# ОПТИМИЗАЦИЯ РАДИОТЕХНИЧЕСКИХ СИСТЕМ

# 1. Постановка задачи оптимизации

$$f(x) \to \min_{Q}(\max_{Q}),$$

$$Q: \begin{cases} H_k(x) = 0,\ k = 1,\ldots,m, \\ g_j(x) \geq 0,\ j = m+1,\ldots,s. \end{cases}$$

$x \in Q$ - допустимые решения, $x^*$ - оптимальное решение.

$$f(\vec{x}) \to \min_{Q},$$

$$f(x) \to \inf, \quad x \in Q.$$

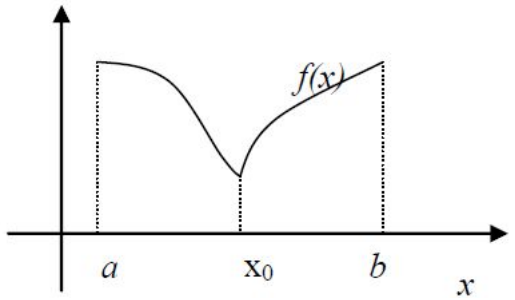
## 2. Методы одномерной оптимизации

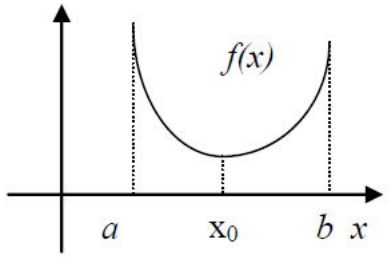
$$\min_{x \in [a,b]} \{z = f(x)\}.$$

*Монотонность функции*

При $x_1 < x_2$ выполняется $f(x_1) < f(x_2)$ или $f(x_1) > f(x_2)$.

# Унимодальность




*Определение глобального минимума.* Функция $f(x)$, определённая на множестве $D$ достигает глобального минимума в точке $x^* \in D$, если $f(x^*) < f(x)$ для всех $x \in D$.

*Определение локального минимума.* Функция $f(x)$, определённая на множестве $D$ имеет локальный минимум в точке $x^* \in D$, если существует такая ε-окрестность точки $x^*$, что для всех $x$ из этой ε-окрестности $f(x^*) < f(x)$.

# Методы прямого поиска состоят из двух групп:

1) все $N$ точек $x_к$., $к = 1,\ldots, N$, в которых будут вычислены значения функции, выбирают заранее (до вычисления функции в этих точках);

2) точки $x_k$ выбирают последовательно (для выбора последующей точки используют значения функции, вычисленные в предыдущих точках).

Будем считать, что стратегия поиска определена, если:

- определен алгоритм выбора точек $x_k$., к = 1,…, $N$;
- определено условие прекращения поиска, т.е. условие, при выполнении которого значение $f(x^*)$ считают найденным с заданной точностью.

**Оптимальный пассивный поиск** состоит в выборе точек, равномерно расположенных на отрезке $[a, b]$, координаты которых

$$x_k = \frac{(b-a)\,k}{N+1}, \quad k = 1, 2, \ldots, N.$$

$$f_k = f(x_k),$$
$$x^* \in (x_{j-1}, x_{j+1}), x^* \approx x_j \pm \varepsilon$$

В алгоритмах этих методов вычисляются значения функции в промежуточных точках $\lambda_k$ и $\mu_k$ $(\lambda_k < \mu_k)$ интервала неопределенности:

если $f(\lambda_k) > f(\mu_k)$, то в качестве границ нового интервала рассматривается интервал $[\lambda_k, b_k]$ (рис. 2)

если $f(\lambda_k) < f(\mu_k)$ это будет интервал $[a_k, \mu_k]$ (рис. 3) если $f(\lambda_k) = f(\mu_k)$, то оставим интервал $[\lambda_k, \mu_k]$.

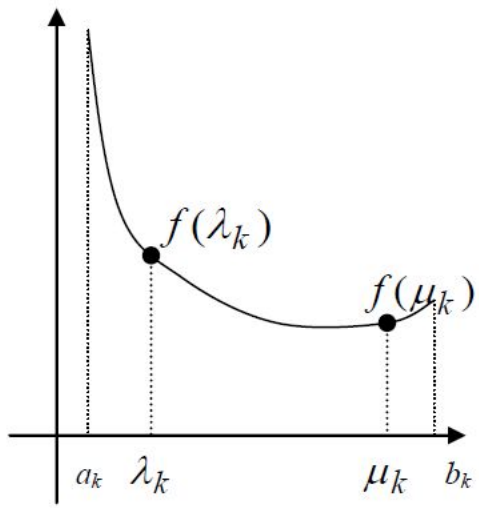



Рис. 2

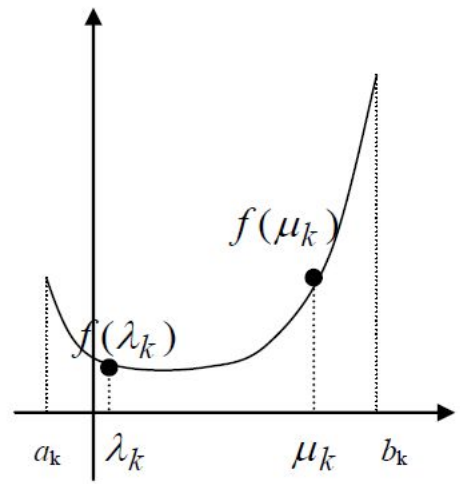



Рис. 3

**Теорема**[1]. Пусть даны монотонно возрастающая переменная $x_n$, и монотонно убывающая переменная $y_n$, причем всегда $x_n < y_n$. Если их разность $x_n - y_n$ стремится к нулю, то обе переменные имеют общий конечный предел: $\lim\limits_{n\to\infty} x_n = \lim\limits_{n\to\infty} y_n = c$.

**Метод дихотомии**

Идея метода состоит в вычислении на каждой очередной итерации двух значений целевой функции в точках, отстоящих на величину $\alpha$ в обе стороны от середины интервала неопределенности. Величина $\alpha$ в этом методе называется константой различимости, такова, что, с одной стороны, величина $2\alpha$ была близка к желаемому конечному значению интервала неопределенности, с другой, значения оптимизируемой функции на краях интервала $2\alpha$ были различимы.

Введем обозначения: ε – конечная длина интервала, определяющая точность, $[a_1, b_1]$ – начальный интервал неопределенности, $k$ – номер итерации. Пусть на $k$-м шаге $x^* \in [a_k, b_k]$. На этом отрезке $[a_k, b_k]$ выберем две точки $\lambda_k = \frac{a_k + b_k}{2} - \alpha$, $\mu_k = \frac{a_k + b_k}{2} + \alpha$ (рис. 4). Вычислим значения в этих точках. $f(\lambda_k), f(\mu_k)$ и выполним процедуру исключения отрезка, то есть

если $f(\lambda_k) > f(\mu_k)$, то $a_{k+1} = \lambda_k$, $b_{k+1} = b_k$, иначе $a_{k+1} = a_k$ и $b_{k+1} = \mu_k$.

Таким образом, построим новый отрезок $[a_{k+1}, b_{k+1}] \subset [a_k, b_k]$. Длина интервала неопределенности после $k$-ой итерации

$$|b_{k+1} - a_{k+1}| = \frac{1}{2^k}(b_1 - a_1) + 2\alpha\left(1 - \frac{1}{2^k}\right)$$

Отсюда можно вычислить число итераций для достижения необходимой точности ε, потребуется $n \geq \frac{\ln((b_1 - a_1)/\varepsilon)}{\ln 2}$ итераций. На каждой итерации минимизируемая функция вычисляется дважды.

**Метод деления пополам.**

На каждой итерации исключается половина интервала. Пусть на $k$-м шаге $x^* \in [a_k, b_k]$, длиной $l_k$ (рис. 5). Найдем середину интервала $\bar{x} = \dfrac{a_k + b_k}{2}$, вычислим $f(\bar{x})$. Найдем промежуточные точки $x_1 = a_k + l_k / 4$, $x_2 = b_k - l_k / 4$ и значения $f(x_1), f(x_2)$. Далее проведем процедуру исключения интервала: если $f(x_1) < f(\bar{x})$ то исключается интервал $(\bar{x}, b_k)$, при этом $b_{k+1} = \bar{x}$, $a_{k+1} = a_k$, $\bar{x} = x_1$; если $f(x_2) < f(\bar{x})$, то исключается интервал $(a, \bar{x})$, при этом $a_{k+1} = \bar{x}$, $b_{k+1} = b_k$, $\bar{x} = x_2$. Средняя точка последовательности получаемых интервалов всегда совпадает с одной из пробных точек $x_1, x_2$, или $\bar{x}$, найденных на предыдущих итерациях.

