

УДК 519.83

ББК 22.18

АЛГОРИТМЫ НАХОЖДЕНИЯ ПРЕД-N-ЯДРА И SM-ЯДРА В КООПЕРАТИВНЫХ ТП-ИГРАХ

СЕРГЕЙ В. БРИТВИН

СВЕТЛАНА И. ТАРАШНИНА

Санкт-Петербургский государственный университет
Факультет прикладной математики – процессов
управления

198504, Санкт-Петербург, Университетский пр., 35
e-mail: serbritvin@gmail.com, tarashnina@gmail.com

В статье рассматриваются два одноточечных решения кооперативных игр n лиц с трансферабельными полезностями: пред-N-ядро и SM-ядро. Предложены новые алгоритмы нахождения этих решений, которые основаны на процедуре нахождения лексикографического минимума, описанной в работе М. Машлера, Б. Пелега и Л.С. Шепли. Введение особой нумерации коалиций обеспечивает нахождение решения в игре за одну итерацию процедуры.

Ключевые слова: Кооперативная игра, эффективно рациональное распределение, лексикографический минимум, эксцесс, пред-N-ядро, би-эксцесс, SM-ядро.

1. Введение

В работе исследуются одноточечные решения кооперативных игр n лиц с трансферабельными полезностями (ТП-игры): пред-N-ядро [11] и SM-ядро [13]. Построение алгоритма нахождения пред-N-ядра

является одной из наиболее важных задач теории кооперативных игр. Существуют различные способы нахождения пред-N-ядра. В основе многих из них лежит метод, предложенный М. Машлером, Б. Пелегом и Л.С. Шепли [6], который основан на последовательном решении не более, чем 4^n задач линейного программирования с 2^n ограничениями относительно 2^n переменных. Ж. Санкаран на основе этого алгоритма предложил свой [10], уменьшив максимальное количество задач линейного программирования в последовательности до 2^n , при этом задачи содержат 2^n ограничений относительно 2^n неизвестных. В работе Дж. Поттерса, Дж. Райниэс и М. Ансинга [8] предложен алгоритм, находящий пред-N-ядро на основе последовательного решения не более, чем $(n - 1)$ -ой задачи линейного программирования, содержащей $(2^n + n - 1)$ ограничение относительно $(2^n - 1)$ -ой переменной. Существуют алгоритмы нахождения пред-N-ядра, предполагающие решение лишь одной задачи линейного программирования. В работе Э. Колберга [5] предложен алгоритм, основанный на решении одной задачи линейного программирования с $(2^n)!$ ограничениями относительно n неизвестных. Г. Оуэн в своей работе [7] предложил алгоритм, основанный на решении одной задачи линейного программирования с 2^n ограничениями относительно 4^n переменных. В работе Х. Пуэрто и Ф. Переа [9] также предложен алгоритм, основанный на одной задаче линейного программирования, содержащей 4^n ограничений относительно 4^n неизвестных, при этом коэффициенты в ограничениях принимают значения из множества $\{0, 1, -1\}$, в отличие от работы Оуэна. В данной работе предлагается новый алгоритм нахождения пред-N-ядра, а также алгоритм нахождения SM-ядра. Отличительной чертой предлагаемых алгоритмов является решение всего лишь одной задачи линейного программирования с $(n + 1)$ -м ограничением относительно 2^n переменных. Алгоритм программно реализован и протестирован на известных примерах кооперативных ТП-игр. Поэтапная реализация алгоритма проиллюстрирована на примере игры 3-х лиц.

Данная работа состоит из введения, пяти разделов и заключения. Сначала приводятся основные понятия и определения, используемые для формализации алгоритмов. Следующий раздел посвящен описанию основных положений процедуры нахождения лекси-

кографического минимума [6]. В разделе 4 вводится определенный порядок следования коалиций, который определяет вид задачи линейного программирования. Непосредственное описание алгоритмов нахождения пред- N -ядра и SM -ядра представлены в разделе 5. Далее предлагаемые алгоритмы поэтапно реализованы на примере игры 3-х лиц в разделе 6.

2. Основные понятия и определения

Рассмотрим класс кооперативных игр n лиц с трансферабельными полезностями (ТП-игр). Обозначим через $N = \{1, 2, \dots, n\}$ конечное множество игроков. Коалицией игроков будем называть любое непустое подмножество S из N .

