НАЦИОНАЛЬНАЯ АКАДЕМИЯ НАУК
АЗЕРБАЙДЖАНСКОЙ РЕСПУБЛИКИ

ИНСТИТУТ КИБЕРНЕТИКИ

Г.Г.АБДУЛЛАЕВА , М.В.МАМЕДОВА

ДИНАМИЧЕСКОЕ ПРОГРАММИРОВАНИЕ

В ТРАНСПОРТНЫХ ЗАДАЧАХ

(Препринт)

Печатается по решению Ученого Совета
ИК НАН Азербайджанской Республики
Протокол №5 от 20.06.2001

Баку - 2001

Рецензент и научный консультант:

к. ф.-м. н. Магеррамов М. А.

Аннотация

Рассматривается пять различных задач оптимизации использования распределённых ресурсов. Применяется метод динамичесого программирования, причём осуществляется сведение бесконечномерной вариационной задачи к конечномерному случаю. Построены математические модели как на основе детерминированных, так и стохастических процессов. Доказаны теоремы существования и единственности решения рекуррентного уравнения многомерного стохастического бесконечношагового процесса управления распределёнными ресурсами, а также доказаны теоремы о свойствах решений. Сформулированы алгоритмы решения двух практически важных постановок задач и выполнена оцененка их сложности. Разработанные алгоритмы легли в основу программного обеспечения, приведённого в работе.

Глава I. Введение.

§1. Метод динамического программирования

Оптимизационные задачи встречаются почти во всех отраслях науки, техники и хозяйства. С такими задачами часто приходится иметь дело в промышленной технологии, организации и управлении производством, в экономическом планировании, в различных областях физики. В подобных задачах часто оказывается целесообразным принимать решение не сразу, а шаг за шагом, как процесс состоящий из нескольких этапов. Систематизация примеров подобного рода показывает, что многие из них могут быть решены некоторым единообразным математическим аппаратом – теорией динамического программирования.

Предметом динамического программирования является изучение оптимальных многошаговых решений. Классические методы нахождения экстремумов функций многих переменных здесь часто оказываются неприменимыми ввиду большого числа параметров, от которых зависит решение. Лежащий же в основе динамического программирования принцип оптимальности часто может быть реализован в виде такого функционального уравнения, решение которого более доступно методам современной математики (в том числе вычислительной математики), чем решение соответствующих уравнений в условиях классической постановки задачи. На таком пути обнаруживается и новый подход к решению обычных задач вариационного исчисления.

Рассмотрим каким образом эта идея приводит к новой трактовке процесса решения. В традиционной трактовке мы рассматриваем весь многошаговый процесс решения как некоторый единый многомерный шаг. Так, если мы имеем N – шаговый процесс, в котором на каждом шаге можно принять M решений, то классический подход приводит к MN – мерному одношаговому

процессу. Как избежать этого нарастания размерности, которое сильно осложняет решение и процесс его вычисления? Для этого мы должны знать характеристики системы, определяющие решения, которые следует принимать на каждом шаге процесса. Иначе говоря, нам следует определить оптимальные решения, которые следует принимать для каждого состояния системы. Преимущество такой постановки заключается в том, что она уменьшает размерность процесса до той, которая присуща самой задаче. Это делает задачу более легко поддающейся аналитической обработке и значительно упрощает её вычисление. Данный подход приводит нас к некоторому типу приближения, обладающему свойством монотонности сходимости.

Преимущество рассуждений в терминах поведений даёт нам возможность ставить и рассматривать такие задачи, которые не поддаются изучению другими методами.

Рассмотрим пример применения метода динамического программирования для задачи многошагового процесса распределения ресурсов.

Сначала сформулируем задачу в классической постановке и проиллюстрируем метод приведения её к многомерной задаче максимизации.

Пусть x – некоторое физическое количество, а y – его неотрицательная часть. Тогда величина x делится на две части: y и x-y. Доход от первой части обозначим $g(y)$, а от второй $h(x-y)$. Задача состоит в том, что для любого $x\geq 0$ подобрать такое $y\in[0,x]$, чтобы величина

$$R_1(x,y)=g(y)+h(x-y)$$

была максимальной. Для существования максимального значения положим g и h непрерывными. Это одношаговый процесс.

Рассмотрим теперь двухшаговый процесс. Пусть для получения дохода $g(y)$ первоначальное количество y уменьшается

до ay, где $0\leq a<1$ и аналогично количество x-y уменьшается до $b(x-y)$, где $0\leq b<1$.

Повторим процесс с суммарным остатком

$$ay+b(x-y)=x_1=y_1+(x_1-y_1),\ \text{где}\ 0\leq y_1\leq x_1.$$

В результате нового распределения на втором шаге получится доход

$$g(y_1)+h(x_1-y_1).$$

Полный доход двух шагов будет:

$$R_2(x,y,y_1)=g(y)+h(x-y)+g(y_1)+h(x_1-y_1).$$

Для любого x максимальный суммарный доход получится при максимизации $R_2(x,y,y_1)$ относительно y и y_1 в двумерной области

$$0\leq y\leq x,$$
$$0\leq y_1\leq x_1$$

Продолжая рассуждения, для N – шагового процесса получим задачу максимизации функции

$$R_N(x,y,y_1,\dots,y_{N-1}) = g(y)+h(x-y)+$$
$$+g(y_1)+h(x_1-y_1)+\dots+g(y_{N-1})+h(x_{N-1}-y_{N-1}),$$

в области $0\leq y\leq x$, $0\leq y_i\leq x_i$, $\overline{i=1,N-1}$, где

$$x_1=ay+b(x-y)$$
$$x_2=ay_1+b(x_1-y_1)$$
$$\dots\dots\dots\dots\dots\dots\dots\dots$$
$$x_{N-1}=ay_{N-2}+b(x_{N-2}-y_{N-2}).$$

Т.о. задача сводится к максимизации функции R_N в N-мерной области по переменным $y,y_1,\dots,y_{N-1}$. Решение этой задачи для больших N приводит к длительным громоздким вычислениям. Кроме этого, если потребуется решать задачу для различных значений x, a, b, либо для определённого класса функций g и h, решение задачи этим методом потребует огромного количества вычислений. Во многих случаях представляет интерес не только получение численного решения, но и изучение структуры решения.

