

Tópicos especiais em Engenharia de Software

Aula 03

Notas de aula Profa. Simone R. S. Souza e Profa. Ellen F. Barbosa.

Aula anterior:

- Técnica Estrutural:
 - Conhecida como teste caixa-branca (em oposição ao teste caixa-preta).
 - Baseia-se no conhecimento da estrutura interna (implementação) do programa.
 - Teste dos detalhes procedimentais.
 - A maioria dos critérios dessa técnica utiliza uma representação de programa conhecida como grafo de programa.

Técnica Estrutural

- Baseados em Fluxo de Controle
 - Todos-Nós
 - Todas-Arestas
 - Todos-Caminhos: Simples, Completo, Livre de Laço,
 - ...
- Baseados em Complexidade
 - Critério de McCabe (teste do caminho base).
- Baseados em Fluxo de Dados
 - Critérios de Rapps & Weyuker
 - Todas-Defs, Todos-Usos, Todos-P-Usos e outros.
- Critérios Potenciais-Usos (Maldonado)
 - Todos-Potenciais-Usos, Todos-Potenciais-Usos/DU e outros.



Critérios Baseados em Fluxo de Dados

- Critérios pertencentes à técnica estrutural.
- Complementares aos critérios baseados em fluxo de controle.
- Objetivo: testar o uso das variáveis em um programa, ou seja, como os dados são usados nas computações.
- Utilizam informações do fluxo de dados do programa para determinar os requisitos de teste.
- Exploram as interações que envolvem definições de variáveis e referências a tais definições.

Critérios Baseados em Fluxo de Dados

Exemplo de erro em Fluxo de Dados:

- Referenciar uma variável sem esta ter sido inicializada.

```
1 main () {  
2   int x;  
3   if (x==42){...}  
4 }
```

- Assumir que o compilador inicializa a variável com algum valor padrão quando ele não o faz. Qual a saída do programa abaixo?

```
1 #include <stdio.h>  
2 main () {  
3   int x;  
4   printf ("%d", x);  
5 }
```



Critérios Baseados em Fluxo de Dados

- O teste baseado em fluxo de dados constitui uma ferramenta poderosa para identificar o uso incorreto de valores resultante de erros de codificação.
- Tornou-se popular com a publicação do trabalho de Rapps e Weyuker (1982):
- “It is our belief that, just as one would not feel confident about a program without executing every statement in it as part of some test, one should not feel confident about a program without having seen the effect of using the value produced by each and every computation.”



Anomalias de Fluxo de Dados

- Uso de variável não inicializada.
- Atribuição de valor a uma variável mais de uma vez sem que tenha havido uma referência a essa variável entre essas atribuições.
- Liberação ou reinicialização de uma variável antes que ela tenha sido criada ou inicializada.
- Liberação ou reinicialização de uma variável antes que ela tenha sido usada.
- Atribuir novo valor a um ponteiro sem que variável tenha sido liberada.

Ciclo de vida das variáveis

- Variáveis são criadas, usadas e destruídas.
- Em algumas linguagens de programação (BASIC e FORTRAN, por exemplo) a criação e destruição são automáticas.
- Em outras (C, C++ e Java, por exemplo), a criação deve ser explícita.
- Exemplos de declaração:
1 `int x;` // *x é criada como um inteiro*
2 `String y;` // *y é criada como uma String*

Ciclo de vida das variáveis

- Declarações, em geral, ocorrem dentro de um bloco.
- Variáveis são criadas quando a definição das mesmas são executadas.
- Variáveis são destruídas no final do bloco (conceito de escopo da variável).

```
1  { // início do bloco 1
2      int x; // x é definida no bloco 1
3      ...; // x pode ser usada aqui
4      { // início do bloco 2
5          int y; // y é definida no bloco 2
6          ...; // x e y podem ser usadas aqui
7      } // y é automaticamente destruída no final do bloco 2
8      ...; // x ainda pode ser usada daqui em diante
9  } // x é automaticamente destruída no final do bloco 1
```

Utilização de variáveis

- No ciclo de vida de uma variável podemos ainda distinguir dois tipos de utilização de uma variável
- Quando o valor da variável é modificado. Por exemplo, via comando de atribuição: $a = 10$ e $b = 5$.
- Quando o valor da variável é usado mas seu valor não é modificado Por exemplo: $a = b * 1$.



Critérios de Rapps e Weyuker

- Objetivo: exercitar caminhos ligando definições globais a usos globais de variáveis do programa.
- Critérios:
 - Todas-Definições.
 - Todos-P-Usos.
 - Todos-P-Usos/Alguns-C-Usos.
 - Todos-C-Usos/Alguns-P-Usos.
 - Todos-Usos.
 - Todos-DU-Caminhos.

