Компьютерные инструменты в образовании, 2018
№ 5: 41–51
УДК: 004.8:372.862
http://ipo.spb.ru/journal
doi:10.32603/2071-2340-2018-5-41-51

ПАСЬЯНС «МАХДЖОНГ»: НАУЧНЫЙ ПРОЕКТ ДЛЯ СТАРШЕКЛАССНИКОВ С ЭЛЕМЕНТАМИ ИСКУССТВЕННОГО ИНТЕЛЛЕКТА

Мельников Б. Ф.[1, 2], Мельникова Е. А.[2], Пивнева С. В.[2]

[1]Университет МГУ-ППИ в Шэнчжэне, Шэньчжэн, Китайская Народная Республика
[2]Российский государственный социальный университет, Москва, Россия

Аннотация

Проет, описываемый в настоящей статье, предназначен для исполнения старшеклассниками (а возможно — студентами младших курсов в качестве курсовой работы). Важно отметить, что реализация первой части проекта (компьютерной программы, решающей конкретную раскладку пасьянса) представляет собой весьма незначительную часть работы, предлагаемой для выполнения в этом проекте. Однако и эта часть — создание компьютерной программы для решения конкретной раскладки — тоже является задачей, относящейся к искусственному интеллекту.

В качестве основного критерия качества программы-решателя мы предлагаем использовать процент решения ею случайно сгенерированных раскладок, причём решение должно быть осуществлено без возможности взятия хода назад. Для программирования решения некоторой заданной раскладки мы предлагаем смоделировать процесс решения этой задачи человеком, мышление которого сильно отличается от «мышления» компьютера, в частности, отличатся объёмом запоминаемой информации. Для этого моделирования запрещается, например, программе запоминать фишки, которые уже вышли из игры.

Мы рассматриваем только «заведомо разрешаемые» раскладки. В качестве первого варианта (то есть в качестве начала реализации), мы предлагаем ученику реализовать программу, которая только решает раскладки, полученные случайным заполнением — заполнением пустого поля «с конца». Возможный подход к реализации программы-решателя — применение генетических алгоритмов. Отметим, что даже в этом случае решатель можно назвать небольшой экспертной системой. Мы также кратко описываем в статье некоторые другие области искусственного интеллекта, знание и применение которых возможно в рассматриваемой задаче.

Ранее авторы уже предагали аналогичные проекты студентам младших курсов, было реализовано несколько из них, но бóльшая часть материала, описанного в статье, ещё не реализована, поэтому документ озаглавлен как научный проект для реализации.

Ключевые слова: *оптимизационная задача, пасьянс Мхаджонг, первый шаг в науке, искусственный интеллект.*

Цитирование: Мельников Б. Ф., Мельникова Е. А., Пивнева С. В. Пасьянс «махджонг»: научный проект для старшеклассников с элементами искусственного интеллекта // Компьютерные инструменты в образовании. 2018. № 5. С. 41–51. doi:10.32603/2071-2340-2018-5-41-51

1. ВВЕДЕНИЕ И МОТИВАЦИЯ

Пасьянс «Махджонг» (или, по-другому, «Шанхай», 上海) является одной из самых интересных головоломок, и при этом эта головоломка, по некоторым данным, была изобретена менее 40 лет назад. Стоит отметить, что *пасьянс* Маджонг не следует путать с гораздо более древней азартной игрой с тем же именем. Общего у них, по-видимому, только то, что они используют один и тот же набор фишек, а также, по мнению авторов, они гораздо более интересны и «более насыщенны» по сравнению с более известными европейскими аналогами (то есть с обычными карточными играми, классическим покером и «европейскими» пасьянсами, [1–3] и т. д.)[1].

Не будет большим преувеличением сказать, что для *практически полного* понимания правил пасьянса Маджонг достаточно одной картинки, см. рис. 1, по-видимому, только одной классической раскладки «черепаха» (龜) достаточно для всего описываемого нами проекта. На этом рисунке приведена «европеизированная» версия фишек[2], но нам достаточно знать только то, что в раскладке участвуют 144 фишки, разделяющиеся на 36 вариантов — по 4 фишки для каждого варианта[3].