Сформулируем теперь задачу так, чтобы сохранить её одномерность. Очевидно, что максимальное значение полного дохода зависит лишь от величины x и числа шагов N. Обозначим

$f_N(x)$ как максимум дохода от N – шагового процесса для величины x, где $N=1,2,...$ и $x\geq 0$. То есть

$$f_N(x) = \max_{(y_1,y_2,\ldots,y_{N-1})} R_N(x, y, y_1, \ldots, y_{N-1})$$

(1)

$$f_1(x) = \max_{0\leq y\leq x}[g(y) + h(x-y)]$$

Получим выражение $f_2(x)$ через $f_1(x)$ для двухшагового процесса. Для двухшагового процесса величина дохода складывается, как доход от первого шага плюс доход от второго шага. После первого шага для распределения остаётся величина $x_1=ay+b(x-y)$. Очевидно, что какой бы ни была первоначально выбранная y, оставшаяся величина $x_1=ay+b(x-y)$ должна быть использована наивыгоднейшим образом. Т.к. второй шаг выполняется по тем же законам, что и первый, то прибыль от второго шага равна:

$$\max_{0\leq y_1\leq ay+b(x-y)} [g(y_1)+h(ay+b(x-y)-y_1)] =$$
$$= \max_{0\leq y_1\leq x_1}[g(y_1)+h(x_1-y_1)] = f_1(x_1) = f_1(ay+b(x-y))$$

Следовательно, для окончательного дохода от двухшагового процесса при начальном распределении y получаем выражение

$$R_2(x,y,y_1)=g(y)+h(x-y)+f_1(ay+b(x-y)).$$

Т.к. y выбирается так, чтобы максимизировать это выражение, то получим рекуррентное выражение

$$f_2(x) = \max_{0\leq y\leq x}[g(y)+h(x-y)+f_1(ay+b(x-y))],$$

которое связывает функции $f_1(x)$ и $f_2(x)$.

Рассуждая аналогично, для N – шагового процесса получим:

$$f_N(x) = \max_{0\leq y\leq x}[g(y)+h(x-y)+f_{N-1}(ay+b(x-y))], \quad (2)$$

для $N\geq 2$, где $f_1(x)$ определяется из (1).

Т.о., значение $f_N(x)$ определяется, начиная с нахождения значения $f_1(x)$. Подставляя f_1 в формулу (2), находим $f_2(x)$, подставляя f_2 находим $f_3(x)$, и т.д. до $f_N(x)$. Одновременно определяем

$y, y_1, y_2, \ldots, y_{N-1}$ – т.е. систему распределений, максимизирующих полный доход.

Приведем пример применения метода динамического программирования для решения одной транспортной задачи оптимизации.

§2. Транспортная задача

Имеется некоторое транспортное средство, способное взять на борт количество топлива не более V. Предполагая, что расход топлива транспортного средства равен одной единице количества топлива на одну единицу расстояния, предложить оптимальную стратегию прохождения транспортным средством некоторого расстояния L, предполагая возможность промежуточного складирования части имеющегося на борту топлива, равно как и промежуточную заправку ранее складированного топлива при любой траектории движения транспортного средства. Определить расход топлива, соответствующий оптимальной стратегии.

Решение.

Прежде всего заметим, что в случае $L\leq V$ оптимальная стратегия заключается в заправке L горючего и безостановочном прохождении всего расстояния. В общем случае приходится рассматривать произольные траектории, описываемые произвольными кусочно-гладкими функциями времени $\phi(t), \phi(0)=0, \phi(T)=L$, что приводит к (бесконечномерной) вариационной задаче. Очевидными являются преимущества дискретных методов, сводящих задачу к конечномерной постановке. В качестве первого шага дискретизации задачи рассмотрим следующую постановку: найти максимальное расстояние x_n, которое может преодолеть транспортное средство, использовав $(n+1)V$ топлива при неограниченном запасе топлива в начале пути и некоторой стратегии, подразумевающей возможность промежуточного складирования и последующей заправки топлива в

пути. Определить также оптимальную стратегию, обеспечивающую достижение требуемого расстояния. Из ранее сказанного следует, что $x_0 = V$. Докажем следующее утверждение.

Теорема 1.

Пусть x_n имеет тот же смысл, что и выше. Оптимальная стратегия прохождения расстояния x_n при расходе количества топлива $(n+1)V$ (n – неотрицательное целое) следующая:

1. Транспортное средство $n+1$ раз выезжает из начала пути, взяв на борт V топлива.
2. В первый раз доехав до точки, находящейся на расстоянии $x_n - x_{n-1}$ от начала пути, в этой точке складируется количество топлива $V - 2(x_n - x_{n-1})$, после чего транспортное средство возвращается в начальную точку.
3. В каждое последующее прибытие транспортного средства в точку, находящуюся на расстоянии $x_n - x_{n-1}$ от начала пути, транспортное средство берёт на борт количество топлива $x_n - x_{n-1}$ и следует оптимальной стратегии прохождения расстояния x_{n-1}, как если бы указанная точка была начальной.
4. При каждом возвращении в точку в точку, находящуюся на расстоянии $x_n - x_{n-1}$ от начала пути, на борт берётся $x_n - x_{n-1}$ топлива и транспортное средство возвращается в начальную точку.

Справедливо равенство:

$$x_n = V\sum_{k=0}^{n}(2k+1)V \qquad (2)$$

Доказательство.

Применим метод математической индукции. Прежде всего, для $n=0$ оптимальность вышеописанного алгоритма, а также формулы (2) очевидны. Докажем справедливость утверждения для $n+1$, предполагая справедливость для n. Для наглядности рассуждений предположим, что точки $x_1, x_2, ..., x_n, x_{n+1}$ откладываются влево от точки назначения. Предположим, что транспортное средство движется по какой-то оптимальной стратегии, забирая некоторое количество топлива в точке x_{n+1}, оставляя часть топлива в промежуточных точках, вновь забирая его впоследствии и т.д. В таком случае, при каждом прибытии в точку x_n слева запас топлива на борту не будет превышать V, а при каждом возвращении в точку

x_n, на борту будет не меньше 0 топлива. Далее заметив, что любые промежуточные запасы, оставляемые правее точки x_n, по необходимости делаются лишь из количеств топлива, привозимых транспортным средством из участка левее точки x_n, заключаем, что отвлекаясь от части стратегии, относящейся к прохождению транспортным средством участка между точками x_{n+1} и x_n, оставшаяся часть стратегии является стратегией прохождения расстояния x_n. Оптимальность всей стратегии требует оптимальности стратегии, относящейся к прохождению расстояния x_n. Следовательно, необходимо обеспечить $n+1$ выезд из точки x_n направо с запасом V топлива на борту и n возвращений в точку x_n слева. И действительно, предположим, что оптимальная для $n+1$ стратегия использует неоптимальную стратегию для прохождения расстояния от x_n до точки назначения. В таком случае количество выездов из точки x_n необходимо будет больше $n+1$ и, соответственно, количество возвращений в x_n будет больше n и на 1 меньше, чем выездов. При оптимальной стратегии прохождения расстояния x_{n+1} при каждом прибытии в x_n транспортное средство оставляет какое-то количество топлива или берёт часть ранее оставленного топлива, двигаясь при этом дальше или возвращаясь назад. Следовательно, можно сформулировать эквивалентную оптимальную стратегию, при которой транспортное средство сначала доставляет в точку x_n всё топливо, которое должно быть оставлено в этой точке, возвращаясь каждый раз назад в x_{n+1}, а при последней доставке запаса топлива в x_n начинает реализовывать стратегию прохождения расстояния x_n. В таком случае при неоптимальности стратегии прохождения расстояния x_n на доставку необходимого количества топлива (большего $(n+1)V$ ввиду неоптимальности) из точки x_{n+1} в x_n будет израсходовано количество менее V, что означает уменьшение расстояния между x_{n+1} в x_n в сравнении со случаем оптимальности стратегии прохождения расстояния от x_n до точки назначения, как части оптимальной стратегии прохождения расстояния x_{n+1}.