Grafo Def-Uso

- Utilizam o Grafo Def-Uso (Def-Use Graph) para derivar os requisitos de teste.
- Informações a respeito do fluxo de dados do programa.
- Extensão do GFC.
 - Grafo Def-Uso
 - GFC + Definição e Uso de Variáveis

Grafo Def-Uso

- Definição:
 - Atribuição de um valor a uma variável.
 - $a = 1$
- Uso Predicativo (p-uso):
 - A variável é utilizada em uma condição.
 - $\text{if } (a > 0)$
- Uso Computacional (c-uso):
 - A variável é utilizada em uma computação.
 - $b = a + 1$



Grafo Def-Uso

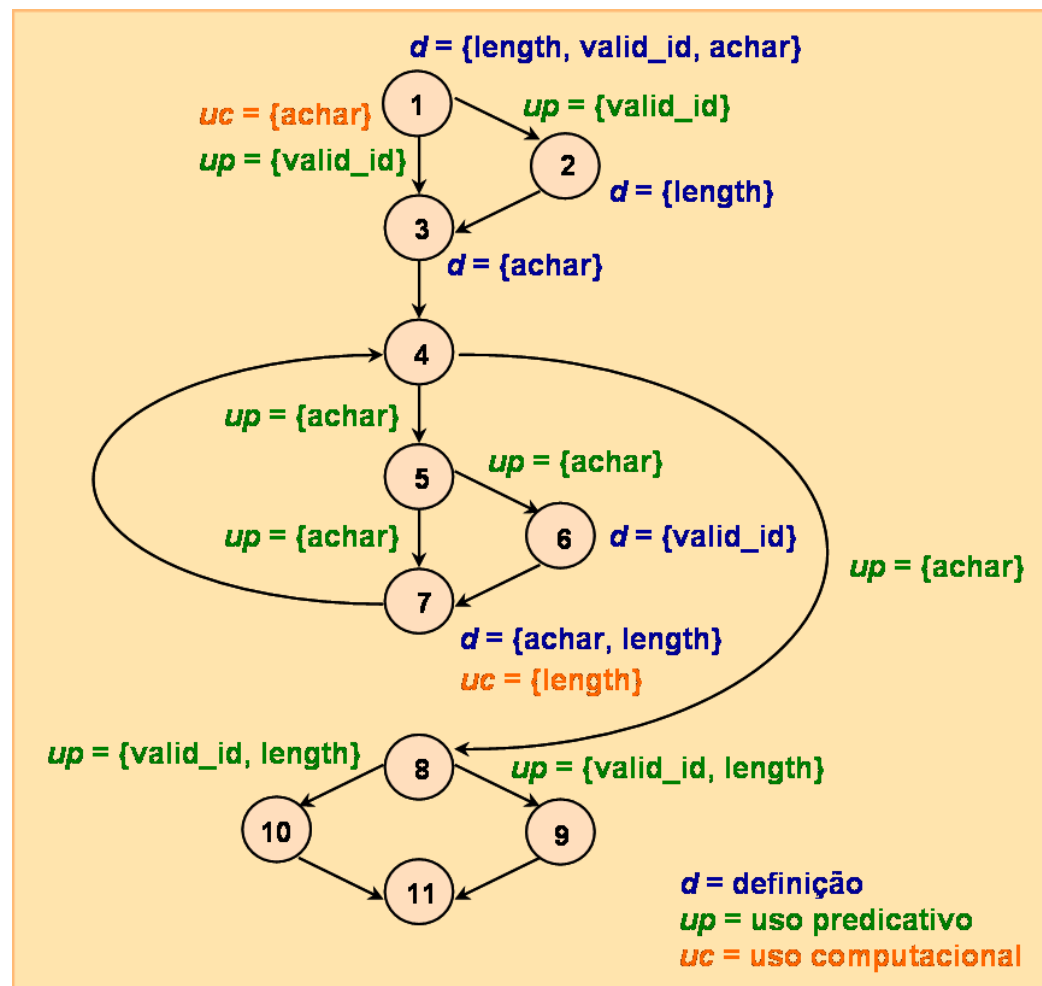
- O programa Identifier determina se um identificador é válido ou não. Um identificador válido deve começar com uma letra e conter apenas letras ou dígitos. Além disso, deve ter no mínimo um caractere e no máximo seis caracteres de comprimento.

