Computer Science Theory

(Master Course)

Chapter 2:

Programs and Computable Functions

Mohammad Farshi
Department of Computer Science
Yazd University

1395-1

A Programming Language

- Computability theory is based on a specific programming language $\mathcal{L}$.

Assumptions and Definitions:

- numbers: positive integers.
- X_is: input variables
- Y: output variable
- Z_is: local variables
- The subscript 1 is often omitted (i.e. X is X_1).
- The output variable Y and the local variables Z_i initially have the value 0.
- A "program" of $\mathcal{L}$ will then consist of a list (i.e., a finite sequence) of instructions.
- Each line of a program can have a "label", which comes at the begining of instruction and within $[\,]$.

List of instructions of $\mathcal{L}$:

instruction	intrepretation
$V \leftarrow V + 1$	increment
$V \leftarrow V - 1$	decrement
If $V \neq 0$ GOTO L	conditional branch

List of instructions of $\mathcal{L}$:

instruction	intrepretation
$V \leftarrow V + 1$	increment
$V \leftarrow V - 1$	decrement
If $V \neq 0$ GOTO L	conditional branch

Example:

$$
\begin{array}{ll}
[A] & X \leftarrow X - 1 \\
 & Y \leftarrow Y + 1 \\
 & \text{IF } X \neq 0 \text{ GOTO } A
\end{array}
$$

Output of the Program:

- If $X \neq 0$, the output is x (i.e. Y has the value of X).
- If $X = 0$, the output is 1 (i.e. Y has the value of 1).

List of instructions of $\mathcal{L}$:

instruction	intrepretation
$V \leftarrow V + 1$	increment
$V \leftarrow V - 1$	decrement
If $V \neq 0$ GOTO L	conditional branch

Example:

$$
\begin{array}{ll}
[A] & X \leftarrow X - 1 \\
 & Y \leftarrow Y + 1 \\
 & \text{IF } X \neq 0 \text{ GOTO } A
\end{array}
$$

Output of the Program:

- If $X \neq 0$, the output is x (i.e. Y has the value of X).
- If $X = 0$, the output is 1 (i.e. Y has the value of 1).

List of instructions of $\mathcal{L}$:

instruction	intrepretation
$V \leftarrow V + 1$	increment
$V \leftarrow V - 1$	decrement
If $V \neq 0$ GOTO L	conditional branch

Example:

$$\begin{array}{ll} [A] & X \leftarrow X - 1 \\ & Y \leftarrow Y + 1 \\ & \text{IF } X \neq 0 \text{ GOTO } A \end{array}$$

Output of the Program:

- If $X \neq 0$, the output is x (i.e. Y has the value of X).
- If $X = 0$, the output is 1 (i.e. Y has the value of 1).

List of instructions of $\mathcal{L}$:

instruction	intrepretation
$V \leftarrow V + 1$	increment
$V \leftarrow V - 1$	decrement
If $V \neq 0$ GOTO L	conditional branch

Example:

$$
\begin{array}{ll}
[A] & X \leftarrow X - 1 \\
 & Y \leftarrow Y + 1 \\
 & \text{IF } X \neq 0 \text{ GOTO } A
\end{array}
$$

Output of the Program:

The program computes the function:

$$f(x) = \begin{cases} 1 & \text{If } x = 0 \\ x & \text{otherwise.} \end{cases}$$

Example:

$$
\begin{array}{ll}
[A] & X \leftarrow X - 1 \\
 & Y \leftarrow Y + 1 \\
 & \text{IF } X \neq 0 \text{ GOTO } A
\end{array}
$$

Output of the Program:

If one consider the above program to copy the value of X to Y, then it has a "bug": it does not work correctly when the input is 0.

