Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra kybernetiky

DIPLOMOVÁ PRÁCE

PLZEŇ, 2012 **Bc. JAN VAVRUŠKA**

P R O H L Á Š E N Í

Předkládám tímto k posouzení a obhajobě diplomovou práci zpracovanou na závěr studia na Fakultě aplikovaných věd Západočeské univerzity v Plzni.

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím odborné literatury a pramenů, jejichž úplný seznam je její součástí.

V Plzni dne .....

Mé poděkování patří v první řadě mému vedoucímu Ing. Janu Švecovi za jeho vstřícnost a trpělivost po celou dobu řešení této práce. Jeho nápady a cenné rady byly vždy velmi inspirativní a vážím si toho, že jsem mohl svou práci dělat právě pod jeho vedením.

Dále bych chtěl poděkovat mým rodičům Janě a Josefovi za podporu, zázemí a lásku, které mně věnovali nejen v průběhu mých studií na vysoké škole.

Stejně tak děkuji i mé přítelkyni Lence, která mi byla oporou ve chvílích, kdy mně práce nešla tak, jak jsem si původně představoval. Mé díky rovněž patří i její mamince Anně Brabcové za podoru a vytvoření příjemného pracovního prostředí při psaní této práce.

Anotace

Tato práce se zabývá metodami pro úlohu vyhledávání slov v rozsáhlém archivu mluvené řeči. Vyhledávání v takovém archivu je umožněno prostřednictvím indexace slovních a fonémových mřížek, které jsou výstupem systému automatického rozpoznávání řeči. Hlavním cílem práce bylo nastudovat a aplikovat přístup k indexaci a vyhledávání v mřížkách s využitím teorie vážených konečných automatů a nástrojů STDTools. Následně jej pak otestovat na zvolených experimentálních datech a porovnat se systémem, založeným na indexaci vybraných mřížkových hran. Porovnávání probíhalo vyhodnocením přístupu z hlediska jeho přesnosti, rychlosti vyhledávání a nároků na datový prostor.

Klíčová slova: vyhledávání slov v řečovém archivu, automatické rozpoznávání řeči, slovní a fonémové mřížky, vážené konečné automaty a transducery, projekt MALACH, STDTools, OpenFST

Annotation

The focus of this thesis is the Spoken Term Detection (STD) task whose aim is to index and search word and phoneme lattices resulting from the Automatic Speech Recognition (ASR) system. The main goal of this thesis was to thoroughly familiarize with the theory of weighted finite-state automata (WFSA) and then implement a spoken term detection system using existing software framework (STDTools). Consequently, the implemented system has been tested on the large real-world data and results were compared with the existing STD engine developed previously within the research group. The comparison was based on the evaluation of precision, search time and also data storage requirements.

Keywords: Spoken Term Detection, Automatic Speech Recognition, word and phoneme lattices, weighted finite-state automata and transducers, STDTools, OpenFST, Multilingual Access to Large Spoken Archives

Obsah

1 Úvod

1.1 Motivace

Život člověka v dnešní moderní civilizaci je už prakticky nerozlučně spjat s rozvojem multimédií všeho druhu. Vývoj IT technologií udává tempo neustále se zvyšující výkonnosti počítačů a prostřednictvím internetu směřuje k daleko větší dostupnosti velmi rozsáhlých multimediálních dat, než tomu bylo možné v dřívějších dobách. Televizní a rozhlasové společnosti zpřístupňují stále více svých pořadů on-line, nejrůznější instituce jako knihovny, historické archivy aj. díky rozvoji digitalizace uchovávají ve svých depozitářích kromě textových i audio a video data, a konečně internetové multimediální servery jako je např. YouTube jsou příkladem za všechny.

Ze snahy o co nejlepší dosupnost těchto multimediálních informací logicky vyvstává potřeba v těchto archivech vyhledávat. S narůstajícím objemem uložených dat není v lidských silách je všechny ručně zpracovávat a na scéně se tak objevují automatické přístupy s využitím metod umělé inteligence. Kromě toho, lidská práce je často nejdražší komoditou a tak automatizace procesů indexace a vyhledávání v multimediálních archivech přináší dnes - v době nejrůznějších úspor a škrtů - i významnou redukci nákladů. Jak plyne přímo z její definice, umělá inteligence jako vědní disciplína, se ve svých úlohách snaží strojově napodobovat některé schopnosti a činnosti člověka, které bychom intuitivně považovali za projev jeho inteligence. To vše s cílem usnadnit mu práci, či pomáhat při řešení některých velmi složitých problémů. V případě úlohy strojového, tj. automatického rozpoznávání řeči[1] se tedy nejčastěji snaží zpřístupnit člověku komunikaci se strojem v jeho nejpřirozenější podobě, tj. mluvenou řečí. Úloha ASR dnes nalézá uplatnění i v mnoha jiných odvětvích lidské činnosti, např. v hlasových dialogových systémech, bezpečnostních systémech s verifikací řečníka, usnadnění života osobám s nejrůznějšími zdravotními handicapy apod.

Předmětem této práce je problematika úlohy vyhledávání slov v rozsáhlém řečovém archivu, resp. rozbor a porovnání dvou různých přístupů k indexaci a vyhledávání nad výstupem systému automatického rozpoznávání řeči z takového archivu. Práce je členěna následovně: Kapitola 1 uvádí čtenáře do problematiky automatického rozpoznávání řeči a úlohy vyhledávání slov v rozsáhlém řečovém archivu. 2. kapitola je věnována teorii vážených

[1] ASR = angl.: Automatic Speech Recognition

konečných automatů, kterou využívá konstrukce indexu, popsaného v kapitole 3. Přehled přístupu k druhému způsobu indexace je nastíněn v kapitole 4. Implementaci přístupu s váženými konečnými automaty popisuje kapitola 5 a jeho porovnání se zkoumaným přístupem a zhodnocení je věnována kapitola 6. Obě činnosti spolu sice bezprostředně souvisí neboť závisí jedna na druhé, ale pro přehlednost jsem je přesto takto rozdělil do dvou samostatných kapitol. A konečně v kapitole 7 jsou prezentovány závěry a budoucí práce.

1.2 Automatické rozpoznávání řeči

Počátky výzkumu této problematiky jsou staré již více než padesát let. Za tu dobu byl na jeho poli učiněn již velký pokrok, ale i přesto je konstrukce systému, který by byl schopen rozpoznat libovolnou promluvu jakéhokoliv řečníka v jakémkoli jazyce, ještě vzdálenou budoucností. Samotná úloha zpracování řečového signálu je značně netriviální a skrývá v sobě mnoho souvisejících dílčích podproblémů. Namátkou jmenujme např.:

- problém odlišnosti hlasu v závislosti na řečníkovi, tj. odlišné parametry hlasového ústrojí a z toho plynoucí rozdíly např. v barvě hlasu, přízvuku, tempu řeči apod. ASR systémy tedy můžeme dělit na systémy na řečníku nezávislé (natrénované na množství hlasů) a závislé (natrénovány na jednoho nebo malou skupinu řečníků);

- problém závislosti hlasu na situaci (klid, rozčilení, promluva potichu nebo nahlas...), a také vliv koartikulace, tj. změny fonetických vlastností na začátku a konci slov v závislosti na kontextu okolních slov;

- měnící se akustické pozadí řeči(problém odlišení ruchů v prostředí) a problém zašumění signálu;

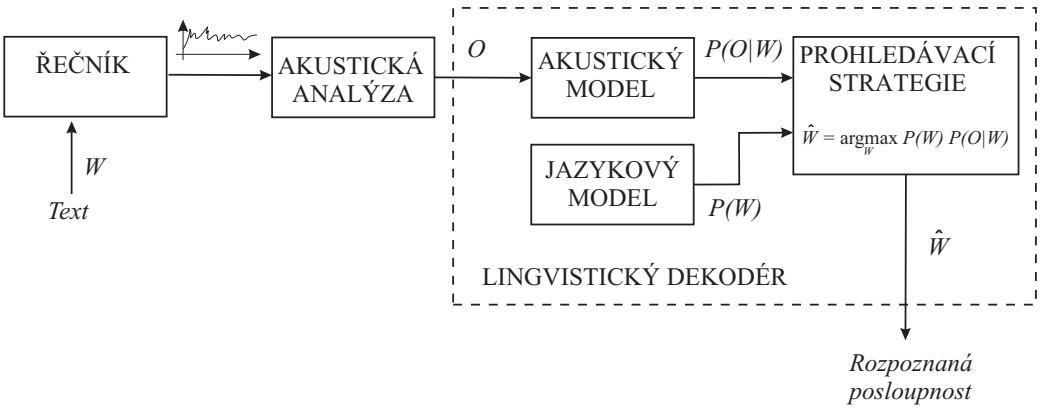
- úloha se také liší podle toho, zda jde o rozpoznávání izolovaně vyslovených slov oproti souvislé mluvené řeči, nebo podle toho, jestli jde o čtenou versus spontánně pronášenou řeč (kde mají významný vliv neřečové události).

Tyto problémy pak zavádí do úlohy rozpoznávání množství nejistot, které je nutno kompenzovat. Typickou úlohou dnešních ASR je vývoj systémů pro rozpoznání souvisle promluvené řeči s velkým slovníkem[2],tj. čítajícím stovky tisíc slov [4]. Dodnes bylo představeno několik základních přístupů (např.: porovnávání obrazů, znalostní přístup), z nichž v současné době dominuje využití statistických rozhodovacích metod [6].

[2]LVCSR = angl.: Large Vocabulary Continuous Speech Recognition

1.2.1 Statistický přístup

V následujícím odstavci čtenáře krátce uvedu do této problematiky tak, jak je popsána v [6]. Základní blokové schéma systému statistického rozpoznávání řeči ukazuje obrázek 1. Předpokládejme že $W = \{w_1, w_2, \ldots, w_n\}$



Obrázek 1: Dekompozice systému automatického rozpoznávání řeči

je posloupnost slov, pronesených řečníkem a $O = \{o_1, o_2, \ldots, o_n\}$ je posloupnost vektorů příznaků, tj. informace odvozená z řečového signálu. Lingvistický dekodér se z této posloupnosti snaží rozpoznat slova, která byla vyslovena. Cílem je tedy nalézt nejpravděpodobnější posloupnost slov $\hat{W}$, která maximalizuje podmíněnou pravděpodobnost $P(W|O)$. Z Bayessova pravidla plyne:

$$\hat{W} = arg\max_{W} P(W|O) = arg\max_{W} \frac{P(W)P(O|W)}{P(O)}, \tag{1}$$

kde:

$P(O\|W)$	...	pravděpodobnost, že při vyslovení slov W budou produkovány vektory příznaků O (akustický model),
$P(W)$	...	apriorní pravděpodobnost vyřčených slov W (jazykový model),
$P(O)$	...	apriorní pravděpodobnost výstupních vektorů.

Protože $P(O)$ není funkcí W, lze tuto část výrazu ignorovat a hledanou posloupnost $\hat{W}$ určit s pomocí sdružené pravděpodobnosti $P(W, O)$:

$$\hat{W} = arg\max_{W} P(W, O) = arg\max_{W} P(W)P(O|W). \tag{2}$$

Z uvedeného plyne, že problém lze řešit dekompozicí na:

1. Provedení akustické analýzy řečového signálu a určit O.
2. Vytvoření akustického modelu pro ocenění pravděpodobnosti $P(O|W)$.
3. Vytvoření jazykového modelu pro ocenění $P(W)$.
4. Vyhodnocení rovnice 2 aplikací účinné prohledávací strategie, tj. dekódování.

Úlohy na konstrukci akustického modelu spočívají v problému nalezení matematického modelu, který co nejvěrohodněji reprezentuje skutečné akustické vlastnosti řeči. Užívají se téměř výhradně tzv. Skryté Markovovy Modely[3]. Modely mají danou strukturu a parametry, které se trénují na základě rozsáhlých trénovacích množin anotovaných promluv. Jazykový model obsahuje znalosti, popisující stavbu a chování jazyka. Trénují se statistickým zpracováním rozsáhlého množství textů. Rozpoznávání je pak formulováno jako problém dekódování, tj.:

a) Nalezení posloupnosti stavů HMM, která s největší pravděpodobností generuje posloupnost pozorování O.

b) Prohledávání sítě všech možných řešení posloupnosti slov W, daných jazykovým modelem $P(W)$, a nalezení takové cesty $\hat{W}$, která nejlépe odpovídá posloupnosti značek O (podle kritéria *maximální aposteriorní pravděpodobnosti* - MAP).

Z předchozích faktů přímo vyplývají omezení, která tento přístup do úlohy přináší. Především je to omezení v množství a povaze trénovacích dat. Aby postihly všechny myslitelné aplikace systému ASR, muselo by jich být obrovské množství. To však v praxi není možné splnit a proto úloha rozpoznávání řeči může přinášet dobré výsledky vždy jen na té doméně, pro kterou byl systém ASR natrénován. Příklad jednoho z nejvíce zkoumaných problémů můžeme nalézt v podobě tzv. out-of-vocabulary (OOV) slov, tj. takových slov, které nejsou ve slovníku ASR systému. Taková slova v mluvené řeči pak systém buď nerozpozná, nebo vrátí neadekvátní výsledky.

1.2.2 Slovní a fonémové mřížky

V typických úlohách rozpoznávání řeči, probíhajících v reálném čase (např. automatický přepis diktované řeči, titulkování televizních pořadů, apod.), je

[3]HMM, angl.: Hidden Markov Model

výsledkem dekódování jedno nejlepší řešení posloupnosti $\hat{W}$. Někdy je ale úloha dekódování definována jako nalezení více než jednoho, tj. n nejlepších hypotéz. Rozpoznávání pak probíhá zpravidla off-line a výstup dekodéru může být reprezentován formou hranově ohodnoceného, orientovaného a topologicky uspořádaného grafu (podle času). Ukázkový příklad vidíme na obrázku 2. Jeho uzly (s přidanou informací o času v akustickém signálu) vymezují začátky a konce slov a hrany představují jednotlivá slova.
Ohodnocení hran představují pravděpodobnosti vyslovení slova v příslušném časovém intervalu. Pravděpodobnosti slov jsou normalizovány tak, že v každém časovém okamžiku grafu je součet všech pravděpodobností paralelních hypotéz roven jedné [11]. Seznam n nejpravděpodobnějších hypotéz posloupností slov je určen množinou všech cest, vedoucích z počátečního do koncového uzlu grafu. Takovéto grafy se nazývají slovní grafy nebo *slovní mřížky*.
Jiným typem může být *fonémová mřížka*, jejímž hranám namísto slov odpovídají jednotlivé fonémy[4]. Tyto mřížky bývají pouze výsledkem akustického modelování řeči, tj. bez použití přidaného jazykového modelu.[6]

Jak je patrné i z obrázku 2, v mřížce vzniká množství nadbytečných paralelních hran (hypotéz) se stejným vstupním symbolem a překrývajícím se časem výskytu. Řešení tohoto problému probíhá nejčastěji na dvou úrovních:

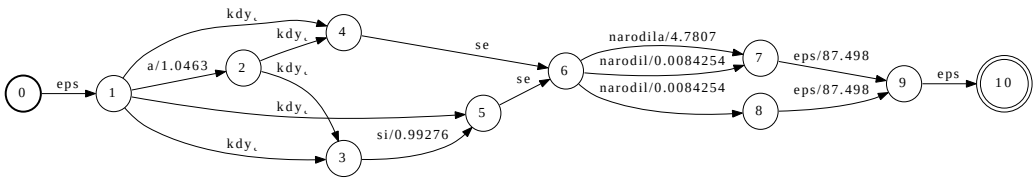
- Buď jsou tyto hrany zahozeny již během samotného procesu dekódování a výstupem je mřížka, obsahující již jen nejlepší cesty [10],
- nebo jejich odstranění probíhá až na úrovni zpracování výstupních mřížek dekodéru buď sloučením těchto hran dohromady nebo prořezáváním (viz odstavec 2.4.7 a 3.1).

Metody, kterými se zabývá tato práce, využívají de-facto kombinaci obou těchto přístupů, kdy z výstupu dekodéru jsou hrany mřížky omezeny parametrem, určujícím pro každý stav mřížky maximální počet vstupních hran do tohoto stavu. Výstupní mřížky jsou pak během indexace ještě dále prořezávány.

1.3 Úloha vyhledávání slov v rozsáhlém řečovém archivu

Abychom mohli definovat úlohu, kterou se zabývá tato práce, musíme ještě zmínit několik základních pojmů. Jedním ze stěžejních odvětví oboru

[4]Fonémy (hlásky) představují přirozené nejmenší jednotky řeči, které rozlišují slova [6]



Obrázek 2: Příklad skutečné slovní mřížky, která je výstupem systému ASR

umělé inteligence je oblast Získávání informací [5] jehož cílem je zprostředkovat uživateli přístup k datům z korpusu dokumentů na základě nějakého jeho dotazu. Jeho předmětem bývá nejčastěji vyhledání dokumentů na zadané téma, popř. vyhledání pozice konkrétních slov v dokumentech, ve kterých se vyskytují. Hlavní komponentu systému IR představuje *index*, což je nějaká datová struktura, obsahující všechny relevantní informace o takovém korpusu. Na úrovni textových dat si můžeme představit dokument tak, že zobrazuje jeho název na množinu slov v něm obsažených. Index představuje de facto jeho inverzi, tj. slova zobrazuje na názvy a příslušné pozice v dokumentech, ve kterých se vyskytuje [4]. Asi nejznámější praktickou aplikaci indexů nalezneme u internetových vyhledávačů.

Úloha vyhledávání slov v rozsáhlém archívu mluvené řeči[6] tedy představuje spojení úlohy IR a ASR, kdy systém pro vyhledávání informací indexuje výstup systému rozpoznávání řeči. V tomto případě se postupuje tak, že řečový archiv je nejprve zpracován systémem ASR a posléze je vytvořen index nad slovními nebo fonémovými mřížkami. Při dotazování tak není nutné znovu provádět akustické zpracování řeči, ale projde se již jednou rozpoznaný text. V úloze STD nebereme v potaz téma dané promluvy, ale zaměřujeme pozornost pouze na konkrétní výskyty slov v nich obsažených. Informace, které by intuitivně měl takový index pro hledaný výraz vracet, jsou: *i*) identifikátor promluvy, *ii*) časový interval výskytu slova a *iii*) míru důvěry nebo-li pravděpodobnost, že se jedná o hledané slovo v místě uvedeného výskytu.

Zatímco v případě souboru textových dokumentů je vyhledávání přesné ve smyslu pozice hledaného slova, v případě řečového archivu musíme počítat s nepřesností danou složitostí úlohy rozpoznávání a faktem, že systémy ASR nejsou univerzální. Další problém ředstavují již zmíněná OOV slova. Pokud se nacházejí v řečovém archivu ale ne v rozpoznávacím slovníku, systém ASR

[5] IR = angl.: Information Retrieval
[6] STD = angl.: Spoken Term Detection

je nerozpozná a nebudou se tak ve výstupních mřížkách nacházet. Možný způsob řešení tohoto problému spočívá ve vygenerování a indexování obou typů mřížek, tj. slovní i fonémové. Ty posledně zmiňované nepotřebují jazykový model a závisí pouze na informaci z akustického zpracování řečového signálu. Pokud se tedy nějaké slovo nenachází ve slovníku ASR systému, může být vygenerován jemu odpovídající fonetický dotaz a provedeno dodatečné vyhledání nad fonémovými mřížkami.

