Základy složitosti a vyčíslitelnosti

NTIN090

Petr Kučera

2024/25 (7. přednáška)

Dnešní plán

- Savičova věta
- Prostorová hierarchie
- Časová hierarchie

Savičova věta

Savičova věta

Věta (Savičova věta)

Pro každou funkci $f(n) \geq \log_2 n$ *platí*

$$\mathrm{NSPACE}(f(n)) \subseteq \mathrm{SPACE}(f^2(n))$$

Důsledek

$$\mathrm{PSPACE} = \mathrm{NPSPACE}$$

Savičova věta (začátek důkazu)

- Předpokládejme, že $L \in \mathrm{NSPACE}(f(n))$
- Existuje NTS M, který přijímá L v prostoru $O(f(n))$
- Popíšeme deterministický TS M', který rozhoduje L v prostoru $O(f^2(n))$

Zjednodušující předpoklad

M pracuje v prostoru $f(n)$

- Pokud M pracuje v prostoru $g(n) = O(f(n))$, pak

$$O(g^2(n)) = O(f^2(n))$$

- Tedy $\mathrm{SPACE}(g^2(n)) \subseteq \mathrm{SPACE}(f^2(n))$

Technické předpoklady

- M nepohne hlavou na pracovní pásce nalevo od počáteční pozice
- M má jednoznačnou přijímající konfiguraci C_F
  - Jediný přijímající stav q_1
  - Hlava na vstupní pásce je nad nejlevějším symbolem vstupu
  - Hlava na pracovní pásce je nad nejlevější buňkou omezeného prostoru
  - Pracovní páska je prázdná
- C_0^x označuje počáteční konfiguraci výpočtů M se vstupem x
- $n = |x|$ označuje délku vstupu x

Prohledáváme graf

Idea

Hledej cestu z C_0^x do C_F v grafu konfigurací $G_{M,x} = (V, E)$.

První návrh řešení: Použij DFS nebo BFS

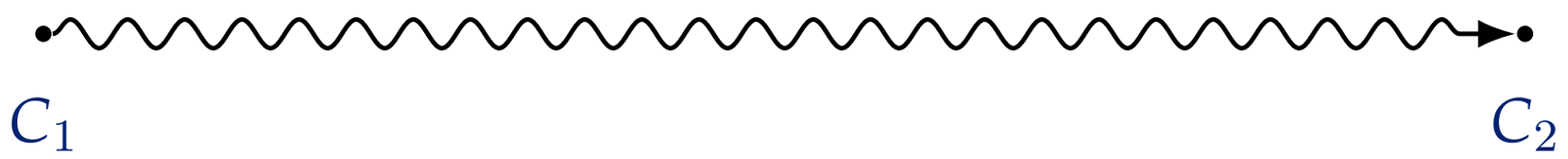
- Oba algoritmy vyžadují paměť, která je přinejmenším lineární ve velikosti grafu
- $G_{M,x}$ může obsahovat až $2^{c_M f(n)}$ vrcholů pro nějakou konstantu c_M, která závisí na M
- DFS i BFS vyžadují prostor exponenciální v $f(n)$

Nemůžeme použít DFS ani BFS

→ samotná jedna větev může být $O(2^{c_M f(n)})$, to nepočítám

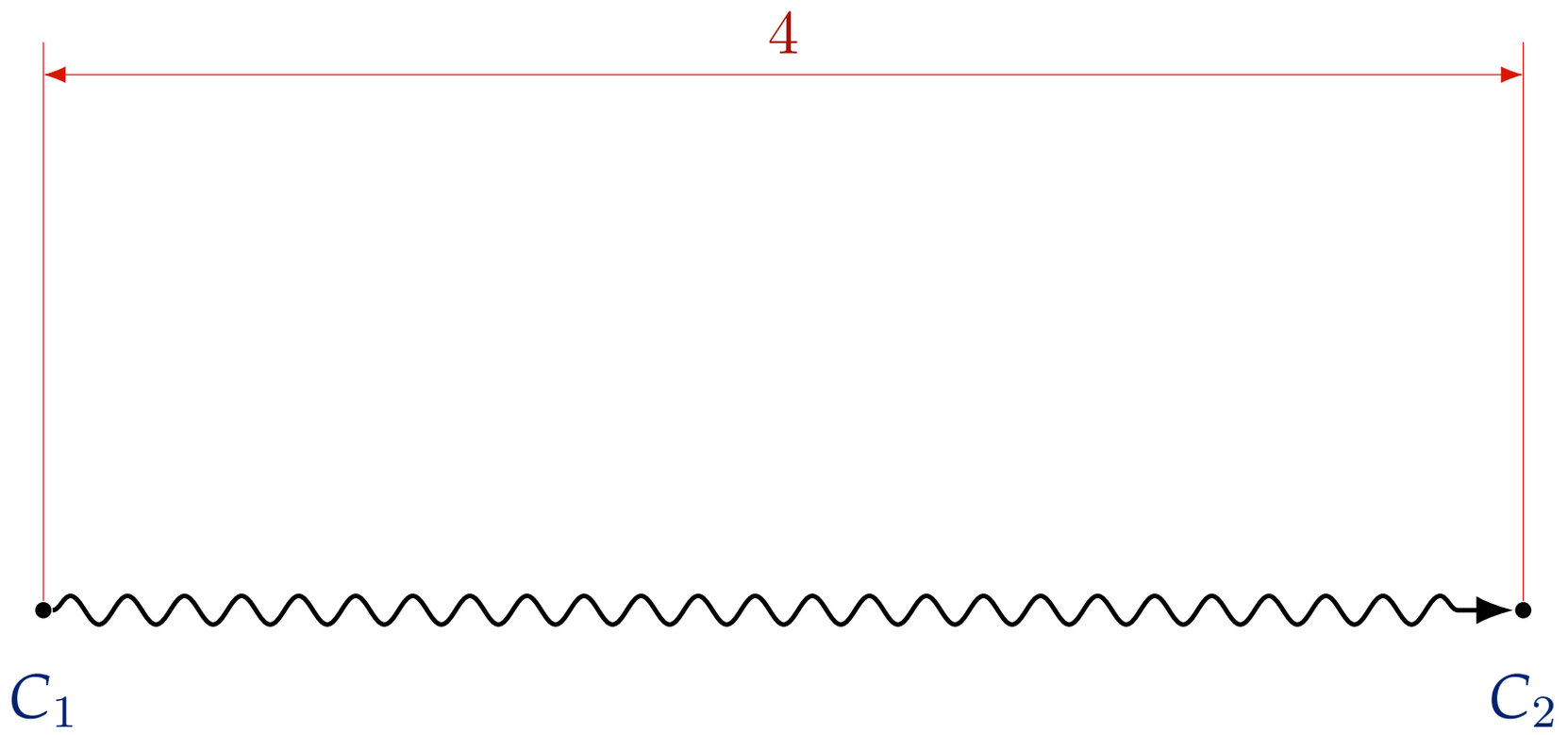
Rozděl a panuj

Existuje cesta z C_1 do C_2?



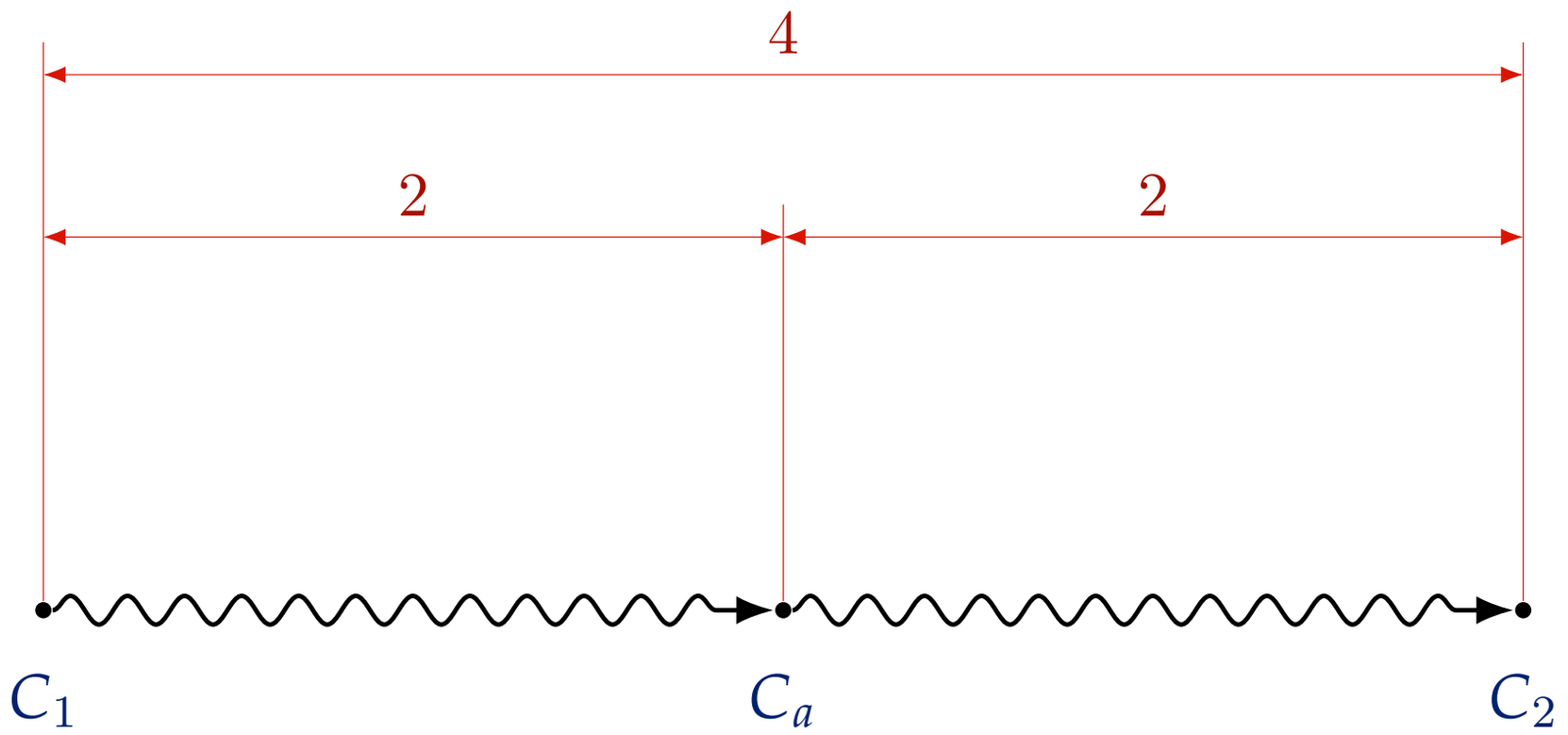
Rozděl a panuj

Existuje cesta z C_1 do C_2 délky nejvýš t?



