Téma 19: Složitost algoritmu – Konečný automat vs. Touringův stroj

Složitost algoritmu udává, jak je daný algoritmus rychlý vzhledem k množině vstupních dat.

Algoritmická analýza se zabývá efektivitou algoritmů (jak z množiny možných algoritmů vybrat ten nejlepší).

Teorie složitosti je otázka efektivity algoritmů, složitost – jak je algoritmus rychlý.

Třída složitosti je obtížnost rozhodnutelnosti algoritmu na turingově stroji. Tříd složitosti je stovky.

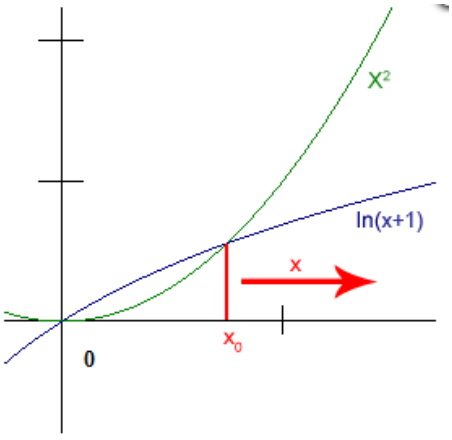
Ke klasifikaci algoritmů se obvykle používá tzv. **asymtotická složitost**, což je rozdělení algoritmů do tříd složitostí, u kterých platí, že od určité velikosti dat, je algoritmus dané třídy vždy pomalejší než algoritmus třídy předchozí, bez ohledu na to, jestli je některý z počítačů c-násobně výkonnější (c je konstanta).

Škála v nekonečnu slouží k rozlišení jednotlivých tříd. Říká, že pokud se n blíží k nekonečnu, tak neexistuje reálná konstanta taková, aby byl algoritmus z vyšší třídy rychlejší než ten z třídy přechozí.

$$1 \ll \log(n) \ll n \ll n \cdot \log(n) \ll n^k \ll k^n \ll n! \ll n^n$$

Pokud máme dva algoritmy o srovnatelné složitosti, první $O(n)$ a druhý $O(2n)$, tak nám stačí ten druhý pustit na 2x rychlejším stroji a nepoznáme rozdíl. Pokud ovšem nejsou ve stejné třídě složitosti, například jeden $O(n)$ a druhý $O(n^2)$, tak nám na srovnání výkonu nepomůže libovolně výkonný počítač, protože dvojnásobný objem dat bude druhému algoritmu trvat 4x tolik času, desetinásobný 100x tolik času.

Jednoduše řečeno: pokud spadají dva algoritmy do různých tříd asymptotické složitosti, pak vždy existuje takové množství dat, od kterého je asymptoticky lepší algoritmus vždy rychlejší, bez ohledu na to, kolikrát je některý z počítačů výkonnější.



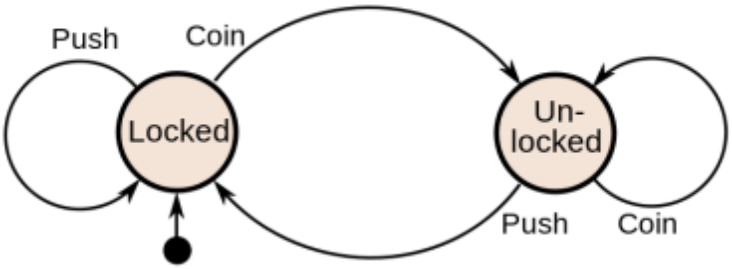
Na obrázku vidíme, že i kdybychom přenásobili libovolnou, byť velmi malou, konstantou c funkci x^2 (tj. zrychlovali bychom počítač, na němž tento algoritmus běží), tak vždy bude existovat bod x_0, od kterého bude algoritmus popsaný logaritmickou funkcí rychlejší na všech datech velikosti $x > x_0$. Změnou konstanty c bychom pouze posouvali bod x_0 po ose x.

Konečný automat je informatický model, pomocí kterého lze určit, zda je daný algoritmus vyčíslitelný. Pokud lze na řešení problému použít konečný automat, je algoritmus vyčíslitelný.

Konečný automat se skládá z paměti, v níž si uchovávám aktuální stav konečného automatu, vstupu, kde čtu jednotlivé data podle kterých se rozhoduji pro další polohu konečného automatu. A nějakou funkci, metodu nebo tabulku, podle které se rozhoduji, kam půjdu na další pozici.

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

Jako příklad konečného automatu je možné uvést následující obrázek, který je modelem turniketu, který se uvolňuje vložením mince. Počáteční stav je „Locked“ – turniket je uzamčen. Vložíme-li minci (automat přijme symbol „Coin“), dojde k přechodu do stavu „Unlocked“ – turniket se odemkne a můžeme projít. Jakmile projdeme (automat přijme symbol „Push“), turniket se opět uzamkne (přechod do stavu „Locked“). →



Turingův stroj je teoretický model počítače popsaný matematikem Alanem Turingem. Skládá se z procesorové jednotky, tvořené konečným automatem a pravostranné nekonečné pásky pro zápis mezivýsledků, kterou smí šoupat na obě strany. Využívá se pro modelování algoritmů v teorii vyčíslitelnosti. Jeden ze způsobu vyjádření Churchovy-Turingovy teze říká, že ke každému algoritmu existuje ekvivalentní Turingův stroj.

Oproti konečnému automatu má nekonečně velkou pásku, na kterou může zapisovat a libovolně se po ní může pohybovat. Turingův stroj je mnohem mocnější než konečný automat, protože umí počítat.

Máme posloupnost symbolů AAABBB…. Pouze Turingův stroj může rozpoznat posloupnost, ve který je libovolné množství symbolů A následované stejným množstvím symbolů B.

Univerzální Turingův stroj je Turingův stroj, který je schopný simulovat činnost libovolného TS. Turingův stroj sám o sobě může být zakódovaný na pásku stroje. Skládá se z konečného automatu (procesoru) a nekonečné pásky