Karlova Univerzita v Praze, Fakulta Filozofie

Katedra Logiky

Obor Logika

Bakalářská práce

Bezestrojová charakterizace polynomiálně počitatelných funkcí

Machine-Free Characterization of Polynomially Computable Functions

Autor:
Michal Profeld

Vedoucí práce:
Doc. RNDr. Vítězslav Švejdar, CSc.

21. května 2017

Abstrakt

Tato bakalářská práce se zabývá bezestrojou definicí polynomiálních funkcí. Hlavním cílem je čtenáře obeznámit nejen s touto definicí, ale i s ostatními důležitými pojmy této práce. Nejdůležitějšími pojmy je myšleno: základní funkce, schéma skládání funkcí, rekuzivní schémata a polynomiální podmínky. Během práce bude čtenář mimo jiné svědkem odvození nejznámějších polynomiálně omezených funkcí, jako jsou násobení, sčítání, nebo jiné aritmetické funkce. Odvozeny však budou i zajímavější a netradiční funkce, jako je funkce smash, nebo mocnění v modulárním prostoru Z_n.

Abstract

This thesis focuses on machine-free definition of polynomial functions. The main goal is to not only make the readers familiar with this definition, but also to introduce them to the other pivotal terms of this thesis. The other pivotal terms are: basic functions, function composition, recursive schemes a polynomial conditions. Throughout the thesis the readers will be introduced, among other things, to derivation of the most used polynomially bounded functions, like multiplication, addition, or other arithmetic functions. Other interesting, less common and non-traditional functions, such as the Smash function or power function in modular space Z_n will be showcased.

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

V Praze dne

.....

Podpis

.....

Obsah

1 Úvodem

Tato práce se bude věnovat definici polynomiálních funkcí beze strojové definice. Práce bude postupovat souběžně s "A. Cobham. The Intrinsic Computational Difficulty of Functions,", nicméně formálně se bude držet spíše "Samuel R. Buss. Bounded Arithmetic"a bude lehce částečně inspirována prací "P. Mach. Bezestrojová charakterizace funkcí počitatelných v polynomiálním čase".

V druhé kapitole bude čtenář seznámen se základními strojovými definicemi. Mezi tyto termíny patří definice složitostních tříd (konkrétně definici třídy $\mathcal{O}(g)$ a třídy poly(g)), časové funkce stroje T a polynomomiálního času.

Na strojové definice navazuje definice sedmi zakladních funkcí, na kterých bude tato práce stavět. Mezi tyto funkce patří konstantní nulová funkce, funkce projekce proměnných, funkce následníka, dvě funkce pro bitové prosuny a dvě větvící funkce. Tyto funkce budou spolu se schématem pro skládánání funkcí detailněji představeny pomocí příkladů odvození několika jednoduchých funkcí (včetně odvození některých triviálních funkcí).

Ve čtvrté kapitole jsou definovány dvě rekurzivní schémata. Konkrétněji schéma pro délkovou rekurzi a schéma pro polynomiální rekurzi. Čtenáři budou opět představeny pomocí odvození několika jednoduchých funkcí. Tyto funkce budou poté použity v důkazu ekvivavelnce těchto schémat. Pomocí základních funkcí, schématu skládání funkcí a rekurzivních schémat definujeme pojem polynomiálně omezené funkce. Pomocí pojmu polynomiálně omezené funkce definujeme pojem polynomiální podmínky a polynomiální množiny. U polynomiálních podmínek dokážeme uzavřenost na výrokové logické operace a na omezenou kvantifikaci. V šesté kapitole jsou vyjmenovány nejznámější funkce, které počítají v polynomiálním čase. O těchto funkcích je dokázáno, že jsou polynomiálně omezenými funkcemi. Mezi zmíněné funkce patří například sčítání, násobení, nebo funkce modulo. Zmíněny jsou ovšem i zajímavější funkce, jako je funkce smash, nebo funkce pro mocnění v prostoru $\mathrm{Z_n}$. Poslední kapitola bude vedena v neformálním duchu v zájmu vyhnutí se zbytečnému formalizmu. Nejdříve budou nastíněny možnosti kódování posloupností pomocí gödelova čísla pro posloupnost. Zmíněny budou jednotlivé funkce, které jsou třeba odvodit, aby bylo kódování použitelné.

2 Počáteční definice

V této kapitole definujme základní strojové definice pro polynomiálně počitatelné funkce. Mezi tyto pojmy patří výpočetní třídy $\mathcal{O}(g)a\text{poly}(g)$, časová funkce stroje T, definice polynomiálního času. Definice jsou přímo přejaté z [6]. Definujme třídy funkcí na N takto:

Definition 2.1. Třídy funkcí na N
Nechť $g : \mathrm{N} \to \mathrm{N}$. Definujme třídy funkcí:

$$\mathcal{O}(g) := \{f : \mathrm{N} \to \mathrm{N} : \text{existuje přirozené číslo } k > 0 \text{ tak, že } f(n) \leq k \cdot g(n)\},$$
$$\text{poly}(g) := \{f : \mathrm{N} \to \mathrm{N} : \text{existuje polynom } p(n) \text{ tak, že } f(n) \leq p(g(n))\}.$$

Dále definujme časovou funkci stroje T, za stoj berme pro jednoduchost jednosměrný jednopáskový turingův stroj. Definici turingova stroje lze nalézt například zde[6].

Definition 2.2. Časová funkce stroje T
Definujme funkci $f : N \to \mathrm{N}$ určenou k výpočtu časové složitosti stroje nad vstupním slovem $y \in \{1, 0\}^n$.

$$f(n) = max\{\text{výpočetní čas nad slovem } y \in \{1, 0\}^n\}$$

Na závěr kapitoly definujme nejdůležitější z definic v této kapitole. Definici stroje pracujícího v polynomiálním čase.

Definition 2.3. Polynomiální čas
Řekneme, že stroj T pracuje v čase (nanejvýš) řádově $g(n)$, jestliže jeho časová funkce $f(n) = \mathcal{O}(g(n))$. Řekněme, že stroj T pracuje v polynomiálním čase, je-li $f(n) = \text{poly}(n)$.

3 Definiční obory v této práci

Dříve než začneme s bezestrojovými definicemi, tak definume definiční obory funkcí v této práci. V dalším textu se omezíme pouze na přirozená čísla. Pokud nebude řečeno, pak lze o každé funkce f s k proměnnými předpokládat, že se jedná o funkci $f : \mathrm{N}^k \to \mathrm{N}$.

