COMP30026 Models of Computation

Lecture 16: PDA-CFG Equivalence and More CFL Properties

Mak Nazecic-Andrlon and William Umboh

Semester 2, 2024

CFLs Have PDAs as Recognisers

Theorem

A language is context-free if and only if it is recognised by a PDA.

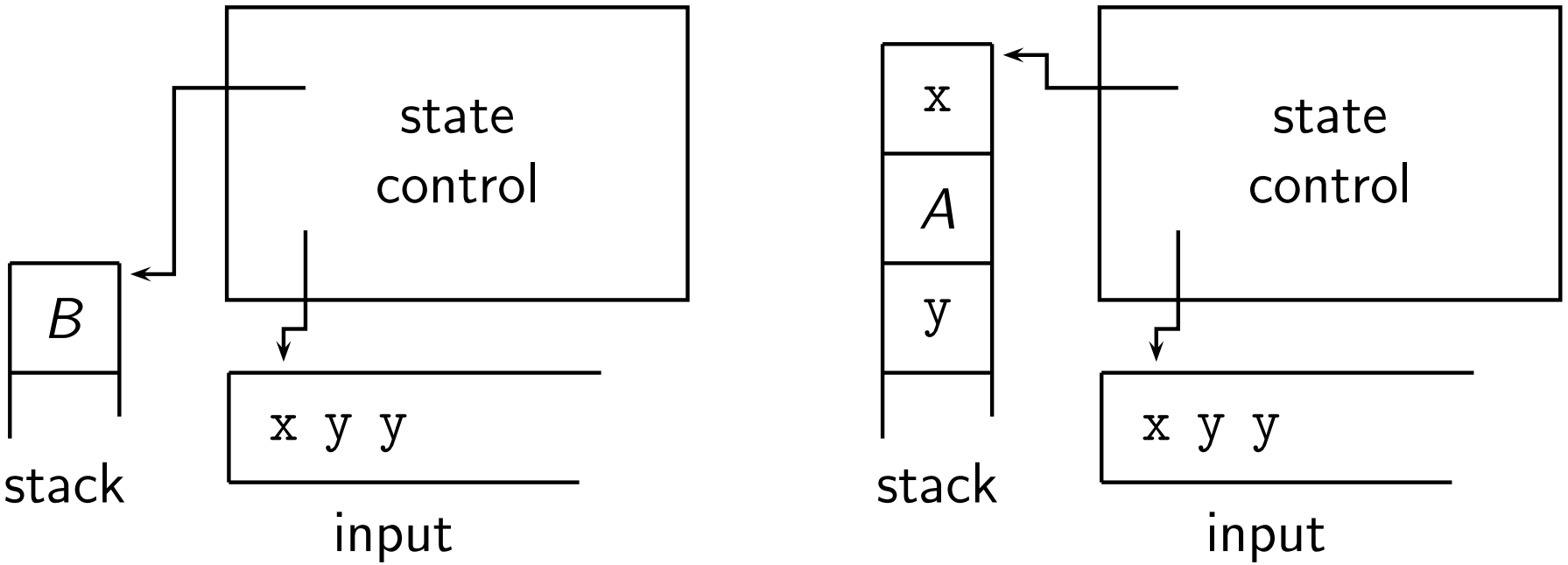
We will only study the forward direction.

(Reverse direction in textbook, if interested.)

From CFGs to PDAs

Proof idea: use nondeterminism to guess a derivation.

