Вестник ЦЭМИ 2013-2022

ISSN 2079-8784

URL - http://ras.jes.su

Выпуск 3-4 Том 4. 2021

Заметка об алгоритме локального поиска равновесия Нэша для игр с участием n лиц в мульти-матричных постановках

Устав Малков
в.н.с., ЦЭМИ РАН
Москва, Нахимовский проспект, 47

Аннотация

Изучена эффективность использования алгоритма локального поиска (альпинизм) для поиска равновесия Нэша в играх с участием n лиц в мульти-матричных постановках с использованием программной среды Питон.

Ключевые слова: игра n лиц, мульти-матричная постановка, смешанные стратегии, равновесие Нэша, Python

Дата публикации: 11.01.2022

Ссылка для цитирования:

Малков У. Заметка об алгоритме локального поиска равновесия Нэша для игр с участием n лиц в мульти-матричных постановках // Вестник ЦЭМИ – 2021. – Выпуск 3-4 [Электронный ресурс]. URL: https://cemi.jes.su/S265838870017210-4-1 (дата обращения: 11.01.2022). DOI: 10.33276/S265838870017210-4

1 Введение

2 Поиск равновесия Нэша в играх с участием n лиц можно рассматривать как задачу нелинейного программирования, которая, фиксируя стратегии всех игроков, кроме одного, превращается в задачу линейного программирования (LP). Решая эти задачи, последовательно, мы получаем алгоритм локального поиска (LS), который сходится к локальному минимуму. Этот алгоритм, называемый «альпинизмом», был предложен в [1] и успешно применен для поиска

равновесия Нэша в двух-матричных играх [7]. Тот же подход (LS) применим для игр трех лиц как в общих [4], так и в мульти-матричных постановках [5]. И применим для игр с n лицами.

[3] В этой статье мы исследуем эффективность рассмотренного подхода для поиска равновесия по Нэшу для игр с n лицами в мульти-матричных постановках, предложенного в [5], где выигрыш каждого из участников конфликта представляет собой сумму $n-1$ выигрышей к остальным игрокам (8), и игра полностью описывается $n(n-1)$ матрицами. Поли-матричную игру с несколькими игроками можно представить, как совокупность биматричных игр, где игроки играют друг с другом попарно, а ищется коллективное равновесие. С помощью такой игры моделируются экономические конфликты на олигополистическом рынке с n участниками, каждый из которых имеет конечное число стратегий [12].

[4] Имеется пример такой постановки [13], где исследуется одна задача конкуренции между тремя основными банками Монголии в секторе крупного кредитования предприятий. Моделирование конфликта проводится с помощью аппарата поли-(гекса)матричных игр трех лиц, где рассматриваемая попарно конкуренция между всеми тремя банками во всех возможных стратегиях кредитования по разным процентным ставкам представляется в виде гекса-матричной игры с шестью матрицами, элементы которых представляют собой возможные объемы кредитов. Авторы этой работы нашли равновесие в рассмотренной модели, применяя идеи глобального поиска, используя четыре итерации глобального поиска, 68 – локального спуска и решив 443 задачи ЛП за пять секунд. В то же время рассматриваемый в данной статье алгоритм получил результат за один локальный поиск, решив при этом три задачи ЛП, и это все за сотую долю секунды.

[5] В каждой из выше приведенных публикаций описывается алгоритм и его программная реализация для конкретного количества игроков (трех, четырех и даже пяти). В данной работе предлагается алгоритм, при помощи программной реализации которого, можно решать игры с разным количеством участников (3, 4, 5, 6, 7).

[6] Поли-матричная игра трех лиц подробно изучена в [2, 5, 6]. В [5] приведена постановка мульти-матричной игры нескольких лиц. В данной работе описан алгоритм решения мульти-матричной игры нескольких лиц и проверена эффективность его работы.

[7] Предлагаемый алгоритм работает, что экспериментально подтверждено результатами тестовых расчетов, где лексикографически упорядоченным перебором стартовых точек (чистых стратегий игроков), применяя алгоритм локального поиска для поиска минимума функции Нэша, находим точку равновесия Нэша за приемлемое время в играх с тремя, четырьмя, пятью, шестью и семью игроками и до 20 стратегий.

[8] Если для общей постановки при вычислении выигрышей игроков участвуют n мерные матрицы, и тем самым объем информации и время поиска равновесия катастрофически растет (в m раз) при увеличении количества игроков и используемых m стратегий, то в случае мульти-матричной постановки из-за существенно меньшего объема исходной информации удается работать с постановками игр с большим количеством игроков и стратегий. Так как объем информации при переходе к $n+1$ игрокам растет в $(n+1)/(n-1)$ раз.

