УДК 519.83 https://doi.org/10.33619/2414-2948/110/02

РАЗРАБОТКА НА PUTHON АЛГОРИТМА ПОИСКА РЕШЕНИЯ В ЧИСТЫХ СТРАТЕГИЯХ АНТАГОНИСТИЧЕСКОЙ МАТРИЧНОЙ ИГРЫ

©Сафина Г. Ф., *ORCID: 0000-0002-7326-0896, SPIN-код: 4562-2453, канд.физ.-мат.наук, Уфимский университет науки и технолологий, г. Нефтекамск, Россия, safinagf@mail.ru*
©Кагарманов И. А., *Уфимский университет науки и технолологий, г. Нефтекамск, Россия, iury.conyaev2016@yandex.ru*

PYTHON DEVELOPMENT OF ALGORITHM TO FIND SOLUTION IN PURE STRATEGIES OF ANTAGONISTIC MATRIX GAME

©Safina G., *ORCID: 0000-0002-7326-0896, SPIN-code: 4562-2453, Ph.D., Ufa University of Science and Technology, Neftekamsk, Russia,safinagf@mail.ru*
©Kagarmanov I., *Ufa University of Science and Technology, Neftekamsk, Russia, iury.conyaev2016@yandex.ru*

Аннотация. Приведен алгоритм поиска оптимального решения матричной антогонистической игры в чистых стратегих. Алгоритм реализует поиск верхней и нижней цен матричной игры посредством реализации максиминной и минимаксной стратегий игроков. Кодирование алгоритма произведено с помощью функционала языка программирования Python. Алгоритм протестирован на численном примере.

Abstract. The article provides an algorithm for finding the optimal solution to the matrix antagonistic game in pure strategies. The algorithm implements the search for the upper and lower prices of the matrix game by implementing the maximal and minimax strategies of the players. The algorithm is encoded using the functionality of the Python programming language. The algorithm is tested using a numerical example.

Ключевые слова: матричная игра, оптимальное решение, чистые стратегии, алгоритм, разработка, язык Python.

Keywords: matrix game, optimal solution, pure strategies, algorithm, development, Python language.

Теория игр является одним из мощных инструментов принятия оптимальных решений как в условиях в условиях информативной определенности (конкуренции), так и условиях неопределенности. Методы и технологии теории игр позволяют моделировать сложные взаимодействия между участниками и разрабатывать стратегии, которые помогают достичь оптимальных результатов в разрешениях конфликтов. Известно, что матричные игры с нулевой суммой являются примером антогонистических конфликтов двух сторон (игроков), в которых задается матрица выигрышей одного из игроков, чаще первого игрока [1-3].

В этой работе приводится алгоритм определения оптимального решения подобной игры в ее чистых стратегих. Алгоритм реализует поиск максиминной стратегии первого игрока и минимаксной стратегии второго игрока, которые широко используется в играх с нулевой суммой. Именно метод «минимакс» («максимин») дает возможность игрокам выбрать такие стратегии, которые минимизируют их максимальные потери, что особенно важно в условиях конфликтных ограничений и неопределенности [4; 5].

Т. 11. №1 2025
https://doi.org/10.33619/2414-2948/110

Для того чтобы понять, как это работает, рассмотрим простую задачу, в которой участвуют два игрока — две стороны конфликта. У каждого игрока есть по несколько стратегий, известна матрицей выигрышей первого игрока, которая в данной антогнонистической игре задает одновременно и матрицу выигрышей второго игрока — лишь с противоположными элементами. Каждый элемент такой платежной матрицы указывает на выигрыш игрока 1 в зависимости от выбранных стратегий обоими игроками.

Поиск оптимального решения в чистых стратегиях (если такое решение существует) состоит в том, что игрок 1 выбирает одну свою стратегию (строку матрицы), а игрок 2 — одну свою стратегию (столбец), причем игроки принимают решения, исходя из того, как они могут минимизировать свои потери и максимизировать свои выигрыши.

При таком выборе по одной стратегии каждым из игроков возможна ситуация разрешения конфликта, которая удовлетворит интересам обоих игроков. Именно такая ситуация, в которой максиминное значение игры (нижняя цена) совпадает с ее минимаксным значением (верхней ценой) и является седловой точкой матричной игры двух игроков с нулевой суммой.

Реализация алгоритма проведена с помощью функционала и библиотек языка программирования Python (https://www.python.org/doc; https://www.pythonanywhere.com).

К алгоритму применим и совместим следующие функции и кодовые команды языка:

payoff_matrix.shape — возвращает размеры матрицы (количество строк и столбцов);

player1_strategy[i] = max(payoff_matrix[i]) — для каждой стратегии игрока 1 определяем максимальный возможный выигрыш (наилучший исход), который он может получить, независимо от того, какие стратегии выберет игрок 2;

player2_strategy[j] = min(payoff_matrix[:, j]) — для каждой стратегии игрока 2 находим минимальный возможный проигрыш, то есть наихудший исход для игрока 1, когда игрок 2 выбирает эту стратегию;

player1_best = min(player1_strategy) — после того как для всех стратегий игрока 1 определены максимальные выигрыши, выбираем минимальный из них, чтобы найти оптимальную стратегию для игрока 1.

player2_best = max(player2_strategy) — для игрока 2 мы выбираем максимальный из минимальных выигрышей, чтобы определить его оптимальную стратегию.

