APLICAÇÃO DA TEORIA *FUZZY* EM REQUISITOS DE QUALIDADE DE SOFTWARE

Arnaldo Dias Belchior (belchior@cos.ufrj.br)
Geraldo B. Xexéo(xexeo@cos.ufrj.br)
Ana Regina Cavalcanti da Rocha (darocha@cos.ufrj.br)
Programa de Engenharia de Sistemas e Computação - COPPE/UFRJ
Caixa Postal 68511 / CEP 21945-970 - Rio de Janeiro-RJ - Brasil

RESUMO

Este trabalho fornece alguns conceitos sobre a teoria fuzzy em geral, utilizando-se deste enfoque para propor um modelo para a medição de requisitos de qualidade de software. Isto porque muitos desses requisitos possuem conceitos subjetivos, que é uma característica peculiar da teoria fuzzy. O modelo fuzzy proposto procura tratar desta subjetividade de uma forma mais objetiva, tendo sido realizados vários experimentos, e cujos resultados ajudaram a refiná-lo.

1. INTRODUÇÃO

A teoria *fuzzy* é utilizada para representar modelos de raciocínio impreciso, que possuem um papel essencial na notável habilidade humana, para tomar decisões racionais em ambientes de incertezas e imprecisões [ZADE 88]. Com o advento dessa teoria, obteve-se uma ferramenta robusta para se representar várias facetas do conhecimento humano [YAGE 91].

Tradicionalmente, na Teoria dos conjuntos, um elemento *pertence* ou *não pertence* a um conjunto. Os conjuntos *fuzzy*, uma generalização de um conjunto ordinário, permitem a definição de um *grau de dependência* para cada elemento, isto é, um número real no intervalo [0,1]. Neste caso, se o grau é *zero*, o elemento não pertence ao conjunto e, se é 1, o elemento *pertence* totalmente ao conjunto [TURK 91].

Portanto, um elemento pode pertencer parcialmente a um conjunto *fuzzy*. Podemos citar, por exemplo, o conjunto de pessoas "*jovens*". Quando uma pessoa não é mais jovem? A definição de um conjunto *fuzzy* pode-nos mostrar, por exemplo, uma pessoa de 20 anos como "90 % *jovem*", enquanto que alguém de 60 anos seria, apenas "30 % *jovem*".

2. TEORIA *FUZZY* E QUALIDADE DE SOFTWARE

Qualquer representação adequada de um conjunto *fuzzy* envolve o entendimento básico de cinco diferentes símbolos conceituais, relacionados entre si [TURK 91]:

- *Conjunto de elementos* $\theta \in \Theta$, como, por exemplo, um "*homem*" em "*homens*" ou um "*item*" em "*estoque*".
- *Variável lingüística V* em um conjunto de variáveis lingüísticas, que é um rótulo para um atributo dos elementos $\theta \in \Theta$, como "*altura de homem*" ou o "*nível de estoque*" de uma empresa.
- *Termo lingüístico A* de uma *variável lingüística*, correspondendo a um adjetivo ou a um advérbio, em um conjunto de termos lingüísticos, como "*homem alto*" associado com a "*altura do homem*" ou "*estoque baixo*", relacionado com possíveis "*níveis de estoque*" de uma empresa.
- *Intervalo numérico mensurável* $X \in [-\infty, \infty]$, conhecido como o *conjunto referencial* para um atributo particular *V*, de um conjunto de elementos $\theta \in \Theta$, como, por exemplo, "[0, 3] *metros*" para "*altura de homem*", ou "[250, 750] *unidades*" para "*nível de estoque*".
- *Atribuição numérica subjetiva* $\mu_A(\theta)$, ou *valor de pertinência*, que é o grau com que um elemento θ pertence ao conjunto de elementos, rotulados por uma *variável lingüística V*, e identificados pelo *termo lingüístico A*. Por exemplo, o valor de pertinência dado a um "*homem*" em um grupo de homens por um observador, que usa o *termo lingüístico* "*alto*", segundo sua

visão de "*altura*" para homens, ou o valor de pertinência atribuído por um gerente para "*estoque*", através do adjetivo "*baixo*", englobando todos os níveis de estoque sob o seu gerenciamento.