4 Množina základních funkcí

V této kapitole definujeme množinu elementárních funkcí inspirovanou převzatou ze zdrojů [1], [3] a doplněnou funkci projekce z [4]. Dále definjeme schéma pro skládání funkcí převzaté z [4]. Jednotlivé elementární funkce si představíme a pomocí skládání funkcí odvodíme pro ukázku několik jednoduchých funkcí.

Definition 4.1. Definice základních funkcí Definujme 7 základních funkcí. Seznam 6 základních funkcí je převzatý z prací [1] a [3]. V rámci zachování formalizmu doplněna 7. Funkce z knihy [4], která slouží pro práci s parametry funkce.

(1) $\mathrm{z} : \underline{x} \longmapsto 0$ (Konstantní nulová funkce)

(2) $\mathrm{i}_j^n : \underline{x} \longmapsto x_j$ (Projekce proměnných)

(3) $\mathrm{s}(x) : \underline{x} \longmapsto x + 1$ (Funkce následníka)

(4) $\mathrm{asr}(x) : \underline{x} \longmapsto \lfloor x/2 \rfloor$ (Bitový posun vpravo)

(5) $\mathrm{asl}(x) : \underline{x} \longmapsto x \cdot 2$ (Bitový posun vlevo)

(6) $x \leq y : \underline{x} \longmapsto \begin{cases} 1 & \text{if } x \leq y \\ 0 & \text{if } x > y \end{cases}$ (Menší rovno)

(7) $\mathrm{Choice}(x, y, z) : \underline{x} \longmapsto \begin{cases} y & \text{if } x > 0 \\ z & \text{if } x = 0 \end{cases}$ (Výběrová funkce)

Když máme definovány základní funkce, definujme schéma pro jejich skládání definované v [4]. Schéma skládání slouží k složení funkce g s dalšími funkcemi. Každá z funkcí je nasycena pomocí k proměnných. Tyto na nasycené funkce jsou následně dosazeny do funkce g.

Definition 4.2. Skládání funkcí
Nechť g je funkcí n proměnných a $f_1 \ldots, f_n$ jsou funkcemi k proměnných. Pak pro odvození funkce f (n proměnných) pomocí složení funkcí g a $f_1, \ldots, f_n$ (značme $g \circ [f_1, \ldots, f_n]$ pro $n > 1$ a $g \circ f_1$ pro $n = 1$). Pro každé z a $\underline{x}$ platí

$$f(\underline{x}) = z$$

Právě tehdy když platí:

$$g(f_1(\underline{x}), \ldots, f_n(\underline{x})) = z$$

4.1 Rozbor množiny základních funkcí

V této kapitole se detailněji podíváme na základní funkce z minulé kapitoly. Jednotlivé funkce si představíme detailněji a ukážeme si příklady použití skládání funkcí. Mimo jiné si v této kapitole představíme a odvodíme některé z triviálních operací převzatých z [4]. V práci však nebudou odvozeny všechny z primitivních operací vzhledem k tomu, že jejich odvození je spíše technickým důkazem. Pokud by čtenář měl zájem, pak může najít odvození všech čtyř funkcí v [4]. Jako první z primitivních operací odvoďme operace přidání jalové proměnné(terminologie převzata ze zdroje). Tato operace slouží pro k odvození funkce $k+1$ proměnných z funkce k proměnných.

4.1.1 Projekce proměnných

Projekce

Theorem 4.3. *Schéma přidání jalové proměnné je odvoditelné pomocí základních funkcí a schématu pro skládání funkcí*

Proof. Nechť f je funkce $\mathrm{N}^n \to \mathrm{N}$, kde $n \geq 0$. Definujme funkci $g : \mathrm{N}^{n+1} \to \mathrm{N}$, pro kterou platí $\forall x, y \in \mathrm{N} : f(\underline{x}) = g(\underline{x}, y)$.

$$g(x_1, \ldots, x_{n+1}) = f(\mathrm{i}_1^n, \ldots, \mathrm{i}_n^n)(x_1, \ldots, x_k)$$

4.1.2 Funkce následníka

Funkce následníka je funkcí jedné proměnné, která přičte jedničku k číslu x.

Odvoďme schéma pro odvození libovolné konstantní funkce. Funkce je odvozeny pomocí složení konstantní funkce pro nulu, která je c-krát složena s následnickou funkcí. Za zmínku stojí, že funkce pro odvození konstantní funkce není jedna, nicméně každá z konstantních funkcí musí být odvozena samostatně.

Theorem 4.4. *Konstatní funkce jsou odvoditelné pomocí základních funkcí a schématu pro skládání funkcí*

Proof. Nechť $c \in \mathrm{N}$ je libovolná konstanta. Odvoďme konstantní funkci $f : \mathrm{N} \to \mathrm{N}$ pro tuto konstantu následovně:

$$c(x) = (\underbrace{\mathrm{s} \circ \cdots \circ \mathrm{s}}_{\text{c-krát použita funkce s}} \circ \mathrm{z})(x)$$

Čímž dostaneme funkci generující konstantu c.

Podobným způsobem, jako jsou odvozeny konstantní funkce, lze odvodit schéma pro přičtení konstanty k číslu. Funkce pro přičtení konstanty je odvozena pomocí c použití následnické funkce na proměnnou x. Opět se jedná o schéma a je tedy třeba pro každé c odvodit samostatnou funkci.

Theorem 4.5. *Funkce pro přičtení konstanty(x+c) jsou odvoditelné pomocí základních funkcí a schématu pro skládání funkcí*

Proof. Nechť c je libovolná konstanta a nechť x je vstupní proměnná, pak funkci $f : \mathrm{N} \to \mathrm{N}$ definujme takto:

$$f(\underline{x}) = (\underbrace{\mathrm{s} \circ \cdots \circ \mathrm{s}}_{\text{c-krát použita funkce s}}) \circ \mathrm{i}_1^1)(x_1, \ldots, x_k)$$

Druhou z primitivních operací je schéma pro dosazení konstanty do funkce. Uvažujme funkci k proměných, kde $k > 0$. Pomocí operace dosazení konstanty do funkce vytvoříme funkci $k-1$ proměnných f_1, jelikož dosadíme konstantu c za jednu z proměných. Schéma je odvozené manipulací proměnných pomomocí funkce projekce proměnných.