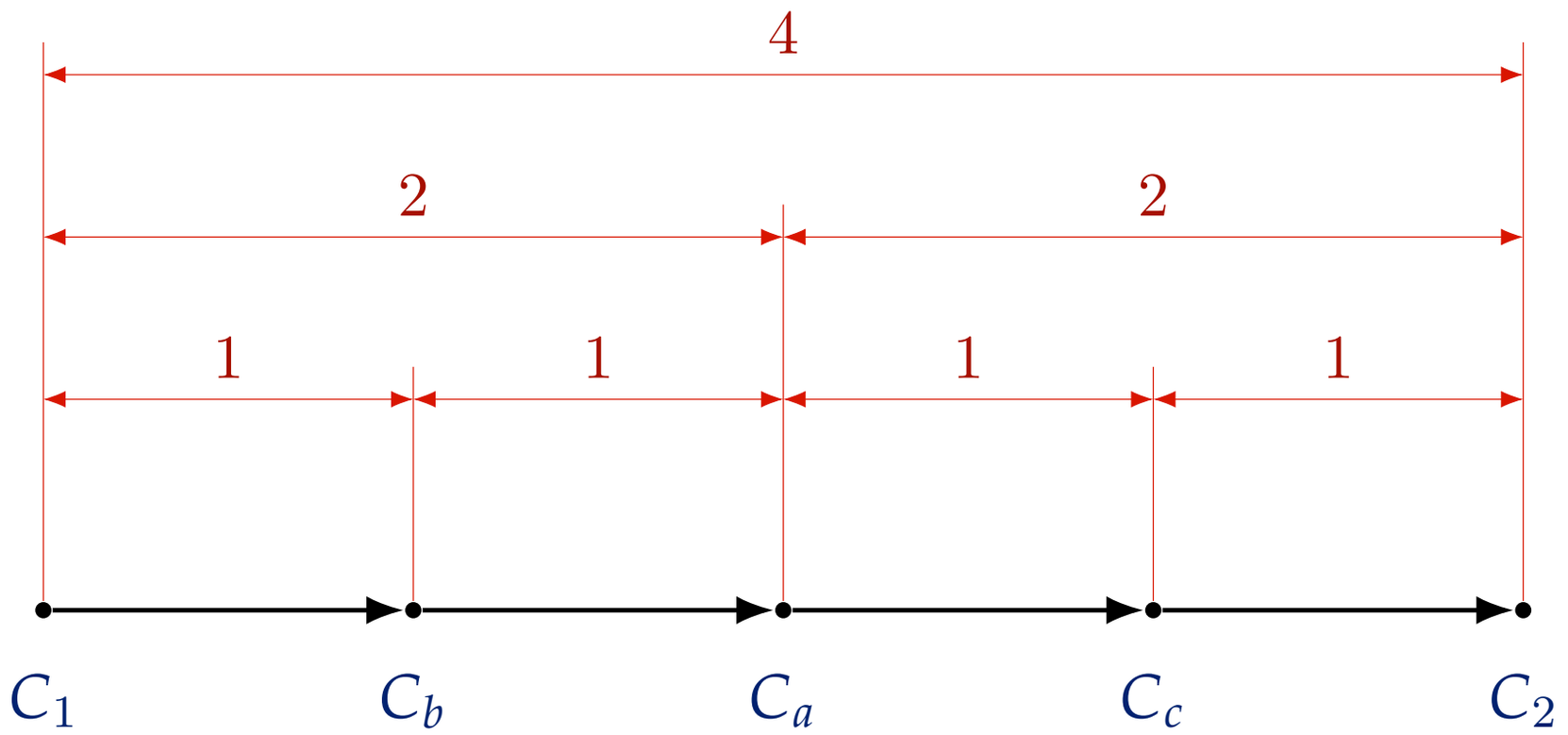
Rozděl a panuj

Existuje, pokud lze najít vrchol uprostřed



Rozděl a panuj

Půlení pokračuje dokud nedosáhneme délky nejvýš 1.



Hrany grafu, snadno ověřitelné

Prohledáváme graf s malou pamětí

- Předpokládejme (pro tuto chvíli), že M' může spočítat $f(n)$ pro vstup x
- Délka cesty z C_0^x do C_F je nejvýš $2^{c_M f(n)}$

Rozděl a panuj

Cesta z C_1 do C_2 délky nejvýš 2^k existuje, právě když

1. $k = 0$ a $C_1 = C_2$ nebo $(C_1, C_2) \in E$, nebo
2. $k > 0$ a existuje prostřední vrchol C_m, pro který platí, že
   - existuje cesta z C_1 do C_m délky nejvýš 2^{k-1} a
   - existuje cesta z C_m do C_2 délky nejvýš 2^{k-1}

- Hloubka rekurze je $O(f(n))$
- Na každé úrovni rekurze je potřeba prostor $O(f(n))$ pro reprezentaci konfigurací C_1, C_2 a C_m —> pouze to si musím v jedné vrstvě počítat
- Celkový prostor $O(f^2(n))$

Dosažitelnost

Funkce `Reachable`(C_1, C_2, k)

Vstup: Konfigurace C_1 a C_2, přirozené číslo k

Výstup: **true** pokud $G_{M,x}$ obsahuje cestu z C_1 do C_2 délky nejvýš 2^k, jinak **false**

if $k = 0$ **then**

  if $C_1 = C_2$ **or** $(C_1, C_2) \in E$ **then**

    return true

  else

    return false

foreach konfiguraci C_m využívající prostor $f(n)$ **do**

  if `Reachable`$(C_1, C_m, k-1)$ **and** `Reachable`$(C_m, C_2, k-1)$ **then**

    return true

return false

prostě hledání zkusím všechny vhodné konfigurace

Volání `Reachable()`

- Pokud M' zná hodnotu $f(n)$, stačí mu zavolat

  C_M známe ze vlastní TS M

  `Reachable`$(C_0^x, C_F, c_M f(n))$

  → v tomhle prostoru pracuje TS M

- Jedna instance `Reachable()` používá prostor velikosti $O(f(n))$
  - konfigurace C_1, C_2, C_m používají prostor $f(n)$
  - $O(f(n))$ bitů stačí k reprezentaci těchto konfigurací
  - $O(f(n))$ bitů stačí k reprezentaci hodnoty k
- Hloubka rekurze je $O(f(n))$
- $O(f(n))$ instancí `Reachable()` je v každém okamžiku na zásobníku

Celkem je stačí prostor velikosti $O(f^2(n))$

Jsme tedy hotovi?

Jsme hotovi?

Co když M' nemůže spočítat hodnotu $f(n)$ pro vstup x?

- Funkce $f(n)$ nemusí být nutně algoritmicky vyčíslitelná
- $f(n)$ může být vyčíslitelná, ale k jejímu vyčíslení může být potřeba prostor $\omega(f^2(n))$
- I kdyby $f(n)$ byla vyčíslitelná v prostoru $O(f^2(n))$, funkce $f(n)$ může být neznámá, známe-li pouze M
- c_M je konstanta závisející na M
  - její hodnotu můžeme určit ze znalosti M

tohle by nám stačilo, protože stejný prostor vyžaduje náš algoritmus výše.

Je-li $f(n)$ neznámá

Neoptimalizujeme čas... proto si můžeme dovolit tuhle úsmatku

Idea

1. Zkoušej hodnoty $f(n) = 1, 2, 3, \ldots$
2. Zastav s hodnotou $f(n) = i$, pro niž
   - je nalezena cesta z C_0^x do C_F nebo
   - z C_0^x není dosažitelná žádná konfigurace využívající prostor velikosti $i + 1$

Někdy ale musím zastavit! Speciálně když ta cesta neexistuje. (vstup nemá být přijat)

To zjistím tak, že z C_0^x se nedostanu do žádné nové konfigurace

= nebo prostě žádná konfigurace na pásce $+1$

Výpočet M'

Výpočet M' se vstupem x

1 $i \leftarrow 1$
   // Volání Reachable() předpokládají, že $f(n) = i$
2 **if** Reachable(C_0^x, C_F, $c_M \cdot i$) **then přijmi**
3 **foreach** konfiguraci C využívající prostor $i + 1$ **do**
4 **if** Reachable(C_0^x, C, $c_M \cdot i$) **then**
5 $i \leftarrow i + 1$
6 **goto** 2
7 **odmítni**

M' rozhoduje L v prostoru $O(f^2(n))$.

Více zdrojů, více síly

Když přidám více času/prostoru, budu schopný vyřešit více problémů?

Prostor

Deterministická Prostorová Hierarchie

Připomenutí

Nechť $f : \mathbb{N} \to \mathbb{N}$ je funkce, která je definovaná pro každý vstup

- Deterministický Turingův stroj M pracuje v prostoru $f(n)$, pokud výpočet M nad libovolným vstupem x délky $|x| = n$ skončí a využije nejvýš $f(n)$ buněk pracovní pásky.
- $\mathrm{SPACE}(f(n))$ je třída jazyků přijímaných Turingovými stroji, které pracují v prostoru $O(f(n))$

Věta (o deterministické prostorové hierarchii)

Pro každou prostorově konstruovatelnou funkci $f : \mathbb{N} \to \mathbb{N}$ existuje jazyk A, který je rozhodnutelný v prostoru $O(f(n))$, nikoli však v prostoru $o(f(n))$.

