МИНИСТЕРСТВО СЕЛЬСКОГО ХОЗЯЙСТВА РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Южно-Уральский государственный аграрный университет»
(ФГБОУ ВО Южно-Уральский ГАУ)
Институт агроинженерии

Кафедра Эксплуатация машинно-тракторного парка, и технология и механизация животноводства

ЛЕКЦИЯ

на тему

Динамическое программирование

по направлению подготовки 35.04.06 «Агроинженерия»

программа подготовки – Технический сервис в сельском хозяйстве

доцент кафедры ЭМТП и ТМЖ,
к.т.н, доцент В.Н. Николаев

1 Сущность динамического программирования

Динамическое программирование (ДП) определяет оптимальное решение n-мерной задачи путем ее декомпозиции на n этапов, каждый из которых представляет подзадачу относительно одной переменной. Вычислительное преимущество такого подхода состоит в том, что мы занимаемся решением одномерных оптимизационных подзадач вместо большой n-мерной задачи. Фундаментальным принципом ДП, составляющим основу декомпозиции задачи, является оптимальность. Так как природа каждого этапа решения зависит от конкретной оптимизационной задачи, ДП не предлагает вычислительных алгоритмов непосредственно для каждого этапа. Вычислительные аспекты решения оптимизационных подзадач на каждом этапе проектируются и реализуются по отдельности (что, конечно, не исключает применения единого алгоритма для всех этапов).

2 Рекуррентная природа вычислений ДП

Вычисления в ДП выполняются рекуррентно в том смысле, что оптимальное решение одной подзадачи используется в качестве исходных данных для следующей. Решив последнюю подзадачу, мы получим оптимальное решение исходной задачи. Способ выполнения рекуррентных вычислений зависит от того, как производится декомпозиция исходной задачи. В частности, подзадачи обычно связаны между собой некоторыми общими ограничениями. Если осуществляется переход от одной подзадачи к другой, то должны учитываться эти ограничения.

Пример 1. Задача о кратчайшем пути. Предположим, необходимо выбрать кратчайший путь между двумя городами. Сеть дорог, показанная на рисунке 1, представляет возможные маршруты между исходным городом, находящимся в узле 1, и конечным пунктом, который находится в узле 7. Маршруты проходят через промежуточные города, обозначенные на сети узлами с номерами 2-6.

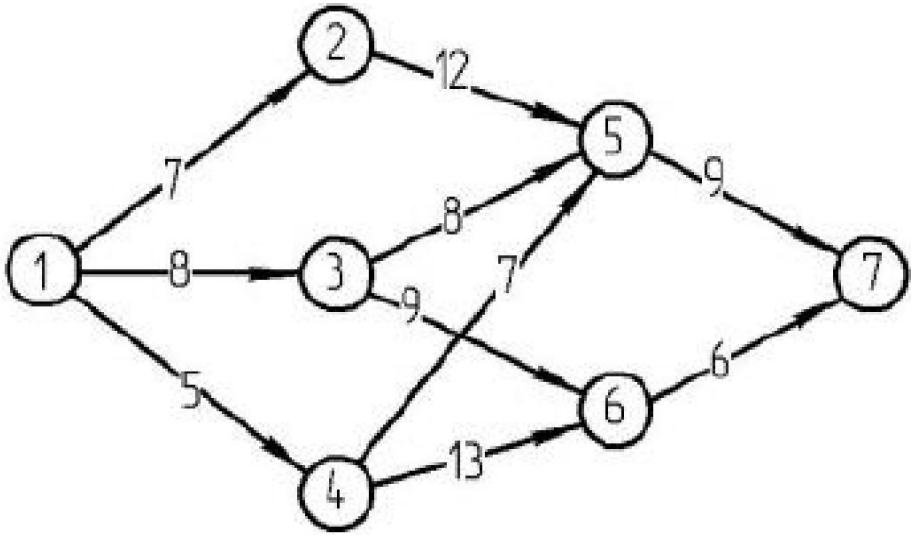

Рисунок 1 – Сеть дорог для примера 1.

