Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Bakalářská práce

Komprese XML souborů

David Fait

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

David Fait

Informatika

Název práce

XML komprese

Název anglicky

Compression of XML

Cíle práce

Hlavní cíl
Otestování a zhodnocení vybraných nástrojů pro kompresi XML souborů.

Dílčí cíle jsou
- zpracovat analýzu současného stavu poznání,
- vybrat některé nástroje pro kompresi XML souborů,
- stanovit hodnotící kriteria,
- provést měření.

Metodika

Na základě studie odborné a vědecké literatury bude provedena analýza současného stavu poznání. Následně v praktické části bude vytvořena hodnotící tabulka a proběhne měření dle stanovených parametrů. Na základě zjištěných poznatků bude syntetizován závěr práce.

Doporučený rozsah práce

40 – 50 stránek

Klíčová slova

Komprese, Značkovací jazyk, Algoritmus, Testování, Výsledky testování, Testovaný HW, XMill

---

Doporučené zdroje informací

A. SAPATE, Suchit. Effective XML Compressor: XMill with LZMA Data Compression [online]. India, 2019 [cit. 2022-05-09].

Principles of Database Management [online]. Belgium, 2014

P. TIWAR, GYAN, ELENI STROULIA a ABHISHEK SRIVASTAVA. Compression of XML and JSON API Responses [online]. Canada, India: IEEE Access, 2021.

Předběžný termín obhajoby

2022/23 LS – PEF

Vedoucí práce

Ing. Alexandr Vasilenko, Ph.D.

Garantující pracoviště

Katedra informačních technologií

Elektronicky schváleno dne 14. 7. 2022

doc. Ing. Jiří Vaněk, Ph.D.

Vedoucí katedry

Elektronicky schváleno dne 27. 10. 2022

doc. Ing. Tomáš Šubrt, Ph.D.

Děkan

V Praze dne 03. 03. 2023

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Komprese XML" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu použitých zdrojů na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 15. 3. 2023 _____

Poděkování

Rád bych touto cestou poděkoval vedoucímu mé bakalářské práce panu
Ing. Alexandru Vasilenkovi, Ph.D. za jeho odborné vedení při vypracování této práce.

Komprese XML

Abstrakt

Tato bakalářská práce se zabývá problematikou komprese XML souborů různými kompresními mechanismy a následným zhodnocením dle hodnotících kritérií. Poskytuje tedy ucelený pohled na problematiku komprese XML souborů a může pomoci při výběru vhodného kompresního mechanismu pro konkrétní účely.

Teoretická část představuje značkovací jazyk XML, především jeho syntaxi. Dále se věnuje způsobům, jakými jsou XML soubory zpracovávány, a shrnutím jejich výhod a nevýhod. Následně jsou popsány rozdíly mezi typy kompresí a uvedeny různé dotazovací a nedotazovací kompresní metody. V poslední řadě jsou objasněny principy fungovaní algoritmů, které jsou používány uvedenými kompresními mechanismy.

Praktická část se zabývá testováním různých souborů XML, které se liší strukturou a velikostí. Tato testování jsou prováděna na počítačích s odlišným hardwarem. Individuální soubory jsou dále komprimovány dílčími kompresními mechanismy a jsou provedena měření na základě zvolených hodnotících kritérií. V závěru jsou shrnuty dílčí kompresní mechanismy včetně doporučení jejich využití v závislosti na konkrétních požadavcích a omezeních.

Klíčová slova: komprese, značkovací jazyk, algoritmus, měření, výsledky testování, testovaný HW, Hodnotící kritéria, LZMA, doba trvání komprese, kompresní mechanismy

Compression of XML

Abstract

This bachelor's thesis deals with the issue of compressing XML files using various compression mechanisms and subsequently evaluating them according to assessment criteria. It provides a comprehensive view of the issue of compressing XML files and can help in selecting the appropriate compression mechanism for specific purposes.

The theoretical part introduces the XML markup language, especially its syntax. It also discusses how XML files are processed and compares its advantages and disadvantages. Subsequently, the differences between compression types are described, and various query and non-query compression methods are presented. Finally, the principles of the algorithms used by the compression mechanisms are explained.

The practical part focuses on testing different XML files that vary in structure and size. These tests are performed on computers with different hardware. Individual files are then compressed using partial compression mechanisms, and measurements are taken based on selected evaluation criteria. Finally, the partial compression mechanisms are summarized, and recommendations are made regarding their use based on specific requirements and limitations.

Keywords: compression, markup language, algorithm, measurement, test results, tested hardware, evaluation criteria, LZMA, compression duration, compression mechanisms

Obsah

1 Úvod

XML již od roku 2001 patří mezi nejrozšířenější textové dokumenty pro přesun dat po internetu. S tím se pojí problematika ukládání velkých objemů dat a následném posílání po síti. Jedním z řešení redukce velkého objemu dat, ať už pro jejich posílání či ukládání, je jejich komprese, která může mít ovšem vysoké nároky na zátěž komponentů počítače či na rychlost zpracování dat. K tomuto účelu byla vyvinuta řada kompresních mechanismů. Na kompresní mechanismy stále vycházejí nové aktualizace či vychází jejich vylepšení např. za pomoci jiného algoritmu. Jelikož existují různé kompresní mechanismy, které se odlišují ať už výsledkem kompresního poměru, či zátěží na hardware, tak je důležité vědět, který kompresní mechanismus v jaké situaci využít.

Zaměřením práce je výběr určitých kompresních mechanismů, u kterých je provedeno měření na základě zvolených hodnotících kritérií a jejich vstupní nastavení. Z daných výsledků měření jsou syntetizovány závěry.

V teoretické části dojde k objasnění základů XML, převážně o jeho syntaxi a jejích pravidel. Toto je rozděleno do jednotlivých podkapitol, kde je tato syntaxe více rozebrána. V následujících kapitolách jsou rozebrány parsery, způsob zpracování dat, výhody, nevýhody XML a následně dílčí kompresní mechanismy. Kompresní mechanismy se obecně dělí na dotazovací a nedotazovací. Používají se pro redukci vstupních dat za účelem snížení nákladů a času na jejich přenos po síti a nižších nároků na jejich uložení. Jejich nevýhodou může být zátěž na hardware či dlouhá doba komprese. Tyto nevýhody jsou součástí testování v praktické části.

V praktické části jsou vybrány XML soubory, které jsou v pozdější fázi komprimovány vybranými kompresními mechanismy. U jednotlivých mechanismů jsou zvolena hodnotící kritéria, podle kterých je hodnocena jejich celková efektivita. Výsledky měření jsou posléze syntetizovány do grafů a tabulek. V závěru jsou zhodnoceny dílčí kompresní mechanismy a zformulována doporučení.

2 Cíl práce a metodika

2.1 Cíl práce

Hlavní cíl

Otestování a zhodnocení vybraných nástrojů pro kompresi XML souborů.

Dílčí cíle jsou

- zpracovat analýzu současného stavu poznání,
- vybrat některé nástroje pro kompresi XML souborů,
- stanovit hodnotící kritéria,
- provést měření.

2.2 Metodika

Na základě studia odborné a vědecké literatury bude provedena analýza současného stavu poznání. Následně v praktické části bude vytvořena hodnotící tabulka a proběhne měření dle stanovených parametrů. Na základě zjištěných poznatků bude syntetizován závěr práce.

3 Teoretická východiska

3.1 XML

XML (eXtensible Markup Language) patří mezi značkovací jazyky podobně jako HTML, XSL či DOC. Je odvozený ze staršího jazyka SGML (Standard Generalized Markup Language). Nejvíce se využívá pro distribuci dat přes internet.

Zásadní informací je, že XML samo o sobě nic nedělá a pro manipulaci s XML soubory je třeba jiný software. XML je využíváno pro výměnu dat mezi aplikacemi a je strukturou těchto předávaných dat. Data jsou ukládána jako ASCII[1] a lze je poté upravit v jakémkoliv textovém editoru.

Primární funkcí XML je vytvořit jednoduchý textový formát se strukturovanými informacemi. Tyto informace jsou reprezentovány počtem tagů[2] a strukturálními vztahy mezi jednotlivými tagy. Ve své podstatě XML uschovává data na jednom místě, kde jsou strukturovaná, dobře čitelná a snáze se s nimi pracuje [1].

V dnešní době byl vydán JSON, který funguje na stejné bázi jako XML a ten spoustu věcí vylepšuje, například zkrácení kódu, rychlost, je rozšířen z JavaScriptu; je ještě jednodušší a lépe čitelný.

Pro dotazování dat byl speciálně navržený XQuery, který je W3C[3] doporučením a je podporován všemi databázemi. Díky němu se dají v dokumentu snáze dohledávat informace za pomocí příkazů jako v SQL jazyce [4].

3.1.1 Vizualizace

XML je sice snadno pochopitelné, ale u větších kódů se může stát nepřehledným. Proto je možné si vzhled vizualizovat pomocí různých stylů. Mezi nejznámější patří zajisté CSS (Style Sheets for HTML).

[1] Tabulka pro reprezentaci znaků v počítačích
[2] Označení určené pro definici elementů či atributů
[3] Mezinárodní organizace zabývající se standardizací technologií

Spíše určené pro XML jsou XSL (eXtensible Stylesheet Language), které umožnují specifikovat vizuální formátování XML dokumentu a jsou W3C doporučením. XSL se skládá ze tří částí:

- XSLT (Extensible Stylesheet Language Transformations) je jazyk pro transformaci XML dokumentu na XHTML či jiné XML dokumenty [45].
- XPath (XML Path Language) je používán pro navigaci v souboru a přebírá tak nejdůležitější část XSL. Jak již bylo řečeno, XPath je jazyk využívaný XSLT, dále ho ale lze využít pro navigaci DOM (3.2.2) nebo pro otestování uzlů, zda odpovídají vzoru. Zápis cesty pro navigaci v XML probíhá podobně jako v URL[4] a může vypadat například takto: /html/body/div[1]/div[2]/div[2]/main/article/div/div/p.
- XSL-FO (XSL Formatting Objects) převádějí transformovaný dokument od XSLT na naformátovaný výstup [2].

3.1.2 XML syntaxe

Jazyk XML je velice náchylný na formátování. Při nesprávném formátování program, který následně zpracovává tato data, vrátí chybu. XML musí být „well-formed“ (správně formátovaný), aby posléze mohl být zpracován. K tomu je zapotřebí, aby odpovídal správné syntaxi a zároveň ho mohl parser (kapitola 3.2) přečíst a zpracovat [4]. Níže jsou uvedeny základní prvky syntaxe.

Elementy

Elementy jsou základním stavebním kamenem XML souborů. Jsou chápany jako kontejnery, které uchovávají data. Všechny elementy jsou ohraničeny takzvanými tagy. Musí být vloženy do speciálních znaků („<“ a „>“) a být uzavřeny (pomocí „/“). Je možné tyto elementy rozšiřovat, aby obsahovaly více informací [6].

Elementy nemusí mít nějaký obsah, stačí, když se do speciálních znaků uvede název elementu a poté je řádně ukončen. Prázdné elementy se dají také napsat zkrácenou formou (<element />), díky které se nemusí uvádět ukončovací tag. [5]

[4] Adresa umožňující přístup ke zdroji na internetu

Elementy mohou obsahovat text, číslice, jiné elementy či kombinaci těchto uvedených. Názvy elementů jsou takzvaně „case-sensitive“ (rozlišují malá a velká písmena) a musí začínat písmenem či podtržítkem. Co se týče interpunkčních znamének, tak jsou všechna, až na výjimky, zakázána (kromě pomlčky "-", podtržítka "_" a tečky ".").

Názvy nemohou začínat slovem „xml“ ani žádnou jinou variací tohoto slova (XmL ani xMl) a také nemohou obsahovat mezery [4].

Atributy

Atributy jsou částí elementu. Obsahují data, která jsou přidělena k určitým elementům a dodávají tak více informací. Mezi hlavní důvody využívání atributů patří jejich oddělení elementů od elementů, které nesou stejné jméno. Jejich využití pomáhá rozlišovat dva nebo více podobných elementů [6]. Atributy se zapisují do deklarace elementů (do počátečního tagu). Způsob zápisu je ukázán na obrázku 1.

```
<osoba pohlaví="muž">
```

Obrázek 1: Atributy XML (Zdroj: vlastní zpracování)

K elementu „osoba“ je připsán atribut „pohlaví“ a jemu je přidána pomocí rovnítka hodnota „muž“, která musí být vložena do apostrofů nebo do uvozovek. Atributy jsou vždy psané ve dvojici („název“ a „hodnota“) [4]. Níže jsou uvedena pravidla správného zápisu atributů:

- Element může mít více unikátních atributů.
- Hodnoty atributů musí být uvedeny za rovnítkem, jinak se nejedná o správnou syntaxi.
- Každý atribut musí být deklarován v DTD[5] (Document Type Definition) za využití „Attribute-List Declaration“[6].
- Hodnoty atributů (na rozdíl od elementů) nemohou obsahovat více než jednu hodnotu.
- Atributy nesmí odkazovat na externí entity a nemohou obsahovat tree structure (3.2.2).

[5] Definice struktury a pravidel pro validaci
[6] Součást DTD, umožňující definovat seznam atributů pro daný element

- Jméno atributu se nemůže vyskytovat v počátečním tagu nebo v prázdném elementu více než jednou [4].

Komentáře

Komentáře slouží k vysvětlení části kódu a mohou být přidávány jako poznámky či řádky. Jsou viditelné pouze ve zdrojovém kódu, nikoli v XML kódu, tudíž nejsou zpracovány programy. Zapisují se stejně jako elementy či atributy do speciálních znaků („<“ a „>“), pouze se na začátek za speciální znak přidá vykřičník a dvě pomlčky a na konec dvě pomlčky (viz obrázek 2):

```
<!--Komentář-->
```

Obrázek 2: Komentáře XML (Zdroj: vlastní zpracování)

Komentáře se mohou nacházet kdekoliv v dokumentu, ale nemohou se objevit před deklarací XML a jednotlivé komentáře nemohou být vloženy do dalších komentářů či hodnot atributů, jednalo by se tak o špatnou syntaxi [4].

Namespaces

Při sjednocení dvou XML dokumentů může dojít k chybě s názvem „name conflict“. Tato chyba vyjadřuje shodu dvou elementů v různých XML dokumentech. Chybě se dá předejít za pomoci použití „name prefixu“. Name prefix je přidání písmene před daný element, a tak se zajistí jeho ojedinělost [4].

```
<h:table>
```

Obrázek 3: Namespaces XML (Zdroj: vlastní zpracování)

Namespace musí být definovaný, pokud je cílem používat prefix v XML. Ten lze definovat pomocí xmlns na začátku vytváření elementu (syntaxe = „xmlns:*prefix*="*URI*"“).

URI (Uniform Resource Identifier) je soubor textových řetězců s definovanou strukturou. Slouží k jednoznačné identifikaci a používají se, jelikož jsou světově jedinečné. Nejpoužívanější URI je URL (Uniform Resource Locator), který je použit k identifikaci doménové adresy. Dalším typem URI je URN (Uniform Resource Name) [36].

Znakové entity

Znakové entity se používají pro zápis znaků, které není možno napsat v textu. Jsou tři typy znakových entit: předdefinované entity (např.: <, >, &), číselné znakové entity a pojmenované znakové entity. Pro předdefinované znaky platí <, >, &. Jsou zde ještě dvě předdefinované znakové entity, a to pro apostrofy a uvozovky ("&quot" a "&apot") [6].

