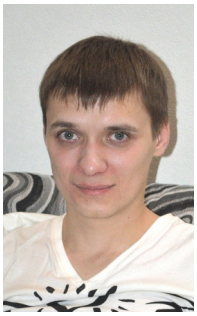
УДК 004.588:004.8



В.Ю. Высоцкий,
аспирант,
Одесский национальный политехнический университет
visot@te.net.ua

В.Д. Гогунский,
д.т.н., профессор,
Одесский национальный политехнический университет
e-mail: vgog@i.ua

ПОИСКОВЫЕ АЛГОРИТМЫ ДЛЯ АВТОМАТИЗИРОВАННОГО ОБУЧЕНИЯ

В.Ю. Высоцкий, В.Д. Гогунський. Поисковые алг-ритмы для автоматизированного обучения. В статье описывается подход к поиску пути в интеллектуальных графах с вершинами, являющимися интеллектуальными агентами. Возможное применение этого подхода основано на логическом выводе в распределенной фреймовой иерархии..

V. Vysotsky, V.D. Gogunsky. Pathfinding algorithms for computer aided learning. This paper describes an approach to path-finding in the intelligent graphs, with vertices being intelligent agents. A possible implementation of this approach is described, based on logical inference in distributed frame hierarchy.

Введение. Одна из больших проблем в информационных технологиях в настоящее время - это растущее количество информации, накопленное в больших и плохо структурированных хранилищах, распределенных базах знаний. Много исследований ведется в области информационного обзора и поиска, в большинстве решений присутствуют некоторые элементы искусственного интеллекта.

Мы рассмотрим подход, основанный на формировании динамических описаний в форме местных (локальных) баз знаний, которые содержат знания в виде продукционных правил, и может обеспечить ссылки на рекомендуемые ресурсы, включая другие базы знаний для консультаций, удовлетворяющие требования пользователя. Алгоритмы для логической связи при выводе (производстве) знаний хорошо известны и инструменты уже существуют, и могут использоваться для логического вывода в распределенном окружении [1, 3].

Главное неудобство этого подхода мы видим в трудности отображения в базе знаний быстро изменяющихся фрейм-ресурса в определенной проблемной области. Чтобы преодолеть эту трудность, интуитивно предлагается использовать предварительно разработанные шаблоны, уже содержащие универсальные динамические знания во многих подобных этой проблеме областях, и таким образом, уменьшают задачу описания индивидуального ресурса, свести к наследованию большей части знаний из имеющегося ресурса.

Необходимые выводы могут быть получены в процессе онлайн просмотра коллекции распределенных баз знаний, это форма, на структуру которой мы будем ссылаться как на интеллектуальный граф. Каждая вершина консультационной базы знаний связана с некоторым возможным вопросом пользователя. В результате этого рассматриваемая проблема пополняется новыми фактами, сведениями, ресурсная цепочка меняется, то есть добавляются новые ресурсы к просмотру, который может быть интересным для пользователя, или в дальнейшем использоваться базой знаний для консультаций.

Далее, мы можем наложить некоторые ограничения на последовательность обработки запроса, на посещение некоторого ресурса ранее другого подобного. Это позволило бы нам проложить путь в коллекции ресурсов, который удовлетворит большинству потребностям пользователей. Например, таким способом мы можем получить учебный материал из набора связанных ресурсов, который окажется чрезвычайно полезным в такой области как дистанционное обучение.

В этой статье используется понятие интеллектуального графа и описывается специальный эвристический алгоритм, который может использоваться как инструмент для формирования многоуровневых обучающих материалов и тестов. Архитектура интеллектуального графа основана на распределенной фрейм иерархии, которая может быть использована для реализации представленных идей интеллектуального формирования учебной информации, тестирования, поиска. В конце обсуждаются потенциальные применения представленного подхода.

