Local Register Allocation
basic idea

• Perform register allocation on a *per basic block* basis
  • Register allocation across basic blocks is *global* — will discuss later
• Perform code generation and register allocation at the same time
  • Find registers for operands when translating 3AC to assembly
• Greedily reuse registers
  • Keep operands in registers if operand is *live*
  • If operand is already in register, no need for new loads
• Only store registers back to the stack if necessary
  • Need register for something else (*spill* register to stack/global memory)
  • At the end of basic block
tracking registers

• As code is generated keep track of:
  • What piece of data is in each register
    • In our case, two possibilities:
      1. Local variable/parameter or global variable
      2. Temporary
    • Whether the data is dirty (has its value changed since it was put into the register) — why?
1:  A = B + C
2:  C = A + B
3:  T1 = B + C
4:  T2 = T1 + C
5:  D = T2
6:  E = A + B
7:  B = E + D
8:  A = C + D
9:  T3 = A + B
10: WRITE(T3)
1: \( A = B + C \)
2: \( C = A + B \)
3: \( T1 = B + C \)
4: \( T2 = T1 + C \)
5: \( D = T2 \)
6: \( E = A + B \)
7: \( B = E + D \)
8: \( A = C + D \)
9: \( T3 = A + B \)
10: \text{WRITE}(T3)

<table>
<thead>
<tr>
<th>Inst</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{A, B}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>{A, B, C}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>{A, B, C, T1}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>{A, B, C, T2}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>{A, B, C, D}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>{C, D, E}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>{B, C, D}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>{A, B}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>{T3}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>{}</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1: LW R1 B
LW R2 C
ADD R2 R1 R2

1: \( A = B + C \)  
2: \( C = A + B \)  
3: \( T1 = B + C \)  
4: \( T2 = T1 + C \)  
5: \( D = T2 \)  
6: \( E = A + B \)  
7: \( B = E + D \)  
8: \( A = C + D \)  
9: \( T3 = A + B \)  
10: WRITE(T3)

<table>
<thead>
<tr>
<th>Inst</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### Example

<table>
<thead>
<tr>
<th>Inst</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>A*</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1: \( A = B + C \)

2: \( C = A + B \)

3: \( T_1 = B + C \)

4: \( T_2 = T_1 + C \)

5: \( D = T_2 \)

6: \( E = A + B \)

7: \( B = E + D \)

8: \( A = C + D \)

9: \( T_3 = A + B \)

10: WRITE(T3)
example

1: LW R1 B
LW R2 C
ADD R2 R1 R2
2: ADD R3 R2 R1
3: ADD R1 R1 R3

1: \( A = B + C \)
2: \( C = A + B \)
3: \( T1 = B + C \)
4: \( T2 = T1 + C \)
5: \( D = T2 \)
6: \( E = A + B \)
7: \( B = E + D \)
8: \( A = C + D \)
9: \( T3 = A + B \)
10: WRITE(T3)

<table>
<thead>
<tr>
<th>Inst</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>A*</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>3</td>
<td>T1*</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1:  LW R1 B
    LW R2 C
    ADD R2 R1 R2
2:  ADD R3 R2 R1
3:  ADD R1 R1 R3
4:  ADD R1 R1 R3

1:  A = B + C  1:  {A, B}
2:  C = A + B  2:  {A, B, C}
3:  T1 = B + C  3:  {A, B, C, T1}
4:  T2 = T1 + C  4:  {A, B, C, T2}
5:  D = T2  5:  {A, B, C, D}
6:  E = A + B  6:  {C, D, E}
7:  B = E + D  7:  {B, C, D}
8:  A = C + D  8:  {A, B}
9:  T3 = A + B  9:  {T3}
10: WRITE(T3)  10: {}
1:  LW R1 B
   LW R2 C
   ADD R2 R1 R2
2:  ADD R3 R2 R1
3:  ADD R1 R1 R3
4:  ADD R1 R1 R3
5:

1:  A = B + C  1:  \{A, B\}
2:  C = A + B  2:  \{A, B, C\}
3:  T1 = B + C  3:  \{A, B, C, T1\}
4:  T2 = T1 + C  4:  \{A, B, C, T2\}
5:  D = T2  5:  \{A, B, C, D\}
6:  E = A + B  6:  \{C, D, E\}
7:  B = E + D  7:  \{B, C, D\}
8:  A = C + D  8:  \{A, B\}
9:  T3 = A + B  9:  \{T3\}
10: WRITE(T3)  10: \{\}

