

E2. GOTO Computability & Comparsion to Turing Computability

GOTO Programs

E2.1 GOTO Programs

Theory of Computer Science May 24, 2023 — E2. GOTO Computability & Comparsion to Turing Computability



E2. GOTO Computability & Comparsion to Turing Computability

GOTO Programs

Motivation

We already know: WHILE programs are strictly more powerful than LOOP programs.

How do DTMs relate to LOOP and WHILE programs?

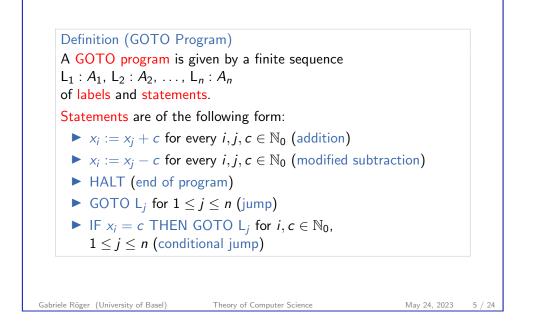
To answer this question, we make a detour over one more programming formalism, GOTO programs.

We will establish:

- ▶ WHILE programs are at least as powerful as GOTO programs.
- DTMs are at least as powerful as WHILE programs.
- GOTO programs are at least as powerful as DTMs.
- $\Rightarrow \mbox{ Turing-computable} = \mbox{ WHILE-computable} = \\ \mbox{ GOTO-computable}$



GOTO Programs: Syntax



E2. GOTO Computability & Comparsion to Turing Computability

GOTO-Computable Functions Definition (GOTO-Computable) A function $f : \mathbb{N}_0^k \to \mathbb{N}_0$ is called GOTO-computable if a GOTO program that computes f exists.

E2. GOTO Computability & Comparsion to Turing Computability

GOTO Programs: Semantics

Definition (Semantics of GOTO Programs)

- Input, output and variables work exactly as in LOOP and WHILE programs.
- Addition and modified subtraction work exactly as in LOOP and WHILE programs.
- Execution begins with the statement A_1 .
- After executing A_i, the statement A_{i+1} is executed. (If i = n, execution finishes.)
- exceptions to the previous rule:
 - ► HALT stops the execution of the program.
 - ► After GOTO L_j execution continues with statement A_j.
 - After IF x_i = c THEN GOTO L_j execution continues with A_j if variable x_i currently holds the value c.

Gabriele Röger (University of Basel)

Theory of Computer Science

May 24, 2023 6 / 24

E2. GOTO Computability & Comparsion to Turing Computability

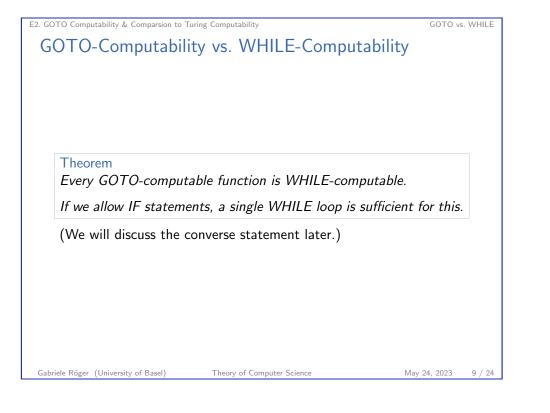
GOTO vs. WHILE

E2.2 GOTO vs. WHILE

Gabriele Röger (University of Basel)

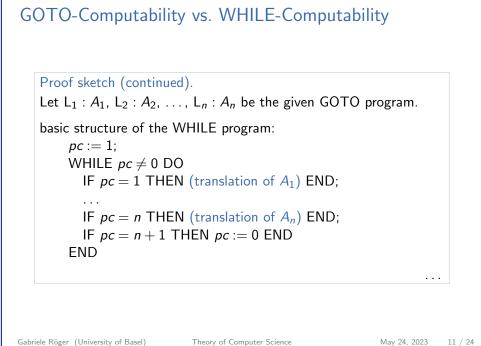
GOTO Programs

GOTO Programs



E2. GOTO Computability & Comparsion to Turing Computability

GOTO vs. WHILE



E2. GOTO Computability & Comparsion to Turing Computability

GOTO-Computability vs. WHILE-Computability

Proof sketch.

Given any GOTO program, we construct an equivalent WHILE program with a single WHILE loop (and IF statements).

Ideas:

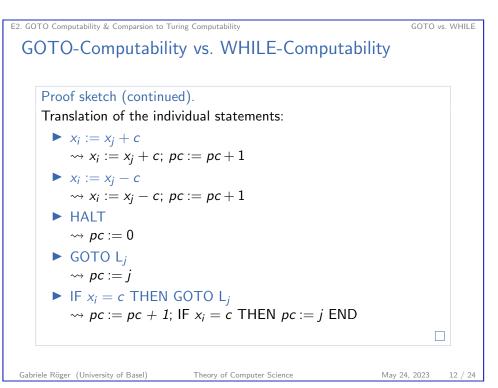
- Use a fresh variable to store the number of the statement to be executed next.
 - \rightsquigarrow The variable of course has the form x_i , but for readability we write it as *pc* for "program counter".
- ► GOTO is simulated as an assignment to *pc*.
- ▶ If *pc* has the value 0, the program terminates.

Gabriele Röger (University of Basel)

Theory of Computer Science

```
May 24, 2023 10 / 24
```

. . .



E2.3 WHILE vs. Turing

Gabriele Röger (University of Basel)

Theory of Computer Science

E2. GOTO Computability & Comparsion to Turing Computability

WHILE vs. Turing

. . .