[9] Поэтому, если бы можно было решить игры для трех человек размером до 100 (т. е. до 100 стратегий на каждого игрока), то в случае игр для четырех игроков можно было бы получить решение только в разумные сроки для нескольких десятков стратегий. В цитируемых публикациях реализовали алгоритм локального поиска в виде отдельной программы для каждого количества игроков. Оказалось, что в случае мульти-матричной постановки можно создать одну программу для игр с n игроками, например, в среде Python.

В этой статье для игр с участием n лиц мы описываем LS-алгоритм компактно при помощи формул (13), (14), (15), и все n задач LP формируются и решаются последовательно с использованием одного и того же набора формул. В случае трех лиц пришлось описать для каждой задачи отдельный набор формул и отдельную процедуру в программной реализации. Такое представление алгоритма позволило компактно писать и реализовывать программы в средах Matlab и Python.

Стало намного проще отлаживать программы, внося изменения в одном месте текста, а не в n. Используемая нами мульти-матричная постановка игр n лиц, представляет собой обобщение гекса-матричной постановки игр трех лиц, где таблицы выигрышей игроков определялись шестью матрицами. В нашем случае таблицы выигрышей n игроков определяются $(n-1)n$ матрицами.

Собирая эти матрицы в одну блочную (14), мы можем представить саму игру n-лиц и алгоритм локального поиска равновесия Нэша в компактной форме. В этом случае игры n-лиц, как задачи нелинейного программирования, становятся задачами квадратичного программирования с линейными условиями.

Имеется несколько других подходов к решению игр для четырех лиц [9, 10], в том числе постановка с несколькими матрицами (трипло), где таблицы выигрышей игроков определяются 12 трехмерными таблицами. Эти подходы используют алгоритм глобальной оптимизации.

Игра n лиц в общей постановке

Конечная бескоалиционная игра n лиц Г определяется множествами $X_1, \dots, X_n$ стратегий n игроков и их функциями выигрышей, заданными на компактном множестве $X = X_1 \times \cdots \times X_n$ евклидова пространства $\mathbf{E}^M$, где для $r \in I = \{1, \dots, n\}$ стратегии $x_r \in X_r \subset \mathbf{E}^{m_r}$ имеют размеры $m_1, \dots, m_n$, вектор $x = (x_1, \dots, x_n) \in \mathbf{E}^M$, $M = m_1 + \cdots + m_n$:

$$X_r = \left\{x_r = \left(x_1^r, \dots, x_{m_r}^r\right) \in \mathbf{E}^{m_r}\colon \langle x_r, e_{m_r}\rangle = \sum_{p=1}^{m_r} x_p^r = 1, \quad x_r \geq o_{m_r}\right\}, \qquad (1)$$

$$f^r(x) = f^r(x_1, \dots, x_n) = \sum_{p_1=1}^{m_1} \cdots \sum_{p_n=1}^{m_n} A^r_{p_1 \dots p_n} x_{p_1}^1 \cdots x_{p_n}^n, \qquad (2)$$

$$f(x) = f(x_1, \dots, x_n) = \sum_{p_1=1}^{m_1} \cdots \sum_{p_n=1}^{m_n} A_{p_1 \cdots p_n} x_{p_1}^1 \cdots x_{p_n}^n, \qquad (3)$$

$A^r_{p_1 \dots p_n}$ — n-мерная таблица выигрышей игрока r ($r \in I$ и $A_{p_1 \dots p_n} = \sum_{r=1}^{n} A^r_{p_1 \dots p_n}$) для любого целого числа $q > 0$ векторы $e_q = (1, \dots, 1) \in \mathbf{E}^q$ и $o_m = (0, \dots, 0) \in \mathbf{E}^q$.

Введем функцию (показатель) Нэша $N(x) = \sum_{r=1}^{n} N^r(x)$,

$$N^r(x) = \max_{x_r \in X_r} f^r(x_1, \dots, x_{r-1}, x_r, x_{r+1}, \dots, x_n) - f^r(x_1, \dots, x_n), \qquad r \in I. \qquad (4)$$

Для всех стратегий $x \in X$ выполняется неравенство $N(x) \geq 0$, и для любой точки равновесия $x^* \in X$ должно соблюдаться условие $N(x^*) = 0$.

