Ляхов Александр Федорович,
Тришин Илья Михайлович

АЛГОРИТМЫ И ПРОГРАММЫ УПРАВЛЕНИЯ КОМПЬЮТЕРОМ В АЗАРТНЫХ ИГРАХ, СОЗДАННЫЕ НА ОСНОВЕ ТЕОРИИ НЕЧЁТКИХ МНОЖЕСТВ

Поиск оптимального управления и исследование сложных динамических систем, обладающих большой степенью неопределенности параметров, начальных данных, внешнего воздействия и целей управления, как правило, невозможны классическими методами. В семидесятых годах прошлого столетия для описания и изучения таких задач была предложена теория нечетких множеств и нечеткая логика [1; 2]. При использовании этих теорий вводятся лингвистические термы, которые качественно описывают динамическую систему. Далее с этими термами в соответствии с правилами нечеткой логики можно выполнять различные преобразования, позволяющие получать новые лингвистические высказывания о состоянии системы, которые либо сразу используются для управления системой, либо переводятся в количественные значения для дальнейшего анализа.

На актуальность данного подхода указывает то, что в универсальную систему компьютерной математики MATLAB включен специальный пакет Fuzzy Logic Toolbox (пакет нечеткой логики).

Постановка задачи в терминах нечеткой логики и ее анализ – это сложная процедура, поэтому наглядных примеров, показывающих принципы работы с нечеткими множествами, очень мало. В Интернете на многих сайтах, посвященных нечеткой логике, представлена всего одна задача – задача об управлении работой светофора (http://fuzzyfly.chat.ru/index.htm).

В данной работе на основе теории нечеткой логики был разработан алгоритм и написана программа, управляющая игрой компьютера в карточной игре «Дурак». Интерфейс программы позволяет наблюдать процесс принятия компьютером решения на основе лингвистических правил во время игры. Для тестирования программы создан вспомогательный файл, в котором отражается процесс создания лингвистических термов.

МОДЕЛЬ ИГРЫ

В качестве модели системы, требующей сложного управления, выберем классическую карточную игру в «дурака». Приведем основные правила игры. В игре участвует колода из 36 карт, игрокам раздается по 6 карт, одна карта открывается и она определяет козырную масть, эта карта кладется под колоду. Игра состоит из локальных партий (один игрок ходит, другой отвечает). После розыгрыша локальной партии игроки дополняют свои наборы карт до шести карт из колоды. Побеждает тот игрок, у которого в конце последней локальной партии не остается карт.

Рассмотрим игру двух игроков. В этом случае игру можно классифицировать как антагонистическую игру с нулевой суммой и неполной информацией.

АНАЛИЗ ПРОЦЕССА ИГРЫ

Сложность большинства карточных игр связана с глобальной неопределенностью игры, которая определяется случайным раскладом карт. Очевидно, что число различных распределений карт в колоде $p = 36!$ В начале игры после раздачи карт для игрока не имеет значения порядок полученных карт как своих, так и карт противника, поэтому общее число различных распределений карт $p_0 = \frac{36!}{6!6!}$.

После раздачи карт каждый игрок знает 6 своих карт и козырь, ему неизвестны 29 карт. Следовательно, при выборе хода степень неопределенности игрока определяется числом неизвестных наборов карт у противника: $N = C_{29}^6$ Вероятность реализации одного из наборов $p_i = \frac{1}{C_{29}^6}$, причем все p_i равны. Степень неопределенности каждого игрока будет характеризоваться энтропией [3]

$$H = -\sum_{i=1}^{N} p_i \cdot log_2\ p_i .$$

В начале игры энтропии первого и второго игрока равны $H_{OA} = H_{OB} = log_2\ C_{29}^6$.

Игра носит характер последовательного разыгрывания локальных партий, которые можно разделить на следующие типы по характеру действий, например, первого игрока:

1) ход первого игрока – второй игрок отвечает;

2) ход второго игрока – первый игрок отвечает.

Локальная партия может закончиться как одним ходом с каждой стороны, так и несколькими последовательными ходами.

Игроки в процессе игры принимают решение сделать ход, находясь в состоянии неопределенности. Эта неопределенность носит неустранимый характер и не позволяет учесть связь локальных партий. Поэтому было принято решение каждую локальную партию анализировать отдельно. В этом случае процесс управления игрой компьютера сводится к последовательному анализу этих партий и принятию соответствующих решений в каждой партии.

