第四章基于属性的软件可信性度量模型

陈仪香

2024年10月22日

Outline

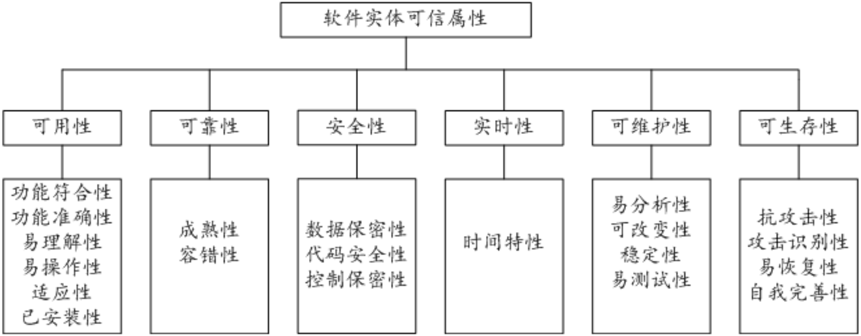
第4章基于属性的软件可信度量模型

本章介绍基于多维属性的软件可信性度量性质集和基于属性分解的软件可信性度量性质集。依据公理化方法分别建立基于属性的软件可信性度量模型、基于属性划分的软件可信性度量模型和基于属性分解的软件可信性度量模型。

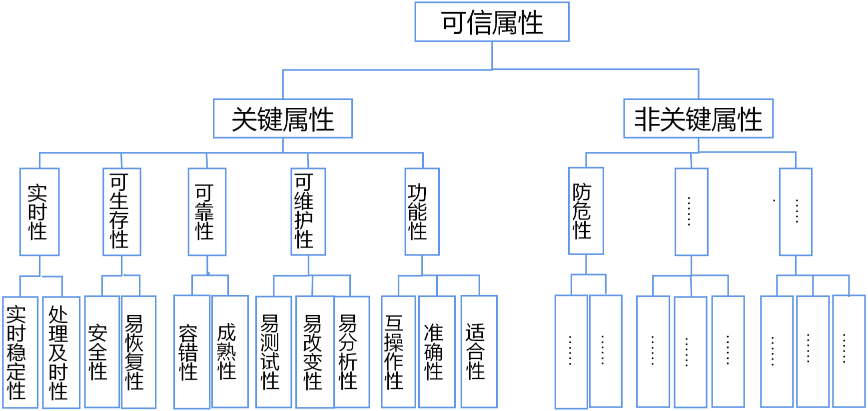
分层原因

构成不同高层次属性的低层次属性集合的交集可能非空，
这样各高层次属性间可能并不是毫不相关的，
意味着属性之间可以发生替代。

软件可信属性分解模型


软件实体可信属性
可用性
可靠性
安全性
实时性
可维护性
可生存性
功能符合性
功能准确性
易理解性
易操作性
适应性
已安装性
成熟性
容错性
数据保密性
代码安全性
控制保密性
时间特性
易分析性
可改变性
稳定性
易测试性
抗攻击性
攻击识别性
易恢复性
自我完善性

软件可信属性分解模型-关键属性与非关键属性



最简单度量模型–不分关键属性和非关键属性

度量模型0：

$$T_0 = y_1^{\alpha_1} y_2^{\alpha_2} \cdots y_m^{\alpha_m}$$

Model 1

$$T_1 = \frac{10}{11}(\frac{y_{\min}}{10})^{\varepsilon} y_1^{\alpha\alpha_1} y_2^{\alpha\alpha_2} \cdots y_m^{\alpha\alpha_m} + \frac{10}{11} y_{m+1}^{\beta\beta_{m+1}} y_{m+2}^{\beta\beta_{m+2}} \cdots y_{m+s}^{\beta\beta_{m+s}}$$

Model 2

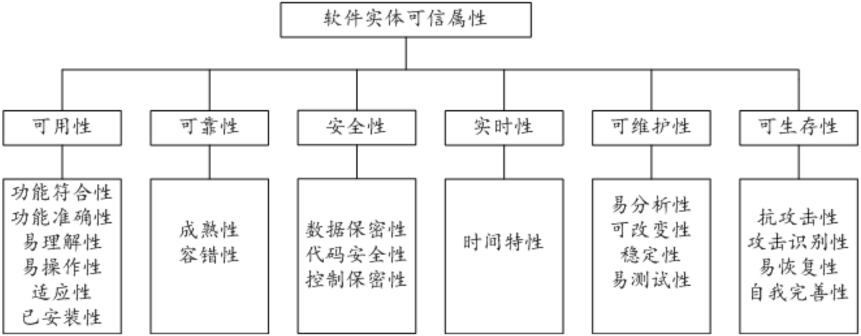
$$T_2 = \frac{10}{11}(\frac{y_{\min}}{10})^{\varepsilon} y_1^{\alpha\alpha_1} y_2^{\alpha\alpha_2} \cdots y_m^{\alpha\alpha_m} + \frac{10}{11} y_{\min'}^{\beta\beta_i},$$

where $y_{\min}, \varepsilon$ and y_i are the same as those t in model 1 and $y_{\min'} = \min\{y_j \mid m+1 \leq j \leq m+s\}$。