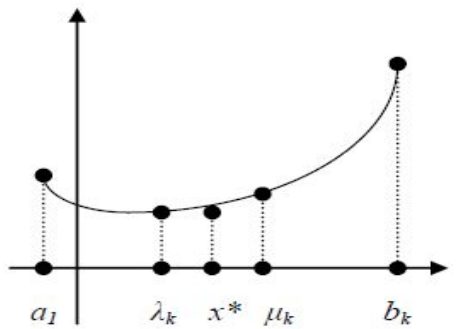



Рис. 4

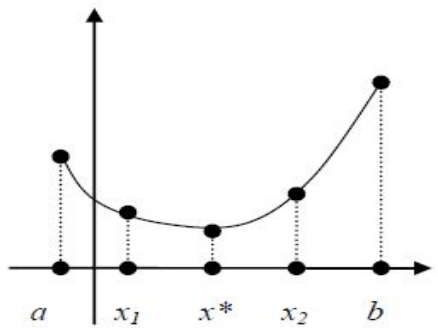



Рис. 5

На каждой итерации требуется не более 2-х вычислений значений функции. После $N$ вычислений длина интервала равна $l = (1/2)^{1/N}$ длинны исходного интервала. Таким образом, для достижения необходимой точности нужно провести $N \geq -\dfrac{\ln 2}{\ln \varepsilon}$ итераций.

# 3. Методы одномерной оптимизации, использующие информацию о производной

**Теорема Ферма.** Пусть $f(x)$ дифференцируема на интервале и достигает своего наименьшего (наибольшего) значения на этом интервале в некоторой внутренней точке $x^*$ этого интервала, тогда $f'(x^*) = 0$.

$$f'(x^*)=0$$

Таким образом, если унимодальная функция $f(x)$ непрерывно дифференцируема на отрезке минимизации, то точку $x^*$ наименьшего значения функции можно вычислять как корень уравнения $f'(x) = 0$ с помощью тех или иных методов численного решения нелинейных уравнений. В этом случае на точность решения задачи решающее влияние оказывает погрешность вычисления производной функции. Рассмотрим некоторые методы одномерной минимизации, основанные на использовании производной минимизируемой функции.

**Метод средней точки.**

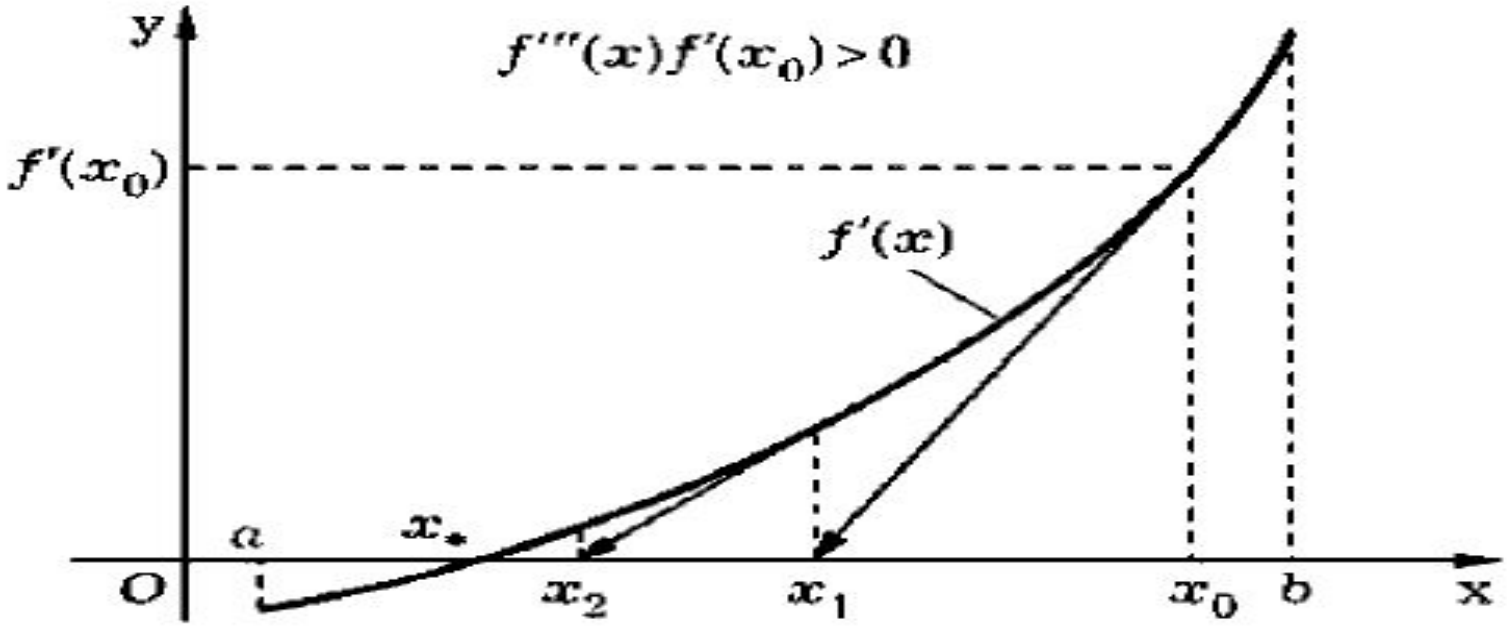
Будем искать минимум функции $f(x)$ непрерывно дифференцируемой и строго унимодальной на отрезке $[a,b]$. В этом случае единственной точкой $x^* \in [a,b]$ минимума будет стационарная точка, в которой $f'(x^*) = 0$. Отметим, что непрерывно дифференцируемая унимодальная на отрезке функция может иметь на нем более одной стационарной точки. На отрезке определяются две точки $a_k, b_k$, в которых производные имеют разные знаки, $f'(a_k) f'(b_k) < 0$. Искомый оптимум находится между ними. Делим интервал пополам: $x_k = \frac{a_k + b_k}{2}$, если $f'(x_k) > 0 \, (< 0)$, то из двух интервалов оставляем тот, на концах которого производная имеет разные знаки.

## Метод Ньютона (метод касательной)

Выбираем $x_0$ .

$$[x_0, f'(x_0)]$$

$$f'(x) \approx f'(x_0) + f''(x_0)(x - x_0)$$




$$x_1 = x_0 - \frac{f'(x_0)}{f''(x_0)}$$

$$x_{k+1} = x_k - \frac{f'(x_k)}{f''(x_k)}$$

Поскольку для дважды непрерывно дифференцируемой функции в точке минимума $f''(x^*) > 0$, то должно быть и $f''(x_0) > 0$. Поэтому говорят, что метод Ньютона обладает *локальной сходимостью* в том смысле, что надо выбрать хорошее начальное приближение, попадающее в такую окрестность точки $x^*$, где $f''(x) > 0$. Однако проверка выполнения этого условия не всегда возможна. Достаточным условием монотонной сходимости метода Ньютона будут постоянство в интервале между точками $x_0$ и $x^*$ знака производной $f'''(x)$ и совпадение его со знаком $f'(x)$.

**Теорема**. Если $f(x)$ трижды дифференцируема в окрестности точки $x^*$, тогда релаксационная последовательность (1) сходится к $x^*$ и имеет место оценка

$$|x_{k+1} - x^*| \le \frac{2 \max\limits_{U(x^*)} |f'''(x)|}{\min\limits_{U(x^*)} |f''(x)|} |x_k - x^*|^2 .$$

## Метод секущих

$$x_{k+1} = x_k - \frac{f'(x_k)(x_k - x_{k-1})}{(f'(x_k) - f'(x_{k-1}))}$$

# 4. Задачи многомерной оптимизации

$$f(\vec{x}) \to \min_{Q},$$

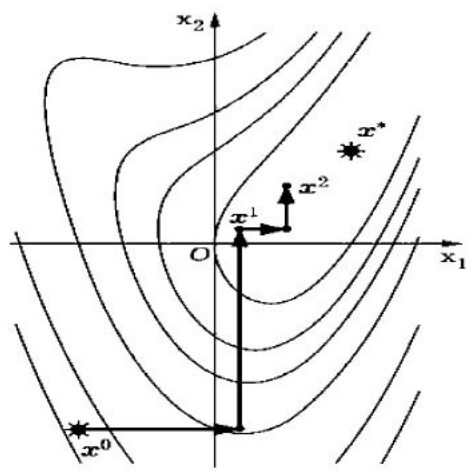
$$Q \subset R^N$$

Все методы решения задач безусловной оптимизации состоят в том, что мы строим последовательность точек $\{x^{(n)}\}$ таким образом, чтобы последовательность функций $f(x^{(n)})$ была убывающей (т. е. спускаемся вдоль функции). На $k$-м шаге $(k > 0)$ определяем вектор $\vec{S}_k$, в направлении которого функция $f(\bar{x})$ уменьшается. В этом направлении делаем шаг величиной $\lambda_k$ и получаем новую точку $x^{k+1} = x^k + \lambda_k \vec{S}_k$, в которой $f(x^{k+1}) < f(x^k)$.. Последовательность $\{x^{(n)}\}_{n \in N}$, удовлетворяющая этому условию, называется *релаксационной последовательностью*, а соответствующие методы – *методами спуска*. Методы решения делятся на методы с использованием информации о производных функции и без использования таковой. Различные методы спуска отличаются выбором направления и величины шага. Как правило, для нахождения $\lambda_k$ используется процедура одномерного поиска.