Předmětem této práce je úloha STD jako rozbor přístupu k indexaci a vyhledávání v souboru ASR mřížek, založeném na matematicky dobře podložené teorii vážených konečných transducerů[7] a operací s nimi. Následně pak i analýza jeho přesnosti, rychlosti, prostorové a výpočetní náročnosti v porovnání s již funkčním systémem, jehož index tvoří seznam hran, vybraných z mřížky na základě zvoleného kritéria.

[7] Angl.: WFST = Weighted Finite State Transducers

2 Vážené konečné transducery

V této kapitole jsou uvedeny základní pojmy z algebry polookruhů tak, jak jsou uvedeny v [5] [12] a [6], dále z teorie jazyků, gramatik a konečných automatů z [13] potažmo [9], které se bezprostředně týkají popisované problematiky tvorby vyhledávacího indexu. V definicích se vyskytují i obecně užívané pojmy z teorie množin, grafů a uspořádání. Ty z nich, které jsem považoval za účelné pro lepší pochopení problematiky, jsou více rozebrány podle [7] a ostatní ponechávám k čtenářovu samostatnému prostudování.

2.1 Polookruhy

Na začátek trochu předbíhám když uvedu, že problematika polookruhů se bude týkat vah na jednotlivých hranách konečných automatů (mřížek). Věřím, že to však později poslouží k rychlejší čtenářově orientaci.

Definice 1 (Monoid) Trojici $(\mathbb{K}, \otimes, \overline{1})$ nazýváme *monoid*, kde $\mathbb{K}$ je nějaká množina, $\otimes$ je uzavřený asociativní binární operátor na množině $\mathbb{K}$ a $\overline{1}$ je neutrální prvek pro tento operátor. Neutrální prvek a je takový prvek, pro který platí $\forall a \in \mathbb{K}$:

$$a \otimes \overline{1} = \overline{1} \otimes a = a.$$

Monoid je *komutativní*, pokud $\otimes$ je komutativní, tj. $\forall a \in \mathbb{K}$:

$$a \otimes b = b \otimes a.$$

Definice 2 (Polookruh) *Polookruh* nad množinou $\mathbb{K}$ nazveme pětici $(\mathbb{K}, \oplus, \otimes, \overline{0}, \overline{1})$, kde $(\mathbb{K}, \oplus, \overline{0})$ je komutativní monoid, $(\mathbb{K}, \otimes, \overline{1})$ je monoid, $\otimes$ je distributivní vůči $\oplus$, tj. $\forall a, b, c \in \mathbb{K}$:

$$c \otimes (a \oplus b) = (c \otimes a) \oplus (c \otimes b)$$

a $\overline{0}$ je ničitel vůči $\otimes$, tj. $\forall a \in \mathbb{K}$:

$$a \otimes \overline{0} = \overline{0} \otimes a = \overline{0}.$$

Pokud operace $\otimes$ je také komutativní, říkáme, že i polookruh je *komutativní*.

Definice 3 Polookruh je *idempotentní* jestliže $\forall a \in \mathbb{K} : a \oplus a = a$.

Název	Značení	Množina $\mathbb{K}$	$\oplus$	$\otimes$	$\overline{0}$	$\overline{1}$
Přirozený	$\mathcal{N}$	$\mathbb{N}$	$+$	$\times$	0	1
Booleovský	$\mathcal{B}$	$\{0,1\}$	$\vee$	$\wedge$	0	1
Pravděpodobnostní	$\mathcal{R}$	$\mathbb{R}_+$	$+$	$\times$	0	1
Logaritmický	$\mathcal{L}$	$\mathbb{R} \cup \{+\infty\}$	$\oplus_{\log}$	$+$	$+\infty$	0
Tropický	$\mathcal{T}$	$\mathbb{R} \cup \{+\infty\}$	min	$+$	$+\infty$	0
Tropický (jinak)	$\mathcal{T}'$	$\mathbb{R} \cup \{-\infty\}$	max	$+$	$-\infty$	0

Tabulka 1: Příklady některých polookruhů

Lemma 1 (Uspořádání) Je-li $(\mathbb{K}, \oplus, \otimes, \overline{0}, \overline{1})$ idempotentní polookruh, potom relace $\leq$, definovaná jako $\forall a, b \in \mathbb{K}$:

$$(a \leq b) \Longleftrightarrow (a \oplus b = a)$$

je *částečným* uspořádáním na množině $\mathbb{K}$ a nazývá se *přirozené uspořádání* na $\mathbb{K}$ [14]. Částečné se jmenuje proto, že připouští pro danou relaci na dané množině existenci dvojic neporovnatelných prvků. Praktický příklad: relace dělitelnosti na $\mathbb{N}$ je částečným uspořádáním, protože existují např. prvky {2, 3}, které nejsou vzájemně dělitelné. [7]
Dále relace $\leq$ jest uspořádáním *úplným* tehdy a jen tehdy, jestliže polookruh má takovou vlastnost cesty, že $\forall a, b \in \mathbb{K}$:

$$(a \oplus b = a) \vee (a \oplus b = b).$$

Úplné uspořádání se nazývá proto, že každé dva prvky z dané množiny jsou danou relací porovnatelné, tj. opět např. pro množinu $\mathbb{R}$ je relace $\leq$ úplným uspořádáním [7].

Definice 4 Idempotentní polookruh $(\mathbb{K}, \oplus, \otimes, \overline{0}, \overline{1})$ je *úplně uspořádaný*, jestliže jeho přirozené uspořádání jest úplným uspořádáním.

Definice 5 (Monotónní uspořádání) Nechť $\mathcal{K} = (\mathbb{K}, \oplus, \otimes, \overline{0}, \overline{1})$ je polookruh a $\leq$ částečné uspořádání na $\mathbb{K}$. $\leq$ je *monotónní* vzhledem ke $\mathcal{K}$, jestliže $\forall a, b, c \in \mathbb{K}$:

$$(a \leq b) \Longrightarrow (a \otimes c \leq b \otimes c) \wedge (c \otimes a \leq c \otimes b).$$

$\leq$ je *negativní* tehdy a jen tehdy, jestliže $\overline{1} \leq \overline{0}$.

Lemma 2 Jestliže polookruh $(\mathbb{K}, \oplus, \otimes, \overline{0}, \overline{1})$ je idempotentní, potom jeho přirozené uspořádání je monotónním uspořádáním.

Orientační přehled některých polookruhů nabízí tabulka 2.1. V odstavcích, popisujících proces vytváření indexu, budeme používat především následující dva polookruhy.

Definice 6 (Logaritmický polookruh) *Logaritmický* polookruh je pětice

$$\mathcal{L} = (\mathbb{R} \cup \{-\infty, +\infty\}, \oplus_{\log}, +, +\infty, 0)$$

kde

$$\forall a, b \in (\mathbb{R} \cup \{-\infty, +\infty\}) : a \oplus_{\log} b = -\log(e^{-a} + e^{-b}),$$

spolu se zavedenými konvencemi v podobě $e^{-\infty} = 0$ a `-log`$(0) = \infty$. $\mathcal{L}$ je skrze operaci `-log` izomorfní (tj. vzájemně jednoznačným zobrazením) k polookruhu kladných reálných čísel:

$$\mathcal{R} = (\mathbb{R}_+, +, \times, 0, 1).$$

Jak později uvidíme, tohoto polookruhu budeme využívat při výpočtech váhy cesty, tj. pravděpodobnosti výskytu hledaného výrazu v nějaké promluvě. Záporný logaritmus vah je zde použit proto, že při hledání cesty v grafu se díky tomu váhy po sobě jdoucích hran sčítají, kdežto v kdyby se násobily (jak je tomu v případě pravděpodobnostního polookruhu), mohlo by tak dojít k vynulování celkové pravděpodobnosti této cesty.

Definice 7 (Tropický polookruh) *Tropický* polookruh je definován jako

$$\mathcal{T} = (\mathbb{R} \cup \{-\infty, +\infty\}, \min, +, +\infty, 0)$$

kde jako tropický součtový operátor je zde užita operace minima. Poznamenejme, že $\mathcal{T}$ je idempotentní a jeho přirozené uspořádání

$$\forall a, b \in \mathbb{R} \cup \{-\infty, +\infty\} : (\min\{a, b\} = a) \iff (a \leq b)$$

jest úplným uspořádáním (obvyklé uspořádání reálných čísel) [14]. Tohoto polookruhu bude využito pro stanovení počátečního času hledaného výrazu v nějaké promluvě. Proces je analogický s tzv. *Vitterbiovým algoritmem* [6], kdy se pro každý symbol hrany spočítá cena cesty grafem operací + a ze všech cest, odpovídajících stejnému symbolu se pak vybere operací *min* ta nejlepší. To jest v našem případě nejmenší počáteční čas výskytu výrazu v mřížce.
Analogicky lze definovat jiný idempotentní polookruh s využitím operace maxima:

$$\mathcal{T}' = (\mathbb{R} \cup \{-\infty, +\infty\}, \max, +, -\infty, 0),$$

jehož přirozené uspořádání je jiný případ úplného uspořádání (převrácené uspořádání reálných čísel). Podobně s předchozím, bude tohoto polookruhu využito pro stanovení koncového času hledaného výrazu v nějaké promluvě.

Budoucí proces tvorby indexu využívá speciální struktury polookruhů, definované nad kartézským součinem uspořádaných množin. Jak později uvidíme, finální index je vlastně vážený transducer (viz dále) nad lexikografickým polookruhem tří tropických polookruhů. Proto zaveďme ještě následující definice.

Definice 8 (Součinový polookruh) *Součinový* polookruh dvou částečně uspořádaných polookruhů $\mathcal{A} = (\mathbb{A}, \oplus_{\mathbb{A}}, \otimes_{\mathbb{A}}, \overline{0}_{\mathbb{A}}, \overline{1}_{\mathbb{A}})$ a $\mathcal{B} = (\mathbb{B}, \oplus_{\mathbb{B}}, \otimes_{\mathbb{B}}, \overline{0}_{\mathbb{B}}, \overline{1}_{\mathbb{B}})$ je definován jako

$$\mathcal{A} \times \mathcal{B} = (\mathbb{A} \times \mathbb{B}, \oplus_{\times}, \otimes_{\times}, (\overline{0}_{\mathbb{A}}, \overline{0}_{\mathbb{B}}), (\overline{1}_{\mathbb{A}}, \overline{1}_{\mathbb{B}}))$$

kde $\oplus_{\times}$ a $\otimes_{\times}$ jsou operátory po složkách, tj. např. $\forall a_1, a_2 \in \mathbb{A}, b_1, b_2 \in \mathbb{B}$:

$$(a_1, b_1) \oplus_{\times} (a_2, b_2) = ((a_1 \oplus_{\mathbb{A}} a_2), (b_1 \oplus_{\mathbb{B}} b_2)).$$

Pro $\otimes_{\times}$ je předpis analogický. Pokud platí, že $\mathcal{A}$ a $\mathcal{B}$ jsou úplně uspořádané, pak přirozené uspořádání nad $\mathcal{A} \times \mathcal{B}$:

$$(a_1, b_1) \leq_{\times} (a_2, b_2) \iff ((a_1 \oplus_{\mathbb{A}} a_2) = a_1, (b_1 \oplus_{\mathbb{B}} b_2) = b_1)$$

definuje částečné uspořádání, zvané *součinové* uspořádání.

Definice 9 (Lexikografický polookruh) *Lexikografický* polookruh dvou částečně uspořádaných polookruhů $\mathcal{A} = (\mathbb{A}, \oplus_{\mathbb{A}}, \otimes_{\mathbb{A}}, \overline{0}_{\mathbb{A}}, \overline{1}_{\mathbb{A}})$ a $\mathcal{B} = (\mathbb{B}, \oplus_{\mathbb{B}}, \otimes_{\mathbb{B}}, \overline{0}_{\mathbb{B}}, \overline{1}_{\mathbb{B}})$ je definován jako

$$\mathcal{A} * \mathcal{B} = (\mathbb{A} \times \mathbb{B}, \oplus_{*}, \otimes_{*}, (\overline{0}_{\mathbb{A}}, \overline{0}_{\mathbb{B}}), (\overline{1}_{\mathbb{A}}, \overline{1}_{\mathbb{B}}))$$

kde $\otimes_{*}$ je operátor násobení po složkách a $\oplus_{*}$ je operátor lexikografické priority , tj. $\forall a_1, a_2 \in \mathbb{A}, b_1, b_2 \in \mathbb{B}$:

$$(a_1, b_1) \oplus_{*} (a_2, b_2) = \begin{cases} (a_1, b_1 \oplus_{\mathbb{B}} b_2) & a_1 = a_2 \\ (a_1, b_1) & a_1 = (a_1 \oplus_{\mathbb{A}} a_2) \neq a_2 \\ (a_2, b_2) & a_1 \neq (a_1 \oplus_{\mathbb{A}} a_2) = a_2 \end{cases}$$

Příkladem takového operátoru z praxe je např. obecně známé uspořádání podle abecedy. V našem případě později uvidíme, že poslouží při řazení výsledků vyhledávání podle časů a pravděpodobnosti výskytu hledaného výrazu. Na rozdíl od $\mathcal{A} \times \mathcal{B}$, zde platí, že pokud jsou $\mathcal{A}$ a $\mathcal{B}$ úplně uspořádané, pak přirozené uspořádání nad $\mathcal{A} * \mathcal{B}$, dané jako:

$$(a_1, b_1) \leq_{*} (a_2, b_2) \iff \begin{array}{c} (a_1 = (a_1 \oplus_{\mathbb{A}} a_2) \neq a_2) \\ \text{nebo} \\ ((a_1 = a_2) \wedge (b_1 = b_1 \oplus_{\mathbb{B}} b_2)) \end{array}$$

definuje úplné uspořádání, známé jako *lexikografické* uspořádání.

Obecněji řečeno, můžeme definovat součinové a lexikografické uspořádání nad kartézským součinem n uspořádaných množin. Nechť $\{\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_n\}$ je n-tice množin spolu s odpovídajícími úplnými uspořádáními $\{\leq_1, \leq_2, \ldots, \leq_n\}$. Součinové uspořádání $\leq_\times$ n-tice $\{\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_n\}$ je definováno jako

$$(a_1, a_2, \ldots, a_n) \leq_\times (b_1, b_2, \ldots b_n) \Longleftrightarrow a_i \leq_i b_i \ \ \forall\, i \leq n.$$

Podobně lexikografické uspořádání $\leq_*$ n-tice $\{\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_n\}$ je definováno jako

$$(a_1, a_2, \ldots, a_n) \leq_* (b_1, b_2, \ldots b_n) \Longleftrightarrow \exists\, m > 0, a_i = b_i,\ \forall\, i < m, a_m \leq_m b_m.$$

To jest, že pro jeden z výrazů platí $a_m \leq_m b_m$ a všechny předcházející výrazy jsou si rovny.

Poznamenejme, že součinový či lexikografický polookruh na $\{\mathbb{A}_1, \mathbb{A}_2, \ldots, \mathbb{A}_n\}$ lze rekurzivně definovat s využitím asociativity operátorů $\times$ či $*$ jako:

$$\mathcal{A}_1 \times \mathcal{A}_2 \times \ldots \mathcal{A}_n = (((\mathcal{A}_1 \times \mathcal{A}_2) \times \ldots) \times \ldots\ \mathcal{A}_n).$$

2.2 Vážené konečné automaty

V literatuře často dochází k záměně pojmů automat a akceptor tak, že se zdá jako by měly stejný význam. V dalších odstavcích budeme užívat pojmu automat v případech, kdy máme na mysli obecný konečný automat, tj. nerozlišujeme mezi pojmem transducer a akceptor.

Definice 10 (Transducer) *Vážený konečný transducer* T nad polookruhem $\mathbb{K}$ je osmice $T = (\Sigma, \Delta, Q, I, F, E, \lambda, \rho)$ kde:

Σ	je konečná vstupní abeceda,
Δ	je konečná výstupní abeceda,
Q	je konečná množina stavů,
$I \subseteq Q$	je množina počátečních stavů,
$F \subseteq Q$	je množina koncových stavů,
E	je konečná množina hran (přechodů): $E \subseteq Q \times (\Sigma \cup \epsilon) \times (\Delta \cup \epsilon) \times \mathbb{K} \times Q$,
$\lambda : I \mapsto \mathbb{K}$	je funkce počátečních vah a
$\rho : F \mapsto \mathbb{K}$	je funkce koncových vah.

Definice 11 (Akceptor) *Vážený konečný akceptor* A nad polookruhem $\mathbb{K}$ je sedmice $A = (\Sigma, Q, I, F, E, \lambda, \rho)$ která vznikne z transduceru prostým vypuštěním výstupní abecedy Δ. *Velikost automatu* M je definována jako $|M| = |Q| + |E|$.

Lemma 3 Nechť V je konečná množina (abeceda) symbolů a V^+ označuje množinu všech konečných neprázdných posloupností (řetězců) utvořených z prvků množiny V. Potom V^* označuje množinu $E^+ \cup \{\lambda\}$, kde λ označuje prázdný prvek množiny V.

Je-li dána hrana $e \in E$, označme jako

$i[e]$	její vstupní symbol,
$o[e]$	její výstupní symbol,
$w[e]$	její váhu,
$p[e]$	její zdrojový nebo-li předcházející stav a
$n[e]$	její cílový nebo následující stav.

Cesta $\pi = e_1 \dots e_k$ je prvkem E^* s po sobě následujícími hranami, splňujícími podmínku $n[e_{i-1}] = p[e_i],\ i = 2, \dots, k$. Rozšiřme n a p na cestu zvolením $n[\pi] = n[e_k]$ a $p[\pi] = p[e_1]$. Stejně tak můžeme rozšířit i vstupně-výstupní označení a váhy definováním

$$\begin{aligned} i[\pi] &= i[e_1] \dots i[e_k], \\ o[\pi] &= o[e_1] \dots o[e_k], \\ w[\pi] &= w[e_1] \otimes \dots \otimes w[e_k]. \end{aligned}$$

Rovněž rozšíříme váhu w na nějakou konečnou množinu cest Π definováním:

$$w[\Pi] = \bigoplus_{\pi \in \Pi} w[\pi].$$

Označme $\Pi(q, q')$ množinu cest z q do q', dále $\Pi(q, x, q')$ množinu cest z q do q' se vstupním řetězcem $x \in \Sigma^*$ a jako $\Pi(q, x, y, q')$ množinu cest z q do q' se vstupním řetězcem $x \in \Sigma^*$ a výstupním řetězcem $y \in \Delta^*$. Tyto definice mohou být rozšířeny na množiny stavů $S, S' \subseteq Q$ jako

$$\Pi(S, x, S') = \bigcup_{q \in S, q' \in S'} \Pi(q, x, q').$$

Úspěšná cesta automatem M je cesta z nějakého jeho počátečního stavu do nějakého jeho koncového stavu. Posloupnost symbolů x je automatem M přijata tehdy, jestliže existuje úspěšná cesta π s tímto vstupním řetězcem x. M je *jednoznačný*, jestliže pro nějaký řetězec $x \in \Sigma^*$ existuje nejvýše jedna úspěšná cesta se vstupním označením x.