Рассмотрим множество всех карт $\Omega = \{\omega_1, \omega_2, ...\omega_{36}\}$. В игре участвуют различные подмножества Ω, состоящие из различного количества карт: A_1 – подмножество карт, состоящее из одной карты, A_2 – подмножество карт, состоящее из двух карт, и т. д. Количество всевозможных подмножеств карт может быть вычислено

$$N = C_{36}^1 + C_{36}^2 + ... + C_{36}^{36} = 2^{36} - 1.$$

Каждый элемент множества $A_2 - a_i^{(2)}$ образуется с помощью объединения двух элементов множества A_1, элементы множества A_3 соответственно образуются из элементов множеств A_2 и элементов множества A_1 и т. д. Следовательно, для всех элементов рассматриваемых подмножеств можно ввести операцию объединения, которая ставит в соответствие этим элементам элемент множества более высокого ранга

$$a_l^{(i)} \cup a_k^{(j)} = a_m^{(i+j)},$$

здесь $m(l,k)$ – некоторая зависимость.

Можно ввести операцию разности подмножеств

$$a_l^{(i)} \setminus a_k^{(j)} = a_m^{(i-j)}, i > j.$$

Для дальнейшего анализа игры введем количественную шкалу оценки карт (таблица 1).

Игроки в процессе игры принимают решения сделать ход, находясь в состоянии неопределённости...

Таблица 1.

Карта	Цена	
	для карт простой масти	для карт козырной масти
Шестерка	1	10
Семерка	2	11
Восьмерка	3	12
Девятка	4	13
Десятка	5	14
Валет	6	15
Дама	7	16
Король	8	17
Туз	9	18

В соответствии с приведенной шкалой, определим среднюю ценность одной карты на руках игрока $m = \sum_{i=1}^{k} r_i \Big/ k$, где k – число карт у игрока, r_i – ценность карты в соответствии с введенной шкалой. Средняя цена карты в колоде – 7,25.

На рисуноке 1 показано, как изменяется ценность карт игроков в некоторых партиях. Из приведенных графиков можно видеть, что при правильной игре, приводящей к выигрышу, средняя ценность карт возрастает в процессе игры.

Введенная шкала ценности карт поставила в соответствие множеству карт Ω множество их числовых значений.

ЛИНГВИСТИЧЕСКИЕ ТЕРМЫ И ТАБЛИЦЫ

Во время игры игроки, как правило, явно или неявно пользуются качественной оценкой своих карт и карт противника.

Во время игры игроки... пользуются качественной оценкой своих карт и карт противника.

Определим следующие лингвистические термы.

Определение 1. Обозначим термом «плохие» карты – карты 6, 7, 8, 9, 10 некозырных мастей.

Определение 2. Обозначим термом «средние» карты – карты валет, дама, король, туз некозырной масти.

Определение 3. Обозначим термом «хорошие» карты – все карты козырной масти.

Количественный диапазон изменения «плохих» карт [1–5], «средних» карт [6–9] и «хороших» карт [10–18]. Таким образом, подмножество карт A_1 будет разделено на три подмножества.

Рассмотрим множество различных пар карт A_2. Пара карт «плохая», если каждая карта «плохая», карты «средние», если они состоят из одной «хорошей» и одной «плохой» карты, и карты «хорошие», если обе карты «хорошие»[1].

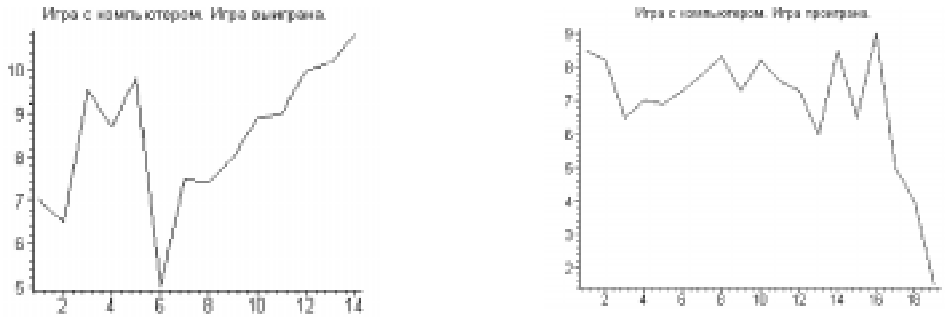


Рисунок 1.