Определение 2.1. *Кооперативной игрой с трансферабельными полезностями (ТП-игрой) или игрой в форме характеристической функции называется пара (N, v) , где $N = \{1, \dots, n\}$ – множество игроков, $v : 2^N \rightarrow \mathbb{R}$ – характеристическая функция, такая что $v(\emptyset) = 0$.*

Множество всех ТП-игр с фиксированным множеством игроков N обозначим через G^N . Пусть задана игра (N, v) из множества G^N . Полагая, что игроки сформировали максимальную коалицию N , рассмотрим задачу распределения величины $v(N)$ между всеми игроками. Множество допустимых распределений определим следующим образом:

$$X^*(N, v) = \{x \in \mathbb{R}^n \mid \sum_{i \in N} x_i \leq v(N)\}.$$

Определение 2.2. *Множество $X^0(N, v) \subset X^*(N, v)$, такое что*

$$X^0(N, v) = \{x \in \mathbb{R}^n \mid \sum_{i \in N} x_i = v(N)\},$$

называется множеством эффективно рациональных распределений в игре (N, v) .

Из этого определения непосредственно следует, что $x \in X^0$ тогда и только тогда, когда для всех $S \subset N$ выполнено:

$$x(S) + x(N \setminus S) = v(N), \quad (2.1)$$

где

$$x(S) = \sum_{i \in S} x_i.$$

Далее определим понятие решения кооперативной ТП-игры.

Определение 2.3. *Решением кооперативной игры на множестве G^N называется отображение f , которое каждой игре $(N, v) \in G^N$ ставит в соответствие подмножество $f(N, v)$ множества $X^*(N, v)$.*

В статье рассматриваются два одноточечных решения кооперативной ТП-игры, принадлежащие к классу эксцессоподобных решений: пред-N-ядро и SM-ядро. Определим эти решения согласно работам [2,11,13].

Определение 2.4. *Эксцессом $e(x, S, v)$ коалиции S для вектора $x \in X^0$ в игре (N, v) называется следующая величина:*

$$e(x, S, v) = v(S) - x(S), \quad (2.2)$$

где

$$x(S) = \sum_{i \in S} x_i. \quad (2.3)$$

Для каждого конкретного вектора $x \in X^0$ можем вычислить вектор эксцессов $e(x, v) = \{e(x, S, v)\}_{S \subseteq N}$ размерности 2^n .

Определим для некоторого вектора $z \in \mathbb{R}^n$ отображение $\theta : \mathbb{R}^n \rightarrow \mathbb{R}^n$ такое, что $y = \theta(z) \in \mathbb{R}^n$, где вектор y получен из вектора z упорядочиванием его компонент в порядке невозрастания.

Теперь введем понятие пред-N-ядра игры $(N, v) \in G^N$.

Определение 2.5. *Пред-N-ядро игры (N, v) – множество векторов $X_{PN} \subset X^0$, таких что для каждого $x \in X_{PN}$ вектор $\theta(\{e(x, S, v)\}_{S \subseteq N})$ является лексикографически наименьшим:*

$$\begin{aligned} X_{PN}(N, v) = \\ = \{x \in X^0 : \theta(\{e(x, S, v)\}_{S \subseteq N}) \preceq_{lex} \theta(\{e(y, S, v)\}_{S \subseteq N}), \forall y \in X^0\}. \end{aligned}$$

В [11] было доказано, что пред-N-ядро игры (N, v) состоит из единственного вектора.

Для определения SM-ядра введем понятие биексцесса.

Определение 2.6. Биэксцессом $\bar{e}(x, S, v)$ коалиции S для вектора $x \in X^0$ в игре (N, v) называется величина

$$\bar{e}(x, S, v) = e(x, S, v) - e(x, N \setminus S, v). \quad (2.4)$$

Для каждого конкретного вектора $x \in X^0$ и для любого конечного множества коалиций J можем вычислить вектор биэксцессов $\{\bar{e}(x, S, v)\}_{S \subseteq J}$.

Введем понятие SM-ядра игры $(N, v) \in G^N$.

Определение 2.7. SM-ядро игры (N, v) – множество векторов $X_{SM} \subset X^0$, таких что для каждого $x \in X_{SM}$ вектор $\theta(\{\bar{e}(x, S, v)\}_{S \subseteq N})$ является лексикографически наименьшим:

$$\begin{aligned} X_{SM}(N, v) = \\ = \{x \in X^0 : \theta(\{\bar{e}(x, S, v)\}_{S \subseteq N}) \preceq_{lex} \theta(\{\bar{e}(y, S, v)\}_{S \subseteq N}), \forall y \in X^0\}. \end{aligned}$$

В [13] было доказано, что SM-ядро игры (N, v) состоит из единственного вектора.