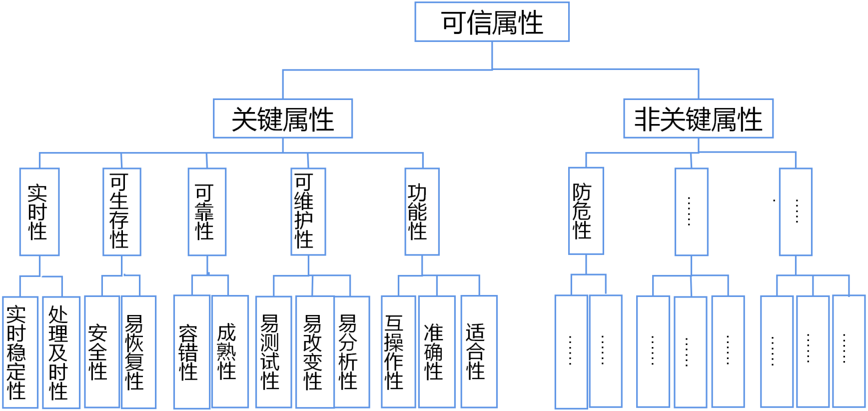
Model 3

$$T_3 = \{\alpha[\min_{1\leq i\leq m}\{(\frac{y_i}{10})^{\epsilon}\}y_1^{\alpha_1}y_2^{\alpha_2}\cdots y_m^{\alpha_m}]^{-\rho}+\beta[y_{m+1}^{\beta_{m+1}}y_{m+2}^{\beta_{m+2}}\cdots y_{m+s}^{\beta_{m+s}}]^{-\rho}\}^{-\frac{1}{\rho}}$$

软件可信属性分解模型


软件实体可信属性
可用性
可靠性
安全性
实时性
可维护性
可生存性
功能符合性
功能准确性
易理解性
易操作性
适应性
已安装性
成熟性
容错性
数据保密性
代码安全性
控制保密性
时间特性
易分析性
可改变性
稳定性
易测试性
抗攻击性
攻击识别性
易恢复性
自我完善性

软件可信属性分解模型-关键属性与非关键属性



可信属性度量

设y是一软件可信属性，有n个子属性：$x_1, x_2, \cdots, x_n$。y是$x_1, x_2, \cdots, x_n$,即

$$y = g(x_1, x_2, \cdots, x_n)$$

我们的任务是寻找合适的函数g。

可信属性度量模型的性质

Claim

1. 非负性： 非负性用来描述软件可信性为非负值。

$$0 \leq y$$

Claim

2. 属性比例合适性： 比例合适性是指各可信子属性应该有合适的比例假设。

$$(\exists c，d \in \Re^{+})\ c \leq \frac{x_i}{x_j} \leq d，\ 1 \leq i，\ j \leq n$$

Claim

3. 单调性： 可信属性值关于可信子属性值单调增加。

$$\partial y/\partial x_i \geq 0，\ 1 \leq i \leq n$$

可信属性度量模型的性质

Claim

4. 凝聚性： 凝聚性表示随着可信子属性值的增加，其对可信属性值增加的贡献效率在减少。

$$\partial^2 y/\partial {x_i}^2 \leq 0，\ 1 \leq i \leq n$$

Claim

5. 灵敏性：灵敏性描述可信子属性的百分比变化所导致可信属性的百分比变化情况。可信值最小子属性改进对相应可信属性改进有较大影响。

$$\delta_i = \frac{\partial y}{\partial x_i}\frac{x_i}{y} \geq 0，\ 1 \leq i \leq n$$

可信属性度量模型的性质

Claim

6. 替代性：可信子属性间可进行替代。

$$(\exists c, \ d \in \Re^{+}) \ \ c \leq \sigma_{ij} \leq d$$

σ_{ij}为构成可信属性y的可信子属性x_i和x_j间替代性值，具体定义如下：对于任意的i,j满足 $1 \leq i,j \leq n$都有

$$\sigma_{ij} = \frac{\frac{d(x_i/x_j)}{x_i/x_j}}{\frac{d(-\frac{\partial y/\partial x_j}{\partial y/\partial x_i})}{-\frac{\partial y/\partial x_j}{\partial y/\partial x_i}}} = \frac{d(x_i/x_j)}{d(-\frac{\partial y/\partial x_j}{\partial y/\partial x_i})} \times \frac{-\frac{\partial y/\partial x_j}{\partial y/\partial x_i}}{x_i/x_j} \qquad (1)$$

σ_{ij}越小，则x_i与x_j间越难于替代。

可信属性度量模型的性质

Claim

7. 可期望性： 期望性是指如果所有可信子属性均达到用户预期，则软件可信属性也要满足用户预期，并且可信属性值不超过最大子属性值。

$$(x_0 \le \min\{x_1, \cdots, x_n\}) \text{ 推出 } (x_0 \le y \le \max\{x_1, \cdots, x_n\})$$

其中，x_0 是用户关于所有可信子属性的最低预期值。

可信属性度量模型

Claim

模型*1*：$y = x_1^{\gamma_1} x_2^{\gamma_2} \cdots x_n^{\gamma_n}$

模型1满足可信属性度量的七条性质。

Claim

模型*2*：$y = (\sum_{i=1}^{n} \omega_i x_i^{-\rho_y})^{-\frac{1}{\rho_y}}$，$1 \leq i \leq n$，$1 \leq x_i \leq 10$。

1. ρ_y：与可信属性y相匹配的参数，它是构成可信属性y的可信子属性间替代性相关的参数，满足$0 < \rho_y$，且其值越大，则可信子属性间替代性越难；
2. w_i：可信属性y的可信子属性$x_i(1 \leq i \leq n)$权重，满足$\sum_{i=1}^{n} \omega_i = 1$，$0 \leq \omega_i \leq 1$。

模型2满足可信属性度量的七条性质吗？

可信属性度量模型2性质

模型2：$y=(\sum_{i=1}^{n}\omega_i x_i^{-\rho_y})^{-\frac{1}{\rho_y}}$，$1\le i\le n$，$1\le x_i\le 10$。

Claim

1. 非负性：$1\le y\le 10$, 这是因为$1\le x_i\le 10$。

Claim

2. 比例合适性：$\frac{1}{10}\le\frac{x_i}{x_j}\le\frac{10}{1}$, 这是因为$1\le x_i\le 10$。

可信属性度量模型2性质

模型2：$y=(\sum_{i=1}^{n}\omega_i x_i^{-\rho_y})^{-\frac{1}{\rho_y}}$，$1\leq i\leq n$，$1\leq x_i\leq 10$。

