ЭКОНОМИКА 1968
И МАТЕМАТИЧЕСКИЕ МЕТОДЫ том IV, вып. 1

НАУЧНЫЕ КОНСУЛЬТАЦИИ

ДИНАМИЧЕСКОЕ ПРОГРАММИРОВАНИЕ

М. М. БЕРКОВИЧ

(Москва)

Динамическое программирование является сравнительно новым разделом той отрасли математики, которая занимается анализом и разработкой численных методов решения различных экстремальных задач. Оно не выделяет среди всех задач математического программирования задач, обладающих определенной математической записью, как, например, задачи линейного программирования, а дает новый своеобразный подход к исследованию экстремальных задач. Конечно, это не означает, что здесь нет общих математических формулировок, но они являются, как правило, следствием самого подхода к исследуемой проблеме.

Впервые термин «динамическое программирование» появился в работах Р. Беллмана в середине 50-х годов. С этого времени идеи динамического программирования были значительно расширены и углублены: появилась общая математическая постановка рассматриваемого класса задач и были разработаны весьма эффективные методы решения. В настоящее время эти методы находят все большее применение при решении конкретных экстремальных задач, так как они дают эффективный подход к анализу внутренней структуры исследуемых процессов.

Цель настоящей консультации — изложить различные постановки задач динамического программирования, указать основные характерные черты, объединяющие эти задачи, и на основании их описать общий подход к задачам с точки зрения динамического программирования. Кроме того, здесь будет дана общая математическая формулировка задач динамического программирования и описаны простейшие методы решения.

ПРИМЕРЫ ЗАДАЧ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

Рассмотрим две простые задачи динамического программирования. На практике приходится встречаться с более сложными задачами, но, как будет показано ниже, большинство из этих дополнительных сложностей не имеет принципиального значения при анализе и решении этих задач.

Задача о планировании производства. Пусть нужно составить план работы предприятия на период времени, который состоит из n хозяйственных лет. Это предприятие производит некоторую продукцию (например, станки), реализуемую в конце каждого года. Результатом этой реализации является прибыль предприятия. Определенная ее часть идет на расширение предприятия. Кроме того, на эти же цели в начале каждого года направляются некоторые дополнительные средства (например, из фонда государственных капиталовложений), размер которых заранее не определен. Требуется в начале каждого года так определять размеры этих дополнительных средств, идущих на расширение предприятия, чтобы: 1) к концу планируемого периода предприятие выполнило план выпуска

продукции, который задан заранее; 2) общая сумма дополнительных капиталовложений за весь планируемый период была как можно меньше.

Будем считать, что основной и единственной характеристикой рассматриваемого предприятия является его мощность, т. е. способность выпускать определенное количество продукции. Будем также считать, что средства, выделенные на развитие предприятия (на увеличение его мощности), реализуются в течение одного года. При этом условии количество продукции, выпущенной в i-м году, однозначно определяется мощностью предприятия в том же году. Отсюда также видно, что сама мощность предприятия к началу года полностью определяется его мощностью в предыдущем году и величиной дополнительных средств, выделенных для его развития, так как прибыль предприятия однозначно определяется его мощностью. Обозначим через x_i мощность предприятия к концу i-го года, а через u_i — величину дополнительных средств, выделенных в i-м году. Тогда все сказанное выше можно записать следующим образом: $x_{i+1} = f(x_i, u_i)$.

Заметим, что дополнительные средства u_i не обязательно выражаются в денежных единицах.

Из постановки задачи видно, что планирование роста предприятия на весь планируемый период распадается на ряд этапов. На каждом этапе, т. е. в начале каждого года, нужно определить дополнительные средства, которые идут на увеличение мощности предприятия.

То, что планирование развития предприятия за весь период времени состоит из ряда этапов, является одной из самых характерных черт задач динамического программирования. Поэтому эти задачи часто называются задачами многоходового выбора.

Что нужно учитывать в первую очередь при определении оптимальных размеров дополнительных средств, выделяемых на развитие предприятия в начале каждого года? Совершенно очевидно, что оптимальный план на весь планируемый период полностью зависит от состояния предприятия в исходный момент времени, а не в предшествующие годы. Поскольку здесь рассматривается упрощенная модель предприятия, то состояние его в любой момент времени характеризуется лишь его мощностью.

То, что оптимальный план зависит не от предыстории, не от того, как предприятие достигло исходного состояния, а только от состояния предприятия в исходный момент времени *, является второй характерной особенностью задач динамического программирования.

Свойство независимости оптимального плана от предыстории в задачах динамического программирования является основным. Именно это свойство указывает на то, что данную задачу можно рассматривать с точки зрения динамического программирования, и именно на его основе строится большинство численных методов решением подобных задач.

