Extrakce N-gramů z rozsáhlých textů

Zdeněk Češka
Ivo Hanák
Roman Tesař

ZÁPADOČESKÁ UNIVERZITA V PLZNI

Znalosti 2008

Obsah

- Využití N-gramů
- Myšlenka algoritmu Teraman
- Předzpracování a indexace
- Výpočet četností N-gramů
- De-indexace
- Experimenty
- Závěr
- Budoucí práce

Co jsou N-gramy?

- Pod sekvence N položek
  - Písmen
  - Čísel
  - Slov
  - Libovolných jiných sekvencí
- Příklad N-gramů na slovech
  - Vzorový text
    - “Lidé mohou řídit auta bezpečně.”
  - Trigramy
    - “lidé mohou řídit”
    - “mohou řídit auta”
    - “řídit auta bezpečně”

Využití N-gramů

- V textových dokumentech reprezentují N-gramy jednotlivé fráze
- K obohacení základního bag-of-word modelu
  - Klasifikace textových dokumentů
  - Shlukování textových dokumentů
  - Filtrace spamů
  - atd.
- V biologii jsou N-gramy extrahovány pro určení výskytu jednotlivých proteinových sekvencí

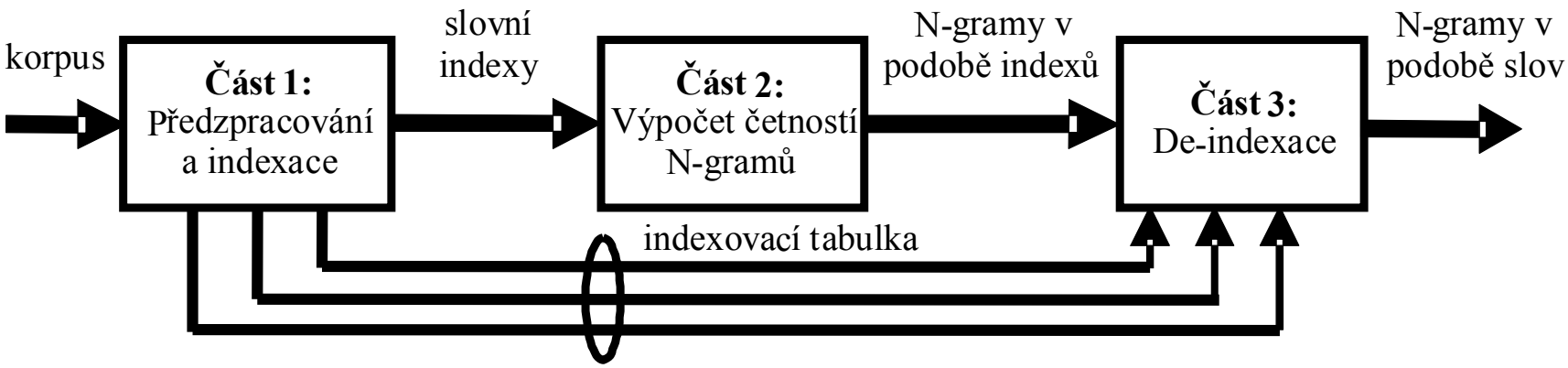
Myšlenka algoritmu Teraman

- Extrakce N-gramů spolu s jejich četností z velmi rozsáhlých textových korpusů
- Dosažení vysokého výkonu extrakce N-gramů i v případě, že vstupní korpus dalece přesahuje velikost operační paměti
- Schopnost běhu na běžném počítači bez potřeby specializovaného hardwarového vybavení