Claim

3. 单调性：$\frac{\partial y}{\partial x_i}\geq 0$

$$
\begin{aligned}
\frac{\partial y}{\partial x_i} &= -\frac{1}{\rho_y}(\sum_{i=1}^{n}\omega_i x_i^{-\rho_y})^{-\frac{1}{\rho_y}-1}\cdot\frac{\partial(\omega_i x_i^{-\rho_y})}{\partial x_i} \\
&= -\frac{1}{\rho_y}(\sum_{i=1}^{n}\omega_i x_i^{-\rho_y})^{-\frac{1}{\rho_y}-1}\cdot\omega_i\cdot-\rho_y\cdot x_i^{-\rho_y-1} \\
&= \omega_i(\sum_{i=1}^{n}\omega_i x_i^{-\rho_y})^{-\frac{1}{\rho_y}-1}\cdot x_i^{-\rho_y-1} \\
&\geq 0
\end{aligned}
$$

可信属性度量模型2性质

$$\frac{\partial y}{\partial x_i} = \omega_i (\sum_{i=1}^{n} \omega_i x_i^{-\rho_y})^{-\frac{1}{\rho_y}-1} \cdot x_i^{-\rho_y-1}$$

Claim

4. 凝聚性：$\frac{\partial^2 y}{\partial x_i^2} \leq 0$。

对于$1 \leq i \leq n$有

$$\frac{\partial^2 y}{\partial {x_i}^2} = \omega_i (1+\rho_y)(\sum_{k=1}^{n} \omega_k x_k^{-\rho_y})^{-\frac{1}{\rho_y}-1} \cdot x_k^{-\rho_y-2} (\frac{\omega_i x_i^{-\rho_y}}{\sum_{k=1}^{n} \omega_k x_k^{-\rho_y}} - 1) \leq 0$$

所以可信属性y关于可信子属性x_i满足凝聚性。

可信属性度量模型2性质

Claim

凝聚性证明

$$
\begin{aligned}
\frac{\partial^2 y}{\partial x_i^2} &= \omega_i \cdot \left(-\frac{1}{\rho_y}-1\right)\cdot \left(\sum_{k=1}^{n}\omega_k x_k^{-\rho_y}\right)^{-\frac{1}{\rho_y}-2}\cdot \omega_i(-\rho_y)\cdot x_i^{-\rho_y-1}\cdot x_i^{-\rho_y-1}\\
&\quad +\omega_i\left(\sum_{k=1}^{n}\omega_k x_k^{-\rho_y}\right)^{-\frac{1}{\rho_y}-1}\cdot(-\rho_y-1)\cdot x_i^{-\rho_y-2}\\
&= \omega_i\cdot(1+\rho_y)\left(\sum_{k=1}^{n}\omega_k x_k^{-\rho_y}\right)^{-\frac{1}{\rho_y}-2}\cdot\omega_i\cdot x_i^{-\rho_y-1}\cdot x_i^{-\rho_y-1}\\
&\quad -\omega_i(1+\rho_y)\left(\sum_{k=1}^{n}\omega_k x_k^{-\rho_y}\right)^{-\frac{1}{\rho_y}-1}\cdot x_i^{-\rho_y-2}\\
&= \omega_i(1+\rho_y)\left(\sum_{k=1}^{n}\omega_k x_k^{-\rho_y}\right)^{-\frac{1}{\rho_y}-1}\cdot x_i^{-\rho_y-2}\left(\left(\left(\sum_{k=1}^{n}\omega_k x_k^{-\rho_y}\right)^{-1}\right)\cdot\omega_i x_i^{-\rho_y}-1\right)\\
&= \omega_i(1+\rho_y)\left(\sum_{k=1}^{n}\omega_k x_k^{-\rho_y}\right)^{-\frac{1}{\rho_y}-1}\cdot x_i^{-\rho_y-2}\left(\frac{\omega_i x_i^{-\rho_y}}{\sum_{k=1}^{n}\omega_k x_k^{-\rho_y}}-1\right)\\
&\le 0
\end{aligned}
$$

可信属性度量模型2性质

$$\frac{\partial y}{\partial x_i} = \omega_i(\sum_{i=1}^{n} \omega_i x_i^{-\rho_y})^{-\frac{1}{\rho_y}-1} \cdot x_i^{-\rho_y - 1}$$

Claim

5 灵敏性：$\delta_i \geq 0$。

可信属性y关于可信子属性x_i $(1 \leq i \leq n)$的灵敏性有

$$\begin{aligned} \delta_i &= \frac{\partial y}{\partial x_i} \cdot \frac{x_i}{y} \\ &= \omega_i(\sum_{i=1}^{n} \omega_i x_i^{-\rho_y})^{-\frac{1}{\rho_y}-1} \cdot x_i^{-\rho_y-1} \cdot \frac{x_i}{(\sum_{k=1}^{n} \omega_k x_k^{-\rho_y})^{-\frac{1}{\rho_y}}} \\ &= \omega_i(\sum_{k=1}^{n} x_k^{-\rho_y})^{-1} \cdot x_i^{-\rho_y} \\ &= \frac{\omega_i x_i^{-\rho_y}}{\sum_{k=1}^{n} \omega_k x_k^{-\rho_y}} \geq 0 \end{aligned}$$

可信属性度量模型2性质

$$\frac{\partial y}{\partial x_i} = \omega_i(\sum_{i=1}^{n} \omega_i x_i^{-\rho_y})^{-\frac{1}{\rho_y}-1} \cdot x_i^{-\rho_y-1} = \omega_i \cdot y^{-1} \cdot x_i^{-\rho_y-1}$$

$$\sigma_{ij} = \frac{d(x_i/x_j)}{d(-\frac{\partial y/\partial x_j}{\partial y/\partial x_i})} \times \frac{-\frac{\partial y/\partial x_j}{\partial y/\partial x_i}}{x_i/x_j} \text{ 记} t = \frac{x_i}{x_j}$$