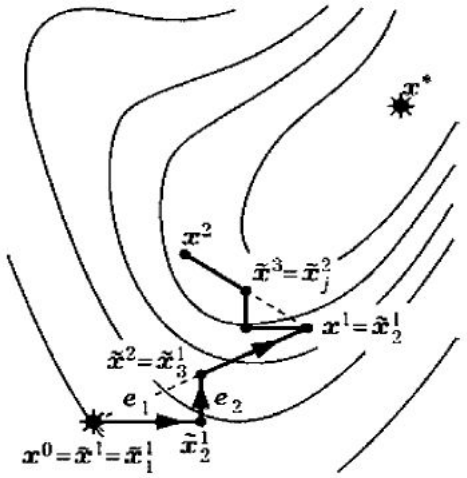
1. Методы нулевого порядка, или прямого поиска, стратегия которых основана на использовании информации только о свойствах целевой функции.
2. Методы первого порядка, в которых при построении итерационной процедуры наряду с информацией о целевой функции используется информация о значениях первых производных этой функции.
3. Методы второго порядка, в которых наряду с информацией о значениях целевой функции и ее производных первого порядка используется информация о вторых производных функции.

## 4.1. Методы нулевого порядка

### 1. Метод Гаусса – Зейделя



## 2. Метод Хука – Дживса




# 4.2. Методы первого порядка

## 1. Метод наискорейшего спуска

Пусть в точке $x$ требуется определить направление наискорейшего спуска (то есть направление наибольшего локального уменьшения $f(x)$. Разложим $f(x)$ в ряд Тейлора в окрестности точки $x$ и отбросим члены второго порядка по $\Delta x$ и выше $f(x) = f(x^{(k)}) + (\nabla f(x^{(k)}), \Delta x) + \ldots$

Локальное уменьшение $f(x)$ определяется вторым слагаемым, т. е. наибольшее уменьшение $f(x)$ будет тогда, когда $(\nabla f(x^{(k)}), \Delta x)$ будет иметь наибольшую отрицательную величину. Этого можно добиться выбором $S^{(k)} = -\nabla f(x^{(k)})$, вектор спуска коллинеарен антиградиенту. Этот случай соответствует наискорейшему локальному спуску $x^{(k+1)} = x^{(k)} - \alpha_k \nabla f(x^{(k)})$.

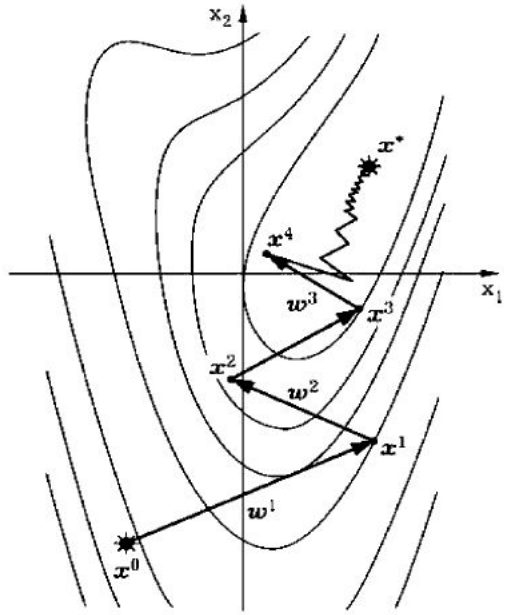



Рис.1

## 2.Метод сопряженных градиентов

$$x^{(k+1)} = x^{(k)} + \lambda_k S^{(k)}$$

$$S^{(k+1)} = -\nabla f(x^{(k)}) + \sum_{i=1}^{k} \alpha_i \, S^{(i)}.$$

# 4.3. Методы 2-го порядка

$$\varphi(x) = f(x^k) + \nabla f(x^k)(x - x^k) + \frac{1}{2}(x - x^k)^T H(x^k)(x - x^k),$$

$$x^{(k)} = x^{(k-1)} - H^{-1}(x^{(k-1)})\nabla f(x^{(k-1)}).$$

Алгоритм оптимизации, в котором направление поиска определяется из этого соотношения, называется *методом Ньютона*, а направление $p_k = -H^{-1}(x^{(k-1)})\nabla f(x^{(k-1)})$ – *ньютоновским направлением*, составляет с вектором градиента тупой угол (рис. 1).

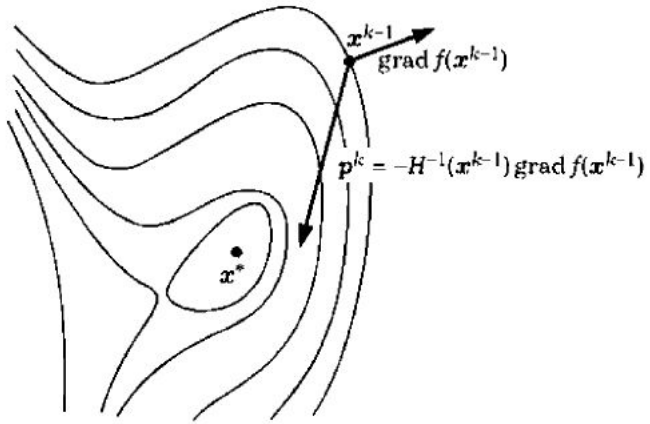



Рис. 1

# 5. АНАЛИТИЧЕСКИЕ МЕТОДЫ РЕШЕНИЯ ЗАДАЧ ОПТИМИЗАЦИИ

## ЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ

$$f(x) = c_1 x_1 + c_2 x_2 + \ldots + c_n x_n \to \min$$

при ограничениях

$$\begin{cases} a_{11} x_1 + a_{12} x_2 + \ldots + a_{1n} x_n = b_1, \\ a_{21} x_1 + a_{22} x_2 + \ldots + a_{2n} x_n = b_2, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{l1} x_1 + a_{l2} x_2 + \ldots + a_{ln} x_n = b_l, \end{cases} \quad (6.1)$$

$$\begin{cases} a_{(l+1)1} x_1 + \ldots + a_{(l+1)n} x_n \le b_{l+1}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m1} x_1 + a_{m2} x_2 + \ldots + a_{mn} x_n \le b_m, \end{cases} \quad (6.2)$$

$$x_1 \ge 0,\ x_2 \ge 0,\ \ldots,\ x_n \ge 0.$$

В условии задачи линейного программирования могут содержаться неравенства и противоположного, чем в (6.2), знака, однако такие неравенства легко сводятся к виду (6.2) умножением на $-1$.

**Определение 6.1.** Если в условии задачи линейного программирования не содержатся ограничения-неравенства (6.2), то есть в (6.1) $l = m$, то она называется *задачей линейного программирования в каноническом виде*.

Вводя дополнительные переменные $x_{n+i} \geq 0$, $i = l+1, \ldots, m$, ограничения-неравенства (6.2) можно записать в виде равенств

$$\begin{cases} a_{(l+1)1}x_1 + \ldots + a_{(l+1)n}x_n + x_{n+l+1} = b_{l+1}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n + x_{n+m} = b_m. \end{cases}$$

Таким образом, любая задача линейного программирования может быть записана в каноническом виде

$$f(x) = c_1x_1 + c_2x_2 + \ldots + c_nx_n \to \min \qquad (6.3)$$

при ограничениях

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n = b_1, \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n = b_2, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n = b_m, \end{cases} \qquad (6.4)$$

$$x_1 \geq 0,\ x_2 \geq 0,\ \ldots,\ x_n \geq 0. \qquad (6.5)$$

Задачу линейного программирования (6.3)—(6.5) можно записать в векторной форме

$$f(x) = (c,\ x) \to \min,$$

$$Ax = b, \qquad (6.6)$$

$$x \geq 0,$$

где $x = (x_1, x_2, \ldots, x_n)$ — вектор неизвестных, $c = (c_1, c_2, \ldots, c_n)$ — вектор коэффициентов целевой функции из (6.3),

$A = \begin{pmatrix} a_{11} & \ldots & a_{1n} \\ \ldots & \ldots & \ldots \\ a_{m1} & \ldots & a_{mn} \end{pmatrix}$ — матрица коэффициентов при неизвестных системы (6.4), $b = (b_1, b_2, \ldots, b_m)$ — вектор правых частей системы (6.4).