Рис. 1. Пример классической раскладки «черепаха»

В настоящей статье рассматривается проект, предназначенный для реализации «продвинутыми» старшеклассниками и студентами младших курсов. Как следует из названия проекта, *часть* задания заключается в *создании программы*, предназначенной для решения конкретной раскладки пасьянса. Специально отметим, что эта часть работы, первая её часть (программа для решения раскладки), составляет очень небольшую долю описанных и предлагаемых ученикам действий; хотя, конечно, даже она сама

[1] Мы, однако, *не* считаем, что все китайские интеллектуальные игры (головоломки) лучше «соответствующих» европейских: «китайские шахматы» (сянци, 象棋) европейским сильно уступают . . .

[2] Например, «центральная» (она же «верхняя») фишка на этом рисунке, «красная тройка», соответствует исходному китайскому варианту «три бамбука» (三竹).

[3] Эта картинка взята из программы “Kyodai Mahjongg”, *свободно распространяемая* версия которой может быть загружена с сайта [4]. Соответствующее лицензионное соглашение, находящееся по этому же адресу (оно также доступно при установке бесплатной версии программы), не запрещает использование их изображений с экрана в некоммерческих целях.

содержит некоторые элементы искусственного интеллекта. Общим критерием качества программы решателя предлагается считать процент решённых раскладок (причём решённых «за один проход», без возможности сделать шаг назад, то есть без бэктрекинга). При решении конкретной раскладки ученику предлагается смоделировать процесс решения этой проблемы человеком, а не компьютером. Для этого запрещается запоминать все пары фишек, которые уже покинули игру; точнее, мы разрешаем помнить только некоторые из них (обычно от 2 до 4 пар). Однако «для снижения отрицательного эффекта от предыдущего запрета» и также моделируя процесс решения пасьянса человеком, мы разрешаем программе в качестве *ответа на текущую позицию* раскладки выдавать *последовательность ходов*, что также можно рассматривать как частный случай использования памяти. По нашему мнению, *большинство решателей-людей* думают подобным образом (и обладают примерно такими способностями).

Мы всегда будем рассматривать только «заведомо разрешимые» раскладки. В качестве первого варианта (то есть как начало реализации всего проекта) мы предлагаем ученику просто реализовать программу, которая решает раскладки, полученные *случайным* добавлением пар фишек «с конца», то есть последовательным добавлением пар, начинающимся с пустой доски [4]. Возможным подходом к такой реализации является применение генетических алгоритмов — с дополнительным использованием критериев, сформулированных экспертом (опытным игроком) и внедрённых программистом. Даже в этом случае программу-решатель можно назвать небольшой экспертной системой, — но такой формулировкой рассматриваемый нами проект далеко не ограничен.

Важно отметить, что мы еще не определили (не описали) использованное нами понятие «случайное добавление пар фишек». Пока мы будем предполагать, что среди возможных (то есть ещё не добавленных пар фишек) случайно выбирается одна согласно простейшему равномерному распределению, и также согласно равномерному распределению выбирается одна возможная пара мест для этих фишек.

В этой статье мы очень кратко рассмотрим и некоторые другие области искусственного интеллекта, элементы которых возможны для применения в этой задаче.

- Во-первых, это возможное *изменение* генетического алгоритма *при переходе процесса решения* раскладки (игры): от «дебюта» к «миттельшпилю», а затем — к «эндшпилю».
- Во-вторых, это создание и использование *программ-предикторов*, предназначенных для «угадывания» алгоритма *исходной случайной генерации* раскладок с целью автоматической настройка решателя (например генетического алгоритма) на этот алгоритм генерации.
- В-третьих, *изменение* такого алгоритма *в процессе самого решения*, если, например, в ходе решения появилась информация, противоречащая ранее принятому предположению об исходном алгоритме генерации.

Таким образом, мы в статье кратко описываем несколько таких областей искусственного интеллекта (здесь мы перечислили не все), которые могут быть применены в этой задаче и вполне доступны для реализации школьниками.

Ранее авторы настоящей статьи уже предлагали аналогичные проекты для исполнения студентам младших курсов. При их выполнении уже были получены некоторые

[4] Конечно же, при этом необходимы небольшие дополнительные ограничения, отражающие сами правила пасьянса. Отметим, что такой тривиальный алгоритм формирования раскладки делает её заведомо разрешимой.