Claim

6. 替代性：$\sigma_{ij} = \frac{1}{\rho_y+1}$。

$$\begin{aligned}
\sigma_{ij} &= \frac{dt}{d(-\frac{\omega_j}{\omega_i}\cdot(\frac{1}{t})^{-\rho_y-1})} \times \frac{-\frac{\omega_j}{\omega_i}(\frac{1}{t})^{-\rho_y-1}}{t} \\
&= \frac{1}{\frac{d(\frac{\omega_j}{\omega_i}\cdot(\frac{1}{t})^{-\rho_y-1})}{dt}} \times \frac{\frac{\omega_j}{\omega_i}(\frac{1}{t})^{-\rho_y-1}}{t} \\
&= \frac{1}{\frac{\omega_j}{\omega_i}(-\rho_y-1)\cdot(\frac{1}{t})^{-\rho_y-2}\cdot\frac{-1}{t^2}} \times \frac{\frac{w_j}{\omega_i}(\frac{1}{t})^{-\rho_y-1}}{t} \\
&= \frac{1}{1+\rho_y}
\end{aligned}$$

可信属性度量模型2性质

模型2：$y = (\sum_{i=1}^{n} \omega_i x_i^{-\rho_y})^{-\frac{1}{\rho_y}}$，$1 \leq i \leq n$，$1 \leq x_i \leq 10$。

Claim

7. 可期望性：$\sigma_{ij} = \frac{1}{\rho_y+1}$。

事实上，若所有的$x_i = x_0$则

有$y = (\sum_{i=1}^{n} \omega_i x_0^{-\rho_y})^{-\frac{1}{\rho_y}} = (x_0^{-\rho_y})^{-\frac{1}{\rho_y}} = x_0$。从而

若$x_0 \leq x_i \ (1 \leq i \leq n)$则$x_0 \leq y \leq \max\{x_i \mid 1 \leq i \leq n\}$。

可信属性度量模拟

模型1：$y = x_1^{\gamma_1} x_2^{\gamma_2} \cdots x_n^{\gamma_n}$

模型2：$y = (\sum_{i=1}^{n} \omega_i x_i^{-\rho_y})^{-\frac{1}{\rho_y}}$，$1 \le i \le n$，$1 \le x_i \le 10$。

设有4个软件系统，分别编号为1,2,3,4。它们都有可信安全属性y，含有4个子属性，编号为x_1, x_2, x_3, x_4, 其权重分别为$\gamma_1(\omega_1) = 0.25, \gamma_2(\omega_2) = 0.20, \gamma_3(\omega_3) = 0.25, \gamma_4(\omega_4) = 0.30$。参数$\rho_y = 0.10, 0.20$。按照模型1($y_1$)和模型2($y_2$)分别计算安全属性$y$的可信度量值，注意：模型1是不需要参数$\rho$的。

编号	x_1	x_2	x_3	x_4	ρ_y	y_1	y_2
1	8	9	10	8	0.10	8.66	8.66
	8	9	10	8	0.20		8.65
2	6	7	5	6	0.10	5.91	5. 90
	6	7	5	6	0.20		5.90
3	9	10	9	8	0.10	8.87	8.87
	9	10	9	8	0.20		8.87
4	3	4	5	4	0.10	3.94	3.93
	3	4	5	4	0.10		3.92

软件可信性度量模型T_3代码

```
m = 4 ♯ 关键属性数
s = 3 ♯ 非关键属性数
sub_alpha = [0.2, 0.3, 0.35, 0.15]
sub_beta = [0.25, 0.35, 0.40]
alpha = 0.75
beta = 0.25
y = list(map(float, input("输入前0个关键属性值与后1个非关键属性值: ".format(m, s)).split()))
epsilon, rho = map(float, input("输入epsilon与rho的值: ").split())
min_part = None
for i in range(0, m):
    val = pow(y[i] / 10, epsilon)
    if min_part is None or val ¡ min_part:
        min_ part = val
```

软件可信性度量模型T_3代码

```
key_part = min_part
for i in range(0, m):
    key_ part *= pow(y[i], sub_alpha[i])
non_key_part = 1
for i in range(0, s):
    non_key_ part *= pow(y[m+i], sub_beta[i])
result = pow(alpha * pow(key_part, -rho) + beta *
pow(non_key_part, -rho), -1 / rho)
print(result)
```

可信属性度量模型1

```
x1, x2, x3, x4 = map(float, input().split())
print(pow(x1, 0.25) * pow(x2, 0.20) * pow(x3, 0.25) * pow(x4,
0.30) )
```

可信属性度量模型2

```
n = 4 ♯ 子属性个数
gamma = [0.25, 0.20, 0.25, 0.30]
x = list(map(float, input("输入子属性值: ".format(n)).split()))
rho=float(input("输入rho的值: "))
♯ rho=0.50
sum=0
for i in range(0, n):
    sum += gamma[i]*pow(x[i], -rho)
result = pow(sum, -1/rho)
print(result)
```

Example

对于一个网络软件，其关键属性包括可靠性y_1和可维护性y_2，非关键属性有可移植性y_3和可测性y_4。可靠性又可以分解为容错性x_1、一致性x_2、简单性x_3和准确性x_4的4个可信子属性。可维护性可分解为一致性x_2、简单性x_3、模块性x_5 和自描述性x_6。可移植性分解为模块性x_5、自描述性x_6、机器无关性x_7 和软件系统无关性x_8。可测性则分解为简单性x_3、模块性x_5、自描述性x_6 和可观测性x_9。