15 / 24

13 / 24

May 24, 2023

WHILE-Computability vs. Turing-Computability

Proof sketch.

Given any WHILE program, we construct an equivalent deterministic Turing machine.

Let x_1, \ldots, x_k be the input variables of the WHILE program, and let x_0, \ldots, x_m be all used variables.

General ideas:

- The DTM simulates the individual execution steps of the WHILE program.
- Before and after each WHILE program step the tape contains the word bin(n₀)#bin(n₁)#...#bin(n_m), where n_i is the value of WHILE program variable x_i.
- It is enough to simulate "minimalistic" WHILE programs (x_i := x_i + 1, x_i := x_i - 1, composition, WHILE loop).

E2. GOTO Computability & Comparsion to Turing Computability

WHILE-Computability vs. Turing-Computability

Theorem

Every WHILE-computable function is Turing-computable.

(We will discuss the converse statement later.)

Gabriele Röger (University of Basel)

WHILE vs. Turing

14 / 24

May 24, 2023

WHILE vs. Turing

WHILE-Computability vs. Turing-Computability

Theory of Computer Science

Proof sketch (continued).

E2. GOTO Computability & Comparsion to Turing Computability

The DTM consists of three sequential parts:

- ► initialization:
 - Write 0# in front of the used part of the tape (move existing content 2 positions to the right).
 - (m-k) times, write #0 behind the used part of the tape.
- execution:

Simulate the WHILE program (see next slide).

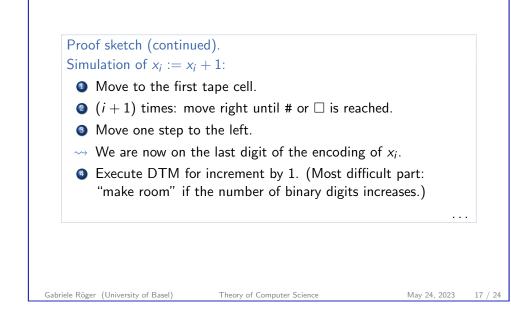
- clean-up:
 - ▶ Replace all symbols starting from the first # with □, then move to the first tape cell.

16 / 24

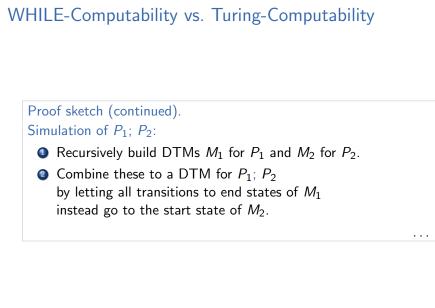
. . .



WHILE-Computability vs. Turing-Computability

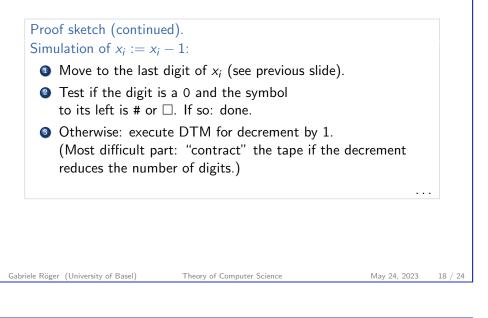


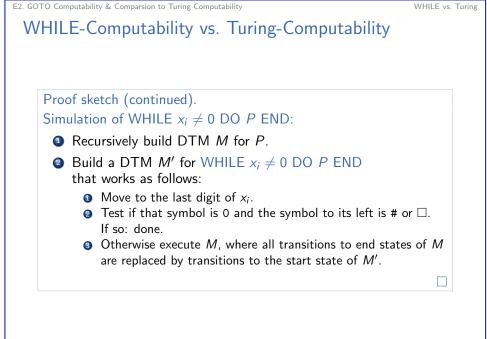
E2. GOTO Computability & Comparsion to Turing Computability



E2. GOTO Computability & Comparsion to Turing Computability

WHILE-Computability vs. Turing-Computability





WHILE vs. Turing

WHILE vs. Turing

WHILE vs. Turing

E2.4 Turing vs. GOTO

Gabriele Röger (University of Basel)

Theory of Computer Science

E2. GOTO Computability & Comparsion to Turing Computability

Turing vs. GOTO

21 / 24

May 24, 2023

Final Result

Corollary

Let $f : \mathbb{N}_0^k \to_p \mathbb{N}_0$ be a function.

The following statements are equivalent:

- ▶ f is Turing-computable.
- ▶ f is WHILE-computable.
- ▶ f is GOTO-computable.

Moreover:

- Every LOOP-computable function is Turing-/WHILE-/GOTO-computable.
- The converse is not true in general.

E2. GOTO Computability & Comparsion to Turing Computability

$Turing-Computability \ vs. \ GOTO-Computability$

Theorem (Turing-Computability vs. GOTO-Computability) Every Turing-computable numerical function is GOTO-computable.

Proof sketch.

- Represent TM configuration (x, q, y) with three numbers, one for x, one for q and one for y.
- The tape content can be accessed and modified using DIV and MOD operations, which are GOTO-computable.
- For each transition, implement the corresponding modification of the configuration in terms of the three numbers.
- Use "IF ... GOTO" statements for each tape symbol and state to jump to the implementation of the corresponding transition.

Theory of Computer Science

Gabriele Röger (University of Basel)

<page-header><page-header><page-header><section-header><section-header><list-item><list-item><list-item><list-item><list-item>

May 24, 2023

22 / 24