Лекция 9

Многокритериальные задачи принятия решений.

Оценка операций по многим критериям.

Во многих задачах при моделировании критерия приходится назначать много критериев. Оказывается, что изменение одного критерия, несёт за собой изменение другого критерия. Существует принципиальная трудность в оценке двух или более вариантов. Особенно если их оценивать безусловно.

Векторная оптимизация:

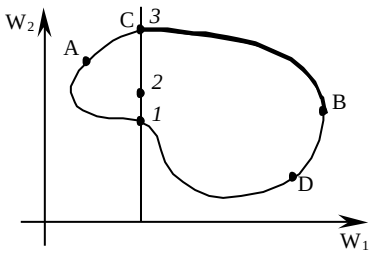
1) Безусловная оптимизация, когда пытаются определить безусловно лучшее решение, но после этапа безусловной оптимизации можно отсеять заведомо невыгодные решения, т. е. безусловно сравнённые и худшие, поэтому несравнимые, но и не плохие остаются, и мы получаем эффективное множество – множество Парето.

2) Для нахождения наилучшего решения приходится вводить некоторые условия, например, предпочтения или приходится выбирать из всех важных критериев, какой из них самый ценный.

Определение множества Парето

Непрерывный случай.

Если $X \in R \Rightarrow \overline{W} = f(X_1, X_2, \ldots, X_n)$ – множество точек, в котором небольшое улучшение X влечёт за собой небольшое изменение W. Пусть есть два показателя (W_1, W_2).



Сравним безусловную пару вариантов: третий безусловно лучше второго и первого. На дуге AB находятся лучшие точки при $W_1 = const$, а на дуге CD лучшие точки при $W_2 = const$. Поэтому точками множества Парето будут являться точки находящиеся на дуге CD.

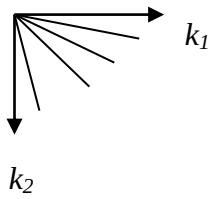
Дискретный случай.

Существует несколько алгоритмов нахождения множества Парето для дискретного множества точек. Они зависят от того, в каком виде задано множество вариантов критериев. В практических заданиях приходится сначала получать само множество критериев.

Существует два основных метода нахождения множества Парето:

1) Метод отбора конусом. Конус – часть телесного угла, ограниченной области, соответствующая направлениям оптимизации конкретного критерия.

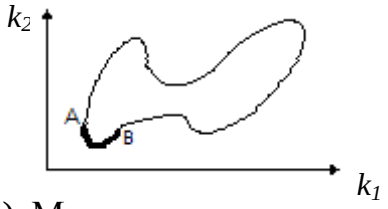
   Пусть для двух критериев: $k_1 \uparrow$ и $k_2 \downarrow$



Если конус установить на границу критериального множества и обойти все точки границы, то: если ни один луч не пересек область, то вершина конуса стоит на точке Парето.

Пример: $k_1 \to \min$

$k_2 \to \min$



2) Метод прямоугольников.

Запишем алгоритм для случая, когда $k_1 \to \min$ и $k_2 \to \min$:

1) Фиксируем самые левые точки, если их несколько, выбираем среди них нижнюю.
2) Через эту точку проводим вертикаль и горизонталь.
3) Фиксируем самые нижние точки, если их несколько – выбираем самую левую. Проводим вертикаль и горизонталь.
4) Выбрасываемые точки являются точками множества Парето.
5) Отбрасываем все точки, лежащие на границе прямоугольника.
6) К внутренним точкам прямоугольника применим алгоритм с пункта №1.

Методы условной оптимизации.

После нахождения множества Парето, если количество точек в нём ≥ 2, встаёт вопрос о выборе единственной точки, которую необходимо выбрать для проведения операций. Так как точки на множестве Парето отбираются так, что каждая из них лучше по одному критерию, но хуже по другому, то совместное улучшение по двум критериям невозможно. Условная оптимизация предполагает введение условия согласованности в компонентах критерия. Их существует четыре вида:

1) Метод скаляризации. Здесь формируется некоторая скалярная функция многих переменных:

$$W_0 = f(W_1, W_2, \ldots, W_k)$$

это скалярная величина. Наиболее распространённый метод – метод скользящей суммы:

$$W_0 = \sum_{i=1}^{k} \alpha_i W_i$$

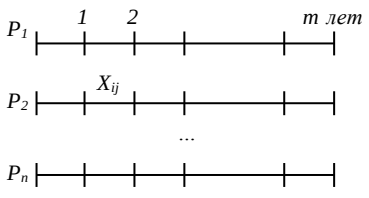
α_i – вес каждой компоненты критерия.