可信属性度量模型使用模型2，即有

$$
\begin{aligned}
y_1 &= (\omega_{11}x_1^{-\rho_{y_1}} + \omega_{12}x_2^{-\rho_{y_1}} + \omega_{13}x_3^{-\rho_{y_1}} + \omega_{14}x_4^{-\rho_{y_1}})^{-\frac{1}{\rho_{y_1}}}\\
y_2 &= (\omega_{22}x_2^{-\rho_{y_2}} + \omega_{23}x_3^{-\rho_{y_2}} + \omega_{25}x_5^{-\rho_{y_2}} + \omega_{26}x_6^{-\rho_{y_2}})^{-\frac{1}{\rho_{y_2}}}\\
y_3 &= (\omega_{35}x_3^{-\rho_{y_3}} + \omega_{36}x_6^{-\rho_{y_3}} + \omega_{37}x_7^{-\rho_{y_3}} + \omega_{38}x_8^{-\rho_{y_3}})^{-\frac{1}{\rho_{y_3}}}\\
y_4 &= (\omega_{43}x_3^{-\rho_{y_4}} + \omega_{45}x_5^{-\rho_{y_4}} + \omega_{46}x_6^{-\rho_{y_4}} + \omega_{49}x_9^{-\rho_{y_4}})^{-\frac{1}{\rho_{y_4}}}
\end{aligned}
$$

Example

假定关键属性与非关键属性的权重分别为$\alpha = 0.7$和$\beta = 0.3$。而关键属性y_1和y_2的权重分别为$\alpha_1 = 0.6, \alpha_2 = 0.4$，非关键属性$y_3$和$y_4$的权重为$\beta_3 = 0.5, \beta_4 = 0.5$。令

$$
\begin{aligned}
(\omega_{11}, \omega_{12}, \omega_{13}, \omega_{14}) &= (0.3, 0.3, 0.2, 0.2) \\
(\omega_{22}, \omega_{23}, \omega_{25}, \omega_{26}) &= (0.4, 0.2, 0.2, 0.2) \\
(\omega_{35}, \omega_{36}, \omega_{37}, \omega_{38}) &= (0.25, 0.25, 0.3, 0.2) \\
(\omega_{43}, \omega_{45}, \omega_{46}, \omega_{49}) &= (0.1, 0.3, 0.3, 0.3) \\
(\rho, \rho_{y_1}, \rho_{y_2}, \rho_{y_3}, \rho_{y_4}) &= (0.4, 0.5, 0.6, 0.5, 0.7)
\end{aligned}
$$

$$T_3=\{\alpha[\min_{1\le i\le m}\{(\frac{y_i}{10})^{\epsilon}\}y_1^{\alpha_1}y_2^{\alpha_2}\cdots y_m^{\alpha_m}]^{-\rho}+\beta[y_{m+1}^{\beta_{m+1}}y_{m+2}^{\beta_{m+2}}\cdots y_{m+s}^{\beta_{m+s}}]^{-\rho}\}^{-\frac{1}{\rho}}$$

模型2：$y=(\sum_{i=1}^{n}\omega_i x_i^{-\rho_y})^{-\frac{1}{\rho_y}}$，$1\le x_i\le 10$。

Table: 关于模型T_3的模拟

编号	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	ϵ	y_1	y_2	y_3	y_4	T_3
1	8	8	9	8	9	8	8	8	8	0.1	8.19	8.41	8.23	8.41	8.13
	8	8	9	8	9	8	8	8	8	0.01	8.19	8.41	8.23	8.41	8.28
2	8	9	9	8	9	8	9	8	9	0.1	8.48	8.82	8.53	8.72	8.52
	8	9	9	8	9	8	9	8	9	0.01	8.48	8.82	8.53	8.72	8.61
3	9	8	9	9	8	8	8	9	8	0.1	8.68	8.22	8.17	8.13	8.27
	9	8	9	9	8	8	8	9	8	0.01	8.68	8.22	8.17	8.13	8.38
4	7	8	9	8	7	8	8	7	8	0.1	7.85	7.99	7.53	7.79	7.70
	7	8	9	8	7	8	8	7	8	0.1	7.85	7.99	7.53	7.79	7.82

作业三

设有4个软件系统，分别编号为1,2,3,4。它们都有可靠性属性y，含有5个子属性，编号为x_1, x_2, x_3, x_4, x_5，其权重分别为$\gamma_1(\omega_1) = 0.15, \gamma_2(\omega_2) = 0.20, \gamma_3(\omega_3) = 0.20, \gamma_4(\omega_4) = 0.25, \gamma_5(\omega_5) = 0.20$。参数$\rho_y = 0.01, 0.55$。按照模型1($y_1$)和模型2($y_2$)分别计算可靠性属性$y$的可信度量值，注意：模型1是不需要参数$\rho_y$的。

Table: 作业三的数据

编号	x_1	x_2	x_3	x_4	x_5	ρ_y	y_1	y_2
1	8.6	9.1	9.2	8.8	8.9	0.01		
	8.6	9.1	9.2	8.8	9.9	0.55		
2	6.8	7.9	5.9	6.6	6.1	0.01		
	6.8	7.9	5.9	6.6	6.1	0.55		
3	9.1	9.9	8.9	8.8	7.8	0.01		
	9.1	9.9	8.9	8.8	7.8	0.55		
4	3.5	4.2	5.6	4.9	5.2	0.01		
	3.5	4.2	5.6	4.9	5.2	0.55		

Weight Vectors

Let us first introduce two concepts .

Definition

A matrix $A=(a_{ij})_{n\times n}$ with $a_{ji}=\frac{1}{a_{ij}}, a_{ij}>0, 1\le i,j\le n$ is called positive reciprocal matrix（正互反判断矩阵）。
If for all $i,j,k=1,\cdots,n$, $a_{ik}=a_{ij}a_{jk}$, then it is said to be consistent（一致性正互反判断矩阵）。

Definition

We call a vector $w=(w_1,\cdots,w_n)^T$ with $\sum_{i=1}^{n} w_i=1, w_i>0,\ (i=1,2,\cdots n)$ weight vector.

The priority method is to derive the weight vector $w=(w_1,\cdots,w_n)^T$ from the positive reciprocal matrix $A=(a_{ij})_{n\times n}$.