Váha, přiřazená transducerem T nějaké dvojici vstupně-výstupních řetězců $(x, y) \in \Sigma^* \times \Delta^*$ je dána jako

$$[T](x,y) = \bigoplus_{\pi \in \Pi(I,x,y,F)} \lambda(p[\pi]) \times w[\pi] \times \rho(n[\pi])$$

a pokud je $\Pi(I, x, y, F) = \emptyset$ pak $[T](x, y)$ definujme rovno $\overline{0}$.

2.3 Faktorový automat

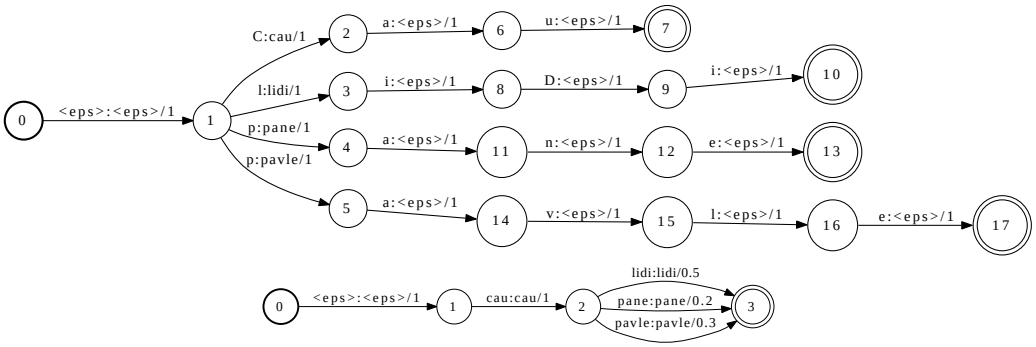
Definice 12 (Faktor) Jsou-li dány dva řetězce $u, v \in \Sigma^*$, v je *faktorem* (podřetězcem) u jestliže $u = xvy$ pro nějaká $x, y \in \Sigma^*$. Obecně v je faktorem jazyka $L \subseteq \Sigma^*$, jestliže v je faktorem nějakého řetězce $u \in L$.

Definice 13 (Faktorový automat) *Faktorový automat* $F(u)$ řetězce u je minimální deterministický konečný akceptor, přijímající (rozpoznávající) přesně množinu faktorů řetězce u. $F(u)$ lze sestrojit v lineárním čase a jeho velikost je lineární funkcí délky řetězce $|u|$ [15] [16].
Podobně označme jako $F(A)$ minimální deterministický akceptor, rozpoznávající množinu faktorů konečného akceptoru A, tj. rozpoznávající množinu všech faktorů všech řetězců, přijímaných automatem A. Dle práce [17] je velikost faktorového automatu $F(A)$ lineární k $|A|$ a též jej lze sestrojit v lineárním čase.

Poznámka: Operace $\otimes$ je u obou typů polookruhů $\mathcal{L}$ a $\mathcal{T}$ definována stejně. Z toho plyne, že v případě hranových vah není rozdíl v tom, jestli se díváme na automat M jako nad logaritmickým nebo tropickým polookruhem. Jak později zjistíme v odstavci o procesu vytváření indexu, $\mathcal{L}$ polookruh umožňuje v automatu sloučit nejednoznačné cesty (tj. se stejným vstupním symbolem a překrývajícími se časy výskytu) do jedné operací $\otimes_{\log}$. Na druhou stranu $\mathcal{T}$ polookruh je idempotentní a jeho přirozené uspořádání je úplným, což umožňuje váženému automatu nad tímto polookruhem použití algoritmů hledání nejlepší cesty a prořezávání. Fakt, že přirozené uspořádání je úplným, také umožňuje porovnávat váhy a jim odpovídající cesty mezi sebou. [4]

2.4 Operace s konečnými automaty

V této sekci jsou popsány některé základní operace nad váženými konečnými automaty. Jejich výčet není zdaleka úplný, ale jsou zde zmíněny ty, které jsou používány v textu této práce. Kompletní přehled všech najde čtenář např. v [5].



Obrázek 3: Příklad lexikálního slovníku a jazykového modelu, reprezentované s pomocí vážených konečných transducerů.

2.4.1 Kompozice

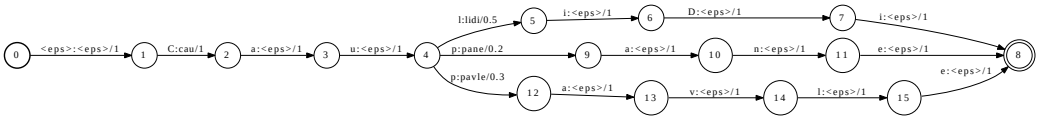
Vážený konečný transducer reprezentuje ohodnocenou binární relaci mezi vstupní a výstupní posloupností symbolů. Kompozice T dvou ohodnocených transducerů R a S je kompozicí jejich relací: $T = R \circ S$. Výsledkem je ohodnocený transducer T takový, že pro všechny cesty v R a S platí: existuje-li v R cesta, zobrazující posloupnost vstupních symbolů x na nějaku posloupnost výstupních symbolů y, a v S cesta, zobrazující posloupnost vstupních symbolů y na posloupnost výstupních symbolů z, pak v T existuje právě jedna cesta, zobrazující x na z [6]. Přitom váha cesty v T je $\otimes$-součinem vah odpovídajících si cest v R a S:

$$[R \circ S](x, z) = \bigoplus_{y} [R](x, y) \otimes [S](y, z),$$

za podmínky, že automaty jsou nad komutativními polookruhy. Protože je výpočet kompozice pro jednotlivé hrany lokální (závisí pouze na jejich počátečních a koncových stavech), lze výpočet provádět za běhu aplikace - tzv. *líná implementace* (angl.: lazy implementation). Přechody s označením ϵ musejí být řešeny speciálně. [18] Kompozici dvou transducerů z obrázku 3 vidíme na obrázku 4.

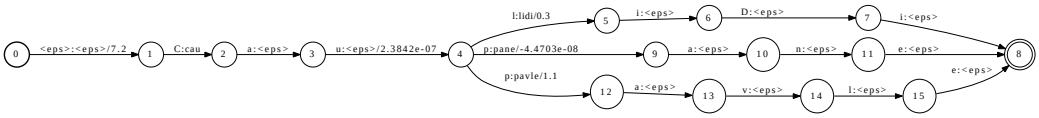
2.4.2 Stlačení vah

Stlačení vah (angl.: weight-pushing) spočívá v nalezení ekvivalentního automatu s jiným ohodnocením přechodů tak, že ohodnocení všech úspěšných cest automatem zůstane nezměněno. Tato ohodnocení se zpravidla přemisťují na jejich začátky (počáteční stavy a hrany), popř. alternativně



Obrázek 4: Kompozice transducerů z obrázku 3.

i na jejich konce. Přičemž platí, že váhy jsou normalizovány tak, že pro všechny výstupní hrany vedoucí ze stejného stavu je $\oplus$ součet jejich vah roven $\overline{1}$. V případě tropického polookruhu to může mít neblahý vliv při dekódování Vitterbiovým algoritmem, protože oproti původní cestě mají jejich jednotlivé části při prořezávání nerovnoměrná ohodnocení. Proto je u transducerů nad tropickým polookruhem lepší při stlačování použít vztahy pro logaritmický polookruh [18]. Složitost algoritmu stlačení vah je pro případ acyklických transducerů lineární k jejich velikosti, tj. $O(|A_i|)$ [5]. Podrobnější popis algoritmu nalezneme např. v [6] a také v [18]. Pro náš případ algoritmus stlačení vah nalezne uplatnění především jako předstupeň algoritmu minimalizace (viz dále), kdy jeho výsledkem je transducer, jehož přechody se stejným vstupně-výstupním označením mají v maximální možné míře stejné ohodnocení.



Obrázek 5: Transducer z obrázku 4 po aplikaci weight-pushingu.

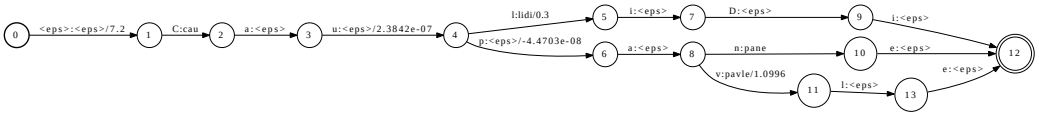
2.4.3 Determinizace

Vážený konečný automat je deterministický, jestliže pro každý jeho stav platí, že pro každé vstupní označení existuje z daného stavu nanejvýš jeden odpovídající přechod. Platí, že ke každému váženému konečnému akceptoru lze sestrojit ekvivalentní deterministický akceptor. Ne tak v případě vážených konečných transducerů. Zde platí, že transducer je determinizovatelný jestliže je funkcionální, tj. každý vstupní řetězec zobrazí na právě jeden výstupní řetězec. Jeho váhy musí být navíc dělitelné zleva [18] a polookruh $\mathbb{K}$ musí

mít takovou vlastnost, že $\forall a, b \in \mathbb{K}$:

$$a \oplus b = \overline{0} \Rightarrow a = b = \overline{0}.$$

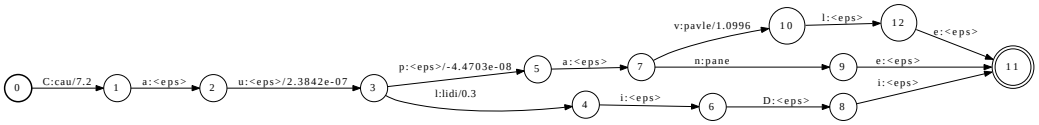
Poznamenejme, že tyto podmínky splňuje např. logaritmický nebo tropický polookruh [18]. Při determinizaci transduceru v porovnání s akceptorem, je třeba vzít na vědomí fakt, že přechody se stejným vstupním symbolem mohou mít různou váhu i různé výstupní symboly. Jako výstupní řetězec se proto volí nejdelší společný začátek výstupních řetězců a nejmenší ohodnocení (viz *Viterbiovo kritérium*). Deterministický transducer tedy neobsahuje redundantní cesty, čímž se minimalizuje velikost prohledávaného prostoru při jeho procházení (např. algoritmem hledání nejkratší cesty). Prohledávání je tedy rychlejší, ovšem za cenu vyšších nároků na úložný prostor, neboť procesem determinizace obvykle narůstá počet stavů automatu. Podrobný popis algoritmu determinizace nalezneme např. v [6]. Poznamenejme ještě, že algoritmus je možné provádět i jako línou implementaci.



Obrázek 6: Determinizace transduceru z obrázku 5.

2.4.4 Odstranění ϵ-hran

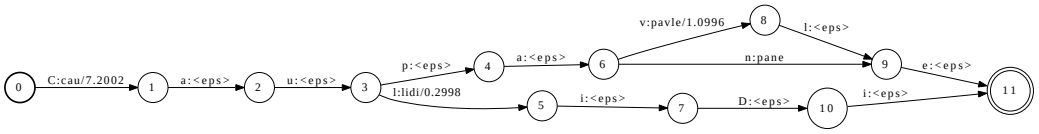
Tato operace ke vstupnímu automatu, obsahujícího hrany s prázdným symbolem ϵ, vytvoří ekvivalentní automat bez těchto hran. V případě transduceru musí být ϵ zároveň jako vstupní i výstupní symbol hrany. U vážených automatů přitom musí i po odstranění zůstat zachovány celkové váhy všech cest, které zahrnovaly původní vážené ϵ hrany. Opět jej lze provádět jako línou implementaci.



Obrázek 7: Odstranění ϵ-hran z transduceru z obrázku 6.

2.4.5 Minimalizace

Vstupem algoritmu je deterministický vážený automat a výstupem je ekvivalentní deterministický automat s minimálním početm stavů a přechodů. Nejjednodušší případ představuje vážený akceptor, kdy se dvojice (vstupní symbol, váha) převedou na jeden symbol. Tím vznikne nevážený akceptor na který se aplikuje nevážený algoritmus minimalizace [5]. V případě transduceru musíme počítat navíc s přítomností výstupního symbolu. Nejlepších výsledků minimalizace lze dosáhnout předchozí aplikací stlačení vah, kdy hrany se stejným vstupně-výstupním označením mají v maximální možné míře i stejnou váhu. Minimalizaci transduceru z obr. 7 vidíme na obr. 8



Obrázek 8: Minimalizace transduceru z obrázku 7.

2.4.6 N nejlepších cest

Algoritmus ze vstupního automatu vypočte n nejlepších cest a vrátí je ve formě jiného odpovídajícího automatu. Postup je analogický s klasickým Vitterbiovým algoritmem (např. v [6]), kdy n-tá nejlepší cesta je cesta s n-tou nejmenší váhou, danou přirozeným uspořádáním polookruhu automatu. Poznamenejme, že v úlohách ASR představují váhy `-log` pravděpodobnosti, proto cesta s nejmenší váhou je cestou s největší pravděpodobností. Podmínkou aplikovatelnosti algoritmu je úplně uspořádaný polookruh. Nejkratší cestu minimalizovaným automatem z obr. 8 ukazuje obr. 9

2.4.7 Prořezávání

Algoritmus z automatu odstraní všechny cesty, jejichž váha je větší než váha nejlepší cesty vynásobená operací $\otimes$ zadaným prahem p. Podmínkou je opět vlastnost úplně uspořádaného polookruhu. Příklad prořezání vidíme na obrázku 10.

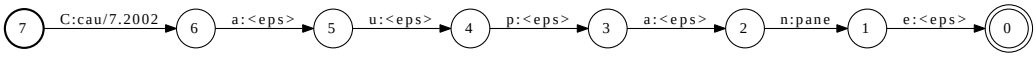
2.4.8 Projekce

Touto operací z transduceru vytvoříme akceptor, který bude výsledkem projekce na vstupní (nebo výstupní) symboly transduceru. Tj. např. v

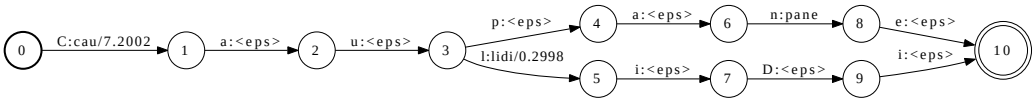
případě projekce na vstup Π_1 se zkopírují vstupní symboly transduceru T_1 i do jeho výstupních:

$$[\Pi_1(T)](x) = \bigoplus_y [T](x, y).$$

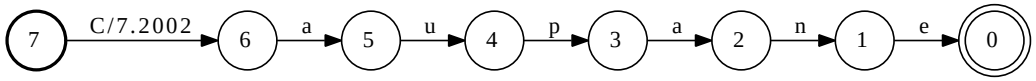
Takový transducer pak lze převést na akceptor prostým vypuštěním výstupních symbolů. Projekci transduceru s nejlepší cestou z obr. 9 na jeho vstupní symboly nalezneme na obr. 11.



Obrázek 9: Nejkratší cesta automatem z obr. 8.



Obrázek 10: Automat z obr. 8 po prořezání s prahem $p = 1$.

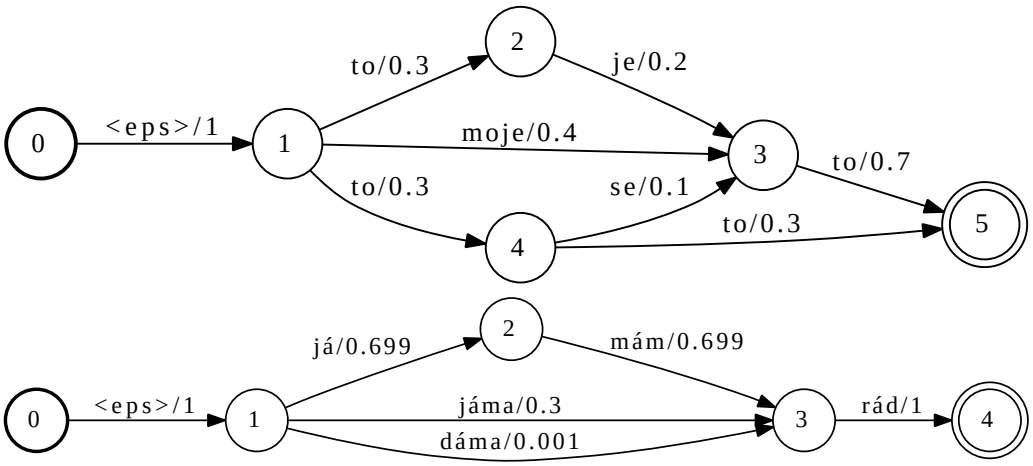


Obrázek 11: Projekce transduceru z obrázku 9 na vstup, čímž z něj vzniká akceptor.

3 Indexace a vyhledávání s váženými konečnými transducery

Následující sekce je v převážné většině citací prací [3], potažmo [4]. I přesto jsem však pro přehlednost a rychlejší orientaci na mnoha místech zachoval i původní odkazy na subzdroje, ze kterých autoři čerpali.

Předpokládejme, že řečový archiv obsahuje množství jednotlivých promluv u_i, $i = 1, \ldots, n$ a pro každou takovou promluvu máme k dispozici odpovídající slovní nebo fonémovou mřížku spolu se seznamem časování jejích vrcholů, která je výstupem dekodéru systému ASR (viz sekce 1.2.2). Mřížka strukturou odpovídá konečnému váženému akceptoru, kde vstupní symboly Σ odpovídají slovům, či fonémům a váhy jednotlivých přechodů odpovídají jejím pravděpodobnostem, daným akustickým a jazykovým modelem. Takovýto automat přijímá pouze řetězce, vyskytující se v jemu odpovídající promluvě. Příklad vidíme na obrázku 12.



Obrázek 12: Ukázkový hypotetický příklad mřížek tak, jak jsou produkovány dekodérem systému ASR. Časování stavů je $t_1 = [0, 1, 2, 4, 3, 5]$, $t_2 = [0, 1, 2, 3, 4]$

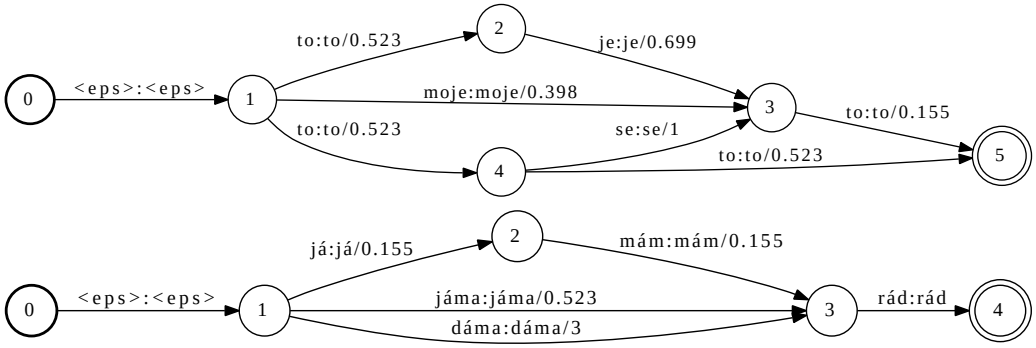
Problém vyhledávání nad množinou těchto mřížek spočívá v konstrukci indexu pro přímé vyhledávání libovolného podřetězce (faktoru) každé takové mřížky. Ten by měl každý vyhledávaný řetězec zobrazit na čtveřici (u_i, t_s, t_e, p) kde u_i je identifikátor mřížky (promluvy), t_s a t_e jsou počátek a konec časového intervalu výskytu hledaného řetězce a p je pravděpodobnost že se jedná o tento řetězec v tomto časovém intervalu. V následující sekci

bude popsána konstrukce takového indexu, který je založen na tzv. časovém faktorovém transduceru (angl. Timed Factor Transducer).