Выбор пути в интеллектуальных графах. Понятие графа широко используется в программировании как средство абстракции при описании таких объектов, как программа, структура данных, а также при описании алгоритмов распознавания различных свойств программы, при разработке инструментов автоматического программирования, при генерации программных кодов и гипертекстовых фреймов. В теории программирования схема программы представляется в виде размеченного мультиграфа. Понятие разметки формализует отдельные свойства графа или его частей. Иллюстративным примером понятия разметки является задача достижимости вершин ориентированного графа от некоторой выделенной его вершины, называемой начальной [6].

Для графического представления состава и структуры распределенных баз знаний, плохо структурированных информационных хранилищ типа баз знаний удобно использовать такое понятие, как интеллектуальные графы [5].

Интеллектуальные графы. Во многих практических ситуациях удобно иметь дело с графами, в которых дуги не статически определены, а скорее зависят от некоторых параметров [4]. Параметрами могут быть случайные числа, статистика работы пользователя, результаты контроля знаний обучаемого, индивидуальные запросы пользователя и т.п. Например, интеллектуальный каталог базы знаний может представить только подмножество

доступных гиперссылок пользователю, зависящий от его предпочтений, так создается граф динамических гиперсвязей (динамический гиперсвязанный граф). Далее, пользовательское предпочтение может быть изменено во время просмотра гипертекстового графа. Набор дуг, представленных пользователю, будет динамически изменен для отдельных вершин в процессе просмотра. Мы отобразим эту идею в следующем определении.

Определение 1. Ориентированный динамический мультиграф мы обозначим $\Gamma = (X, E, \Phi)$,

где $\{X\}$ – множество вершин, $x_i \in X = \{X_i\}$, $i = 0 \dots n$,

$\{E\}$ – множество дуг, $e_j \in E = \{E_j\}$, $j = 1 \dots m$,

$\{\Phi\}$ – набор активирующих дуги функций (правил), зависящих от параметров, $\phi_r \in \Phi = \{\Phi_r\}$, $r = 1 \dots k$, $\phi_r : S \to \{true - 1, false - 0, P\}$ – каждая функция определена начальным состоянием S графа (из набора true - 1, false - 0) или динамически корректируется параметром P. Для каждого значения ϕ_r граф $\Gamma(\Phi)$становится нормально ориентированным графом $\Gamma = (\{X_i\} ,V)$,

где V – программные агенты, состоящие из дуг и вершин,

$$V = \{(u_j, v_j) | \phi_r(S) = true \cup \rho_r\}.$$

Количество дуг и функций у каждого агента конечно.

Описание свойств и особенностей функционирования программных агентов V ограничим конечными графами. Дуги и функции будем снабжать метками, которые формализуют интересующие нас свойства графа или его частей. Подобную процедуру сопоставления меток вершинам и дугам назовем *разметкой*. В качестве размеченного анализируемого графа будут выступать схемы программ. Метод разметки будет использовать тот факт, что свойства вершины графа определяются свойствами некоторых “соседей” этой вершины.

Путем ω в графе Γ называется всякая последовательность $\dots x_i, e_i, x_{i+1} \dots$ такая, что для всех i $\Phi(e_i) = (x_i, x_{i+1})$ или $\forall i \Phi(e_i) = (x_i, x_{i+1})$. Путь по вершинам содержит только вхождения вершин, т.е. ω $(x_0, \dots , x_i, x_{i+1}, \dots, x_n)$. Путь по дугам содержит только вхождения дуг, т.е. ω $(e_0, \dots , e_i, e_{i+1}, \dots, e_n)$. Здесь $0 \le i \le n$, тогда e_0 – начало пути, e_n - конец пути, n – длина пути. Если граф размечен, то *образом пути* будет набор слов, составленный из меток проходимых дуг или вершин.

Размеченный граф назовем *строго возрастающим*, если всякая последователь-ность вершин $x_0, x_1, \dots, x_n$ из X такая, что $\forall i \, x_{i+1} > x_i$. Граф L, в котором строго возрастающие цепи конечны, назовем *ограниченным* (ограниченная полурешетка L). В таком графе $\forall x \in L$ существует такое число b, что длины всех строго возрастающих цепей ограничены числом b.

