ISSN 2223–5620 (Print), ISSN 2411–1554 (Online)
Електромагнітна сумісність та
безпека на залізничному транспорті, 2013, № 6

УДК 519.6:519.9

В. В. ЛАГУТА – к.т.н., доцент, Днепропетровский национальный университет железнодорожного транспорта имени академика В. Лазаряна, vvlaguta@mail.ru

ВЫБОР ЭФФЕКТИВНОГО РЕШЕНИЯ В ДИСКРЕТНОМ ДИНАМИЧЕСКОМ ПРОГРАММИРОВАНИИ

Статью представил д.т.н., проф. А. А. Босов

Введение

Эффективным подходом к решению задач современной теории управления является построение вычислительных процедур в форме метода динамического программирования.

Динамическое программирование (ДП) связано с принятием решений в многошаговых процессах. Впервые данный термин был введен Беллманом в его монографии [1, 2]. В этой работе на основе рассмотрения функциональных уравнений были заложены математические основы ДП.

ДП используется, когда решение задачи на каждом этапе является решением некоторой частной задачи. Решение целевой задачи формируется на основе решенных подзадач на каждом этапе.

Как известно прямое применение метода ДП для определения решения связано с большим объемом вычислений. Существует множество методов основанных на идеях динамического программирования преобразованных через дополнительную модификацию вычислительных процедур [3, 4, 5].

Идея, предложенной в статье процедуры, возникла при решении задачи векторной оптимизации тяговых расчетов [6].

В данной работе предложено обоснование возможной модификации метода ДП, позволяющего в некоторых случаях уменьшить при анализе промежуточных решений количество состояний при оптимизации многостадийного управляемого процесса. Данный подход применим для тех задач, которые допускают введение дополнительного компромиссного показателя оптимизации, исходя из естественных условий решаемой задачи.

1. Схема метода динамического программирования

Обычно задача, к которой применяется метод динамического программирования, представляется многостадийным процессом. Исследуемый процесс S разбивается на стадии, которые характеризуются N координатами S_j, $j=\overline{0,N}$. Процесс на каждой стадии S_j может находиться в нескольких состояниях s_{ij}, $i=\overline{0,M}$. На каждой стадии осуществляется выбор управления для перехода в необходимое состояние следующей стадии. Назначение перехода из состояния S_j в любое допустимое состояние S_{j+1} назовем локальным управлением u. Каждому из состояний процесса S приписывается множество $U(S)$ возможных локальных управлений, осуществляющих переходы в другие состояния. Реализация локального управления $u \in U(S)$ имеет некоторое качество $c(u)$. Требуется определить последовательность локальных управлений, переводящих систему из начального состояния S_0 в некоторое конечное состояние S_N, с наилучшим показателем качества $C(S_0)$, равную сумме составляющих качества ее локальных управлений. Такая последовательность локальных управлений называется оптимальным

управлением для осуществления перехода из состояния S_0 с показателем качества $C(S_0)$. Аналогичный смысл вкладывается и в понятие оптимального управления из любого другого состояния S_j процесса S.

В основу метода динамического программирования положен принцип оптимальности Беллмана: в рассматриваемой стадии S_j следует выбирать такое локальное управление $u \in U(S)$, которое в совокупности с оптимальным управлением из состояния S_{j+1} составит оптимальное управление из состояния S_j. Этот принцип записывается так

$$C(S_j) = \min_{u \in U(S_j)} \{c(u) + C(U(S_{j+1}))\}. \quad (1)$$

Критерий качества локального управления может быть и противоположным критерием оптимального управления, чем в выражении (1).

2. Поиск минимального пути в графе

Если множество локальных управлений конечно, то возможна интерпретация принципа оптимальности с помощью графов [7]. Определим взвешенный ориентированный граф $G = (V, E)$ (далее граф), вершины которого соответствуют состояниям системы, а дуги – всевозможным переходам из состояния в состояние; веса дуг полагаются равными оценкам качества соответствующих локальных управлений.

Оптимальным ориентированным путем из некоторой вершины $v \in V$ назовем путь минимальной длины $C(v)$ из вершины v в одну из вершин, соответствующих конечному состоянию.

