УДК 681.3

И.Н. ПАРАСЮК, Ф.В. КОСТУКЕВИЧ

ОБ ОДНОМ ЭФФЕКТИВНОМ АЛГОРИТМЕ РАСПРОСТРАНЕНИЯ ВЕРОЯТНОСТЕЙ В НЕЧЕТКИХ БАЙЕСОВСКИХ СЕТЯХ ДОВЕРИЯ

Рассмотрена общая схема алгоритма распространения вероятностей для байесовских сетей. Построен унифицированный алгоритм распространения доверия для нечетких байесовских сетей доверия, который базируется на новых принципах обхода узлового дерева, является более прозрачным и быстродействующим. Описана структура этого алгоритма и исследованы условия его корректной работы. Описаны результаты моделирования основных операций с размытыми потенциалами, заданными над нечеткими байесовскими сетями доверия.

Введение. Для эффективного вероятностного вывода на классических байесовских сетях доверия Перлом был разработан алгоритм распространения вероятностей (доверия), использующий трансформированную байесовскую сеть в виде узлового дерева, в котором для достижения согласованности по разделяемым переменным между вершинами использовался метод передачи сообщений [1]. По этой же схеме Шпигельхальтер и Лауритцен разработали подобные алгоритмы и реализовали их в широко известной системе HUGIN [2]. Алгоритм распространения доверия (АРД) в задачах анализа сетей доверия играет главную роль – он позволяет вычислить состояние всей исследуемой системы, имеющее наибольшее значение вероятности, т. е. найти оптимальное ее состояние. Другая интересная задача, решаемая с помощью этого алгоритма, – это вычисление маргинальной вероятности состояний для некоторого подмножества элементов исследуемой системы. Кроме того, АРД, использующий трансформированную байесовскую сеть, осуществляет как априорный (без свидетельств), так и апостериорный (со свидетельствами) вероятностный вывод.

Представляется, что развитие АРД применительно к нечетким байесовским сетям доверия будет способствовать более адекватному оцениванию как причинно-следственных связей систем в условиях неопределенности и неполноты данных, так и апостериорных оценок доверия для различных конфигураций свидетельств. Такой подход

продолжительное время используется, например, в системе [3], в основе которой – нечеткие деревоподобные байесовские сети доверия с недетерминированными состояниями.

Настоящая статья посвящена вопросам построения унифицированного, заметим, нерекурсивного АРД для нечетких сетей доверия. Кроме того, созданный алгоритм является с технологической точки зрения более конструктивным и прозрачным, имеет лучшие показатели временной сложности.

Определение нечеткой байесовской сети и узлового дерева. Случайные переменные или их множества, здесь и далее, обозначаются большими буквами, а их отдельные значения – малыми буквами: $X=x$ означает, что переменная X получает значение x, или, что вектор переменных $X=(X_1,\ldots, X_n)$ получает вектор значений $x = (x_1,\ldots, x_n)$. Область определения для X обозначается $\mathrm{dom}(X)$; $||X||=|\mathrm{dom}(x)|$ – количество возможных различных значений переменной X.

Если $X = X_1,\ldots, X_n)$, тогда $\mathrm{dom}(X)$ является декартовым произведением над областями определения переменных в X, т. е. $||X||=\prod_i || X_i ||$. Оно образует пространство состояний, а его элементы – так называемые конфигурации [2]).

Размытый (нечеткий) потенциал – это функция $\varphi:\mathrm{dom}(X)\to M$, где M – область значений этой функции, состоящая из нечетких нормализованных чисел [4]. Для обозначения области определения потенциала φ будем использовать нижний индекс (например, φ_X – это потенциал, определенный над $\mathrm{dom}(X)$) или записывать в виде $\mathrm{dom}(\varphi)$, а для обозначения области значений – Im (к примеру, $\mathrm{Im}(\varphi)$ –область значений потенциала φ) . Обозначив $\mathrm{Im}(\varphi)$ символом M, можем записать $M =\{ \mu(t) \mid t\in \mathrm{R}, \mu(t):\mathrm{R}\to[0;1]\}$, где R – множество действительных чисел, $\mu(t)$ – функции принадлежности определенного вида [3–5].