Construction of positive reciprocal matrix

- We compare any two attributes y_i and y_j and by aid of 9-point Saaty's scale, assign a value a_{ij} that represents our judgment of the relative importance of y_i and y_j.
- The reciprocal property

$$a_{ji} = \frac{1}{a_{ij}}, \ 1 \leq i,j \leq n$$

  by assumption always holds.
- This way a positive reciprocal matrix $A = (a_{ij})_{n \times n}$ about critical attributes is constructed.

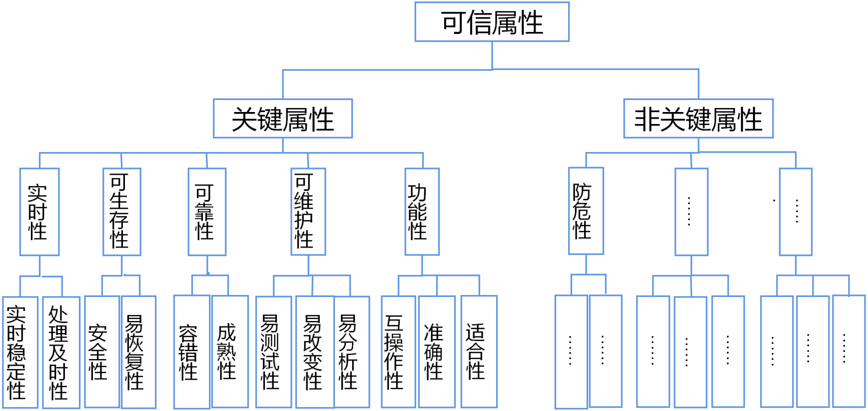
9-point Saaty's scale

9-point Saaty's scale:

Scale	Meaning
1	y_i and y_j are equally(同等) important
3	y_i is weakly more（稍微）important than y_j
5	y_i is strongly more （明显）important than y_j
7	y_i is demonstrably more （强烈）important than y_j
9	y_i is absolutely more （绝对）important than y_j
2,4,6,8	compromising between slightly differing judgement.（相邻判断的中间值）

若$a_{ij} = 3$则$a_{ji} = \frac{1}{3}$。

软件可信属性分解模型-关键属性与非关键属性



下面通过例子来展示如何运用基于合理排序方法求权向量的组合方法求属性权重。

Example

假定关键属性的数目为5，非关键属性的数目分别为2，关于关键属性的正互反矩阵为

$$A_1 = \begin{bmatrix} 1 & 2 & 4 & 2 & 2 \\ 1/2 & 1 & 2 & 1 & 1/2 \\ 1/4 & 1/2 & 1 & 1/2 & 2 \\ 1/2 & 1 & 2 & 1 & 2 \\ 1/2 & 2 & 1/2 & 1/2 & 1 \end{bmatrix}$$

关于非关键属性的正互反矩阵为

$$A_2 = \begin{bmatrix} 1 & 2 \\ 1/2 & 1 \end{bmatrix}$$

Remark

- There is currently no accepted priority method for deriving weight vector from positive reciprocal matrix.
- The reasonable priority methods should possess some basic properties: correctness in the consistent case, comparison order invariance, smoothness preserve ranks strongly, independence of description, harmony.

Remark (Cont.)

- In our combinational algorithm, we choose LLSM and CSM as candidates,
  - since they not only possess the basic properties that reasonable priority methods should have,
  - but also it is easy to derive the weight vector from positive reciprocal matrix.
- Moreover, we also identify EV as one of the candidates, even EV does not satisfy independence of description and harmony,
  - it was shown by various researchers that for small deviations around the consistent ratios $\frac{w_i}{w_j}$, EV method gives reasonably good approximation of the weight vector,
  - and it is very popular, of course it is also easy to solve the weight vector from positive reciprocal matrix.

方法1：特征向量法Eigenvector method (EV)

Saaty proposed the principal eigenvector of A as the desired weight vector w. To find this vector, we just have to solve the following system

$$AX = \lambda_{\max} X$$

where $\lambda_{\max}$ is the principal eigenvalue of matrix A，X是$\lambda_{\max}$对应的特征向量。对X进行归一化处理，就可以得到权值向量w。使用matlab进行求解。

$>> A = [1\ 2; 3\ 4]$

$>> [ev\ dv] = eig(A)$

ev是特征向量矩阵，每一列对应着一特征向量。dv是特征值矩阵，主对角线上的数字是特征值，按照从大到小排列。因而主对角线的对一个数字是最大特征值$\lambda_{\max}$，EV的第一列是$\lambda_{\max}$的对应的特征向量，也是要求的特征向量X。

方法1：特征向量法Eigenvector method (EV)

Example

正互反判断矩阵：

$$A = \begin{bmatrix} 1 & 2 & 4 & 2 & 2 \\ 1/2 & 1 & 2 & 1 & 1/2 \\ 1/4 & 1/2 & 1 & 1/2 & 2 \\ 1/2 & 1 & 2 & 1 & 2 \\ 1/2 & 2 & 1/2 & 1/2 & 1 \end{bmatrix}$$

求出其最大特征值$\lambda_{\max} = 5.4137$，其对应的特征向量$X = (-0.7317, -0.3368, -0.2700, -0.4239, -0.3141)$，归一化处理后得到权重向量$w = (0.3524, 0.1622, 0.1300, 0.2041, 0.1513)$。

方法2：对数最小二乘法Logarithmic least-squares method (LLSM)

LLSM minimizes L^2 distance function defined for unknown $\ln(w_i/w_j)$ and known $\ln a_{ij}$ by solving the following optimization problem:

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n}[\ln a_{ij} - (\ln w_i - \ln w_j)]^2$$

subject to

$$\begin{cases} \sum_{i=1}^{n} w_i = 1, \\ w_i > 0, \ i = 1, 2, \cdots n. \end{cases}$$

