VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Srovnání booleovského modelu a modelu založeného na signaturních souborech

Comparison of Boolean Model and Model Based on Signature Files

2015 Radek Zika

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Radek Zika**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Srovnání booleovského modelu a modelu založeného na signaturních souborech
Comparison of Boolean Model and Model Based on Signature Files

Zásady pro vypracování:

Booleovský model je v současnosti velmi používaný pro vyhledávání a indexaci textových dokumentů. Funkce modelu je založena na existenci indexovaných termů v dokumentu.
Model založený na signaturních souborech je velmi podobný booleovskému modelu, neboť indexuje termy obsažené v dokumentu do signatur pomocí hashovací funkce.
Cílem práce je porovnat booleovský model a model založený na signaturních souborech v oblasti prostorové a časové složitosti nad vybranou kolekcí dokumentů a porovnat také výsledky obou modelů na položené dotazy.
1. Popište vlastnosti, výhody a nevýhody booleovského modelu.
2. Naimplementujte booleovský model.
3. Prozkoumejte možnosti tvorby signaturních souborů a možnosti pro ukládání a vyhledávání signaturních souborů. Popište vlastnosti signaturních souborů
4. Naimplementujte model založený na signaturních souborech.
5. Porovnejte výsledky dotazů mezi booleovským modelem a modelem založeným na signaturních souborech.

Seznam doporučené odborné literatury:

[1] Pokorný J., Snášel V., Kopecký M.:Dokumentografické informační systémy.ISBN 80-246-1148-1
[2] Wartik, Steven (1992). "Boolean operations". Information Retrieval Data Structures & Algorithms. Prentice-Hall, Inc. ISBN 0-13-463837-9

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. Petr Berek**

Datum zadání: 01.09.2013

Datum odevzdání: 07.05.2015

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 7. května 2015 .....

Rád bych na tomto místě poděkoval Ing. Petru Berkovi za odborné vedení, za pomoc a rady při zpracování této práce.

Abstrakt

Cílem práce bylo porovnat booleovský model a model založený na signaturních souborech. Práce uvádí teoretické základy obou modelů, je zde tedy popsáno jak reprezentují dokumenty a jak přistupují k vyhodnocení dotazů zadaných uživatelem. Vytvořené aplikace byly použity pro srovnání, kde se porovnávalo vytváření indexu z hlediska času potřebného pro vytvoření, tak i paměti, kterou tento index zabírá na disku. Poté byly porovnány výsledky vyhodnocení pro různé dotazy a čas, který byl u obou modelů potřebný. Z důvodu, že signaturní soubory pro dotaz vybírají i ty dokumenty, které nejsou relevantní, bylo provedeno porovnání nastavovaných hodnot pro zjištění nejvhodnějšího nastavení, které by vykazovalo nejlepší výsledky.

Klíčová slova: Booleovský model, signaturní soubory, invertovaný index, signatura, S-strom, vrstvené kódování, Dokumentografický informační systém, term, dokument, Hammingova vzdálenost

Abstract

The aim of the study was to compare boolean model and model based on signature files. The study presents the theoretical foundations of both models, so here is described how they represent documents and how they approach to the evaluation of the queries entered by the user. Created applications were used for comparison, where the index creation was compared in terms of required time for the creation and the memory that index occupies on the disk. After that the evaluation results were compared for different queries and time, which was needed for evaluation in both models. For the reason, that signature files select for the query even those documents, that are not relevant, so a comparison of set values was made to determine the best settings, which showed the best results.

Keywords: Boolean model, signature files, inverted index, signature, S-tree, superimposed coding, Documentographic information system, term, document, Hamming distance

Seznam použitých zkratek a symbolů

BCG	–	Bit-block compression with concatenated groups
BCP	–	Bit-block compression with concatenated parts
BM	–	Booleovský model
DIS	–	Dokumentografické informační systémy
MIME	–	Multipurpose Internet Mail Extensions
SS	–	Signaturní soubory

Obsah

1 Úvod **4**

2 Modely DIS **5**
2.1 Booleovský model 5
2.2 Signaturní soubory 10

3 Popis datasetu **20**

4 Porovnání modelů **21**
4.1 Tvorba indexu 21
4.2 Vyhodnocení dotazů 23
4.3 Vliv počtu termů v dotazu na vyhodnocení dotazu 25
4.4 Detekce spamu 28

5 Nastavení signaturních souborů **31**
5.1 Velikost signatury 31
5.2 Počet nastavovaných bitů 33

6 Závěr **36**

7 Reference **37**

Přílohy **37**

A Obsah CD **38**

Seznam tabulek

1 Logické operátory 7
2 Proximitní operátory 7
3 Konstrukce regulárních výrazů 8
4 Operátory Oracle SQL*Text 9
5 Parametry 11
6 Spojování slovních signatur 12
7 Vrstvené kódování 12
8 Vyhodnocení dotazu pro vrstvené kódování 12
9 Bitově bloková komprese 13
10 Spojení skupin signatur 13
11 Vytváření indexu 21
12 Dotazy 23
13 Výsledky pro dotazy 24
14 Výsledky booleovský model 26
15 Výsledky signaturní soubory 27
16 Počet dokumentů pro daný rozsah 30
17 Počet chybně vybraných dokumentů pro dané velikosti signatury 31
18 Průměrný váha signatury pro dané velikosti 32
19 Počet chybně vybraných dokumentů pro daný počet nastavovaných bitů . 34
20 Průměrný váha signatury pro daný počet nastavovaných bitů 34

Seznam obrázků

1 Invertovaný index 6
2 Sekvenční soubor 15
3 Bitové řezy 15
4 Pevný prefix 16
5 Plovoucí klíč 17
6 S-strom 18
7 Dvouúrovňové vrstvené kódování 18
8 Čas potřebný pro vytváření indexu 22
9 Velikost indexu na disku 23
10 Porovnání časů pro dotazy 25
11 Porovnání časů pro dotazy bez načtení offsetu 25
12 Chybovost u signaturních souborů 26
13 Čas pro vyhodnocení dotazu pro různý počet termů v dotazu 28
14 Chybně vybrané dokumenty pro různou váhu dotazu 28
15 Průměrná chybovost v závislosti na váze dotazu 29
16 Procentuální navštívenost uzlů pro jednotlivé dotazy 29
17 Průměrný počet navštívených uzlů v závislosti na velikosti signatury dotazu 30
18 Chybně vybrané dokumenty pro různě velké signatury 32
19 Chybovost v závislosti na velikosti signatury 33
20 Průměrná využitost signatury pro daný počet nastavovaných bitů 33
21 Chybovost v závislosti na počtu nastavovaných bitů 35
22 Průměrná využitost signatury pro daný počet nastavovaných bitů 35

1 Úvod

Booleovský model a signaturní soubory jsou modely používané v dokumentografických informačních systémech. Dokumentografické informační systémy slouží k práci s textovými daty, jako je například ukládání a výběr dokumentů odpovídajících uživatelskému dotazu. Modely DIS poté představují jednak pohled na reprezentaci dokumentu, tak i poskytují formální pozadí pro dotazování. Booleovský model je poté typickým modelem DIS, jelikož se zaobírá oběma body, kdežto signaturní soubory primárně řeší problém reprezentace dokumentu, ale i tak se dají použít jako model pro vyhledávání. Práce se poté zaobírá srovnáním těchto modelů při použití pro vyhledávání v rozsáhlých kolekcích dokumentů. Je provedeno porovnáním těchto modelů a to jak z hlediska paměťové složitosti, tak i časové, ale je i provedeno porovnání výsledků pro různé dotazy.

V kapitole modely DIS jsou popsány teoretické základy obou modelů. U booleovského modelu je popsáno jakou datovou strukturu používá, tedy je popsán invertovaný index. Dále je popsáno jakými možnostmi pro tvorbu dotazu oplývá. Na závěr je booleovský model popsán z hlediska jeho výhod a nevýhod. Poté jsou popsány signaturní soubory. Zde jsou popsány dva základní problémy a to metody pro vytváření signatur a dále pak jejich organizace.

V kapitole popis datasetu je popsáno jaká kolekce dat je použita u obou aplikací a poté je popsán problém předzpracování těchto dat pro další použití. Na závěr je nastíněn problém s nastavením u signaturních souborů pro dosažení co nejlepších výsledků.

Kapitola porovnání modelů se zaobírá srovnáním booleovského modelu a modelu založeného na signaturních souborech se zvoleným nastavením. Je provedeno porovnání tvorby indexu a to jak z hlediska času, tak i paměti, kterou tento index zabírá na disku. Dále je provedeno porovnání výsledků pro různé dotazy a čas potřebný pro jejich vyhodnocení. Dále je ukázáno jak počet termů v dotazu ovlivňuje oba modely. Na závěr jsou oba modely porovnány jako aplikace na detekci spamu.

Poslední kapitola se zaobírá porovnáním nastavení signaturních souborů, které vykazovalo nejlepší výsledky, tedy bylo zvoleno v aplikaci a bylo na něm provedeno porovnání s booleovským modelem. Je tedy ukázáno srovnání počtu chybně vybraných dokumentů pro různé velikosti signatury a jak jednotlivé velikosti signatury byly efektivně využity. Stejné srovnání je provedeno i pro počet nastavovaných bitů pro jednotlivé termy.

2 Modely DIS

Jelikož v DIS se pracuje s dokumenty a jejich textovým obsahem, tak je vhodné s těmito texty pracovat na více úrovních abstrakce, tedy kromě textů je vhodné popsat jejich schéma pomocí odpovídajícího modelu. Model DIS je definován jako soubor pojmů či nástrojů pro reprezentaci dokumentu k popisu informace obsažené v dokumentu, dále pak k reprezentaci dotazu umožňující uživateli specifikovat požadavek na informace a k reprezentaci postupu jak určit shodu mezi požadavkem od uživatele a dokumenty, které tomuto požadavku vyhovují. [1]

2.1 Booleovský model

Booleovský model a jeho modifikace je nejstarší model používaný v dokumentografických informačních systémech, přesto je stále poměrně velice často používaný. Teoretické základy tohoto modelu byly popsány už v 50. letech minulého století, jednalo se o první systémy automatizace knihovnictví. Přesto se však tyto systémy ve velké míře používají dodnes pro jejich snadnou implementaci a lze jej vidět například u internetového vyhledávače AltaVista3. [1, 8]

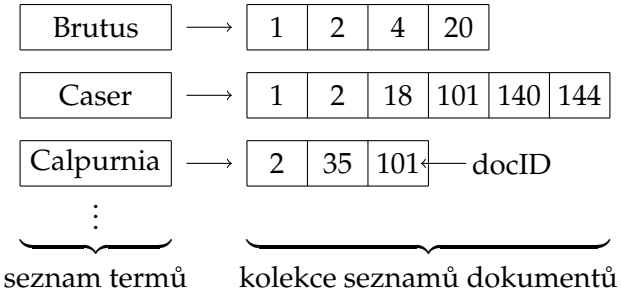
V booleovském modelu je každý dokument reprezentován pomocí množiny termů, které jej co nejlépe popisují. U booleovského modelu se jako datová struktura používá invertovaný index, ten nepřiřazuje k dokumentům jednotlivé termy, ale naopak k jednotlivým termům je přiřazen seznam, který obsahuje ukazatele na dokumenty, ve kterých se daný term nachází. Kvalitní invertovaný index je jednou z nejdůležitějších částí, který je zapotřebí pro vytvoření kvalitního booleovského modelu. Další důležitou částí jsou možnosti pro kladení dotazu nad vytvořeným invertovaným indexem. Popis a tvorba invertovaného indexu a možnosti pro kladení dotazu jsou detailněji popsány v následujících částech. [3, 4]

2.1.1 Invertovaný index

Jak již bylo zmíněno, tak invertovaný index je použit jako datová struktura u booleovského modelu. Důvodem pro vytvoření indexu je to, že vytvořením indexu odpadá potřeba neustálého procházení textu při vyhodnocování dotazů. Při základní variantě se invertovaný index skládá ze seznamu termů a kolekce seznamů dokumentů, kde každý term je asociován se seznamem dokumentů, ve kterém se vyskytuje. Struktura takového indexu je poté znázorněna na Obrázku 1.

