7. Barevnost grafu

Definice 7.1. *Graf G se nazývá* k-obarvitelný, *jestliže každému jeho uzlu lze přiřadit jednu z "barev"* $1 \dots k$ *tak, že žádné dva sousední uzly nemají stejnou barvu.*

Definice 7.2. *Nejmenší přirozené číslo k, pro které je graf G k-obarvitelný, se nazývá* chromatické číslo (barevnost) *grafu G a značí se $\chi(G)$.*

Tvrzení 7.1. *Nechť G obsahuje jako podgraf úplný graf K_k. Pak*

$$\chi(G) \geq k.$$

Věta 7.1. *Pro každý graf G platí*

$$\chi(G) \leq \Delta(G) + 1,$$

kde $\Delta(G)$ je maximální stupeň grafu G.

Věta 7.2 (Brooks). *Pro každý graf G platí*

$$\chi(G) \leq \Delta(G)$$

až na tyto dvě výjimky:

I. G má komponentu $K_{(\Delta(G)+1)}$,

II. $\Delta(G) = 2$ a G má za komponentu kružnici liché délky.

Důsledek 7.1. *Je-li G souvislý graf, který není úplným grafem ani kružnicí liché délky, pak $\chi(G) \leq \Delta(G)$.*

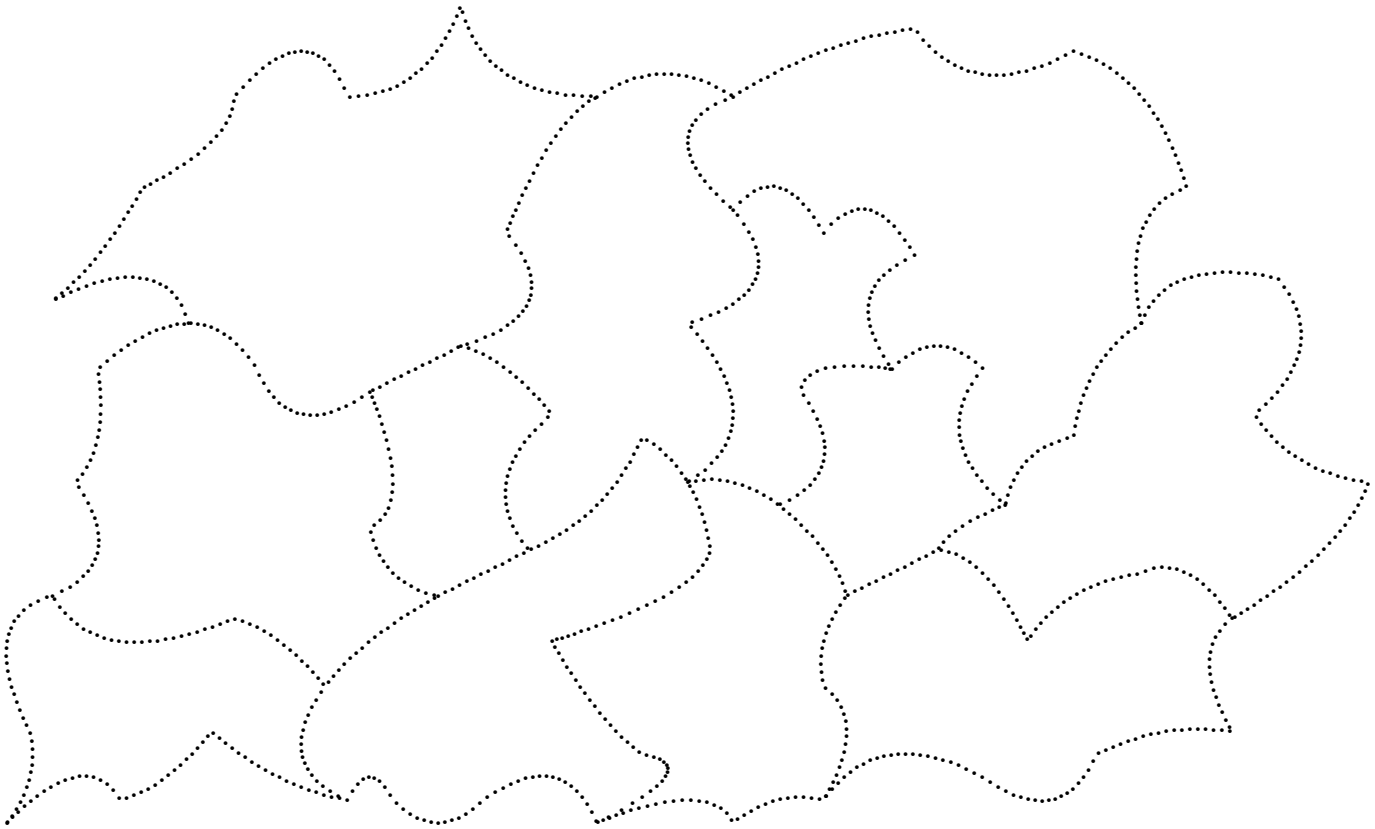
Věta 7.3. *Nechť G je graf. Pak je $\chi(G) = 2$ právě když $H(G) \neq \emptyset$ a G neobsahuje kružnici liché délky.*