Третья особенность рассматриваемой задачи состоит в ее аддитивности. Это означает, что целевая функция в рассматриваемой задаче имеет вид:

$$F(u_0, \ldots, u_{n-1}) = F(u) = \sum_{0}^{n-1} \varphi(u_i),$$

где u_i — размер дополнительных средств, выделенных в начале i-го года, а $\varphi(u_i)$ — стоимость этих средств.

Это свойство не является столь существенным, как первые два, однако оно присуще большинству задач динамического программирования и зна-

* Это свойство не следует смешивать с тем, что при составлении планов необходимо учитывать развитие предприятия до планируемого периода для выявления и уточнения различных характеристик этого предприятия.

чительно упрощает вычислительный процесс решения. Существуют задачи динамического программирования, в которых свойство аддитивности не выполняется. К таким задачам относятся, например, задачи о максимизации прибыли в конце планируемого периода и др.

Приведем теперь математическую постановку данной задачи. Пусть $x(0)$ — состояние предприятия (его мощность) в начале планируемого периода, а $x(n)$ — в конце планируемого периода. Так как в конце планируемого периода предприятие должно выполнить заданный план выпуска продукции, то можно считать, что $x(n)$ также известна заранее.

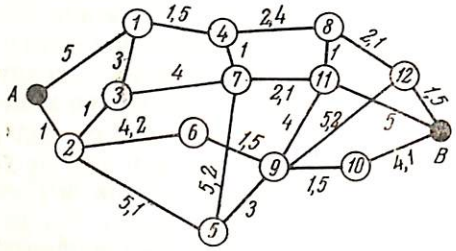

Рис. 1

Величины дополнительных средств, вкладываемых в предприятие, очевидно, не могут быть совершенно произвольными, а изменяются в некоторых пределах, т. е. u_i может принимать значения только из некоторого ограниченного множества, которое обозначим U_i $(i = 0, 1, \ldots, n-1)$.

Теперь задачу можно записать в следующей форме: минимизировать

$$\left.\begin{array}{c} F(u) = \sum_{0}^{n-1} \varphi(u_i) \\ x_{i+1} = f(x_i, u_i), \\ x_0 = x_{(0)};\ x_n = x(n), \\ u_i \in U_i;\ i = 0, 1, \ldots, n-1 \end{array}\right\} \quad \text{(I)}$$

при условии

Итак, это типичная задача динамического программирования. Хотя рассматривалась только грубая схематичная модель, но нетрудно видеть, что при конкретизации этой модели названные выше основные свойства сохраняются. Следует также заметить, что структура приведенных функциональных зависимостей и допустимых областей может быть совершенно произвольной; это никак не отразится на основных свойствах задач.

Задача о кратчайшем пути. Рассмотрим еще одну задачу из другой области — задачу о нахождении кратчайшего пути.

Пусть нужно добраться на машине из пункта A в пункт B. Имеется ряд возможных вариантов путей. Они составлены из участков дорог, различных по длине (качество дорог принимать во внимание не будем). Общая схема дорог изображена на рис. 1.

Кружочками отмечены промежуточные пункты, которые для удобства занумерованы. Цифры около отрезков прямых указывают длины соответствующих дорог. Задача состоит в том, чтобы выбрать кратчайший путь следования машины из A в B.

Эту задачу можно рассматривать с разных позиций. Рассмотрим ее с точки зрения динамического программирования. Для этого разобьем все пункты на группы. К первой группе отнесем пункт A; ко второй — те пункты, в которые можно попасть непосредственно из пункта A (эта группа состоит из пунктов с номерами 1; 2); к третьей — те пункты, в которые можно попасть непосредственно из любого пункта второй группы (к этой группе не относятся пункты, входящие в первую и вторую группу; третья группа состоит из пунктов 3; 4; 5; 6).

Аналогично образуется четвертая группа из пунктов 7, 8, 9; пятая группа — из пунктов 10, 11, 12 и шестая группа, в которую входит

пункт B. Можно с достаточной степенью точности считать, что движение машины из A и B состоит из нескольких этапов. На первом этапе движения машина перемещается из A в какой-то пункт группы 2, на втором — из пункта группы 2 в пункт группы 3 и т. д. Таким образом, весь процесс движения автомобиля является многоэтапным процессом движения. Кроме того, процесс нахождения кратчайшего пути следования тоже распадается на этапы. На каждом этапе требуется так выбирать путь следования машины в соседнюю группу пунктов, чтобы весь путь следования был наименьшим. Отсюда следует, что рассматриваемая задача является задачей многоходового выбора.

