УДК 519.81

ОБОБЩЕНИЕ СХЕМЫ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ НА ОСНОВАНИИ ЛЕКСИКОГРАФИИ БИНАРНЫХ ОТНОШЕНИЙ

Ю.В. Бугаев, Ф.А. Музалевский

Кафедра «Информационные технологии моделирования и управления», ФГБОУ ВПО «Воронежский государственный университет инженерных технологий», г. Воронеж; y_bugaev52@mail.ru

Представлена членом редколлегии профессором В.И. Коноваловым

Ключевые слова и фразы: бинарное отношение; вычислительная сложность; динамическое программирование; функция выбора; эффективные пути.

Аннотация: Предложено обобщение схемы динамического программирования на случай трактовки понятия оптимальности в виде механизма выбора альтернатив, недоминируемых по заданному бинарному отношению. Введено понятие выбора по лексикографии бинарных отношений для сужения множества недоминируемых решений.

1. Постановка задачи

В работе [1] предложено обобщение схемы динамического программирования (**ДП**), описанное в терминах языка функций выбора (**ФВ**), – наиболее универсального и удобного для анализа описания концепции выбора. Кратко это описание сводится к следующему.

Пусть имеем процесс, содержащий N стадий. Состояние процесса после завершения k-й стадии определяется значением некоторого параметра x_k, $k = 1, \ldots, N$; x_0 определяет начальное состояние, x_N – конечное. Формально k-я стадия представима переходом из состояния x_{k-1} в x_k под действием управления, характеризующегося значением параметра $u_k \in U_k(x_{k-1})$. Иными словами k-я стадия определяется уравнением вида

$$x_k = \chi_k (x_{k-1}, u_k), \tag{1}$$

где χ_k – некоторые функции. На каждом k-м шаге необходимо выбрать определенное множество состояний $\{x_k\}$ и в результате получить набор лучших траекторий.

Пусть X – некоторое множество траекторий процесса, движение по которым происходит согласно уравнению (1). Рассматривая каждую траекторию $p \in X$ как допустимое альтернативное решение некоторой задачи выбора, а X – как некое предъявление, введем ФВ $C(X)$.

Назовем C-оптимальными все траектории предъявления, попавшие в выбор.

Предположим, что ФВ определена не только на целых траекториях, но и на каждой их части. Пусть все траектории множества X имеют общий фрагмент b, то есть для некоторой совокупности Y фрагментов траекторий имеем соотношение

$$X = Y \,\&\, b, \tag{2}$$

символ “&” означает операцию “склеивания” или соединения траекторий.

Определение 1. Будем говорить, что ФВ обладает свойством слабой аддитивности, если для любого множества Y неполных траекторий, удовлетворяющего соотношению (2) при некоторой совокупности траекторий X и фиксированном фрагменте b, справедливо равенство

$$C(Y \,\&\, b) = C(Y) \,\&\, b.$$

Пусть $D_k(x_{k-1})$ – множество C-оптимальных траекторий вида $x_{k-1} \to x_N$. Предлагается при начальном условии $D_{N+1}(x_N) = \varnothing$ рекуррентное соотношение

$$D_k(x_{k-1}) = C\left[\bigcup_{u \in U_k}\left[\left(x_{k-1} \to \chi_k(x_{k-1}, u)\right) \,\&\, D_{k+1}\left(\chi_k(x_{k-1}, u)\right)\right]\right],\ k = N,\ldots,1, \quad (3)$$

которое является обобщением известного уравнения Беллмана

$$F(x_{k-1}) = \operatorname*{opt}_{u \in U_k} \{f(x_{k-1}, u) + F[\chi(x_{k-1}, u)]\}. \quad (4)$$

Было доказано, что в случае, когда ФВ C (•), определенная на множестве траекторий многошагового процесса и их отдельных фрагментов, обладает свойствами наследования, отбрасывания и слабой аддитивности, то набор $D_1(x_0)$, полученный посредством рекуррентного соотношения (3), совпадает с множеством C-оптимальных траекторий процесса.

Этот результат позволил с единых позиций провести анализ ряда алгоритмов, вписывающихся в данную концепцию. Однако вскоре выяснилось, что автоматическое перенесение на схему обобщенного ДП известных ФВ не вполне корректно и нуждается в определенных доработке и формальном обосновании, которые и предлагаются в настоящей статье.