Мы можем решить эту задачу посредством полного перебора всех маршрутов между узлами 1 и 7 (имеется пять таких маршрутов). Однако в большой сети полный перебор является неэффективным с вычислительной точки зрения.

Чтобы решить эту задачу с помощью методов динамического программирования, сначала разделим ее на этапы. Вертикальные пунктирные линии на рисунке 2 очерчивают три этапа задачи. Далее выполняются вычисления для каждого этапа в отдельности.

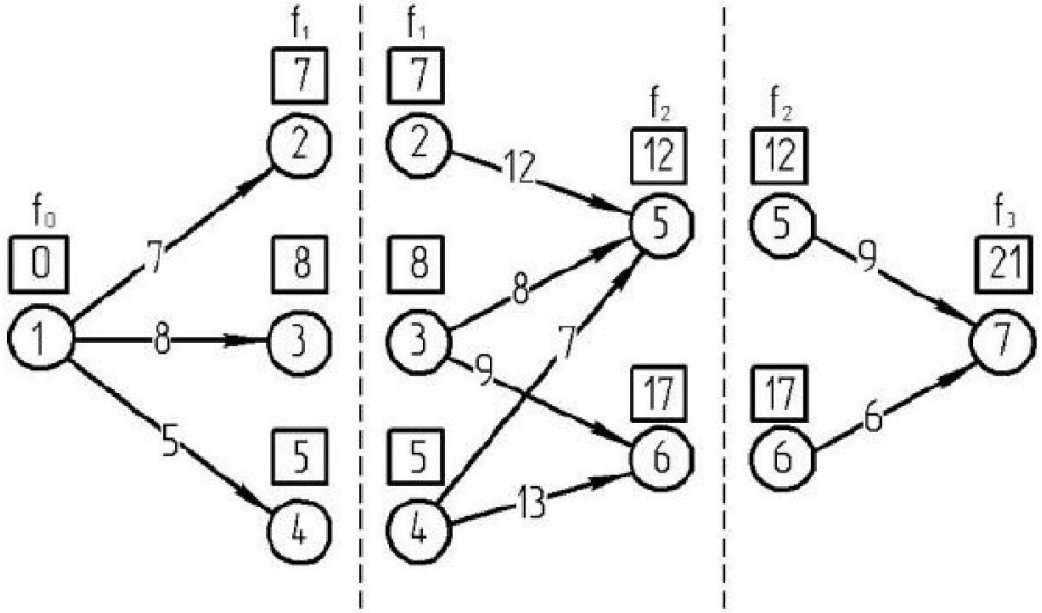

Рисунок 2 – Декомпозиция задачи на три этапа

Общая задача состоит в вычислении кратчайших (постепенно накапливаемых) расстояний ко всем вершинам этапа с последующим использованием этих расстояний в качестве исходных данных для следующего этапа. Рассматривая узлы, относящиеся к первому этапу, замечаем, что каждый из узлов 2, 3 и 4 связан с начальным узлом 1

единственной дугой (рис. 2). Следовательно, для первого этапа имеем следующее.

Этап 1. Итоговые результаты.

Кратчайший путь к узлу 2 равен 7 миль (из узла 1). Кратчайший путь к узлу 3 равен 8 миль (из узла 1). Кратчайший путь к узлу 4 равен 5 миль (из узла 1).