3. Процедура нахождения лексикографического минимума

Рассмотрим процедуру нахождения пред-N-ядра, приведенную в работе [6].

Для нахождения пред-N-ядра будем последовательно отбрасывать векторы x из множества

$$X^0 = \{x \in \mathbb{R}^n \mid \sum_{i \in N} x_i = v(N)\}$$

согласно некоторому правилу.

Пусть J^0 – множество всех коалиций в игре (N, v) , кроме пустой и максимальной, поскольку они равны нулю для всех $x \in X^0$. Для каждого $x \in X^0$ и для множества коалиций J^0 вычислим вектор эксцессов $\{e(x, S, v)\}_{S \subseteq J^0}$. Найдем максимальную компоненту каждого из векторов эксцессов:

$$u^0(x, v) = \max_{S \subseteq J^0} e(x, S, v).$$

Далее найдем минимальное из всех $u^0(x, v)$ на множестве X^0 :

$$u^0(x', v) = \min_{x \in X^0} u^0(x, v).$$

Найденный вектор x' может быть не единственным. Тогда рассмотрим множество

$$X^1 = \{x \in X^0 \mid x = \arg \min_{x \in X^0} \max_{S \subset J^0} e(x, S, v)\}.$$

Максимальная компонента вектора эксцессов, равная $u^0(x, v)$, совпадает для всех $x \in X^1$. Отбросим из множества J^0 те коалиции S , для которых эксцесс $e(x, S, v)$ совпадает с $u^0(x, v)$ для $x \in X^1$. Таким образом, размерность вектора эксцессов для каждого $x \in X^1$ уменьшается как минимум на единицу. Обозначим через $J^1 \subset J^0$ множество оставшихся коалиций. Для каждого $x \in X^1$ и для множества коалиций J^1 вычислим вектор эксцессов $\{e(x, S, v)\}_{S \subset J^1}$. Найдем максимальную компоненту каждого из векторов эксцессов:

$$u^1(x, v) = \max_{S \subset J^1} e(x, S, v).$$

Далее найдем минимальное из всех $u^1(x, v)$ на множестве X^1 :

$$u^1(x'', v) = \min_{x \in X^1} u^1(x, v).$$

Найденное x'' может быть не единственным. Определим множество

$$X^2 = \{x \in X^1 \mid x = \arg \min_{x \in X^1} \max_{S \subset J^1} e(x, S, v)\}.$$

Будем продолжать выполнять эту процедуру до тех пор, пока размерность вектора эксцессов не станет равной нулю. В дальнейшем будет показано, что при определенных условиях множество X^1 всегда будет состоять из единственного вектора, то есть для того, чтобы получить пред-N-ядро игры (N, v) , достаточно выполнить процедуру только один раз.

Итак, после определения начальных множеств X^0 и J^0 , i -ая итерация процедуры выглядит следующим образом.

Шаг 1. Определяем множество

$$X^i = \{x \in X^{i-1} \mid x = \arg \min_{x \in X^{i-1}} \max_{S \subset J^{i-1}} e(x, S, v)\}.$$

Шаг 2. Отбрасываем из множества J^{i-1} те коалиции S , для которых эксцесс $e(x, S, v)$ максимален для $x \in X^i$. Таким образом, размерность вектора эксцессов для каждого $x \in X^i$ уменьшилась. Определяем множество $J^i \subset J^{i-1}$ – множество оставшихся коалиций.

Шаг 3. Для каждого $x \in X^i$ и для всех $S \subset J^i$ вычисляем вектор эксцессов $\{e(x, S, v)\}_{S \subset J^i}$.

Количество компонент в векторе эксцессов на i -ой итерации зависит от множества коалиций J^i , следовательно, в векторе эксцессов не останется компонент тогда и только тогда, когда множество оставшихся коалиций будет пусто. Таким образом, определен момент, когда нужно остановить выполнение процедуры.

Множество X^i на i -ой итерации можно определить с помощью решения задачи линейного программирования:

$$\begin{cases} \min u, \\ u \geq e(x, S, v), \\ x \in X^{i-1}, \\ S \subset J^{i-1}. \end{cases}$$

4. Нумерация коалиций

В игре (N, v) может быть сформировано 2^n коалиций. Не будем учитывать пустую коалицию. Пронумеруем оставшиеся коалиции следующим образом.