Для иллюстрации этого факта рассмотрим вариант известной задачи о рюкзаке, названный в [2] «целочисленный рюкзак».

Пример 1. Заданы n вещей, их размеры w_i, стоимости s_i, и некоторое число H – вместимость рюкзака (все числа – натуральные). Найти количество штук z_i каждой вещи, чтобы выполнялось условие $\sum_{i=1}^{n} z_i w_i \le H$ и при этом суммарная ценность вещей $f = \sum_{i=1}^{n} z_i s_i$ была бы максимальной.

Решим задачу методом ДП при следующих данных: n = 4; w= (2; 3; 5; 7); s = (3; 5; 8; 11) и H = 10. Получим ответ: z = (2; 2; 0; 0), $f^{\max}$ = 16, а таблица условно-оптимальных решений задачи будет иметь вид

Таблица 1

Условно-оптимальные решения задачи о целочисленном рюкзаке (традиционный подход)

k	10	9	8	7	6	5	4	3	2	1	0
$F(x_k)$	0	0	3	5	6	8	10	11	13	15	16
$u^{\text{opt}}(x_k)^*$	0	0	2	3	2	2	3	2	2	3	2

$^*u^{\text{opt}}(x_k)$ – условно-оптимальный размер вещи, выбираемой на k-й стадии.

Однако несложно убедиться, что $z = (1; 1; 1; 0)$ тоже является оптимальным решением. Очевидно, алгоритм ДП выбрал первый вариант из-за того, что при поиске максимального элемента в некотором наборе ответом всегда является первый из одинаковых максимальных элементов. Исследуем корректность такого подхода с точки зрения формальной теории ФВ.

Функция выбора, реализующая скалярный оптимизационный механизм, имеет следующее формальное представление [3]

$$C_{CO}(X) = \{x \in X \mid x = \operatorname{argmax} f(x)\}, \quad (5)$$

где $f(x)$ – некоторая скалярная функция. Далее, согласно классическому определению [4], точка x^* называется точкой (глобального) максимума функции $f(x)$ на множестве X, если $f(x^*) \geq f(x)$ для всех $x \in X$ – нестрогое неравенство.

Пусть X – конечное множество, имеющее несколько точек максимума. Тогда из сформулированных определений следует, что при буквальном следовании представлению (5) в выбор должны попасть **все** эти точки. Это означает, что при подстановке в формулу (3) ФВ (5) набор условно-оптимальных решений задачи о целочисленном рюкзаке должна иметь вид, представленный в табл. 2. Тогда, как несложно проверить, при обратном ходе схемы ДП мы получим 15 вариантов оптимального управления или без учета порядка следования вещей в укладке – четыре решения (2, 2, 0, 0); (1, 1, 1, 0); (0, 1, 0, 1); (0, 0, 2, 0) (см. табл. 2).

Очевидно, что при увеличении размерности задачи число подобных дублирующих вариантов будет быстро расти, а так как их критериальные оценки одинаковы, то, скорее всего, нет никакого смысла все их отыскивать. Так что традиционный подход к поиску оптимума в схеме ДП оправдан, разве что некоторое сомнение в справедливости такого пути возникает из-за того, что ответ задачи будет различным при разной нумерации в исходных данных.

Ясно, что с подобной ситуацией мы столкнемся при использовании и других ФВ, например, реализующей паретовский механизм, и в общем случае уже нельзя с полной уверенностью ратовать за отбраковку альтернатив по их порядковым номерам. Так что актуальной представляется задача корректировки формальных определений традиционных ФВ с целью сделать их полностью пригодными для реализации обобщенной схемы ДП. Сокращение числа «хороших» альтернатив на каждой стадии процесса позволило бы значительно уменьшить вычислительную сложность задач, решаемых при помощи рекурсивных и итерационных методов с большой глубиной вложенности. Для метода ДП, как одного из распространенных способов решения широкого круга задач, вывод условий подобного сокращения представляется также крайне полезным.

В данной работе исследуется достаточно распространенный случай использования ФВ, основанной на механизме блокировки по некоторому бинарному отношению (**БО**).