Example:

$$\begin{array}{ll} [A] & X \leftarrow X - 1 \\ & Y \leftarrow Y + 1 \\ & \text{IF } X \neq 0 \text{ GOTO } A \end{array}$$

New Program:

$$\begin{array}{ll} [A] & \text{IF } X \neq 0 \text{ GOTO } B \\ & Z \leftarrow Z + 1 \\ & \text{IF } Z \neq 0 \text{ GOTO } E \\ [B] & X \leftarrow X - 1 \\ & Y \leftarrow Y + 1 \\ & Z \leftarrow Z + 1 \\ & \text{IF } Z \neq 0 \text{ GOTO } A \end{array}$$

Note: We use Z because we can not use instructions like GOTO A.

Macro expansion:

- We can use GOTO A in our programs, but we replace it with
  $Z \leftarrow Z + 1$
  IF $Z \neq 0$ GOTO A
  that satisfies in $\mathcal{L}$.
- Later, we will talk about the way we choose variable Z distinct to all other local variables.

New Program:

$$
\begin{array}{ll}
[A] & \text{IF } X \neq 0 \text{ GOTO } B \\
 & Z \leftarrow Z + 1 \\
 & \text{IF } Z \neq 0 \text{ GOTO } E \\
[B] & X \leftarrow X - 1 \\
 & Y \leftarrow Y + 1 \\
 & Z \leftarrow Z + 1 \\
 & \text{IF } Z \neq 0 \text{ GOTO } A
\end{array}
$$

Note: The program does copy the value of X into Y but it destroyes X (i.e. the value of X is 0 at the end).

Fixed Program:

$[A]$ IF $X \neq 0$ GOTO B
GOTO C
$[B]$ $X \leftarrow X - 1$
$Y \leftarrow Y + 1$
$Z \leftarrow Z + 1$
GOTO A
$[C]$ IF $Z \neq 0$ GOTO D
GOTO E
$[D]$ $Z \leftarrow Z - 1$
$X \leftarrow X + 1$
GOTO C

Note: Later, we will use $V \leftarrow V'$ with above code as a macro.

Macro for $V \leftarrow 0$:

$$
\begin{array}{ll}
[L] & V \leftarrow V - 1 \\
 & \text{IF } V \neq 0 \text{ GOTO } L
\end{array}
$$

Macro for $V \leftarrow V'$:

$$
\begin{array}{ll}
 & V \leftarrow 0 \\
[A] & \text{IF } V' \neq 0 \text{ GOTO } B \\
 & \text{GOTO } C \\
[B] & V' \leftarrow V' - 1 \\
 & V \leftarrow V + 1 \\
 & Z \leftarrow Z + 1 \\
 & \text{GOTO } A \\
[C] & \text{IF } Z \neq 0 \text{ GOTO } D \\
 & \text{GOTO } E \\
[D] & Z \leftarrow Z - 1 \\
 & V' \leftarrow V' + 1 \\
 & \text{GOTO } C
\end{array}
$$

Macro for $V \leftarrow V'$:

$$
\begin{array}{ll}
 & V \leftarrow 0 \\
[A] & \text{IF } V' \neq 0 \text{ GOTO } B \\
 & \text{GOTO } C \\
[B] & V' \leftarrow V' - 1 \\
 & V \leftarrow V + 1 \\
 & Z \leftarrow Z + 1 \\
 & \text{GOTO } A \\
[C] & \text{IF } Z \neq 0 \text{ GOTO } D \\
 & \text{GOTO } E \\
[D] & Z \leftarrow Z - 1 \\
 & V' \leftarrow V' + 1 \\
 & \text{GOTO } C
\end{array}
$$

Notes:

- Adding $Z \leftarrow 0$ at the beginning of the macro is unnecessary.
- Z should be distinct from all local variables of the main program.
- Same for Labels in the marco.
- Label E in the macro should replace by label of the instruction after the macro in the main program.