Описанный алгоритм теперь применим к следующему численному примеру:

```
import numpy as np

payoff_matrix = np.array([[3, -1, 2],
                          [0, 4, -2],
                          [-1, 2, 3]])

def minimax(payoff_matrix):
    rows, cols = payoff_matrix.shape
    player1_strategy = np.zeros(rows)
    player2_strategy = np.zeros(cols)

    for i in range(rows):
        player1_strategy[i] = max(payoff_matrix[i])

    for j in range(cols):
```

Т. 11. №1 2025
https://doi.org/10.33619/2414-2948/110

```
        player2_strategy[j] = min(payoff_matrix[:, j])

    player1_best = min(player1_strategy)
    player2_best = max(player2_strategy)

    return player1_best, player2_best

player1_best, player2_best = minimax(payoff_matrix)
print(f"Оптимальная стратегия для Игрока 1: {player1_best}")
print(f"Оптимальная стратегия для Игрока 2: {player2_best}")
```

Описанный метод минимакса является одним из важнейших методов теории игр, и помогает найти такие пути решения антогонистической матричной игры, которая будет наилучшей для обоих игроков в самых неблагоприятных условиях.

Метод активно используется во многих практических и производственных областях, использующих анализ конкурентных ситуаций и оптимизацию выходов их этих ситуаций.

В экономике, например, он помогает анализировать конкурентные ситуации, где важно минимизировать убытки и максимизировать прибыль. В бизнесе метод может быть использован для определения стратегий на рынке с конкуренцией, а также для оптимизации ценовых и производственных решений в условиях неопределенности.

В военных стратегиях метод минимакса также применяется для планирования действий в условиях возможных конфликтов. Здесь важно предсказать наихудший исход для каждой из сторон и выбрать стратегию, которая минимизирует потери. Аналогичные подходы используются в политике, где принятие решений часто зависит от действий других стран или политических сил.

Метод минимакса также активно применяется в ситуациях, где необходимо учитывать несколько факторов одновременно (например, в многокритериальных задачах), что делает его универсальным инструментом для принятия решений в условиях неопределенности.

Таким образом, разработанный для автоматизации метода минимакса (максимина) алгоритм на Python можно отнести к оному из инструментов анализа и оптимизации стратегий в играх с нулевой суммой. Дальнейшее расширение алгоритма возможно в его расширении для применения в более сложных многопользовательских играх и более детального анализа взаимодействий между множеством участников.

Список литературы:

1. Видрич Р., Пирс С. Теория игр: основы и приложения. М.: Наука, 2015. 320 с.
2. Peters M, Sharp P. Game theory and operations research: Models and methods. London: Springer, 2017. 456 p.
3. Гамильтон Л., Ван дер Вудсен П. Игры, оптимизация и принятие решений: исследования и приложения. М.: Физматлит, 2016. 400 с.
4. Берн Э. Игры с многими игроками: от теории к практике. М.: Научное издательство УРСС, 2018. 350 с.
5. Миллер Р., Дэвис Т. Математика для анализа и принятия решений. М.: Академический проект, 2019. 480 с.

References:

1. Vidrich, R., & Pirs, S. (2015). Teoriya igr: osnovy i prilozheniya. Moscow. (in Russian).

2. Peters, M, & Sharp, P. (2017). Game theory and operations research: Models and methods. London.

3. Gamil'ton, L., & Van der Vudsen, P. (2016). Igry, optimizatsiya i prinyatie reshenii: issledovaniya i prilozheniya. Moscow. (in Russian).

4. Bern, E. (2018). Igry s mnogimi igrokami: ot teorii k praktike. Moscow. (in Russian).

5. Miller, R., & Devis, T. (2019). Matematika dlya analiza i prinyatiya reshenii. Moscow. (in Russian).

Работа поступила
в редакцию 06.12.2024 г.

Принята к публикации
12.12.2024 г.

Ссылка для цитирования:

Сафина Г. Ф., Кагарманов И. А. Разработка на Python алгоритма поиска решения в чистых стратегиях антагонистической матричной игры // Бюллетень науки и практики. 2025. Т. 11. №1. С. 18-21. https://doi.org/10.33619/2414-2948/110/02

Cite as (APA):

Safina, G., & Kagarmanov, I. (2025). Python Development of Algorithm to find Solution in Pure Strategies of Antagonistic Matrix Game. *Bulletin of Science and Practice*, *11*(1), 18-21. (in Russian). https://doi.org/10.33619/2414-2948/110/02