Таблица 2

Условно-оптимальные решения задачи о целочисленном рюкзаке (использование функции выбора (5))

k	10	9	8	7	6	5	4	3	2	1	0
$F(x_k)/u^{\text{opt}}$	0/0	0/0	3/2	5/3	6/2	8/2	10/3	11/2	13/2	15/3	16/2
						8/3		11/3	13/3		16/3
						8/5		11/5	13/5		16/5
								11/7			16/7

2. Построение модифицированного бинарного отношения

В случае, когда ФВ порождается некоторым бинарным отношением, свойства этого отношения во многом влияют на свойства самой ФВ. В частности, известно [3, с. 41], что структуры, представленные асимметричными и транзитивными БО, порождают непустые ФВ, обладающие свойствами наследования, согласия и отбрасывания. Именно эти свойства ФВ, как было сказано в п. 1, дают возможность применять метод ДП. Кроме того, ФВ должна обладать слабой аддитивностью. Данное свойство было впервые введено в работе [1], и важно установить его связь со свойствами БО, порождающего применяемую ФВ.

Определение 2. Пусть на множестве траекторий и их фрагментов некоторого многостадийного процесса определено БО R. Будем говорить, что R обладает свойством инвариантности относительно сдвига, если выполнено условие

$$\forall\, x, y \in A,\ \ \forall b \mid x\&b, y\&b \in A' : (x, y) \in R \Leftrightarrow (x\&b, y\&b) \in R. \qquad (6)$$

Теорема 1. Для того чтобы ФВ, основанная на механизме блокировки по бинарному отношению R, обладала свойством слабой аддитивности, необходимо и достаточно, чтобы R было инвариантно относительно сдвига.

Доказательство. Напомним, что механизм блокировки определяется следующим правилом [3]

$$C^R(X) = \{x \in X \mid \forall y \in X\, (y, x) \notin R\}.$$

Достаточность. Пусть для R выполнено соотношение (6), и пусть Y – некоторое множество неполных траекторий. Выберем произвольную альтернативу x, что $x \in C^R(Y)$. Тогда $(x\&b) \in C^R(Y)\&b$ и для любой альтернативы y будет $(y, x) \notin R$. Следовательно $(y\&b, x\&b) \notin R$. Значит $(x\&b) \in C^R(Y\&b)$. Отсюда $C^R(Y)\&\ b \subseteq C^R(Y\&b)$. Аналогично доказывается и обратное включение. Значит $C^R(Y\&b) = C^R(Y)\&\ b$.

Необходимость. Пусть $(x\&b) \in C^R(Y\&b)$. Значит $x \in C^R(Y)$. Тогда для любой траектории $(y\&b) \in (Y\&b)$ будет $(y\&b, x\&b) \notin R$. Предположим, что отношение $(y, x) \notin R$ не выполняется. Но тогда $x \notin C^R(Y)$ – противоречие. Значит

$$(y\&b, x\&b) \notin R \Rightarrow (y, x) \notin R.$$

Аналогично доказывается и обратное следование. В итоге получим условие (6). Теорема доказана ■

Известно [3], что для однозначности выбора, то есть чтобы он состоял из единственного элемента, необходимо и достаточно, чтобы БО было ациклично и слабо полно. Поэтому для уменьшения объема множества недоминируемых альтернатив необходимо использовать для сравнения вариантов такое БО, которое будет удовлетворять перечисленным выше требованиям.

В целом, их набор получается достаточно обширным. Более приемлемым кажется формирование такой ФВ, которая сужает набор равноценных вариантов, но при этом ориентируется на более объективный критерий, чем порядковый номер решения. Отталкиваться будем от случая, когда для организации выбора в том или ином виде применяется отношение качественного порядка [3], то есть асимметричное и транзитивное БО P. Таковыми являются, в частности, часто используемые в приложениях отношения Парето и Слейтера. Тогда при выборе на основе механизма блокировки отбрасываются те варианты, которые «хуже» хотя бы одного из находящихся в предъявлении. Зачастую, многие альтернативы, по-

павшие в выбор, в некотором смысле эквивалентны между собой. Для их отсева и требуется построить решающее правило путем модификации P.

Пусть P – исходное отношение качественного порядка, заданное на некотором множестве альтернатив A; I – отношение эквивалентности, определенное на множестве недоминируемых по P альтернатив и S – некоторое новое БО, введенное для ужесточения выбора на множестве эквивалентных решений. Модифицируем отношение P по следующей формуле

$$P^M = P \cup (I \cap S). \tag{7}$$

Наша задача – установить свойства вводимых отношений I и S, при которых ФВ, основанная на блокировке по P^M, будет обладать свойствами наследования, отбрасывания и слабой аддитивности. Иными словами, P^M также должно быть асимметричным и транзитивным.

