CFG

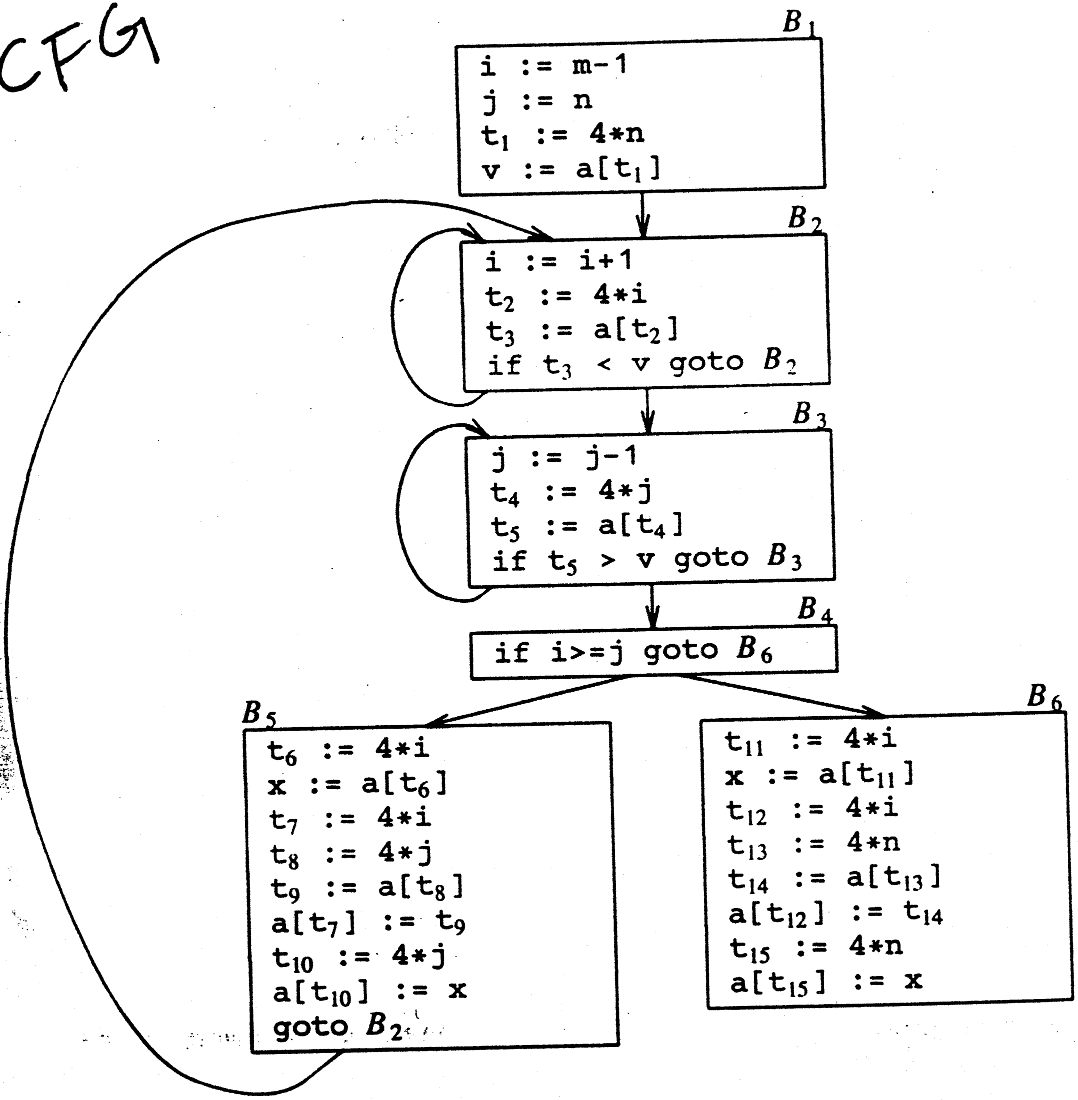


Fig. 10.5. Flow graph.

