],
        [110, 100, 90],
        [100,  70, 60]],
    2: [[110,  80, 50],
        [100,  60, 40],
```

```
        [ 80,  70, 60]],
}

# Состояния и действия
state_names = ["Отличный", "Хороший", "Удовлетворительный"]
action_names = ["3% скидка", "Бесплатная доставка", "Ничего"]

# Параметры задачи
num_states = 3  # Количество состояний
num_actions = 3  # Количество действий
num_variables = num_states * num_actions  # Общее количество переменных (9)


# 2. Формируем вектор вознаграждений r для каждого состояния и действия
#print("\n1. ФОРМИРОВАНИЕ ВЕКТОРА ВОЗНАГРАЖДЕНИЙ")
#print("-"*60)

reward_vector = np.zeros(num_variables)
#print("\nРасчет ожидаемых моментальных вознаграждений r(s,a):")

for state in range(num_states):
    #print(f"\nСостояние {state} ({state_names[state]}):")

    for action in range(num_actions):
        idx = state * num_actions + action
        # Вычисляем детальное ожидаемое вознаграждение
        reward = 0
        #print(f"  Действие {action} ({action_names[action]}):")

        for next_state in range(num_states):
            contribution = P[action][state][next_state] * R[action][state][r
            reward += contribution
            #print(f"    Переход в {next_state} ({state_names[next_state]}):
            #      f"P={P[action][state][next_state]:.3f}, R={R[action][stat
            #     f"вклад = {contribution:.3f}")

        reward_vector[idx] = reward
        #print(f"  Итоговое r({state},{action}) = {reward:.3f} (индекс в век

#print("\nИтоговый вектор вознаграждений reward_vector:")
for i in range(num_variables):
    state = i // num_actions
    action = i % num_actions
    #print(f"  reward_vector[{i}] = r({state},{action}) = {reward_vector[i]:

# 3. Формируем матрицу A_eq и вектор b_eq для решения задачи линейного прогр
#print("\n2. ФОРМИРОВАНИЕ СИСТЕМЫ ОГРАНИЧЕНИЙ")
#print("-"*60)

# A_eq и b_eq для условия балансировки потоков и нормировки
A_eq = np.zeros((num_states + 1, num_variables))
b_eq = np.zeros(num_states + 1)

#print("\nФормирование условий балансировки потоков:")
# а) Баланс потоков: для каждого состояния j, учитываем все действия a
for next_state in range(num_states):
```

```
    #print(f"\nДля состояния {next_state} ({state_names[next_state]}):")
    equation_terms = []

    for state in range(num_states):
        for action in range(num_actions):
            idx = state * num_actions + action
            coef = 0

            # Если это состояние j, добавляем +1*x(j,a)
            if state == next_state:
                A_eq[next_state, idx] += 1.0
                coef += 1.0
                equation_terms.append(f"+1.0·x({state},{action})")

            # Вычитаем вероятность перехода в j из любого состояния
            transition_prob = P[action][state][next_state]
            A_eq[next_state, idx] -= transition_prob
            coef -= transition_prob

            if abs(coef) > 1e-10:  # Показываем только значимые коэффициенты
                #print(f"  x({state},{action}) с коэффициентом {coef:.3f}")
                if state == next_state:
                    #print(f"    = 1.0 (потому что это текущее состояние) -
                    pass
                else:
                    pass
                    #print(f"    = 0.0 (не текущее состояние) - {transition_

    #print(f"  Полное уравнение баланса (в матричном виде): ∑ коэффициентов

#print("\nУсловие нормировки:")
# б) Нормировка: сумма всех переменных x_{s,a} равна 1
A_eq[num_states, :] = 1.0
b_eq[num_states] = 1.0

#print("  Сумма всех x(s,a) = 1.0:")
for state in range(num_states):
    for action in range(num_actions):
        #print(f"    + x({state},{action})")
        pass
#print("\nИтоговая матрица коэффициентов A_eq:")
for i in range(A_eq.shape[0]):
    if i < num_states:
        row_desc = f"Уравнение баланса для состояния {i} ({state_names[i]})"
    else:
        row_desc = "Условие нормировки"
    #print(f"  Строка {i} ({row_desc}):")