Известно, что из антирефлексивности транзитивного БО следует его асимметричность и что качественный порядок антирефлексивен. Поэтому выясним условия антирефлексивности P^M. Если оно рефлексивно, то

$$x[P \cup (I \cap S)]x \Leftrightarrow [x(P \cup I)x] \wedge [x(P \cup S)x].$$

Антирефлексивность же означает ложность записанных отношений. Известно [3, с. 13], что объединение произвольного числа антирефлексивных отношений является антирефлексивным отношением, поэтому вторая половина конъюнкции будет ложной, если S антирефлексивно. Следовательно, в этом случае ложной будет и вся правая часть следования.

Теперь выясним условие транзитивности. Известны два свойства транзитивных БО [3, с. 14]:

1) пересечение любого числа транзитивных отношений транзитивно;

2) объединение двух бинарных отношений снова транзитивно, если одно из них транзитивно относительно другого.

Значит, из транзитивности I и S следует транзитивность $I \cap S$. Поскольку I транзитивно по определению, получаем, что для транзитивности $I \cap S$ требуется лишь транзитивность S, а для транзитивности $P \cup (I \cap S)$, надо чтобы либо

а) P было транзитивно относительно $I \cap S$, либо

б) $I \cap S$ было транзитивно относительно P.

Выясним условия, при которых выполняется соотношение а). Напомним, что отношение P транзитивно относительно отношения Q в том случае, если выполняются следующие условия

$$\begin{cases} xPy \wedge yQz \Rightarrow xPz \\ xQy \wedge yPz \Rightarrow xPz \end{cases}. \tag{8}$$

Пусть для некоторых x, y, z имеем $xPy \wedge yIz \wedge ySz$. Тогда для выполнения условия xPz достаточно, чтобы хотя бы одно из отношений I и S было транзитивно относительно P. Для второго соотношения свойства относительной транзитивности имеют аналогичное требование.

Несложно показать, что для соблюдения условия б) требуется, чтобы оба отношения I и S были транзитивны относительно P.

Таким образом, получаем следующие условия, при которых $P \cup (I \cap S)$ антирефлексивно и транзитивно.

1°. S должно быть антирефлексивно и транзитивно.

2°. Должно выполняться хотя бы одно из условий:

– P транзитивно относительно I или S;

– оба отношения I и S транзитивны относительно P.

Наконец, для однозначности выбора, то есть чтобы построенная ФВ позволяла из каждого класса эквивалентных решений выбирать единственного представителя, БО P^M должно быть слабо полным на каждом классе эквивалентности. Иными словами,

$$\forall x, y \in A \quad ((x, y) \in I \wedge x \neq y) \Rightarrow ((x, y) \in P^M \vee (y, x) \in P^M).$$

Очевидно, что для этого достаточно, чтобы подобным свойством обладало отношение S.

Для иллюстрации полученного результата рассмотрим несколько важных частных случаев.

Пример 2. Пусть каждое из отношений P и S задается некоторым скалярным критерием q_P и q_S соответственно, каждый из которых надо максимизировать

$$(x, y) \in P \Leftrightarrow q_P(x) > q_P(y); \ (x, y) \in S \Leftrightarrow q_S(x) > q_S(y).$$

Очевидно, множеством решений, эквивалентных по P, будет набор вариантов, для которых значения q_P совпадают. Тогда отношение P^M будет задаваться следующим правилом

$$(x, y) \in P^M \Leftrightarrow [q_P(x) = q_P(y)] \wedge [q_S(x) > q_S(y)].$$

Построенное БО P^M представляет собой отношение лексикографии по последовательно применяемым критериям q_P и q_S. В дальнейшем будем БО, построенное по правилу (7), по аналогии называть *лексикографией бинарных отношений* P и S и обозначать $P^M = \mathrm{Lex}(P, S)$. В рассмотренном скалярном случае q_S может быть каким-нибудь дополнительным, неглавным критерием. К примеру, при решении однокритериальной задачи о рюкзаке таким критерием может быть минимум количества выбранных вещей не по массе, а в штуках. В отличие от минимума номера альтернативы, выбор по такому критерию не зависит от порядка нумерации альтернатив. В этом случае для исходных данных, приведенных в примере 1, получим три варианта оптимального управления или два набора вещей: (0, 0, 2, 0) и (0, 1, 0, 1). Причина наличия двух решений, а не одного, заключается в том, что для имеющихся исходных данных при одинаковой суммарной стоимости вещей их количество также может быть одинаковым. Иными словами, введенное бинарное отношение не является слабо полным.

