УДК 004.89
DOI: 10.15827/0236-235X.125.012-019
Дата подачи статьи: 15.11.18
2019. Т. 32. № 1. С. 012–019

Алгоритмическое и программное обеспечение когнитивного агента на основе методологии Д. Пойа

С.С. Курбатов [1], *ведущий научный сотрудник, curbatow.serg@yandex.ru*
И.Б. Фоминых [2], *д.т.н., профессор, igborfomin@mail.ru*
А.Б. Воробьев [2], *аспирант, abvorobyev@bk.ru*

[1] *Научно-исследовательский центр электронной вычислительной техники, г. Москва, 117218, Россия*
[2] *Национальный исследовательский университет «МЭИ», г. Москва, 111250, Россия*

В статье описывается оригинальный подход к созданию интегральной системы решения задач. Система (когнитивный агент) предполагает тесную интеграцию этапов лингвистической обработки, онтологического представления задачи, эвристически-ориентированного решения и концептуальной визуализации. Концепция системы базируется на методологии Пойа, но в трактовке алгоритмического и программного воплощения. Система реализована в макетном варианте и протестирована в предметной области «школьная геометрия».

Лингвистическая составляющая системы использует метод получения канонического описания задачи путем перефразирования и отображения в семантическую структуру.

Автоматический поиск решения основан на выполнении правил, отражающих аксиоматику соответствующих предметных областей. Выбор правил при поиске решения определяется эвристиками, представленными в онтологии. Эвристики оформлены как структуры семантической сети, что позволяет организовать многоаспектный поиск подходящего правила, а также обоснование выбора в виде естественно-языкового комментария.

Концептуальная (когнитивная) визуализация обеспечивает наглядное отображение решения путем интерпретации текстового файла, содержащего информацию для вывода графических объектов, а также комментарии о процессе решения. Комментарии включают естественно-языковое описание правил (аксиом, теорем), эвристические и эмпирические обоснования их выбора, а также ссылки на визуализируемые объекты.

Проведены эксперименты, демонстрирующие возможности визуализации как чертежей задач, так и фрагментов онтологии, фраз естественного языка, формул математики, в том числе формальной логики. Онтология реализована в программной среде СУБД Progress. Программы визуализации реализованы на JavaScript с использованием JSXGraph и MathJax. Реализация обеспечивает возможность пошагового просмотра решения в различных направлениях с динамическим изменением чертежа и соответствующих комментариев. Разнообразная модификация пользователем чертежа с сохранением условий задачи позволяет эмпирически продемонстрировать корректность условий.

Результаты эксперимента интерпретированы, намечено исследование, развивающее описанный подход.

Ключевые слова: *интегральная система, когнитивный агент, естественно-языковой интерфейс, онтология предметной области, визуализация решения, школьная геометрия.*

В настоящее время в области *искусственного интеллекта* (ИИ) достигнуты значимые результаты в таких направлениях, как обработка естественного языка (включая генерацию лингвистических трансляторов [1], методы глубокого обучения [2]), онтологии, автоматическое решение задач и концептуальные средства визуализации [3, 4]. Однако системы, интегрирующие достигнутые результаты, пока далеки от идейной и технологической зрелости. Целью данной работы являются исследование и разработка возможностей интеграции в перечисленных направлениях, и именно этим определяется ее актуальность. Разумеется, интеграция в широком понимании предполагает различные источники информации (например видео), автономное функционирование в реальном времени (роботы) и т.п., что значительно выходит за рамки данного исследования.

Результаты на отдельных этапах (лингвистический, онтологический, этап решения, визуализация) с трудом интегрируются в целостную систему, особенно при решении не слишком тривиальных задач. Выявление причин ошибок на отдельных этапах обработки и, тем более, их исправление серьезно осложняются разнородностью методов. Новизна исследования состоит в разработке унифицированных механизмов обработки на всех этапах функционирования системы, обеспечивающих качественно новый уровень интеграции.

Концепция интегральной системы (когнитивного агента), включающей естественно-языковой интерфейс, эвристически-ориентированный решатель и концептуальную визуализацию, дана в [5]. Система базируется на методологии известного ученого и педагога Пойа [6], но его рекомендации адресованы специалисту-человеку, в то время как на уровне данного исследования они рассматриваются в аспектах алгоритмизации, программного воплощения и представления в базе знаний.

Визуализация в системе базируется на универсальном базовом механизме, формирующем текстовый файл, который используется далее интерпретаторами для графического отображения. Файл может содержать данные для чертежа, математических формул, естественно-языковых описаний, 3D-графики [7] и т.д.

Специфика соответствующих интерпретаторов описывается в онтологии и учитывается базовым механизмом при формировании текстового файла.

Весьма развитая система автоматического решения задач разработана в [3, 4], где описаны более 40 000 математических приемов, намечены пути создания интеллектуальной версии, способной к обучению по источникам. Однако разработчиков в основном интересовали масштабность системы и собственно этап решения задачи. Вопросам организации базы знаний, иерархической организации эвристик, лингвистической поддержки решения (естественно-языковой ввод формулировок задачи и организация уточняющего диалога, ЕЯ-описание решения и обоснование эмпирических догадок) и, наконец, развитой когнитивной визуализации уделялось значительно меньше внимания. Но именно эти вопросы являются центральными в предлагаемой концепции интегральной системы.

Основная гипотеза состоит в том, что компьютерное воплощение методологии Пойа позволит получить знания, помогающие понять когнитивные механизмы и модели, используемые человеком при решении задач. В свою очередь, в практическом плане эти знания дадут возможность проектировать качественно более совершенные обучающие системы.