<table>
<thead>
<tr>
<th>Inst</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>A*</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>3</td>
<td>T1*</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>4</td>
<td>T2*</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>5</td>
<td>D*</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
1: LW R1 B  
   LW R2 C  
   ADD R2 R1 R2
2: ADD R3 R2 R1
3: ADD R1 R1 R3
4: ADD R1 R1 R3
5:               
6: SW R3 C  
   LW R3 B  
   ADD R2 R2 R3

1: A = B + C  
1: \{A, B\}
2: C = A + B  
2: \{A, B, C\}
3: T1 = B + C  
3: \{A, B, C, T1\}
4: T2 = T1 + C  
4: \{A, B, C, T2\}
5: D = T2  
5: \{A, B, C, D\}
6: E = A + B  
6: \{C, D, E\}
7: B = E + D  
7: \{B, C, D\}
8: A = C + D  
8: \{A, B\}
9: T3 = A + B  
9: \{T3\}
10: WRITE(T3)  
10: \{\}

\begin{tabular}{|c|c|c|c|}
\hline
Inst & R1 & R2 & R3 \\
\hline
1 & B & A* & \\
\hline
2 & B & A* & C* \\
\hline
3 & T1* & A* & C* \\
\hline
4 & T2* & A* & C* \\
\hline
5 & D* & A* & C* \\
\hline
6 & D* & E* & \\
\hline
7 & & & \\
\hline
8 & & & \\
\hline
9 & & & \\
\hline
10 & & & \\
\hline
\end{tabular}
1: \text{LW R1 B}  
   \text{LW R2 C}  
   \text{ADD R2 R1 R2}  
2: \text{ADD R3 R2 R1}  
3: \text{ADD R1 R1 R3}  
4: \text{ADD R1 R1 R3}  
5: \text{ADD R1 R1 R3}  
6: \text{SW R3 C}  
   \text{LW R3 B}  
   \text{ADD R2 R2 R3}  
7: \text{ADD R2 R2 R1}  
8: \text{A = B + C}  
9: \text{T1 = B + C}  
10: \text{T2 = T1 + C}  
11: \text{D = T2}  
12: \text{E = A + B}  
13: \text{B = E + D}  
14: \text{A = C + D}  
15: \text{T3 = A + B}  
16: \text{WRITE(T3)}  
17: \text{}  

\begin{tabular}{|c|c|c|c|}
\hline
\text{Inst} & \text{R1} & \text{R2} & \text{R3} \\
\hline
1 & B & A* & \\
2 & B & A* & C* \\
3 & T1* & A* & C* \\
4 & T2* & A* & C* \\
5 & D* & A* & C* \\
6 & D* & E* & \\
7 & D* & B* & \\
8 & & & \\
9 & & & \\
10 & & & \\
\hline
\end{tabular}
example

<table>
<thead>
<tr>
<th>Inst</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>A*</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>3</td>
<td>T1*</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>4</td>
<td>T2*</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>5</td>
<td>D*</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>6</td>
<td>D*</td>
<td>E*</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>D*</td>
<td>B*</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>A*</td>
<td>B*</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1:  $A = B + C$
2:  $C = A + B$
3:  $T1 = B + C$
4:  $T2 = T1 + C$
5:  $D = T2$
6:  $E = A + B$
7:  $B = E + D$
8:  $A = C + D$
9:  $T3 = A + B$
10: WRITE(T3)

---

1:  LW R1 B
2:  LW R2 C
3:  ADD R2 R1 R2
4:  ADD R3 R2 R1
5:  ADD R1 R1 R3
6:  ADD R1 R1 R3
7:  SW R3 C
8:  LW R3 B
9:  ADD R2 R2 R3
10: ADD R2 R2 R1
11: LW C R3
12: ADD R1 R3 R1

---

<table>
<thead>
<tr>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A, B}</td>
<td>{A, B, C}</td>
<td>{A, B, C, T1}</td>
<td>{A, B, C, T2}</td>
<td>{A, B, C, D}</td>
<td>{C, D, E}</td>
<td>{B, C, D}</td>
<td>{A, B}</td>
<td>{T3}</td>
<td>{}</td>
</tr>
</tbody>
</table>