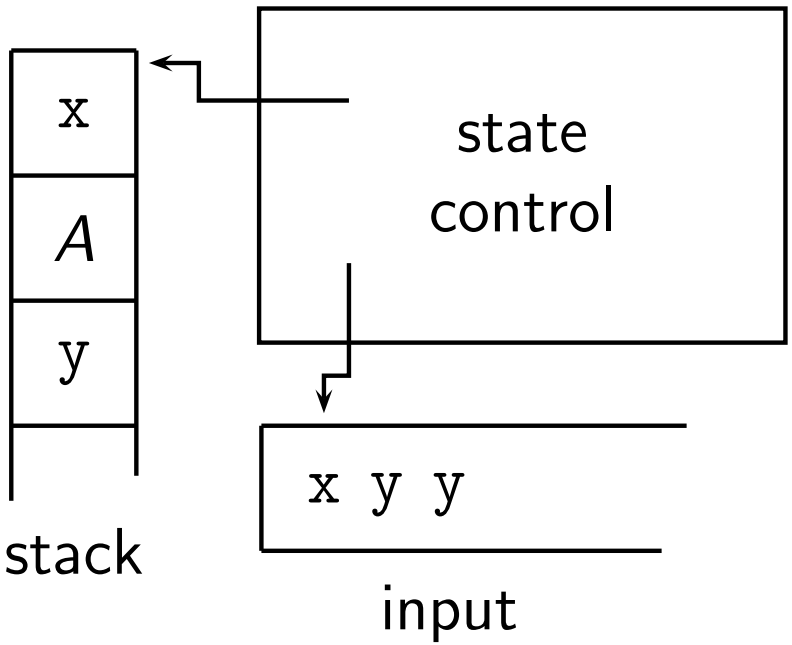
Say $B \to xAy$ is a rule. If B is on top of the stack, we can pop B and push y, A, and x, in that order.

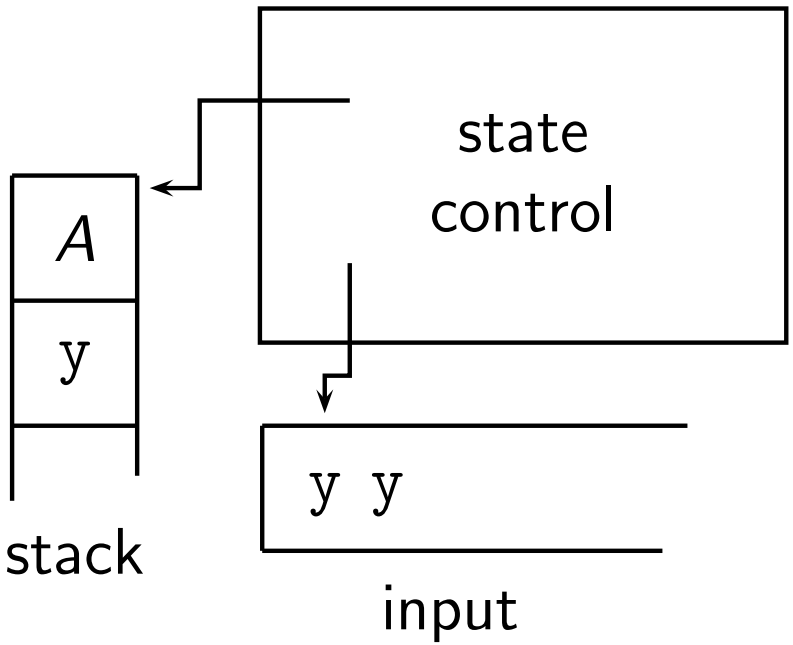


From CFGs to PDAs

Proof idea: use nondeterminism to guess a derivation.

Terminal x on top of stack? Must pop x and consume x from input.





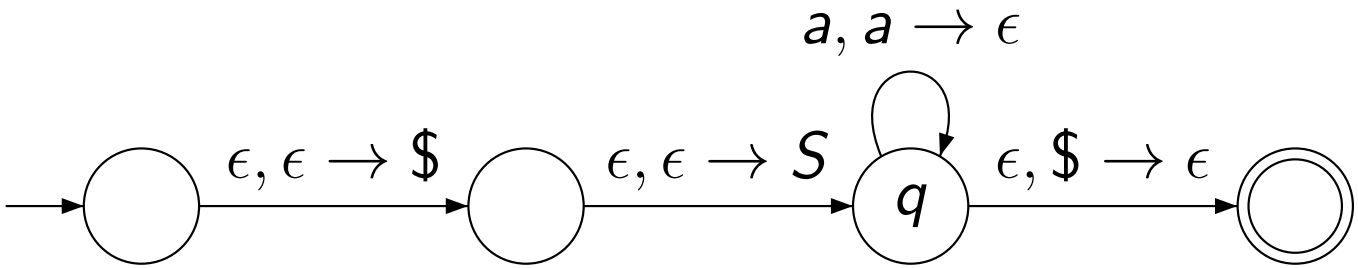
From CFGs to PDAs

What the PDA will do:

1. Push \$ and start symbol onto stack.
2. Repeat forever:
    1. Pop x off stack.
    2. If x is a variable, nondeterministically push RHS of a matching rule onto stack.
    3. If x is a terminal, consume x from input.
    4. If $x = \$$ and input is empty, accept.
    5. Otherwise, or if any action fails, reject on this branch.

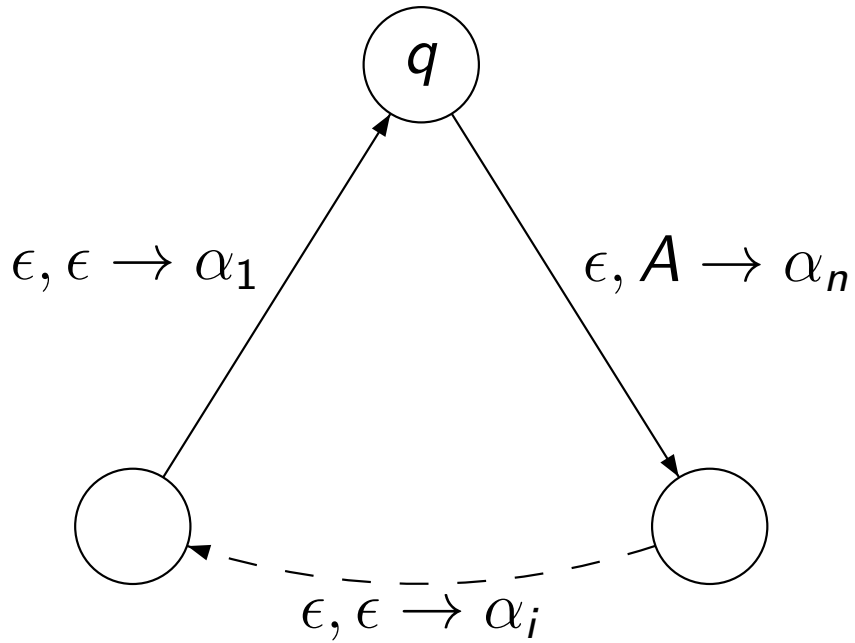
From CFGs to PDAs

Construct the PDA like this:



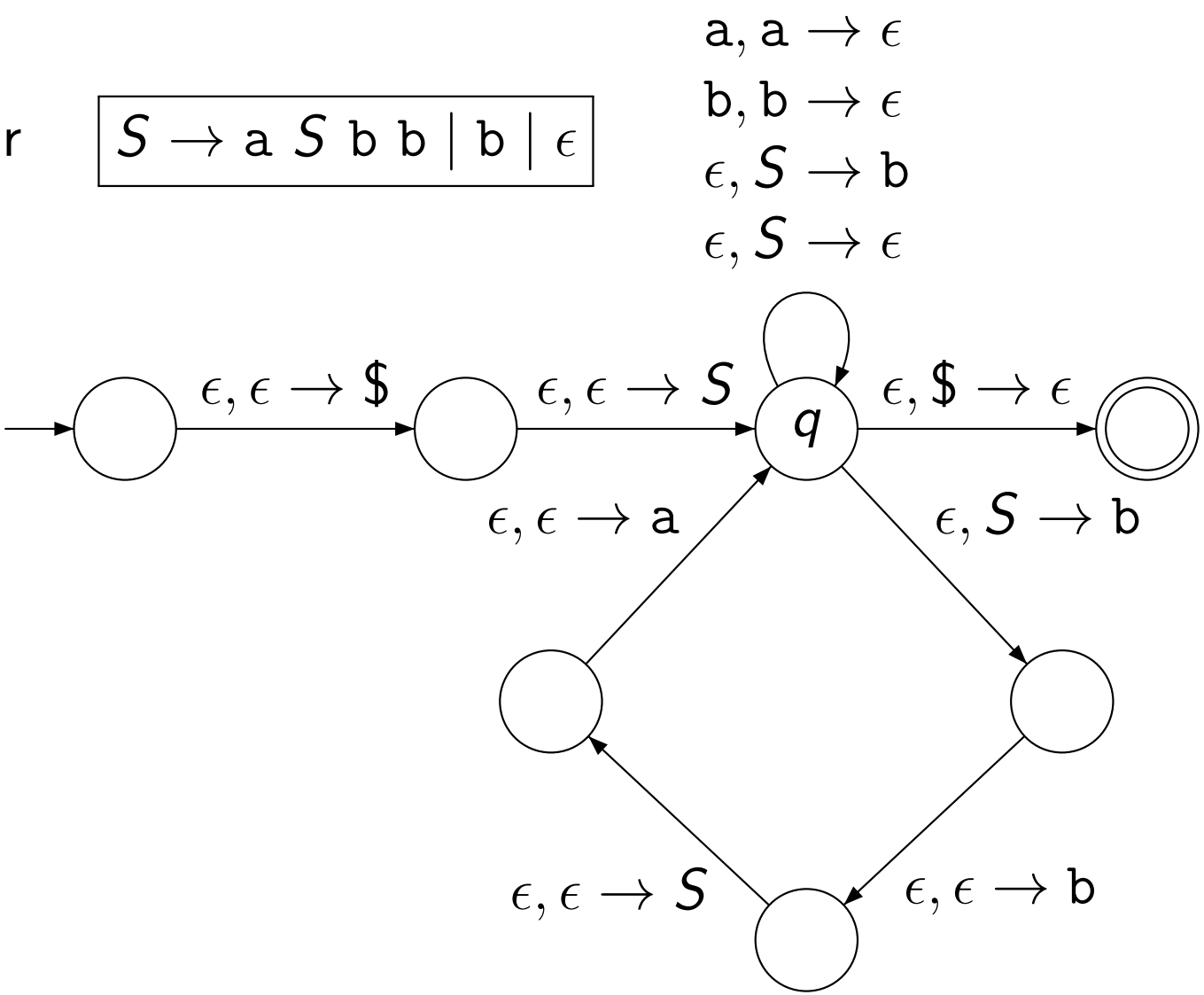
with a self-loop from q for each terminal a
(S is the grammar's start symbol).

For each rule $A \to \alpha_1 \ldots \alpha_n$,
add this loop from q to q:



Example Recogniser

For the grammar $\boxed{S \to \mathtt{a}\ S\ \mathtt{b}\ \mathtt{b} \mid \mathtt{b} \mid \epsilon}$



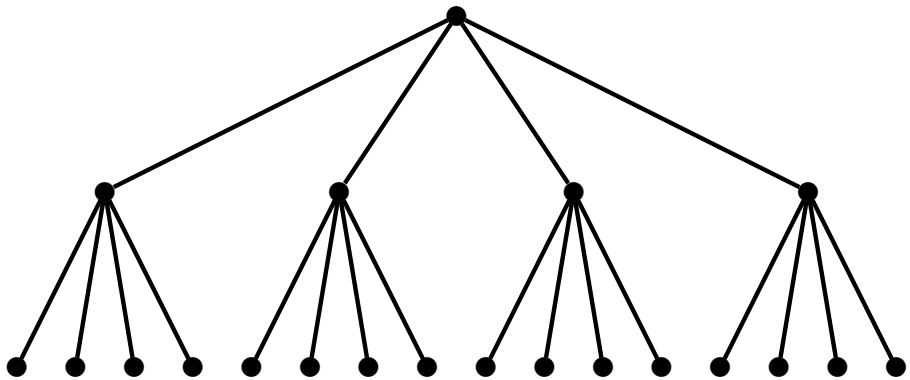
Pumping Lemma for CFLs

If A is context-free then there is a number p such that for any string $s \in A$ with $|s| \geq p$, there exist strings u, v, x, y, z such that $s = uvxyz$, and