Свойство аддитивности для задачи вытекает из того, что длина пути, состоящего из нескольких отрезков дорог, равна сумме длин этих дорог. И, наконец, нетрудно видеть, что кратчайший путь следования машины из любого пункта в пункт B зависит не от того, как машина попала в этот пункт, а только от расположения этого пункта в общей схеме дорог. Т. е. для этой задачи выполняется основное свойство задач динамического программирования: свойство независимости оптимального поведения (в данном случае оптимального движения автомобиля) от предыстории. Из этого свойства, в частности, вытекает следующее: предположим, что оптимальный путь из A в B проходит через пункты 2, 6, 9, 10, тогда кратчайший путь следования машины из пункта 2 в пункт B будет проходить через пункты 6, 9, 10, т. е. будет совпадать с оптимальной траекторией всей задачи на участке от пункта 2.

Чтобы подчеркнуть, какую большую роль играет это простое, но важное свойство, опишем вкратце метод решения рассматриваемой задачи, который целиком на нем основан. Этот метод состоит в следующем. Предположим, что мы уже нашли кратчайшие пути l_1 и l_2 из пунктов 1 и 2 до пункта B; тогда, чтобы найти кратчайший путь из A в B, достаточно рассмотреть всего две траектории: первая траектория L_1 проходит через пункт 1, а дальше совпадает с траекторией l_1, соединяющей этот пункт с пунктом B, вторая траектория L_2 аналогична первой, но проходит через пункт 2.

Для выбора кратчайшего пути из A в B нужно сравнить длины полученных двух путей и выбрать из них самый короткий, который дает решение исходной задачи.

Докажем этот факт. Пусть L — оптимальный путь следования из A в B. Покажем, что он совпадает с одной из двух описанных выше траекторий. Так как любая траектория, соединяющая A с B, проходит либо через пункт 1, либо через пункт 2, то предположим, что L проходит через пункт 1, тогда из основного свойства оптимальной траектории вытекает, что L совпадает с l_1 на участке от пункта 1 до пункта B и, следовательно, L совпадает с L_1. Если же L проходит через пункт 2, то аналогичными рассуждениями можно показать, что он будет совпадать с траекторией L_2. Таким образом, чтобы найти траекторию L, достаточно найти траектории l_1 и l_2, т. е. найти кратчайшие пути из всех пунктов группы 2 в пункт B. Нетрудно показать теми же рассуждениями, что для нахождения траекторий l_1 и l_2 достаточно найти кратчайшие пути из каждого пункта группы 3, т. е. из пунктов 3, 4, 5, 6 в пункт B.

Применяя последовательно аналогичные рассуждения, мы приходим к следующему алгоритму для нахождения кратчайшего пути из A в B. Сначала находим кратчайшие пути l_{10}, l_{11}, l_{12} из каждого пункта группы 5 в пункт B, затем, используя полученные результаты, находим кратчайшие пути l_7, l_8, l_9 из каждого пункта группы 4 в пункт B и т. д. до тех пор, пока не найдем кратчайшего пути L из группы 1, которая состоит из одного пункта A. Этот путь будет решением задачи.

Проиллюстрируем этот метод на численном примере. Найдем кратчайший путь между A и B (рис. 1). Длины всех дорог указаны на схеме. Весь алгоритм состоит из 5 шагов:

1 шаг. Вычисляются кратчайшие пути из пунктов 10, 11, 12 группы 5 до пункта B. Они, очевидно, равны: $l_{10} = 4{,}1$; $l_{11} = 5$; $l_{12} = 1{,}5$ (l_j — кратчайший путь из пункта j в пункт B).

2 шаг. Вычисляются кратчайшие пути из пунктов 7, 8, 9 группы 4 до пункта B. Вычислим, например, l_9. Так как из пункта 9 можно попасть в пункт B через пункты 10, 11, 12, то длина кратчайшего пути равна наименьшему из трех чисел: $l_{10} + 1{,}5$; $l_{11} + 4$; $l_{12} + 5{,}2$, т. е. $l_9 = \min\{l_{10} + 1{,}5;\ l_{11} + 4;\ l_{12} + 5{,}2\} = 5{,}6$. Кратчайший путь из пункта 9 будет проходить через пункт 10, поэтому будем писать: $l_9 = l_9(10)$.

Аналогично получаем $l_7 = \min\{l_{11} + 2{,}1\} = 7{,}1 = l_7(11)$, $l_8 = \min\{l_{12} + 2{,}1;\ l_{11} + 1\} = 3{,}6 = l_8(12)$.

3 шаг. Вычисляются l_3; l_4; l_5; l_6. $l_3 = \min\{l_7 + 4\} = 11{,}1 = l_3(7)$; $l_4 = \min\{l_7 + 1;\ l_8 + 2{,}4\} = 6 = l_4(8)$; $l_5 = \min\{l_7 + 5{,}2;\ l_9 + 3\} = 8{,}6 = l_5(9)$; $l_6 = \min\{l_9 + 1{,}5\} = 7{,}1 = l_6(9)$.

