Program Optimisation
Solutions of Homework 2

1. a) Let $h(\mathbb{D}_1) = m$ and $h(\mathbb{D}_2) = n$.
   - We show that $h(\mathbb{D}_1 \times \mathbb{D}_2) \geq m + n$. $\mathbb{D}_1$ has a chain $\perp \sqsubset d_1 \sqsubset \ldots \sqsubset d_m$ and $\mathbb{D}_2$ has a chain, $\perp \sqsubset d'_1 \sqsubset \ldots \sqsubset d'_n$. Then $(\perp, \perp) \sqsubset (d_1, \perp) \sqsubset \ldots \sqsubset (d_m, \perp) \sqsubset (d_m, d'_1) \sqsubset \ldots \sqsubset (d_m, d'_n)$ is a chain in $\mathbb{D}_1 \times \mathbb{D}_2$ of length $m + n$.
   - Now we show that $h(\mathbb{D}_1 \times \mathbb{D}_2) \leq m + n$. First we show by induction that

     (*) if $(a_0, b_0) \sqsubset \ldots \sqsubset (a_i, b_i)$ is any chain in $\mathbb{D}_1 \times \mathbb{D}_2$ then $|\{a_1, \ldots, a_i\}| + |\{b_1, \ldots, b_i\}| \geq i + 2$

     Now let $(\perp, \perp) \sqsubset (a_1, b_1) \sqsubset \ldots \sqsubset (a_k, b_k)$ be any chain in $\mathbb{D}_1 \times \mathbb{D}_2$. We have to show that $k \leq m + n$. Since $h(\mathbb{D}_1) = m$ hence $|\{\perp, a_1, \ldots, a_k\}| \leq m + 1$. Similarly $|\{\perp, b_1, \ldots, b_k\}| \leq n + 1$. Hence $|\{\perp, a_1, \ldots, a_k\}| + |\{\perp, b_1, \ldots, b_k\}| \leq m + n + 2$. But from (*) we have $|\{\perp, a_1, \ldots, a_k\}| + |\{\perp, b_1, \ldots, b_k\}| \geq k + 2$. Hence $k + 2 \leq m + n + 2$, i.e. $k \leq m + n$.

   b) We show by induction on k that $h({\mathbb{D}_1}^k) = k \cdot h(\mathbb{D}_1)$. For $k = 1$ the result is clear. Now suppose we have shown for some k that $h({\mathbb{D}_1}^k) = k \cdot h(\mathbb{D}_1)$. From part (a), $h({\mathbb{D}_1}^{k+1}) = h({\mathbb{D}_1}^k) + h(\mathbb{D}_1)$. Hence we have $h({\mathbb{D}_1}^{k+1}) = k \cdot h(\mathbb{D}_1) + h(\mathbb{D}_1) = (k + 1)h(\mathbb{D}_1)$.

   c) Let $|\mathbb{D}_1| = m$ and $h(\mathbb{D}_2) = n$. Let the elements of $\mathbb{D}_1$ be $d_1, \ldots, d_m$ such that if $d_i \sqsubset d_j$ then $i > j$. (I.e. we enumerate the elements in such a way that smaller elements always occur after larger elements, while the order of incomparable elements is not important. Such an enumeration can always be done for a finite partial order.)
   - We show that $h([\mathbb{D}_1 \rightarrow \mathbb{D}_2]) \leq mn$. For this consider any chain $\perp \sqsubset f_1 \sqsubset \ldots \sqsubset f_k$ in $[\mathbb{D}_1 \rightarrow \mathbb{D}_2]$. We show that $k \leq mn$. Define elements $x_0, x_1, \ldots, x_k \in {\mathbb{D}_2}^m$ as follows. $x_0 = (\perp, \ldots, \perp)$, and $x_i = (f_i(d_1), \ldots, f_i(d_m))$ for $1 \leq i \leq k$. Then clearly $x_0 \sqsubset x_1 \sqsubset \ldots \sqsubset x_k$ is a chain in ${\mathbb{D}_2}^m$. We know that $h({\mathbb{D}_2}^m) = mh(\mathbb{D}_2) = mn$. Hence $k \leq mn$.
   - Now we show that $h([\mathbb{D}_1 \rightarrow \mathbb{D}_2]) \geq mn$. Since $h(\mathbb{D}_2) = n$ we have a chain $\perp \sqsubset e_1 \sqsubset \ldots \sqsubset e_n$ in $\mathbb{D}_2$. Define functions $g_0, g_1, \ldots, g_{mn} : \mathbb{D}_1 \rightarrow \mathbb{D}_2$ as in the table.