Пример 3 [5]. На множестве альтернатив водится исходное отношение R нестрогого превосходства – транзитивное, рефлексивное («не хуже, чем»). Оно порождает следующие отношения:

1) строгого превосходства P – транзитивное, антирефлексивное («лучше, чем»): $xPy \Leftrightarrow (xRy) \wedge \neg(yRx)$;

2) несравнимости N – симметричное, антирефлексивное («несравнимо»): $xNy \Leftrightarrow \neg(xRy) \wedge \neg(yRx)$;

3) безразличия I – транзитивное, рефлексивное, симметричное («одинаково хороши»): $xIy \Leftrightarrow (xRy) \wedge (yRx)$.

Отношение строгого превосходства P является, очевидно, качественным порядком, а безразличие I является эквивалентностью. Несложно показать, что при организации выбора недоминируемых по P альтернатив мы получим множество таких вариантов, что для любой отобранной пары имеет место отношение несравнимости N или безразличия I.

Утверждение 1. Отношение строгого превосходства P транзитивно относительно БО безразличия I.

Доказательство. Пусть xPy и yIz. Надо получить xPz. Распишем представление посылки и предполагаемого следствия в (8) через нестрогое превосходство R:

$$xPy \wedge yIz \Leftrightarrow xRy \wedge \neg(yRx) \wedge yRz \wedge zRy;$$
$$xPz \Leftrightarrow xRz \wedge \neg(zRx).$$

Следствие xRz получаем из $xRy \wedge yRz$. Предположим, что $\neg(zRx)$ не верно, то есть на самом деле zRx. Тогда из условия $yRz \wedge zRx$ получаем yRx, что по предположению не верно. Следовательно, $\neg(zRx)$ верно.

Следствие $xIy \wedge yPz \Rightarrow xPz$ доказывается аналогично.

Таким образом, при синтезе Lex(P, S) нам достаточно выбрать в качестве S произвольный качественный порядок, так как условие 2° выполняется автоматически.

В случае установления БО R на множестве многокритериальных альтернатив по правилу $xR\,y \Leftrightarrow \forall\ i, f_i(x) \geq f_i(y)$ отношение P – не что иное, как отношение Парето, наличие N означает несравнимость по Парето, а I – совпадение критериальных оценок x и y по всем критериям. Отношение S может, как и в скалярном случае, задаваться строгим превосходством по дополнительному критерию.

Пример 4. Пусть дано БО качественного порядка P. Качественным безразличием $I_{\text{КБ}}$ называется рефлексивное, симметричное отношение, отрицающее наличие качественного порядка между альтернативами

$$(x, y) \in I_{\text{КБ}} \Leftrightarrow (x, y) \notin P \wedge (y, x) \notin P.$$

Качественное безразличие индуцирует по формуле

$$(x, y) \in I \Leftrightarrow \forall u\ ((u, x) \in I_{\text{КБ}} \Leftrightarrow (u, y) \in I_{\text{КБ}}), \qquad (9)$$

рефлексивное, симметричное, транзитивное отношение качественной эквивалентности I.

Положив в (9) $u = x$, получим, что из $(x, y) \in I$ следует $(x, y) \in I_{\text{КБ}}$. То есть $I \subseteq I_{\text{КБ}} = \overline{P} \cap \overline{P}^{-1}$, а, значит $I \cap P = \varnothing$, $I \cap P^{-1} = \varnothing$.

Утверждение 2. Отношение качественного порядка транзитивно относительно качественной эквивалентности.

Доказательство. Пусть $(x, y) \in P$ и $(y, z) \in I$. Предположим противное, то есть $(x, z) \notin P$. Тогда возможны два случая:

1) $(x, z) \in I_{\text{КБ}}$;

2) $(z, x) \in P$.