4 шаг. Вычисляются l_1 и l_2. $l_1 = \min\{l_3 + 3;\ l_4 + 1{,}5\} = 7{,}5 = l_1(4)$; $l_2 = \min\{l_3 + 1;\ l_5 + 5;\ l_6 + 4{,}2\} = 11{,}3 = l_2(6)$.

5 шаг. Вычисляется кратчайший путь из A в B. $l_A = \min\{l_5 + 5;\ l_2 + 1\} = 12{,}3 = l_A(2)$.

Таким образом, кратчайший путь из A в B равен 12,3. Этот путь проходит через пункт 2, так как $l_A = l_A(2)$. Вспоминая, что кратчайший путь из пункта 2 в пункт B проходит через пункт 6 и т. д., получаем кратчайший путь из A в B, который проходит через пункты 2, 6, 10. На этом решение задачи заканчивается. Этот метод дает не только искомую оптимальную траекторию, но и всю структуру оптимальных поведений относительно пункта B для рассматриваемой сети дорог. Например, кратчайший путь из пункта 5 до пункта B равен $l_5(9) = 8{,}6$, и этот путь проходит через пункты *9* и *10*. Этот факт для практических нужд часто более ценен, чем нахождение только одной оптимальной траектории.

О применении этого метода к общей задаче динамического программирования, о его достоинствах и недостатках будет сказано в последнем разделе этой статьи.

Приведем математическую формулировку этой задачи. Для этого введем необходимые обозначения: X_j — множество всех пунктов группы j. Например, для $j = 2$ X_2 состоит из двух пунктов — 1 и 2. U_j — множество всех путей, ведущих из группы j в группу $j + 1$. Для $j = 2$ U_2 состоит из путей, соединяющих пункты: (1,3); (1,4); (2,3); (2,5); (2,6). $\varphi(x_j, x_{j+1})$ — длина пути между пунктами $x_j \in X_j$ и $x_{j+1} \in X_{j+1}$. Эта функция может задаваться либо графически, либо при помощи таблиц.

Предположим, что из некоторого пункта $x_j \in X_j$ проходит дорога $u_j \in U_j$, которая ведет в пункт $x_{j+1} \in X_{j+1}$, тогда будем писать: $x_{j+1} = f(x_j, u_j)$.

Так как из каждого пункта ведет несколько дорог, то, подставляя в f различные u_j, будем получать различные x_{j+1}. Заметим, что функция f задана при помощи схемы, изображенной на рис. 1. Рассмотрим последовательность $u = (u_1, \ldots, u_5)$, где $u_j \in U_j$. Назовем такую последовательность допустимой, если ей сопоставляется последовательность пунктов $x = (x_1, \ldots, x_6)$, где $x_j \in X_j$, такая, что $x_{j+1} = f(x_j, u_j)$ для $j = 1, \ldots, 5$. Ясно, что допустимая последовательность U_2 определяется не только множествами U_j, но и множествами X_j.

Задача состоит в нахождении допустимой последовательности $u = (u_1^*, \ldots, u_5^*)$, минимизирующей функцию

$$\left.\begin{array}{c} F(u) = \sum_{1}^{5} \varphi(x_j, x_{j+1}) \\ x_{j+1} = f(x_j, u_j); \\ x_j \in X_j;\ u_j \in U_j; \\ x_1 = A;\ x_6 = B; \\ j = 1, \ldots, 5. \end{array}\right\} \quad \text{(II)}$$

при условии

ОБЩАЯ ПОСТАНОВКА ЗАДАЧИ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

В предыдущем разделе были рассмотрены две задачи из различных областей экономики. Эти задачи на первый взгляд весьма далеки одна от другой. Однако, рассмотрев их с позиций динамического программирования, мы обнаружили много общего в их структуре: и та и другая задачи могут быть представлены как задачи многоходового выбора; оптимальные траектории и той и другой задачи обладают очень важными свойствами «независимости»; кроме того, эти задачи являются аддитивными. Единство этих задач нашло свое выражение в их математических моделях, которые являются почти идентичными. Хотя математическая модель для задачи о кратчайшем пути выглядит несколько искусственной, но именно благодаря этой математической модели можно легко перенести метод решения, описанный в специфических терминах, на общую задачу динамического программирования, а следовательно, и на задачу о «планировании производства». Вообще единая терминология и общая постановка позволяют исследовать весь класс задач целиком и применять методы, разработанные для конкретных задач этого класса, к другим задачам этого же класса. Поэтому любую задачу желательно попытаться сформулировать в общих терминах, даже если формулировка будет слишком громоздкой и неудобной. При этом, конечно, нельзя забывать, что существует постановка, которая является наиболее приемлемой для рассматриваемой задачи.