3.1 Předzpracování

Počáteční fázi konstrukce představuje předzpracování, jehož cílem je normalizace vah přechodů v mřížce nad logaritmickým polookruhem $\mathcal{L}$ a také vzájemné oddělení časově se nepřekrývajících výskytů stejných vstupních symbolů ze stejné mřížky. Volitelně lze též aplikovat operaci prořezávání a změnšit tak množství hypotéz vstupní mřížky a tedy i objem zpracovávaných dat (viz odstavec 1.2.2).

Vstupem algoritmu je transducer A_i ve standardním formátu knihovny OpenFST [5], odpovídající slovní nebo fonémové mřížce, spolu se seznamem časování jeho vrcholů. Váhy jednotlivých přechodů představují pravděpodobnost, přiřazenou vstupnímu symbolu přechodu systémem ASR a jsou nad tropickým polookruhem $\mathcal{T}$. Ukázkový příklad takových mřížek vidíme na obrázku 13. Na každou takovou mřížku lze aplikovat obecný algoritmus stlačení vah nad logaritmickým polookruhem (viz 2.4.2), který převede pravděpodobnosti na jejich `-log` podobu, pokud se v ní již nenachází. Zároveň je normalizuje tak, že $\oplus$ součet všech vah, vycházejících z nějakého stavu je roven jedné.



Obrázek 13: Mřížky z obrázku 12 převedené do podoby WFST nad polookruhem $\mathcal{T}$, časy vrcholů $t_1 = [0, 1, 2, 4, 3, 5]$, $t_2 = [0, 1, 2, 3, 4]$.

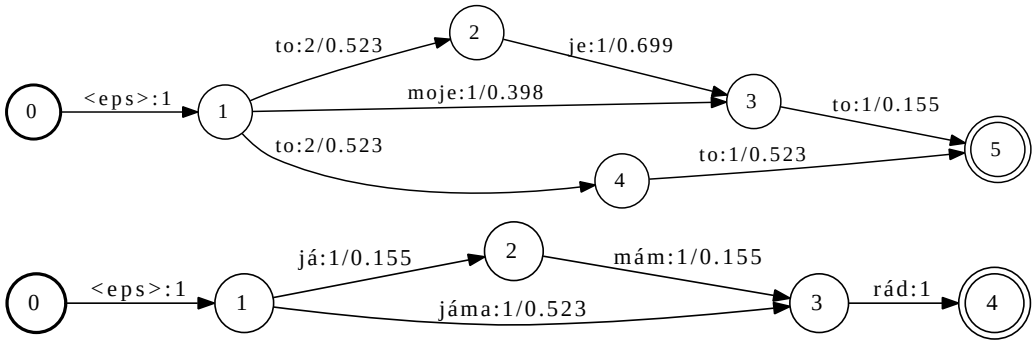
V případě úlohy STD je nezbytné indexovat samostatně každý výskyt i stejných podřetězců, pokud se v promluvě nacházejí v odlišných časových intervalech. Toho lze dosáhnout seskupením hran mřížky, nesoucích stejné

vstupní symboly s překrývajícím se časovým intervalem, dohromady. Jednoprůchodový shlukovací algoritmus, popsaný v [4], procházel mřížkou dle topologického uspořádání a mohlo tak dojít k nežádoucímu sloučení nepřekrývajících se hran (prostřednictvím nějaké paralelní hrany se stejným symbolem, časově překrývající obě slučované hrany). Proto autoři přistoupili k jinému, tentokrát dvouprůchodovému algoritmu:

Pro každý vstupní symbol $i \in \Sigma$:

- Roztřiď odpovídající dvojice (t_s, t_e) podle t_e.
- Identifikuj největší množinu nepřekrývajících se dvojic (t_s, t_e) a vytvoř z ní množinu identifikátorů shluků hran.
- Klasifikuj do těchto shluků všechny ostatní hrany podle maximálního překrytí časových intervalů.

Takto postavený algoritmus zaručí převedení původního akceptoru A_i na transducer B_i, kde každá hrana nese na svém výstupním symbolu identifikátor shluku časově se překrývajících hran se stejným vstupním symbolem. Jinými slovy každá dvojice (vstupní symbol, výstupní symbol) představuje identifikátor nějakého hranového shluku. Výsledek algoritmu aplikovaného na mřížku v obrázku 13 vidíme na obrázku 14. Z obrázku je mimo jiné patrné, že prořezáním zmizely některé hrany s nízkou váhou.



Obrázek 14: Mřížky z obrázku 13 po aplikaci předzpracování s prořezáním $p = 1$ nad polookruhem $\mathcal{L}$, časy vrcholů $t_1 = [0, 1, 2, 4, 3, 5]$, $t_2 = [0, 1, 2, 3, 4]$.

3.2 Odvození pokročilého faktorového transduceru

V tomto kroku je z každé předzpracované mřížky odvozen tzv. pokročilý faktor transducer, který přijímá přesně množinu všech jejích faktorů a

je deterministický. Děje se tak prostřednictvím rozšíření každého faktoru předzpracované mřížky o jednoznačný symbol a dále aplikací optimalizačních kroků.

Nechť $B_i = (\Sigma, \Delta, Q_i, I_i, F_i, E_i, \lambda_i, \rho_i)$ označuje transducer bez ϵ-přechodů (tj. přechodů s prázdným vstupním symbolem), nad logaritmickým polookruhem $\mathcal{L}$, získaný aplikací předzpracování akceptoru A_i. Výstupní řetězec, přiřazený transducerem (mřížkou) B_i nějakému přijatému vstupnímu řetězci, představuje identifikátor shluku hran. Váhy této mřížky lze interpretovat jako logaritmické věrohodnosti výskytu hledaného výrazu v promluvě u_i, dané modelem, generujícím mřížku. Obecně řečeno, B_i definuje pravděpodobnostní rozložení P_i nad všemi dvojicemi vstupně-výstupních řetězců $(x, y) \in \Sigma^* \times \Delta^*$, kde $P_i(x, y)$ je dáno součtem všech cest (x, y) v B_i.
Pro každý stav $q \in Q_i$ označme jako $d[q]$ nejkratší vzdálenost z I_i do q (nebo `-log` dopředné pravděpodobnosti - viz *forward-backward* algoritmus např. v [6]) a jako $f[q]$ nejkratší vzdálenost z q do F_i (nebo `-log` zpětné pravděpodobnosti):

$$d[q] = \bigoplus_{\pi \in \Pi(I_i, q)}^{log} (\lambda_i(p[\pi]) + w[\pi]) \tag{3}$$

$$f[q] = \bigoplus_{\pi \in \Pi(q, F_i)}^{log} (w[\pi] + \rho(n[\pi])) \tag{4}$$

Protože je B_i acyklický, lze vzdálenosti $d[q]$ a $f[q]$ spočítat pro všechny stavy $q \in Q_i$ v lineárním čase [14].
Nechť Π_i označuje množinu cest v B_i a $C_{x,y}(u_i)$ označuje počet výskytů dvojice vstupně-výstupních řetězců (x, y) v promluvě u_i. Potom rovnice

$$-\log(E_{P_i}[C_{x,y}(u_i)]) = \bigoplus_{\substack{i[\pi] = x, o[\pi] = y, \\ \pi \in \Pi_i}}^{\log} d[p[\pi]] + w[\pi] + f[n[\pi]] \tag{5}$$

dává očekávaný počet výskytů dvojice (x, y) v u_i nebo ekvivalentně aposteriorní pravděpodobnost $p((x, y)|u_i)$. Podobně nechť $t_i[q]$ označuje časovou informaci o stavu $q \in Q_i$ a $t_i^s(x, y)$ resp. $t_i^e(x, y)$ označují počáteční resp. koncový čas shluku hran (x, y) v promluvě u_i. Potom

$$t_i^s(x, y) = \min_{\substack{i[\pi] = x, o[\pi] = y, \\ \pi \in \Pi_i}} t_i[p[\pi]], \tag{6}$$

$$t_i^e(x,y) = \max_{\substack{i[\pi]=x,\, o[\pi]=y,\\ \pi \in \Pi_i}} t_i[n[\pi]], \tag{7}$$

Rovnice 3, 4 a 5 definují hodnoty, které chceme uložit pro každý shluk faktorů, tj. zkonstruovat index v podobě transduceru, který bude zobrazovat každý shluk faktorů (x, y) na trojici $\{-\log(E_{P_i}[C_{x,y}(u_i)]), t_i^s(x,y), t_i^e(x,y)\}$. Proces tvorby takového indexu, tj. odvození časového faktorového transduceru T_i nad $\mathcal{T} * \mathcal{T} * \mathcal{T}$ z váženého konečného transduceru B_i nad $\mathcal{L}$ a časováním jeho stavů t_i je popsán v následujících čtyřech krocích:

1. *Generování faktorů.* Nejprve z B_i zkonstruujeme transducer, který indexuje každý výskyt faktorů odděleně. Tj. každá cesta mřížkou odpovídá výskytu jednoho faktoru a váha této cesty dává odpovídající trojici (apost. pravděpodobnost, počáteční čas, koncový čas). V obecném případě indexujeme všechny faktory takto:

   - Zobraz váhu každé hrany tak, že:

     $$w \in \mathcal{L} \longrightarrow \{w, \overline{1}, \overline{1}\} \in \mathcal{L} \times \mathcal{T} \times \mathcal{T}';$$

   - Vytvoř nový počáteční stav $q_I \notin Q_i$;
   - Vytvoř nový koncový stav $q_F \notin Q_i$;
   - $\forall q \in Q_i$ vytvoř dvě nové hrany:
     - počáteční hranu $(q_I, \epsilon, \epsilon, \{d[q], t_i[q], \overline{1}\}, q)$,
     - koncovou hranu $(q, \epsilon, i, \{f[q], \overline{1}, t_i[q]\}, q_F)$.

   Výsledek po této operaci ukazuje obrázek 15.

2. *Slučování faktorů.* Pro finální zobrazení, kterého chceme dosáhnout, optimalizujeme transducer nad součinovým polookruhem $\mathcal{L} \times \mathcal{T} \times \mathcal{T}'$ z předchozího kroku tak, že se na něj díváme jako na akceptor a sloučíme všechny cesty s překrývajícími se dvojicemi stejných faktorů (x, y). Tj. zakódujeme vstupně-výstupní symboly do jednoho vstupního symbolu, aplikujeme vážené odstranění ϵ-hran, determinizaci a minimalizaci nad polokruhem $\mathcal{L} \times \mathcal{T} \times \mathcal{T}'$. Výskyty stejných faktorů jsou zde sloučeny do jedné cesty operací $\oplus_{\log}$ pro jejich váhy nad polookruhem $\mathcal{L}$, operací `min` pro jejich počáteční časy nad polookruhem $\mathcal{T}$ a operací `max` pro jejich koncové časy nad polookruhem $\mathcal{T}'$ (obr. 16).
   Po tomto sloučení je výhodnější dále pracovat s těmito váhami jako s

lexikografickým polokruhem, proto zobrazíme stávající součinový polookruh $\mathcal{L} \times \mathcal{T} \times \mathcal{T}'$ na $\mathcal{T} * \mathcal{T} * \mathcal{T}$:

$$w_1, w_2, w_3 \in \mathcal{L} \times \mathcal{T} \times \mathcal{T}' \longrightarrow w_1, w_2, w_3 \in \mathcal{T} * \mathcal{T} * \mathcal{T}.$$

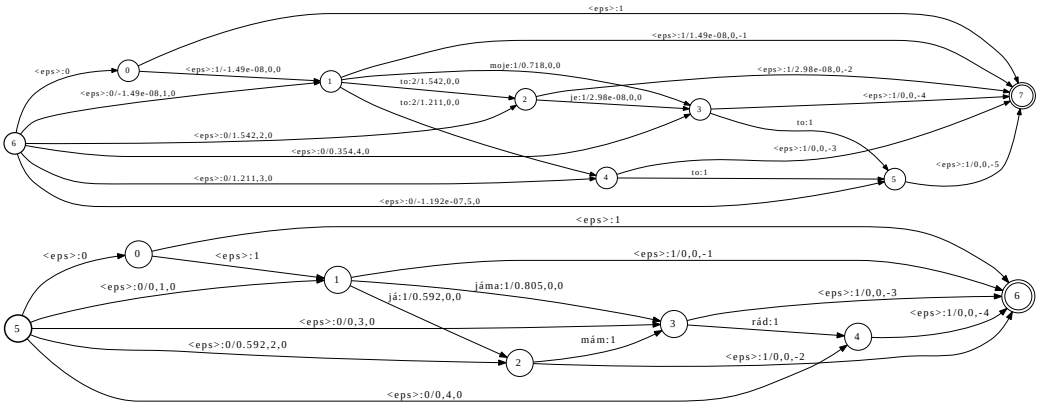
Poznamenejme, že lexikografický polookruh $\mathcal{T} * \mathcal{T} * \mathcal{T}$ je ekvivalentní s tropickým polookruhem nad $\mathbb{R}^3$. Toto zobrazení umožňuje operace prořezávání, hledání nejlepší cesty ale také např. pozdější řazení výsledků vyhledávání (v přirozeném uspořádání tohoto lexikografického polookruhu). Jak již bylo zmíněno v odstavci 2.3, tento rozdílný pohled na polookruhy je možný proto, že operace $\otimes$ je u obou typů polookruhů $\mathcal{L}$ a $\mathcal{T}$ definována stejně a celkové váhy cest se tak touto změnou polookruhu nemění.

3. *Zjednoznačnění faktorů.* Pokročilý transducer z předchozího kroku (označme jej $\tilde{T}_i$) není deterministický na množině vstupních symbolů. Abychom toho dosáhli, nejprve před vlastní determinizací odstraníme již nepotřebné identifikátory shluků a rozšíříme každou cestu o jednoznačný symbol, který při následné optimalizaci zaručí, že nepřekrývající se výskyty stejných faktorů budou od sebe odlišeny. V tomto kroku tedy vložíme do koncových hran mřížky jednoznačný vstupní symbol v podobě čísla výchozího stavu. Tj. $\forall e \in E_i$:
   - pokud $n[e] \notin F_i$ pak přiřaď $o[e] = \epsilon$;
   - pokud $n[e] \in F_i$ pak přiřaď $i[e] = p[e]$.
4. *Optimalizace.* Výsledek předchozího kroku lze tedy optimalizovat opět tak, že se na něj díváme jako na akceptor a aplikujeme váženou determinizaci a minimalizaci nad $\mathcal{T} * \mathcal{T} * \mathcal{T}$ polookruhem. Výsledný transducer T_i je deterministický a každá jeho cesta odpovídá nějaké vstupně-výstupní dvojici faktorů transduceru $\tilde{T}_i$.

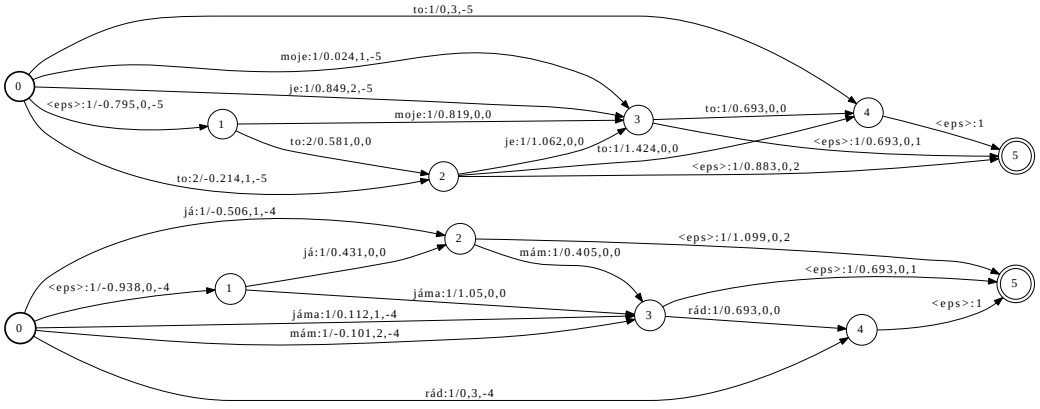
Proces generování faktorů vytváří pokročilý faktorový transducer, který zobrazuje přesně množinu faktorů B_i na identifikátor promluvy i (možno i vícekrát s rozdílnými váhami). Jak již bylo zmíněno, během slučování faktorů dochází ke sjednocení překrývajících se faktorů do jedné cesty. Nechť $\tilde{T}_i$ označuje výsledek této operace (obr. 16). Z rovnic 3, 4 a 5 tedy plyne, že váha, přiřazená tímto transducerem nějakému faktoru $(x, y) \in \Sigma^* \times \Delta^*$ bude skutečně trojice:

$$[\tilde{T}_i](x, yi) = \{-\log(E_{P_i}[C_{x,y}(u_i)]), t_i^s(x, y), t_i^e(x, y)\}, \tag{8}$$

kde yi označuje zřetězení výstupního symbolu (identifikátoru shluku) a identifikátoru promluvy (mřížky).



Obrázek 15: Konstrukce pokročilého faktorového transduceru z předzpracovaných automatů na obr. 14 - po operaci generování faktorů.

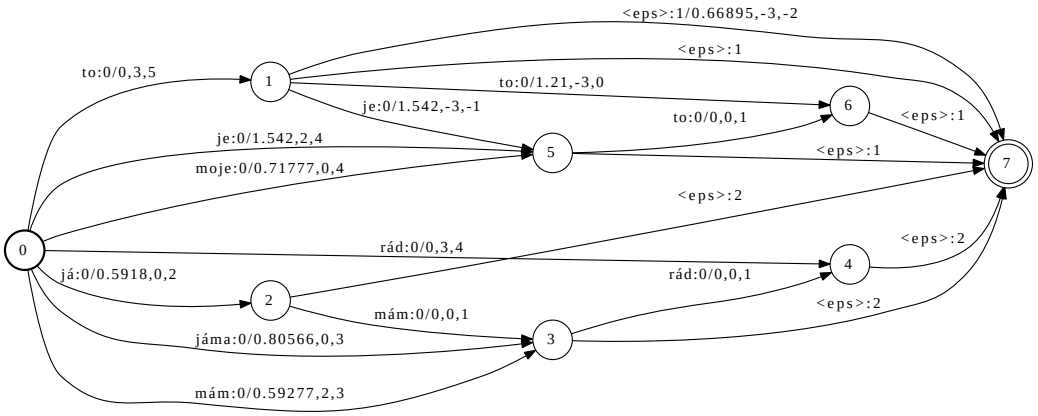


Obrázek 16: Konstrukce pokročilého faktorového transduceru z předzpracovaných automatů na obr. 14 - po optimalizaci nad $\mathcal{L} \times \mathcal{T} \times \mathcal{T}'$ během operace slučování faktorů.