достаточно интересные результаты, однако бóльшая часть материала, описанного в статье, школьниками и студентами младших курсов ещё не реализована, поэтому статья и озаглавлена как научный проект для реализации. Кроме того, мы считаем, что в настоящее время появляются всё новые и новые области искусственного интеллекта, и, по мнению авторов, практически любая из этих новых областей может быть применена для реализации алгоритмов, связанных с решением пасьянса Маджонг.

Настоящая статья имеет следующую структуру. В разделе 2 мы кратко рассмотрим некоторые оставшиеся *правила* рассматриваемой головоломки, а также некоторые связанные с нею понятия, необходимые в дальнейшем. В разделе 3 мы приводим точное *описание ограничений*, необходимых для простых вариантов реализации программы-решателя. В разделе 4 мы даем возможный *подход к построению программы-решателя* для заданной раскладки Маджонга. А в заключении (раздел 5) мы рассмотрим некоторые возможные пути дальнейшего развития описываемого нами проекта.

2. ПРЕДВАРИТЕЛЬНЫЕ СВЕ́ДЕНИЯ

В этом разделе мы очень кратко рассмотрим оставшиеся правила рассматриваемой головоломки и некоторые другие понятия, необходимые для нашей статьи. По-видимому, почти все правила пасьянса понятны на основе только одного рис. 1. Дополнительно о правилах головоломки скажем следующее.

- Как мы уже отмечали, имеется 36 «четвёрок», каждая из которых состоит из одних и тех же фишек.
- Зафиксированы 144 (то есть $36 \cdot 4$) места для размещения этих фишек. (Повторим, что мы будем использовать только «самую классическую» раскладку — «черепаху».)
- За один ход мы можем снять 2 *одинаковые* фишки, то есть 2 элемента одной и той же «четвёрки». При этом обе удаляемые фишки должны быть крайними в своих горизонтальных рядах (каждая — либо крайняя левая, либо крайняя правая) и, конечно, не должны быть «накрыты» другими фишками. Например, на приведённом выше рис. 1 ни одна пара из трёх «видимых» синих девяток не может быть удалена.
- Целью игры является удаление всех 144 фишек, никакое «почти полное удаление» ничего не даёт и приравнивается к поражению. Таким образом, головоломка заканчивается только с одним из 2 исходов: «выиграл» или «проиграл», а «промежуточных результатов» не существует.

Пример «хорошего эндшпиля» (в котором осталось 8 ходов до победы) приведён на рис. 2.

Итак, очень сильно упрощая, можно сказать, что цель проекта — следующая: написать прорамму, решающую заданную раскладку Маджонга. Однако эта небольшая формулировка содержит большое число нюансов (как раз и связанных с различными областями искусственного интеллекта). Мы рассмотрим некоторые из них в настоящей статье.

3. ОПИСАНИЕ РАБОТЫ

В этом разделе мы уточняем описание необходимой работы по проекту, начиная с простых вариантов для программы-решателя. Мы рассматриваем несколько вариантов условий этого описания, но нужно обратить внимание, что в принципе *обязательным* ни одно из них *не является*, однако мы всё-таки начнем с такой «полной» версии.

Рис. 2. «Хороший» эндшпиль

- Программа-решатель в процессе решения раскладки обладает той же самой информацией, что и решатель-человек.
- Мы будем рассматривать только *заведомо решаемые* раскладки[5]. А *как* получить (сгенерировать) такую раскладку, которая заведомо решаема? Простейший метод для этого очевиден: мы последовательно добавляем на пустые поля (не прерывая формируемые ряды) пары фишек до полного заполнения всех мест «черепахи».
- В простейшей версии это реализуется следующим образом[6]:
  - во-первых, мы случайным образом выбираем пару фишек (при этом используя *равномерное распределение* среди ещё не выбранных пар, иными словами, выбирая с равной вероятностью каждую из тех пар фишек, которые возможны для добавления);
  - затем мы случайным образом выбираем пару ещё не заполненных мест в «черепахе» (эта пара также может быть выбрана с помощью равномерного распределения среди пар ещё не выбранных мест)[7], следя при этом за тем, чтобы не получалось разрыва ни в каком формируемом горизонтальном ряду
  - выбранные места заполняются выбранными фишками;
  - и так далее . . .