[1] Можно ввести более сложные лингвистические термы, например: «очень хорошие», «хорошие», «средние», «плохие» и «очень плохие» карты.

Введем определение лингвистических термов для любого подмножества карт A_i.

Определение 4. Элемент любого подмножества назовем «хорошим», если количество «хороших» карт, образующих этот элемент, строго больше 2/3 общего количества карт.

Определение 5. Элемент любого подмножества назовем «плохим», если количество «плохих» карт, образующих этот элемент, строго больше 2/3 общего количества карт.

Определение 6. Элемент любого подмножества назовем «средним», если не выполняются условия определения 4 и условия определения 5.

Рассмотрим лингвистическую переменную «Качество расклада». Для реализации лингвистической переменной необходимо определить значения термов этой переменной. В соответствии с выше приведенными определениями зададим ее термами «Плохой», «Средний» и «Хороший».

Значение переменной «Качество расклада» определяется средним значением одной карты расклада. Степень принадлежности расклада к тому или иному терму лингвистической переменной «Качество расклада» определяется так называемой функцией принадлежности $\mu(d)$, где d – средняя стоимость одной карты.

Рассмотрим функцию принадлежности для расклада из шести карт. В таблице 2 приведены диапазоны изменения средней цены карт для раскладов, описываемых соответствующими лингвистическими термами.

Один из возможных видов функции принадлежности показан на рисунке 2.

Приведем пример получения значения лингвистической переменной «Качество расклада». Пусть средняя стоимость одной карты расклада равна 11. В этом случае степень принадлежности к терму «Хороший» равна 0,33, а к терму «Средний» – 0,67 (см. рисунок 2). Конкретное определение степени принадлежности, как правило, осуществляется экспертами. В рассматриваемом случае, по-видимому, лингвистическая переменная «Качество расклада» примет значение «Средний».

Описав карты игрока, ожидаемые карты из колоды и предполагаемые карты противника в лингвистических термах, можно построить управляющие лингвистические выражения. Например: «если у игрока А «хорошие карты», и карта, которая придет из колоды, «хорошая», и, если у игрока В, вероятней всего, «плохие карты», то игрок А должен ходить «хорошей» картой».

Для оценки ожидаемой карты из колоды и расклада карт противника могут быть использованы как теоретические вероятностные оценки, так и оценки, полученные с помощью прямых экспериментов.

Таблица 2.

	Минимальное среднее значение карты	**Максимальное среднее значение карты**
«Плохой» расклад	1,5	6,83
«Средний» расклад	2,83	14
«Хороший» расклад	10,7	15,51

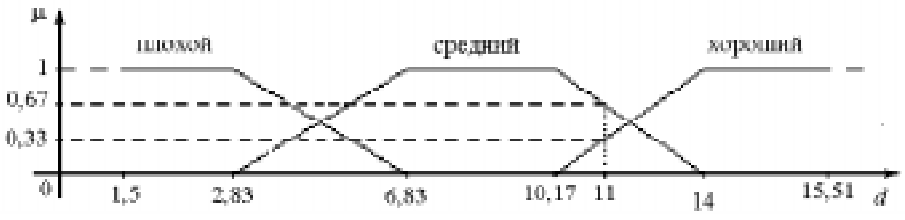


Рисунок 2.

В работе, при создании программы, управляющей игрой компьютера, для оценки карт противника и ожидаемой карты из колоды использовался следующий подход. Эти карты случайным образом выбирались из неизвестных карт колоды (максимальное число неизвестных карт 29). После выбора карт производился перевод их значения в лингвистические термы.

Для управления действиями компьютера, моделирующего одного из игроков, были созданы четыре таблицы лингвистических термов.

1. Таблица, управляющая действиями компьютера при выборе: брать карту со стола или из колоды. После хода игрока (человека), компьютер принимает решение покрыть карту (отдать одну свою карту и взять одну из колоды) или взять (взять карту со стола) (таблица 3).

2. Таблица, управляющая действиями компьютера при выборе карты, которой будет сделан ход. В данном случае таблица состоит из двух частей, так как учитываются два фактора: ожидаемая карта из колоды и предполагаемый расклад противника (таблица 4).

3. Таблица, управляющая действиями компьютера при выборе добавления карты, когда противник берет карту. Если противник берет карту, то компьютер проверяет, стоит добавить карту в придачу, если есть, что добавлять, или нет (таблица 5).