В виду требования оптимальности, на все прохождения участка справа от x_n в любом направлении будет израсходовано $(n+1)V$, следовательно, на все прохождения участка между точками x_{n+1} и x_n в обоих направлениях должно быть израсходовано количество топлива V. Отсюда, для достижения максимальности расстояния

между точками x_{n-1} и x_n и удовлетворения требований оптимальной стратегии для n (в смысле количеств топлива при подъёзде к точке x_n справа и слева), получаем следующую оптимальную стратегию:

1. Взяв на борт V топлива, выезжаем в точку x_{n+1}, где оставляем $V-2(x_{n+1}-x_n)$ топлива и возвращаемся в начальную точку.
2. $n+1$ раз выезжаем из начальной точки, в точке x_n берём на борт $x_{n+1}-x_n$ топлива из оставленного вначале запаса, а при каждом из n возвращений в x_n также берём на борт $x_{n+1}-x_n$ и возвращаемся в начальную точку.

Заметим, что в соответствии со сказанным ранее об альтернативных оптимальных стратегиях, оптимальной будет являться также следующая стратегия:

1. $n+1$ раз выезжаем в точку x_{n+1}, взяв на борт V топлива, где оставляем $V-2(x_{n+1}-x_n)$ топлива и возвращаемся в начальную точку.
2. Выезжаем из начальной точки, взяв на борт V топлива, оставляем $V-(x_{n+1}-x_n)$ топлива в точке x_n и реализовываем оптимальную стратегию прохождения расстояния от x_n до точки назначения.

При первой стратегии запас топлива, оставленный в точке x_n на первом шаге, должен покрыть все дальнейшие дозаправки. Учитывая, что из x_n транспортное средство выезжает $n+1$ раз, а возвращается n раз, что означает прохождение расстояния между точками x_{n+1} и x_n $(2n+1)$ раз, исключая первый проезд туда и обратно для доставки запаса топлива $V-2(x_{n+1}-x_n)$, получим формулу:

$$V-2(x_{n+1}-x_n)=(2n+1)(x_{n+1}-x_n), \qquad (3)$$

откуда

$$x_{n+1}=x_n+\frac{V}{(2n+3)}, \qquad (4)$$

т.е. получена формула (2).

Аналогично, при второй стратегии всё оставленное в точке x_n топливо должно равняться минимальному количеству топлива,

необходимого для прохождения расстояния x_{n+1}, т.е. $(n+1)V$, опять-таки

$$(n+1)(V-2(x_{n+1}-x_n))+V-(x_{n+1}-x_n)=(n+1)V,$$
$$x_{n+1}=x_n+\frac{V}{2n+3}.$$

Докажем, что при оптимальной стратегии из начальной точки следует выезжать ровно $n+2$ раз. И действительно, при большем числе выездов расход, связанный с прохождением этого расстояния возрастёт, что означает доставку в точку x_n меньшего количества топлива для хранения. Если же количество выездов менее $n+2$, то общее количество израсходованного топлива будет менее $(n+2)V$, что противоречит определению x_{n+1}. Теорема доказана.

Следствие 1.

Пусть функция $y(q)$ равна максимальному расстоянию, которое может преодолеть транспортное средство, использовав количество топлива q. Верна следующая формула:

$$y(q)=V\sum_{k=0}^{n}(2k+1)^{-1}+(q-(n+1)V)(2n+3)^{-1}, \qquad (5)$$

где

$$(n+1)V<q\le(n+2)V. \qquad (6)$$

Интересно отметить, что из асимптотики расходящегося гармонического ряда следует, что $y(q)-const\,V\ln(qV^{-1})$, $q\to\infty$, т.е. количество требуемого топлива растёт экспоненциально при росте преодолеваемого расстояния и постоянстве максимального бортового запаса топлива.

Глава II. Постановки задач.

В этой главе приводятся постановки задач, решённых в настоящей работе.

§1. Детерминированный и стохастический дискретные процессы.

Многие задачи управления распределёнными ресурсами приводятся к некоторому стохастическому процессу, в котором исход принятия

того или иного решения не определён точно, но задаётся некоторым распределением вероятностей. В подобных обстоятельствах наиболее удобным критерием оптимальности может служить некоторая величина, характеризирующая «среднее» или ожидаемое значение некоторой весовой функции. Следует отметить, что конкретное определение «средних», а также «весовых» функций крайне специфично и должно осуществляться в контексте конкретной задачи. Изучаемые в настоящей работе типовые постановки задач для детерминированного и стохастического дискретных процессов иллюстрируются следующими примерами.

Задача 1 (детерминированный процесс).
Допустим, что состояние системы определяется n-мерным вектором $(x_1,...,x_n) \in D$, где D – некоторая область. Пусть $F_y : D \to D$ параметризованное семейство преобразований множества параметров состояния на себя, причём параметр y принадлежит некоторому множеству C. Будем полагать, что весовая функция («критерий оптимальности») является некоторой функцией состояния $f(x_1,...,x_n)$. Рассматривая дискретные процессы, состоящие из конечного числа шагов, проиндексируем весовую функцию номером шага и, вводя начальной состояние $(x_1^0,...,x_n^0)$ в качестве аргумента в предположении оптимальности поведения, получим $f_N(x_1^0,...,x_n^0) = f(x_1^N,...,x_n^N)$, где $(x_1^N,...,x_n^N)$ - состояние системы после N-ого шага. Задача определения оптимальной стратегии при начальном состоянии $(x_1^0,...,x_n^0)$ определяется теперь как задача определения такой последовательности преобразований $F_{y_1},...,F_{y_N}, (x_1^1,...,x_n^1) = F_{y_1}(x_1^{i-1},...,x_n^{i-1})$, $i = 1,2,...,n$, что величина $f_N(x_1^0,...,x_n^0)$ будет максимально возможной.

Основное функциональное уравнение.

Заметив, что $f_N(x_1^0,...,x_n^0) = f_{N-1}(x_1^1,...,x_n^1) = f_{N-1}(F_{y_1}(x_1^0,...,x_n^0))$ и учитывая требование максимизации, получим:

$$f_1(x_1^0,...,x_n^0) = f(x_1^1,...,x_n^1) \qquad (1)$$

$$f_N(x_1^0,...,x_n^0) = \sup_{y \in C} f_{N-1}(F_y(x_1^0,...,x_n^0)). \qquad (2)$$

Данное уравнение является центральным рекуррентным соотношением, применяемым при решении задач для

детерминированных дискретных процессов методом динамического программирования.

Следует отметить, что при выводе уравнения мы воспользовались следующими характеристиками задач, допускающих решение методами динамического программирования:

- Любое состояние системы однозначно определяется некоторым конечным числом «параметров состояния».
- Результатом принятия того ли иного решения на каждом шаге процесса является изменение параметров состояния.
- Предыстория (т.е. набор предыдущих параметров состояния) не имеет никакого значения при определении последующих решений (преобразования $F_y : D \to D$ могут зависеть лишь от текущего состояния).
- Целью процесса является максимизация некоторой функции параметров состояния.