Рассмотрим элементы орграфа G: вершину v стадии j и множество всех дуг (v, v_1), (v, v_2),…, (v, v_m), выходящих из v. Дуга (v, v_i) весом $c(v, v_i) = c(u^i)$ соответствует локальному управлению $u^i \in U(S)$, $i = \overline{1, m}$, переводящему систему S в некоторое состояние стадии S_{j+1}. Принцип оптимальности в терминах теории графов формулируется так: будучи в вершине v, следует выбирать ту дугу (v, v_i), которая вместе с оптимальным путем из вершины v_i,- составит оптимальный путь из вершины v

$$C(v) = \min_{i=\overline{1,m}} \{c(v, v_i) + C(v_i)\}.$$

При определении $C(v)$ необходимо знать величины $C(v_i)$, $i = \overline{1, m}$. Это возможно, если граф G допускает топологическую сортировку (является бесконтурным).

Алгоритм поиска оптимальных путей в связном бесконтурном орграфе, использующий принцип оптимальности Беллмана.

Шаг 0. Произвести топологическую сортировку вершин орграфа $G = (V, E)$, пронумеровав их числами $1, 2, \ldots, n$. Положить $step_1 = n$ и перейти к следующему шагу.

Шаг k, $(k \geq 1)$. 1. Если $step_k = 0$, то алгоритм окончен. Если $step_k \neq 0$, то выбрать вершину v с номером $step_k$. Если нет дуг с началом в вершине v, то положить

$$C(v) = 0,\ top(v) = \varnothing,\ step_{k+1} = step_k - 1$$

и перейти к следующему шагу.

2. Если имеются дуги с началом в вершине v, то выбрать среди них дугу (v, w) с минимально возможным значением

$$c(v, w) + C(w).$$

Положить

$$C(v) = c(v, w) + C(w),\ top(v) = w,$$

$$step_{k+1} = step_k - 1$$

и перейти к следующему шагу.

Временная сложность алгоритма равна [8]

$$O\left(|E| \max_{e \in E} \log_2 c(e)\right), \quad (2)$$

где $c(e)$ – вес дуги e.

3. Метод сокращения перебора в ДП

При сведении непрерывной задачи оптимального управления к ее дискретному варианту (приближению) необходимо принять решение о выборе шага сетки. Малый шаг сетки приводит к необходимости проведения большого объема вычислений. Рассматриваемый в статье вариант сокращения перебора не относится непосредственно к самому методу динамического программирования. Предлагаемая модификация, скорей всего, относится к процессу решения некоторого класса задач. В классическом дискретном случае динамического программирования управление перехода из одного состояния в другое состояние последующей стадии оценивается одной величиной C. Иногда в прикладной задаче выбор управления можно оценивать не одной величиной, а двумя $C(S,U)$ и $H(S,U)$, где S – стадия, U – множество допустимых управлений. Например, если переход из состояния в состояние характеризуется затратами энергии, то наряду с энергетическими затратами можно рассматривать и затраты времени (или другую характеристику процесса). Второй показатель выбирается исходя из естественных условий оптимизируемого процесса или имеющихся ограничений задачи (ограничения на ресурс, время и т.п.). Существенно только то, что второй (вспомогательный) показатель $H(S,U)$ должен быть компромиссным (противоречивым) по отношению к первому (основному) показателю в том смысле, что их экстремальные значения достигаются на различных траекториях.

Под компромиссными решениями обычно понимают полуэффективные и эффективные решения [9]. Траекторию σ называют полуэффективной, если, выбирая другую траекторию, нельзя получить значение векторного критерия лучше полуэффективного одновременно по всем компонентам, по которым оптимизируют критерий. Требования к эффективным траекториям более однозначны. Среди эффективных траекторий не должно быть траекторий не худших по всем компонентам векторного критерия и лучшей траектории хотя бы для одной компоненты. Эффективные (оптимальные по Парето) решения таковы, что при переходе от одного эффективного решения к другому некоторые компоненты векторного критерия строго убывают, но другие при этом строго возрастают.