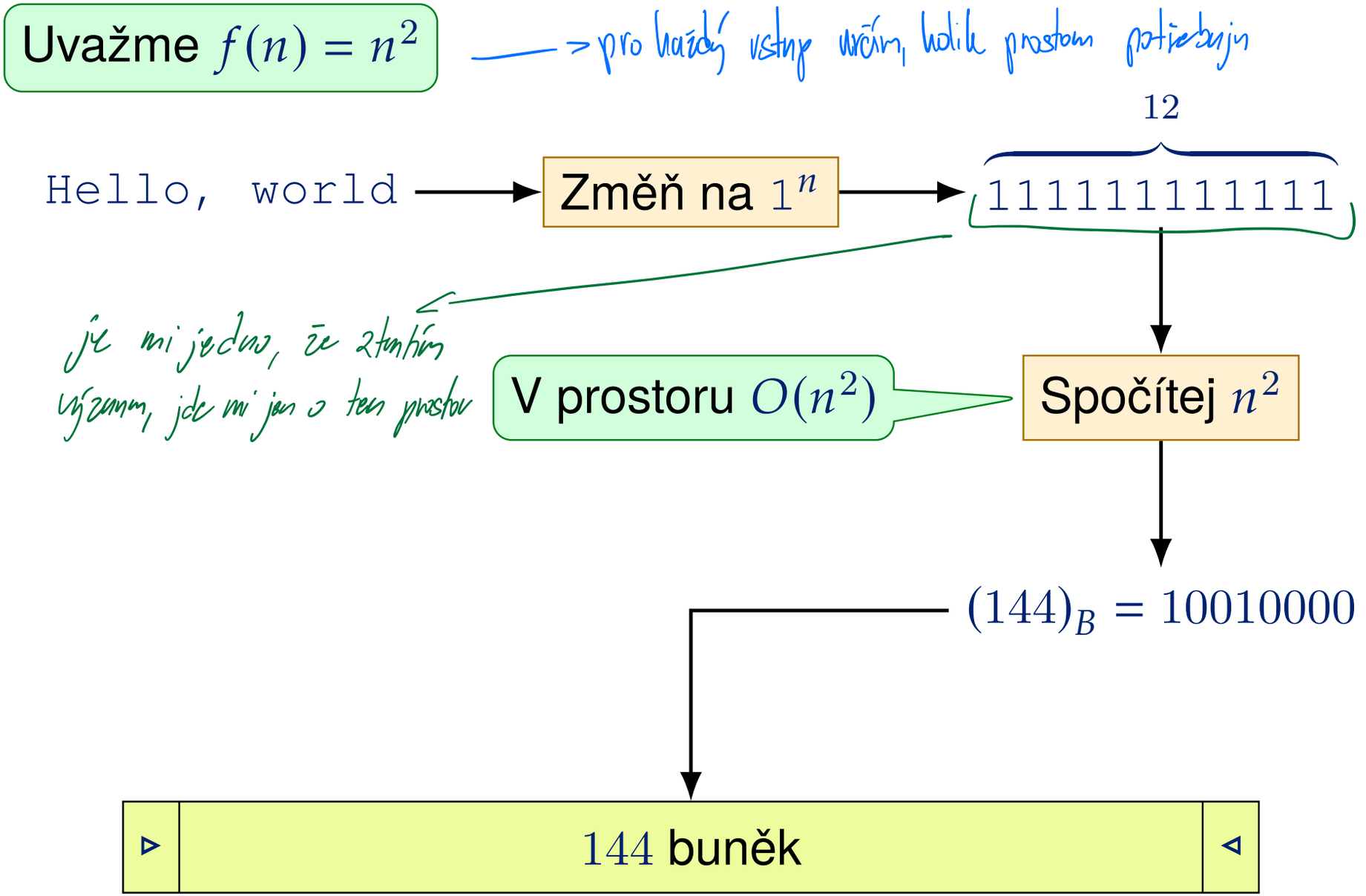
Prostorová konstruovatelnost

Definice

Funkci $f : \mathbb{N} \to \mathbb{N}$, kde $f(n) \geq \log_2 n$, nazveme prostorově konstruovatelnou, je-li funkce, která zobrazuje 1^n na binární reprezentaci $f(n)$ vyčíslitelná v prostoru $O(f(n))$.

- Funkce obvykle používané pro měření prostorové složitosti jsou prostorově konstruovatelné, například
  - $\lceil \log_2 n \rceil$
  - $\lceil \sqrt{n} \rceil$
  - polynomy
  - $\lceil n \log_2 n \rceil$

Efektivní alokace paměti



Efektivní alokace paměti

Předpokládejme, že $f(n)$ je prostorově konstruovatelná strojem M_f

1. Se vstupem x délky $n = |x|$
2. Sestav řetězec $w = \mathtt{1}^n$
   - Každý znak x změň na $\mathtt{1}$
3. Vypočítej $k = f(n)$
   - Spusť $M_f(w)$
4. Vyznač k buněk na pásce

Využívá prostor $O(f(n))$, ne více, než potřebujeme alokovat.

Idea důkazu

Věta (o deterministické prostorové hierarchii)

Pro každou prostorově konstruovatelnou funkci $f : \mathbb{N} \to \mathbb{N}$ existuje jazyk A, který je rozhodnutelný v prostoru $O(f(n))$, nikoli však v prostoru $o(f(n))$.

Idea důkazu:

- A definujeme popsáním stroje D, který rozhoduje A
- D pracuje v prostoru $O(f(n))$
- Pro každý stroj M, který pracuje v prostoru $o(f(n))$ platí, že $L(M) \neq L(D)$
- Použijeme diagonalizaci

První nápad

První nápad konstrukce D

1. Se vstupem $x = \langle M \rangle$
2. Simuluj $M(\langle M \rangle)$ v prostoru $f(n)$
   - Pokud simulace potřebuje více prostoru, odmítni
3. Pokud M odmítl, přijmi, jinak odmítni

pak si to vystačilo se svým prostorem a dolnálo rozhodnout.

- Nechť M pracuje v prostoru $g(n) = o(f(n))$, pak

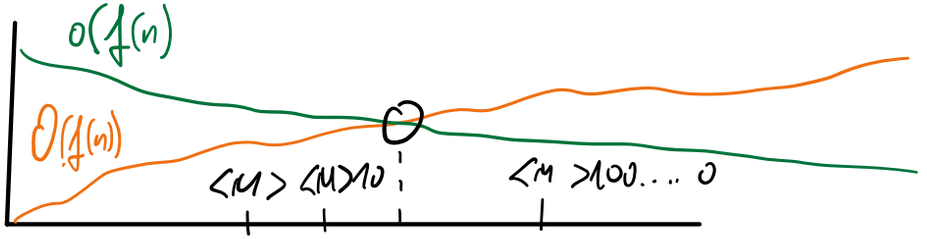
$$(\forall c \in \mathbb{R}^+)(\exists n_0 \in \mathbb{N})(\forall n \geq n_0)[cg(n) \leq f(n)]$$

prostor, který je menší než $f(n)$

- Prostor $f(n)$ postačuje k simulaci M na dost dlouhých vstupech x
- Nezáleží na chování se stroji M, které nepracují v prostoru $o(f(n))$

Závorek ale může vyhlit v malém prostoru, to taky nechceme.

Nevíme, jestli jsme dostali M zleva nebo zprava od průsečíku



Problém s asymptotikou

Není-li řetězec $\langle M \rangle$ dost dlouhý, prostor $f(n)$ nemusí stačit k simulaci $M(\langle M \rangle)$.

Řešení

- Uvážíme řetězce tvaru $\langle M \rangle \mathtt{10}^*$
- $x = \langle M \rangle \mathtt{10}^{n_0}$ je dost dlouhý pro nějaké n_0
- Prostor $f(n)$ stačí k simulaci $M(x)$
- $D(x)$ přijme, právě když $M(x)$ odmítne
- Tedy $L(M) \neq L(D)$

$\langle M \rangle \mathtt{1}$
$\langle M \rangle \mathtt{10}$
$\langle M \rangle \mathtt{100}$
$\langle M \rangle \mathtt{1000}$
$\langle M \rangle \mathtt{10000}$
$\langle M \rangle \mathtt{100000}$
$\vdots$

Problém se zastavením

I je-li prostor $f(n)$ dostatečný pro simulaci $M(\langle M \rangle)$, výpočet se může zacyklit.

Řešení

Zastav, pokud simulace vyžaduje více než $2^{f(n)}$ kroků.

→ v tomhle případě jsem už prošel všechny možnosti a jsem někde znovu

- Nechť M pracuje v prostoru $g(n) = o(f(n))$
- Uvažme vstup x délky $n = |x|$
- Je-li $M(x)\downarrow$, pak výpočet skončí do $2^{c_M g(n)}$ kroků pro nějakou konstantu c_M
- Je-li n dost velké, simulace $M(x)$ skončí do $2^{f(n)}$ kroků

Stroj D

Výpočet D se vstupem x

1. $n \leftarrow |x|$
2. Vypočti $f(n)$ pomocí prostorové konstruovatelnosti
3. Označ $f(n)$ buněk na pracovní pásce
4. **if** v následujících krocích hlava opustí vyznačený prostor **then**
5. $\quad$ **odmítni**
6. **if** x nemá tvar $\langle M \rangle 10^*$ **then odmítni**
7. Simuluj $M(x)$ s počítáním kroků simulace
8. **if** počet simulovaných kroků překročí $2^{f(n)}$ **then odmítni**
9. **if** M přijal **then odmítni else přijmi**

Definujeme $A = L(D)$

> pouze když zastavil a odmítl, tak prošel všechny možnosti (vyčerpal svůj veškerý prostor)

Prostor použitý strojem D

- Výpočet $f(n)$ vystačí s prostorem $O(f(n))$ díky prostorové konstruovatelnosti
- Poté hlava D zůstane v rámci $f(n)$ vyznačených buněk
- D tedy pracuje v prostoru $O(f(n))$
- Čítač kroků lze reprezentovat pomocí $f(n)$ bitů
  - Může být na další stopě

A je rozhodnutelný v prostoru $O(f(n))$.