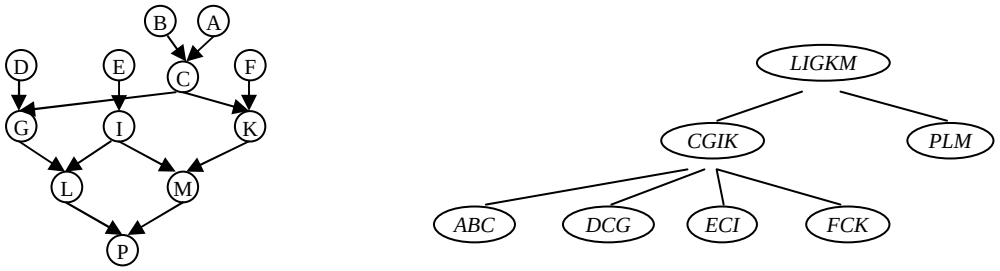
Для графического представления взаимосвязей между переменными, над которыми определены размытые потенциалы, используется, согласно [2, 6], байесовская сеть (БС) $N=(X , G, M)$, которая состоит из

1) ацикличного ориентированного графа $G=(V, E)$ с вершинами $V=\{v_1,\ldots,v_n\}$ и дугами $E=V\times V$ (рис. 1);

2) множества переменных $X = (X_{v_1},\ldots,X_{v_n})$, элементы которого сопоставлены вершинам графа G, причем для каждой вершины сети определен размытый потенциал $\varphi_{v_i}(X_{v_i})$, где $i=1,\ldots,n$;

3) множества размытых потенциалов $\boldsymbol{\Phi}=\{ \varphi(X_{v_1}\mid X_{pa(v_1)}),\ldots,\varphi(X_{v_n}\mid X_{pa(v_n)}) \}$, где $X_{pa(v)}$ – родительские переменные для переменной X_v (если для вершины v множество родительских переменных пусто, тогда $\varphi(X_v\mid X_{pa(v)})=\varphi(X_v)$, такую вершину называют маргиналом).

Применение АРД к трансформированной БС приводит к узловому дереву [2, 7] со множеством клик и сепараторов. На рис. 2 показано узловое дерево (УД) $\tilde{T} =(\tilde{C} , \tilde{S})$, где $\tilde{C}$ – это множество клик, $\tilde{S}$ – множество сепараторов, для БС (рис. 1). Клики представляют собой вершины этого дерева $\tilde{T}$, а сепараторы обозначают его ребра. Напомним, что ребро между двумя соседними кликами C_i и C_j является их пересечением, т. е. $S = C_i\cap C_j$, где $S\in \tilde{S}$.



а б

РИС. 1. Пример фрагмента байесовской сети (а) и соответствующего ему узлового дерева (б)

Приведем модельный пример для наглядного представления введенных ключевых понятий нечеткой байесовской сети. Пусть A, B, C – переменные, каждая из которых может находиться в одном из трех четких взаимно независимых состояний: $A=(a1, a2, a3)$, $B=(b1, b2, b3)$, $C=(c1, c2, c3)$. Связь между этими переменными графически показана на рис. 1. Соответствующие размытые потенциалы $\varphi(A)$, $\varphi(B)$ и фрагмент потенциала для переменной C, представляющий собой распределение условных размытых оценок $\varphi(C \mid B, A)$, с функциями принадлежности треугольного типа, представлены на рис. 2.

Для потенциалов определены основные операции: *комбинация*($\otimes$), *проекция*($\downarrow$) (*маргинализация* [2, 6]) и дополнительная – *деление*. Для размытых потенциалов φ_1 и φ_2 комбинация, определенных над $\mathrm{dom}(X)$ и $\mathrm{dom}(Y)$ соответственно, для произвольного $z \in \mathrm{dom}(X \cup Y)$, поэлементное произведение $\varphi_1 \otimes \varphi_2$ определяется, согласно [6], по формуле

$$(\varphi_1 \otimes \varphi_2)(z) = \varphi_1(z_X)\varphi_2(z_Y), \qquad (1)$$

где z_X и z_Y – проекция конфигурации z на $\mathrm{dom}(X)$ и $\mathrm{dom}(Y)$ соответственно. Например, произведение потенциалов (рис. 2) на основе (1) создает потенциал, фрагмент которого показан на рис. 3.