Когнитивный агент

В работе ставится задача разработки и исследования алгоритмического и программного обеспечения когнитивного агента на основе методологии Д. Пойа. Концепция когнитивного агента (интегральной системы решения задач) предложена в [5], она включает лингвистический транслятор для получения онтологического представления задачи, сформулированной на *естественном языке* (ЕЯ), эвристически-ориентированный решатель и подсистему визуализации решения в виде чертежа, формул и комментария с обоснованием шагов решения. Разработка обеспечивает тесную интеграцию алгоритмов и программ лингвистической обработки, решения задач и визуализации.

Такая интеграция предполагает возможность на каждом этапе функционирования агента информировать об истории получения визуализированного объекта (графического образа, элемента чертежа, формулы, онтологической структуры, синтаксической структуры, переформулировки задачи и т.п.) вплоть до исходного ЕЯ-описания как первичного документа.

В процессе разработки использовались методы искусственного интеллекта, когнитивного анализа и компьютерной графики.

Были разработаны алгоритмы когнитивного агента, значительная часть которых реализована в программном макете. Тестирование агента выполнялось в предметной области «школьная геометрия» большей частью на уровне программных экспериментов. Некоторые алгоритмы тестировались в режиме автономной отладки отдельных компонент. Макетный вариант системы (рис. 1) был программно реализован с использованием инструментальных возможностей СУБД Progress [8] (лингвистическая трансляция, онтология, решатель), визуализация реализована на javascipt с использованием библиотек jsxgraph [9] и mathjax [10]. Далее описываются общая схема системы, функционирование отдельных компонент, результаты эксперимента и их интерпретация. Общая схема системы приведена на рисунке 1.

На входе система получает естественно-языковое описание задачи, которое после лингвистической обработки переводится в онтологическую структуру. Решатель системы на основе правил предметной области (аксиом, теорем и базовых операций) осуществляет

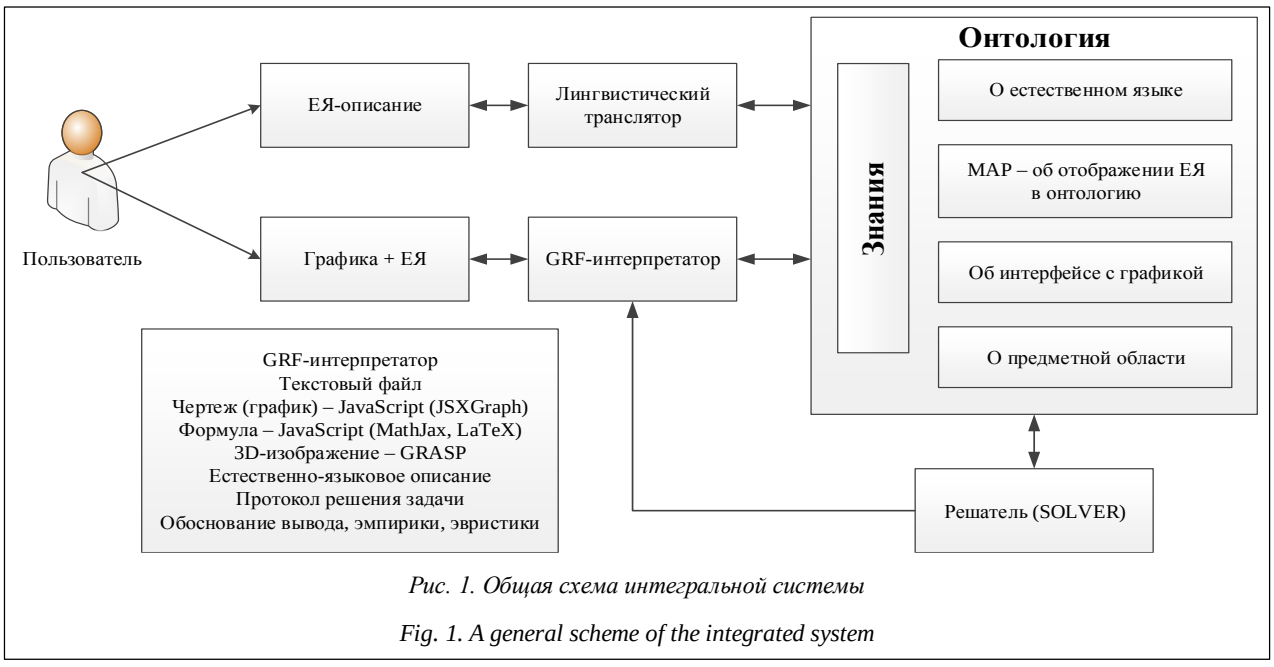


Рис. 1. Общая схема интегральной системы

Fig. 1. A general scheme of the integrated system

поиск решения (построения, доказательства) и в случае успеха формирует текстовый файл для визуализации. Файл содержит протокол с шагами решения и данные для графической интерпретации (чертеж, формула, ЕЯ-описание примененного правила и его эвристического обоснования).