Číselné entity mohou být v desítkové nebo hexadecimální soustavě. Číselné znaky odkazují na číselný znak v Unicode. Před kód je potřeba umístit prefix („#“ nebo „#x“) viz obrázek 4 [4].

```
&# decimalni cislo;
&#x hexadecimalni cislo;
```

Obrázek 4: Znakové entity XML (Zdroj: vlastní zpracování)

Mimo znakové entity existují i „textové entity“, které jsou děleny na externí a interní. Interní slouží ke zkrácení textu a externí slouží k vložení kódu do XML dokumentu [6].

CDATA

CDATA se uznávají jako značky, ale nejsou analyzovány parserem. Slouží k uložení dlouhých textů, které se mají zobrazit. Bez využití CDATA je nutné používat znakové entity. Tyto entity nařizují parseru, aby se k této konkrétní části přistupovalo jako k textovému řetězci [6].

Na první pohled se může zdát, že CDATA jsou podobná komentářům, ale existují mezi nimi určité rozdíly. Na rozdíl od komentářů jsou CDATA stále součástí XML dokumentu a odkazy na entity jsou rozpoznány. Avšak CDATA nemohou obsahovat znaky "]]>" v žádné části dokumentu [6].

Syntaxe začíná „<![CDATA[“ a uzavírá se pomocí „]]>“ viz obrázek 5 [6].

```
<![CDATA[
dlouhy text
]]>
```

Obrázek 5: CDATA XML (Zdroj: vlastní zpracování)

Pravidla syntaxe

U syntaxe je mnoho pravidel, která je pro správný zápis XML nutno dodržet. V bodech jsou vypsána pouze ty nejzásadnější:

- Každý XML dokument musí obsahovat jeden "root" element, který bude nadřazený všem ostatním elementům. Tento element se často nazývá rodičovský (parent). Do něj jsou zabalené všechny následující elementy a větve či pod-větve.
- Všechny tagy musí být zakončeny a zároveň veškeré elementy musí být správně vnořeny do sebe.
- Všechny hodnoty atributů musí být v uvozovkách [4].

Aby bylo XML funkční a mohlo být úspěšně parsováno, musí být jeho struktura „well-formed“. Well-formed se dokument stane poté, co odpovídá standardům W3C.

U souboru je třeba také dodržet validitu. I když dokument splňuje kritéria pro well-formed XML, neznamená to automaticky, že je také validní. Pro ověření validity XML souboru je zapotřebí použít DTD nebo XSD (soubory pravidel). Pokud validátor nalezne v dokumentu chybu, zastaví činnost [7].

3.2 Zpracování dat a parsování XML

3.2.1 Parsování

Důležitým krokem je zpracování dat, které je možné až po proběhnutí parsování, které převede dokument tak, aby byl přístupný.

Parsování je operace, která je prováděna předtím, než se začne s dokumentem jakkoliv manipulovat. Jejím cílem je tedy převést XML na kód, který si může adresát a jiné aplikace přečíst a pracovat s nimi. Parsery zpracovávají data a strukturu dat obsažené v dokumentu XML. Parser nejprve prozkoumá dokument, jestli se v něm nenachází nějaké chyby, a poté ho ověří vůči DTD (Document Type Definition) nebo schématu (jestli je

validní). Pokud splňuje veškerá pravidla W3C a je validní, tak s dokumentem mohou pracovat ostatní aplikace, jako je DOM (3.2.2) či SAX (3.2.3) [8].

Existuje široké spektrum možností výběru softwaru, který parsuje XML dokumenty, např. MSXML, Saxon, Xerces atd. Každý parser je vhodný pro jiný programovací jazyk. Je důležité podotknout, že v dnešní době mají prohlížeče samy zabudované parsery [8].

3.2.2 DOM

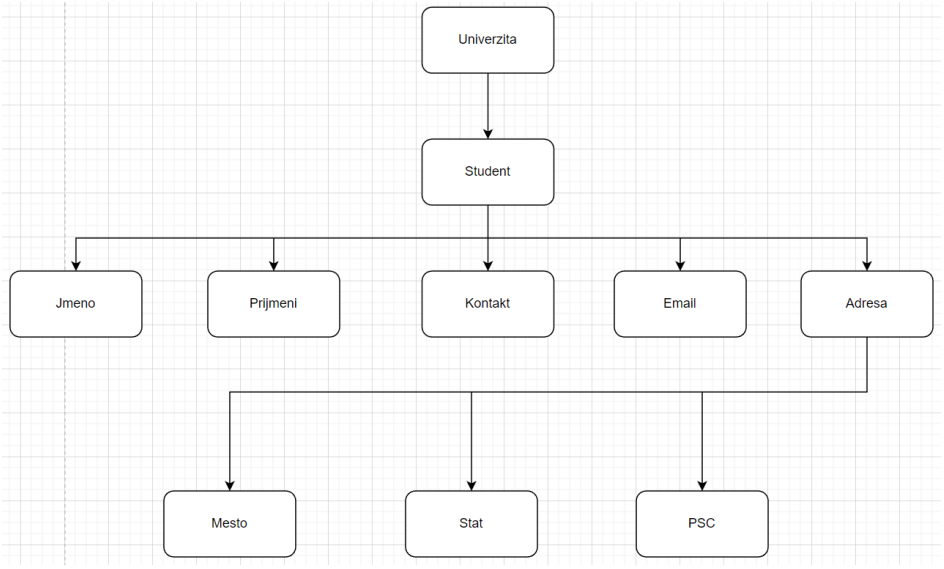
Na rozdíl od SAX je DOM (Document Object Model) W3C standardem. Jelikož je W3C standardem, tak jeho důležitým cílem je poskytnout nějaké programovací rozhraní. Značnou výhodou je, že může být použit s jakýmkoliv programovacím jazykem. Je vhodné zmínit možnost využívání xPath, který značně ulehčuje práci s dokumenty [10].

DOM přistupuje k určitým částem souboru. Umožnuje manipulaci, úpravu a čtení XML dokumentu. XML dokumenty jsou tvořeny takzvanými stromovými strukturami. Aby byla tato struktura vytvořena, je potřeba přečíst celý dokument.

Stromové struktury neboli „Tree structures“ jsou důležitým nástrojem pro snadný popis XML dokumentů. Tyto struktury se vždy skládají z kořenových (nadřazených) elementů (root element), podřízených elementů atd... [9].

```
<?xml version = "1.0"?>
<Univerzita>
    <Student>
        <Jmeno>David</Jmeno>
        <Prijmeni>Fait</Prijmeni>¨
        <Kontakt>159753846</Kontakt>
        <Email>davidfait@szn.cz</Email>
        <Adresa>
            <Mesto>Tokyo</Mesto>
            <Stat>Japonsko</Stat>
            <PSC>15748</PSC>
        </Adresa>
    </Student>
</Univerzita>
```

Obrázek 6: Náhled na zápis XML kódu (Zdroj: vlastní zpracování)



Obrázek 7: Stromová struktura XML dokument (Zdroj: 37, upraveno)

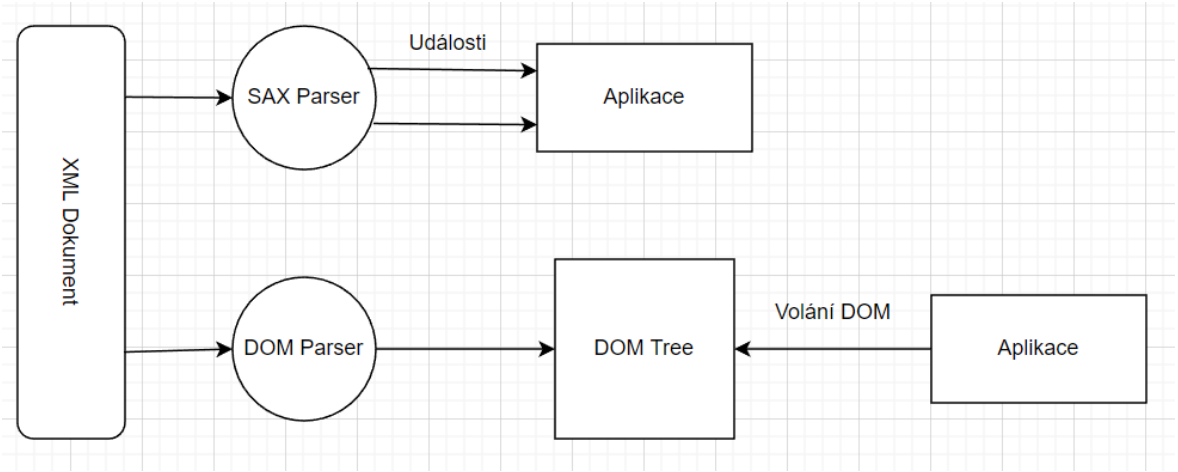
Na obrázku 6 je vidět ukázka kódu a na obrázku 7 je poté znázorněný stejný kód v Tree structure. Na úplném vrcholu se nachází root element, který nese jméno „Univerzita“. Uvnitř tohoto root elementu se nachází další element „Student“. Tento element v sobě má 5 větví „Jmeno“, „Prijmeni“, „Kontakt“, „Email“ a „Adresu“. Adresa má v sobě další tři pod-větve.

3.2.3 SAX

SAX (Simple API for XML) je komunitním standardem. Na rozdíl od DOM, který je založen na tree structure, je SAX založen na událostech, které jsou generovány v průběhu čtení dokumentu. Díky API[7] rozhraní má taky efektivnější správu paměti než DOM, díky následnému zahození události, která již byla převzata [5].

Jak už bylo řečeno o pár řádků výše, SAX využívá události. Události jsou vyvolávány k jednotlivým částem kódu (elemenům, komentářům, tagům atd.). V událostech se nacházejí důležité informace, jako je třeba obsažený text uvnitř elementu či začátek a konec elementu. Tyto události jsou následně zpracovány obsluhou událostí, ale ke zpracování touto obsluhou je potřeba naimplementovat funkce. Funkce je třeba zaregistrovat ve vytvořeném parseru a po jejich zaregistrování probíhá čtení dokumentu. Na obrázku 8 je vidět porovnání DOM vs SAX [11].

[7] Sada pravidel, protokolů a nástrojů umožňující dvěma různým zařízením vzájemnou komunikaci



Obrázek 8: Rozdíl mezi SAX a DOM (Zdroj: 38, upraveno)

3.3 Výhody a nevýhody XML

Většina těchto výhod či nevýhod vyplývá z informací, které již byly v tomto dokumentu uvedeny.

3.3.1 Výhody

Jelikož je XML velice rozšířené, tak k němu existuje i spousta softwaru, který je stále upravován a nově vydáván. Je jednoduchý, adaptivní, nezávislý na jiných platformách nebo na programovacích jazycích a snadno srozumitelný i pro uživatele, kteří se s tímto jazykem v životě nesetkali. K nezávislosti na platformě se ještě váže jednoduchost sdílení mezi různými systémy.

Sdílená data se dají kdykoliv měnit, a to aniž by to ovlivnilo prezentaci dat. XML podporuje řadu funkcí, jako je třeba validace za pomocí DTD a schema, které zabraňuje vypuštění XML dokumentu s errory nebo také podporuje UNICODE (umožnuje přenášet informace v jakémkoliv lidském jazyce). Mimo jiné lze XML otevřít v jakémkoliv textovém editoru [13].

3.3.2 Nevýhody

Podobně jako každý jiný jazyk má i XML řadu nevýhod. XML soubory bývají často velice obsáhlé kvůli podrobnosti zápisu, takže jejich komprese může zabrat spoustu času a zabere i spoustu místa.

Data XML se nedají zobrazit bez pomoci jiného programu. Při deklaraci je potřeba uvést jazyk, aby prohlížeč byl schopný porozumět napsanému kódu. Bez deklarace jazyka prohlížeč nebude schopný zobrazit XML dokument.

Je potřeba uzavřít tagy a deklarovat kořenový element, jinak se bude jednat o neplatnou syntaxi XML a validátor vrátí dokument s chybou [12].

3.4 Komprese

V dnešní době je mnoho aplikací, které obsahují velký obnos dat a je potřeba je přenášet prostřednictvím internetu či ukládat je na disk. Cílem komprese je tedy zmenšit výsledný soubor před jeho následným přenosem po síti [31]. Komprese zmenšením souborů značně urychlila posílání dat přes internet, především pro velké webové služby, které doručují obsáhlé množství souborů. Kompresi obstarávají různé algoritmy či funkce. Data se dají jak komprimovat, tak i dekomprimovat (opak komprese). Obecně se komprese dat dělí na dva způsoby, které jsou uvedeny níže [24].

Komprese textu odstraňuje všechny nepotřebné znaky vložením jediného znaku, který bude referencí řetězce opakujících se znaků. Velikost souboru se tak zmenší o 50 % a víc [24].

3.4.1 Bezeztrátová komprese

Jedná se o typ komprese, která neodstraňuje žádné informace z dokumentu, ale na druhou stranu není komprese natolik efektivní jako u ztrátové. Komprese dat se dosahuje za pomoci odstranění redundance v datech. Tento způsob komprese je využíván v případě nutnosti zkomprimovat soubor, na kterém je potřeba zachovat kvalitu, např. formáty PNG [25].

Algoritmy, které používají tento typ komprese, jsou například: Huffman encoding, dictionary encoding či delta encoding. Všechny tyto algoritmy jsou vysvětleny v pozdějších kapitolách 4.9 [26].

3.4.2 Ztrátová komprese

Tato komprese na rozdíl od bezeztrátové odstraňuje data, která nejsou v souboru důležitá. Tímto způsobem ztrátová komprese docílí větší komprese, ale za cenu možné degradace dat. Po použití této metody je soubor neobnovitelný ve své původní hodnotě. Nevhodná je hlavně u souborů, které chtějí autoři zachovat, např. tabulky (výsledek komprese by byl nepoužitelný). Čím více je soubor komprimován, tím více je snižována kvalita výstupního souboru. Ideální použití této komprese je pro MP3 formát a nejčastější použití je pro textová data [25].

3.5 Nedotazovací kompresory

Kompresory tohoto typu nepodporují zpracování dotazů, ale na druhou stranu se soustředí na dosažení lepších kompresních poměrů. Pro vyhodnocení dotazů je třeba dekomprimovat celý soubor [25].

Nedotazovací kompresory často pro účel komprese oddělují strukturu od dat, tato data následně vloží do kontejnerů. Kontejnery jsou kompresovány vhodným kompresorem, který odpovídá datovému typu daných kontejnerů např: GZip [25].

3.5.1 BZip2

Tento kompresor byl vydán roku 1996 a je open source[8]. Převážně je využíván na zařízeních s operačním systémem Linux, ale využívá se i u všech ostatních. Je založený na XBW (4.9.1) transformaci dat. Umožnuje ovšem i dekompresi [14].

Pro kompresi dat využívá BZip2 algoritmus Burrows-wheeler, který převádí často se opakující znaky na identické řetězce písmen. Po tomto kroku se použije Huffmanův kód k přesunu informací. BZip2 slouží pouze ke kompresi jednoho souboru, a ne více souborů najednou. Kvůli tomuto omezení se v linuxu tato komprese kaskáduje do archivu TAR (data a metadata více souborů). Data nelze šifrovat ani archivovat. Kompresované soubory mají výstupní příponu ve tvaru „.bz2“ [14].