Далее переходим ко второму этапу для вычисления кратчайших (накопленных) расстояний к узлам 5 и 6. Рассматривая узел 5 первым, из рисунка. 2 замечаем, что есть три возможных маршрута, по которым можно достичь узла 5, а именно (2, 5), (3, 5) и (4, 5). Эта информация вместе с кратчайшими расстояниями к узлам 2, 3, и 4 определяет кратчайшее (накопленное) расстояние к узлу 5 следующим образом.

$$\begin{pmatrix}\text{Кратчайший}\\ \text{путь к узлу 5}\end{pmatrix} = \underset{i=2,3,4}{min}\left\{\begin{pmatrix}\text{Кратчайший}\\ \text{путь к узлу } i\end{pmatrix} + \begin{pmatrix}\text{Расстояние от}\\ \text{узла } i \text{ к узлу 5}\end{pmatrix}\right\} = min\begin{Bmatrix}7+12=19\\ 8+8=16\\ 5+7=12\end{Bmatrix}$$

$$= 12 \text{ (из узла 4)}$$

Аналогично из узла 6 получаем следующее:

$$\begin{pmatrix}\text{Кратчайший}\\ \text{путь к узлу 6}\end{pmatrix} = \underset{i=3,4}{min}\left\{\begin{pmatrix}\text{Кратчайший}\\ \text{путь к узлу } i\end{pmatrix} + \begin{pmatrix}\text{Расстояние от}\\ \text{узла } i \text{ к узлу 6}\end{pmatrix}\right\} = min\begin{Bmatrix}8+9=17\\ 5+13=18\end{Bmatrix}$$

$$= 17 \text{ (из узла 3)}$$

Этап 2. Итоговые результаты.

Кратчайший путь к узлу 5 равен 12 миль (из узла 4).

Кратчайший путь к узлу 6 равен 17 миль (из узла 3).

Последним шагом является третий этап. Конечный узел 7 можно достичь как из узла 5, так и 6. Используя итоговые результаты этапа 2 и расстояния от узлов 5 и 6 к узлу 7, получаем следующее:

$$\begin{pmatrix}\text{Кратчайший}\\ \text{путь к узлу 7}\end{pmatrix} = min\begin{Bmatrix}12+9=21\\ 17+6=23\end{Bmatrix} = 21\ (\text{из узла 5})$$

Этап 3. Итоговые результаты.

Кратчайший путь к узлу 7 равен 21 миле (из узла 5).

Приведенные вычисления показывают, что кратчайшее расстояние между узлами 1 и 7 равно 21 миле. Города, через которые проходит кратчайший маршрут, определяются следующим образом. Из итоговых результатов третьего этапа следует, что узел 7 связывается с узлом 5. Далее из итоговых результатов второго этапа следует, что узел 4 связывается с узлом 5. Наконец, из итоговых результатов первого этапа следует, что узел 4 связывается с узлом 1. Следовательно, оптимальным маршрутом является последовательность 1 → 4 → 5 → 7.

Теперь покажем, как рекуррентные вычисления динамического программирования можно выразить математически. Пусть $f_i(x_i)$ - кратчайшее расстояние до узла x_i, на этапе i, d (x_{i-1}, x_i - расстояние от узла x_{i-1} до узла x_i. Тогда f_i вычисляется на основе значений f_{i-1} с помощью следующего рекуррентного уравнения

$$f_i(x_i) = \min_{\substack{\text{все допустимые} \\ (x_{i-1},x_i)-\text{маршруты}}} \{d(x_{i-1},x_i) + f_{i-1}(x_{i-1})\}, i = 1,2,3$$

При i = 1 полагаем $f_0(x_0) = 0$. Это уравнение показывает, что кратчайшие расстояния $f_i(x_i)$ на этапе i должны быть выражены как функции следующего узла x_i. В терминологии динамического программирования x_i именуется состоянием системы на этапе i.

В действительности состояние системы на этапе i - это информация, связывающая этапы между собой, при этом оптимальные решения для оставшихся этапов могут приниматься без повторной проверки того, как были получены решения на предыдущих этапах. Такое определение состояния системы позволяет рассматривать каждый этап отдельно и гарантирует, что решение является допустимым на каждом этапе.

Определение состояния системы приводит к следующему унифицированному положению.

Принцип оптимальности. На каждом этапе оптимальная стратегия определяется независимо от стратегий, использованных на предыдущих этапах.

Применение принципа оптимальности демонстрируется вычислениями из примера 1. Например, на этапе 3 мы используем кратчайшие пути к узлам 5 и 6 и не интересуемся, как эти узлы были достигнуты из узла 1.