2) Метод главного критерия. Критерии располагаются в порядке убывания важности: W_1 объявляется собственным критерием, а остальные становятся управляемыми переменными:

$$W_1 \to \max$$
$$\left.\begin{array}{l} W_2 \le C_3 \\ W_3 \ge C_3 \\ \ldots \\ W_k \le C_k \end{array}\right\}$$

Такой метод чаще всего используется при оптимизации технических систем (в системах оптимизации и проектирования).

3) Метод уступок. Располагаем критерии в порядке убывания важности: $W_1, W_2, \ldots$. Считаем критерий W_1, а остальные отбрасываем и вычисляем $W_{1\max}$. Назначается уступка на критерий $W_1 - \Delta W_1$, которую мы готовы отдать в пользу других критериев. Далее проделываем то же самое для всех остальных критериев.



$$W_2 \to \max$$
$$W_{1\max} - \Delta W_1 < W_1 < W_{1\max}$$

далее:

$$W_3 \to \max$$
$$W_{1\max} - \Delta W_1 < W_1 < W_{1\max}$$
$$W_{2\max} - \Delta W_2 < W_2 \le W_{2\max}$$

и т. д.

Очевидно, успех такого метода зависит от того, насколько тупой максимум. Процедура может быть повторена для других ΔW.

4) Метод последовательной оптимизации. В некоторых случаях критерии системы не слишком связаны друг с другом. Поэтому, сначала оптимизируют систему по важнейшему показателю, отбросив все остальные, затем по второму, и т. д. Затем прогоняют оптимизацию в другом порядке.

Динамическое программирование.

Это особый метод, он специально приспособлен для оптимизации динамических задач, в которых операция состоит из элементов, сильно влияющих друг на друга. ДП связано с именем Ричарда Беллмана, который сформулировал принцип Беллмана. Он позволяет существенно сократить перебор решений в многоэтапных нелинейных задачах.

Рассмотрим экономическую задачу распределения ресурсов: пусть есть начальный капитал k_0. Его можно потратить на несколько предприятий $P_1, P_2, \ldots, P_n$. X_{ij} – количество средств вкладываемых в $i-$ом году, в $j-$ое предприятие. В результате получается эффект:

$$W_{ij} = f\left(X_{ij}\right)$$

В общем случае это не линейная функция.

$$W = \sum f\left(X_{ij}\right) \to \max$$
$$\sum X_{ij} \le k_0$$

Так как функция W нелинейная, то получаем задачу нелинейного программирования. Решать её сложно, кроме того, часто X_{ij} – дискретные значения. Вопрос: нельзя ли решить задачу последовательно, т. е. найти оптимальное вложение для первого года, второго и т. д. т. е. провести последовательную оптимизацию. В большинстве задач так нельзя, т. к. то, что мы решили оказывает влияние на последующие шаги, например:

> Бег на 800 метров. Каждый бегун имеет запас энергии, который он тратит на каждые 100 метров. t_i – время на i – й 100 метров. $\Sigma t_i(x_i) \to \min$; $\Sigma x_i \le x_0$. Оказывается, на первых 100 метров бегун будет обеспечивать минимальное время.

Принцип оптимальности Беллмана

Принцип оптимальности Беллмана ставит вопрос о том, что такое оптимальность отдельного элемента системы с точки зрения оптимальности всей системы. Принимая решение на отдельном этапе, мы должны выбирать управление на этом этапе с прицелом на будущее, т. к. нас интересует результат в целом за все шаги. Беллман предложил рассматривать величину выигрыша от $i-$го шага и до конца, если $i-$ый шаг начинается с некоторого состояния S. Такую величину называют условным оптимальным выигрышем $W_i(S)$.