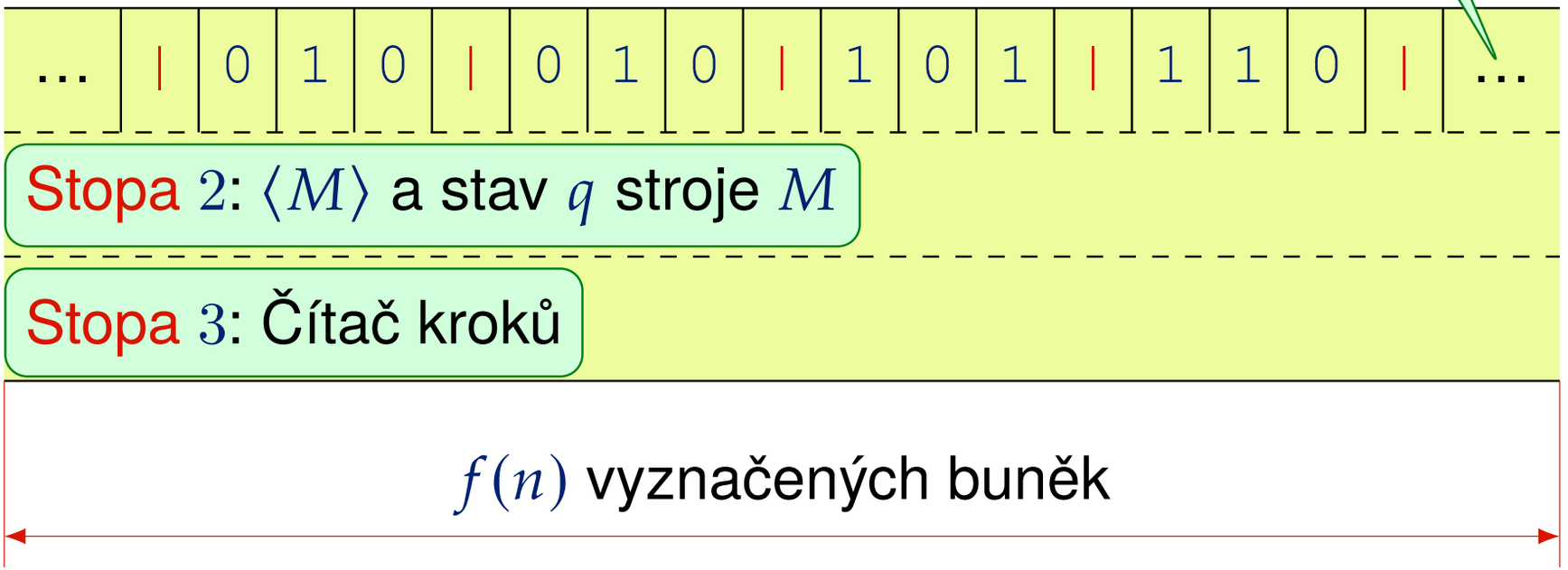
D simuluje M

Vstupní páska D (i M)

$\langle M \rangle 1000000000000$

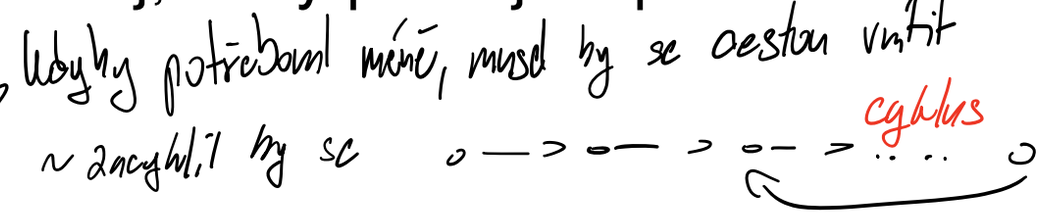
Stopa 1: páska M
každý znak zakódován
$b = \lceil \log_2 |\Sigma| \rceil$ bity

Pracovní páska D má tři stopy



Menší prostor nestačí

- Nechť $M = (Q, \Sigma, \delta, q_0, F)$ je stroj, který pracuje v prostoru $g(n) = o(f(n))$
- Ukážeme, že $A \neq L(M)$



Pro nějakou konstantu c_M platí, že se vstupem x délky $n = |x|$

- $M(x)$ lze simulovat v prostoru $c_M g(n)$
- $M(x)$ skončí výpočet do $2^{c_M g(n)}$ kroků

- Připomeňme univerzální TS
- Existuje konstanta c_M taková, že M se vstupem x má nejvýš $2^{c_M g(n)}$ různých konfigurací
- Prostor $c_M g(n)$ stačí k reprezentaci jedné konfigurace
  - $\lceil \log_2 |\Sigma| \rceil$ bitů pro každé políčko pracovní pásky M
  - $\lceil \log_2 |Q| \rceil$ bitů pro reprezentaci stavu M

Menší prostor nestačí

$$g(n) = o(f(n)) \implies (\exists n_0 \in \mathbb{N})(\forall n \geq n_0)[c_M g(n) \leq f(n)]$$

Existují konstanty c_M a n_0 takové, že se vstupem x délky $n \geq n_0$

- $M(x)$ lze simulovat v prostoru $c_M g(n) \leq f(n)$
- $M(x)$ skončí výpočet do $2^{c_M g(n)} \leq 2^{f(n)}$ kroků

- Simulace M se vstupem $x = \langle M \rangle 10^{n_0}$ skončí a
- $D(x)$ přijme právě když $M(x)$ odmítne

$$L(D) \neq L(M)$$

Prostorová hierarchie

Věta (o deterministické prostorové hierarchii)

Pro každou prostorově konstruovatelnou funkci $f : \mathbb{N} \to \mathbb{N}$ *existuje jazyk* A*, který je rozhodnutelný v prostoru* $O(f(n))$*, nikoli však v prostoru* $o(f(n))$*.*

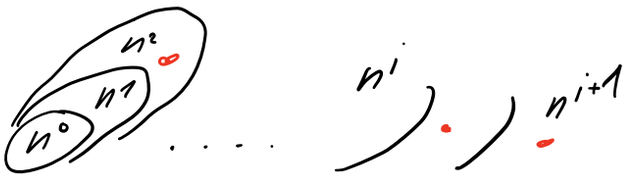
Důsledek

Jsou-li $f_1, f_2 : \mathbb{N} \to \mathbb{N}$ *funkce, pro které platí, že* $f_1(n) \in o(f_2(n))$ *a* f_2 *je prostorově konstruovatelná, potom*

$$\mathrm{SPACE}(f_1(n)) \subsetneq \mathrm{SPACE}(f_2(n))$$

Tohle říká: $PSPACE = \bigcup_{i \in \mathbb{N}} SPACE(n^i)$

– nikdy nebude skončit konečné sjednocení



Polynomy a související funkce

Důsledek

Pro každá dvě reálná čísla $0 \leq \epsilon_1 < \epsilon_2$ *platí, že*

$$\mathrm{SPACE}(n^{\epsilon_1}) \subsetneq \mathrm{SPACE}(n^{\epsilon_2})$$

- Je-li ϵ_2 racionální číslo, pak
  - n^{ϵ_2} je prostorově konstruovatelná
    - Lze jednoduše ukázat pro přirozená čísla
    - Lze ukázat i pro racionální čísla
  - Ostrá inkluze plyne z prostorové hierarchie
- Je-li ϵ_2 iracionální číslo
  - Racionální čísla jsou hustá v reálných číslech
  - Existuje racionální číslo ϵ splňující $\epsilon_1 < \epsilon < \epsilon_2$
  - Z prostorové hierarchie a prostorové konstruovatelnosti n^{ϵ}

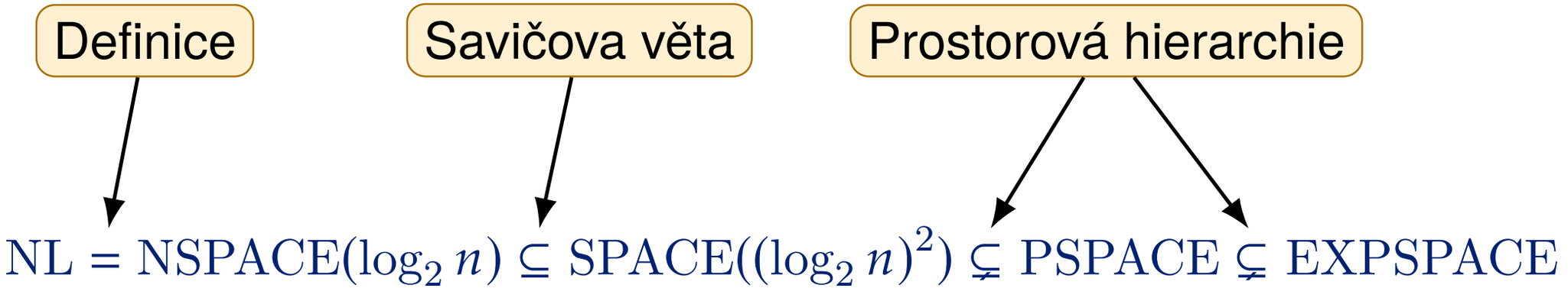
i mezi reálná čísla vždy můžeš najít číslo „mezi"

$$\mathrm{SPACE}(n^{\epsilon_1}) \subsetneq \mathrm{SPACE}(n^{\epsilon}) \subseteq \mathrm{SPACE}(n^{\epsilon_2})$$

Logaritmický, polynomiální a exponenciální prostor

Důsledek

$\mathrm{NL} \subsetneq \mathrm{PSPACE} \subsetneq \mathrm{EXPSPACE} = \bigcup_{k \in \mathbb{N}} \mathrm{SPACE}(2^{n^k})$.