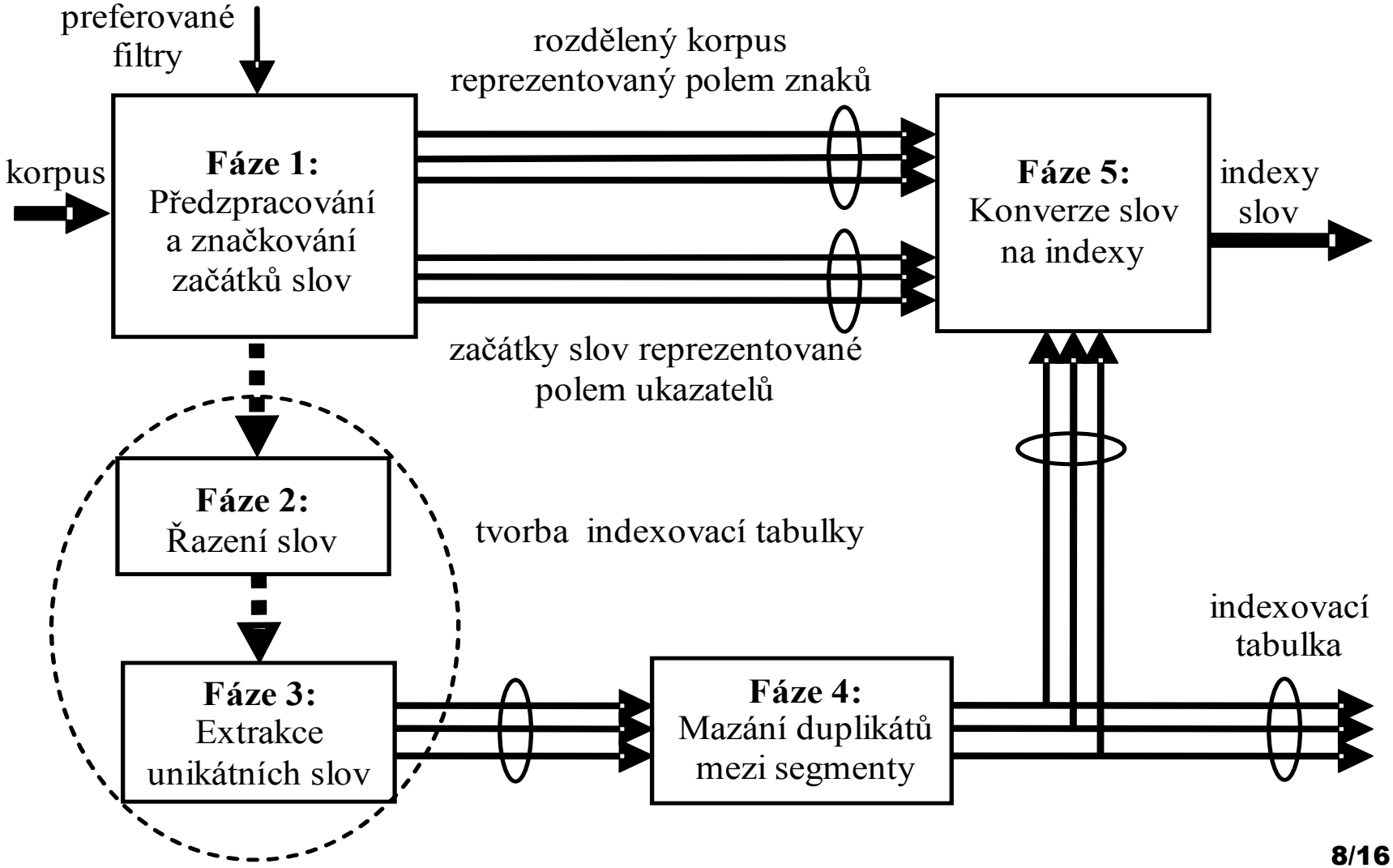
Použité postupy

- Dávkové zpracování
- Klouzavé okénko
- Quick Sort
- Optimalizace kódu
  - C# .NET Framework 2.0
  - Využití neřízeného kódu v kritických sekcích kde se provádí velké množství operací nad poli
  - Přímá adresace struktur nesoucí N-gramy prostřednictvím ukazatelů

