Register Allocation

TEACHING ASSISTANT: DAVID TRABISH

Register Allocation

- We have more IR variables than CPU registers
- We need to reduce the number of IR variables:
  - A **mapping** between **IR variables** and **CPU registers**

Register Allocation

For each function:

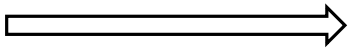
- Construct the CFG (from the IR)
- Run liveness analysis
- Construct the interference graph
- Compute a *k-coloring* of the graph
- Use the coloring to build the required mapping

Control Flow Graph

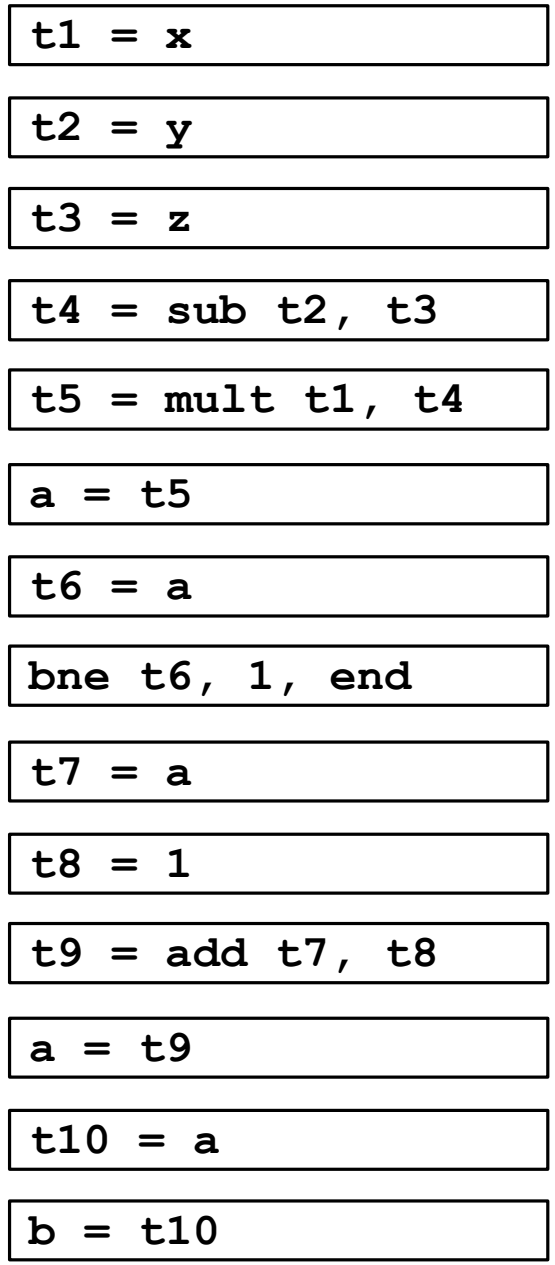
- Create a node for each IR instruction
- Create an edge between an instruction and its next instruction
- If the instruction is a **branch**:
  - Connect it to the instruction the comes after the **target label**
  - If the branch is conditional, connect to it’s next instruction

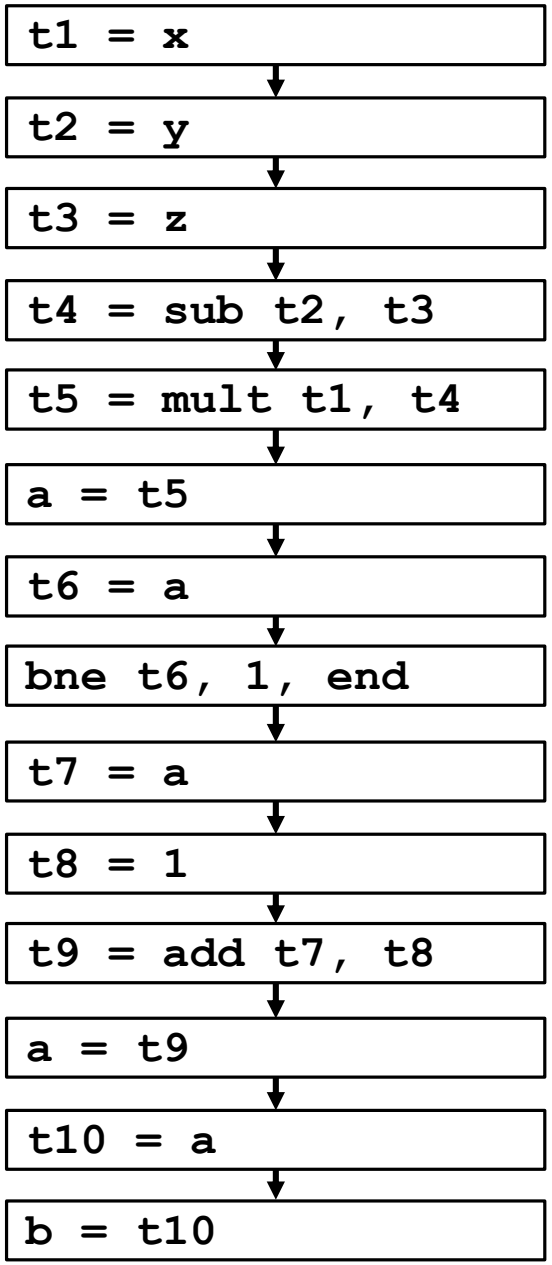
Control Flow Graph

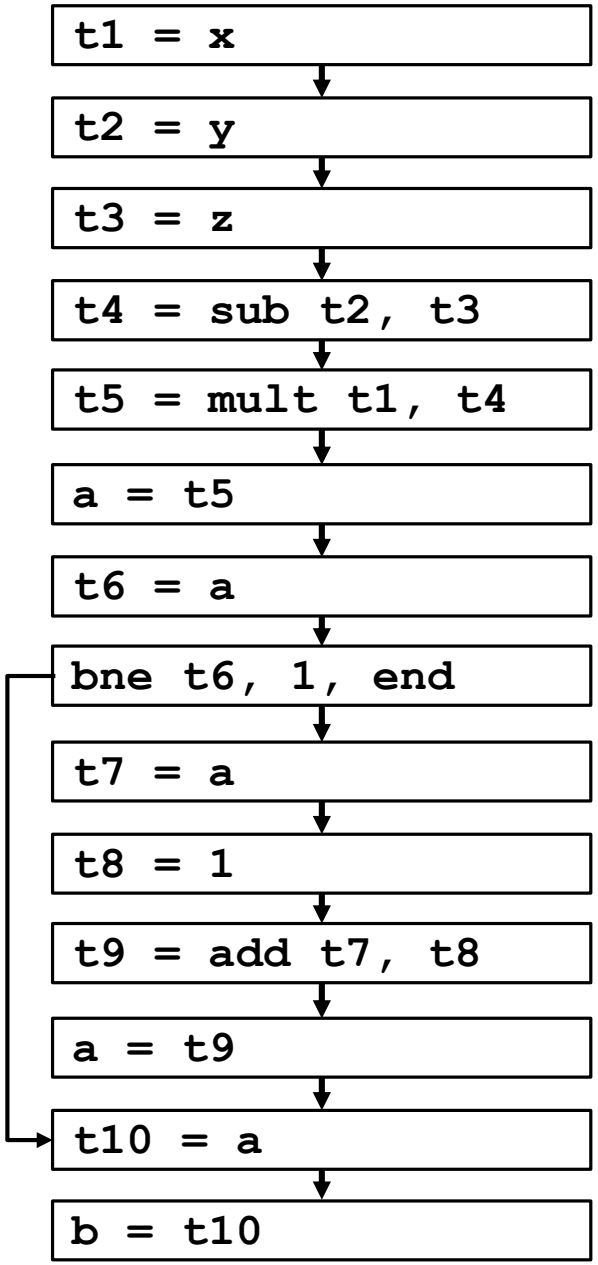
```
a = x * (y - z)
if (a) {
  a = a + 1;
}
b = a
```



```
t1 = x
t2 = y
t3 = z
t4 = sub t2, t3
t5 = mult, t1, t4
a = t5
t6 = a
bne t6, 1, end
t7 = a
t8 = 1
t9 = add t7, t8
a = t9
end:
t10 = a
b = t10
```

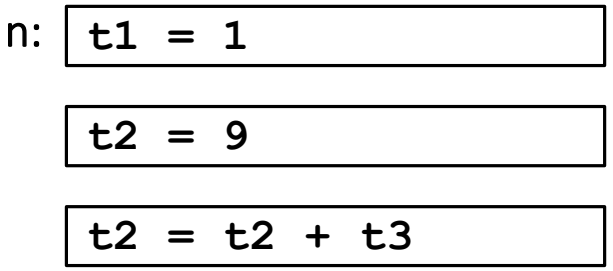

t1 = x
t2 = y
t3 = z
t4 = sub t2, t3
t5 = mult t1, t4
a = t5
t6 = a
bne t6, 1, end
t7 = a
t8 = 1
t9 = add t7, t8
a = t9
t10 = a
b = t10


t1 = x
t2 = y
t3 = z
t4 = sub t2, t3
t5 = mult t1, t4
a = t5
t6 = a
bne t6, 1, end
t7 = a
t8 = 1
t9 = add t7, t8
a = t9
t10 = a
b = t10

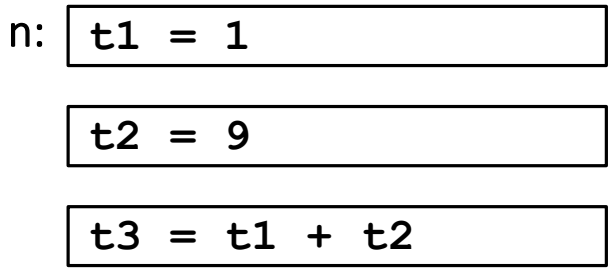

t1 = x
t2 = y
t3 = z
t4 = sub t2, t3
t5 = mult t1, t4
a = t5
t6 = a
bne t6, 1, end
t7 = a
t8 = 1
t9 = add t7, t8
a = t9
t10 = a
b = t10

Liveness Analysis

- Determine live variables at each program location
- A variable x is **live** at location n if:
  - There is a path from n where x is read before it's overwritten



t3 is live



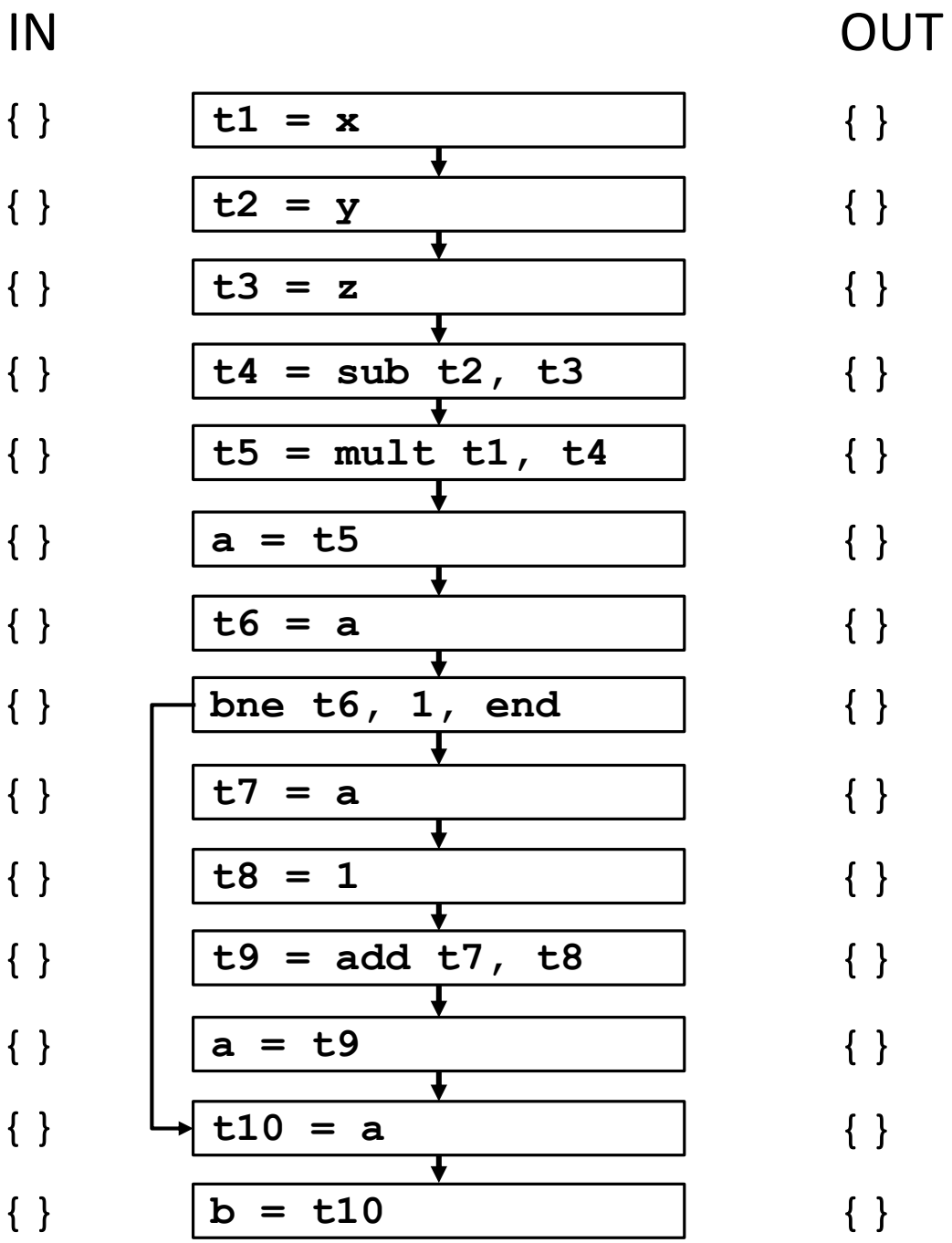
t3 is dead

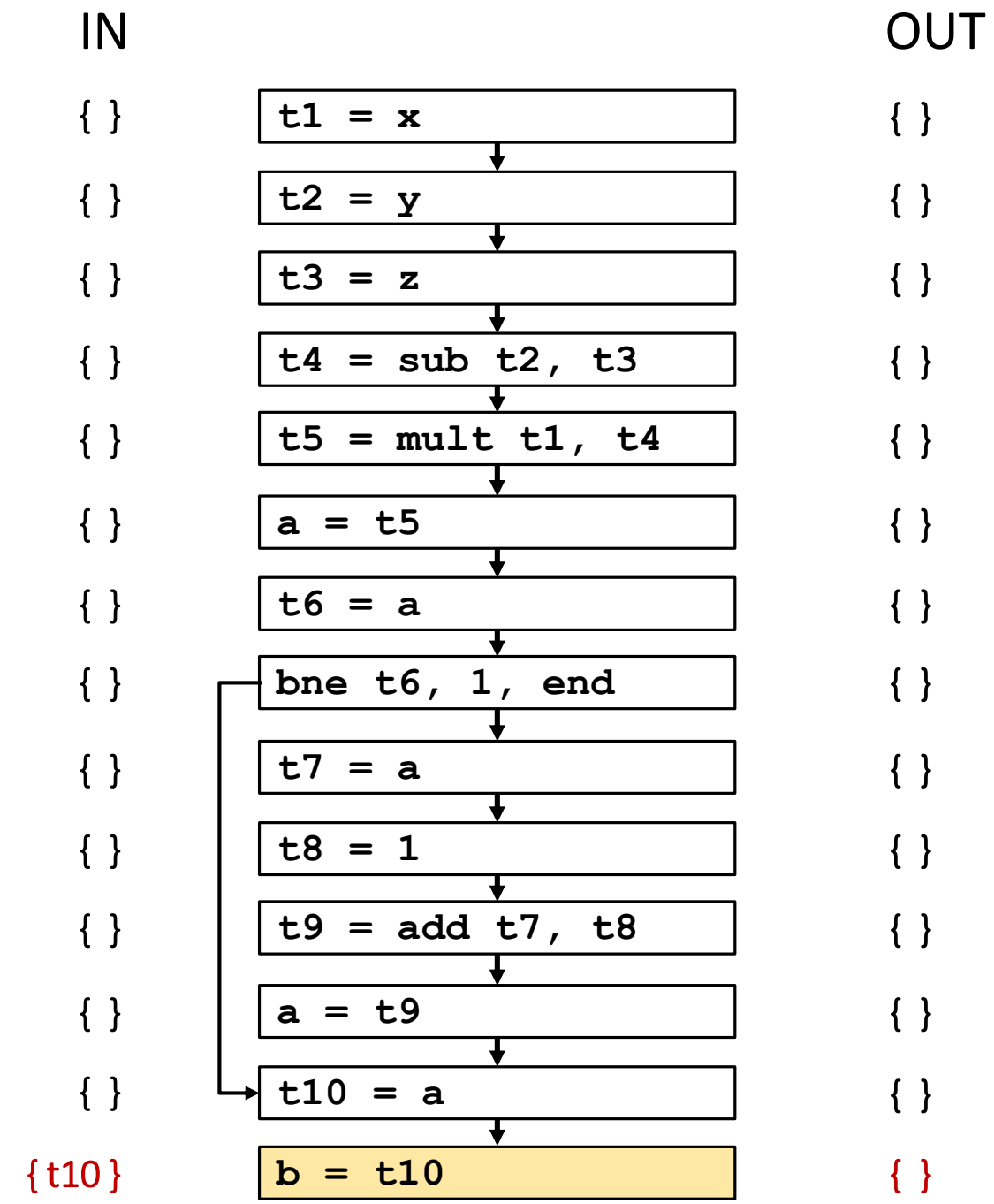
Liveness Analysis

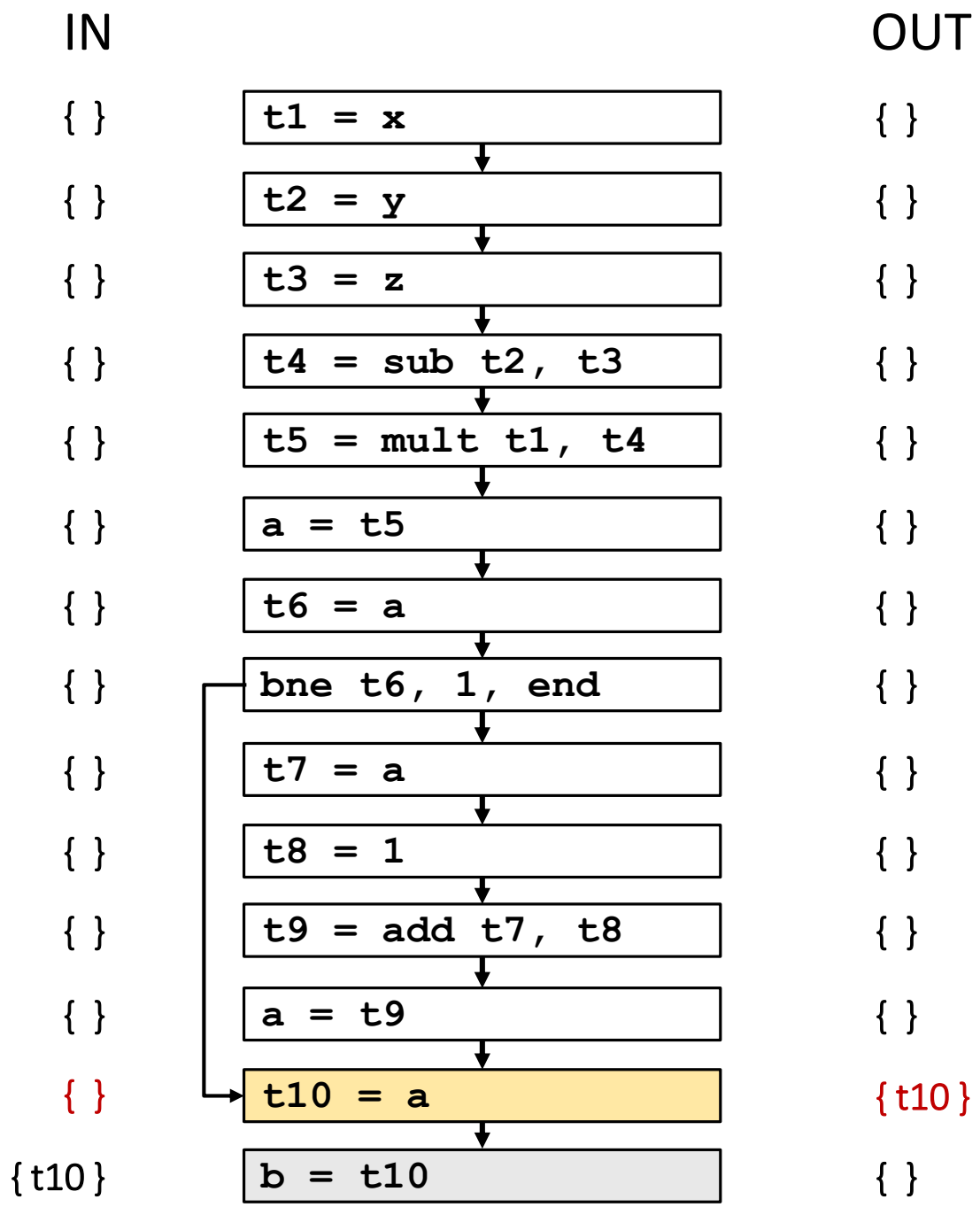
- Step 1: Initialize $out(n)$ and $in(n)$ to empty sets for each n
- Step 2:
  - Compute $out(n)$ and $in(n)$ for each node n
    - Assuming that n is the instruction $y = f(x_1, x_2, \dots)$
    - $out(n) = \cup\, in(s)$:
      - for each successor s of n
    - $in(n) = \{x_1, x_2, \dots\} \cup (out(n) \setminus \{y\})$
- Step 3:
  - If at least one node was changed (in or out)
    - Go to step 2