Стоит отметить следующее. Мы *уже* описали простейший вариант указаний, в соответствии с которыми возможно создание программы-решателя. Однако именно в таком виде программа-решатель представляется малоинтересной: практика показывает, что совсем несложно написать программу, решающую около 90% от заданных таким образом «черепах». Мы же описыаем проект, в котором предлагаем школьнику создать компьютерную программу, *имитирующую мышление решателя-человека*. (Точнее можно сказать так: описывая некоторые примитивы, которые, как мы полагаем, в чём-то

[5] Заметим, что в вышеупомянутой программе “Kyodai Mahjongg” мы для генерации таких раскладок можем установить соответствующий параметр («флажок»).

[6] Заметим, что в ходе последовательного усложнения проекта этот мини-алгоритм будет изменён.

[7] Существует несколько возможностей описанной здесь реализации, однако это, по-видимому, не очень принципиально.

похожи на применяемые человеком, мы на *основе программирования этих примитивов и их применения* пытаемся создать наиболее успешную программу-решатель.)

Основным различием между мышлением человека и «мышлением компьютера» (различием, относящимся к нашей задаче) можно считать ограничение по памяти. Это приводит нас к следующим ограничениям на разрабатываемые программы-решатели:

- Самое важное: *мы не будем запоминать вышедшие из игры фишки!* (Подробности см. далее.)
- Мы будем использовать только «конечную память»: а именно, мы позволим программе-решателю запоминать только 3 числа в диапазоне от 1 до 36 (то есть 3 *цифры* в системе счисления с основанием 36). Такие «ячейки памяти» могут быть использованы программой, например, для хранения информации о тех оставшихся фишках, которые, «по её мнению», для процесса дальнейшего решения «более важны», чем остальные [8].

Итак, мы можем предположить, что на вход программы-решателя подаётся некоторое *состояние* решаемой раскладки (то есть «из черепахи» уже удалены некоторые пары фишек, однако мы, вообще говоря, не знаем, какие именно.) Кроме того, входные данные включают 3 числа от 1 до 36. На выходе программы должна быть пара удаляемых фишек, при этом путём последовательного применения такой программы мы получаем искомую программу-решатель.

Отметим ещё раз следующее важное обстоятельство. Наша цель — не создавать программу, которая решает наибольшее количество заданных ей раскладок; целью является создание такой программы *в случае специально сформулированных ограничений*, которую мы считаем *моделированием человеческого мышления*! Высокий процент решённых раскладок — это тоже требование к программе, но требование, имеющее второй приоритет.

Продолжим описание наших ограничений (требований к программе), которые также (с нашей точки зрения) моделируют процесс решения некоторой заданной раскладки человеком.

- В качестве ещё одного «послабления» к вышеупомянутому «отсутствию памяти у программы-решателя» (также, с нашей точки зрения, проявляющегося у решателя-человека) мы разрешим программе такую возможность: её результатом (выходными данными) может быть не одна пара удаляемых фишек, а некоторая *последовательность* таких пар. В этом случае, конечно, все фишки всех таких пар должны быть «видны» в заданной программе позиции; применяя аналогию с шахматами и им подобными играми, это можно назвать «форсированным вариантом игры». Приведём пример такой «видимой» в заданной позиции (рис. 1) последовательности: красные тройки, потом жёлтые восьмёрки, потом красные девятки [9].

[8] «Почему используются именно 3 элемента памяти?» — «А почему бы и нет?..» Мы считаем, что это ограничение также в чём-то похоже на использование памяти человеком, решающим пасьянс: обычно его мозг «загружен другими вычислениями», представляющимися в текущий момент решения более важными. Поэтому запоминать большое количество вышедших пар фишек способны далеко не все решатели-люди, либо такое запоминание вредит самому процессу решения.

[9] Отметим, что мы выбрали такой пример, в котором для каждой снимаемой пары фишек имеется единственный вариант выбора — то есть остальные 2 фишки четвёрки не снимаемы. В противном случае надо дополнительно указывать, *какие именно* две фишки нужно снимать — реализацию этой возможности тоже должен провести разработчик программы.