Crawford and Williams have shown that the solution for this problem is

$$w_i = \frac{\sqrt[n]{\prod_{j=1}^{n} a_{ij}}}{\sum_{i=1}^{n} \sqrt[n]{\prod_{j=1}^{n} a_{ij}}}, \ i = 1, \cdots, n$$

方法2：对数最小二乘法Logarithmic least-squares method (LLSM)

Example

正互反判断矩阵：

$$A_2 = \begin{bmatrix} 1 & 2 \\ 1/2 & 1 \end{bmatrix}$$

使用权重求解公式得到

$$w_1 = \frac{\sqrt{2}}{\sqrt{2}+\sqrt{0.5}} = \frac{1.414}{1.414+0.707} = 0.667$$

$$w_2 = \frac{\sqrt{0.5}}{\sqrt{2}+\sqrt{0.5}} = \frac{0.707}{1.414+0.707} = 0.333$$

方法2：LLSM–Python代码

```
n = 5 ♯ 5 方阵阶数
a = []
print(”输入方阵: ”)
for i in range(0, n):
    row = list(map(float, input().split()))
    a.append(row)
A = []
for i in range(0, n):
    product = 1
    for j in range(0, n):
        product *= a[i][j]
    A.append(pow(product, 1 / n))
♯ A[i]即为a[i][0]到a[i][n-1]的乘积的n次方根
sum = sum(A)
w = []
for i in range(0, n):
    w.append(A[i] / sum)
print(”result: 0”.format(w))
```

方法2：LLSM-Python

Example

正互反判断矩阵：

$$A = \begin{bmatrix} 1 & 2 & 4 & 2 & 2 \\ 1/2 & 1 & 2 & 1 & 1/2 \\ 1/4 & 1/2 & 1 & 1/2 & 2 \\ 1/2 & 1 & 2 & 1 & 2 \\ 1/2 & 2 & 1/2 & 1/2 & 1 \end{bmatrix}$$

```
ds/LLSM.py
输入方阵:
1 2 4 2 2
0.5 1 2 1 0.5
0.25 0.5 1 0.5 2
0.5 1 2 1 2
0.5 2 0.5 0.5 1
result: [0.36785931114848636, 0.1601200652670195, 0.12134831777797382, 0.21127969279330439, 0.13939261301321598]
```

方法3：卡方最小二乘法Chi-square method (CSM)

The CSM also uses L^2 metric in defining objective function of the following optimization problem:

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\frac{a_{ij} - \frac{w_i}{w_j}}{\frac{w_i}{w_j}} \right)^2$$

subject to

$$\begin{cases} \sum_{i=1}^{n} w_i = 1, \\ w_i > 0, \;\; i = 1, 2, \cdots n. \end{cases}$$

Wang and Fu(王应明，傅国伟，判断矩阵的χ^2方法，管理工程学报，1994年第8卷第1期：26-32) developed a convergent iteration algorithm which is simple enough and easy to implement.

方法3：CSM-Python

```
n = 5 ♯ 方阵阶数
a = []
print(”输入方阵: ”)
for i in range(0, n):
    row = list(map(float, input().split()))      a.append(row)
♯ 步骤(1)
epsilon = 1e-10 ♯ 迭代精度
w = [1 / n for i in range(0, n)] ♯ 初始解
while True:
    ♯ 步骤(2)
    m = None
    max-val = None
    for i in range(0, n):
        val = 0
        for j in range(0, n):
            val += (1 + a[j][i] * a[j][i]) * (w[i] / w[j]) - (1 + a[i][j] * a[i][j]) *
(w[j] / w[i])
```

方法3：CSM-Python

```
        val = abs(val)
        if max_val == None or val ¿ max-val:
            max_val = val
            m = i
if max-val ¡= epsilon:
    break
♯ 步骤(3)
up = 0
bottom = 0
for j in range(0, n):
    if j != m:
        up += (1 + a[m][j] * a[m][j]) * (w[j] / w[m])
        bottom += (1 + a[j][m] * a[j][m]) * (w[m] / w[j])
T = pow(up / bottom, 1 / 2)
```

方法3：CSM-Python

```
X = w
X[m] *= T
sum-X = sum(X)
for i in range(0, n):
    w[i] = X[i] / sum-X
print("最终解: 0".format(w))
```

```
PS C:\Users\YXChen> & C:/Users/YXChen/AppData/Local/Programs/Python/Python312/python.exe c:/Users/YXChen/Downloads/CSM.py
输入方阵:
1 2 4 2 2
0.5 1 2 1 0.5
0.25 0.5 1 0.5 2
0.5 1 2 1 2
0.5 2 0.5 0.5 1
最终解: [0.3766703113826757, 0.15459360515184867, 0.11859209445452762, 0.21219212515053232, 0.13795186386041577]
```

三种计算方法总结：EV，LLSM，CSM

Example

正互反判断矩阵：

$$A = \begin{bmatrix} 1 & 2 & 4 & 2 & 2 \\ 1/2 & 1 & 2 & 1 & 1/2 \\ 1/4 & 1/2 & 1 & 1/2 & 2 \\ 1/2 & 1 & 2 & 1 & 2 \\ 1/2 & 2 & 1/2 & 1/2 & 1 \end{bmatrix}$$

方法	权重向量
EV	$(0.3524, 0.1622, 0.1300, 0.2041, 0.1513)$
LLSM	$(0.3679, 0.1601, 0.1213, 0.2113, 0.1394)$
CSM	$(0.3767, 0.1546, 0.1186, 0.2121, 0.1379)$

三种方法求出来的权重向量不同，如何选择？

Evaluating Criteria for Assessing Quality of Weight vectors Estimates

- For any given positive reciprocal matrix, different priority methods would give different weight vector.
- In order to assess quality of weight vectors estimates, people proposed various evaluating criteria in "strengths（程度）" and "directions（方向）".
- An estimating method is deemed to be better if the weight vector $w = (w_1, \cdots, w_n)$ obtained from $A = (a_{ij})_{n\times n}$ by this method is closer to satisfying

$$\frac{w_i}{w_j} = a_{ij},\ i,j = 1,\cdots,n$$

and

$$\frac{w_i}{w_j} \leq 1 \ if\ and\ only\ if\ a_{ij} \leq 1,\ i,j = 1,\cdots,n$$

Evaluating Criteria in “程度”

