# 19. Složitost algoritmu - konečný automat vs. Touringův stroj

## Složitost algoritmu

Jedna z možností jak porovnat algoritmy v určitých ohledech. Počet kroků nebo operací, které musí procesor vykonat pro různý počet vstupních hodnot. Vyjadřuje, jak se bude měnit chování algoritmu v závislosti na změně vstupních dat. O tom, zda je algoritmus realizovatelný, rozhodují tzv. **matematické modely**, které vyhodnocují složitost pomocí **vyčíslitelnosti** (teorii vyčíslitelnosti)

## Matematické Modely:

## Konečný automat – deterministický

Deterministic Finite Automaton (**DFA**) je matematický model, pomocí kterého lze určit, zda je daný algoritmus vyčíslitelný.

**Skládá se z:**

- „Paměti“ (matematický model paměť nemá; pouze při programování)
  - Proměnná
  - Uchovává aktuální stav konečného automatu
  - „Množinu vnitřních stavů“ ve kterých se může nacházet
- Vstupu
  - Čte jednotlivá data, podle kterých se rozhoduje pro další polohu konečného automatu
- Funkce
  - Metoda nebo tabulka, podle které rozhoduje o dalším kroku

Pokud lze na řešení problému použít konečný automat, je algoritmus vyčíslitelný.

Konečný automat musí mít minimálně **2 stavy**, stav pro začátek a stav pro konec.

Z **matematického** pohledu je Konečný automat Bez paměti, ale při simulování tohoto automatu je potřeba ukládat do nějaké proměnné aktuální stav. Při programování se místo 2 stavů používají 3 (4 – chyba): **bez přenosu, s přenosem a koncový**.

**Výhoda**

- Snadno realizovatelný

**Nevýhoda**

- Neumí moc věcí

**Použití**

- Binární sčítačka
- Lexikální analyzátory (v překladačích)

**Nedeterministický konečný automat**

- Pro stejný vstup existují 2 stavy
- Matematicky mocnější než **DKA**
- Umí simulovat více věcí než DKA
- Jednodušší na kreslení
- Nenaprogramovatelný

## Touringův stroj

Matematický model definovaný **Alanem Touringem**. Jedná se o nejsilnější matematický model.

**Skládá se z:**

- CPU
  - Uvnitř CPU se nachází konečný automat
- Čtecí a zapisovací hlavy
  - Mohou se posouvat na obě strany
- Paměti
  - Nekonečná páska
  - Zápis mezivýsledků

**Nevýhoda**

- Programátorsky nerealizovatelný
- Pouze teoretický návrh

**Použití**

- Násobení binárních čísel

**Church-Turingova Teze**

Podle této teze má každý algoritmus ekvivalentní Touringův stroj.

**Halting Problem – Problém zastavení**

*Znáte-li zdrojový kód programu a jeho vstup, rozhodněte, zda program zastaví, nebo zda poběží navždy bez zastavení.*

Problém vymyšlený Alane Touringem. Tento problém nedokáže vyřešit Touringův stroj. Alan Touring dokázal, že obecný algoritmus, který by řešil problém zastavení pro všechny vstupy všech programů, neexistuje.