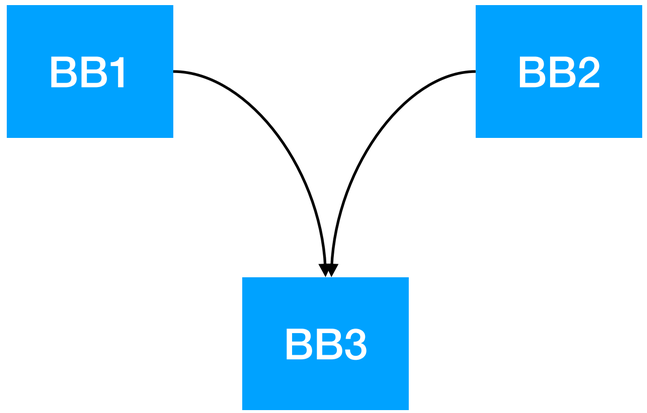
Global Register Allocation

drawbacks of local register allocation

- Why do we have to store all live/dirty registers at the end of a basic block?
  - Because we only consider per-basic block register allocation, and information may not match across basic blocks
- Consider the CFG



- In BB1, x is mapped to r1; in BB2, x is mapped to r2
  - What should x be mapped to in BB3?

global register allocation

- To make sure that a temporary/local/global has consistent mapping across basic blocks, we want to assign that variable to a register *for the entire function*
  - Isn't this kind of like our naïve register allocation approach?
  - Key: a register might have *multiple variables assigned to it*
- All variables with the same color can be assigned to the same register in this code

```
1:   T1 = A + B
2:   T2 = A + T1
3:   T3 = A + T2
4:   D = A + T3
5:   T4 = C + B
6:   T5 = T4 + C
7:   E = T5 + D
```

global register allocation

- Issues:
  - How do we know that two variables can be assigned to the same register?
  - How do we find the right assignment of variables to registers?
  - What do we do if we don't have enough registers to make the assignment?

```
1:   T1 = A + B
2:   T2 = A + T1
3:   T3 = A + T2
4:   D = A + T3
5:   T4 = C + B
6:   T5 = T4 + C
7:   E = T5 + D
```

co-locating variables

- Two variables can be assigned to the same register *if they are not live at the same time*
  - They don't have values we need at the same time

```
1:   T1 = A + B
2:   T2 = A + T1
3:   T3 = A + T2
4:   D = A + T3
5:   T4 = C + B
6:   T5 = T4 + C
7:   E = T5 + D
```

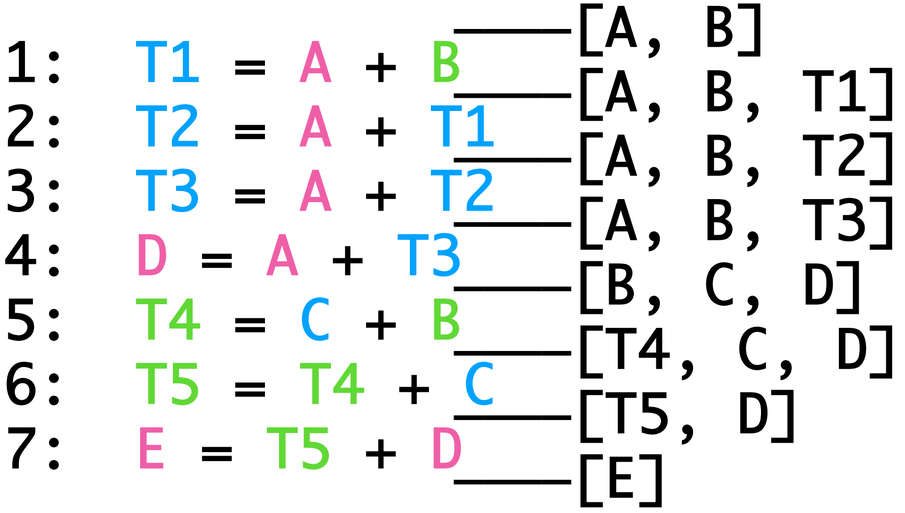
co-locating variables

- Two variables can be assigned to the same register *if they are not live at the same time*
  - They don't have values we need at the same time

```
1:  T1 = A + B      ____[A, B]
2:  T2 = A + T1     ____[A, B, T1]
3:  T3 = A + T2     ____[A, B, T2]
4:  D = A + T3      ____[A, B, T3]
5:  T4 = C + B      ____[B, C, D]
6:  T5 = T4 + C     ____[T4, C, D]
7:  E = T5 + D      ____[T5, D]
                    ____[E]
```

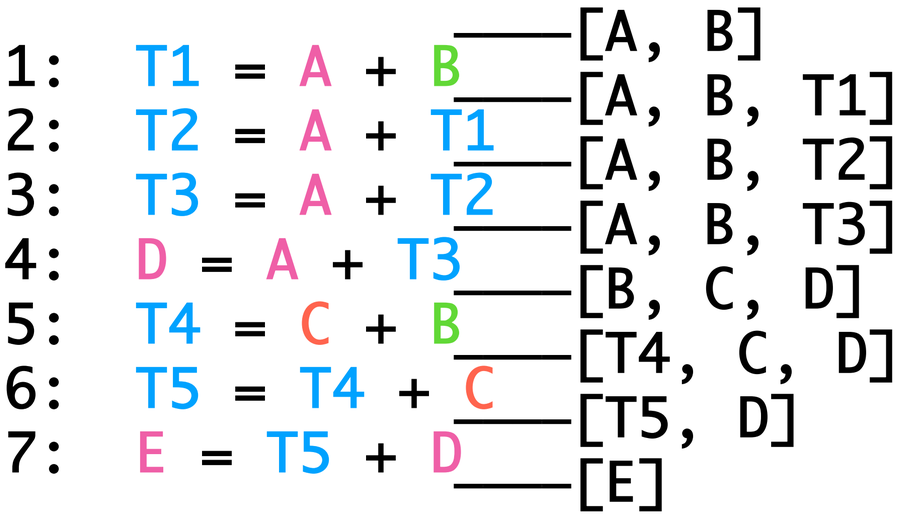
co-locating variables

- Just because you make sure that variables that are not live at the same time do not go in the same register doesn't mean you make the right assignments



making the right assignments

- Just because you make sure that variables that are not live at the same time do not go in the same register doesn't mean you make the right assignments
- If we put all the temporaries in the same register, then we need an extra register for C



next: graph coloring