U seznamu termů každý term ukazuje na seznam prvků, kde každý prvek obsahuje jednoznačný identifikátor dokumentu (docID) a přídavná data. V nejjednodušším případě jsou přídavná data prázdná, protože pro jednoduchý booleovský model nejsou žádné dodatečné informace, kromě docID, zapotřebí. Existence prvku s daným identifikátorem již indikuje přítomnost termu v dokumentu. [2]

Nejčastějším obsahem přídavných dat bývá frekvence termu, což umožňuje ohodnotit dokumenty podle počtu výskytů termu v daném dokumentu. Více precizní obsah



Obrázek 1: Invertovaný index

přídavných dat obsahuje pozici pro každý výskyt termu, což umožňuje využití proximitních operátorů při dotazování. Při webovém kontextu je možné zaznamenávat informace o tom zda term byl například asociován s hypertextovým odkazem, což může být využito při vyhodnocováni dotazu. [2]

U invertovaného indexu je vícenásobný výskyt termu ve stejném dokumentu sloučen do jednoho. Dále pak bývá seznam jednotlivých termů setříděn abecedně a jednotlivé seznamy dokumentů jsou poté setříděny podle identifikátoru dokumentu.

Co se týče paměti kterou invertovaný index zabírá na disku, tak velikost výsledného indexu se liší podle toho co je uloženo jako obsah přídavných dat pro každý prvek v kolekci seznamu dokumentů. Pokud se jako obsah přídavných dat ukládá jenom frekvence termu nebo je obsah prázdný, tak dobře optimalizovaný index může být jenom desetinou velikosti původních dat. Pokud se jako přídavná data ukládají i další věci jako poziční informace, tak pak může být výsledný index několikanásobně větší.

Z důvodu zvýšení výkonu bývá u invertovaného indexu dost často využito přístupu kdy seznam jednotlivých termů bývá uložen ve vnitřní paměti jelikož jeho velikost nebývá moc vysoká a seznamy dokumentů pro jednotlivé termy bývají uloženy na disku.

Při tvorbě výsledného invertovaného indexu se postupuje podle následujících kroků.

1. Shromáždění dokumentů, které mají být indexovány
2. Obsah dokumentu rozdělen na seznam slov
3. Seznam slov zpracován, čehož výsledkem je seznam termů pro daný dokument
4. Zaindexovat termy pro jednotlivé dokumenty, tím vytvořit výsledný index

2.1.2 Dotazování

Další podstatnou věcí, která ovlivňuje kvalitu booleovského modelu jsou možnosti pro kladení dotazů. Základní a nejjednodušších možností je použití termů a logických spojek sloužících k zadávání vztahů mezi jednotlivými termy. Výsledkem jsou vytvářeny logické výrazy, které jsou vyhodnocovány na základě Boolovy algebry. Přehled používaných logických spojek je zobrazen v Tabulce 1. [1, 3]

X AND Y	Výběr dokumentů, obsahujících jak term X, tak term Y.
X OR Y	Výběr dokumentů, obsahujících buď term X, nebo term Y, nebo oba termy současně.
X XOR Y	Výběr dokumentů, obsahujících buď term X, nebo term Y, ale ne oba současně.
NOT Y	Výběr dokumentů, neobsahujících term Y.

Tabulka 1: Logické operátory

Při vyhodnocování dotazu sestaveného z těchto logických spojek nad invertovaným indexem se postupuje podle těchto kroků. Postup je demonstrován nad následujícím jednoduchým konjunktivním dotazem.

Brutus AND Calspurnia

1. Vyhledat term Brutus v seznamu termů
2. Získat jeho seznam dokumentů
3. Vyhledat term Calpurnia v seznamu termů
4. Získat jeho seznam dokumentů
5. Udělat průnik těchto dvou seznamů

Jak je vidět, tak kritickou operací pro výkon je průnik u logické operace AND, proto průnik musí být proveden co nejefektivněji a nejrychleji. Stejné problémy jsou spojeny i s ostatními operátory, jako sjednocení seznamů u operace OR apod.

Jelikož použití pouze logických spojek pro vytváření dotazů by nebylo dostačující, tak bylo nutné rozšířit možnosti sestavování dotazů o další možnosti. Jednou z možností jak daný dotaz upřesnit je použití tzv. proximitních operátorů, pomocí kterých lze stanovovat vzdálenost a posloupnost mezi vyhledávanými termy v textu. Přehled používaných proximitních operátorů je znázorněn v Tabulce 2. [1]

X adj Y	(adjacent) Výběr dokumentů, ve kterých se vyskytuje term X následovaný termem Y.
X (n)words Y	Výběr dokumentů, ve kterých se vyskytuje term X následovaný termem Y nejdále ve vzdálenosti n slov.
X sentence Y	Výběr dokumentů, ve kterých se vyskytuje term X a Y ve stejné větě.
X paragraph Y	Výběr dokumentů, ve kterých se vyskytuje term X a Y ve stejném odstavci.

Tabulka 2: Proximitní operátory

Další možností pro rozšíření dotazu je použití symbolů, které slouží pro definici obecnějších regulárních výrazů. Patří mezi ně znaky jejichž význam je popsán v Tabulce 3. [1]

Znak	Význam
.	Tečka odpovídá libovolnému znaku.
*	Znak následovaný hvězdičkou odpovídá libovolnému počtu výskytů (včetně nulového) tohoto znaku. Např. xy* odpovídá x, xy, xyy atd.
+	Znak následovaný plus odpovídá libovolnému počtu výskytů (kromě prázdného) tohoto znaku. Např. xy+ odpovídá xy, xyy, xyyy atd.
[ ]	Znaky v hranatých závorkách odpovídají libovolnému jednomu znaku, který je v závorkách uveden, ale ne jinému. Např. $[xyz]$ odpovída x, y nebo z.
[^]	Stříška na začátku řetězce v závorce znamená negaci. Např. [^xyz] odpovídá libovolnému znaku kromě x, y nebo z.
[−]	Pomlčka mezi znaky v závorkách označuje rozsah znaků. Např. $[a - x]$ odpovídá libovolnému znaku od a do x.

Tabulka 3: Konstrukce regulárních výrazů

Další možností je zadání slovního kmene pomocí lemmatizátoru, což je algoritmus, který pro zadané slovo nalezne jeho základní tvar. V indexu je poté místo všech tvarů termu „počítač“ („počítače“, „počítači“, „počítačům“ atd.) uložen pouze základní tvar termu, což je „počítač“. Stejně tak termy v dotazu se před porovnáním převedou na základní tvar. Použití lemmatizátoru výrazně zvyšuje kvalitu informačního systému. Implementace lemmatizátoru je však značně složitá.

Moderní DIS používají pro zvýšení informativnosti indexy se složitější vnitřní strukturou - tzv. tezaury. Tezaury umožňují provázanost termů mezi sebou vztahem synonym a nebo vztahem nadtermů a podtermů. To ve výsledku umožňuje snadnější vytváření dotazů pro uživatele, jelikož už nemusí přemýšlet o všech možných vztazích mezi termy, které v dotazu zadává. Pokud například v dotazu uvede, že potřebuje dokumenty obsahující term „operační systém“, pak díky tezauru může DIS do výsledku zahrnout i ty dokumenty, které sice uvedený term neobsahují, ale obsahují podtermy „Windows“ nebo „MacOS“. Pokud tedy DIS umožňuje definovat tezaury, pak je způsob kladení dotazu dále rozšířen o možnost ptát se na jednotlivé úrovně provázanosti termů. Tezaurus je implementovaný například v produktu Oracle SQL*Text retrieval, který rozšiřuje jazyk SQL o speciální operátory, kterými určujeme nejen term, ale i to, zda ve výsledku mají být dokumenty obsahující jeho nadřazené či podřazené termy. Díky tomu se můžeme na Oracle dotazovat na množinu úplných textů. Tabulka 4 obsahuje tyto operátory a jejich význam. Uvedené operátory vycházejí ze standardu ISO-2788. [4]

Objevují se i možnosti klást dotazy přímo v přirozeném jazyce, např. „Nalezni všechny dokumenty o databázích a mineralogii“. Takové dotazy bývají transformovány do konjunktivního dotazu. Pokusy o složitější typy dotazů v přirozeném jazyce ovšem vyžadují hlubší analýzu, obvykle i použití lingvistických metod. [1]

Znak	Význam	Popis
NT (A)	NARROWED TERM	Užší term k termu A
BT (A)	BROADER TERM	Širší term k termu A
TT (A)	TOP TERM	Nejširší term k term A
RT (A)	RELATED TERM	Příbuzné termy k termu A
PT (A)	PREFERRED TERM	Preferovaný term k termu A
SYN (A)	SYNONYMUM	Synonyma k termu A

Tabulka 4: Operátory Oracle SQL*Text

2.1.3 Výhody a nevýhody

Booleovský model je snadno implementovatelný a je velmi efektivní z pohledu času potřebného k vyhodnocení dotazu. Pokud je dotaz správně formulován, jsou systémy na něm založené schopné poskytnout poměrně přesné výsledky. Navíc booleovský model se stal základem dalších modelů.

Nevýhodou booleovského modelu je, že i přes uvedené možnosti pro sestavení dotazu je jeho vyjadřovací síla stále omezená. Problémem je, že i když jakákoliv množina dokumentů, která je popsatelná pomocí termů, může být vhodným dotazem vybrána, tak pro uživatele není snadné takové dotazy sestavovat.

Dalším omezením booleovského modelu je, že při vyhodnocování dotazů je logický výraz pravdivý nebo nepravdivý a z tohoto důvodu je dokument buď vybrán nebo ne, problém této interpretace je ukázán na následujícím dotazu.

A AND B AND C AND D AND E

U uvedeného dotazu budou vybrány pouze ty dokumenty, které obsahují všechny termy, ale dá se očekávat, že pro uživatele by mohlo být užitečné, kdyby byly vybrány i ty dokumenty, které například neobsahují pouze jeden z termů.

U booleovského modelu dále není umožněno ohodnocení významnosti vybraných dokumentů, které by uživateli umožňovali prohlídnout si nejdříve ty dokumenty, které byly pro dotaz vybrány jako nejvíce relevantní. Ukázka tohoto problému je znázorněna na následujícím dotazu.

A OR B OR C OR D OR E

Po vyhodnocení dotazu jsou vybrané dokumenty obsahující pouze jeden z uvedených termů chápány stejně významně jako vybrané dokumenty, které obsahují všechny uvedené termy.

U booleovského modelu není ani umožněno nějakým způsobem ohodnocovat významnost nebo přiřazovat váhu jednotlivých termů v dotazu, což by například umožňovalo indikovat, že absence jednoho termu je významnější než jiného termu.

Dalším omezením booleovského modelu je, že neumožňuje uživateli jakýmkoliv způsobem seřadit výsledným seznam vybraných dokumentů. Existuje zde pouze možnost

řazení podle časové řady apod. Navíc není umožněno řízení velikosti výstupu, proto mohou nastat případy, kdy je na výstupu velké množství dokumentů a nebo naopak je výstup prázdný. Dále není umožněna realizace zpětné vazby, což by umožnilo uživateli označit z výsledných dokumentů ty, které pro něj byly relevantní a automaticky tak modifikovat dotaz.

