

1 Alcançabilidade de Definições

Gen ...

Kill ...

IN ...

OUT ...

	Gen	Kill	IN	OUT
B_1	1000	1000	0000	1000
B_2	0100	0101	1000	1100
B_3	0010	0010	1000	1010
B_4	0001	0101	1110	1011
B_5	0000	0000	1100	1100
B_6	0000	0000	1010	1010

Table 1: Alcançabilidade de Definições — $(b := 0, a := b + c, x := b + 1, a := b + c)$

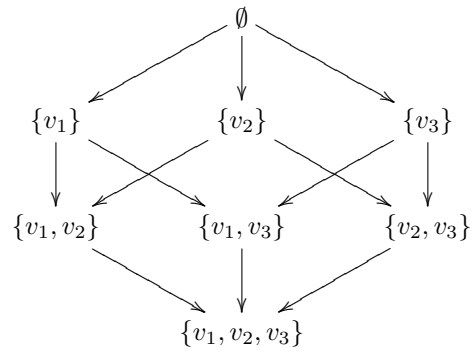
2 Vivacidade de Variáveis

Gen ...

Kill ...

IN ...

OUT ...



	Gen	Kill	IN	OUT
B_1	0000	0100	0010	0110
B_2	0110	1000	0110	0110
B_3	0100	0001	0110	0110
B_4	0110	1000	0110	0000
B_5	0000	0000	0110	0110
B_6	0000	0000	0110	0110

Table 2: Vivacidade de Variáveis — (a, b, c, x)

3 Disponibilidade de Expressões

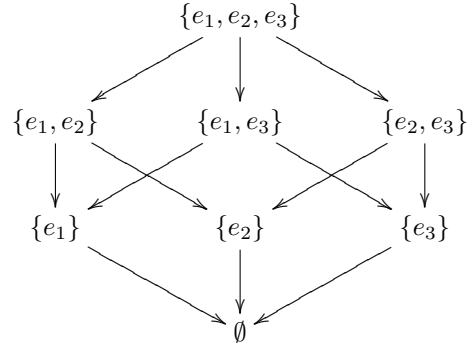
A análise é para frente (forward) e sua intenção é determinar em cada ponto do código, quais expressões estão disponíveis, isto é, foram seguramente executadas e, caso fossem executadas novamente (naquele ponto) produziriam o mesmo resultado.

Gen Indica quais expressões foram geradas dentro do bloco e que não foram “mortas” por redefinições de seus operandos dentro do mesmo bloco. **É igual à entrada das expressões antecipáveis.**

Kill Indica quais expressões (considerando o universo inteiro) foram mortas por redefinições (posteriores ¹) de seus operandos que ocorrem dentro do bloco.

IN Indica quais expressões estão disponíveis na entrada do bloco. É uma interseção das saídas dos blocos predecessores.

OUT Indica quais expressões estão disponíveis na saída do bloco. **É igual ao último *Gen*.**



	Gen	Kill	IN	OUT
B_1	00	11	00	00
B_2	10	00	00	10
B_3	01	00	00	01
B_4	10	00	00	10
B_5	00	00	10	10
B_6	00	00	01	01

Table 3: Disponibilidade de Expressões — $((+, b, c), (+, b, 1))$

¹Só faz sentido em análises internas ao bloco.

4 Disponibilidade de Expressões Antecipáveis

A análise é para trás (backward) e sua intenção é determinar em cada ponto do código, quais expressões podem ser movidas para o início do bloco (ou para blocos antecedentes).

Gen Indica quais expressões podem ser movidas para o início do bloco (ou para blocos antecedentes).

Kill Indica quais expressões (considerando o universo inteiro) foram mortas por redefinições (anteriores ²) de seus operandos que ocorrem dentro do bloco.

IN Indica quais expressões podem ser movidas para blocos antecedentes.

OUT Indica quais expressões de blocos subseqüentes podem ser movidas para o final do bloco atual – estas expressões podem ou não serem antecipadas pelo bloco atual.

	Gen	Kill	IN	OUT
B_1	00	11	00	10
B_2	10	00	10	10
B_3	01	00	11	10
B_4	10	00	10	00
B_5	00	00	10	10
B_6	00	00	10	10

Table 4: Disponibilidade de Expressões Antecipáveis — $((+, b, c), (+, b, 1))$

²Só faz sentido em análises internas ao bloco.

5 Disponibilidade Parcial de Expressões

Gen ...

Kill ...

IN ...

OUT ...

	Gen	Kill	IN	OUT
B_1	00	11	00	00
B_2	10	00	00	10
B_3	01	00	00	01
B_4	10	00	11	11
B_5	00	00	10	10
B_6	00	00	01	01

Table 5: Disponibilidade Parcial de Expressões — $((+, b, c), (+, b, 1))$

6 Mortalidade de Variáveis

Gen ...

Kill ...

In ...

In ...

	Gen	Kill	IN	OUT
B_1	0100	0000	1101	1001
B_2	1000	0110	1001	1001
B_3	0001	0100	1001	1001
B_4	1000	0110	1001	1111
B_5	0000	0000	1001	1001
B_6	0000	0000	1001	1001

Table 6: Mortalidade de Variáveis — (a, b, c, x)

7 Alcançabilidade de Definições para Propagação de Cópias

Gen ...

Kill ...

In ...

In ...