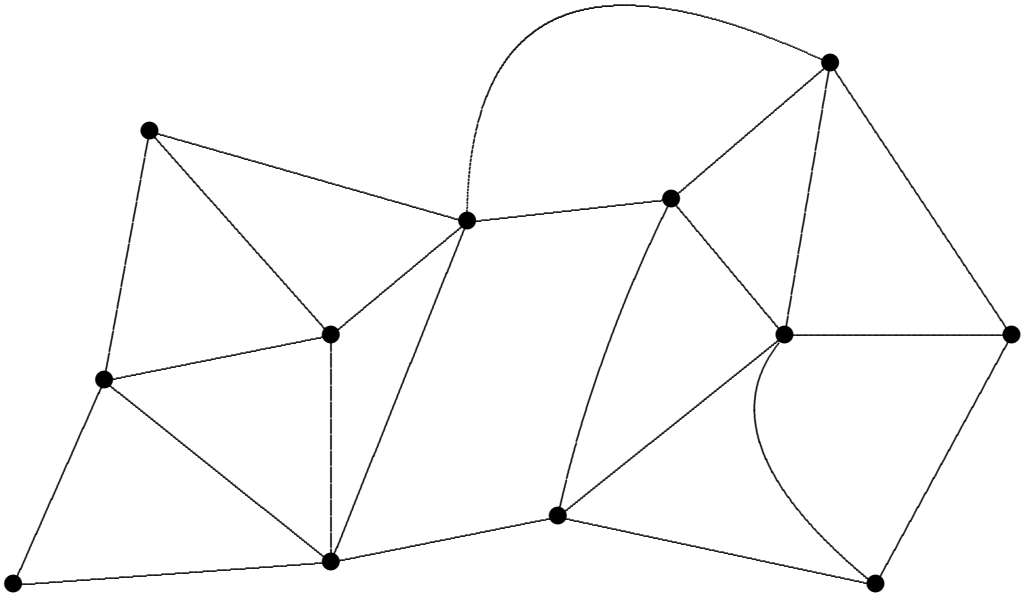
Věta 7.4. *Úloha: "určete, zda je daný graf G 3-obarvitelný" je NP-úplná.*

Věta 7.5. *Pro graf G s chromatickým číslem $\chi(G)$ a nezávislostí $\alpha(G)$ platí:*

1. $\chi(G)\ \alpha(G) \geq |U(G)|$,

2. $\chi(G) + \alpha(G) \leq |U(G)| + 1$.