Лингвистический транслятор использует стандартные методы морфологического анализа, а построение синтаксической структуры базируется на отношениях, в основном соответствующих вопросительным словам естественного языка (что, когда, где и т.д.). Отображение синтаксических структур в семантику предметной области (например геометрии) выполняется с помощью адаптированного для целей системы метода перефразирования [11]. Далее приведен пример простейшего ЕЯ-описания, отображаемого в единственное семантическое представление (в скобках даны английские эквиваленты):

– прямая, проходящая через точку (a straight line passing through a point);

– прямая проходит через точку (the line passes through the point);

– прямая, которая проходит через точку (a straight line that passes through a point);

– прямая, которой принадлежит точка (the line to which the point belongs);

– точка на прямой (the line to which the point belongs, point on the line);

– точка, принадлежащая прямой (point belonging to a straight line);

– точка принадлежит прямой (the point belongs to the line);

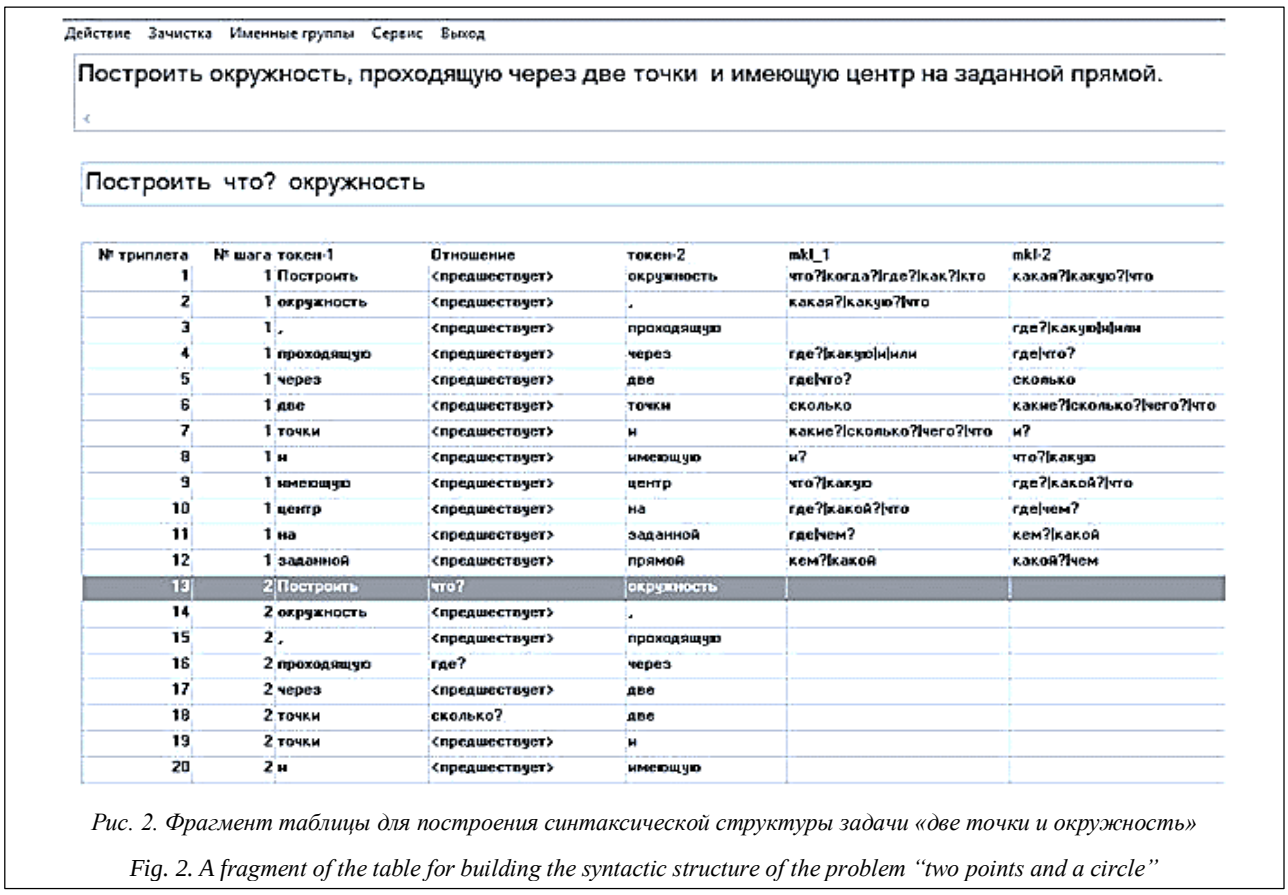
– точка находится на прямой (point is on a straight line);

– точка, которая принадлежит прямой (the point that belongs to the line).

Все эти ЕЯ-описания порождают одну концептуальную структуру: <point ON line>.

Правила перефразирования представлены в онтологии с помощью таблиц БД с соответствующими левыми и правыми частями. Синтаксические структуры также представлены в таблицах. Процесс перефразирования заканчивается, когда удается получить каноническую структуру. Последняя непосредственно отображается в концептуальную (семантическую) структуру типа <point ON line>. Далее с семантической структурой работает решатель, вызывая с помощью эвристик правила, пополняющие эту структуру. После получения целевой структуры (решения) формируется текстовый файл, передаваемый интерпретатору для визуализации.