Локальный поиск минимума функции Нэша сводится к итеративному процессу, где на каждой итерации последовательно (одна за другой) решаются n задач линейного программирования (ЛП).

Для произвольного вектора $x \in X$ фиксируем значения всех «координат» $x_i = \overline{x}_i$, $i \in I$, кроме $i = r$. Определим множество

$$I \backslash r = \{s_1, \dots, s_{n-1}\} = \begin{cases} \{2,3, \dots, n\}, & r = 1, \\ \{1,2, \dots, n-1\}, & r = n, \\ \{1, \dots, r-1, r+1, \dots, n\}, & 1 < r < n; \end{cases} \qquad I = \{I \backslash r\} \cup r.$$

Тогда значение показателя $N(x)$ как функции от x_r может быть уменьшено (по крайней мере не увеличено) после решения задачи ЛП $P^r(\overline{x}_1, \dots, \overline{x}_{r-1}, x_r, \overline{x}_{r+1}, \dots, \overline{x}_n)$:

$$\sum_{p_r=1}^{m_r} \left(\sum_{p_{s_1}=1}^{m_{s_1}} \cdots \sum_{p_{s_{n-1}}=1}^{m_{s_{n-1}}} A_{p_r\, p_{s_1} \cdots p_{s_{n-1}}} \overline{x}_{p_{s_1}}^{s_1} \cdots \overline{x}_{p_{s_{n-1}}}^{s_{n-1}} \right) x_{p_r}^r - \sum_{u \in I \setminus r} \alpha_u^r \uparrow \max_{x_r, \alpha_u^r} \quad (5)$$

при ограничениях

$$\sum_{p_r=1}^{m_r} \left(\sum_{p_{t_1}=1}^{m_{t_1}} \cdots \sum_{p_{t_{n-2}}=1}^{m_{t_{n-2}}} A_{p_u p_r\, p_{t_1} \cdots p_{t_{n-2}}}^r \overline{x}_{p_{t_1}}^{t_1} \cdots \overline{x}_{p_{t_{n-2}}}^{t_{n-2}} \right) x_{p_r}^r \le \alpha_u^r, \;\; t = s \setminus u, \; u \in I \setminus r, \quad (6)$$

$$\langle x_r, e_{m_r} \rangle = 1, \qquad x_r \ge o_{m_r}, \qquad \alpha_u^r \in \mathbf{E}^1, \qquad u \in I \setminus r. \quad (7)$$

Для полученного оптимального плана x_r^* задачи (5)–(7) и произвольного вектора $x_r \in X_r$ выполняется неравенство

$$0 \le N(\overline{x}_1, \dots, \overline{x}_{r-1}, x_r^*, \overline{x}_{r+1}, \dots, \overline{x}_n) \le N(\overline{x}_1, \dots, \overline{x}_{r-1}, x_r, \overline{x}_{r+1}, \dots, \overline{x}_n).$$

Мульти-матричная постановка

Используемая нами мульти-матричная постановка игр n лиц представляет собой обобщение гекса-матричной постановки игр трех лиц [5], где таблицы выигрышей игроков определялись шестью матрицами. В нашем случае таблицы выигрышей игроков определяются $(n-1)n$ матрицами.

Оказалось, что мульти-матричная постановка после сборки всех матриц, определяющих таблицы игроков в одну блочную матрицу, позволяет написать одну программу для поиска равновесия в играх нескольких лиц, что невозможно делать в случае общей постановки, где для каждого количества игроков надо написать отдельную программу.

Рассмотрим игру Υ с n игроками. Конечная некооперативная игра n лиц Υ определяется n множествами X_r $(1 \le r \le n)$ стратегий игроков соответственно,

$$X_r = \left\{ x_r = (x_1^r, \dots, x_{m_r}^r) \in E^{m(r)} : \langle x_r, e_{m(r)} \rangle = 1, x_r \ge 0_{m(r)} \right\},$$

вместе с их функциями выигрышей по формуле

$$f^r(x) = f^r(x_1, \dots, x_n) = \langle x_r, \sum_{q \in I \setminus r}^n a_{m(r),m(q)}^{r,q} x_q \rangle, \;\; 1 \le r \le n, \quad (8)$$

где $a_{m(r),m(q)}^{r,q}$ – $m(r) \times m(q)$ -матрица, $I = [1,2,\dots,n]$, $(r,q) \in I$, $a_{m(r),m(r)}^{r,r}$ – нуль матрицы.