Тогда, принимая решение на $i-$ шаге, мы должны выбрать X_i так, чтобы условный оптимальный выигрыш был максимальным от $i-$шага и до конца.

Определение: оптимальность в малом понимается через оптимальность в большом.

Любой процесс имеет где-то окончание, т. е. имеет горизонт планирования. Тогда последний этап «не имеет будущего». Вот именно его можно оптимизировать только из условий данного этапа. После этого приступают к оптимизации $(m-1)-$го этапа. Выбирают такое X_{m-1}, чтобы при применении этого X_{m-1} его внести в управление последнего шага. При этом мы задаём состояние, с которого начинается $(m-1)-$ый шаг. Поэтому функцию $W_i(S)$ называют условным оптимальным выигрышем. Таким образом, процесс оптимизации разворачивается от конца к началу, и начальное состояние становится известно. Принцип Беллмана нашёл

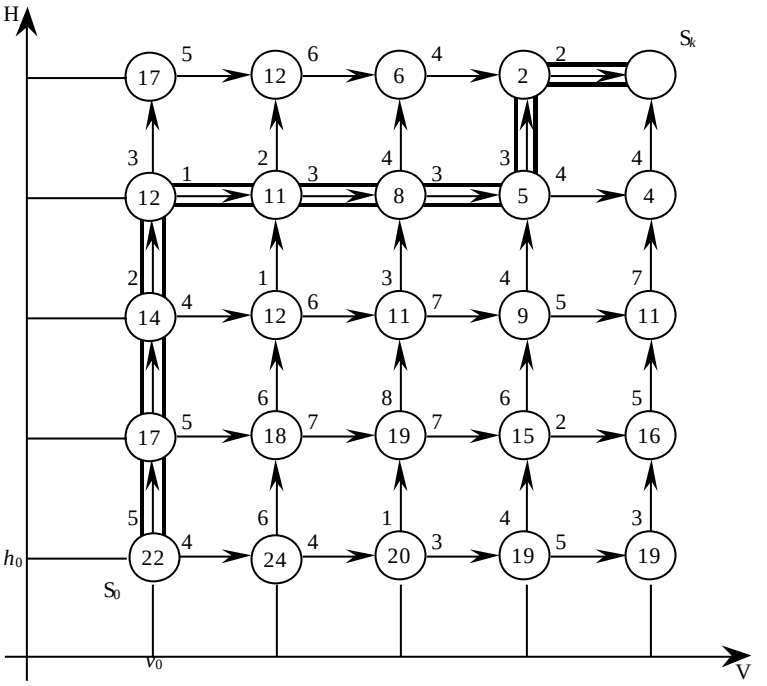
применение в методе программно-целевого планирования (любое действие планируется некоторым проектом).

Задача о наборе высоты и скорости летательного аппарата.

Летательный аппарат находится на высоте h_0 и летит со скоростью v_0. Необходимо перевести его на высоту h_1 со скоростью v_1. Причём $h_1 > h_0,\ v_1 > v_0$. Разобьём участок от h_0 до h_1 на n частей:

$$\Delta h = \frac{h_1 - h_0}{n}$$

$$\Delta v = \frac{v_1 - v_0}{m}$$



Известен расход горючего при переводе системы на Δh при $v = const$ и на Δv при $h = const$. Таким образом, из каждого состояния есть лишь два управления.

За каждое такое управление получим расход горючего (выигрыш и потери). Суммарные потери равны сумме всех выигрышей на всех шагах. Дойдя до конца и получив 22, мы получим минимальную величину потерь горючего.

Любая задача, сводится к поиску минимального пути в графе и решается методом динамического программирования.

Функциональное уравнение Беллмана.

Назовём состоянием системы вектор координат: $S = (\xi_1, \xi_2, ..., \xi_L)$. В некоторых задачах состояние – одна величина. Тогда работу системы можно представить как движение в фазовом пространстве – пространстве состояний. Назовём шаговым управлением – управление на $i-$ом шаге. Рассмотрим процесс управления системой за m шагов. Функция $W_i(S, U_i)$ называется выигрышем на i-ом шаге. Здесь $S-$состояние перед $i-$ым шагом, а U_i- управление на $i-$ом шаге.