4. Таблица, управляющая действиями компьютера при выборе добавления карты при условии, что игрок карту покрыл. Данная таблица совпадает с предыдущей, но оценки раскладов производятся иначе. В этом случае учитывается возможность того, что игрок покроет данную ему карту и возьмет из колоды карт больше, чем в предыдущем случае, когда было точно известно, что игрок берет все карты со стола (таблица 6).

Таблица 3.

		Расклад после взятия карты из колоды		
		Плохой	**Средний**	**Хороший**
Расклад после взятия карты со стола	**Плохой**	Из колоды	Из колоды	Из колоды
	Средний	Со стола	Из колоды	Из колоды
	Хороший	Со стола	Со стола	Со стола

Таблица 4.

		Расклад до хода		
		Плохой	**Средний**	**Хороший**
Карта, ожидаемая из колоды	**Плохая**	Плохая	Плохая	Плохая
	Средняя	Плохая	Средняя	Средняя
	Хорошая	Плохая	Средняя	*

*

	Предполагаемый расклад игрока		
	Плохой	**Средний**	**Хороший**
Карта, которой будет сделан ход	Хорошая	Средняя	Плохая

Таблица 5.

		Расклад без добавления карт и взятия карты из колоды		
		Плохой	**Средний**	**Хороший**
Расклад после добавления карты и взятия карты из колоды	**Плохой**	Да	Нет	Нет
	Средний	Да	Да	Нет
	Хороший	Да	Да	Нет

Таблица 6.

		Расклад без добавления карт и взятия карты из колоды		
		Плохой	**Средний**	**Хороший**
Расклад после добавления карты и взятия карты из колоды	**Плохой**	Да	Нет	Нет
	Средний	Да	Да	Нет
	Хороший	Да	Да	Нет

Заметим, что в конце игры карты в колоде кончаются, и игра приобретает характер игры с полной информацией.

ОПИСАНИЕ ПРОГРАММЫ FOOL.EXE

Для реализации выше приведенного управления игрой компьютера была написана программа Fool.exe на языке программирования *Microsoft Visual Basic 6.0.* Интерфейс программы позволяет ее использовать как для исследования работы управляющего блока программы, вероятностных характеристик игры, так и для обычной игры в «дурака» (рисунок 3).

ПРИНЦИПИАЛЬНЫЙ АЛГОРИТМ ПРОГРАММЫ

Все игровые карты записываются в двумерный массив $Karta[i, j]$, $i = 1..9$ – номинальное значение карты, $j = 1..4$ – масть карты. В начале игры компьютеру и игроку раздаются наборы из шести карт. Раздача карт в начале и в процессе игры происходит следующим образом: производится проверка наличия карт в колоде, и если карт достаточно, то производится случайная генерация карт из колоды. Для компьютера карты противника (человека) остаются неизвестными и сохраняются вместе с неизвестными картами колоды.

Управление игрой компьютера сводится к последовательному рассмотрению локальных партий. В каждой такой партии ожидаемая карта из колоды и расклад карт противника выбирался случайным образом из неизвестных карт колоды. В соответствии с введенными правилами, компьютер оценивает сгенерированные карты в лингвистических термах и на основе таблиц управления принимает управляющее решение.

Генерация выборок карт происходит периодически с некоторой частотой, то есть за время обдумывания игроком хода компьютер проводит выборку несколько раз, при этом, соответственно, меняются значения лингвистических переменных и, следовательно, меняется решение компьютера.

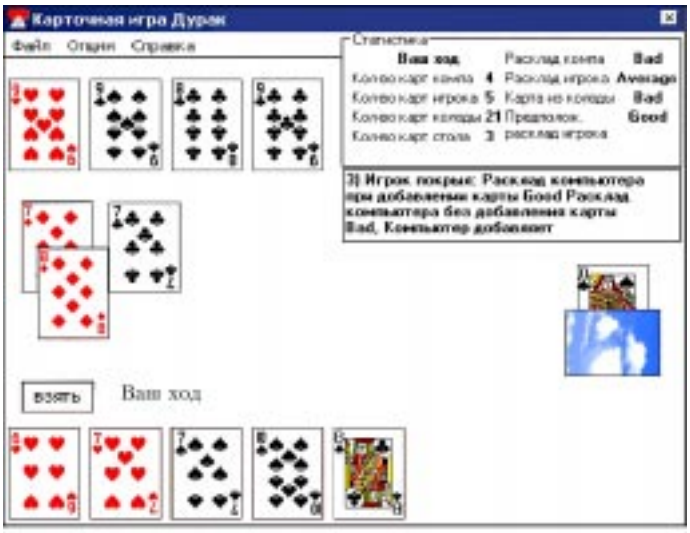


Рисунок 3.

