Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Южно-Уральский государственный университет»
(Национальный исследовательский университет)
Институт естественных и точных наук
Факультет математики, механики и компьютерных технологий
Кафедра математического и компьютерного моделирования

Решение практических задач

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА К КУРСОВОЙ РАБОТЕ
по дисциплине «Теория принятия решений»
ЮУрГУ-010400.62.2015.11-001-1909 КР

Преподаватель,
____________ А.В. Панюков
« » ________ 2016 г.

Автор работы
студент группы ЕТ-224
____________ В.А. Безбородов
« » ________ 2016 г.

Челябинск 2016

ОГЛАВЛЕНИЕ

1 Задания для самостоятельной работы

1.1 Задание 1

1.1.1 Постановка задачи

Является ли заданная матрица A матрицей парных сравнений? Для матрицы A найти приближенное $\overline{W}$ и точное W значение главного собственного вектора. Оценить погрешность $\Delta\overline{W} = |\overline{W} - W|$. Определить, является ли матрица парных сравнений согласованной.

$$A = \begin{pmatrix} 1 & 4 & 6 & 8 \\ 1/4 & 1 & 3 & 2 \\ 1/6 & 1/3 & 1 & 3 \\ 1/8 & 1/2 & 1/3 & 1 \end{pmatrix}$$

1.1.2 Ход решения

Метод парных сравнений заключается в сравнении изучаемых объектов между собой. Объекты сравниваются попарно по отношению к их воздействию (“весу”, или “интенсивности”) на общую для них (вышестоящую в иерархии) характеристику.

Для матрицы парных сравнений всегда должно выдерживаться соотношение, отвечающее условию: если при сравнении i-го объекта с j-м объектом ставится оценка a_{ij}, то при сравнении j-го объекта с i-м, оценка a_{ji} должна быть обратной.

Видно, что для матрицы A такое соотношение строго выдерживается для каждой оценки. Поэтому матрица **A** является матрицей парных сравнений.

Определим приближенное $\overline{W}$ значение главного собственного вектора. Введем матрицу A (здесь и далее все вычисления выполняются на языке Julia; секция In соответствует введенному коду, секция Out - результатам вычислений; секции для удобства нумеруются).

```
In [1]: A = [1 4 6 8; 1/4 1 3 2; 1/6 1/3 1 3; 1/8 1/2 1/3 1]
```

```
Out[1]: 4x4 Array{Float64,2}:
 1.0       4.0       6.0       8.0
 0.25      1.0       3.0       2.0
 0.166667  0.333333  1.0       3.0
 0.125     0.5       0.333333  1.0
```

Приближенное значение $\overline{W}$ главного собственного вектора с некоторой точностью ε можно получить по формуле

$$\overline{W} = \lim_{k \to N} \frac{A^k e}{e^T A^k e}, e = (1, 1, K, 1)^T,$$

при таком N, что $|\overline{W}_i - \overline{W}_{i-1}| < \varepsilon$. Выберем ε такое, чтобы абсолютная разница текущего и предыдущего значений не превышала 9 знаков после запятой.

```
In [2]: eps = 1e-9
```

```
Out[2]: 1.0e-9
```

Введем единичный вектор, соответствующий размерности матрицы A.

```
In [3]: e = ones(size(A)[1])
```

```
Out[3]: 4-element Array{Float64,1}:
 1.0
 1.0
 1.0
 1.0
```

Выполним вычисления согласно приведенному алгоритму.

```
In [4]: num = A * e
        denom = e' * num
        prev = num / denom
        num = A * num
        denom = e' * num
        next = num / denom
        while true
            prev = next
            num = A * num
            denom = e' * num
            next = num / denom
            result = abs(next - prev) .> eps
            done = true
            for r in result
                if !r
                    done = false
                    break
                end
            end
```

```
        if done
            break
        end
    end
    approx = next
```

Out[4]: 4x1 Array{Float64,2}:

```
 0.623881
 0.195949
 0.113291
 0.0668799
```

Таким образом, мы получили приближенное значение $\overline{\mathbf{W}}$ главного собственного вектора с точностью до 9 знаков после запятой.