Theorem 4.6. *Schéma pro dosazení konstanty do funkce je odvoditelná pomocí základních funkcí a schématu pro skládání funkcí*

Proof. Nechť f je funkce $\mathrm{N}^n \to \mathrm{N}$, kde $n{>}0$. Definujme funkci $g : \mathrm{N}^{n-1} \to \mathrm{N}$ pro kterou platí: $\forall x_1, \ldots, x_n \in \mathrm{N} : f(x_1, \ldots, x_{j-1}, c, x_{j+1}, \ldots, x_n) = g(x_1, \ldots, x_{n-1})$, kde c, j jsou předem zvolené konstanty.

$$g(x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n) = f(\mathrm{i}_0^n, \ldots, \mathrm{i}_{j-1}^n, c, \mathrm{i}_{j+1}^n, \ldots, \mathrm{i}_n^n)(x_1, \ldots, x_k)$$

4.1.3 Bitové posuny

Bitové posuny jsou funkcemi dvou proměnných, které pracují s bitovou podobou čísla. Bitový posun vpravo zkrátí číslo o poslední (nejméně významný bit), zatímco bitový posun vlevo přidá k číslu jeden nulový bit zprava (na místo nejméně významného bitu). Pokud použijeme na číslo x bitový posun vpravo a poté na výsledek bitový posun vlevo, pak dostaneme opět číslo x. Definume takovýmto způsobem funkci identity.

Theorem 4.7. *Funkce identity* - $(\mathrm{Id}(x))$ *je odvoditelná pomocí základních funkcí a schématu pro skládání funkcí*

Proof. Nechť x je proměnná. Odvoďme funkci identity $f : \mathrm{N} \to \mathrm{N}$

$$\mathrm{asr}(\mathrm{asl}(\underline{x}))$$

Definici bitových posunů umožňuje bitový posun pouze o jeden bit. V pozdějších kapitolách této práce odvodíme bitové posuny vlevo a vpravo o délku vstupní proměnné. Nyní odvoďme bitové posuny o konstatní počet bitů.

Theorem 4.8. *Konstantní bitový posun vlevo ($x \times 2^c$) a vpravo($\frac{x}{2^c}$) o c bitů jsou odvoditelné pomocí základních funkcí a schématu pro skládání funkcí*

Proof. Odvození probíhá pomocí složení c funkcí stejně jako v sekci 4.1.2

4.1.4 Větvící funkce (funkce menší rovno a funkce choice)

Poslední dvě základní funkce jsou funkcemi umožňující větvení. Funkce budou používány v dalších kapitolách a to primárně v kombinaci s délkovou rekurzím, která pracuje s jednotlivými bity.

5 Definice délkově rekurzivních funkcí

V této kapitole definujeme dvě rekurzivní schémata. Kokrétněji schéma pro délkovou rekurzi a schéma pro polynomiální rekurzi. Definice rekuzivnách schémat jsou převzata ze zdrojů [1] a [3] Každé ze schémat následně popopíšeme a uvedeme několik příkladů použití, aby se s nimi čtenář lépe obeznámil. Následně s použitím odvozených funkcí dokážeme ekvivalenci obou schémat. Prvním rekurzivním schématem je schéma délkové rekurze:

5.1 Definice rekuzivních funkcí

Definition 5.1. Délková rekurze Nechť g je libovolná funkce $g : \mathrm{N}^k \to \mathrm{N}$ pro $k \leq 0$ a f je libovolná funkce $f : \mathrm{N}^{k+2} \to$ Npro $k \leq 0$. Dále nechť q je libovolný vhodný polynom. Pak funkce $f : \mathrm{N}^k \to \mathrm{N}$ je odvozena omezenou rekurzí z funkcí f,g s časovým omezením $|y|$ a prostorovým omezením pomocí polynomu q, jestliže pro libovolné $|y|$ platí následující:

- $f(\underline{x}) = \tau(\underline{x}, |y|)$
- $\tau(\underline{x}, 0) \ = \ g(\underline{x})$
- $\tau(\underline{x}, n+1) \ = \ h(\underline{x}, n, \tau(\underline{x}, n))$

A navíc:
$\forall n \leq |y|(|\tau(\underline{x}, n)| \leq q(|\underline{x}|, |y|))$

Schéma délkové rekurze pracuje s množinou parametrů $\underline{x}$ a hloubkovou proměnnou y. Rekuze v tomto schématu má celkovou hloubku $|y|$. Aktuální hloubka rekurze je zachycena proměnnou n, která nabývá hodnot 0 až $|y|$ (kde 0 představuje nejhlubší část rekurze). Pro $n = 0$ je hodnota funkce definována, jako výsledek funkce $g(\underline{x})$. Hodnota funkce pro n je definována, jako hodnota fukce h, do které je dosazena aktuální hloubka n, hodnota funkce v hloubce $n-1$ a množina proměnných $\underline{x}$. Pro funkci musí existovat polynom q takový, že pro libovolnou hloubku n musí platit $(|\tau(\underline{x}, n)| \leq q(|\underline{x}|, |y|))$

Definition 5.2. Polynomiální rekurze
Nechť g je libovolná funkce $g : \mathrm{N}^k \to \mathrm{N}$ pro $k \leq 0$ a f je libovolná funkce $f : \mathrm{N}^{k+2} \to \mathrm{N}$ pro $k \leq 0$. Dále nechť p je libovolný vhodný polynom. Pak funkce $f : \mathrm{N}^k \to \mathrm{N}$ je odvozena polynomiální rekurzí z funkcí f,g s časovým omezením pomocí polynomu q, jestliže platí následující:

- $f(\underline{x}, y) \ = \ h(\underline{x}, y, f(\underline{x}, \lfloor y/2 \rfloor))$
- $f(\underline{x}, 0) \ = \ g(\underline{x})$

A navíc:
$|f(\underline{x}, y)| \leq q(|\underline{x}|, |y|)$

Schéma polynomální rekurze pracuje s přidáním jednoho bitu na konec čísla. Celková hloubka rekurze je opět definována jako $|y|$. V nulté hloubce je hodnota funkce f definována, opět pomocí funkce $g(\underline{x})$. pro $y \gneqq 0$

5.2 Pomocné funkce

V této podkapitole provedeme odvození několika jednoduchých funkcí, které budou použity v důkazu ekvivalence rekurzivních schémat. Pokusme se odvodit funkci $x_1 \ll x_2$.