LOCAL CSE

B_5

```
t6 := 4*i
x := a[t6]
t7 := 4*i
t8 := 4*j
t9 := a[t8]
a[t7] := t9
t10 := 4*j
a[t10] := x
goto B2
```

(a) Before

B_5

```
t6 := 4*i
x := a[t6]
t8 := 4*j
t9 := a[t8]
a[t6] := t9
a[t8] := x
goto B2
```

(b) After

Fig. 10.6. Local common subexpression elimination.

GLOBAL CSE

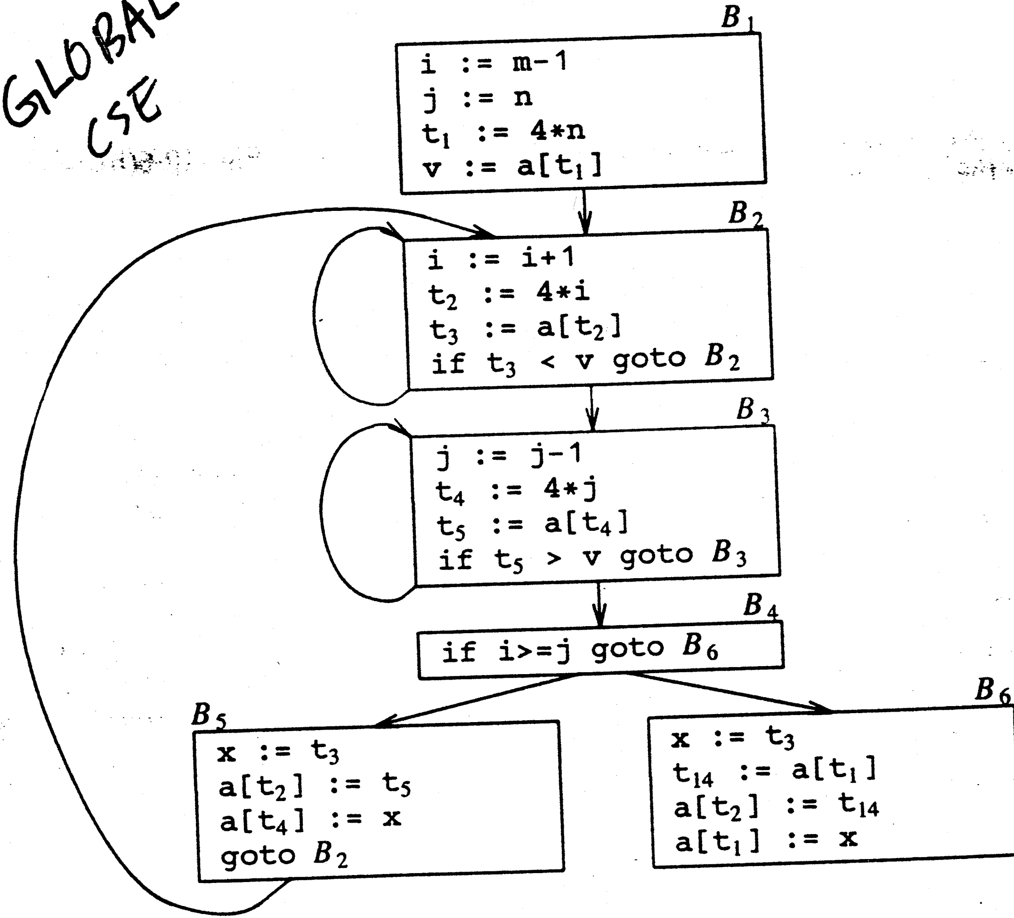
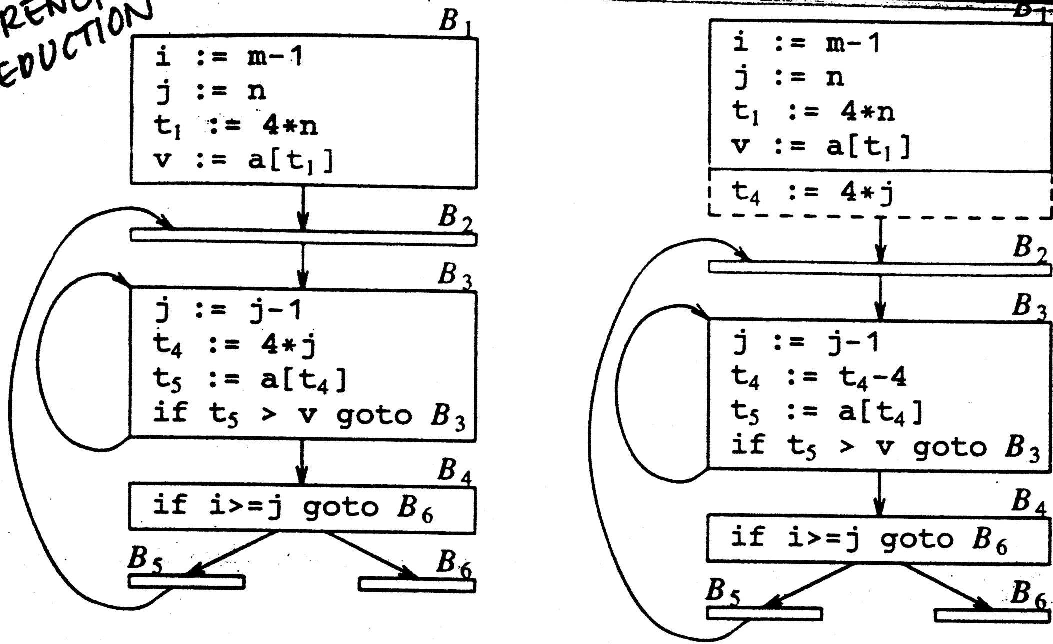