Задача $f(x) = (c, x) \to \max$ на множестве $X$ может быть сведена к минимизации функции $-f(x) = (-c, x)$ на том же множестве $X$.

Градиент $\nabla f(x) = c$ указывает направление возрастания целевой функции $f(x) = (c, x)$. Тогда вектор антиградиент $-\nabla f(x) = -c$ — направление убывания функции $f(x)$.

Если задача линейного программирования содержит только две переменные и в ее условии нет ограничений-равенств (6.1), то такую задачу можно исследовать и решить графически.

Рассматривается задача линейного программирования

$$f(x) = c_1 x_1 + c_2 x_2 \to \min \tag{6.8}$$

при ограничениях

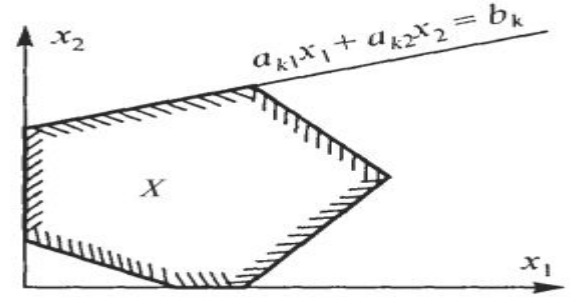
$$\begin{cases} a_{11} x_1 + a_{12} x_2 \le b_1, \\ a_{21} x_1 + a_{22} x_2 \le b_2, \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m1} x_1 + a_{m2} x_2 \le b_m, \end{cases} \tag{6.9}$$

$$x_1 \ge 0, x_2 \ge 0. \tag{6.10}$$

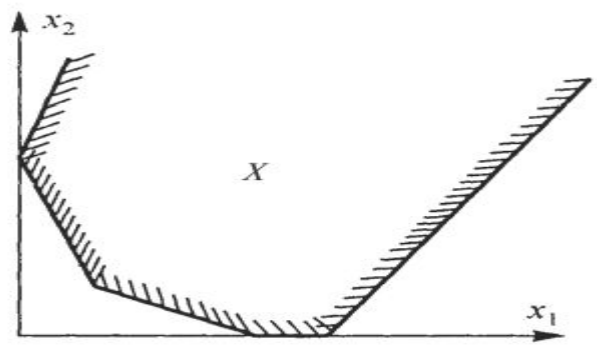
На плоскости $(x_1, x_2)$ любое из неравенств (6.9) определяет полуплоскость, лежащую по одну из сторон от прямой $a_{i1} x_1 + a_{i2} x_2 \le b_i$, $i = 1, 2, \ldots, m$. Для того чтобы определить расположение этой полуплоскости относительно граничной прямой, можно подставить координаты какой-либо точки (при $b_i \ne 0$ проще всего взято начало координат) в соответствующее неравенство (6.9) и проверить его выполнение.

Таким образом, допустимое множество $X$ задачи (6.8)–(6.10) является пересечением первого квадранта $x_1 \geq 0$, $x_2 \geq 0$ и полуплоскостей, соответствующих неравенствам (6.9). Поэтому множество $X$ представляет собой либо:

1) пустое множество, тогда задача (6.8)–(6.10) не имеет решений из-за несовместимости ограничений (6.9), (6.10);
2) многоугольник;




3) неограниченное многоугольное множество.



Для решения задачи (6.8)—(6.10) в случае $X \neq \varnothing$ рассмотрим семейство линий уровня функции $f(x)$ из (6.8):

$$c_1 x_1 + c_2 x_2 = C = \text{const}, \tag{6.11}$$

которые являются параллельными прямыми. Антиградиент $-\nabla f(x) = (-c_1, -c_2) = e$ перпендикулярен прямым (6.11) и указывает направление убывания функции $f(x)$. Если перемещать параллельно самой себе произвольную прямую (6.11), проходящую через допустимое множество $X$, в направлении $e$ убывания функции $f(x)$ до тех пор, пока эта прямая будет иметь

хотя бы одну общую точку с множеством $X$, то в своем крайнем положении указанная прямая пройдет через точку множества $X$, в которой целевая функция $f(x)$ принимает минимальное значение.

Используя графический метод, найти решение задачи линейного программирования

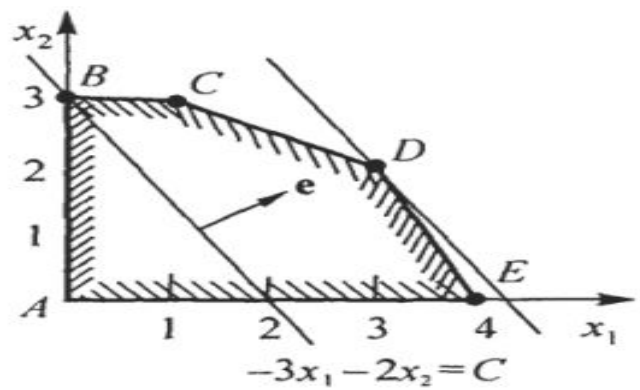
$$f(x) = -3x_1 - 2x_2 \to \min,$$

$$\begin{cases} x_1 + 2x_2 \le 7, \\ 2x_1 + x_2 \le 8, \\ x_2 \le 3, \end{cases}$$

$$x_1 \ge 0,\ x_2 \ge 0.$$

Р е ш е н и е

Изобразим на плоскости $(x_1, x_2)$ допустимое множество $X$ данной задачи (многоугольник $ABCDE$) и одну из линий уровня $-3x_1 - 2x_2 = C$ целевой функции.




Антиградиент $-\nabla f(x) = (3,\ 2) = e$ указывает направление убывания функции $f(x)$.

Совершая параллельный перенос линии уровня вдоль направления $e$, находим ее крайнее положение. В этом положении прямая $-3x_1 - 2x_2 = C$ проходит через вершину $D = (3,\ 2)$ многоугольника $ABCDE$. Поэтому целевая функция $f(x)$ принимает минимальное значение $f^*$ в точке $x^* = (3,\ 2)$, причем

$$f^* = f(x^*) = f(3,\ 2) = -13.$$

Пусть ранг $r$ матрицы системы ограничений (6.4) (то есть матрицы $A$ из (6.6)) равен рангу расширенной матрицы $(A|b)$ этой системы. Иначе система (6.4) несовместна и задача линейного программирования (6.3)–(6.5) не имеет решения, так как ее допустимое множество $X$ пусто.

Выберем произвольный базисный минор матрицы $A$. Без ограничения общности будем считать, что этот минор порядка $r$ соответствует первым $r$ столбцам и строкам матрицы $A$. Если $r < m$, то уравнения (6.4) с номерами $i = r+1,\ \ldots,\ m$ являются следствиями остальных уравнений и их следует опустить. Поэтому считаем, что $r = m$.

Допустим, что $1 \le n - m \le 2$. Тогда, считая переменные $x_j$, $j = 1,\ 2,\ \ldots,\ m$, базисными, а остальные — свободными, решим

систему (6.4), то есть выразим базисные переменные через свободные, после чего исключим базисные переменные из условия задачи (6.3)–(6.5). В результате получим задачу линейного программирования (6.8)–(6.10), эквивалентную исходной задаче и содержащую только свободные переменные исходной задачи, а их число не превосходит двух. Для решения полученной задачи можно использовать графический метод.

# Пример

Используя графический метод, найти решение задачи линейного программирования

$$f(x) = x_1 + 9x_2 + 5x_3 + 3x_4 + 4x_5 + 14x_6 \to \min,$$

$$\begin{cases} x_1 + x_4 = 20, \\ x_2 + x_5 = 50, \\ x_3 + x_6 = 30, \\ x_4 + x_5 + x_6 = 60, \end{cases}$$

$$x_j \geq 0,\ j = 1,\ 2,\ \ldots,\ 6.$$

**Решение**

Матрица $A$ ограничений-равенств имеет вид

$$A=\begin{pmatrix}1 & 0 & 0 & 1 & 0 & 0\\ 0 & 1 & 0 & 0 & 1 & 0\\ 0 & 0 & 1 & 0 & 0 & 1\\ 0 & 0 & 0 & 1 & 1 & 1\end{pmatrix}$$

Ее ранг $r=4=m$, причем за базисный минор примем минор, образованный первыми четырьмя столбцами. Число свободных переменных

$$n-m=6-4=2\le 2,$$

поэтому для решения задачи можно использовать графический метод.