Проекция размытого потенциала $\varphi^{\downarrow X}$ на X определяется [7] как суммирование или нахождение максимального значения над всеми переменными из $\mathrm{dom}(\varphi)$, кроме X. Применяя операцию *sum-маргинализации* [7] к потенциалу $\varphi(C \mid B,A)$ (см. рис. 1), получаем $\varphi^{\downarrow A}(C \mid B, A)=\varphi(A)$, а $\varphi^{\downarrow B}(C \mid B, A)=\varphi(B)$ (рис. 2). Применение *max-маргинализации* [7] к потенциалу позволяет найти конфигурацию с наибольшей оценкой: сравнение оценок осуществляется на основе сравнения их центров тяжести. Например, для потенциалов $\varphi(A)$ и $\varphi(B)$ результатом max-маргинализации будет конфигурация $A=a2$ и $B=b3$ соответственно (рис. 2), а для $\varphi(C \mid B, A)$ – ($C=c3, B=b3, A=a1$) с центром тяжести 10,96. Применение операции max-маргинализации над размытым потенциалом позволяет находить оптимальную конфигурацию для всех переменных БС, а реализация sum-маргинализации обеспечивает вычисление апостериорного распределения каждой переменной отдельно.

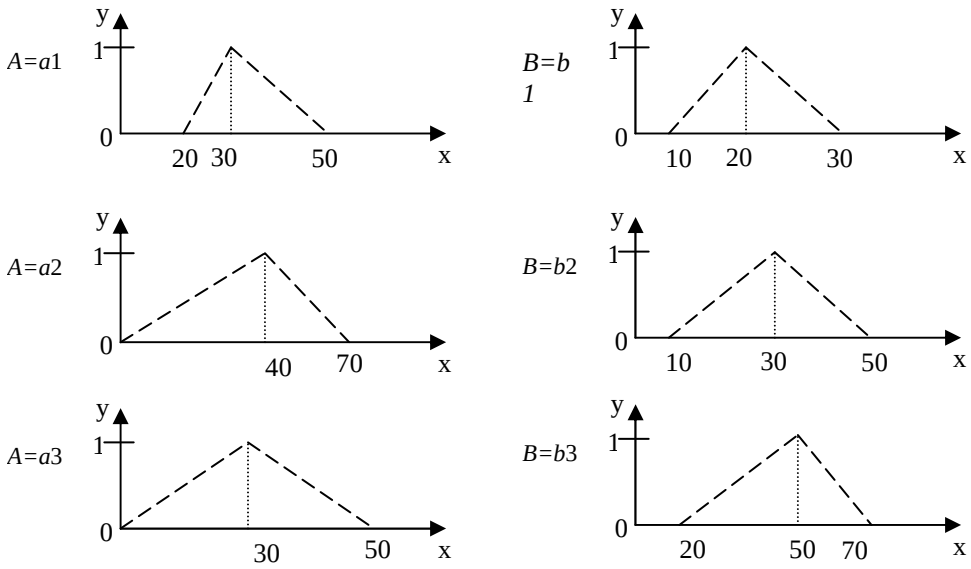


РИС. 2. Размытые потенциалы переменных *A,B,C*

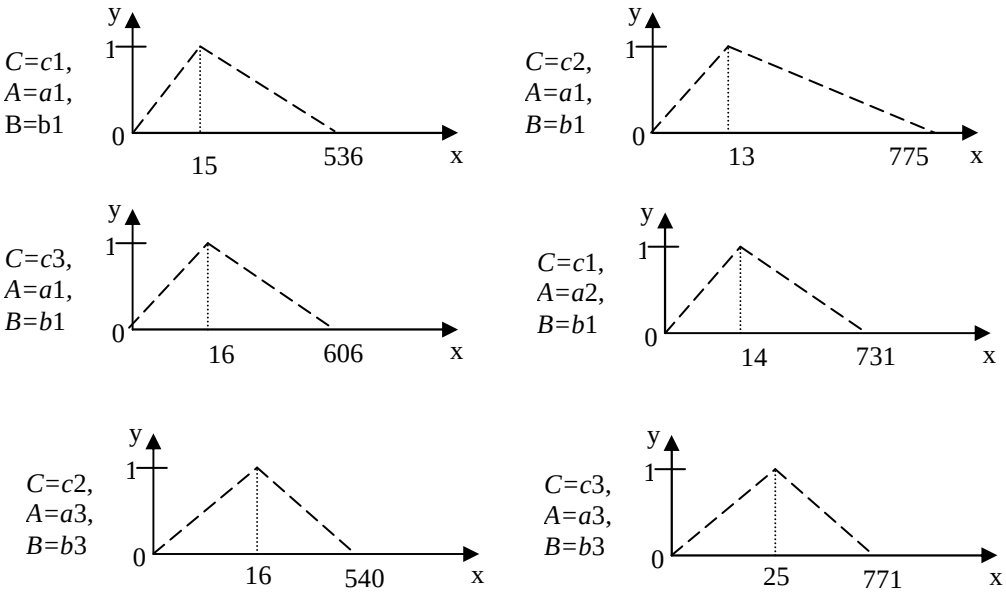


РИС. 3. Произведение размытых потенциалов переменных *A,B,C*

После создания узлового дерева $\tilde{T}=(\tilde{C},\tilde{S})$ (см. рис. 1, б) каждая его вершина инициализируется такими потенциалами: потенциал $\varphi\in\Phi$ присоединяется к клике так, чтобы $\mathrm{dom}(\varphi)\subseteq C$ для $C\in\tilde{C}$ Тогда над всеми $\varphi\in\Phi$, которые

присоединяются к одной и той же клике, – выполняется операция комбинация потенциалов. Например, для клики (*ABC*) (см. рис. 1, б) комбинируются потенциалы $\varphi(C \mid A, B)$, $\varphi(A)$, $\varphi(B)$, т. е. $\varphi(ABC) = \varphi(C|A, B)\bullet\varphi(A)\bullet\varphi(B)$ (см. рис. 3). Для ребра между кликами (*ABC*) и (*CGIK*) создается сепаратор $\varphi(C)\equiv 1$.

Алгоритм распространения вероятностей. Предположим, что в результате проведенных трансформаций сети доверия построено УД, к которому нужно применить АРД. В основе этого алгоритма находится процедура (алгоритм) вычисления значений потенциала одной клики на основе значений потенциала соседних клик (такие вычисления называются *передачей информации*, в данном случае нечеткой, между кликами). Различают две реализации этой процедуры: алгоритм S-S (Shenoy-Shafer) и его разновидность – алгоритм HUGIN [1, 2].