Zásadní omezení booleovského modelu bylo ukázáno na pokusu Blaira a Marona z roku 1985. V pokusu, který zahrnoval 40 000 právnických textů, bylo studováno několik desítek dotazů z hlediska hodnot koeficientů R a P. Koeficient přesnosti P (Precision) je určen jako poměr počtu vybraných relevantních dokumentů k počtu všech vybraných dokumentů. Koeficient přesnosti P tedy určuje, jak dobře systém vyhledá jen relevantní dokumenty. Koeficient úplnosti R (Recall) je určen jako poměr počtu vybraných relevantních dokumentů k počtu všech relevantních dokumentů. Koeficient úplnosti R měří, jak dobře systém vyhledá všechny relevantní dokumenty. Výsledkem pokusu byly průměrné hodnoty pro P rovny 80 %, kdežto pro R pouze 20 %, což je velmi žalostné, zvláště jednalo-li se o právní texty. Navíc vysoká hodnota P současně vedla k nízké hodnotě R. [1]

2.2 Signaturní soubory

Signatury byly intenzivně studovány v 2. polovině 80. let. Jejich hlavní výhodou je jejich prostorová nenáročnost. Na rozdíl od invertovaných souborů, které můžou bez komprese zabrat i několikanásobně více prostoru než původní dokumenty, zaberou pouze zlomek tohoto místa. Modely používající signatury se používají pro vyhledávání textů ve velmi rozsáhlých databázích a tvoří alternativu k indexaci. Jako příklad používající tento model lze například uvést databáze novinových článků, chemické databáze, programy pro kontrolu pravopisu, při zpracování obrazu nebo v databázích pro vyhledávání v řetězcích DNA. [1]

Signatury jsou komprimované verze reálných dat a při jejich tvorbě se vytváří jistá kódová slova nazývané bitové vektory. Jinak řečeno, tvorba signaturních souborů je založena na generování bitových vektorů popisujících jednotlivé dokumenty. Tyto vektory jsou sestaveny pomocí některé metody pro vytváření signatur z bitových vektorů popisujících jednotlivé termy v dokumentu. Takto vytvořené signatury lze použít jako datovou strukturu pro vyhledávání dokumentů. Jednotlivé signatury jsou nějakou metodou organizovány do signaturního souboru, nad kterým je prováděno dotazování kvůli minimalizaci počtu přístupů na disk. Jelikož signatury jsou komprimovaná data, tak při jejich vytváření dochází ke ztrátě informací, což může mít za následek, že při vyhledávání dojde k chybnému výběru, což je jev, kdy je dokument vybrán jako relevantní, i když nevyhovuje dotazu.

Vyhledávání pomocí signaturních souborů může probíhat ve dvou krocích. V prvním kroku se signatury dokumentů porovnávají se signaturou dotazu. Výsledkem je množina obsahující dokumenty, které se mohou kvalifikovat pro daný dotaz. Signaturní soubor zde slouží jako filtrovací mechanismus, jelikož eliminuje dokumenty, které neobsahují termy obsažené v dotazu. Jelikož v prvním kroku byl výsledkem seznam dokumentů vyhovujících dotazu zatížený o chybně vybrané dokumenty, jsou pak v druhém kroku tyto

vybrané dokumenty porovnány textově s položeným dotazem, čímž dojde k odstranění chybně vybraných dokumentů z prvního kroku. V praxi jsou však uživateli předloženy dokumenty obsahující i chybné výběry. Problémem však je, že jako výsledek chybného výběru mohou být i dokumenty, které jsou zcela mimo dotaz. [1, 7]

Symbol	Definice
d	Velikost signatury v bitech
w	Váha signatury
m	Počet dokumentů v kolekci
n	Počet termů v dokumentu
S_q	Signatura dotazu
S_d	Signatura dokumentu
Q	Počet termů v dotazu
P_d	Pravděpodobnost chybného výběru

Tabulka 5: Parametry

2.2.1 Metody vytváření signatur

Metody vytváření signatur se snaží o dosáhnutí co nejlepšího P_d, což je pravděpodobnost jevu, že dokument je vybrán jako odpovídající dotazu, ale přitom jde o chybný výběr. P_d je podíl chybně vybraných dokumentů všemi dokumenty nevyhovujících dotazu.

Při vytváření signatur se používá hašovací funkce, která převádí jednotlivé termy z dokumentu na d-bitové vektory předem určené velikosti, kdy je nastaveno m bitů na jedničku. Kvalitní hašovací funkce by měla být jednoduchá, rychlá a mít co nejmenší počet zobrazení různých termů na stejný bitový vektor a pozice nastavovaných bitů v signatuře by měly být rozkládány rovnoměrně.

Rozdílem u jednotlivých metod je způsob vytváření bitových vektorů pro termy, nazývaných binární reprezentace termů nebo také signatury termů, které jsou poté spojeny do unikátního bitového vektoru, který je poté signaturou celého dokumentu.

2.2.1.1 Slovní signatury V této metodě je každý term dokumentu hašován do bitového vektoru o délce d. Takto vytvářené signatury termů, nazývané slovní signatury, jsou poté spojeny do signatury dokumentu. Při vyhledávání se sekvenčně prochází signatura dokumentu. Aby byl dokument vybrán jako relevantní musí signatura obsahovat všechny slovní signatury, které obsahuje signatura dotazu. Příklad takto vytvářených signatur je znázorněn v Tabulce 6 ($n = 3$, $w = 2$, $d = 12$). [7, 6]

Slovní signatury na rozdíl od jiných metod vytvářející signatury stejné velikosti neposkytují příliš kvalitní P_d, ale mají přiměřený čas vyhledávání a pracují dobře pro dokumenty s proměnným počtem termů. Při použití této metody musí být signatury organizovány jako sekvenční soubor. [5]

pes	1001
stan	0011
kost	0101
výsledek	100100110101

Tabulka 6: Spojování slovních signatur

2.2.1.2 Vrstvené kódování Při vrstveném kódování se vytváří signatura dokumentu tak, že jsou všechny signatury termů na sebe vrstveny pomocí logické funkce OR. Jednotlivé signatury termů se vytváří tak, že v signatuře o délce d je pomocí hašovací funkce vybráno m pozic, které jsou nastaveny na jedničku, což určuje váhu signatury, bity na ostatních pozicích zůstanou na nule. Příklad vytváření signatur touto metodou je znázorněn v Tabulce 7 ($n = 3$, $w = 3$, $d = 12$). [7, 6]

strom	000100101001
auto	010010000011
slovo	110000000101
výsledek	110110101111

Tabulka 7: Vrstvené kódování

Při vyhledávání se pro term nebo skupinu vyhledávaných termů vytvoří signatura dotazu a to stejným způsobem jako při vytváření signatury dokumentu. Dokument je vybrán za relevantní právě tehdy, když je splněna podmínka S_d AND $S_q = S_q$, což je splněno tehdy, když všechny pozice nastavené na jedničku v signatuře dotazu jsou nastaveny na jedničku i v signatuře dokumentu. Tato podmínka se také dá vyjádřit jako S_q AND NOT $S_d = 0$. Příklad vyhodnocení dotazu z Tabulky 3 pro tuto metodu je zachycen v Tabulce 8 ($S_q = 110110101111$, $Q = 1$). [1, 5]

strom	000100101001	vybraný dokument
autobus	010011000011	nevybraný dokument
disk	010100001001	chybný výběr

Tabulka 8: Vyhodnocení dotazu pro vrstvené kódování

Problémem u vrstveného kódování může být, že pokud dokument obsahuje určitý počet termů a signatura není dost dlouhá, tak výsledkem může být velmi velká váha signatury, až může dojít k případu, kdy signatura bude obsahovat samé jedničky, pak ale tato signatura dokumentu bude vyhovovat každému dotazu. Řešením může být zvětšení délky signatury, což ale zvětší výslednou velikost celého signaturního souboru. Další možností je rozdělit dokument do bloků. Jednou z možností je použít metodu bloků pevné délky, kdy se text dělí na bloky stejné délky a nebo použít metodu bloků pevné

váhy, kdy je dokument rozdělen na bloky s přibližně stejnou váhou, kde váha může být počet termů nebo je blok ukončen pokud obsahuje přibližně polovinu jedniček a nul. [1, 7]

Výhodou vrstveného kódováni je krátký čas, který je potřebný k porovnáni dvou signatur. Pokud se počet termů v dokumentech příliš neliší, tak má velmi přijatelnou P_d. Kvůli jednoduchosti a pevné struktuře, která nabízí mnoho možností uložení na disku je tato metoda nejpoužívanější v praxi. [7, 5]

2.2.1.3 Metody založené na kompresi Metody založené na kompresi jsou podobné vrstvenému kódování. Rozdílem je však, že při vytváření signatur termů je počet nastavovaných bitů na jedničku typicky roven 1, což ve výsledku vede k velmi řídkému bitovému vektoru, a proto jsou data komprimovány.

2.2.1.3.1 Bitově bloková komprese Jednou z metod je bitově bloková komprese, která řídký vektor rozděluje na skupiny po sobě jdoucích bitů, tyto skupiny jsou poté samostatně komprimovány a to tak, že každá skupina je rozdělena na 3 části. První část obsahuje pouze jeden bit, který udává jestli se v dané skupině nachází nějaký jedničkový bit. Pokud je bit v první části nastaven na jedničku, což znamená, že se ve skupině nachází alespoň jeden jedničkový bit, pak jsou použity i části 2 a 3. Druhá část udává celkový počet jedničkových bitů x v bloku a to sekvencí $(x-1)$ po sobě jdoucích jedniček ukončených nulou. Třetí část udává offset jedniček od začátku skupiny. Příklad řídkého vektoru a jeho komprimovaných skupin je znázorněn v Tabulce 9 ($S_d = 1001000000100000$). [7, 5]

	Skupina 1 1001	Skupina 2 0000	Skupina 3 0010	Skupina 4 0000
část I	1	0	1	0
část II	10		0	
část III	0011		10	

Tabulka 9: Bitově bloková komprese

Pro vytvoření výsledné signatury dokumentu z komprimovaných skupin lze použít způsob, kdy jsou jednotlivé skupiny spojeny tak, že jednotlivé části jsou uloženy odděleně (BCG). Další možností je, že jsou ukládány jednotlivé skupiny tak, že první části jsou uloženy bezprostředně za sebou následovány druhými částmi a ty pak třetími (BCP). Oba způsoby jsou znázorněny v Tabulce 10. [7]

BCG signatura	1100011	0	1010	0
BCP signatura	1010	100	001110	

Tabulka 10: Spojení skupin signatur

Nevýhodou bitově blokové komprese je, že je pomalejší při porovnávání, protože zde přibývají bitové operace a porovnáváni pro určení zda blok obsahuje části dvě a tři. Výhodou je však velmi kvalitní P_d, které je stejné i pro dokumenty s proměnlivým počtem termů. [5]

2.2.1.3.2 Kódování vzdáleností jedniček Další kompresní metodou je kódování vzdáleností jedniček. Tato metoda je velmi podobná bitově-blokové kompresi, rozdílem je, že metoda kódování vzdáleností jedniček ukládá vzdálenosti mezi sousedícími jedničkovými bity v řídkém bitovém vektoru na rozdíl od ukládání offsetu. Tato metoda vykazuje nejlepší kompresní poměr, což znamená, že vytváří nejmenší signatury. Také vykazuje velmi kvalitní P_d. Nevýhodou je však, že při porovnávání dvou signatur je zapotřebí signatury rozkódovat, což vede k velmi pomalému porovnání. [7, 5]

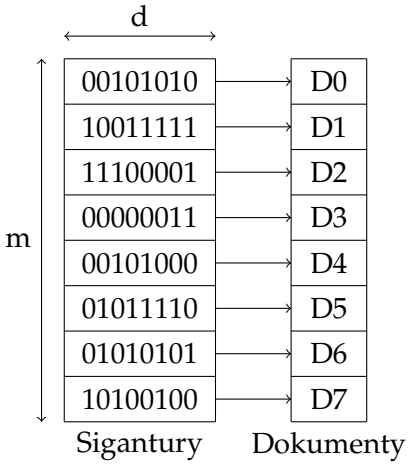
2.2.2 Organizace signaturních souborů

Metody pro vytvářeni signatur se snažili o zlepšení výkonu tím, že se snažili dosáhnout co nejlepšího P_d. Způsoby organizace signaturních souborů se snaží o snížení počtů $I/0$ operací a to hlavně snížením počtu fyzických stránek, které musí být zpřístupněny k vyhodnocení dotazu. Většina technik pro organizaci signaturních souborů je postavena na předpokladu, že je použito vrstvené kódování pro tvorbu signatur a to z důvodu, že mají konstantní délku a lze je chápat jako matici o m řádcích a d sloupcích.