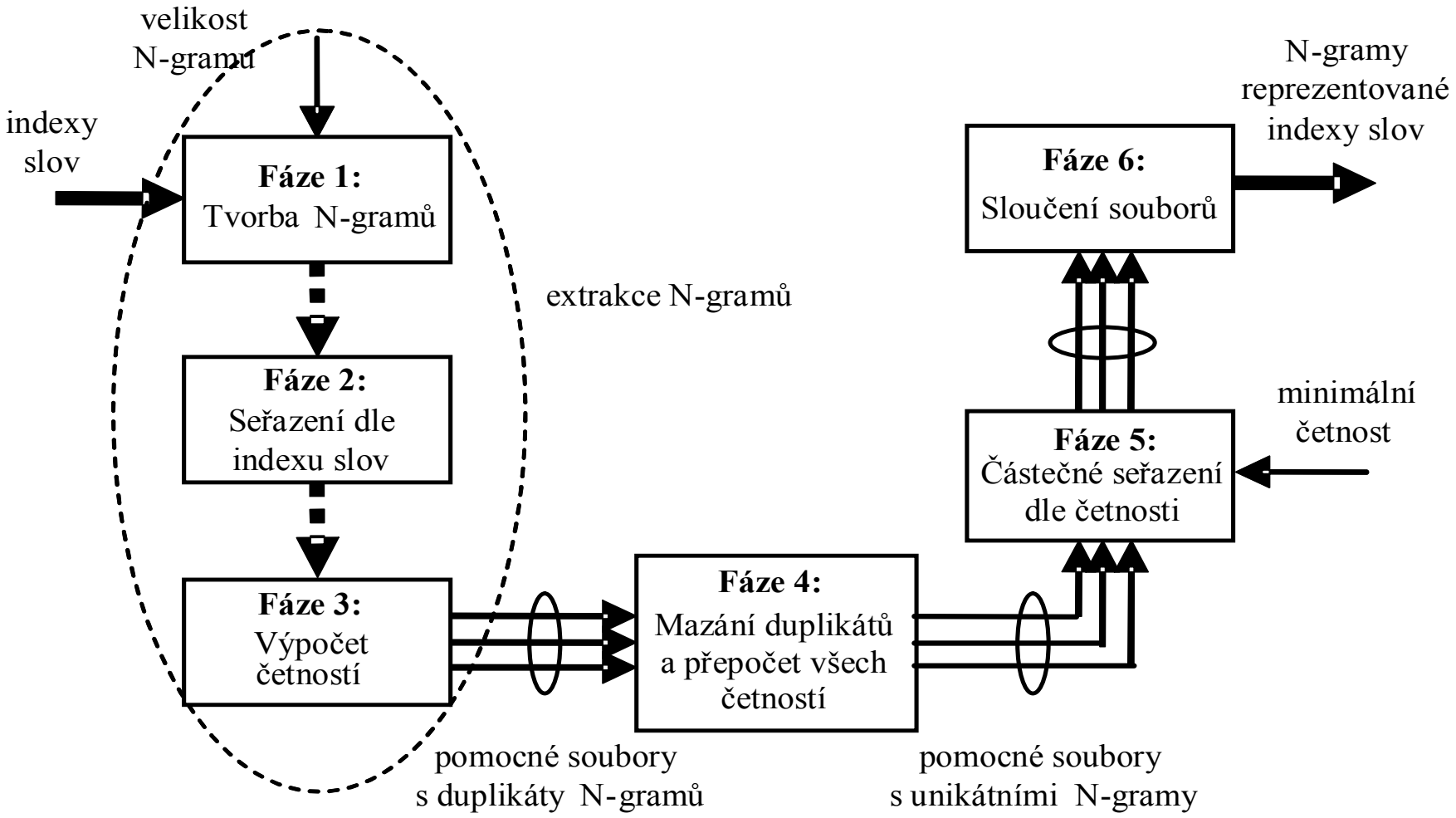
Hlavní části algoritmu



Předzpracování a indexace



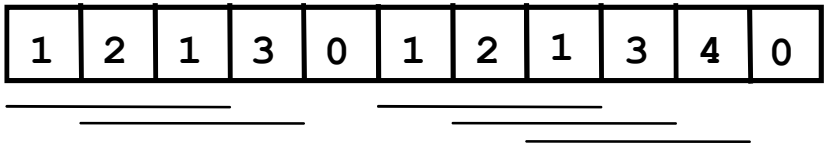
Výpočet četnosti N-gramů



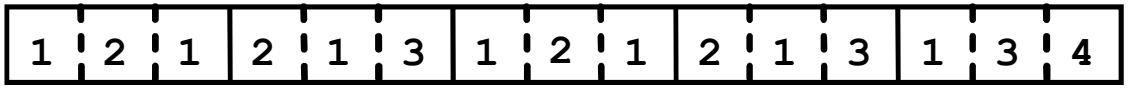
Příklad extrakce N-gramů

(Fáze 1 až 3)