Причем, если процедура PASS_INFORMATION_S_S вызывается из клики C_i для клики C_j, то алгоритм S–S вычисляет апостериорное распределение оценок состояний тех переменных, которые входят в состав клики C_i. Чтобы вычислить апостериорные распределения для всех переменных, необходимо выполнить процедуру PASS_INFORMATION_S_S по направлению к другим кликам. При этом все вычисления выполняются с теми значениями потенциалов, которые были получены на этапе инициализации УД. Последнее условие значит, что процедура PASS_INFORMATION_S_S нуждается во время выполнения в дополнительной памяти для отдельного сохранения результатов промежуточных вычислений в сепараторах или в кликах.

Другой алгоритм – алгоритм HUGIN – является такой разновидностью алгоритма S-S, которая не нуждается в дополнительной памяти, поскольку хранит промежуточные вычисления в потенциалах соответствующих сепараторов и клик. Это достигается, согласно [1, 2], введением операции деления одного и того же сепаратора: новое значение сепаратора, полученное во время передачи информации из клики C_i в клику C_j, делится на старое значение, которое было вычислено на предыдущем шаге АРД во время передачи информации из клики C_j в клику C_i.

Известно [8], что множество потенциалов, с операциями комбинация и маргинализация, примеры которых указаны выше, образуют алгебру $(\Phi, D, \otimes, \downarrow)$, позволяющую выполнять локальные вычисления для потенциалов над УД, объединяя затем отдельные результаты в общий. Такая схема составляет основу алгоритма распространения доверия [1, 2]. Далее, выполнение АРД делится на два этапа (две фазы):

1) вычисление потенциала одной из клик УД (избранную произвольным способом клику называют корневой) на основе потенциалов остальных клик УД (рис. 4, а) – это выполнение процедуры COLLECT;

2) распространение вычислений на потенциалы остальных клик УД, начиная от корневой клики, избранной на этапе 1 (рис. 4, б), – это выполнение процедуры DISTRIBUTE.

