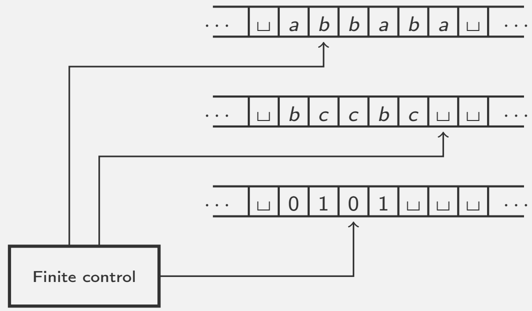
Fundamentals of theory of computation 2

4th lecture

lecturer: Tichler Krisztián
ktichler@inf.elte.hu

Multitape Turing machines



- Reading k tape symbols and the current state the TM moves to a new state, rewrites the k tape symbols and each of the k tape heads move to neighboring cells (or stay).
- The concept of accepting a word is analogous to that of 1-tape TM's.
- The concepts of running time and time complexity are analogous to those of 1-tape TM's.

Multitape Turing machines

Definition

A k-tape TM is a 7-tuple $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r \rangle$ where

- Q is a finite, nonempty set of states,
- $q_0, q_a, q_r \in Q$, q_0 is the starting, q_a is the accepting and q_r is the rejecting state,
- Σ and Γ are the input and the tape alphabets, respectively, where $\Sigma \subseteq \Gamma$ and $\sqcup \in \Gamma \backslash \Sigma$,
- $\delta : (Q \backslash \{q_a, q_r\}) \times \Gamma^k \to Q \times \Gamma^k \times \{L, S, R\}^k$ is the transition function.

Definition

$(q, u_1, v_1, \ldots, u_k, v_k)$ is called a configuration of a k-tape TM, where $q \in Q$ and $u_i, v_i \in \Gamma^*, v_i \neq \lambda$ $(1 \leqslant i \leqslant k)$.

Multitape Turing machines

Configuration is a finite representation of the machine at a given time. It represents the current state q, the content of the ith tape $u_i v_i$, and the position of the ith head as the first letter of v_i $(1 \leqslant i \leqslant k)$.

Definition

Starting configuration of the word u is $u_i = \lambda$ $(1 \leqslant i \leqslant k)$, $v_1 = u\sqcup$, and $v_i = \sqcup$ $(2 \leqslant i \leqslant k)$.

[Why v_1 is defined to be $u\sqcup$ and not u? To avoid $u = \lambda$ being another case. The two words represent the same tape content.]

Definition

For a configuration $(q, u_1, v_1, \ldots, u_k, v_k)$ where $q \in Q$ and $u_i, v_i \in \Gamma^*, v_i \neq \lambda$ $(1 \leqslant i \leqslant k)$, it is an accepting configuration if $q = q_a$, rejecting configuration, if $q = q_r$, halting configuration, if $q = q_a$ or $q = q_r$.

Multitape Turing machines

Definition

For a k-tape Turing machine $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_i, q_n \rangle$ let us introduce the one-step transition relation $\vdash \subseteq C_M \times C_M$ as follows

Let $C = (q, u_1, a_1 v_1, \ldots, u_k, a_k v_k)$ be a configuration, where $a_i \in \Gamma,\ u_i, v_i \in \Gamma^*$ $(1 \leqslant i \leqslant k)$. Let furthermore $\delta(q, a_1, \ldots, a_k) = (r, b_1, \ldots, b_k, D_1, \ldots, D_k)$ be a transition, where $q, r \in Q,\ b_i \in \Gamma, D_i \in \{L, S, R\}$ $(1 \leqslant i \leqslant k)$. Then $C \vdash (r, u_1', v_1', \ldots, u_k', v_k')$, where for every tape $1 \leqslant i \leqslant k$

- if $D_i = R$ then $u_i' = u_i b_i$ and $v_i' = v_i$ in the case of $v_i \neq \lambda$, otherwise $v_i' = \sqcup$,
- if $D_i = S$ then $u_i' = u_i$ and $v_i' = b_i v_i$,
- if $D_i = L$ then $u_i = u_i' c$ $(c \in \Gamma)$ and $v_i' = c b_i v_i$ in the case of $u_i \neq \lambda$, otherwise $u_i' = \lambda$ és $v_i' = \sqcup b_i v_i$.