Пусть игра n лиц строится динамически путем добавления одного игрока на каждом шаге, начиная с игры с одним участником. В игре с одним игроком единственная коалиция $\{1\}$ имеет номер 1. Добавим второго игрока и новые коалиции, которые при этом образуются. Последовательность коалиций в порядке возрастания их номеров для игры двух лиц выглядит следующим образом: $\{1\}, \{2\}, \{1, 2\}$. Далее для игры трех лиц имеем: $\{1\}, \{2\}, \{1, 2\}, \{3\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}$. И так далее.

Предположим, что сформирована последовательность коалиций в порядке возрастания их номеров для игры k лиц. Добавление в эту игру $(k + 1)$ -го игрока влечет за собой добавление еще 2^k коалиций. Определим порядок расположения добавленных коалиций. Пусть коалиция $\{k + 1\}$ будет первой среди добавленных. Если в оставшихся из добавленных коалиций не будем обращать внимание на $(k + 1)$ -го игрока, то получим набор коалиций для игры k лиц, который уже выстроен в порядке возрастания номеров. Выстроим добавленные коалиции в этом порядке, не обращая внимание на $(k + 1)$ -го игрока.

Продолжим эту нумерацию на добавленные 2^k коалиций. Таким образом, каждой коалиции в игре с $(k + 1)$ игроком присвоен номер.

Например, для игры 4-х лиц коалиции в порядке возрастания их номеров:

$$\{1\}, \{2\}, \{1, 2\}, \{3\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}, \\ \{4\}, \{1, 4\}, \{2, 4\}, \{1, 2, 4\}, \{3, 4\}, \{1, 3, 4\}, \{2, 3, 4\}, \{1, 2, 3, 4\}.$$

Предложенная нумерация позволит сократить до одной количество итераций в процедуре нахождения пред-N-ядра, предложенной в разделе 3.

5. Алгоритмы нахождения пред-N-ядра и SM-ядра

5.1. Пред-N-ядро

Найдем пред-N-ядро игры (N, v) , используя процедуру, предложенную в разделе 3. При этом будем придерживаться определенной в разделе 4 нумерации для любого встретившегося набора коалиций.

Сначала определим пару (X_{PN}^0, J_{PN}^0) для первой итерации процедуры. Пусть X_{PN}^0 совпадает с множеством эффективно рациональных распределений:

$$X_{PN}^0 = X^0 = \{x \in \mathbb{R}^n \mid \sum_{i=1}^n x_i = v(N)\}.$$

Множество J^0 состоит из всех возможных коалиций для игры n лиц кроме тривиальных – пустой и максимальной. Пусть множество J_{PN}^0 совпадает с множеством J^0 . Для игры с конечным количеством игроков множество J_{PN}^0 содержит конечное количество элементов, пронумеруем их согласно введенной нумерации.

Составим задачу линейного программирования для определения множества X_{PN}^1 для пред-N-ядра:

$$\begin{cases} \min u, \\ u \geq e(x, S, v), \\ x \in X_{PN}^0, \\ S \subset J_{PN}^0. \end{cases} \quad (5.1)$$

Рассмотрим более детально одно из условий задачи:

$$u \geq e(x, S, v) \stackrel{(2.2)}{=} v(S) - x(S) \stackrel{(2.3)}{=} v(S) - \sum_{i \in S} x_i.$$

Таким образом, это условие можно записать в виде неравенства:

$$u + \sum_{i \in S} x_i \geq v(S).$$

Приведем задачу (5.1) к стандартному виду. Неизвестными в задаче являются число u , а также n -мерный вектор x . Объединим все переменные в одну векторную переменную:

$$z = \begin{pmatrix} u \\ x \end{pmatrix}, \quad u \in \mathbb{R}^1, \quad x \in X_{PN}^0. \quad (5.2)$$

В результате введения переменной z получили стандартный вид задачи линейного программирования (5.1):

$$\begin{cases} \min c^T z, \\ A_1 z \geq b_1, \\ A_{eq} z = b_{eq}. \end{cases} \quad (5.3)$$

Перебирая все коалиции $S \subset J_{PN}^0$ согласно выбранной нумерации, получаем определенный вид матриц A_1 , A_{eq} и векторов b_1 , b_{eq} и c .