4. ПОДХОД К РЕАЛИЗАЦИИ ПРОГРАММЫ-РЕШАТЕЛЯ

В этом разделе мы приводим возможный подход к построению программы-решателя для заданной раскладки Махджонга. Как мы уже отмечали выше, цель состоит *не в том*, чтобы создать программу, решающую наибольшее количество заданных раскладок. Однако, если ограничения на программу *уже даны* (мы имеем в виду ограничения на работу программы, операции с данными, использование памяти и т. д.), то в этом случае, наоборот, *нужно* создать программу, решающую наибольшее количество заданных раскладок — столько, сколько получится.

Заранее отметим следующее. По мнению авторов, в последние годы в самых разных интеллектуальных программах преобладают подходы, которые можно считать представителями так называемых “non rule-based approaches” (то есть подходы, *не* использующие предварительные знания эксперта-человека)[10]. Однако мы не забываем и *подход, основанный на правилах* (rule-based approach), и в данном проекте мы попытаемся описать последовательность действий, приводящую к алгоритму (программе), «расположенному между» этими двумя подходами; это можно называть *гибридной моделью*, см. [8, 9] и многочисленные последующие публикации.

Итак, разрабатываемая программа может быть построена, например, с помощью такого подхода:

- Вводятся некоторые «критерии оценки качества», которые считаются для каждой из крайних (то есть возможных к снятию в данный момент) фишек. (Вот несколько простых примеров: ширина ряда, в котором лежит фишка, «высота» этого ряда. В обоих этих случаях значение критерия тем больше, чем больше входное значение, то есть ширина либо «высота».) При этом выбор критериев сам по себе является *частью предмета* нашего проекта.
- Для рассматриваемой позиции мы суммируем все критерии; при этом суммы считаются для любой возможной пары (то есть возможной для снятия пары фишек). Пара, получившая в результате наибольшее значение суммы критериев, возвращается программой-решателем в качестве ответа.
- При этом программист как-то должен предусмотреть возможность возвращать в качестве ответа и *последовательность* пар фишек. (О такой возможности было сказано ранее, мы назвали это «форсированной игрой».)

И, конечно, созданная программа должна быть самообучающейся, то есть в процессе своей работы (рассматривая и решая различные раскладки) последовательно улучшать коэффициенты, применяемые в критериях. Целевая функция получающегося оптимизационного алгоритма при этом очевидна: увеличение доли решаемых программой раскладок.

5. ЗАКЛЮЧЕНИЕ

В предыдущих разделах мы уже упоминали — явно или неявно — некоторые области искусственного интеллекта, связанные с рассматриваемым нами проектом. Отметим еще раз, что в нём предлагается использовать подходы как «основанные на правилах» (rule-based), так и им противоположные; точнее, в проекте предполагается использовать *промежуточные* подходы.

[10] Включая так называемое модельно-ориентированное проектирование (model based design, model based systems) и проектирование на основе прецедентов (case based systems), см. [6, 7] и др.

Рассмотрим некоторые возможные варианты дальнейшего развития описываемого проекта.

«С алгоритмической точки зрения» разработка нашего проекта — хорошая возможность для применения генетических алгоритмов. Геномом здесь можно считать набор критериев, обсуждавшихся выше, и некоторых других. (Весьма просто, кратко и интересно о генетических алгоритмах написано в [10], хотя и со многими орфографическими ошибками; см. также Википедию и многие другие доступные источники.) Но ещё более интересным является не алгоритмическое совершенствование программы-решателя, а модификация *алгоритмов генерации* стартовых раскладок.

Например, можно специально генерировать такие раскладки, в которых какие-либо две одинаковые фишки находятся «одна над другой». Но это самый простой вариант так называемых “badness”, а описание и реализация других вариантов подобных badness также входит в тему проекта, но при этом не стоит забывать, что, несмотря на все эти badness, все генерируемые раскладки должны быть *решаемыми* (см. выше).

Выше мы также упоминали про то, что наш проект можно рассматривать как *промежуточный* подход, комбинацию подходов, основанных на правилах (rule-based), а также им противоположных. Однако выше практически не было примеров подходов, основанных на правилах (предлагавшиеся генетические алгоритмы надо рассматривать как пример противоположного подхода), поэтому приведём один из вспомогательных алгоритмов, которые могут быть названы rule-based.