Multitape Turing machines

So by the definitions $C \vdash C'$ holds for configurations C, C' if we can get C' from C by following the instructions of δ for every tape.

Example:

Let k=2 and $\delta(q, a_1, a_2) = (r, b_1, b_2, R, S)$ be a transition of a TM. Then $(q, u_1, a_1 v_1, u_2, a_2 v_2) \vdash (r, u_1 b_1, v_1', u_2, b_2 v_2)$, where $v_1' = v_1$, if $v_1 \neq \lambda$, otherwise $v_1' = \sqcup$.

Notice, that the heads can move independently of each other.

Definition

Multistep transition relation (a configuration yields an other one in finitely many steps) is formally defined the same way as we did it for one-tape TM's. Notation: $\vdash^*$.

Multitape Turing machines

Definition

The language recognized by a k-tape Turing machine $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r \rangle$ is the following:
$L(M) = \{ u \in \Sigma^* \mid (q_0, \lambda, u\sqcup, \lambda, \sqcup, \ldots, \lambda, \sqcup) \vdash^* (q_a, x_1, y_1, \ldots x_k, y_k),\ x_1, y_1 \ldots, x_k, y_k \in \Gamma^*, y_1, \ldots, , y_k \neq \lambda \}$.

As in the case of one tape TM's multitape TM's are accepting those words for which the TM can reach its accepting state q_a.

Languages that can be recognized or can be decided by a multitape TM is defined the same way as for one-tape TM's.

Definition

Running time of word u on TM M is the number of computational steps from the starting configuration of u to a halting configuration.

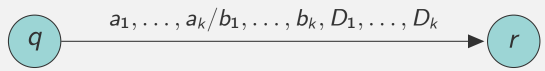
Time complexity of a multitape TM is defined the same way as we did for 1-tape TM's.

Multitape Turing machines

An Example

Exercise: Construct a 2-tape TM M having $L(M) = \{ww^{-1} \mid w \in \{a, b\}^*\}$.

Transition diagram is a vertex and edge labelled directed graph, where



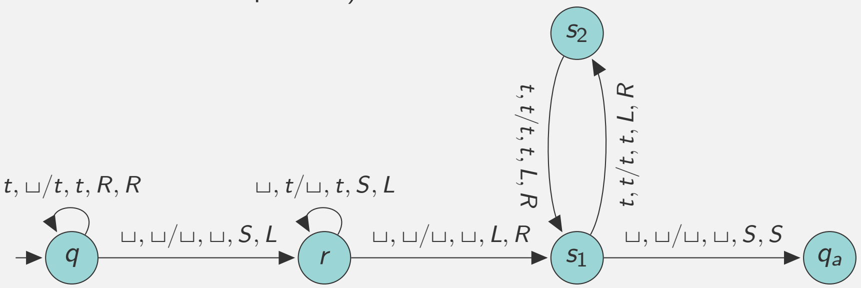
$$\delta(q, a_1, \ldots, a_k) = (r, b_1, \ldots, b_k, D_1, \ldots, D_k)$$
$$(q, r \in Q, a_1, \ldots, a_k, b_1, \ldots, b_k \in \Gamma, D_1, \ldots, D_k \in \{L, S, R\})$$

Multitape Turing machines

An Example

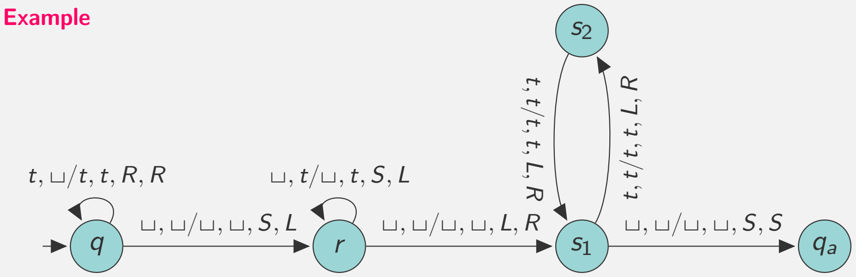
Exercise: Construct a 2-tape TM M having $L(M) = \{ww^{-1} \mid w \in \{a,b\}^*\}$.