Величина $W_i(S, U_i)$ должна быть известна до начала динамического программирования. Если состояние перед $i-$ым шагом было S и мы приняли какое-то управление U_i, то система перейдёт в новое состояние $S' = \varphi_i(S, U_i)$. Эта функция должна быть так же известна. Если эти функции не заданы, то их надо сформулировать. Введём $W_i(S)-$условный оптимальный выигрыш. Это выигрыш на всех этапах от i до конца, если $i-$ый шаг начинается с состояния S.

Рассмотрим m шагов. Начнём с $(i+1)-$го шага. Мы системой

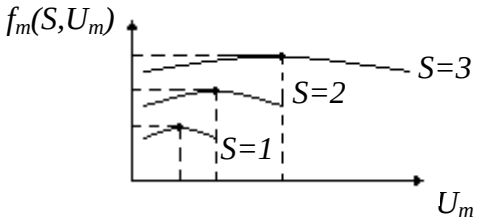
управляем оптимально, величина оптимального выигрыша $W_{i+1}(S')$. На $i-$ом шаге - любое управление. $\tilde{W}_i(S)-$неоптимальный выигрыш, т. к. на $i-$ом шаге мы применяем управление U_i.

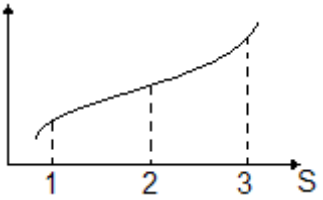
$$W_i(S)=\max_{U_i}\{W_i(S,U_i)+W_{i+1}(S')\};\qquad S'=\varphi_i(S,U_i)$$

$$\underset{\text{неизвестно}}{W_i(S)}=\max_{U_i}\left\{\underset{\text{известно}}{W_i(S,U_i)}+\underset{\text{неизвестно}}{W_{i+1}[\varphi_i(S,U_i)]}\right\}$$

Это функциональное уравнение Беллмана. Для использования уравнения Беллмана начинают с конца:

1) $i=m$





$$W_m(S)=\max_{U_m}\{f_m(S,U_m)\}.$$

2) $i=m-1$

$$W_{m-1}(S)=\max_{U_{m-1}}\{f_{m-1}(S,U_{m-1})+W_m[\varphi_{m-1}(S,U_{m-1})]\}$$

Итак, идя от конца к началу, мы получаем последовательно:

$W_m(S),W_{m-1}(S),...,W_1(S)$

$U_m(S),U_{m-1}(S),...,U_1(S)$

Придя в начальное состояние $W_1(S)$, мы можем подставить $S = S_0$ и $W_1(S_0) = W_{max}$ – это безусловный выигрыш.

Теперь необходимо получить, идя от начала к концу по цепочке, безусловно оптимальное уравнение:

$U_1(S)=U_1(S_0)=U_1^*$

$\varphi_1(S,U_1)=\varphi(S_0,U_1^*)=S_1^*$

$U_2(S)=U_2^*$

$\varphi_2(S,U_2)=\varphi(S_1^*,U_2^*)$

Итак, в конце мы получаем:

$$U_1^*,U_2^*,...,U_m^*;W_{\max}$$

Задача распределения ресурсов.

Это едва ли не самая распространённая операция. Под ресурсом в общем случае понимают физическую или абстрактную величину, которую система использует для производства полезного продукта. Например: горючее, деньги, время, объём склада. Как правило – ресурс ограничен, поэтому встаёт задача так распределить ресурс между отдельными элементами системы, чтобы суммарный эффект был максимальным. Рассмотрим классическую задачу распределения ресурсов.

Имеется начальное количество ресурсов k_0, которые необходимо распределить между двумя отраслями. Каждая отрасль работает в течении m лет. Если в первую отрасль в $i-$ый год вкладываются средства X_i, то доход $f(X_i)$, если же во вторую вкладываются Y_i, тогда доход $g(Y_i)$. Средства тратятся, принося доход, а новых средств не поступает и полученный доход не вкладывается.

Нас интересует суммарный доход: $W=\sum_{i=1}^{m} f(X_i)+g(Y_i)$. Суммарный выигрыш равен сумме выигрышей на каждом шаге. Состоянием системы является количество средств перед $i-$ым шагом. Так как новых средств не поступает, то ресурсы «тают».