В первом случае по определению I имеем

$$(y, z) \in I \Rightarrow ((x, z) \in I_{\text{КБ}} \Leftrightarrow (x, y) \in I_{\text{КБ}}).$$

Но в силу предположения $(x, y) \in P$ включение $(x, y) \in I_{\text{КБ}}$ не может выполняться, следовательно и $(x, z) \notin I_{\text{КБ}}$.

Во втором случае в силу транзитивности P получим, что из $(z, x) \in P$ и $(x, y) \in P$ следует $(z, y) \in P$. То есть, не может выполняться $(y, z) \in I$.

Следовательно, в обоих случаях получаем противоречие.

Вторая половина условия относительной транзитивности доказывается аналогично.

Таким образом, в описанном случае при синтезе Lex(P, S) нам достаточно учитывать только свойства БО S, то есть выполнение условия 1°, поскольку 2° выполняется автоматически.

3. Алгоритмы восстановления решений при наличии эквивалентности

На первом этапе схемы ДП, в ее традиционном варианте, на каждой k-й стадии запоминаются условно-оптимальные решения, то есть оптимальные управления для каждого состояния x_{k-1} процесса записываются в множества $U^k_{\text{opt}}(x_{k-1})$. На втором этапе восстанавливается оптимальная последовательность состояний.

Для многокритериального случая и других механизмов выбора, отличающихся от скалярно-оптимизационного, этот подход также можно использовать. Оптимальная последовательность состояний восстанавливается посредством следующей рекурсивной процедуры (этот и последующие алгоритмы удобно описать на неформальной версии языка Pascal).

```
Procedure  Search (k);
begin if  k = n + 1 then writeln(STACK)
        else
for u ∈ U_opt(k) do
begin  y := top(STACK);
x[k] := χ (y, u);
x[k] => STACK;
Search (k+1);
v <= STACK;
end;
end;
```

```
{Головная программа}
begin   STACK := x[0];    Search (1);
end.
```

Здесь $\chi\ (x_{k-1},\ u_k)$ – функция (1), определяющая состояние процесса x_k на k-й стадии по значениям предыдущего состояния x_{k-1} и управления u_k; оптимальные последовательности состояний накапливаются в информационной структуре *STACK*; операции *x => STACK* и *y <= STACK* означают запись в *STACK* элемента *x* и исключение из *STACK* его верхнего элемента и запись в переменную *y*, соответственно; *y:= top*(*STACK*) – запись первого элемента набора *STACK* в переменную *y* без его исключения из *STACK*.

В частности, при поиске оптимальных путей в графе между двумя заданными вершинами *s* и *t* восстановление решений будет выглядеть так:

```
Procedure  Search_G;
begin y := top(STACK);
if  y = s then writeln(STACK);
        else
for u ∈ ПРЕДШ_opt(y) do
begin  u => STACK;
Search_G;
v <= STACK;
end;
end;
```

```
{ Головная программа }
begin  read (s, t);  STACK := t;    Search_G;
end.
```

Нерекурсивный вариант:

```
begin      read (s, t);
STACK := t;
while (STACK  <> ∅) do
begin v := top(STACK);
while (v <> s)  and  (ПРЕДШ_opt(v)  <> ∅) do
begin u <= ПРЕДШ_opt(v);   v := u;
u => STACK;
end;
if  v = s  then writeln(STACK);
w <= STACK;
< Восстановить ПРЕДШ_opt(w) >;
end;
end.
```

Здесь *ПРЕДШ_opt* [*v*] – список вершин ориентированного графа, предшествующих вершине *v* и включенных в список условно-оптимальных решений. Он организован по принципу *STACK*.

Очевидно, описанные алгоритмы будут корректно функционировать как при отсутствии, так и при наличии эквивалентных решений.

Однако в некоторых алгоритмах, основных на численной схеме ДП, условно-оптимальные решения в явном виде не получаются. Это относится, например, к известному методу Флойда, предназначенному для поиска оптимальных путей между всеми парами вершин произвольного графа. В этом методе, зная значения $D[i, j]$ критериальных оценок кратчайших путей между всеми парами вершин (i, j), можно построить сами пути между двумя заданными вершинами s и t с помощью алгоритма, описанного в [6]. В нем используется достаточно простая идея обнаружения последней вершины u восстанавливаемого пути по совпадению

$$D[s, v] = D[s, u] + A[u, v],$$

где $A[u, v]$ – вес дуги $[u, v]$.

```
read(s, t);
write(t);    v := t;
 while v <> s do
 for u :=1 to  n do
 if  ( D[s, v] = D[s, u] + A[u, v] ) and  (u<>v)  then
 begin  write(u);  v:= u;  break;  end;
```

Подобный подход удобен тем, что он более универсален, не использует память для хранения самих условно-оптимальных решений (а только для их критериальных оценок), и при этом затраты на дополнительное время для расчетов минимальны.