Fig. 10.7. B_5 and B_6 after common subexpression elimination.

STRENGTH REDUCTION



(a) Before　　(b) After

Fig. 10.9. Strength reduction applied to `4*j` in block B_3.

INDUCTION VARIABLE ELIMINATION

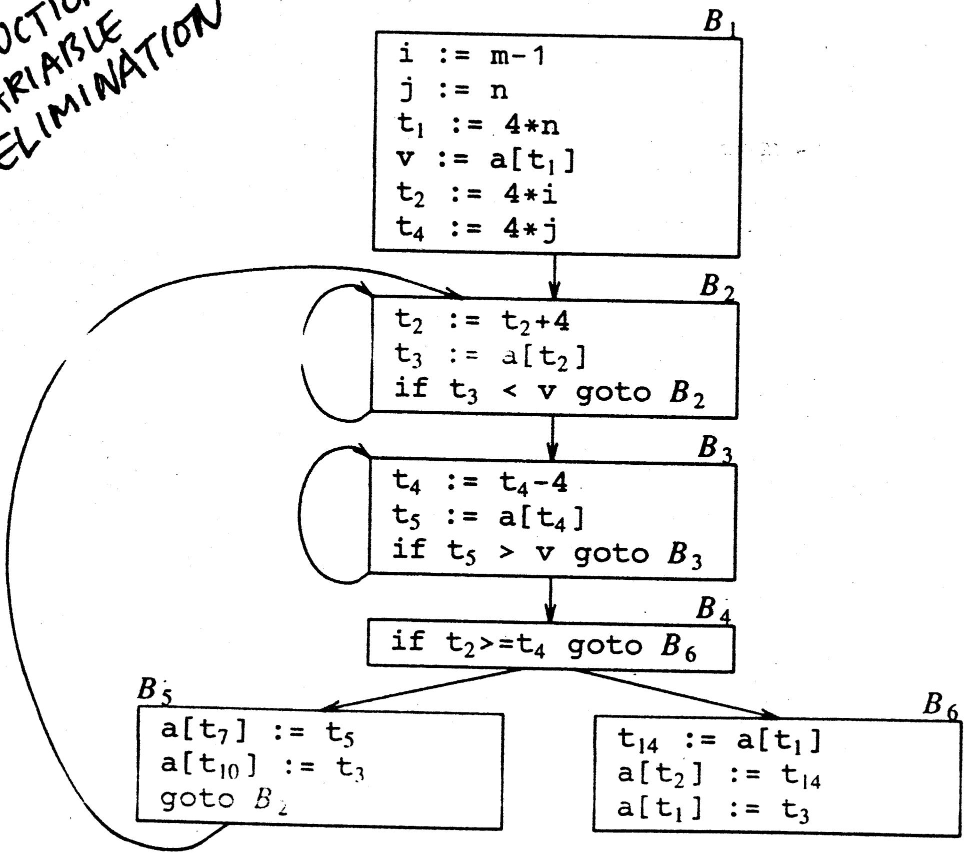


