The unary loop-computable functions are recursively enumerable.

1. The class of all loop programs is r.e.

Proof: on the slides, one can find a grammar (context-free) which generates the class of all loop programs.

2. Problem:

- Not every loop program computes a function.
- From the form of a loop program one cannot see if the program computes a function or not.

(And one cannot see if the program computes a unary function: we would need a way of checking whether for every value of register $x_1$, at the end the values stored in registers $x_3, \ldots x_n$ are 0 or not.)

3. Idea:

Change every loop program $P$ to obtain a loop program $\hat{P}$ which computes a function $f_P : N \rightarrow N$.

Such that if $P$ computes a function $f_P : N \rightarrow N$ then $\hat{P}$ computes the same function (i.e., $f_P = f_{\hat{P}}$).

Construction: $P$ loop program using registers $\{x_1, \ldots x_n\}$.

Let $x_n$ be a new register name.

$\hat{P} := \left\{ \begin{array}{l}
\text{x}_n := \text{x}_1; \\
\hat{P}; \\
\text{x}_1 := \text{x}_n; \text{x}_n := 0; \\
\text{x}_2 := 0; \text{x}_2 := 0; \ldots \ldots \\
x_3 := 0 \text{ x_til all register} + \text{x}_2 \\
\text{x}_3 \ldots \text{x}_n, \text{x}_n := 0
\end{array} \right.$

4. Show that the class of unary loop computable functions is r.e.

The $i$-th loop computable function (unary) $f_i : N \rightarrow N$ is

$f_i = \{ \begin{array}{l}
te_P \text{ if } i \text{ is the Gödel number of the loop program } P \\
g_0 \text{ otherwise}
\end{array} \right.$

When $g_0 : N \rightarrow N$ is defined by $g_0(n) = 0$ for all $n \in N$.

A Turing Machine $M_\mathbb{N}$ can recursively enumerate $\{f_1, f_2, \ldots \}$.

[EN] $i \leftrightarrow i$ encodes correct loop program $P \in \mathbb{N}$.