A teoria *fuzzy* é usada basicamente para mapear modelos qualitativos de tomada de decisões, e para métodos de representação imprecisa. Neste contexto, é que se pode utilizar a teoria *fuzzy* em requisitos (atributos) de qualidade de software, uma vez que são, em sua maioria, conceitos subjetivos e de avaliação não trivial.

Não basta, apenas, identificar que atributos determinam a qualidade do software, mas também que procedimentos adotar, para controlar seu processo de desenvolvimento, de forma a atingir o nível de qualidade desejado. Esse processo é realizado através da aplicação de métricas de qualidade, que são medidas ou avaliações das características de qualidade do produto. O uso de medidas, de uma forma organizada e projetada, possui efeito benéfico, tornando os desenvolvedores mais conscientizados da relevância do gerenciamento e dos compromissos para com a qualidade [BELC 92].

Um dado atributo de qualidade pode ter um escopo amplo e ser composto por outros atributos de qualidade. Quando há esta composição, nomearemos esse atributo de *agregado*. Aquele que não se compõe de outros atributos, chamaremos de *atributo primitivo*, sendo passível de medição, segundo o Modelo Rocha Extendido para qualidade de software [XEXE 95]. Esse modelo propõe a utilização de funções *fuzzy*, para interpretar medidas e definir processos de agregação de atributos.

O *conjunto referencial* para a medição dos *atributos primitivos* utilizará uma *escala de pesos* no intervalo [0 a 5], conforme a *tabela 2.1*, no desenvolvimento do modelo *fuzzy* para qualidade de atributos de software, proposto em [BELC 95].

ESCALA DE PESOS		
Peso	**Simbologia**	**Termo Lingüístico**
5,0	E	Extremamente Alto
4,0	A	Alto
3,0	ME	Médio
2,0	B	Baixo
1,0	MB	Muito Baixo
0,0	NE	Nenhum ou Ausente

Tabela 2.1: Escala de valores para medidas de qualidade de software [BELC 92]

3. MODELO *FUZZY* PARA ATRIBUTOS DE QUALIDADE DE SOFTWARE

Um dado software possui um conjunto de atributos de qualidade (*variáveis lingüísticas*). Cada atributo pode ser relacionado a um conjunto de *termos lingüísticos*, dentro de um *conjunto referencial* (vide *tabela 2.1*), e podem ser associados a uma *função de pertinência*.

Tomemos os *termos lingüísticos* A_0, A_1, A_2, ..., A_n com seus valores correspondentes no eixo das abcissas x_0, x_1, x_2, ..., x_n , respectivamente, dentro de um *conjunto referencial* [0, *n*]. No eixo das ordenadas tomemos o intervalo [0, 1], caracterizando um conjunto *fuzzy*, que medirá o grau de pertinência de um elemento x_i. Chegaremos à função linear [*eq. 3.0*], que servirá para gerar *funções de pertinência* de *termos lingüísticos* de qualidade de software, onde $i = 1, 2, 3, ..., n$, onde *n* corresponde ao número desses *termos lingüísticos* considerados. Essa funções estão baseadas na representação de *funções de pertinência* desenvolvidas em [VIOT 93].

$$f_A(x) = [1 / (x_i - x_{i-1})]\, x - [1 / (x_i - x_{i-1})]\, x_{i-1}\,; \quad \text{para } x_{i-1} \leq x \leq x_i \qquad [eq.\ 3.0]$$

$$[1 / (x_{i+1} - x_i)]\, x_{i+1} - [1 / (x_{i+1} - x_i)]\, x\,; \quad \text{para } x_i \leq x \leq x_{i+1}$$

Pela *tabela 2.1* e [*eq 3.0*], geramos as *funções de pertinência* para os *termos lingüísticos* de qualidade de software considerados: f_{NE}, f_{MB}, f_B, f_{ME}, f_A e f_E, apresentadas na *figura 3.1*.