1. $uv^i xy^i z \in A$ for all $i \geq 0$
2. $|vy| > 0$
3. $|vxy| \leq p$

String Length—Parse Tree Height

A tree of height h with branching factor b has at most b^h leaves.



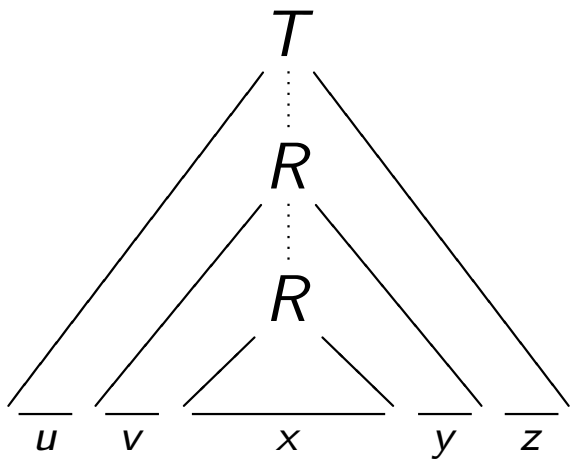
Let b and k be positive integers.

$$|s| > b^k \Rightarrow \text{height of parse tree } > k$$
$$\Rightarrow \text{some path has } k+1 \text{ nodes, of which } k \text{ are internal}$$

Proving the Pumping Lemma

Let T be the start variable of a CFG G which generates A. Let b be the length of the longest right-hand side in G. Let $p = b^{|V|+2}$.

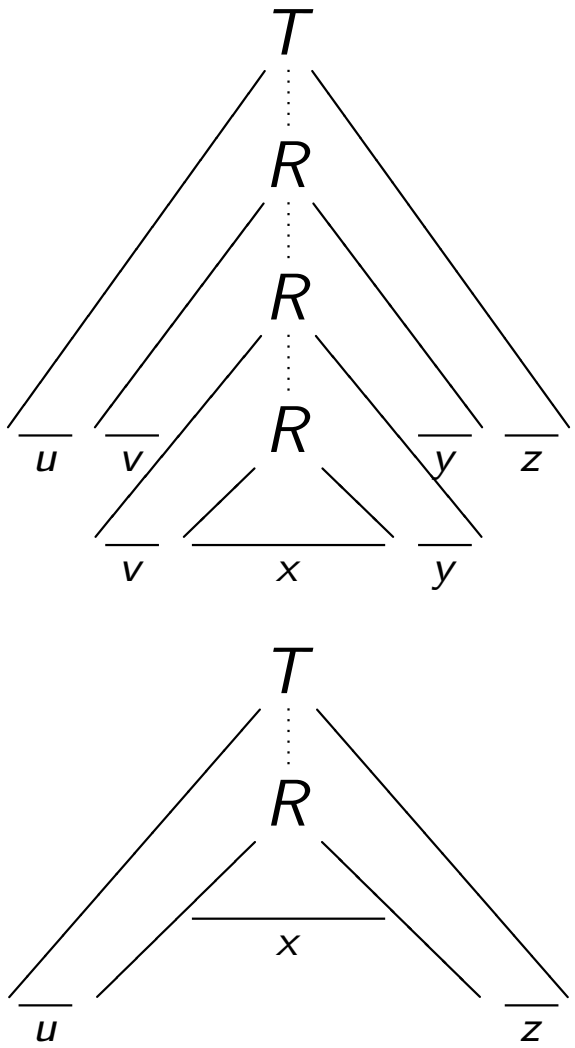
Consider some string s derived from T. If $|s| \geq p$ then the height of the parse tree is at least $|V| + 2$, as the tree has branching factor b or less.