Пример табличного представления реально формируемой синтаксической структуры для задачи «две точки и окружность» приведен на рисунке 2. Проблемы синтаксического и семантического анализа весьма сложны и в рамках интегральной системы ре-



№ триплета	№ шага	токен-1	Отношение	токен-2	mkl_1	mkl-2
1	1	Построить	<предшествует>	окружность	что?|когда?|где?|как?|кто	какая?|какую?|что
2	1	окружность	<предшествует>	,	какая?|какую?|что	
3	1	,	<предшествует>	проходящую		где?|какую|и|или
4	1	проходящую	<предшествует>	через	где?|какую|и|или	где|что?
5	1	через	<предшествует>	две	где|что?	сколько
6	1	две	<предшествует>	точки	сколько	какие?|сколько?|чего?|что
7	1	точки	<предшествует>	и	какие?|сколько?|чего?|что	и?
8	1	и	<предшествует>	имеющую	и?	что?|какую
9	1	имеющую	<предшествует>	центр	что?|какую	где?|какой?|что
10	1	центр	<предшествует>	на	где?|какой?|что	где|чем?
11	1	на	<предшествует>	заданной	где|чем?	кем?|какой
12	1	заданной	<предшествует>	прямой	кем?|какой	какой?|чем
13	2	Построить	что?	окружность		
14	2	окружность	<предшествует>	,		
15	2	,	<предшествует>	проходящую		
16	2	проходящую	где?	через		
17	2	через	<предшествует>	две		
18	2	точки	сколько?	две		
19	2	точки	<предшествует>	и		
20	2	и	<предшествует>	имеющую		

Рис. 2. Фрагмент таблицы для построения синтаксической структуры задачи «две точки и окружность»

Fig. 2. A fragment of the table for building the syntactic structure of the problem "two points and a circle"

шались в достаточно ограниченном контексте. В ряде случаев они решались ad hoc в соответствии с общими целями интегральности. Отметим, что проблема автоматического синтаксического анализа в лингвистике еще не получила унифицированного решения и поиски подходов для повышения качества анализа продолжаются [12].

Фрагмент таблицы на рисунке 2 демонстрирует процесс построения синтаксической структуры с использованием вопросительных слов и морфологических классов (вопросительные слова, относящиеся к этим классам, перечислены в двух последних столбцах фрагмента). Системное отношение <предшествует> определяет порядок токенов (лексем) в исходном ЕЯ-описании. В целом при построении синтаксической структуры в системе комбинируются известные методы дерева зависимостей и дерева составляющих.

Алгоритм построения многопроходной: по мере построения текущих элементов структуры они маркируются (не отражено на фрагменте) и далее используются только крупные составляющие. Например, связь «где?» от предлога «через» устанавливается только после формирования целостной именной группы «две точки». Ограничения на установление связей записываются в онтологии, что облегчает их редактирование и обеспечивает объяснительные возможности системы.

Семантическая структура для данного ЕЯ-описания имеет вид триплетов:
<*point_A* ON *circle*>, <*point_B* ON *circle*>, <*circle* HAS_A_CENTER *point_C* >,
<*point_C* ON *line*>, <*point_C* HAVE_THE_STATUS ?>

Структура приведена в упрощенной нотации, реальные триплеты записываются в строке БД с рядом дополнительных полей: типы объектов (точка, прямая), имя объекта, статус объекта (задан или требует нахождения) и т.д. Статус точки С (центр окружности) помечен знаком вопроса, то есть требует нахождения. Онтологический решатель с помощью правил (в данном случае правил построения) предметной области, выбираемых по эвристическим критериям, расширяет исходную структуру до смены статуса точки С.

Если статус оказывается измененным, решение считается полученным и по примененным правилам формируется протокол. Он содержит последовательность примененных правил, данные для визуализации чертежа и комментарии к решению. Пример визуализации задачи «две точки и окружность», полученной в ранней версии системы (с помощью макросов MS Word), дан на рисунке 3.

Громоздкая онтологическая структура решения приведена в [13]. В этой версии онтология была неполна и не учитывались граничные условия. В текущей версии система учитывает три фиксированных в онтологии взаимоисключающих отношения между прямыми: различны и пересекаются, параллельны, совпадают. На основе этих отношений система формирует три возможных случая: единственное решение, нет решения, бесконечное множество решений. В последнем случае система на чертеже отображает несколько решений, а в комментариях отражает, что каждая точка на заданной прямой может быть центром окружности.

Онтология, правила, эвристики

Онтология в системе включает два механизма: выбор правила и выполнение правила. Выбор правила –