deslocamento à direita de μ, determinando assim um intervalo, no qual o atributo é considerado de qualidade. Quando σ_1 é subtraído de μ ou σ_2 é adicionado a μ, o grau de pertinência é zero, como mostram [*eq. 3.2*] e [*eq. 3.4*]. Embora σ_1 possa ser diferente de σ_2, assumiremos, a partir deste ponto, uma simplificação do modelo, particularizando-o para $\sigma = \sigma_1 = \sigma_2$. Geraremos, então, os seguintes valores componentes:

- *c'*: valor à esquerda de *c*, onde $c' = \mu - \sigma$

  $f_C(\mu - \sigma) = 0$, onde $(\mu - \sigma) \geq 0$ [*eq. 3.2*]

- *c*: valor intermediário, que fica entre c' e c", onde $c = \mu$

  $f_C(\mu) = 1$; [*eq. 3.3*]

- *c"*: valor à direita de *c*, onde $c'' = \mu + \sigma$

  $f_C(\mu + \sigma) = 0$, onde $(\mu + \sigma) \leq n$ [*eq. 3.4*]

Apropriando-nos da simplificação feita acima ($\sigma = \sigma_1 = \sigma_2$), pode-se, por exemplo, utilizar os valores estatísticos, *média aritmética* para μ, e *desvio padrão* para σ, apurados para um conjunto de atributos de qualidade de software, em um determinado domínio de aplicação. Considerando-se os dados obtidos provenientes de uma população normal, σ pode ser igual ao valor de desvio padrão, representando cerca de 68% das observações feitas, ou σ pode ser igual ao dobro do desvio padrão, abrangendo cerca de 95% das observações feitas.

Dos pontos P_1 (*c'*, 0) e P_2 (*c*, 1), teremos a primeira equação linear do atributo primitivo *C*, com coeficiente angular, $a = (1 / \sigma)$. Dos pontos P_2 (*c*, 1) e P_3 (*c"*, 0), teremos a segunda equação linear desse atributo com coeficiente angular, $a = -(1 / \sigma)$. Portanto, a função de pertinência, f_C, para o atributo primitivo *C*, é:

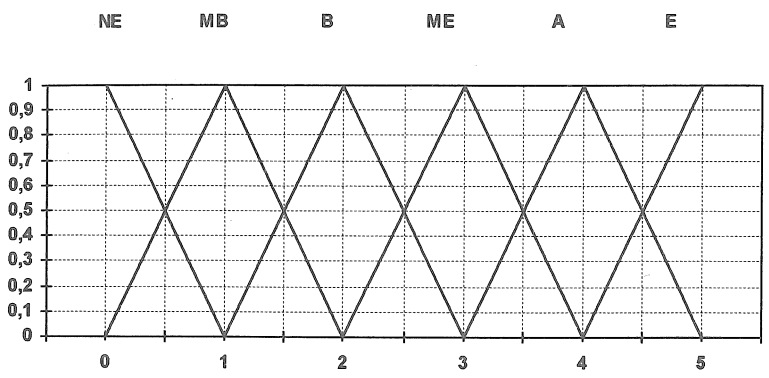

Figura 3.1: Funções de pertinência para atributos de qualidade de software [BELC 95]

Nas funções lineares representadas na *figura 3.1*, de formato $f(x) = ax + b$, o coeficiente angular a possui valor 1 ou -1, dependendo do intervalo considerado para o cálculo. Assim sendo, podemos deduzir as funções de pertinência, para os termos lingüísticos da *tabela 2.1*, como, por exemplo, o *termo lingüístico* Médio (ME):

$$f_{ME}(x) = x - 2; \quad \text{para } 2 \leq x \leq 3 \qquad [eq.\ 3.1]$$

$$4 - x; \quad \text{para } 3 \leq x \leq 4$$

Assim, podemos obter todas as equações [EQ], para cada termo lingüístico considerado na *figura 3.1*. Com base nessas equações, poderemos conceber *funções de pertinência* para atributos de qualidade primitivos C, no *intervalo numérico mensurável* X [0, 5], como veremos a seguir.