Hence the longest path has $|V| + 1$ variables or more, so some variable occurs repeatedly.

Proving the Pumping Lemma

This gives the desired splitting into $uvxyz$. Clearly these are also valid parse trees:



Proving the Pumping Lemma

The condition that $|vxy| \leq p$ is satisfied if we make sure that the occurrences of R we consider are in the lowest part of the tree.

If both occurrences of R fall within the bottom $|V| + 1$ variables on the longest path, then the tree that generates vxy has height at most $|V| + 2$.

And so it generates a string of length at most $b^{|V|+2}$, that is, p.

Pumping Example 1

$A = \{ww \mid w \in \{0,1\}^*\}$ is not context-free.

Assume it is, let p be the pumping length, take $0^p1^p0^p1^p$.

By the pumping lemma, $0^p1^p0^p1^p = uvxyz$, with uv^ixy^iz in A for all $i \geq 0$, and $|vxy| \leq p$.

There are three ways that vxy can be part of

$$00\ldots0011\ldots1100\ldots0011\ldots11$$

If it straddles the midpoint, it has form 1^n0^m, so pumping down, we are left with $0^p1^i0^j1^p$, with $i < p$, or $j < p$, or both.

If it is in the first half, uv^2xy^2z will have pushed a 1 into the first position of the second half.

Similarly if vxy is in the second half.

Pumping Example 2

$B = \{\mathtt{a}^n\mathtt{b}^n\mathtt{c}^n \mid n \in \mathbb{N}\}$ is not context-free.

Assume it is, let p be the pumping length, and take $\mathtt{a}^p\mathtt{b}^p\mathtt{c}^p \in B$.

By the pumping lemma, $\mathtt{a}^p\mathtt{b}^p\mathtt{c}^p = uvxyz$, with uv^ixy^iz in B for all i.

Either v or y is non-empty (or both are).

If one of them contains two different symbols from $\{\mathtt{a}, \mathtt{b}, \mathtt{c}\}$ then uv^2xy^2z has symbols in the wrong order, and so cannot be in B.

So both v and y must contain only one kind of symbol. But then uv^2xy^2z can't have the same number of as, bs, and cs.

In all cases we have a contradiction.

Regular Grammars

There is a grammar notion that corresponds to the regular languages.

If we restrict the kind of rules allowed in CFGs, so they must be either of form

$$A \to w$$

or

$$A \to w\ B$$

with $w \in \Sigma^*$ and $A, B \in V$, then we have "regular grammars".

These generate exactly the regular languages.

(Here we chose the so-called right-linear form — we could have said $A \to B\ w$ instead of $A \to w\ B$.)

Deterministic PDAs (DPDAs)

Determinism: at most one possible move at any time.

Definition

A PDA $(Q, \Sigma, \Gamma, \delta, q_0, F)$ is *deterministic* if and only if

$$|\delta(q, v, a)| + |\delta(q, v, \epsilon)| + |\delta(q, \epsilon, a)| + |\delta q, \epsilon, \mathit{epsilon}| \leq 1$$

for all states $q \in Q$, input symbols $v \in \Sigma$, and stack symbols $a \in \Gamma$.

Sipser's book has a slightly different but equivalent definition for a DPDA.

Deterministic PDAs

Is a deterministic PDA (a DPDA) as powerful as a PDA?

No. A DPDA can recognise the context-free

$$\{wcw^{\mathcal{R}} \mid c \in \Sigma, w \in (\Sigma \setminus \{c\})^*\}$$

but not the context-free $\{ww^{\mathcal{R}} \mid w \in \Sigma^*\}$.

Nondeterminism lets us "guess" where the middle is.

Without it, cannot know where middle of input is, so cannot know when to start popping stack.

Imagine reading 000011000000000110000 from left to right, without reading far ahead/behind.

After the Teaching-Free Week

We discuss Turing machines!

Have a nice break!