Приступим к описанию общей задачи динамического программирования. Рассмотрим некоторую, развивающуюся систему в дискретные моменты времени $t = 0, \ldots, T$. Состояние этой системы в каждый момент времени можно характеризовать вектором $x(t) = (x_t^1, \ldots, x_t^n)$. Этот вектор будем называть вектором состояния системы. Обозначим через X_m множество всех состояний, в которых может находиться система в момент времени $t = m$. Состояние системы в начальный момент $t = 0$ считается заданным $x(0) = x_0$. Процесс развития системы состоит в последовательном переходе из одного состояния в другое. В каждый момент t на это развитие можно воздействовать при помощи системы управлений. Если система находится в состоянии $x(t)$, то состояние ее в следующий момент времени $x(t+1)$ определяется не только вектором $x(t)$, но и соответствующим управлением. Запишем это следующим образом: $x(t+1) = f(x(t), u(t))$, где $u(t)$ — выбранное управление. Функция f задает правило перехода от состояния $x(t)$ в состояние $x(t+1)$, если выбрано управление $u(t)$. Ясно, что управление в каждый момент не может быть совершенно произвольным. Обозначим через U_m множество всевозможных управлений, которые можно выбирать в момент $t = m$. Развитие системы в течение всего периода можно полностью определить последовательностью $x = (x(0), \ldots, x(T))$, где $x(m) \in X_m$ — вектор состояния системы при $t = m$.

Эта последовательность указывает на то, что система последовательно переходит от состояния $x(m)$ к состоянию $x(m+1)$. Такие последова-

тельности называются стратегиями. Так как существуют такие состояния $x(m) \in X_m$ и $x(m+1) \in X_{m+1}$, что нет управления переводящего систему из одного состояния в другое, то естественно ввести понятие допустимого множества стратегий. Это множество состоит из стратегий, для которых существуют управления, такие, что $x(m+1) = f(x(m)u_m)$ для любого $m = 0, \ldots, T-1$. Для полного описания процесса каждой стратегии x нужно сопоставить некоторую оценку $F(x)$: функцию цели. Таким образом, процесс движения описывается заданием допустимых множеств состояний X_m, допустимых множеств управлений u_m, правилами перехода от одного состояния в другое согласно выбранному управлению и функцией цели.

Задача состоит в нахождении допустимой стратегии, обеспечивающей экстремум (для определенности минимум) функции цели. Функция цели может быть задана в двух формах: либо непосредственно на множестве заключительных состояний X_T (например, когда определяется максимум дохода в конце периода), либо в виде суммы оценочных функций $Q_m(x(m); x(m+1))$, получаемых при каждом ходе из $x(m)$ в $x(m+1)$:

$$F(x) = \sum_{0}^{T-1} \theta_m(x(m); x(m+1)).$$

Первая форма целевой функции соответствует неаддитивной задаче динамического программирования, а вторая — аддитивной.

Первый вид целевой функции можно рассматривать как частный случай второго вида, когда $\theta_m \equiv 0$ при $m \neq T$. Поэтому можно рассматривать только второй вид целевой функции. Так как любая допустимая стратегия x полностью определяется последовательностью допустимых управлений $u = (u_0, \ldots, u_{T-1})$, то целевую функцию будем считать функцией от управления. Таким образом, задача динамического программирования состоит в нахождении последовательности управлений $u^* = (u_0^*, \ldots, u_{T-1}^*)$, минимизирующей функцию

$$\left.\begin{array}{r} F(x(u)) = \sum_{m=0}^{T-1} \theta_m(x(m); x(m+1)); \\ \text{при условии} \qquad x(m+1) = f(x(m), u_m) \\ x(m) \in X_m;\ x(0) = x_0; \\ u_m \in U_m;\ m = 0, 1, \ldots, T-1 \end{array}\right\} \qquad \text{(III)}$$

Покажем, что рассмотренные здесь задачи могут быть представлены формулой (III).

Математическая формулировка задачи (I) отличается от общей формулировки (III) только тем, что оценочная функция в ней задана на множестве управлений.

Положим:

$$\theta_j(x(j); x(j+1)) = \min_{uj} \varphi(u_j), \quad x_{j+1} = f(x_j u_j)$$

и

$$F^1(u) = F^1(x(u)) = \sum_{j=0}^{n-1} \theta_j(x(j); x(j+1)).$$

Тогда оптимальная стратегия в задаче (I) с функцией цели F будет совпадать с оптимальной стратегией той же задачи, но с функцией цели F^1.

Следовательно, математическая модель задачи (I) может быть представлена в форме (III). Математическая модель задачи (II) полностью совпадает с математической моделью общей задачи. Из постановки общей задачи динамического программирования видно, что она является задачей многоходового выбора, так как нахождение оптимального управления разбивается на ряд этапов, на каждом из которых выбирается некоторое допустимое управление.

Легко проверить для этой задачи выполнение свойства независимости оптимального поведения от предыстории. Оно часто называется принципом оптимальности Беллмана или просто принципом оптимальности. Приведем формулировку этого принципа, данную Беллманом в [1].