A Program for $f(x_1, x_2) = x_1 + x_2$:

$$
\begin{array}{ll}
 & Y \leftarrow X_1 \\
 & Z \leftarrow X_2 \\
[B] & \text{IF } Z \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z \leftarrow Z - 1 \\
 & Y \leftarrow Y + 1 \\
 & \text{GOTO } B
\end{array}
$$

A Program for $f(x_1, x_2) = x_1 \times x_2$:

$$
\begin{array}{ll}
 & Z_2 \leftarrow X_2 \\
[B] & \text{IF } Z_2 \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z_2 \leftarrow Z_2 - 1 \\
 & Z_1 \leftarrow X_1 + Y \\
 & Y \leftarrow Z_1 \\
 & \text{GOTO } B
\end{array}
$$

Note:

We used $Z_1 \leftarrow X_1 + Y$
$Y \leftarrow Z_1$
instead of $Y \leftarrow X_1 + Y$ since in the summation macro all 3 variables should be distinct. If not, the result is not correct.

A Programming Language

Some Examples of Programs

A Program for $f(x_1, x_2) = x_1 + x_2$:

$$
\begin{array}{ll}
 & Y \leftarrow X_1 \\
 & Z \leftarrow X_2 \\
[B] & \text{IF } Z \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z \leftarrow Z - 1 \\
 & Y \leftarrow Y + 1 \\
 & \text{GOTO } B
\end{array}
$$

A Program for $f(x_1, x_2) = x_1 \times x_2$:

$$
\begin{array}{ll}
 & Z_2 \leftarrow X_2 \\
[B] & \text{IF } Z_2 \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z_2 \leftarrow Z_2 - 1 \\
 & Z_1 \leftarrow X_1 + Y \\
 & Y \leftarrow Z_1 \\
 & \text{GOTO } B
\end{array}
$$

Note:

We used $Z_1 \leftarrow X_1 + Y$
$Y \leftarrow Z_1$
instead of $Y \leftarrow X_1 + Y$ since in the summation macro all 3 variables should be distinct. If not, the result is not correct.

A Program for $f(x_1, x_2) = x_1 + x_2$:

$$
\begin{array}{ll}
 & Y \leftarrow X_1 \\
 & Z \leftarrow X_2 \\
[B] & \text{IF } Z \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z \leftarrow Z - 1 \\
 & Y \leftarrow Y + 1 \\
 & \text{GOTO } B
\end{array}
$$

A Program for $f(x_1, x_2) = x_1 \times x_2$:

$$
\begin{array}{ll}
 & Z_2 \leftarrow X_2 \\
[B] & \text{IF } Z_2 \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z_2 \leftarrow Z_2 - 1 \\
 & Z_1 \leftarrow X_1 + Y \\
 & Y \leftarrow Z_1 \\
 & \text{GOTO } B
\end{array}
$$

Note:

We used $Z_1 \leftarrow X_1 + Y$
$Y \leftarrow Z_1$
instead of $Y \leftarrow X_1 + Y$ since in the summation macro all 3 variables should be distinct. If not, the result is not correct.

A Program for $f(x_1, x_2) = x_1 + x_2$:

$$
\begin{array}{ll}
 & Y \leftarrow X_1 \\
 & Z \leftarrow X_2 \\
[B] & \text{IF } Z \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z \leftarrow Z - 1 \\
 & Y \leftarrow Y + 1 \\
 & \text{GOTO } B
\end{array}
$$

Macro for $Y \leftarrow X_1 + Y$:

$$
\begin{array}{ll}
 & Y \leftarrow X_1 \\
 & Z \leftarrow Y \\
[B] & \text{IF } Z \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z \leftarrow Z - 1 \\
 & Y \leftarrow Y + 1 \\
 & \text{GOTO } B
\end{array}
$$

Note:

It computes $2x_1$ instead of $x_1 + y$.