Fig. 10.10. Flow graph after induction-variable elimination.

CFG

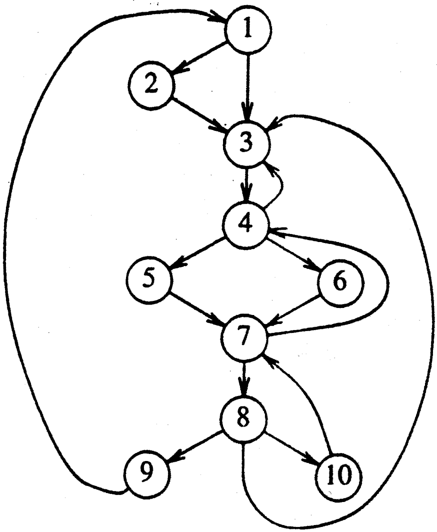


Fig. 10.13. Flow graph.

Dominator Tree

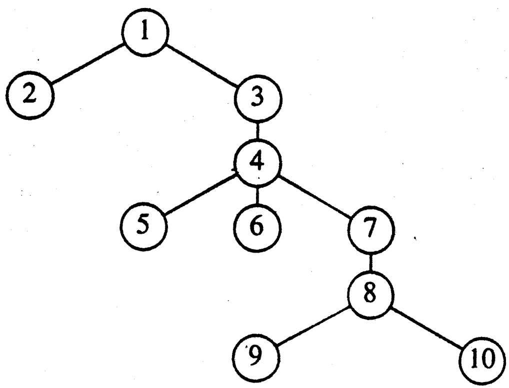
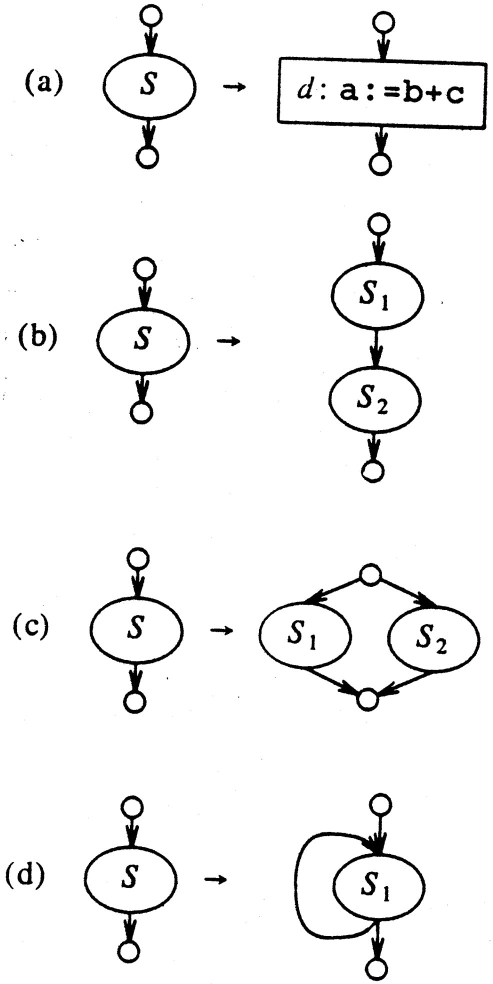


Fig. 10.14. Dominator tree for flow graph of Fig. 10.13.