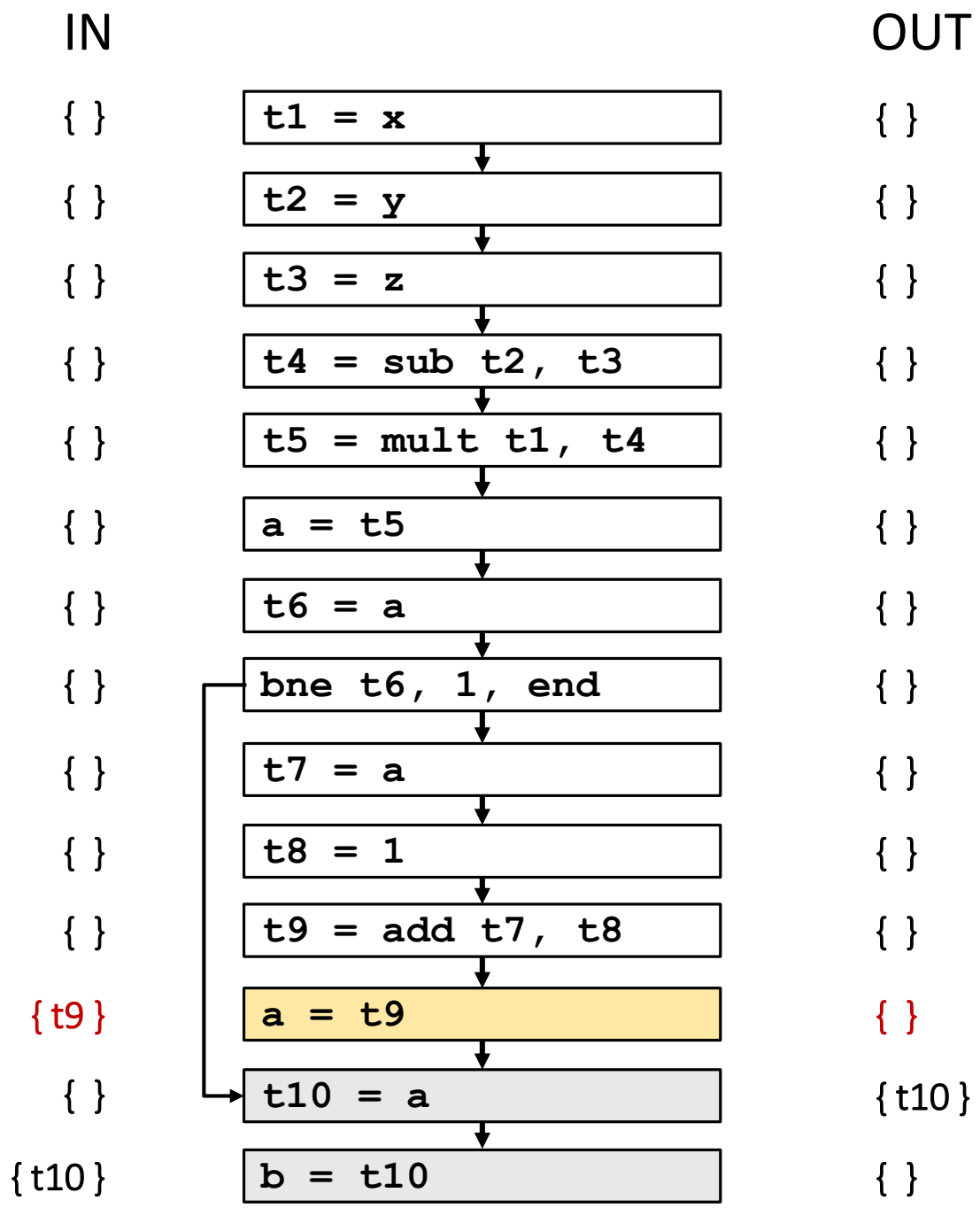
Liveness Analysis

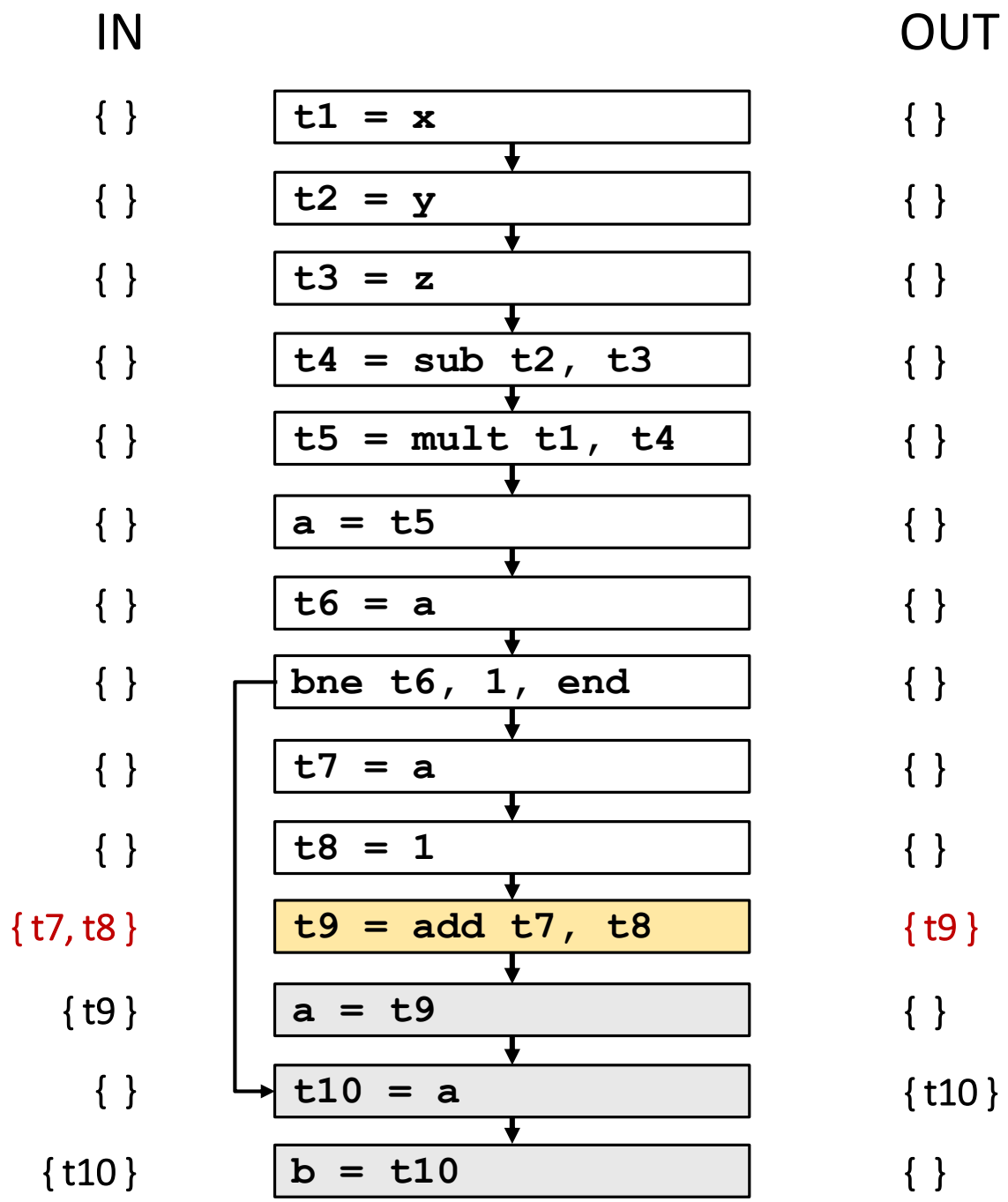
First Iteration…

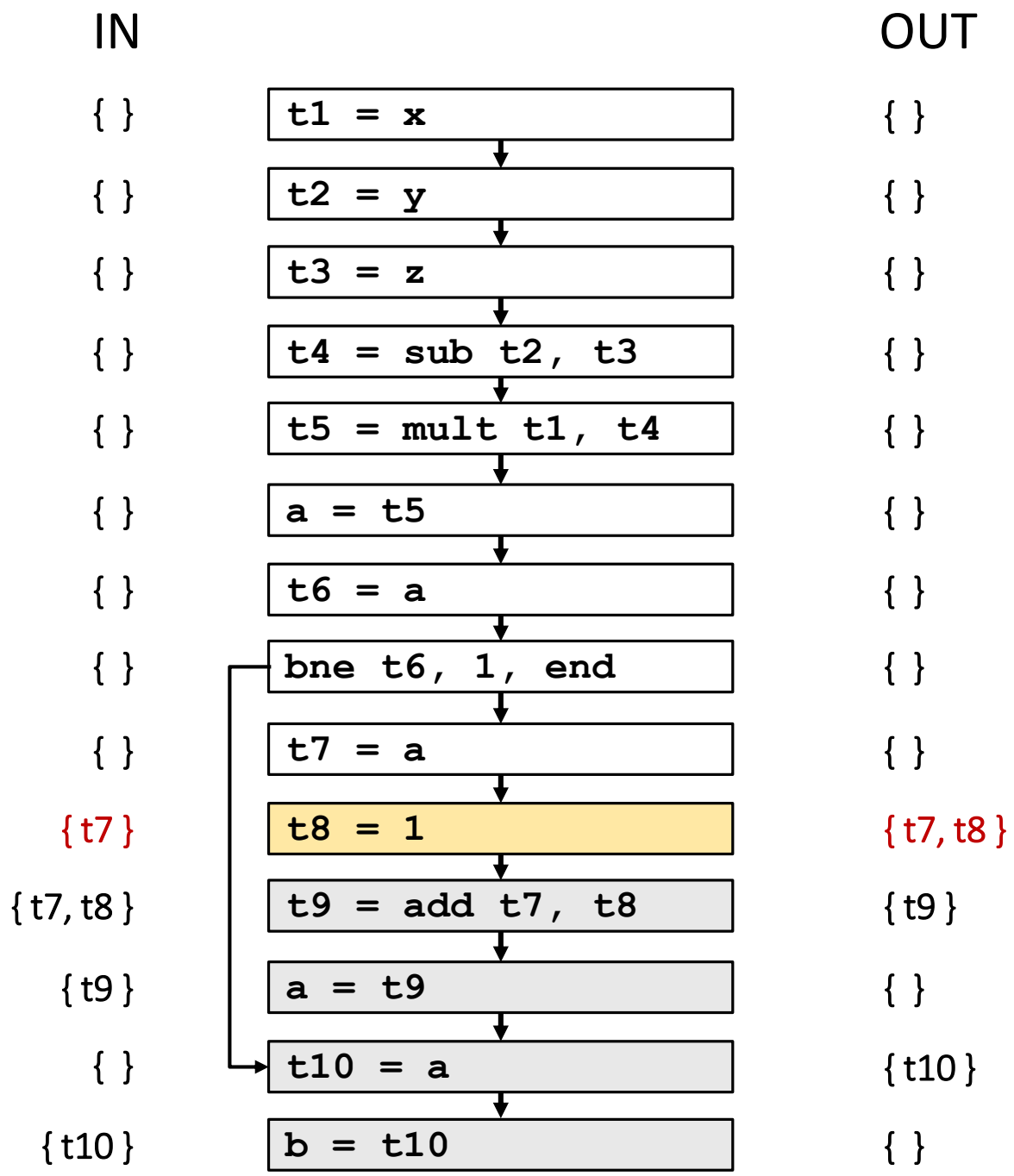

IN
OUT
{ }
t1 = x
t2 = y
t3 = z
t4 = sub t2, t3
t5 = mult t1, t4
a = t5
t6 = a
bne t6, 1, end
t7 = a
t8 = 1
t9 = add t7, t8
a = t9
t10 = a
b = t10

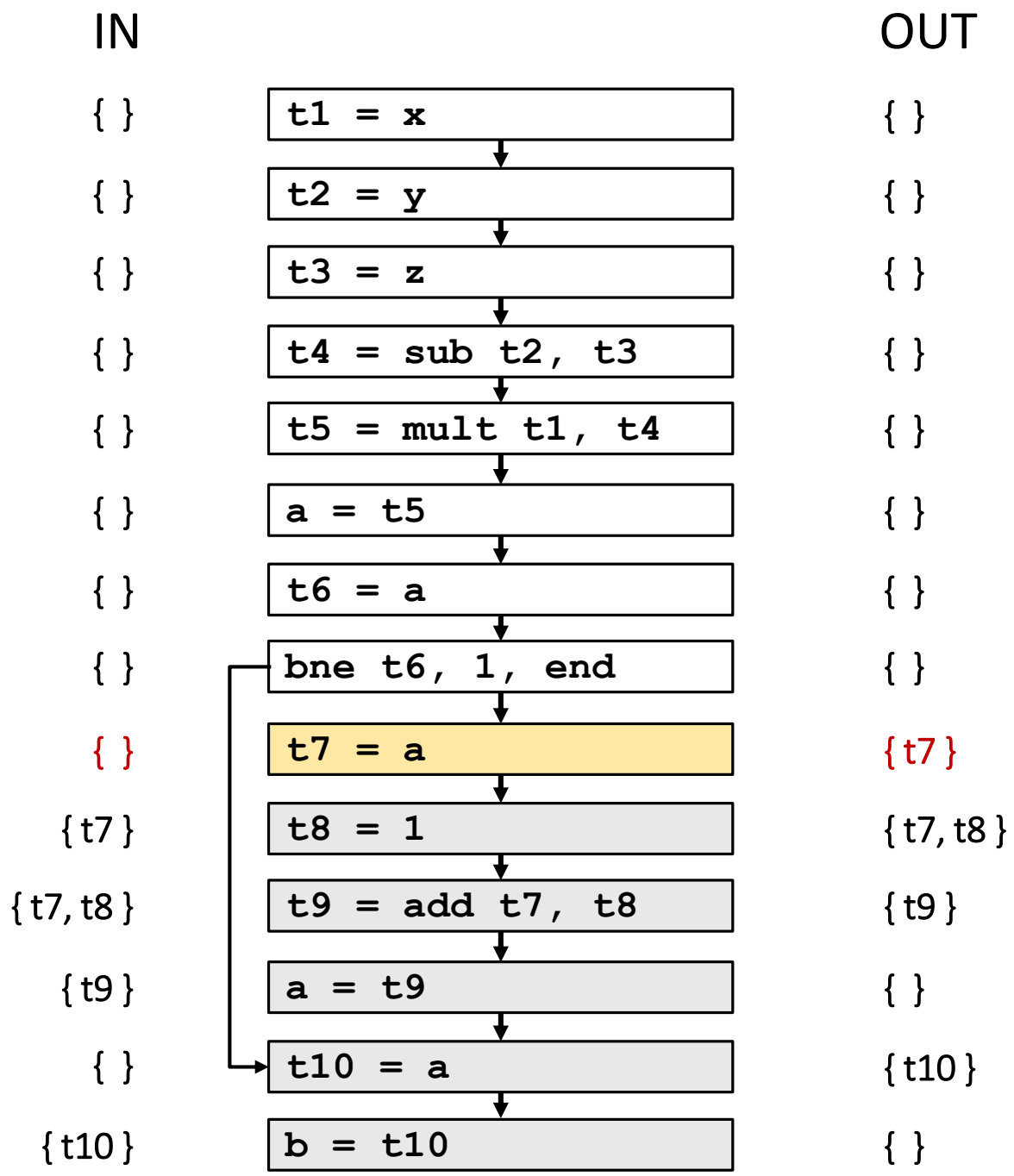

IN
OUT
{ }
t1 = x
{ }
{ }
t2 = y
{ }
{ }
t3 = z
{ }
{ }
t4 = sub t2, t3
{ }
{ }
t5 = mult t1, t4
{ }
{ }
a = t5
{ }
{ }
t6 = a
{ }
{ }
bne t6, 1, end
{ }
{ }
t7 = a
{ }
{ }
t8 = 1
{ }
{ }
t9 = add t7, t8
{ }
{ }
a = t9
{ }
{ }
t10 = a
{ }
{t10}
b = t10
{ }

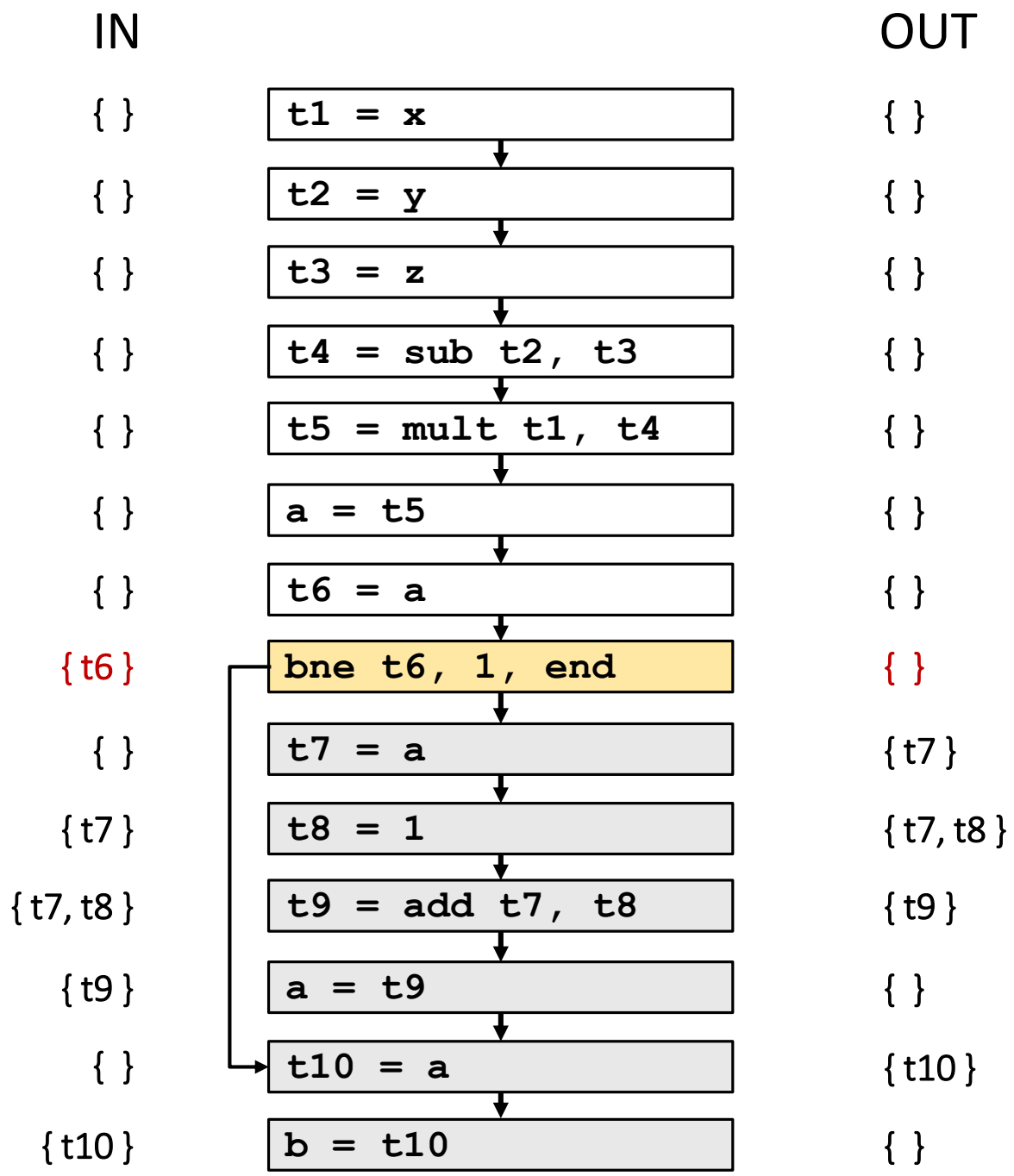

IN
OUT
{ }
t1 = x
{ }
{ }
t2 = y
{ }
{ }
t3 = z
{ }
{ }
t4 = sub t2, t3
{ }
{ }
t5 = mult t1, t4
{ }
{ }
a = t5
{ }
{ }
t6 = a
{ }
{ }
bne t6, 1, end
{ }
{ }
t7 = a
{ }
{ }
t8 = 1
{ }
{ }
t9 = add t7, t8
{ }
{ }
a = t9
{ }
{ }
t10 = a
{t10}
{t10}
b = t10
{ }

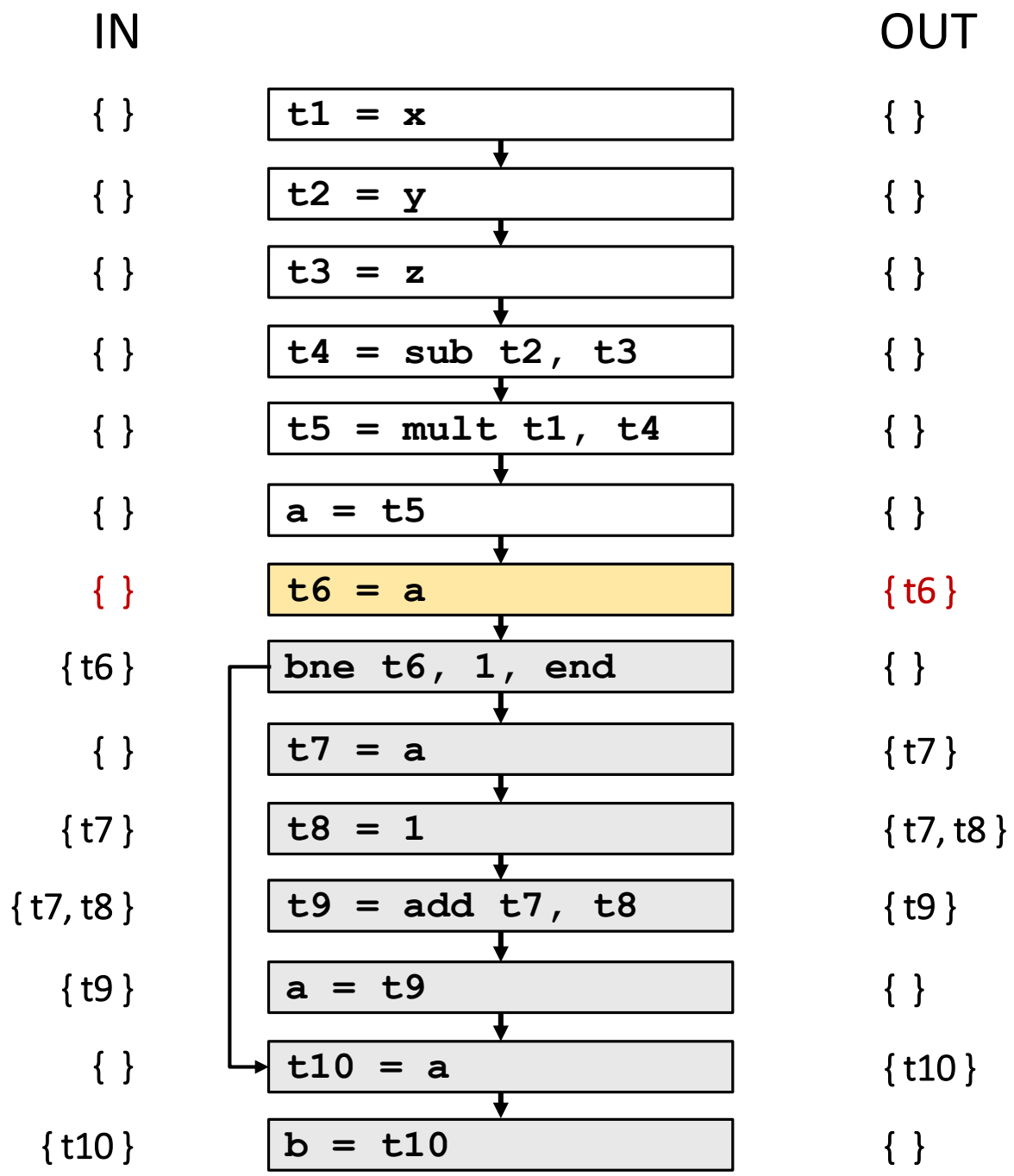

IN
OUT
{ }
t1 = x
{ }
{ }
t2 = y
{ }
{ }
t3 = z
{ }
{ }
t4 = sub t2, t3
{ }
{ }
t5 = mult t1, t4
{ }
{ }
a = t5
{ }
{ }
t6 = a
{ }
{ }
bne t6, 1, end
{ }
{ }
t7 = a
{ }
{ }
t8 = 1
{ }
{ }
t9 = add t7, t8
{ }
{t9}
a = t9
{ }
{ }
t10 = a
{t10}
{t10}
b = t10
{ }

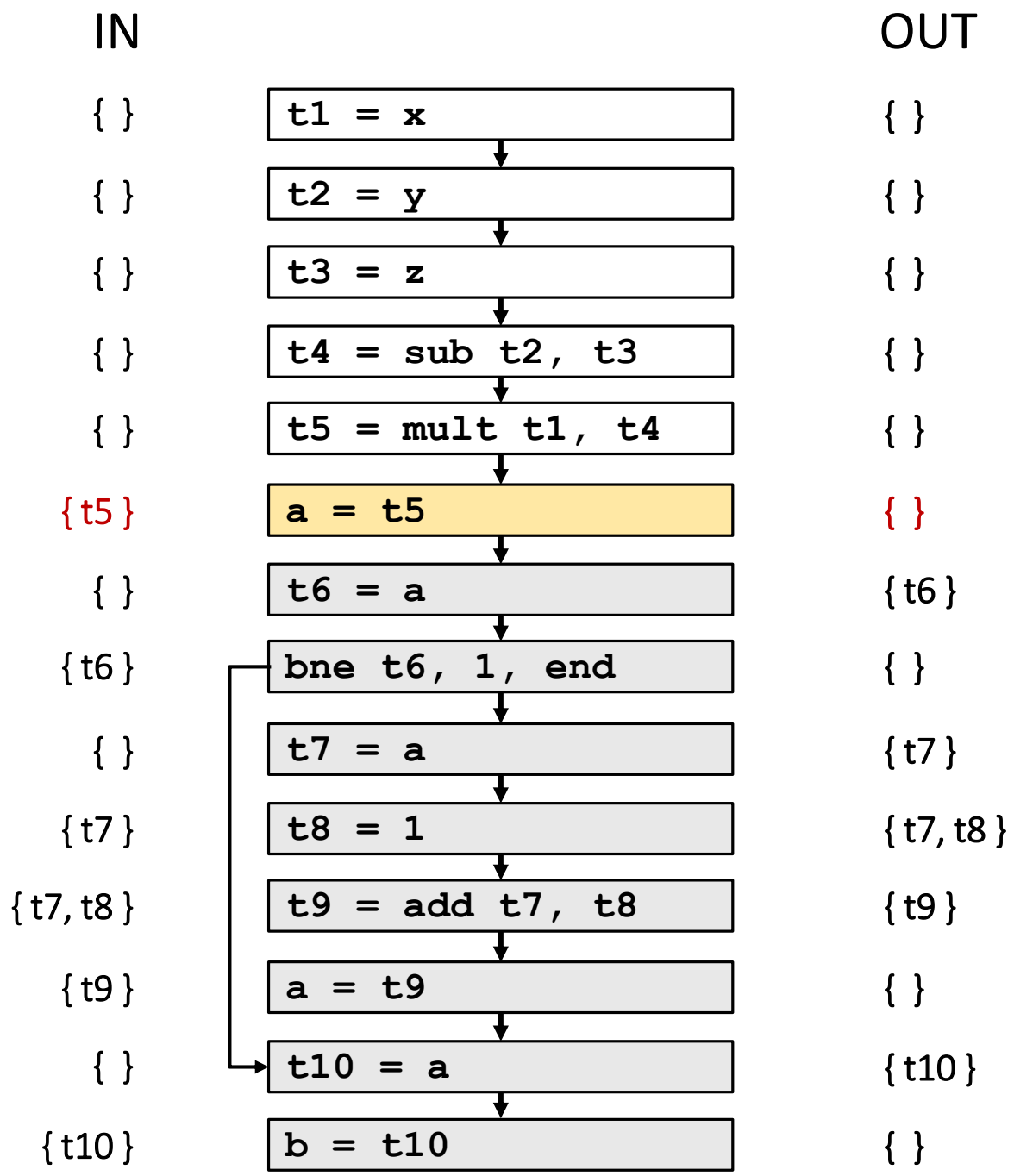

IN
OUT
{ }
t1 = x
{ }
{ }
t2 = y
{ }
{ }
t3 = z
{ }
{ }
t4 = sub t2, t3
{ }
{ }
t5 = mult t1, t4
{ }
{ }
a = t5
{ }
{ }
t6 = a
{ }
{ }
bne t6, 1, end
{ }
{ }
t7 = a
{ }
{ }
t8 = 1
{ }
{t7, t8}
t9 = add t7, t8
{t9}
{t9}
a = t9
{ }
{ }
t10 = a
{t10}
{t10}
b = t10
{ }