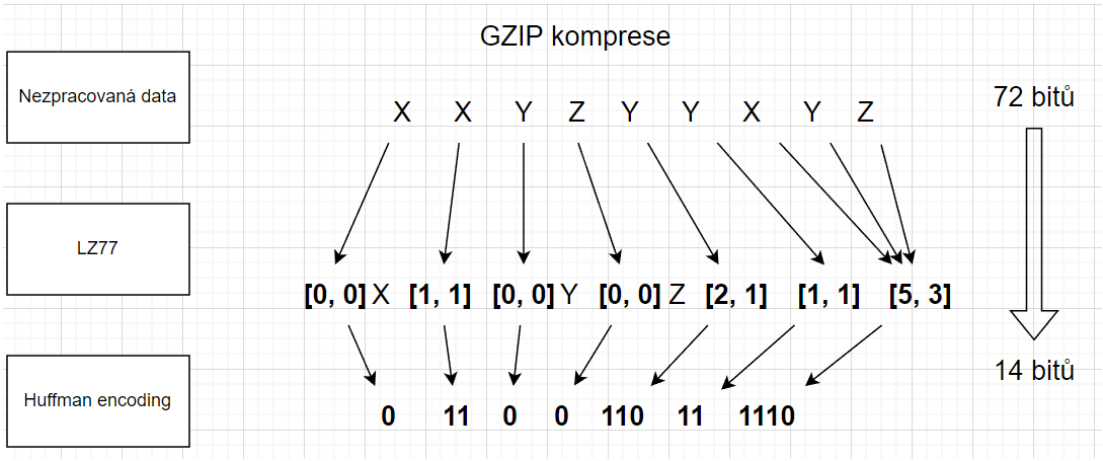
[8] Aplikace jejíž zdrojový kód je možné šířit, upravovat, vylepšovat a je zdarma.

3.5.2 GZip

GZip slouží jak pro kompresi, tak i pro následnou dekompresi. Pro kompresi využívá algoritmus „Deflate“, který je kombinací LZ77 (Lempel-Ziv) a Huffmanova kódování. Používá se na Unixových systémech, a to již od roku 1992. Je podporován prohlížeči a je součástí protokolu HTTP i HTTP2 [15].

Jeho hlavní výhodou je jeho rychlost. Vlastníci webové stránky si mohou nastavit server, aby data při vyžádání komprimoval. Pokud klient požádá o obsah, server zkontroluje hlavičku této žádosti pro informaci, zdali prohlížeč podporuje GZip. Pokud je podporován, tak před kompresí je vygenerováno označení stránky, aby prohlížeči předal informaci, čím bylo kompresováno. Poté se data na serveru zkomprimují (zabalí) za pomoci algoritmu Deflate a data, která jsou opakovaně nalezena, jsou nahrazena jedinečným identifikátorem. Po těchto krocích jsou data poslána prostřednictvím prohlížeče, který je poté dekomprimuje (rozbalí).

Díky tomuto zmenšení souborů je značně urychleno načítaní webové stránky. GZip také nabízí možnosti, jak moc chce autor, aby data byla komprimována. Čím větší chce zmenšení, tím větší je zatížení na CPU (procesor). GZip nemůže komprimovat obrázky [16].



Obrázek 9: Princip GZip kompresoru (Zdroj: 39, upraveno)

3.5.3 XMLPPM

XMLPPM (XML Prediction by Partial Matching) používá SAX pro parsování a model multiplexovaného hierarchického PPM (MHM) určený pro kompresi. Je kombinací několika modelů k predikci dalších znaků [28].

MHM[9] se skládá ze čtyř různých PPM modelů. ‚Chars' pro textové řetězce, ‚Syms' pro elementy a názvy atributů a ‚Atts' pro atributy. PPM odhaduje rozsah pravděpodobnosti pro daný symbol a s tímto odhadovaným rozsahem pravděpodobnosti dále pracuje aritmetická metoda pro kompresi souboru [23].

Všechna vstupní data jsou zpracována jako sekvence událostí SAX a každý token v těchto událostech je následně zpracován vhodným PPM modelem (jedním ze čtyř). V konečné fázi jsou všechny značky nahrazeny znakem ‚FF' [23].

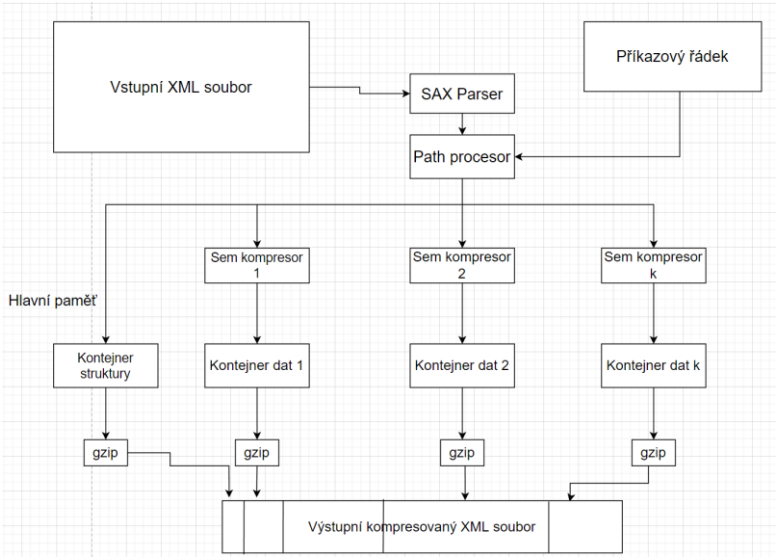
3.5.4 XMill

XMill byl vyvinut roku 1999 v New Jersey. Není to ve své podstatě kompresor, ale využívá jiné kompresory, jako je třeba GZip. XMill nejprve analyzuje celý XML soubor a na každou část kódu použije jiný kompresor [5].

V dokumentu XMill jsou všechna data XML roztříděna do tří druhů: značky, datové hodnoty a atributy. Ty jsou propojeny do stromové struktury, kde jsou vnitřní uzly označeny značkami či atributy a listy jsou označeny datovými hodnotami. Pro tyto uvedené druhy XMill funguje následovně: nejprve jsou metadata ve tvaru XML tagů i atributů odděleně komprimována od dat. Poté veškeré datové položky, které jsou (co se týče sémantiky) stejné, jsou seskupeny do kontejnerů. Toto se dá nazvat jako rozšíření semistrukturované domény pojmu sloupcové nebo doménové komprese. Motivace pro toto seskupení spočívají ve stejnosti dat ve stejných skupinách. Tato data mají obvykle velice podobné vlastnosti a díky tomu se dají lépe komprimovat. Poté je na každý kontejner použit jiný vhodný kompresor. Všechny výstupy jsou v poslední řadě komprimovány pomocí kompresoru GZip. Pro implementaci je soubor parsován SAX8 parserem [17].

Podle určitých studií se ukázalo, že XMill nabízí zlepšení komprese, na rozdíl od obyčejného GZip, který zachází s celým souborem jako s nepřetržitým proudem bajtů a s obsahem nesdružuje žádnou sémantiku [18].

[9] Používá se k modelování vztahů mezi elementy, atributy atd.



Obrázek 10: Princip XMill (Zdroj: 40, upraveno)

V momentě, kdy Xmill používá GZip, nově nastoupil tým z Indie, který pro navýšení efektivity komprese přidává do nástroje XMill knihovnu 7zip. Tato knihovna ke kompresi používá algoritmus LZMA (3.5.5), který je vylepšením již existujícího algoritmu LZ77 (Lempel-Ziv 1977) následujícími prvky: pro odstranění duplicitních dat využívá 4 GB, namísto 32 kB, ovšem tato velikost slovníku zapříčiňuje zpomalení algoritmu. Používá optimální parsování pro kratší kódy nedávno opakovaných shod. Místo hladového[10] přístupu je využíván přístup "look-ahead"[11] a používá zpracování kontextu [18].

XMill má procesor cesty, který má na starosti mapování datových hodnot do kontejnerů. Procesor zkontroluje každou z těchto cest proti kontejneru a určí, jestli hodnota bude uložena v daném kontejneru, nebo vytvoří pro tento účel nový kontejner. Tyto kontejnery jsou ukládány do paměti o velikosti 8 MB. Po zaplnění této paměti jsou všechny uložené kontejnery zpracovány algoritmem LZMA. Po zpracování těchto kontejnerů jsou kontejnery uloženy na disk a komprese pokračuje [18].

3.5.5 LZMA

LZMA (Lempel-Ziv-Markov Chain Algorithm) vznikl v roce 1998 a byl navržen panem Igorem Pavlovem. Skládá se ze dvou algoritmů LZ77 a algoritmu, který je založený na kódování rozsahu [34].

[10] Technika, která pro minimalizaci dat využívá co nejvíce prostředků (např. paměť)
[11] Technika, která předpovídá a analyzuje následující symboly

Funguje na stejné bázi jako většina ostatních kompresorů. Nahrazuje často se opakující řetězce slov ukazateli na předchozí výskyty. Vypočítá odhady frekvence, které jsou dané sadou symbolů nebo podřetězců. Poté se aplikují výsledné odhady k nalezení shod [35].

3.6 Dotazovací kompresory

Na rozdíl od nedotazovacích kompresí podporují vyhodnocování dotazů. Hlavním cílem těchto typů kompresí je dosáhnout kvalitního kompresního poměru co nejbližšího nedotazovacím kompresorům a zajistit přitom efektivní vyhodnocení dotazů.

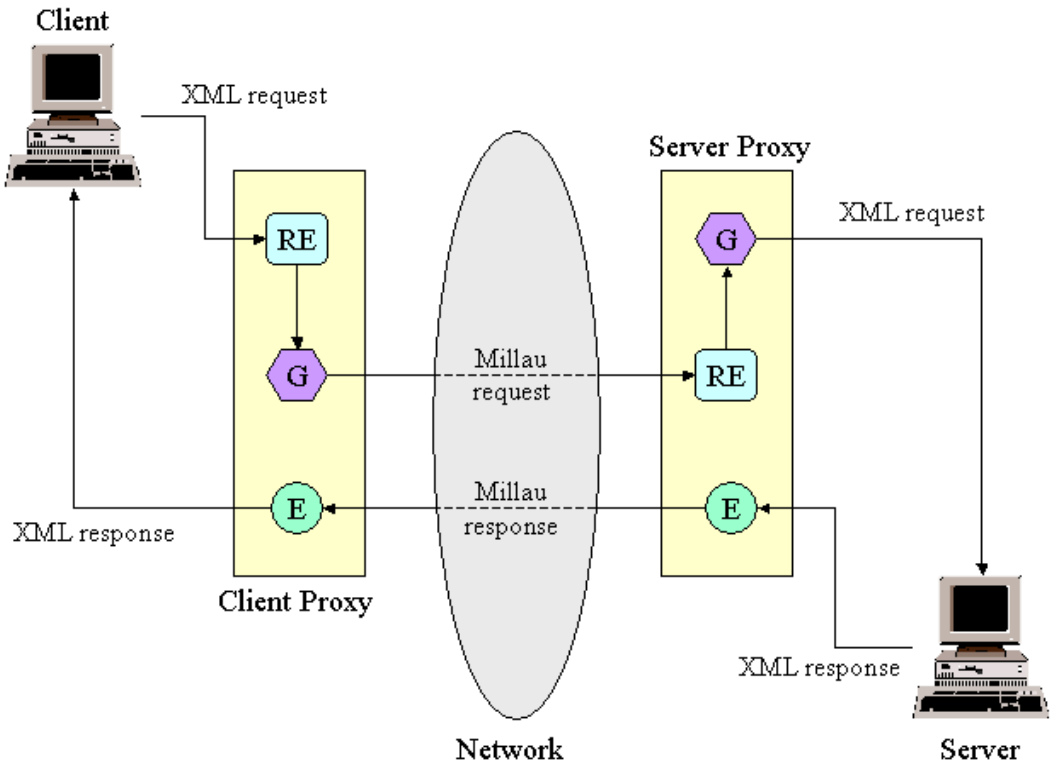
3.6.1 Millau

Je rozšířením formátu WAP Binary[12]. Specifikace WBXML (Wireless Application Protocol Binary XML) slouží k definici binární reprezentace XML. WBXML zmenšuje celkovou přenosovou velikost XML dokumentu bez toho, aniž by přišel o funkcionalitu, a také zachovává strukturu elementů, což pomáhá prohlížeči k přehlížení neznámých elementů. Millau je kódovací formát, který umožnuje efektivní kódování a streamování XML dokumentů [30].

Sady kompresních a kódovacích technik, které slouží pro kompresi XML, jsou obsaženy v tomto systému. Millau využívá techniku, která nese název diferenciální DTD (Distributed Data-structures for Text processing) stromová komprese, jež napomáhá k usnadnění komprese tím, že si bere informace z DTD. DDT kóduje jen informace, které není možné odvodit z DDT, jako jsou informace
o struktuře či datové hodnoty [29].

Je třeba speciálního algoritmu, aby mohly být parsovány DTD a zároveň DDT stromy. V tomto algoritmu mají uzly zpracovány ve stromech stejnou sémantiku. Algoritmus prozkoumá strukturní informace dokumentu, který je odeslán do toku. Do obsahového toku jsou odeslány datové hodnoty vstupu dokumentu a poté je komprimován [29].

[12] Formát pro přenos dat mezi mobilním zařízením a serverem



Obrázek 11: Cesta a odpověď webového požadavku (Zdroj: 30)

3.6.2 Xaust

Jedná se o kompresor, který je online a je závislý na schématu. Používá znalosti z DTD schématu, který určuje pravidla syntaxe. Kompresor může předvídat další prvek, který má být komprimován nebo dekomprimován. Zajímavá věc přichází při dekompresi, kdy stačí, aby si kompresor a dekompresor sdílely DTD schema a kompresor pro navrácení dokumentu do původního stavu zakódoval pouze část, kterou pomocí DTD nelze odvodit.

Pro kompresi dat využívá Xaust aritmetický algoritmus. Každý element může být transformován do DFA[13] (Deterministic Finite Automaton). Jednotlivé přechody jsou označeny jménem elementu. Xaust seskupuje všechna data pro stejný element a umístí je do kontejneru. Kontejnery jsou poté kompresovány aritmetickou metodou řádu 4. Díky DTD schématu je Xaust schopný předpovídat očekávané symboly, jelikož může sledovat strukturu dokumentu [23].

[13] Automat pro vyhledávání během procesů komprese

3.6.3 XGrind

XGrind je podobný ostatním kompresorům, ale na rozdíl od nich umožňuje dotazování na kompresovaná data. Tato funkce je užitečná při práci s databázemi. XGrind také zachovává původní strukturu kompresovaných dat. Tato možnost umožnuje XML opakovaně zpracovat daný soubor, tomuto se říká „homomorfní komprese“ [19].

XGrind pro kompresi využívá různé metody pro struktury, atributy výčtového typu či data a řídí se tak stejným přístupem jako XMill. U struktur je každý počáteční tag nahrazen písmenem ´T´, označen jedinečným ID prvkem, a u atributů je stejný postup, jen se mění označení na ´A´. Uzavírající tagy jsou nahrazeny znakem ´/´. Hodnoty výčtového typu jsou kompresovány pomocí DTD a zakódovány pomocí jednoduchého ()+*-, k reprezentaci výčtových hodnot. Pokud je v souboru atribut definován jako výčtový typ, tak jsou kompresovány za pomoci metody kódování slovníku. Zbytek dat v souboru je kompresován za pomoci Hoffmanovy metody [19].