Вектор c содержит $(n + 1)$ компоненту:

$$c = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}. \quad (5.4)$$

Матрица A_1 состоит из двух блоков:

$$A_1 = \begin{pmatrix} I & A_{PN} \end{pmatrix}, \quad (5.5)$$

где I представляет собой вектор, состоящий из единиц и содержащий $(2^n - 2)$ компоненты:

$$I = \begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{pmatrix}. \quad (5.6)$$

Матрица A_{PN} имеет размерность $[(2^n - 2) \times n]$. Заметим, что строки этой матрицы могут быть получены последовательным переводом первых $(2^n - 2)$ чисел из десятичной системы исчисления в двоичную:

$$A_{PN} = \begin{pmatrix} 1 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 & 0 \\ 1 & 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 & 0 \\ \vdots & & & & \ddots & & \vdots \\ 1 & 0 & 1 & 1 & \dots & 1 & 1 \\ 0 & 1 & 1 & 1 & \dots & 1 & 1 \end{pmatrix}. \quad (5.7)$$

Вектор b_1 содержит $(2^n - 2)$ компоненты и имеет следующий вид:

$$b_1 = \begin{pmatrix} v(1) \\ v(2) \\ v(1, 2) \\ v(3) \\ v(1, 3) \\ \vdots \\ v(1, 3, \dots, n) \\ v(2, 3, \dots, n) \end{pmatrix}.$$

Матрица A_{eq} имеет размерность $[1 \times (n + 1)]$:

$$A_{eq} = (0 \ 1 \ 1 \ 1 \ \dots \ 1). \quad (5.8)$$

Вектор b_{eq} содержит только одну компоненту:

$$b_{eq} = v(1, 2, 3, \dots, n).$$

Класс задач линейного программирования был сформулирован Л.В. Канторовичем [1]. Д. Данциг продолжил изучение этого класса задач и разработал симплекс-метод, позволяющий решать задачи линейного программирования [3]. В статье [4] сформулированы альтернативы решения задачи линейного программирования, которые в книге Х. Тахи [12] представлены в следующем виде:

- решение существует и достигается в единственной точке или неограниченно стремится к бесконечности;
- существует бесконечно много решений;
- множество решений пусто.

Следующая теорема оставляет лишь первую альтернативу для решения задачи линейного программирования (5.3).

Теорема 5.1. *В выбранной нумерации множество решений задачи линейного программирования (5.3) состоит из одной точки z_{PN}^* или неограниченно стремится к бесконечности.*

Доказательство. В статье [6] было доказано, что используя процедуру, описанную в разделе 3, за конечное количество итераций можно найти пред- N -ядро. При этом утверждается, что пред- N -ядро является общей точкой множеств X_{PN}^0 и X_{PN}^1 , откуда можно сделать вывод, что множество X_{PN}^1 , являющееся решением задачи линейного программирования (5.1), не пусто.

Согласно [4,12], задача линейного программирования в стандартной форме (5.3) имеет бесконечное множество решений, если вектор-строка c^T компланарен с любой парой векторов-строк матриц A_1 и A_{eq} . В нашем случае вектор c имеет вид (5.4), матрица A_1 , согласно (5.5), состоит из матриц I (5.6) и A_{PN} (5.7), а матрица A_{eq} имеет вид (5.8).

Таким образом, у задачи линейного программирования (5.3) не может быть бесконечного множества решений, так как ни одна пара векторов-строк матриц A_1 и A_{eq} не компланарна вектору-строке c^T . □

В случае, когда множество X_{PN}^1 ограничено и состоит из одной точки x_{PN}^* , получим, что в точке x_{PN}^* достигается лексикографический минимум на множестве векторов эксцессов всех эффективно ра-

циональных распределений, следовательно, x_{PN}^* – пред-N-ядро игры (N, v) .

Алгоритмическая сложность в работах [5–10] определяется количеством задач линейного программирования и количеством ограничений и переменных в них. Также существенным является вид матрицы ограничений. В представленном алгоритме нахождения пред-N-ядра решается одна задача линейного программирования (5.3), матрица ограничений которой имеет размерность $[(2^n - 1) \times (n + 1)]$ и состоит из нулей и единиц.

5.2. SM-ядро

По аналогии найдем теперь SM-ядро в игре (N, v) , используя процедуру, описанную в разделе 3. При этом будем придерживаться определенной в разделе 4 нумерации для любого встретившегося набора коалиций.