Найдем аналитические представления максимального собственного числа и главного собственного вектора для матрицы A через ее элементы. Введем символьные обозначения для наддиагональных элементов матрицы и выполним вспомогательные вычисления.

In [5]:
```
a = A[1, 2]
b = A[1, 3]
c = A[1, 4]
d = A[2, 3]
e = A[2, 4]
f = A[3, 4]

B = (d*f/e+e/d/f)+(a*e/c+c/a/e)+(b*f/c+c/b/f)+(a*d/b+b/a/d)
```

Out[5]: 11.916666666666668

In [6]:
```
C = 3-(a*d*f/c+c/a/d/f)-(a*e/b/f+b*f/a/e)-(c*d/a/e)-(c*d/b/e+b*e/c/d)
```

Out[6]: -9.916666666666668

In [7]:
```
x = B^2/2+8C-8
```

Out[7]: -16.3298611111111

In [8]:
```
y = (4(C+3)/3)^3
```

Out[8]: -784.3443072702335

In [9]:
```
r = Float64((x+sqrt(Complex(y)+x^2))^(1/3) +
(x-sqrt(Complex(y)+x^2))^(1/3))
```

Out[9]: 4.5214327417548645

Вычислим точное значение $\lambda_{\mathbf{max}}$ максимального собственного числа.

In [10]:
```
lmax = Float64((2+sqrt(r+4))/2 + sqrt((8-r)/4) + B/(2sqrt(r+4)))
```

Out[10]: 5.433240289413801

Определим точное значение главного собственного вектора.

```
In [11]: Q = (lmax-1)^3 + (c+f+e)*(lmax-1)^2 +
((a*e-3)+(b+d)f+c/a+c/b+e/d)*(lmax-1) +
(a*d*f-c-e-f+(b*e/d+b*f/a)+(c*d+a*e-a*d)/b+(c-b)/a/d)
```

Out[11]: 537.2229210859609

```
In [12]: first = c*(lmax-1)^2 + (a*e+b*f)*(lmax-1) + (a*d*f+b*e/d-c)
```

Out[12]: 304.4932032342129

```
In [13]: second = e*(lmax-1)^2 + (d*f+c/a)*(lmax-1) + (b*f/a+c*d/b-e)
```

Out[13]: 94.57288211091533

```
In [14]: third = f*(lmax-1)^2 + (e/d+c/b)*(lmax-1) + (c/a/d+a*e/b-f)
```

Out[14]: 66.82733896987288

```
In [15]: fourth = (lmax-1)^3 - 3*(lmax-1) - (b/a/d+a*d/b)
```

Out[15]: 71.32949677095984

```
In [16]: accurate = [first; second; third; fourth]
```

Out[16]: 4-element Array{Float64,1}:
```
304.493
 94.5729
 66.8273
 71.3295
```

Вычислим точное значение $\overline{\mathbf{W}}$ главного собственного вектора.

```
In [17]: accurate /= Q
```

Out[17]: 4-element Array{Float64,1}:
```
0.566791
0.17604
0.124394
0.132774
```

Оценим погрешность $\Delta\overline{W} = |\overline{W} - W|$.

```
In [18]: abs(accurate - approx)
```

Out[18]: 4x1 Array{Float64,2}:
```
0.0570894
0.0199085
0.0111034
0.0658946
```

Расчеты показывают, что абсолютная разница между точным и приближенным значениями главного собственного вектора составляет не более 10%.

Вычислим ранг матрицы A.

In [19]: rank(A)

Out[19]: 3

Ранг, не равный 1, означает, что матрица A не является согласованной.

1.2 Задание 2

1.2.1 Постановка задачи

Преобразовать матрицу парных сравнений A из задания 1 таким образом, чтобы она стала абсолютно согласованной. При этом оставить без изменений: * первую строку матрицы, * последнюю строку матрицы.