Ускорение вычислительного процесса возможно, если в каждой стадии будем исключать из рассмотрения некоторые состояния, которые не ведут к успеху, руководствуясь «разумной стратегией». В качестве способа исключения состояний оптимизируемого процесса взята процедура фильтрации с помощью принципа Парето определения эффективных траекторий, но оптимизируя требуемый показатель. Этот способ позволяет не рассматривать в процессе оптимизации в дискретном фазовом пространстве состояния, принадлежащие неэффективным траекториям.

Вместе с показателем $C(S)$, рассматриваемой задачи, качество траектории будем оценивать также вспомогательным показателем $H(S)$. Для определенности полагаем, что показатель $H(S)$ минимизируется для достижения компромисса по отношению к показателю $C(S)$.

Для каждой стадии S_k, $k=1,2,...,N$, N - количество стадий, будем рассматривать множество пар $(C(S_k), H(S_k))$, для которых определим бинарное отношение частичного порядка

$$R_k : (C'(S_k), H'(S_k)) \succ (C''(S_k), H''(S_k)) \Leftrightarrow C'(S_k) \le C''(S_k), \quad H'(S_k) \le H''(S_k). \qquad (3)$$

Значок $\succ$ означает, что пара слева «лучше» пары справа.

Оценка управления $C(S)$ аддитивная (или мультипликативная) функция

$$C(S)=\sum_{j=1}^{N} c_j\left(S_j,u_j\right),\ u_j\in U\left(S_j\right),$$

где $U\left(S_j\right)$ - множество допустимых управлений стадии j, $c_j\left(S_j,u_j\right)$ – оценка эффективности применения управления на стадии j. Требование аддитивности вспомогательного показателя

$$H(S)=\sum_{j=1}^{N} h_j\left(S_j,u_j\right),\ u_j\in U\left(S_j\right)$$

необязательно. Это, к примеру, может быть значение фазовой переменной (координаты). Мы будем рассматривать функцию $H(S)$ как аддитивную.

Обозначим $C_k\left(S_k\right)$ оптимальное значение задачи

$$C_k\left(S_k\right)=\min_{u\in U(S)}\sum_{j=1}^{k} c_j\left(S_j,u\right), \qquad (4)$$

а через $H_k\left(S_k\right)$ соответствующее значение вспомогательного показателя, обусловленное выбранному управлению u в выражении (4).

Расчет величины $C_k\left(S_k\right)$ в вычислительной процедуре производится по рекуррентному соотношению

$$C_k\left(S_k\right)=$$
$$\min_{u_k\in U(S_k)}\left\{C_{k-1}\left(\varphi\left(S_{k-1},u_k\right)\right)+c_k\left(S_k,u_k\right)\right\}, \qquad (5)$$

$\varphi\left(S_{k-1},u_k\right)$ – функция изменения состояния на стадии $k-1$ под воздействием управления u_k. Необходимо также вычислить значение $H_k\left(S_k\right)$ для определенного в (5) локального управления u_k^*

$$H_k\left(S_k\right)=H_{k-1}\left(\varphi\left(S_{k-1},u_k^*\right)\right)+h_k\left(S_k,u_k^*\right).$$

Вначале процедуры оптимизации полагаем $C_0\left(S_0\right)=0$ и $H_0\left(S_0\right)=0$. Если на какой-то стадии k минимум $C_k\left(S_k\right)$ не достигается, то полагаем $C_k\left(S_k\right)=\infty$.

Обозначим

$$P_k=\left\{\left(C_k^i,H_k^i\right), i=\overline{1,M_k}\right\},$$

множество паретовских точек стадии S_k. Множество P_k, $k=\overline{1,N}$ определяются на каждой стадии k.

Опишем общий алгоритм. Пусть определено множество

$$P_{k-1}=\left\{\left(C_{k-1}^i,H_{k-1}^i\right), i=\overline{1,M_{k-1}}\right\}.$$

Обозначим

$$C_k^j=\min_{u_k\in U(S_k)}\left\{C_{k-1}\left(\varphi\left(P_{k-1},u_k\right)\right)+c_k\left(S_k,u_k\right)\right\},\ j=\overline{1,m_k}, \qquad (6)$$

$$H_k^j=H_{k-1}\left(\varphi\left(P_{k-1},u_k^*\right)\right)+h_k\left(S_k,u_k^*\right),\ j=\overline{1,m_k}, \qquad (7)$$

где j – индекс дискретного состояния процесса на k-й стадии, m_k – количество возможных дискретных состояний стадии k.

