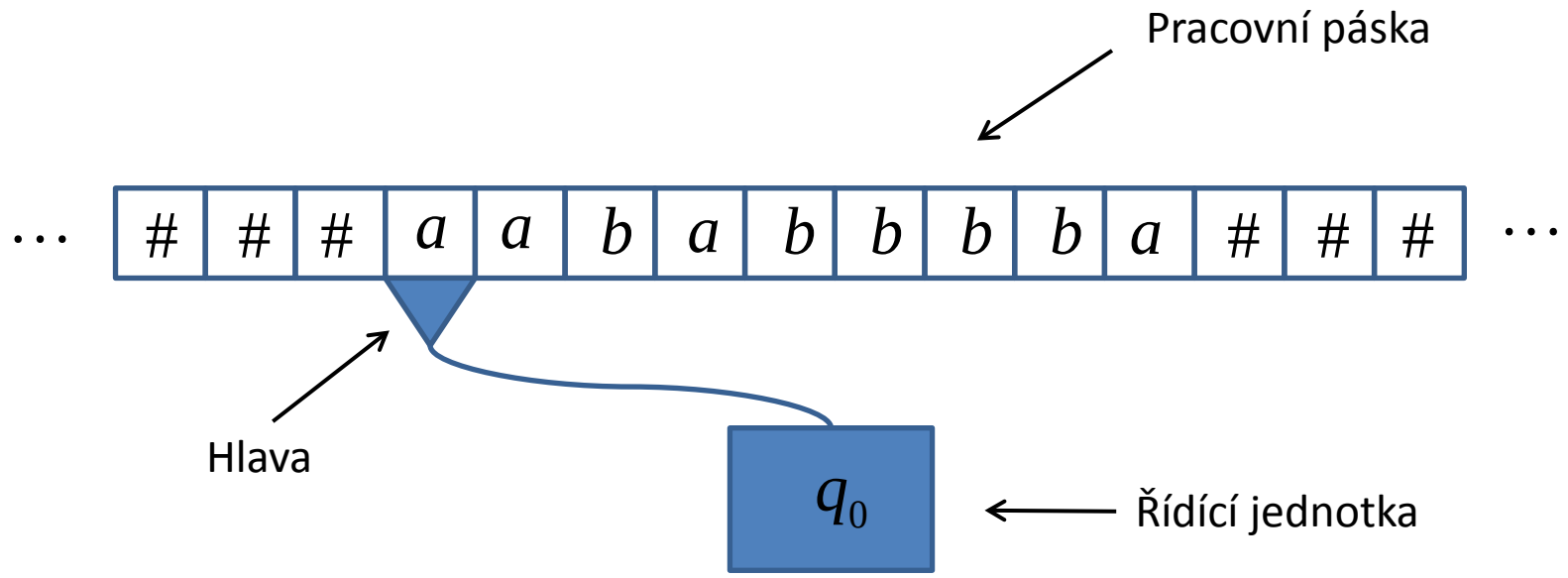
Abstraktní výpočetní modely

1. Turingův stroj
   - Jednoduchý model, jehož výpočetní síla je ekvivalentní s reálným počítačem.
   - Umožňuje studovat meze algoritmických výpočtů.
   - Popsán v roce 1936 A. Turingem, použit k řešení tzv. „Rozhodovacího problému“ – *existuje postup, který by rozhodoval, jestli dané matematické tvrzení je pravdivé ?*

2. RAM (Random Access Machine)
   - Navržen později, lépe vystihuje architekturu počítačů.
   - Vhodný pro návrh konkrétních algoritmů.