Данный подход несложно распространить и на другие алгоритмы, основанные на численной схеме ДП. Например, в статье [7] приведен алгоритм восстановления «первых подходящих» Парето – оптимальных путей по множеству их многокритериальных весов $\{D[s, t]\}$ в векторном варианте метода Флойда. Данный алгоритм (назовем его «Алгоритм 1») имеет вид:

```
read (s, t);
   for z ∈ {D[s, t]} do
    begin  v := t;   STACK := t;   q := z;
while v <> s  do
 for  u∈ ПРЕДШ [v]  do
for p ∈ {D[s, u]} do
if   q = p + A[u, v]   then
begin u => STACK;  q := p;   v := u;
break (for u );
end;
 write(STACK);
end;
```

Здесь *break* (*for u*) означает прерывание цикла по *u*. Вершины восстанавливаемого пути накапливаются в *STACK*, содержимое которого распечатывается, как только первая вершина пути совпадет с заданной начальной вершиной *s*.

Оценим вычислительную сложность алгоритма. Пусть k – число критериев; n, m – количество вершин и ребер графа соответственно; $L = \max\limits_{i,j} |D[i, j]|$ – максимальная мощность множества эффективных путей из i-й вершины в j-ю. При выполнении цикла *for* $p \in \{D[s, u]\}$ *do* необходимо найти сумму не более чем L k-мерных точек. Следовательно, вычислительная сложность выполнения цикла составит $O(kL)$. Поскольку в прикладных задачах величина k ограничена обычно некоторой константой, то можно считать вычислительную сложность данного цикла равной $O(L)$.

В алгоритме необходимо просмотреть все предшествующие вершины для каждой вершины предполагаемого пути. Общее число таких вершин не превышает m. Наконец, внешний цикл *for* $z \in \{D[s, t]\}$ *do* повторяется не более L раз. Следовательно, вычислительную сложность алгоритма можно оценить как $O(mL^2)$. Для сравнения, вычислительная сложность первой части векторного варианта метода Флойда оценивается [7] величиной $O(n^3L^3)$, то есть, по крайней мере, на два порядка выше.

При наличии эквивалентных альтернатив описанный алгоритм 1 ищет среди них только первую, подходящую по критериальным оценкам. Поэтому в том случае, когда необходимо получить все решения, имеющие векторный вес $z \in \{D[s, t]\}$, алгоритм 1 не работает. Тогда можно предложить другой алгоритм (назовем его «Алгоритм 2»). В нем сформированные пути также накапливаются в *STACK*, а после получения всего пути – выводятся на печать.

```
Function Back;
begin h:= nil; Back := nil;
 while (h = nil) and (STACK <>∅)  do
begin g <= STACK;
       < Восстановить ПРЕДШ [g] >
        w := top(STACK);   h := top (ПРЕДШ [w]);
Back := w;
end;
end;
```

```
{Головная программа }
read (s, t);
for z ∈ {D[s, t]} do
begin v := t; STACK:= t; q := z;
while STACK <> ∅ do
begin while (v <> s) and (STACK <> ∅) do
begin Y:= True;
        while (ПРЕДШ [v] <> ∅) and Y do
begin u<= ПРЕДШ [v];
for p∈ {D[s, u]} do
begin if q = p + A[u, v] then
begin u => STACK; q := p; v := u;
Y:= False; break;
end;
end;
end;
if Y then {Не нашли подходящее u – откат назад}
v := Back;
end;

if v = s then
begin write(STACK);
v := Back;
end;
end;
end;
```

Здесь функция *Back* осуществляет откат назад, для чего производится циклическое исключение элементов стека до тех пор, пока не найдется элемент w:= *top*(*STACK*), у которого не пусто множество еще не использованных предшествующих вершин.