Мы будем рассматривать размеченные динамические графы с точки зрения формирования пути ω, то есть поиск определенной конечной последо-

вательности вершин $x_0..., x_n$, начинающийся с определенной точки x_0, где конечная вершина x_n, удовлетворяет некоторому условию $C(x_n)$. Практически мы должны были бы также наложить некоторые ограничения на формируемый (необходимый) путь. Ограничения представим в виде конечной последовательности $\sqsubset s$ отношений, заданных на множестве X. Отношения задаются на x_i, и зависят от разметки (статуса, состояния, значения) набора отношений S . Мы рассмотрим изменение разметки (состояния, статуса) во время формирования пути, то есть во время просмотра графа. Для этого к отдельным вершинам графа применим корректирующие функции Φ. Вводим следующее

Определение 2. Ориентированный граф с ограничениями

$\Gamma = (\{X, \Phi\}, <u_j, v_j, \phi_j>, \sqsubset s)$,

где $\Phi : S \rightarrow S'$ — корректирующая функция, $\sqsubset s$ — ограничения на порядок отношений, зависит от текущего состояния. Мы также положим, что S - частично определенный набор (полурешетка - чередующийся набор значений), и что отношения ϕ_j и $\sqsubset s$ – *монотонны*, то есть для $\forall s_1 \geq s_2 \in S$ значит, что $\forall j \ \phi_j(s_1) \geq \phi_j(s_2)$ и следует, что для $\forall u, v \in X$ следует $(u \sqsubset s_1 v) \geq (u \sqsubset s_2 v)$.

Корректирующие функции ϕ_i - монотонны в традиционном смысле, то есть для $\forall i \forall s_i \in S$ следует – ограничение s_i определяет результат $\phi_i(s)$.

Перемещение (Выполняя поиск) в таком графе будет зависеть и от стартового шага $s_0 \in S$. Так, каждый путь $x_0 \rightarrow ... \rightarrow x_n$ также соответствует пути $s_0 \sqsubseteq s_1 \sqsubseteq .. \sqsubseteq s_n$ в наборе ограничений. В соответствии с этим мы сделаем следующее

Определение 3. Путь $(x_0, s_0) \rightarrow ... \rightarrow (x_n, s_n)$ назовем действующим (валидным), если $\forall i \ (x_i, x_{i+1}, \phi_j) \in \omega$ и $\phi(s_i) = true$ (то есть все дуги в пути ω действующие), и $x_i \sqsubseteq s_j \ x_j \supset (i \leq j)$ (то есть все последующие ограничения, включая конечные, удовлетворены). Для действующего пути можем также потребовать непрерывности ограничений, то есть $s_{i+1} \sqsubseteq \phi(s_i)$, и назовем его монотонно действующим путем.

На практике, выполняя эвристические алгоритмы, действенность (валидность) отношений трудно достигнуть (согласовать), пока вершины в действующем пути не будут удовлетворять всем ограничениям, даже тем, которые могут появиться на более поздних стадиях просмотра графа(состояниях). Мы можем ввести специальный вид выборочных условных связей, где $\sqsubset s_{i+1} \Rightarrow \sqsubset s_i \cup \sqsubset i$ и $\sqsubset i \subseteq \{(u, x_i), u \in X\}$ (то есть условия и ограничения, введенные при просмотре вершин (vertice), в последующих состояниях (стадиях)

ограничения на просмотренные вершины, не будут накладываться). В этом случае валидное отношение может быть написано в более слабой форме $x_i \sqsubseteq s_j\, x_j \sqsupset\ (i \le j)$.