There are many evaluating criteria in “程度” which measures the accuracy of the weight vector, such as

$$\sum_{i=1}^{n}[\sum_{j=1}^{n}(a_{ij}-\frac{w_i}{w_j})^2]^{1/2}$$

$$[\sum_{i=1}^{n}\sum_{j=1}^{n}(a_{ij}-\frac{w_i}{w_j})^2]^{1/2}.$$

$$\sum_{i=1}^{n}\sum_{j=1}^{n}|a_{ij}-\frac{w_i}{w_j}|$$

Evaluating Criteria in ”Direction”

There are also several evaluating criteria in “方向” which measures the ranking order property, such as Minimum Violation (MV), The MV criterion is expressed as

$$MV = \sum_{i=1}^{n} \sum_{j=1}^{n} I_{ij}$$

where

$$I_{ij} = \begin{cases} 1 & if \quad w_i > w_j \quad and \quad a_{ij} > 1 \\ 0.5 & if \quad w_i = w_j \quad and \quad a_{ij} \neq 1 \\ 0.5 & if \quad w_i \neq w_j \quad and \quad a_{ij} = 1 \\ 0 & \qquad\qquad\qquad\qquad otherwise \end{cases}$$

计算KD和I的代码

```
n = 5 ♯ 方阵阶数
a = []
print(”输入方阵a: ”)
for i in range(0, n):
    row = list(map(float, input().split()))
    a.append(row)
w = list(map(float, input(”输入向量w: ”).split()))
TD = 0
for i in range(0, n):
    for j in range(0, n):
        TD += abs(a[i][j] - w[i] / w[j])
epsilon = 1e-10 ♯ 浮点数相等的判断精度
♯ 判断x大于y
def greater_than(x, y):
    return x > = y + epsilon
```

计算KD和I的代码

```
♯ 判断x等于y
def equal-to(x, y):
    return abs(x - y) ¡ epsilon
I = [[0 for j in range(0, n)] for i in range(0, n)] ♯ 初始化矩阵I全为0
MV = 0
for i in range(0, n):
    for j in range(0, n):
        if greater_than(w[i], w[j]) and greater_than(a[i][j], 1):
            I[i][j] = 1
        elif equal_to(w[i], w[j]) and not equal_to(a[i][j], 1):
            I[i][j] = 0.5
        elif not equal_to(w[i], w[j]) and equal_to(a[i][j], 1):
            I[i][j] = 0.5
        else:
            0
        MV += I[i][j]
```

计算KD和I的代码

```
print("\nTD: 0".format(TD))
print("矩阵I: ")
for i in range(0, n):
    for j in range(0, n):
        print("{:> 3}".format(I[i][j]), end=" ")
    print()
print("MV: 0".format(MV))
```

Combinational algorithm based on priority methods

Here we give a polynomial-time combinational algorithm for estimating the weight vectors based on the priority methods.

Step1 Estimate weight vectors from positive reciprocal matrix with LLSM, CSM, EM, and suppose the weight vectors be $w^{(1)} = (w_1^{(1)}, \cdots, w_n^{(1)}), w^{(2)} = (w_1^{(2)}, \cdots, w_n^{(2)})$ and $w^{(3)} = (w_1^{(3)}, \cdots, w_n^{(3)})$ respectively.

Combinational algorithm based on priority methods

Step2 Evaluate the result under evaluating criterion in strengths:

$$TD^{(k)} = \sum_{i=1}^{n}\sum_{j=1}^{n} |a_{ij} - \frac{w_i^{(k)}}{w_j^{(k)}}|, \ k = 1, 2, 3$$

Suppose the rankings of the results $TD^{(1)}$, $TD^{(2)}$, $TD^{(3)}$
最小TD对应的权重向量为最优的权重向量。

三种计算方法总结：EV，LLSM，CSM

Example

正互反判断矩阵：

$$A = \begin{bmatrix} 1 & 2 & 4 & 2 & 2 \\ 1/2 & 1 & 2 & 1 & 1/2 \\ 1/4 & 1/2 & 1 & 1/2 & 2 \\ 1/2 & 1 & 2 & 1 & 2 \\ 1/2 & 2 & 1/2 & 1/2 & 1 \end{bmatrix}$$

方法	权重向量
EV	$(0.3524, 0.1622, 0.1300, 0.2041, 0.1513)$
$LLSM$	$(0.3679, 0.1601, 0.1213, 0.2113, 0.1394)$
CSM	$(0.3767, 0.1546, 0.1186, 0.2121, 0.1379)$

三种方法求出来的权重向量不同，如何选择？计算这三个权重向量的TD值！$TD^{EV} = 8.793, TD^{LLSM} = 8.536, TD^{CSM} = 8.589$。$TD^{LLSM}$最小，选LLSM计算的权重向量为可信属性的权重向量。

作业四

设C919飞行控制软件有5个可信属性：实时性、可靠性、可生存性、可维护性、功能性（其含义见第4讲第3.2节），其正互反判断矩阵A为

$$A = \begin{bmatrix} 1 & 1/2 & 3 & 2 & 1/2 \\ 2 & 1 & 2 & 3 & 2 \\ 1/3 & 1/2 & 1 & 2 & 1/3 \\ 1/2 & 1/3 & 1/2 & 1 & 2 \\ 2 & 1/2 & 3 & 1/2 & 1 \end{bmatrix}$$

分别使用右特征向量法EV、对数最小二乘法LLSM、卡方最小二乘法CSM求出权重向量W^{EV}、W^{LLSM}、W^{CSM}，在此基础上使用“强度”方法求出最优的权重向量。

Thanks!