Основной подход, применяемый при решении оптимизационных задач динамическим программированием, сконцентрирован в следующем эвристическом правиле («принцип оптимальности», см. [1]): *оптимальное поведение обладает тем свойством, что, каковы бы ни были первоначальное состояние и решение в начальный момент, последующие решения должны составлять оптимальное поведение относительно состояния, получающегося в результате первого решения.*

Уравнение (1), в частности, является строгой переформулировкой принципа оптимальности для детерминированного дискретного процесса.

Задача 2 (стохастический процесс).
Допустим, как и ранее, что состояние системы определяется n-мерным вектором $(x_1,...,x_n) \in D$, где D – некоторая область. Как и выше допустим существование множества параметров преобразования состояния $y \in C$ (т.е. каждое значение y определяет некоторое преобразование параметров состояния). Предположим, однако, что результат применения преобразования y не определяется точно, но задаётся некоторым распределением вероятности:

$$dP(y; x_1,\dots,x_n; z_1,\dots,z_n) = p(y; x_1,\dots,x_n)\,dz_1\dots dz_n, \qquad (3)$$

где $(x_1,\dots,x_n),(z_1,\dots,z_n)$ - начальное и конечное состояние соответственно. Ввиду того, что результат каждого шага не определён точно, в соответствии со сказанным выше об использовании «средних» величин используем в качестве критерия оптимальности значение математического ожидания некоторой весовой функции $\phi(x_1,\dots,x_n)$.

Основное функциональное уравнение.

Вводя, как и выше, функцию $f_N(x_1^k,\dots,x_n^k)$ (математическое ожидание), получим:

$$f_1(x_1^2,\dots,x_n^2) = \sup_{y\in C}\int_D \phi(z_1,\dots,z_n)\,dP(y; x_1^k,\dots,x_n^k; z_1,\dots,z_n) \qquad (4)$$

и

$$f_{N+1}(x_1^2,\dots,x_n^0) = \sup_{y\in C}\int_D f_N(z_1,\dots,z_n)\,dP(y; x_1^0,\dots,x_n^k; z_1,\dots,z_n). \qquad (5)$$

В некоторых случаях, когда число шагов велико или заведомо неограниченно, целесообразно применять аналоги уравнений (2),(5) для бесконечно продолжающихся процессов:

$$f(x_1^2,\dots,x_n^2) = \sup_{y\in C} f(F_y(x_1^0,\dots,x_n^k)) \qquad (6)$$

$$f(x_1^2,\dots,x_n^0) = \sup_{y\in C}\int_D f(z_1,\dots,z_n)\,dP(y; x_1^0,\dots,x_n^0; z_1,\dots,z_n). \qquad (7)$$

Во введении (Глава I) приведено два примера детерминированных дискретных процессов. В следующих параграфах приводятся важные с точки зрения приложений стохастические дискретные процессы.

§2. Распределённые сетевые ресурсы.

Примером промышленного стохастического процесса является процесс организации доступа к распределённым сетевым ресурсам (см. [2]). Стохастичность данного процесса может определяться как

характером информации о доступности тех или иных ресурсов (например, возможна ситуация, когда местонахождение того или иного ресурса не известно точно, но имеются эвристические соображения о вероятности нахождения ресурса на том или ином носителе), так и особенностями самой физической системы (загруженность сети, «всплески» активности пользователей, вопросы надёжности). Рис.1 иллюстрирует типичную ситуацию, характеризующуюся наличием множества ранжированных носителей (предполагается, что индекс отражает вероятность нахождения ресурса на соответствующем носителе, причём 1 соответствует наибольшей вероятности), различной пропускной способностью каналов связи с носителями и объединением всех компонентов системы в глобальную сеть со звёздной топологией.

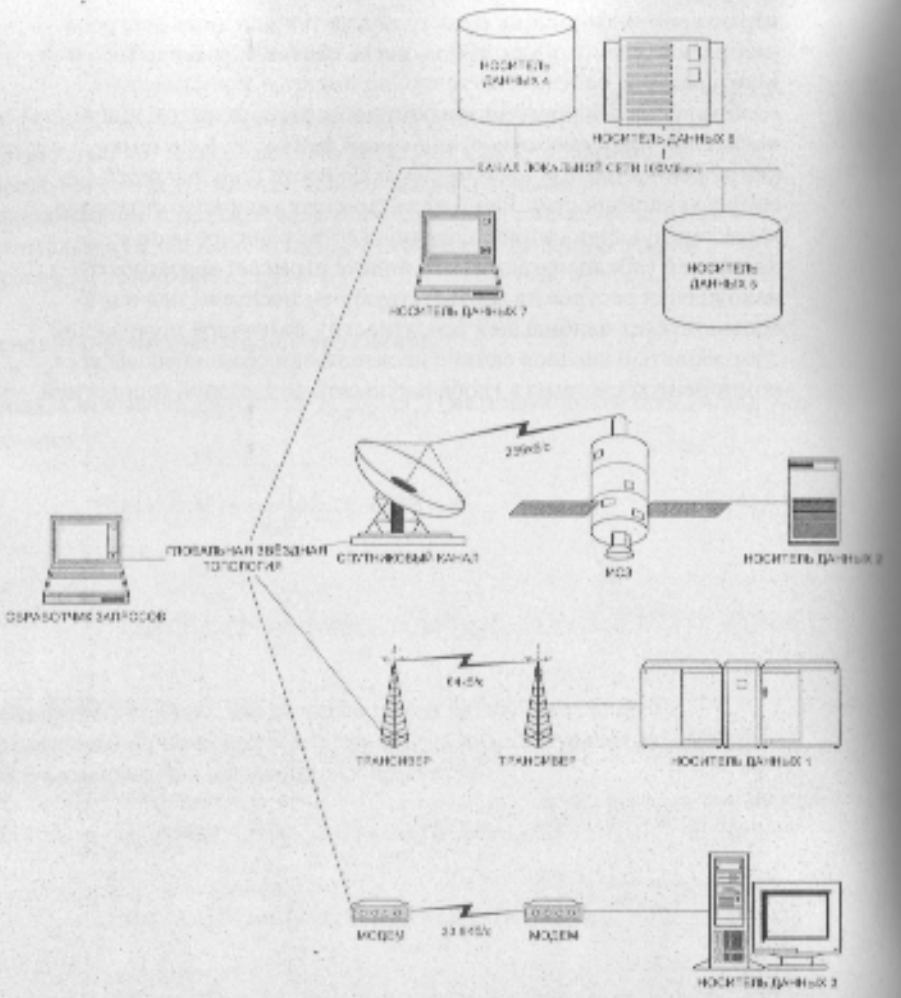

Рис. 1. Типичная конфигурация распределённых сетевых ресурсов.

Подобная схема применима как при описании сложных поисковых систем, так и многокомпонентных систем обработки данных (см. [2]). В настоящей работе рассматриваются две основные задачи: задача минимизации времени поиска сетевого ресурса и задача максимизации эффективности многозадачной обработки данных. Характерным для приложений является низкая пропускная

способность наиболее предпочтительных носителей, что на практике соответствует загруженности «популярных» обработчиков запросов.