Программирование окончания партии связано с большими практическими трудностями. Существует около $(2^{36} - 1)$ различных вариантов окончания. Поэтому было принято решение, что компьютер и в конце партии действует по правилам нечеткой логики. В этом случае в отличие от основного варианта при покрытии карты и при ходе не учитывается карта из колоды, а при отбивании компьютер кроет по возможности все.

В приложении на диске к журналу приводится подробный алгоритм программы Fool.exe.

ТЕСТИРОВАНИЕ ПРОГРАММЫ

Тестирование программы Fool.exe осуществлялось многократным разыгрыванием игры с различными игроками. Сравнение программы Fool.exe с игровыми программами из Интернета показало, что большинство игровых программ используют простой перебор всех возможных комбинаций карт и выбирают возможную наименьшую карту для хода.

ИНТЕРФЕЙС ПРОГРАММЫ

Интерфейс программы Fool.exe позволяет использовать ее для исследования работы управляющего блока программы, вероятностных характеристик игры и для обычной игры в «дурака».

Используя кнопку меню «Файл» можно начать новую игру, при этом значения всех переменных и расклады карт обнуляются.

Имеется возможность сохранить игру, при нажатии этой кнопки происходит запись массивов карт, козырь на столе, в текстовый файл под названием Расклад.txt. В этот же файл записывается информация компьютера о картах игрока и номер хода. Возможность сохранения игры позволяет рассмотреть варианты действия компьютера при различных ходах игрока при одном и том же раскладе. Файл Расклад.txt создается в основной папке программы Fool.exe.

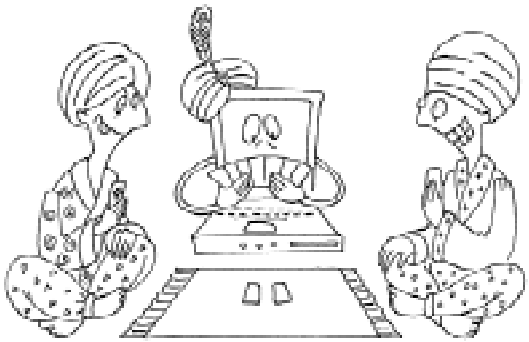
При нажатии кнопки загрузки игры происходит чтение всех значений из файла Расклад.txt. Во время запуска программы создается второй служебный текстовый файл Ходы.txt, в который при каждом ходе записывается значение лингвистических термов и принятое решение компьютера.

Кнопка **«Опции»** позволяет включать окно «статистика», в котором в лингвистических термах отображается текущее состояние компьютера и игрока и основания, по которым компьютер принял решение (см. таблицы 1–4). Эта же кнопка позволяет выбрать режим игры с показом карт компьютера или игровой закрытый вариант.

Кнопка **«Справка»** – в программе имеется файл помощи с кратким описанием основных возможностей программы.

Программа Fool.exe показала свою эффективность и наглядность, с одной стороны, как исследовательская программа, а с другой стороны, – как игровая программа.

В заключение заметим, что данный подход к созданию управляющей программы позволяет поставить вопрос о создании самообучающейся программы. Действительно, можно предусмотреть возможность того, чтобы компьютер, анализируя исходы локальных партий, мог вносить изменения в управляющие таблицы.



Программирование окончания партии связано с большими практическими трудностями.

Литература

1. *Кофман А.* Введение в теорию нечетких множеств. М: Радио и связь, 1982.
2. *Семухин М.В.* Теория нечетких множеств. Учебно-методическое пособие. Тюмень: ТюмГУ, 1999.
3. *Яглом А.М., Яглом И.М.* Вероятность и информация. М, 1973.

Ляхов Александр Федорович,
доцент кафедры теоретической механики механико-математического факультета Нижегородского государственного университета имени Н.И. Лобачевского.

Тришин Илья Михайлович,
студент первого курса механико-математического факультета НГГУ.