Определим пару (X_{SM}^0, J_{SM}^0) для первой итерации процедуры. Пусть X_{SM}^0 совпадает с множеством эффективно рациональных распределений:

$$X_{SM}^0 = X^0 = \{x \in \mathbb{R}^n \mid \sum_{i=1}^n x_i = v(N)\}.$$

Множество J^0 состоит из всех возможных коалиций, кроме тривиальных – пустой и максимальной. Пусть множество J_{SM}^0 совпадает с множеством J^0 . Для игры с конечным количеством игроков множество J_{SM}^0 содержит конечное количество элементов. Пронумеруем их согласно введенной нумерации.

Составим задачу линейного программирования для определения множества X_{SM}^1 для SM-ядра:

$$\begin{cases} \min u, \\ u \geq \bar{e}(x, S, v), \\ x \in X_{SM}^0, \\ S \subset J_{SM}^0. \end{cases} \quad (5.9)$$

Рассмотрим более детально одно из условий задачи:

$$\begin{aligned}
 u \geq \bar{e}(x, S, v) &\stackrel{(2.4)}{=} e(x, S, v) - e(x, N \setminus S, v) \stackrel{(2.2)}{=} v(S) - x(S) - v(N \setminus S) + \\
 + x(N \setminus S) &\stackrel{(2.1)}{=} v(S) - v(N \setminus S) + v(N) - 2 \cdot x(S) \stackrel{(2.3)}{=} v(S) - v(N \setminus S) + \\
 &+ v(N) - 2 \cdot \sum_{i \in S} x_i.
 \end{aligned}$$

Таким образом, это условие можно записать в виде неравенства:

$$u + 2 \cdot \sum_{i \in S} x_i \geq v(S) - v(N \setminus S) + v(N).$$

Приведем задачу (5.9) к стандартному виду. Неизвестными в задаче являются число u , а также n -мерный вектор x . Объединим все переменные в одну векторную переменную следующим образом:

$$z = \begin{pmatrix} u \\ x \end{pmatrix}, \quad u \in \mathbb{R}^1, \quad x \in X_{SM}^0. \quad (5.10)$$

В результате введения переменной z получили стандартный вид задачи линейного программирования (5.9):

$$\begin{cases} \min c^T z, \\ A_2 z \geq b_2, \\ A_{eq} z = b_{eq}. \end{cases} \quad (5.11)$$

Перебирая все коалиции $S \subset J_{SM}^0$ согласно выбранной нумерации, получаем определенный вид матриц A_2 , A_{eq} и векторов b_2 , b_{eq} и c .

Вектор c содержит $(n + 1)$ компоненту:

$$c = \begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{pmatrix}. \quad (5.12)$$

Матрица A_2 состоит из двух блоков:

$$A_2 = (I \quad 2A_{SM}), \quad (5.13)$$

где I представляет собой вектор, состоящий из единиц и содержащий $(2^n - 2)$ компоненты:

$$I = \begin{pmatrix} 1 \\ 1 \\ 1 \\ \vdots \\ 1 \\ 1 \end{pmatrix}. \quad (5.14)$$

Матрица A_{SM} имеет размерность $[(2^n - 2) \times n]$. Заметим, что строки этой матрицы могут быть получены последовательным переводом первых $(2^n - 2)$ чисел из десятичной системы исчисления в двоичную:

$$A_{SM} = \begin{pmatrix} 1 & 0 & 0 & 0 & \dots & 0 & 0 \\ 0 & 1 & 0 & 0 & \dots & 0 & 0 \\ 1 & 1 & 0 & 0 & \dots & 0 & 0 \\ 0 & 0 & 1 & 0 & \dots & 0 & 0 \\ \vdots & & & & \ddots & & \vdots \\ 1 & 0 & 1 & 1 & \dots & 1 & 1 \\ 0 & 1 & 1 & 1 & \dots & 1 & 1 \end{pmatrix}. \quad (5.15)$$

Вектор b_2 содержит $(2^n - 2)$ компоненты:

$$b_2 = \begin{pmatrix} v(1) - v(2, 3, \dots, n) + v(N) \\ v(2) - v(1, 3, \dots, n) + v(N) \\ v(1, 2) - v(3, 4, \dots, n) + v(N) \\ v(3) - v(1, 2, 4, \dots, n) + v(N) \\ v(1, 3) - v(2, 4, \dots, n) + v(N) \\ \vdots \\ v(1, 3, \dots, n) - v(2) + v(N) \\ v(2, 3, \dots, n) - v(1) + v(N) \end{pmatrix}.$$

Матрица A_{eq} имеет размерность $[1 \times (n + 1)]$.

$$A_{eq} = (0 \ 1 \ 1 \ 1 \ \dots \ 1). \quad (5.16)$$

Вектор b_{eq} содержит только одну компоненту:

$$b_{eq} = v(1, 2, 3, \dots, n).$$

Следующая теорема оставляет лишь первую альтернативу для решения задачи линейного программирования (5.11).