Т.е. коэффициенты матрицы выигрышей игрока r определяются как суммы коэффициентов $n-1$ матриц по формуле

$$A_{i_1,\dots,i_n}^r = \sum_{q \in I-r} a_{i_r,i_q}^{r,q}, \;\; 1 \leqslant r \leqslant n. \quad (9)$$

$(a_{m(r),m(q)}^{r,q}, a_{m(q),m(r)}^{q,r}$ – матрицы выигрышей в биматричной игре участников r и $q)$.

Введем функцию (индикатор) Нэша $N(x) = \sum_{r=1}^n N^r(x)$,

$$N^r(x) = \max_{x_r \in X_r} f^r(x_1, \dots, x_{r-1}, x_r, x_{r+1}, \dots, x_n) - f^r(x_1, \dots, x_n). \quad (10)$$

Мульти-матричная игра n лиц в этих обозначениях представляется как задача квадратичного программирования

$$\sum_{r=1}^{n}\{\langle x_r, \sum_{q\in I\setminus r} a_{m(r),m(q)}^{r,q}\, x_q\rangle - \alpha^r\} \uparrow \max_{x,\alpha} \qquad (11)$$

при ограничениях

$$\langle x_r, \sum_{s\in I\setminus r}\ a_{m(r),m(s)}^{r,s}\ \rangle \le \alpha^r e_m,\quad r \in I. \qquad (12)$$

$$\langle x_r, e_{m(r)}\rangle\ =\ 1,\ x_r\ \ge 0_{m(r)},\ \ \alpha^r \ge 0_1. \qquad (13)$$

Алгоритм локального поиска

Для компактного представления алгоритма локального поиска мы собрали эти матрицы в блочную матрицу

$$H = \begin{pmatrix} O_{m(1)} & a_{m(1),m(2)}^{1,2} & a_{m(1),m(3)}^{1,3} & \cdots & a_{m(1),m(n)}^{1,n} \\ a_{m(2),m(1)}^{2,1} & O_{m(2)} & a_{m(2),m(3)}^{2,3} & \cdots & a_{m(2),m(n)}^{2,n} \\ a_{m(3),m(1)}^{3,1} & a_{m(3),m(2)}^{3,2} & O_{m(3)} & \cdots & a_{m(3),m(n)}^{3,n} \\ & \cdots & & & \\ a_{m(n),m(1)}^{n,1} & a_{m(n),m(2)}^{n,2} & a_{m(n),m(3)}^{n,3} & \cdots & O_{\mathrm{m(n)}} \end{pmatrix}, \qquad (14)$$

где, например, подматрица $\mathbf{H}_{1,2} = a_{m(1),m(2)}^{1,2}$.

Алгоритм локального поиска равновесия Нэша для игры n-лиц в мульти-матричной постановке принимает следующий вид.

Фиксируя все стратегий $(\bar{x}_v)$ кроме одной (x_r) в представлении функции Нэша в цикле мы построим функционалы (15) и матрицы (17, 18) условий задач линейного программирования (ЛП) P_r $(\mathrm{r} = 1, 2, \ldots, \mathrm{n})$ n игроков и будем решать эти задачи последовательно $1,2, \ldots, n$ друг за другом до тех пор, пока не будет достигнут минимум функции Нэша $N(x)$.

После решения очередной задачи P_r фиксируем полученную стратегию x_r как $\bar{x}_r$, при этом значение показателя $N(x)$ как функция от x_r может уменьшится (по крайней мере не увеличиватся). Т.е. функция Нэша монотонно сходится к локальному минимуму.

В этих задачах функционал задачи P_r, полученной фиксацией всех стратегий кроме одной в выше приведённой задаче квадратичного программирования (11, 12, 13), задается формулой

$$\langle \mathrm{x}_r, \sum_{v\in I\setminus r}\left(a_{m(r),m(v)}^{r,v} + (a_{m(v),m(r)}^{v,r})^T\right)\bar{x}_v\rangle - \sum_{v\in I\setminus r}\alpha_v^r \uparrow \max_{x_r,\alpha_v^r} \qquad (15)$$

или, используя вместо обозначении $a_{m(r),m(v)}^{r,v}$ символы $H_{r,v}$.

$$\{\langle \mathrm{x}_r, \sum_{v\in I\setminus r}\left(H_{r,v} + H_{v,r}^T\right)\bar{x}_v\rangle - \sum_{v\in I\setminus r}\alpha_v^r\} \uparrow \max_{x_r,\alpha_v^r}. \qquad (16)$$