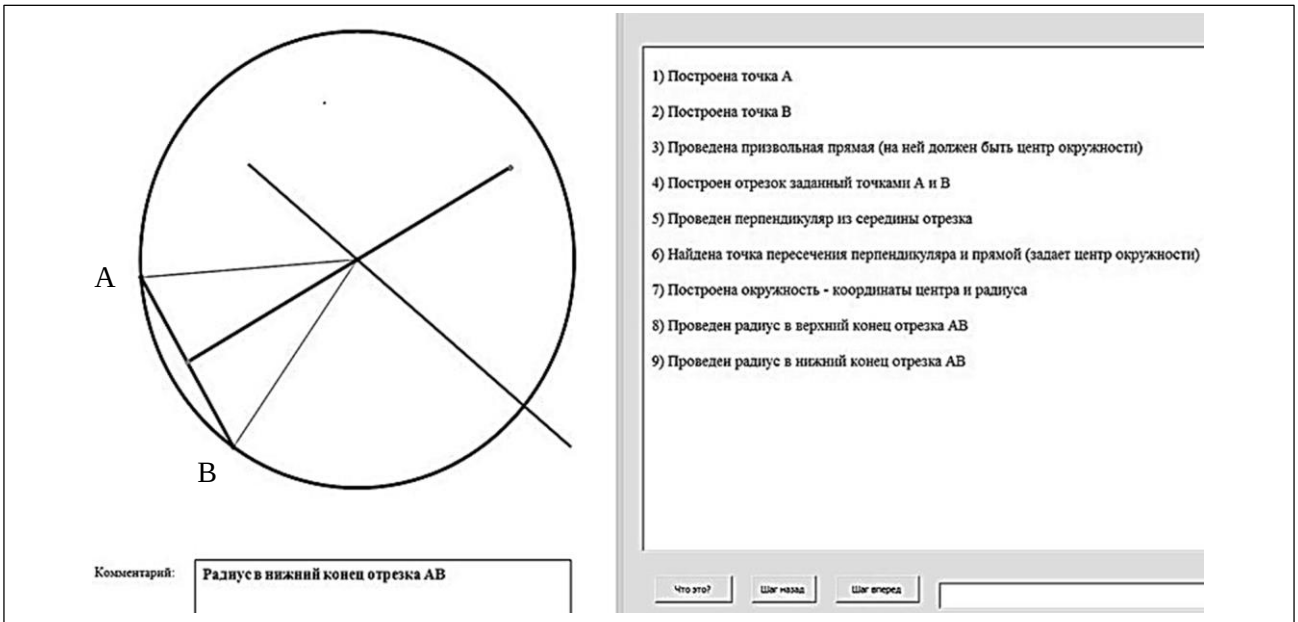


Рис. 3. Пример визуализации задачи «две точки и окружность»

Fig. 3. An visualization example of the problem "two points and a circle"

это эвристическая составляющая. Она не гарантирует результат, но может существенно сократить перебор для получения оптимизированного решения. Выполнение правила – дедуктивная составляющая, в принципе гарантирующая истинность результата. Именно на эвристические соображения делается акцент в методологии Пойа [6], но, как уже отмечено, важно трактовать их в аспекте алгоритмической и программной реализации. Именно взаимодействие механизмов эвристического выбора и строгого выполнения обеспечивает связь дедукции и правдоподобных рассуждений в стиле Пойа [14].

Наиболее интересны предметно-независимые эвристики типа статистических данных о применяемых методах, отбрасывания частей условия и анализа на пересечение множеств, сведения задачи к алгебраической формулировке и т.д. В текущей версии системы разработана логика использования (частично реализованная в макете) около 10 таких эвристик. Эвристики записываются в семантической сети и извлекаются в соответствии с индексацией, использующей текущую онтологическую структуру (типы как заданных, так и требующих нахождения объектов и отношений, статистику операций, глубину поиска и т.п.). Правила записываются в таблицы, фрагмент правила, программно поддерживающего аксиому, приведен на рисунке 4. Левая часть правила (l-part) ссылается на две различные точки A и B, а правая (r-part) утверждает, что существует прямая, которой принадлежат обе точки. Дополнительные поля, не отраженные на рисунке 4, определяют, в частности, единственность такой прямой.

В таблице используется отношение «различны», обладающее большой общностью, обратное отношение – «совпадают». Эти отношения часто применялись при решении задач методом доведения до абсурда: вывод, что два объекта обладают и тем, и другим отношениями, дает противоречие. Подчеркнем, что под объектами для этих отношений могут пониматься не только точки, но практически произвольные сущности. Такие отношения улучшают естественность и компактность описаний, а также позволяют по возможности избегать отрицаний, то есть (not) отношений. Для точки и прямой использованы аналогичные отношения: «на» и «вне» (вместо «принадлежит» и «не принадлежит».

В аналогичном виде представлены необходимые для решения задач на построение базовые операции: «*создать точку*» (*отрезок, прямую, окружность*), «*найти середину отрезка*», «*провести перпендикуляр из точки на отрезке прямой*», «*опустить перпендикуляр из точки на прямую*» и т.п. Дополнительно были добавлены правила работы с алгебраическими выражениями, что позволило решать задачи на построение с привлечением алгебры. Пример такой задачи: «построить прямоугольный треугольник по гипотенузе и биссектрисе прямого угла», детали решения которой приведены в [15].

Визуализация решения

Стиль визуализации решения, базирующейся на макросах MS Word, отражен ранее на рисунке 3. В текущей версии визуализация существенно более развита и реализована на JavaScript с использованием JSXGraph и MathJax. Программа взаимодействия с онтологией формирует текстовый файл с протоколом решения аналогично файлу, интерпретируемому макросами. Однако инструментальные средства уровня JavaScript с указанными библиотеками обеспечивают гораздо больше возможностей активного взаимодействия с чертежом. Фрагмент скриншота для чертежа, сформированного по ЕЯ-описанию задачи, приведен на рисунке 5.

Лингвистическая обработка позволила построить чертеж, полностью соответствующий условию задачи. В универсальной схеме решения Пойа это соответствует ответу на первый вопрос: *как* ***от формулировки*** *перейти* ***к формализации***. Далее для этой задачи система не дала полного решения, но предложила дополнительное построение и использование теоремы Чевы. Тем не менее, эта задача приводится, чтобы подчеркнуть важность автоматического перехода от ЕЯ-описания к онтологической структуре и далее к чертежу.