A Program for $f(x_1, x_2) = x_1 + x_2$:

$$
\begin{array}{ll}
 & Y \leftarrow X_1 \\
 & Z \leftarrow X_2 \\
[B] & \text{IF } Z \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z \leftarrow Z - 1 \\
 & Y \leftarrow Y + 1 \\
 & \text{GOTO } B
\end{array}
$$

Macro for $Y \leftarrow X_1 + Y$:

$$
\begin{array}{ll}
 & Y \leftarrow X_1 \\
 & Z \leftarrow Y \\
[B] & \text{IF } Z \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z \leftarrow Z - 1 \\
 & Y \leftarrow Y + 1 \\
 & \text{GOTO } B
\end{array}
$$

Note:

It computes $2x_1$ instead of $x_1 + y$.

A Program for $f(x_1, x_2) = x_1 \times x_2$:

$$
\begin{array}{ll}
 & Z_2 \leftarrow X_2 \\
[B] & \text{IF } Z_2 \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z_2 \leftarrow Z_2 - 1 \\
 & Z_1 \leftarrow X_1 + Y \\
 & Y \leftarrow Z_1 \\
 & \text{GOTO } B
\end{array}
$$

A Program for $f(x_1, x_2) = x_1 + x_2$:

$$
\begin{array}{ll}
 & Y \leftarrow X_1 \\
 & Z \leftarrow X_2 \\
[B] & \text{IF } Z \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z \leftarrow Z - 1 \\
 & Y \leftarrow Y + 1 \\
 & \text{GOTO } B
\end{array}
$$

The program with macro expansion:

$$
\begin{array}{ll}
 & Z_2 \leftarrow X_2 \\
[B] & \text{IF } Z_2 \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z_2 \leftarrow Z_2 - 1 \\
 & Z_1 \leftarrow X_1 \\
 & Z_3 \leftarrow Y \\
[B_2] & \text{IF } Z_3 \neq 0 \text{ GOTO } A_2 \\
 & \text{GOTO } E_2 \\
[A_2] & Z_3 \leftarrow Z_3 - 1 \\
 & Z_1 \leftarrow Z_1 + 1 \\
 & \text{GOTO } B_2 \\
[E_2] & Y \leftarrow Z_1 \\
 & \text{GOTO } B
\end{array}
$$

A Program for $f(x_1, x_2) = x_1 \times x_2$:

$$
\begin{array}{ll}
 & Z_2 \leftarrow X_2 \\
[B] & \text{IF } Z_2 \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z_2 \leftarrow Z_2 - 1 \\
 & Z_1 \leftarrow X_1 + Y \\
 & Y \leftarrow Z_1 \\
 & \text{GOTO } B
\end{array}
$$

A Program for $f(x_1, x_2) = x_1 + x_2$:

$$
\begin{array}{ll}
 & Y \leftarrow X_1 \\
 & Z \leftarrow X_2 \\
[B] & \text{IF } Z \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z \leftarrow Z - 1 \\
 & Y \leftarrow Y + 1 \\
 & \text{GOTO } B
\end{array}
$$

The program with macro expansion:

$$
\begin{array}{ll}
 & Z_2 \leftarrow X_2 \\
[B] & \text{IF } Z_2 \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z_2 \leftarrow Z_2 - 1 \\
 & Z_1 \leftarrow X_1 \\
 & Z_3 \leftarrow Y \\
[B_2] & \text{IF } Z_3 \neq 0 \text{ GOTO } A_2 \\
 & \text{GOTO } E_2 \\
[A_2] & Z_3 \leftarrow Z_3 - 1 \\
 & Z_1 \leftarrow Z_1 + 1 \\
 & \text{GOTO } B_2 \\
[E_2] & Y \leftarrow Z_1 \\
 & \text{GOTO } B
\end{array}
$$

A Program for $f(x_1, x_2) = x_1 \times x_2$:

$$
\begin{array}{ll}
 & Z_2 \leftarrow X_2 \\
[B] & \text{IF } Z_2 \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z_2 \leftarrow Z_2 - 1 \\
 & Z_1 \leftarrow X_1 + Y \\
 & Y \leftarrow Z_1 \\
 & \text{GOTO } B
\end{array}
$$