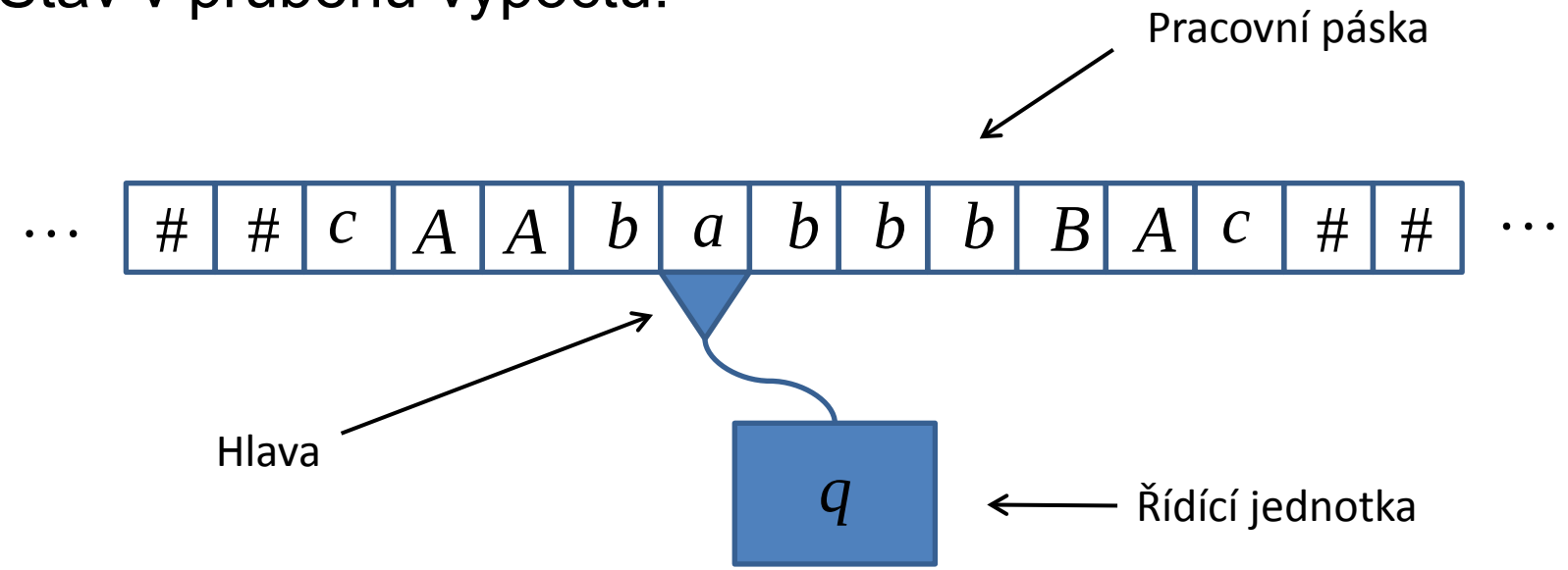
Turingův stroj

Zesilujeme schopnosti konečného automatu
(i zásobníkového automatu):



Turingův stroj

Stav v průběhu výpočtu:



Jeden výpočetní krok:

- závisí na čteném symbolu a stavu řídící jednotky
- provede se přepis symbolu na pásce, změní se stav a hlava se může posunout o jedno pole vlevo nebo vpravo

Turingův stroj – formální definice

Turingův stroj (TS) M je šestice $(Q, \Sigma, \Gamma, \delta, q_0, F)$, kde

- Q je konečná množina stavů
- Σ je vstupní abeceda
- Γ je pracovní abeceda, $\Sigma \cup \{\#\} \subseteq \Gamma$, $\# \notin \Sigma$
- $F \subseteq Q$ je množina koncových stavů
- $q_0 \in Q$ je počáteční (iniciální) stav
- δ je přechodová funkce