Управление Y_i может быть записано как $Y_i=k-X_i$. После $i-$го шага в первой отрасли остаются средства $\varphi(X_i)$, а во второй $\psi(Y_i)=\psi(k-X_i)$. Эти функции называются функциями траты. Мы можем составить уравнение Беллмана. В этой задаче на *i*-ом шаге одно управление X_i и одно состояние k

$$W_i(k)=\max_{X_i}\{f(X_i)+g(k-X_i)+W_{i+1}[\varphi(X_i)+\psi(k-X_i)]\}$$

$$i=m:\ W_m(k)=\max_{X_m}\{f(X_m)+g(k-X_m)\}\ \text{и ттд.}$$

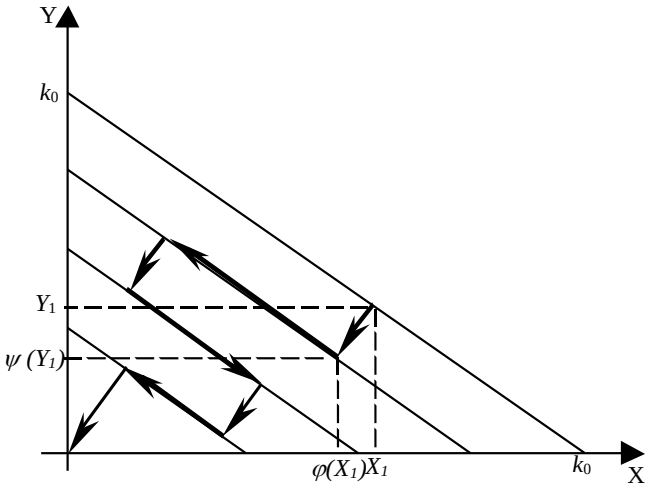
$$W_1(k), k=k_0, W_1(k_0)=W_{\max};\quad X_1(k_0)=X_1^*, Y_1^*=k_0-X_1^*$$

Исследуя функции траты, получим количество средств после $i-$го шага:

$\varphi(X_1^*)+\psi(Y_1^*)=k_1^*;\ X_2(k_1^*)=X_2^*$ и ттд.

Задача о распределении ресурсов допускает геометрическую интерпретацию.

$$X_1+Y_1=k_0$$



Распределение на первом шаге – указание точки на гипотенузе. После этого средства тратятся. Распределение средств – движение внутрь треугольника. Рассмотрим частные случаи задач о распределении ресурсов.

Распределение по неоднородным этапам.

Выше мы считали, что все функции одинаковы на всех этапах. Во многих задачах функции меняются от этапа к этапу: $f_i(X_i), g_i(Y_i); \varphi_i(X_i), \psi_i(Y_i)$. Покажем, что процедура динамического программирования принципиально не меняется. Запишем уравнение Беллмана:

$$W_i(k)=\max_{X_i}\{f_i(X_i)+g_i(k-X_i)+W_{i+1}[\varphi_i(X_i)+\psi_i(k-X_i)]\}$$

Распределение ресурсов между тремя и более отраслями.

В этом случае на каждом шаге будет уже n управлений, но одно из них может быть выражено как: $X_i^n=k-\sum_{j=1}^{n-1}X_i^j$. В этом случае, в правой части уравнения Беллмана будет две и более переменных, по которым ищется максимум, и задача усложняется.

Распределение ресурсов с резервированием.

В такой модели если средства распределяются между двумя отраслями, то какое-то количество средств можно оставить до последующего распределения. В этом случае задача имеет смысл даже для одной отрасли. Начальное количество средств разделяется на первом этапе на X_1 и на $k-X_1$ (резерв), на втором этапе подлежат разделению средства из резерва. Такую задачу можно представить как с одной отраслью реальной, а другой фиктивной (не приносящей доход и не расходующей средства). Решение такой задачи сводится к классической, задав функции дохода и трат.

$$f(X),g(Y)=0\text{ - функции дохода};\ \varphi(X),\psi(Y)=0\text{ - функции трат}$$

Подставив их в уравнение Беллмана, можно решить задачу как классическую. Задача может быть упрощена до следующей:

$$\left.\begin{array}{l} W=\sum_{i=1}^{m}f(X_i)\to\max \quad (1)\\ \sum_{i=1}^{m}X_i\le k_0 \quad (2)\end{array}\right\}$$

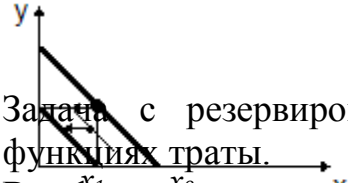
Задача линейного программирования с одной переменной.