g	$g(d_1)$	$g(d_2)$	$g(d_3)$	$\dots$	$g(d_m)$
g_0	$\perp$	$\perp$	$\perp$	$\dots$	$\perp$
g_1	e_1	$\perp$	$\perp$	$\dots$	$\perp$
g_2	e_2	$\perp$	$\perp$	$\dots$	$\perp$
		$\dots$			
g_n	e_n	$\perp$	$\perp$	$\dots$	$\perp$
g_{n+1}	e_n	e_1	$\perp$	$\dots$	$\perp$
g_{n+2}	e_n	e_2	$\perp$	$\dots$	$\perp$
		$\dots$			
g_{2n}	e_n	e_n	$\perp$	$\dots$	$\perp$
g_{2n+1}	e_n	e_n	e_1	$\dots$	$\perp$
g_{2n+2}	e_n	e_n	e_2	$\dots$	$\perp$
		$\dots$			
g_{3n}	e_n	e_n	e_n	$\dots$	$\perp$
		$\dots$			
g_{mn}	e_n	e_n	e_n	$\dots$	e_n

Because of the chose enumeration of $d_1, \dots, d_m$, each g_i is monotone, i.e. $g_i \in [\mathbb{D}_1 \to \mathbb{D}_2]$. Also it is clear that $g_0 \sqsubset g_1 \sqsubset \dots \sqsubset g_{mn}$. Hence $h([\mathbb{D}_1 \to \mathbb{D}_2]) \geq mn$.

$h([\mathbb{D}_1 \to \mathbb{D}_2]) = |\mathbb{D}_1| \cdot h(\mathbb{D}_2)$ where $[\mathbb{D}_1 \to \mathbb{D}_2]$ is the set of monotone functions $f : \mathbb{D}_1 \to \mathbb{D}_2$, and $|\mathbb{D}_1|$ is the cardinality of $\mathbb{D}_1$

2. The edge-effects defined in the lecture for computing available expressions as as follows:

$$\begin{array}{rcl} [\![;]\!]^\sharp A & = & A \\ [\![Pos(e)]\!]^\sharp A & = & A \cup \{e\} \\ [\![Neg(e)]\!]^\sharp A & = & A \cup \{e\} \\ [\![R = e;]\!]^\sharp A & = & (A \cup \{e\}) \setminus Expr_R \\ [\![R_1 = M[R_2];]\!]^\sharp A & = & A \setminus Expr_{R_1} \\ [\![M[R_1] = R_2;]\!]^\sharp A & = & A \end{array}$$

To deal with load operations define the new set of expressions $EXPR_R = Expr_R \cup \{M[R]\}$. Now the sets A can contain expressions of the form $M[R]$ to remember load operations already performed. The edge-effects for the edges ;, $Pos(e)$, $Neg(e)$ remain as before. The edge-effects for the remaining edges are changed as follows:

$$\begin{array}{rcl} [\![R = e;]\!]^\sharp A & = & (A \cup \{e\}) \setminus EXPR_R \\ [\![R_1 = M[R_2];]\!]^\sharp A & = & (A \cup M[R_2]) \setminus EXPR_{R_1} \\ [\![M[R_1] = R_2;]\!]^\sharp A & = & A \setminus Mem \end{array}$$

where Mem is the set of all expressions of the form $M[R]$. The explanation for the given translation of store operations is that once we modify the memory at any position, the value of any $M[R]$ already computed may now be different because it is possible that R points to the same memory location.

Transformation 1 in case of assignments (Abbildung 1) and conditions (Abbildung 2) is as before.

But now transformation 1 also deals with load operations (Abbildung 3).