3.6.4 XQzip

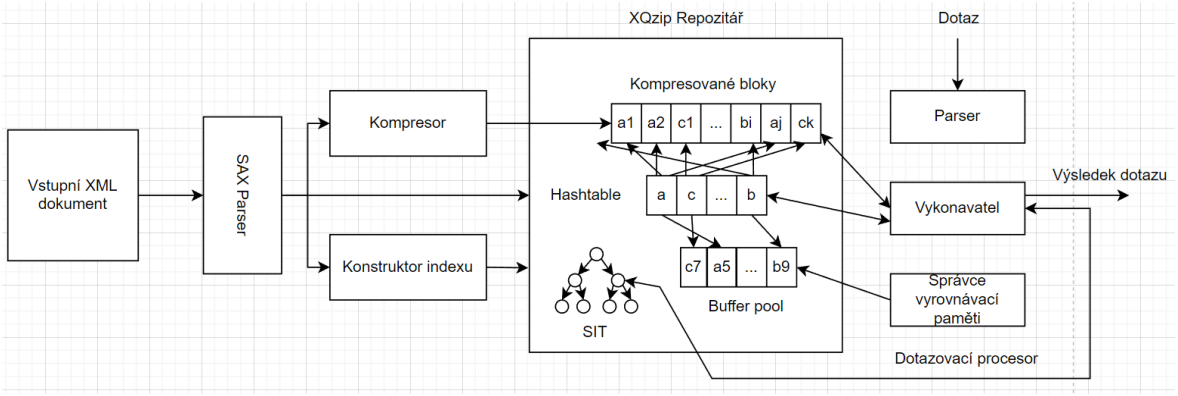
XQzip je nehomomorfní komprese a podporuje přímé vyhodnocování dotazů. Využívá sub stromy při kompresi. Je zde využívána metoda SIT (Structured Index Tree). SIT umožnuje odstranění duplicitních struktur, zlepšení výkonnosti dotazů a především udržuje informace o struktuře dokumentu [32, 23].

XQzip nedekomprimuje celý dokument, ale nejprve dokument rozdělí do bloků a dekomprimuje jeden blok po druhém. Účinně snižuje režii dekomprese vyhodnocování dotazů, využitím „buffer pool“ pro dekomprimované bloky dat XML. Jednotlivé bloky jsou přístupné pomocí „Hashtable“[14], kde jsou názvy elementů či atributů uloženy.

Je využíván index pro dotazování komprimovaných dat. Podporuje velkou část dotazů XPath.

Proces dotazování „Query Parser“ parsuje vstupní dotaz a následně používá „Query Executor“, který používá index k vyhodnocení dotazů. Query Executor uplatňuje pravidlo LRU (Least Recently Used) pro správu „Buffer pool“[15] pro dekomprimované bloky [32].

[14] Struktura pro rychlé vyhledávání v slovníku
[15] Technika správy paměti



Obrázek 12: Princip XQzip komprese (Zdroj: 32)

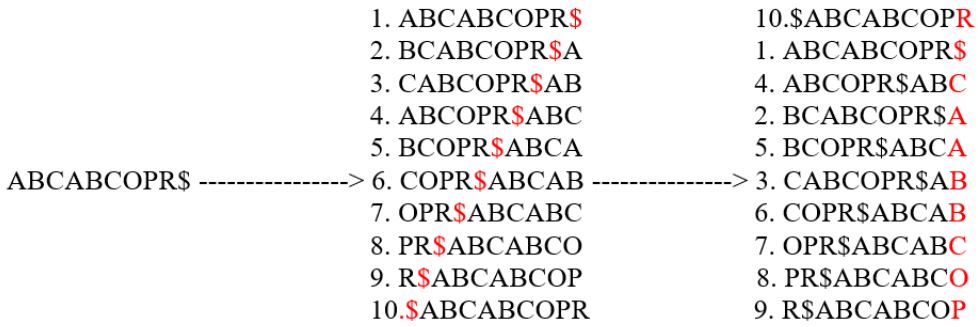
3.7 Algoritmy

Každý z kompresorů využívá různé algoritmy, transformace či kódování k zmenšení vstupních souborů.

3.7.1 Burrows-Wheleer Transform (BWT)

Algoritmus byl vyvinut v roce 1983 Michaelem Burrowsem a Davidem Wheelerem. Jedná se o blokový kompresní algoritmus [22]. Používán je například kompresorem BZip a patří mezi bezeztrátové komprese. Výsledkem jsou řetězce stejných znaků, které se dají jednoduše uložit v kompaktní formě [21].

BWT seřazuje vstupní textový řetězec lexograficky (abecedně) a poté vezme poslední sloupec z výstupu, což představuje BWT. Algoritmus je vysvětlen na následujícím obrázku 13 [21].

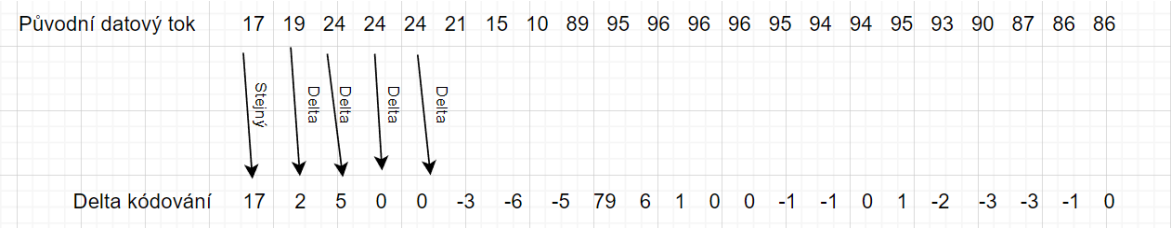


Obrázek 13: Princip Burrows-Wheleer Transofrm (Zdroj: 41, upraveno)

Dolar na konci značí první písmeno, jelikož znaky jsou v abecedě na začátku, a také by se neměl vyskytovat ve výstupním sloupci. Algoritmus si nejprve vezme jednotlivé znaky a utvoří z nich všechny možné variace. Následně přichází na řadu lexografické seřazení, které vždy vezme první znak abecedy, což je v tomto případě $. Podívá se, jestli je zde ještě jeden, a pokud ne, přesune se na další. Další v pořadí je A, nacházejí se 2 verze, co začínají na A. V tomto případě se orientuje podle druhého znaku, a pokud jsou znaky stejné, tak podle třetího atd. Po tomto utřídění se v posledním řádku vezmou všechny znaky, což v tomto případě jsou R$CAABBCOP a zde se použije jednodušší zápis: R$C2A2BCOP. Čím více se nachází stejných znaků v textovém řetězci, tím efektivněji BWT komprimuje [21].

3.7.2 Delta encoding

Je využívána převážně při vysokém stupni redundancí mezi referenčními soubory a cílovými soubory. Pro lepší kompresi tedy využívá redundance mezi páry nebo skupinami. Tato metoda zaručuje výrazně efektivnější komprimaci souboru. Vyžaduje, aby kodér měl úplnou znalost o referenčních souborech. Vývoj delty byl zaměřen na kompresi textových a binárních souborů [31]. Na obrázku 14 je vidět princip kódování.



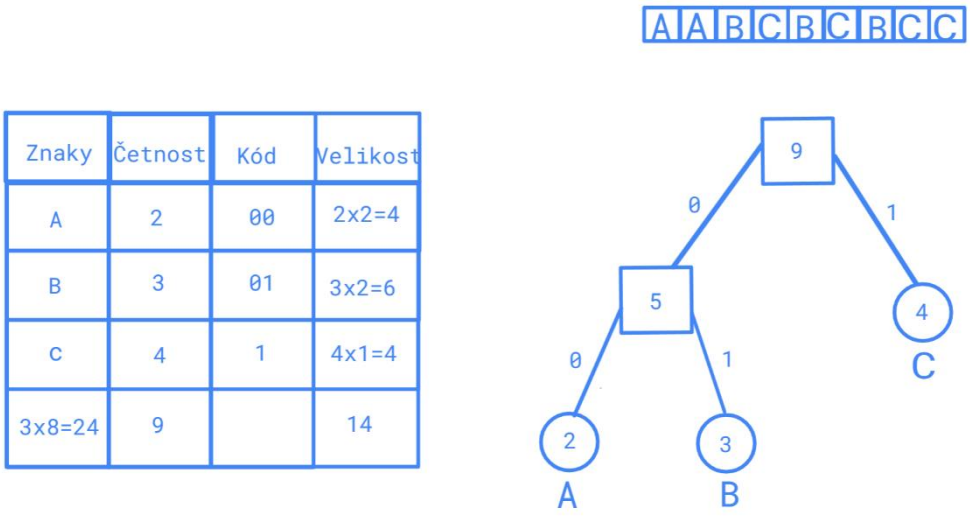
Obrázek 14:Princip Delta encoding (Zdroj: 42, upraveno)

3.7.3 Huffman encoding

Huffmanův algoritmus byl vytvořen v 19. století Davidem Huffmanem. Jak bylo výše uvedeno (3.4.1), používá se pro bezeztrátovou kompresi. Základním principem je kódovat a kompresovat data v závislosti na frekvenci znaků nacházejících se v souboru [27].

Fungování algoritmu je založeno na přiřazení kódu proměnné délky vstupním znakům na základě jejich odpovídající frekvence. Nejvíce se vyskytující znak obdrží

nejmenší kód a nejméně se vyskytující zase největší kód. Tyto přiřazené kódy se nazývají „prefixové kódy“, které zaručují jedinečnost přiřazeného kódu. Právě díky této metodě při dekódování nedochází k nejednoznačnosti. Následující obrázek 15 slouží k objasnění [27].



Obrázek 15: Princip Huffmanova algoritmu (Zdroj: 43, upraveno)

Každý z těchto znaků potřebuje 8 bitů. Pro odeslání tohoto dokumentu je tedy zapotřebí 72 bitů. Algoritmus nejprve spočítá četnost jednotlivých znaků obsažených v dokumentu. Následně jsou data seřazena vzestupně podle četnosti znaků. Vytvoří stromovou strukturu z přechozího seřazení a prázdný uzel, který bude součtem dvou nejmenších četností. Poté se proces zopakuje. Když je strom hotový, tak se doplní binární hodnoty na jednotlivé větve stromu, vlevo je doplněna 0 a vpravo 1. V tabulce jsou viditelné finální hodnoty a součet bitů, který činí 47 [27].

3.7.4 Dictionary encoding

Tato metoda přidá každému jedinečnému slovu v dokumentu celočíselnou hodnotu, takže každé z těchto slov je zakódováno jedinečnou celočíselnou hodnotou. Například tedy pokud jsou vstupní data napsaná takto: ‚kočky seděly vedle kočky‘, tak se mezi ně rozdělí hodnoty ‚1‘, ‚2‘ a ‚3‘. Díky tomuto rozdělení se data komprimují jako „1 2 3 1“, jelikož slovo ‚kočky‘ se vyskytuje dvakrát, tak číslo ‚1‘ je obsažené také dvakrát. Obecně celočíselná čísla, která jsou metodou přiřazena, neuchovávají pořadí mezi daty. Kvůli tomu

byla navržena metoda ALM, a ta zachovává pořadí mezi slovy. Mimo metodu ALM existují i jiné dictionary encoding metody, jako jsou LZ77 a LZ78 [23].

Tato metoda je velmi efektivní pro vstupní data, která mají velký výskyt opakujících se slov, jelikož základní kompresní jednotkou dictionary encoding je slovo. V případě menšího výskytu opakujících se slov tato metoda může být časově náročná, protože slovník obsahuje velký počet slov. Mimo jiné by bylo zapotřebí více místa [23].

4 Vlastní práce

Vlastní práce spočívala v seznámení se s individuálními kompresními mechanismy a následném vybrání některých z nich. Poté proběhlo měření komprese zvolených vstupních XML souborů na základě předem zvolených hodnotících kritérií. Z jednotlivých výsledků těchto měření byly syntetizovány závěry.

4.1 Příprava na měření

Pro měření komprese bylo zapotřebí určit soubory XML, které budou pro tyto záměry použity, tak, aby vyhovovaly jednotlivým kritériím měření. Nedílnou součástí měření byl hardware, na kterém bylo provedeno testování. Využity byly notebooky v osobním vlastnictví, které zastoupily nižší a střední třídu, a na náročnější testování byl obstarán počítač vyšší třídy.

V rámci dostupnosti byly vybrány kompresní mechanismy, kterými byly soubory komprimovány. Na výběr byla široká škála programů obsahující kompresní mechanismy, např. WinRAR, JZip. 7Zip a Bandizip. Pro účely měření byl zvolen program 7Zip, který obsahoval nezbytné kompresní mechanismy a mimo jiné je open source.

Pro určení účinnosti jednotlivých kompresních mechanismů a porovnání jejich výkonů byla zvolena hodnotící kritéria. Tato kritéria nabízejí objektivní podklad pro hodnocení efektivity kompresních mechanismů. Z individuálních měření byly syntetizovány výsledky ve formě tabulek, které byly následně popsány.

4.1.1 Použité XML soubory

Základním stavebním kamenem pro provedení měření bylo vybrání určitých XML souborů. Jelikož v pozdější fázi byly provedeny zátěžové testy na hardware, potřebné soubory se liší především velikostí před komprimací. Veškeré soubory byly použity z veřejně dostupných webových stránek vlády Spojených států amerických.

Názvy vybraných souborů a jejich velikosti před komprimací jsou uvedeny v tabulce 1. Velikosti souborů jsou seřazeny vzestupně a jsou uváděny v bajtech pro co nejpřesnější reprezentaci.

Název souboru	Velikost souboru
Vehicle, Snowmobile, and Boat Registrations	9 130 687 921 B
U.S. Chronic Disease Indicators	1 284 070 975 B
Health Care Provider Credential Data	1 136 868 518 B
Police Department Incident Reports 2018 to Present	894 252 637 B
Crime Data from 2020 to Present	570 228 647 B
U.S. COVID-19 Self-Test Data	307 823 316 B
Active Businesses - County Data	230 695 941 B
Oregon Active Workers' Compensation Employer Database	147 493 711 B
Electric Vehicle Population Data	93 228 248 B
Heart Disease Mortality Data Among US Adults (35+)	55 725 227 B
Power Outages - County	17 754 394 B
Voter Registration Data	8 603 486 B
ABS Store Inventory and Sale Items	4 680 906 B

Tabulka 1: Použité XML soubory (Zdroj: 44)

U všech souborů byl změřen čas komprese i dekomprese, ovšem pro měření zátěže hardwaru bylo vybráno prvních jedenáct souborů pro měření zátěže procesoru a prvních devět souborů pro měření zátěže operační paměti. Těchto souborů v tomto typu měření je využito díky jejich větší velikosti. U tohoto měření se tedy hodnotí zátěž při kompresi a dekompresi. Zbývající XML soubory nebyly zahrnuté do měření zátěže na hardware, jelikož byla zjištěna jejich malá velikost, což znamená, že doba komprese je příliš krátká pro provedení měření zátěže.

4.1.2 Použitý hardware

Pro jednotlivá měření byl použít počítač střední třídy, který postačil pro účely měření. Klíčovými komponenty pro tuto úlohu byly procesor (CPU) a operační paměť (RAM), které hrají primární roli při řízení a zpracování kompresních algoritmů. Navíc, jelikož se kompresované soubory ukládají na pevný disk, tak i tento komponent je využíván při této operaci, ovšem nebyl začleněn do měření. Komponenty a parametry počítače střední třídy jsou uvedeny v tabulce č. 2.