1.2.2 Ход решения

Матрица A из задания 1 имеет вид.

In [20]: A

Out[20]: 4x4 Array{Float64,2}:

```
1.0       4.0       6.0       8.0
0.25      1.0       3.0       2.0
0.166667  0.333333  1.0       3.0
0.125     0.5       0.333333  1.0
```

Добьемся абсолютной согласованности матрицы, не изменяя первую строку. Запомним исходные значения первого столбца матрицы.

In [21]:
```
tmp = deepcopy(A)
origin = A[:,1]
```

Out[21]: 4-element Array{Float64,1}:

```
1.0
0.25
0.166667
0.125
```

Для каждой из строк (2-3) будем последовательно вычитать строку (1), умноженную на первый элемент текущей строки для обнуления первого столбца.

```
In [22]: for i in 2:size(origin)[1]
             A[i,:] -= A[i,1] * A[1,:]
         end
         A
```

```
Out[22]: 4x4 Array{Float64,2}:
 1.0   4.0        6.0       8.0
 0.0   0.0        1.5       0.0
 0.0  -0.333333   0.0       1.66667
 0.0   0.0       -0.416667  0.0
```

Обнулим все элементы получившейся матрицы, кроме элементов 1-го столбца и 1-ой строки.

```
In [23]: A[2:4, 2:4] = 0
         A
```

```
Out[23]: 4x4 Array{Float64,2}:
 1.0  4.0  6.0  8.0
 0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0
 0.0  0.0  0.0  0.0
```

“Вернем” сделанные изменения обратно. После этого имеем матрицу, полученную путем применения преобразований Жордана-Гаусса к исходной матрице без изменения первой строки.

```
In [24]: for i in 2:size(origin)[1]
             A[i,:] += origin[i] * A[1,:]
         end
         A
```

```
Out[24]: 4x4 Array{Float64,2}:
 1.0       4.0       6.0  8.0
 0.25      1.0       1.5  2.0
 0.166667  0.666667  1.0  1.33333
 0.125     0.5       0.75 1.0
```

Вычислим ранг полученной матрицы.

```
In [25]: rank(A)
```

```
Out[25]: 1
```

Ранг равен 1. Это означает, что полученная матрица абсолютно согласована.

Теперь проделаем то же самое, только на этот раз оставим неизменной последнюю строку.

```
In [26]: A = tmp
```

```
Out[26]: 4x4 Array{Float64,2}:
         1.0       4.0       6.0       8.0
         0.25      1.0       3.0       2.0
         0.166667  0.333333  1.0       3.0
         0.125     0.5       0.333333  1.0
In [27]: origin = A[:,4]
Out[27]: 4-element Array{Float64,1}:
         8.0
         2.0
         3.0
         1.0
In [28]: for i in 1:size(A)[1]-1
             A[i,:] -= A[i,4] * A[4,:]
         end
         A
Out[28]: 4x4 Array{Float64,2}:
          0.0        0.0      3.33333   0.0
          0.0        0.0      2.33333   0.0
         -0.208333  -1.16667  0.0       0.0
          0.125      0.5      0.333333  1.0
In [29]: A[1:3, 1:3] = 0
         A
Out[29]: 4x4 Array{Float64,2}:
         0.0    0.0  0.0       0.0
         0.0    0.0  0.0       0.0
         0.0    0.0  0.0       0.0
         0.125  0.5  0.333333  1.0
In [30]: for i in 1:size(origin)[1]-1
             A[i,:] += origin[i] * A[4,:]
         end
         A
Out[30]: 4x4 Array{Float64,2}:
         1.0    4.0  2.66667   8.0
         0.25   1.0  0.666667  2.0
         0.375  1.5  1.0       3.0
         0.125  0.5  0.333333  1.0
In [31]: rank(A)
Out[31]: 1
```

Таким образом, получили абсолютно согласованную матрицу, не изменяя последнюю строку.