Оптимальное поведение системы (u^*x^*): $u^* = u_0^*, \ldots, u_{T-1}^*$; $x^* = x^*(0), \ldots, x(T)$ обладает тем свойством, что, каково бы ни было состояние системы $x(0)$ в начальный момент времени $t = 0$ и управление в начальный момент времени, которое переводит систему из $x(0)$ в $x^*(1)$, т. е. $x(0) \xrightarrow{u_0^*} x^*(1)$, оптимальное поведение V^*y^* относительно состояния $x^*(1)$ будет равно: $V^* = (u_1^*, \ldots, u_{T-1}^*)$; $y^* = (x^*(1), \ldots, x^*(T))$. Другими словами, оптимальное поведение системы относительно состояния $x(i)$ зависит не от того, как мы попали в это состояние, а только от самого состояния.

Важнейшей особенностью задач динамического программирования является возможность эффективного использования функциональных уравнений, которые являются точной записью принципа оптимальности Беллмана. Выведем эти уравнения. Рассмотрим систему в момент $t = 1$. Предположим, что для каждого состояния $x(1) \in X_1$ известна последовательность управлений $V^*(x(1)) = (u_1^*, \ldots, u_{T-1}^*)$. Эта последовательность обращает в минимум функционал $\sum_1^{T-1} \theta_m$ при выполнении условий задачи (III). Поскольку минимальное значение определяется только вектором $x(1)$, то обозначим это значение $F_1(x(1))$. Из принципа оптимальности следует: для нахождения оптимальной последовательности управлений задачи (III) достаточно рассмотреть только последовательности вида: $u = (u_0, V^*(x(1)))$ для всех $x(1) \in X_1$ и выбрать из них оптимальную. Здесь u_0 — допустимое управление, переводящее систему из состояния $x(0)$ в состояние $x(1)$. Таким образом, если известны $V^*(x(1))$ и $F_1(x(1))$ для всех $x(1) \in X_1$, то задача сводится к выбору соответствующего управления в начальный момент времени. Значение функционала, соответствующее последовательности управлений $u_0, V^*(x(1))$, равно $[\theta_0(x(0), x(1)) + F_1(x(1))]$. Поэтому оптимальное управление u_0 находится из соотношения:

$$\min_{u_0 \in U_0} [\theta_0(x(0), x(1)) + F_1(x(1))] = F_0(x(0)); \qquad x(1) = f(x(0), u_0)$$

Последовательность управлений $u_0^* V^*(x(1))$, где $x(1) = f(x(0), u_0)$, будет оптимальной последовательностью управлений задачи (III). Обозначим через $F_k(x(k))$ минимальное значение функционала $\sum_k^{T-1} \theta_m$ для $\{x(m)\}$; $\{u_m\}$: $m = k+1, \ldots, T$, удовлетворяющих условиям задачи (III). Тогда, применяя последовательно принцип оптимальности к функ-

циям F_1, F_2 и т. д., приходим к системе функциональных уравнений:

$$\left.\begin{aligned} F_0(x(0)) &= \min_{u_0}[\theta_0(x(0), x(1)) + F_1(x(1))] \\ F_1(x(1)) &= \min_{u_1}[\theta_1(x(1), x(2)) + F_2(x(2))] \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ F_i(x(i)) &= \min_{u_i}[\theta_i(x(i), x(i+1)) + F_{i+1}(x(i+1))] \\ &\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ F_{T-1}(x(T-1)) &= \min_{u_{T-1}}[\theta_{T-1}(x(T-1), x(T))] \end{aligned}\right\} \quad \text{(IV)}$$

Функции $F_i(x(i))$ называются функциями Беллмана. Решение этой системы уравнений надо начинать с конца, т. е. сначала находить F_{T-1}, затем F_{T-2} и т. д.

Основное достоинство этих функциональных уравнений состоит в том, что решение задач большой размерности сводится к последовательному решению задач с меньшей размерностью. Так например, если $x(i)$ — одномерный вектор для всех i, то первоначальная задача состоит в минимизации функции T переменных. Каждая функция Беллмана является функцией одной переменной, поэтому функциональные уравнения (IV) сводят решение исходной задачи к решению последовательности одномерных задач, что значительно упрощает решение.