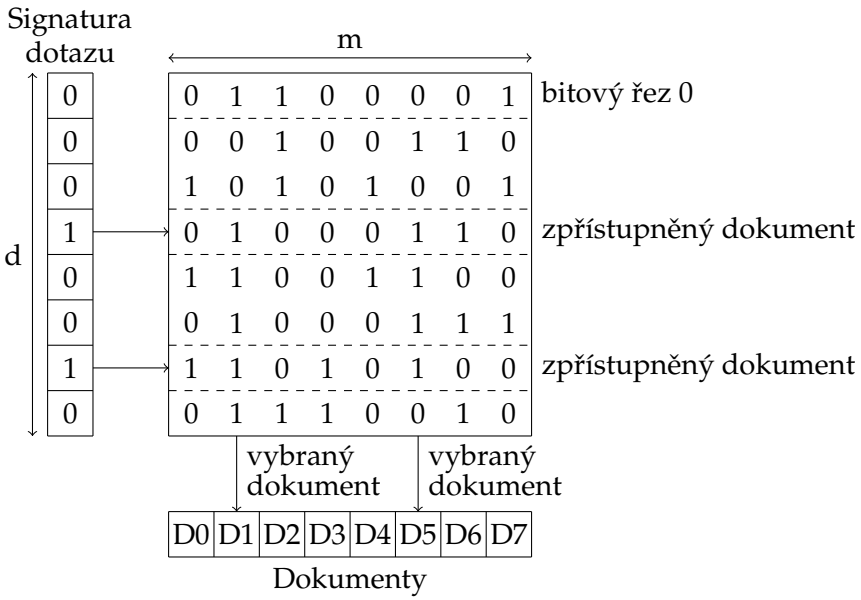
2.2.2.1 Sekvenční soubor Nejjednodušší možností je organizace do sekvenčního souboru. U tohoto způsobu je nová signatura vkládána na konec souboru jako nový řádek, což vede k tomu, že je velikost signaturního souboru závislá na počtu dokumentů. U vyhodnocování se signatura dotazu musí porovnávat s každou signaturou dokumentu v sekvenčním souboru, což vede k tomu, že vyhledávací čas závisí na velikosti souboru. I když jsou signatury menší než původní data je vyhledávání často časově neúnosné. Způsob organizace je naznačen na Obrázku 2. [7, 5]

2.2.2.2 Vertikální rozklad Tento způsob rozložení signaturního souboru se nazývá bitové řezy. Vylepšení oproti sekvenčnímu souboru spočívá v tom, že signatury nejsou ukládány po řádcích, ale po sloupcích, tím vznikne soubor d bitových vektorů (řezů) velikosti m. Tento způsob organizace dat způsobí, že při vyhodnocování dotazu se přistoupí pouze k těm sloupcům, ve kterých má signatura dotazu jedničku. Přistoupí se tedy pouze k $w(S_q)$ sloupcům na rozdíl od sekvenčního souboru, kde se přistupuje ke všem sloupcům. Pro určení signatur, které jsou relevantní pro dotaz se na obdržené bitové vektory provede operace AND. Výsledkem bude bitový vektor, kde každá jednička indikuje pozici relevantního dokumentu. Na Obrázku 3 je naznačen způsob této organizace. [1, 7]

U této metody je vyhledávací čas závislý na váze dotazu, což je na počtu zpřístupněných sloupců. Proto je tato metoda nejlepší pro dotazy s velmi nízkou váhou. Údržba u této metody je velmi časově náročná a z tohoto důvodu je metoda vhodná pouze pro stabilní soubory. [5]



Obrázek 2: Sekvenční soubor



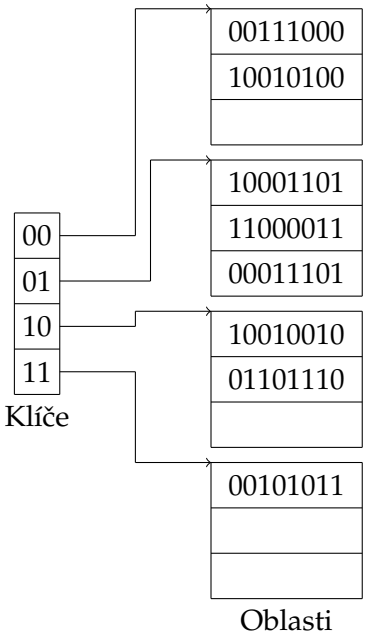
Obrázek 3: Bitové řezy

2.2.2.3 Horizontální rozklad U horizontálního rozkladu dochází ke snížení vyhledávacího prostoru při provádění dotazu. Obecně tyto metody definují klíč pro každou signaturu a uchovávají signatury se stejným klíčem ve stejné unikátní oblasti.

2.2.2.3.1 Pevný prefix Snížení vyhledávacího prostoru je dosaženo tak, že se vezme prvních x bitů tzv. klíčů ze signatury dokumentu K_d, které jsou uloženy na vnitřní paměti a použity jako ukazatele. Signatury se stejným klíčem jsou uloženy ve stejné oblasti

(stránce) na vnější paměti, na které ukazují jednotlivé ukazatele. Při vyhodnocování dotazu se poté přistoupí jen k těm oblastem, které zahrnují klíč signatury dotazu K_q, tedy platí K_d AND $K_q = K_q$. [1, 7]

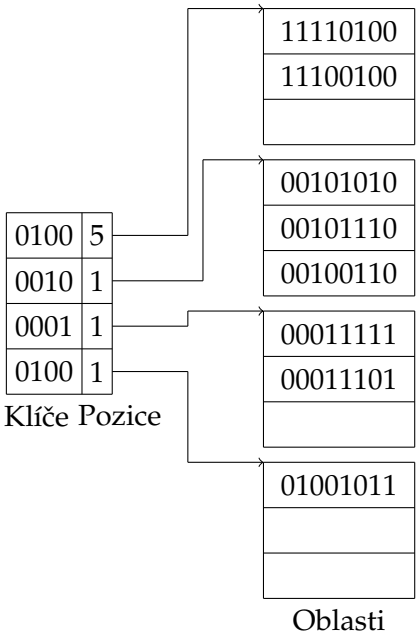
Hlavní výhodou této organizace je její jednoduchost, naopak problémem je, že pro oblasti, jejichž klíč tvoří samé jedničky bude klíč dotazu vždy vyhovovat, a proto se k těmto oblastem bude přistupovat pro každý dotaz. Organizace signaturních souborů pomocí pevného prefixu je naznačena na Obrázku 4.



Obrázek 4: Pevný prefix

2.2.2.3.2 Plovoucí klíč Organizace pomocí plovoucího klíče je velmi podobná organizaci, kdy je použit pevný prefix, rozdíl je však při vytváření klíče. U této metody se klíč vytváří tak, že se prochází všechny nepřekrývající x-bitové podřetězce a jako klíč je vybrán ten, který obsahuje nejmenší počet jedniček. Kvůli tomuto způsobu vytváření klíčů je potřeba uchovávat kromě klíčů samotných i jejich pozici, což je pozice prvního bitu podřetězce v signatuře. Vyhodnocování dotazu je stejné jako u organizace s pevným prefixem. [1, 7]

Na rozdíl od organizace s pevným prefixem je dosaženo snížení počtů signatur, jejichž klíč by obsahoval samé jedničky, čímž dojde ke sníženi počtu přistoupených oblastí. Problémem u této organizace je však, že velikosti oblastí a přistup k nim není rovnoměrný. Způsob organizace signaturních souborů pomocí plovoucího klíče je naznačen na Obrázku 5.

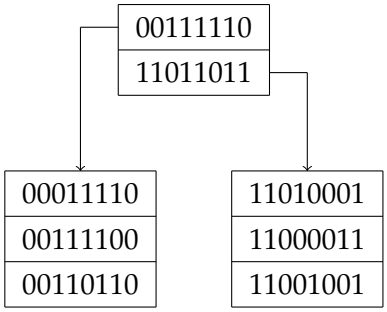


Obrázek 5: Plovoucí klíč

2.2.2.4 Stromová organizace Další možností jak jednotlivé signatury ukládat je organizace do stromů. U této organizace se nejčastěji používá struktura podobná B-stromu, jelikož tato struktura je vhodná pro použití, kdy celá struktura není uložena ve vnitřní paměti, ale na disku a tím pádem se snažíme a co nejmenší počet přístupů do této paměti.

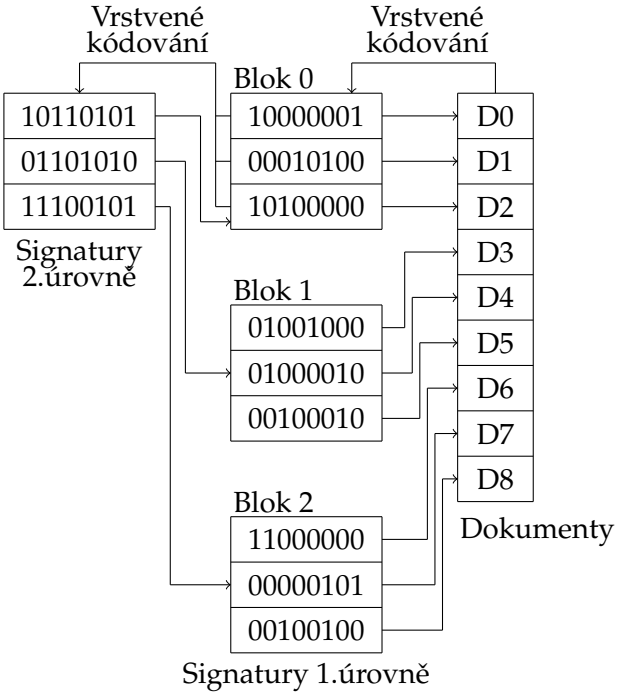
2.2.2.4.1 Indexované deskriptory V této metodě je princip takový, že signatury vyšších řádů vznikají vrstvením signatur z nižších řádů. Vyhledávací proces spočívá v rekurzivním prohledávání stromu se signaturou dotazu jako filtru, který odřezává cesty, které nevyhovují dotazu. Ve statickém prostředí může být dosaženo dobrého výkonu. Velikost signatur by však měla být zvolena dostatečně velká pro stromy s mnoha úrovněmi. [1]

2.2.2.4.2 S-stromy Jedná se o další stromovou organizaci signaturních souborů. S-stromy jsou velmi podobné indexovaným deskriptorům. Hlavní myšlenkou této organizace je seskupení signatur, které se liší ve velmi malém počtu bitů, tj. mají malou Hammingovu vzdálenost a postavit nad nimi strukturu podobnou B-stromu, která poskytuje rychlý přístup k jednotlivým uzlům. Z důvodu, že se jedná o strukturu podobnou B-stromu je S-strom vyvážený. Výkon filtrace u této struktury závisí na váze signatury dotazu, proto tato organizace pracuje dobře pro signatury s velkou váhou. S-stromy ovšem nejsou vhodné pro velké soubory, protože vrstvené kódování ze signatur ve stránkách může vytvořit signatury s velkou váhou, které způsobují velkou chybovost. Na Obrázku 6 je naznačena organizace jako S-strom. [5]



Obrázek 6: S-strom

2.2.2.5 Víceúrovňový soubor signatur Jedním ze způsobů víceúrovňové organizace signatur je dvouúrovňové vrstvené kódování. U této metody jsou dokumenty rozděleny do bloků a jsou vytvořeny dvě úrovně signatur. V první úrovni jsou pomocí vrstveného kódování vytvořeny signatury z termů v dokumentu. Ve druhé úrovni jsou opět pomocí vrstveného kódování vytvářeny signatury pro celé bloky a to ze signatur dokumentů obsažených v daném bloku. Jelikož signatury druhé úrovně jsou vytvářeny vrstvením kódováním signatur dokumentů z první úrovně, tak mají větší váhu než signatury první úrovně. Způsob organizace dvouúrovňovým vrstveným kódováním je naznačen na Obrázku 7. [1]



Obrázek 7: Dvouúrovňové vrstvené kódování

Při vyhodnocování dotazu se signatura dotazu nejprve porovná se signaturami bloků. Poté se signatura dotazu porovnává s dokumenty, které jsou obsaženy v kvalifikovaných blocích. Tato metoda je efektivní pro velké soubory a vyhodnocování signatur dotazů s nízkou váhou.

Ve výsledku se dá tato metoda zobecnit a místo dvou používaných úrovní, může těchto úrovní být vytvořeno několik a rozdílem může být i to, že pro každou úroveň je použita jiná mapovací metoda.

3 Popis datasetu

Pro srovnání obou modelů je zapotřebí mít testovací kolekcí dokumentů. U obou aplikací byla použita kolekce 2005 TREC Public Spam Corpus od NIST[1]. Tato kolekce obsahuje celkový počet 92189 dokumentů, kde dokumentem je emailová zpráva, která je spam. Tyto zprávy byly zvoleny z důvodu, že jsou krátké, což dokazuje i to, že průměrný počet termů pro indexaci je u jednotlivých zpráv 257.