    # Выводим значения по частям для лучшей читаемости
    for j in range(num_variables):
        state_j = j // num_actions
        action_j = j % num_actions
        #if abs(A_eq[i, j]) > 1e-10:  # Только значимые коэффициенты
            #print(f"    A_eq[{i},{j}] (для x({state_j},{action_j})) = {A_eq

print("\nИтоговый вектор правых частей b_eq:")
```

```python
for i in range(len(b_eq)):
    if i < num_states:
        row_desc = f"Уравнение баланса для состояния {i} ({state_names[i]})"
    else:
        row_desc = "Условие нормировки"
    print(f"  b_eq[{i}] ({row_desc}) = {b_eq[i]}")

print("\nСистема уравнений в развернутом виде:")
for i in range(num_states):
    equation_parts = []
    for j in range(num_states * num_actions):
        if abs(A_eq[i, j]) > 1e-10:
            state_j = j // num_actions
            action_j = j % num_actions
            sign = "+" if A_eq[i, j] > 0 and equation_parts else ""
            equation_parts.append(f"{sign}{A_eq[i, j]:.4f}·y({state_j},{acti

    equation = " ".join(equation_parts)
    print(f"  {equation} = {b_eq[i]}")


# 4. Решаем задачу линейного программирования для максимизации r^T * x
print("\n3. РЕШЕНИЕ ЗАДАЧИ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ")
print("-"*60)

print("\nЗадача: максимизировать r^T·x при ограничениях A_eq·x = b_eq, x ≥ 0
print("  где x - вектор переменных x(s,a), r - вектор моментальных вознагра
print("  При этом для решения задачи максимизации через linprog (который мин
print("  мы минимизируем -r^T·x")

print("\nВектор коэффициентов целевой функции (-r):")
for i in range(num_variables):
    state = i // num_actions
    action = i % num_actions
    print(f"  c[{i}] для x({state},{action}) = {-reward_vector[i]:.4f}")

# Задача сводится к минимизации -r^T * x
result = linprog(
    c=-reward_vector,  # Минмизируем -r^T * x
    A_eq=A_eq,
    b_eq=b_eq,
    bounds=[(0, None)] * num_variables,
    method='highs'  # Используем метод 'highs', возможен также 'revised simp
)

# Проверка успешности решения задачи
if not result.success:
    print(f"\nОшибка в решении LP задачи: {result.message}")
else:
    #print(f"\nЗадача успешно решена за {result.nit} итераций.")
    #print(f"Статус: {result.message}")
    pass

# Оптимальное решение
optimal_x = result.x
optimal_reward = reward_vector.dot(optimal_x)
```

```
#print("\nОптимальные значения переменных x(s,a):")
for i in range(num_variables):
    state = i // num_actions
    action = i % num_actions
    print(f"  x({state},{action}) = {optimal_x[i]:.6f}")

#print(f"\nОптимальное значение целевой функции (средний доход): {optimal_re

# 5. Восстанавливаем политику для каждого состояния
#print("\n4. ВОССТАНОВЛЕНИЕ ОПТИМАЛЬНОЙ ПОЛИТИКИ")
#print("-"*60)

#print("\nДля каждого состояния выбираем действие с наибольшим значением x(s
optimal_policy = []

for state in range(num_states):
    #print(f"\nСостояние {state} ({state_names[state]}):")
    values = [optimal_x[state * num_actions + action] for action in range(nu

    #print("  Значения x(s,a) для разных действий:")
    for action in range(num_actions):
        #print(f"    x({state},{action}) = {values[action]:.6f}")
        pass
    best_action = int(np.argmax(values))  # Действие с максимальной вероятно
    optimal_policy.append(best_action)

    #print(f"  Максимальное значение: x({state},{best_action}) = {values[bes
    #print(f"  Выбранное оптимальное действие: {best_action} ({action_names[

# 6. Выводим итоговые результаты
# 6. Выводим итоговые результаты
print("\n5. ИТОГОВЫЕ РЕЗУЛЬТАТЫ")
print("-"*60)

print(f"\nОптимальное среднее вознаграждение g* = {optimal_reward:.4f}")
print("\nОптимальная стационарная детерминированная политика:")
for state in range(num_states):
    print(f"  Состояние {state} ({state_names[state]}) → Действие {optimal_p

print("\nСтационарное распределение по состояниям:")
for state in range(num_states):
    state_sum = sum(optimal_x[state * num_actions + action] for action in ra
    print(f"  Состояние {state} ({state_names[state]}): {state_sum:.6f}")

# Дополнительная информация о базисных и небазисных переменных
print("\nИнформация о базисных и небазисных переменных:")
print("-"*60)

# Определяем порог для идентификации базисных переменных
threshold = 1e-10

# Находим базисные и небазисные переменные
basic_vars = []
nonbasic_vars = []
```

```python
for i in range(num_variables):
    state = i // num_actions
    action = i % num_actions
    value = optimal_x[i]

    if abs(value) > threshold:
        basic_vars.append((i, state, action, value))
    else:
        nonbasic_vars.append((i, state, action, value))