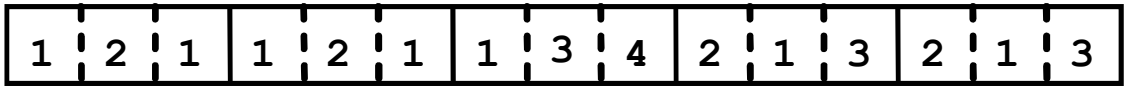
(a) slovní indexy



(b) vytvořené struktury N-gramů



(c) seřazené struktury N-gramů



(d) četnosti N-gramů uložené v souboru

		3	3
1	2	1	2
1	3	4	1
2	1	3	2

Odstraňování duplikátů

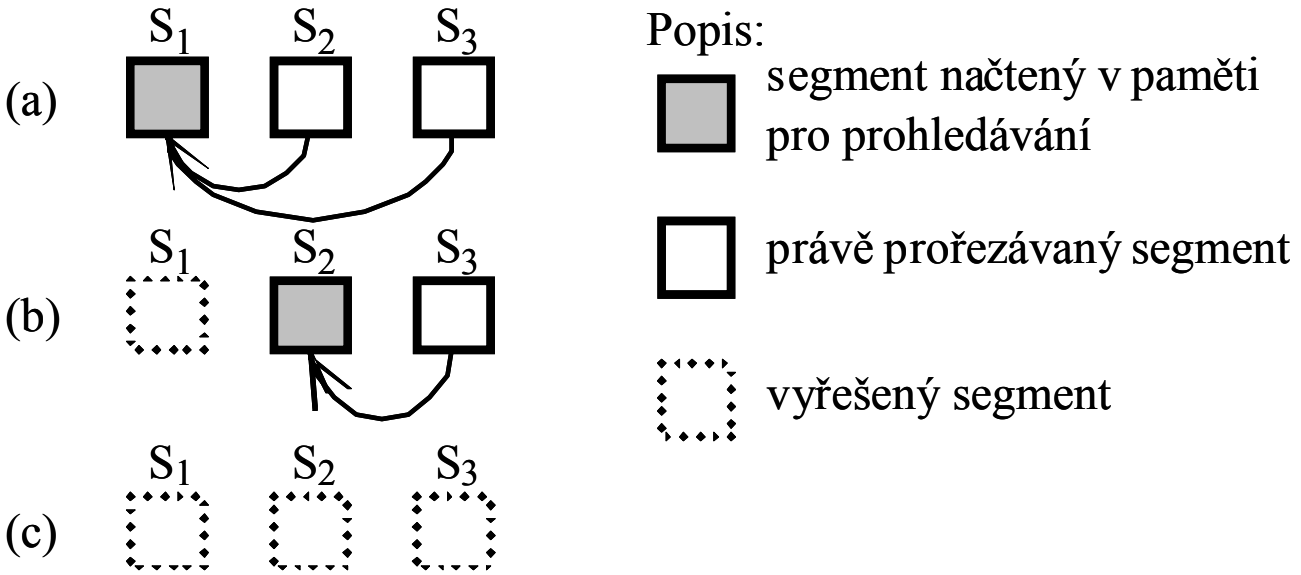
Nedostatek paměti

⇩

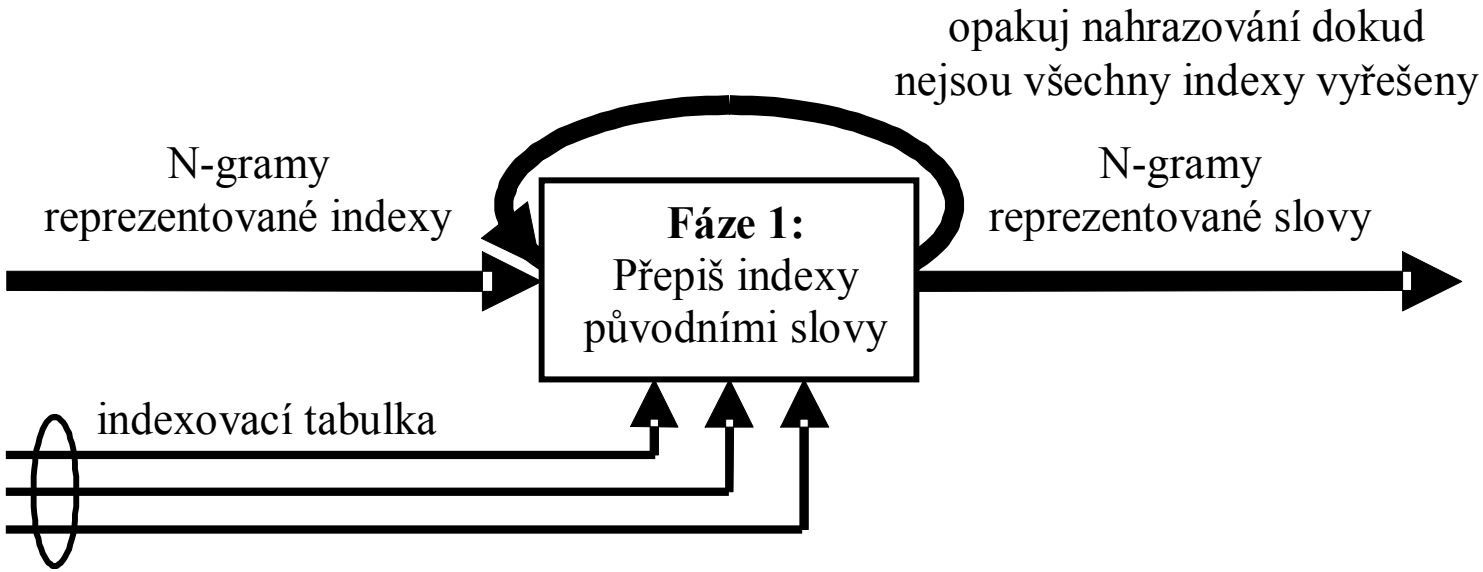
Data jsou rozdělena na více segmentů (bloků) a zpracována odděleně

⇩

Nutnost odstranit duplikáty vzniklé samostatným zpracováním jednotlivých segmentů