Excersise 5.3. Funkce $x_1 \ll x_2$ není odvoditelná pomocí ze základních funkcí a pomocí schémat rekurze, skládání funkcí.

Uvažujme délkovou funkci, pomocí které omezíme délku funkce. Vzhledem k tomu, že délka výsledné funkce je $x \times 2^c$, tak je jasné, že polynomiální funkcí omezit nejde. Jedná se jednoznačně o funkci exponciální.

Funkce $x_1 \ll x_2$ je funkcí exponenciální a z tohoto důvodu se nám jí nepodařilo odvodit. Uvažujme nyní funkci $x_1 \ll |x_2|$, která již exponenciální není. Funkce bude odvozena délkovou rekurzí podle délky proměnné $|x_2|$. Za každý bit čísla x_2 bude na číslo x_1 použita funkce asl.

Theorem 5.4. *Funkce $f : \mathrm{N}^2 \to \mathrm{N}$ pro bitový posun vlevo o y bitů $(x_1 \ll |x_2|)$ je odvoditelná pomocí ze základních funkcí a pomocí schémat rekurze, skládání funkcí.*

Proof. pomocí rekurze

- $\underline{x} = \{x_1, x_2\}$
- $y = x_2$
- $g(\underline{x}) = \underline{x}$
- $h(\underline{x}, n, \tau(\underline{x}, n)) = \tau(\underline{x}, n) \ll 1$

Opět zkusme rozdělit funkci na dvě funkce a obě omezit polynomem. $p_1 = x_1$ a $p_2 = |2^{|x_2|}| = |x_2|$. Neboli $q = |x_1| + |x_2| = |\underline{x}|$

Stejným způsobem můžeme odvodit funkci $x_1 \gg |x_2|$. Funkce bude odvozena délkovou rekurzí podle délky proměnné $|x_2|$. Za každý bit čísla x_2 bude na číslo x_1 použita funkce asr.

Theorem 5.5. *Funkce $f : \mathrm{N}^2 \to \mathrm{N}$ pro bitový posun vpravo o $|y|$ bitů $(x_1 \gg |x_2|)$ je funkce odvoditelná pomocí ze základních funkcí a pomocí schémat rekurze, skládání funkcí.*

Proof. Odvoďme pomocí délkové rekurze.

- $\underline{x} = \{x_1, y\}$
- $y = y$
- $g(\underline{x}) = \underline{x}$
- $h(\underline{x}, n, \tau(\underline{x}, n)) = \tau(\underline{x}, n) \gg 1$

Vzhledem k tomu, že délku čísla x_1 zmenšujeme, tak můžeme omezit polynomem $q = (|x_1|)$.

Odvoďme funkci pro podmíněné odečtení jedničky. Algoritmus pro odečtení čísla jedna spočívá v inverzi všech nulových bitů na konci čísla až po první nenulový bit(včetně tohot bitu). Odvození probíhá pomocí délkové rekuze probíhá tak, že jsou umazány všechny nulové bity. Dále je invertován první nenulový bit a číslo na číslo je používáná funkce asl dokud $\text{asl}(\tau(\underline{x}, n)) \leq x$ (dokud by použitím funkce asl nezniklo číslo větší než x) .

Theorem 5.6. *Funkce $f : \mathrm{N}^2 \to \mathrm{N}$ pro odečtení jedničky $(x \dot{-} 1)$ je funkce odvoditelná pomocí ze základních funkcí a pomocí schémat rekurze, skládání funkcí.*

Proof. Nejdříve si připravíme funkci f, která smaže všechny nuly na konci čísla.

- $\underline{x} = \{x\}$
- $y = x$
- $g(\underline{x}) = \underline{x}$
- $h(\underline{x}, n, \tau(\underline{x}, n)) = \begin{cases} \text{asr}(\tau(\underline{x}, n)) & \text{if } \tau(\underline{x}, n)[0] = 0 \\ \tau(\underline{x}, n) & \text{if Jinak} \end{cases}$

Vzhledem k tomu, že délku čísla x_1 zmenšujeme, tak můžeme omezit polynomem $q = |x|$
Dále si odvodíme funkci $\dot{-}1$, tak že jedničku na koci čísla přepíšeme na nulu a doplníme adekvátní počet nul pomocí rekurze. Za zmínku stojí fakt, že takováto funkce nijak neovlivní číslo 0.

- $\underline{x} = \{x\}$
- $y = x$
- $g(\underline{x}) = \text{AddToEnd}(\text{asr}(f(x)), 0)$
- $= h(\underline{x}, n, \tau(\underline{x}, n)) = \begin{cases} asl(\tau(\underline{x}, n)) & if \ asl(\tau(\underline{x}, n)) \leq x \\ \tau(\underline{x}, n) & if \ Jinak \end{cases}$

Vzhledem k tomu, že délku čísla x_1 zmenšujeme, tak můžeme omezit polynomem $q = |x|$

V další podkapitole budeme dokazovat rovnost polynomiální a délkové rekurze. Pro důkaz této rovnosti bude potřeba odvodit funkci délkového odčítání, neboli rozdíl čísla x_1 a délky čísla x_2. Odvození probíhá pomocí délkové rekurze, tak že je za každý bit čísla x_2 od čísla x_1 odečtena jednička.

Theorem 5.7. *Funkce $f : \mathrm{N}^2 \to \mathrm{N}$ pro délkové odčítání $(x_1 - |x_2|)$ je funkce odvoditelná pomocí ze základních funkcí a pomocí schémat rekurze, skládání funkcí.*

.

- $\underline{x} = \{x_1, x_2\}$
- $y = x_2$
- $g(\underline{x}) = x_1|$
- $h(\underline{x}, n, \tau(\underline{x}, n)) = \begin{cases} \tau(\underline{x}, n)) \dot{-} 1 & \textit{if } |n| \leq x_2 \\ \tau(\underline{x}, n)) & \textit{if Jinak} \end{cases}$

Vzhledem k tomu, že délku čísla $|x_1|$ zmenšujeme, tak můžeme omezit polynomem $q = (|\underline{x}|)$.

6 Ekvivalence délkově rekurzivních funkcí

V této sekci ukážeme rovnost rekurzivních schémat. Při odvozování budeme značit odvozované schéma a jeho proměnné s aspostrofy, aby se mohli ve funkcích vyznat.