Строим множество Парето для точек (6)-(7) в соответствии с введенным бинарным отношением R_k, (3)

$$P_k=\left\{\left(C_k^i,H_k^i\right), i=\overline{1,M_k}\right\}.$$

Просмотрев все стадии, в конечной точке сформируется множество, состоящее из одной пары, в случае одноточечного краевого условия

$$P_N=\left\{\left(C_N^1,H_N^1\right)\right\}.$$

Оптимальным значением будет C_N^1.

Рассмотренный порядок построения оптимального решения соответствует схеме динамического программирования «вперед», т.е. процесс оптимизации происходит из начального состояния в конечное состояние. В этом алгоритме рассматриваются лишь траектории, проходящие через паретовские точки, другие траектории не рассматриваются.

Аналогично можно построить процедуру оптимизации по схеме «назад».

Как в схеме «вперед», так и в схеме «назад» на каждой стадии сформировано управление. По «обратному ходу» определяем соответствующие управления поочередно в требуемом направлении для каждой последующей стадии. Восстановив всю цепочку оптимальных управлений, получим необходимое решение.

В дискретном случае возможна интерпретация рассмотренной вычислительной процедуры динамического программирования с помощью графа. Разбив процесс на N стадий, для каждой стадии k определим множество вершин $V_k = \{v_{ik}\}$, $k = \overline{0, N}$, $i = \overline{1, m_k}$, вершина v_{ij} соответствует некоторому состоянию оптимизируемого процесса, m_k - количество состояний процесса для стадии k, обусловленное выбором сетки расчета и существующими ограничениями, включая ограничения на допустимые управления. Допустимому переходу из одного состояния в другое состояние последующей стадии соответствует дуга с векторным весом $\left[c\left(v_{jk-1}, v_{ik}\right), h\left(v_{jk-1}, v_{ik}\right)\right]$, здесь v_{jk-1} – начальная вершина дуги, v_{ik} – конечная вершина дуги, $c\left(v_{jk-1}, v_{ik}\right)$ – эффект от реализации управления при переходе из начальной вершины в конечную вершину, $h\left(v_{jk-1}, v_{ik}\right)$ – введенный вспомогательный показатель эффективности реализации выбранного управления. Все значения $c\left(v_{jk-1}, v_{ik}\right)$ и $h\left(v_{jk-1}, v_{ik}\right)$ считаем ограниченными величинами.

Пусть $C\left(v_{ik}\right)$ эффект управления на стадии k соответствующий траектории проходящей через вершину v_{ik} (условно оптимальный выигрыш), $H\left(v_{ik}\right)$ – значение вспомогательного показателя, соответствующего условно оптимальному выигрышу. В процессе оптимизации каждой вершине ставится в соответствие пара $\left(C\left(v_{ik}\right), H\left(v_{ik}\right)\right)$. Определим множества Парето P_k, $k = \overline{1, N}$, для пар $\left(C\left(v_{ik}\right), H\left(v_{ik}\right)\right)$ каждой стадии $P_k = \left\{\left(C\left(v_{ik}\right), H\left(v_{ik}\right)\right), i = \overline{1, M_k}\right\}$. Вместе с множеством P_k будем строить множество $\tilde{V}_k = \left\{\tilde{v}_{ik}, i = \overline{1, M_k}\right\}$ состоящее из вершин стадии k, образовавших пары, входящих в множество P_k.

Вначале алгоритма полагаем $C\left(v_{10}\right) = 0$ и $H\left(v_{10}\right) = 0$, $P_0 = \left\{(0,0)\right\}$ и $\tilde{V}_0 = \left\{\tilde{v}_{10}\right\}$.

На каждой стадии $k = \overline{1, N}$ выполняем следующие действия

$$C\left(v_{ik}\right) = \min_{v \in \tilde{V}_{k-1}} \left\{C(v) + c\left(v_{ik}\right)\right\}, \qquad (8)$$

для индекса j^*, определившего условно оптимальный выигрыш $C\left(v_{ik}\right)$ в (8) определяем значение вспомогательного показателя

$$H\left(v_{ik}\right) = H\left(v_{j^* k-1}\right) + h\left(v_{ik}\right). \qquad (9)$$

Вычисления величин (8) и (9) проводятся для каждой вершины $i = \overline{1, m_k}$ рассматриваемой стадии k.