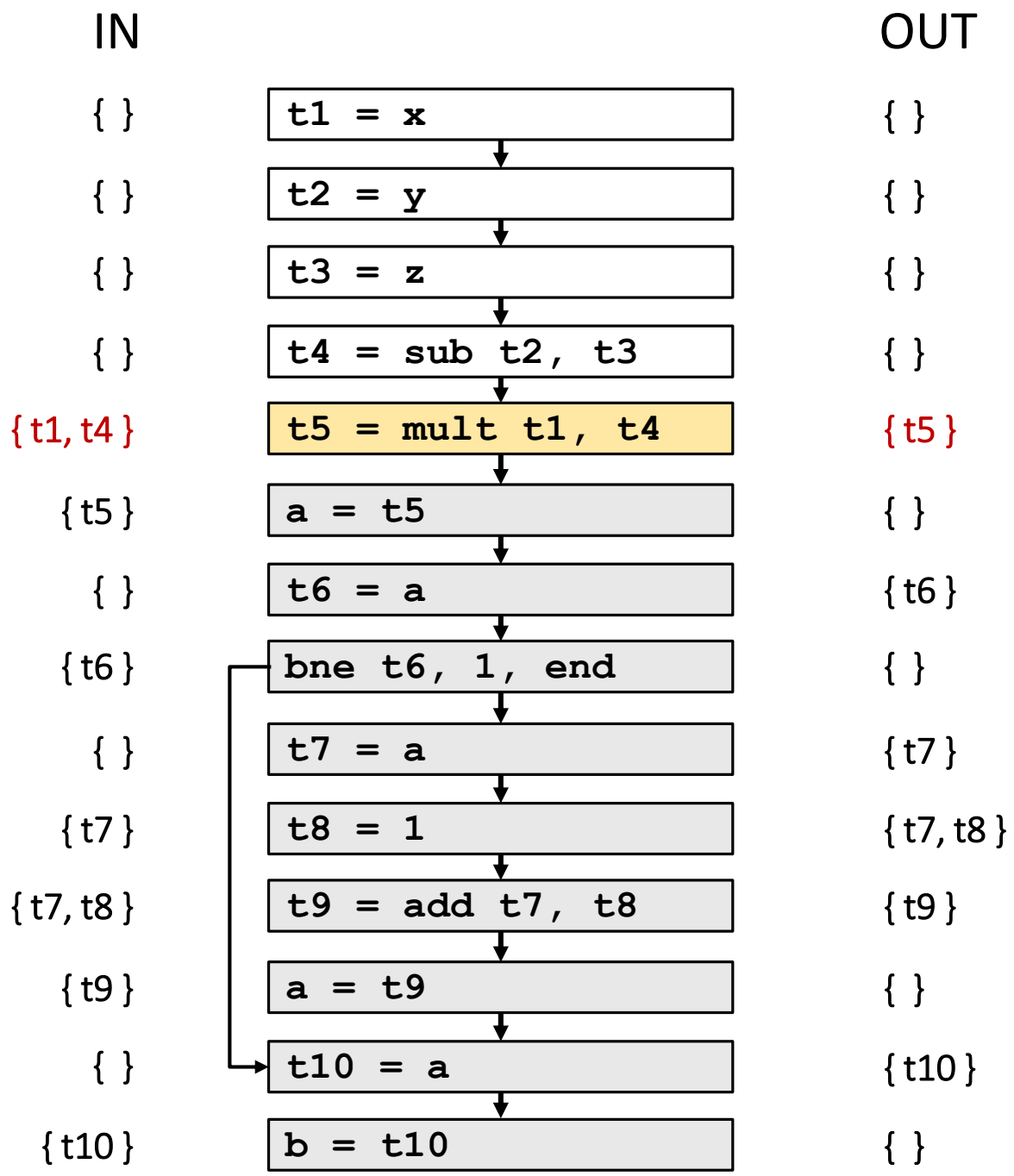

IN
OUT
{ }
t1 = x
{ }
{ }
t2 = y
{ }
{ }
t3 = z
{ }
{ }
t4 = sub t2, t3
{ }
{ }
t5 = mult t1, t4
{ }
{ }
a = t5
{ }
{ }
t6 = a
{ }
{ }
bne t6, 1, end
{ }
{ }
t7 = a
{ }
{t7}
t8 = 1
{t7, t8}
{t7, t8}
t9 = add t7, t8
{t9}
{t9}
a = t9
{ }
{ }
t10 = a
{t10}
{t10}
b = t10
{ }

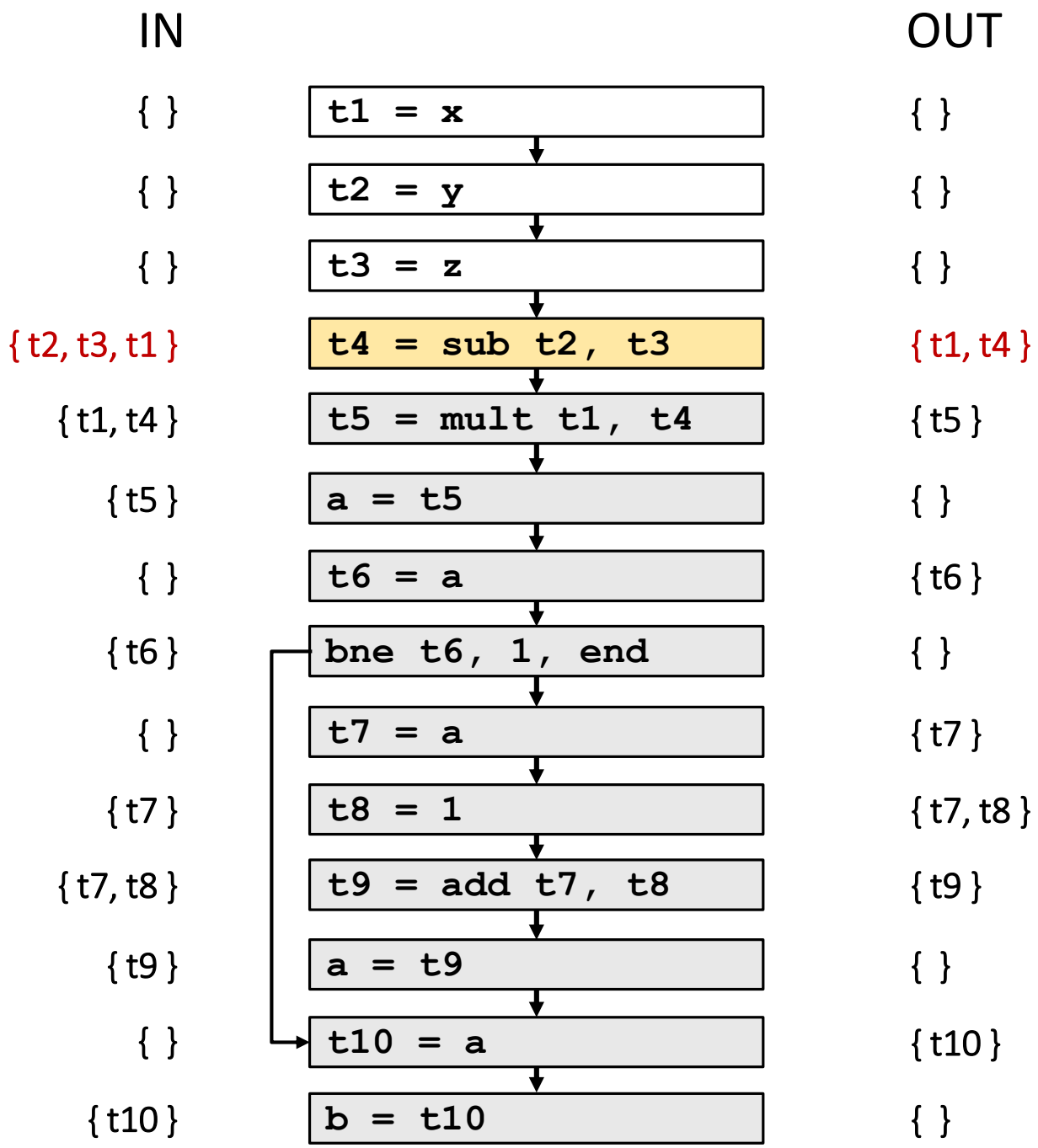

IN
OUT
{ }
t1 = x
{ }
{ }
t2 = y
{ }
{ }
t3 = z
{ }
{ }
t4 = sub t2, t3
{ }
{ }
t5 = mult t1, t4
{ }
{ }
a = t5
{ }
{ }
t6 = a
{ }
{ }
bne t6, 1, end
{ }
{ }
t7 = a
{t7}
{t7}
t8 = 1
{t7, t8}
{t7, t8}
t9 = add t7, t8
{t9}
{t9}
a = t9
{ }
{ }
t10 = a
{t10}
{t10}
b = t10
{ }

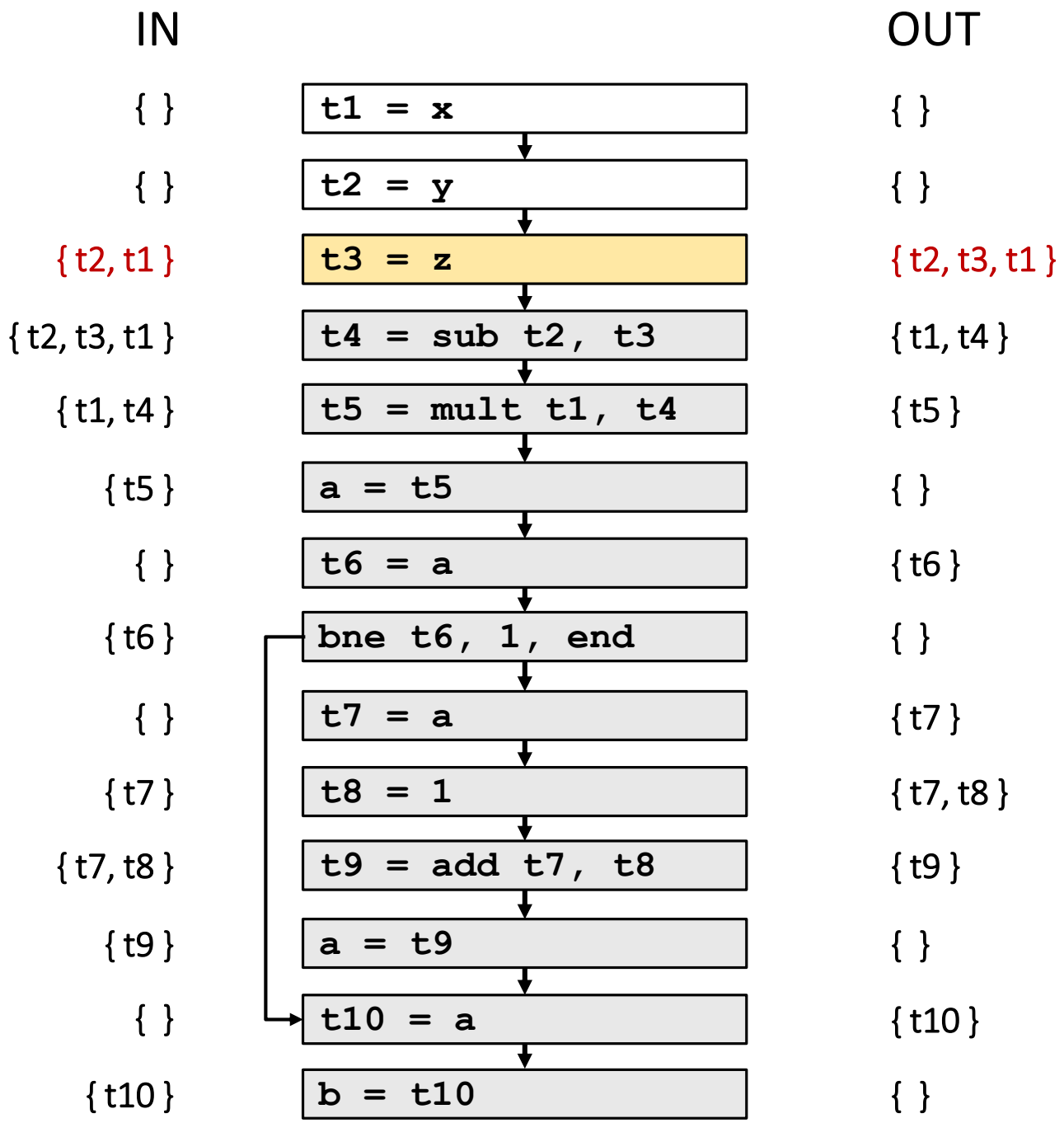

IN
OUT
{ }
t1 = x
{ }
{ }
t2 = y
{ }
{ }
t3 = z
{ }
{ }
t4 = sub t2, t3
{ }
{ }
t5 = mult t1, t4
{ }
{ }
a = t5
{ }
{ }
t6 = a
{ }
{t6}
bne t6, 1, end
{ }
{ }
t7 = a
{t7}
{t7}
t8 = 1
{t7, t8}
{t7, t8}
t9 = add t7, t8
{t9}
{t9}
a = t9
{ }
{ }
t10 = a
{t10}
{t10}
b = t10
{ }

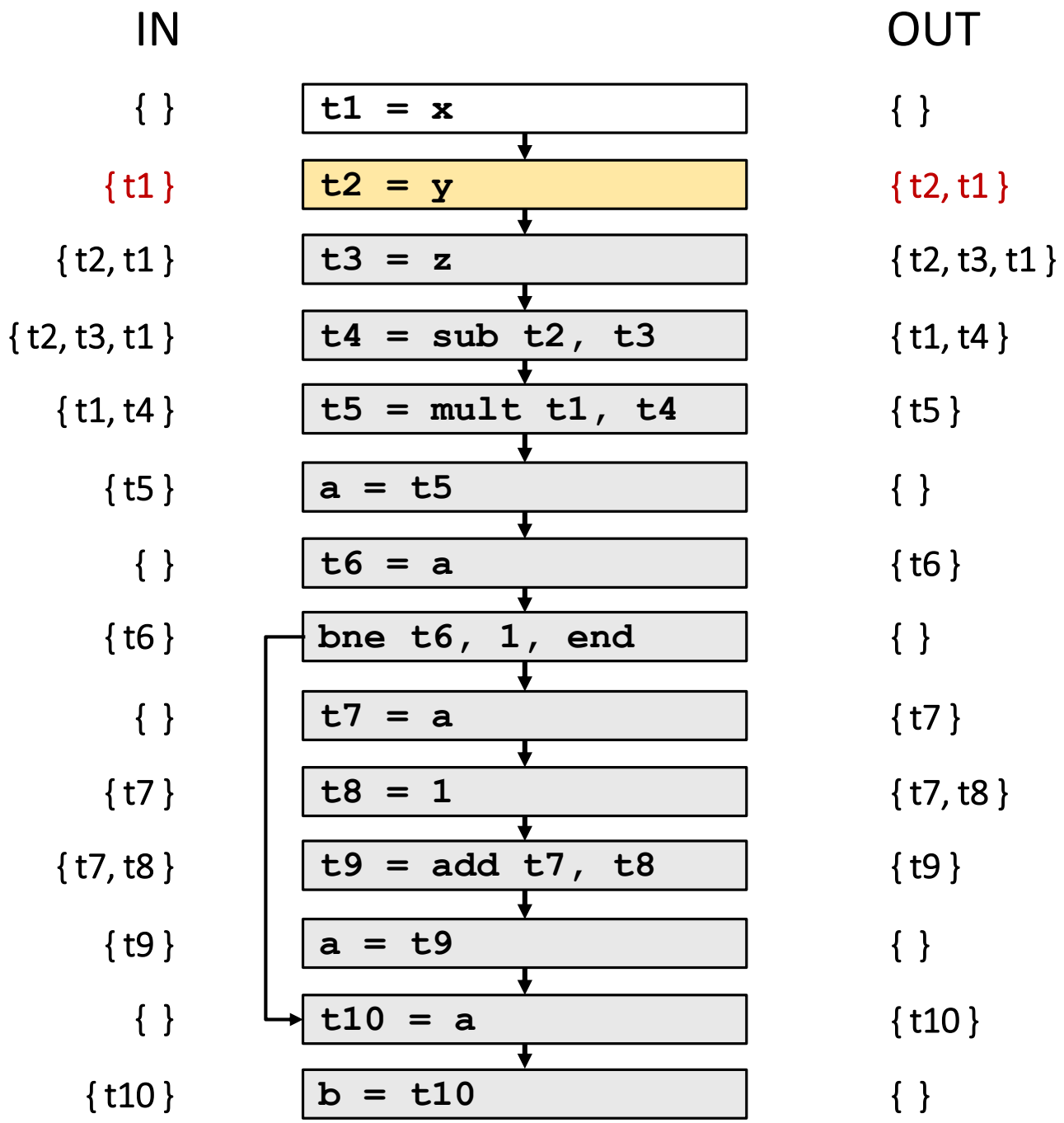

IN
OUT
{ }
t1 = x
{ }
{ }
t2 = y
{ }
{ }
t3 = z
{ }
{ }
t4 = sub t2, t3
{ }
{ }
t5 = mult t1, t4
{ }
{ }
a = t5
{ }
{ }
t6 = a
{t6}
{t6}
bne t6, 1, end
{ }
{ }
t7 = a
{t7}
{t7}
t8 = 1
{t7, t8}
{t7, t8}
t9 = add t7, t8
{t9}
{t9}
a = t9
{ }
{ }
t10 = a
{t10}
{t10}
b = t10
{ }

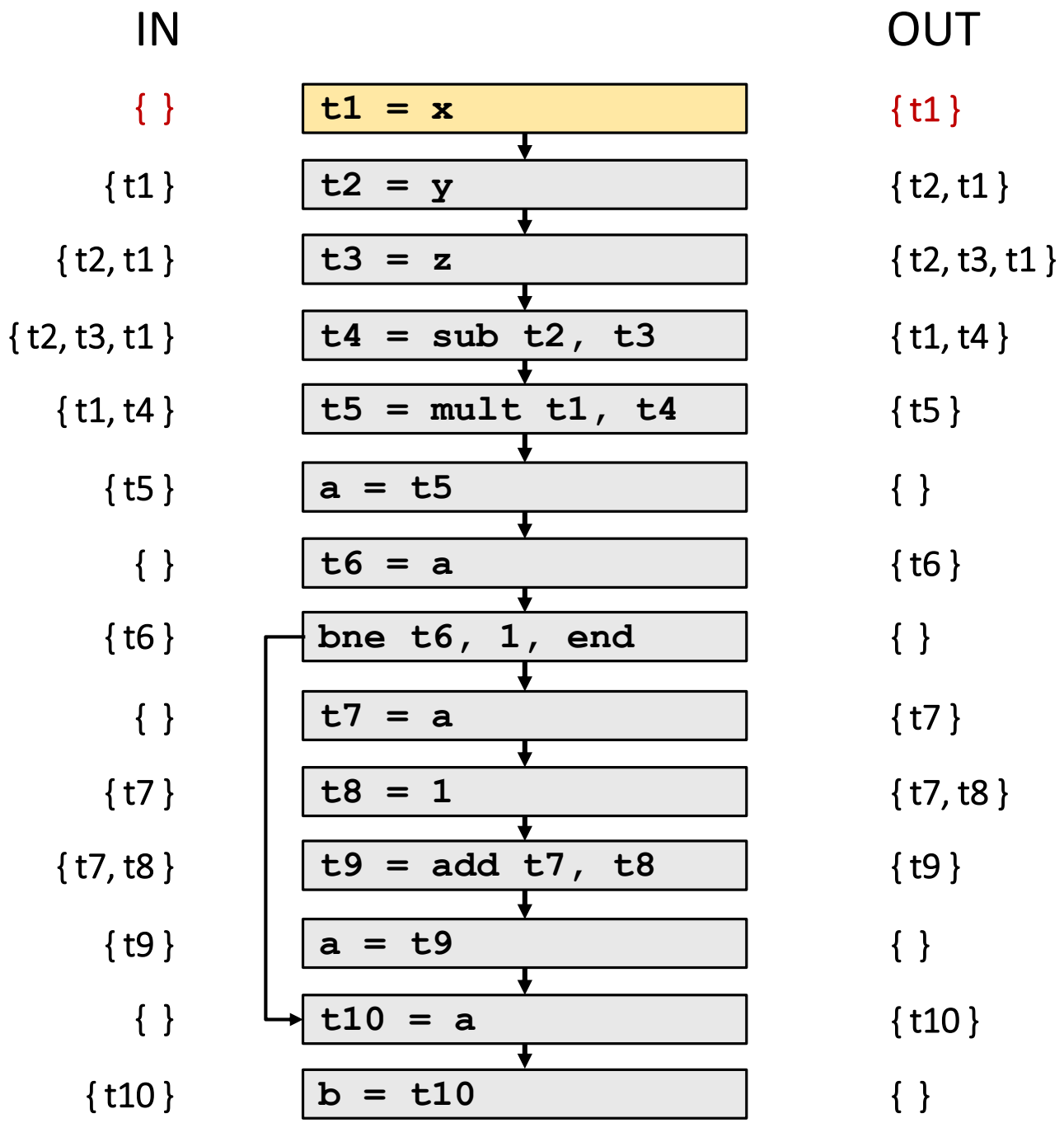

IN
OUT
{ }
t1 = x
{ }
{ }
t2 = y
{ }
{ }
t3 = z
{ }
{ }
t4 = sub t2, t3
{ }
{ }
t5 = mult t1, t4
{ }
{t5}
a = t5
{ }
{ }
t6 = a
{t6}
{t6}
bne t6, 1, end
{ }
{ }
t7 = a
{t7}
{t7}
t8 = 1
{t7, t8}
{t7, t8}
t9 = add t7, t8
{t9}
{t9}
a = t9
{ }
{ }
t10 = a
{t10}
{t10}
b = t10
{ }

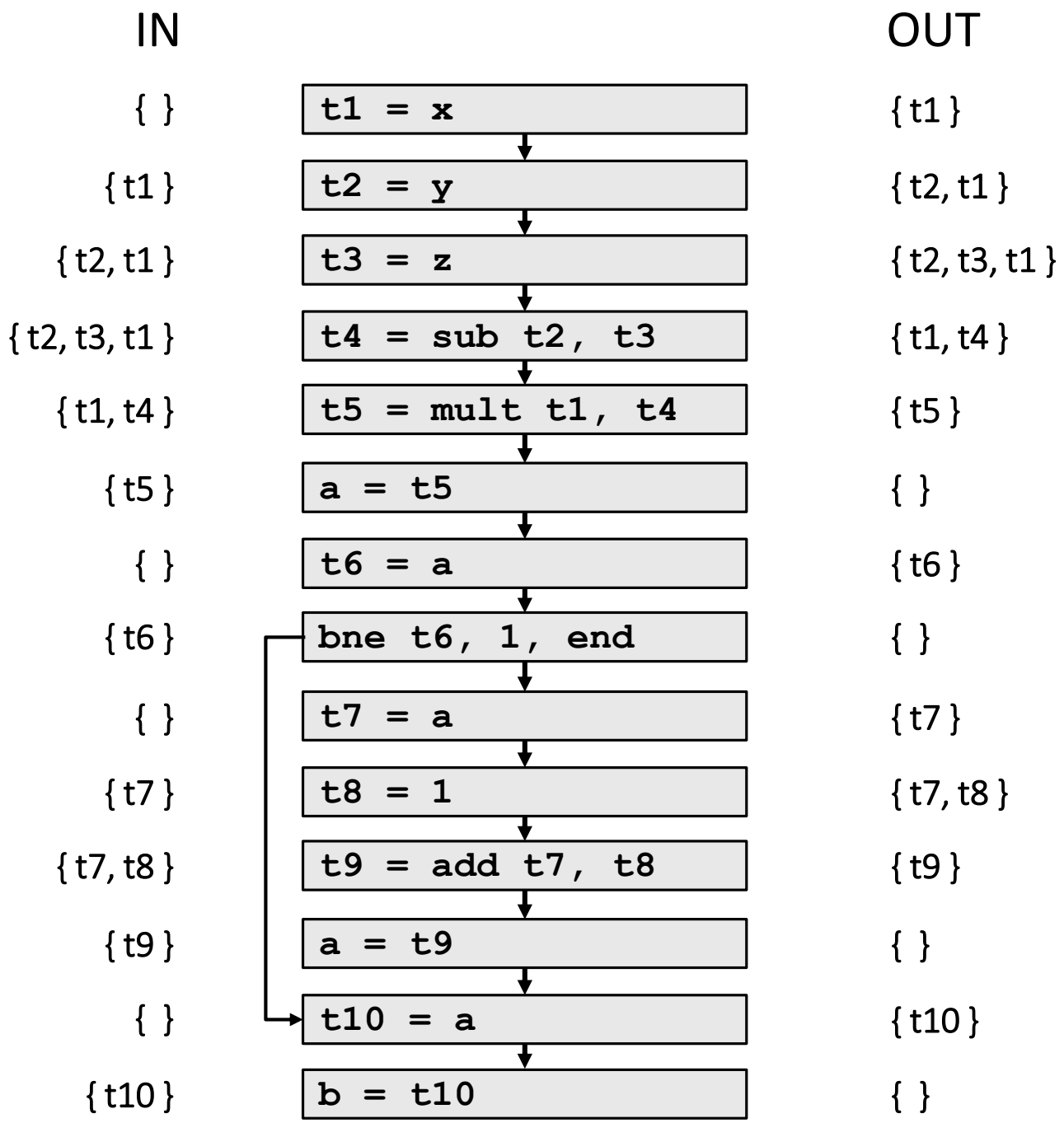

IN
OUT
{ }
t1 = x
{ }
{ }
t2 = y
{ }
{ }
t3 = z
{ }
{ }
t4 = sub t2, t3
{ }
{t1, t4}
t5 = mult t1, t4
{t5}
{t5}
a = t5
{ }
{ }
t6 = a
{t6}
{t6}
bne t6, 1, end
{ }
{ }
t7 = a
{t7}
{t7}
t8 = 1
{t7, t8}
{t7, t8}
t9 = add t7, t8
{t9}
{t9}
a = t9
{ }
{ }
t10 = a
{t10}
{t10}
b = t10
{ }


IN
OUT
{ }
t1 = x
{ }
{ }
t2 = y
{ }
{ }
t3 = z
{ }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }


IN
OUT
{ }
t1 = x
{ }
{ }
t2 = y
{ }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }

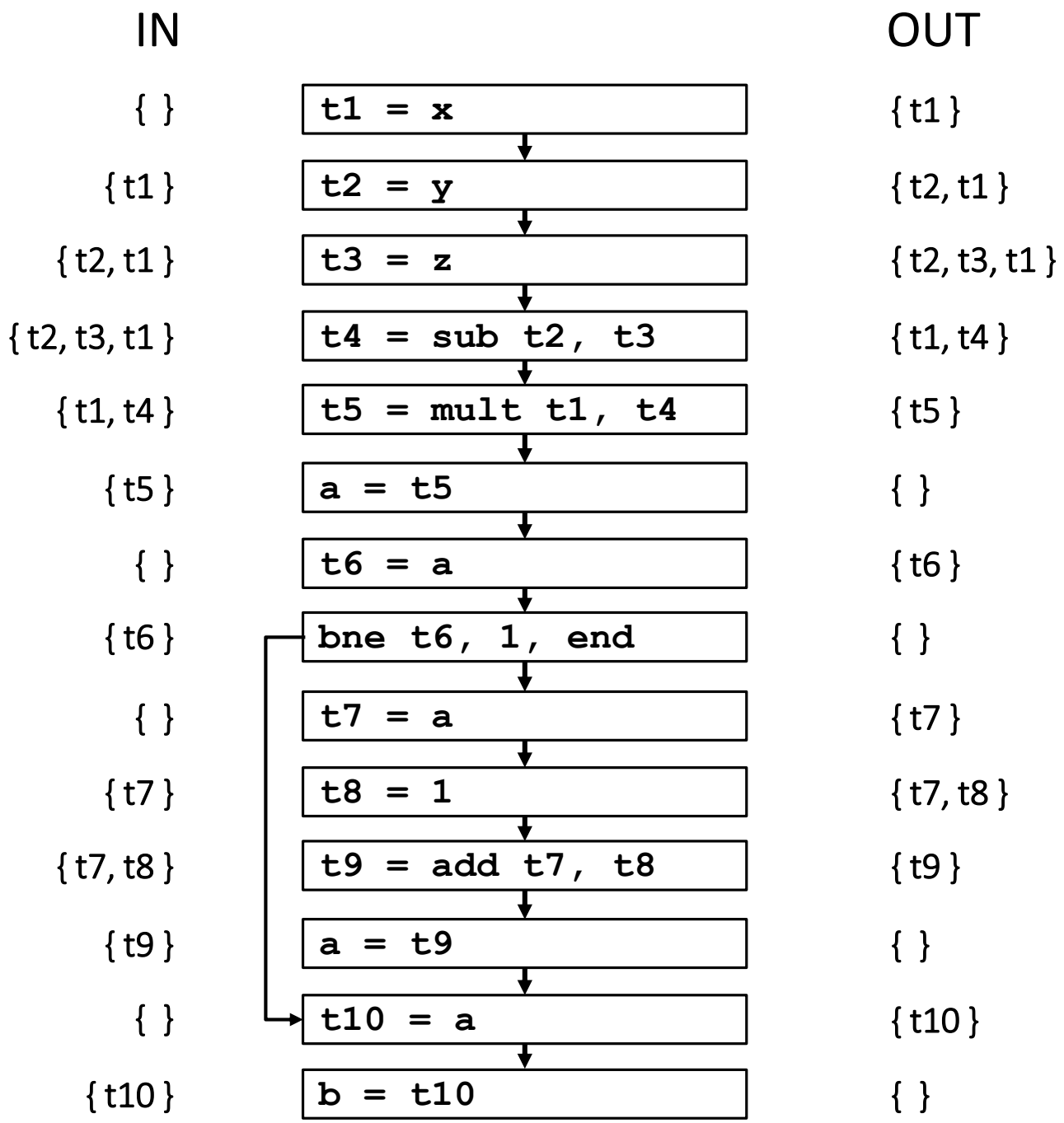

IN
OUT
{ }
t1 = x
{ }
{t1}
t2 = y
{t2, t1}
{t2, t1}
t3 = z
{t2, t3, t1}
{t2, t3, t1}
t4 = sub t2, t3
{t1, t4}
{t1, t4}
t5 = mult t1, t4
{t5}
{t5}
a = t5
{ }
{ }
t6 = a
{t6}
{t6}
bne t6, 1, end
{ }
{ }
t7 = a
{t7}
{t7}
t8 = 1
{t7, t8}
{t7, t8}
t9 = add t7, t8
{t9}
{t9}
a = t9
{ }
{ }
t10 = a
{t10}
{t10}
b = t10
{ }

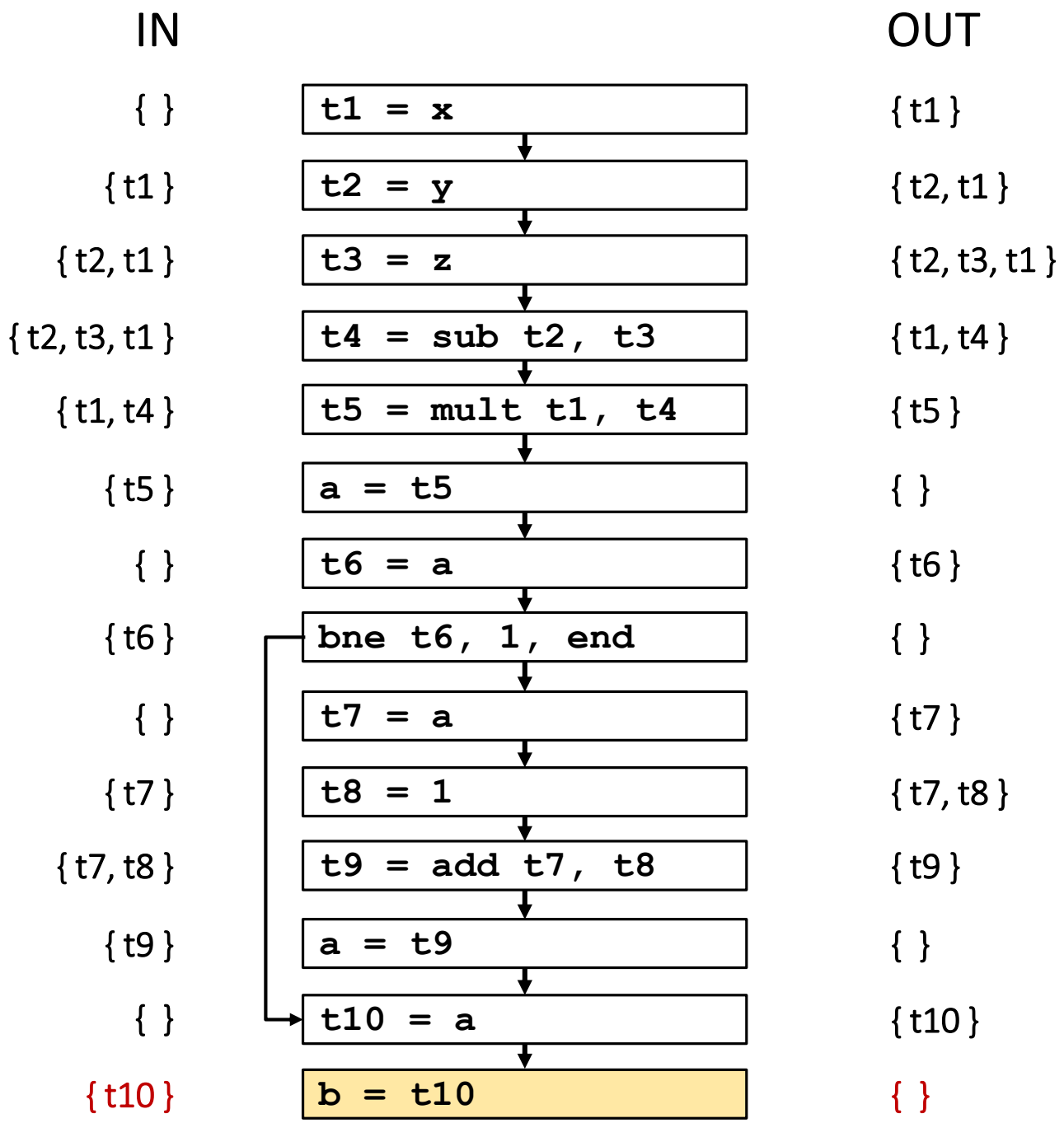

IN
OUT
{ }
t1 = x
{ t1 }
{ t1 }
t2 = y
{ t2, t1 }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }

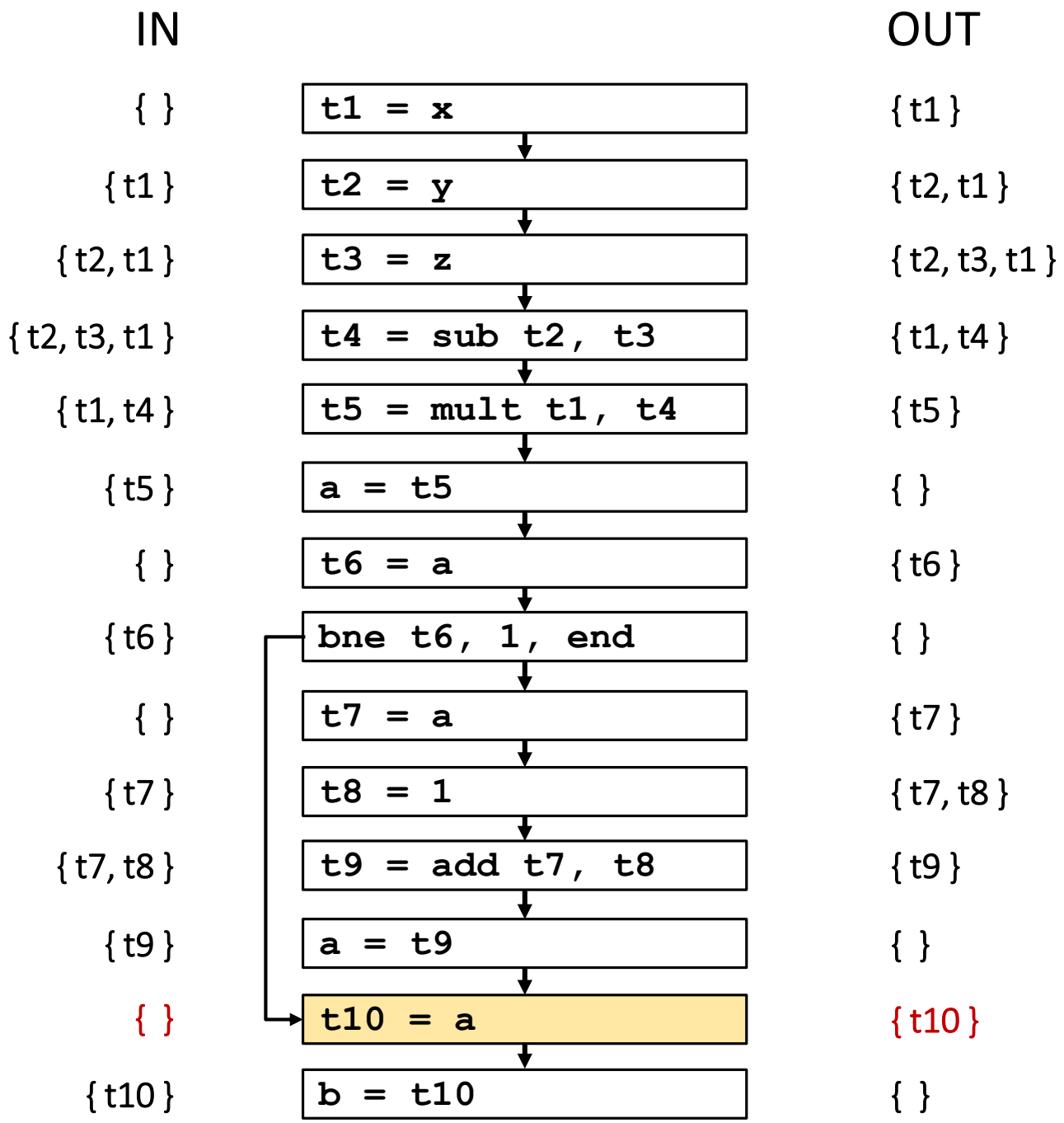

IN
OUT
{ }
t1 = x
{ t1 }
{ t1 }
t2 = y
{ t2, t1 }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }

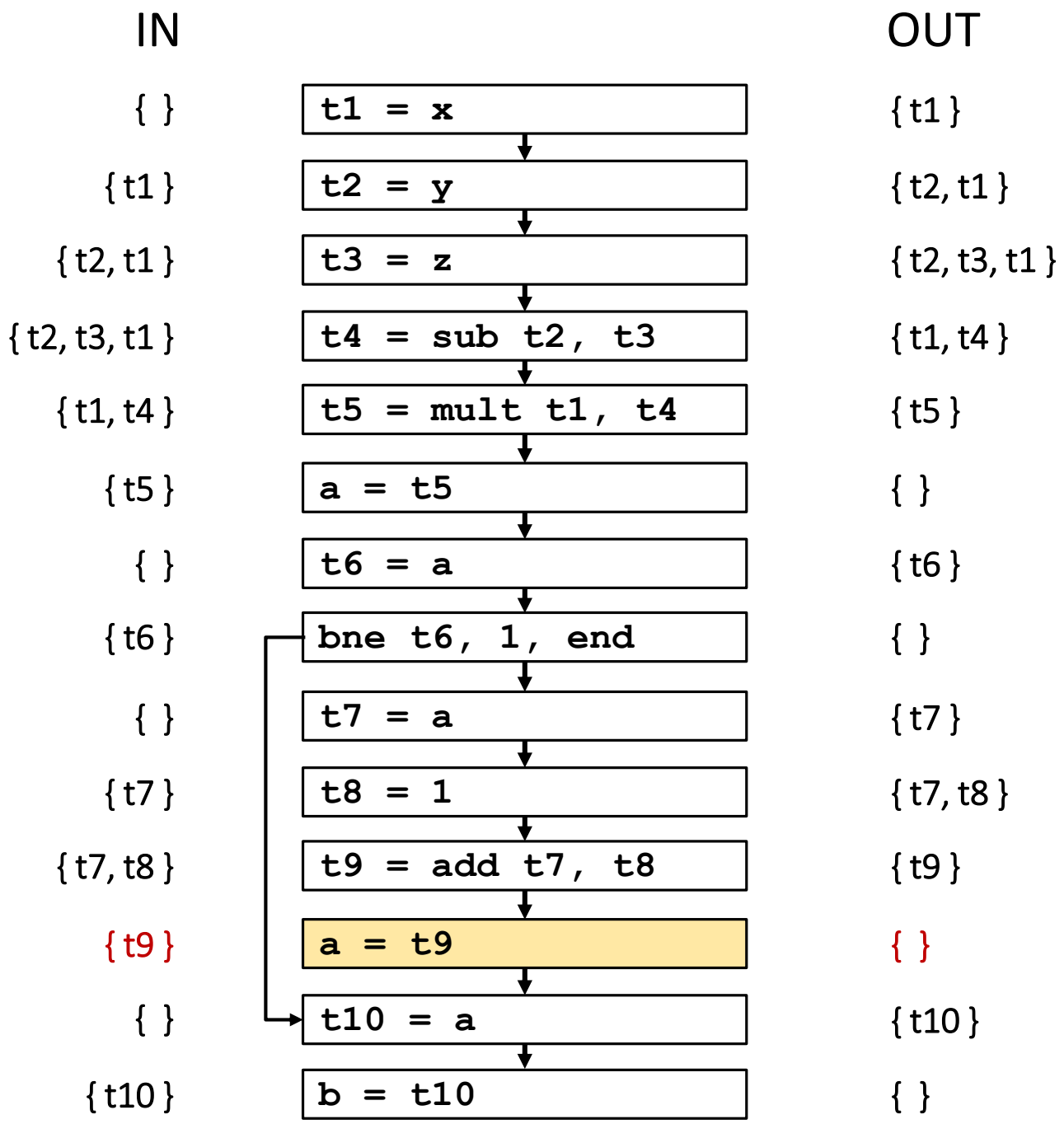
Liveness Analysis

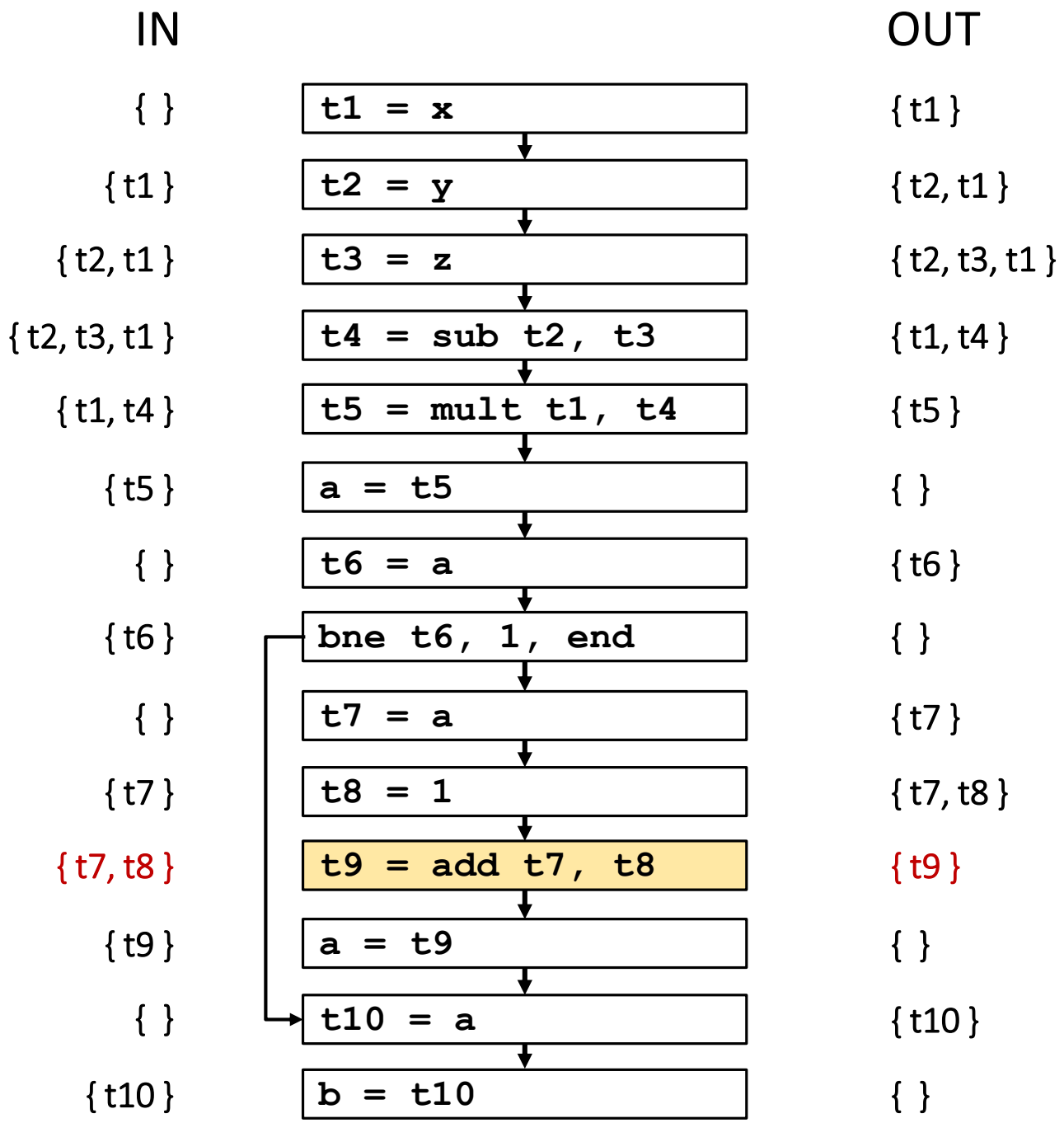
Second Iteration…

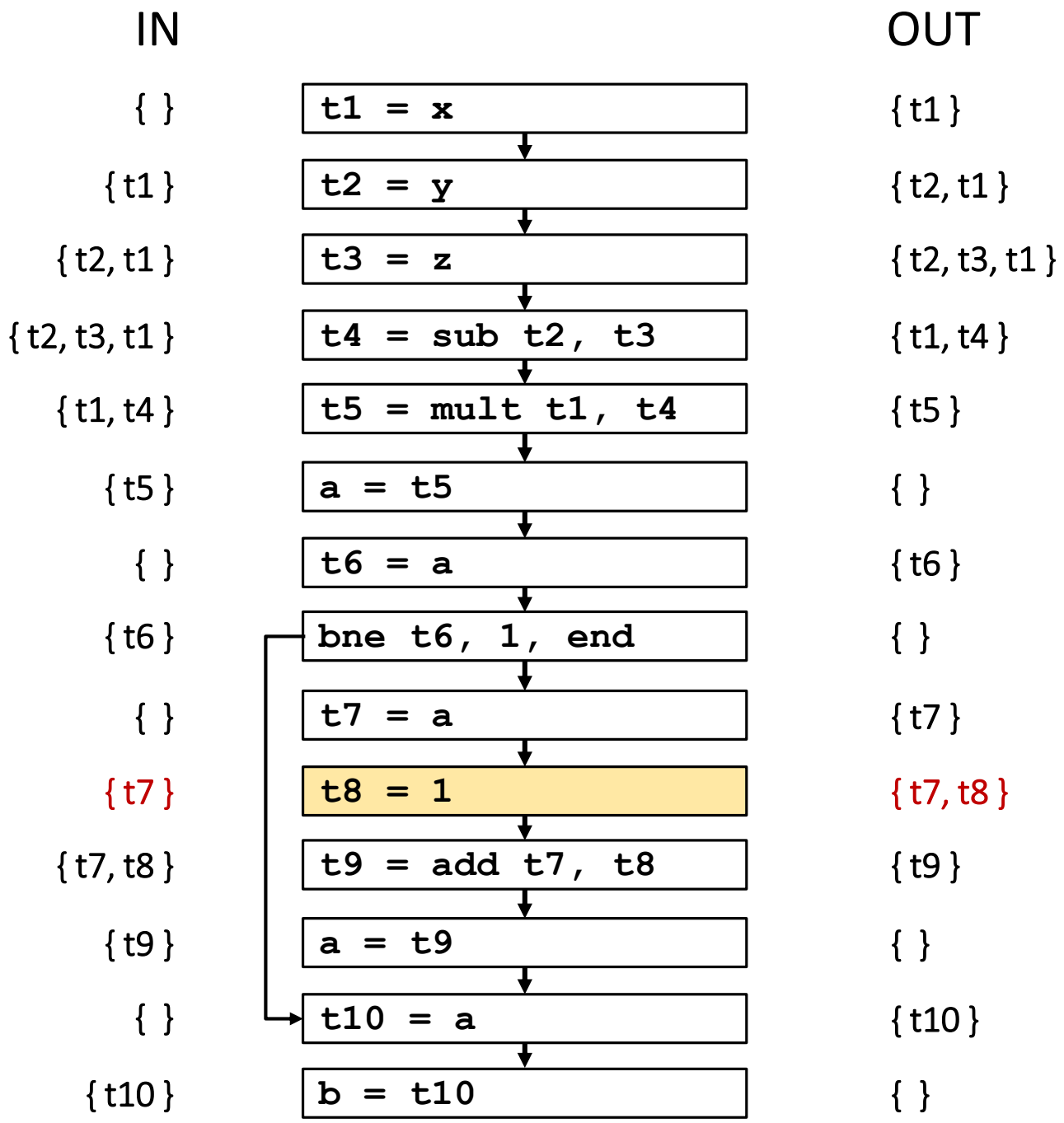

IN
OUT
{ }
t1 = x
{ t1 }
{ t1 }
t2 = y
{ t2, t1 }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }

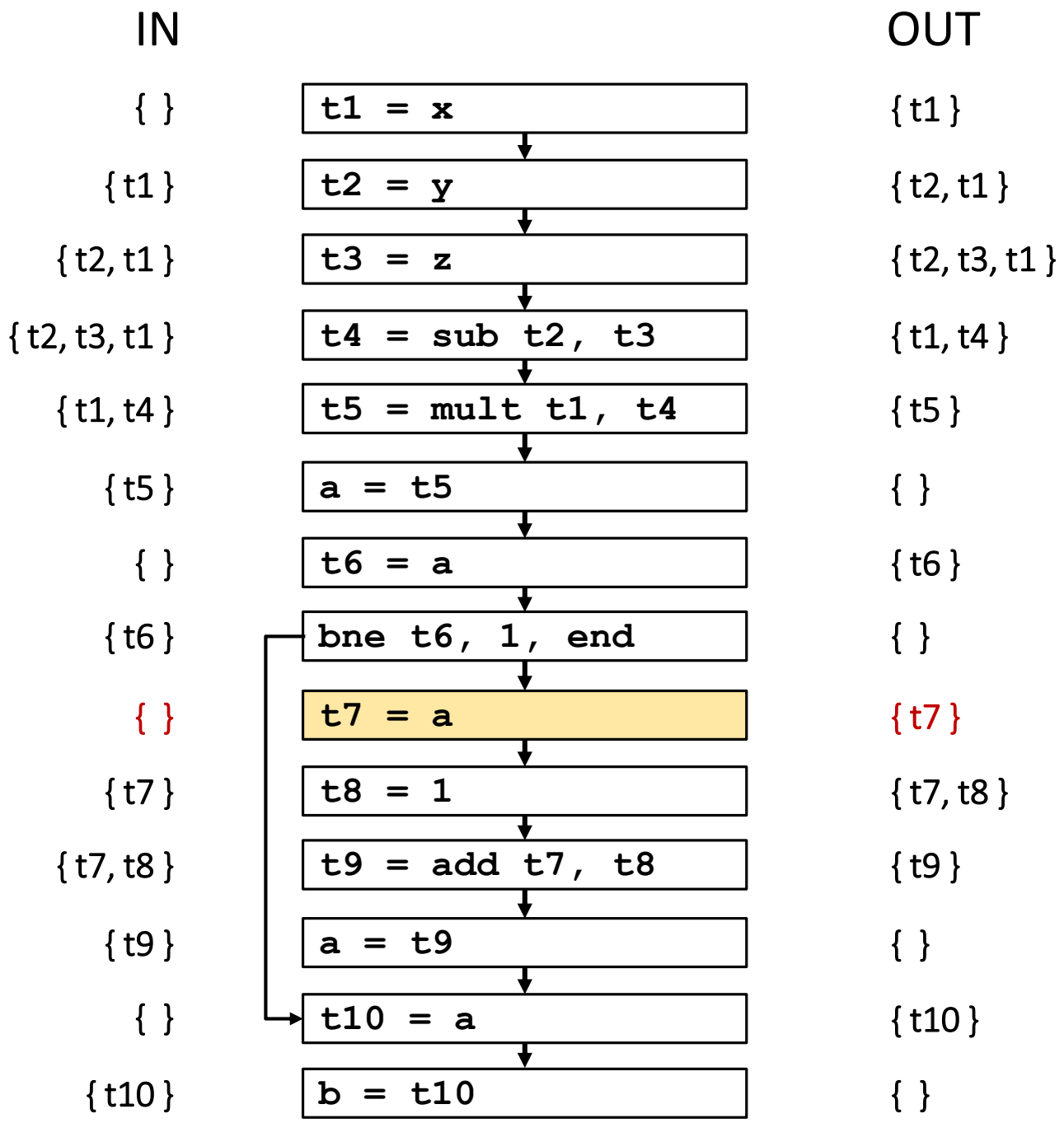

IN
OUT
{ }
t1 = x
{ t1 }
{ t1 }
t2 = y
{ t2, t1 }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }

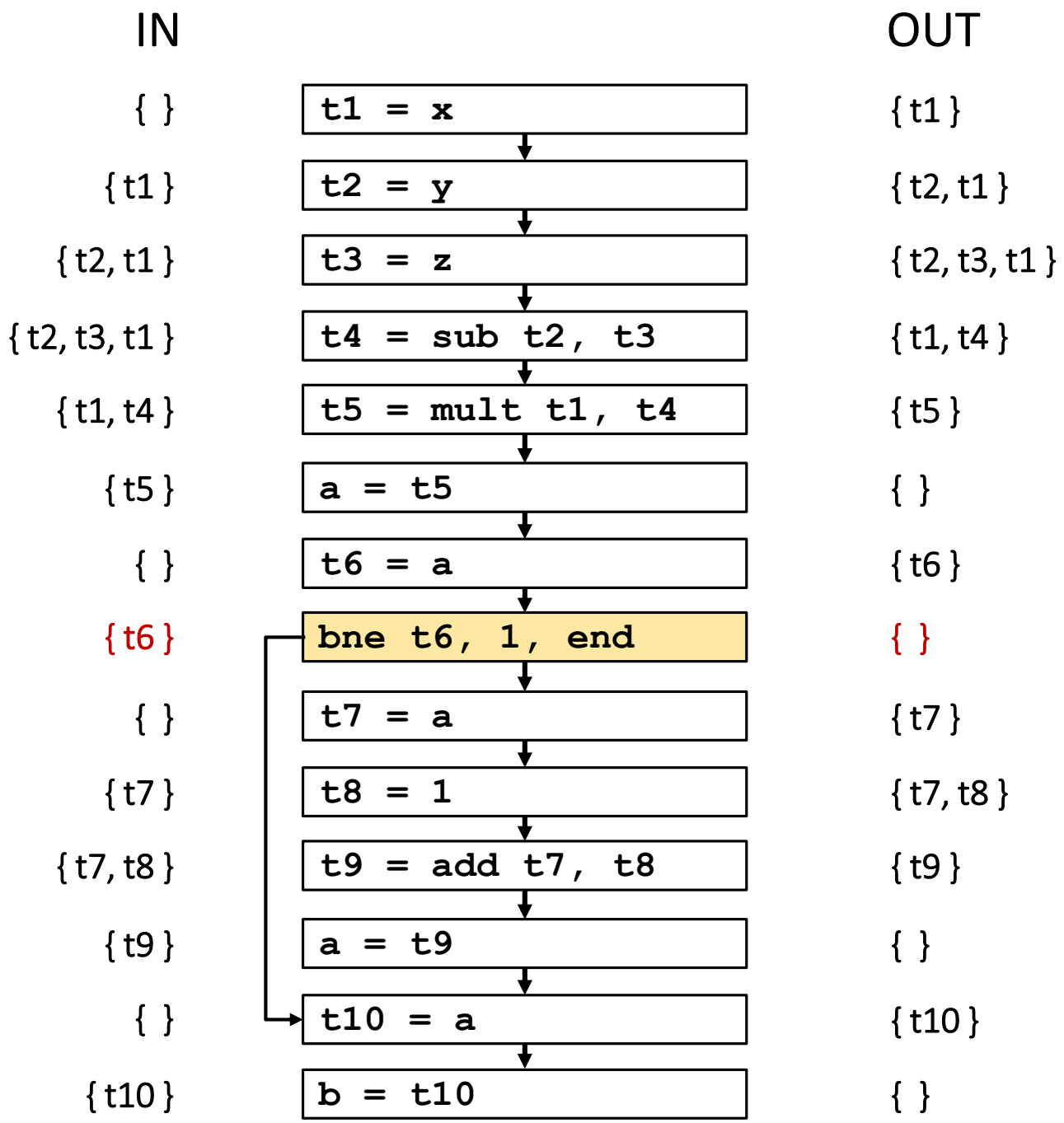

IN
OUT
{ }
t1 = x
{ t1 }
{ t1 }
t2 = y
{ t2, t1 }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }

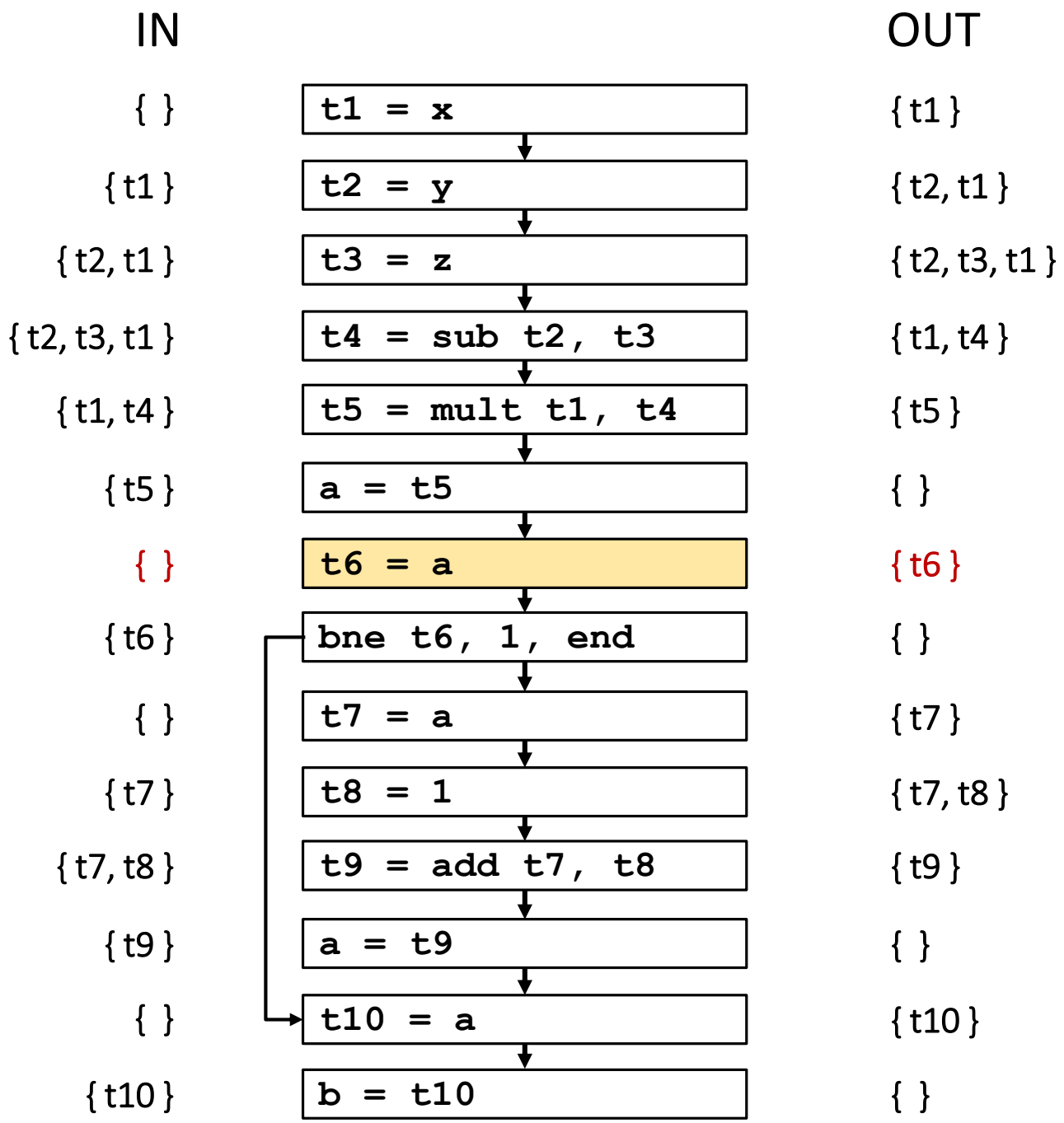

IN
OUT
{ }
t1 = x
{ t1 }
{ t1 }
t2 = y
{ t2, t1 }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }

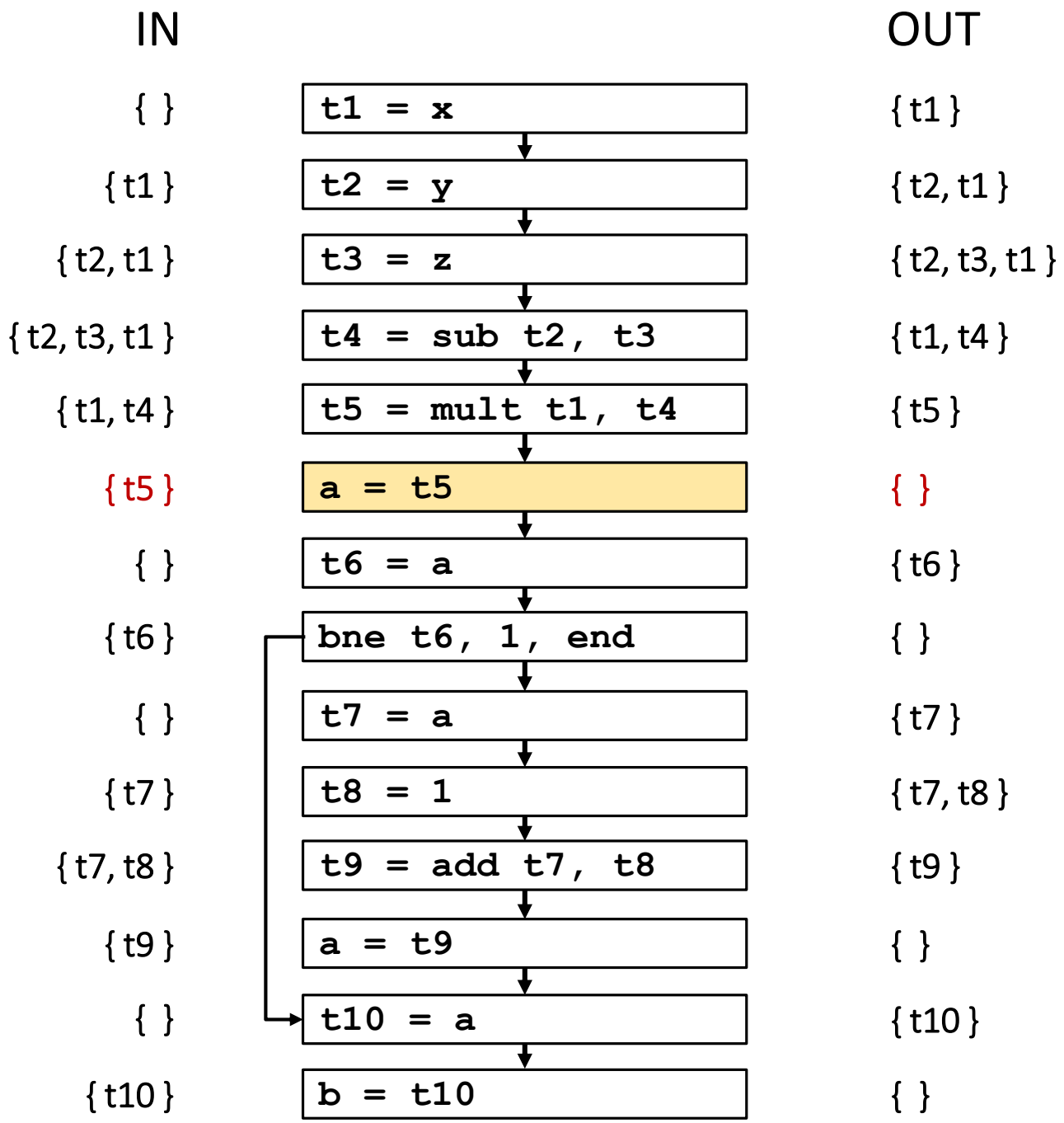

IN
OUT
{ }
t1 = x
{ t1 }
{ t1 }
t2 = y
{ t2, t1 }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }

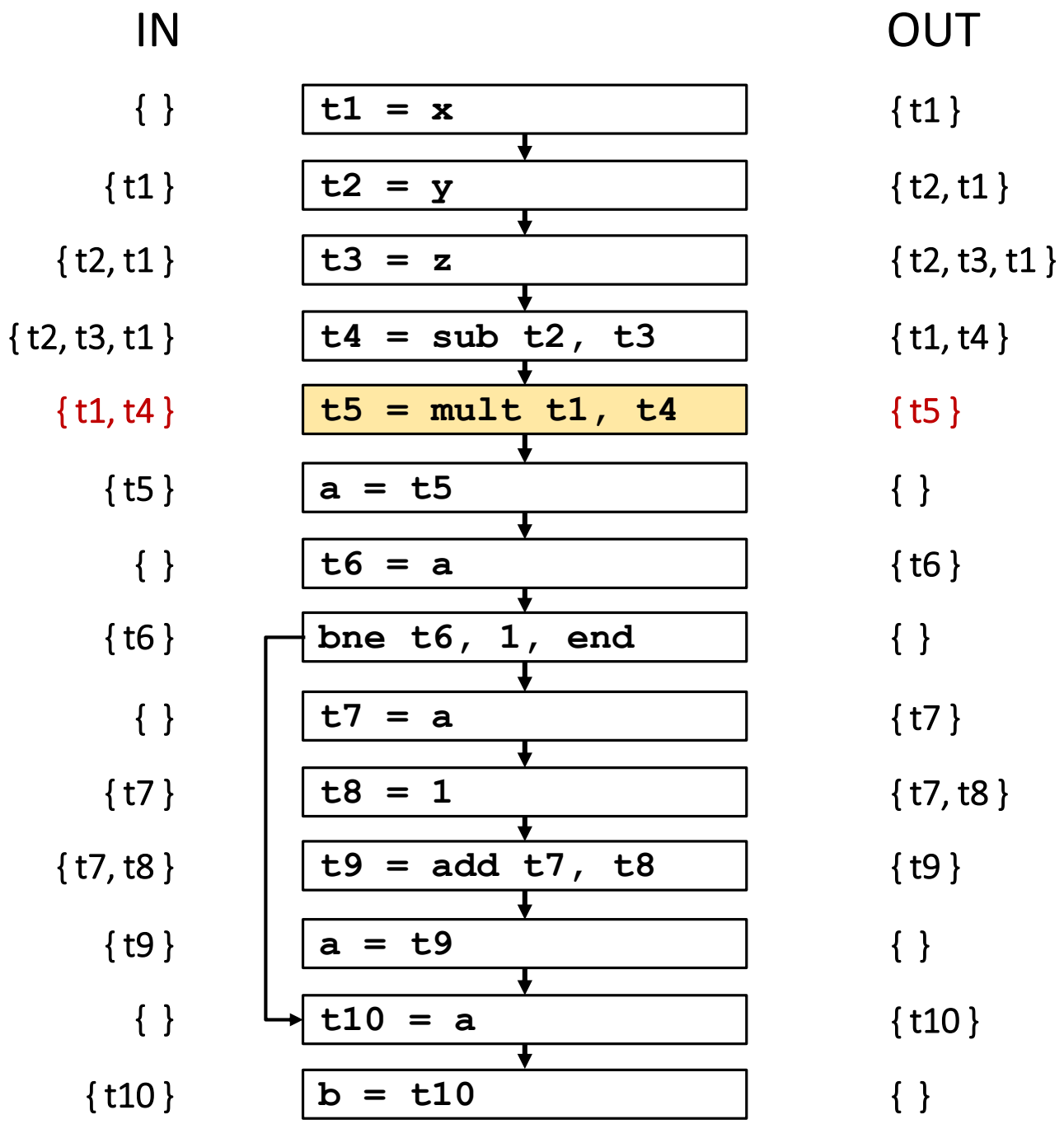

IN
OUT
{ }
t1 = x
{t1}
{t1}
t2 = y
{t2, t1}
{t2, t1}
t3 = z
{t2, t3, t1}
{t2, t3, t1}
t4 = sub t2, t3
{t1, t4}
{t1, t4}
t5 = mult t1, t4
{t5}
{t5}
a = t5
{ }
{ }
t6 = a
{t6}
{t6}
bne t6, 1, end
{ }
{ }
t7 = a
{t7}
{t7}
t8 = 1
{t7, t8}
{t7, t8}
t9 = add t7, t8
{t9}
{t9}
a = t9
{ }
{ }
t10 = a
{t10}
{t10}
b = t10
{ }

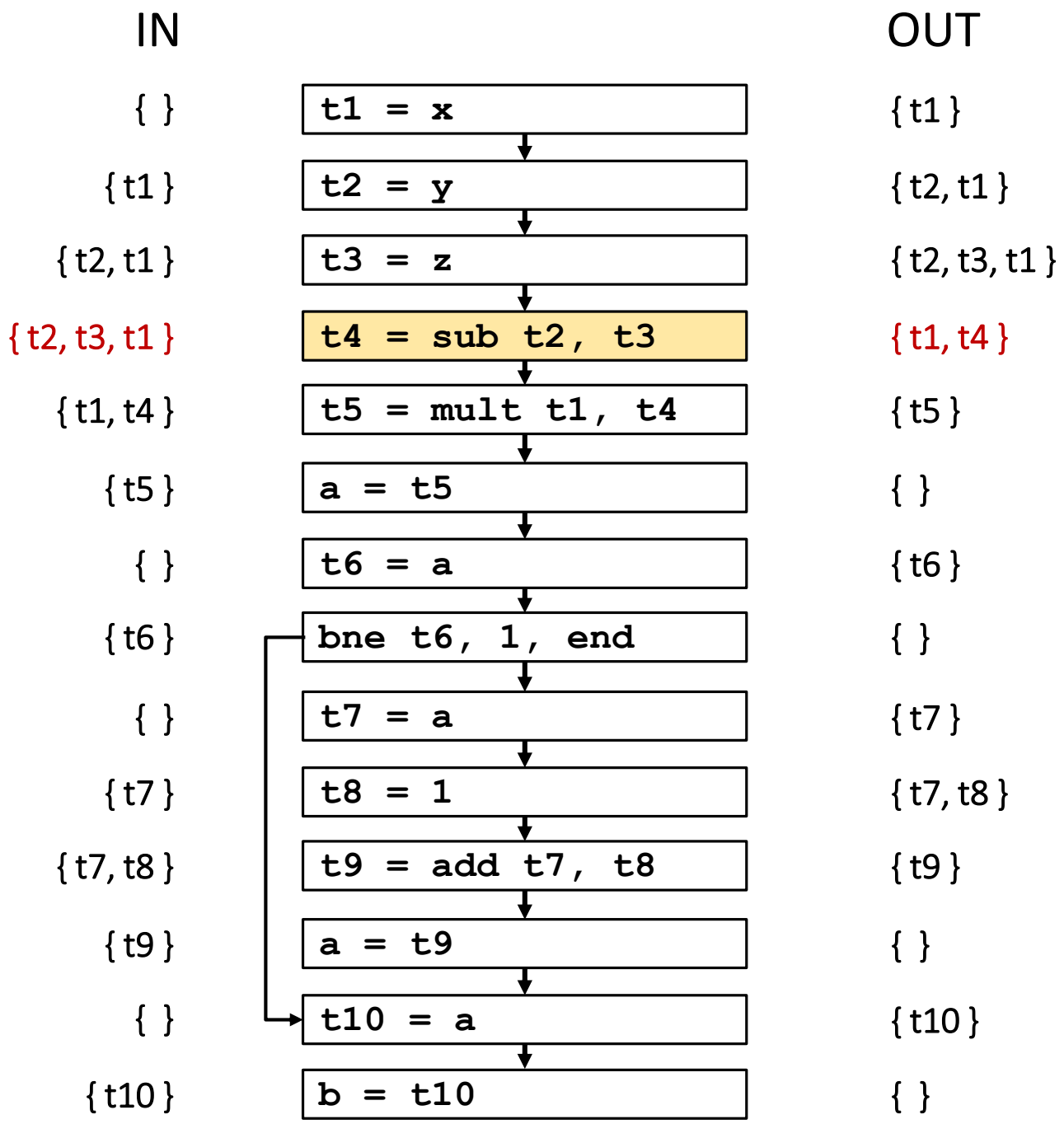

IN
OUT
{ }
t1 = x
{ t1 }
{ t1 }
t2 = y
{ t2, t1 }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }

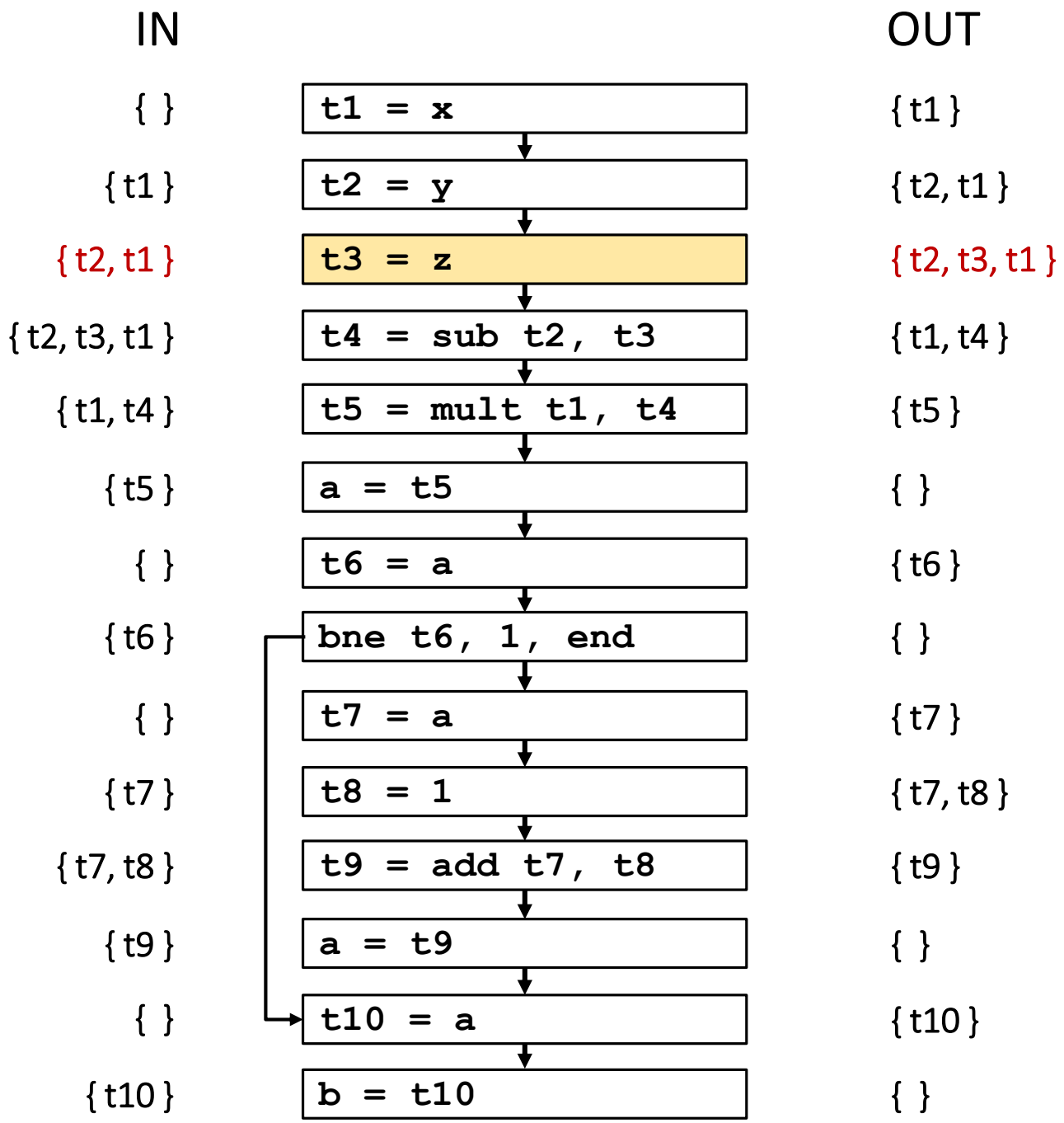

IN
OUT
{ }
t1 = x
{ t1 }
{ t1 }
t2 = y
{ t2, t1 }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }

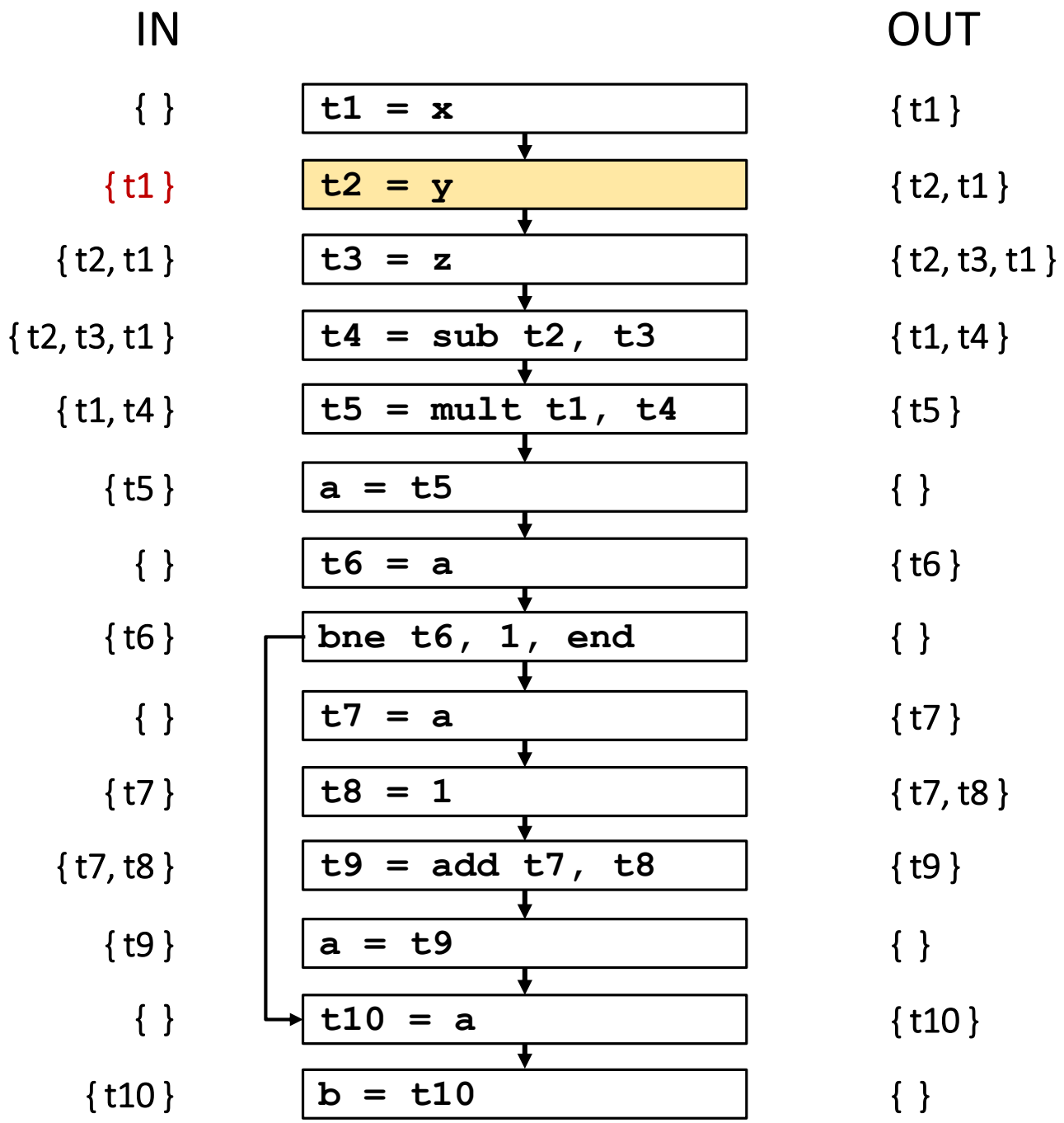

IN
OUT
{ }
t1 = x
{ t1 }
{ t1 }
t2 = y
{ t2, t1 }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }

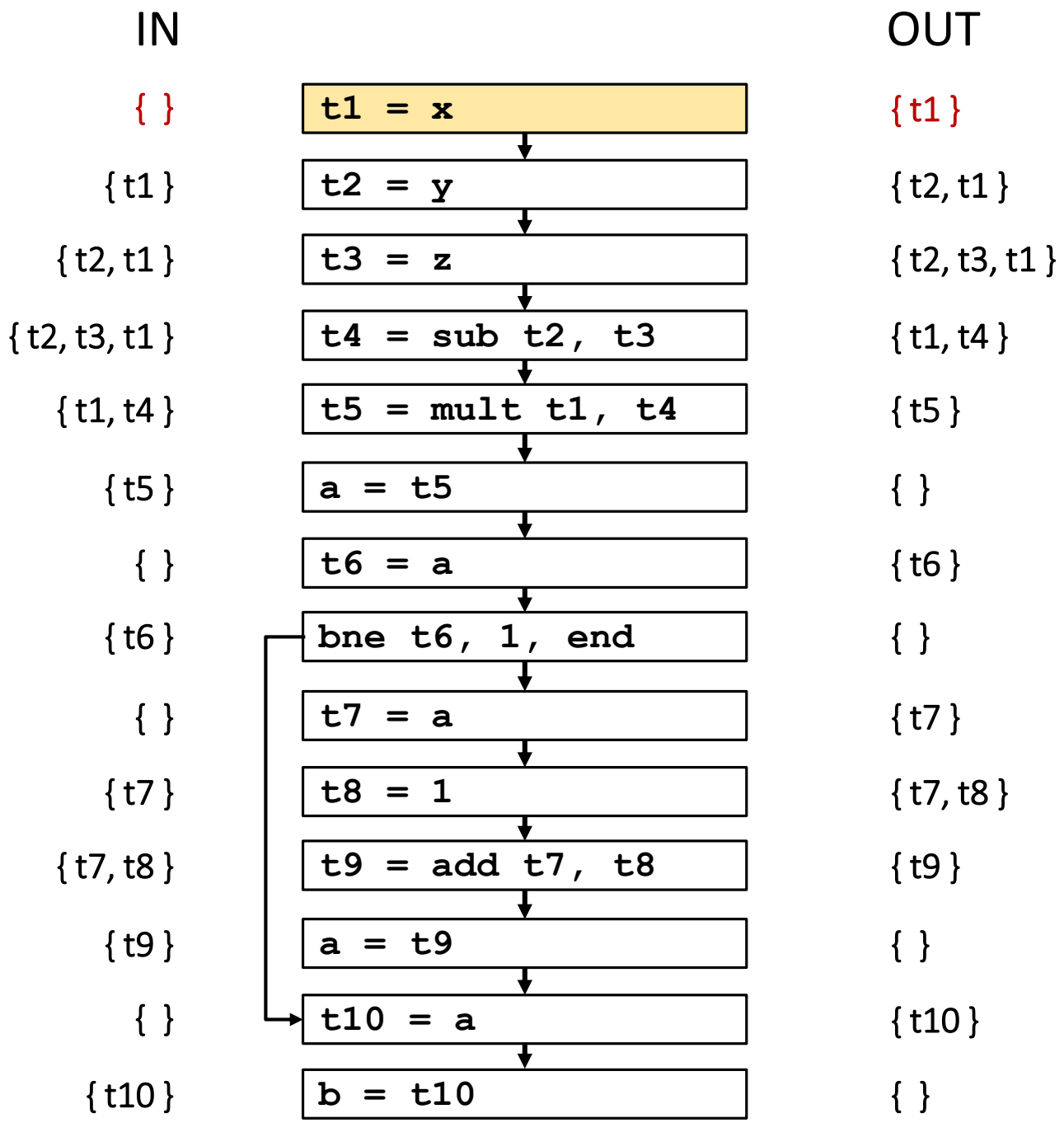

IN
OUT
{ }
t1 = x
{ t1 }
{ t1 }
t2 = y
{ t2, t1 }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }

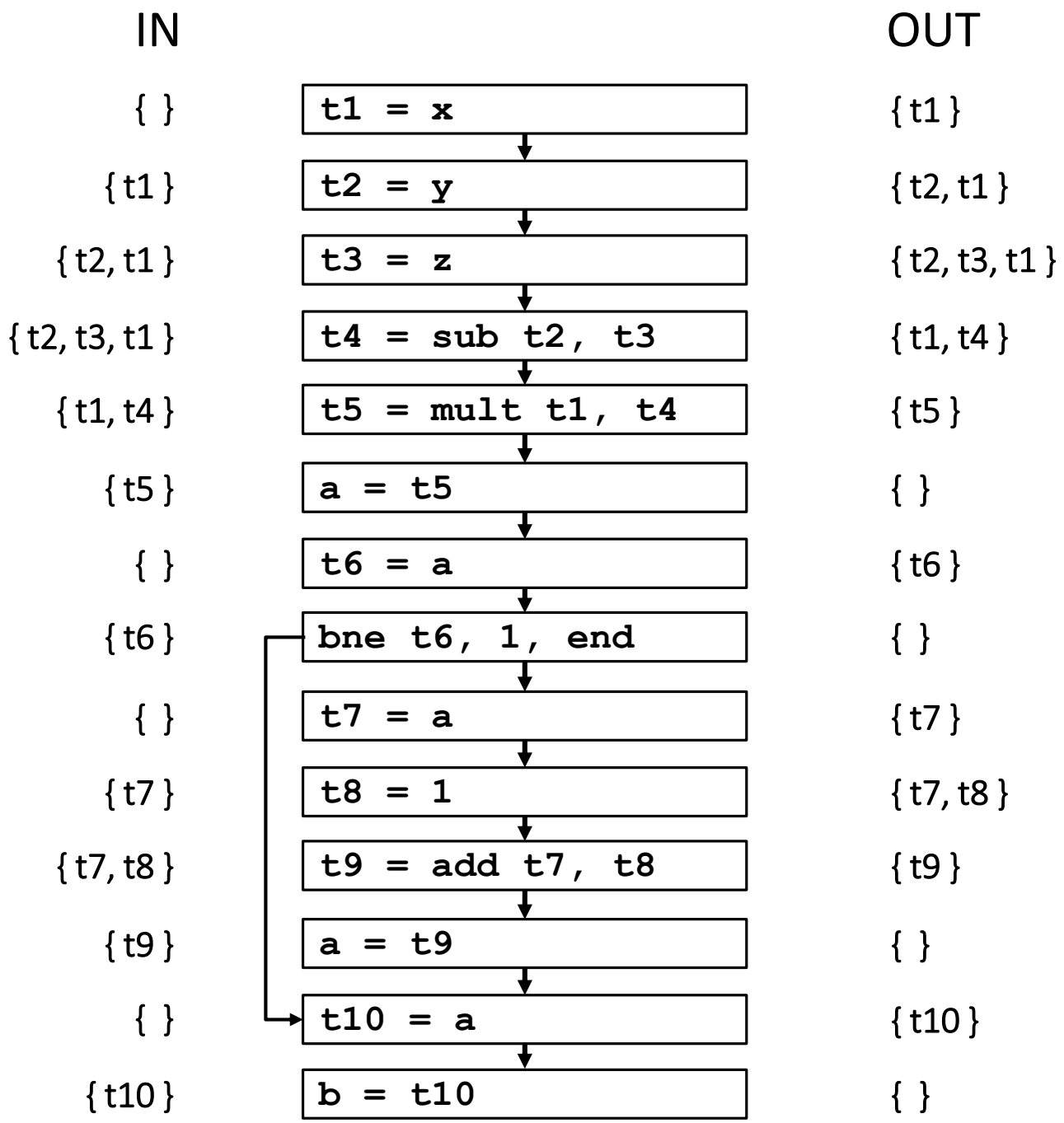

IN
OUT
{ }
t1 = x
{ t1 }
{ t1 }
t2 = y
{ t2, t1 }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }


IN
OUT
{ }
t1 = x
{ t1 }
{ t1 }
t2 = y
{ t2, t1 }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }


IN
OUT
{ }
t1 = x
{ t1 }
{t1}
t2 = y
{ t2, t1 }
{t2, t1}
t3 = z
{ t2, t3, t1 }
{t2, t3, t1}
t4 = sub t2, t3
{ t1, t4 }
{t1, t4}
t5 = mult t1, t4
{ t5 }
{t5}
a = t5
{ }
{ }
t6 = a
{ t6 }
{t6}
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{t7}
t8 = 1
{ t7, t8 }
{t7, t8}
t9 = add t7, t8
{ t9 }
{t9}
a = t9
{ }
{ }
t10 = a
{ t10 }
{t10}
b = t10
{ }


IN
OUT
{ }
t1 = x
{ t1 }
{t1}
t2 = y
{ t2, t1 }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }


IN
OUT
{ }
t1 = x
{t1}
{t1}
t2 = y
{t2, t1}
{t2, t1}
t3 = z
{t2, t3, t1}
{t2, t3, t1}
t4 = sub t2, t3
{t1, t4}
{t1, t4}
t5 = mult t1, t4
{t5}
{t5}
a = t5
{ }
{ }
t6 = a
{t6}
{t6}
bne t6, 1, end
{ }
{ }
t7 = a
{t7}
{t7}
t8 = 1
{t7, t8}
{t7, t8}
t9 = add t7, t8
{t9}
{t9}
a = t9
{ }
{ }
t10 = a
{t10}
{t10}
b = t10
{ }

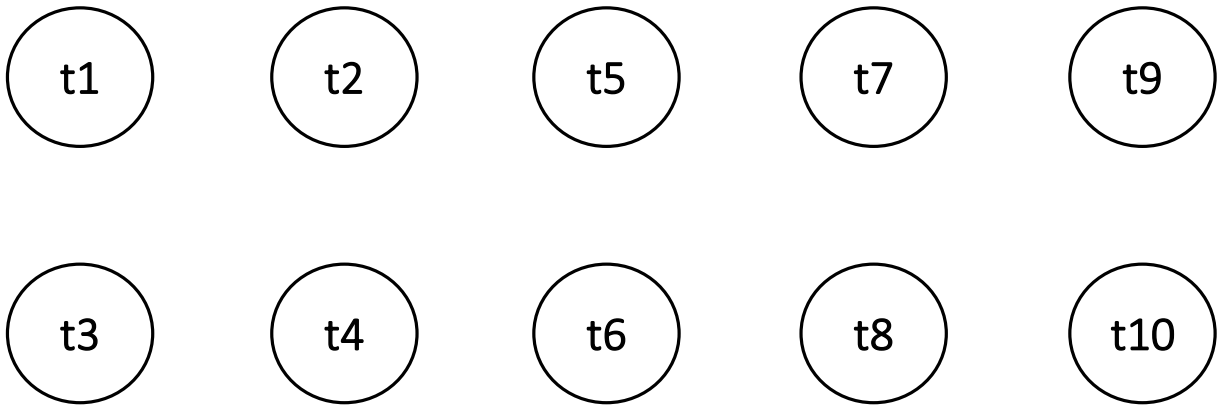

IN
OUT
{ }
t1 = x
{ t1 }
{ t1 }
t2 = y
{ t2, t1 }
{ t2, t1 }
t3 = z
{ t2, t3, t1 }
{ t2, t3, t1 }
t4 = sub t2, t3
{ t1, t4 }
{ t1, t4 }
t5 = mult t1, t4
{ t5 }
{ t5 }
a = t5
{ }
{ }
t6 = a
{ t6 }
{ t6 }
bne t6, 1, end
{ }
{ }
t7 = a
{ t7 }
{ t7 }
t8 = 1
{ t7, t8 }
{ t7, t8 }
t9 = add t7, t8
{ t9 }
{ t9 }
a = t9
{ }
{ }
t10 = a
{ t10 }
{ t10 }
b = t10
{ }

Interference Graph

- Use liveness analysis to construct the **interference graph**
- Create a node for each IR variable (*t1*, *t2*, …)
- If *t1* and *t2* appear together in one of the liveness sets:
  - Create an edge between *t1* and *t2*

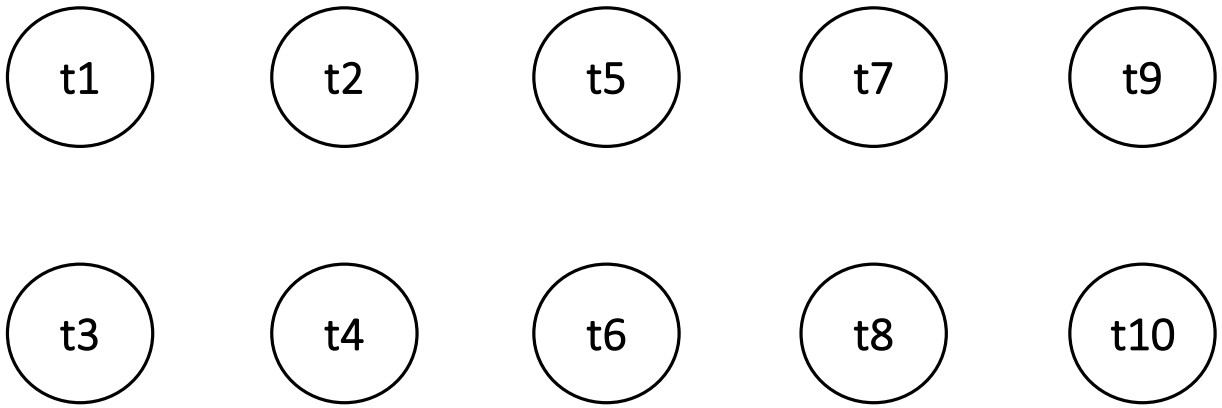
Interference Graph

{ t1 }
{ t2, t1 }
{ t2, t3, t1 }
{ t1, t4 }
{ t5 }
{ t6 }
{ t7 }
{ t7, t8 }
{ t9 }
{ t10 }



Interference Graph

{ t1 }
{ t2, t1 }
{ t2, t3, t1 }
{ t1, t4 }
{ t5 }
{ t6 }
{ t7 }
{ t7, t8 }
{ t9 }
{ t10 }