(a)

$$gen[S] = \{d\}$$
$$kill[S] = D_a - \{d\}$$
$$out[S] = gen[S] \cup (in[S] - kill[S])$$

(b)

$$gen[S] = gen[S_2] \cup (gen[S_1] - kill[S_2])$$
$$kill[S] = kill[S_2] \cup (kill[S_1] - gen[S_2])$$
$$in[S_1] = in[S]$$
$$in[S_2] = out[S_1]$$
$$out[S] = out[S_2]$$

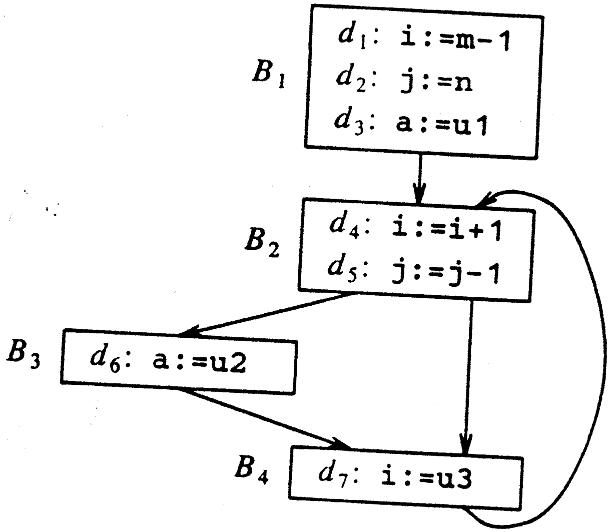
(c)

$$gen[S] = gen[S_1] \cup gen[S_2]$$
$$kill[S] = kill[S_1] \cap kill[S_2]$$
$$in[S_1] = in[S]$$
$$in[S_2] = in[S]$$
$$out[S] = out[S_1] \cup out[S_2]$$

(d)

$$gen[S] = gen[S_1]$$
$$kill[S] = kill[S_1]$$
$$in[S_1] = in[S] \cup gen[S_1]$$
$$out[S] = out[S_1]$$

Fig. 10.21. Data-flow equations for reaching definitions.



$$gen[B_1] = \{d_1, d_2, d_3\}$$
$$kill[B_1] = \{d_4, d_5, d_6, d_7\}$$

$$gen[B_2] = \{d_4, d_5\}$$
$$kill[B_2] = \{d_1, d_2, d_7\}$$

$$gen[B_3] = \{d_6\}$$
$$kill[B_3] = \{d_3\}$$

$$gen[B_4] = \{d_7\}$$
$$kill[B_4] = \{d_1, d_4\}$$

Fig. 10.27. Flow graph for illustrating reaching definitions.

BLOCK B	INITIAL		PASS 1		PASS 2	
	in [B]	*out* [B]	*in* [B]	*out* [B]	*in* [B]	*out* [B]
B_1	000 0000	111 0000	000 0000	111 0000	000 0000	111 0000
B_2	000 0000	000 1100	111 0001	001 1100	111 0111	001 1110
B_3	000 0000	000 0010	001 1100	000 1110	001 1110	000 1110
B_4	000 0000	000 0001	001 1110	001 0111	001 1110	001 0111

Fig. 10.28. Computation of *in* and *out*.

```
in[B1] := ∅;
out[B1] := e_gen[B1];   /* in and out never change for the initial node, B1 */
for B ≠ B1 do out[B] := U − e_kill[B];   /* initial estimate is too large */
change := true;
while change do begin
        change := false;
        for B ≠ B1 do begin
                in[B] :=        ∩         out[P];
                          P a prede-
                          cessor of B
                oldout := out[B];
                out[B] := e_gen[B] ∪ (in[B] − e_kill[B]);
                if out[B] ≠ oldout then change := true
        end
end
```

Fig. 10.32. Available expressions computation.