3.3 Finální časový faktorový transducer

Finální časový faktorový transducer T nad souborem všech mřížek získáme

- sjednocením jednotlivých transducerů, vzniklých procesem z přechozí sekce 3.2:
$$U = \bigcup_i T_i,\ i = 1, \ldots, n;$$
- zakódováním vstupně-výstupních symbolů U jako jednoho vstupního symbolu a aplikací váženého odstranění ϵ-hran, determinizace a minimalizace nad $\mathcal{T} \times \mathcal{T} \times \mathcal{T}$ polookruhem (opět díky jednoznačnému symbolu z operace 3.2.3 nedojde ke sloučení nepřekrývajících se hran se stejnými vstupními symboly);
- dekódování symbolů zpět a odstranění již nepotřebných jednoznačných vstupních symbolů na koncových hranách;
- a definováním T jako transduceru, získaného po setřídění hran s ohledem na jejich vstupní symboly [5].

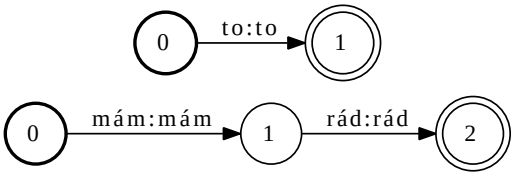


Obrázek 17: Finální index nad celou množinou mřížek z obrázku 14.

Druhý krok s optimalizací pouze slučuje jednotlivé cesty mezi mřížkami $T_i,\ i = 1, \ldots, n$ a každá úspěšná cesta indexem T dává na výstup jednoznačný identifikátor mřížky, ze které pochází. Pokud je to nutné, lze možné provádět během optimalizace operaci prořezávání díky úplnému uspořádání polookruhu $\mathcal{T} * \mathcal{T} * \mathcal{T}$. Poslední krok s tříděním hran je volitelný a lze jej vypustit. Výsledný optimální transducer nad celou množinou mřížek z obrázku 14 vidíme na obrázku 17.

3.4 Vyhledávání v časovém faktorovém transduceru

Silnou stránkou této formy indexace je, že uživatelský dotaz může být libovolný konečný automat X, tj. např. řetězec, booleovský dotaz nebo regulární výraz, zkompilované do podoby konečného automatu. Malou ukázku vidíme na obrázku 18.

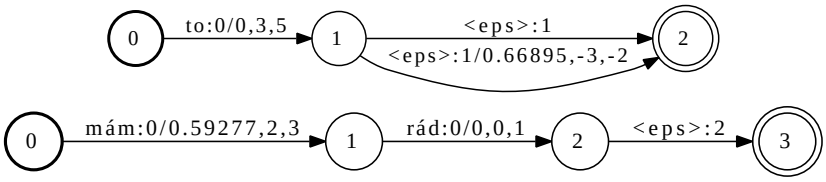


Obrázek 18: Ukázka jednoduchých dotazů na textové řetězce ve formě WFST.

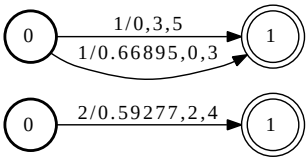
Odpověď R na tento dotaz bude jiný automat, získaný

- kompozicí transducerů X a T (obr. 19) a projekcí výsledného transduceru na jeho výstupní symboly: $\Pi_2(X \circ T)$ [5],
- odstraněním ϵ-hran a setřídění algoritmem nejlepší cesty.

R je jednoduchý akceptor s r hranami mezi počátečním a koncovým stavem (příklad opět na obrázku 20). Každá jeho úspěšná cesta π má jako vstupní řetězec $i[\pi]$ symbol identifikátoru automatu (promluvy, mřížky) a trojici (`-log` apost. pravděpodobnosti, počáteční čas, koncový čas) na svojí váze $w[\pi] \in \mathcal{T} * \mathcal{T} * \mathcal{T}$. Průchod přes cesty R dává výsledky v přirozeném uspořádání, daném polookruhem $\mathcal{T} * \mathcal{T} * \mathcal{T}$. R lze před tímto průchodem prořezávat a tím získat jen určitý počet nejlepších výsledků, navíc stanovením různých hodnot prořezání lze stanovit různé pracovní body získaných výsledků.



Obrázek 19: Kompozice transducerů na obr. 18 s indexem na obr. 17.



Obrázek 20: Příklad transducerů R s výsledky vyhledávání. Vstupní symbol nese identifikátor mřížky, váha pak udává skóre, počáteční a koncový čas výskytu.

4 Indexace a vyhledávání nad seznamy mřížkových hran

Následující kaptiola je věnována obecnému principu systému pro indexaci a vyhledávání ve slovních a fonémových mřížkách, popsanému v pracích [1] [2]. Ten je, narozdíl od předchozího přístupu, v současné době již v provozu nad českou částí multimediálního archivu projektu MALACH. Více informací o indexovaných datech a o tomto projektu je také pojednáno v odstavci 5.1.

4.1 Indexace

Důležitým rozdílem oproti technologii založené na vážených konečných transducerech je, že v tomto systému nebyly indexovány celé mřížky ale jen jen jejich vybrané hrany s výskyty jednotlivých slov nebo hlásek. Index tak zcela upouští od jejich strukturálních vlastností. Realizován byl s pomocí SQL databáze, která umožňuje ukládání a indexaci velkého počtu záznamů. Pro zvýšení přesnosti vyhledávání a řešení problému OOV slov byl zkonstruován index nad oběma typy mřížek - slovními i fonémovými.

4.1.1 Slovní mřížky

V případě slovních mřížek byla indexace poměrně jednoduchá, do databáze byly uloženy záznamy v podobě pětic pro každou hranu mřížky:

$$(t_s, t_e, slovo, skore, id\_videa),$$

kde *slovo* je slovo (položka ve slovníku ASR systému), *skore* je aposteriorní pravděpodobnost, *id_videa* je identifikátor původního video souboru a (t_s, t_e) je časový interval výskytu, relativní k začátku daného videosouboru. Kvůli zmenšení velikosti výsledného indexu byly aplikovány dva způsoby prořezávání. Zaprvé, do indexace byly zahrnuty jen ty hrany, jejichž skóre bylo větší nebo rovno stanovenému prahu $\theta_w = 0,05$. Zadruhé, pokud se vyskytovaly nějaké dvě hrany se stejným slovem a překrývajícím se časem výskytu (s prahem časových rozdílu překrytí $\Delta t_w < 0,5s$), byla z nich vybrána pouze hrana s nejvyšším skóre.

4.1.2 Fonémové mřížky

U fonémových mřížek byla situace o něco složitější. Při indexaci jednotlivých fonémů nebyly dotazy do databáze dostatečně specifické a vracely velmi často mnoho nesouvisejících výsledků (falešných poplachů - viz odstavec 6.1). Za základní jednotku indexace byly proto zvoleny překrývající

se trigramy po sobě jdoucích fonémů. Tyto tri-gramy byly získány z vygenerovaných mřížek použitím prohledávání do šířky. I zde bylo použito prořezávání a sice tak, že do indexu nebyly zahrnuty ty tri-gramy, které splňovaly některou z následujících podmínek:

- jeden nebo více fonémů v daném tri-gramu bylo symbolem pro ticho,
- některé dva fonémy byly totožné,
- skóre některého fonému bylo menží než práh $\theta_p = 0,05$.

Každému tri-gramu byl přiřazen odpovídající časový interval a jejich skóre vypočítáno jako geometrický průměr dílčích skóre jednotlivých fonémů. Tj. pro skóre $\{w_1, w_2, w_3\}$ odpovídajících hlásek každého tri-gramu, bylo celkové skóre $\mathcal{W}$:

$$\mathcal{W}(w_1, w_2, w_3) = \sqrt[3]{w_1 \cdot w_2 \cdot w_3}.$$

V porovnání s aritmetickým měl geometrický průměr tu vlastnost, že vracel nižší celkové skóre v případech, kdy některé z dílčích skóre trifonu bylo výrazně nižší. To ve výsledku vedlo k žádoucí eliminaci málo pravděpodobných cest v mřížce. V neposlední řadě tento způsob ulehčoval i výpočet vlastního skóre, protože pravděpodobnosti v mřížkách byly dány v `-log` podobě.
Další dvě kritéria prořezávání jsou analogická se slovními mřížkami. Celkové skóre tri-gramu muselo být větší nebo rovno než $\theta_C = 0,1$ a posun překrytí časových intervalů tri-gramů se stejnými symboly nesměl být menší než $\Delta t_p < 0,03s$.

4.2 Vyhledávání

V případě vyhledávání nad slovními mřížkami je princip následující. Dotazy na jednotlivá slova mohou být zobecněny tak, že k nim lze vygenerovat všechny uvažované fonetické transkripce (tj. převod na posloupnost jednotlivých fonémů) tak, jak mohou být alternativně vysloveny. Příkladem budiž slovo Shoah[8] v Angličtině vs Šoá v Češtině. Tyto vygenerované fonetické transkripce jsou pak zpětně zobrazeny na odpovídající slova ve slovníku. Fonetickou transkripci zabezpečuje systém na bázi pravidlového přístupu, který byl na Katedře kybernetiky ZČU k tomu již dříve vyvinutý.

[8]Shoah je hebrejským ekvivalentem pro Holokaust, někdy užívaný i v Anglickém jazyce. Doslova znamená katastrofu nebo zničení.

Kromě toho je volitelně možné vyhledávat všechny tvary hledaného slova. Tento problém je řešen tak, že je hledaný výraz převeden na tzv. lemmatizovaný tvar, tj. slovo v jeho základním tvaru. Poté je možné ve slovníku vyhledat všechny uvažované tvary hledaného slova na základě jeho lemmatu.

Pokud se však základní tvar slova ve slovníku LVCSR systému nevyskytuje (jedná se o OOV slovo) proběhne vyhledávání nad fonémovými mřížkami:

1. Opět je vygenerována fonetická transkripce hledaného výrazu, tj. např. pro slovo "válka"je to "v aa l", "aa l k", "l k a".

2. Všechny tyto vygenerované trigramy jsou vyhledány v databázi a seřazeny podle ID video nahrávky a počátečního času výskytu.

3. Pro každé ID videa jsou nalezené trigramy seřazeny a seskupeny podél časové osy tak, že časové roezstupy mezi jednotlivými skupinami trigramů jsou větší nebo rovno $\theta_s = 0,2s$.

4. Algoritmus nevyžaduje nutně výskyt všech trigramů z hledaného slova. Proto je každému seskupení trigramů přiřazeno skóre S, definované jako:
$$S = (1 - \lambda) \cdot S_{ACM} + \lambda \cdot S_{hit},$$
kde S_{ACM} je aritmetický průměr skóre jednotlivých trigramů ve skupině nalezených trigramů, S_{hit} je poměr mezi počtem trigramů ve skupině nalezených trigramů a počtem všech trigramů, reprezentujících hledané slovo, a λ je interpolační koeficient. Jeho hodnota byla experimentálně stanovena jako $\lambda = 0,6$.
Toto finální skóre S pak zastupuje míru důvěry nalezeného výskytu slova.

Bystrý čtenář snadno nahlédne, že takový způsob vyhledávání neumožňuje dotazování víceslovných frází. K tomu je určen ještě jiný algoritmus, popsaný následujícím způsobem. Každé slovo v hledané frázi může být označeno buď za *povinné* nebo *doplňkové*. Vyhledávací algoritmus poté:

1. Vyhledá jednotlivá slova a seřadí je podél časové osy obdobně jako v předchozím odstavci u tri-gramů OOV slov. Časový rozestup mezi skupinami slov je zde zvolen $\theta_f = 10s$.

2. Vyloučí všechny výsledky, které neobsahují povinná slova.

3. Každé skupině slov pak přiřadí míru důvěry, vypočítanou jako aritmetický průměr skóre jednotlivých slov ve frázi.

5 Implementace

5.1 Data

5.1.1 Stručně o projektu MALACH

V souvislosti s exeprimentálními daty nejprve krátce uvedu, odkud pocházejí a o jaká data se jedná. V létech 2002 - 2007 spolupracovala Katedra Kybernetiky Západočeské Univerzity v Plzni, spolu Karlovou Univerzitou v Praze a mnoha dalšími subjekty z USA na projektu nazvaném MALACH[9]. Stěžejním úkolem projektu byl vývoj metod pro snadnější přístup k vyhledávání v rozsáhlých vícejazyčných archivech mluvené řeči. Konkrétně se jednalo o archiv video rozhovorů se svědeckými výpověďmi přeživších pamětníků Holokaustu, který obsahoval více než 52 000 výpovědí v 32 světových jazycích, čítajících dohromady kolem 116 000 hodin nahrávek. Výpovědi byly shromážděny mezi roky 1994 - 1997 nadací Survivors of the Shoah VHF[10]. Katedra Kybernetiky ZČU se podílela na vývoji a ověřování technologií pro rozpoznávání řeči, indexaci a vyhledávání těch částí archivu, týkajících se výpovědí v Českém jazyce (více než 550 výpovědí, čítajících téměř 1000 hodin videozáznamu)[1].

5.1.2 Trénování LVCSR systému

Hlavní náplní projektu bylo vytvoření odpovídajícího LVCSR systému. Pro vytvoření jeho akustických a jazykových modelů, byla vybrána a zpracována část tohoto archivu, konkrétně 100 hodin náhodně vybraných nahrávek. Poté byla celá část archivu s českými výpověďmi tímto systémem automaticky rozpoznána a jako výstup systému byly vygenerovány slovní a fonémové mřížky. Kvůli redukování objemu dat bylo použito speciálního algoritmu, který zajišťoval zpracování jen těch částí nahrávek, které obsahovaly promluvu alespoň jednoho řečníka. Průměrná délka jednoho videozáznamu byla 1,9 hodiny a jednotlivé záznamy byly rozděleny na segmenty po půlhodině. Pro zajímavost uveďme, že tvůrci se museli při návrhu systému potýkat s dalším obtížným problémem: nahrávky s výpověďmi byly stereofonní, kdy jeden kanál obsahoval promluvy pamětníka a druhý promluvy reportéra. Tyto se však ve výsledku vzájemně překrývaly a bylo tak nutné signály v obou kanálech od sebe odlišit. Další podrobnosti už přesahují rámec této práce a čtenář je nalezne v [1] a [2].

[9]Z angl.: Multilingual Acces to Large Spoken Archives = Vícejazyčný přístup k rozsáhlým řečovým archivům.

[10]VHF angl.: Visual History Foundation, v překladu: Nadace visuální historie přeživších pamětníků Holokaustu.

5.1.3 Slovní a fonémové mřížky

Vlastní rozpoznávání LVCSR systému bylo navrženo ve dvou průchodech. V prvním průchodu byly akustické modely postupně adaptovány na každého z 550 řečníků za současného použití tri-gramového jazykového modelu. V druhém průchodu pak byly vygenerovány mřížky. Slovní byly vypočteny retrospektivně tak, že ke slovu, rozpoznanému normalizovaným věrohodnostním akustickým modelem, bylo přidáno n nejlepších hypotéz (alternativ) trigramového jazykového modelu. Výsledná míra důvěry, přiřazená každé hypotéze (přechodu) v mřížce, byla vyjádřená aposteriorní pravděpodobností, vypočtenou *forward-backward* [6] algoritmem. Fonémové mřížky byly vygenerovány stejným způsobem jen na základě informace z akustického modelu adaptovaného na každého řečníka v prvním průchodu rozpoznávání [1] [2].

Soubory s mřížkami byly uloženy ve formě textových souborů SLF (angl.: Standard Lattice Format), což je standardní formát mřížky, užívaný nástroji HTK[11] [19]. Ten obsahuje hlavičku se základními informacemi (verze, název promluvy, počet uzlů a hran, apod.) a definice jednotlivých hran a vrcholů mřížky. Níže vidíme malou ukázku:

```
VERSION=1.1
UTTERANCE=vetsi/25819_002_(1285.090)_F.htk
base=10
dir=f
tscale=0.01
hmms=
lmname=
vocab=
start=0
end=7
NODES=8 LINKS=12
I=0 t=0
I=1 t=1
I=2 t=187
I=3 t=309
I=4 t=370
I=5 t=371
```

[11] HTK (z angl.: Hidden Markov Model Toolkit) je nástroj pro tvorbu a operace se Skrytými Markovskými Modely, užívaný v úlohách rozpoznávání řeči. Byl vyvinut během 90-tých let v Laboratořích Strojové Inteligence na univerzitě v Cambridge.

```
I=6 t=406
I=7 t=406
J=0 S=0 E=1   W=<s>  s=<s>  a=0.0000   p=1
J=1 S=1 E=2   W=KDYŽ         a=-71.9112 p=0.0161303
J=2 S=1 E=2   W=TO           a=-67.0614 p=0.0144275
J=3 S=1 E=3   W=ŠESTNÁCTÉHO   a=-191.0700 p=1
J=4 S=2 E=3   W=ŠESTNÁCTÉHO   a=-123.2100 p=1
J=5 S=3 E=4   W=LISTOPADU     a=-109.3069 p=0.956816
J=6 S=3 E=4   W=LISTOPAD     a=-94.8409 p=0.0431839
J=7 S=3 E=5   W=LISTOPADU     a=-112.2513 p=0.956816
J=8 S=3 E=5   W=LISTOPAD     a=-97.8583 p=0.0431839
J=9 S=4 E=6   W=</s> s=</s> a=-15.5895 p=0
J=10 S=5 E=6  W=</s> s=</s> a=-13.8618 p=0
J=11 S=6 E=7  W=!NULL s=!NULL a=0.0000   p=1
```

Povšimněme si zejména hodnot `a` a `p` v definici hran (řádky začínající písmenem `J`). Ty označují akustickou pravděpodobnost $P(O|W)$ a výslednou věrohodnost $P(W,O)$. Tato věrohodnost byla použita pro převod na `-log` váhy hran mřížek ve formátu WFST. Další důležitou informaci představuje časování vrcholů mřížky (hodnoty `t` u řádek, začínajících písmenem `I`). Podrobnosti k významu zmíněných veličin čtenář nalezne v odstavci 1.2.1.

5.2 Tvorba indexu, vyhledávání

Nástroje pro indexaci a vyhledávání s váženými konečnými transducery (dále jen STDtools) jsou k dispozici ke stažení na `www.busim.ee.boun.edu.tr/õgan/research.html` a jejím autorem je MSc. Dogan Can z Univerzity v Bogaziçi. Je implementována v jazyce C++ nad knihovnou OpenFST (`www.openfst.org`), která tvoří framework pro práci s WFST. Součástí nástrojů jsou skripty pro předzpracování, indexaci a vyhledávání tak, jak byly popsány v kapitole 3.

Kromě nich, bylo k implementaci celého procesu indexace MALA-CHovských mřížek zapotřebí napsat skripty pro převod z textového HTK formátu do binárního formátu WFST, kompatibilních s knihovnou OpenFST. Ty byly napsány v jazyce Python s využitím volně dostupné knihovny PyOpenFST (`http://code.google.com/p/pyopenfst/`). a knihovny PyFST, vyvíjené na Katedře Kybernetiky ZČU v Plzni. Při převodu bylo mimo jiné důležité vzít na vědomí fakt, že HTK mřížky byly reprezentací grafů nad pravděpodobnostním polookruhem, kdežto standardní formát transducerů knihovny OpenFST počítá s polookruhem tropickým.