1.3 Задание 3

1.3.1 Постановка задачи

Найдите агрегированную оценку двух экспертов, если матрица парных сравнений первого эксперта имеет вид, представленный в задании 1, а матрица парных сравнений второго имеет вид:

$$A = \begin{pmatrix} 1 & 3 & 6 & 8 \\ 1/3 & 1 & 4 & 5 \\ 1/6 & 1/4 & 1 & 3 \\ 1/8 & 1/5 & 1/3 & 1 \end{pmatrix}.$$

1.3.2 Ход решения

Для объединения оценок суждений двух экспертов строится матрица с средним геометрическим оценок. В данной задаче будем считать, что суждения двух экспертов обладают одинаковой степенью значимости. Введем оценки первого (A_1) и второго (A_2) экспертов.

```
In [1]: A1 = [1 4 6 8; 1/4 1 3 2; 1/6 1/3 1 3; 1/8 1/2 1/3 1]

Out[1]: 4x4 Array{Float64,2}:
        1.0       4.0       6.0       8.0
        0.25      1.0       3.0       2.0
        0.166667  0.333333  1.0       3.0
        0.125     0.5       0.333333  1.0

In [2]: A2 = [1 3 6 8; 1/3 1 4 5; 1/6 1/4 1 3; 1/8 1/5 1/3 1]

Out[2]: 4x4 Array{Float64,2}:
        1.0       3.0   6.0       8.0
        0.333333  1.0   4.0       5.0
        0.166667  0.25  1.0       3.0
        0.125     0.2   0.333333  1.0
```

Построим результирующую матрицу A со средним геометрическим оценок.

```
In [3]: using Stats
        A = map((x, y) -> geomean([x, y]), A1, A2)

Out[3]: 4x4 Array{Float64,2}:
        1.0       3.4641    6.0       8.0
        0.288675  1.0       3.4641    3.16228
        0.166667  0.288675  1.0       3.0
        0.125     0.316228  0.333333  1.0
```

Таким образом, получили агрегированную оценку двух экспертов при условии, что суждения экспертов обладают одинаковой степенью значимости.

1.4 Задание 4

1.4.1 Постановка задачи

Найти агрегированную оценку экспертов из задания 3, при условии, что квалификация первого эксперта имеет вес 3 (первый эксперт более квалифицированный), а второго - 1.

1.4.2 Ход решения

Расчет агрегированной оценки в случае привлечения n экспертов, имеющих различную значимость α_k, $k = \overline{1,n}$, осуществляется по формуле:

$$\alpha_{ij} = \sqrt[n]{\prod_{k=1}^{n} a_{ij}^{\alpha_k}}, \sum_{k=1}^{n} \alpha_k = 1.$$

Введем в рассмотрение веса w^s (квалификацию) экспертов.

```
In [4]: ws = [3 1]
```

```
Out[4]: 1x2 Array{Int32,2}:
    3  1
```

Рассчитаем значимость суждения каждого эксперта таким образом, чтобы общая значимость была равна 1.

```
In [5]: a = [w / sum(ws) for w in ws]
```

```
Out[5]: 1x2 Array{Float64,2}:
    0.75  0.25
```

Рассчитаем результирующую матрицу.

```
In [6]: A = map((x, y) -> geomean([x^a[1], y^a[2]]), A1, A2)
```

```
Out[6]: 4x4 Array{Float64,2}:
    1.0       1.92936   2.44949  2.82843
    0.518307  1.0       1.79547  1.58583
    0.408248  0.556957  1.0      1.73205
    0.353553  0.630583  0.57735  1.0
```

Таким образом, получили агрегированную оценку двух экспертов при условии, что суждения экспертов обладают разной степенью значимости.

1.5 Задание 5

1.5.1 Постановка задачи

Для иерархической структуры, представленной на рисунке, определите приоритет провайдера, выполнив иерархический синтез.