Čas

Časová složitost Turingova stroje

Připomenutí

Nechť $f : \mathbb{N} \to \mathbb{N}$ je funkce, která je definovaná pro každý vstup

- Deterministický Turingův stroj M pracuje v čase $f(n)$, pokud výpočet M nad libovolným vstupem x délky $|x| = n$ skončí po provedení nejvýše $f(n)$ kroků.
- $\mathrm{TIME}(f(n))$ je třída jazyků přijímaných Turingovými stroji, které pracují v čase $O(f(n))$

Věta (O deterministické časové hierarchii)

Pro každou časově konstruovatelnou funkci $f : \mathbb{N} \to \mathbb{N}$ existuje jazyk A, který je rozhodnutelný v čase $O(f(n))$, nikoli však v čase $o(f(n)/\log_2 f(n))$.

Myšlenkově je to stejné jako s prostorem

Časová konstruovatelnost

Definice

Funkci $f : \mathbb{N} \to \mathbb{N}$, kde $f(n) \in \Omega(n \log_2 n)$, nazveme časově konstruovatelnou, je-li funkce, která zobrazuje 1^n na binární reprezentaci $f(n)$ vyčíslitelná v čase $O(f(n))$.

- Funkce obvykle používané pro měření časové složitosti jsou časově konstruovatelné, například
  - $\lceil n \log_2 n \rceil$
  - $\lceil n\sqrt{n} \rceil$
  - polynomy
  - 2^n

Efektivní počítání kroků

Předpokládejme, že $f(n)$ je časově konstruovatelná strojem M_f

1. Se vstupem x
2. Sestav řetězec $w = 1^n$
   - Každý znak x změň na 1
3. Vypočítej $k = f(n)$
   - Spusť $M_f(w)$
4. Inicializuj binární čítač hodnotou k
   - Používá $\lceil \log_2 k \rceil$ bitů
5. Sniž hodnotu čítače o jedna po každém kroku a skonči pokud dosáhne hodnota čítače nuly

Pracuje v čase $O\left(f(n) \cdot t(\lceil \log_2 k \rceil)\right)$, kde $t(\lceil \log_2 k \rceil)$ je čas potřebný k aktualizaci hodnoty čítače s $\lceil \log_2 k \rceil$ bity.

Idea důkazu

Věta (O deterministické časové hierarchii)

Pro každou časově konstruovatelnou funkci $f : \mathbb{N} \to \mathbb{N}$ existuje jazyk A, který je rozhodnutelný v čase $O(f(n))$, nikoli však v čase $o(f(n)/\log_2 f(n))$.

Idea důkazu

- Podobný postup jako v případě prostoru
- Je potřeba simulovat $M(x)$ s počítáním kroků
- Manipulace s čítačem kroků přidává faktor $\Theta(\log_2 f(n))$

Předpokládáme, že všechny stroje mají jednu pásku.

Nemusíme řešit prostor, protože bude vždy $\leq$ času

Stroj D

Výpočet D se vstupem x

1. $n \leftarrow |x|$
2. Vypočti $f(n)$ pomocí časové konstruovatelnosti
3. Inicializuj binární čítač hodnotou $\lceil f(n)/\log_2 f(n) \rceil$
4. Sniž hodnotu čítače o 1 po každém kroku při provádění kroků 6-8
5. **if** čítač dosáhne nulové hodnoty **then** odmítni
6. **if** x není tvaru $\langle M \rangle 10^*$ **then** odmítni
7. Simuluj $M(x)$
8. **if** M přijal **then** odmítni **else** přijmi

Z = vytekl jsem z prostoru

Definujeme $A = L(D)$

Čas výpočtu D

- $f(n)$ lze vypočítat v čase $O(f(n))$ díky časové konstruovatelnosti
- $\lceil f(n)/\log_2 f(n) \rceil$ (binárně) lze vypočítat v čase $O(f(n))$
- Ověření, zda x má tvar $\langle M \rangle 10^*$ lze provést v čase $O(n) = O(f(n))$

Implementační detaily

1. Jak provést simulaci $M(x)$ tak, aby jedna instrukce M byla provedena v c_M krocích simulace
   - kde c_M je konstanta závisející na M
2. Jak snížit hodnotu čítače o 1 po každém kroku D v čase $O(\log_2 f(n))$ → amortizovaně je to $O(1)$, TS bude ale hlocht ten čítač...

Simulace s konstantním zpožděním

- Předpokládejme $M = (Q, \Sigma, \delta, q_0, F)$
- Předpokládejme vstup x tvaru $\langle M \rangle 10^*$
- $\langle M \rangle$ kóduje přechodovou funkci δ
- $|\langle M \rangle|$ je konstantní, je-li M zafixovaný
- Při simulaci jednoho kroku $M(x)$
  - D hledá v $\langle M \rangle$ přechod pro aktuální displej
  - Nejprve musí hlava D najít začátek $\langle M \rangle$ na pásce
  - D potřebuje také rychlý přístup k aktuálnímu stavu M

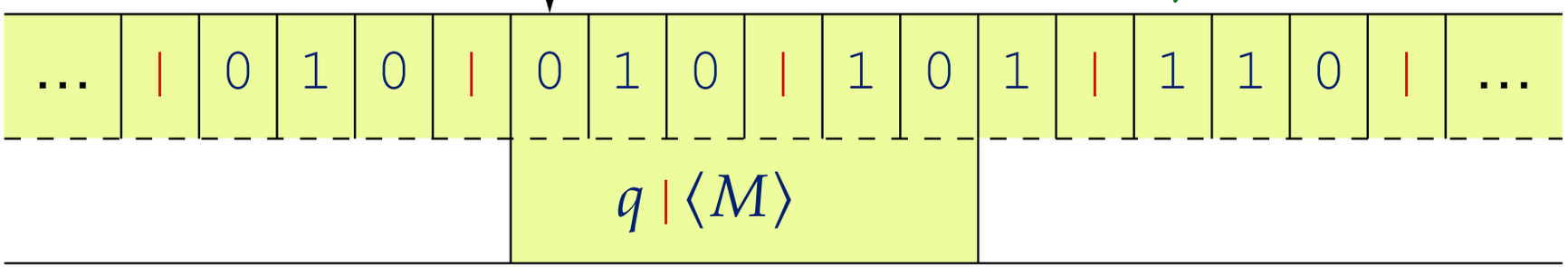
$\langle M \rangle$ a stav stroje M musí být v každém okamžiku poblíž hlavy D.

Páska D

Páska stroje D má dvě stopy

Hlava D nad blokem pod hlavou M

Stopa 1: páska M každý znak zakódován $b = \lceil \log_2 |\Sigma| \rceil$ bity



Stopa 2: aktuální stav q a přechodová funkce $\langle M \rangle$

Stopa 2 je vždy zarovnaná s blokem pod hlavou M

Simulace s konstantním zpožděním

- Páska D má dvě stopy:

  Stopa 1 páska M, každý znak je zakódován $b = \lceil \log_2 |\Sigma| \rceil$ bity
  Stopa 2 aktuální stav q stroje M a přechodová funkce $\langle M \rangle$

  - Vždy zarovnaná s hlavou M
  - Po odsimulování jednoho kroku M může být potřeba obsah stopy 2 posunout

- Simulace jednoho kroku M pak zabere čas c_M pro nějakou konstantu c_M, která závisí na M
  - Nalezení instrukce v čase $O(|\langle M \rangle|^2)$
  - Posunutí stopy 2 v čase $O(|\langle M \rangle|^2)$
  - Konstantní čas pro fixní TS M

Pokud M pracuje v čase $g(n)$, jeho simulaci lze provést $O(g(n))$.

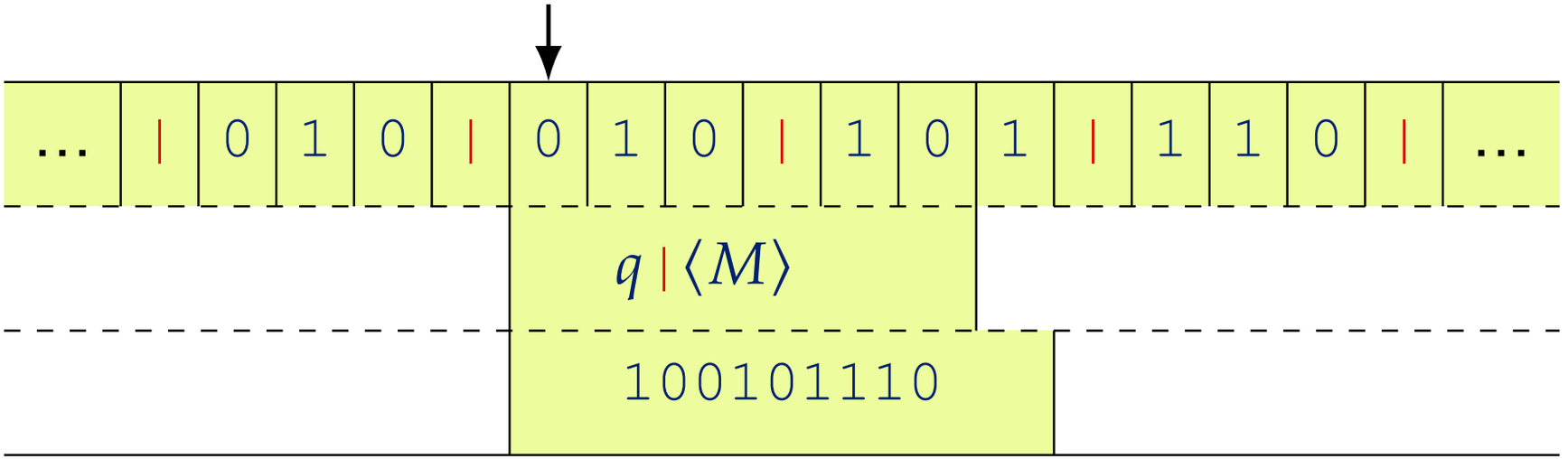
Aktualizace čítače kroků

- Čítač má $O(\log_2 f(n))$ bitů
- Snížení hodnoty o 1 lze provést v lineárním čase vzhledem k počtu bitů
- Nejprve však musí hlava M přejít k čítači na pásce