Solution:
(All other transitions go to q_r. Let $t \in \{a,b\}$ be arbitrary for the transitions where t is present.)



Multitape Turing machines

An Example



$(q, \lambda, abba, \lambda, \sqcup) \vdash (q, a, bba, a, \sqcup) \vdash (q, ab, ba, ab, \sqcup) \vdash$
$(q, abb, a, abb, \sqcup) \vdash (q, abba, \sqcup, abba, \sqcup) \vdash (r, abba, \sqcup, abb, a) \vdash$
$(r, abba, \sqcup, ab, ba) \vdash (r, abba, \sqcup, a, bba) \vdash (r, abba, \sqcup, \lambda, abba) \vdash$
$(r, abba, \sqcup, \lambda, \sqcup abba) \vdash (s_1, abb, a, \lambda, abba) \vdash$
$(s_2, ab, ba, a, bba) \vdash (s_1, a, bba, ab, ba) \vdash (s_2, \lambda, abba, abb, a) \vdash$
$(s_1, \lambda, \sqcup abba, abba, \sqcup) \vdash (q_i, \lambda, \sqcup abba, abba, \sqcup)$

What is the time complexity of M? It is a $3n+3 =$ **O(n)** time TM.

Multitape Turing machines – 1-tape simulation

Definition

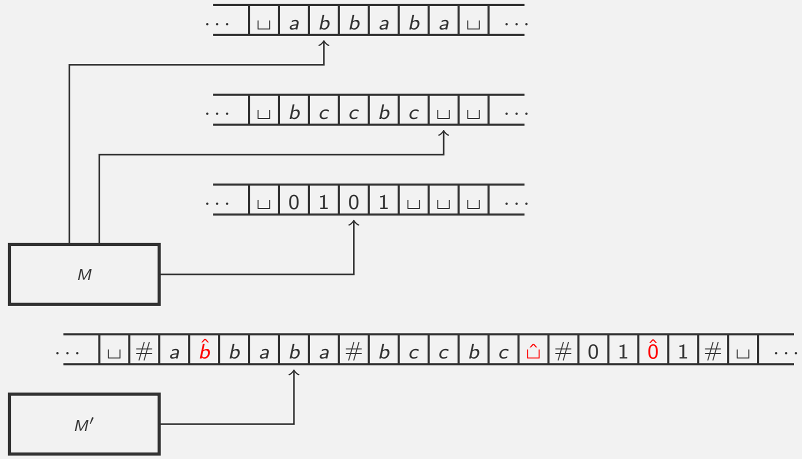
Two TM's are considered to be **equivalent**, if they recognize the same language.

Theorem

For any k-tape TM M there is an equivalent 1-tape TM M'. Furthermore, if M has time complexity $f(n)$, which is at least linear, i.e, $f(n) = \Omega(n)$, then M' has time complexity $O(f(n)^2)$.

Multitape Turing machines – 1-tape simulation

Proof (sketch): Main idea of the simulation



Multitape Turing machines – 1-tape simulation

Steps of the simulation on input $a_1 \cdots a_n$:

1. Let the starting configuration M' be $q_0' \# \hat{a}_1 a_2 \cdots a_n \# \hat{\sqcup} \# \cdots \hat{\sqcup} \#$
2. As M' scans its tape for the first time counts the $\#$'s and stores symbols denoted by a $\hat{}$ in its states. (E.g., for the case on the figure, M is in state q then M' changes its state from (q) to (q, b), to $(q, b, \sqcup)$ and finishing in $(q, b, \sqcup, 0)$.)
3. M' goes through its tape for the another time updating it's content according to its transition function.
4. if the length of the content on a tape of M increases M' shifts it's content by 1 cell to make room for the new letter (for $\sqcup$ in fact). This can be done in O(# letters to be moved).
5. If M reaches its accepting or rejecting state so does M'.
6. Otherwise M' continues with step 2.