Решая систему ограничений-равенств относительно базисных переменных $x_j$, $j=1,\ 2,\ 3,\ 4$, получим

$$\begin{cases}x_1=-40+x_5+x_6,\\ x_2=50-x_5,\\ x_3=30-x_6,\\ x_4=60-x_5-x_6.\end{cases} \qquad (6.12)$$

Исключая переменные $x_j$, $j=1,2,3,4$, из выражения для целевой функции, имеем

$$f(x)=740-7x_5+7x_6. \qquad (6.13)$$

Учитывая условия неотрицательности $x_j \geq 0$, $j = 1, 2, \ldots, 6$, и равенства (6.12) и (6.13), получим следующую задачу линейного программирования:

$$f(x) = 740 - 7x_5 + 7x_6 \to \min,$$

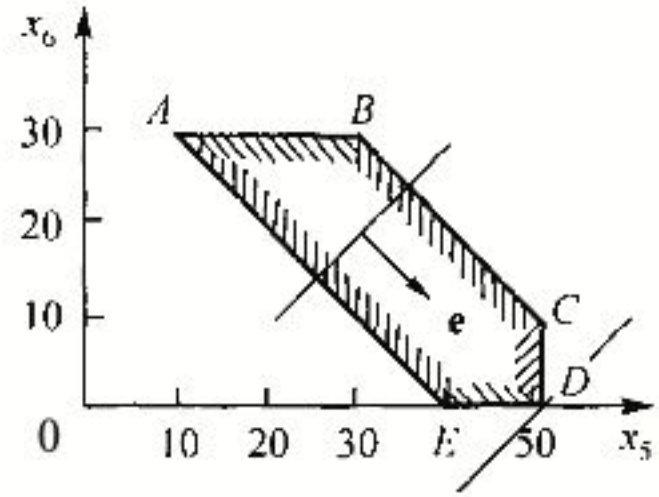
$$\begin{cases} x_5 + x_6 \geq 40, \\ x_5 \leq 50, \\ x_6 \leq 30, \\ x_5 + x_6 \leq 60, \end{cases}$$

$$x_5 \geq 0,\ x_6 \geq 0.$$

Изобразим на плоскости $(x_5, x_6)$ допустимое множество $X$ данной задачи (многоугольник $ABCDE$).

Антиградиент $-\nabla f(x) = (7, -7) = e$ указывает направление убывания функции $f(x)$.

Совершая параллельный перенос линии уровня $-7x_5 + 7x_6 = C$ функции $f(x)$ вдоль направления $e$, находим ее крайнее положение. В этом положении прямая $-7x_5 + 7x_6 = C$ проходит через вершину $D = (50, 0)$ многоугольника $ABCDE$.




Подставив значения $x_5 = 50$, $x_6 = 0$ в равенства (6.12), находим

$$x^* = (10,\ 0,\ 30,\ 10,\ 50,\ 0).$$

Следовательно, целевая функция $f(x)$ принимает минимальное значение $f^*$ в точке $x^* = (10,\ 0,\ 30,\ 10,\ 50,\ 0)$, причем

$$f^* = f(x^*) = f(10,\ 0,\ 30,\ 10,\ 50,\ 0) = -390.$$

Рассматривается задача линейного программирования в каноническом виде

$$f(x) = c_1 x_1 + c_2 x_2 + \ldots + c_n x_n \to \min \tag{6.14}$$

при ограничениях, определяющих допустимое множество $X$,

$$\begin{cases} a_{11} x_1 + a_{12} x_2 + \ldots + a_{1n} x_n = b_1, \\ a_{21} x_1 + a_{22} x_2 + \ldots + a_{2n} x_n = b_2, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a_{m1} x_1 + a_{m2} x_2 + \ldots + a_{mn} x_n = b_m, \end{cases} \tag{6.15}$$

$$x_1 \geq 0,\ x_2 \geq 0,\ \ldots,\ x_n \geq 0. \tag{6.16}$$

Пусть ранг $r$ матрицы $A = (a_{ij})$ системы ограничений равенств (6.15) задачи линейного программирования в каноническом виде равен рангу расширенной матрицы $(A|b)$ этой системы. Иначе система (6.15) несовместна и задача линейного программирования (6.14)—(6.16) не имеет решения, так как ее допустимое множество $X$ пусто.

Выберем произвольный базисный минор матрицы $A$. Без ограничения общности будем считать, что этот минор порядка $r$ соответствует первым $r$ столбцам и строкам матрицы $A$. Если $r < m$, то уравнения (6.15) с номерами $i = r+1,\ \ldots,\ m$, являющиеся следствиями остальных уравнений системы, отбросим. Поэтому считаем в дальнейшем, что $r = m$.

Разрешим систему (6.15) относительно базисных переменных $x_j$, $j = 1,\ \ldots,\ m$. Для этого с помощью эквивалентных преобразований приведем систему (6.15) к виду

$$\begin{cases} x_1 + \alpha_{1m+1}^{(0)} x_{m+1} + \ldots + \alpha_{1n}^{(0)} x_n = \beta_1^{(0)}, \\ x_2 + \alpha_{2m+1}^{(0)} x_{m+1} + \ldots + \alpha_{2n}^{(0)} x_n = \beta_2^{(0)}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_m + \alpha_{mm+1}^{(0)} x_{m+1} + \ldots + \alpha_{mn}^{(0)} x_n = \beta_m^{(0)}. \end{cases} \quad (6.17)$$

Отсюда общее решение системы (6.15) запишется следующим образом:

$$\begin{cases} x_1 = \beta_1^{(0)} - \alpha_{1m+1}^{(0)} x_{m+1} - \ldots - \alpha_{1n}^{(0)} x_n, \\ x_2 = \beta_2^{(0)} - \alpha_{2m+1}^{(0)} x_{m+1} - \ldots - \alpha_{2n}^{(0)} x_n, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ x_m = \beta_m^{(0)} - \alpha_{mm+1}^{(0)} x_{m+1} - \ldots - \alpha_{mn}^{(0)} x_n, \end{cases} \quad (6.18)$$

где свободные переменные $x_{m+1}$, ..., $x_n$ могут принимать произвольные значения.

Положим значения свободных переменных $x_{m+1}$, ..., $x_n$ равными нулю, тогда получим частное решение системы (6.15):

$$x_1 = \beta_1^{(0)},\ x_2 = \beta_2^{(0)},\ \ldots,\ x_m = \beta_m^{(0)},\ x_{m+1} = x_{m+2} = \ldots = x_n = 0$$

или

$$x^{(0)} = (\beta_1^{(0)},\ \beta_2^{(0)},\ \ldots,\ \beta_m^{(0)},\ 0,\ 0,\ \ldots,\ 0), \quad (6.19)$$

которое будем называть *базисным решением* системы (6.15). Каждому выбору базисных переменных соответствует свое базисное решение системы (6.15).

**Определение 6.2.** Базисное решение (6.19) называется *допустимым базисным решением* системы (6.15) или *угловой точкой* допустимого множества $X$ задачи линейного программирования (6.14)— (6.16), если все компоненты базисного решения (6.19) неотрицательны, то есть $\beta_i^{(0)} \geq 0$, $i = 1,\ 2,\ \ldots,\ m$.

**Определение 6.3.** Допустимое базисное решение (6.19) называется *вырожденным* (*вырожденной угловой точкой*), а соответствующая задача линейного программирования — *вырожден-*

*ной*, если среди неотрицательных чисел $\beta_i^{(0)}$ в допустимом базисном решении (6.19) есть равные нулю.

В основе симплекс-метода лежит факт.

Если задача линейного программирования (6.14)—(6.16) разрешима, то минимум целевой функции $f(x)$ из уравнения (6.14) достигается хотя бы в одной из угловых точек допустимого множества $X$ этой задачи.

Поскольку различные базисные решения системы (6.15) соответствуют различным способам выбора $m$ базисных из общего числа $n$ переменных $x_j$, то число допустимых базисных решений (угловых точек) не превышает $C_n^m = \frac{n!}{m!(n-m)!}$. Поэтому задачу линейного программирования можно решать методом перебора конечного числа угловых точек допустимого множества $X$, сравнивая значения целевой функции в этих точках. Однако при большой размерности $n$ задачи линейного программирования это будет затруднительно. Идея симплекс-метода состоит в направленном переборе угловых точек допустимого множества $X$ с последовательным уменьшением целевой функции $f(x)$.

### Схема симплекс-метода

Предположим, что задача линейного программирования (6.14)—(6.16) является невырожденной, а базисное решение (6.19) — допустимым. Выразим целевую функцию $f(x)$ из выражения (6.14) через свободные переменные $x_j$, $j = m+1, \ldots, n$, используя соотношение (6.18):

$$f(x) = p_0^{(0)} + \sum_{j=m+1}^{n} p_j^{(0)} x_j, \qquad (6.20)$$

$$p_0^{(0)} = \sum_{i=1}^{m} c_i \beta_i^{(0)}, \; p_j^{(0)} = c_j - \sum_{i=1}^{m} c_i \alpha_{ij}^{(0)}, \; j = m+1, \ldots, n.$$

**Случай 6.1.** Если в выражении (6.20) все коэффициенты $p_j^{(0)}$, $j = m+1, \ldots, n$, неотрицательны, то в угловой точке допустимого базисного решения (6.19) достигается минимум функции $f(x)$ из выражения (6.14) на допустимом множестве $X$ задачи (6.14)—(6.16) и этот минимум равен $p_0^{(0)}$.