Grafo Def-Uso

```
/* 01 */      {
/* 01 */      char  achar;
/* 01 */      int length, valid_id;
/* 01 */      length = 0;
/* 01 */      printf ("Identificador: ");
/* 01 */      achar = fgetc (stdin);
/* 01 */      valid_id = valid_s(achar);
/* 01 */      if (valid_id)
/* 02 */          length = 1;
/* 03 */      achar = fgetc (stdin);
/* 04 */      while (achar != '\n')
/* 05 */      {
/* 05 */          if (!(valid_f(achar)))
/* 06 */              valid_id = 0;
/* 07 */          length++;
/* 07 */          achar = fgetc (stdin);
/* 07 */      }
/* 08 */      if (valid_id && (length >= 1) && (length < 6))
/* 09 */          printf ("Valido\n");
/* 10 */      else
/* 10 */          printf ("Invalido\n");
/* 11 */      }
```

- Implementação do Programa Identifier (função main).
 - Função valid s(): determina se o primeiro caractere é válido.
 - Função valid f(): determina se o próximo caractere é válido.

Grafo Def-Uso do programa Identifier



Critério Todas as Definições

- Requer que cada definição de variável seja exercitada pelo menos uma vez, não importa se por um c-uso ou por um p-uso.
 - Para todas as definições de variáveis deve ser exercitado um caminho livre de definição para pelo menos um de seus usos.
 - Caminho onde a variável não é redefinida.

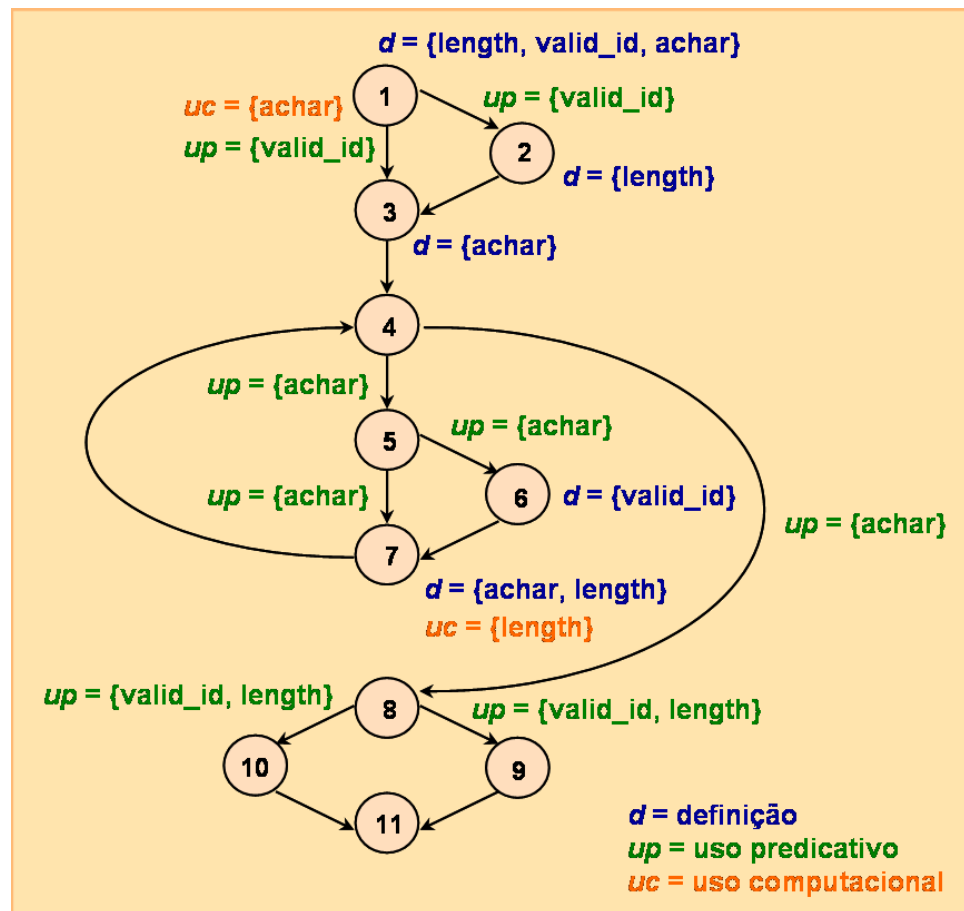
Critério Todas as Definições

- Associações Definição-Uso
 - Tripla $\langle i, j, \text{var} \rangle$ ou $\langle i, (j, k), \text{var} \rangle$ indicando que a variável var é definida no nó i e existe um uso computacional de var no nó j ou um uso predicativo de var no arco (j, k) , respectivamente, bem como pelo menos um caminho livre de definição do nó i ao nó j ou ao arco (j, k) .