Pro vyhledávání nad slovními mřížkami byl sestaven dotaz z 20 jednotlivých klíčových slov, o kterých bylo známo, že se vyskytují jak v samotných mřížkách, tak i ve slovníku LVCSR systému. Dotaz byl jednoduše sestrojen jako vážený konečný transducer s jedním přechodem mezi počátečním a koncovým stavem, jehož vstupní i výstupní symbol tvořilo hledané slovo (viz odstavec 3.4). V případě fonémových mřížek byl sestaven dotaz ze 108 slov mimo slovník rozpoznávače (OOV slov). Převod dotazu na transducer byl o něco složitější, protože bylo třeba řešit problémy jednak s možnou různou výslovností stejného slova a také uvažovat vliv nepřesného rozpoznávání konkrétních fonémů z akustického signálu. Podrobněji je o tom pojednáno v sekci 6.2.3. Výstupní soubory s výsledky vyhledávání byly vygenerovány ve formátu MLF (Master Label File), použitelných pro nástroje HTK. Obsahovaly vždy ID nahrávky extrahované z názvu mřížky, následované řádky s nalezenými položkami s počátečním a koncovým časem výskytu $[s \cdot 10^{-7}]$, slovem a s jeho skóre. Opět malá ukázka:

```
#!MLF!#
"*/28105_001.rec"
12647980566 12653567868 židi 0.332351 0.332351 1
12647980566 12654166507 židi 0.332351 0.332351 1
6738651088 6742741791 židi 0.829029 0.829029 1
.
"*/12608_003.rec"
10665570997 10670858979 zdeněk 1 1 1
10955729433 10961316734 zdeněk 1 1 1
11177312244 11182500453 zdeněk 1 1 1
11357040362 11364124263 zdeněk 1 1 1
11357040362 11365720634 zdeněk 1 1 1
11603966417 11607857573 praze 1 1 1
12038345170 12043333832 praze 1 1 1
12482791179 12487380748 zdeněk 1 1 1
17203688571 17206781541 pavel 0.939413 0.939413 1
2282219727 2286809297 praze 1 1 1
698248253 704035102 praze 1 1 1
7060991678 7066080113 praze 1 1 1
7060991678 7069472403 praze 1 1 1
7719095328 7726079455 židi 0.0235867 0.0235867 1
7751721179 7755712108 praze 1 1 1
7880647913 7884339523 praze 1 1 1
```

Nad těmito soubory, pak byly prováděny experimenty pro hledání optimálních parametrů a vyhodnocení systému. Na závěr uveďme pro

zajímavost ještě něco málo o stroji, na kterém byly výpočty prováděny. Byl jím databázový server DELL PowerEdge R310 se čtyřjádrovým procesorem Intel Xeon X3450 s frekvencí 2.67GHz, fyzická paměťí RAM 32 GB a s pevným diskem o kapacitě 8 TB. Nebylo výjimkou, že výpočty některých indexů, zejména v případě fonémových mřížek, si vyžádaly i více než 25 GB paměti a při plném výkonu procesoru trvaly mnohdy i několik dní.

6 Experimenty

6.1 Teorie k vyhodnocení úlohy STD

Mějme soubor $Q_k,\ k = 1 \dots N_Q$ výrazů v hledaném dotazu a T_u, což je soubor všech výrazů v promluvě u. Nechť

$R(Q_k)$	je počet všech výrazů z T_u skutečně relevantních k dotazu Q_k
$A(Q_k)$	je počet všech výrazů vyhledaných pro dotaz Q_k
$C(Q_k)$	je počet výrazů správně vyhledaných pro dotaz Q_k, tj. $C(Q_k) = A(Q_k) \cap R(Q_k)$.

Během činnosti STD systému nastávají čtyři základní jevy:

Výrazy	Vyhledané, $A(Q_k)$	Nevyhledané, $T_u - A(Q_k)$
Relevantní, $R(Q_k)$	Správně získané $C(Q_k)$	Nesprávně ztracené (FM)
Irelevantní, $T_u - R(Q_k)$	Nesprávně získané (Falešný poplach - FA)	Správně zamítnuté

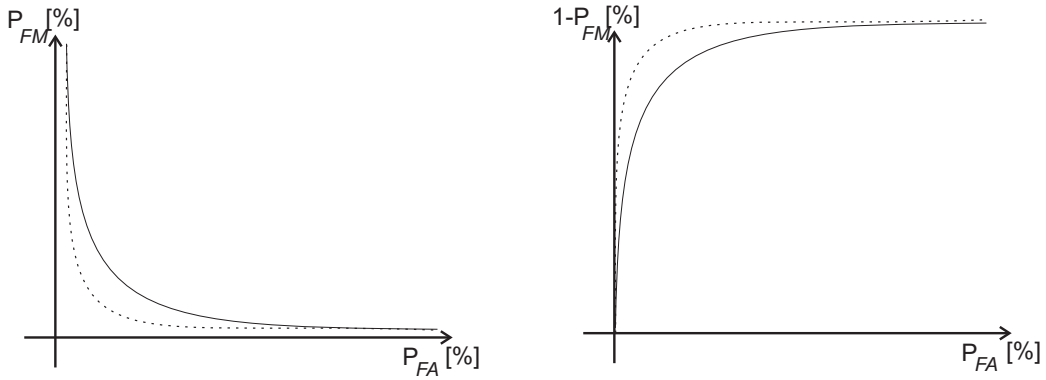
Činnost STD systému pak lze ohodnotit např. pomocí pravděpodobnosti falešného poplachu P_{FA} a nesprávné ztráty P_{FM} [4]:

$$P_{FM} = 1 - \frac{C(Q_k)}{R(Q_k)}, \quad P_{FA} = \frac{A(Q_k) - C(Q_k)}{T_u - C(Q_k)}$$

6.1.1 Křivka ROC

Nahlížíme-li na úlohu STD jako na statistický rozhodovací problém, pak pravděpodobnost rozpoznaného slova $P(W|O)$ reprezentuje míru věrohodnosti pro hypotézu výskytu (vyhledání) slova v daném místě promluvy. Zavedeme-li θ jako nějaký práh věrohodnosti, tj. hypotéza je přijata, pokud $P(W|O) > \theta$, pak P_{FM} můžeme nazvat pravděpodobnost nesprávného odmítnutí této hypotézy (statistická chyba I. druhu) a P_{FA} pravděpodobnost jejího nesprávného přijetí (chyba II. druhu). S klesajícím prahem θ klesá počet chyb N_{FM}, ale roste počet falešných poplachů N_{FA} a naopak. Práh θ tedy tvoří jakýsi parametr závislosti $N_{FM}(\theta)$ na $N_{FA}(\theta)$. Tuto závislost je možné vyčíslit prostřednictvím pravděpodobností obou chyb pro všechny promluvy v archivu a vyjádřit pomocí tzv. *křivky ROC* (angl.: Receiver Operating Characteristic Curve). Její příklad vidíme na obrázku 21 vlevo.

Optimální bod křivky leží v počátku souřadného systému a čím více se tomuto bodu křivka blíží (její ohyb je ostřejší), tím jsou pravděpodobnosti obou chyb menší. Stejnou myšlenku můžeme vyjádřit také tak, že namísto pravděpodobnosti ztráty, uvažujeme míru přesnosti, tj. $1-P_{FM} = \frac{C(Q_k)}{R(Q_k)}$ (obr. 21 vpravo), což je jiný případ křivky ROC. Míry se zde zpravidla uvádějí v procentech [6].



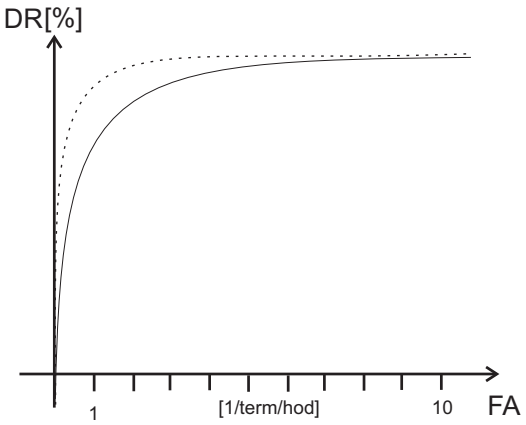
Obrázek 21: Ukázkový hypotetický příklad ROC křivek pro dva různé STD systémy. Čárkovaný průběh je ostřejší, více se blíží optimálnímu bodu v počátku souřadného systému a je tedy lepší, než-li průběh s plnou čárou.

6.1.2 Detekční schopnost

ROC charakteristiku lze vyjádřit i s pomocí jiných, avšak velmi podobných veličin. Její vypovídací hodnota se tím může ještě zvýšit. Pravděpodobnost P_{FA} můžeme nahradit četností N_{FA} a posuzovat vůči různým měřítkům. Z nich nejuniverzálnější je patrně míra počtu falešných poplachů na jedno relevantní referenční slovo a jednu hodinu testovaného signálu [8]:

$$FA[1/term/hod] = \frac{N_{FA}}{R_{u_t}(Q_k) \times l(u_t)},$$

kde $l(u_t)$ je délka testované promluvy u_tv hodinách. Míra přesnosti $(1-P_{FM})$ se také nazývá jako *detekční schopnost* (angl.: Detection Rate - DR) a je i stejně definována, tj. $DR = \frac{C(Q_k)}{R(Q_k)}$. Křivka ROC se zde vyčísluje v počtech $FA/term/hod$ a v procentech pro DR (obr. 22).



Obrázek 22: Příklad ROC křivek, zobrazujících závislost detekční schopnosti DR na četnosti N_{FA}. Čárkovaně opět lepší průběh.

Chceme-li vyjádřit hodnocení STD systému pouze jediným číslem, můžeme použít tzv. *průměrnou detekční schopnost FOM* (angl.: Figure Of Merit), definovanou jako průměrná detekční schopnost systému pro nula až deset falešných poplachů na jedno klíčové slovo a jednu hodinu vstupního signálu [8]:

$$FOM = \int_{FA=0}^{10} DR(FA)dFA.$$

Na tuto hodnotu má největší vliv počáteční (rostoucí) část ROC charakteristiky a měli bychom se snažit dosáhnout co největší hodnoty FOM.

6.1.3 Míra rovnosti EER

Jak již bylo zmíněno v odstavci 6.1.1, chyby FM a FA jsou na sobě závislé prostřednictvím parametru θ. **Míra rovnosti chyb** $N_{FM}(\theta)$ a $N_{FA}(\theta)$ (EER - angl.: Equal Error Rate) při parametru θ, je další možností, jak hodnotit kvalitu STD systému, a je definována jako [6]:

$$EER = FM(\theta) = FA(\theta)$$

Míra rovnosti staví obě chyby na stejnou váhu a říká, jaká je jejich nejmenší možná hodnota a s jakým parametrem prahu θ. Pro určení EER je tedy třeba najít takovou hodnotu prahu θ, pro kterou jsou si obě chyby rovny. Při návrhu systému bychom se měli snažit dosáhnout co nejmenší hodnoty EER.

6.2 Hledání optimálních parametrů

Při ladění systému pro úlohu STD je potřeba mít k dispozici informaci o tom, jak má vypadat správný výsledek. Vlastní experimenty proto probíhaly nad podmnožinou MALACHovských mřížek, které byly výsledkem rozpoznávání těch částí zvukových stop, ke kterým byla známa i přesná referenční data z ručně vytvořených anotací (tj. přepisů s uvedenými místy výskytu výrazů ve zvukové stopě). Jsou to stejné části archivu, které byly použity i pro trénování LVCSR systému (viz kapitola 5.1.2).

6.2.1 Konfigurovatelnost systému založeného na WFST

Před vyhodnocením přínosu systému s WFST bylo nutné nejprve nalézt optimální parametry indexu s ohledem na hodnoty FOM, EER a křivku ROC. Dobrým vodítkem pro ladění zde byla možnost vygenerovat tyto hodnoty stávajícím systémem, popsaným v kapitole 4. Indexace nástroji STDtools je konfigurovatelná několika parametry:

- Při přezdpracování je to parametr práh hodnoty prořezávání, který je velmi důležitý a o jeho volbě se ještě zmíním. Další možnosti parametrizace spočívají v:
  - Stanovení minimálního časového překrytí při shlukování překrývajících se hran se stejnými vstupními symboly.
  - Dále je možno definovat, jak se bude zacházet s po sobě jdoucími hranami se symbolem pro ticho. Pro náš případ toto není třeba řešit, protože jak již bylo zmíněno v odstavci 5.1.2, ASR systém rozpoznával jen ty části nahrávek, které obsahovaly nějaký řečový signál.
  - Užitečnou volbou je i možnost determinizace předzpracované mřížky, což pomáhá redukovat množství zpracovávaných dat. Tuto volbu jsem použil pro redukci překrývajících se hran se stejným symbolem, kterých se v předzpracovaných mřížkách vyskytovalo značné množství.
- Při indexaci lze odstranit hrany se zadanými faktory nebo faktory, jejichž délka řetězce převyšuje stanovenou hodnotu. Ukázalo se, že omezení na délku řetězce má významný vliv na velikost budoucího indexu. Rovněž o tomto bude ještě v nadcházejících odstavcích řeč.
- U vyhledávání je možno nastavit parametr, omezující vyhledání jen na n nejlepších hypotéz namísto všech dostupných. Tato možnost zhoršuje

výsledky vyhledávání a odebírá tak jednu z hlavních výhod indexace mřížek, totiž možnost archivovat více hypotéz ASR systému. Proto nebyla v mojí práci použita.

Hledání optimálních parametrů probíhalo následovně. Označme I jako maximální přípustný počet vstupních hran pro každý stav mřížky, tj.: $I\{p[q]\}\ \forall q \in Q_i, i = 1, \ldots, n$ kde n je počet všech mřížek. Mřížka s hodnotou $I = 1$ tedy odpovídá pouze jedné nejlepší hypotéze ASR systému. Soubory slovních mřížek byly k dispozici v deseti variantách parametru I od 1 do 10. Fonémové mřížky se vyskytovaly ve dvou základních variantách se 0-gramovým a 5-gramovým jazykovým modelem, které se pak ještě dělily na další podsoubory podle parametru I s hodnotami od 1 do 5. Pro oba typy mřížek tedy bylo k dispozici celkem deset souborů a pro každý z nich byly sestaveny indexy s prořezáním od 1 do 5 a s neomezenou délkou faktorů. Výsledkem tedy bylo více než padesát různých indexů nad oběma typy mřížek, které byly s pomocí validačního skriptu porovnány s referenčními daty. Z výsledných tabulek FOM a EER pak byl jako optimální index zvolen ten, který poskytoval maximální hodnotu FOM při minimální míře rovnosti chyb EER. S těmito optimálními indexy bylo posléze prováděno další vyhodnocení.

6.2.2 Optimální index slovních mřížek

Výsledné hodnoty FOM a EER pro slovní mřížky ukazují tabulky 2 a 3. Optimálních hodnot bylo dosaženo pro $I = 4$ a $p = 5$, při nichž bylo vyčísleno FOM = 91,485 a EER = 21,39. Z tabulek 2 a 3 je také patrné, že nerozsáhlejší index v počtu uložených ASR hypotéz nemusí nutně poskytovat nejlepší výsledky. Tato vlastnost plyne především ze vzájemného vztahu chyb FA a FM. Pro více dat v indexu totiž sice dosáhneme menších ztrát, ale za cenu nežádoucího vzrůstu počtu falešných poplachů.

6.2.3 Tvorba dotazů do fonémového indexu

Zatímco dotazy nad slovními mřížkami vznikaly jednoduše vytvořením transducerů se slovy na vstupně-výstupních symbolech, v případě fonémového indexu byla situace o něco komplikovanější. Experimenty dávaly zprvu velmi špatné výsledky (FOM kolem nuly a EER nad 90%), vlivem velmi nízkého skóre nalezených hypotéz. Důvod byl ten, že celková skóre hledaných slov byla vypočtena prostým logaritmickým součtem dílčích fonémů a byla velice nízká. Vysvětlení plyne z vlastnosti logaritmického polookruhu, kde se váhy podel cesty sčítají - což v případě polookruhu reálných čísel odpovídá

$I\{n[q]\}$	**Hodnota prořezání** p				
	1	2	3	4	5
1	81.878	81.878	81.878	81.878	81.878
2	85.343	86.691	87.225	87.755	88.289
3	85.619	87.491	88.033	89.337	90.134
4	85.619	87.748	89.355	90.404	**91.485**
5	85.094	87.224	89.351	90.653	91.202
6	85.357	86.965	89.371	90.434	91.213
7	85.349	86.968	89.612	90.433	91.487
8	85.092	86.981	88.829	90.683	91.470
9	85.098	86.986	89.092	90.665	91.469
10	85.364	86.988	89.101	90.415	91.243

Tabulka 2: Závislost hodnoty FOM na parametrech I a p u slovního indexu. Nejlepší hodnota je zvýrazněna.

$I\{n[q]\}$	**Hodnota prořezání** p				
	1	2	3	4	5
1	17.647	17.647	17.647	17.647	17.647
2	21.925	21.925	21.925	21.925	21.925
3	21.123	20.588	20.321	20.321	20.856
4	21.123	21.123	20.856	20.856	**21.390**
5	21.123	21.390	20.856	20.856	21.390
6	21.123	21.390	21.390	21.123	21.390
7	21.390	21.390	21.123	20.856	21.658
8	21.123	21.658	21.390	20.856	21.390
9	20.856	21.390	20.856	20.588	21.123
10	21.390	21.390	21.123	20.856	21.390

Tabulka 3: Závislost hodnoty EER na parametrech I a p u slovního indexu. Nejlepší hodnota je zvýrazněna.

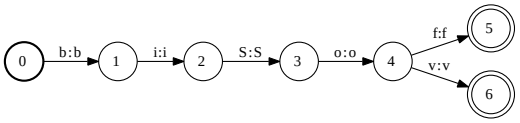
jejich součinu. A jak známo výsledkem součinu čísel menších než nula jsou postupně menší a menší čísla. Celkové skóre se ještě dále snižovalo v závislosti na rostoucím počtu fonémů, ze kterých se slova skládají. Potom byl-li dán nějaký práh věrohodnosti θ, pak tato delší slova měla oproti kratším menší šanci dostat se do výsledků vyhledávání. Z tohoto důvodu byla aplikována normalizace v podobě podílu výsledného skóre slova počtem fonémů v něm obsažených:

$$\mathcal{W}_{W'}(w_1, \ldots, w_n) = \frac{w_1 \cdot \ldots \cdot w_n}{n}.$$

kde w_i je skóre i-tého fonému a n je počet fónémů v nalezeném slově W'. Rovnice odpovídá geometrickému průměru fonémů z kapitoly 4.1.2 protože

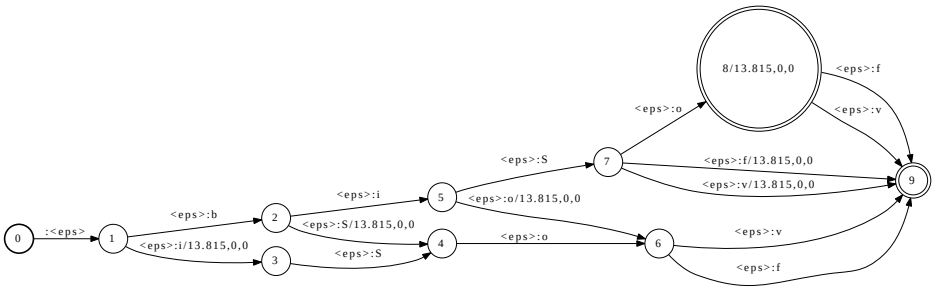
$$\log \sqrt[n]{w_1 \cdots w_n} = \frac{1}{n} \cdot \log(w_1 \cdot \ldots \cdot w_n).$$

Dotazy do fonémového indexu byly výsledkem fonémové transkripce hledaného slova, přičemž bylo nutné počítat s více možnostmi jeho vyslovení (viz ukázkový příklad na obr. 23).