3.1. Proposta *Fuzzy* para Atributos de Qualidade de Software

Para a geração das *funções de pertinência* para cada atributo primitivo C, utilizaremos os seguintes conceitos: o valor μ e a amplitude σ. O valor μ representa a qualidade desejável para o atributo considerado, correspondendo ao valor de pico de sua função de pertinência, isto é, para o grau de pertinência igual a 1. A amplitude σ_1 é o deslocamento à esquerda de μ, e σ_2 é o

$f_C(x) = (1/\sigma)\,x - (1/\sigma)\,c'$; para $c' \leq x \leq c$ [*eq. 3.5*]

$(1/\sigma)\,c'' - (1/\sigma)\,x$; para $c \leq x \leq c''$

Aplicando-se individualmente cada valor componente c', c, e c'' nas equações *EQ*, conforme for o caso, geraremos dois *valores de pertinência* para cada um desses componentes, isto é, as *atribuições numéricas subjetivas* $\mu_{A1}(\theta)$ e $\mu_{A2}(\theta)$.

Seja, por exemplo, um produto de software θ (*contabilidade*), inserido em um determinado domínio de aplicação Θ (*software financeiro*), ao ser avaliado para um certo atributo de qualidade, ou variável lingüística *V* (*documentação*), caracterizado por um conjunto de termos lingüísticos *A*. Para um certo observador (usuário ou desenvolvedor, por exemplo), a *documentação* (*V*) de um software contábil (θ), seria avaliada como tendo 70% ($\mu_A(\theta)$) *médio* (*A*) e 30% ($\mu_A(\theta)$) *alto* (*A*).

Nos intervalos de valores, das equações *EQ*, verificamos que quase todos seus valores limites pertencem a mais de uma equação. Quando algum dos valores componentes (c', c, c'') incidirem no limite do intervalo, a equação mais apropriada a ser usada, deverá ser aquela que esteja mais próxima do *valor componente c* ou que o contenha.

Quando o *valor componente c*, incidir no limite do intervalo, a equação a ser usada deverá ser a mais à direita de *c*. Isto se justifica, porque para se alcançar um maior nível de qualidade, buscamos produtos ou processos cada vez melhores.

Uma vez conhecidas as *funções de pertinência* de cada atributo primitivo C, poderemos fazer uma composição das medidas dessas funções [TURK 91], gerando-se assim as *funções de pertinência* dos *atributos agregados* de software.

3.2. Funções de Pertinência para Atributos Agregados

A agregação dos atributos primitivos de qualidade será obtida, considerando-se o *valor agregado* μ_a e também a *amplitude agregada* σ_a, que corresponde ao valor que dista de μ_a tanto à direita, σ_{a1}, quanto à esquerda, σ_{a2}. Consideraremos aqui, também, a mesma simplificação feita para atributos primitivos $\sigma_a = \sigma_{a1} = \sigma_{a2}$. Vejamos, então, estes conceitos:

- o *valor agregado* μ_a, compõe-se dos valores dos atributos primitivos *C*, segundo as seguintes equações, onde *h* é o peso de C, isto é, o grau de contribuição do atributo primitivo c_i:

$$t_c = c_1 + c_2 + \ldots + c_n \qquad [eq.\ 3.6]$$

$$h_{ci} = c_i / t_c \text{, onde i = 1, 2, ... , n} \qquad [eq.\ 3.7]$$

$$\mu_a = \sum_{i=1}^{n} (c_i \,.\, h_{ci}) \text{ ou simplesmente,} \qquad [eq.\ 3.8]$$

$$\mu_a = (\sum_{i=1}^{n} c_i^2) / (\sum_{i=1}^{n} c_i) \qquad [eq.\ 3.9]$$

- a *amplitude agregada* σ_a dos atributos primitivos *C*, compõem o atributo agregado resultante *R*, através das seguintes equações, obtidas experimentalmente:

$$t_\sigma = [(1 / \sigma_1) + (1 / \sigma_2) + \ldots + (1 / \sigma_n)] \qquad [eq.\ 3.10]$$

$$w_{ci} = \sigma_i / t_\sigma \text{, onde i = 1, 2, ... , n} \qquad [eq.\ 3.11]$$

$$\sigma_a = \sum_{i=1}^{n} (c_i \,.w_{ci}) \text{ ou, simplesmente,} \qquad [eq.\ 3.12]$$

$$\sigma_a = n / [\sum_{i=1}^{n} (1 / \sigma_i)], \text{ onde} \qquad [eq.\ 3.13]$$

A variável t_σ é o somatório dos inversos das *amplitudes*, que constituem o atributo agregado, uma vez que quanto menor é a *amplitude*, maior o grau de certeza do atributo de qualidade

considerado. Se algum atributo possuir amplitude igual a zero, podemos considerá-la como sendo 5%, correspondendo a uma margem de erro estatístico aceitável, um vez que estes dados geralmente provêm de pesquisas de campo. Se para a composição do atributo agregado, todos os atributos primitivos tiverem amplitude nula, a amplitude agregada também será considerada nula.

A variável w_c é o peso da amplitude do atributo primitivo C, isto, é o grau de contribuição da amplitude de c para o atributo agregado R considerado. Assim sendo, para a geração de funções de pertinência lineares para o *atributo agregado resultante*, R, teremos os seguintes componentes, onde $0 \leq \mu_a \leq n$, de acordo com o *conjunto referencial adotado*:

- r': faremos $f_R(\mu_a - \sigma_a) = 0$, onde $(\mu_a - \sigma_a) \geq 0$ [*eq. 3.14*]
- r: faremos $f_R(\mu_a) = 1$ [*eq. 3.15*]
- r'': faremos $f_R(\mu_a + \sigma_a) = 0$, onde $(\mu_a + \sigma_a) \leq n$ [*eq. 3.16*]

Utilizando-se, então, os pontos P_1 (r', 0) e P_2 (r, 1) teremos a primeira equação linear do atributo agregado R, com coeficiente angular, $a = (1 / \sigma_a)$. Dos pontos P_2 (r, 1) e P_3 (r'', 2) teremos a segunda equação linear do mesmo atributo com o coeficiente angular, $a = -(1 / \sigma_a)$. Desta forma, a função de pertinência, f_R, para o atributo primitivo, R será:

$$f_R(x) = (1 / \sigma_a)\, x - (1 / \sigma_a)\, r'; \quad \text{para } r' \leq x \leq r \quad [eq.\ 3.17]$$

$$(1 / \sigma_a)\, r'' - (1 / \sigma_a)\, x; \quad \text{para } r \leq x \leq r''$$

Com os valores r', r, e r'' e aplicando-os às equações *EQ*, conforme for o caso, geraremos seus *valores de pertinência*. Uma vez mais, como podemos observar na *figura 3.1*, cada valor componente r', r e r'' possuem mais de uma função. Quando $f(r') = 0$ ou $f(r'') = 0$, isto é, os

valores componentes incidem exatamente nos limites do intervalo dos termos lingüísticos, deverá ser utilizada suas equações correspondentes, mais próximas do valor componente central *r*.

Quando o valor componente central, *r*, incidir no limite do intervalo, a equação a ser utilizada deverá ser a mais à direita de *r*, por significar uma maior excelência nos atributos de *qualidade* de software.

Quando a composição de dois ou mais *atributos agregados R*, constituírem um novo atributo agregado *R'*, procede-se, também, como se os atributos agregados *R*, que o originaram, fossem atributos primitivos.

Quando já se conhece o padrão de qualidade desejado de um determinado domínio de aplicação Θ, pode-se, baseando-se nesse padrão, avaliar a qualidade de outros produtos de software, dentro do mesmo domínio de aplicação, como veremos a seguir.