Подобной возможностью, (ранее уже реализованной одним из наших предыдущих студентов-дипломников, однако ещё не включённой в общий комплекс алгоритмов для рассматриваемой задачи) можно считать следующую эвристику. Мы можем в какой-то определённый момент сделать не самый хороший («с объективной точки зрения» либо «с точки зрения описанных выше предпочтений» [11]) ход , но при этом получая от него какие-нибудь иные преимущества [12]. Например мы можем добавить *правило* о том, чтобы выполнить не самый удачный, с нашей точки зрения, ход, чтобы получить возможность снять сразу 4 фишки. (Напомним, что памяти у нас практически нет, и информацию про снятые 2 фишки мы вскоре забываем.)

Опишем ещё одну возможность комбинирования подходов. Созданная ранее одним из наших студентов программа *интеллектуального анализа сгенерированных программой “Kyodai Mahjongg” раскладок* [13] показала, что в процессе генерации используются 6 вспомогательных алгоритмов. Поэтому в процессе разработки описанной программы можно создать вспомогательный алгоритм (назовём его А), с приемлемой вероятностью определяющий по заданной раскладке применённый для неё «чужой» алгоритм генерации, и на основе ответа алгоритма А *скорректировать применяемый алгоритм решения*

[11] Не будет преувеличением сказать, что одна из целей проекта — «совмещение» этих двух понятий.

[12] Выше мы уже проводили одну аналогию с шахматами (форсированные ходы), несмотря на то, что шахматы — игра, а мы рассматриваем головоломку, “puzzle”. А здесь возникает ещё одна аналогия, а именно, жертва (причём, скорее позиционная), «гамбит»: за счёт каких-то сознательных ухудшений (в шахматах обычно за материальные уступки) мы приобретаем что-то иное (возможность активной игрыи др.).

[13] Входом программы являлось *большое* множество раскладок, сгенерированных этой программой. Необходимы также следующие замечания. Во-первых, мы рассматриваем такую программу отдельно от всех остальных программ нашего проекта. Во-вторых, она была создана полностью на основе нейросетевого подхода. В-третьих, в соответствии с упомянутым выше лицензионным соглашением, *мы не имеем права анализировать код* программы “Kyodai Mahjongg”, но мы этого и не делали, ограничиваясь анализом результатов её работы.

раскладки[14]. Важной частью темы рассматриваемого нами проекта является описание и реализация других вариантов badness'а — как аналогичных рассматриваемому выше примеру, так и мало похожих на него. Ещё раз отметим, что при этом все генерируемые раскладки должны быть разрешимыми (см. выше).

Список литературы

1. *Law M., Deane G.* General Card Game Playing. URL: https://www.doc.ic.ac.uk/teaching/distinguished-projects/2013/m.law.pdf (Retrieved: 06.09.2018).
2. *Law M., Russo A., Broda K.* The complexity and generality of learning answer set programs // Artificial Intelligence. 2018. № 259. P. 110–146.
3. *Brown N. Sandholm T.* Superhuman AI for heads-up no-limit poker: Libratus beats top professionals // Science (American Association for the Advancement of Science). 2017. № 359 (6374). doi:10.1126/science.aao1733
4. Cyna Games. URL: http://cynagames.com/ (Retrieved: 06.09.2018).
5. *Yen Sh.-J., Chen J.-C., Yang T.-N., Hsu Sh.-Ch.* Computer Chinese Chess // ICGA journal. 2004. № 27 (1). P. 3–18.
6. *Luger G.* Artificial Intelligence: Structures and Strategies for Complex Problem Solving. Boston: Pearson Education, 2008.
7. *Russel S., Norvig P.* Artificial Intelligence: A Modern Approach. Prentice Hall, NJ. 2010.
8. *Ben-Bassat M., Beniaminy I., Joseph D.* Combining Model-Based and Case-Based Expert Systems. In: Research Perspectives and Case Studies in System Test and Diagnosis. Frontiers in Electronic Testing // Research Perspectives and Case Studies in System Test and Diagnosis. Frontiers in Electronic Testing. 1998. Vol. 13. P. 179–199.
9. *Weiss M., Indurkhya S.* Rule-based Machine Learning Methods for Functional Prediction // Journal of Artificial Intelligence Research, 1995. № 3. doi:10.1613/jair.199
10. Генетический алгоритм. Просто о сложном. URL: https://habr.com/post/128704/ (дата обращения: 06.09.2018).
11. *Мельников Б. Ф., Пивнева С. В., Рогова О. А.* Репрезентативность случайно сгенерированных недетерминированных конечных автоматов с точки зрения соответствующих базисных автоматов // Стохастическая оптимизация в информатике. 2010. № 6 (1-1). С. 74–82.
12. *Мельников Б. Ф., Сайфуллина Е. Ф.* Применение мультиэвристического подхода для случайной генерации графа с заданным вектором степеней // Известия высших учебных заведений. Поволжский регион. Физико-математические науки. 2013. № 3 (27). С. 70–83.