Dříve než jednotlivé dokumenty začneme indexovat a poté vkládat do invertovaného indexu v případě booleovského modelu nebo do S-stromu v případě signaturních souborů, tak je zapotřebí si data před vložením předpřipravit. Po předzpracování dat je jejich obsah daleko menší, což má za následek rychlejší zpracování dat a přesnější výsledky.

V prvním kroku jsou všechny dokumenty postupně načteny. Jelikož se jedná o emailové zprávy, tak je zkontrolováno jestli se jedná o validní MIME zprávu. Pokud se nejedná o validní MIME zprávu, tak není indexována a to z toho důvodu, že při použití obou modelů jako detekce spamu je zpráva, která není validní zprávou v tomto formátu kandidátem na spam. Tímto zpracováním dojde ke snížení celkového počtu zpráv na 80000, se kterým se dále pracuje. Poté co je zkontrolováno, že se jedná o validní MIME zprávu je ze zprávy vytažen její obsah, který je poté dále zpracován.

V dalším kroku se tedy provádí zpracování obsahu zprávy. Při tomto zpracování je text rozdělen na jednotlivá slova, což způsobí, že jsou odstraněny všechny interpunkční znaky a také všechny bílé znaky. Poté jsou u těchto slov všechna velká písmena převedena na malé písmena, aby se v indexu neuchovávaly termy se stejným významem. Tyto slova jsou postavena proti seznamu nevýznamových slov, což znamená, že slova v textu jako spojky, předložky apod. jsou odstraněny. Výsledkem zpracování obsahu textu je tedy seznam termů pro daný dokument.

Poté co již máme vytvořený seznam termů, který popisuje daný dokument, pak mohou být jednotlivé termy vloženy do invertovaného indexu v případě booleovského modelu. Pokud se jedná o signaturní soubory, tak jednotlivé termy jsou pomocí vrstveného kódování spojeny do výsledné signatury dokumentu, která je poté vložena do S-stromu.

U signaturních souborů při vytváření signatury dokumentu je nutné mít zapotřebí určité nastavení. První podstatnou věcí, která je zapotřebí určit je velikost signatury. Druhou věcí, která je potřeba nastavit je na kolik bitů se jednotlivé termy budou hašovat v případě vrstveného kódování. Zvolené nastavení má ve výsledku vliv na celkovém počtu chybně vybraných dokumentů, na výsledné velikosti kterou bude S-strom na disku zabírat a také jaká doba bude zapotřebí pro vyhodnocení dotazu. U S-stromu je také potřeba určit velikost uzlů. Dobře zvolená velikost způsobí, že se bude přistupovat k co nejefektivnějšímu počtu uzlů, tedy co nejmenšímu počtu dat.

[1] http://trec.nist.gov/data/spam.html

4 Porovnání modelů

V následující části textu je provedeno porovnání booleovského modelu a signaturních souborů s daným nastavením. Porovnávalo se vytváření indexu pro jednotlivé modely a to z hlediska času, který pro své vytváření potřebují a paměti jakou index na disku zabírá. Dále je provedeno porovnání výsledků pro různé dotazy. Také je provedeno porovnání času potřebného pro vyhodnocení dotazu a jaký vliv na čas má počet termů v dotazu u jednotlivých modelů. U signaturních souborů je dále ukázána jak efektivní je organizace signatur do S-stromu a to tak, že je měřen počet přistoupených uzlů. Na závěr jsou oba modely porovnány na konkrétní aplikaci a to, že jsou použity jako detekce spamu.

Pro porovnání bylo u signaturních souborů zvoleno nastavení, které vykazovalo nejlepší výsledky. Toho bylo dosaženo tehdy, když délka signatury byla nastavena na 1000 bitů, dále pak jako metoda pro vytváření signatur bylo zvoleno vrstvené kódování, kde jednotlivé termy byly hašovány do 3 bitů v signatuře. Signatury pak byly organizovány jako S-strom s velikostí uzlů nastavenou na 30.

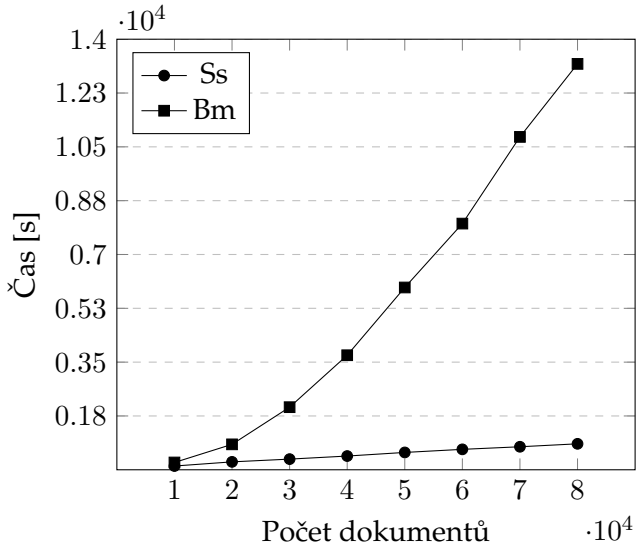
4.1 Tvorba indexu

První porovnání obou modelů bylo provedeno na porovnání času potřebného pro vytvoření indexu a paměti na disku, kterou tento index zabírá. Výsledná data pro oba modely jsou znázorněna v Tabulce 11.

Model	Počet dokumentů	Čas$[s]$	Paměť$[kB]$		
			index	offset	názvy
BM	10000	237,3	24734257	1728797	190000
SS		121,8	1774128		
BM	20000	826,9	44017438	2972195	380000
SS		257,3	3559253		
BM	30000	2035,4	74439150	4968381	570000
SS		346,9	5344028		
BM	40000	3724,2	103745301	7906901	760000
SS		446,4	7129153		
BM	50000	5927,2	127287973	10321995	950000
SS		563,8	8914108		
BM	60000	8004,7	149021503	13898879	1140000
SS		664,3	10698878		
BM	70000	10823,3	170168237	15200730	1330000
SS		748,3	12504147		
BM	80000	13197,2	188314947	16286348	1515877
SS		844,1	14230131		

Tabulka 11: Vytváření indexu

Pokud se nejdříve podíváme na čas potřebný pro vytvoření indexu, tak již na první pohled je z výsledků patrné, že signaturní soubory potřebují pro svou indexaci daleko méně času. Srovnání časů pro oba modely je znázorněno na Obrázku 8. U signaturních souborů je vidět lineární průběh a se zvyšujícím se počtem dokumentů se čas zvyšuje konstantně. Zato u booleovského modelu se zvyšujícím se počtem dokumentů se čas zvyšuje daleko strměji, čehož důkazem je i to, že pro vytvoření indexu z 80000 dokumentů oproti 10000 dokumentů je čas u signaturních souborů 7 násobný, tak u booleovského modelu je tento čas 56 násobný. Navíc pokud srovnáme čas pro vytvoření indexu proti sobě, tak pro 10000 dokumentů je čas u booleovského modelu dvojnásobný oproti signaturním souborům a u 80000 dokumentů je rozdíl v časech už 16 násobný.

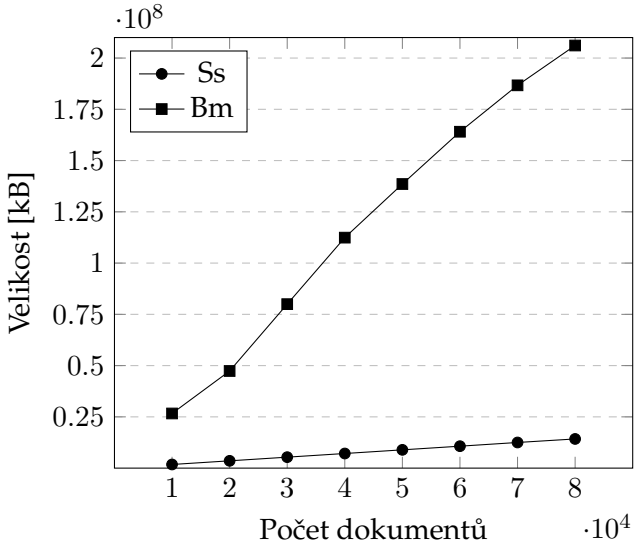


Obrázek 8: Čas potřebný pro vytváření indexu

Druhým měřeným parametrem u vytváření indexu byla paměť, kterou tento index zabírá na disku. U booleovského modelu je tento index složen ze 3 částí. První část tvoří index obsahující seznam ukazatelů na dokumenty pro jednotlivé termy, dále pak z offsetu, který obsahuje seznam všech termů a ukazatelů na seznam dokumentů pro jednotlivé termy v indexu. Tento offset je načítán do vnitřní paměti při vyhodnocování dotazů. Poslední částí je seznam, který obsahuje názvy všech dokumentů, pro rychlejší vyhodnocení logické operace NOT.

I u srovnání velikosti paměti, kterou index zabírá na disku, jsou na první pohled patrné lepší výsledky u signaturních souborů. Porovnání pro oba modely je znázorněno na Obrázku 9.

U obou modelů roste velikost indexu s narůstajícím počtem dokumentů lineárně. U booleovského modelu je narůst strmější, přesto však nárůst je dosti podobný, což dokazuje i to, že pro index vytvořený z 10000 dokumentů je u booleovského modelu index 15 krát větší a u indexu vytvořeného z 80000 dokumentů je nárůst přibližně také 15 násobný. Pokud porovnáme velikost indexu s původními daty, tak booleovský model do-



Obrázek 9: Velikost indexu na disku

sahuje přibližně 30 % velikosti oproti původním datům a signaturní soubory pak pouhé 2 %, zde je však nutné brát v úvahu, že pro indexaci se použije jen část z původních dat a navíc místo souborů samotných se ukládá ukazatel na ně.

Signaturní soubory dosáhly lepších výsledků jak v čase potřebném pro vytváření indexu, tak i ve velikosti paměti, kterou tento index zabírá na disku. Tyto výsledky se však daly očekávat, jelikož signaturní soubory jsou komprimované verze reálných dat, což má za následek, že budou prostorově méně náročné. Navíc menší velikost dat způsobí i rychlejší zpracování a proto byly signaturní soubory rychlejší při vytváření indexu. Dalším důvodem proč vytváření indexu u signaturních souborů bylo rychlejší, než u booleovského modelu je i to, že signaturní soubory jsou organizovány do S-stromu, tedy do struktury podobné B+ stromu a operace vložení zabere daleko méně času než vložení do lineární seznamu jak je tomu v případě invertovaného indexu u booleovského modelu.

4.2 Vyhodnocení dotazů

Další porovnání obou modelů bylo provedeno pro dotazování u jednotlivých modelů pro dotazy uvedeny v Tabulce 12.

Dotaz	Definice
Q1	east and power and web
Q2	east and power and web and site and information
Q3	east and power and web and site and information and western and price

Tabulka 12: Dotazy

Nad těmito dotazy bylo provedeno měření doby potřebné pro vyhodnocení jednotlivých dotazů. U booleovského modelu je doba potřebná pro vyhodnocení dotazu rozdělena do dvou časů. Prvním je doba potřebná pro načtení offsetu do vnitřní paměti a druhý čas uvádí samotné vyhodnocení dotazu. Dále pak pro tyto dotazy bylo měřeno kolik dokumentů bylo vybráno jako relevantní. Výsledná data jsou uvedena v Tabulce 13.