3.3. Aplicação do Modelo *Fuzzy* em novos Produtos de Software

Neste caso, o cálculo das *funções de pertinência* para seus atributos primitivos será o mesmo desenvolvido na *seção* anterior. No entanto, na composição dos atributos agregados, adotaremos um novo enfoque, para evitar as grandes variações de seus atributos primitivos, isto é, aqueles que possuem valores muito diferentes (para mais ou para menos) daqueles experimentalmente obtidos e estabelecidos como padrão de qualidade. Estas discrepâncias poderiam influenciar, indevidamente, no padrão de qualidade do produto de software, que está sendo aferido. Neste caso, redefiniremos os cálculos para o valor agregado μ_a e a amplitude σ_a, como se segue:

- Cálculo do novo *valor agregada* μ_a :

- tomando-se *H* como sendo o peso do atributo primitivo *C*, considerado como padrão de qualidade, para um determinado domínio de aplicação, e com base na [*eq. 3.8*], teremos:

$$\mu_a = \sum_{i=1}^{n} [c_i \, . \, H_i] \qquad [eq.\ 3.18]$$

- Cálculo da nova *amplitude* σ_a :

- tomando-se *W* como sendo o peso da amplitude do atributo primitivo *C*, considerado como padrão de qualidade, e com base na [*eq. 3.12*]:

$$\sigma_a = \sum_{i=1}^{n} [c_i \, . \, W_i] \qquad [eq.\ 3.19]$$

A partir destes cálculos, seguiremos os mesmo procedimentos, já explanados anteriormente, no Modelo *Fuzzy* para atributos de qualidade de software proposto.

4. CONCLUSÃO

Neste trabalho, propõe-se a utilização de um modelo *fuzzy* para atributos de qualidade de software. Todos os experimentos realizados, para a validação e refinamento desse modelo proposto, encontram-se em [BELC 95].

Inicialmente, trata-se dos atributos (primitivos) de qualidade de software, enquadrando-os em uma escala de *termos lingüísticos*. A seguir, faz-se um proposta para a agregação desses atributos. O processo de agregação desses atributos permite-nos chegar a uma medição do software, isto é, à sua avaliação final, desde que se possa deduzir uma equação linear para seus requisitos de qualidade, conforme propõe o Modelo *Fuzzy* desenvolvido.

Propõe-se, também, dentro do mesmo modelo, que quando já se conhece o padrão de qualidade de um determinado domínio de aplicação, baseando-se nesse padrão, pode-se avaliar a qualidade de outros produtos de software, dentro desse mesmo domínio de aplicação.

BIBLIOGRAFIA

[BELC 92] - Belchior, A. D.; *Controle da Qualidade de Software Financeiro*; Tese de Mestrado, COPPE/UFRJ; Rio de Janeiro, julho 1992.

[BELC 95] - Belchior, A.D.; *Um Modelo Fuzzy para Qualidade de Software*; Relatório Técnico ES-344/95, COPPE/UFRJ; Rio de Janeiro, junho de 1995.

[TURK 91] - Turksen, I. B.; *Measurement of membership functions and their acquisition*; Fuzzy Sets and Systems, IFSA; Special Memorial Volume: 25 years of fuzzy sets; North-Holland - Amsterdam; 1991.

[VIOT 93] - Viot, G.; *Fuzzy Logic in C*; Dr. Dobb's Journal; February 1993.

[XEXE 95] - Xexeo, G. B. et alli; *Uma Proposta Fuzzy para Medição de Atributos de Qualidade de Software*; Workshop de Qualidade de Software; IX Simpósio Brasileiro de Engenharia de Software; Recife, 1995.

[YAGE 91] - Yager, R. R.; *Connectives and quantifiers in fuzzy sets*; Fuzzy Sets and Systems, IFSA; Special Memorial Vol.: 25 years of fuzzy sets; North-Holland - Amsterdam; 1991.

[ZADE 88] - Zadeh, L. A.; *Fuzzy logic*, IEEE Transaction Comput. 35; 1988; in [TURK 91].