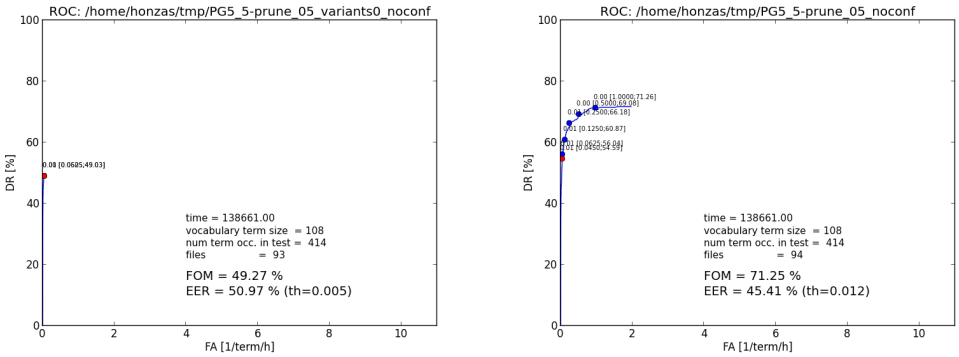
Obrázek 23: Fonetický dotaz ve formě WFST pro různé výslovnosti slova "Bischof".

Takovéto dotazy však nacházely málo výsledků a detekční schopnost systému byla stejná nebo i menší, než chyba EER. Navíc byla i konstantní pro různé hodnoty prahu θ, viz levý graf na obrázku 25. To proto, že dotazům v prohledávaném indexu odpovídalo jen velmi malý počet úspěšných cest. Dotazy totiž nereflektovaly skutečnost, že některé fonémy se v rozpoznaných slovech mřížky nemusejí vůbec nacházet. Důvodem je poměrně častý jev, kdy při vyslovování slov se některé hlásky vypouštějí. Po vzoru stávajícího systému indexace byly proto dotazy zformulovány tak, že akceptovaly možnost vypuštění jednoho libovolného fónému. Dělo se tak prostřednictvím přidání paralelních cest s nízkou váhou pro všechny kombinace vypuštěného fonému z hledaného slova. Váhu označme γ a její hodnotu jako $\gamma = -\log(1 \cdot 10^{-6}) = 13,815$. Příklad této operace pro transducer z obrázku 23 vidíme na obrázku 24. Z grafů na obrázku 25 a z tabulky 4 je jasně vidět výrazné zlepšení detekční schopnosti vlivem nalezení více úspěšných cest v indexu. Z posledního řádku lze vyčíst, že navýšení detekční schopnosti má s přibývajícím počtem falešných poplachů rostoucí charakter. Při



Obrázek 24: Dotaz z obrázku 23, vzniklý přidáním paralelních hran se všemi kombinacemi možných chybějících fonémů, po determinizaci a minimalizaci.

vypuštění dvou libovolných fonémů už nedochází ke zlepšení detekční schopnosti, jak ukazuje předposlední řádek, a další navyšování počtu vypuštěných fonémů je tedy již zbytečné.

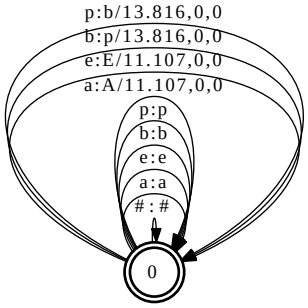


Obrázek 25: Srovnání křivek ROC, vlevo pro dotazy bez vypuštění fonému, vpravo pro dotazy, uvažující vypuštění jednoho libovolného fonému.

FA/term/hod	1/16	1/8	1/4	1/2	1	2
$DR_{V0}(FA)$	46,86	46,86	46,86	46,86	46,86	46,86
$DR_{V1}(FA)$	56,04	60,87	66,18	69,08	71,26	72,21
$DR_{V2}(FA)$	56,04	60,87	66,18	69,08	71,26	72,46
$DR_{V1} - DR_{V0}$	9,18	14,01	19,32	22,22	24,4	25,35

Tabulka 4: Závislost detekční schopnosti DR na parametru Vx a počtu falešných poplachů, x označuje počet libovolně vypuštěných fonémů.

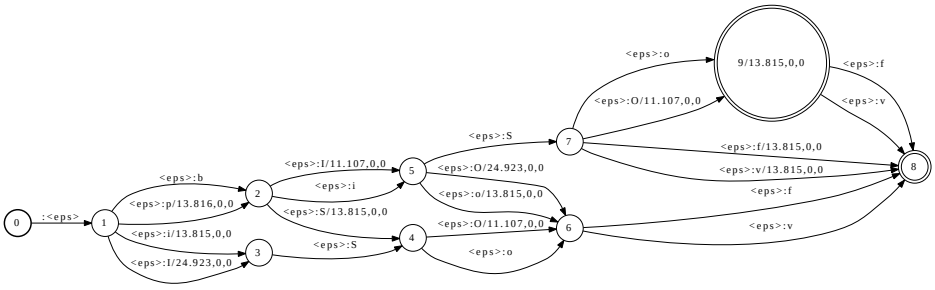
Další vliv na schopnost detekce má také fakt, že akustické rozpoznávání slov bez dodatečné informace jazykového modelu není tak přesné a při rozpoznávání může způsobit záměnu některých hlásek. Typickým příkladem je záměna krátkých samohlásek za dlouhé nebo záměna neznělých souhlásek za znělé (a naopak). Vyzkoušeli jsme tedy dotaz z předchozího kroku ještě rozšířit přidáním dalších hran, reprezentujícím tyto možné záměny. Dělo se tak prostřednictvím kompozice dotazu s konfuzní tabulkou ve formě WFST, která obsahovala všechny uvažované záměny, ke kterým může v praxi docházet. Oba typy záměn byly ohodnoceny každý jinou váhou: $\alpha = -\log(15 \cdot 10^{-6}) = 11,107$ pro záměnu krátké samohlásky za dlouhou a $\beta = -\log(1 \cdot 10^{-6}) = 13,816$ pro záměnu neznělé souhlásky za znělou (a naopak). Příklady konfuzní tabulky a výsledek operace vidíme na obrázcích 26 a 27.



Obrázek 26: Ukázková (neúplná) fonémová konfuzní tabulka, přidávající ke standardním fonémovým hranám možnost záměny některých fonémů, např. "a" (= krátké *a*) za "A" (= dlouhé *á*) nebo *p* za *b* a naopak (spodoba znělosti).

Jak ukazuje tabulka 5, rozšíření dotazu konfuzní tabulkou také ještě přispělo k nárůstu detekční schopnosti vlivem dalších nalezených úspěšných cest indexem, ale již ne tak výrazně jako krok předchozí.

Na závěr poznamenejme, že hodnoty parametrů α, β, γ rozšiřujících cest byly stanoveny odhadem na základě několika pokusů s experimentálními daty. Prozatím poskytly vcelku dobré výsledky a výzkum jejich vlivu na výsledná skóre nalezených hypotéz a tedy i hodnoty DR a FA bude předmětem další práce do budoucna.



Obrázek 27: Definitivní podoba fonetického dotazu, vzniklá kompozicí konfuzní tabulky s dotazem na obr. 24.

FA/term/hod	1/16	1/8	1/4	1/2	1	2
$\Delta DR_{V0}(FA)$	2,17	2,17	2,17	2,17	2,17	2,17
$\Delta DR_{V1}(FA)$	1,45	1,45	0,73	0,49	1,69	1,94

Tabulka 5: Nárůst hodnot z tabulky 4 při použití konfuzní tabulky.

6.2.4 Optimální index fonémových mřížek

Po vyřešení problémů kolem tvorby fonémových dotazů mohlo být přistoupeno k vlastnímu nalezení optimálních parametrů I a p. Tabulky 6 a 7 ukazují opět závislost hodnot FOM a EER na parametrech I a p. Povšimněme si, že ve srovnání se slovními mřížkami, nezahrnují všechny hodnoty zkoumaných fonémových N-gramů, ale jen hodnoty pro 5-gramové fonémy a navíc nový vzorek fonémů 6-gramových. To proto, že již během prvních experimentů se ukázalo, že 0-gramové fonémy nemohou poskytnout požadované výsledky ani pro nejméně prořezané mřížky. Od dalších experimentů s touto sadou tedy bylo upuštěno a namísto toho byla přidána ona pokusná 6-gramová sada (pouze s hodnotou $I = 4$, poněvažd ostatní varianty by byly výpočetně velmi náročné a trvaly by neúnosně dlouho). V případě fonémového indexu tedy bylo optimálních hodnot FOM = 74.757 a EER = 44.928 dosaženo pro $I = 5$ a $p = 5$. Pokusná sada 6-gramů již lepší výsledky nepřinášela.

K otázce hledání optimálních parametrů indexu je třeba zmínit fakt, že fonémové mřížky obsahovaly řádově mnohem více stavů i hran a kladly tak nepoměrně větší nároky na výpočetní čas než mřížky slovní. Indexy pro některé hodnoty I a p nešly standardním způsobem na DB serveru spočítat (alokovaly si kolem 28 GB virtuální paměti) a proces ustrnul ve

N-gram	$I\{n[q]\}$	Hodnota prořezání p				
		1	2	3	4	5
5	1	32.090	32.090	32.090	32.090	32.090
5	2	40.532	46.048	49.646	52.760	55.636
5	3	43.364	50.778	56.991	60.742	64.514
5	4	46.921	54.053	60.968	67.695	71.401
5	5	45.955	56.220	64.271	71.870	**74.757**
6	4	46.921	54.053	49.340	67.695	71.400

Tabulka 6: Závislost hodnoty FOM na parametrech I, p a N-gramu pro případ fonémového indexu.

N-gram	$I\{n[q]\}$	Hodnota prořezání p				
		1	2	3	4	5
5	1	69.324	69.324	69.324	69.324	69.324
5	2	62.560	57.971	55.556	53.382	50.725
5	3	62.077	56.039	52.174	50.725	49.275
5	4	59.903	55.797	51.208	47.826	46.135
5	5	59.903	53.382	48.792	45.169	**44.928**
6	4	59.903	55.797	57.246	47.826	46.135

Tabulka 7: Závislost hodnoty EER na parametrech I, p a N-gramu pro případ fonémového indexu.

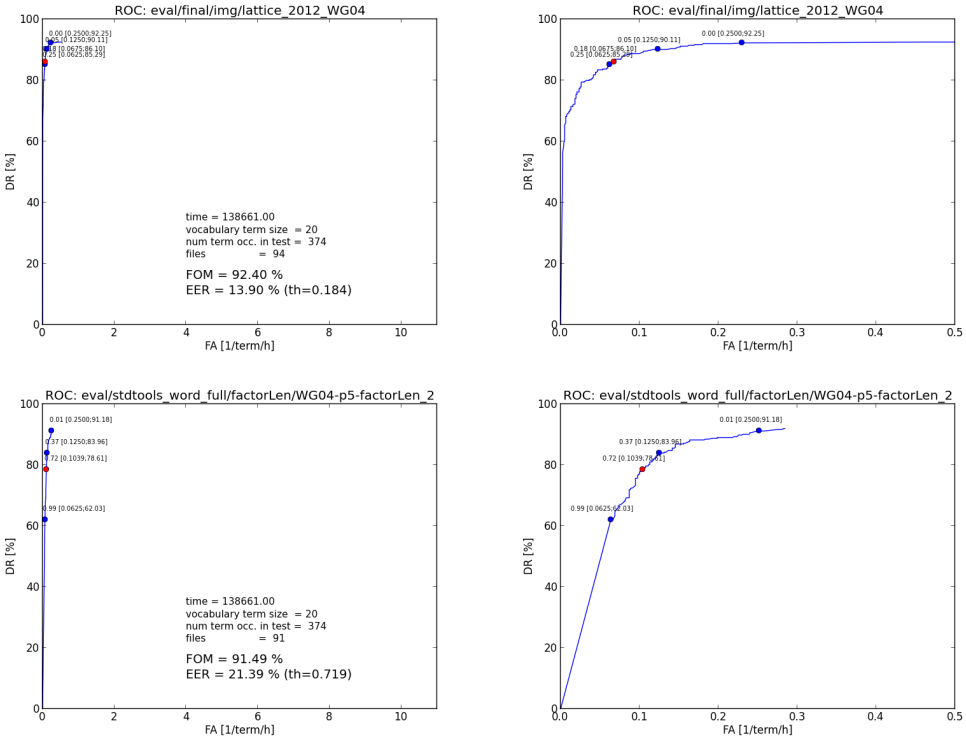
fázi stránkování paměti na disk. Problém byl vyřešen tak, že byl stanoven omezující práh maximální délky faktoru (tj. maximálnímu počtu fonémů ve slově) na 15, a soubory indexovaných mřížek byly rozděleny do několika podsouborů, nad kterými byly vygenerovány samostatné indexy. Všem těmto indexům byly pak kladeny dotazy zvlášť a výsledky vyhledávání se pak při vyhodnocování sečetly.

6.3 Vyhodnocení indexu slovních mřížek

6.3.1 Přesnost

Srovnávací grafy ROC charakteristik stávajícího a zkoumaného systému indexace slovních mřížek vidíme na obrázku 28. Z nich si můžeme udělat následující závěry:

- Co do hodnoty FOM je systém, založený na WFST a indexaci celých



Obrázek 28: Srovnání křivek ROC pro systém indexace hran mřížky (nahoře) a systém s WFST (dole). Levý graf zobrazuje rozsah osy x pro hodnoty[0;10], zatímco pravý graf je výřez pro průběh charakteristiky do 0,5 FA/term/hod. Body v křivce představují pracovní body při daném parametru θ (v grafu ozn. "th"), červeně označený je pracovní bod, při kterém bylo dosaženo rovnosti chyb EER.

mřížek srovnatelný s dosavadním systémem, založeným na indexaci jen vybraných mřížkových hran.

- Vyšší hodnotu EER u systému s WFST lze vysvětlit v zásadě třemi způsoby. Jednak je to tím, že systém s WFST narozdíl od dosavadního indexuje celou mřížku, tedy všechny podřetězce zvolené délky. To znamená, že se oproti stávajícímu systému v indexu vyskytne mnohem více hran s nižším skóre, tedy více irelevantních dat. Navíc během normalizace weight-pushingem se váhy všech hran mřížky oproti původním hodnotám snižují. Výše zmíněné aspekty mají následně vliv na vyšší hodnotu četnosti falešných poplachů (cca 1 falešný poplach na 10 výrazů a 1 hodinu záznamu) a vyšší hodnotu prahu θ při

rovnosti chyb FA a FM. Naopak protože index dosavadního systému díky výběru jen "nejlepších"mřížkových hran obsahuje poměrně více relevantních dat, je hodnota četnosti chyb falešných poplachů a ztrát nízká i při velmi malé hodnotě prahu θ (cca 7 falešných poplachů na 100 výrazů a 1 hodinu záznamu). Jistý vliv může mít zřejmě také fakt, že tvůrci nástrojů *stdtools* počítají s tím, že váhy mřížek jsou tvořeny součinem akustické pravděpodobnosti a jazykového modelu (sdruženou pravděpodobností $P(W, O)$ viz sekce 1.2.1), kdežto váhy mnou zkoumaných mřížek představují již finální aposteriorní pravděpodobnost $P(W|O)$, normalizovanou tak, aby součet paralelních hypotéz byl v každém časovém okamžiku roven jedné. Otázka vlivu tohoto faktu zatím zůstává otevřená pro další experimenty.

6.3.2 Nároky na úložný prostor, rychlost vyhledávání

Vedle ukazatelů přesnosti indexace a vyhledávání nás také zajímají jeho nároky na úložný prostor a rychlost vyhledávání. V průběhu hledání optimálních parametrů indexu se ukázalo, že na jeho výslednou velikost má významný vliv omezení maximální délky indexovaných faktorů. Proto jsem si sestavil 10 indexů (pro již výše zmíněnými parametry $I = 4$ a $p = 5$) s hodnotami omezení délky faktoru od jedné do deseti. Z těchto jsem nalezl takovou minimální velikost indexu, pro kterou by i při tomto omezení odpovídal přesností stávajícímu systému. Takový index jsem získal pro maximální délku faktoru rovno 2. V této souvislosti je třeba poznamenat, že oba systémy byly porovnávány na základě dotazů na jednotlivá klíčová slova, tj. faktory délky 1. To hlavně z toho důvodu, že stávající sytém kladl důraz především na vyhledávání jednotlivých klíčových slov. Proto má smysl zde srovnávat právě tento minimální a přitom přesností srovnatelný WFST index.

Pro další experimenty jsem tedy vzal výše zmíněný minimální index a k němu pro představu ještě index bez omezení délky faktoru. U nich jsem pak zkoumal jejich nároky na úložný prostor a vyhledávací čas. Protože jsem pro experimenty měl k dispozici jen zlomek skutečného množství dat z celého MALACHovského archivu (měřeno počtem půlhodinových pásek je to cca 47 z celkového počtu 1000 hodin), zkusil jsem stanovit závislost velikosti indexu a vyhledávacího času na objemu indexovaných dat. K tomu účelu jsem si sestavil skripty, které rozdělily vstupní sedmačtyřicetihodinová data na části (100, 75, 50 a 25 %) a nad nimi vygeneroval samostatné indexy. Pro ně jsem pak zjistil jejich velikosti a časy vyhledávání. Ukázalo se, že obě zkoumané veličiny mají k objemu dat lineární závislost, čehož jsem mohl s využít k

estimaci jejich hodnot pro celý archiv s 1000 hodinami. Na tomto místě je třeba poznamenat, že se jedná pouze o odhad stanovený na základě pouhého počtu indexovaných pásek. Skutečný indexovaný čas a objem dat se závisí především na tom, kolik se v pásce skutečně vyskytuje řečových promluv a tedy jak objemné jsou odpovídající mřížky. Výsledky pro minimální i neomezený index ukazují grafy na obrázku 29. Ukazuje se, že v případě neomezeného indexu je nárůst úložného prostoru zhruba trojnásobný, zatímco na vyhledávací čas to nemá praktický žádný vliv. To je velice zajímavé zjištění. Vyhledávací čas závisí mnohem více na počtu slov v hledaném dotazu, což nakonec uvádějí i autoři v [3].