Interference Graph

{ t1 }

{ t2, t1 }

{ t2, t3, t1 }

{ t1, t4 }

{ t5 }

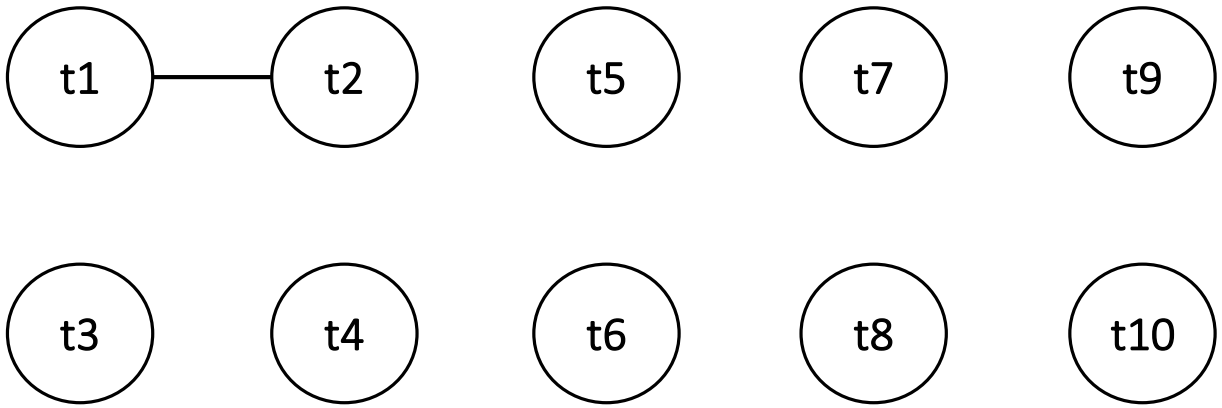
{ t6 }

{ t7 }

{ t7, t8 }

{ t9 }

{ t10 }



Interference Graph

{ t1 }

{ t2, t1 }

{ t2, t3, t1 }

{ t1, t4 }

{ t5 }

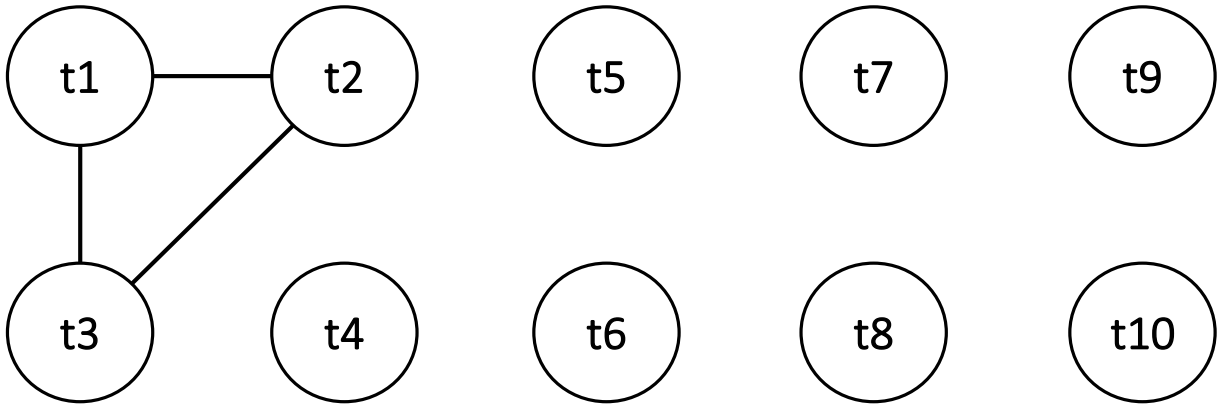
{ t6 }

{ t7 }

{ t7, t8 }

{ t9 }

{ t10 }



Interference Graph

{ t1 }

{ t2, t1 }

{ t2, t3, t1 }

{ t1, t4 }

{ t5 }

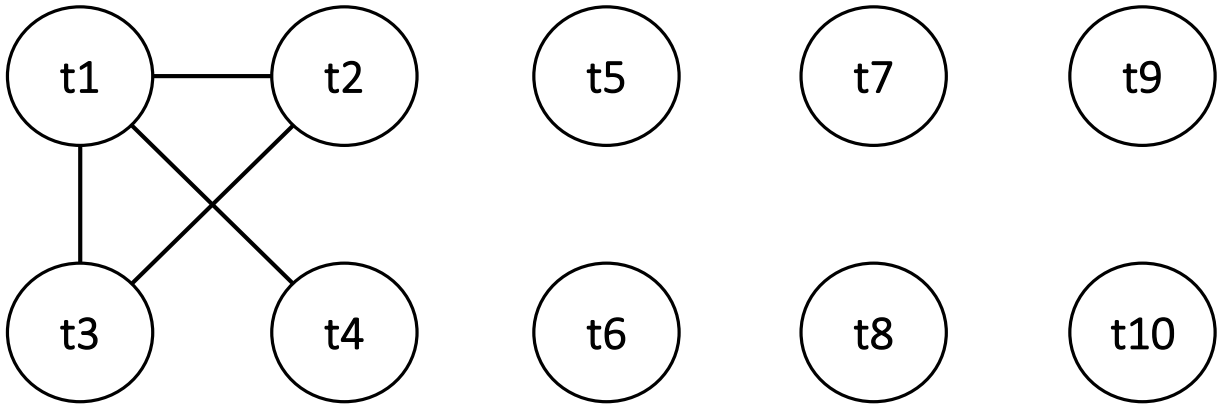
{ t6 }

{ t7 }

{ t7, t8 }

{ t9 }

{ t10 }



Interference Graph

{ t1 }

{ t2, t1 }

{ t2, t3, t1 }

{ t1, t4 }

{ t5 }

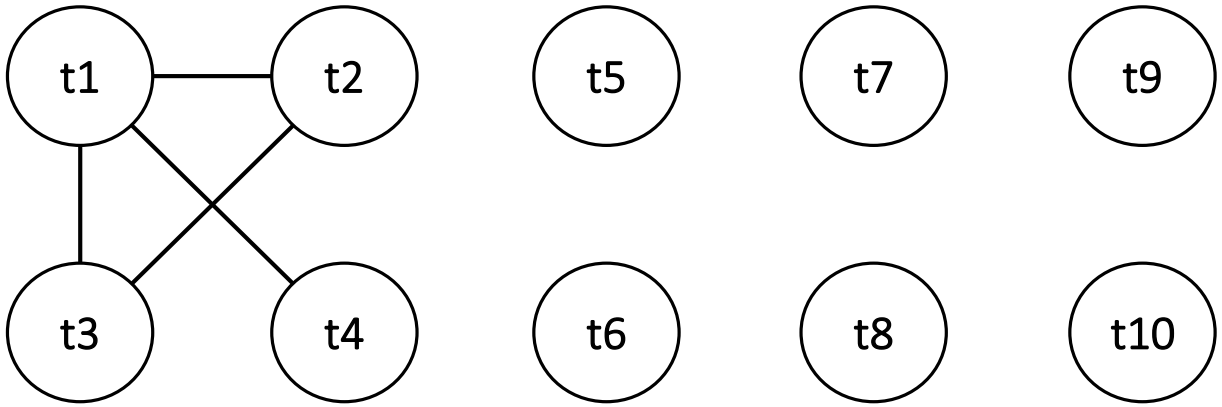
{ t6 }

{ t7 }

{ t7, t8 }

{ t9 }

{ t10 }



Interference Graph

{ t1 }

{ t2, t1 }

{ t2, t3, t1 }

{ t1, t4 }

{ t5 }

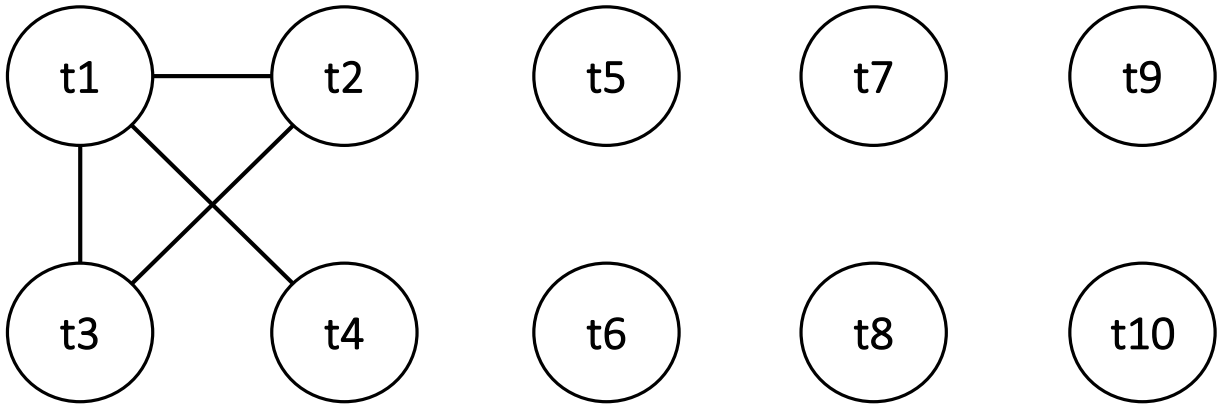
{ t6 }

{ t7 }

{ t7, t8 }

{ t9 }

{ t10 }



Interference Graph

{ t1 }

{ t2, t1 }

{ t2, t3, t1 }

{ t1, t4 }

{ t5 }

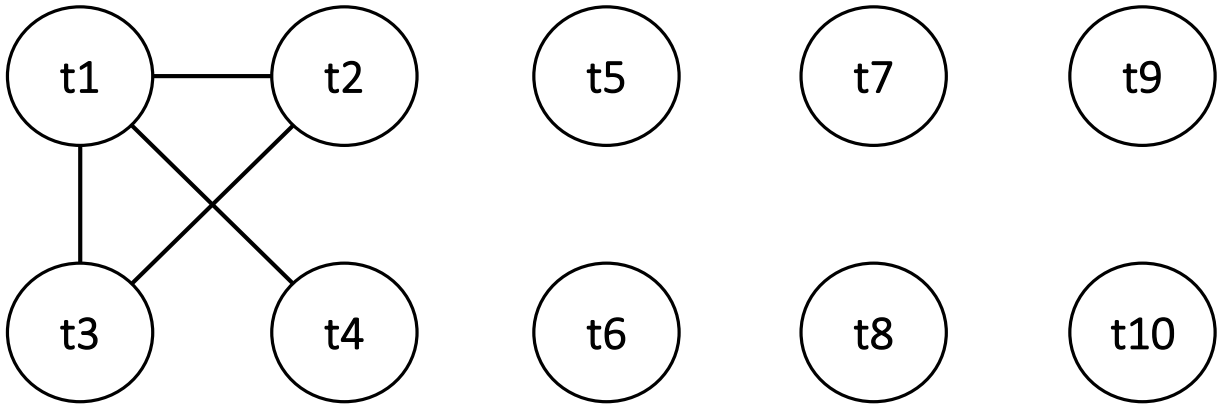
{ t6 }

{ t7 }

{ t7, t8 }

{ t9 }

{ t10 }



Interference Graph

{ t1 }

{ t2, t1 }

{ t2, t3, t1 }

{ t1, t4 }

{ t5 }

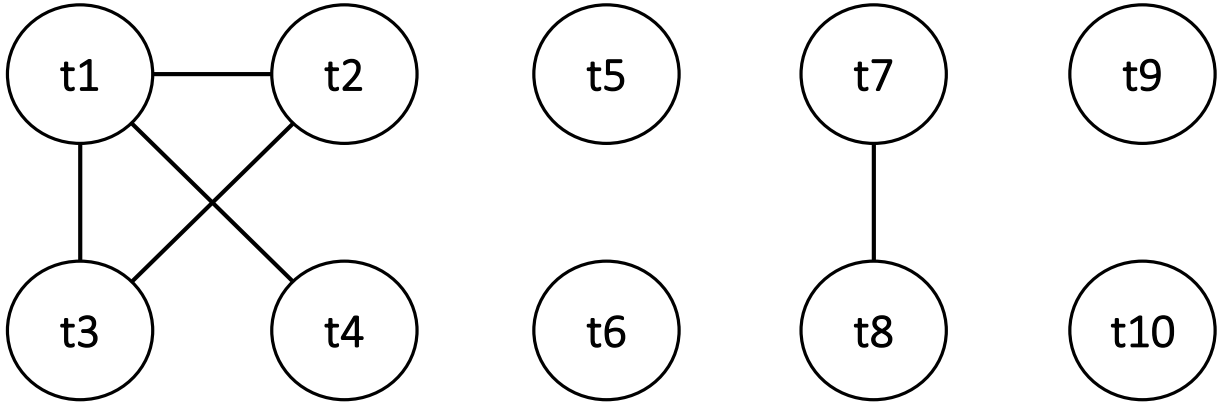
{ t6 }

{ t7 }

{ t7, t8 }

{ t9 }

{ t10 }



Interference Graph

{ t1 }

{ t2, t1 }

{ t2, t3, t1 }

{ t1, t4 }

{ t5 }

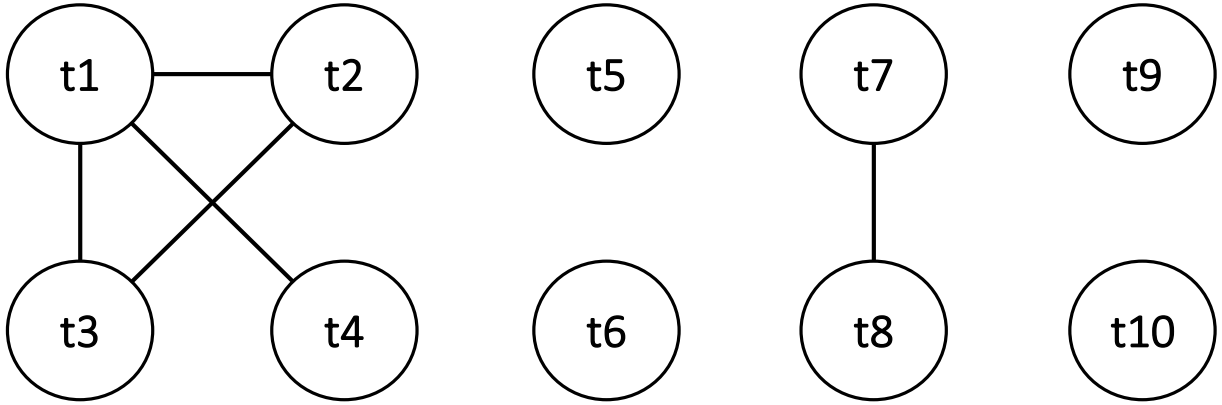
{ t6 }

{ t7 }

{ t7, t8 }

{ t9 }

{ t10 }



Interference Graph

{ t1 }

{ t2, t1 }

{ t2, t3, t1 }

{ t1, t4 }

{ t5 }

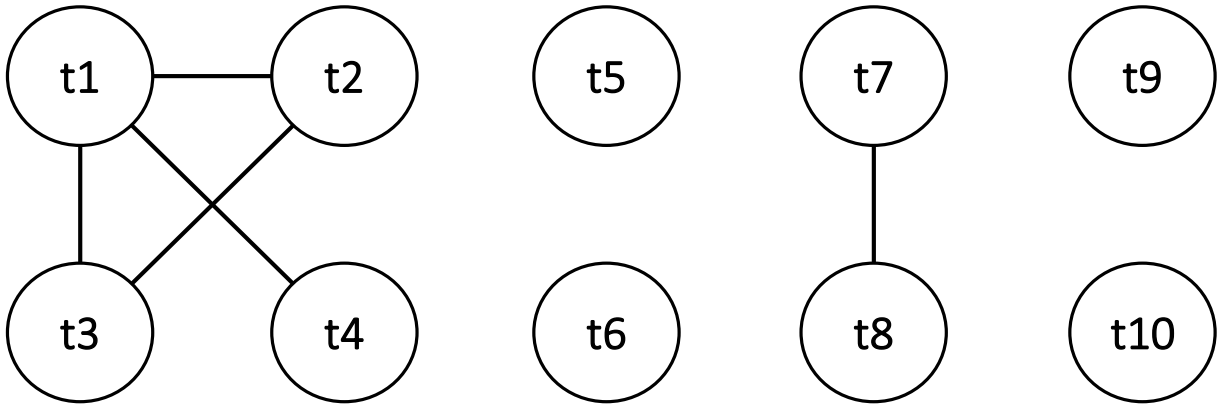
{ t6 }

{ t7 }

{ t7, t8 }

{ t9 }

{ t10 }



Graph Coloring

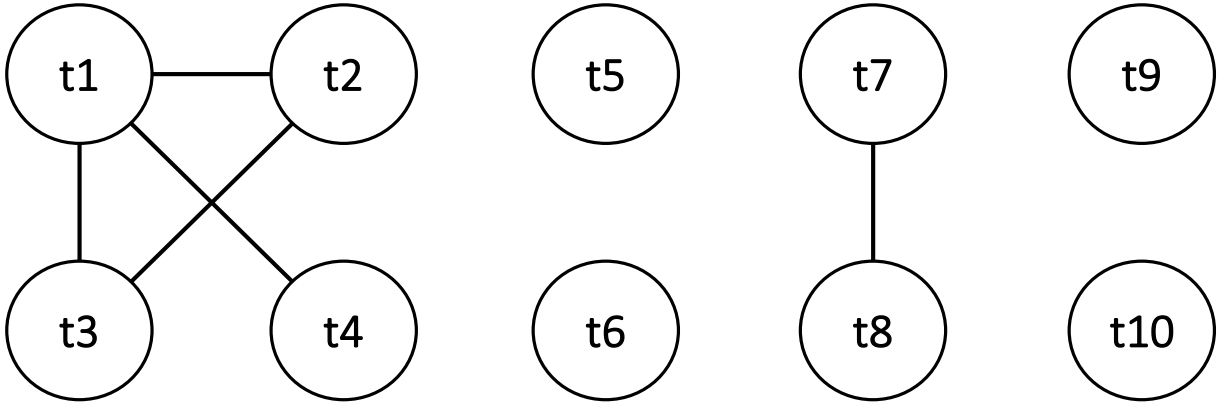
- Graph coloring is hard (NP-complete problem)
- We need a heuristic…

Graph Coloring

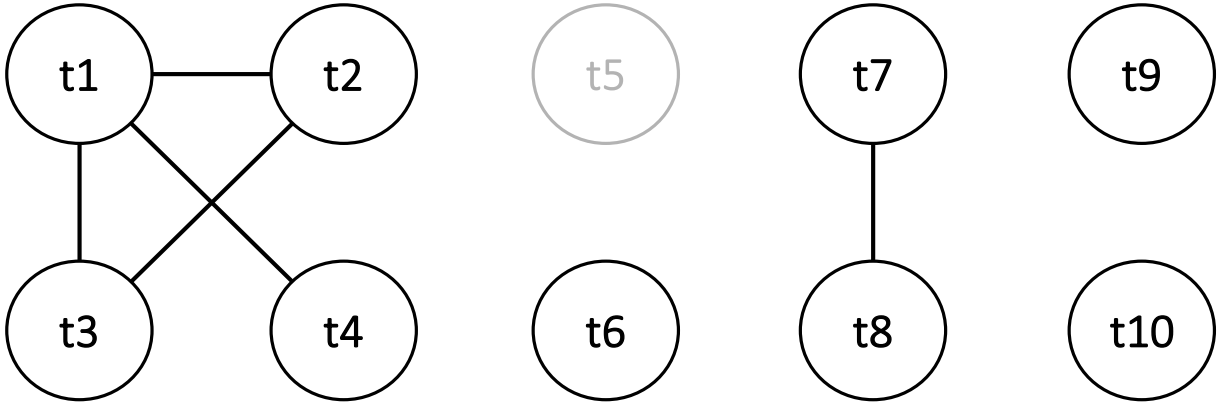
Chaitin's Algorithm for *k-coloring* (simplified):

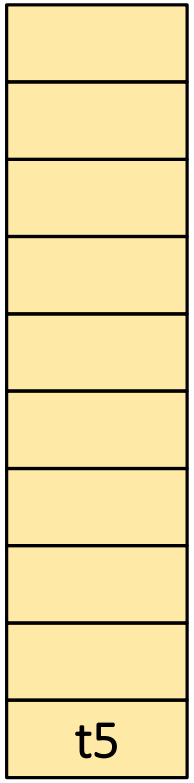
- While there is a node with less than k neighbors:
  - Remove it and it's edges from the graph
  - Push it on the stack
- If the entire graph was removed, then it's k-colorable
- While the stack is not empty:
  - Pop a node from the stack and assign it a color

Graph Coloring (k = 3)

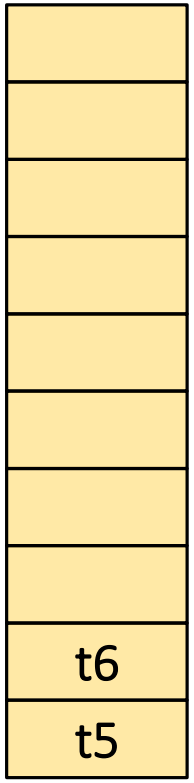


Graph Coloring (k = 3)

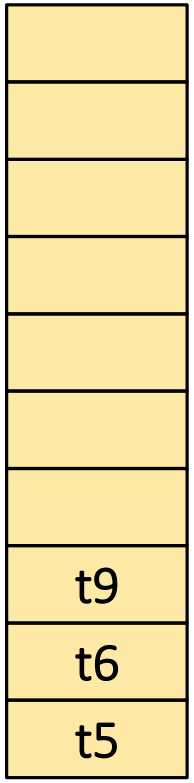




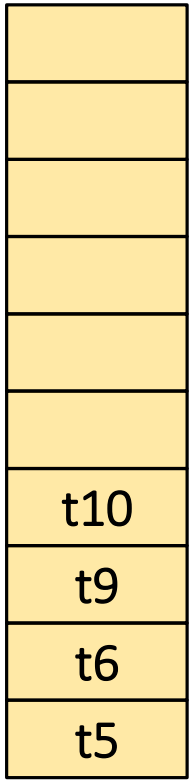
Graph Coloring (k = 3)



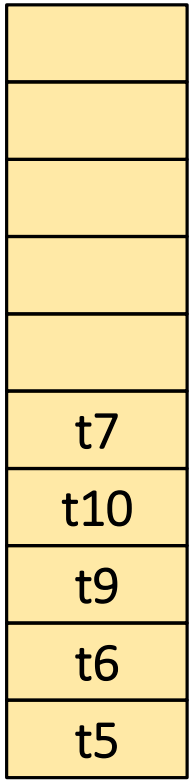
Graph Coloring (k = 3)



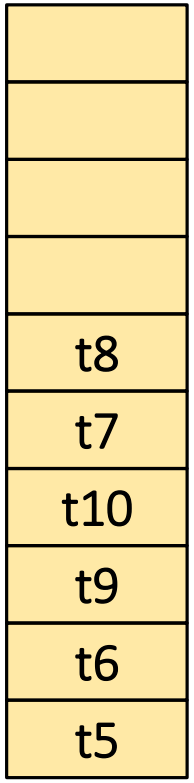
Graph Coloring (k = 3)



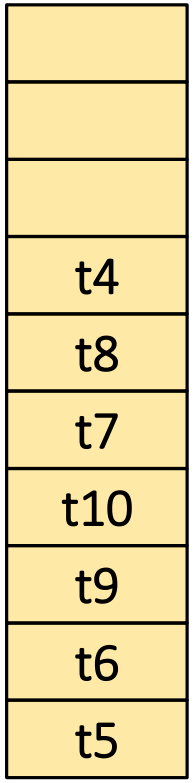
Graph Coloring (k = 3)



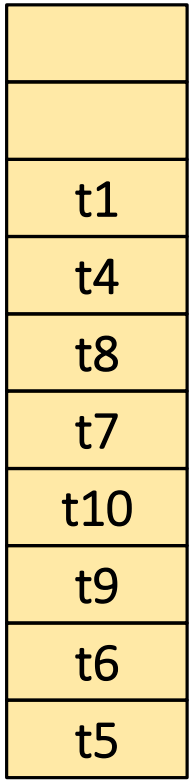
Graph Coloring (k = 3)



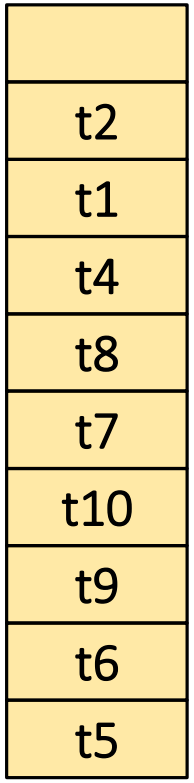
Graph Coloring (k = 3)



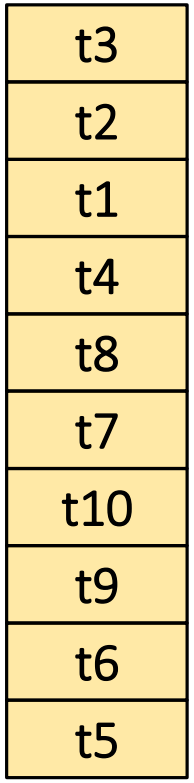
Graph Coloring (k = 3)



Graph Coloring (k = 3)



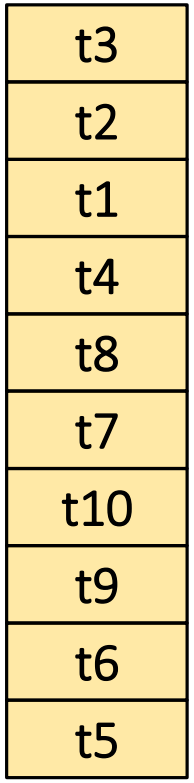
Graph Coloring (k = 3)



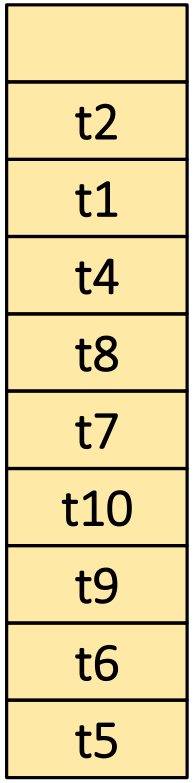
Graph Coloring (k = 3)

Pop nodes and assign colors…

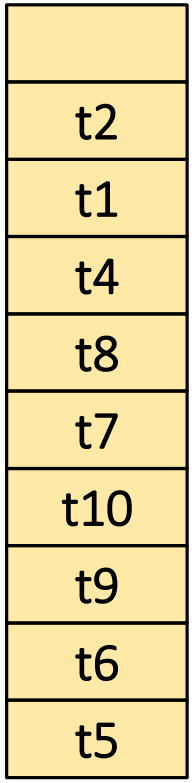
Graph Coloring (k = 3)



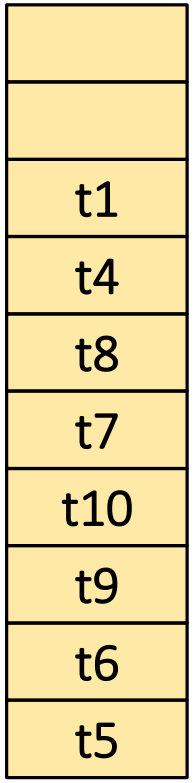
Graph Coloring (k = 3)



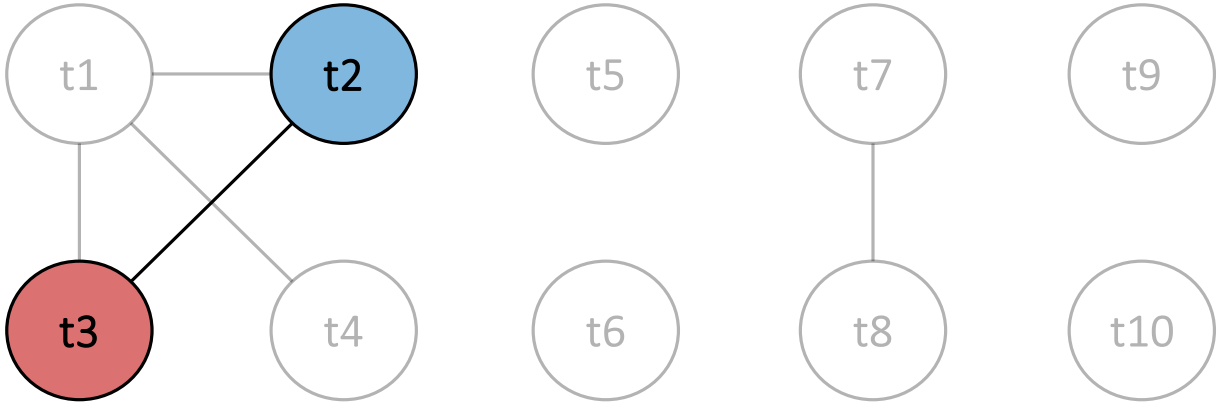
Graph Coloring (k = 3)

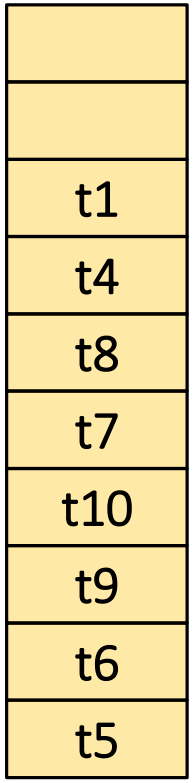


Graph Coloring (k = 3)

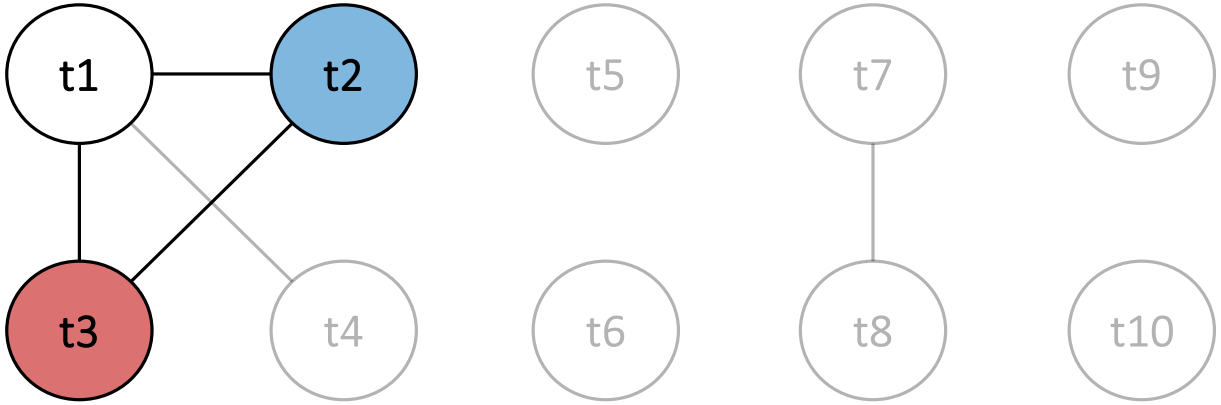


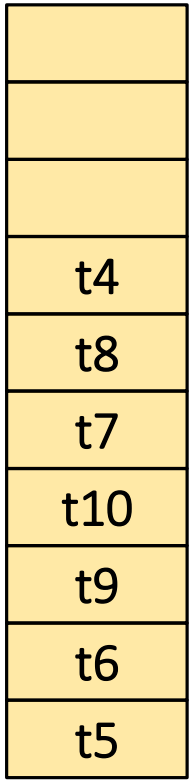
Graph Coloring (k = 3)



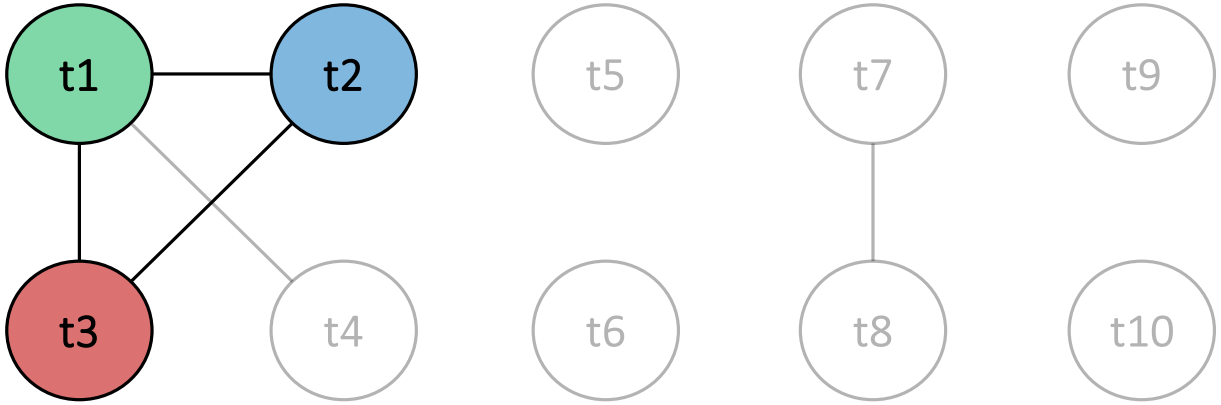


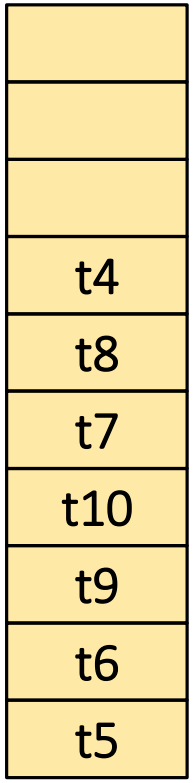
Graph Coloring (k = 3)



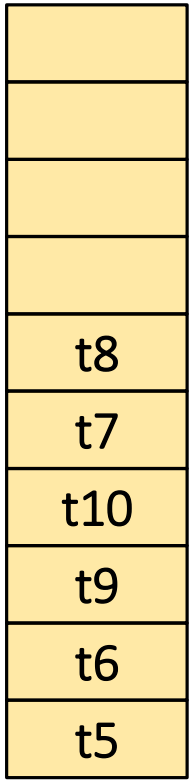


Graph Coloring (k = 3)

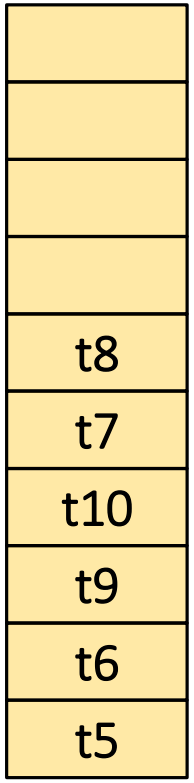




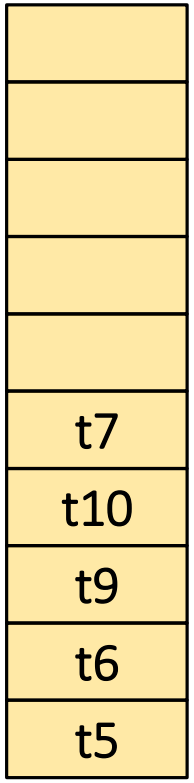
Graph Coloring (k = 3)



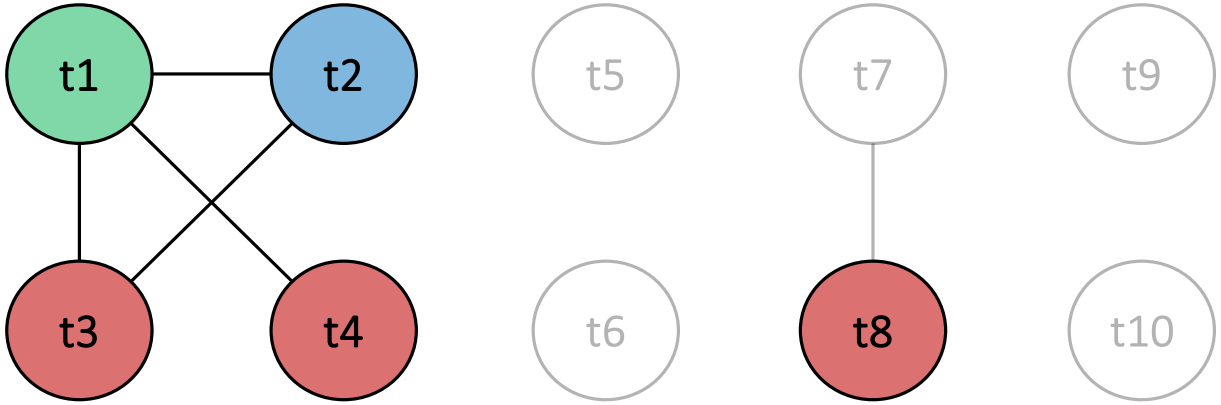
Graph Coloring (k = 3)