Практическое применение представленных определений может быть применено как в мультиагентных системах, ориентированных на информационный поиск так и в гипертекстовом навигационном планировании. Рассматривается связанное мультиагентное общество, где каждый агент имеет смысл в некоторых общих терминах (словах), и знает об определенном наборе других агентов, которых он может рекомендовать для дальнейшей консультации. В этом случае агенты формируют динамический граф, и находят путь, удовлетворяющий определенным ограничениям (условиям, требованиям), связанным с выполнением распределенной консультацией, обучением, тестированием. Состояние S в этом случае соответствует внешним требованиям всех агентов в библиотеке, обществе, и это состояние подчинялось бы всем монотонным (непрерывным) взаимосвязям, если стремиться к удовлетворению запросов в этой мультиагентной системе.

Поисковые алгоритмы. Для исследования ориентированных графов, были использованы два (алгоритма) приложения. В первый, к большинству классических поисковых алгоритмов были добавлены ограничения (restrictions), чтобы удовлетворить корректирующие требования.

Алгоритм 1: основанный на предварительной последовательности поисковый алгоритм для выделения ограничений из динамического графа.

Input: Избирательный ориентированный динамический граф (**Separable constrained dynamic graph**) $\mathbf{Q = [\{X_i,\Phi_i\},U,\sqsubset s]}$, начальное поисковое состояние (**initial search state**) $\mathbf{S_0}$, начальная вершина (**initial point**) $\mathbf{X_0}$, критерий поиска (**search criteria**) **C.**

Output: набор всех действительных путей (**A set of all valid paths**) $\mathbf{R = \{[(X_0,S_0),\ldots,(X_n,S_n)]\}}$

```
QueueSearch(Q,X0,S0)        ПросмотрОчереди –Заголовок ПП
R ← ∅                       ‘очищаем путь для результата
Q ← {(X0,S0)}               ‘ удаляем начальную вершину из множ.акт.дуг графа
while Q ≠ ∅ do              ‘пока множество активных дуг не пусто повторять
   select Φ ∈ Q (first, last, best, etc.)
   Q ← Q \ {Φ}  ‘ удалить результат корректир.функции из множества активных дуг графа
   (Xl,Sl) ← last(Φ)  ‘заменить пометку выбранной вершины
   if Xl satisfies C then
      R ← R ∪ {Φ} ‘объединяем вершины, удовлетвор.критерию, в действующий путь
      else
```

S ← {(x, $T_x(S_l)$))|(X_l, x, ф) ∈ U, ф(s)= true} ‘рабочая память, в которой вершины и дуги с результатом корректирующих функций = TRUE

End if

continue ‘ конец цикла

For each (x, s)∈ S ‘ цикл для каждой записи, входящей в набор требований

if [Ф| (x, s)] satisfies ⊏ s (or ⊏ x) then Q ← Q ∪ {[Ф | (x, s)]} ‘восстанавливаем активный путь с учетом требований ⊏ s, наложенных на вершину, или с учетом свойств вершины ⊏ x

Альтернативно, ограничения могут использоваться не только для фильтрации недействительных путей во время поиска, но скорее для выбора направления в ходе поиска пути. Например, если, во время поиска, мы перемещаемся от X_k до определенной вершины X_i, а условия указывают, что X_{j1} и X_{j2} нужно посетить до X_i, тогда мы двигаемся к X_{j1} , помещая X_{j2} и X_i в очередь. В этом примере допустим, путь X_k к X_{j1} может не существовать — алгоритм возвращает нас назад. На практике, ограничения указывают, что соответствующая вершина "знает" о предыдущей, и таким образом, путь между ними существует, то есть

$$x_i \sqsubseteq s\ x_j \supset (x_i, x_j, \phi_j) \in U \wedge \phi_j(s) = true \qquad (1)$$

Эти соображения приводят нас в семейство из эвристических поисковых алгоритмов [3], где мы, главным образом, удовлетворяем ограничения и наконец выполняем условия, формирующие валидный путь. Эти алгоритмы дают действительный (валидный) путь, заданный условиями (1) на корректирующих функциях ϕ_j и ограничениях $\sqsubseteq s$. Пример такого алгоритма приведен в Алгоритм 2.