Čítač musí být neustále poblíž hlavy D

- Přidáme novou stopu pro uložení čítače
- Po každém kroku simulace je stopa posunuta posunuta, aby čítač byl u hlavy D

Třetí stopa pásky stroje D



Stopa 3: Binární čítač
$O(\log_2 f(n))$ bitů

Zarovnaná s hlavou D

Posouvá se po každém kroku simulace

Zarovnání stop

Stopa 2 obsahuje dvojici $q \mid \langle M \rangle$

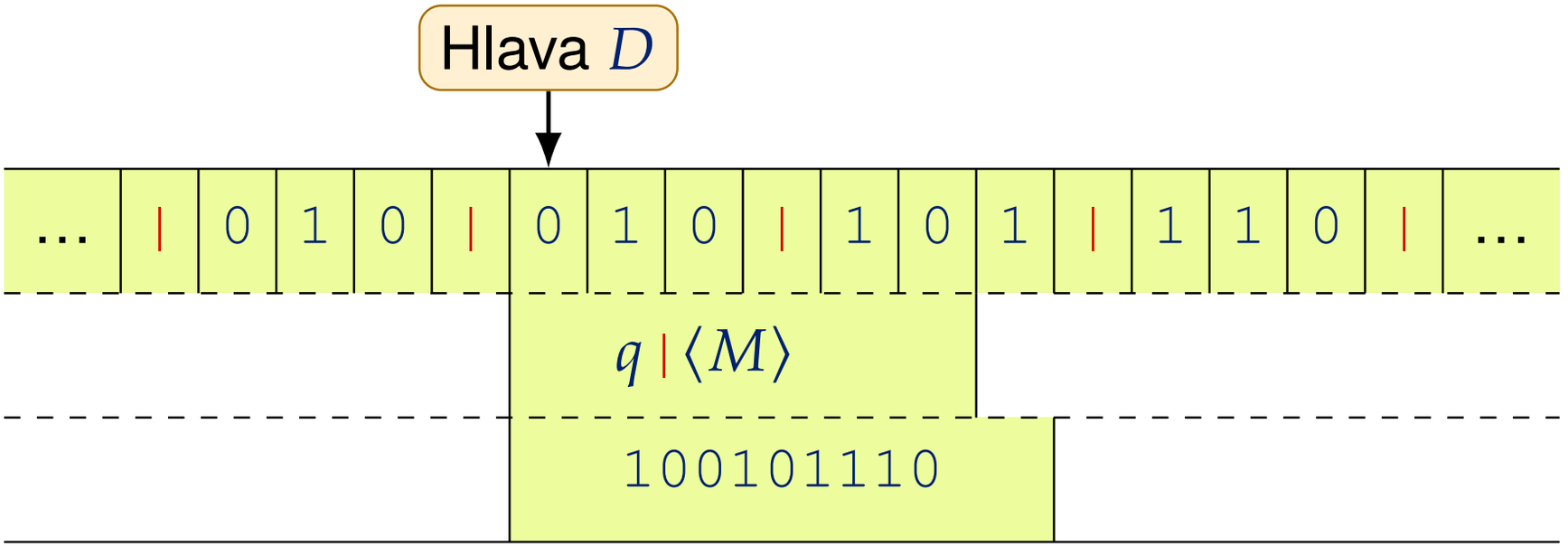
- Na začátku simulace kroku M musí být zarovnaná s blokem pod hlavou M
- Poté, co D dokončí simulaci kroku M, je stopa posunuta podle toho, kam se pohne hlava M

Stopa 3 obsahuje čítač

- Na začátku každého kroku D v rámci simulace musí být zarovnaná s hlavou D
- Jeden krok M je proveden pomocí c_M kroků simulace
- Po každém kroku simulace dojde k posunu čítače
- Čítač se tedy posune c_M-krát během simulace jednoho kroku M