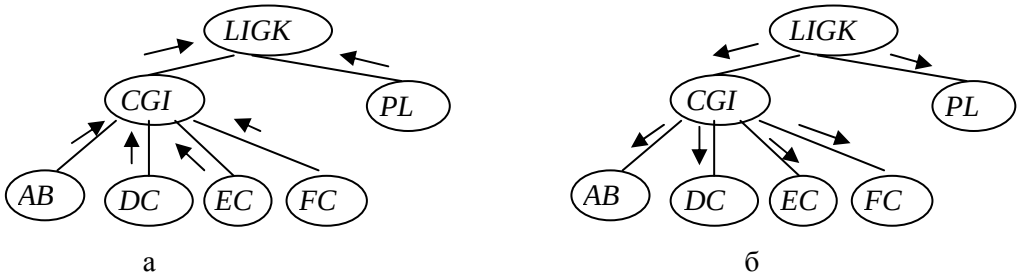


РИС. 4. Этапы выполнения АРД (корень УД – (*LIGKM*)):
а) выполнение процедуры COLLECT; б) выполнение процедуры DISTRIBUTE

Процедура COLLECT вычисляет для переменных корневой клики общее распределение нечетких вероятностей, которое учитывает все (непосредственные и опосредствованные) связи между всеми переменными БС и наличие свидетельств, поэтому вычисления начинаются от листьев дерева и заканчиваются в корневой вершине. Чтобы получить общее вероятностное распределение нечетких вероятностей для каждой оставшейся клики, выполняют вычисления в обратном порядке (от корневой клики к листьям) с помощью процедуры DISTRIBUTE. Используя рекурсивное описание обеих процедур, COLLECT реализовывают с помощью алгоритма поиск в «глубину» [2, 6, 9], а DISTRIBUTE – алгоритмом поиск в «ширину» [2, 6, 9]. Выполнение АРД осуществляется как для сети доверия с априорными числовыми параметрами, так и при наличии свидетельств (апостериорный вывод). В последнем случае изменяются только оценки состояний. Порядок выполнения АРД остается прежним.

После завершения выполнения АРД применяется операция sum-маргинализации к произвольной клике. Результатом применения является вероятностное распределение для каждой переменной байесовской сети. Если применяется max-маргинализация, тогда вычисляется наиболее вероятная конфигурация всех переменных байесовской сети.

Вероятностный вывод в байесовских сетях доверия в общем случае является NP-трудной задачей [8]. Поэтому повышение скорости работы АРД задача весьма актуальна. В данной статье предлагается один из подходов к ее решению. Он состоит в использовании эвристик, которые позволят избежать рекурсивных вычислений.

Нерекурсивный алгоритм распространения вероятностей. Рассмотрим, придерживаясь [10], алгоритмы обхода ориентированного графа – составных частей алгоритмов распространения вероятностей. Технологически – это рекурсивные алгоритмы, реализующие обход ориентированного графа в «глубину» и в «ширину», представленные единым обобщенным алгоритмом обхода графа с запоминанием дуг. Содержательно схема работы таких алгоритмов такова.

Сначала посещается (маркируется) начальная вершина обхода, а затем в цикле, с помощью дополнительной структуры данных, выполняется обход остальных вершин графа. Алгоритм изменяет состояние непомеченной вершины тогда и только тогда, когда она достижима из вершины, помещенной (маркированной) на предыдущей итерации цикла. Если дополнительная структура данных является стеком, то алгоритм маркирует достижимые вершины графа в порядке поиска в «глубину», а если очередью, тогда достижимые вершины графа в порядке поиска в «ширину». Такой алгоритм по временной сложности является весьма эффективным – имеет место линейная зависимость от числа дуг графа, достижимых из начальной вершины графа.

Для выполнения обхода УД, инициализированного размытыми потенциалами, с одновременным распространением доверия предложен модифицированный алгоритм обхода графа с запоминанием ребер, который объединяет все этапы АРД без использования рекурсивных вызовов процедур. Понятно, что такой алгоритм будет не только более прозрачным, но и более быстрым.

Нерекурсивный АРД в своей работе использует следующие структуры данных и процедуры:

1) структура данных Т –УД, например, рис. 1, б. Каждая из вершин дерева Т может иметь одно из двух значений маркера: mark_false (*неотмеченная)* или mark_true (*отмеченная);* в начале работы АРД – все вершины имеют значения маркера mark_false; список вершин смежных с вершиной *C* обозначается Adj(*C*), а их количество – count_Adj;

2) дополнительная структура данных Q – это список, содержащий указатель на соответствующие вершины УД; над элементами списка Q допустимы следующие операции:

- push_frontQ(element) – вставить данные element в начало списка Q;
- pop_frontQ() – удалить данные element из начала списка Q;
- push_backQ(element) –вставить данные element в конец списка Q.