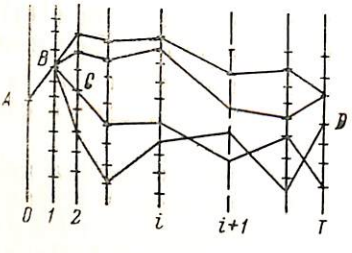

Рис. 2

Задачи динамического программирования обладают простой геометрической интерпретацией, которая часто является весьма полезной для их понимания и изучения. На рис. 2 вертикальным линиям соответствуют моменты времени, в которые рассматривается исследуемая задача. Вертикали O соответствует момент времени $t = 0$, вертикали T — момент времени $t = T$, а точкам на вертикалях — допустимые состояния системы в соответствующие моменты времени. Так как состояние системы в каждый момент времени характеризуется вектором, то каждому вектору состояния соответствует точка на рис. 2. Конечному числу точек соответствует конечное число состояний, бесконечному числу — бесконечное состояние. Для удобства рассмотрим только конечное число состояний. Пара точек $x(i)$; $x(i+1)$, расположенных на соседних вертикалях, называется допустимой, если существует допустимое управление, переводящее систему состояния $x(i)$ в состояние $x(i+1)$. Каждой допустимой паре точек естественно сопоставляется число $\theta_i(x(i); x(i+1))$. Каждой допустимой стратегии сопоставляется ломаная, соединяющая точку A с вертикалью T. Такие ломаные называются допустимыми. Каждой допустимой ломаной естественным образом сопоставляется число, равное $F = \sum_{i=0}^{T-1} \theta_i(x(i); x(i+1))$. Оптимальной стратегии соответствует ломаная с наименьшим значением суммы чисел θ_i.

Следовательно, исходную задачу можно сформулировать в следующем виде: требуется среди всех допустимых ломаных, соединяющих точку A с вертикалью T, найти такую, которой соответствует наименьшее значе-

ние суммы F. Очевидно, что эта формулировка эквивалентна формулировке исходной задачи.

Принцип оптимальности в такой интерпретации означает следующее. Пусть ломаной $ABCD$ соответствует наименьшее значение F. Тогда среди всех ломаных, соединяющих точку B с вертикалью T, ломаной BCD соответствует наименьшее значение $F_1 = \sum_{i=1}^{T-1} \theta_i$. Это наименьшее значение суммы есть не что иное как значение функции Беллмана $F_1(x(1))$ при $x(1) = B$. Справедливость этого утверждения вытекает из того, что оптимальный путь из точки B не зависит от траектории, ведущей в точку B.

НЕКОТОРЫЕ МЕТОДЫ РЕШЕНИЯ

Опишем вкратце основные методы решения задач динамического программирования. Желающие более детально ознакомиться с данными алгоритмами решения смогут воспользоваться литературой [1—8].

Для изложения методов решения воспользуемся геометрической интерпретацией, изложенной выше.

Метод последовательного анализа вариантов. Данный метод применяется в основном для задач, в которых система может находиться лишь в конечном числе состояний в каждый момент времени.

Суть этого метода, описанного выше на примере решения задачи о нахождении кратчайшего пути, состоит в следующем.

Пусть для каждой точки $(i+1)$-й вертикали найдена «оптимальная» ломаная, соединяющая эту точку с вертикалью T. На рис. 3 изображены все эти ломаные. Тогда для нахождения «оптимальной» ломаной из точки A достаточно рассмотреть только ломаные, изображенные на рис. 3, и выбрать из них оптимальную. Если пара точек A и C является не допустимой, то ломаная, проходящая через эти точки, не рассматривается.

Предположим, что ломаная, проходящая через точку B, является лучшей среди остальных; тогда при дальнейшем рассмотрении нет необходимости рассматривать ломаные, проходящие через точку A и не проходящие через точку B. Основным достоинством этого метода является то, что он позволяет найти не только одну оптимальную стратегию, ведущую из начального состояния, а дает всю структуру оптимальных стратегий.

Недостатком этого метода является большой объем вычислений, который быстро возрастает с ростом размерности вектора состояния системы $x(i)$ и с ростом количества состояний в каждый момент времени.

Метод «трубок». Существуют различные варианты этого метода. Здесь будет описан только один.

Всегда можно приближенно считать, что множество состояний системы в каждый момент времени является конечным. Так как этих состояний может быть чрезвычайно много, то метод последовательного анализа вариантов оказывается не применим. Рассмотрим только часть множества состояний, отмеченную на рис. 4 штрихами. Вся область допустимых состояний лежит между дугами S_1 и S_2. Назовем заштрихованную область «трубкой». Найдем стратегию, лучшую среди всех стратегий, лежащих внутри «трубки», относительно выбранных состояний системы *. Это можно сделать, например, методом, описанным выше. После того, как «локально-оптимальная» стратегия найдена, «трубка» корректируется: там, где траектория не выходит на границу, «трубка» сужается, но добавляется

* Так как в «трубку» могло попасть тоже довольно большое число состояний, то выбирается только некоторая часть из них. Другие состояния исследуются в процессе решения.

количество рассматриваемых допустимых состояний системы. Там, где траектория выходит на границу, «трубка» изгибается в соответствующую сторону. Такой процесс повторяется до тех пор, пока приращение значения целевой функции становится достаточно малым.