Пусть вид функции $f(X_i)$ не убывающий в этом случае недоиспользовать средства не выгодно. В этом случае решение допускают теоремы, обосновывающие, если:

1) $f(X)$ неубывающая и выпуклая вверх, оптимальное распределение ресурсов равномерное.

2) $f(X)$ возрастающая и выпуклая вниз, оптимальное решение – вложить все средства в один этап, и ничего не зарезервировать.

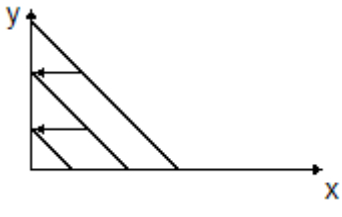
Таким образом приходим к классической задаче.



Трата || оси X.

Задача с резервированием в одной отрасли при параллельных функциях траты.

Все функции траты $\varphi(x_i)=0$.



В этом случае задача сводится к более простой.

$$\varphi(x_i) = 0$$
$$W = \sum_{i=1}^{m} f_i(x_i) \to \max$$
$$\sum x_i \le x_0$$

Рассмотрим более частный случай: все функции одинаковые на всех шагах .

$f_i(x) = f(x), \forall i$

Эти функции неубывающие.

$$W = \sum_{i=1}^{m} f(x_i) \to \max \qquad (1)$$

$$\sum x_i = x_0 \qquad (2)$$

(2) – равенство, т.к. функция неубывающая и недоиспользование средств невыгодно. Это имеет теоретическое обоснование:

-1- если функция неубывающая и выпуклая вверх, то оптимальным распределением является равномерное распределение.

-2- Если функция неубывающая и выпуклая вниз, то оптимальным распределением является такое: все распределение в один этап (элемент) и ничего в другие.

Распределение ресурсов «с вложением доходов в производство».

В классической задаче считается, что полученный доход на $i-$ом шаге в производство не вкладывается, т. е. он отчисляется и подсчитывается как эффект. Во многих задачах полученный эффект можно использовать как ресурс для следующего шага объединяя его с оставшимся ресурсом. Если ресурс не деньги, то средства можно привести к единому эквиваленту с оставшимися средствами. Такая модель является развитием классической модели. Так как оставшиеся средства и доход объединяются, то можно ввести единую интегральную функцию – функцию <u>изменения средств</u>. $F(X_i)-$количество оставшихся средств плюс доход после $i-$го шага, если вложили X_i.

I. $F(X_i)$

II. $G(Y_i) = G(k - X_i)$

$k-$количество средств перед $i-$м шагом.

Выигрыш на $i-$ом шаге зависит от того, как мы подсчитываем доход (эффект) от управления всеми ресурсами. Поставим задачу: максимальный доход в конце $m-$го шага. Тогда на всех шагах $i = \overline{1, m-1}$, доход = 0, $W_i = 0$. На

$m-$ом шаге выигрыш $W_m = F_m(X_m) + G_m(k - X_m)$. Подставив эти выражения в уравнение Беллмана, мы программируем задачу от начала к концу, если имеется начальное количество средств k_0. Здесь функция траты: $k' = F_i(X_i) + G_i(k - X_i)$.

Частный случай: когда F и G неубывающие. В этом случае чем больше значение **доход + средства** получается в конце $i-$го шага, тем лучшим условием это будет для проведения $(i+1)-$го шага. Поэтому можно не заботиться о следующих шагах, достаточно обеспечить максимум на каждом шаге.