**Случай 6.2.** Если в выражении (6.20) среди отрицательных коэффициентов $p_j^{(0)}$, $j \neq 0$, есть такой (например, $p_l^{(0)}$), что в выражении (6.17) все коэффициенты $\alpha_{il}^{(0)} \leq 0$, $i = 1, \ldots, m$, то целевая функция $f(x)$ не ограничена снизу на допустимом множестве $X$ и задача (6.14)—(6.16) не имеет решений.

**Случай 6.3.** Если в выражении (6.20) хотя бы один из коэффициентов $p_j^{(0)}$, $j \neq 0$, отрицателен (например, $p_l^{(0)} < 0$) и при этом среди коэффициентов $\alpha_{il}^{(0)}$ в выражении (6.17) есть хотя бы один положительный, то существует угловая точка $x^{(1)}$ множества $X$ такая, что $f(x^{(1)}) < f(x^{(0)})$.

В случаях 6.1 и 6.2 процесс решения задачи линейного программирования на этом заканчивается.

Рассмотрим случай 6.3. Пусть в выражении (6.20) коэффициент $p_l^{(0)} < 0$ и в выражении (6.17) имеются положительные коэффициенты $\alpha_{il}^{(0)}$, $i = 1, \ldots, m$. Найдем номер $k$ базисной переменной из условия

$$\frac{\beta_k^{(0)}}{\alpha_{kl}^{(0)}} = \min_{i:\alpha_{il}^{(0)}>0} \left\{ \frac{\beta_i^{(0)}}{\alpha_{il}^{(0)}} \right\}, \qquad (6.21)$$

где минимум берется по всем номерам $i = 1, \ldots, m$, для которых $\alpha_{il}^{(0)} > 0$.

Найдем решение системы (6.15), полагая свободными переменные $x_{m+1}, \ldots, x_{l-1}, x_k, x_{l+1}, \ldots, x_n$, то есть поменяв местами свободную переменную $x_l$ с базисной $x_k$. Система уравнений вида (6.17) в этом случае запишется следующим образом:

$$x_i + \sum_{\substack{j=m+1 \\ j \neq l}}^{n} \left( \alpha_{ij}^{(0)} - \alpha_{il}^{(0)} \frac{\alpha_{kj}^{(0)}}{\alpha_{kl}^{(0)}} \right) x_j - \frac{\alpha_{il}^{(0)}}{\alpha_{kl}^{(0)}} x_k = \beta_i^{(0)} - \alpha_{il}^{(0)} \frac{\beta_k^{(0)}}{\alpha_{kl}^{(0)}},$$

$$i = 1, \ \ldots, \ m, \ i \neq k, \qquad (6.22)$$

$$x_l + \sum_{\substack{j=m+1 \\ j \neq l}}^{n} \frac{\alpha_{kj}^{(0)}}{\alpha_{kl}^{(0)}} x_j + \frac{1}{\alpha_{kl}^{(0)}} x_k = \frac{\beta_k^{(0)}}{\alpha_{kl}^{(0)}},$$

а зависимость целевой функции от новых свободных переменных имеет вид

$$f(x) = \sum_{\substack{j=m+1 \\ j \neq l}}^{n} \left( p_j^{(0)} - p_l^{(0)} \frac{\alpha_{kj}^{(0)}}{\alpha_{kl}^{(0)}} \right) x_j - \frac{p_l^{(0)}}{\alpha_{kl}^{(0)}} x_k + p_0^{(0)} + p_l^{(0)} \frac{\beta_k^{(0)}}{\alpha_{kl}^{(0)}}. \quad (6.23)$$

Компоненты нового базисного решения $x^{(1)}$ можно определить, приравняв нулю свободные переменные $x_j$, $j = m+1, \ \ldots, \ n, \ j \neq l, \ x_k$ и найдя при этом условии значения базисных переменных из выражения (6.22). Базисное решение $x^{(1)}$ является допустимым, то есть угловой точкой множества $X$, причем $f(x^{(1)}) < f(x^{(0)})$.

По знакам коэффициентов в системе (6.22) и в выражении для целевой функции (6.23) можно получить один из трех случаев, приведенных выше, как это было сделано для угловой точки $x^{(0)}$. В случае 3 следует совершить переход к очередной угловой точке $x^{(2)}$, аналогичный переходу от $x^{(0)}$ к $x^{(1)}$ и т. д.

Поскольку число угловых точек допустимого множества $X$ не превышает $C_n^m$, то случай 3 может повторяться конечное число раз, то есть в результате конечного числа шагов перехода к новой угловой точке будет либо найдено решение задачи, либо сделано заключение о том, что она не имеет решений.

### Симплекс-таблицы

Реализация описанного выше симплекс-метода значительно упрощается при использовании симплекс-таблиц.

Записав коэффициенты уравнений (6.17) и целевой функции (6.20) соответствующим образом

| | $x_{m+1}$ | $\cdots$ | $x_j$ | $\cdots$ | $x_n$ | |
|---|---|---|---|---|---|---|
| $x_1$ | $\alpha_{1m+1}^{(0)}$ | $\cdots$ | $\alpha_{1j}^{(0)}$ | $\cdots$ | $\alpha_{1n}^{(0)}$ | $\beta_1^{(0)}$ |
| $\vdots$ | $\vdots$ | $\cdots$ | $\vdots$ | $\cdots$ | $\vdots$ | $\vdots$ |
| $x_k$ | $\alpha_{km+1}^{(0)}$ | $\cdots$ | $\alpha_{kj}^{(0)}$ | $\cdots$ | $\alpha_{kn}^{(0)}$ | $\beta_k^{(0)}$ |
| $\vdots$ | $\vdots$ | $\cdots$ | $\vdots$ | $\cdots$ | $\vdots$ | $\vdots$ |
| $x_m$ | $\alpha_{mm+1}^{(0)}$ | $\cdots$ | $\alpha_{mj}^{(0)}$ | $\cdots$ | $\alpha_{mn}^{(0)}$ | $\beta_m^{(0)}$ |
| | $p_{m+1}^{(0)}$ | $\cdots$ | $p_j^{(0)}$ | $\cdots$ | $p_n^{(0)}$ | $-p_0^{(0)}$ |

получим симплекс-таблицу задачи (6.14)—(6.16) для угловой точки $x^{(0)}$ из допустимого базисного решения (6.19).

Совершим переход от симплекс-таблицы, соответствующей угловой точке $x^{(0)}$, к симплекс-таблице для угловой точки $x^{(1)}$.

Пусть номера $k$ и $l$ найдены так, как это было сделано выше. Элемент $\alpha_{kl}^{(0)}$, строка и столбец симплекс-таблицы, на пересечении которых он находится, называются *разрешающими* или *опорными*, а сам элемент $\alpha_{kl}^{(0)}$ — *опорным* элементом симплекс-таблицы. Из формул (6.22) и (6.23) следует, что преобразование исходной симплекс-таблицы с опорным элементом $\alpha_{kl}^{(0)}$ приводит к новой симплекс-таблице

| | $x_{m+1}$ | $\cdots$ | $x_k$ | $\cdots$ | $x_n$ | |
|---|---|---|---|---|---|---|
| $x_1$ | $\alpha_{1m+1}^{(1)}$ | $\cdots$ | $\alpha_{1k}^{(1)}$ | $\cdots$ | $\alpha_{1n}^{(1)}$ | $\beta_1^{(1)}$ |
| $\vdots$ | $\vdots$ | $\cdots$ | $\vdots$ | $\cdots$ | $\vdots$ | $\vdots$ |
| $x_l$ | $\alpha_{lm+1}^{(1)}$ | $\cdots$ | $\alpha_{lk}^{(1)}$ | $\cdots$ | $\alpha_{ln}^{(1)}$ | $\beta_l^{(1)}$ |
| $\vdots$ | $\vdots$ | $\cdots$ | $\vdots$ | $\cdots$ | $\vdots$ | $\vdots$ |
| $x_m$ | $\alpha_{mm+1}^{(1)}$ | $\cdots$ | $\alpha_{mk}^{(1)}$ | $\cdots$ | $\alpha_{mn}^{(1)}$ | $\beta_m^{(1)}$ |
| | $p_{m+1}^{(1)}$ | $\cdots$ | $p_k^{(1)}$ | $\cdots$ | $p_n^{(1)}$ | $-p_0^{(1)}$ |