Приведенное на рисунке 5 вверху слева меню обеспечивает сервис, позволяющий перемещаться вперед и назад по шагам решения, видеть обоснование шагов (применяемых теорем или построений), включать подсветку ключевых объектов чертежа и т.п. Дополнительно пользователь может с помощью мыши модифицировать чертеж с сохранением условий задачи (пере-

Код	Область	№ акс	Тип объекта-1	Имя объекта-1	Отношение	Тип объекта-2	Имя объекта-2	Левая/правая
1	Геометрия	1	точка	T-A	различны	точка	T-B	l-part
2	Геометрия	1	точка	T-A	на	прямая	Pr-1	r-part
3	Геометрия	1	точка	T-B	на	прямая	Pr-1	r-part

Рис. 4. Фрагмент правила, программно поддерживающего аксиому

Fig. 4. A fragment of the rule that supports the axiom

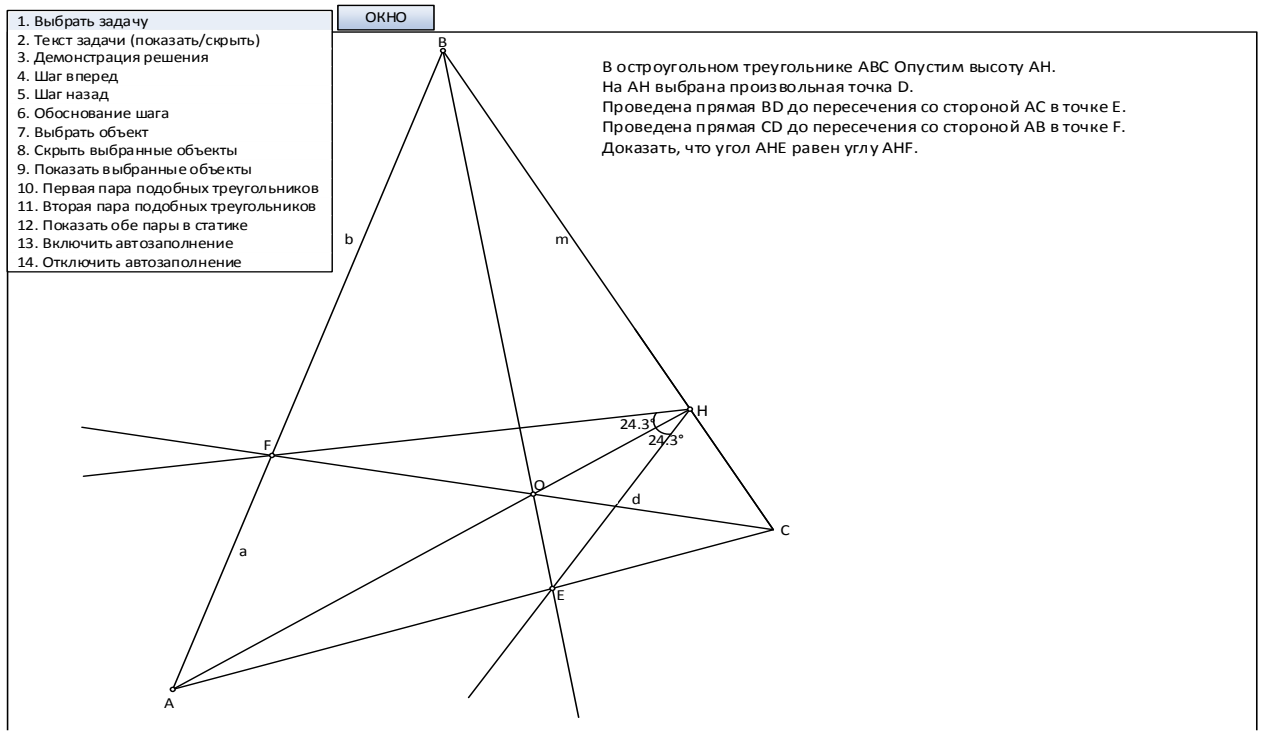


Рис. 5. Фрагмент чертежа, сформированного по естественно-языковому описанию задачи

Fig. 5. A fragment of the drawing, formed by the natural language task description

мещать выбранную на высоте точку, изменять размеры треугольника и т.п.). На логическом уровне разработаны возможности получения от онтологии информации о выбранных с помощью мыши объектах.

Статический чертеж на рисунке, к сожалению, не отражает динамику визуализации. Видеоролики, демонстрирующие пошаговую работу с решением, а также возможности модификации чертежа, приведены на HTML-странице [16].

Подчеркнем, что разработанный механизм визуализации из концептуальной структуры не ограничивается только областью геометрии. Наличие в онтологии соответствующих правил и информации об интерпретаторе графики позволяет формировать текстовый файл не только для визуализации чертежа, но и для графического вывода формул (MathJax), естественно-языковых описаний, 3D-графики и т.д. Примеры такой визуализации приведены на HTML-странице [16].

Обсуждение результатов

Эксперименты в основном подтвердили перспективность концепции и работоспособность макета. В процессе экспериментов был решен или частично решен (решение намечено, но не завершено) ряд задач. Интересно, что для достаточно нетривиальных задач система предлагала (или намечала) решения, отличающиеся от приведенных в источниках, примеры в [5, 17]. При этом, по мнению авторов, решения отличались большей наглядностью и естественностью, что вполне согласуется с методологией Пойа. Согласно Пойа, кристальная ясность доказательства достигается не только безупречностью каждого логического шага, но и обоснованием шагов эмпирическими соображениями, аналогиями с уже решенными задачами, визуализацией и т.д.

При описании визуализации отмечалось, что пользователь может активно взаимодействовать с чертежом, перемещая мышью объекты, подсвечивая их и т.д., что демонстрируется на видеороликах. Такое перемещение и подсветка во многих случаях подсказывают направление решения, обеспечивая пользователя наводящими (эмпирическими) данными. Разумеется, ни чертеж (по сути модель), ни манипуляции с ним не являются доказательством, но это служит хорошей базой для правдоподобных догадок в стиле методологии Пойа. Необходимость онтологии для анализа решения отмечена для задачи на рисунке 3.