Multitape Turing machines – 1-tape simulation

The following holds for simulating a single step of M:

- Used up space (number of used cells) is an asymptotic upper bound for the number of steps M' takes. (Goes through its content twice, it needs to make space for a $\sqcup$ at most k times which is O(used up space))
- used up space is increased by $O(1)$ cells. (by $\leqslant k$, in fact)

In the beginning M' uses $\Theta(n)$ cells. In each step the used up space is increased by $O(1)$, so $O(n + f(n)O(1)) = O(n + f(n))$ is a common asymptotic upper bound for the used up cells after each step of the simulation.

So $O(n + f(n))$ is a general asympotic upper bound for the time complexity of a single step.

Altogether M' has a time complexity of $f(n) \cdot O(n + f(n))$, which is $O(f(n)^2)$, if $f(n) = \Omega(n)$.

Nondeterministic Turing machines

Definition

A **nondeterministic TM** (NTM) is a 7-tuple $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r \rangle$ where

- $Q, \Sigma, \Gamma, q_0, q_a, q_r$ are the same as for deterministic TM's
- $\delta : (Q \backslash \{q_a, q_r\}) \times \Gamma \to \mathcal{P}(Q \times \Gamma \times \{L, S, R\})$

Let C_M be the set of all possible configurations for a NTM M.

$\vdash \subseteq C_M \times C_M$ one-step transition relation

Let $uqav$ be a configuration, where $a \in \Gamma$, $u, v \in \Gamma^*$.

- If $(r, b, R) \in \delta(q, a)$, then $uqav \vdash ubrv'$, where $v' = v$, if $v \neq \lambda$, otherwise $v' = \sqcup$,
- if $(r, b, S) \in \delta(q, a)$, then $uqav \vdash urbv$,
- if $(r, b, L) \in \delta(q, a)$, then $uqav \vdash u'rcbv$, where $c \in \Gamma$ and $u'c = u$, if $u \neq \lambda$, otherwise $u' = u$ and $c = \sqcup$.

If $C \vdash C'$ we say that **C yields C' in one step**.

Nondeterministic Turing machines

Multistep transition relation, denoted by $\vdash^*$, is the reflexive, transitive closure of one-step transition relation $\vdash$. I.e.,

$\vdash^* \subseteq C_M \times C_M$ multistep transition relation

Let $C, C' \in C_M$ be configurations of a NTM M. $C \vdash^* C' \Leftrightarrow$

- $C = C'$ or
- $\exists\, n > 0 \wedge C_1, C_2, \ldots \in C_n \in C_M$, such that $\forall\, 1 \leqslant i \leqslant n-1$ $C_i \vdash C_{i+1}$ holds. Furthermore $C_1 = C$ and $C_n = C'$.

If $C \vdash^* C'$ we say that **C yields C' in finitely many steps**

NTM's may have several computations for the same word. It accepts a word if and only if it has at least on computation ending in q_a.

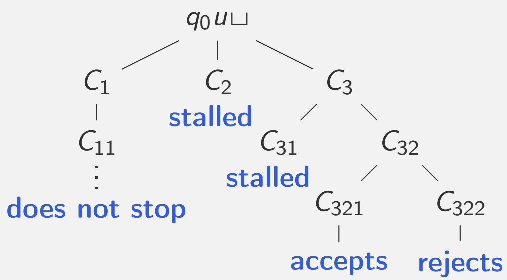
Example: Let $\delta(q_2, a) = \{(q_5, b, L), (q_1, d, R)\}$ and $C_1 = bcq_2a\sqcup b$, $C_2 = bq_5cb\sqcup b$, $C_3 = bcdq_1\sqcup b$. Then $C_1 \vdash C_2$ and $C_1 \vdash C_3$. If we have another configuration C_4 with the property $C_2 \vdash C_4$. Then $C_1 \vdash^* C_1$, $C_1 \vdash^* C_2$, $C_1 \vdash^* C_3$ and $C_1 \vdash^* C_4$.