Примером первой задачи является, скажем, определение адреса получателя электронной почты при использовании протокола SMTP. Вследствие того, что не все SMTP-сервера поддерживают релейные функции, зачастую отправитель вынужден определять маршрут самостоятельно, используя, к примеру, DNS-функции. Как известно, серверные имена в ресурсовых таблицах приводятся ранжированными по предпочтительности использования, что естественным образом приводит к вышеописанной ситуации.

Примером второй задачи является задача эффективной организации вычислительного процесса в многозадачном режиме. В этом случае максимизируется математическое ожидание количества операций, выполненных всей вычислительной системой за конечный промежуток времени.

Ниже обе задачи рассматриваются более подробно.

§3. Оптимизация поиска

Постановка 1.
Предположим, что на обработчик поступил запрос о поиске некоторой единицы данных в глобальной сети. Известно, что искомые данные могут находиться на любом из n носителей, причём вероятность нахождения ресурса на k-том носителе составляет $\alpha_k, k = 1,\dots,n$. Требуется максимизировать вероятность нахождения ресурса за произвольное число N шагов.

Рекуррентные соотношения.

Прежде всего заметим, что произвольная стратегия поиска определяется вектором $p_1,\dots,p_N, p_k \in \{1,\dots,n\}$, где p_k означает, что на k-том шаге поиск ведётся на носителе с номером p_k. Вероятность обнаружения искомых данных за N шагов при стратегии $p_1,\dots,p_N$, таким образом, равна

$$A_N(p_1,\dots,p_N) = \alpha_{p_1} + (1-\alpha_{p_1})\alpha_{p_2} + \dots + (1-\alpha_{p_1})(1-\alpha_{p_2})\dots(1-\alpha_{p_{N-1}})\alpha_{p_N} \qquad (1)$$

Отсюда,

$$A_{N-1}(p_1,\dots,p_{N-1}) = \alpha_{p_1} + (1-\alpha_{p_1})x_{p_1} + \dots + (1-\alpha_{p_1})(1-\alpha_{p_2})\dots(1-\alpha_{p_{N-1}})x_{p_{N-1}} =$$
$$= \alpha_{p_1} + (1-\alpha_{p_1})\left[\alpha_{p_2} + (1-\alpha_{p_2})x_{p_2} + \dots + (1-\alpha_{p_2})\dots(1-\alpha_{p_{N-1}})x_{p_{N-1}}\right] =$$
$$= \alpha_{p_1} + (1-\alpha_{p_1})A_{N-1}(p_2,\dots,p_{N-1})$$

Критерием оптимальности является $G_N = \max_{p_1,\dots,p_N} A_N(p_1,\dots,p_N)$, но легко видеть, что

$$G_{N+1} = \max_{p_1,\dots,p_{N+1}} A_{N+1}(p_1,\dots,p_{N+1}) = \max_{p_1,\dots,p_{N+1}} \left(\alpha_{p_1} + (1-\alpha_{p_1})A_N(p_2,\dots,p_{N+1})\right) =$$
$$= \max_{p_1} \max_{p_2,\dots,p_{N+1}} \left(\alpha_{p_1} + (1-\alpha_{p_1})A_N(p_2,\dots,p_{N+1})\right) = \max_{p_1}\left(\alpha_{p_1} + (1-\alpha_{p_1})G_N\right) \qquad (2)$$

Т.к. всегда $0 \le G_N \le 1$, то оптимальная стратегия в данном случае приводит к необходимости поиска на самом предпочтительном носителе. Множество индексов p_i, на котором ищется максимум с каждым шагом сужается (из него исключаются индексы, составляющие оптимальную стратегию для меньшего числа шагов).

Заметим, что хотя процесс и описывает события с неизвестным исходом, вероятность, превращённая в весовую функцию, позволила сформулировать задачу в терминах детерминированных процессов.

Постановка 2.
Предположим, что на обработчик поступил запрос о поиске некоторой единицы данных в глобальной сети. Известно, что искомые данные могут находиться на любом из n носителей, причём вероятность нахождения ресурса на k-том носителе составляет $\alpha_k, k = 1,\dots,n$. Предположим, что вероятность того, что в случае нахождения ресурса на носителе поиск окажется успешным (т.е. ресурс будет найден) равна $\beta_k, k = 1,\dots,n$. Требуется максимизировать вероятность нахождения ресурса за произвольное число N шагов.

Рекуррентные соотношения.

Произвольная стратегия поиска определяется, опять-таки, вектором $p_1,\dots,p_N, p_k \in \{1,\dots,n\}$, где p_k означает, что на k-том шаге поиск ведётся на носителе с номером p_k. Вероятность обнаружения искомых данных за N шагов при стратегии $p_1,\dots,p_N$ из множества индексов I, таким образом, равна

$$A_N(p_1,\dots,p_N) = \alpha_{p_1}\beta_{p_1} + (1-\alpha_{p_1}) \underset{p_2,\dots,p_N \in I\setminus\{p_1\}}{A_{N-1}}(p_2,\dots,p_N) +$$
$$+ (1-\beta_{p_1})x_{p_1} \underset{p_1,\dots,p_N \in I}{A_{N-1}}(p_1,\dots,p_N) \qquad (3)$$

Для критерия оптимальности $G_N(I) = \max_{p_1,\dots,p_N \in I} A_N(p_1,\dots,p_N)$ тогда получим:

$$G_{N+1} = \max_{p_1,\dots,p_{N+1} \in I} A_N(p_1,\dots,p_N) =$$
$$= \max_{p_1 \in I}\left\{\max_{p_2,\dots,p_{N+1} \in I\setminus\{p_1\}}\left[\alpha_{p_1}\beta_{p_1} + (1-\alpha_{p_1}) \underset{p_2,\dots,p_N \in I\setminus\{p_1\}}{A_{N-1}}(p_2,\dots,p_N)\right] + \right.$$
$$\left. + \max_{p_2,\dots,p_{N+1} \in I\setminus\{p_1\}}\left[(1-\beta_{p_1})x_{p_1} \underset{p_2,\dots,p_N \in I}{A_{N-1}}(p_2,\dots,p_N)\right]\right\} = \qquad (4)$$
$$= \max_{p_1 \in I}\left[\alpha_{p_1}\beta_{p_1} + (1-\alpha_{p_1})G_N(I\setminus\{p_1\}) + (1-\beta_{p_1})x_{p_1}G_N(I)\right]$$

причём $G_1(I) = \max_{p_1 \in I}\{\alpha_{p_1}\beta_{p_1}\}$. На каждом шаге необходимо вычислять и 2^n величин $G_N(I)$, где множества I являются всеми неупорядоченными подмножествами множества $\{1,\dots,N\}$. Формула (3) означает, что вероятность того, что искомый ресурс будет успешно найден при стратегии $p_1,\dots,p_N$, равна вероятности нахождения ресурса на носителе p_1, помноженной на вероятность успешного поиска, в сумме с условной вероятностью не нахождения ресурса на данном носителе, но успеха поиска $p_2,\dots,p_N$ при исключении p_1 и условной вероятностью неуспешного поиска ресурса на носителе p_1 при успехе стратегии поиска $p_2,\dots,p_N$ на том же множестве носителей.