Komponenty	Parametry
Procesor	AMD Ryzen 5 5600H, 6 jader
RAM	16GB
Grafická karta	RTX 3060 Laptop, 6GB
Disk	1TB SSD
Operační systém	Windows 10/Pro

Tabulka 2: Parametry počítače střední třídy (Zdroj: vlastní zpracování)

Jakožto objektivní hodnocení náročnosti komprese na různý hardware byl vybrán počítač nižší třídy, jehož komponenty jsou obsaženy v tabulce č. 3. Při výběru tohoto počítače se především bral ohled na horší procesor a nižší operační paměť. Zároveň počítač obsahuje novější verzi operačního systému Windows.

Komponenty	Parametry
Procesor	Lenovo Core i3-1115G4, 3GHz
RAM	8GB
Grafická karta	Intel UHD
Disk	512GB SSD
Operační systém	Windows 11/Pro

Tabulka 3: Parametry počítače nižší třídy (Zdroj: vlastní zpracování)

Pro náročnější měření, především pro kompresní metody LZMA a LZMA2, byl použit hardware z vyšší třídy.

Komponenty	Parametry
Procesor	Intel Core i7 10700F, 2,9 GHz
RAM	64GB
Grafická karta	RTX 3060, 12GB
Disk	1TB HDD, 500GB SSD
Operační systém	Windows 10

Tabulka 4: Parametry vyšší třídy (Zdroj: vlastní zpracování)

4.1.3 Vybrané kompresní mechanismy

Kompresní mechanismy byly především vybrány díky jejich dostupnosti. Všechny tyto mechanismy obsahuje aplikace 7Zip, ve které taky byly komprese vykonávány. Testováno bylo i více programů, např. Bandzip a IZArc, ovšem žádný z nich neobsahoval tolik nastavení a kompresních mechanismů jako aplikace 7Zip. V aplikaci byly vybrány následující kompresní mechanismy:

- GZip
- BZip2
- LZMA
- LZMA2
- PPMd
- Deflate64

Individuální kompresní mechanismy byly obsaženy v různých formátech archivu jako jsou 7Z a zip. Zvoleným byl převážně formát 7z, díky jeho nízkému výslednému kompresnímu poměru. Kompresní mechanismus Deflate64 nebyl obsažený v archivním formátu 7z, proto pouze v tomto případě byl využit formát zip.

Naneštěstí u ostatních kompresních mechanismů, které byly uvedeny v kapitole 4.7 a 4.8, nebyly nalezeny žádné aplikace a ani zdrojové kódy, tudíž nemohly být součástí měření.

4.1.4 Hodnotící kritéria a vstupní nastavení

Výběr hodnotících kritérií a vstupních nastavení je klíčovým faktorem, který slouží k zajištění spolehlivosti a objektivnosti výsledků měření. Bylo nutno vybírat na základě efektivnosti a dopadů těchto kritérií na výslednou kompresi dat. Všechna kritéria a vstupní nastavení byla otestována a jejich dopad na kompresi zhodnocen.

Jedním z hlavních zvolených vstupních nastavení kompresních mechanismů byla úroveň komprese. Ta představuje „míru redukce velikosti souboru“. Dále byla zvolena velikost slovníku, u té platí: čím větší je nastavená velikost slovníku, tím větší je poté kompresní poměr, ale současně se tímto zvyšuje spotřeba paměti. Kompresní poměr je udáván v procentech a představuje podíl mezi velikostí souboru před kompresí a výslednou velikostí po kompresi. V neposlední řadě byla zvolena komprimační metoda (kompresní mechanismus). Některé kompresní mechanismy umožňovaly nastavení velikosti slova a u těchto kompresních mechanismů bylo také zahrnut do součásti měření.

V kategorii hodnotící kritéria byla zvolena doba trvání komprese a dekomprese, tudíž čas, za který komprese či dekomprese proběhne. Dále byl zvolen kompresní poměr, jehož výpočet je uveden v kapitole 4.2.4 a také byla zvolena zátěž komprese na různý hardware. Hodnotil jsem:

- vytíženost operační paměti
- vytíženost procesoru

4.2 Provedená měření

Prvním cílem měření bylo určit vstupní nastavení jednotlivých kompresních mechanismů. Pro tento požadavek byly provedeny testy, které zhodnocovaly, jaká nastavení

mají vliv na výsledný kompresní poměr, tak aby byl co nejnižší. V této kategorii byla hodnocena úroveň komprese, velikost slovníku, velikost bloku, velikost slova a formát archivu. Téměř všechna vybraná nastavení měla vliv na výsledný kompresní poměr, kromě velikosti bloku. Zvolena byla pokaždé (kromě úrovně komprese) co nejvyšší možná hodnota.

V tabulce č. 5 jsou uvedena vybraná vstupní nastavení každého kompresního mechanismu. Veškerá pozdější měření byla prováděna s těmito uvedenými hodnotami. Označení znakem „x“, znamená, že u případného kompresního mechanismu nebylo možné zvolit dané nastavení.

Následně v tabulce č. 6 jsou znázorněna provedená měření na dvou velikostně odlišných souborech. Pokaždé bylo dbáno na odlišnost jednotlivých nastavení, hlavně u velikosti slovníku a velikosti slova. Úroveň komprese neměla buď žádný vliv na výsledný kompresní poměr, anebo minimální, ovšem velice ovlivňovala dobu trvání komprese, tudíž pro všechny kompresní mechanismy byla zvolena úroveň „nejrychlejší“.

Veškerá měření byla provedeno na počítači střední třídy, pozdější testování pro metody LZMA a LZMA2 bylo provedeno, na počítači vyšší třídy, a to kvůli jejich možnosti nastavení velikosti slovníku. Velikost slovníku je velice náročná na operační paměť a pro zvolení nejvyšších hodnot bylo za potřebí 64 GB operační paměti. V tomto ohledu bylo zohledněno i maximální možné vstupní nastavení u počítače nižší třídy.

Kompresní mechanismus	Úroveň komprese	Velikost slovníku	Velikost slova	Formát archivu
BZip2	Nejrychlejší	900 KB	x	bzip2
Gzip	Nejrychlejší	32 KB	258	gzip
LZMA	Nejrychlejší	768 MB	273	7z
LZMA2	Nejrychlejší	768 MB	273	7z
PPMd	Nejrychlejší	1024 MB	32	7z
Deflate64	Nejrychlejší	64 KB	257	zip

Tabulka 5: Použitá vstupní nastavení pro jednotlivé kompresní mechanismy (Zdroj: vlastní zpracování)

Název souboru	Formát archivu	Úroveň komprese	Velikost slovníku	Čas komprese	Komprimační metoda	Zátěž procesoru	Zátěž Ram	Velikost	Kompresní poměr	Velikost slova
Active Businesses - County Data	zip	Nejrychlejší	100 KB	3s	Bzip2	84%	21,2 MB	26 440 835	11,00%	x
	zip	Nejrychlejší	500KB	4s	Bzip2	89%	64,3 MB	24 964 696	9,00%	x
	zip	Nejrychlejší	900KB	7s	Bzip2	88,80%	105 MB	20 888 773	9,00%	x
	bzip2	Nejrychlejší	900KB	7s	Bzip2	88,80%	106 MB	20 888 773	9,00%	x
	7z	Nejrychlejší	900KB	7s	Bzip2	88,00%	101,6 MB	20 888 951	9,00%	x
Vehicle, Snowmobile, and Boat Registrations	zip	Nejrychlejší	100 KB	2m 9s	Bzip2	90,00%	21.1 MB	698 490 739	7,00%	x
	zip	Nejrychlejší	500KB	3m 57s	Bzip2	90,70%	63,4 MB	556 121 322	6,00%	x
	zip	Nejrychlejší	900KB	5m 47s	Bzip2	90,90%	104,8 MB	524 424 247	5,00%	x
	bzip2	Nejrychlejší	900KB	5m 16s	Bzip2	94,00%	100,6 MB	524 424 247	5,00%	x
	7z	Nejrychlejší	900KB	5m 24s	Bzip2	92,00%	100,6 MB	524 424 247	5,00%	x
Active Businesses - County Data	gzip	Nejrychlejší	32KB(uzamčeno)	1s	Deflate (uzamčeno)	2,40%	4,4 MB	31 800 598	13,00%	32
	gzip	Nejrychlejší	32KB(uzamčeno)	2s	Deflate (uzamčeno)	5,30%	4,4 MB	31 099 264	13,00%	128
	gzip	Nejrychlejší	32KB(uzamčeno)	2s	Deflate (uzamčeno)	5,30%	4,4 MB	31 022 401	13,00%	258
Vehicle, Snowmobile, and Boat Registrations	gzip	Nejrychlejší	32KB(uzamčeno)	42s	Deflate (uzamčeno)	11,00%	3,4 MB	843 585 792	9,00%	32
	gzip	Nejrychlejší	32KB(uzamčeno)	55s	Deflate (uzamčeno)	11,90%	3,4 MB	816 530 832	8,00%	128
	gzip	Nejrychlejší	32KB(uzamčeno)	1m 4s	Deflate (uzamčeno)	11,70%	3,4 MB	810 599 099	8,00%	258
Active Businesses - County Data	7z	Nejrychlejší	32 MB	6s	PPmD	11,30%	35,8 MB	21 946 420	9,00%	4
	7z	Nejrychlejší	128 MB	9s	PPmD	12,20%	131,8 MB	16 552 411	7,00%	16
	7z	Nejrychlejší	1024 MB	15s	PPmD	11,80%	1027,8 MB	15 472 657	6,00%	32
	zip	Nejrychlejší	16 MB	5s	PPmD	11,80%	23,7 MB	21 652 120	9,00%	4
	zip	Nejrychlejší	64 MB	8s	PPmD	11,80%	71,8 MB	17 672 576	7,00%	8
	zip	Nejrychlejší	256 MB	11s	PPmD	11,80%	263,9 MB	16 368 431	7,00%	16
Vehicle, Snowmobile, and Boat Registrations	7z	Nejrychlejší	32 MB	3m 1s	PPmD	11,90%	35,4 MB	658 826 974	7,00%	4
	7z	Nejrychlejší	128 MB	4m 9s	PPmD	11,90%	131,7 MB	413 578 670	4,00%	16
	7z	Nejrychlejší	1024 MB	6m 45s	PPmD	12,20%	1 027,7 MB	381 413 006	4,00%	32
	zip	Nejrychlejší	16 MB	2m 54s	PPmD	11,70%	23,8 MB	646 425 118	7,00%	4
	zip	Nejrychlejší	64 MB	3m 42s	PPmD	11,70%	71,8 MB	454 323 937	4,00%	8
	zip	Nejrychlejší	256 MB	4m 58s	PPmD	12,20%	263,7 MB	410 478 190	4,00%	16
Active Businesses - County Data	zip	Nejrychlejší	64KB(Uzamčených)	1s	Deflate64	3,20%	7,6MB	30 663 304	13,00%	32
	zip	Nejrychlejší	64KB(Uzamčených)	2s	Deflate65	3,20%	7,6MB	29 852 244	12,00%	128
	zip	Nejrychlejší	64KB(Uzamčených)	2s	Deflate66	8,30%	7,6MB	29 758 544	12,00%	257
Vehicle, Snowmobile, and Boat Registrations	zip	Nejrychlejší	64KB(Uzamčených)	43s	Deflate67	11,20%	7,6MB	825 650 118	9,00%	32
	zip	Nejrychlejší	64KB(Uzamčených)	58s	Deflate68	11,60%	7,6MB	793 863 465	8,00%	128
	zip	Nejrychlejší	64KB(Uzamčených)	1m 13s	Deflate69	11,60%	7,6MB	785 566 230	8,00%	257
Active Businesses - County Data	7z	Nejrychlejší	64 MB	7s	LZMA	11,90%	418,3 MB	24 415 467	10,00%	32
	7z	Nejrychlejší	256 MB	20s	LZMA	11,90%	1350,6 MB	23 414 924	10,00%	128
	7z	Nejrychlejší	768 MB	32s	LZMA	12,10%	1355 MB	23 165 814	10,00%	273
	zip	Nejrychlejší	64 MB	7s	LZMA	11,00%	422,3 MB	24 415 482	10,00%	32
	zip	Nejrychlejší	256 MB	18s	LZMA	11,40%	1353 MB	23 414 939	10,00%	128
	zip	Nejrychlejší	768 MB	32s	LZMA	11,70%	1358 MB	23 165 829	10,00%	273
Vehicle, Snowmobile, and Boat Registrations	7z	Nejrychlejší	64 MB	2m 58s	LZMA	11,90%	418,2 MB	661 888 729	7,00%	32
	7z	Nejrychlejší	256 MB	9m 5s	LZMA	12,00%	1 666,2 MB	609 744 456	6,00%	128
	7z	Nejrychlejší	768 MB	20m 33s	LZMA	12,00%	5 250,2 MB	596 248 151	6,00%	273
	zip	Nejrychlejší	64 MB	2m 52s	LZMA	11,70%	422,3 MB	661 888 744	7,00%	32
	zip	Nejrychlejší	256 MB	9m 42s	LZMA	11,70%	1 670,3 MB	609 744 471	6,00%	128
	zip	Nejrychlejší	768 MB	19m 34s	LZMA	14,00%	5 254,2 MB	596 248 166	6,00%	273
Active Businesses - County Data	7z	Nejrychlejší	64 MB	11s	LZMA2	5,50%	418,3 MB	24 420 203	10,00%	32
	7z	Nejrychlejší	256 MB	26s	LZMA2	5,30%	1351 MB	23 419 456	10,00%	128
	7z	Nejrychlejší	768 MB	41s	LZMA2	5,60%	1341 MB	23 170 300	10,00%	273
Vehicle, Snowmobile, and Boat Registrations	7z	Nejrychlejší	64 MB	44s	LZMA2	55,60%	7 130,9 MB	663 136 781	7,00%	32
	7z	Nejrychlejší	256 MB	2m 32s	LZMA2	28,80%	9 342,3 MB	614 141 853	6,00%	128
	7z	Nejrychlejší	768 MB	27m 4s	LZMA2	7,60%	5 250,3 MB	596 363 650	6,00%	273

Tabulka 6: Testování jednotlivých vstupních nastavení (Zdroj: vlastní zpracování)

4.2.1 Čas komprese

Čas komprese může být kritickým rozhodnutím pro výběr správného kompresního mechanismu. Odvíjí se převážně od velikosti vstupního souboru a nastavení komprese individuálních kompresních mechanismů.

Soubory v tabulce č.7 jsou seřazeny podle velikosti, od největšího po nejmenší. Jednotlivé hodnoty v tabulce jsou uvedeny v sekundách, pro přesnější zobrazení. Z tabulky lze vyčíst, že jednoznačně nejkratší dobu komprese provede kompresní metoda GZip, která dominovala u všech čtrnácti souborů. Hned na druhé příčce je metoda Deflate64, ta dosáhla u pěti souborů stejného času komprese jako GZip. Rozdíl mezi těmito dvěma kompresními metodami není tak markantní jako u ostatních metod. Nejhůře je na tom LZMA společně s LZMA2, kde komprese velkých souborů zabere čas v řádech desítek minut.

Při kompresi větších souborů (větších než 500 MB) je třeba dát důraz na použití vhodného kompresního mechanismu, který dosahuje nízkých hodnot doby komprese. V těchto případech se jednoznačně vyplatí použít GZip či Deflate64, které mají dobu komprese až 5x menší než ostatní testované metody.