Theorem 6.1. *5.1 $\longmapsto$ 5.2*

Důkaz.
Nejdříve zvolme vhodně proměnné pro dosazení do funkce 5.1

- $\underline{x}' = \underline{x}$,
- $y' = y$ (v nultém kroku rekurze)
- $g'(\underline{x}') := g(\mathrm{i}_1^1)$
- $h'(\underline{x}', y', f'(\underline{x}', \lfloor y'/2 \rfloor)) := h(\mathrm{i}_1^3, |\mathrm{i}_2^3|, \mathrm{i}_3^3)$

Dokažme ekvivalenci na nulté hladině rekurze. Tato ekvivalence je triviální, vzhledem k tomu že byla zvolena totožná funkce g a do ní dosazena totožná množina $\underline{x}$.

$$f'(\underline{x}', 0) = \ g'(\underline{x}') = g(\mathrm{i}_1^1) = g(\underline{x}') = g(\underline{x}) = \tau(\underline{x}, 0) = f(\underline{x}, 0)$$

Dále si dokažme ekvivalenci $f(\underline{x}', y') = f(\underline{x}, y)$

$$\begin{aligned} f'(\underline{x}', y') &= h'(\underline{x}', y', f'(\underline{x}', \lfloor y'/2 \rfloor)) = h(\mathrm{i}_1^3, |\mathrm{i}_2^3|, \mathrm{i}_3^3) = h(\underline{x}', |y'|, \tau'(\underline{x}', n')) \\ &= h(\underline{x}, n, \tau(\underline{x}, n)) = \tau(\underline{x}, n+1) = f(\underline{x}, y) \end{aligned}$$

Funkce h' na n-té hladině rekurze dosadí proměnné $\underline{x}$, $|y|$ a $\lfloor y'/2 \rfloor$) do funkce h(do funkce h, která byla použita při odvození pomocí délkové rekurze). Vzhledem k tomu, že $|y|$ je současnou hloubkou rekuze a, že $\lfloor y'/2 \rfloor$) je výsledkem funkce f v hloubce $n-1$, tak jsme funkci h předali stejné parametry, jako funkce τ při odvození pomocí délkové rekurze. Z tohoto důvodu je jasné, že pro všechny $\underline{x}, n$ platí $f'(\underline{x}', y') = \tau(\underline{x}, n+1)$, což jsme chtěli dokázat.

(Jinými slovy jsme opět použili volání funkce h tentokrát z důkazu pomocí polynomiální rekurze a této funkci jsme předali stejné parametry, jako polynmiální rekurze. Díky čemuž Dostaneme na každé úrovni rekurze stejný výsledek, jako polynomiální rekurze.

□

Proof. $f'''(x_1, x_2)$ zkrátí číslo x na délku y lze odvodit pomocí délkové rekurze

Důkaz.

- $\underline{x} = \{x_1, x_2\}$
- $y = x_1$
- $g(\underline{x}) = x_1$
- $h(\underline{x}, n, \tau(\underline{x}, n)) = \begin{cases} \text{asr}(tau(\underline{x}, n)) & \text{if } tau(\underline{x}, n) > x_2 \\ tau(\underline{x}, n) & \text{if Jinak} \end{cases}$

Vzhledem k tomu, že délku čísla $|x_1|$ zmenšujeme, tak můžeme omezit polynomem $q = (|\underline{x}|)$. □

Theorem 6.2. *4.2* $\longmapsto$ *4.1*

Důkaz.

- $\underline{x}' = \underline{x}, y$ (kde se jedná o y v nultém kroku rekurze)
- $y' = y$ (v nultém kroku rekurze)
- $g'(\underline{x}, y) = g(\mathrm{i}_1^1)$
- $h'(\underline{x}, y, y', f'(\underline{x}', \lfloor y'/2 \rfloor)) =$

Dokažme ekvivalenci na nulté hladině rekurze. Tato ekvivalence je triviální, vzhledem k tomu že byla zvolena stejná funkce g a do ní dosazena stejná množina $\underline{x}$.

$$f'(\underline{x}', 0) = \tau'(\underline{x}', 0) \;=\; g'(\underline{x}') = g'(\underline{x}, y) = g(\mathrm{i}_1^1) = g(\underline{x}) = f(\underline{x}, 0)$$

Rád bych není probral jednotlivé rovnosti v této ekvivalenci. První a druhá rovnost plyne z definice délkové rekuze. Třetí rovnost plyne z toho, jak jsme si definovali x. Čtvrtá rovnost plyne z toho, jak jsme si nastavili funkci g'. Pátá rovnost plyne z definice projekce proměných. Šestá rovnost pak plyne z definice funkce $\underline{x}'$. Šestá rovnost plyne z definice délkové rekurze. Tímto jsme dokázali bázi indukce.

$$\begin{aligned} f'(\underline{x}', y, y') &= \tau'(\underline{x}, y, n'+1) = h'(\underline{x}, y, n', \tau'(\underline{x}, y, n')) = h(\mathrm{i}_1^4, f'''(\mathrm{i}_2^4, \mathrm{i}_3^4), \mathrm{i}_4^4) \\ &= h(\underline{x}, f'''(y, n'), \tau(\underline{x}, n)) = h(\underline{x}, y, f(\underline{x}, \lfloor y/2 \rfloor)) = f(\underline{x}, y) \end{aligned}$$

Opět proberme rovnosti. První rovnost plyne z naší definice x a definice délkové rekurze. Druhá rovnost vyplývá z definice délkové rekurze. Třetí z rovnost vyplývá z toho jak jsme si definovali funkci h. Čtvrtá rovnost vyplývá z definice funkce projekce. Další rovnost je složitější. Rovnost prvních argumentů je triviální. Druhé argumenty jsou výsledkem definice funkce f'''. Rovnost třetích argumentů vyplývá z indukčního předpokladu. Poslední funkce vyplývá z definice plynomiální funkce. Rovnost délkových omezení je triviální.

□

7 Polynomiálně počitatelné funkce s polynomiální podmínky

V této kapitole definujme pojmy polynomiálně počitatelné funkce, polynomiální podmínky a pojem polynomiální množiny. Definice polynomiální podmínky je inspirována knihou[4], definice polynomiálně počitatelné funkce je převzata z [1]. Kapitola se bude zabývat především rozborem polynomiálních podmínek. Bude dokázaná uzavřenost polynomiálních podmínek na výrokové operace a omezenenou kavantifikaci (které bude také definována v této kapitole).