Рекуррентные алгоритмы прямой и обратной прогонки

В примере 1 вычисления проводились последовательно: от первого этапа до третьего. Такая последовательность вычислений известна как алгоритм прямой прогонки. Этот же пример может быть решен с помощью алгоритма обратной прогонки, в соответствии с которым вычисления проводятся от третьего этапа до первого.

Алгоритмы прямой и обратной прогонки приводят к одному и тому же решению. Несмотря на то, что алгоритм прямой прогонки представляется более логичным, в специальной литературе, посвященной динамическому программированию, неизменно используется алгоритм обратной прогонки. Причина этого в том, что в общем случае алгоритм обратной прогонки может быть более эффективным с вычислительной точки зрения. Продемонстрируем использование алгоритма обратной прогонки на примере 1. Мы также представим вычисления динамического программирования в компактной табличной форме.

Пример 2. Рекуррентное уравнение для алгоритма обратной прогонки в примере 1 имеет вид

все допустимые (xj, xi+1)-маршруты

где $f_4(x_4) = 0$ для $x_4 = 7$.

Соответствующей последовательностью вычислений будет $f_3 \rightarrow f_2 \rightarrow f_1$.

Этап 3. Поскольку узел 7 ($x_4 = 7$) связан с узлами 5 и 6 ($x_3 = 5$ и 6) только одним маршрутом, альтернативы для выбора отсутствуют, а результаты третьего этапа можно подытожить следующим образом (табл. 1).

Таблица 1 – Результаты этапа 3 задачи о кратчайшем пути

	$d(x_3, x_4)$	Оптимальное решение	
x_3	$x_4 = 7$	$f_3(x_3)$	x_4
5	9	9	7
6	6	6	7

Этап 2. Так как маршрута (2, 6) не существует, соответствующая альтернатива не рассматривается. Используя значения f3(x3), полученные на третьем этапе, мы можем сравнить допустимые альтернативные решения, как показано в таблице 2.

Таблица 2 – Результаты этапа 2 задачи о кратчайшем пути

	$d(x_2, x_3) + f_3(x_3)$		Оптимальное решение	
x_2	$x_3 = 5$	$x_3 = 6$	$f_2(x_2)$	x_3
2	$12 + 9 = 21$	-	21	5
3	$8 + 9 = 17$	$9 + 6 = 15$	15	6
4	$7 + 9 = 16$	$13 + 6 = 19$	16	5

Этап 2. Так как маршрута (2, 6) не существует, соответствующая альтернатива не рассматривается. Используя значения $f_3(x_3)$, полученные на третьем этапе, мы можем сравнить допустимые альтернативные решения, как показано в таблице 2.

Оптимальное решение второго этапа означает следующее: если вы находитесь в узле (городе) 2 или 4, кратчайший путь к узлу 7 проходит через узел 5, а если в узле 3 - через узел 6.

Этап 1. Из узла 1 начинаются три альтернативных маршрута: (1, 2), (1, 3) и (1, 4).

Используя значения $f_2(x_2)$, полученные на втором этапе, вычисляем данные таблицы 3.

Таблица 3 – Результаты этапа 1 задачи о кратчайшем пути

	$d(x_1, x_2) + f_2(x_2)$			Оптимальное решение	
x_1	$x_2 = 2$	$x_2 = 3$	$x_2 = 4$	$f_1(x_1)$	x_2
1	$7 + 21 = 28$	$8 + 15 = 23$	$5 + 16 = 21$	21	4

Оптимальное решение на первом этапе показывает, что кратчайший путь проходит через город 4. Далее из оптимального решения на втором этапе следует, что из города 4 необходимо двигаться в город 5. Наконец, из оптимального решения на третьем этапе следует, что город 5 связан с городом 7. Следовательно, полным маршрутом, имеющим кратчайшую длину, является $1 \rightarrow 4 \rightarrow 5 \rightarrow 7$, и его длина равна 21 миле.