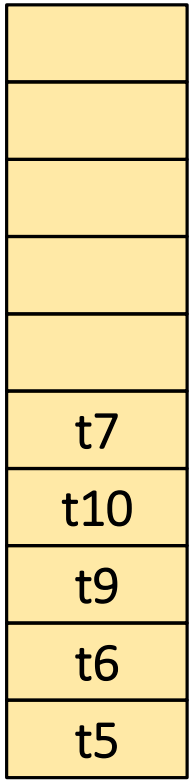


Graph Coloring (k = 3)

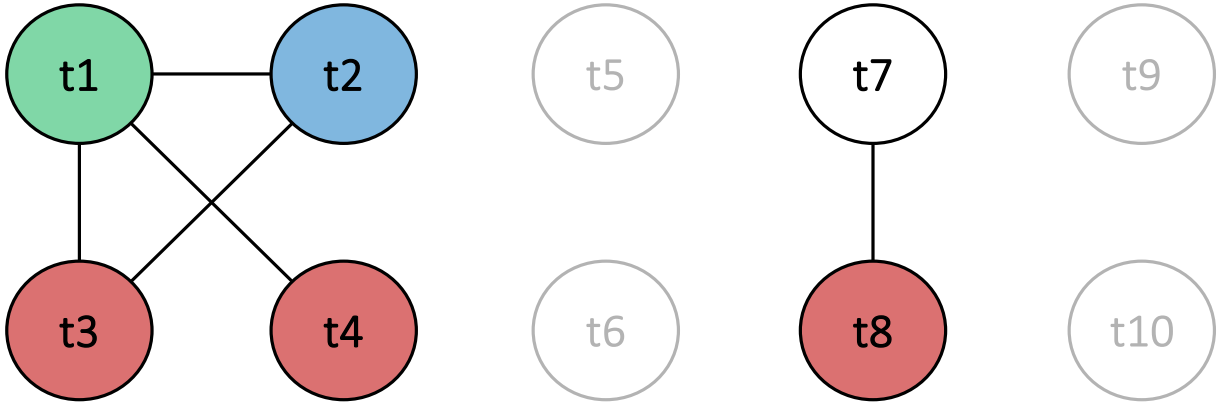


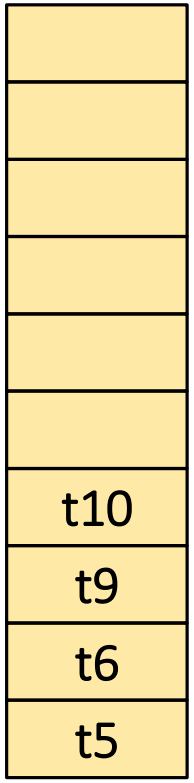
Graph Coloring (k = 3)



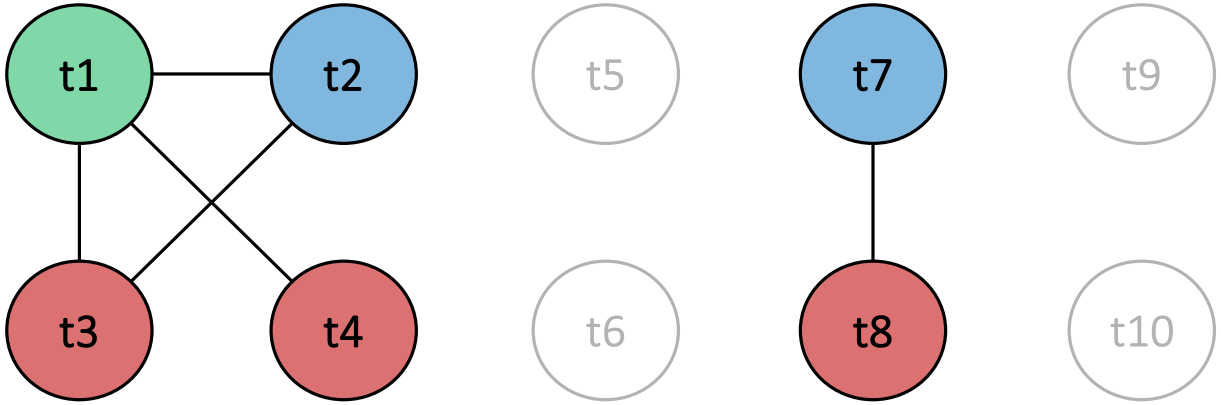


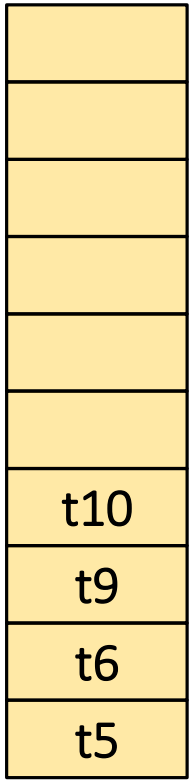
Graph Coloring (k = 3)



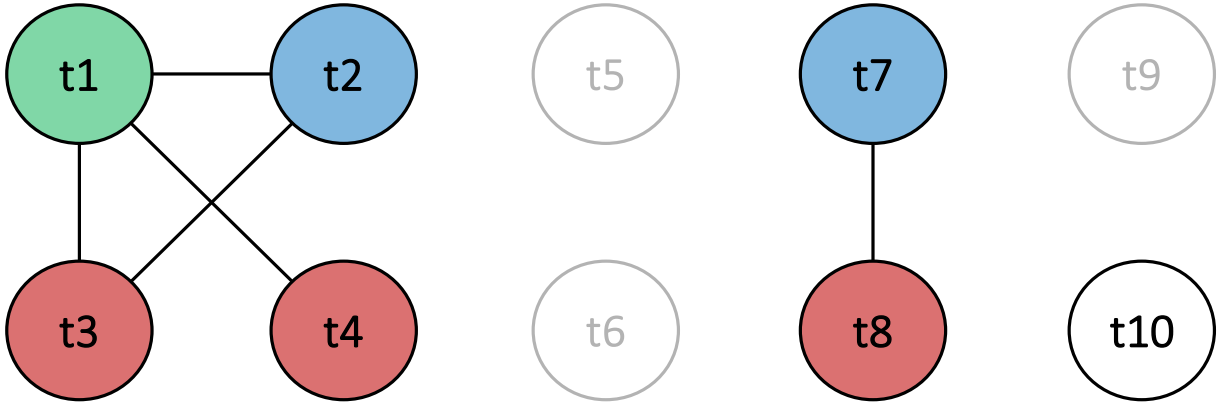


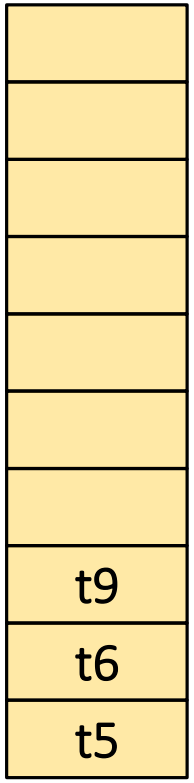
Graph Coloring (k = 3)



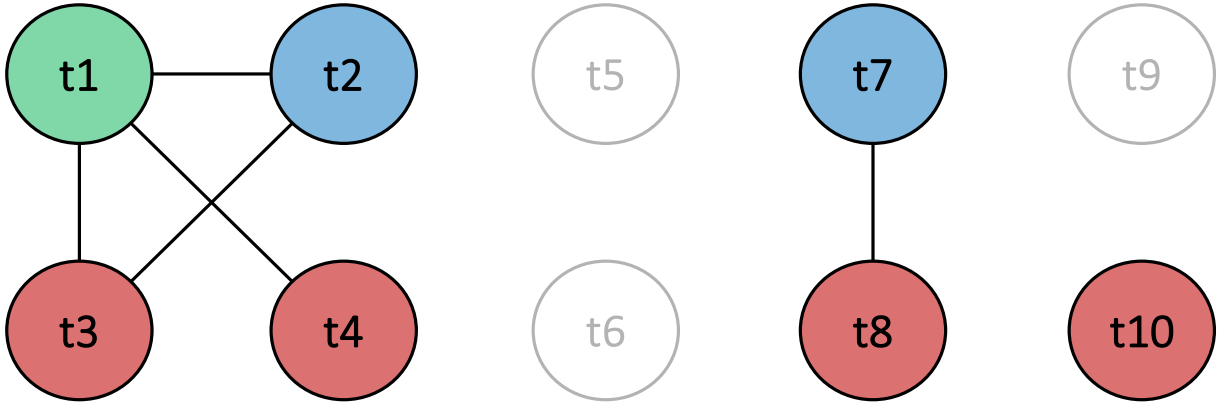


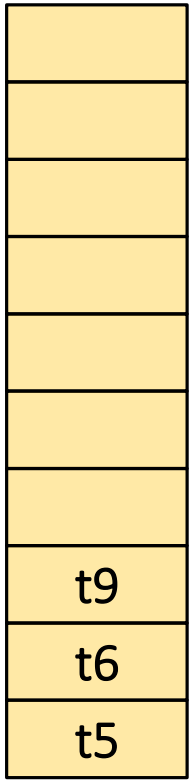
Graph Coloring (k = 3)



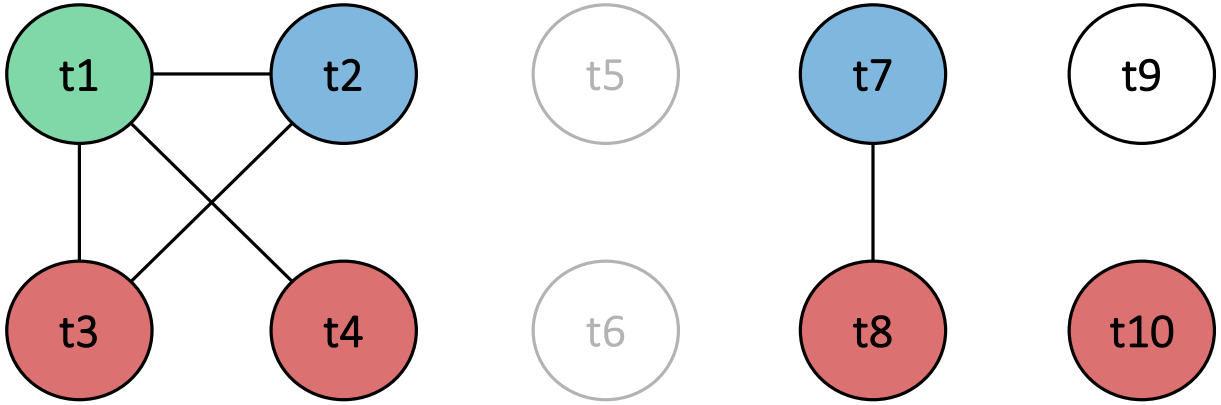


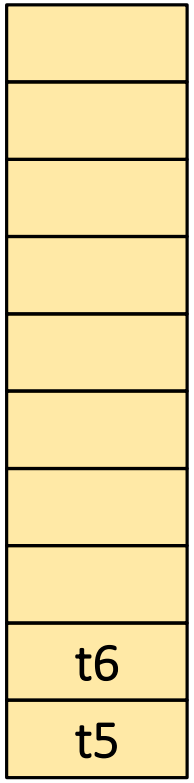
Graph Coloring (k = 3)



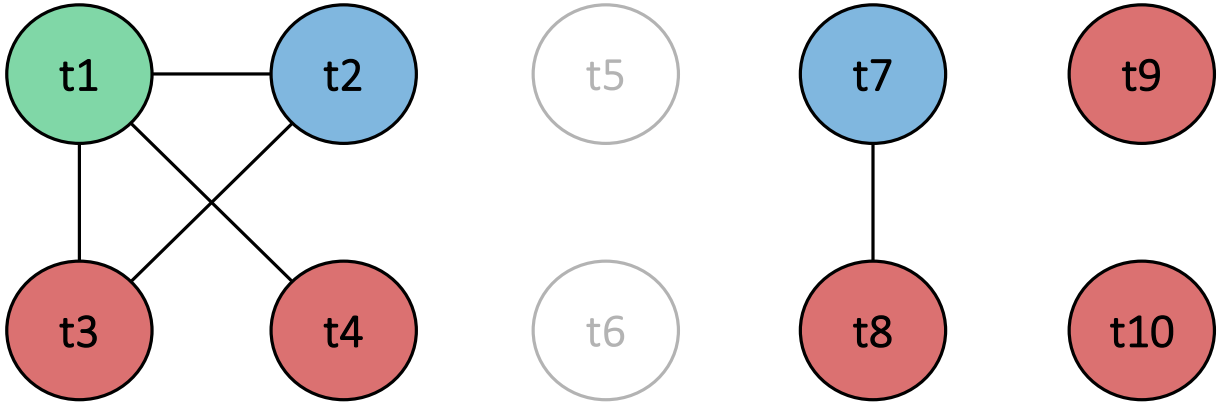


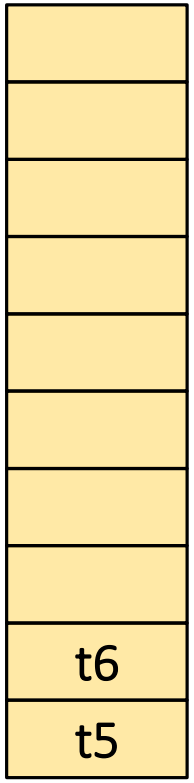
Graph Coloring (k = 3)



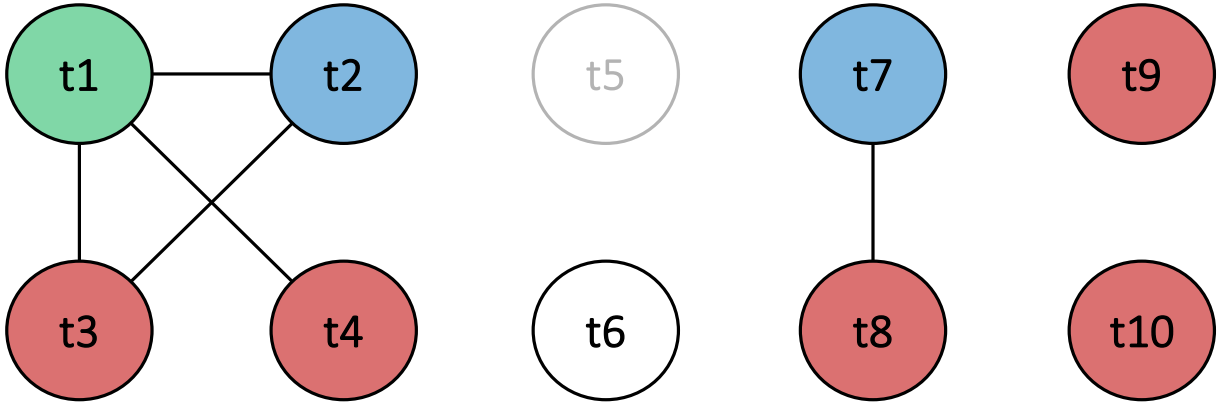


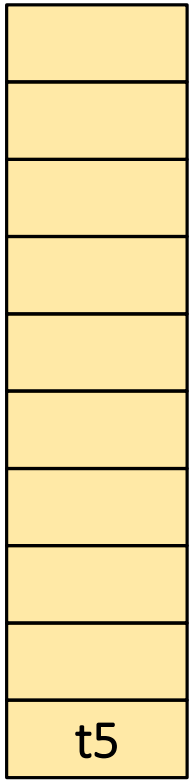
Graph Coloring (k = 3)



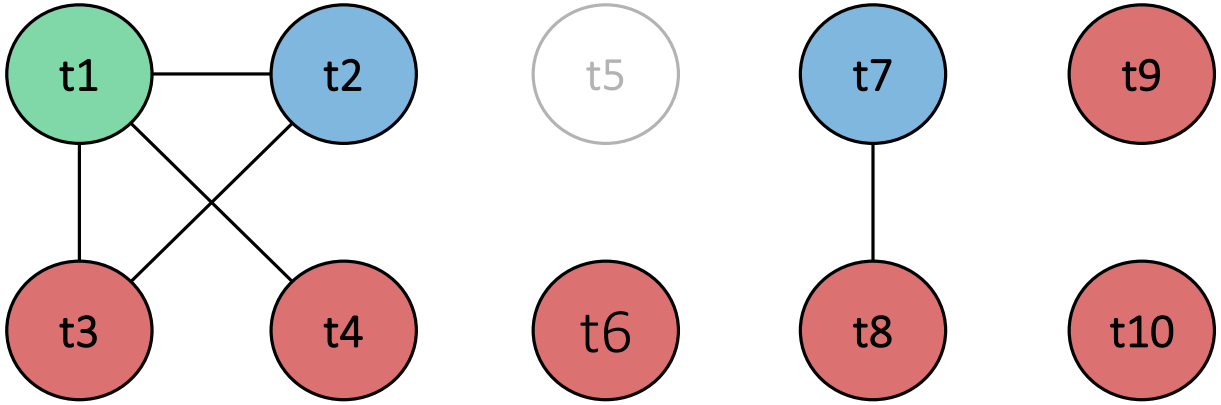


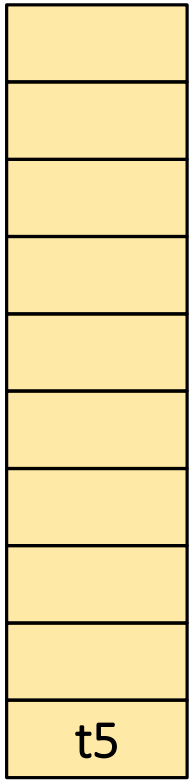
Graph Coloring (k = 3)



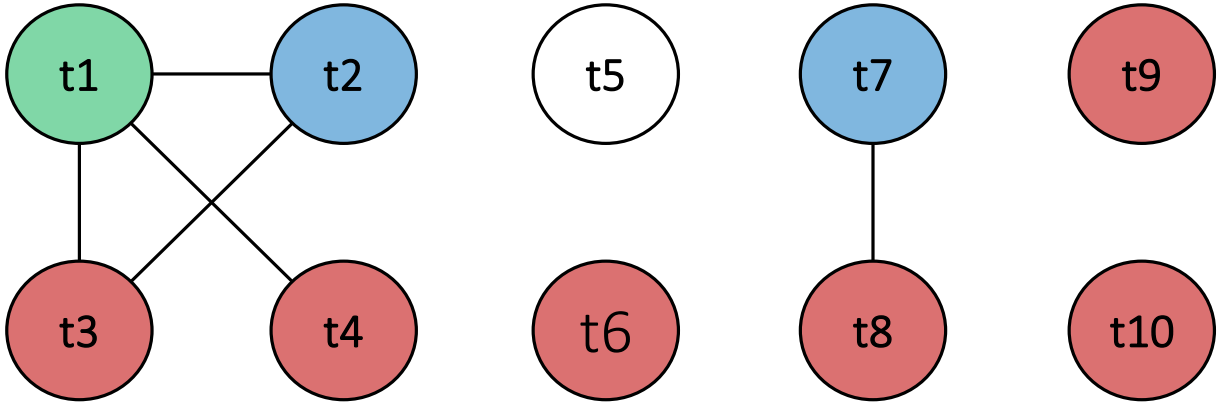


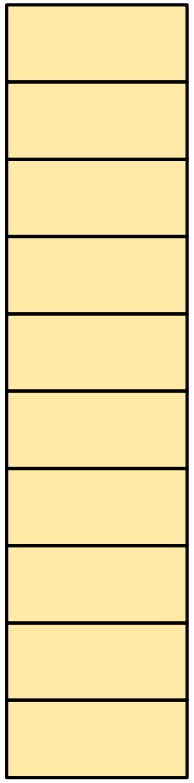
Graph Coloring (k = 3)



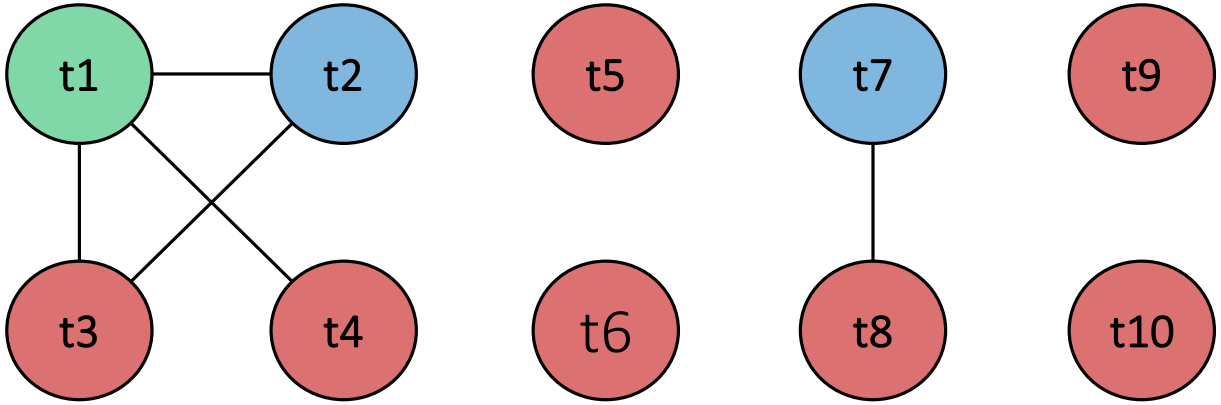


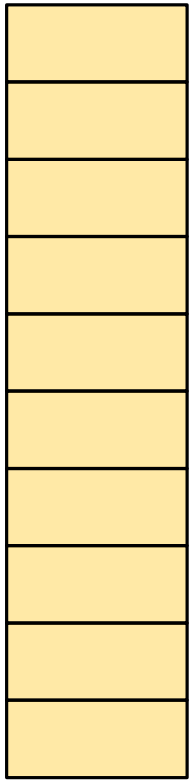
Graph Coloring (k = 3)





Graph Coloring (k = 3)





Register allocation

According to the coloring, our register allocation is:

IR Register	Color	MIPS Register
t1	R3	t2
t2	R2	t1
t3	R1	t0
t4	R1	t0
t5	R1	t0
t6	R1	t0
t7	R2	t1
t8	R1	t0
t9	R1	t0
t10	R1	t0

Register allocation

```
t1 = x
t2 = y
t3 = z
t4 = sub t2, t3
t5 = mult, t1, t4
a = t5
t6 = a
bne t6, 1, end
t7 = a
t8 = 1
t9 = add t7, t8
a = t9
end:
t10 = a
b = t10
```

IR Register	MIPS Register
t1	t2
t2	t1
t3	t0
t4	t0
t5	t0
t6	t0
t7	t1
t8	t0
t9	t0
t10	t0

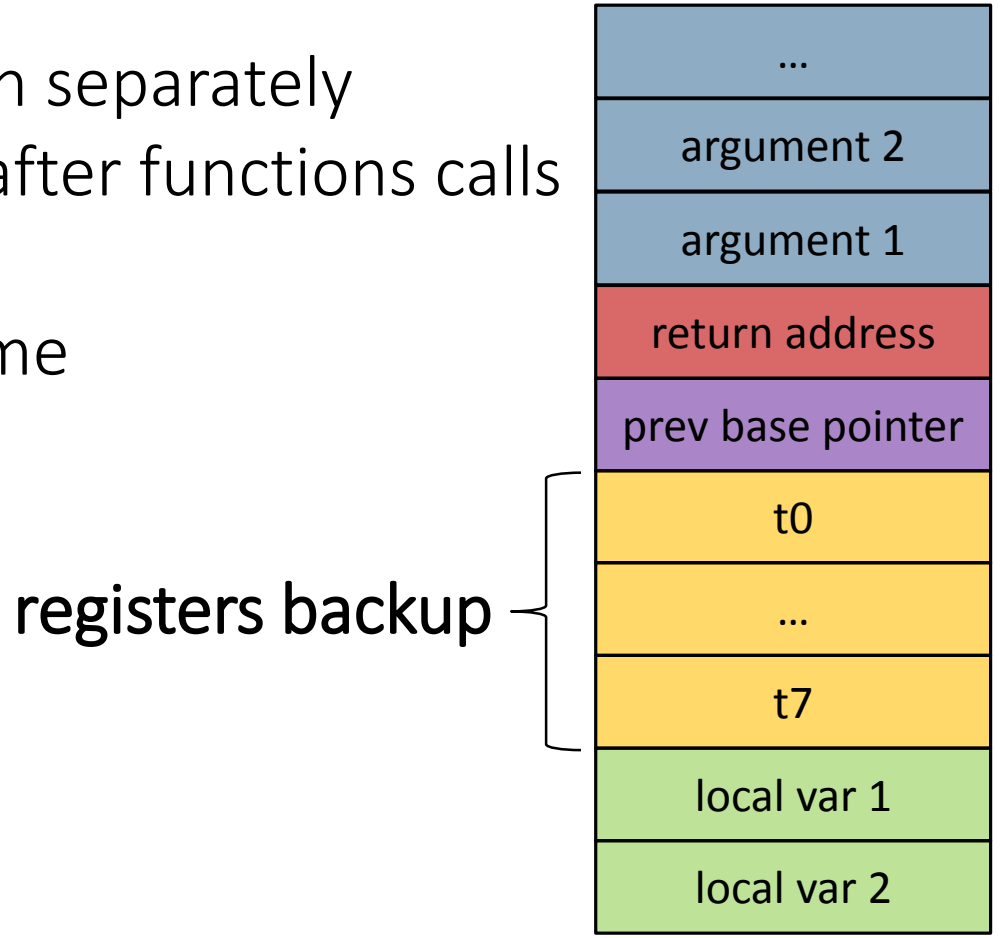
```
lw $t2, 8($fp)
lw $t1, 12($fp)
lw $t0, 16($fp)
sub $t0, $t1, $t0
mul $t0, $t2, $t0
sw $t0, -4($fp)
lw $t0, -4($fp)
bne $t0, 1, end
lw $t1, -4($fp)
li $t0, 1
add $t0, $t1, $t0
sw $t0, -4($fp)
end:
lw $t0, -4($fp)
sw $t0, -8($fp)
```

Register Backup

- Register allocation is run for each function separately
- We assume that called functions don’t change registers
- But how do we achieve that?

Register Backup

- Register allocation is run for each function separately
- We assume registers remain unchanged after functions calls
- But how do we achieve that?
  - Backup CPU registers on the stack frame
  - Registers: t0, t1, … t7



Tips

- Write some programs **manually** in SPIM
- Represent MIPS instructions with Java classes
- Use the interactive debugger (xspim)