И $n-1$ систем неравенств ($u \in I\setminus r$) описываются формулой

$$H_{u,r}x_r\ + \sum_{s\in I\setminus r,u} H_{u,s}\bar{x}_s \le \alpha_u^r\, e_m,\ u \in I\setminus r, \qquad (17)$$

$$\langle x_r, e_{m(r)}\rangle\ =\ 1,\ x_r\ \ge 0_{m(r)},\ \alpha_u^r\ \ge 0_1. \qquad (18)$$

При помощи формул (15, 16, 17, 18) описываются n $(r = 1, 2, \ldots, n)$ задач и их $n-1$ (u ∈ I\r) систем неравенств. При написании алгоритма в среде Питон эти же формулы реализуются при помощи одной процедуры с параметрами r и u.

66 Решение, т. е. равновесие Нэша, ищется путем целенаправленного перебора начальных точек, наборов чистых стратегий игроков, с использованием локального поиска, пока мы не найдем точку, для которой минимум функции Нэша равен нулю, т. е. найдено равновесие. Упорядоченное лексикографическое генерирование начальных точек может быть выполнено следующим образом.

67 Пусть будет m стратегий и n игроков. Используя начальный список (счетчик) start [1: n]= [1, ...,1], мы устанавливаем начальный набор стартовых точек, то есть первые чистые стратегии игроков. Переход к следующему набору стартовых точек, начиная со второго игрока, осуществляется путем циклического просмотра стартового списка от двух до n путем увеличения рассматриваемого компонента на единицу, если он меньше предельного значения m, и переходим к следующему набору. При обнаружении предельного значения присвоим этому компоненту списка номер первой чистой стратегии и продолжим просмотр списка. Тем самым осуществим лексикографическое продвижение по стартовому списку.

68 В среде Python (где нумерация компонентов списка начинается с 0) переход к следующему набору начальных точек выполняется с помощью следующего цикла:

```
start=[0 for c in range(n+1)] # начальная установка счетчика как набор первых чистых стратегий
for i in range(n):
 if start[i+1]== m-1: # обнаружили последнюю чистую стратегию i+1 игрока
  start[i+1]=0; # назначим i+1 игроку первую чистую стратегию и продолжим просмотр набора
 else:
  start[i+1]=start[i+1]+1; # переходим к следующей чистой стратегии i+1 игрока
  break #  выбор следующего набора завершен
[nc,XR3]=N_LS.N_LS(N,m,ncc,I,H,XR) # Вызов процедуры локального поиска
```

69

70 Мы будем перебирать начальные точки, пока не найдем равновесие или не исчерпаем список этих точек, т. е. start[n] = m.

71 **Результаты тестовых расчетов**

72 В среде Python была составлена программа для поиска равновесия Нэша в играх с n лиц и m стратегиями с мульти старт, где локальный поиск начинался с набора чистых стратегий игроков с последующим лексикографическим сканированием этого набора, пока результат не был найден или этот набор не был исчерпан. Такая программа может быть написана для мульти-матричной постановки, но не для общей постановки.

73 Как вы можете видеть, результаты в таблице показывают, что увеличение числа лиц и стратегий значительно увеличивает время нахождения равновесия.

74 Более того, для тех игр, у которых более десяти стратегий, невозможно получить результаты с приемлемым временем для игр более пяти лиц.

75 В таблице результатов тестирования показана сумма количества начальных точек, итераций и времени решения пяти игр, сгенерированных с использованием датчика случайных чисел в серии трех, четырех, пяти, шести, семи лиц и 3, 5, 10, 20 стратегий (рис. 1).

76

n Persons →m Strategies↓	3	4	5	6	7	
3	12 34 <1	21 72 <1	86 471 <1	210 1455 11	541 6785 56	startp itn time
5	18 64 <1	87 427 <1	517 3777 23	984 9425 101	5438 62062 1054	startp itn Time
10	302 1489 3	484 3495 29	5201 47061 797	25014 308170 9291	139667 2019306 95681	startp ltn Time
20	105 607 4	7958 6996 1850	199390 2359645 124494	>∞	>∞	Startp ltn Time

Рис. 1. Поиск равновесия по Нэшу (начальные точки startp, итерации itn и время time) для пяти игр с участием трех, четырех, пяти, шести, семи лиц и 3, 5, 10, 20 стратегий по локальному алгоритму поиска.

77 Здесь itn – это шаги алгоритма локального поиска, где на каждом шаге решается n задач линейного программирования, а время указывается в секундах.