Алгоритм 2. эвристический поисковый алгоритм

Input: Избирательный ориентированный динамический граф (**Separable constrained dynamic graph**) **G = ({X_i,T_i},U,⊏ s), initial search state S_0, initial point X_0, search criteria C.**

Output: Действующий путь **P = [(X_0,S_0),…, (X_n,S_n)]**

HeurSearch (G,X_0,S_0) ‘ ЭвристическПоиск

P ← [(X_0,S_0)] ‘ помещаем начальную вершину в результирующий путь

S ← ∅ ‘очищаем рабочую память

R ← ∅

while last(P) does not satisfy **C do**

(x_l, s_l)← **last(P)** ‘ заменить пометку выбранной вершины

if Q ≠ ∅ then ‘ если множество активных дуг не пусто

select x ∈ Q

Q ← Q \ {x} ‘ удаляем выбранную вершину из множ.акт.дуг графа

else if {ξ|(x_l,ξ,ф)∈ U, ф(s_l)= true} ≠ ∅

S ← {ξ|(x_l,ξ, ф)∈ U, ф (s_l)= true} ‘рабочая память, в которой вершины и дуги с результатом корректир. функций = TRUE

select x ∈ S

R ← R ∪ (S \ {x})

else if R ≠ ∅

select x ∈ R

R ← R \ {x}

else

break ‘Выход из ПП или перейти к след.активн. вершине

s ← $T_x(s_l)$ ‘ выделяем соответствующее ограничение для вершины x

Ω ← {ξ|ξ ⊏ s x} ‘ выполняется функция преобразования свойств в соотв.с огранич.

if Ω= ∅ then

P ← [P|(x, s)] ‘ помещаем текущую вершину в результирующий путь, если маршрут состоит из единственной дуги

else

Q ← Q ∪ Ω ∪ {x} ‘ формируем активный путь (стационарную разметку) с учетом всех ограничений

continue

Принципы интеллектуального поиска в динамическом описании ресурсов. Формализм, представленный выше, может быть использован для описания следующей ситуации. Предположим, что существует несколько источников (ресурса), и мы хотим предоставить автоматическим способом путь для пользователя, чтобы выбрать указанный ресурс или путь внутри ресурса, основанный на каких-то предварительных условиях. Это может быть выполнено в виде инвариантных динамических учебных материалов из базы знаний. Алгоритм должен определить, является ли ресурс полезным для требований пользователя, а также рекомендовать, какие ресурсы следует посетить как для знакомства, так и в качестве основного источника информации.

В этом случае, ресурсы следует оформить в виде ограниченных ориентированных графов, где ориентация сформирована в общей рабочей памяти (или в буфере, доске), используемой всеми самостоятельными системами. Требования пользователя будут определены в начальном запросе (в статистике со-

стояния) S_0, и исходной точкой поиска X_0 в известном ресурсе. В процессе поиска исходный граф будет преобразован (модифицирован), и консультационная база знаний оформляется отдельным узлом. Результирующее улучшенное состояние исходного графа будет пополнено новейшими фактами (что соответствует применению монотонной функции Φ_x).

В ходе работы с учебным материалом может потребоваться уточнить некоторые дополнительные вопросы у пользователя для принятия более точного решения. В этом случае, ответы пользователя также хранятся в статистике состояния.

Ограничения будут использованы для уточнения того, что ресурсы просмотрены в ”правильном” порядке, т.е. определенные темы проработаны перед новым материалом. В результате, удовлетворены как требования пользователя так и логика работы баз знаний, все это отражено в виде индивидуального пути в ресурсном графе.

В целях обеспечения уверенной совместимости между базами знаний, распространяющими различные данные, запросы должны быть сформулированы и организованы в некоторых общих терминах, которые определены в тематике требования. На практике, используются запросы двух типов [2]:

Ориентированные на архитектуру каталогов, которые отражают основные (базовые) структуры в описании ресурсов (предисловия, аннотации, обзоры и т.д.), а также шаблоны для различных типов запросов, таких как ссылки на источники, на оформления и т.д.