Nondeterministic Turing machines

Definition

The nondeterministic configuration tree for $u \in \Sigma^*$ is a vertex labelled directed tree with the following properties. The root has label $q_0 u \sqcup$. If C is a label of a node, then it has $|\{C' \mid C \vdash C'\}|$ children labelled by the elements of $\{C' \mid C \vdash C'\}$.

Example:



This TM accepts u since $q_0 u \sqcup \vdash C_3 \vdash C_{32} \vdash C_{321}$ is a computation leading to an accepting configuration. Only one accepting computation is needed to accept a word.

Nondeterministic Turing machines

Definition

The language recognized by a NTM $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r \rangle$ is $L(M) = \{u \in \Sigma^* \mid q_0 u \sqcup \vdash^* x q_a y \text{ for some } x, y \in \Gamma^*, y \neq \lambda\}$.

Definition

A NTM M recognizes L if $L(M) = L$. A NTM M decides a language L if M recognizes L and for all $u \in \Sigma^*$ the configuration tree has a finite height and all leaves are halting configurations.

Definition

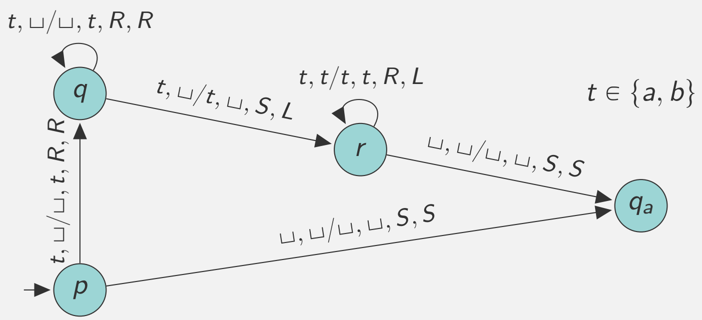
M has time complexity $f(n)$, if for all $u \in \Sigma^*$ of length n the height of the configuration tree is at most $f(n)$.

Remark: Definition of a multitape NTM is analogous to that of previous definitions.

Nondeterministic TM

Example

Excercise: Construct a NTM M with $L(M) = \{ww^{-1} \mid w \in \{a, b\}^*\}$.



$(p, \lambda, abba, \lambda, \sqcup) \vdash (q, \lambda, bba, a, \sqcup) \vdash (r, \lambda, bba, \lambda, a) \vdash (q_r, \lambda, bba, \lambda, a)$

$(p, \lambda, abba, \lambda, \sqcup) \vdash (q, \lambda, bba, a, \sqcup) \vdash (q, \lambda, ba, ab, \sqcup) \vdash (r, \lambda, ba, a, b) \vdash (r, b, a, \lambda, ab) \vdash (r, ba, \sqcup, \lambda, \sqcup ab) \vdash (q_a, ba, \sqcup, \lambda, \sqcup ab)$

Nondeterministic TM

Simulating NTM's by deterministic TM's

Theorem

For all $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_a, q_r \rangle$ NTM's of time complexity $f(n)$ there is an equivalent deterministic TM of time complexity $2^{O(f(n))}$.

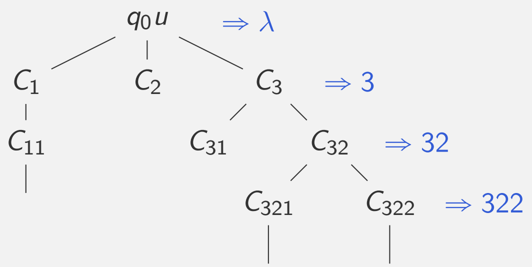
Proof (sketch): M' simulates all partial computations for an input $u \in \Sigma^*$ by doing a breadth first search on its configuration tree.