3) элементом element списка Q является структура данных, которая имеет два информационных поля: а) element.child – для сохранения ссылки на вершину-наследницу; б) element.parent – для сохранения ссылки на вершину-родителя;

4) процедура mark(C) устанавливает для вершины *C* значение mark_true; *в*ершина, с которой начинается выполнение АРД, обозначается как Root;

5) процедура IsChild(C) возвращает значение true, если вершина *C* имеет немаркированные смежные вершины-потомки;

6) процедура pass_message(C_j, C_i) реализует один из способов, описанных выше, для пересылки сообщений между соседними кликами: каждой клике соответствует размытый потенциал, полученный на этапе трансформации БС, поэтому их комбинация и маргинализация выполняются согласно [5].

Сценарий выполнения АРД определяется списком Q, который имеет информационное поле – Q.kind, получающее значение из множества {"стек","очередь"}. Если во время вызова АРД установлено Q.kind="стек", тогда над элементами Q выполняются операции push_frontQ и pop_frontQ, а если

Q.kind="очередь ", тогда – операции pop_frontQ и push_backQ. Такая структура позволяет правильно выбирать направление передачи сообщений между кликами во время обхода УД.

При указанных выше условиях модифицированный АРД на языке C++ выглядит следующим образом:

Алгоритм распространения доверия (T, Root, Q.kind)

```
1. Q=∅ ;                                                    // сначала список пуст
2. C=Root;                                                  // выбор корня УД
3. mitka_L: mark(C)                                         // установка маркера
4.      if (Q.kind == "очередь ") {                         // выбор сценария
выполнения
       for (int i=0; i<count_Adj; i++) {                    //  для всех смежных с C
вершин
            C_i =Adj(C); pass_message(C, C_i) ;        // переслать сообщение_i
            element.child= C_i ;  push_backQ(element);  // добавить в конец списка Q
       }
     }
     else  {
     for (int i=0; i<count_Adj; i++) {                      // для всех смежных с C
вершин
          element.child=C_i  ;  element.parent=C  ;          //запомнить соседние
вершины
          push_frontQ(element) ;                        // добавить в начало списка Q
     }
     5. while  (Q<>∅)  {                                   // пока список Q не пуст
               element= pop_frontQ() ;                    //получить элемент списка
               C= element.child;                                  //получить ссылку на
    вершину-наследницу
               if ( IsChild(C)==false )  mark(C) ;      //если все потомки пройдены
    – маркировать C
          if (C.mark == mark_true) goto mitka_L;   //перейти к следующей
    вершине дерева T
                  else   pass_message(element.child, element.parent) ;  // иначе –
               переслать сообщение
          }
```

К полученным в результате выполнения алгоритма размытым потенциалам применяются вышеописанные операции маргинализации. Например, после выполнения АРД и sum-маргинализации (свидетельства отсутствуют) первоначальные значения размытых оценок для переменных *A*, *B* (см. рис. 2) изменятся так, как показано на рис. 5 (пунктиром показано начальная оценка, а сплошной линией – результат вычислений):

Для БС (см. рис. 1 – фрагмент интерфейса авторской компьютерной системы) при наличии свидетельств $A=a1$, $C=c2$ результат sum-маргинализации показан на рис. 6, а, а max-маргинализации – на рис. 6, б (как видно из рис. 6, б, наиболее вероятной апостериорной конфигурацией является конфигурация {*a*1, *b*3, *c*2, *d*3, *e*3, *f*2, *g*3, *i*3, *k*2, *l*2, *m*2, *p*2}). В обоих случаях для получения конечного результата выполняется дефузификация размытых оценок на основе вычисления их центра тяжести, согласно [5].