для определения элементов которой необходимо выполнить следующие действия:

1. Поменять местами переменные $x_k$ и $x_l$, остальные переменные таблицы оставить на прежних местах.
2. На место опорного элемента записать 1.
3. На остальных местах разрешающей строки поставить соответствующие элементы исходной таблицы.
4. На свободные места разрешающего столбца записать соответствующие элементы исходной таблицы, взятые со знаком минус.
5. На оставшихся свободных местах элементов $\alpha_{ij}^{(1)}$, $\beta_i^{(1)}$, $p_j^{(1)}$ в новой симплекс-таблице написать числа $\overline{\alpha}_{ij}$, $\overline{\beta}_i$, $\overline{p}_j$, найденные следующим образом:

$$\overline{\alpha}_{ij} = \alpha_{kl}^{(0)}\alpha_{ij}^{(0)} - \alpha_{kj}^{(0)}\alpha_{il}^{(0)}, \ \overline{\beta}_i = \alpha_{kl}^{(0)}\beta_i^{(0)} - \alpha_{il}^{(0)}\beta_k^{(0)},$$

$$\overline{p}_j = \alpha_{kl}^{(0)} p_j^{(0)} - \alpha_{kj}^{(0)} p_l^{(0)}.$$

6. Все полученные элементы новой таблицы разделить на опорный элемент $\alpha_{kl}^{(0)}$.

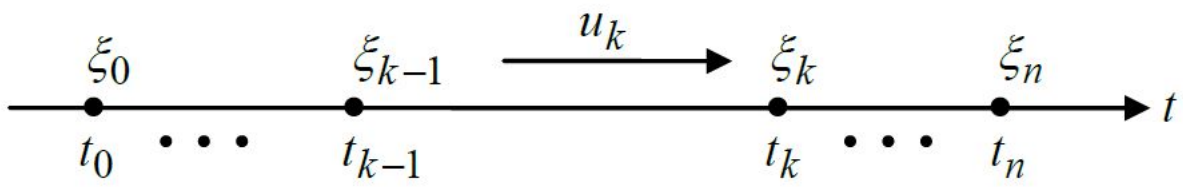
# 6. ДИНАМИЧЕСКОЕ ПРОГРАММИРОВАНИЕ

$$t_0, t_1, \ldots, t_n, \ldots$$

$$t_0 \to \xi_0$$

$$t_k \to \xi_k$$

$$\xi_k = \varphi(\xi_{k-1}, u_k) \qquad (1)$$




$$\xi_0, \xi_1, \ldots, \xi_n \qquad (2)$$

$$S = S(u_1, u_2, \ldots, u_n).$$

$$S = f_1 + f_2 + \ldots + f_n \qquad (3)$$

$$f_k = f_k(\xi_{k-1}, u_k). \qquad (4)$$

$$S = \sum_{k=1}^{n} f_k(\xi_{k-1}, u_k). \qquad (5)$$

Задача динамического программирования ставится следующим образом.

I. Имеется управляемая динамическая система, что означает:

1) выделено конечное число $n$ шагов;

2) на каждом шаге указаны все возможные состояния $(\xi_k)$, через которые проходит динамическая система, причем начальное состояние $\xi_0$ фиксировано;

3) на каждом шаге заданы управления $(u_k)$, причем указана связь (1), по которой состояние к концу шага однозначно определяется состоянием в начале шага и выбранном на этом шаге управлением.

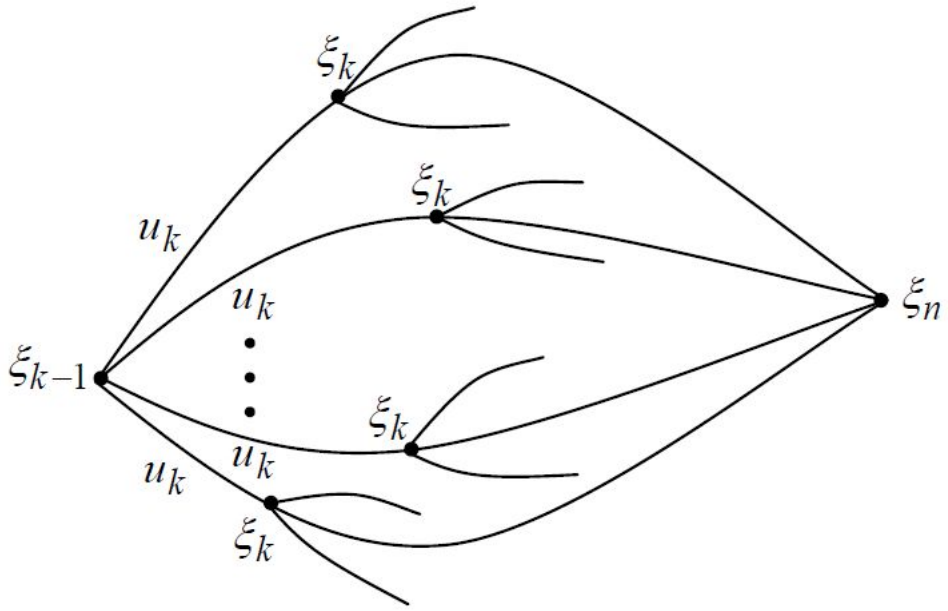
II. Задана аддитивная целевая функция, то есть на каждом шаге заданы доходы (затраты) для всех возможных состояний и для всех возможных управлений (функции 4) и функция (5).

$$S = \max; \qquad (u_1, u_2, \ldots, u_n) = ?$$

В случае, если показатель качества в (5) есть не доход, а затраты, целевая функция (5) минимизируется:

$$S = \min; \qquad (u_1, u_2, \ldots, u_n) = ?$$

Задача динамического программирования состоит в поиске оптимальной (близкой к оптимальной) траектории



$$S_k = f_k + f_{k+1} + \cdots + f_n.$$

Обозначим через $S_k^*(\xi_{k-1})$ - максимальный суммарный доход с k-го до n-го (последнего) шага.

$$S_k^*(\xi_{k-1}) = \max_{u_k} \{ f_k(\xi_{k-1}, u_k) + S_{k+1}^*(\xi_k) \}, \qquad (6)$$

где максимум берется по всем возможным управлениям на $k$-м шаге, $\xi_k$ – состояние, в которое переходит система из состояния $\xi_{k-1}$ под действием управления $u_k$ :

$$\xi_k = \varphi(\xi_{k-1}, u_k),$$

где $\varphi(\xi_{k-1}, u_k)$ определяется (1).

$$S_k^*(\xi_{k-1}) = \max_{u_k, u_{k+1}, \ldots, u_n} [f_k + f_{k+1} + \ldots + f_n] =$$

$$= \max_{u_k} \left[ \max_{u_{k+1}, \ldots, u_n} (f_k + f_{k+1} + \ldots + f_n) \right] =$$

$$= \max_{u_k} \left\{ f_k(\xi_{k-1}, u_k) + \max_{u_{k+1}, \ldots, u_n} [f_{k+1}(\xi_k, u_{k+1}) + \ldots + f_n] \right\} =$$

$$= \max_{u_k} \left[ f_k(\xi_{k-1}, u_k) + S_{k+1}^*(\xi_k) \right],$$

что и требовалось.

**Теорема** (принцип оптимальности). При построении оптимальной траектории нужно выбирать управление на каждом шаге так, чтобы доход на этом шаге плюс максимальный доход на последующих шагах был наибольшим.

Управление, на котором реализуется максимум в (6), будем обозначать $u_k^*(\xi_{k-1})$ и называть ***условным оптимальным управлением*** на $k$-м шаге.