Definition 7.1. Polynomiálně počitatelné funkce
Definujme množinu polynomiálně počitatelných funkcí, jako množinu všech funkcí, které jsou ze základních funkcí odvozeny pomocí schémat skládání funkcí, délkové rekurze a polynomiální rekurze. Následující definice charakteristické funkce je převzata z [6].

Definition 7.2. Charakteristická funkce
Charakteristickou funkcí množiny A je funkce(s definičním oborem A) $\chi_A : A \to \{0, 1\}$ definovaná předpisem

$$\chi_{\mathrm{A}}(A) = \begin{cases} 1 & \text{if } x \in A \\ 0 & \text{if Jinak} \end{cases}$$

Definition 7.3. Polynomiální množina a polynomiální podmínka
Definujme polynomiální množinu, jako libovolnou množinu, jejíž charakteristická funkce je polynomiálně počitatelnou funkcí. Takovouto charakteristickou funkci budeme nazývat polynomiální podmínkou.

Uveďme důkazy uzavřenosti polynomiálních podmínek na jednotlivé operace. Samotné důkazy nejsou nijak složité a jsou jednoduše pochopitelné. Principem důkazů je fakt, že polynomiální podmínkou jsou zároveň polynomiálně počítatelnými characterisctickými funkcemy a proto na ně můžeme aplikovat schémata pro skládání funkcí a rekurzivní schémata. Pokud bude výsledná funkce f funkcí do množiny $\{0, 1\}$, pak je polynomiální podmínkou.

Theorem 7.4. *Uzavřenost polynomiálních podmínek na negaci($\neg\varphi$)*

Proof. Nechť φ je polynomiální podmínkou. Pak polynomiální podmínku ψ, jež reprezentuje její negaci, odvodíme takto:

$$\psi = \varphi \leq 0$$

Theorem 7.5. *Uzavřenost polynomiálních podmínek na konjunkci($\varphi \wedge \psi$)*

Proof. Nechť φ a ψ jsou polynomiální podmínky. Pak polynomiální podmínku χ, jež reprezentuje jejich konjunkci, odvodíme takto:

$$\chi = \text{Choice}(\varphi, \psi, 0)$$

Theorem 7.6. *Uzavřenost polynomiálních podmínek na disjunkci($\varphi \vee \psi$)*

Proof. k Nechť φ a ψ jsou polynomiální podmínky. Pak polynomiální podmínku χ, jež reprezentuje jejich disjunkci, odvodíme takto:

$$\chi = \neg(\neg\varphi \wedge \neg\psi)$$

Theorem 7.7. *Uzavřenost polynomiálních podmínek na všechny výrokové logické operace*

Proof. V předchozích důkazech bylo dokázána uzavřenost na negace, disjunkci a konjunkci. Vzhledem k tomu, že tyto tři operace tvoří úplný systém logických spojek, tak máme dokázáno.

Definition 7.8. Délkově omezená kvantifikace

Nechť $(\psi)(\underline{x})$ je polynomiální podmínkou a nechť $x, y \in \underline{x}$, pak definujme délkově omezenou kvantifikaci jako jednu z formulí ve tvaru:

- $\forall y \leq |x|(\psi)(\underline{x})$
- $\forall y < |x|(\psi)(\underline{x})$
- $\exists y \leq |x|(\psi)(\underline{x})$
- $\exists y < |x|(\psi)(\underline{x})$

V následujícím odvození bude z polynomiální podmínky $(\psi)(\underline{x}, y)$ odvozena polynomiální podmínka $\forall y \leq |x|(\psi)(\underline{x}, y)$. Odvození proběhne odvozením funkce $(\chi)(z, \underline{x})$ pomocí polynomiální rekurze dle proměnné z. v každém rekurzivním kroku bude do funkce (ψ) dosazena hodnota proměnné z a výsledek bude zkonjunktován s $(\chi)(\lfloor z/2 \rfloor, \underline{x})$. Bude tedy platit:

$$(\chi)(y, \underline{x}) = (\psi)(\underline{x}, 0) \wedge (\psi)(\underline{x}, 1) \wedge, \ldots, \wedge(\psi)(\underline{x}, y)$$

Definition 7.9. Polynomiální podmínky jsou uzavřeny na omezenou kvantifikaci

Proof. Nechť $(\psi)(\underline{x}, y)$ je polynomiální podmínkou. Odvoďme funkci $\forall z \leq |x|(\psi)(\underline{x})$

$(\chi)(0, \underline{x}) = 1$
$(\chi)(2z, \underline{x}) = (\chi)(z, \underline{x}) \wedge (\psi)(\underline{x}), |2z|)$
$(\chi)(2z + 1, \underline{x}) = (\chi)(z, \underline{x}) \wedge (\psi)(\underline{x}, |2z + 1|)$

8 Odvození funkcí s použitím rekurze

8.1 Přípravné funkce

Před tím, než si odvodíme polynomální funkce je však třeba si odvodit jednodušší funkce, které nám pomohou k odvození dálších funkcí. První z funkcí, které budeme odvozovat je funkce pro přístup k $n-$tému bitu zprava, s tím že se budeme držet programátorského kodexu a budeme číslovat od nuly.

Theorem 8.1. $x[y]$
Nechť x je číslo, ve kterém chceme znát velikost y-tého bitu. Dohodněme se, že podle programátorské tradice budeme číslovat bity od nultého bitu. Nejdříve odvoďme funkci g, která nám vrátí výsledek pro $y \leq |x|$

$$g(x,y) : \begin{cases} 0 & if \ \ (x \gg y) \leq \mathrm{asl}(x \gg \mathrm{s}(y)) \\ 1 & if \ \ Jinak \end{cases}$$

Nyní odvodíme funkci f, která bude fungovat pro libovolné číslo.

$$f(x,y) : \begin{cases} 0 & if \ \ y > |x| \\ g(x,y) & if \ \ Jinak \end{cases}$$

Omezme polynomem $q = 1$

Další z funkcí, kterou budeme odvozovat, bude konkateční funkce. Tato funkci provede konkatenaci čísla s bitem. Dalo by se tedy říci, že provede konkatenaci libovolného čísla s číslem jedna, nebo nula. Tuto funkci oceníme především při kódování posloupností v další kapitole.