§4. Оптимизация использования распределенных ресурсов

Постановка 1.
Пусть $x_k, k = 1,\dots,n$ - ресурс k-того носителя, $\alpha_k, k = 1,\dots,n$ - вероятность успешного использования части ресурсов носителя при запросе, $\beta_k, k = 1,\dots,n$ - часть ресурсов, используемая запрашивающим при успехе, причём $x_k \ge 0$, $1 \ge \beta_k, \alpha_k \ge 0$. Предполагается, что при каждом успешном использовании ресурса k-того носителя на носителе остаётся $(1-\beta_k)x_k$ запаса ресурса, а при неудаче дальнейшие действия прекращаются. Требуется сформулировать стратегию,

обеспечивающую максимальное математическое ожидание количества использованных ресурсов за произвольное число N шагов.

Рекуррентные соотношения.

Как и выше, стратегия определяется вектором $p_1,...,p_N, p_k \in \{1,...,n\}$, где p_k означает, что на k-том шаге работа ведётся на носителе с номером p_k. Пусть $f_N(u_1,...,u_n)$ обозначает (максимальное) математическое ожидание количества использованных ресурсов за N шагов при оптимальной стратегии при начальных количествах ресурсов на носителях равных $u_1,...,u_n$. Нетрудно видеть, что

$$f_1(u_1,...,u_n) = \max_{p\in\{1,...,n\}} \{\alpha_p \beta_p u_p\}. \qquad (1)$$

Далее заметим, что в случае выбора некоторого носителя k на $N+1$ шаге при оптимальности первых N шагов, математическое ожидание количества использованных ресурсов составит:

$$\tilde{f}^k(u_1,...,u_n) = \alpha_k[\beta_k u_k + f_N(u_1,...,u_{k-1},(1-\beta_k)u_k,u_{k+1},...,u_n)], \qquad (2)$$

следовательно,

$$f_{N+1}(u_1,...,u_n) = \max_{k\in\{1,...,n\}} \alpha_k[\beta_k u_k + f_N(u_1,...,u_{k-1},(1-\beta_k)u_k,u_{k+1},...,u_n)] \qquad (3)$$

-- частный случай уравнения (1.5).

Данное уравнение представляет наибольший интерес, так как является рекуррентным соотношением для стохастического процесса с широким применением.

Глава III. Существование и свойства решений

В настоящей главе формулируются и доказываются основные теоремы о существовании и поведении решений уравнений вида (II.4.3).

§1. Существование и единственность

Прежде всего, докажем теорему существования и единственности для бесконечношагового стохастического процесса.

Теорема 1.
Предположим, что в терминах Постановки II.4.1 выполнены следующие условия:

$$0 \le \alpha_k < 1, k = 1,...,n, \qquad (1)$$
$$0 \le \beta_k < 1, k = 1,...,n.$$

Тогда в любом прямоугольнике $[0,X_1]\times...\times[0,X_n]$ существует и единственно ограниченное непрерывное решение уравнения

$$f(u_1,...,u_n) = \max_{k\in\{1,...,n\}} \alpha_k[\beta_k u_k + f(u_1,...,u_{k-1},(1-\beta_k)u_k,u_{k+1},...,u_n)]. \qquad (2)$$

Доказательство.

Докажем теорему методом последовательных приближений. Построим последовательность

$$f_1(u_1,...,u_n) = \max_{k\in\{1,...,n\}} \{\alpha_k \beta_k u_k\} \qquad (3)$$

$$f_{N+1}(u_1,...,u_n) = \max_{k\in\{1,...,n\}} \{\alpha_k[\beta_k u_k + f_N(u_1,...,u_{k-1},(1-\beta_k)u_k,u_{k+1},...,u_n)]\}. \qquad (4)$$

Введём целочисленную функцию $k(N;u_1,...u_n)$ равную индексу, при котором достигается максимум в правой части (3) (для простоты будем опускать аргументы $(u_1,...,u_n)$). Далее, обозначая выражение в фигурных скобках формулы (4) $S_k(f)$[1], получим:

$$\begin{aligned} f_{N+1}(u_1,...,u_n) &= S_{k(N)}(f_N) \ge S_{k(N+1)}(f_N), \\ f_{N+2}(u_1,...,u_n) &= S_{k(N+1)}(f_{N+1}) \ge S_{k(N)}(f_{N+1}) \end{aligned} \qquad (5)$$

Оценим разность двух последовательных членов приближения из левой части (5):

[1] Следует заметить, что это – не функция, а функционал, т.к. аргументы функции f, входящей в выражение, зависят от k.

$$
\begin{aligned}
&|f_{N+2}(u_1,\dots,u_n)-f_{N+1}(u_1,\dots,u_n)|\le\\
&\le\max\{|S_{k(N)}(f_N)-S_{k(N)}(f_{N+1})|,|S_{k(N+1)}(f_N)-S_{k(N+1)}(f_{N+1})|\}\le\\
&\le\max_{i=1,\dots,n}\{|S_i(f_N)-S_i(f_{N+1})|\}\le\\
&\le\max_{i=1,\dots,n}\{\alpha_i|f_N(u_1,\dots,u_{i-1},(1-\beta_i)u_i,u_{i+1},\dots,u_n)-\\
&-f_{N+1}(u_1,\dots,u_{i-1},(1-\beta_i)u_i,u_{i+1},\dots,u_n)|\}
\end{aligned}
\qquad (6)
$$

Отсюда

$$
\Delta_{N+1}(u_1,\dots,u_n)\le\max_{k=1,\dots,n}\{\alpha_k\}\Delta_N(u_1,\dots,u_n),
\qquad (7)
$$

где

$$
\Delta_N(x_1,\dots,x_n)=\max_{0\le u_k\le x_k,\,k=1,\dots,n}|f_N(u_1,\dots,u_n)-f_{N+1}(u_1,\dots,u_n)|.
\qquad (8)
$$

Ввиду равномерной сходимости ряда $\sum_{N=1}^{\infty}\Delta_N(x_1,\dots,x_n)$ на любом прямоугольнике $[0,X_1]\times\dots\times[0,X_n]$ и непрерывности первого приближения, получаем существование непрерывного ограниченного решения.

Докажем единственность. Пусть f и g – два решения уравнения (2), $k(f)$ и $k(g)$ соответствующие индексные функции (аргументы $(u_1,\dots,u_n)$ опущены). Имеем:

$$
\begin{aligned}
f(u_1,\dots,u_n)&=S_{k(f)}(f)\ge S_{k(g)}(f),\\
g(u_1,\dots,u_n)&=S_{k(g)}(g)\ge S_{k(f)}(g)
\end{aligned}
\qquad (9)
$$

Аналогично (6) получим:

$$
\begin{aligned}
&|f(u_1,\dots,u_n)-g(u_1,\dots,u_n)|\le\\
&\le\max\{|S_{k(f)}(f)-S_{k(f)}(g)|,|S_{k(g)}(f)-S_{k(g)}(g)|\}\le\\
&\le\max_{i=1,\dots,n}\{|S_i(f)-S_i(g)|\}\le\\
&\le\max_{i=1,\dots,n}\{\alpha_i|f(u_1,\dots,u_{i-1},(1-\beta_i)u_i,u_{i+1},\dots,u_n)-\\
&-g(u_1,\dots,u_{i-1},(1-\beta_i)u_i,u_{i+1},\dots,u_n)|\}
\end{aligned}
\qquad (10)
$$

Применяя это неравенство произвольное число раз, убеждаемся в тождестве $f\equiv g$. Теорема полностью доказана.