U menších souborů (menších než 500 MB) naopak není kladen důraz na dobu komprese, jelikož je v řádech milisekund až několika desítek sekund. Je tedy možné u těchto souborů zvážit rozhodnutí spíše podle výsledků kompresních poměrů a zátěže na hardware.

Název souboru	BZip2	GZip	LZMA	LZMA2	PPMD	Deflate64
Vehicle, Snowmobile, and Boat Registrations	316 s	64 s	1233 s	1624 s	405 s	73 s
U.S. Chronic Disease Indicators	62 s	17 s	111 s	110 s	68 s	18 s
Health Care Provider Credential Data	40 s	32 s	373 s	389 s	141 s	35 s
Police Department Incident Reports 2018 to Present	35 s	20 s	179 s	175 s	74 s	23 s
Crime Data from 2020 to Present	20 s	13 s	115 s	118 s	56 s	16 s
U.S. COVID-19 Self-Test Data	12 s	5 s	36 s	36 s	26 s	4 s
Active Businesses - County Data	7 s	2 s	32 s	41 s	15 s	2 s
Oregon Active Workers' Compensation Employer Database	6 s	3 s	26 s	26 s	15 s	4 s
Electric Vehicle Population Data	4 s	2 s	9 s	9 s	7 s	2 s
Heart Disease Mortality Data Among US Adults (35+)	2 s	0,5 s	3 s	3 s	3 s	0,5 s
Power Outages - County	0,5 s	0,5 s	2 s	2 s	2 s	0,5 s
Voter Registration Data	0,2 s	0,2 s	0,5 s	0,5 s	1 s	0,2 s
ABS Store Inventory and Sale Items	0,2 s	0,1 s	0,5 s	0,5 s	1 s	0,2 s

Tabulka 7: Čas komprese (Zdroj: Vlastní zpracování)

4.2.2 Zátěž procesoru

Při měření zatížení procesoru byla omezena množina souborů, které byly podrobeny testování, pouze na prvních jedenáct. Toto omezení bylo způsobeno tím, že u ostatních souborů byla velikost příliš nízká a doba komprese příliš krátká na změření zátěže procesoru.

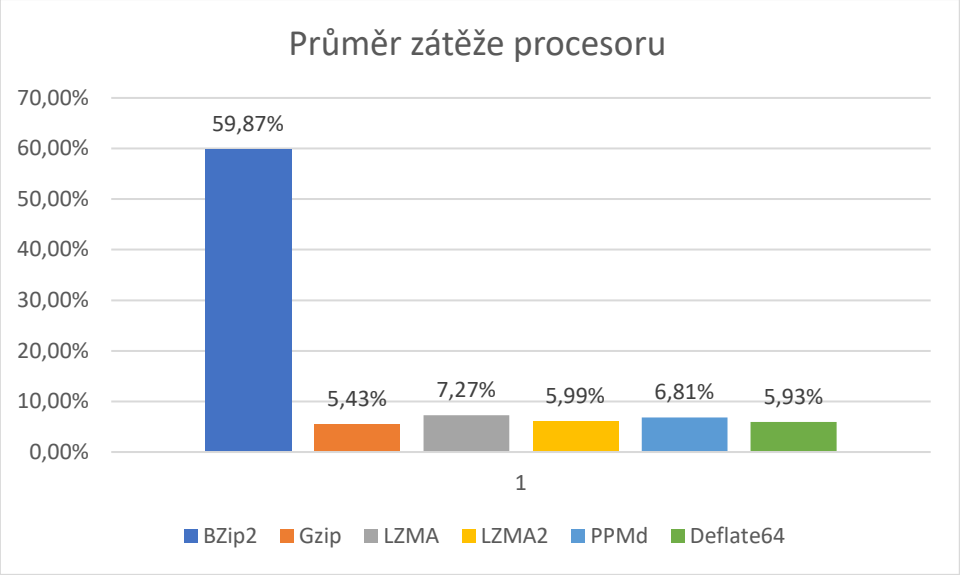
Z tohoto důvodu nebyly tyto soubory zahrnuty do výsledné tabulky. U souboru „Heart Disease Mortality Data Among US Adults“ u metody Deflate64 je uvedeno znaménko „x“, což znamená nemožnost změření.

Mezi tři metody s nejnižším využitím procesoru patří GZip, LZMA2 a Deflate64. Největší zátěž na procesor představovala metoda BZip2, která nejčastěji využívala okolo 50 %, ovšem u souborů „Vehicle, Snowmobile, and Boat Registrations“ a „Active Business – County Data“ využívala veškerou volnou kapacitu procesoru. Při následně opakované kompresi některých souborů bylo zjištěno, že BZip2 využíval veškerou volnou paměť procesoru u souborů, u kterých v minulosti využíval pouze cca 50 %. Bohužel nebyla zjištěna příčina těchto neshod v měření.

Název souboru	BZip2	Gzip	LZMA	LZMA2	PPMd	Deflate64
Vehicle, Snowmobile, and Boat Registrations	94,00%	11,70%	12,00%	7,60%	12,20%	11,60%
U.S. Chronic Disease Indicators	51,00%	5,10%	6,90%	5,60%	5,40%	5,20%
Health Care Provider Credential Data	55,20%	5,20%	7,30%	8,50%	5,90%	5,10%
Police Department Incident Reports 2018 to Present	55,30%	5,30%	5,90%	5,30%	5,50%	5,10%
Crime Data from 2020 to Present	55,10%	5,40%	5,60%	5,40%	5,80%	5,10%
U.S. COVID-19 Self-Test Data	53,70%	5,30%	7,10%	5,80%	5,50%	5,40%
Active Businesses - County Data	88,80%	5,40%	12,10%	5,60%	11,80%	8,30%
Oregon Active Workers' Compensation Employer Database	52,80%	5,50%	5,40%	5,50%	5,70%	5,20%
Electric Vehicle Population Data	51,50%	2,80%	5,60%	5,60%	5,80%	2,40%
Heart Disease Mortality Data Among US Adults (35+)	41,30%	2,60%	4,80%	5,00%	4,50%	x

Tabulka 8: Zátěž na procesor při kompresi (Zdroj: vlastní zpracování)

Jelikož jsou naměřené hodnoty v tabulce č. 8 až moc nejednotné, byl vytvořen graf s vypočítaným průměrem naměřených hodnot, který lépe shrnuje zátěž individuálních kompresních metod. Zde lze vidět, že v průměru nejméně zatěžuje procesor metoda GZip s hodnotou využití 5,43 %. Následující metody, které jsou téměř na stejné úrovni zátěže jako metoda GZip, jsou Deflate64 s hodnotou 5,93 % a těsně za ní LZMA2 s hodnotou 5,99 %. Tyto tři metody vykazují minimální zátěž procesoru. Podobně jsou na tom metody PPMd se zátěží 6,81 % a LZMA se zátěží 7,27 %. Na druhé straně, BZip2 s průměrnou naměřenou hodnotou 59,87 % jednoznačně nejvíce zatěžuje procesor.



Graf 1: Průměr zátěže procesoru při kompresi (Zdroj: vlastní zpracování)

4.2.3 Zátěž operační paměti

Během komprese XML souborů dochází k zátěži operační paměti, především při kompresi velkých souborů. Při procesu komprese dochází k načtení celého souboru do paměti a následné manipulaci s daty, toto může vést k velkému množství dočasných dat v paměti. Pokud je paměť plná, může dojít k pádu komprese.

V tabulce 9 jsou uvedeny nejvyšší naměřené hodnoty (v MB) pro jednotlivá provedená měření. Měření proběhlo pouze u devíti souborů z důvodu jejich dostatečné velikosti. U ostatních souborů kvůli jejich malé velikosti nebyla doba komprese dostatečně dlouhá na to, aby bylo možno změřit zátěž na operační paměť.

Nejnižší zátěže dosahuje kompresní mechanismus GZip s naměřenou zátěží 3,4 MB a s maximální zátěží 4,4 MB. U Deflate64 byla pro veškeré soubory naměřena konstantí hodnota 7,6 MB. Trochu hůře si vedla metoda BZip2 s nejnižší naměřenou hodnotou 100,3 MB a nejvyšší 101,8 MB. Při měření zátěže na operační paměť u metod LZMA, LZMA2 a PPMd hrála významnou roli doba trvání komprese. Důvodem je postupné navyšování zátěže odvíjející se od doby trvání celkové komprese. Proto u zmíněných metod nedosahují hodnoty u menších souborů tak vysokých hodnot, jako u těch větších. Lze konstatovat, že maximální hodnoty, kterých tyto kompresní mechanismy dosáhnou u velkých souborů, jsou: LZMA: 5 250,20 MB, LZMA2: 5 250,30 MB a PPMd: 1 027,80 MB.

Název souboru	BZip2	GZip	LZMA	LZMA2	PPMD	Deflate64
Vehicle, Snowmobile, and Boat Registrations	100,6 MB	3,4 MB	5250,2 MB	5250,3 MB	1027,8 MB	7,6 MB
U.S. Chronic Disease Indicators	100,3 MB	3,4 MB	5250,2 MB	5250,3 MB	1027,7 MB	7,6 MB
Health Care Provider Credential Data	101,8 MB	3,4 MB	5163,3 MB	5182,2 MB	1027,8 MB	7,6 MB
Police Department Incident Reports 2018 to Present	101,4 MB	3,4 MB	4951,1 MB	4719,2 MB	1027,8 MB	7,6 MB
Crime Data from 2020 to Present	100,7 MB	3,4 MB	3742,5 MB	3732 MB	1027,8 MB	7,6 MB
U.S. COVID-19 Self-Test Data	100,7 MB	3,4 MB	1970,6 MB	1969,2 MB	1027,8 MB	7,6 MB
Active Businesses - County Data	106 MB	4,4 MB	1355 MB	1341 MB	1027,8 MB	7,6 MB
Oregon Active Workers' Compensation Employer Database	101,4 MB	3,4 MB	940,1 MB	950,5 MB	1027,8 MB	7,6 MB
Electric Vehicle Population Data	101,6 MB	4,4 MB	551,3 MB	542,5 MB	506,5 MB	7,6 MB

Tabulka 9: Zátěž na operační paměť při kompresi (Zdroj: vlastní zpracování)

4.2.4 Kompresní poměr

Jedním z nejdůležitějších, ne-li nejdůležitějším výsledkem měření je kompresní poměr. Ten udává poměr mezi vstupní velikostí souboru a velikostí po provedení komprese. Kompresní poměr lze interpretovat dvěma způsoby:

- Výsledný kompresní poměr ukazuje, na kolik procent byl vstupní soubor zmenšen.
- Výsledný kompresní poměr ukazuje, o kolik procent byl vstupní soubor zmenšen.

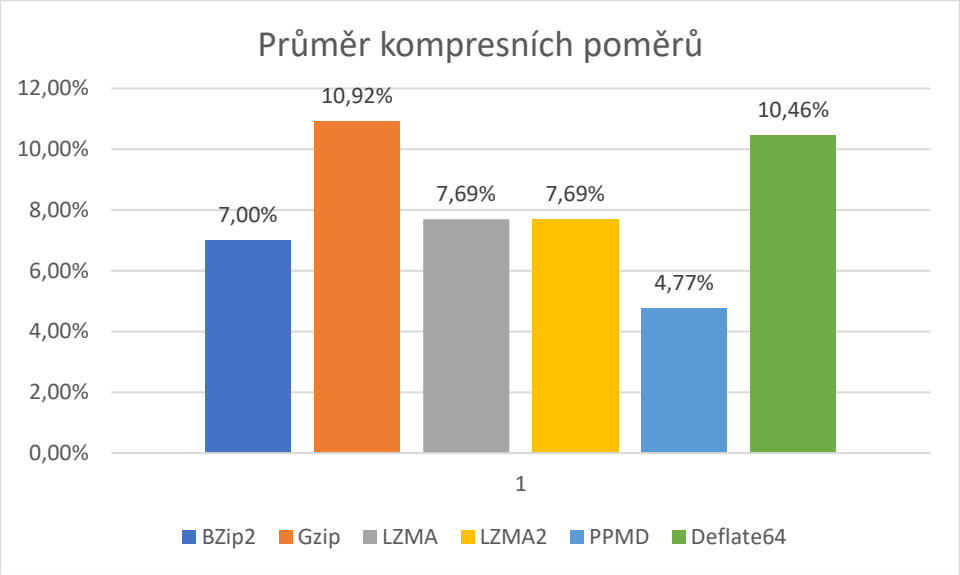
V této práci byl tedy použit kompresní poměr, který čím je nižší, tím úspěšněji kompresní mechanismus zvládl kompresi. Vzorec použitý pro vypočítání kompresního poměru byl „(1 - (velikost souboru před kompresí – velikost souboru po kompresi) / velikost souboru před kompresí) * 100 %“. Výsledkem je tedy na kolik % dat byl soubor zmenšen z původního vstupního souboru.

Výsledky kompresních poměrů jsou interpretovány v tabulce 10. Testováno bylo všech třináct souborů z tabulky 1. a veškeré uváděné hodnoty byly zaokrouhleny dolů na celá čísla. PPMd metoda dosáhla u všech třinácti souborů nejlepších kompresních poměrů. Metoda BZip2 dosáhla u deseti souborů lepších kompresních poměrů, něž metoda LZMA a LZMA2. Nejhoršími metodami pro docílení nízkého kompresního poměru jsou Deflete64 a GZip. Deflate64 dosáhl ale i tak lepších výsledků než GZip, a to hned u šesti souborů.

Název souboru	BZip2	Gzip	LZMA	LZMA2	PPMd	Deflate64
Vehicle, Snowmobile, and Boat Registrations	5,00%	8,00%	6,00%	6,00%	4,00%	8,00%
U.S. Chronic Disease Indicators	4,00%	7,00%	4,00%	4,00%	2,00%	6,00%
Health Care Provider Credential Data	9,00%	14,00%	10,00%	10,00%	6,00%	14,00%
Police Department Incident Reports 2018 to Present	7,00%	12,00%	5,00%	5,00%	4,00%	11,00%
Crime Data from 2020 to Present	7,00%	12,00%	8,00%	8,00%	4,00%	12,00%
U.S. COVID-19 Self-Test Data	6,00%	9,00%	8,00%	8,00%	5,00%	9,00%
Active Businesses - County Data	9,00%	13,00%	10,00%	10,00%	6,00%	12,00%
Oregon Active Workers' Compensation Employer Database	7,00%	13,00%	7,00%	7,00%	4,00%	12,00%
Electric Vehicle Population Data	6,00%	10,00%	6,00%	6,00%	4,00%	9,00%
Heart Disease Mortality Data Among US Adults (35+)	3,00%	6,00%	4,00%	4,00%	2,00%	6,00%
Power Outages - County	9,00%	12,00%	11,00%	11,00%	7,00%	12,00%
Voter Registration Data	6,00%	8,00%	7,00%	7,00%	5,00%	8,00%
ABS Store Inventory and Sale Items	13,00%	18,00%	14,00%	14,00%	9,00%	17,00%

Tabulka 10: Kompresní poměry (Zdroj: vlastní zpracování)

Jakožto jednodušší zobrazení byl použit průměr kompresních poměrů v grafu č.2, ze všech souborů pro individuální kompresní mechanismy. Nejlepších kompresních poměrů dosáhla metoda PPMd s průměrem kompresního poměru 4,77 %. Druhé místo patří metodě BZip2 s kompresním poměrem 7 %. Metody LZMA a LZMA2 po zaokrouhlení dosahují stejných hodnot, a proto bylo provedeno přesnější měření. Jednoznačně nejhůře si vedl GZip a Deflate64, které přesahují hranici 10 %.