Transformation 2 now deals with ordinary expressions, as well as expressions of the

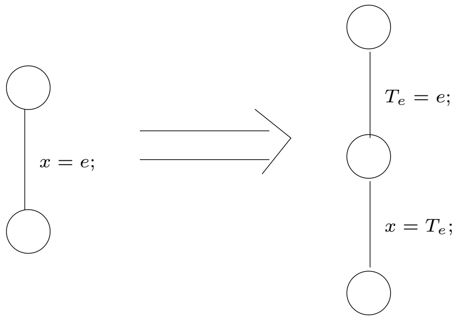

Abbildung 1: Transformation 1 for assignments

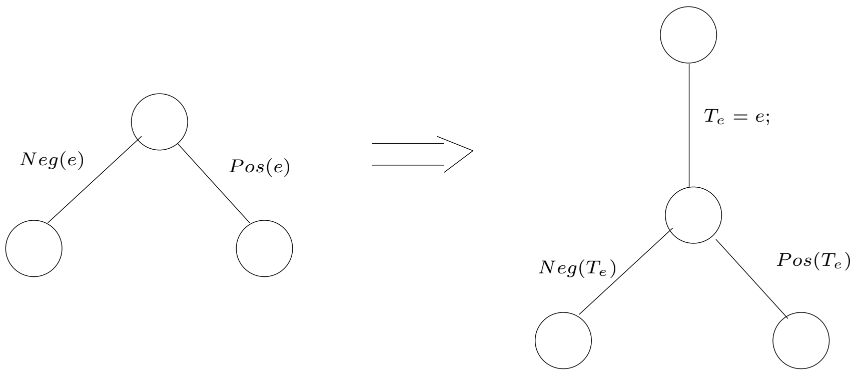

Abbildung 2: Transformation 1 for conditions

form $M[R]$ (Abbildung 4). Note that the set $\mathcal{A}[u]$ in Abbildung 4 may also have expressions of the form $M[R]$.

3. a) The required lattice is $\mathbb{D} = 2^{Vars}$ with $\sqsubseteq = \supseteq$ (the ordering is the superset relation).

   b) For every edge $k = (u, lab, v)$ we define $[\![k]\!]^\sharp = [\![lab]\!]^\sharp$ which transforms the dead variables at point v into dead variables at point u (backward propagation of information, as for the analysis of live variables).

$$\begin{array}{rcl}
[\![;]\!]^\sharp D & = & D \\
[\![Pos(e)]\!]^\sharp D & = & D \setminus Vars(e) \\
[\![Neg(e)]\!]^\sharp D & = & D \setminus Vars(e) \\
[\![x = e;]\!]^\sharp D & = & (D \cup \{x\}) \setminus Vars(e) \\
[\![R_1 = M[R_2];]\!]^\sharp D & = & (D \cup \{R_1\}) \setminus \{R_2\} \\
[\![M[R_1] = R_2;]\!]^\sharp & = & D \setminus \{R_1, R_2\}
\end{array}$$

   For a path $\pi = k_1 \dots k_r$ we have $[\![\pi]\!]^\sharp = [\![k_1]\!]^\sharp \circ \dots \circ [\![k_r]\!]^\sharp$. The set of dead variables at a point u is $\mathcal{D}^*[u] = \bigcap\{[\![\pi]\!]^\sharp Vars \mid \pi : u \longrightarrow^* stop\}$.
   The constraints that we will use for the analysis are of the form $\mathcal{D}[stop] \sqsupseteq Vars$ and $\mathcal{D}[u] \sqsupseteq [\![lab]\!]^\sharp \mathcal{D}[v]$ for every edge (u, lab, v) (recall that $\sqsupseteq = \subseteq$).
   To show the correctness of the above edge transformations, we show that $\mathcal{D}^*[u] = Vars \setminus \mathcal{L}^*[u]$. For this we show that for any sets $D = Vars \setminus L$, for any edge $k = (\_, lab, \_)$, if the edge transformation above gives $[\![lab]\!]^\sharp D = D_1$ and if the edge transformation defined in the lecture for live variables gives $[\![lab]\!]^\sharp L = L_1$ then we must have $D_1 = Vars \setminus L_1$.
   As example we show how to prove this for assignment statements. Assume