---

1:   LW R1 B
2:   LW R2 C
3:   ADD R2 R1 R2
4:   ADD R3 R2 R1
5:   ADD R1 R1 R3
6:   ADD R1 R1 R3
7:   SW R3 C
8:   LW R3 B
9:   ADD R2 R2 R3
10:  ADD R2 R2 R1
11:  LW C R3
12:  ADD R1 R3 R1

---

1:  A = B + C
2:  C = A + B
3:  T1 = B + C
4:  T2 = T1 + C
5:  D = T2
6:  E = A + B
7:  B = E + D
8:  A = C + D
9:  T3 = A + B
10: WRITE(T3)
1:  LW R1 B  
    LW R2 C  
    ADD R2 R1 R2
2:  ADD R3 R2 R1
3:  ADD R1 R1 R3
4:  ADD R1 R1 R3
5:  
6:  SW R3 C  
    LW R3 B  
    ADD R2 R2 R3
7:  ADD R2 R2 R1
8:  LW C R3  
    ADD R1 R3 R1
9:  ADD R1 R1 R2

1:  A = B + C  

2:  C = A + B  

3:  T1 = B + C  

4:  T2 = T1 + C  

5:  D = T2  

6:  E = A + B  

7:  B = E + D  

8:  A = C + D  

9:  T3 = A + B  

10: WRITE(T3)

<table>
<thead>
<tr>
<th>Inst</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>A*</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>3</td>
<td>T1*</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>4</td>
<td>T2*</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>5</td>
<td>D*</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>6</td>
<td>D*</td>
<td>E*</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>D*</td>
<td>B*</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>A*</td>
<td>B*</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>T3*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
example

1:  LW R1 B
   LW R2 C
   ADD R2 R1 R2  
2:  ADD R3 R2 R1
3:  ADD R1 R1 R3
4:  ADD R1 R1 R3
5:  
6:  SW R3 C
   LW R3 B
   ADD R2 R2 R3
7:  ADD R2 R2 R1
8:  LW C R3
   ADD R1 R3 R1
9:  ADD R1 R1 R2
10: PUTI R1

1:  A = B + C  1:  \{A, B\}
2:  C = A + B  2:  \{A, B, C\}
3:  T1 = B + C  3:  \{A, B, C, T1\}
4:  T2 = T1 + C  4:  \{A, B, C, T2\}
5:  D = T2  5:  \{A, B, C, D\}
6:  E = A + B  6:  \{C, D, E\}
7:  B = E + D  7:  \{B, C, D\}
8:  A = C + D  8:  \{A, B\}
9:  T3 = A + B  9:  \{T3\}
10: WRITE(T3)  10: \{

<table>
<thead>
<tr>
<th>Inst</th>
<th>R1</th>
<th>R2</th>
<th>R3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>B</td>
<td>A*</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>B</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>3</td>
<td>T1*</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>4</td>
<td>T2*</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>5</td>
<td>D*</td>
<td>A*</td>
<td>C*</td>
</tr>
<tr>
<td>6</td>
<td>D*</td>
<td>E*</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>D*</td>
<td>B*</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>A*</td>
<td>B*</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>T3*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
key operations

- **ensure**: make sure that a value exists in a register (put the value in the register, if necessary)
- **allocate**: find a register for a value (kick another value out of a register, if necessary)
- **free**: kick a value out of a register (save the value to the stack/global space if necessary)
For each tuple $C = A \text{ op } B$ in a BB, do

- $R_x = \text{ensure}(A)$
- $R_y = \text{ensure}(B)$
- If $A$ dead after this tuple, $\text{free}(R_x)$
- If $B$ dead after this tuple, $\text{free}(R_y)$
- $R_z = \text{allocate}(C)$ //could use $R_x$ or $R_y$
- Generate code for op
- Mark $R_z$ dirty

At end of BB, for each dirty register

- Generate code to store register into appropriate variable

---

Ensure(oopr)

- If oopr is already in register $r$
  - Return $r$
- Else
  - $r = \text{allocate}(oopr)$
  - Generate load from oopr into $r$
  - Return $r$

Allocate(oopr)

- If there is a free $r$
  - Choose $r$
- Else
  - Choose $r$ to free
  - $\text{free}(r)$
  - Mark $r$ associated with oopr
  - Return $r$

Free(r)

- If $r$ is marked dirty and variable is live
  - Generate store
- Mark $r$ as free