Матрица сравнения критериев относительно цели имеет вид:

$$A = \begin{pmatrix} 1 & 4 & 6 & 2 & 7 \\ 1/4 & 1 & 3 & 4 & 2 \\ 1/6 & 1/3 & 1 & 2 & 1 \\ 1/2 & 1/4 & 1/2 & 1 & 1/3 \\ 1/7 & 1/2 & 1 & 3 & 1 \end{pmatrix}.$$

Матрицы сравнения альтернатив относительно критериев считаются заданными.

1.5.2 Ход решения

Иерархический синтез используется для общего ранжирования альтернатив относительно цели, т.е. для подсчета количественной оценки качества альтернатив. Действовать будем согласно алгоритму.

Шаг 1. Для каждого элемента иерархии построить матрицы парных сравнений элементов иерархии следующего уровня. На первом уровне иерархии будем использовать матрицу сравнения относительно удовлетворения провайдером [УП].

$$[\text{УП}] = \begin{pmatrix} & & \underline{\text{Т}} & \underline{\text{С}} & \underline{\text{Д}} & \underline{\text{О}} & \underline{\text{У}} \\ \text{Т} & | & 1 & 4 & 6 & 2 & 7 \\ \text{С} & | & 1/4 & 1 & 3 & 4 & 2 \\ \text{Д} & | & 1/6 & 1/3 & 1 & 2 & 1 \\ \text{О} & | & 1/2 & 1/4 & 1/2 & 1 & 1/3 \\ \text{У} & | & 1/7 & 1/2 & 1 & 3 & 1 \end{pmatrix}.$$

```
In [1]: A = [1 4 6 2 7; 1/4 1 3 4 2; 1/6 1/3 1 2 1; 1/2 1/4 1/2 1 1/3; 1/7 1/2 1 3 1]
Out[1]: 5x5 Array{Float64,2}:
    1.0        4.0        6.0  2.0  7.0
    0.25       1.0        3.0  4.0  2.0
    0.166667   0.333333   1.0  2.0  1.0
    0.5        0.25       0.5  1.0  0.333333
    0.142857   0.5        1.0  3.0  1.0
```

Введем функцию вычисления приближенных значений максимального собственного числа и главного собственного вектора.

```
In [2]: function eigvalvecapprox(A)
            eps = 1e-9
            e = ones(size(A)[1])
            num = A * e
            denom = e' * num
            prev = num / denom
            num = A * num
            denom = e' * num
            next = num / denom
            while true
                prev = next
                num = A * num
                denom = e' * num
                next = num / denom
                result = abs(next - prev) .> eps
                done = true
                for r in result
                    if !r
                        done = false
                        break
                    end
                end
                if done
                    break
                end
            end
            Wa = next
            lmax = (e' * A * Wa) / (e' * Wa)
            return lmax[1], Wa
        end
```

```
Out[2]: eigvalvecapprox (generic function with 1 method)
```

Вычислим приближенные значения максимального собственного числа и главного собственного вектора для матрицы A.

```
In [3]: val, vec = eigvalvecapprox(A)
```

```
Out[3]: (5.643273525681777,
        [0.505324; 0.205149; … ; 0.0867938; 0.110393])
```

Введем функцию для вычисления индекса однородности (индекса согласованности)

$$I_S = \frac{\lambda_{max} - n}{n - 1}.$$

In [4]:
```
function ids(lmax, n)
    return (lmax - n) / (n - 1)
end
```

Out[4]: ids (generic function with 1 method)

Вычислим индекс однородности для матрицы A.

In [5]:
```
Is = ids(val, size(A)[1])
```

Out[5]: 0.16081838142044425

Также, потребуется функция для вычисления отношения однородности (отношения согласованности)

$$R_S = \frac{I_S}{\hat{I}_S}.$$

In [6]:
```
function rs(idx, n)
    mean = 0
    if n == 3
        mean = 0.58
    elseif n == 4
        mean = 0.9
    elseif n == 5
        mean = 1.12
    end
    return idx / mean
end
```

Out[6]: rs (generic function with 1 method)

Вычислим отношение однородности для матрицы A.