Поступила в редакцию 12.07.2018, окончательный вариант — 06.09.2018.

[14] А в «более научных» областях (то есть *не* связанных с головоломками и играми) аналогичные подходы уже были применены аспирантами, работавшими под руководством авторов настоящей статьи: см. [11] про недетерминированные конечные автоматы и [12] про графы.

Computer tools in education, 2018
№ 5: 41–51
http://ipo.spb.ru/journal
doi:10.32603/2071-2340-2018-5-41-51

MAHJONGG SOLITAIRE: A HIGH-SCHOOL STUDENT SCIENTIFIC PROJECT, CONTAINING ELEMENTS OF ARTIFICIAL INTELLIGENCE

Melnikov B. F.[1, 2], Melnikova E. A.[2], Pivneva S. V.[2]

[1]Shenzhen MSU-BIT University, Shenzhen, China
[2]Russian State Social University, Moscow, Russia

Abstract

The project described in this article is intended to be performed by high school students (and possibly by undergraduate students as a term paper).

It is important to note that the implementation of the first part of the project (the computer program that solves the specific layout of the solitaire) is a very small part of the work proposed for implementation in this project. However, this part — the creation of a computer program for solving a particular layout - is also a task related to artificial intelligence.

As a general criterion for the quality of the solver program, we propose to use the percentage of the solved layouts — which are randomly generated by the program — with the condition that the solution was found without the possibility of taking a step back. To program the solution of a given layout, we propose to model the process of solving this problem by a person — whose thinking is very different from the computer's "thinking", in particular, it will differ in the amount of information stored. For this simulation, for example, we prohibit the program to memorize the chips that have already left the game.

We consider only "knowingly solvable" solitaire layouts. As the first option (that is, the beginning for implementation), we suggest to the student to implement a program that solves the layouts obtained by random generation "from the end". A possible approach for the implementation of the solver program is the application of genetic algorithms. Note that even in this case, the solver can be called a small expert system. We also briefly describe in the article some other areas of artificial intelligence, the knowledge and application of which is possible in the task at hand.

Previously, the authors had already presented similar projects to undergraduate students, several of them were implemented, but most of the material described in the article has not yet been implemented; therefore, the document is entitled as a science project for implementation.

Keywords: *Mahjongg solitaire, optimization problem, the first step in science, artificial intelligence.*

Citation: B. F. Melnikov, E. A. Melnikova, and S. V. Pivneva, "Mahjongg Solitaire: a High-School Student Scientific Project, Containing Elements of Artificial Intelligence", *Computer tools in education*, no. 5, pp. 41–51, 2018 (in Russian). doi:10.32603/2071-2340-2018-5-41-51

Received 12.07.2018, the final version — 06.09.2018.

Boris F. Melnikov, Dr. habil., Professor of Shenzhen MSU-BIT University bf-melnikov@yandex.ru
Elena A. Melnikova, Assistant Professor; 129226, Russia, Moscow, Wilhelm Pieck str., 4-1, ya.e.melnikova@yandex.ru
Svetlana V. Pivneva, Assistant Professor, tlt-swetlana@yandex.ru

Мельников Борис Феликсович,
доктор физико математических наук,
профессор Университета МГУ-ППИ
в Шэнчжэне, профессор Российского
государственного социального
университета,
bf-melnikov@yandex.ru

Мельникова Елена Анатольевна,
кандидат физико математических наук,
доцент Российского государственного
социального университета; 129226, Москва,
ул. Вильгельма Пика, дом 4, стр. 1,
ya.e.melnikova@yandex.ru

Пивнева Светлана Валентиновна,
кандидат педагогических наук, доцент
Российского государственного социального
университета,
tlt-swetlana@yandex.ru