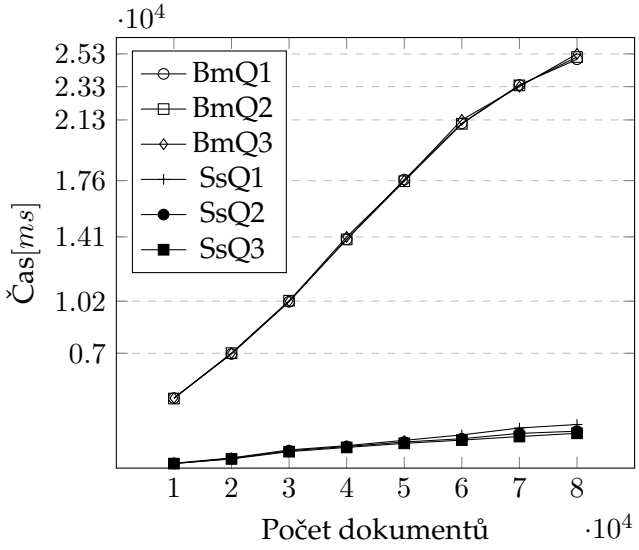
Počet dokumentů $[\cdot 10^3]$			10	20	30	40	50	60	70	80
Q1	BM	čas offsetu$[s]$	4,2	6,9	10,1	13,8	17,1	20,8	23,1	24,6
		čas$[ms]$	28,8	54,7	82,4	131,6	190,7	237,7	271,1	305,4
		relevantní	4	24	42	65	74	82	91	95
	SS	čas$[s]$	0,29	0,62	1,11	1,37	1,69	2,03	2,45	2,66
		relevantní	59	233	461	710	873	1015	1080	1101
Q2	BM	čas offsetu$[s]$	4,2	6,9	10,1	13,8	17,1	20,7	23,1	24,7
		čas$[ms]$	39,8	79,0	113,2	157,1	215,5	275,1	320,4	355,3
		relevantní	3	14	23	35	43	48	56	60
	SS	čas$[s]$	0,28	0,58	1,06	1,32	1,59	1,79	2,12	2,25
		relevantní	40	172	362	561	674	746	797	814
Q3	BM	čas offsetu$[s]$	4,2	6,9	10,1	13,9	17,0	20,9	22,9	24,9
		čas$[ms]$	45,2	86,4	115,6	165,0	233,3	302,0	343,2	384,8
		relevantní	2	10	10	16	21	22	28	31
	SS	čas$[s]$	0,27	0,56	1,01	1,26	1,51	1,71	1,92	2,12
		relevantní	33	134	305	492	593	646	691	708

Tabulka 13: Výsledky pro dotazy

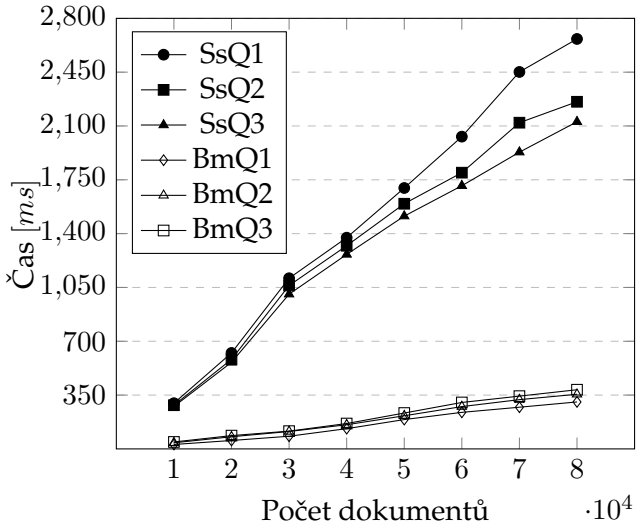
Z dat je patrné, že se časy pro dotazy v rámci stejného modelu příliš neliší. Pokud porovnáme časy potřebné pro vyhodnocení dotazu u jednotlivých modelů, tak je vidět, že signaturní soubory vyžadují daleko menší čas pro vyhodnocení dotazu a to hlavně díky době potřebné pro načtení offsetu, která je zhruba 10 násobně delší než doba potřebná pro vyhodnocení dotazu u signaturních souborů, což je i znázorněno na Obrázku 10.

Pokud je však u booleovského modelu již offset načtený ve vnitřní paměti, tak by pak booleovský model vykazoval lepší výsledky a to až 10 násobně, což je i znázorněných na Obrázku 11.

Na výsledcích pro položené dotazy je také vidět jedna z hlavních nevýhod signaturních souborů oproti booleovskému modelu a to, že signaturní soubory vybírají i ty dokumenty, které pro daný dotaz nejsou relevantní. Z dat je patrné, že s přibývajícím počtem dokumentů roste počet chybně vybraných dokumentů. Dále je také vidět, že největší procentuální chybovost vzhledem k celkovému počtu dat je pro všechny 3 dotazy dosažena u indexu s 40000 dokumenty. Tato chybovost pro jednotlivé dotazy je znázorněna na Obrázku 12.



Obrázek 10: Porovnání časů pro dotazy

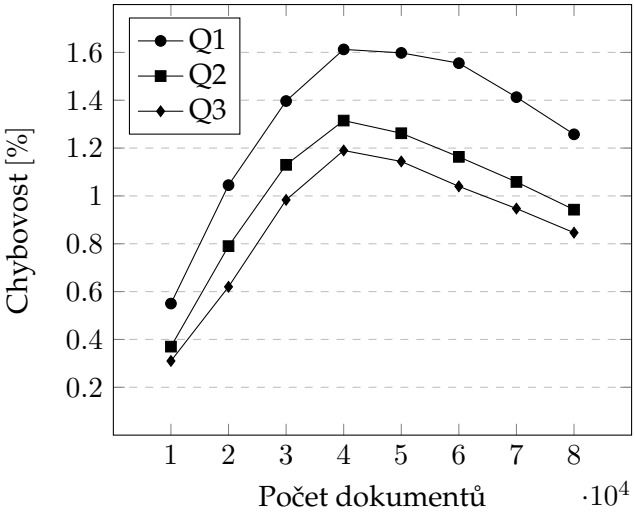


Obrázek 11: Porovnání časů pro dotazy bez načtení offsetu

4.3 Vliv počtu termů v dotazu na vyhodnocení dotazu

V této části textu je ukázáno jaký vliv má počet termů v dotazu, tedy váha dotazu u signaturních souborů, na čas potřebný pro vyhodnocení dotazu a jak váha dotazu ovlivní počet chybně vybraných dokumentů a počet uzlů, ke kterým je přistupováno v S-stromu v případě signaturních souborů. Výsledná data pro booleovský model jsou znázorněna v Tabulce 14, pro signaturní soubory poté v Tabulce 15.

Pokud se nejdříve podíváme jak počet termů v dotazu ovlivňuje čas potřebný pro



Obrázek 12: Chybovost u signaturních souborů

Počet dokumentů $[\cdot 10^3]$		10	20	30	40	50	60	70	80
termů									
10	čas offsetu$[s]$	4,0	6,6	9,5	12,8	16,3	19,6	21,9	23,0
	čas$[ms]$	56,9	79,1	89,7	106,5	116,7	124,2	153,6	171,2
	relevantní	1	8	8	8	10	11	15	15
20	čas offsetu$[s]$	3,9	6,5	9,5	12,9	15,9	19,6	21,5	23,3
	čas$[ms]$	73,0	92,3	111,6	136,2	152,6	174,4	198,8	210,6
	relevantní	1	1	1	1	1	1	1	1
50	čas offsetu$[s]$	3,9	6,4	9,5	13,1	16,1	19,6	21,7	23,6
	čas$[ms]$	85,3	99,8	137,6	188,2	246,4	319,5	377,7	411,5
	relevantní	1	1	1	1	1	1	1	1
100	čas offsetu$[s]$	3,9	6,5	9,4	12,9	16,0	19,6	21,7	23,2
	čas$[ms]$	152,2	210,5	267,7	337,4	414,9	510,1	571,8	631,4
	relevantní	1	1	1	1	1	1	1	1
150	čas offsetu$[s]$	3,9	6,4	9,4	13,0	15,9	19,8	21,9	23,5
	čas	271,9	303,4	370,1	458,9	528,2	628,1	722,6	781,4
	relevantní	1	1	1	1	1	1	1	1

Tabulka 14: Výsledky booleovský model

vyhodnocení dotazu, tak pokud srovnáme pouze čas potřebný pro vyhodnocení dotazu, tedy nebudeme počítat čas potřebný pro načtení offsetu u booleovského modelu, tak je vidět, že booleovský model dosahuje lepších výsledků. Dalším rozdílem jak počet termů v dotazu ovlivňuje jednotlivé modely je to, že s přibývajícím počtem termů v dotazu roste čas potřebný pro vyhodnocení dotazu u booleovského modelu, ale u signaturních

Počet dokumentů $[\cdot 10^3]$		10	20	30	40	50	60	70	80
w(Q)									
30	uzlů navštívených	193	476	873	1276	1598	1871	2123	2286
	čas$[s]$	0,24	0,50	0,83	1,20	1,40	1,70	1,89	2,11
	relevantní	23	76	191	303	362	395	429	444
59	uzlů navštívených	125	352	710	1047	1318	1490	1671	1783
	čas$[s]$	0,16	0,38	0,71	1,01	1,17	1,36	1,46	1,64
	relevantní	18	55	146	238	269	292	313	320
121	uzlů navštívených	87	238	481	725	912	1022	1131	1201
	čas$[s]$	0,12	0,26	0,49	0,70	0,86	0,93	1,06	1,12
	relevantní	11	30	85	157	177	186	196	198
242	uzlů navštívených	72	188	389	602	760	853	944	1005
	čas$[s]$	0,10	0,21	0,40	0,59	0,72	0,79	0,90	0,91
	relevantní	10	25	64	123	138	145	152	153
367	uzlů navštívených	65	167	336	511	633	714	796	852
	čas$[s]$	0,097	0,19	0,35	0,49	0,62	0,65	0,73	0,75
	relevantní	10	20	52	104	117	124	129	130

Tabulka 15: Výsledky signaturní soubory

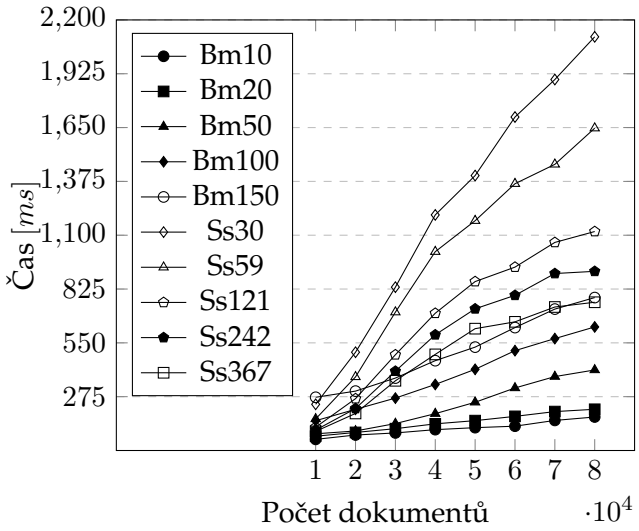
souborů klesá. To je způsobeno tím, že v případě booleovského modelu s přibývajícím počtem termů v dotazu roste počet operací potřebných pro vyhodnocení dotazu, naopak u signaturních souborů se zvyšuje váha signatury dotazu, která způsobí, že se přistoupí k menšímu počtu uzlů v S-stromu, tedy i menšímu počtu dat a čas proto klesne. Porovnání časů pro vyhodnocení dotazu pro oba modely je znázorněno na Obrázku 13.

Počet termů v dotazu má i značný vliv na celkovém počtu chybně vybraných dokumentů u signaturních souborů. Srovnání počtu chybně vybraných dokumentů pro různě velké váhy dotazu je znázorněno na Obrázku 14.

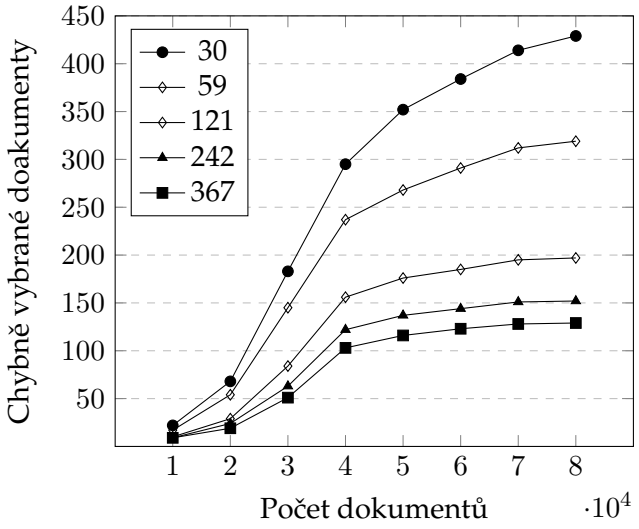
Zde je patrné, že s klesající váhou dotazu roste počet chybně vybraných dokumentů a naopak se zvyšující se váhou klesá. Důvodem proč při zvětšování váhy dotazu klesá počet chybně vybraných dokumentů je to, že větší váha způsobí, že více dat bude odfiltrováno a tím se přistoupí k menšímu počtu uzlů, tedy i k menšímu počtu dat. Závislost počtu chybně vybraných dokumentů na váze dotazu je poté znázorněna na Obrázku 15.