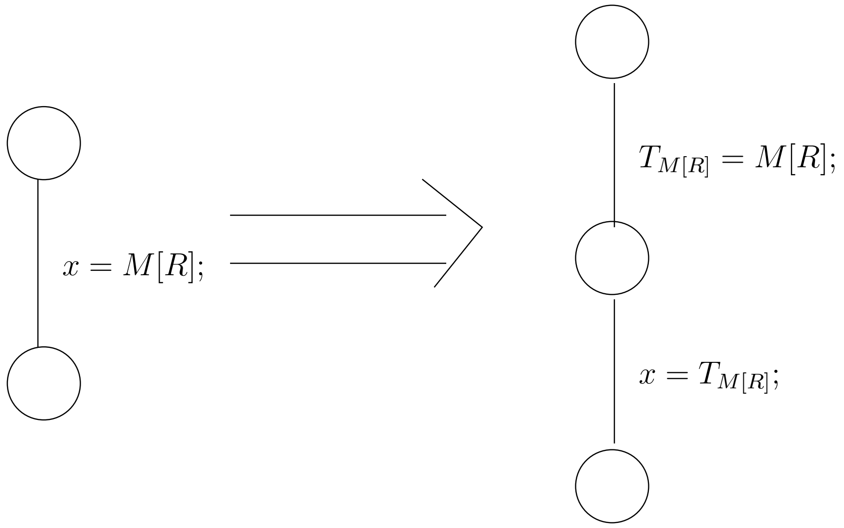

Abbildung 3: Transformation 1 for loads

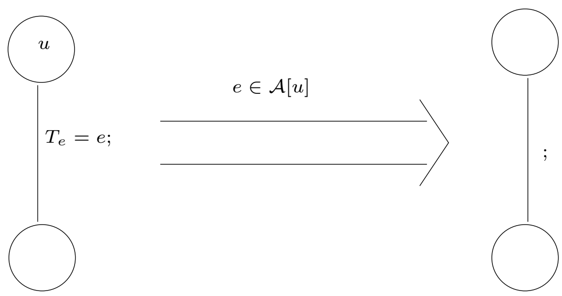

Abbildung 4: Transformation 2

that $D = Vars \setminus L$. Consider label *lab* of the form $x = e;$. The above edge transformation for dead variables gives $D_1 = (D \cup \{x\}) \setminus Vars(e)$. The edge transformation given in the lecture for live variables gives $L_1 = (L \setminus \{x\}) \cup Vars(e)$. Hence we have $Vars \setminus L_1 = (Vars \setminus (L \setminus \{x\})) \cap (Vars \setminus Vars(e)) = ((Vars \setminus L) \cup \{x\}) \cap (Vars \setminus Vars(e)) = (D \cup \{x\}) \cap (Vars \setminus Vars(e)) = (D \cup \{x\}) \setminus Vars(e) = D_1$.

c) For real deadness analysis we use the following edge transformations:

$$\begin{array}{rcl} [\![;]\!]^\sharp D & = & D \\ [\![Pos(e)]\!]^\sharp D & = & D \setminus Vars(e) \\ [\![Neg(e)]\!]^\sharp D & = & D \setminus Vars(e) \\ [\![x = e;]\!]^\sharp D & = & (D \cup \{x\}) \setminus (x \in D) ? \emptyset : Vars(e) \\ [\![R_1 = M[R_2];]\!]^\sharp D & = & (D \cup \{R_1\}) \setminus (R_1 \in D) ? \emptyset : \{R_2\} \\ [\![M[R_1] = R_2;]\!]^\sharp & = & D \setminus \{R_1, R_2\} \end{array}$$

Correctness of this analysis is shown as in the previous part.

4. The result of doing the analysis of available expressions (Homework 1) is shown in Abbildung 5.

   Now for each expression we compute which variables contain its value. This is shown in Abbildung 6.

   Then we apply the transformation 4 of the lecture to obtain the CFG in Abbildung 7.

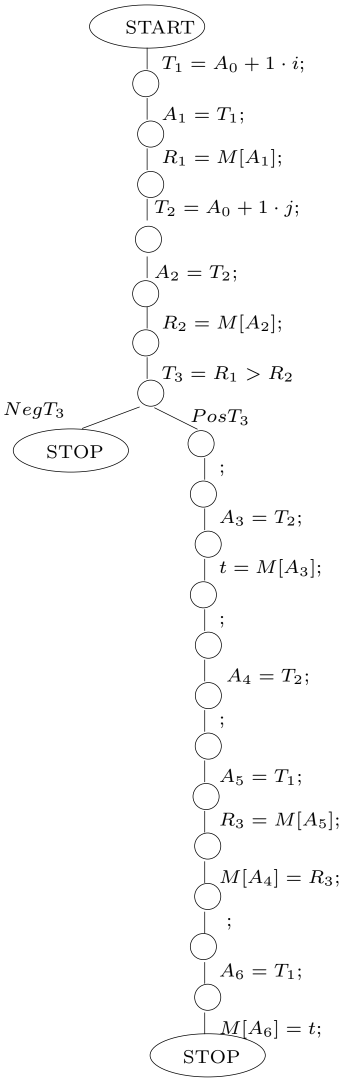

Abbildung 5: Avoiding unnecessary computations

The modifications are in red color.