Graf 2: Průměr kompresních poměrů (Zdroj: vlastní zpracování)

Po důkladném přepočítání kompresních poměrů u metod LZMA a LZMA2 bylo zjištěno, že LZMA dosahuje lepších poměrů než LZMA2.

4.2.5 Dekomprese

Jedná se o krok, který následuje po kompresi souborů a slouží k obnovení dat do původních hodnot. O dekompresi jednotlivých souborů se stejně jako při jejich kompresi staral program 7Zip.

Při dekompresi jsou zatíženy stejné komponenty jako u komprese. Takže zde bylo hodnoceno zatížení procesoru a operační paměti, mimo jiné i čas dekomprese. Pro měření byla zvolena tedy výše uvedená hodnotící kritéria, jež byla aplikována na prvních sedm souborů, neboť ostatní soubory byly příliš malé, a proto nebyly zahrnuty do měření.

V tabulce č.11 jsou uvedeny časy dílčích kompresních mechanismů pro dané soubory. Čas dekomprese byl značně nižší, než jak tomu bylo při kompresi, tudíž bylo změřeno pouze prvních 7 souborů. U zbylých šesti souborů dekomprese proběhla v řádu milisekund. Nejrychleji proběhla dekomprese u Deflate64 společně s GZip. LZMA a LZMA2 jsou na tom dosti podobně a rozdíly mezi nimi jsou téměř nulové. O něco hůře zvládl dekompresi kompresní mechanismus BZip2, který u větších souborů oproti předcházejícím metodám vykazuje rozdíl v řádu desítek sekund. Metoda PPMd zvládla dekompresi ze všech mechanismů nejhůře a u prvního souboru přesáhla hranici čtyř set sekund.

Název souboru	BZip2	Gzip	LZMA	LZMA2	PPMd	Deflate64
Vehicle, Snowmobile, and Boat Registrations	54 s	25 s	27 s	27 s	441 s	23 s
U.S. Chronic Disease Indicators	17 s	2 s	3 s	4 s	42 s	2 s
Health Care Provider Credential Data	15 s	3 s	5 s	6 s	57 s	3 s
Police Department Incident Reports 2018 to Present	11 s	2 s	3 s	3 s	48 s	2 s
Crime Data from 2020 to Present	7 s	2 s	3 s	3 s	36 s	2 s
U.S. COVID-19 Self-Test Data	4 s	1 s	1 s	1 s	16 s	1 s
Active Businesses - County Data	1 s	1 s	1 s	1 s	16 s	1 s

Tabulka 11: Čas dekomprese (Zdroj: vlastní zpracování)

Při dekompresi se od komprese také liší zátěž metod na procesor. Bylo zjištěno, že BZip2, který při kompresi zatěžoval procesor nejvíce ze všech měřených kompresních metod, zde sice dopadl opět nejhůře, ale příliš se neliší od PPMd. Stejně jako u komprese nejméně zatěžuje procesor metoda GZip společně s Deflate64. LZMA a LZMA2 zatěžují procesor více při dekompresi než při kompresi, a to díky jejich nastaveným vstupním hodnotám.

Název souboru	BZip2	Gzip	LZMA	LZMA2	PPMd	Deflate64
Vehicle, Snowmobile, and Boat Registrations	14,20%	6,00%	11,50%	10,30%	12,40%	6,00%
U.S. Chronic Disease Indicators	11,70%	8,30%	11,50%	11,40%	11,60%	7,70%
Health Care Provider Credential Data	12,80%	10,40%	11,50%	11,50%	11,90%	10,30%
Police Department Incident Reports 2018 to Present	12,00%	8,80%	9,40%	8,70%	11,80%	7,90%
Crime Data from 2020 to Present	12,00%	5,30%	8,70%	8,30%	11,70%	5,30%
U.S. COVID-19 Self-Test Data	11,40%	2,50%	4,20%	4,00%	11,70%	3,10%
Active Businesses - County Data	6,10%	2,30%	2,30%	2,30%	11,80%	2,40%

Tabulka 12: Zátěž na procesor při dekompresi (Zdroj: vlastní zpracování)

Zátěž na operační paměť se také liší, jak lze vidět v tabulce č.13. Metoda BZip2 zatěžuje RAM paměť pouze s 10 % jako při kompresi. GZip dekomprimoval konstantní zátěží 3,4 MB podobně jako při kompresi. Měření ukazuje, že Deflate64 je schopen dosáhnout výrazné redukce zatížení operační paměti při dekompresi, kde dosahuje pouze poloviční zatížení oproti kompresi. Metody LZMA a LZMA2 jednoznačně nezatěžují operační paměť, tak jako při kompresi. PPMd metoda při kompresi i dekompresi zatěžuje RAM paměť stejnou mírou, což vedlo k nejhorším výsledkům z měřených metod.

Název souboru	BZip2	Gzip	LZMA	LZMA2	PPMd	Deflate64
Vehicle, Snowmobile, and Boat Registrations	3.2 MB	3,4 MB	771,4 MB	771,5 MB	1027,5 MB	3,6 MB
U.S. Chronic Disease Indicators	7,8 MB	3,4 MB	771,4 MB	772,4 MB	1027,5 MB	3,5 MB
Health Care Provider Credential Data	7,8 MB	3,4 MB	771,4 MB	772,4 MB	1027,5 MB	3,6 MB
Police Department Incident Reports 2018 to Present	7,8 MB	3,3 MB	568 MB	461,7 MB	1027,5 MB	3,5 MB
Crime Data from 2020 to Present	7,8 MB	3,4 MB	507,6 MB	461,7 MB	1027,5 MB	3,5 MB
U.S. COVID-19 Self-Test Data	7,8 MB	3,4 MB	220 MB	283,4 MB	1027,5 MB	2,5 MB
Active Businesses - County Data	2,3 MB	3,4 MB	216,7 MB	205,1 Mb	1027,4 MB	2,5 MB

Tabulka 13: Zátěž na operační paměť při dekompresi (Zdroj: vlastní zpracování)

5 Výsledky a diskuse

5.1 Zhodnocení jednotlivých kompresních mechanismů a počítačů

Tato kapitola slouží ke shrnutí preferencí jednotlivých kompresních mechanismů. Pro tento účel bylo vytvořeno sedm tabulek, kde u šesti z nich jsou popsány výsledky měření testovaných kritérií (tabulka 14) a v tabulce 15 jsou shrnuty výsledné velikosti souborů za použití dílčích kompresních mechanismů.

Metoda BZip2, která dosahuje nízkých hodnot u času komprese i nízkých hodnot zátěže operační paměti, představuje obrovskou zátěž pro procesor. V určitých případech využívala veškerý prostor procesoru. Na druhou stranu nabízí dobrý výsledek kompresního poměru.

GZip exceluje v době trvání komprese, která dosahuje v průměru hodnoty 12,25 sekundy. Mimo jiné téměř nezatěžuje ani procesor, ani operační paměť. Bohužel dosahuje nejhorších kompresních poměrů.

Kvůli schopnosti nastavit vysoké hodnoty vstupních parametrů způsobuje LZMA metoda největší zatížení operační paměti ze všech měřených metod a zároveň nedosahuje dobrých výsledků v čase komprese. Ovšem na rozdíl od BZip2 zatěžuje procesor pouze minimálně. Výsledný kompresní poměr představuje průměrné výsledky.

Novější mechanismus LZMA2 se od svého předchůdce liší v času komprese. Průměrná doba komprese LZMA2 je delší o 30 sekund, což z něj činí mechanismus s nejdelší dobou trvání komprese. Dosáhl ale o trochu menšího zatížení operační paměti. V průměru dosahuje i lepších hodnot zatížení procesoru než LZMA2. Avšak po detailnějším měření kompresního poměru v kapitole 4.2.4. bylo zjištěno, že LZMA2 dosahuje horších kompresních poměrů než LZMA.

PPMd metoda dosáhla jednoznačně nejlepších kompresních poměrů ze všech metod, které byly zahrnuty v měření, a to s průměrem 4,77 %. Nicméně průměrný čas potřebný k provádění komprese byl nižší u jiných metod, například Deflate64 a GZip. Ovšem v zátěži na operační paměť se řadí hned před LZMA a LZMA2. V zátěži na procesor dosahuje průměrných výsledků.

Deflate64, podobně jako GZip, zvládne kompresi souborů značně rychleji než ostatní kompresní mechanismy. Mimo jiné také skoro vůbec nezatěžuje procesor ani operační paměť. Nedosahuje příliš dobrých výsledků kompresního poměru, ale je na tom o něco lépe než GZip.

Kompresní metoda - BZip2	Průměr hodnot
Čas komprese	38,83 s
Zátěž procesoru	59,87%
Zátěž na operační pamět	101,61 MB
Kompresní poměr	7,00%

Kompresní metoda - GZip	Průměr hodnot
Čas komprese	12,25 s
Zátěž procesoru	5,43%
Zátěž na operační pamět	3,62 MB
Kompresní poměr	10,92%

Kompresní metoda - LZMA	Průměr hodnot
Čas komprese	163,07 s
Zátěž procesoru	7,27%
Zátěž na operační pamět	3241,58 MB
Kompresní poměr	7,69%

Kompresní metoda - LZMA2	Průměr hodnot
Čas komprese	194,92 s
Zátěž procesoru	5,99%
Zátěž na operační pamět	3215,24 MB
Kompresní poměr	7,69%

Kompresní metoda - PPMd	Průměr hodnot
Čas komprese	62,61 s
Zátěž procesoru	6,81%
Zátěž na operační pamět	969,86 MB
Kompresní poměr	4,77%

Kompresní metoda - Deflate64	Průměr hodnot
Čas komprese	13,72 s
Zátěž procesoru	5,93%
Zátěž na operační pamět	7,6 MB
Kompresní poměr	10,46%

Tabulka 14: Shrnutí kompresních mechanismů (Zdroj: vlastní zpracování)

Název souboru	Velikost před kompresí	BZip2	Gzip	LZMA	LZMA2	PPMD	Deflate64
Vehicle, Snowmobile, and Boat Registrations	9 130 687 921	524 424 247	810 599 099	596 248 151	596 363 650	381 413 006	785 566 230
U.S. Chronic Disease Indicators	1 284 070 975	51 450 752	92 792 653	59 493 031	59 504 566	34 223 547	82 603 131
Health Care Provider Credential Data	1 136 868 518	106 950 504	164 684 607	124 209 110	124 233 161	78 167 486	160 888 990
Police Department Incident Reports 2018 to Present	894 252 637	64 013 516	111 845 705	49 541 860	49 551 448	36 856 681	104 774 352
Crime Data from 2020 to Present	570 228 647	41 239 379	73 645 532	46 397 277	46 406 240	26 429 327	68 686 339
U.S. COVID-19 Self-Test Data	307 823 316	20 442 974	30 191 102	25 727 174	25 732 168	16 246 144	28 379 075
Active Businesses - County Data	230 695 941	20 888 773	31 022 401	23 165 814	23 170 300	15 472 657	29 758 544
Oregon Active Workers' Compensation Employer Database	147 493 711	11 442 852	19 428 257	11 614 189	11 616 443	7 132 788	18 353 262
Electric Vehicle Population Data	93 228 248	5 850 827	9 820 989	6 277 595	6 278 813	4 005 206	9 118 228
Heart Disease Mortality Data Among US Adults (35+)	55 725 227	2 332 652	3 694 419	2 630 168	2 630 680	1 633 112	3 519 314
Power Outages - County	17 754 394	1 460 065	2 253 839	1 956 432	1 956 808	1 333 243	2 190 461
Voter Registration Data	8 603 486	550 481	699 214	665 447	665 580	470 551	706 525
ABS Store Inventory and Sale Items	4 680 906	594 590	861 610	685 419	685 551	458 491	837 531

Tabulka 15: Výsledné velikosti souborů po kompresi (Zdroj: vlastní zpracování)

Ze tří testovaných počítačů byly veškeré výsledky shrnuty z počítače ze střední třídy. U počítače z vyšší třídy byly testovány dva soubory za použití nejvýše nastavených vstupních nastavení kompresních mechanismů LZMA a LZMA2, které nemohly být otestovány na počítači nižší a střední třídy. Výsledky testování a použitá vstupní nastavení jsou uvedena v tabulce 16 a 17. Po porovnání těchto výsledků s výsledky v tabulkách 7, 8, 9 a 10 vyplývá, že rozdíl od výsledků na středním počítači není příliš velký, co se týče výsledné velikosti souboru a doby trvání komprese. Přesto bylo dosaženo lepšího výsledného kompresního poměru než u nastavení na počítači střední třídy. Výrazně přitom ale vzrostla zátěž operační paměti, což je způsobené maximálním nastavením velikosti slovníku.

Název souboru	Active Businesses - County Data	Vehicle, Snowmobile, and Boat Registrations
Komprimační metoda	LZMA	LZMA
Formát archivu	7z	7z
Úroveň komprese	Nejrychlejší	Nejrychlejší
Velikost slovníku	3840 MB	3840 MB
Velikost slova	273	273
Čas komprese	26 s	1605 s
Zátěž procesoru	11,20%	11,30%
Zátěž RAM paměti	1 357,8 MB	23 554,1 MB
Velikost po kompresi	23 165 814	594 270 866
Kompresní poměr	10,00%	6,00%

Tabulka 16: Shrnutí komprese LZMA na počítači vyšší třídy (Zdroj: vlastní zpracování)

Název souboru	Active Businesses - County Data	Vehicle, Snowmobile, and Boat Registrations
Komprimační metoda	LZMA2	LZMA2
Formát archivu	7z	7z
Úroveň komprese	Nejrychlejší	Nejrychlejší
Velikost slovníku	3840 MB	3840 MB
Velikost slova	273	273
Čas komprese	27 s	1606 s
Zátěž procesoru	11,40%	11,60%
Zátěž RAM paměti	1 326,6 MB	23 554,1 MB
Velikost po kompresi	23 170 300	594 385 953
Kompresní poměr	10,00%	6,00%

Tabulka 17: Shrnutí komprese LZMA2 na počítači vyšší třídy (Zdroj: vlastní zpracování)

5.2 Shrnutí a doporučení

Celkově lze říct, že pro kompresi XML souborů je potřeba zhodnotit více parametrů pro výběr vhodného kompresního mechanismu. Při kompresi menších souborů (s velikostí menší než 500 MB) se může rozhodovat primárně na základě výsledků kompresního poměru, jelikož doba komprese je v řádu milisekund až několika desítek sekund, a tak výrazně neovlivňuje výkon systému. V tomto případě může být nejlepším kandidátem PPMd, LZMA a LZMA2. BZip2 také není špatnou volbou, ovšem je třeba brát ohled na jím způsobenou vysokou zátěž procesoru. Z tohoto důvodu nedoporučuji současně vykonávat jiné úkony na počítači, mimo procesu komprese, poněvadž by vykonávané úkony trvaly příliš dlouho.

Pro větší soubory (s velikostí větší než 500 MB) je třeba kromě kompresního poměru brát v úvahu i dobu trvání komprese, protože zde může dojít k výraznému zatížení operační paměti, což může mít negativní dopad na výkon celého systému. Zde se dělí uvedená doporučení na dvě možnosti. Pro účely nízké doby trvání a nezatížení počítačových komponentů se hodí metody GZip a Deflate64, které sice nedosahují tak kvalitních výsledků

kompresního poměru, ale soubor komprimují velice rychle a příliš při tom nezatěžují hardware. Pokud je ovšem trváno na co největším zmenšení vstupního souboru, tak doporučuji metodu PPMd či na výkonnějších počítačích metody LZMA a LZMA2.