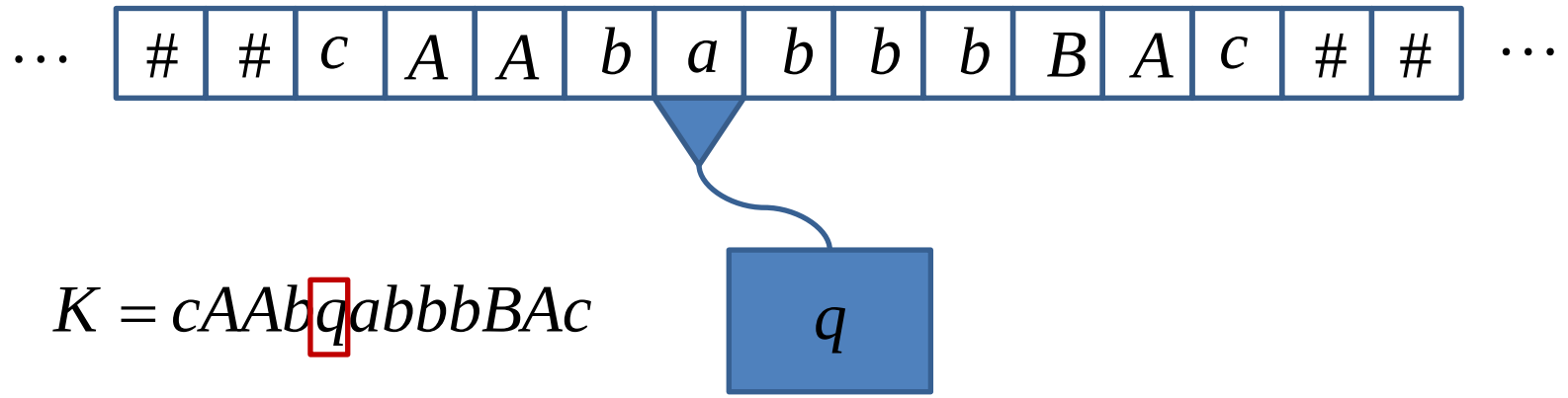
$$\delta : (Q - F) \times \Gamma \to Q \times \Gamma \times \{L, N, R\}$$

$\#$ .. speciální prázdný symbol („blank“ symbol)
$\{L, N, R\}$ .. změny pozice hlavy (vlevo, žádný posun, vpravo)

Konfigurace

- Potřebujeme reprezentovat obsah pásky, pozici hlavy a stav řídící jednotky.

Konfigurace: $K = uqv$, kde $u, v \in \Gamma^*$ a $q \in Q$



$K = cAAbqabbbBAc$

- Počáteční (iniciální) konfigurace: $q_0 v$
- Koncová konfigurace: uqv, kde $q \in F$

Výpočet

(Konečný) výpočet Turingova stroje je posloupnost konfigurací $K_0, K_1, \ldots, K_n$, kde K_0 je počáteční konfigurace, K_n je koncová konfigurace a K_{i+1} vznikne z K_i provedením jednoho výpočetního kroku.

Dvě možné interpretace výsledku výpočtu:

1. Rozdělíme koncové stavy na přijímací a zamítací. Turingův stroj M přijímá jazyk obsahující slova, pro které skončí M v přijímacím stavu.

2. Výsledkem je slovo, které je na konci na pracovní pásce (v takovém případě počítá TS funkci $f : \Sigma^* \to \Gamma^*$).

Rozhodování jazyka

- Výpočet TS M nemusí pro daný vstup u nikdy skončit. V takovém případě M slovo u nepřijímá.

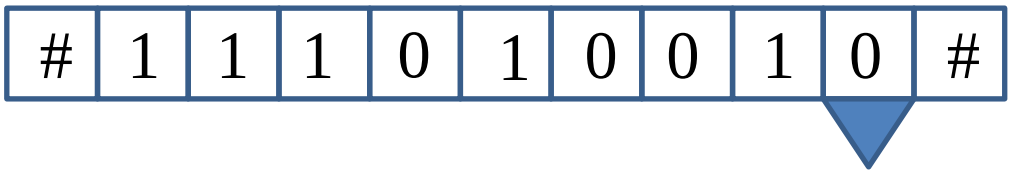
- Říkáme, že M rozhoduje jazyk L, pokud M přijímá L a každý výpočet stroje M je konečný.

Příklad 1

Vstup je binární číslo, TS násobí 3-mi.

TS nejprve přesune hlavu na druhý konec vstupu.
Instrukce:

$$(q_0,0) \rightarrow (q_0,0,R)$$
$$(q_0,1) \rightarrow (q_0,1,R)$$
$$(q_0,\#) \rightarrow (q_{P0},\#,L)$$



Poté provede násobení průchodem od nižších bitů k vyšším.