Jak již bylo zmíněno, tak u signaturních souborů s přibývajícím počtem termů v dotazu roste váha signatury dotazy, což způsobí, že se odfiltruje větší počet dat a přistoupí se k menšímu počtu uzlů. Na Obrázku 16 je znázorněno ke kolika procentům uzlů se pro jednotlivé dotazy přistupovalo.

Je vidět, že nejhorší případ je pro dotaz vytvořený z nejmenšího počtu termů, což znamená, že má signatura takového dotazu nejmenší váhu a přistoupí se k největšímu počtu uzlů tedy i dat. Ale i u tohoto dotazu způsobila signatura, že bylo odfiltrováno přes 50 % uzlů, tedy i dat obsažených v nich. Naopak pro dotaz jehož signatura má největší váhu bylo v nejhorším případě navštíveno pouhých 18 % uzlů, naopak v nejlepším případě je



Obrázek 13: Čas pro vyhodnocení dotazu pro různý počet termů v dotazu

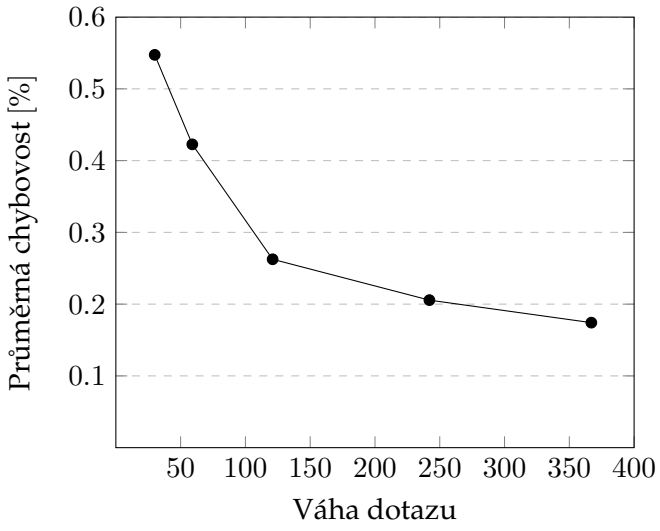


Obrázek 14: Chybně vybrané dokumenty pro různou váhu dotazu

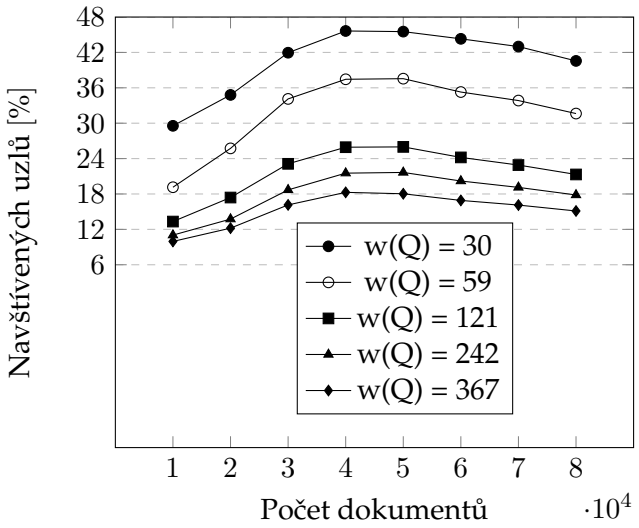
to jenom 10 %. Samotná závislost procent navštívených uzlů v závislosti na váze dotazu je znázorněna na Obrázku 17.

4.4 Detekce spamu

Další porovnání booleovského modelu a signaturních souborů bylo provedeno na konkrétní aplikaci a to detekci spamu. Aplikaci funguje tak, že 50000 dokumentů bylo použito pro vytvoření indexu a zbylých 30000 bylo postaveno proti tomuto indexu. Obsah



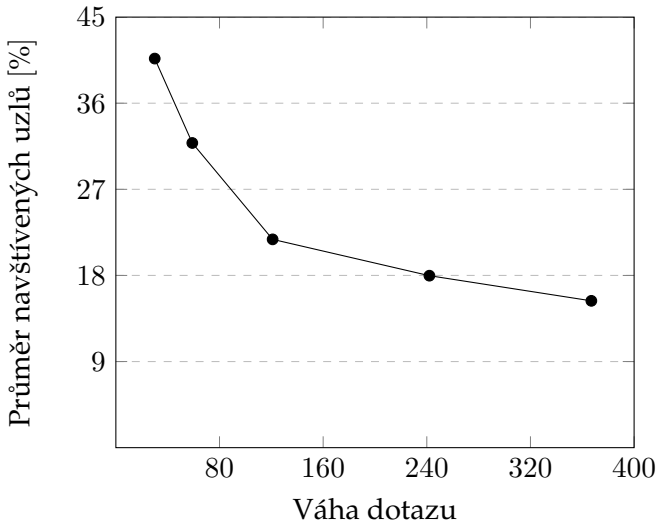
Obrázek 15: Průměrná chybovost v závislosti na váze dotazu



Obrázek 16: Procentuální navštívenost uzlů pro jednotlivé dotazy

jednotlivých dokumentů byl rozdělen na věty. Z daných vět byl postaven dotaz. U signaturních souborů se vytvořila signatura dotazu navrstvením jednotlivých termů ve větě. U booleovského modelu byly termy spojeny do konjunkce jednotlivých termů. Poté se počítalo kolik procent vět z daného dokumentu je již v indexu. Počty dokumentů pro rozsahy jednotlivých procentuálních vyjádření obsahu jsou znázorněny v Tabulce 16.

Z dat je patrné, že výsledky se poměrně liší pro jednotlivé modely, což je způsobeno pravděpodobností chybného výběru u signaturních souborů. Je vidět, že u signaturních souborů dané nastavení způsobilo, že větší počet dokumentů v procentuálním vyjádření



Obrázek 17: Průměrný počet navštívených uzlů v závislosti na velikosti signatury dotazu

Rozsah [%]	Booleovský model	Signaturní soubory
0 - 20	7433	8937
21 - 40	3281	5971
41 - 60	4898	5207
61 - 80	6732	4322
81 - 100	7656	5563

Tabulka 16: Počet dokumentů pro daný rozsah

obsahu je v rozmezí 0 - 40 % a naopak menší počet v rozmezí 60 - 100 % oproti booleovskému modelu. Pokud bychom tedy braly, že dokument je vybrán jako spam tehdy, pokud více než 60 % jeho obsahu je obsaženo v indexu, tak poté pro booleovský model by jako spam bylo vybráno 48 % dokumentů a u siganturních souborů by to bylo 33 % dokumentů.

Výhodou u booleovského modelu je to, že pro konkrétní data obdržíme pouze relevantní data, a proto detekce spamu je daleko přesnější, než u signaturních souborů, jelikož u nich se při rozhodování zda je dotaz obsažený v indexu musí brát v potaz pravděpodobnost chybného výběru. V aplikaci se počítalo s hodnotou kolem 1 %. Nevýhodou u booleovského modelu je to, že z lepší vyjadřovací schopnosti, kterou oproti signaturním souborům má bylo využito pouze operátoru AND a tudíž tak ztrácí jednu z největších výhod, kterou oproti signaturním souborům má.

5 Nastavení signaturních souborů

Jak již bylo zmíněno, tak jednou z hlavních nevýhod signaturních souborů je, že vykazují jistou pravděpodobnost chybného výběru. Pro dosažení co nejlepších výsledků je možné pravděpodobnost chybného výběru do jisté míry ovlivnit. Způsobem jak tuto pravděpodobnost ovlivnit je zvolit co nejlepší nastavení. V následující části textu je provedeno porovnání pro určení co nejvhodnější nastavení pro signaturní soubory, které bylo použito i u porovnání v předchozí části. Pro porovnání byl použit dotaz Q1 z Tabulky 12.

5.1 Velikost signatury

První a nejspíše i nejvíce důležitým parametrem, který je potřeba dobře zvolit je velikost signatury. Pro zvolení co nejvhodnější velikosti signatury je potřeba aby daná velikost vykazovalo co nejmenší počet chybně vybraných dokumentů a byla co nejefektivněji využita. Z těchto důvodů byl měřen počet chybně vybraných dokumentů a průměrná váha signatury pro různé velikosti signatur. Výsledná data počtu chybně vybraných dokumentů jsou znázorněna v Tabulce 17. V Tabulce 18 je poté znázorněna průměrná váha signatury.

Velikost signatury / Počet dokumentů	200	500	700	1000	2000	5000
10000	2280	742	161	55	21	4
20000	4064	1322	582	209	91	10
30000	6248	1997	954	419	183	14
40000	8494	2825	1469	645	270	27
50000	10357	3390	1784	799	325	34
60000	12470	3917	2118	933	359	36
70000	14254	4314	2338	989	380	37
80000	15897	4577	2518	1006	386	37

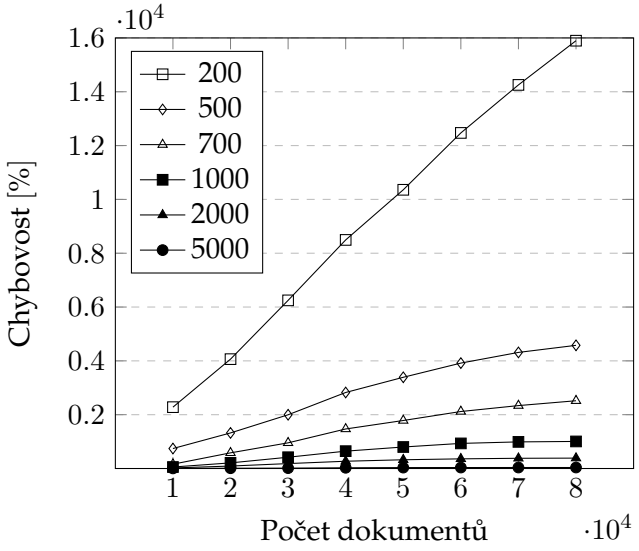
Tabulka 17: Počet chybně vybraných dokumentů pro dané velikosti signatury

U porovnání počtu chybně vybraných dokumentů uvedených v Tabulce 17 je patrné, že se zvětšující se velikostí signatury klesá počet chybně vybraných dokumentů. Srovnání počtu chybně vybraných dokumentů pro dané velikosti signatury je znázorněno na Obrázku 18.

Pro zvolení nejvhodnější délky signatury je důležité, aby pro danou délku signatury byla pravděpodobnost chybného výběru co nejmenší a signatura byla co nejefektivněji využita. Důležitost dobře zvolené délky je vidět i na datech v Tabulce 17, protože pokud byla zvolená signatura příliš malá, tak pravděpodobnost chybného výběru byla až 23%, což už je dost neúnosné. Průměrná pravděpodobnost chybného výběru pro jednotlivé velikosti signatury je znázorněna na Obrázku 19. Zde je vidět, že se zvětšující se velikostí signatury klesá pravděpodobnost chybného výběru exponenciálně, proto pro signatury větší než 800 bitů už pokles není tak výrazný.