Je důležité zohlednit i výkon počítače při výběru vhodného kompresního mechanismu, protože pro maximální využití některých kompresních mechanismů může být výkon hardwaru klíčový. V průběhu testování bylo zjištěno že, operační paměť je počítačovou komponentou, která nejvíce limituje možné nastavení komprese. PPMd metoda je vhodná pro počítače s velikostí operační paměti alespoň 4 GB. Nabízí nejlepší výsledky kompresních poměrů ze všech testovaných metod, ovšem za cenu vyšší zátěže operační paměti a delší doby trvání komprese. Metody LZMA a LZMA2 je nejlepší používat na počítačích vyšší třídy. Tyto metody nabízejí dobré výsledky kompresních poměrů, širokou škálu nastavení komprese a nízkou zátěž procesoru. Na druhou stranu pro plné využití těchto metod je zapotřebí více GB operační paměti, jelikož, jak je patrné z tabulky 16 a 17, při kompresi větších souborů využívají až 23 GB paměti. Pro nejrychlejší kompresi souborů a takřka přehlédnutelné zatížení počítačových komponentů (vhodné například pro nižší třídu a kompresi velikostně vyšších souborů) doporučuji metodu GZip či Deflate64. V poslední řadě může metoda BZip2 v určitých situacích, jak již bylo řečeno, využívat pro potřeby komprese veškerou volnou kapacitu procesoru, tudíž ji nemohu doporučit používat při souběžně vykonávané práci na počítači. Lze ji ovšem doporučit pro všechny typy počítačů, protože příliš nezatěžuje operační paměť a dosahuje lepších výsledků kompresních poměrů než ostatní metody.

Mimo jiné bylo v průběhu testování zjištěno, konkrétně u souborů „U.S. Chronic Disease Indicators“ a „Health Care Provider Credential Data“, že přesto, že první uvedený soubor je větší než druhý, tak dosahuje lepších celkových výsledků. Po důkladnějším rozboru jednotlivých souborů se ukázalo, že se oba soubory liší ve vnitřní struktuře. Menší velikost souboru tedy automaticky neznamená, že soubor dosáhne lepších výsledků komprese. Při kompresi XML souborů je tedy třeba zhodnotit i vnitřní strukturu individuálních souborů.

6 Závěr

Hlavním cílem mé bakalářské práce bylo otestování a zhodnocení vybraných nástrojů pro kompresi XML souborů. Tohoto cíle bylo dosaženo na základě plnění dílčích cílů. První dílčí cíl byl splněn v teoretické části jakožto představení značkovacího jazyka XML společně s kompresními mechanismy, ať už specializovanými na XML či nikoliv. K jednotlivým kompresním mechanismům se váže kapitola s algoritmy, které dané kompresní mechanismy využívají. Východiska načerpaná z teoretické části posloužila pro vybrání vhodných hodnotících kritérií a kompresních mechanismů v praktické části a celkovému pochopení dané problematiky.

V této práci tedy proběhlo měření účinností vybraných kompresních mechanismů, mezi které spadá BZip2, GZip, LZMA, LZMA2, PPMd a Deflate64 na určité XML soubory odlišující se svojí velikostí a vnitřní strukturou. Každý z těchto mechanismů byl zhodnocen na základě zvolených hodnotících kritérií. Z výsledků daných měření vyplývá, že je potřeba brát v úvahu výkon počítače, na kterém je komprese vykonávána, protože každý z testovaných kompresních mechanismů zatěžuje odlišnou komponentu počítače. Nejvíce je kladen důraz na operační paměť a procesor. Hodnotící kritéria, na která je také dbáno, jsou doba komprese a kompresní poměr. Tato kritéria se odvíjí primárně od velikosti vstupního XML souboru a také od jeho interní struktury. Výsledky měření byly shrnuty a byla formulována doporučení na základě specifikací výkonu hardwaru a velikosti vstupních souborů pro každý z dílčích kompresních mechanismů.

Jelikož je každý kompresní mechanismus něčím jedinečný, tak lze konstatovat, že výběr optimálního kompresního mechanismu záleží na komponentách počítače, konkrétních požadavcích a situaci, ve které bude používán. Každý z konkrétních kompresních mechanismů má vlastní charakteristiku a omezení, která určují jeho efektivitu v daných situacích. Nelze tedy určit, který je jednoznačně nejlepší, a proto je před výběrem vhodného kompresního mechanismu nutné vyhodnotit konkrétní situaci, používaný hardware a požadavky na kompresi dat.

Budoucí možnost vylepšení měření může vést k hlubší analýze vstupních nastavení kompresních mechanismů a přesnějšímu zjištění dopadů těchto nastavení na zvolená hodnotící kritéria. Je možné začlenit i více kompresních mechanismů do součásti měření.

7 Seznam použitých zdrojů

1. XML (Extensible Markup Language). *Techtarget* [online]. 2021 [cit. 2023-03-13]. Dostupné z: https://www.techtarget.com/whatis/definition/XML-Extensible-Markup-Language
2. XPath. *Mdn web docs* [online]. 2022 [cit. 2022-08-02]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/XPath
3. The Extensible Stylesheet Language Family (XSL). *W3C* [online]. 2009 [cit. 2022-08-02]. Dostupné z: https://www.w3.org/Style/XSL/
4. XML Tutorial. *W3schools* [online]. 1999 [cit. 2022-08-02]. Dostupné z: https://www.w3schools.com/xml/xml_whatis.asp
5. XML|Elements. *Geeksforgeeks* [online]. 2022 [cit. 2022-08-02]. Dostupné z: https://www.geeksforgeeks.org
6. Základy XML – struktura dokumentu [online]. 2011 [cit. 2022-08-02]. Dostupné z: http://geomatika.kma.zcu.cz/studium/pok/Materialy/02_XML_zaklady_testy.pdf
7. JALIL JAMAL, Shene, Chnoor M. RAHMAN a Mzhda S. ABDULKARIM. *XML Schema Validation Using Java API for XML Processing* [online]. 2022 [cit. 2022-08-02]. Dostupné z: https://doaj-org.infozdroje.czu.cz/article/5d7042ba8f214a82b8aa3d139db490ef
8. *XML Tutorial* [online]. [cit. 2022-08-02]. Dostupné z: https://www.tutorialspoint.com/xml/index.htm
9. XML DOM. *Javapoint* [online]. [cit. 2022-08-03]. Dostupné z: https://www.javatpoint.com/xml-dom
10. Parsing and Processing XML. *MSc-IT Study Material* [online]. 2010 [cit. 2022-08-03]. Dostupné z: https://www.cs.uct.ac.za/mit_notes/web_programming/html/ch09s04.html
11. PHP a XML: SAX – čteme pěkně popořádku. *Zdrojak* [online]. 2009 [cit. 2022-08-03]. Dostupné z: https://zdrojak.cz/clanky/php-a-xml-sax-cteme-pekne-poporadku/
12. Advantages of XML | disadvantages of XML. *Rfwireless-world* [online]. 2012 [cit. 2022-08-03]. Dostupné z: https://www.rfwireless-world.com/Terminology/Advantages-and-Disadvantages-of-XML.html

13. Advantages and Disadvantages of XML. *Beginnersbook* [online]. [cit. 2022-08-03]. Dostupné z: https://beginnersbook.com/2018/10/advantages-and-disadvantages-of-xml/
14. Bzip2. *Bzip* [online]. [cit. 2022-08-03]. Dostupné z: http://www.bzip.org/
15. Gzip. *Strafelda* [online]. [cit. 2022-08-03]. Dostupné z: https://www.strafelda.cz/gzip
16. What Is GZIP Compression?. *Wiredelta* [online]. 2020 [cit. 2022-08-03]. Dostupné z: https://wiredelta.com/gzip-compression/
17. M. TOLANI, Pankaj. *XGRIND: A Query-friendly XML Compressor* [online]. 29 [cit. 2022-08-03]. Dostupné z: https://cse.hkust.edu.hk/~wilfred/Seminars/3%20XGrind.pdf
18. SAPATE, Suchit. *Effective XML Compressor: XMill with LZMA Data Compression* [online]. 2019 [cit. 2022-08-03]. Dostupné z: https://www.mecs-press.org/ijeme/ijeme-v9-n4/IJEME-V9-N4-1.pdf
19. About XGrind. *Xgrind.sourceforge* [online]. 2002 [cit. 2022-08-03]. Dostupné z: http://xgrind.sourceforge.net/
20. MIN, Jun-Ki, Myung-Jae PARK a Chin-Wan CHUNG. *XPRESS: A Queriable Compression for XML Data* [online]. 12 [cit. 2022-08-03]. Dostupné z: http://islab.kaist.ac.kr/chungcw/InterConfPapers/sigmod2003_jkmin.pdf
21. Burrows-Wheeler Transform. *Medium* [online]. 2019 [cit. 2022-08-03]. Dostupné z: https://medium.com/@mr-easy/burrows-wheeler-transform-d475e0aacad6
22. *Design & Analysis of Algorithms* [online]. 2018 [cit. 2022-08-03]. Dostupné z: https://www.cs.cmu.edu/~15451-f18/lectures/lec25-bwt.pdf
23. WANG LING, Tok a Changqing LI. *Advanced Applications and Structures in XML Processing: Label Streams, Semantics Utilization and Data Query Technologies* [online]. U.S.A., 2010 [cit. 2022-08-03]. ISBN 1615207279
24. What is Data Compression?. *Barracuda* [online]. 2018 [cit. 2022-08-04]. Dostupné z: https://www.barracuda.com/glossary/data-compression
25. How does data compression work?. *Netmotionsoftware* [online]. 2019 [cit. 2022-08-04]. Dostupné z: https://www.netmotionsoftware.com/blog/connectivity/how-does-data-compression-work

26. Difference between Lossy Compression and Lossless Compression. *Geeksforgeeks* [online]. 2020 [cit. 2022-08-04]. Dostupné z: https://www.geeksforgeeks.org/difference-between-lossy-compression-and-lossless-compression/
27. Huffman Coding Implementation in Python with Example. *Favtutor* [online]. 2022 [cit. 2022-08-04]. Dostupné z: https://favtutor.com/blogs/huffman-coding
28. G. CLEARY, JOHN a W. J. TEAHAN. *Unbounded Length Contexts for PPM* [online]. 2003 [cit. 2022-08-05]. Dostupné z: https://www.researchgate.net/publication/2473004_Unbounded_Length_Contexts_for_PPM
29. NG, Wilfred, Lam WAI YEUNG a James CHENG. *Comparative Analysis of XML Compression Technologies* [online]. 32 [cit. 2022-08-05]. Dostupné z: https://cse.hkust.edu.hk/~wilfred/paper/wwwj05.pdf
30. Millau: an encoding format for efficient representation and exchange of XMLover the Web [online]. [cit. 2022-08-05]. Dostupné z: https://www.ra.ethz.ch/cdstore/www9/154/154.html
31. Delta Compression Techniques [online]. 2018 [cit. 2022-08-06]. Dostupné z: https://link.springer.com/referenceworkentry/10.1007/978-3-319-63962-8_63-1
32. XQzip: Querying Compressed XML Using Structural Indexing [online]. 2004 [cit. 2022-08-09]. Dostupné z: http://www.cse.cuhk.edu.hk/~jcheng/papers/XQzip_edbt04.pdf
33. LZMA: THE NEW COMPRESSION ALGORITHM. *4D Blog* [online]. 2021 [cit. 2023-03-13]. Dostupné z: https://blog.4d.com/lzma-the-new-compression-algorithm/
34. LZMA: What is LZMA Compression?. *WinZip* [online]. [cit. 2023-03-13]. Dostupné z: https://www.winzip.com/en/learn/tips/what-is-lzma/
35. Namespaces in XML 1.0 (Third Edition*)* [online]. [cit. 2023-03-14]. Dostupné z: https://www.w3.org/TR/xml-names/
36. XML Tree Structure [online]. [cit. 2023-03-14]. Dostupné z: https://www.javatpoint.com/xml-tree-structure
37. Memory-Side Acceleration for XML Parsing [online]. 2011 [cit. 2023-03-14]. Dostupné z: https://www.researchgate.net/publication/220718096_Memory-Side_Acceleration_for_XML_Parsing

38. How to Enable GZIP Compression to Speed Up WordPress Sites [online]. [cit. 2023-03-14]. Dostupné z: https://kinsta.com/blog/enable-gzip-compression/
39. ÖZSU, M. Tamer a Ling LIU. *Encyclopedia of Database Systems 2nd Edition*. 2018. Springer-Verlag New York. ISBN 978-1-4614-8266-6.
40. FERRAGINA, Paolo, Raffaele GIANCARLO, Giovanni MANZINI a Matinella SCIORTINO. *Boosting Textual Compression in Optimal Linear Time* [online]. 26 [cit. 2023-03-14]. Dostupné z: https://people.unipmn.it/manzini/papers/jacm05b.pdf
41. W. SMITH, Steven. *The Scientist and Engineer's Guide to Digital Signal Processing*. 1997. California Technical Pub. ISBN 978-0966017632.
42. LIN, Ming-Bo a Gene Eu JAN. *A Lossless Data Compression and Decompression Algorithm and Its Hardware Architecture* [online]. [cit. 2023-03-14]. Dostupné z: https://www.researchgate.net/publication/3337841_A_Lossless_Data_Compression_and_Decompression_Algorithm_and_Its_Hardware_Architecture
43. Data catalog. *Data.GOV* [online]. [cit. 2023-03-15]. Dostupné z: https://catalog.data.gov/dataset/?res_format=XML
44. How to: Use XSLT Transformations with Project XML Data Interchange Files. *Microsoft* [online]. 2021 [cit. 2023-03-15]. Dostupné z: https://learn.microsoft.com/en-us/office-project/xml-data-interchange/how-to-use-xslt-transformations-with-project-xml-data-interchange-files?view=project-client-2016

7.1 Seznam obrázků

Seznam tabulek

7.2 Seznam grafů

7.3 Seznam použitých zkratek

API – Application Programming Interface
ASCII – American Standard Code for Information Interchange
BWT – Burrows-Wheleer Transform
CPU – Central Processing Unit
DDT – Distributed Data-structures for Text processing
DFA – Deterministic Final Automat
DOM – Document Object Model
DTD – Data Type Definition
HTTP – Hypertext Transfer Protocol
HTTP2 – Hypertext Transfer Protocol version 2
LZ77 – Lempel-Ziv 1977
LZMA – Lempel-Ziv-Markov chain Algorithm
MHM – Markov Histograms for Multi-dimensional Data
PPM – Prediction by Partial Matching
SAX – Simple API for XML
SGML – Standard Generalized Markup Language
SIT – Structured Index Tree
URI – Uniform Resource Identifier
W3C – World Wide Web Consortium
WAP – Wireless Application Protocol
WBXML – Wireless Application Protocol Binary XML
XBW – XMill Binary Format for XML
XHTML – eXtensible Hypertext Markup Language
XML – eXtensible Markup Language
XMLPPM – XML Prediction by Partial Matching
XPath – XML Path Language
XSD – XML Schema Definition
XSL – eXtensible Stylesheet Language
XSL-FO – XSL - Formatting Objects
XSLT – XSL Transformations