78 **Заключение**

79 В рассмотренных выше публикациях алгоритм локального поиска был реализован в виде отдельной программы для каждого числа (трех, четырех) лиц. В случае мульти-матричной постановки мы реализовали алгоритм LS как одну программу в среде Python для игр с n лиц $(n = 3, 4, 5, 6, 7)$. Это работает!

80 Модификация [8] метода Лемке-Хаусона [7] хорошо работает для игр с участием трех лиц в мульти-матричной постановке. Мы попытались реализовать этот алгоритм для игр с несколькими участниками в виде одной программы. Оказалось, что в случае более чем четырех игроков эта программа работает плохо.

Библиография:

1. Konno, H. A cutting plane algorithm for solving bilinear programs / H. Konno // Mathematical Programming. – 1976. – V. 11, Issue 1. – p. 14–27.

2. Orlov, A. V. Numerical search for equilibria in bimatrix games / A. V.Orlov, A. S. Strekalovskii // Comput. Math. Math. Phys. – 2005. – V. 45, N6. – p. 947–960.

3. Porter, R. W. Simple Search Methods for Finding a Nash Equilibrium / R. W. Porter, E. Nudelman, Y. Shoham // Games and Economic Behavior. – 2009. – V. 63, N. 2. – p. 642-662.

4. Гольштейн, Е. Г. Приближенный метод решения конечной игры трех лиц / Е. Г.

Гольштейн // Экономика и математические методы. – 2014. – Т. 50, № 1. – с. 110-116.

5. Strekalovskii, A. S. Polymatrix games and optimization problems / A. S. Strekalovskii, R. Enkhbat // Autom. Remote Control. – 2014. – V. 75, N4. – p. 632–645.

6. Orlov, A. On computational search for Nash equilibrium in hexamatrix games / A.Orlov, A.Strekalovsky, S.Batbileg // Optimization Letters. – 2014. – V.10, No.2. p. 369-381.

7. Lemke, C. Equilibrium Points of Bimatrix Games / C. Lemke, C. J. Howson // Journal of the Societyfor Industrial and Applied Mathematics. – 1964. – V. 12. – p. 778-780.

8. Golshteyn, E. The Lemke–Howson Algorithm Solving Finite Non-Cooperative Three-Person Games in a Special Setting / E. Golshteyn, U. Malkov, N. Sokolov // DEStech Transactions on Computer Science and Engineering. – 2018.

9. Batbileg, S. A global optimization algorithm for solving a four-person game / S. Batbileg, N. Tungalag, A. Anikin [and others] // Optimization Letters. – 2019. – V.13. – p.587-596.

10. Enkhbat, R. A Note on Four-Players Triple Game/ R. Enkhbat, S. Bathileg, A. Anikin [and others] // Contributions to Game Theory and Management. – 2019. – V.XII. – p.100-112.

11. Гольштейн, Е. Г. Эффективность приближенного метода решения конечной игры трех лиц (вычислительный опыт) / Е. Г. Гольштейн, У. Х. Малков, Н. А. Соколов // Экономика и математические методы. – 2017. – Т. 53, № 1. – с. 94-107.

12. Neumann, John Von. Theory of games and economic behavior / John Von Neumann, O. Morgenstern. – Princeton University Press, 1944. – 674 p.

13. Орлов, А. В. С. Батбилэг, Олигополистический банковский сектор Монголиии и полиматричные игры трех лиц / А. В. Орлов, С. Батбилэг // Известия Иркутского государственного университета. – 2015. – Т. 11. Серия «Математика». – С. 80–95.

A Note: Nash Equilibrium Local Search Algorithm for n-Person Games in Multi-Matrix Settings

Ustav Malkov
leading researcher, CEMI
RAS Moscow,
Nakhimovsky prospect, 47

Abstract

The effectivity of using the local search algorithm (mountain climbing) for searching Nash equilibrium in n-person games in multi-matrix settings using the Python software environment has been studied.

Keywords: n-person Game, Multi-matrix setting, Mixed strategies, Nash equilibrium, Python

Publication date: 11.01.2022

Citation link:

Malkov U. A Note: Nash Equilibrium Local Search Algorithm for n-Person Games in Multi-Matrix Settings // Vestnik CEMI – 2021. – Issue 3-4 [Electronic resource]. URL: https://cemi.jes.su/S265838870017210-4-1 (circulation date: 11.01.2022). DOI: 10.33276/S265838870017210-4