In [7]:
```
Rs = rs(Is, size(A)[1])
```

Out[7]: 0.14358784055396806

In [8]:
```
As = []
push!(As, [1 4 6 8; 1/4 1 3 2; 1/6 1/3 1 3; 1/8 1/2 1/3 1])
push!(As, [1 3 6 9; 1/3 1 1/4 2; 1/6 4 1 3; 1/9 1/2 1/3 1])
push!(As, [1 8 3 2; 1/8 1 3 2; 1/3 1/3 1 1; 1/2 1/2 1 1])
push!(As, [1 4 6 1; 1/4 1 3 2; 1/6 1/3 1 3; 1 1/2 1/3 1])
push!(As, [1 3 2 8; 1/3 1 1/4 1/2; 1/2 1/4 1 3; 1/8 2 1/3 1])
```

Для каждой из матриц найти максимальные собственные значения (они потребуются для оценки однородности суждений) и главные собственные векторы, элементы которых равны приоритетам соответствующих элементов следующего уровня иерархии. Для этого воспользуемся алгоритмом вычисления точного значения максимального собственного числа и главного собственного вектора из задания 1, который оформим в виде функции.

In [9]:
```
function eigvalvecaccurate(A)
    a = Complex(A[1, 2])
    b = Complex(A[1, 3])
    c = Complex(A[1, 4])
    d = Complex(A[2, 3])
    e = Complex(A[2, 4])
    f = Complex(A[3, 4])
    B = (d*f/e+e/d/f)+(a*e/c+c/a/e)+(b*f/c+c/b/f)+(a*d/b+b/a/d)
    C = 3-(a*d*f/c+c/a/d/f)-(a*e/b/f+b*f/a/e)-(c*d/a/e)-(c*d/b/e+b*e/c/d)
    x = B^2/2+8C-8
    y = (4(C+3)/3)^3
    r = (x+sqrt(y+x^2))^(1/3) + (x-sqrt(y+x^2))^(1/3)
    lmax = (2+sqrt(r+4))/2 + sqrt((8-r)/4) + B/(2sqrt(r+4))
    Q = (lmax-1)^3 + (c+f+e)*(lmax-1)^2 +
    ((a*e-3)+(b+d)f+c/a+c/b+e/d)*(lmax-1) +
    (a*d*f-c-e-f+(b*e/d+b*f/a)+(c*d+a*e-a*d)/b+(c-b)/a/d)
    first = c*(lmax-1)^2 + (a*e+b*f)*(lmax-1) + (a*d*f+b*e/d-c)
    second = e*(lmax-1)^2 + (d*f+c/a)*(lmax-1) + (b*f/a+c*d/b-e)
    third = f*(lmax-1)^2 + (e/d+c/b)*(lmax-1) + (c/a/d+a*e/b-f)
    fourth = (lmax-1)^3 - 3*(lmax-1) - (b/a/d+a*d/b)
    accurate = [first; second; third; fourth]
    accurate /= Q
    return lmax, accurate
end
```

Out[9]: eigvalvecaccurate (generic function with 1 method)

In [10]:
```
Vals = []
Vecs = []
Iss = []
Rss = []

for A in As
    maxval, mainvec = eigvalvecaccurate(A)
    Is = ids(maxval, size(A)[1])
```

```
        Rs = rs(Is, size(A)[1])
        push!(Vals, real(maxval))
        push!(Vecs, map(real, mainvec))
        push!(Iss, real(Is))
        push!(Rss, real(Rs))
    end
```

Выполним иерархический синтез.

In [11]:
```
W = Vecs' * vec
```

Out[11]:
```
1x1 Array{Any,2}:
 [0.530109 0.152196 0.153613 0.164082]
```

Таким образом, видно, что в системе предпочтений индивидуума наибольший приоритет имеет провайдер 1 (“Связьинформ”).

1.6 Задание 6

1.6.1 Постановка задачи

Оценить отношение согласованности иерархии из задания 5.