Theorem 8.2. $\mathrm{AddToEnd}(\mathrm{x}_1, \mathrm{x}_2)$
Odvoďme si funkci $\mathrm{AddToEnd}(x_1, x_2)$ *která na konec binárního zápisu čísla x_1 doplní číslici x_2, kde x_2 je číslice 1, nebo 0.*

$$f(x,y) : \begin{cases} \mathrm{asl}(x_1) & if \ \ x_2 \leq 0 \\ \mathrm{s}(\mathrm{asl}(x_1)) & if \ \ Jinak \end{cases}$$

Omezme polynomem $q = |x_1| + 1$

Předposlední z odvozovaných funkcí je funkce pro reverzi bitů. Tato funkce provede reverzi bitů v čísle. Nicméně pro některá čísla by se některé bity ztratily. Právě z tohoto důvodu je tato funkce binární s tím, že jako druhý argument je číslo délky nula, neboli bit. Tato bit je funkcí umístěn na začátek čísla a může tak zamezit ztrátě nul. Je třeba poznamenat, že zvolení nuly, jako druhého argumentu nijak neupraví výsledek.

Theorem 8.3. $\mathrm{ReverseBits}(x_1, x_2)$
Definujeme funkci, která bude sloužit, jako reverze bitů čísla. Máme navíc

možnost přidat x_2 před reverzi, což se nám hodí, pokud hrozí ztráta nulových bitů na konci čísla.

- $\underline{x} = x;2$
- $y = |x|$
- $g(\underline{x}) = x_2$
- $h(\underline{x}, y, f(\underline{x}, \lfloor y/2 \rfloor)) = \text{AddToEnd}(f(\underline{x}, \lfloor y/2 \rfloor), x_1[y])$

Omezme polynomem $q = |x_1 + x_2|$

Poslední z pomocných funkcí je zbytek po dělení dvěma. Tato funkce je shodná s funkcí, jež přečte poslední bit.

Theorem 8.4. $x \,\%\, 2$
Odvodíme funkci pro modulo dvěma

$$x \,\%\, 2 = \begin{cases} 1 & \textit{if } x[0] = 1 \\ 0 & \textit{if } x[0] = 0 \end{cases}$$

Omezme polynomem $q = 1$

8.2 Aritmetické funkce

Dále odvodíme aritmetické funkce, tedy ty polynoimální funkce, jež používáme nejčastěji. Jmenovitě se jedná o funkce plus, mínus, krát, děleno a modulo. Začneme s funcí sčítání, která bude odvozena pomocí délkové rekurze. Rekurze půjde od nejvíce signifikantního bitu (zleva). Rekurze

Theorem 8.5. $x_1 + x_2$
r Odvodvodíme funkci pro sčítání dvou čísel, pomocí délkové rekurze. Funkce f bude na hloubce n vracet součet horních n bitů z obou čísel (jsou počítány i nuly v případě různě louhých čísel).

- $\underline{x} = \{x, y\}$
- $y = y$
- $g(\underline{x}) : 0$
- $h(\underline{x}, y, f(\underline{x}, \lfloor y/2 \rfloor)) : \begin{cases} \mathit{AddToEnd}(\text{s}(\tau(\underline{x}, n)), x[n] \wedge y[n]) & \textit{if } x[n] \wedge y[n] \\ \mathit{AddToEnd}(\tau(\underline{x}, n)), x[n] \wedge y[n]) & \textit{if } \mathit{Jinak} \end{cases}$

Omezme polynomem $q = |x_1| + |x_2|$

Funkce f bude však fungovat pouze v případě, že y je větší z obou čísel. Uvažujme tedy funkci $f_1 : f_1 : \begin{cases} f(x, y) y > x \\ f(y, x) \qquad \text{if Jinak} \end{cases}$

Dále si odvodíme funkce, pro součet dvou čísel. Je třeba poznamenat, že s pomocí polynomiální rekurze a funkce pro sčítání je odvození násobící funkce velice intuitivní a jednuduché.

Theorem 8.6. $x_1 * x_2$
Odvodíme funkci pro násobení dvou čísel.

$x_1 * 2(x_2) = \text{asr}(x_1 * x_2)$
$x_1 * (2x_2 + 1) = \text{asr}(x_1 * x_2) + x_1$
$x_1 * 0 = 0$

Omezme polynomem $q = |x_1 + x_2|$
Další z aritmetických funkcí je funkce pro rodíl dvou čísel. Její odvození nebudu úvádět, jelikož školní algoritmy pro sčítání a odčítání fungují velice podobně.

Theorem 8.7. $x_1 - x_2$
Odvození mínusové funkce je velice podobné odvození funkce pro sčítání. Nicméně se při odvození používá funkce $\dot{-}1$.

Jako předposlední z aritmetických máme funkci pro zbytek při dělení, neboli funkce modulo. Odvoznení funkce je velice prosté, jelikož výnásobení čísla dvěma znamená i vynásobení zbytku dvěma. Stejně zvětšení čísla o jedna znamená zvětšení zbytku o jedna.

Theorem 8.8. $x_1 \,\%\, x_2$
Nejdříve si odvodíme funkci g:
$$g(x_1, x_2) = \begin{cases} x_1 & \text{if } x_1 < x_2 \\ x_1 - x_2 & \text{if } Jinak \end{cases}$$

Nyní odvoďme funkci modulo:

$2x_1 \,\%\, x_2 = g(\text{asl}(x_1 \,\%\, x_2), x_2)$
$(2x_1 + 1) \,\%\, x_2 = g(\text{s}(\text{asl}(x_1) + x_2))$
$0 \,\%\, x_2 = 0$

Omezme polynomem $q = |x_2|$

Poslední z aritmetických funkcí je funkce pro dělení dvou čísel. Odvození pracuje s tím, že pokud vynásobím číslo dvěma, pak je vynésoben dvěma i výsledek dělení a zbytek po dělení.

Theorem 8.9. x_1/x_2
Odvodíme si funkci pro dělení:

$(2x_1 + 1)/x_2 = 2(x/y) + ((2(x \,\%\, y) + 1) \,\%\, y)$
$(2x_1)/x_2 = 2(x/y) + ((2(x \,\%\, y)) \,\%\, y)$
$0/x_2 = 0$

Omezme polynomem $q = |x_1|$

8.3 Zajímavé funkce

V této sekci se podíváme na funkce, kterě jsou zajímavé tím, že obsahují exponenciální funkci. To je zajímavé z toho hlediska, že všechny problémy, které jsou exponenciálně náročné nejsou zpravidla polynomiální. Nicméně, všechny funkce které se nám povede odvodit pracují v polynomiálním čase i přesto, že obsahují ve svém předpisu funkci mocniny. První z takových to funkcí bude x_1 na délku x_2.