A Program for $f(x_1, x_2) = x_1 + x_2$:

$$
\begin{array}{ll}
 & Y \leftarrow X_1 \\
 & Z \leftarrow X_2 \\
[B] & \text{IF } Z \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z \leftarrow Z - 1 \\
 & Y \leftarrow Y + 1 \\
 & \text{GOTO } B
\end{array}
$$

The program with macro expansion:

$$
\begin{array}{ll}
 & Z_2 \leftarrow X_2 \\
[B] & \text{IF } Z_2 \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & Z_2 \leftarrow Z_2 - 1 \\
 & Z_1 \leftarrow X_1 \\
 & Z_3 \leftarrow Y \\
[B_2] & \text{IF } Z_3 \neq 0 \text{ GOTO } A_2 \\
 & \text{GOTO } E_2 \\
[A_2] & Z_3 \leftarrow Z_3 - 1 \\
 & Z_1 \leftarrow Z_1 + 1 \\
 & \text{GOTO } B_2 \\
[E_2] & Y \leftarrow Z_1 \\
 & \text{GOTO } B
\end{array}
$$

A program for $f(x_1, x_2) = x_1 - x_2$:

$$
\begin{array}{ll}
 & Y \leftarrow X_1 \\
 & Z \leftarrow X_2 \\
[C] & \text{IF } Z \neq 0 \text{ GOTO } A \\
 & \text{GOTO } E \\
[A] & \text{IF } Y \neq 0 \text{ GOTO } B \\
 & \text{GOTO } A \\
[B] & Y \leftarrow Y - 1 \\
 & Z \leftarrow Z - 1 \\
 & \text{GOTO } C
\end{array}
$$

Notes:

If $x_1 < x_2$, then the program will never terminate. So it computes

$$f(x_1, x_2) = \begin{cases} x_1 - x_2 & \text{If } x_1 \geq x_2 \\ \text{undefined} & \text{If } x_1 < x_2. \end{cases}$$

Syntax

Definitions and assumptions:

- $\mathcal{L}$: Language
- X_is: Input variables
- Z_is: Local variables
- Y: Output variable
- A_i, B_i, C_i, D_i: labels
- statements: (V: any variable, L: any label)
  $V \leftarrow V + 1$
  $V \leftarrow V - 1$
  $V \leftarrow V$ (dummy statement)
  IF $V \neq 0$ GOTO L
- Instructions: Either an statement or a label followed by statement.
- program: a list (i.e., a finite sequence) of instructions.
- state of a program: value of all the variables of the program.

Definitions and assumptions:

- Instructions: Either an statement or a label followed by statement.
- program: a list (i.e., a finite sequence) of instructions.
- state of a program: value of all the variables of the program.
- snapshot of a program of length n: status of the program in each step. (σ, i): status σ at the point that ith instruction is about to be executed.
- $(n+1, \sigma)$: Terminal (n is the length of the program).
- $(i+1, \sigma)$ is the seccessor of (i, σ).

Seccessor of (i, σ):

If ith instruction is	successor
$V \leftarrow V + 1$	$(i{+}1, \tau), \tau$ is σ except that the value of V incremented by 1.
$V \leftarrow V - 1$	$(i{+}1, \tau), \tau$ is σ except that the value of V decremented by 1.
$V \leftarrow V$	$(i + 1, \sigma)$.
IF $V \neq 0$ GOTO L	If $V = 0$, then $(i + 1, \sigma)$. If $V \neq 0$, then if there is instruction with label L and j is the number of it, (j, σ), and otherwise $(n + 1, \sigma)$ (stop).

A computation of a program

a sequence (i.e., a list) $s_1, s_2, \ldots, s_k$ of snapshots of the program such that s_{i+1} is the successor of s_i, for $i = 1, 2, \ldots, k - 1$ and s_k is terminal.