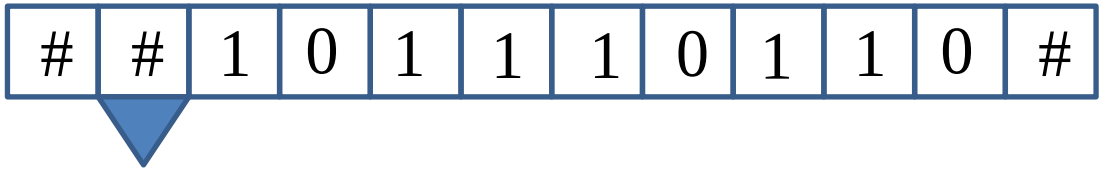
Příklad 1

Přenos do sousedního bitu je 0, 1 nebo 2 – zapamatujeme si jej ve stavu TS.

$(q_{P0},0) \to (q_{P0},0,L)$ $(q_{P1},0) \to (q_{P0},1,L)$ $(q_{P2},0) \to (q_{P1},0,L)$

$(q_{P0},1) \to (q_{P1},1,L)$ $(q_{P1},1) \to (q_{P2},0,L)$ $(q_{P2},1) \to (q_{P2},1,L)$

V okamžiku, kdy se hlava dostane mimo vstup, je potřeba ještě zapsat na pásku další bity odpovídající přenosu.

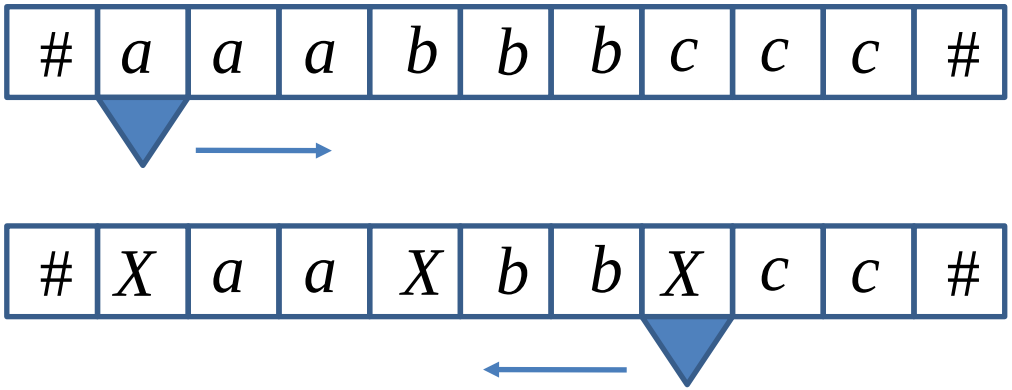


$(q_{P2},\#) \to (q_{P1},0,L)$ $(q_{P1},\#) \to (q_{P0},1,L)$ $(q_{P0},\#) \to (q_F,\#,N)$

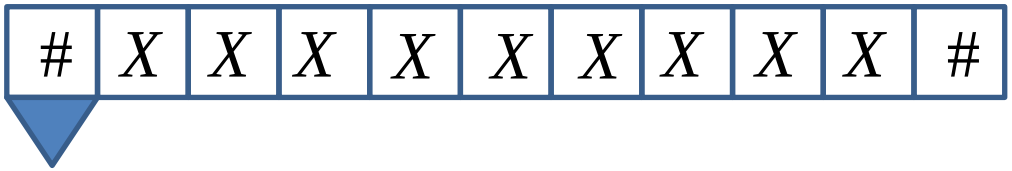
Příklad 2

Navrhneme TS, který rozhoduje jazyk $L = \{a^i b^i c^i \mid i \in \mathbf{N}\}$.

- průběh jednoho cyklu:



- ukončení výpočtu:



Časová a paměťová složitost

$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$

Nechť $u \in \Sigma^*$ je vstupní slovo. Označme počet kroků výpočtu stroje M pro u jako $t(u)$ a počet (různých) navštívených polí pásky jako $s(u)$.