1.6.2 Ход решения

Завершается алгоритм анализом однородности построенной иерархии. Для этого вычислим индекс однородности и отношение однородности в соответствии с третьим шагом алгоритма.

In [12]:
```
I = real(Is + vec' * Iss)
```

Out[12]:
```
1-element Array{Float64,1}:
 1.05641
```

In [13]:
```
Ihat = real(Rs + vec' * Rss)
```

Out[13]:
```
1-element Array{Float64,1}:
 1.17379
```

In [14]:
```
Rs = I / Ihat
```

Out[14]:
```
1x1 Array{Float64,2}:
 0.9
```

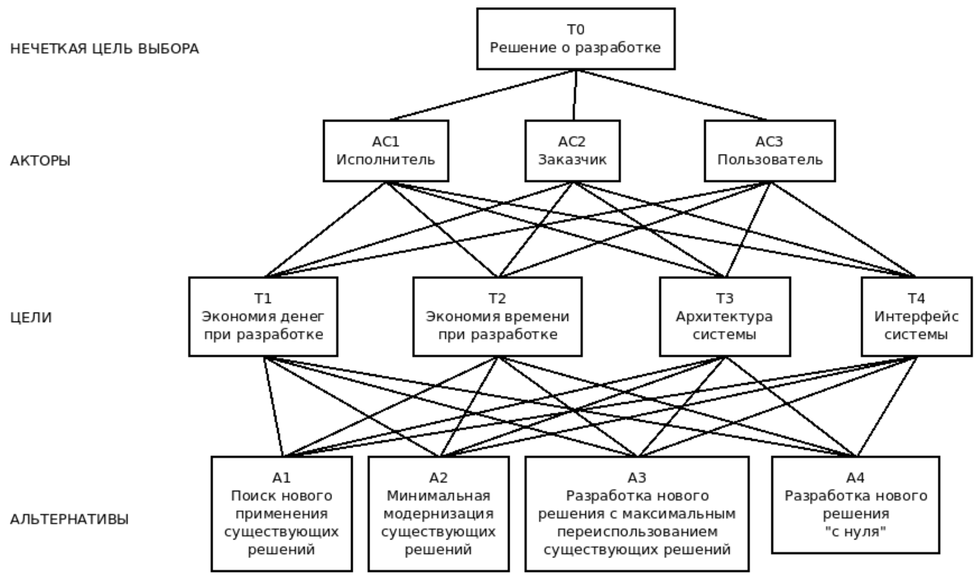
Однородность иерархии принято считать удовлетворительной при $R_s \leq 0.1$. Неудовлетворительную однородность построенной иерархии можно объяснить тем, что на шаге 1 были построены несогласованные матрицы.

2 Метод анализа иерархий

С задачами принятия решений человек сталкивается повседневно. При этом в ситуациях, когда последствия принимаемых решений не связаны с заметным материальным или социальным ущербом, предпочтение отдается, как правило, той или иной альтернативе на основании интуиции и опыта субъекта, осуществляющего выбор.

Однако существует и другой класс задач принятия решений, которые можно охарактеризовать как "ответственные". Это задачи, где принятые решения влияют на разработку сложных систем, которыми будут пользоваться тысячи человек. В таких задачах лицо (лица), принимающее решение, зачастую должно учитывать не только множество критериев качества, но и различные цели, которые преследуют различные участники.

Рассмотрим следующую задачу выбора. Необходимо принять решение о разработке новой программной системы. При этом существуют различные подходы к процессу разработки (см. рисунок 1).



Иерархическая структура.

Чтобы понять, какая из альтернатив предпочтительнее, необходимо произвести экспертную оценку и проанализировать доминантные иерархии (один из методов принятия решений).

Конечный результат определяется различными социальными группами (акторами) и целями, которые преследуют акторы.