Syntax

Note:

We have not forbidden a program to contain more than one instruction having the same label. But only the first occurance of the label take effect, i.e. the following programs are the same:

$$\begin{array}{ll} [A] & X \leftarrow X - 1 \\ & \text{IF } X \neq 0 \text{ GOTO } A \\ [A] & X \leftarrow X + 1 \end{array} \qquad \begin{array}{ll} [A] & X \leftarrow X - 1 \\ & \text{IF } X \neq 0 \text{ GOTO } A \\ & X \leftarrow X + 1 \end{array}$$

Computable Functions

- Consider a function of m variables.
- Let P be a program and $r_1, \ldots, r_m$ be m inputs.
- initial state σ: $X_1 = r_1, \ldots, X_m = r_m, Y = 0$ plus $V = 0$ for all other variables.
- $(1, \sigma)$: initial snapshot.
- **Case 1:** $\exists$ computation $s_1, s_2, \ldots, s_k$ of P beginning with $s_1 = (1, \sigma)$. Then $f_P^{(m)}(r_1, r_2, \ldots, r_m)$ is the value of Y at the (terminal) snapshot s_k.
- **Case 2:** There is no such computation; i.e., the sequence of snapshots is infinite. In this case $f_P^{(m)}(r_1, r_2, \ldots, r_m)$ is undefined.

Computable Functions

Program: $f(x) = x$

$$
\begin{array}{lll}
[A] & \text{IF } X \neq 0 \text{ GOTO } B & (1) \\
 & Z \leftarrow Z + 1 & (2) \\
 & \text{IF } Z \neq 0 \text{ GOTO } E & (3) \\
[B] & X \leftarrow X - 1 & (4) \\
 & Y \leftarrow Y + 1 & (5) \\
 & Z \leftarrow Z + 1 & (6) \\
 & \text{IF } Z \neq 0 \text{ GOTO } A & (7)
\end{array}
$$

Assumptions:

- If # inputs is less than # input variables, it assigns 0 to the rest of input variables .
- If it is more, the rest are ignored.

Computation for input r

#	Status		
	X	Y	Z
1	r	0	0
4	r	0	0
5	$r-1$	0	0
6	$r-1$	1	0
7	$r-1$	1	1
1	$r-1$	1	1
$\vdots$			
1	0	r	r
2	0	r	r
3	0	r	$r+1$
8	0	r	$r+1$

Computable Functions

Program: $f(x) = x$

$$
\begin{array}{lll}
[A] & \text{IF } X \neq 0 \text{ GOTO } B & (1) \\
 & Z \leftarrow Z + 1 & (2) \\
 & \text{IF } Z \neq 0 \text{ GOTO } E & (3) \\
[B] & X \leftarrow X - 1 & (4) \\
 & Y \leftarrow Y + 1 & (5) \\
 & Z \leftarrow Z + 1 & (6) \\
 & \text{IF } Z \neq 0 \text{ GOTO } A & (7)
\end{array}
$$

Assumptions:

- If # inputs is less than # input variables, it assigns 0 to the rest of input variables .
- If it is more, the rest are ignored.

Computation for input r

	Status		
#	X	Y	Z
1	r	0	0
4	r	0	0
5	$r-1$	0	0
6	$r-1$	1	0
7	$r-1$	1	1
1	$r-1$	1	1
$\vdots$			
1	0	r	r
2	0	r	r
3	0	r	$r+1$
8	0	r	$r+1$

Computable Functions

Definitions:

- For any program P and any m, the function $F_P^{(m)}(x_1, \ldots, x_m)$ is said to be computed by P.
- A given partial function g is said to be partially computable or partial recursive if it is computed by some program.
- A given total function g is said to be computable or recursive if it is computed by some program.
- So far, it is shown that x, $x + y$, $x \times y$ and $x - y$ is (partial) computable.
- Computability theory (also called recursion theory) studies the class of partially computable functions.