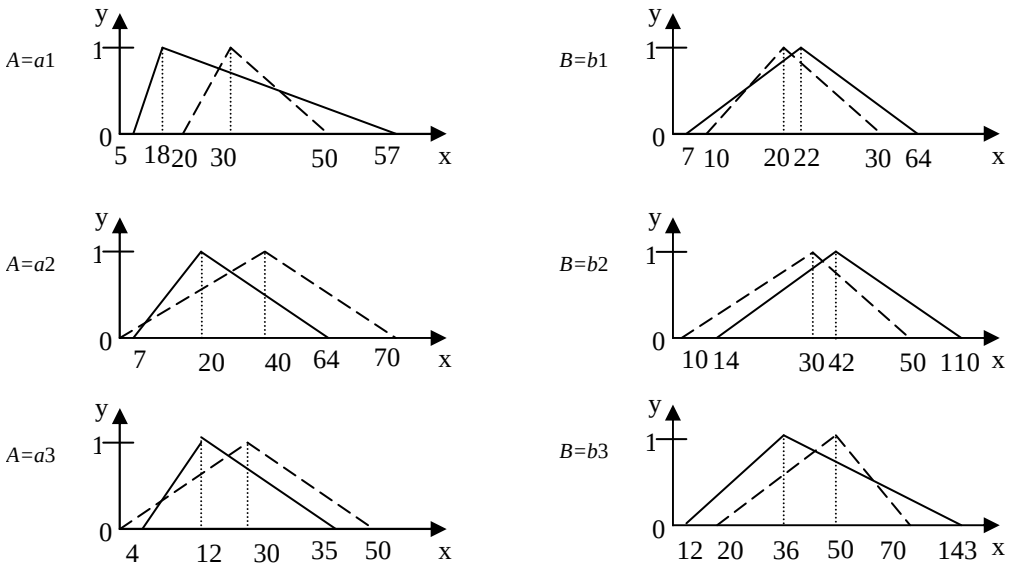


РИС. 5. Результаты распространения доверия на БС с размытыми потенциалами

	A	B	C	D	E	F	G	I	K	L	M	P
1	100,00	0,65	0,00	22,49	46,48	24,50	39,63	6,60	23,92	38,14	31,41	27,93
2	0,00	16,84	100,00	31,38	4,63	44,16	32,21	34,66	50,78	34,62	35,78	32,27
3	0,00	82,51	0,00	46,13	48,89	31,34	28,16	58,74	25,31	27,24	32,81	39,80

а

	A	B	C	D	E	F	G	I	K	L	M	P
1	100,00	0,78	0,00	85,32	91,86	79,07	91,86	9,43	58,99	85,02	85,02	68,84
2	0,00	20,41	100,00	74,00	5,66	100,00	83,37	91,86	100,00	100,00	100,00	84,24
3	0,00	100,00	0,00	100,00	100,00	85,02	100,00	100,00	85,02	71,65	75,67	100,00

б

РИС. 6. Результаты распространения доверия на БС с размытыми потенциалами

Корректность работы нерекурсивного АРД в нечеткой информационной среде. Корректность **выполнения** описанного выше нерекурсивного АРД для нечетких сетей доверия вытекает из следующих фактов:

1) процедуры *S*–*S* и HUGIN для размытых потенциалов однозначно соответствуют общим схемам алгоритмов [5, 8], сконструированных для потенциалов алгебры $(\Phi, D, \otimes, \downarrow)$;

2) операции над размытыми потенциалами, согласно [5, 8] и принципа расширения, сохраняют все свойства для корректного выполнения вычисления над узловым деревом;

3) нерекурсивный АРД выполняет в зависимости от параметра Q.kind обход дерева в «глубину» или в «ширину», в соответствии с [10].

Количество операций, выполняемых нерекурсивным АРД при обходе УД в «глубину» или в «ширину», линейно зависит от числа ребер УД. Таким образом, для нерекурсивного АРД, более строго, временная сложность обхода дерева составляет $O(k)$, где, k – количество ребер дерева T, которые достигаются во время обхода из вершины Root.

Поскольку дополнительная структура данных (список), используемый в нерекурсивном АРД, содержит лишь указатели на вершины УД, то объем памяти, который она занимает, пропорционален количеству вершин УД, т. е. $O(p(k+1))$, где k – количество ребер в УД, p – количество байт, отводимых для размещения в памяти указателя. Из этого следует, что построенный АРД использует также и меньше оперативной памяти по сравнению с рекурсивными версиями АРД, в которых отдельный вызов выполняется для каждой вершины УД.

Выводы. Таким образом, построен унифицированный алгоритм распространения вероятностей над нечеткими байесовскими сетями. Этот алгоритм объединил все этапы распространения доверия, избежав при этом рекурсивных вызовов для каждой клики узлового дерева. Построенный алгоритм основывается на модифицированном общем алгоритме обхода графа с запоминанием дуг и дополнительной структуре данных, сохраняющей указатели на объекты. Модифицированный алгоритм распространения вероятностей позволяет вычислять априорные и апостериорные оценки состояний сложных систем на основе размытых знаний о предметной области путем использования размытых потенциалов. Для вычисления оценок доверия могут использоваться альтернативные стратегии вывода, что позволяет сравнивать их результаты для разных моделей одного процесса или системы. Построенный алгоритмы выполняется корректно и требует меньших затрат вычислительных ресурсов по сравнению с применением аналогичных рекурсивных алгоритмов распространения доверия.