Оценим вычислительную сложность алгоритмов. При выполнении цикла *for* $p \in \{D[s, u]\}$ *do* необходимо найти сумму не более чем L k-мерных точек. Следовательно, вычислительная сложность выполнения цикла составит $O(kL)$. Поскольку величина k ограничена некоторой константой, то можно считать вычислительную сложность данного цикла равной $O(L)$. От Алгоритма 1 данный алгоритм отличается лишь тем, что вместо цикла *for* $u \in$ *ПРЕДШ* [v] *do* выполняется цикл *while* (*ПРЕДШ* [v] <> ∅), который, хотя и с прерываниями, просматривается весь список *ПРЕДШ* [v]. Это означает, что оценка вычислительной сложности Алгоритма 2 будет такой же, как у первого – $O(mL^2)$.

Список литературы

1. Бугаев, Ю.В. Обобщение схемы динамического программирования / Ю.В. Бугаев, С.В. Чикунов // Автоматика и телемеханика. – 2009. – № 2. – С. 90–100.

2. Гэри, М. Вычислительные машины и труднорешаемые задачи : пер. с англ. / М. Гэри, Д. Джонсон. – М. : Мир, 1982. – 416 с.

3. Юдин, Д.Б. Вычислительные методы теории принятия решений / Д.Б. Юдин. – М. : Наука, 1989. – 316 с.

4. Сухарев, А.Г. Курс методов оптимизации : учеб. пособие / А.Г. Сухарев, А.В. Тимохов, В.В. Федоров. – М. : ФИЗМАТЛИТ, 2005. – 368 с.

5. Гафт, М.Г. Выделение множества неподчиненных решений и их оценок в задачах принятия решений при векторном критерии / М.Г. Гафт, В.М. Озерной // Автоматика и телемеханика. – 1973. – № 11. – С. 85–94.

6. Липский, В. Комбинаторика для программистов : пер. с польск. / В. Липский. – М. : Мир, 1988. – 213 с.

7. Блинов, И.В. Обобщение алгоритма Флойда–Уоршалла на случай нескольких критериев / И.В. Блинов, Ю.В. Бугаев, С.В. Чикунов // Вестн. Тамб. гос. техн. ун-та. – 2009. – Т. 15, № 4. – С. 885–892.

Generalization of the Dynamic Programming Scheme by the Binary Relations of Lexicography

Yu.V. Bugaev, F.A. Musalevsky

Department "Information Technologies of Modeling and Management", Voronezh State University of Engineering Technology, Voronezh; y_bugaev52@mail.ru

Key words and phrases: binary relation; computing difficulty; dynamic programming; efficient ways; function of the choice.

Abstract: The paper offers the generalization of dynamic programming scheme in case of interpretation of the notion of optimality in the form of the mechanism of choice alternatives, non-dominated by a given binary relation. The notion of choice on lexicography of binary relations to narrow the set of non-dominated solutions.

Verallgemeinerung des Schemas der dynamischen Programmierung auf Grund der Lexikographie der binären Beziehungen

Zusammenfassung: Es wird die Verallgemeinerung des Schemas der dynamischen Programmierung im Fall der Deutung des Begriffes der Optimalität in der Form des Mechanismus der Wahl der nach der eingestellten binären Beziehung undominierenden Alternativen vorgeschlagen. Es wird den Begriff der Wahl nach der Lexikographie der binären Beziehungen für die Verengung der Menge der undominierenden Lösungen eingeführt.

La généralisation du schéma de la programmation dynamique en vertu de la lexicographie des relations binaires

Résumé: On propose la généralisation du schéma de la programmation dynamique sur le cas de l'interprétation de la notion d`optimalité en forme du mécanisme du choix des alternatives non dominées selon la relation binaire donnée. On introduit la notion du choix selon la lexicographie des relations binaires pour le rétrécissement de la multitude de décisions non dominées.

Авторы: ***Бугаев Юрий Владимирович*** – доктор физико-математических наук, профессор кафедры «Информационные технологии моделирования и управления»; ***Музалевский Федор Александрович*** – аспирант кафедры «Информационные технологии моделирования и управления», ФГБОУ ВПО «Воронежский государственный университет инженерных технологий», г. Воронеж.

Рецензент: ***Абрамов Геннадий Владимирович*** – доктор технических наук, профессор, заведующий кафедрой «Информационные технологии моделирования и управления», ФГБОУ ВПО «Воронежский государственный университет инженерных технологий», г. Воронеж.