Для вновь полученного множества пар $\left(C\left(v_{ik}\right), H\left(v_{ik}\right)\right)$, $i = \overline{1, m_k}$ строим множество Парето $P_k = \left\{\left(C\left(v_{ik}\right), H\left(v_{ik}\right)\right), i = \overline{1, M_k}\right\}$ в соответствии с бинарным отношением (3) и строим множество $\tilde{V}_k = \left\{\tilde{v}_{ik}, i = \overline{1, M_k}\right\}$ – множество вершин, образовавших паретовские точки P_k.

Минимальному значению пути будет соответствовать величина $C_N^1 \in P_N$. Двигаясь со стадии N в обратном следовании, получим необходимую оптимальную траекторию, точки которой принадлежат множеству Парето.

Временная сложность предложенного алгоритма оценивается величиной

$$O\left(\left|E_P\right| \max_{\substack{v\in P_k, w\in P_{k+1} \\ k=0,N-1}} \left\{\log_2 c(v,w), \log_2 h(v,w)\right\}\right),$$

где $(c(v,w),h(v,w))$ – векторный вес дуги, имеющей начало в вершине принадлежащей множеству Парето; E_P – множество дуг, начало которых начинается в вершине из множества Парето.

Выводы

Предложенная модификация метода ДП в дискретном варианте для класса задач допускающих введение дополнительного показателя эффективности выбора управления позволяет уменьшить в переборе количество состояний системы. Модификация базируется на введении соответствующего бинарного отношения и построения множества Парето для пар «критерий» – «дополнительный показатель», на котором осуществляется процедура оптимизации.

Библиографический список

1. Bellman, R. E. Dynamic Programming / R. E. Bellman. Reprint edition. Dover Publications, INC. N.Y. –2003. –366 p.
2. Bellman, R. E. Applied dynamic programming / R. E. Bellman, S. E. Dreyfus // Princeton University Press. –1962. –388 p.
3. Григорьев, А. М. Динамическое программирование в обобщенной задаче курьера с внутренними работами: элементы параллельной структуры [Текст] / А. М. Григорьев, Е. Е. Иванко, А. Г. Ченцов // Моделирование и анализ информационных систем. –Ярославль. –2011. –Том 18. – №3. –С.101-124.
4. Ченцов, А. Г. Динамическое программирование в одной нестационарной задаче маршрутизации [Текст] / А. Г. Ченцов, П. А. Ченцов // Изв. Института математики и информатики УдмГУ. –Ижевск, –2012. –Выпуск 1(39). – С. 151-154.
5. Струченков, В. И. Новые алгоритмы оптимального распределения ресурса [Текст] / В. И. Струченков // Прикладная дискретная математика. –М. –2010. –№4(10). –С.73-78.
6. Лагута, В. В. Тягові розрахунки на множені Парето за двома показниками [Текст] / В. В. Лагута // Вісник Дніпропетровського національного університету залізничного транспорту імені акад. В. Лазаряна. –Д. № 38. –С. 194-200.
7. Bellman, R. E. Algorithms, graphs, and computers / R. E. Bellman, K. L. Cook, J. A. Lockett // Volume 62. Academic Press. –New York and London. –1970. – 246 p.
8. Кузюрин, Н. Н. Эффективные алгоритмы и сложность вычислений [Текст] / Н. Н. Кузюрин, С. А. Фомин. –М.: МФТИ. –2007. –210с.
9. Подиновский, В. В. Парето-оптимальные решения многокритериальных задач [Текст] / В. В. Подиновский, В. Д. Ногин. – М.: Наука. –1982. – 254 с.

Ключові слова: векторна оптимізація, дискретні системи, динамічне програмування.

Ключевые слова: векторная оптимизация, дискретные системы, динамическое программирование.

Keywords: vector optimization, discrete systems, dynamic programming.

Надійшла до редколегії 21.10.2013