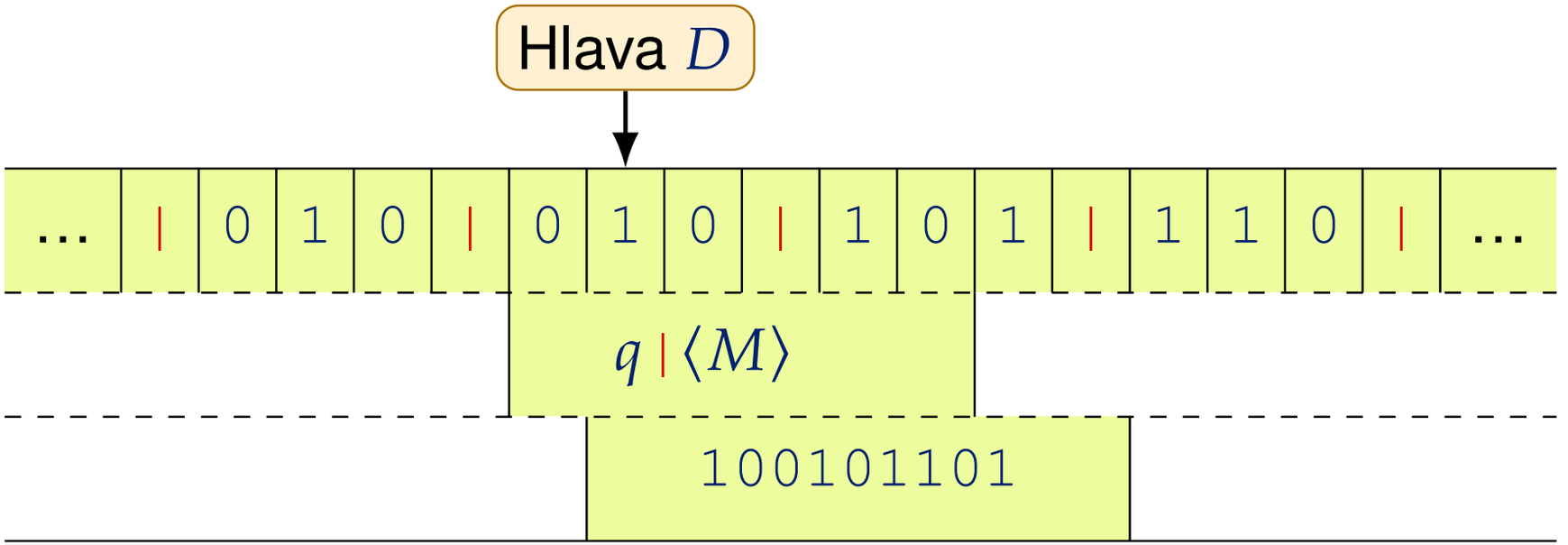
Zarovnání stop

Stopa 3 je zarovnaná s hlavou D při simulaci



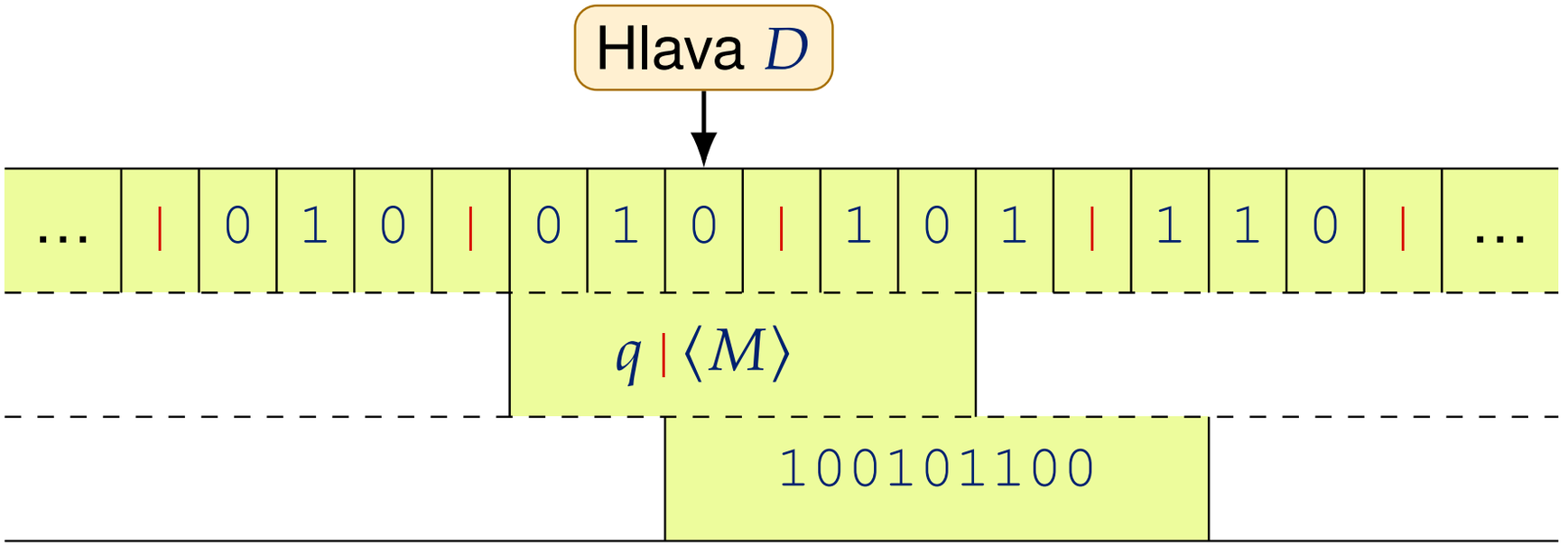
Zarovnání stop

Stopa 3 se posouvá s hlavou D po každém kroku simulace



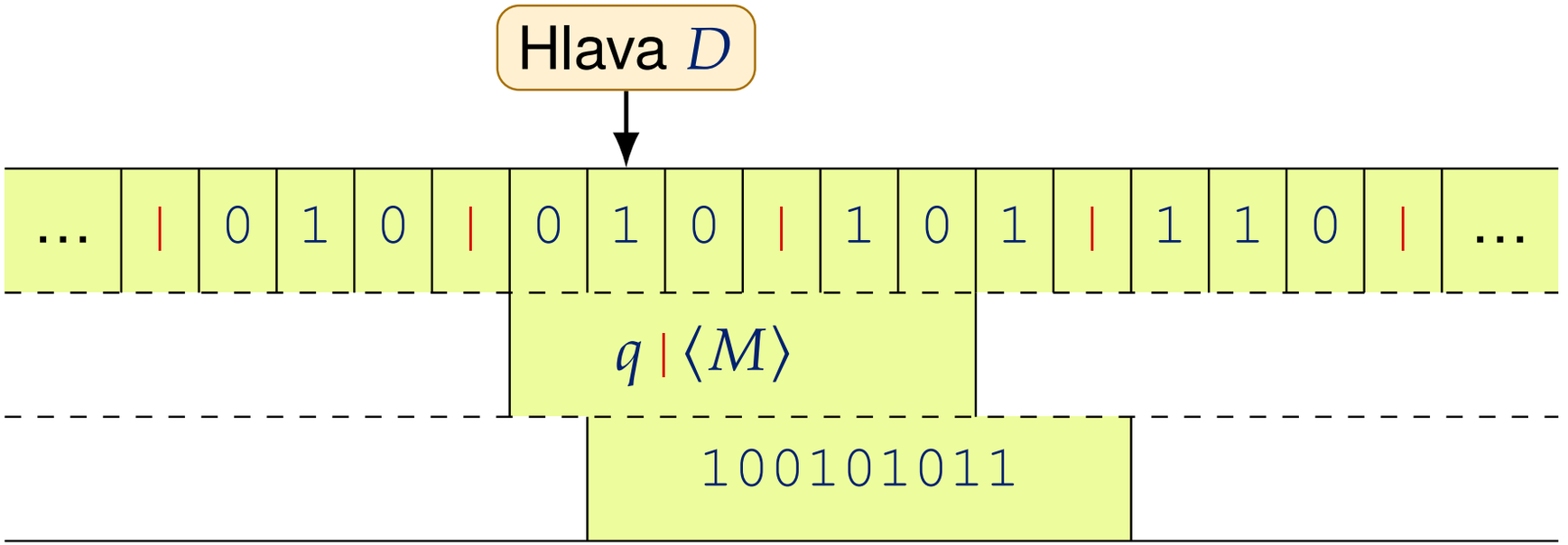
Zarovnání stop

Stopa 3 se posouvá s hlavou D po každém kroku simulace



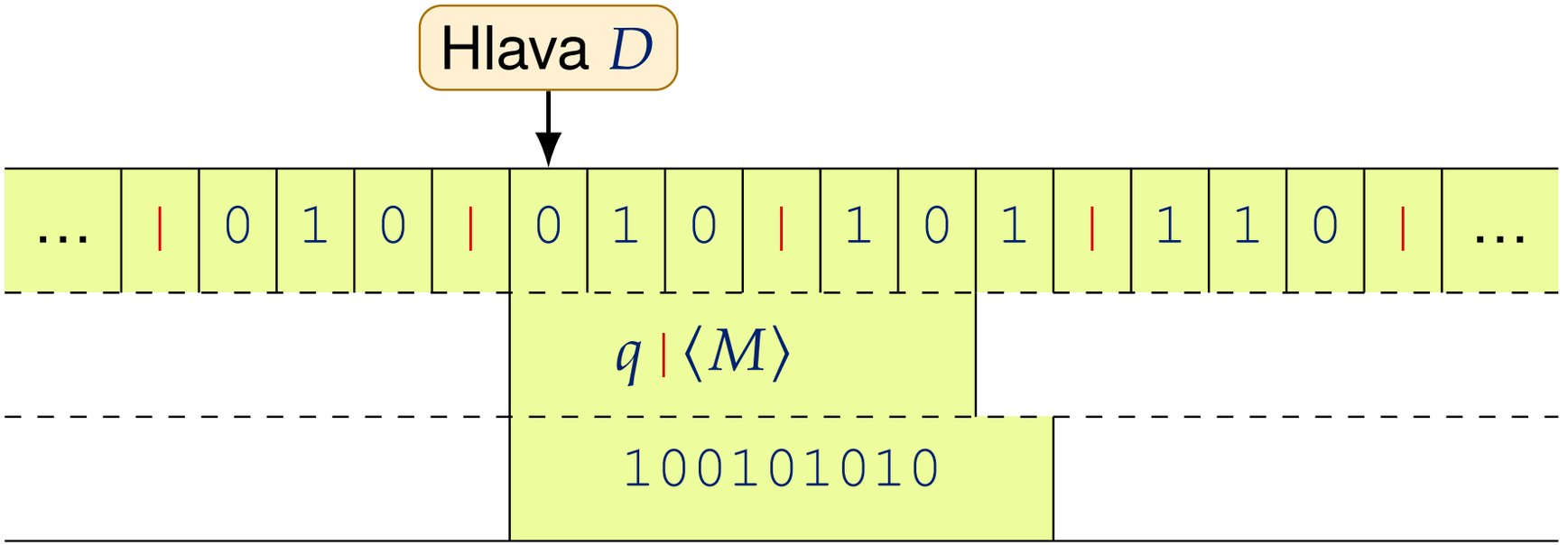
Zarovnání stop

Stopa 3 se posouvá s hlavou D po každém kroku simulace



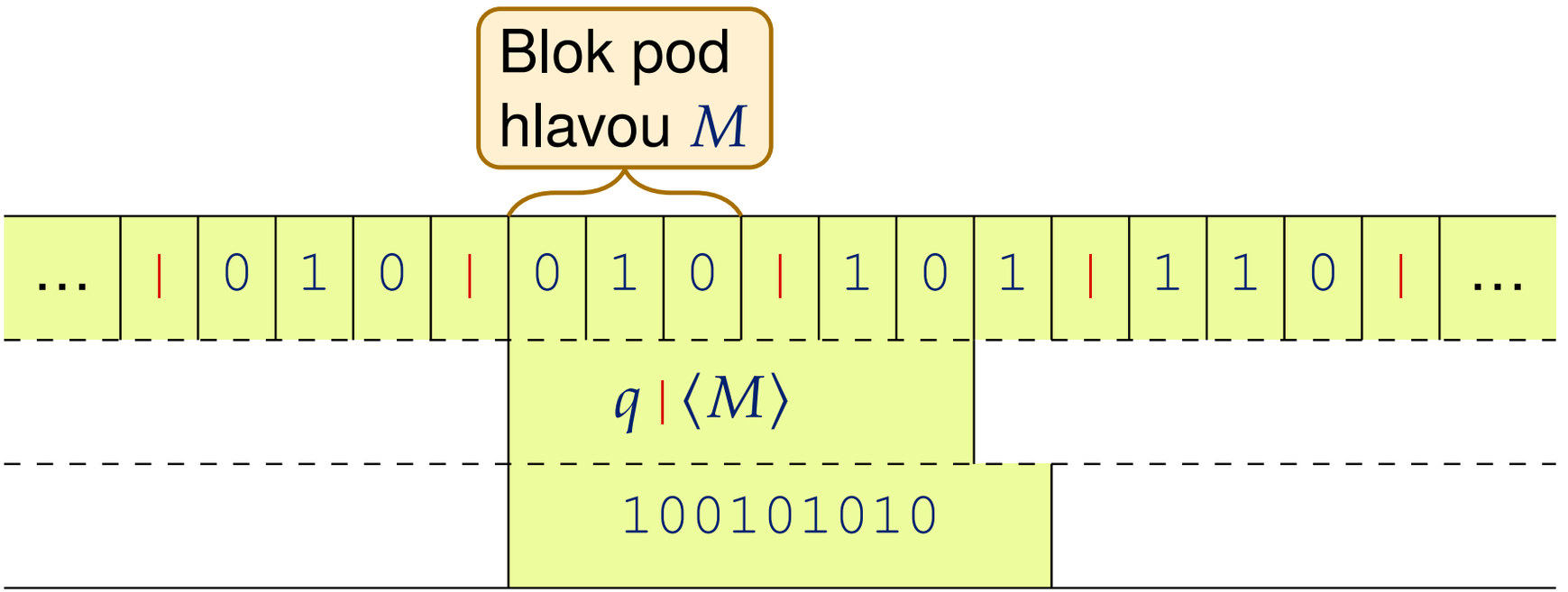
Zarovnání stop

Stopa 3 se posouvá s hlavou D po každém kroku simulace



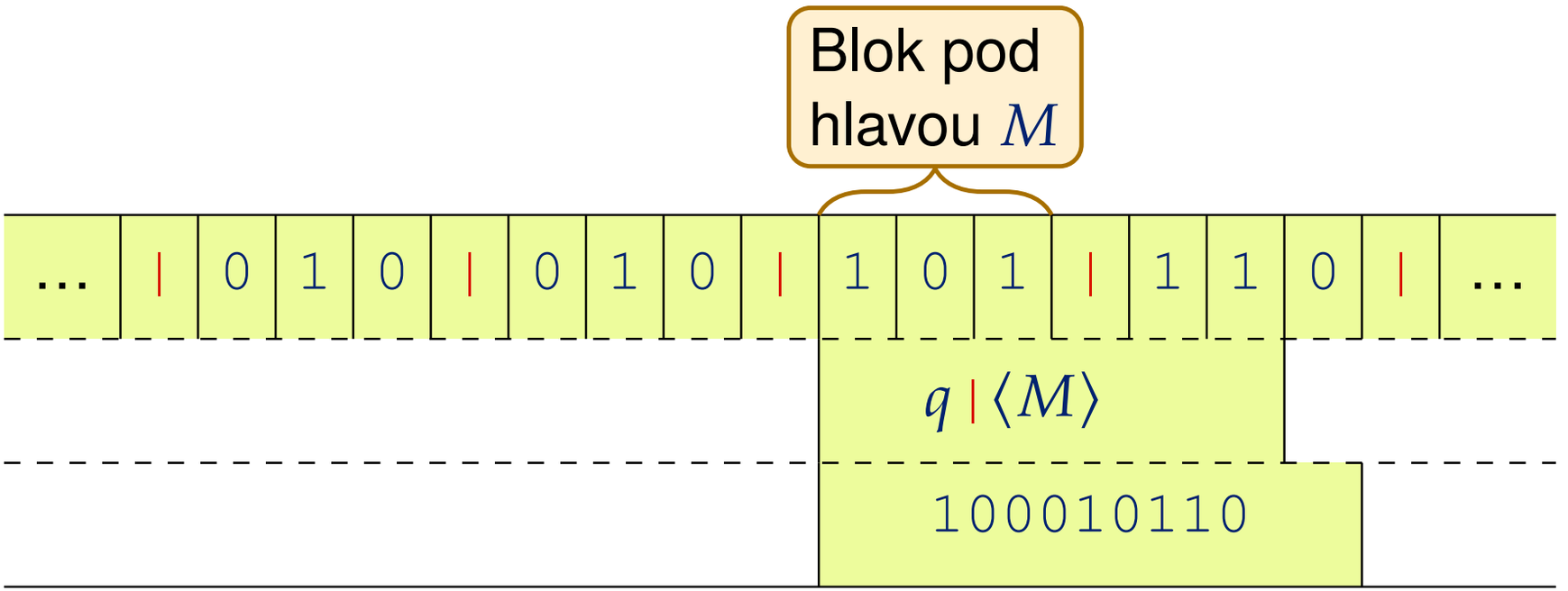
Zarovnání stop

Stopa 2 je zarovnaná s blokem pod hlavou M



Zarovnání stop

Pokud se hlava M pohne, stopa 2 je posunuta



Stopa 3 se posunula spolu s hlavou D

Časová složitost D

- Simulace M je ukončena nejpozději po provedení $\lceil f(n)/\log_2 f(n) \rceil$ kroků simulace
  - D tedy odsimuluje zhruba $\frac{f(n)}{c_M \log_2 f(n)}$ kroků M
- Hodnota čítače se sníží o 1 a případně je posunut po každém kroku simulace
  - $O(\log_2 f(n))$ kroků stačí pro snížení hodnoty
  - $O(\log_2 f(n))$ kroků stačí na posunutí
- Dohromady dostáváme čas

$$O\left(\frac{f(n)}{\log_2 f(n)} \log_2 f(n)\right) = O(f(n))$$

TS D pracuje v čase $O(f(n))$.

Časová složitost rozhodování A (horní odhad)

Jazyk $A = L(D)$ lze rozhodnout v čase $O(f(n))$.

Ukážeme, že A nelze rozhodnout v čase $o(f(n)/\log_2 f(n))$.

Menší čas nestačí

- Nechť $M = (Q, \Sigma, \delta, q_0, F)$ je TS, který pracuje v čase $g(n) = o(f(n)/\log_2 f(n))$
- Ukážeme, že $A \neq L(M)$
- Dříve jsme ukázali, že

Simulaci $M(x)$ lze provést pomocí $c_M g(n)$ kroků, kde c_M je konstanta, jež závisí na M.

Menší čas nestačí

- Z toho, že $g(n) = o(f(n)/\log_2 f(n))$, plyne

$$(\exists n_0 \in \mathbb{N})(\forall n \geq n_0)[c_M g(n) \leq f(n)/\log_2 f(n)]$$

- Předpokládejme vstup $x = \langle M \rangle 10^{n_0}$
- Tedy $|x| > n_0$
- D simuluje M po $f(n)/\log_2 f(n)$ kroků
  - Simulace M se vstupem $x = \langle M \rangle 10^{n_0}$ skončí a
  - $D(x)$ přijme, právě když $M(x)$ odmítne

$$L(D) \neq L(M)$$

Časová hierarchie

Věta (O deterministické časové hierarchii)