Заключение

Основной вывод из проведенного исследования состоит в том, что алгоритмическое и компьютерное воплощение методологии Пойа открывает перспективы создания систем искусственного интеллекта, интегрирующих достижения в обработке естественного языка, автоматического решения задач и концептуальной визуализации. В фундаментальном аспекте системы

такого класса позволят по-новому взглянуть на когнитивные механизмы и модели, используемые человеком при решении задач.

В прикладном аспекте данное исследование целесообразно ориентировать на создание обучающих систем качественно нового уровня (естественно-языковой интерфейс, решатель на базе онтологии и концептуальная графика), например, по сравнению с [18]. Дальнейшие исследования предполагают модернизацию и доработку программ макета, тестирование системы на представительном множестве задач, а также расширение текущей онтологии на другие аксиоматики.

Интересным направлением было бы использование системы для компьютерной реализации идей работы [19]. Темпоральная аксиоматика и динамичная визуализация, воплощенные в интегральной системе, могли бы конкретнее представить прикладную значимость вышеупомянутой работы. Важно также проверить предположение о более простой реализации предложенного в [19] формализма и его лучшие характеристики вычислительной сложности по сравнению с известными системами активной логики.

Авторы выражают благодарность Кулакиной Н.С. и Стринже А. за помощь при тестировании системы и подготовке материалов для регистрации программ.

Работа выполнена при финансовой поддержке РФФИ, проекты №№ 18-07-00098, 18-29-03088, 18-07-00213, 18-51-00007, 19-07-00213.

Литература

1. Хорошевский В.Ф. Генерация лингвистических процессоров для платформы GATE под управлением онтологий // КИИ-2018: сб. тр. конф. М., 2018. Т. 1. С. 288–295.

2. Adid Deshpande (Deep learning research review week 3: natural language processing). URL: https://adeshpande3.github.io/adeshpande3.github.io/Deep-Learning-Research-Review-Week-3-Natural-Language-Processing (дата обращения: 07.11.2018).

3. Подколзин А.С. Исследование логических процессов путем компьютерного моделирования // Интеллектуальные системы. Теория и приложения. 2016. Т. 20. С. 164–168.

4. Подколзин А.С. Компьютерное моделирование логических процессов. Архитектура и языки решателя задач. М.: Физматлит, 2008. 1024 с.

5. Курбатов С.С., Фоминых И.Б., Воробьев А.Б. Пойа-метод: компьютерное воплощение методологии Д. Пойа // КИИ-2018: сб. тр. конф. М., 2018. Т. 2. С. 96–104.

6. Polya G. Mathematical discovery: on understanding, learning and teaching problem solving. Wiley, 1981, 432 p.

7. Литвинович А.В. Язык описания графических объектов GRASP // Нейрокомпьютеры: разработка, применение. 2012. № 10. С. 26–30.

8. Progress (СУБД). URL: https://www.progress-tech.ru/ (дата обращения: 07.11.2018).

9. JSXGraph. Dynamic Mathematics with JavaScript, JSXGraph is a cross-browser JavaScript library for interactive geometry, function plotting, charting, and data visualization in the web browser. URL: http://jsxgraph.uni-bayreuth.de/wp/index.html (дата обращения: 07.11.2018).

10. MathJax. URL: https://radioprog.ru/post/74 (дата обращения: 07.11.2018).

11. Апресян Ю.Д., Богуславский И.М., Иомдин Л.Л. Лингвистическое обеспечение системы ЭТАП-2. М.: Наука, 1989. 295 с.

12. Шелманов А.О. Исследование методов автоматического анализа текстов и разработка интегрированной системы семантико-синтаксического анализа: дис. … канд. тех. наук. М.: ИСА РАН, 2015. 210 с.

13. Курбатов С.С. URL: http://www.eia--dostup.ru/onto_geom.htm (дата обращения: 07.11.2018).

14. Polya G. Mathematics and Plausible Reasoning. Princeton Univ. Press, 1954, 2 volumes (vol. 1: Induction and Analogy in Mathematics, vol. 2: Patterns of Plausible Inference).

15. Воробьев А.Б. URL: http://www.eia--dostup.ru/фрагмент%20магистерской.mht (дата обращения: 07.11.2018).

16. Курбатов С.С. URL: http://www.eia--dostup.ru/Динамика чертежа.htm (дата обращения: 07.11.2018).

17. Тригг Ч. Задачи с изюминкой. М.: Мир, 1975. 302 с.

18. Сергеева Т.Ф., Шабанова М.В., Гроздев С.И. Основы динамической геометрии. М.: Изд-во АСОУ, 2016. 152 с.

19. Фоминых И.Б., Виньков М.М., Пожидаев А.К. Активная логика и логическое программирование: объединение двух концепций // Программные продукты и системы. 2015. № 3. С. 42–48.