Ориентированные на фреймовую структуру предметных знаний, что предоставляет некоторые домен-специфические знания (например, развлекательные материалы, тематические шаблоны, типичные пользовательские предпочтения, статистическая информация, например по различным возрастам, и т.д.).

Для реализации инвариантности представления учебных данных, удобно использование такой архитектуры, где объявлены статические и динамические домен-знания, и использовать эти домен-знания непосредственно в процессе обоснований каждого вывода. Архитектура распределенной фреймовой структуры, описанной в [1] хорошо подходит для таких целей, как единое представление знаний в системе, которая может непосредственно использоваться в процессе распределенного логического вывода для серии баз знаний, расположенный в разных узлах сети.

Приложение в системе дистанционного обучения. В современных системах дистанционного обучения, одна из проблем - автоматическое планирование курса, основанное на пользовательских требованиях [4]. Такая система, после консультации с пользователем или предварительным тестом будет выбирать набор материалов, который должен быть изучен пользователем, и представить его в виде плана, удовлетворяя потребности оптимальным образом. Эта проблема является, по существу, такой же, как поиск оптимального пути в ориентированном графе.

Для реализации такой системы в упомянутой фрейм-технологии должен быть разработан инвариантный учебный план, а все индивидуальные учебные темы должны быть описаны с помощью динамических характеристик с ограничениями. Учебный план может быть построен в процессе распределенного доступа к набору ресурсов. Начальное состояние, требуемое для этого построения, может быть получено путем опроса пользователя, возможно, подключая адаптивную базу знаний, управляющую тестированием [5].

Выводы. В статье представлена архитектура системы мультиагентного поиска на основе фреймовых описаний. В отличие от многих поисковых систем, это перспективная онлайн стратегия поиска, т.к. основана на пользовательских предпочтениях или на предварительном тестировании пользователя. Стратегия строится на наборе динамически выбранных ресурсах, созданных для удовлетворения нужд пользователя. Пока эта архитектура не очень хорошо подходит для традиционных обучающих поисков в широком масштабе, поскольку множественные стандарты ограничивают возможности аннотирования. Однако она может использоваться в различных проектах, связанных с поиском навигации в наборе систематизированных ресурсов. Примером применения такого подхода может быть система дистанционного обучения, и сайты с интеллектуальной навигации. Однако, с дальнейшим развитием стандарта фрейм-технологий и описанных методов, идея ресурсной систематизации динамических баз знаний может быть использована для более амбициозных задач.

Литература

1. Soshnikov D. An Architecture of Distributed Frame Hierarchy for Knowledge Sharing and Reuse in Computer Networks. In Proc. of 2002 IEEE Int. Conf. on Artificial Intelligence Systems, IEEE Computer Society Press, 2002. pp. 115-119.

2. Sizikov E., Soshnikov D. Using Dynamic Ontologies based on Production-Frame Knowledge Representation for Intelligent Web Retrieval. In Proc.ofthe 4th Int. Workshop on Computer Science and Information Technologies,Patras, Greece, 2002.

3. Davison A., Loke S.W. LogicWeb: Enhancing the Web with Logic Programming, Journal of Logic Programming, Vol. 36(3), September 1998, pp.195-240.

4. Panteleev M.G., Puzankov D.V. et al. Intelligent Educational Environments Based on the Semantic Web Technologies. In Proc. of the 2002 IEEE Int. Conf. on Artificial Intelligence Systems, IEEE Computer Society Press, 2002. pp. 457–462.

5. Malkina O.I., Soshnikov D.V. Creating Adaptive Testing Systems on the Internet using Artificial Intelligence Technologies. In Sel. Abstracts of 9th Int. Student Conf. on New Information Technologies, MGIEM Publising, 2001. pp.390-392.

6. Котов В.Е. и др. Теория схем программ. – М.: Наука, 1991