- Let d be the following number. $d = \max_{(q,a) \in Q \times T} |\delta(q, a)|$.
- Let $T = \{1, 2, \ldots, d\}$ be an alphabet.
- for all $(q, a) \in Q \times \Gamma$ let us fix an order of the set $\delta(q, a)$

Nondeterministic TM

Simulating NTM's by deterministic TM's

For each node of the configuration tree one can associate a unique word over T, the **selector** of that partial computation.



Shortlex order

Definition

Let $X = \{x_1 \prec x_2 \prec \cdots \prec x_s\}$ be an ordered alphabet. The **short-lexicographic** (shortlex) order of X^* is the following order, denoted by $\prec_{\text{shortlex}}$. For every $u_1 \cdots u_n, v_1 \cdots v_m \in X^*$ let
$u_1 \cdots u_n \prec_{\text{shortlex}} v_1 \cdots v_m \Leftrightarrow (n < m) \vee$
$((n = m) \wedge (u_k \prec v_k)$, where k is the smallest i having $u_i \neq v_i)$.

Example 1 Let $X = \{a, b\}$ and $a \prec b$, then the shortlex order of X^* is

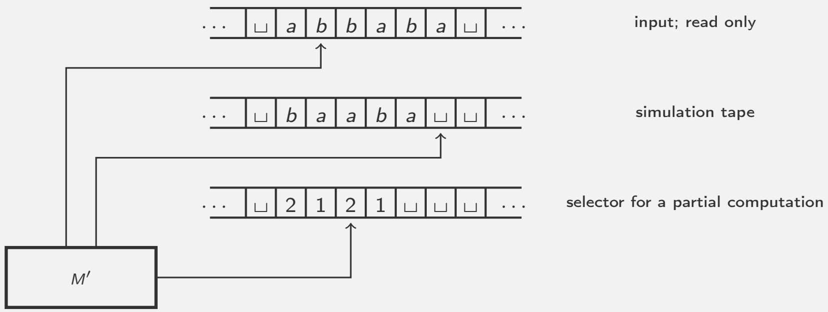
$\lambda, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, \ldots$

Example 2 Consider natural numbers as finite sequences of digits (no extra opening 0's, except for the number 0 itself).

Then $n < m$ if and only if $n \prec_{\text{shortlex}} m$ for the ordered alphabet $X = \{0 \prec 1 \prec 2 \prec 3 \prec 4 \prec 5 \prec 6 \prec 7 \prec 8 \prec 9\}$.

Nondeterministic TM

Simulating NTM's by deterministic TM's



How does M' work?

Nondeterministic TM

- starting configuration of M': input on 1st tape, the other tapes are empty
- WHILE there's no accept
  - copy the content of tape 1 of M' to tape 2
  - WHILE the head of the 3rd tape does not read ⊔
    - Let k be the current letter on tape 3
    - Let a be the current letter on tape 2 and q be the current state of M
    - if $\delta(q, a)$ has a kth element, then
      - M' simulates one step of M according to this
      - if this leads to q_a of M, then M' accepts
      - if this leads to q_r of M, then abort loop

      else ($\delta(q, a)$ has no kth element) abort loop
    - M' moves one cell to the right on the 3rd tape
  - M' deletes the content of tape 2 and creates the next word on tape 3 according to shortlex order over T.

Nondeterministic TM

Simulating NTM's by deterministic TM's

- M' goes to it accepting state if and only if M do so, so they are equivalent TM's,
- the number of partial computations of length at most $f(n)$ is at most the number of nodes of complete d-ary tree, i.e. at most
$$\sum_{i=0}^{f(n)} d^i = \frac{d^{f(n)+1} - 1}{d - 1} = O(d^{f(n)}),$$
- for every partial computation M' takes $O(n + f(n))$ steps,
- altogether the time complexity of M' is $O(n + f(n))O(d^{f(n)}) = 2^{O(f(n))}$.

Remarks:

- The above simulation has exponential time complexity. Other simulations may do better.
- Conjecture: there's no efficient (polynomial) simulation of a NTM by a deterministic one.