Полученные результаты использованы при построении соответствующей нечеткой информационной технологии моделирования состояний сложных систем в условиях неопределенности и размытости причинно-следственных связей. Примеры, использованные в статье, являются результатом применения нерекурсивного алгоритма распространения вероятностей к модельным нечетким байесовским сетям.

І.М. Парасюк, Ф.В. Костукевич

ПРО ОДИН ЕФЕКТИВНИЙ АЛГОРИТМ ПОШИРЕННЯ ЙМОВІРНОСТЕЙ
В НЕЧІТКИХ БАЙЄСІВСЬКИХ МЕРЕЖАХ ДОВІРИ

Розглянута загальна схема алгоритма розповсюдження ймовірностей над нечіткими байєсівськими мережами довіри. Побудовано уніфікований алгоритм розповсюдження довіри на основі розмитих потенціалів. Доведена коректність виконання цього алгоритма та вказані умови його застосування, за яких цей алгоритм є найбільш ефективним.

I.M. Parasyuk, F.V. Kostukevich

AN EFFECTIVE ALGORITHM FOR PROPAGATION OF PROBABILITIES
IN FUZZY BAYESIAN BELIEF NETWORKS

A general scheme of belief propagation algorithms above the fuzzy Bayesian networks of belief is considered. The new belief propagation algorithm is built on the basis of fuzzy potentials. Correctness of this implementation of algorithms is proved and conditions of his applications at which this algorithm is most effective are indicated.

1. *Pearl J.* Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. – Morgan Kaufmann, San Mateo, CA, 1991. – 552 p.
2. *Cowell R.G., Dawid A.P., Spiegelhalter D.J., Lauritzen S.L.* Probabilistic Networks and Expert Systems. – New York: Springer–Verlag, Inc., 1999. – 321 p.
3. *Верьовка О.В., Парасюк И.Н.* О распространении вероятностей в нечетких байесовских сетях с недетерминированными состояниями // Кибернетика и системный анализ. – 2008. – № 6. – С. 153–169.
4. *Zadeh L.A.*, Fuzzy Sets, Information and Control, 1965. –**8**. – P. 338–353.
5. *Парасюк И.Н., Костукевич Ф.В.* Нечеткие потенциалы и вопросы их применения в алгоритмах распространения доверия на байесовских сетях // Компьютерная математика, – 2009. – № 1. – С. 70–76.
6. *Uffe B.Kjaerulff, Anders L.Madsen* Bayesian Networks and Influence Diagrams. – Springer Science+Business Media, LLC, 2008. – 318 p.
7. *Парасюк И.Н., Костукевич Ф.В.* Методы трансформации байесовской сети для построения узлового дерева и их модификация // Компьютерная математика. – 2008. – № 1. – С. 70–80.
8. *Shenoy P.P.* Valuation-based systems for discrete optimization. // In Uncertainty in Artificial Intelligence (ed. P.P.Bonissone, M.Henrion,L.N.Kanal and J.F.Lemmer): North-Holland, Amsterdam, The Netherlands. 1991. – N 6. – P. 385–400.
9. *Кормен Т., Лейзерсон Ч., Ривест Р.* Алгоритмы: построение и анализ / Пер. с англ. под ред. А. Шеня. – М.:МЦНМО: БИНОМ. Лаборатория знаний, 2004. – 960 с.
10. *Касьянов В.Н., Евстигнеев В.А.* Графы в программировании: обработка, визуализация и применение. – СПб.: БХВ-Петербург, 2003. – 1104 с.

Получено.15.04.2010

Об авторах:

Парасюк Иван Николаевич,
доктор технических наук, профессор, член-корреспондент НАН Украины,
зав. отделом Института кибернетики имени В.М. Глушкова НАН Украины,
E-Mail: ivpar1@i.com.ua

Костукевич Феликс Витальевич,
аспирант Института кибернетики имени В.М. Глушкова НАН Украины.
E-Mail: internat_m@fk.lutsk.ua