	Gen	Kill	IN	OUT
B_1	1000	1000	0000	1000
B_2	0000	0000	1000	1000
B_3	0000	0000	1000	1000
B_4	0000	0000	1000	1000
B_5	0000	0000	1000	1000
B_6	0000	0000	1000	1000

Table 7: Alcançabilidade de Definições para Propagação de Cópias — $(b := 0, a := b + c, x := b + 1, a := b + c)$

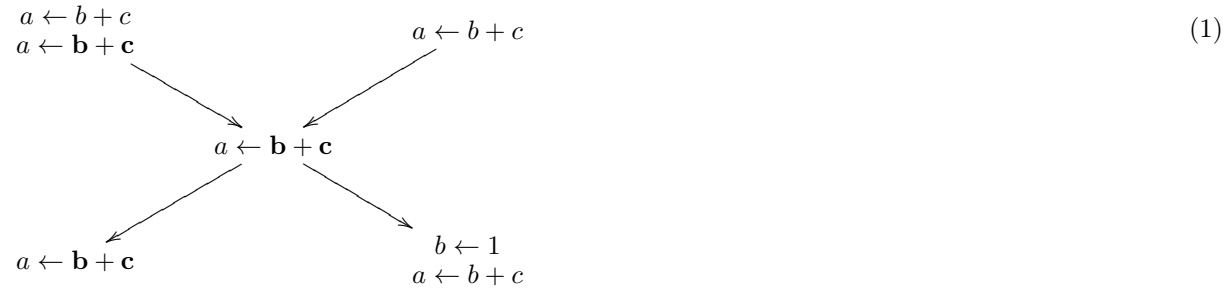
8 Eliminação de Redundâncias Parciais

8.1 Expressão Redundante

Uma expressão é *redundante* no ponto p se em cada caminho até p :

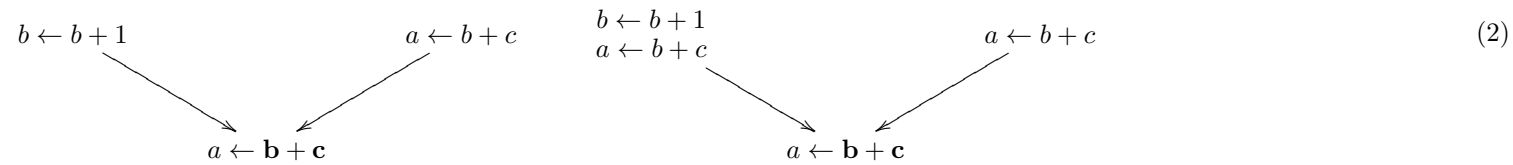
1. Ela é avaliada antes de alcançar p , e
2. Nenhum de seus operandos constituintes é redefinido antes de p .

Por exemplo, na Equação 1, as ocorrências de expressões em negrito são redundantes.



Uma expressão é *parcialmente redundante* no ponto p se ela é redundante ao longo de alguns caminhos, mas não todos, até p .

Por exemplo, na Equação 2, a expressão $b + c$ em negrito no diagrama da esquerda é parcialmente redundante. A inserção de uma cópia de $b + c$ depois da definição de b pode tornar uma expressão parcialmente redundante em uma totalmente redundante como mostra o diagrama da direita.



	ENTRY	B_1	B_2	B_3	B_4	B_5	B_6	EXIT
e_gen	{00}	{00}	{10}	{01}	{10}	{00}	{00}	{00}
e_kill	{00}	{11}	{00}	{00}	{00}	{00}	{00}	{00}
anticipated_out	{00}	{10}	{10}	{10}	{00}	{10}	{10}	{00}
anticipated_in	{00}	{00}	{10}	{11}	{10}	{10}	{10}	{00}
available_in	{00}	{00}	{00}	{00}	{10}	{10}	{11}	{10}
available_out	{00}	{00}	{10}	{11}	{10}	{10}	{11}	{10}
earliest	{00}	{00}	{10}	{11}	{00}	{00}	{00}	{00}
postponable_in	{00}	{00}	{00}	{00}	{00}	{00}	{10}	{00}
postponable_out	{00}	{00}	{00}	{10}	{00}	{00}	{10}	{00}
latest	{00}	{00}	{10}	{01}	{00}	{00}	{10}	{00}
used_out	{00}	{00}	{10}	{00}	{00}	{10}	{10}	{00}
used_in	{00}	{00}	{00}	{00}	{10}	{10}	{00}	{00}
<i>cond_1</i>	{00}	{00}	{10}	{00}	{00}	{00}	{10}	{00}
<i>cond_2</i>	{00}	{00}	{10}	{00}	{10}	{00}	{00}	{00}

Table 8: Eliminação de Redundâncias Parciais — $((+, b, c), (+, b, 1))$

9 Propagação de Constantes

Gen ...

Kill ...

In ...

In ...

	IN	OUT
B_1	$(\top, \top, \top, \top, \top, \top)$	$(\top, 0, \top, \top, \top, \top)$
B_2	$(\top, 0, \top, \top, \top, \top)$	$(\top, 0, \top, \top, \top, \top)$
B_3	$(\top, 0, \top, \top, \top, \top)$	$(\top, 0, \top, 1, \top, \top)$
B_4	$(\top, 0, \top, 1, \top, \top)$	$(\top, 0, \top, 1, \top, \top)$
B_5	$(\top, 0, \top, \top, \top, \top)$	$(\top, 0, \top, \top, \top, \top)$
B_6	$(\top, 0, \top, 1, \top, \top)$	$(\top, 0, \top, 1, \top, \top)$

Table 9: Propagação de Constantes — (a, b, c, x, t_5, t_6)