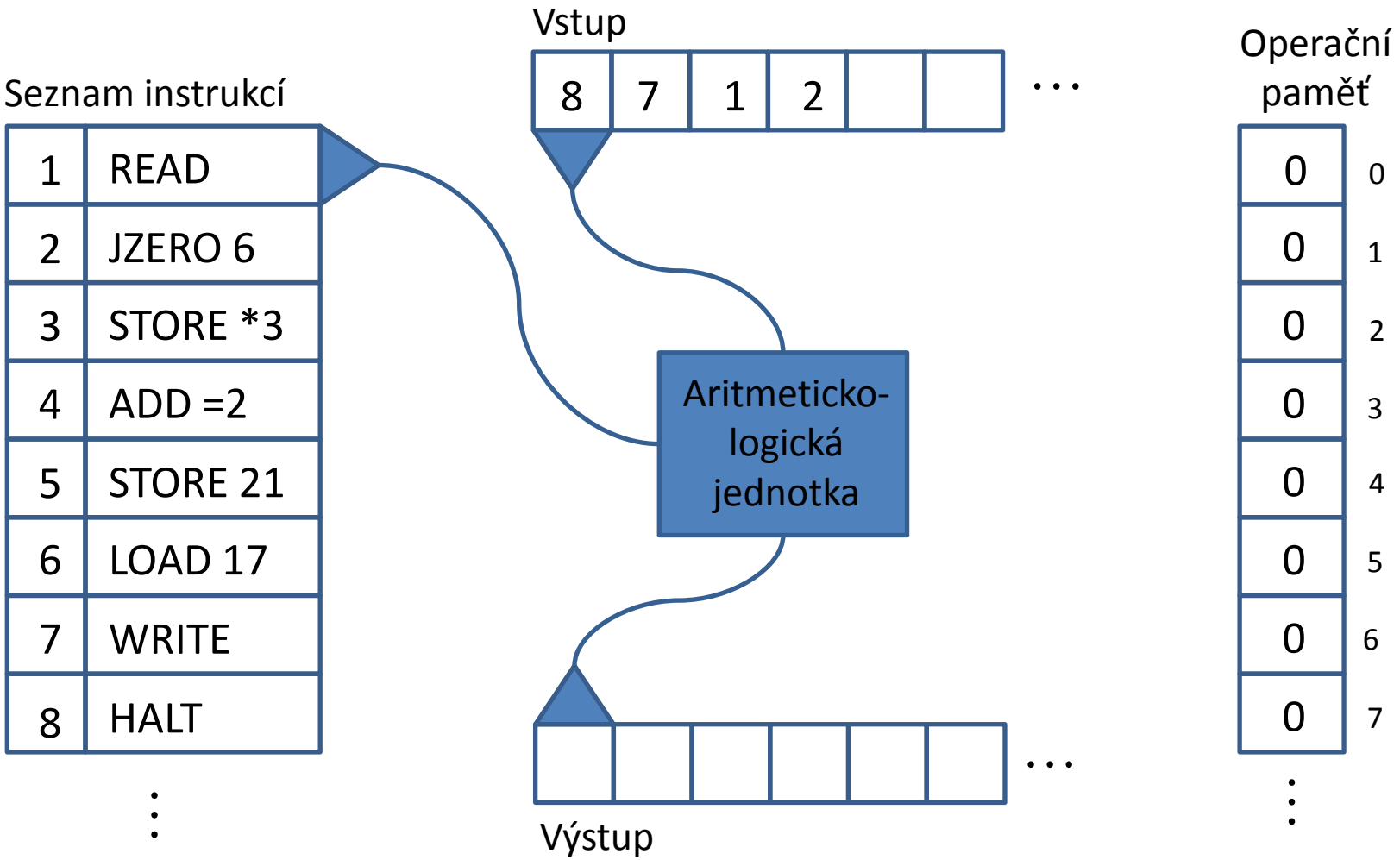
Časová složitost stroje M je funkce $t : \mathbf{N} \to \mathbf{N}$ definovaná následovně:

$$t(n) = \max_{u \in \Sigma^*, |u|=n} t(u)$$

Analogicky, paměťová (též prostorová) složitost stroje M je funkce:

$$s(n) = \max_{u \in \Sigma^*, |u|=n} s(u)$$

Stroj RAM (Random Access Machine)



Stroj RAM

- Do každé paměťové buňky lze uložit libovolné celé číslo.
- Buňky na adrese 0 a 1 mají speciální význam:
  - Buňka 0 je pracovní registr
  - Buňka 1 je indexový registr