Věta 7.6 (Haken, Appel). *Každý rovinný graf je 4-obarvitelný.*

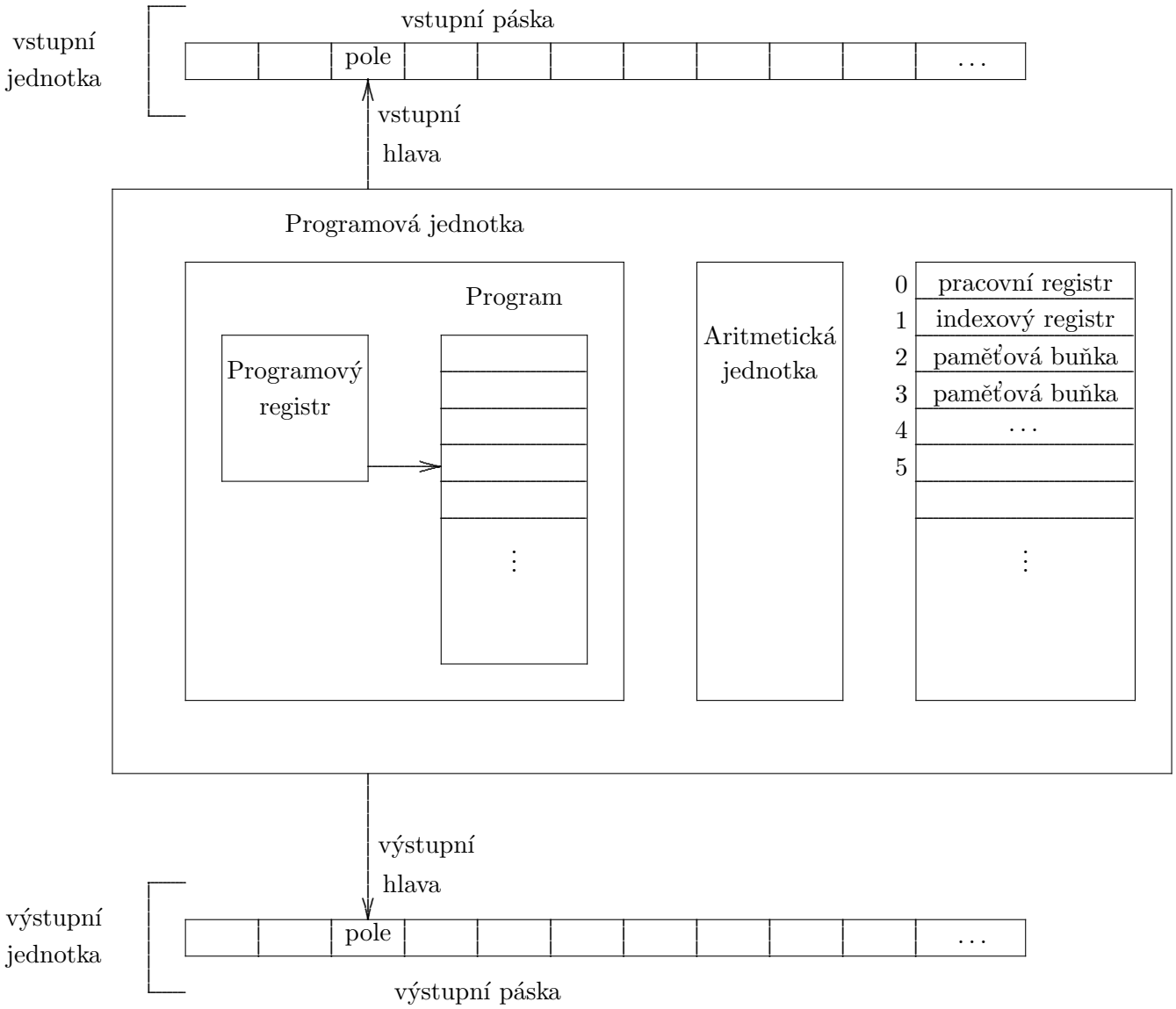
8. Modely výpočtu

Definice algoritmu:

- Turingův počítač
- rekurzivní funkce
- program v některém programovacím jazyku
- ...

„**Churchova teze**“: *každá úloha, algoritmizovatelná podle některé definice, je algoritmizovatelná i podle všech ostatních definic.*

8.1 Počítač s libovolným přístupem



V polích vstupní i výstupní pásky jsou celá čísla libovolně velká.

Paměťových buněk je neomezený počet a lze do nich vkládat neomezeně velká čísla.

Čísla před registry a paměťovými buňkami udávají adresu.

Konfigurace počítače: přiřazení, které každému

- poli vstupní pásky
- poli výstupní pásky
- paměťové buňce
- programovému registru

přiřazuje celé číslo (= popis okamžitého stavu počítače).

Počáteční konfigurace: existuje n takové, že

- pole vstupní pásky s adresami n, $n+1$, ...
- všechna pole výstupní pásky
- všechny paměťové buňky

obsahují nuly a programový registr má hodnotu 1
(tj. na polích vstupní pásky s adresami $0, \ldots, n-1$ jsou vstupní data.)

Výpočet počítače: posloupnost konfigurací C_0, C_1, ... taková, že C_0 je počáteční konfigurace, a *krok* je dán některým z *příkazů* (viz dále).

Program počítače: konečná posloupnost p_1, ... , p_n příkazů.