Critério Todas as Definições

- Elementos Requeridos
 - (definição da variável length no nó 1):
- Associações: $\langle 1, 7, \text{length} \rangle$; $\langle 1, (8, 9), \text{length} \rangle$ ou $\langle 1, (8, 10), \text{length} \rangle$.
- Basta ser executado um dos seguintes sub-caminhos:
 - $(1, 3, 4, 5, 7)$; $(1, 3, 4, 5, 6, 7)$; $(1, 3, 4, 8, 9)$; $(1, 3, 4, 8, 10)$.

Critério Todas as Definições



Critério Todas as Definições

- Para satisfazer o critério Todas-Definições, a análise anterior deve ser realizada para toda definição que ocorre no programa.

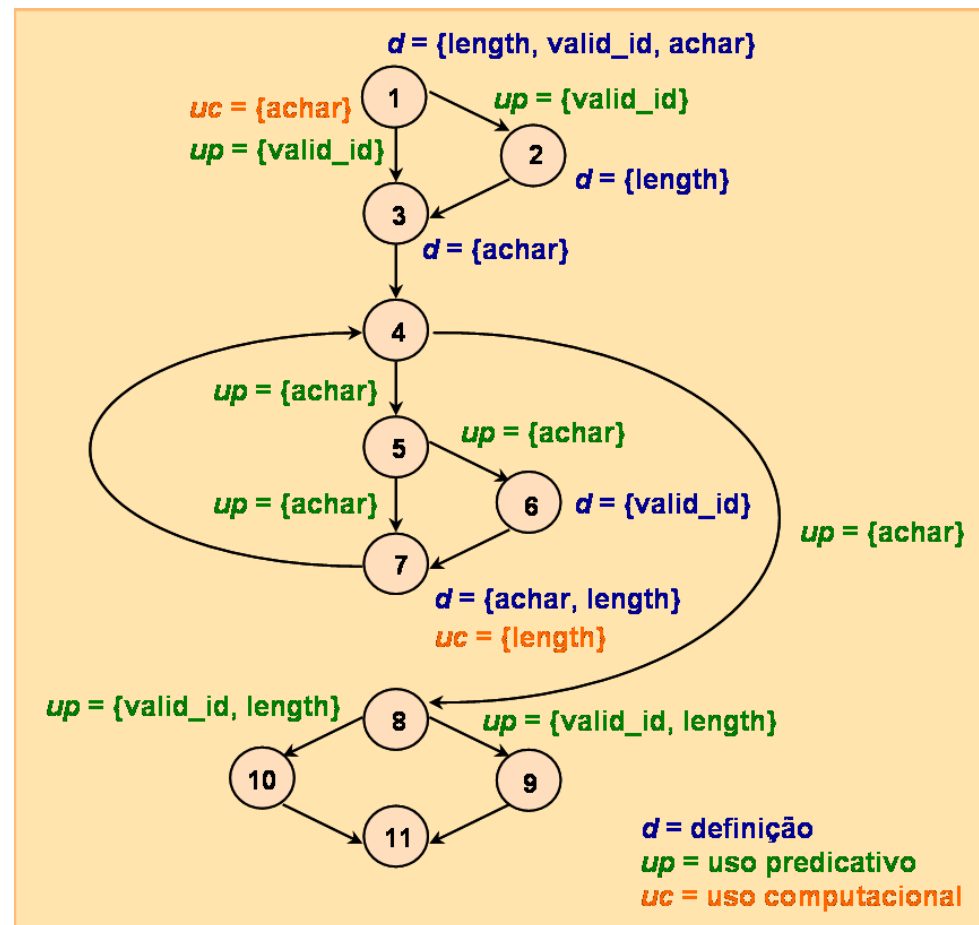
Nó	Variável	Caminhos Requeridos
1	length	(1,3,4,5,7)
		(1,3,4,5,6,7)
		(1,3,4,8,9) ×
		(1,3,4,8,10)
	valid_id	(1,2,3,4,8,9)
		(1,3,4,8,10)
	achar	(1,3,4,5,7)
		(1,3,4,5,6)
		(1,3,4,8)
2	length	(2,3,4,5,7)
		(2,3,4,5,6,7)
		(2,3,4,8,9)
		(2,3,4,8,10) ×
3	achar	(3,4,5,7)
		(3,4,5,6)
		(3,4,8)
6	valid_id	(6,7,4,8,9) ×
		(6,7,4,8,10)
7	length	(7,4,5,7)
		(7,4,5,6,7)
		(7,4,8,9)
		(7,4,8,10)
	achar	(7,4,5,7)
		(7,4,5,6)
		(7,4,8)

Critério Todos-Usos

- Requer que todas as associações entre uma definição de variável e seus subsequentes usos (c-usos e p-usos) sejam exercitadas pelos casos de teste, através de pelo menos um caminho livre de definição.
 - Para todas as definições de variáveis deve ser exercitado um caminho para todos os seus c-usos e para todos os seus p-usos.