# Выводим базисные переменные
print("\nБазисные переменные (положительные значения):")
for idx, state, action, value in basic_vars:
    print(f"  x({state},{action}) = {value:.6f} ({state_names[state]}, {acti

# Выводим небазисные переменные
print("\nНебазисные переменные (нулевые значения):")
for idx, state, action, value in nonbasic_vars:
    print(f"  x({state},{action}) = {value:.6e} ({state_names[state]}, {acti
```

```
Итоговый вектор правых частей b_eq:
  b_eq[0] (Уравнение баланса для состояния 0 (Отличный)) = 0.0
  b_eq[1] (Уравнение баланса для состояния 1 (Хороший)) = 0.0
  b_eq[2] (Уравнение баланса для состояния 2 (Удовлетворительный)) = 0.0
  b_eq[3] (Условие нормировки) = 1.0

Система уравнений в развернутом виде:
  0.7000·y(0,0) +0.8000·y(0,1) +0.7000·y(0,2) -0.2000·y(1,0) -0.1000·y(1,1)
-0.2000·y(1,2) -0.1000·y(2,0) -0.1000·y(2,1) -0.1000·y(2,2) = 0.0
  -0.5000·y(0,0) -0.7000·y(0,1) -0.4000·y(0,2) +0.4000·y(1,0) +0.6000·y(1,1)
+0.4000·y(1,2) -0.2000·y(2,0) -0.2000·y(2,1) -0.3000·y(2,2) = 0.0
  -0.2000·y(0,0) -0.1000·y(0,1) -0.3000·y(0,2) -0.2000·y(1,0) -0.5000·y(1,1)
-0.2000·y(1,2) +0.3000·y(2,0) +0.3000·y(2,1) +0.4000·y(2,2) = 0.0

3. РЕШЕНИЕ ЗАДАЧИ ЛИНЕЙНОГО ПРОГРАММИРОВАНИЯ
------------------------------------------------------------

Задача: максимизировать r^T·x при ограничениях A_eq·x = b_eq, x ≥ 0
  где x - вектор переменных x(s,a), r - вектор моментальных вознаграждений
  При этом для решения задачи максимизации через linprog (который минимизиру
ет),
  мы минимизируем -r^T·x

Вектор коэффициентов целевой функции (-r):
  c[0] для x(0,0) = -97.0000
  c[1] для x(0,1) = -101.0000
  c[2] для x(0,2) = -80.0000
  c[3] для x(1,0) = -78.0000
  c[4] для x(1,1) = -96.0000
  c[5] для x(1,2) = -64.0000
  c[6] для x(2,0) = -48.0000
  c[7] для x(2,1) = -66.0000
  c[8] для x(2,2) = -65.0000
  x(0,0) = 0.000000
  x(0,1) = 0.111111
  x(0,2) = 0.000000
  x(1,0) = 0.000000
  x(1,1) = 0.382716
  x(1,2) = 0.000000
  x(2,0) = 0.000000
  x(2,1) = 0.000000
  x(2,2) = 0.506173

5. ИТОГОВЫЕ РЕЗУЛЬТАТЫ
------------------------------------------------------------

Оптимальное среднее вознаграждение g* = 80.8642

Оптимальная стационарная детерминированная политика:
  Состояние 0 (Отличный) → Действие 1 (Бесплатная доставка)
  Состояние 1 (Хороший) → Действие 1 (Бесплатная доставка)
  Состояние 2 (Удовлетворительный) → Действие 2 (Ничего)

Стационарное распределение по состояниям:
  Состояние 0 (Отличный): 0.111111
  Состояние 1 (Хороший): 0.382716
```

```
  Состояние 2 (Удовлетворительный): 0.506173

Информация о базисных и небазисных переменных:
------------------------------------------------------------

Базисные переменные (положительные значения):
  x(0,1) = 0.111111 (Отличный, Бесплатная доставка)
  x(1,1) = 0.382716 (Хороший, Бесплатная доставка)
  x(2,2) = 0.506173 (Удовлетворительный, Ничего)

Небазисные переменные (нулевые значения):
  x(0,0) = 0.000000e+00 (Отличный, 3% скидка)
  x(0,2) = 0.000000e+00 (Отличный, Ничего)
  x(1,0) = 0.000000e+00 (Хороший, 3% скидка)
  x(1,2) = 0.000000e+00 (Хороший, Ничего)
  x(2,0) = 0.000000e+00 (Удовлетворительный, 3% скидка)
  x(2,1) = 0.000000e+00 (Удовлетворительный, Бесплатная доставка)
```