Таким образом процедура оптимизации возможна в одном направлении от начала к концу, т. е. задача динамического программирования вырождается в задачу последовательной оптимизации.

$$\max_{X_1}\{F_1(X_i) + G_1(k_0 - X_1)\} = k_1^*$$
$$\max_{X_2}\{F_2(X_2) + G_2(k_1^* - X_2)\} = k_2^*$$
$$\dots \qquad W_{\max} = k_m^*$$
$$\max_{X_m}\{F_m(X_m) + G_m(k_{m-1}^* - X_m)\} = k_m^*$$

Рассмотрим задачу <u>распределения ресурсов с вложением доходов в производство и отчислением</u>. Это наиболее общий случай. Разделим функции дохода и функции траты: $\begin{matrix} f(X_i), g(Yi) \\ \varphi(X_i), \psi(Y_i) \end{matrix}$ и максимальный суммарный отчисленный **доход + оставшиеся средства** после $m-$го шага. Введём функцию отчисления $r_i(D_i)$; $D-$доход. Тогда выигрыш на каждом шаге:

$$W_i = r_i[f(X_i) + g(k - X_i)], \quad i = \overline{1, m-1}$$
$$W_m = r_m[f(X_m) + g(k - X_m)] + \varphi(X_m) + \psi(k - X_m) \quad (*)$$
$$W = \sum_{i=1}^{m} W_i \to \max$$
$$k' = \varphi(X_i) + \psi(k - X_i) + f(X_i) + g(k - Xi) - r_i[f(X_i) + g(k - X_i)]$$

Уравнение Беллмана для $i-$го шага будет выглядеть так:

$$W_i(k) = \max_{X_i}\{r_i[f(X_i) + g(k - X_i)] + W_{i+1}[\varphi(X_i) + \psi(k - X_i) + f(X_i) + g(k - X_i) - r_i[f(X_i) + g(k - X_i)]]\}$$
$$i = \overline{1, m-1}$$

для $i = m$ надо учесть $(*)$.

Если $r_i = 1$, то мы получаем классическую задачу.

Учёт предыстории процесса.

Мы считаем, что функции как выигрыша, так и траты зависят от состояния перед $i-$ым шагом, т. е. не зависят от более ранних состояний. Такие процессы называются <u>процессами без памяти</u>. Но иногда при рассмотрении процессов, связанных с «живыми» организациями требуется помнить всю историю происходящего. Такая задача более сложна. Введём расширенное состояние:

$$S=(S,S_{i-1},S_{i-2},\ldots,S_{i-L})$$

S_{i-L} – состояние за L шагов до i – го. Тогда $W_i(S,U_i),\varphi_i(S,U_i)$. Но задача сложна вычислительном аспекте. Пусть S имеет k координат и предыстория распространяется на L шагов, тогда результат $k\times L$. Вот почему подобные задачи можно решать если $k\times L\le 3$.

Задача с мультипликативным критерием.

До сих пор мы считали, что суммарный выигрыш равен сумме выигрыш на i – ом шаге. Но есть задачи, где общий критерий равен произведению критериальных величин на каждом шаге. В этом случае так же можно применить уравнение Беллмана. $W=\prod_{i=1}^{m}W_i$, но вместо этого можно взять функцию $W'=\ln W$. Оптимальные решения будут одинаковы ввиду многоэтапности функций. Но можно при вводе уравнения Беллмана учесть, что:

$$W_i(S)=\max_{U_i}\{W_i\times W_{i+1}(S')\}$$

$$W=F(W_1,W_2,\ldots,W_n)=F(W_1)\otimes F(W_2)\otimes\ldots\otimes F(W_n)$$

Пример: устройство состоит из n узлов. Имеется некоторое устройство k_0, которое может использоваться для повышения надёжности каждого узла. Необходимо так распределить ресурс, чтобы суммарная надёжность была максимальной.

$q(X_i)$ – надёжность каждого узла. $Q=\prod_{i=1}^{m}q_i(X_i)\to\max$. $\sum X_i\le k_0$.

Операции не связанные со временем.

Во многих задачах распределение ресурсов не связано с временными шагами. Ресурс обычно распределяется по объектам. Например, если расписать распределение ресурсов между n объектами и на каждый объект задана функция выигрыша, то такая задача эквивалентна рассмотренной нами задаче о распределении ресурсов с резервированием в одной отрасли по n шагам.