§2. Свойства решений.

При решении (1.2) встаёт вопрос об индексе $k(u_1,\dots,u_n)$, при котором достигается максимум в правой части, как функции переменных $(u_1,\dots,u_n)$. Предполагается, что в ходе вычислительного процесса должны храниться значения этой функции. С вычислительной точки зрения это может означать в общем случае, что значения индекса должны храниться для всех узлов некоторой сетки. Подобные данные весьма громоздки и выдвигают неоправданно высокие требования к оперативной памяти. Между тем, информация о геометрии областей, где функция $k(u_1,\dots,u_n)$ принимает постоянное значение, может позволить сократить накладные вычислительные расходы. Область всех значений $(u_1,\dots,u_n)$ разделяется на не более n подмножеств, в каждом из которых функция $k(u_1,\dots,u_n)$ принимает постоянное значение из диапазона $\{1,\dots,n\}$. Для определения оптимальной стратегии, таким образом, достаточно будет знать расположение границ упомянутых подмножеств.

Справедлива

Теорема 2 (геометрия областей одинаковых оптимальных поведений).
Функция $k(u_1,\dots,u_n)$ для уравнения (2) при условиях (1) и $u_k\ge 0, k=1,\dots,n$ определяется следующим образом: $k(u_1,\dots,u_n)$ равно индексу i, при котором выражение

$$
\frac{\alpha_i\beta_i}{1-\alpha_i}u_i
$$

максимально при фиксированных $(u_1,\dots,u_n)$. Области постоянного значения функции $k(u_1,\dots,u_n)$ однородны и ограничены гиперплоскостями

$$
\frac{\alpha_i\beta_i}{1-\alpha_i}u_i=\frac{\alpha_j\beta_j}{1-\alpha_j}u_j.
\qquad (1)
$$

Все области имеют общий участок границы вдоль луча

$$\frac{\alpha_1\beta_1}{1-\alpha_1}u_1 = \ldots = \frac{\alpha_n\beta_n}{1-\alpha_n}u_n, \qquad (2)$$

в точках которого любой выбор индекса приводит к оптимальной стратегии[2].

Доказательство.

Однородность рассматриваемых множеств следует из (1.3), (1.4) и сходимости построенной последовательности к решению.

Ограничим множества индексов двумя, скажем *i* и *j*, и будем искать максимум в (1.2) лишь по указанным индексам. На координатной гиперплоскости $u_j = 0$ оптимальным окажется выбор индекса *i* и наоборот. Действительно,

$$\begin{aligned} &f(u_1,\ldots,u_i,\ldots,u_{j-1},0,u_{j-1},\ldots,u_n) = \\ &= \max_{k\in\{i,j\}} \alpha_k\left[\beta_k u_k + f(u_1,\ldots,u_{k-1},(1-\beta_k)u_k,u_{k+1},\ldots,u_n)\right] = \\ &= \max\left\{\alpha_i\left[\beta_i u_i + f(u_1,\ldots,u_{i-1},(1-\beta_i)u_i,u_{i+1},\ldots,u_{j-1},0,u_{j+1},\ldots,u_n)\right]\right. \\ &\left.\alpha_j f(u_1,\ldots,u_{i-1},(1-\beta_i)u_i,u_{i+1},\ldots,u_{j-1},0,u_{j+1},\ldots,u_n)\right\} = \\ &= \alpha_i\left[\beta_i u_i + f(u_1,\ldots,u_{i-1},(1-\beta_i)u_i,u_{i+1},\ldots,u_{j-1},0,u_{j+1},\ldots,u_n)\right] \end{aligned} \qquad (3)$$

Ввиду непрерывности функции (1.2) заключаем, что существует такое (малое) δ, что в области $u_j < \delta u_i$ индекс *i* всё ещё будет оптимальной стратегией. Применяя к функции *f* стратегию *j*, а затем *i*, получим величину

$$\begin{aligned} &f^{i,j} = \alpha_j\beta_j u_j + \alpha_j\alpha_i\beta_i u_i + \\ &+ \alpha_j\alpha_i f(u_1,\ldots,u_{i-1},(1-\beta_i)u_i,u_{i+1},\ldots,u_{j-1},(1-\beta_j)u_j,u_{j+1},\ldots,u_n) \end{aligned} \qquad (4)$$

Аналогично, применяя к функции *f* стратегию *i*, а затем *j*, получим величину

$$\begin{aligned} &f^{j,i} = \alpha_i\beta_i u_i + \alpha_i\alpha_j\beta_j u_j + \\ &+ \alpha_i\alpha_j f(u_1,\ldots,u_{i-1},(1-\beta_i)u_i,u_{i+1},\ldots,u_{j-1},(1-\beta_j)u_j,u_{j+1},\ldots,u_n) \end{aligned} \qquad (5)$$

Величина (4) превосходит (5) в области первого квадранта, где

[2] Функция $k(u_1,\ldots,u_n)$, вообще говоря, определена почти всюду, а на множестве лебеговой меры 0 существует произвол в выборе значений: на луче любые значения, на гиперплоскостях любые из двух, на многообразиях размерности m – n-m+1 значений и т.д.

$$\frac{\alpha_j\beta_j}{1-\alpha_j}u_j > \frac{\alpha_i\beta_i}{1-\alpha_i}u_i \qquad (6)$$

и наоборот во внутренности дополнительной области первого квадранта. Рассмотри точку $(u_1,\ldots,u_n)$ в области

$$\frac{\alpha_j\beta_j}{1-\alpha_j}u_j < \frac{\alpha_i\beta_i}{1-\alpha_i}u_i \qquad (7)$$

причём такую, что $u_j > \delta u_i, (1-\beta_j)u_j < \delta u_i$ (т.е. точка находится между гиперплоскостями $u_j = \delta u_i$ и $u_j = (1-\beta_j)^{-1}\delta u_i$). Докажем, что в этой точке из двух стратегий *i* и *j* предпочтительной окажется *i*. Рассуждая от противного, предположим, что предпочтителен индекс *j*. Тогда на следующем шаге должен быть выбран индекс *i*, так как после применения в выбранной выше точке $(u_1,\ldots,u_n)$ стратегии *j* оставшиеся ресурсы

$$(u_1,\ldots,u_{i-1},u_i,u_{i+1},\ldots,u_{j-1},(1-\beta_j)u_j,u_{j+1},\ldots,u)$$

обязательно попадут в область $u_j < \delta u_i$, где предпочтительным является индекс *i*. В таком случае, в точке $(u_1,\ldots,u_n)$ будем иметь:

$$f^{i,j} > f^{j,i} \text{ (т.е. (4) больше (5))}$$

причём $(u_1,\ldots,u_n)$ принадлежит множеству

$$\frac{\alpha_j\beta_j}{1-\alpha_j}u_j < \frac{\alpha_i\beta_i}{1-\alpha_i}u_i, \qquad (8)$$

где (5) должно превосходить (4) – противоречие. Далее аналогично рассматривается произвольная точка $u_j > (1-\beta_j)^{-1}\delta u_i, u_j < (1-\beta_j)^{-2}\delta u_i$ и т.д., охват зоны предпочтительности стратегии *i* достигает гиперплоскости (1). Таким образом, из двух стратегий *i* и *j* в области (8) предпочтительной оказывается *i*, а в области (6) – стратегия *j*. Остальные утверждения теоремы являются прямыми следствиями доказанного.