Pro každou časově konstruovatelnou funkci $f : \mathbb{N} \to \mathbb{N}$ *existuje jazyk* A*, který je rozhodnutelný v čase* $O(f(n))$*, nikoli však v čase* $o(f(n)/\log_2 f(n))$*.*

Důsledek

Jsou-li $f_1, f_2 : \mathbb{N} \to \mathbb{N}$ *funkce, pro které platí, že* $f_1(n) \in o(f_2(n)/\log_2 f_2(n))$ *a* f_2 *je časově konstruovatelná, potom*

$$\mathrm{TIME}(f_1(n)) \subsetneq \mathrm{TIME}(f_2(n))$$

Polynomy a související funkce

Důsledek

Pro každá dvě reálná čísla $1 \leq \epsilon_1 < \epsilon_2$,

$$\mathrm{TIME}(n^{\epsilon_1}) \subsetneq \mathrm{TIME}(n^{\epsilon_2})$$

- Je-li ϵ_2 racionální číslo, pak
  - n^{ϵ_2} je časově konstruovatelná
    - Lze jednoduše ukázat pro přirozená čísla
    - Lze ukázat i pro racionální čísla
  - Ostrá inkluze plyne z časové hierarchie
- Je-li ϵ_2 iracionální číslo
  - Racionální čísla jsou hustá v reálných číslech
  - Existuje racionální číslo ϵ splňující $\epsilon_1 < \epsilon < \epsilon_2$
  - Z časové hierarchie a časové konstruovatelnosti n^{ϵ}

$$\mathrm{TIME}(n^{\epsilon_1}) \subsetneq \mathrm{TIME}(n^{\epsilon}) \subseteq \mathrm{TIME}(n^{\epsilon_2})$$

Polynomiální vs. exponenciální čas

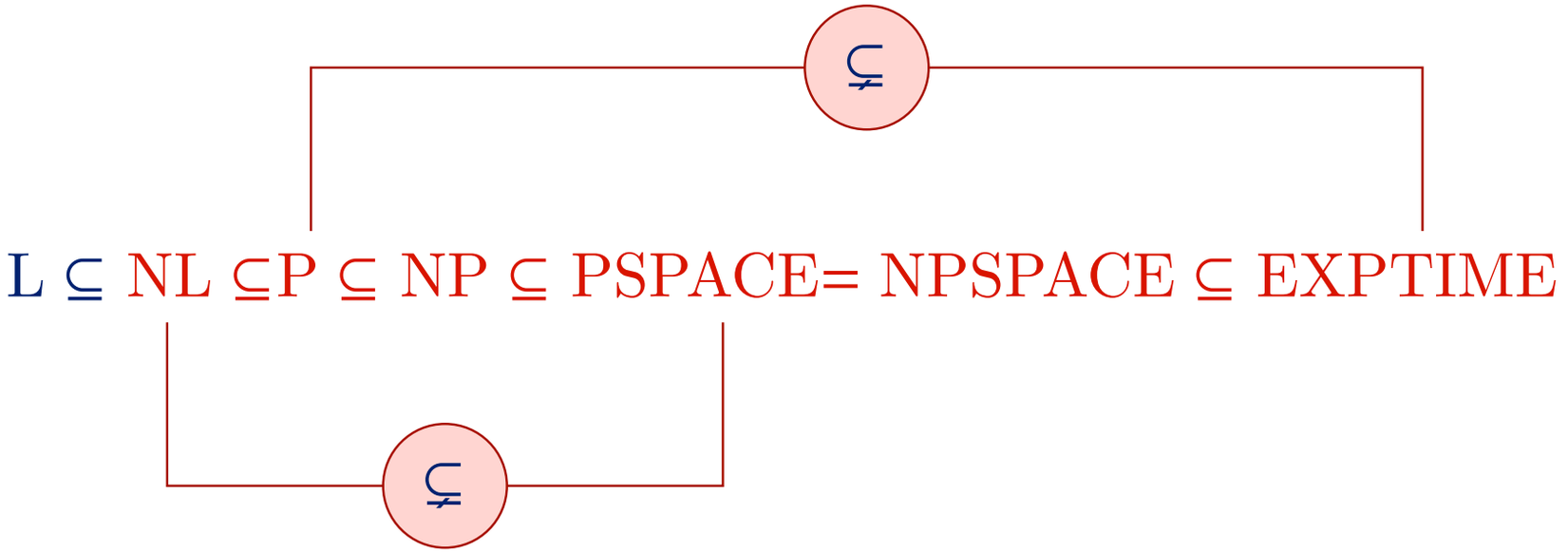
Důsledek

$\mathrm{P} \subsetneq \mathrm{EXPTIME}$

- Pro každé $k \in \mathbb{N}$ platí $\mathrm{TIME}(n^k) \subseteq \mathrm{TIME}(2^n)$
- Podle časové hierarchie tedy

$$\mathrm{P} \subseteq \mathrm{TIME}(2^n) \subsetneq \mathrm{TIME}(2^{n^2}) \subseteq \mathrm{EXPTIME}$$

Vztahy mezi třídami



- Jedna z inkluzí $\mathrm{NL} \subseteq \mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE}$ musí být ostrá
- Jedna z inkluzí $\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} \subseteq \mathrm{EXPTIME}$ musí být ostrá

Nevíme, která z inkluzí je ostrá

Další věty o hierarchii

Prostor

- Nedeterministická prostorová hierarchie
  - $\mathrm{NSPACE}(f_1(n)) \subsetneq \mathrm{NSPACE}(f_2(n))$ pokud $f_1(n) = o(f_2(n))$

Čas

- Platí v tomtéž znění i pro k-páskové stroje
  - Faktor $\log_2 f(n)$ je způsoben redukcí počtu pásek z k na 2
- Nedeterministická časová hierarchie
  - $\mathrm{NTIME}(f_1(n)) \subsetneq \mathrm{NTIME}(f_2(n))$ pokud $f_1(n+1) = o(f_2(n))$
- Časová hierarchie pro RAM
  - Není třeba faktor $\log_2 f(n)$