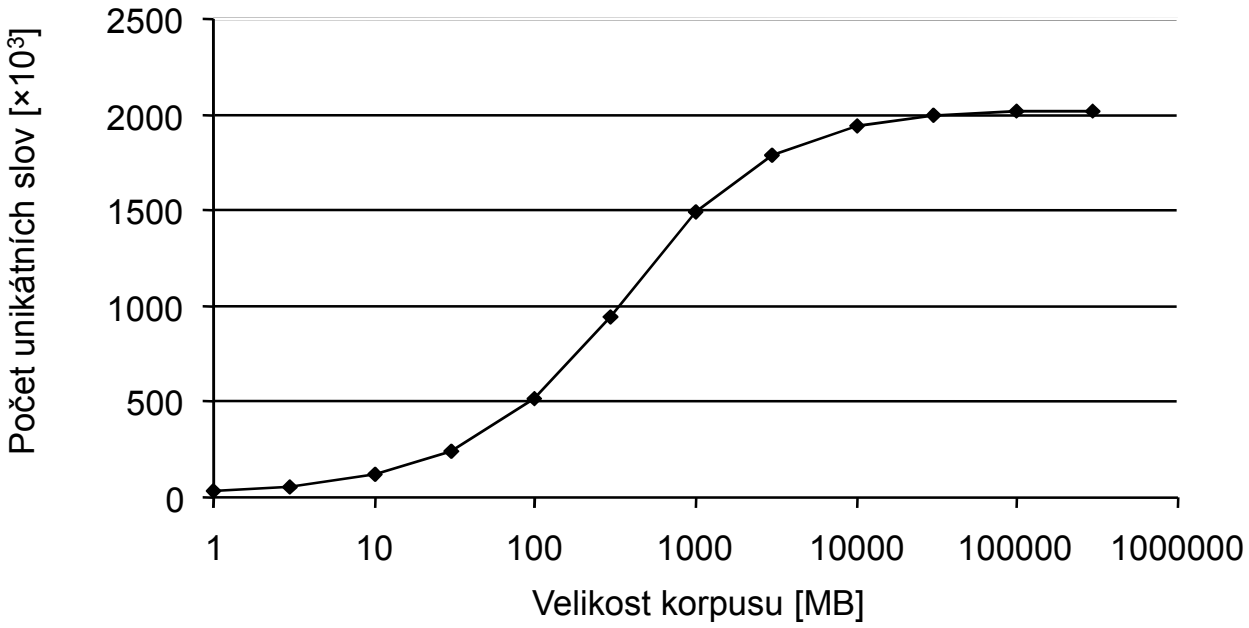


De-indexace

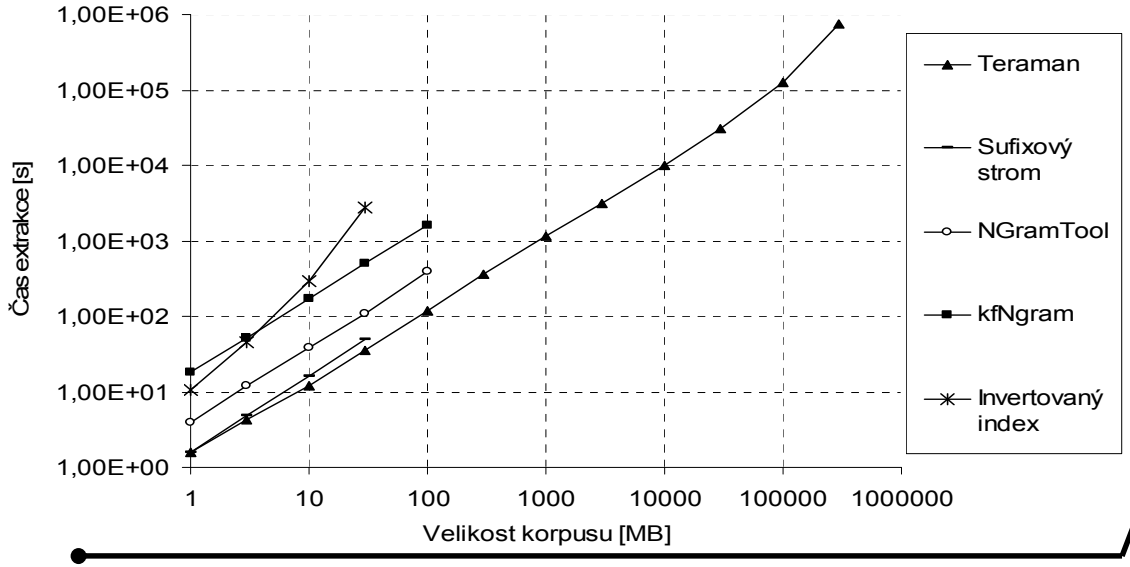

opakuj nahrazování dokud
nejsou všechny indexy vyřešeny
N-gramy
reprezentované indexy
Fáze 1:
Přepiš indexy
původními slovy
N-gramy
reprezentované slovy
indexovací tabulka