Следствие 1.
Рассмотрим рекуррентное соотношение

$$f_{N+1}(u_1,...,u_n) = \max_{k\in\{1,...,n\}} \alpha_k [\beta_k u_k + f_N(u_1,...,u_{k-1},(1-\beta_k)u_k,u_{k+1},...,u_n)] \quad (9)$$

в прямоугольнике $[0,X_1]\times...\times[0,X_n]$,

$$0 \le \alpha_k < 1, k = 1,...,n, \quad (10)$$
$$0 \le \beta_k < 1, k = 1,...,n,$$

причём $f_0(u_1,...,u_n)$ непрерывна в $[0,X_1]\times...\times[0,X_n]$. Для каждого шага имеются n областей решения, которые стремятся к областям для решения (1.2).

Глава IV. Алгоритмы решения

В данной главе приводится описание алгоритмов решения задач об оптимальном использовании распределённых сетевых ресурсов в обеих постановках, а также описывается разработанное программное обеспечение, приведённое в приложении.

§1. Оптимальный поиск

В терминах постановки II.3.2 имеем:

$$G_{N+1}(I) = \max_{p_1\in I}[\alpha_{p_1}\beta_{p_1} + (1-\alpha_{p_1})G_N(I\setminus\{p_1\}) + (1-\beta_{p_1})\alpha_{p_1}G_N(I)] \quad (1)$$

Функция $G_N(I)$ равна вероятности успешного поиска в множестве I сетевых ресурсов. Заметим, что всего необходимо рассматривать 2^n всевозможных множеств I, где n – общее число сетевых ресурсов. Однако, использование бинарного представления целых чисел и операций порязрядного сдвига позволяют осуществить эффективную алгоритмизацию решения:

Описываются массивы $G_0[0..2^{n-1}]$, $G_1[0..2^{n-1}]$, инициализированные нулями.
N раз в цикле выполняется:

для $i = 0, i < 2^n, i++$ {
    вещ m=0
    для $k = 1, i \le n, k++$
        если (i>>(k-1)&1 = 1)
            m = max{m, $\alpha_k\beta_k + (1-\alpha_k)G_0[i \,\&\, \overline{(1<<(k-1))}] + (1-\beta_k)\alpha_k G_0[i]$}
    $G_1[i] = m$
}
$G_0 = G_1$

Искомый результат равен $G_0[2^{n-1}]$.
Сложность алгоритма составляет $O(N2^n)$, т.е. сложность экспоненциальна по количеству ресурсов и линейна по числу шагов. Заметим, что экспоненциальны как временная сложность, так и затраты памяти.

Следует отметить, что экспоненциальная сложность алгоритмов – случай довольно типичный для задач динамического

программирования и напрямую связана со структурой процессов динамического программирования. Однако, приведённый выше пример демонстрирует также эффективность применения методики динамического программирования: при альтернативной методике перебора сложность алгоритма составила бы $O(n^N)$, т.е. возрастала бы экспоненциальна с возрастанием числа шагов, в то же время наш алгоритм имеет линейную сложность по основному переменному параметру N – числу шагов.

Программная реализация приведённого выше алгоритма приведена в Приложении 1.

§2. Использование ресурсов

В терминах постановки II.4.1 имеем:

$$f_{N+1}(u_1,\ldots,u_n) = \max_{k\in\{1,\ldots,n\}} \alpha_k \left[\beta_k u_k + f_N(u_1,\ldots,u_{k-1},(1-\beta_k)u_k,u_{k+1},\ldots,u_n)\right]$$

В практических приложениях приходится рассматривать задачи с неограниченным числом шагов, выражающем продолжительные по времени стохастические процессы использования распределённых ресурсов. Основываясь на теореме III.2.2 и следствии III.2.1, заменим конечношаговый процесс бесконечношаговым. Используя теорему III.2.2, получаем следующий алгоритм:
Задан массив: $u_1, u_2, \ldots, u_n$
В бесконечном цикле повторять:
Цикл {
цел j = 1
для $i = 2, i \le n, i++$
если $\frac{\alpha_i\beta_i}{1-\alpha_i}u_i < \frac{\alpha_j\beta_j}{1-\alpha_j}u_j$ то j=i
вывести j
использовать сетевой ресурс № j
<зависящее от реализации использование>
если <успех> то $u_j = (1-\beta_j)u_j$
иначе прекратить цикл
}

Временная сложность алгоритма составляет $O(Nn)$, где N – число шагов.

Программная реализация алгоритма приведена в Приложении 2. Предлагаемая реализация основывается на моделировании сетевых ресурсов в режиме реального времени в преемптивной многозадачной операционной среде.

Приложение 1.

```
////////////////////////////////////////////////////////////////////////
//
//      STANDARD OBJECTS LIBRARY SAMPLE APP
//
//      (C) 1993-2000
//
//      DYNAMIC PROGRAMMING SAMPLE 1
//
//      FILENAME:       SOC.CPP (SOC.H)
//
////////////////////////////////////////////////////////////////////////


#include "stdafx.h"
#include "soc.h"


OSTREAM serr("", _OUT_, _CON_);

const n = 5;
const SETS = (1<<n);

int main(int argc, char* argv[])
{       OSTREAM out("", _OUT_, _CON_);
        OSTREAM in("", _IN_, _CON_);
        POSTR s;

        double G0[SETS], G1[SETS];
        double alpha[n]={0.05, 0.21, 0.15, 0.1, 0.09};
        double beta[n]= {0.99, 0.6, 0.98, 0.89, 0.97};
        int num = 0, i=0, j=0, k=0;


        out << "No of steps=";
        in >> s;
        sscanf(s, "%d", &num);

        for (j=0;j<num;j++) {
                for (i=0;i<SETS;i++) {
```



```
                        double m = 0;
                        for (k=1;k<=n;k++)
                                if ((i>>(k-1))&1) m = max(m,
alpha[k]*beta[k]+
                                        (1-alpha[k])*G0[i&((1<<(k-1))^(1<<(n-
1)))]+
                                        (1-beta[k])*alpha[k]*G0[i]);
                        G1[i]=m;
                }
                for (i=0;i<SETS;i++) G0[i]=G1[i];
        }

        char __buff[MAX_STRING];
        sprintf(__buff, "%f", G0[(1<<n)-1]);
        out << "max probability = " << __buff << "\r\n";

        return 0;
}
```