Critério Todos-Usos

- Elementos Requeridos (definição da variável length no nó 1):
- Associações: $\langle 1, 7, \text{length} \rangle$; $\langle 1, (8, 9), \text{length} \rangle$ e $\langle 1, (8, 10), \text{length} \rangle$.



Critério Todos-Usos

- Para satisfazer o critério Todos-Usos, a análise anterior deve ser realizada para todas as demais variáveis e associações pertinentes.

Associações Requeridas
$\langle 1, 7, \{\text{length}\} \rangle$
$\langle 1, (8, 9), \{\text{length}, \text{valid_id}\} \rangle \times$
$\langle 1, (8, 10), \{\text{length}, \text{valid_id}\} \rangle$
$\langle 1, (1, 2), \{\text{valid_id}\} \rangle$
$\langle 1, (1, 3), \{\text{valid_id}\} \rangle$
$\langle 1, 1, \{\text{achar}\} \rangle$
$\langle 1, (4, 5), \{\text{achar}\} \rangle$
$\langle 1, (4, 8), \{\text{achar}\} \rangle$
$\langle 1, (5, 6), \{\text{achar}\} \rangle$
$\langle 1, (5, 7), \{\text{achar}\} \rangle$
$\langle 2, 7, \{\text{length}\} \rangle$
$\langle 2, (8, 9), \{\text{length}\} \rangle$
$\langle 2, (8, 10), \{\text{length}\} \rangle \times$
$\langle 3, (4, 5), \{\text{achar}\} \rangle$
$\langle 3, (4, 8), \{\text{achar}\} \rangle$
$\langle 3, (5, 6), \{\text{achar}\} \rangle$
$\langle 3, (5, 7), \{\text{achar}\} \rangle$
$\langle 6, (8, 9), \{\text{valid_id}\} \rangle \times$
$\langle 6, (8, 10), \{\text{valid_id}\} \rangle$
$\langle 7, 7, \{\text{length}\} \rangle$
$\langle 7, (8, 9), \{\text{length}\} \rangle$
$\langle 7, (8, 10), \{\text{length}\} \rangle$
$\langle 7, (4, 5), \{\text{achar}\} \rangle$
$\langle 7, (4, 8), \{\text{achar}\} \rangle$
$\langle 7, (5, 6), \{\text{achar}\} \rangle$
$\langle 7, (5, 7), \{\text{achar}\} \rangle$

Critério Todos-Usos

- Os critérios Todos-P-Usos, Todos-Usos/Alguns-C-Usos e Todos-C-Usos/Alguns-P-Usos representam variações do critério Todos-Usos.

Critério	Descrição
Todos-P-Usos	Para todas as definições de variáveis deve ser exercitado um caminho para todos os seus p-usos.
Todos-P-Usos/ Alguns-C-Usos	Para todas as definições de variáveis deve ser exercitado um caminho para todos os seus p-usos e alguns c-usos.
Todos-C-Usos/ Alguns-P-Usos	Para todas as definições de variáveis deve ser exercitado um caminho para todos os seus c-usos e para alguns p-usos.



Critérios Baseados em Fluxo de Dados

Resumo

- Critérios Baseados em Fluxo de Dados identificam pares definição-uso para variáveis de um programa.
- Critérios de Rapps e Weyuker
 - Grafo Def-Uso é a base a partir do qual os requisitos de testes são derivados.
 - Requisitos definidos como associações definição-uso.
- Critérios Potenciais-Usos
 - Grafo Def é a base a partir do qual os requisitos de testes são derivados.
 - Requisitos definidos como potenciais-associações.
 - Preenchem a lacuna entre os critérios de fluxo de controle Todos-Arcos e Todos-Caminhos.

Referências

- Notas de aula Profa. Ellen Francine Barbosa e Profa. Simone do Rocio Senger de Souza {francine, srocio}@icmc.usp.br.
- Notas de aula Prof. Marcio Eduardo Delamaro.
- E. F. Barbosa.; J. C. Maldonado; A. M. R. Vicenzi; M. E. Delamaro; S. R. S. Souza; & M. Jino. Introdução ao teste de software. In: XIV Simpósio Brasileiro de Engenharia de Software. João Pessoa, PB.
- G. J. Myers. The Art of Software Testing. Wiley, New Jersey, 2004.