Какой из акторов обладает наибольшим влиянием на разработку системы?				
Решение о разработке	AC_1	AC_2	AC_3	Собственный вектор
AC_1	1	0.41	0.70	0.2103
AC_2	2.47	1	0.70	0.3813
AC_3	1.45	1.45	1	0.4086
			λ_{max}	3.11
			R_s	0.095
Наибольшее влияние на разработку системы имеют заказчик (AC_2) и пользователь (AC_3)				

Какие из целей важнее для исполнителя?					
Исполнитель	T_1	T_2	T_3	T_4	Собственный вектор
T_1	1	0.59	0.15	0.20	0.0640
T_2	1.71	1	0.17	0.24	0.0898
T_3	6.81	6.09	1	1.92	0.5228
T_4	5.13	4.33	0.53	1	0.3236
				λ_{max}	4.07
				R_s	0.026
Для исполнителя важнее всего архитектура (T_3) и интерфейс системы (T_4)					

Какие из целей важнее для заказчика?					
Заказчик	T_1	T_2	T_3	T_4	Собственный вектор
T_1	1	0.59	5.13	0.85	0.2528
T_2	1.71	1	6.09	1.00	0.3587
T_3	0.20	0.17	1	0.15	0.0532
T_4	1.19	1.00	6.81	1	0.3355
				λ_{max}	4.05
				R_s	0.019
Для заказчика важнее всего экономия времени при разработке (T_2) и интерфейс системы (T_4)					

Какие из целей важнее для пользователя?					
Пользователь	T_1	T_2	T_3	T_4	Собственный вектор
T_1	1	0.59	1.71	0.14	0.1085
T_2	1.71	1	1.71	0.34	0.1763
T_3	0.59	0.59	1	0.14	0.0833
T_4	7.40	3.01	7.40	1	0.6321
				λ_{max}	4.09
				R_s	0.034
Для пользователя важнее всего интерфейс системы (T_4)					

Какие из альтернатив предпочтительнее для экономии денег при разработке?					
Экономия денег	A_1	A_2	A_3	A_4	Собственный вектор
A_1	1	1.00	2.47	6.09	0.4097
A_2	1.00	1	1.00	2.47	0.2631
A_3	0.41	1.00	1	6.09	0.2589
A_4	0.17	0.41	0.17	1	0.0685
				λ_{max}	4.23
				R_s	0.084
Для экономии денег при разработке лучше всего попытаться найти новое применение существующих решений (A_1)					

Какие из альтернатив предпочтительнее для экономии времени при разработке?					
Экономия времени	A_1	A_2	A_3	A_4	Собственный вектор
A_1	1	1.53	1.71	5.60	0.3724
A_2	0.66	1	4.72	4.72	0.3995
A_3	0.59	0.22	1	4.72	0.1702
A_4	0.18	0.22	0.2	1	0.0581
				λ_{max}	4.33
				R_s	0.120
Для экономии времени при разработке лучше попытаться найти новое применение существующих решений (A_1) либо минимально их модернизировать (A_2)					

Какие из альтернатив предпочтительнее для проектирования наилучшей архитектуры системы?					
Архитектура системы	A_1	A_2	A_3	A_4	Собственный вектор
A_1	1	2.47	0.20	0.17	0.0783
A_2	0.41	1	0.20	0.17	0.0533
A_3	5.13	5.13	1	0.15	0.2102
A_4	6.09	6.09	6.81	1	0.6583
				λ_{max}	4.48
				R_s	0.177
Для проектирования качественной архитектуры системы лучше начать разрабатывать ее "с нуля"(A_4)					

Какие из альтернатив предпочтительнее для создания удобного интерфейса системы?					
Интерфейс системы	A_1	A_2	A_3	A_4	Собственный вектор
A_1	1	1.71	0.41	0.31	0.1438
A_2	0.59	1	0.59	0.31	0.1196
A_3	2.47	1.71	1	0.31	0.2265
A_4	3.28	3.28	3.28	1	0.5102
				λ_{max}	4.16
				R_s	0.058
Для создания удобного интерфейса системы лучше начать разрабатывать ее "с нуля"(A_4)					