# АЛГОРИТМ ДИНАМИЧЕСКОГО ПРОГРАММИРОВАНИЯ

**I этап** (движение от конца к началу). Начиная с конца последовательно находим

$$S_n^*(\xi_{n-1}), u_n^*(\xi_{n-1}); S_{n-1}^*(\xi_{n-2}), u_{n-1}^*(\xi_{n-2}); \ldots; S_1^*(\xi_0), u_1^*(\xi_0);$$

для всех возможных на соответствующих шагах состояний с использованием на каждом шаге, начиная с *n-1*-го, формулы (6). $S_n^*$ вычисляется непосредственно по формуле

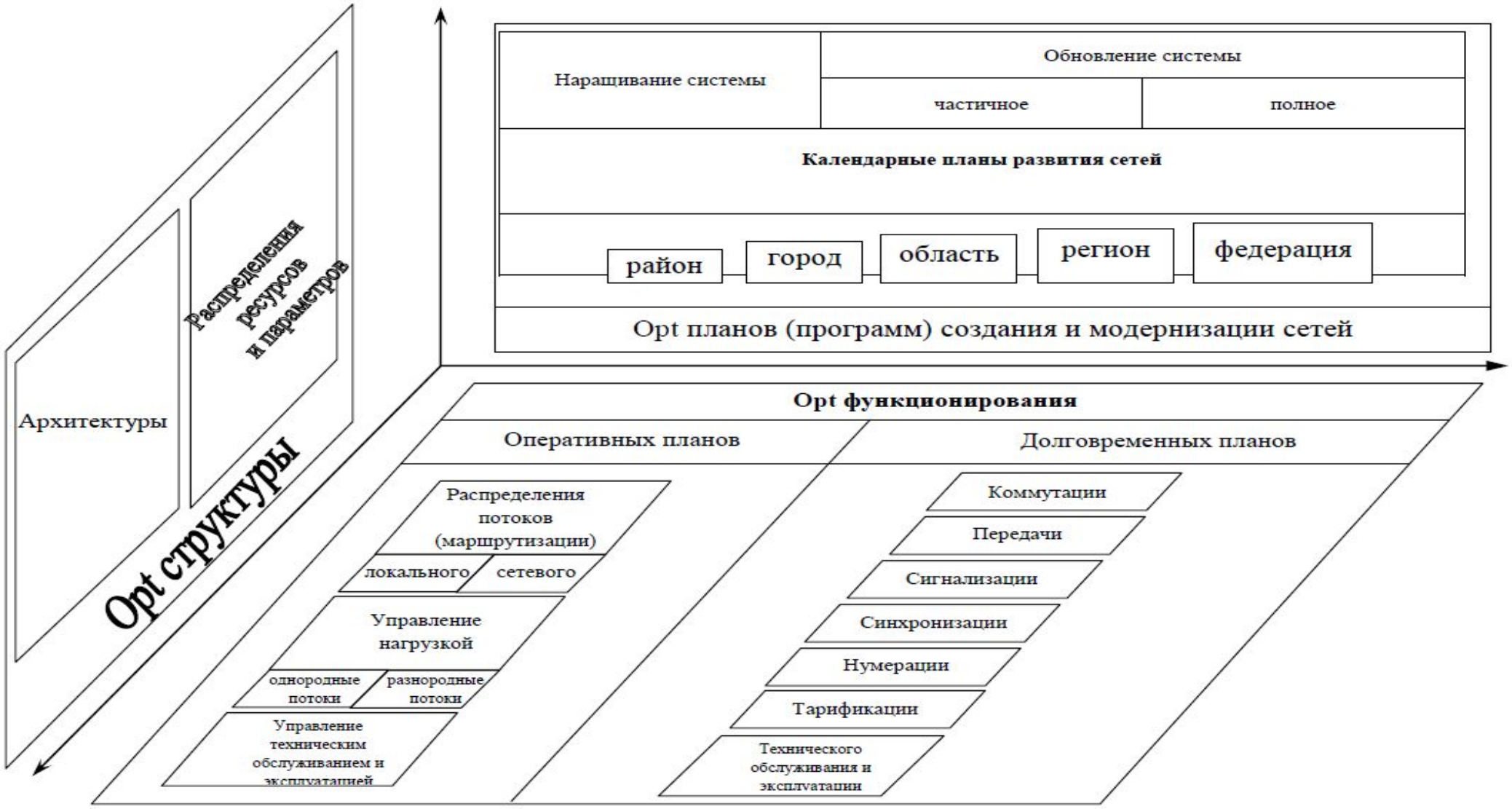
$$S_n^*(\xi_{n-1}) = \max_{u_n} f_n(\xi_{n-1}, u_n). \qquad (7)$$

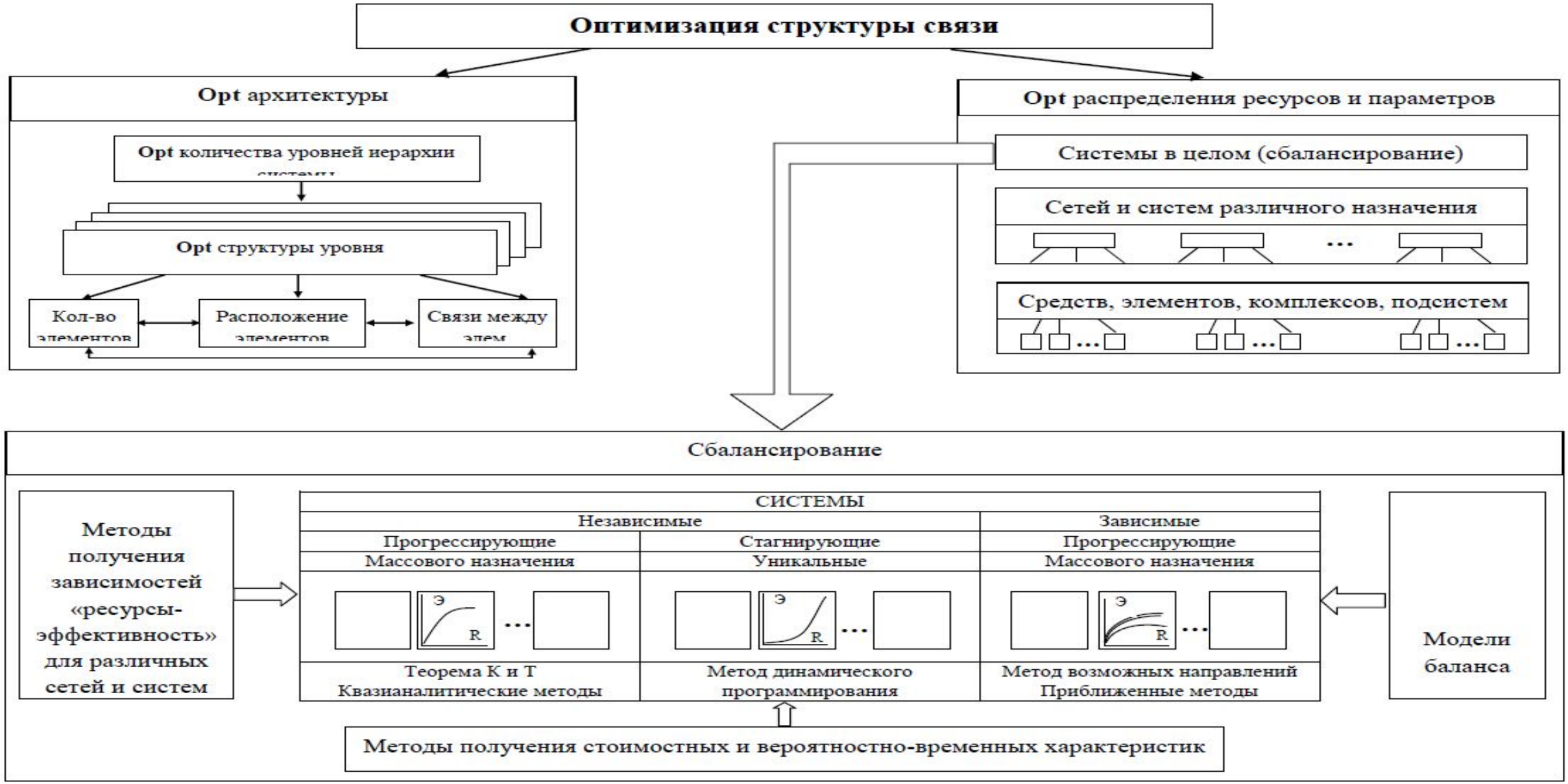
**II этап** (движение от начала к концу). Двигаясь от начала, существенно используя закрепленность начального состояния, строим безусловную оптимальную траекторию

$$\xi_0 \xrightarrow{u_1^*(\xi_0)} \xi_1^* \xrightarrow{u_2^*(\xi_1^*)} \xi_2^* \longrightarrow \ldots \longrightarrow \xi_n^*.$$

Здесь $\xi_k^* = \varphi[\xi_{k-1}^*, u_k^*(\xi_{k-1}^*)]$.

# 7. Оптимизация сетей связи




Оптимизация структуры связи
Opt архитектуры
Opt количества уровней иерархии
Opt структуры уровня
Кол-во элементов
Расположение элементов
Связи между элем
Opt распределения ресурсов и параметров
Системы в целом (сбалансирование)
Сетей и систем различного назначения
Средств, элементов, комплексов, подсистем
Сбалансирование
Методы получения зависимостей «ресурсы-эффективность» для различных сетей и систем
СИСТЕМЫ
Независимые
Зависимые
Прогрессирующие
Стагнирующие
Прогрессирующие
Массового назначения
Уникальные
Массового назначения
Э
R
Теорема К и Т
Квазианалитические методы
Метод динамического программирования
Метод возможных направлений
Приближенные методы
Модели баланса
Методы получения стоимостных и вероятностно-временных характеристик