More about Macros

- Let $f(x_1, \ldots, x_n)$ be some partially computable function computed by P.
- The variables that occur in P are included in $Y, X_1, \ldots, X_n, Z_1, \ldots, Z_k$.
- The labels that occur in P are all included in $E, A_1, \ldots, A_\ell$.
- for each instruction IF $V \neq 0$ GOTO A_i there is an instruction labeled A_i.
- E is the only "exit" label.
- $P = P(Y, X_1, \ldots, X_n, Z_1, \ldots, Z_k, E, A_1, \ldots, A_\ell)$
- We replace all variables and labels by others: $Q_m =$ $P(Z_m, Z_{m+1}, \ldots, Z_{m+n}, Z_{m+n+1}, Z_{m+n+k}, E_m, A_{m+1}, \ldots, A_{m+\ell})$

More about Macros

- $P = P(Y, X_1, \ldots, X_n, Z_1, \ldots, Z_k, E, A_1, \ldots, A_\ell)$
- We replace all variables and labels by others:
  $Q_m =$
  $P(Z_m, Z_{m+1}, \ldots, Z_{m+n}, Z_{m+n+1}, Z_{m+n+k}, E_m, A_{m+1}, \ldots, A_{m+\ell})$

We replace $W \leftarrow f(V_1, \ldots, V_n)$ by

$$
\begin{array}{ll}
 & Z_m \leftarrow 0 \\
 & Z_{m+1} \leftarrow V_1 \\
 & \vdots \\
 & Z_{m+n} \leftarrow V_n \\
 & Z_{m+n+1} \leftarrow 0 \\
 & \vdots \\
 & Z_{m+n+k} \leftarrow 0 \\
 & Q_m \\
[E_m] & W \leftarrow Z_m
\end{array}
$$

Notes:

- The variables and labels used in Q_m should be unique.
- The expansion should sets output and local variables to 0.

- $P = P(Y, X_1, \ldots, X_n, Z_1, \ldots, Z_k, E, A_1, \ldots, A_\ell)$
- We replace all variables and labels by others:
  $Q_m =$
  $P(Z_m, Z_{m+1}, \ldots, Z_{m+n}, Z_{m+n+1}, Z_{m+n+k}, E_m, A_{m+1}, \ldots, A_{m+\ell})$

We replace $W \leftarrow f(V_1, \ldots, V_n)$ by

$$
\begin{array}{ll}
 & Z_m \leftarrow 0 \\
 & Z_{m+1} \leftarrow V_1 \\
 & \vdots \\
 & Z_{m+n} \leftarrow V_n \\
 & Z_{m+n+1} \leftarrow 0 \\
 & \vdots \\
 & Z_{m+n+k} \leftarrow 0 \\
 & Q_m \\
[E_m] & W \leftarrow Z_m
\end{array}
$$

Notes:

- The variables and labels used in Q_m should be unique.
- The expansion should sets output and local variables to 0.

Note:

- If one of the macros of the program is undefined, then the program will not terminate.
  Example:
  $Z \leftarrow X_1 - X_2$
  $Y \leftarrow Z + X_3$
  does not work if $x_1 < x_2$.

Macros of the form IF $P(V_1, \ldots, V_n)$ GOTO L

- Let $P(x_1, \ldots, x_n)$ be any computable predicate (its output is 1 (=TRUE) or 0(=FALSE).
- The macro expansion of

  IF $P(V_1, \ldots, V_n)$ GOTO L

  is simply

  $Z \leftarrow P(V_1, \ldots, V_n)$
  IF $Z \neq 0$ GOTO L
- Note that, now we know how to expand

  $Z \leftarrow P(V_1, \ldots, V_n)$

Example: IF $V = 0$ GOTO L

- Note that it is not an instruction in the language.
- Define

$$P(x) = \begin{cases} \text{TRUE} & \text{if } x = 0 \\ \text{FALSE} & \text{otherwise} \end{cases}$$

- The following program computes $P(x)$:

$$\text{IF } X \neq 0 \text{ GOTO E}$$
$$Y \leftarrow Y + 1$$

End of Chapter 2.