Теорема 5.2. *В выбранной нумерации множество решений задачи линейного программирования (5.11) состоит из одной точки z_{SM}^* или неограниченно стремится к бесконечности.*

Доказательство. Для нахождения SM-ядра, как и для пред-N-ядра, используется одна и та же процедура, описанная в разделе 3. При этом SM-ядро является общей точкой множеств X_{SM}^0 и X_{SM}^1 , откуда можно сделать вывод, что множество X_{SM}^1 , являющееся решением задачи линейного программирования (5.9), не пусто.

Согласно [4,12], задача линейного программирования в стандартной форме (5.11) имеет бесконечное множество решений, если вектор-строка c^T компланарен с любой парой векторов-строк матриц A_2 и A_{eq} . В нашем случае вектор c имеет вид (5.12), матрица A_2 , согласно (5.13), состоит из матриц I (5.14) и $2A_{SM}$ (5.15), а матрица A_{eq} имеет вид (5.16).

Таким образом, у задачи линейного программирования (5.11) не может быть бесконечного множества решений, так как ни одна пара векторов-строк матриц A_2 и A_{eq} не компланарна вектору-строке c^T . \square

В случае, когда множество X_{SM}^1 ограничено и состоит из одной точки x_{SM}^* , получим, что в точке x_{SM}^* достигается лексикографический минимум на множестве векторов биэксцессов всех эффективно рациональных распределений, следовательно, x_{SM}^* – SM-ядро игры (N, v) .

В представленном алгоритме нахождения SM-ядра решается одна задача линейного программирования (5.11), матрица ограничений которой имеет размерность $[(2^n - 1) \times (n + 1)]$ и состоит из нулей, единиц и двоек.

6. Иллюстрация поэтапной реализации предложенных алгоритмов

Рассмотрим игру трех лиц с характеристической функцией следующего вида:

$$v(\{1\}) = v(\{2\}) = v(\{1, 2\}) = 1,$$

$$v(\{3\}) = 3, \quad v(\{1, 3\}) = 4, \quad v(\{2, 3\}) = 5, \quad v(\{1, 2, 3\}) = 8.$$

6.1. Пред-N-ядро

1. Следуя предложенному алгоритму, построим пред-N-ядро рассматриваемой игры. Составим задачу линейного программирования(5.3):

$$\left\{ \begin{array}{l} \min (1 \ 0 \ 0 \ 0) z, \\ \begin{pmatrix} 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \end{pmatrix} z \geq \begin{pmatrix} 1 \\ 1 \\ 1 \\ 3 \\ 4 \\ 5 \end{pmatrix} . \\ \begin{pmatrix} 0 & 1 & 1 & 1 \end{pmatrix} z = 8. \end{array} \right.$$

2. В полученной задаче количество ограничений равно 7: 6 нестрогих и одно строгое. Количество переменных равно 4. Все они могут принимать как положительные, так и отрицательные значения. Решим полученную задачу двухфазным симплекс-методом. Получим, что минимальное значение целевой функции достигается при

$$z^* = \begin{pmatrix} -1 \\ 2 \\ 2 \\ 4 \end{pmatrix}.$$

3. Согласно замене (5.2), из z^* получим пред-N-ядро игры (N, v) в следующем виде:

$$x_{PN}^*(N, v) = \begin{pmatrix} 2 \\ 2 \\ 4 \end{pmatrix}.$$

6.2. SM-ядро

1. Составим задачу линейного программирования (5.11):

$$\left\{ \begin{array}{l} \min (1 \ 0 \ 0 \ 0) z, \\ \left(\begin{array}{cccc} 1 & 2 & 0 & 0 \\ 1 & 0 & 2 & 0 \\ 1 & 2 & 2 & 0 \\ 1 & 0 & 0 & 2 \\ 1 & 2 & 0 & 2 \\ 1 & 0 & 2 & 2 \end{array} \right) z \geq \left(\begin{array}{c} 4 \\ 5 \\ 6 \\ 10 \\ 11 \\ 12 \end{array} \right), \\ \left(\begin{array}{cccc} 0 & 1 & 1 & 1 \end{array} \right) z = 8. \end{array} \right.$$