Testovací data

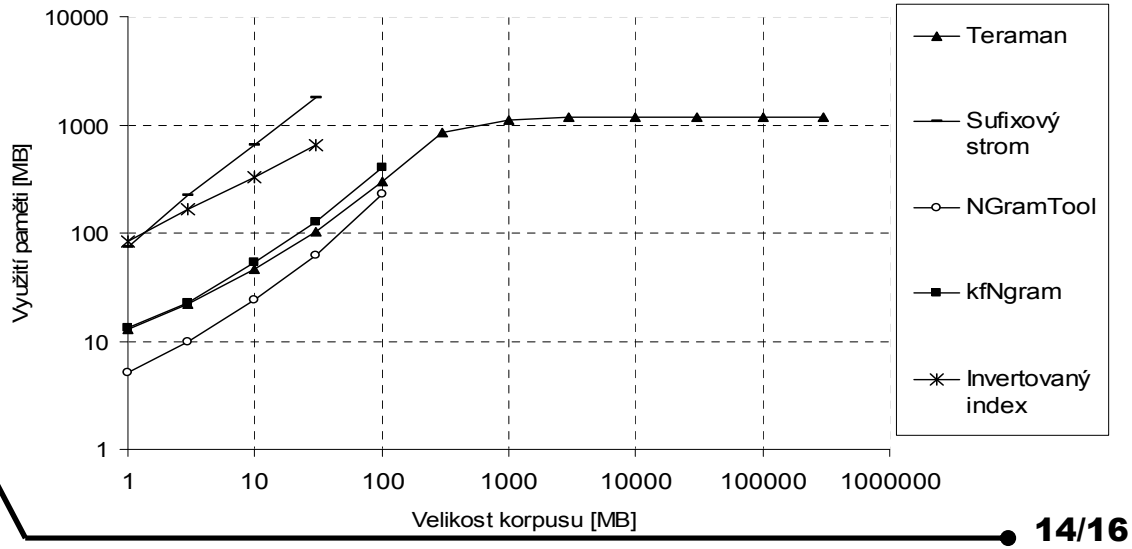
- Množina textových korpusů od 1MB do 300GB
- Vytvořeno z korpusu “Web 1T 5-gram Version 1” uvolněného společností Google
- Na disku je nutné mít 2.2 násobek volného místa vzhledem k velikosti vstupního korpusu; při použití 1TB disku jsem mohli zpracovat nejvýše 300GB korpus



Časové a paměťové požadavky



Čas vyžadovaný pro extrakci 1- až 4-gramů



Maximální využití paměti během extrakce 1- až 4-gramů

Závěr

- Vyšší výkon než ostatní dostupné nástroje
- Schopnost zpracovat libovolné množství dat bez ohledu na velikost operační paměti
- Superlineární časová složitost

		Nejlepší dosažitelná časová složitost	Nejhorší dosažitelná časová složitost	Složitost rozšířené paměti
Část 1	Fáze 1	$O(n)$		$O(1)$
	Fáze 2	$O(n \cdot \log_2(n / k))$		
	Fáze 3	$O(n)$		
	Fáze 4	$O(t \cdot \log_2(t / k))$	$O(k \cdot t \cdot \log_2(t / k))$	$O(1)$
	Fáze 5	$O(k \cdot n \cdot \log_2(t / k))$		$O(1)$
Část 2	Fáze 1	$O(s \cdot n)$		$O(s \cdot n / k)$
	Fáze 2	$O(s \cdot n \cdot \log_2(s \cdot n / k))$		
	Fáze 3	$O(s \cdot n)$		
	Fáze 4	$O(n \cdot \log_2(n / k))$	$O(k \cdot n \cdot \log_2(n / k))$	$O(1)$
	Fáze 5	$O(n \cdot \log_2(n / k))$		$O(1)$
	Fáze 6	$O(\log_2(k) \cdot n)$		$O(1)$
Část 3	Fáze 1	$O(k \cdot n)$		$O(1)$

Budoucí práce

- Využití *Trie* pro indexaci slov a porovnání se současným přístupem
- Vylepšení časové složitosti algoritmu při odstraňování duplikátů N-gramů
- Podpora nezávislého čtení a zápisu na více disků současně

Děkuji za pozornost

Nyní je prostor na případné dotazy