Основное достоинство этого метода состоит в том, что на каждом шаге исследуется не вся область допустимых значений, а лишь часть ее. Поэтому количество вычислений в этом методе в меньшей степени зависит от размерности $x(i)$, чем в предыдущем методе. Если начальное приближе-

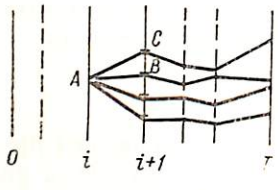

Рис. 3

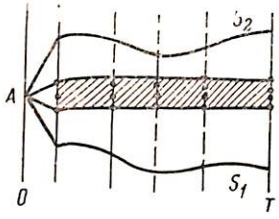

Рис. 4

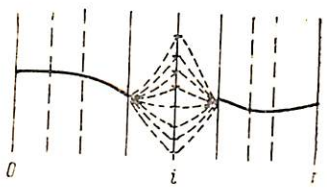

Рис. 5

ние выбрано достаточно хорошо, т. е. начальная «трубка» выбрана удачно, то метод довольно быстро приводит к ответу. Однако, если «трубка» выбрана неудачно, то стратегия, полученная этим методом, не всегда является оптимальной. Для проверки найденной стратегии на оптимальность, как правило, нужно проводить дополнительные исследования.

Следует отметить, что методы, описанные выше, являются методами решения функциональных уравнений. Например, значение целевой функции для стратегии ABD (рис. 3) есть не что иное как значение i-й функции Беллмана в точке $x_i = A$, т. е. $F_i(A)$.

Метод локальных вариаций. Этот метод не является «чистым» методом динамического программирования, но он тесно примыкает к ним и даже может быть рассмотрен как частный случай метода «трубок».

Выбирается некоторая допустимая стратегия и затем строится итеративный процесс, улучшающий эту стратегию. Каждая итерация состоит из T шагов. На i-м шаге каждой итерации рассматривается множество допустимых стратегий, изображенных на рис. 5, и выбирается лучшая из них. (Эти стратегии отличаются одна от другой только состоянием в i-й момент времени.)

Метод локальных вариаций является очень простым для реализации его на ЭВМ, не требует хранения большого количества промежуточной информации и еще в меньшей степени зависит от размерности $x(i)$, чем метод «трубок». Единственным недостатком этого метода является то, что область сходимости его еще более узкая, чем у метода «трубок».

Рассмотренные методы обладают интересной особенностью: чем больше имеется ограничений на область допустимых управлений и на область допустимых состояний системы, тем эти методы быстрее дают решение задачи. Это происходит в результате того, что эти методы являются, по существу, методами направленного перебора вариантов, а чем больше ограничений, тем меньше допустимых вариантов и, значит, тем быстрее их можно проанализировать.

Кроме того, эти методы не требуют знания аналитических выражений рассматриваемых функций. Нужно только уметь вычислять их значения в любых точках. Это обстоятельство часто является весьма существенным при применении тех или иных методов расчета.

* * *

Мы рассмотрели модели только детерминированные, время предполагалось дискретным. Однако не представляет особого труда перенести аналогичные рассуждения для стохастических моделей или для моделей с непрерывным временем.

В заключение отметим большие возможности динамического подхода к анализу широкого класса практических задач. Хотя многие задачи в своей традиционной постановке не имеют ничего общего с задачами динамического программирования, однако их можно сформулировать так, чтобы формулировка полностью укладывалась в рамки описанной здесь схемы. К таким задачам относятся задачи оптимального управления, задачи вариационного исчисления, задачи линейного и целочисленного линейного программирования и др. Однако такое сведение не всегда является целесообразным, так как оно не всегда учитывает специфические особенности данного класса задач.

РЕКОМЕНДУЕМАЯ ЛИТЕРАТУРА

1. Р. Беллман. Динамическое программирование. М., Изд-во иностр. лит., 1960.
2. Р. Беллман, С. Дрейфус. Прикладные задачи динамического программирования. М., «Наука», 1965.
3. Б. Т. Поляк. Задачи многоходового выбора. В сб. Вопросы теории математических машин. Вып. 2. М., 1962.
4. Е. С. Вентцель. Элементы динамического программирования. М., «Наука», 1964
5. Р. А. Ховард. Динамическое программирование и марковские процессы. М., «Сов. радио», 1964.
6. В. С. Михайлевич, Н. З. Шор. О численных методах решения многовариантных плановых и технико-экономических задач. Научно-метод. материалы экономико-матем. семинара ВЦ АН УССР. Вып. 1. Киев, Изд-во ВЦ АН УССР, 1962.
7. И. А. Крылов, Ф. Л. Черноусько. Решение задач оптимального управления методом локальных вариаций. Ж. Вычисл. матем. и матем. физ., 1966, т. 6, № 2.

Поступила в редакцию
4 VIII 1967