Software & Systems
DOI: 10.15827/0236-235X.125.012-019

Received 15.11.18
2019, vol. 32, no. 1, pp. 012–019

Algorithmic and software implementation of a cognitive agent based on G. Polya's methodology

S.S. Kurbatov [1], *Leading Researcher, curbatow.serg@yandex.ru*
I.B. Fominykh [2], *Dr.Sc. (Engineering), Professor, igborfomin@mail.ru*
A.B. Vorobev [2], *Postgraduate Student, abvorobyev@bk.ru*

[1] *Scientific Research Centre for Electronic Computer Technology, Moscow, 117218, Russian Federation*
[2] *National Research University "Moscow Power Engineering Institute", Moscow, 111250, Russian Federation*

Abstract. The paper describes an original approach to creating an integrated problem solving system (cognitive agent). The system involves a tight integration of linguistic processing stages, an ontological representation, a heuristically oriented solution and visualization. The system concept is based on the Polya's methodology interpreted in algorithmic and software implementation. The system is implemented in a mock-up version and tested in the subject area "school geometry".

The linguistic component uses the problem canonical description obtaining method through paraphrasing and mapping it into a semantic structure.

An automated solution search is based on implementing the rules that reflect the axioms of the respective subject areas. The heuristics presented in the ontology define the rules. The heuristics are designed as semantic network structures, which allows organizing a rule multiple-aspect search and selection justification as a natural language comment.

Conceptual (cognitive) visualization provides the solution visual representation by interpreting a text file with information to display graphical objects, as well as comments on the solution process. Comments include natural language descriptions of rules (axioms, theorems), heuristic and empirical justifications for their choice and links to visualized objects.

The paper defines experiments that demonstrate visualization possibilities of task drawings and ontology fragments, natural language phrases, mathematical and formal logic formulas. The ontology is implemented in the DBMS Progress. Visualization programs are implemented in JavaScript using JSXGraph and MathJax. The implementation provides a step-by-step solution view in different directions with dynamic changing in drawing and related comments.

The authors have interpreted experimental results and planned the study to develop the described approach.

Keywords: integrated system, cognitive agent, natural language user interface, domain ontology, solution visualization, school geometry.

Acknowledgements. *The authors are grateful to N. Kulakina and A. Stringa for assistance in testing the system and preparing materials to register programs. The work has been supported by the Russian Foundation for Basic Research, projects no. 18-07-00098, 18-29-03088, 18-07-00213, 18-51-00007, 19-07-00213.*

References

1. Khoroshevsky V.F. Generation of linguistic processors for the GATE platform under the ontology control. *Proc. Conf. on Artificial Intelligence (KII-2018)*. Moscow, vol. 1, pp. 288–295 (in Russ.).
2. Deshpande A. *Deep Learning Research Review Week 3: Natural Language Processing*. Available at: https://adeshpande3.github.io/adeshpande3.github.io/Deep-Learning-Research-Review-Week-3-Natural-Language-Processing (accessed November 7, 2018).
3. Podkolzin A.S. The study of logical processes by computer simulation. *Intelligent Systems. Theory and Applications*. 2016, vol. 20, Moscow, pp. 164–168 (in Russ.).
4. Podkolzin A.S. *Computer Simulation of Logical Processes. Architecture and Problem Solver Languages*. Moscow, Fizmatlit Publ., 2008, 1024 p.
5. Kurbatov S.S., Fominykh I.B., Vorobev A.B. Polya's method: computer implementation of Polya's methodology. *Proc. Conf. on Artificial Intelligence (KII-2018)*. Moscow, vol. 2, pp. 96–104.
6. Polya G. *Mathematical Discovery: On Understanding, Learning and Teaching Problem Solving*. Wiley Publ., 1981.
7. Litvinovich A.V. GRASP graphical object description language. *J. Neurocomputers*. 2012, no. 10, pp. 26–30 (in Russ.).
8. *Progress (DBMS)*. Available at: https://www.progress-tech.ru/ (accessed November 7, 2018).
9. *JSXGraph (Dynamic Mathematics with JavaScript, JSXGraph is a Cross-Browser JavaScript Library for Interactive Geometry, Function Plotting, Charting, and Data Visualization in the Web Browser)*. Available at: http://jsxgraph.uni-bayreuth.de/wp/index.html (accessed November 7, 2018).
10. *MathJax*. Available at: https://radioprog.ru/post/74 (accessed November 7, 2018).
11. Apresyan Yu.D., Boguslavsky I.M., Iomdin L.L. *ETAP-2 System Linguistic Support*. Moscow, Nauka Publ., 1989, 295 p.
12. Shelmanov A.O. *Research on Automatic Text Analysis Methods and the Development of an Integrated System of Semantic-Syntactic Analysis*. PhD Thesis. Moscow, ISA RAS Publ., 2015, 210 p.
13. Kurbatov S.S. Available at: http://www.eia--dostup.ru/onto_geom.htm (accessed November 7, 2018).
14. Polya G. *Mathematics and Plausible Reasoning*. Princeton Univ. Press, 1954.
15. Vorobev A.B. Available at: http://www.eia--dostup.ru/фрагмент%20магистерской.mht (accessed November 7, 2018).
16. Kurbatov S.S. Available at: http://www.eia--dostup.ru/Динамика чертежа.htm (accessed November 7, 2018).
17. Trigg Ch.W. *Mathematical Quickies*. McGraw-Hill, 1967, 270 p. (Russ. ed.: Moscow, Mir Publ., 1975).
18. Sergeeva T.F., Shabanova M.V., Grozdev S.I. *Basics of Dynamic Geometry*. Monograph. Moscow, ASOU Publ., 2016, 152 p.
19. Fominykh I.B., Vinkov M.M., Pozhidaev A.K. Active logic and logic programming: integration of the concepts. *Software & Systems*. 2015, no. 3, pp. 42–48.