Next we compute the set of live variables at each point. This is shown in Abbilding 8

Then we apply transformation 3 to eliminate redundant assignments. The result is shown in Figure 9. The result is satisfactory except that the redundant load operations are not saved, and there are too many empty edges.

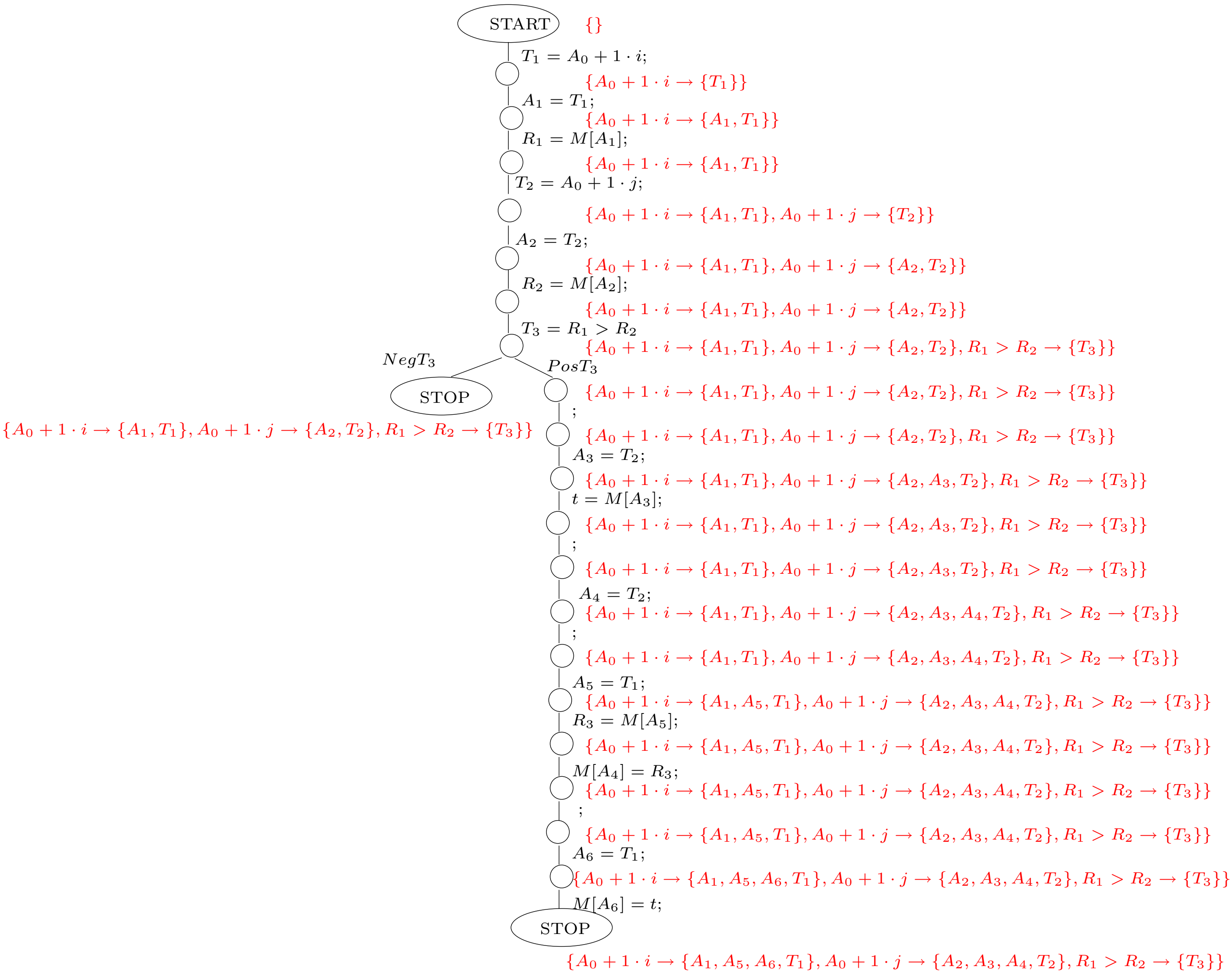

Abbildung 6: Computation of which variables contain which value.