Theorem 8.10. $2^{|x_1|}$

- $\underline{x} = \{x_1\}$
- $y = x_1$
- $g(\underline{x}) : 1$
- $h(\underline{x}, y, f(\underline{x}, \lfloor y/2 \rfloor)) : \tau(\underline{x}, n) * 2$

Omezme polynomem $q = |x_1 + 2|$.

Další ze zajímavých funkcí je takzvaná funkce "Smash". Tato funkce dokonce obsahuje násobení uvnitř exponenciální funkce a mohlo by se zdát, že se jedná o funkci exponenciální. Jedná se však o funkci polynomiální.

Theorem 8.11. $Smash(2^{|x_1|*|x_2|})$

- $\underline{x} = \{x_1, x_2\}$
- $y = x_2$
- $g(\underline{x}) : 1$
- $h(\underline{x}, y, f(\underline{x}, \lfloor y/2 \rfloor)) : \tau(\underline{x}, n) * 2^{|x_1|}$

Omezme polynomem $q = (|x_1| + 2) * (|x_2| + 2)$.

Další z odvozovaných funkcí je několikrát zmíněná exponenciální funkce. O této funkci je jasné, že nepatří do množiny polynomiálních funkcí a důkaz musí tedy někde selhat. Odvození ukazuje kde přesně důkaz selže.

Excersise 8.12. $x_1^{x_2}$
Pokusíme se odvodit mocninou funkci:

- $\underline{x} = \{x_1, x_2\}$
- $y = x_2$
- $g(\underline{x}) : 1$
- $h(\underline{x}, y, f(\underline{x}, \lfloor y/2 \rfloor)) = \begin{cases} (\tau(\underline{x}, n) * \tau(\underline{x}, n)) & \text{if } x_2[n-1] = 0 \\ (\tau(\underline{x}, n) * \tau(\underline{x}, n) * x_1) & \text{if Jinak} \end{cases}$

Pokud se podíváme na funkci, tak je jasné, že počet růstu cifer je exponenciální a nepůjde tedy omezit polynomem.

Poslední ze zajímavých funkcí je exponenciální funkce v prostoru modulo x_3. Fuknce se opět zdá exponenciální, nicméně nikdy nelze dojít k mezivýsledku vetšímu než x_3. A proto půjdě tato funkce pomocí x_3 omezit.

Theorem 8.13. $x_1^{x_2} mod\ x_3$

- $\underline{x} = \{x_1, x_2, x_3\}$
- $y = x_2$
- $g(\underline{x}) : 1$
- $h(\underline{x}, y, f(\underline{x}, \lfloor y/2 \rfloor)) : \begin{cases} (\tau(\underline{x}, n) * \tau(\underline{x}, n)) \,\%\, x_3 & if\ \ x_2[n-1] = 0 \\ (\tau(\underline{x}, n) * \tau(\underline{x}, n) * x_1) \,\%\, x_3 & if\ \ Jinak \end{cases}$

Stačí nám omezit polynomem $q = |x_3|$

9 Kódování posloupností

V této kapitole neformálně představíme jeden z možných způsobů kódování posloupností. nejdříve definujeme gödelovo číslo pro posloupnost. Pro kódování posloupností je potřeba definovat tří základní funkce. Těmi jsou funkce pro odmázíní posledního členu posloupnosti, funkce pro přidání nového členu posloupnosti a funkce pro výběr členu z posloupnosti. Všechny tři funkce jsou velice jednoduše odvoditelné pomocí délkové rekurze a jsou přenéchány čtenáři. N8sledující definice Gödelova čísla je přejatá ze zdroje [3].

Definition 9.1. Gödelovo číslo pro posloupnost
Nechť $M = \{x_0, \ldots, x_n\}$ je množina proměnných, na které můžeme definovat uspořádání pomocí indexů čísel. Gödelovo číslo pro množinu M, na které jsme si definovali upořádání pomocí indexů, dostaneme takto. Vezmeme konkatenaci binarních reprezentací čísel x_0 až x_n s tím, že jednotlivá číla oddělíme čárkami (čísla konkatenujeme pomocí našeho uspořádání). Výsledkem bude řetězec nul, čárek a jedniček. Vezměme reverzi tohoto řetězce a proveďme simultální substituci 10 za 0, 11 za 1 a 00 . Výsledek můžeme přečíst, jako binární číslo. Toto číslo prohlašme za Gödelovo číslo pro posloupnost $< x_0, \ldots, x_n >$. Číslo 0 považujme za prázdnou posloupnost (značme $<>$).

10 Závěrem

Práce se zabývala bezestrojovou definicí polynomiálně počitatelných funkcí a byla silně inspirována pracemi [1] a [3]. Ve dvou úvodních kapitolách jsme nastínili, jakým směrem se se bude práce ubírat a definovali jsme základní definice strojové definice ohledně polynomiálnosti. Dále jsme definovali základní funkci a schéma pro jejich skládání a dvě rekurzivní schémata. Čtenář byl se základními funkcemi obeznámen díky odvození několika funkcí a důkazu ekvivalence mezi rekurzivnímy schématy. Poté byly definovány pojmy polynomiálně počitatelné funkce, polynomiální podmínky a polynomiální množiny. U polynomiálních podmínek byla dokázána uzavřenost na výrokové logické operace a omezenou kvantifikaci. Poslední dvě kapitoly této práce byly vedeny v neformálním duchu. Na závěr práce byly nastíněny možnosti pro definici kódování.

11 Zdroje

[1] Samuel R. Buss. Naples, Italy, 1986. University of California, Berkeley. [2] A. Cobham. The Intrinsic Computational Difficulty of Functions, in: Proc. Logic, Methodology, and Philosophy of Science II, Y. Bar-Hillel ed., pp. 24- 30. North Holland, 1965. [3] P. Mach. Bezestrojová charakterizace funkcí počitatelných v polynomiálním čase. Předběžná verze nefinalizované ročníkové práce. Katedra logiky FF UK, 1998. [4] SVEJDAR, Vítězslav. Logika: neúplnost, složitost a nutnost. Praha: Academia, 2002. ISBN 80-200-1005-X. [5] S. Bellantoni, S. Cook. A New Recursion-Theoretic Characterization of the Polytime Functions. Computational Complexity 2:297-110, 1992. [6] ČERNÝ, Michal. Výpočty. Praha: Professional Publishing, 2012. ISBN 978-80-7431-068-3.