2. В полученной задаче количество ограничений равно 7: 6 нестрогих и одно строгое. Количество переменных равно 4. Все они могут принимать как положительные, так и отрицательные значения. Решим полученную задачу двухфазным симплекс-методом. Получим, что минимальное значение целевой функции достигается при

$$z^* = \begin{pmatrix} 1 \\ \frac{3}{2} \\ 2 \\ \frac{9}{2} \end{pmatrix}.$$

3. Согласно замене (5.10) получим SM-ядро игры (N, v) в следующем виде:

$$x_{SM}^*(N, v) = \begin{pmatrix} \frac{3}{2} \\ 2 \\ \frac{9}{2} \end{pmatrix}.$$

7. Заключение

В работе представлены алгоритмы нахождения пред-N-ядра и SM-ядра в кооперативных играх с трансферабельными полезностями. Основным достоинством алгоритмов является то, что для нахождения решения необходимо решить только одну задачу линейного программирования.

Исследования могут быть продолжены в направлении оптимизации алгоритмов с целью уменьшения количества простейших операций, выполняемых в процессе работы алгоритмов. Также оптимизацию можно вести в направлении уменьшения объема используемой памяти при работе программы, реализующей алгоритмы. Кроме того, можно модифицировать симплекс-метод, используемый в алгоритмах, для решения конкретной задачи линейного программирования. Действительно, размерность начальной таблицы симплекс-метода определяется количеством игроков в игре, а вектор правых частей зависит от значения характеристической функции игры.

СПИСОК ЛИТЕРАТУРЫ

1. Канторович Л.В. *Математические методы организации и планирования производства* // Издательство ЛГУ. 1939.
2. Смирнова Н.В., Тарашнина С.И. *Геометрические свойства $[0,1]$ -N-ядра в кооперативных ТП-играх* // Математическая теория игр и ее приложения. 2012. Т. 4. Вып. 1. С. 55–73.
3. Dantzig G.B. *Programming in a linear structure* // *Econometrica*. 1949. V. 17. P. 73–74.
4. Dantzig G.B. *Linear programming under uncertainty* // *Management science*. 1955. V. 1. P. 297–306.
5. Kohlberg E. *The nucleolus as a solution of a minimization problem* // *SIAM Journal on Applied Mathematics*. 1972. V. 23. P. 34–39.
6. Maschler M., Peleg B. and Shapley L.S. *Geometric properties of the kernel, nucleolus and related solution concepts* // *Mathematics of operations research*. 1979. V. 4. N 4. P. 303–338.
7. Owen G. *A note on the nucleolus* // *International journal on Game theory*. 1974. V. 3. P. 101–103
8. Potters J.A.M., Reijnierse J.H., Ansing M. *Computing the nucleolus by solving a prolonged simplex algorithm* // *Mathematics of operations research*. 1996. V. 21. N 3. P. 757–768.

9. Puerto J., Perea E. *Finding the nucleolus of any n-person cooperative game by a single linear program* // Computers and operations research. 2013. V. 40. P. 2308–2313.
10. Sankaran J.K. *On finding the nucleolus of an n-person cooperative game* // International Journal of Game Theory. 1991. V. 19. N 4. P. 329–338.
11. Schmeidler D. *The nucleolus of a characteristic function game* // SIAM Journal on Applied Mathematics. 1969. V. 17. N 6. P 1163–1170.
12. Taha H.A. *Operations research: an introduction* // Prentice Hall. 2006. P. 95–193.
13. Tarashnina S. *The simplified modified nucleolus of a cooperative TU-game* // Operations Research and Decision Theory. 2011. V. 19. N 1. P. 150–166.

ALGORITHMS OF FINDING THE PRENUCLEOLUS AND THE SM-NUCLEOLUS OF COOPERATIVE TU-GAMES

Sergei V. Britvin, Saint-Petersburg State University, postgraduate (serbritvin@gmail.com),

Svetlana I. Tarashnina, Saint-Petersburg State University, Cand.Sc., associate prof. (tarashnina@gmail.com).

Abstract: Two one-pointed solutions of cooperative games with transferable utility: the prenucleolus and the SM-Nucleolus, are presented in this article. New algorithms of finding these solutions based on the procedure, described in the paper of M. Maschler, B. Peleg and L.S. Shapley, are proposed in this article. Introducing the numbering of the coalitions provides finding solution with the use of only one iteration of the procedure.

Keywords: cooperative game, effectively rational distribution, lexicographical minimum, excess, the prenucleolus, biexcess, the SM-nucleolus.