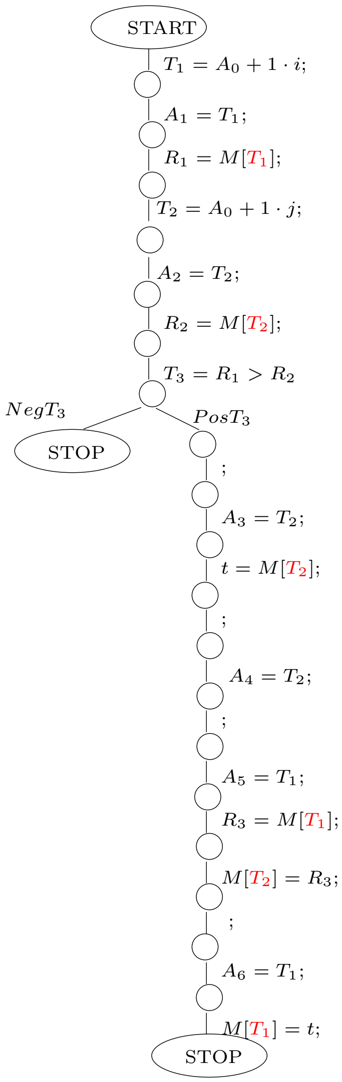

Abbildung 7: Transformation 4 for eliminating redundant moves

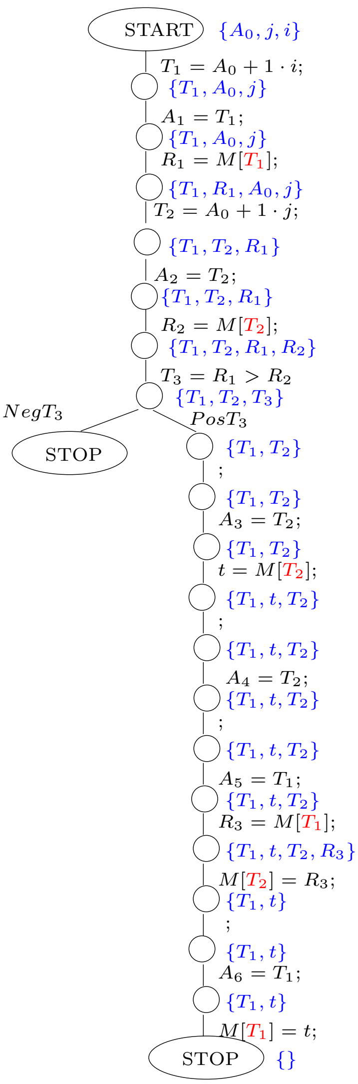

Abbildung 8: Computation of live variables

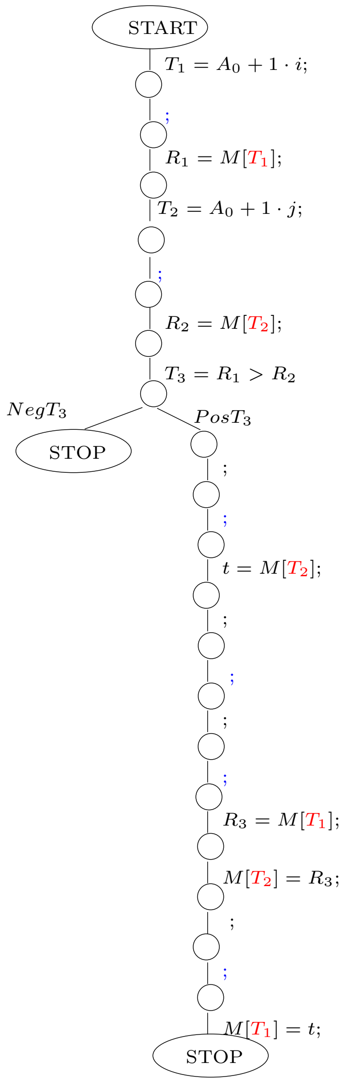

Abbildung 9: Transformation 3 for eliminating redundant assignments