Konvence pro zápis čísel a adresování paměti:

$=i$ … číslo i

i … hodnota čísla obsaženého v buňce s adresou i

$*i$ … hodnota čísla v buňce s adresou $i+j$, kde j je aktuální hodnota v indexovém registru

Stroj RAM – typy instrukcí

READ – do pracovního registru načte číslo, které snímá hlava na vstupní pásce

WRITE – číslo z pracovního registru zapíše na výstupní pásku

LOAD *op* – do prac. registru načte hodnotu operandu

STORE *op* – do paměti uloží číslo z prac. registru

ADD *op*, SUB *op*, MUL *op*, DIV *op* – aritmetické operace, modifikují pracovní registr

Stroj RAM – typy instrukcí

JUMP *index* – výpočet pokračuje instrukcí na uvedeném indexu

JZERO *index* – skok v případě, že pracovní registr obsahuje hodnotu 0

JGTZ *index* – skok v případě, že pracovní registr obsahuje hodnotu větší než 0

HALT – ukončení výpočtu

Vzájemná simulace TS a RAM

Mějme TS $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$. Můžeme navrhnout program pro RAM, který simuluje M a skončí vždy se stejným výsledkem jako M.

- pásku reprezentujeme v operační paměti
- v indexovém registru je uložená pozice hlavy stroje M

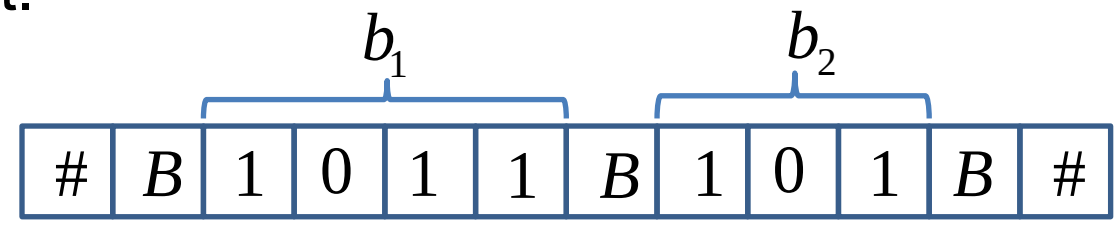
Časová i paměťová složitost navrženého RAMu je řádově stejná jako složitost TS M.

Vzájemná simulace TS a RAM

Opačná situace: máme stroj RAM a jeho program, chceme jej simulovat pomocí TS.

Lze též realizovat, je to ale komplikovanější:

Jednu paměťovou buňku reprezentujeme souvislým blokem polí na pásce. Bloky je nutné podle potřeby průběžně zvětšovat.



Pokud RAM má časovou složitost t_1, pak jej TS simuluje v čase $t_2 = O(t_1^{\,d})$, kde d je nějaká vhodná konstanta.

Church-Turingova teze

Ke každému algoritmu lze zkonstruovat s ním ekvivalentní Turingův stroj.

Ekvivalentní ve smyslu: algoritmus i TS se zastaví právě pro tytéž vstupy, přičemž příslušné výstupy jsou totožné.

Další formalismy ekvivalentní s TS:

- λ-kalkulus (funkcionální programování)
- rekurzivní funkce
- gramatiky

Vztah TS k Chomského hierarchii

- **Typ 0** (obecné gramatiky)
  - generují právě ty jazyky, které lze přijímat pomocí TS
- **Typ 1** (kontextové gramatiky)
  - generují právě ty jazyky, jež lze rozhodovat pomocí TS, který má povoleno pohybovat hlavou pouze nad buňkami obsahujícími vstup (tzv. LBA – linear bounded automaton)
- **Typ 2** (bezkontextové gramatiky)
  - ekvivalentní jsou zásobníkové automaty
- **Typ 3** (regulární gramatiky)
  - ekvivalentní jsou konečné automaty

Varianty TS

- Vícepáskový TS
- Vstupní páska pouze čtecí + pracovní páska
- Páska nekonečná pouze v jednom směru

Všechny uvedené varianty jsou ekvivalentní se základní definicí Turingova stroje.