3. The following fragment of 3-address code was produced by a non-optimizing compiler:

\[
\begin{align*}
\text{start: } & \quad x := 1 \\
& \quad y := 1 \\
& \quad \text{sum := x} \\
& \quad \text{sum := sum + y} \\
& \quad \text{loop: if x = n then goto out2} \\
& \quad \quad t1 := y \\
& \quad \quad t2 := t1 * t1 \\
& \quad \quad t3 := t1 + x \\
& \quad \quad \text{if t3 < n then goto skip} \\
& \quad \quad t3 := t3 - n \\
& \quad \quad \text{skip: sum := sum + t3} \\
& \quad \quad y := y + 1 \\
& \quad \quad \text{goto loop} \\
& \quad \quad x := \text{sum} \\
& \quad \quad \text{sum := x + 1} \\
& \quad \quad x := x + 1 \\
& \quad \quad \text{out1: } x := x - 1 \\
& \quad \quad t1 := x \\
& \quad \quad \text{print t1} \\
& \quad \quad \text{if x = 0 then goto out} \\
\end{align*}
\]

Assume that there are no entry points into the code from outside other than at \text{start}.

(a) Decompose the code into basic blocks \(B_1, B_2, \ldots\), giving a range of line numbers for each.

(b) Draw the control flow graph, and describe any unreachable code.

(c) For each control point, list which variables are live (fill in the table below). Treat the array \(a\) as a single variable. Assume that \(n\) and \text{sum} are the only live variables immediately before line 24.

<table>
<thead>
<tr>
<th>Before line</th>
<th>Live variables</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td></td>
</tr>
<tr>
<td>\ldots</td>
<td></td>
</tr>
</tbody>
</table>

Apr 19-11:01 AM
Detect loops

L

A loop is a set of vertices in a control-flow digraph satisfying

1. L is strongly connected [can follow a path in L from any node in L to any other node in L]

2. L has a unique entry point

An inner loop is a loop that has no proper subloops.

most execution time is spent in inner loops.
concentrate optimization here.

Liveness analysis

\[ x = y \times z \]

\[ x \] is set \[ y \] and \[ z \] are used

at any given point in a program: a variable is live (or alive) if there is some control flow path to a spot where that variable is used with no intervening set of the variable. Otherwise, var is dead.

\[ x = x \times 2 \]

\[ x \] before \[ x \] after

\[ \rightarrow x = y \times z \rightarrow x = 6 \]

\[ \text{can remove if } x \text{'s dead} \]