Příkazy

Přesuny v paměti

LOAD *operand*	do pracovního registru uloží hodnotu operandu (ostatní nezměněno)
STORE *operand*	do paměť. buňky s adresou rovnou adrese operandu uloží obsah prac. registru

Aritmetické příkazy

ADD *operand*	k obsahu prac. registru přičte hodnotu operandu
SUBTRACT *operand*	od obsahu prac. registru odečte hodnotu operandu
MULTIPLY *operand*	obsah prac. registru násobí hodnotou operandu
DIVIDE *operand*	obsah prac. registru dělí hodnotou operandu

Vstupy, výstupy

READ	do prac. registru dá obsah aktuálního pole vstupní pásky a posune hlavu o 1 vpravo
WRITE	obsah prac. registru uloží do aktuálního pole výstupní pásky a posune hlavu o 1 vpravo

Skoky

JUMP *návěští*	uloží návěští do programového registru
JZERO *návěští*	provede příkaz JUMP, pokud je obsah pracovního registru roven nule
JGE *návěští*	provede příkaz JUMP, pokud obsah prac. registru je ≥ 0

Návěštím rozumíme číslo instrukce či příkazu programu, tj. přirozené číslo.

Zastavení

STOP	ukončí výpočet
ACCEPT	ukončí výpočet, u rozhodovacích úloh má pravdivostní hodnotu 1
REJECT	totéž jako ACCEPT, ale dává hodnotu 0

Způsoby zadání operandu:

j (j je přirozené číslo nebo nula) - adresa operandu je j, hodnota operandu je obsah buňky s adresou j.

$*j$ (j je celé číslo) - adresa operandu je $i + j$, kde i je obsah indexového registru, hodnota je obsah buňky s adresou $i + j$.

$= j$ (j je celé číslo) - hodnota je j, adresa není definována.

Adresovací chyba: nastane, když se u operandu $*j$ objeví okamžitá hodnota $i + j$ záporná. Výpočet se v takovémto případě zastaví.

8.2 Časová a paměťová náročnost výpočtu

Definice 8.1. *řekneme, že výpočet počítače* trval dobu t, *jestliže*

- *v t-tém kroku došlo k:*
  - *provedení příkazu zastavení, nebo*
  - *adresovací chybě, nebo*
  - *dělení nulou,*
- *v krocích* $0, 1, \ldots, t-1$ *žádný z uvedených případů nenastal.*

Řekneme, že výpočet počítače pracoval s pamětí velikosti m, *jestliže*

- *nebyl proveden příkaz s adresou* $> m$,
- *byl proveden příkaz s adresou* $= m$.

Omezení 1. *Nechť p je pevně daný polynom. Připouštíme jen výpočty, pro něž v žádné buňce paměti není číslo v absolutní hodnotě větší než $p(\max\{n, |c_1|, |c_2|, \ldots, |c_n|\})$, kde $c_1, \ldots, c_n$ jsou vstupní data.*

Věta 8.1. *Nechť f je funkce a M počítač, který každou vstupní posloupnost délky n zpracuje v čase $f(n)$. Pak existuje počítač M', který zpracuje týž vstup v čase $O\left((f(n))^2\right)$ a v paměti $O(f(n))$ a dá týž výstup.*

Důsledek 8.1. *Jestliže existuje počítač, který každou vstupní posloupnost délky n zpracuje v polynomiálním čase, pak existuje počítač, který každou vstupní posloupnost zpracuje v polynomiálním čase i paměti.*

8.3 Problémy (jazyky) třídy P

Definice 8.2. Vstupními daty *nebo* slovem *nazveme konečnou posloupnost nul a jedniček.*
Délkou slova *rozumíme počet členů posloupnosti vstupních dat.*
Jazykem *nazveme konečnou (nebo i nekonečnou) množinu slov.*

Definice 8.3. Přijímací počítač *je počítač, který má následující dvě vlastnosti:*

(*i*) *jeho program neobsahuje příkazy WRITE ani STOP,*

(*ii*) *pro každé slovo w se výpočet zastaví po konečném počtu kroků provedením příkazů ACCEPT nebo REJECT (aniž by došlo k adresovací chybě nebo dělení nulou).*

Řekneme, že přijímací počítač přijímá *slovo w, pokud se výpočet zastaví příkazem ACCEPT* , a odmítá *slovo q, pokud výpočet skončí příkazem REJECT.*
Množina přijímaných slov se nazývá jazyk přijímaný počítačem.

Definice 8.4. *Nechť J je jazyk a $f : \mathbf{N} \to \mathbf{N}$. Časová (paměťová) složitost jazyka J je nejvýše f, jestliže existuje přijímací počítač M, který přijímá J a každé slovo jazyka J délky n zpracuje v čase (paměti) $f(n)$.*

Definice 8.5. Třída P *je třída všech jazyků J, pro něž existuje polynom p takový, že časová složitost jazyka J je nejvýše p.*