Velikost signatury / Počet dokumentů	200	500	700	1000	2000	5000
10000	114	190	222	252	306	351
20000	113	185	213	241	290	333
30000	118	192	222	250	302	348
40000	120	196	227	256	310	360
50000	119	194	224	253	305	354
60000	119	193	222	250	300	346
70000	118	192	221	248	296	340
80000	118	190	218	244	290	332

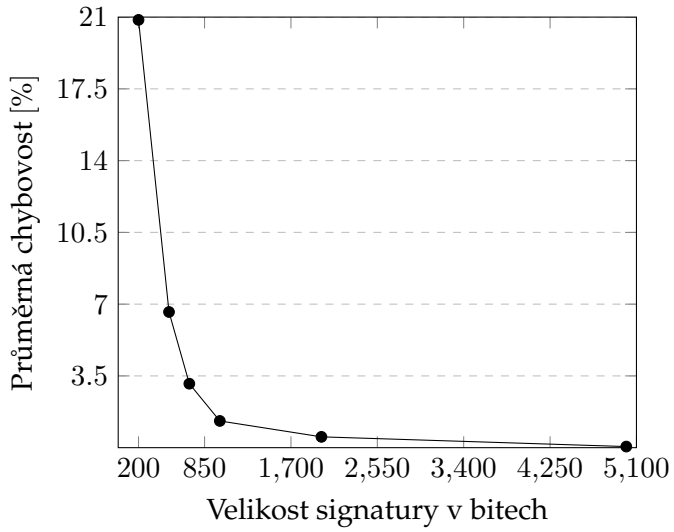
Tabulka 18: Průměrný váha signatury pro dané velikosti



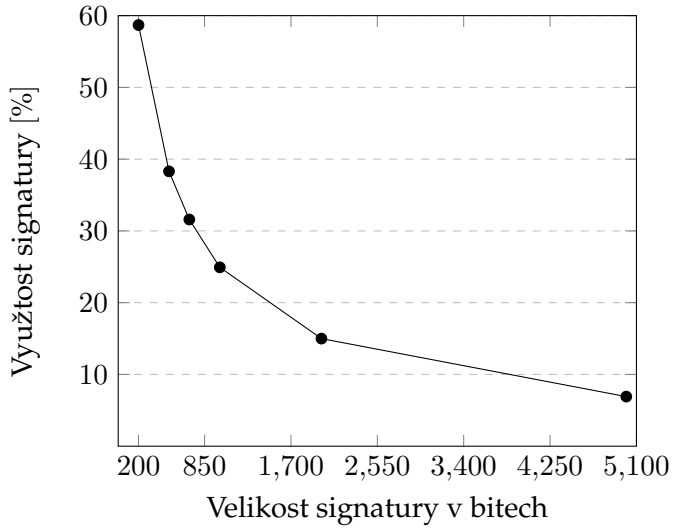
Obrázek 18: Chybně vybrané dokumenty pro různě velké signatury

Pokud by se bralo v úvahu jen počet chybně vybraných dokumentů, tak by se mohlo zdát, že nejvhodnější by bylo zvolit velikost signatury co největší, ale problémem zde je, že pokud zvolíme signaturu příliš velkou, tak výsledná váha signatury bude malá a signatura bude z velké části nevyužita, což dokazuje i Obrázek 20, kde je vidět, že u největší velikosti signatury je využito pouhých 7% z celkové velikost signatury. Navíc velká velikost signatury zbytečně zabírá více místa na disku.

Z předešlých měření byla jako nejvhodnější vybrána signatura s velikostí 1000 bitů, jelikož poskytuje kvalitní procento efektivního využití signatury a také proto, že má přijatelnou pravděpodobnost chybného výběru.



Obrázek 19: Chybovost v závislosti na velikosti signatury



Obrázek 20: Průměrná využitost signatury pro daný počet nastavovaných bitů

5.2 Počet nastavovaných bitů

Dalším parametrem, který stejně jako velikost signatury ovlivňuje počet chybně vybraných dokumentů a využitost signatury, i když ne v takové míře, je počet nastavovaných bitů v signatuře pro jednotlivé termy. Výsledné počty chybně vybraných dokumentů pro jednotlivé počty nastavovaných bitů jsou znázorněny v Tabulce 19.V Tabulce 20 je poté uvedená průměrná využitost signatury s daným počtem nastavovaných bitů.

Pokud se nejdříve podíváme na počet chybně vybraných dokumentů pro jednotlivé

Nastavovaných bitů / Počet dokumentů	1	3	5	7	10
10000	75	55	291	381	1252
20000	224	209	514	752	2043
30000	383	419	922	1260	2887
40000	557	645	1219	1922	3889
50000	677	799	1454	2381	4609
60000	774	933	1645	2642	5183
70000	874	989	1858	2967	5657
80000	898	1006	1961	3090	6004

Tabulka 19: Počet chybně vybraných dokumentů pro daný počet nastavovaných bitů

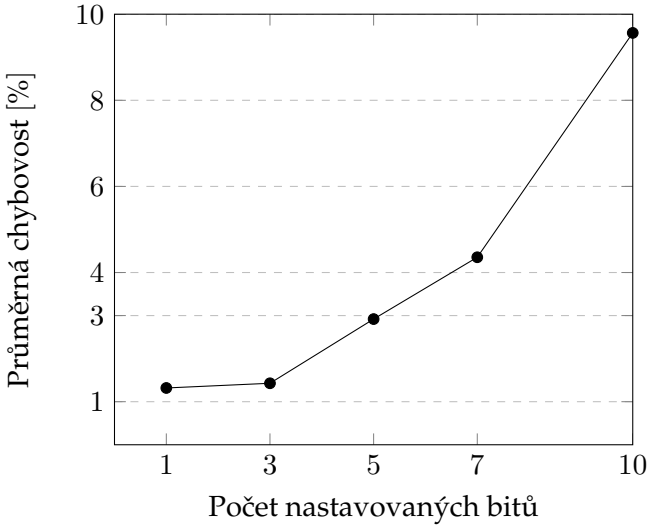
Nastavovaných bitů / Počet dokumentů	1	3	5	7	10
10000	109	252	345	411	485
20000	104	241	332	400	476
30000	108	250	346	418	498
40000	111	256	353	426	507
50000	110	253	349	422	503
60000	108	250	347	420	501
70000	106	248	344	417	499
80000	104	244	341	414	497

Tabulka 20: Průměrný váha signatury pro daný počet nastavovaných bitů

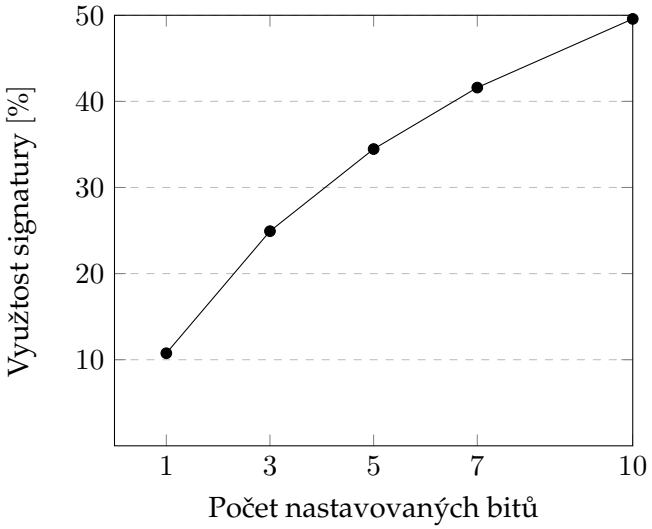
počty nastavovaných bitů, tak je vidět, že se zvětšujícím se počtem nastavovaných bitů roste počet chybně vybraných dokumentů. Průměrná chybovost pro jednotlivé počty je znázorněna na Obrázku 21.

Zde je vidět, že nejlepších výsledků bylo dosaženo pokud počet nastavovaných bitů byl roven 1, ale pokud se podíváme na chybovost pokud se hašovalo do 3 bitů, tak vidíme, že rozdíl není moc velký, ale spíše téměř totožný. Přesto však nastavení kdy jednotlivé termy byly nastavovány do 3 bitů v signatuře vykazuje o dost lepší výsledky, což je znázorně na Obrázku 22, kde je zobrazena průměrná využitost signatury pro daný počet nastavovaných bitů.

Zde je vidět, že nejvíce využitá signatura je v případě, kdy jednotlivé termy byly nastavovány do 10 bitů v signatuře, ale to je v kontrastu s průměrnou chybovostí 9,6 %, což je nejhorší výsledek pro dané nastavení. Pokud se vrátíme k porovnání, kdy si jednotlivé termy nastavovaly do 1, respektive 3 bitů, tak zde je vidět, že při téměř shodné chybovosti je využitost signatury o 15 % lepší, a proto nastaví, kdy si jednotlivé termy nastavovaly do 3 bitů v signatuře bylo zvoleno jako nejvhodnější.



Obrázek 21: Chybovost v závislosti na počtu nastavovaných bitů



Obrázek 22: Průměrná využitost signatury pro daný počet nastavovaných bitů

6 Závěr

V práci byl vytvořen booleovský model a model založený na signaturních souborech. Oba modely byly poté porovnány. Porovnání bylo provedeno na vytváření indexu, na výsledcích pro různé dotazy a času potřebného pro jejich vyhodnocení a na vlivu počtu termů v dotazu na oba modely. U signaturních souborů byla ukázána jedna z hlavních nevýhod a to, že pro daný dotaz vybírá i nerelevantní dokumenty. Pak bylo ukázáno jak je možné počet chybně vybraných dokumentů s danou kolekcí dokumentů ovlivnit správným nastavením signaturních souborů.

Booleovský model dosahuje přesnějších výsledků než signaturní soubory a pokud se u invertovaného indexu v jeho případě udržuje seznam termů ve vnitřní paměti, tak i dotazování dosahuje rychlejších výsledků, než u signaturních souborů. Přesto, že má booleovský model daleko větší možnosti pro dotazování než signaturní soubory, tak stále jsou zde značné omezení. Signaturní soubory sice nedosahují takových výsledků pro dotazování, ale za to jsou prostorově značně méně náročné a tvorba indexu je daleko rychlejší než u booleovského modelu. Signaturní soubory se tedy nedají použít pro vyhledávání přesných výsledků, ale pokud by byly použity jako první stupeň při dotazování, tedy sloužili by jako způsob filtrace nerelevantních dokumentů, kde ve druhé fázi by neodfiltrované dokumenty byly použity proti jinému modelu, tak se dají velmi dobře uplatnit. U booleovského modelu by mohlo dojít ke zlepšení pokud by invertovaný index nebyl organizovaný jako sekvenční soubor, ale třeba jako nějaká stromová struktura.

7 Reference

[1] POKORNÝ, Jaroslav, Václav SNÁŠEL a Michal KOPECKÝ. *Dokumentografické informační systémy.* Praha: Karolinum, 1998, 184 s. ISBN 80-246-1148-1.

[2] FRAKES, William B a R BAEZA-YATES. *Information retrieval: data structures*. Englewood Cliffs, N.J.: Prentice Hall, c1992, viii, 504 p. ISBN 01-346-3837-9.

[3] RYDVAL, Slávek. Dokumentografické informační systémy 1. In: MAREK, Vlastimil. *Softwarové noviny* [online]. 08.08.2005 [cit. 2013-11-17]. Dostupné z: http://www.rydval.cz/phprs/view.php?cisloclanku=1980010101

[4] RYDVAL, Slávek. Dokumentografické informační systémy 2. In: MAREK, Vlastimil. *Softwarové noviny* [online]. 08.08.2005 [cit. 2013-11-17]. Dostupné z: http://www.rydval.cz/phprs/view.php?cisloclanku=1980010102

[5] ZEZULA, P. a RABITTI, F..:*Dynamic Partitioning of Signature Files*, University of Bologna

[6] FALOUTSOS, CH. a CHRISTODOULAKIS, S.:*Signature Files: An Access Method for Documents and Its Analytical Performance Evaluation*, University of Toronto

[7] KORČÁK, Zdeněk. *Signaturní soubory se signaturami proměnné délky* [online]. Brno, CZ, 2002. Dostupné z: http://www.fit.vutbr.cz/research/view_pub.php?id=7127. Disertační práce. Fakulta informačních technologií VUT v Brně.

[8] SEDLÁČEK, Josef. *Algoritmy pro shlukování textových dat* [online]. Brno, CZ, 2011. Dostupné z: http://dspace.vutbr.cz/xmlui/handle/11012/1363. Diplomová práce. Fakulta elektrotechniky a komunikačních technologií VUT v Brně.

A Obsah CD

sf.jar	–	konzolová aplikace signaturních souborů
bm.jar	–	konzolová aplikace booleovského modelu
manual.txt	–	návod pro spuštění aplikací
5k.txt	–	ukázkový seznam 5000 dokumentů pro vytvoření indexu v aplikacích
spam.txt	–	ukázkový seznam 100 dokumentů pro spuštění aplikace jako detekce spamu
src	–	adresář se zdrojovými kódy aplikací
documents	–	adresář pro kolekci dokumentů
data	–	adresář pro uložení vytvořených indexů aplikacemi