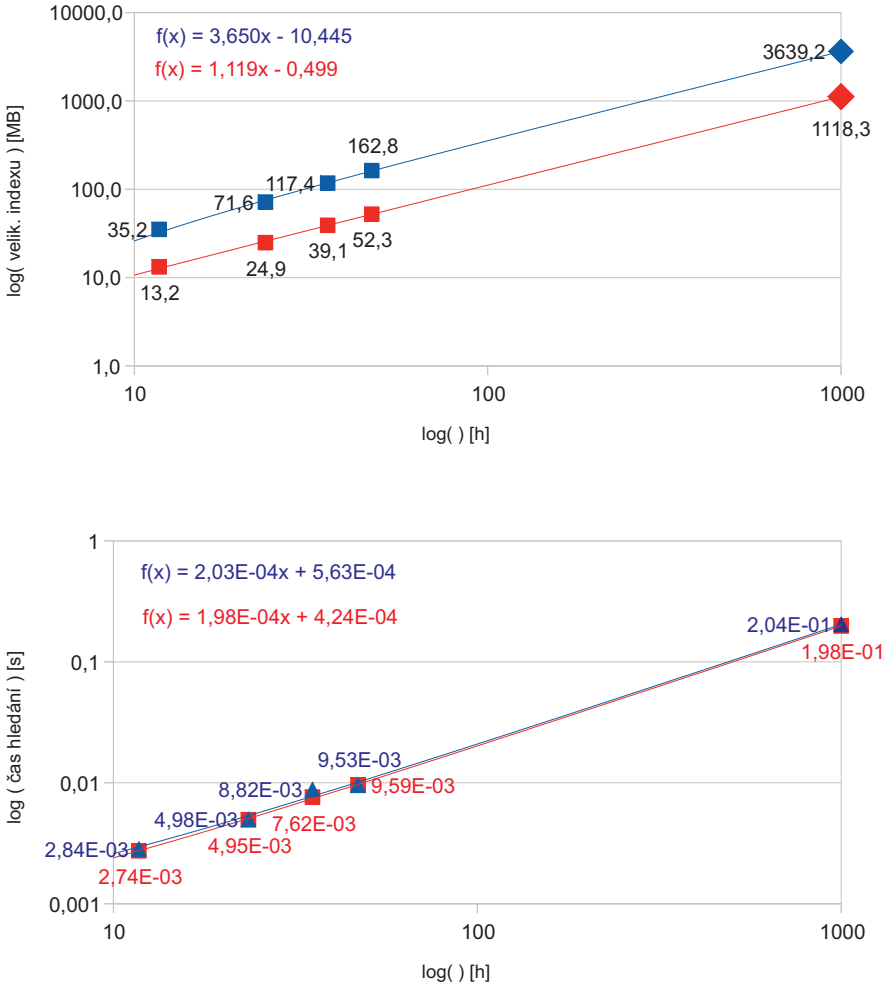
6.4 Vyhodnocení indexu fonémových mřížek

6.4.1 Přesnost

Podíváme-li se na grafy ROC charaketristik na obr. 30 můžeme z nich zřetelně vidět lepší průběh křivky pro systém s WFST oproti stávajícímu systému. Ohyb je ostřejší což značí, že detekční schopnost u novějšího přístupu při stejném množství falešných poplachů narůstá rychleji. Vezmeme-li kupříkladu bod při kterém dostáváme 0,25 FA/term/hod, tak dosavadní přístup s indexací vybraných hran mřížky má detekční schopnost 57,49%, zatímco nový systém s transducery dává hodnotu 66,91%. Zatímco hodnota FOM nového a starého systému je prakticky stejná, tak nový systém poskytuje navíc nižší hodnotu rovnosti chyb EER.

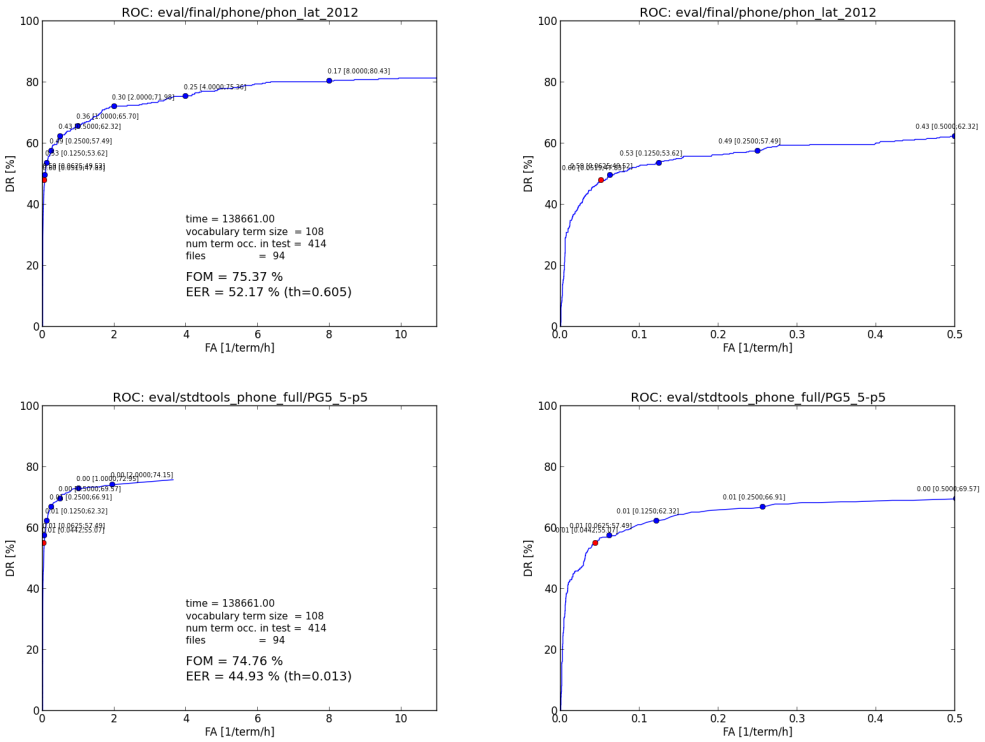
6.4.2 Nároky na úložný prostor, rychlost vyhledávání

Stejně jako u fonémových mřížek jsem i zde sestavil grafy závislostí velikosti indexu a doby vyhledávání na počtu indexovaných hodin (obr. 31). Ukazuje se, že fonémový index nad stejnou množinou dat má vskutku enormní prostorové nároky - více než 255 krát větší než index slovní. Extrapolovaná hodnota pro index nad 1000-hodinovým MALACHovským archivem říká, že by při tomto stavu zabíral bezmála 1 TB dat! To zatím představuje vážný problém, který však není neřešitelný. Především je třeba si uvědomit, že takovýto index obsahuje všechna rozpoznaná slova, čímž de facto supluje i úlohu indexu slovního. Hlavním důvodem indexace fonémových mřížek je ale řešení problému OOV slov. Do budoucna tedy bude potřeba zaměřit se na redukci indexovaných dat ve smyslu odstranění slov, která jsou ve slovníku rozpoznávače a která jsou tedy již obsažena i ve slovních mřížkách.



Obrázek 29: Grafy závislosti velikosti indexu v [MB] (nahoře) a doby vyhledávání v [s] (dole) na počtu hodin indexovaných řečových nahrávek. Červený průběh je pro index s omezením délky faktoru 2, modrý je pro index bez tohoto omezení. První čtyři body představují [11,75; 23,5; 35,25; 47] hod, tj. [25, 50, 75, 100] % referenčních dat. Poslední body jsou hodnoty pro celý archiv, získané extrapolací s pomocí regresní přímky.

Z grafů pro dobu vyhledávání je patrné, že pro experimentální data závislost doby prohledávání nevyšla zcela přesně jako lineární. Jak již bylo zmíněno v odstavci 6.3.2, bude to patrně způsobeno nerovnoměrnou velikostí mřížek v jednotlivých indexovaných páskách. Vyhledávací časy jsou oproti

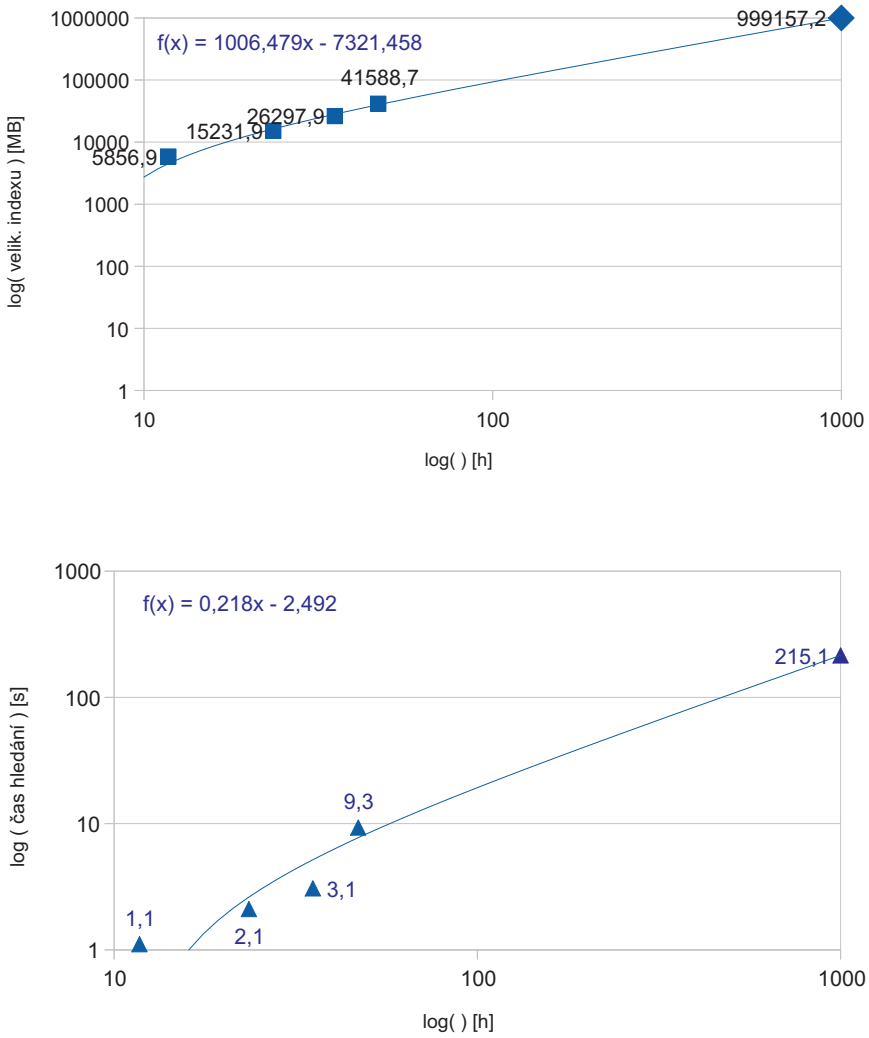


Obrázek 30: Srovnání křivek ROC pro systém indexace hran mřížky (nahoře) a systém s WFST (dole). Levý graf zobrazuje rozsah osy x pro hodnoty[0;10], zatímco pravý graf je výřez pro průběh charakteristiky do 0,5 FA/term/hod. Body v křivce představují pracovní body při daném parametru θ (v grafu ozn. "th"), červeně označený je pracovní bod, při kterém bylo dosaženo rovnosti chyb EER.

slovnímu indexu obecně daleko vyšší. To je v první řadě způsobeno mnohonásobně větší velikostí indexu (co do počtu hran a stavů) a také tím, že fonetické dotazy jsou nepoměrně delší, tj. obsahují více hran a stavů, než dotazy na jedno slovo s pouhou jedinou odpovídající hranou.

6.5 Shrnutí

Závěrečné shrnutí všech zkoumaných hledisek pro stávající i nový systém ukazuje tabulka 8. V případě slovních mřížek lze tedy obecně říci, že při srovnatelné úspěšnosti (FOM kolem 92%) nový přístup s transducery zaostává jen mírou rovnosti chyb (EER = 21,39 oproti původnímu EER = 13,9) a také o něco vyššími nároky na úložný prostor. Na druhou stranu však



Obrázek 31: Grafy závislosti velikosti indexu v [MB] (nahoře) a doby vyhledávání v [s] (dole) na počtu hodin indexovaných řečových nahrávek. Červený průběh je pro index s omezením délky faktoru 2, modrý je pro index bez tohoto omezení. První čtyři body představují [11,75; 23,5; 35,25; 47] hod, tj. [25, 50, 75, 100] % referenčních dat. Poslední body jsou hodnoty pro celý archiv, získané extrapolací s pomocí regresní přímky.

tyto nevýhody kompenzuje výrazně nižší dobou vyhledávání a do budoucna přináší i širší možnosti pro způsob tvorby dotazů včetně těch na řetězce libo-

volné délky. U fonémového indexu jsme novým přístupem s jistotou dosáhli zlepšení detekčních schopnosti v jednotlivých bodech ROC křivky a snížením EER z 52,17 na 44,93. Co je však oproti stávajícímu systému výrazně horší, je velikost jeho úložného prostoru. Rozdíl v rychlosti vyhledávání není tak dramatický, jako v případě slovního indexu ovšem tomuto údaji nelze přičítat příliš velkou váhu vzhledem k jeho zřejmě nepřesnému odhadu.

Systém indexace	FOM [%]	EER	θ	Velikost indexu [MB]	Doba vyhledání [s]
Současný	92,4	**13,9**	0,184	**894,3**	7,74
WFST	91,485	21,39	0,719	1118,3	**0,19**

Systém indexace	FOM [%]	EER	θ	Velikost indexu [MB]	Doba vyhledání [s]
Současný	75,37	52,17	0,605	**6539,82**	229.61
WFST	74,76	**44,93**	0,013	999157,2	**(215,1)**

Tabulka 8: Srovnání všech zkoumaných hledisek systému indexace vybraných mřížkových hran a systému indexace kompletních mřížek s pomocí WFST. Horní tabulka je pro index slovní, spodní pro index fonémový. V tabulce jsou zvýrazněné ty hodnoty, u kterých byl naměřen výraznější rozdíl. Pozn.:Doba vyhledání je dána součtem časů ze samostatně kladených jednoslovných dotazů.

7 Závěr

7.1 Shrnutí dosažených výsledků

Cílem této práce bylo nastudovat problematiku úlohy vyhledávání slov v rozsáhlém řečovém archivu spolu s teorií vážených konečných transducerů a jejich aplikací v této úloze. S využitím knihoven a nástrojů pro indexaci a vyhledávání pak tento přístup použít pro slovní i fonémové mřížky a vyhodnotit jeho přínos v porovnání s přístupem, založeným na indexaci jen vybraných mřížkových hran.

Čtenáři se na první pohled může zdát, že pouhá aplikace již hotových metod není nikterak složitá, ale během této práce se ukázalo, že tomu tak ani zdaleka není. Výsledky použitých metod se mohou i dramaticky lišit, pokud jsou jejich vstupem odlišná data, než pro jaká byly v základní podobě navrženy. Proto bylo nutné s využitím jejich parametrizace nejprve nalézt jejich optimální nastavení a teprve potom začít s vlastním vyhodnocováním a analýzou jeho přínosu pro úlohu STD pro případ MALACHovského archivu.

Základní rozdíl v obou zkoumaných přístupech indexace je, že ten stávající vybírá z mřížek jen hrany s nejlepší váhou, kdežto přístup s WFST indexuje celou strukturu mřížky se všemi jejími faktory. Díky tomu obsahuje stávající index prakticky jen relevantní data s nejvyšší pravděpodobností, jeho přesnost je vysoká a samotný index nezabírá tolik úložného prostoru. Nový systém však se srovnatelnou nebo i ještě o něco lepší přeností (fonémové mřížky) poskytuje jednak výrazně rychlejší časy vyhledávání ale umožňuje navíc i vyhledávání všech podřetězců o libovolném počtu slov. Tato možnost sice způsobí navýšení nároků na úložný prostor indexu, ale bez výrazného vlivu na dobu jeho prohledávání. Ta zde totiž závisí jen na délce hledaného dotazu, který je výsledkem jeho kompozice se samotným indexem. Enormní prostorové nároky fonémového indexu tak představují největší nevýhodu nového systému s transducery. Prozatím dosažené výsledky však nejsou definitivní a lze zde ještě uplatnit několik možností redukce velikosti tohoto indexu. O tom se ještě zmíním v následujících odstavcích.

Rozdíl mezi oběma přístupy lze spatřit i v řešení jak zacházet s redundantními hypotézami výskytu stejných faktorů ve stejném čase mřížky. Zatímco starý přístup opět vybírá hrany s nejlepším skóre, tak přístup s transducery se snaží určitým způsobem respektovat jejich pravděpodobnostní rozložení v indexované promluvě a tyto redundantní hypotézy slučuje dohromady prostřednictvím logaritmického součtu jejich jednotlivých vah.

Nakonec lze zmínit i fakt, že indexace a vyhledávání s WFST jsou zastřešeny jednotnou, dobře podloženou matematickou teorií. Ta ve své podstatě vychází z obecné teorie grafů, která je do dnešních dob již poměrně rozsáhle prostudovaná a na jejím poli je, zejména v oblasti IT, vyřešeno mnoho souvisejících otázek a problémů. Díky tomu lze tuto metodu ještě dále rozvíjet a vylepšovat.

7.2 Budoucí práce

Implementace a experimenty popsané v této práci představovaly teprve první pokusy, jejichž cílem bylo de facto posoudit opodstatněnost myšlenky nasazení technologie WFST pro úlohu STD. Ta se sice ukázala jako přínosná, ale vyplynulo z ní mnoho dalších otázek a problémů, které je pro její reálné nasazení v praxi nutno dořešit.

V první řadě je to optimalizace prostorových nároků indexu, potřebná zejména v případě fonémových mřížek. Během experimentů se vynořilo několik myšlenek, které by mohly výrazně přispět k redukování prohledávaného prostoru mřížek a v konečném důsledku ještě i dále snížit čas vyhledávání:

- Účelem fonémových mřížek je především vyhledávání OOV slov, ale dosud zkoumaný fonémový index zahrnuje všechna rozpoznaná slova z dané promluvy. S využitím operací difference [5], tj. odečtení dvou automatů, by bylo možné z indexu odstranit např. stop-slova a slova, která se nacházejí ve slovníku rozpoznávače. Tato jsou již obsažena ve slovních mřížkách a je tedy zbytečné, aby zabíraly další místo navíc.

- Rychlost vyhledávání v takto rozsáhlých datových struturách lze ještě dále snížit. Kromě samotné kompozice automatů dotazu a indexu, zabírá nejvíce času jeho načítání do virtuální paměti. Toto lze urychlit s použitím metod, které umožňují provádět mapování transducerů z disku do formátu datových bloků tak, jak jsou přímo uloženy v paměti. Prostřednictvím adres těchto datových bloků se pak připojí do virtuální paměti a načítají se líně přímo za běhu kompozice. Navíc se načítají jen ty bloky, které jsou při kompozici skutečně potřeba, tj. jen stavy a hrany, které odpovídají hledanému dotazu. Tento přístup je již v současné době ve vývoji jako součást knihovny *pyfst* a již první experimenty se jeví jako velice slibné.

Ve stati 6.2.3 byl popsán problém detekční schopnosti v závislosti na možné (ne)přítomnosti některých fonémů nebo jejich záměně v rozpoznaných slovech. Experimenty ukázaly, že ke zlepšení výsledku postačí vynechání jednoho libovolného fonému a přidání jednoduché konfuzní tabulky. Ohodnocení těchto operací bylo prozatím stanoveno z několika pokusů nad malou částí testovacích dat a budoucí práce tedy bude směřovat k rozvinutí a podrobnějšímu zkoumání této myšlenky.

V případě slovních mřížek byl též zmíněn fakt, že tvůrci nástrojů *stdtools* počítají se skóre na hranách mřížek jako se sdruženou pravděpodobností $P(W, O)$, vypočtenou součinem váhy akustického a jazykového modelu. Kdežto naše data jsou výsledkem aposteriorní pravděpodobnosti $P(W|O)$, normalizovanou tak, aby součet paralelních hypotéz byl v každém časovém okamžiku roven jedné - odstavec 6.3. Otázka vlivu tohoto faktu na podobu pravděpodobností ve výsledném indexu bude předmětem dalšího zkoumání, protože během tvorby indexu se vlivem operace stlačení vah skóre jednotlivých faktorů zmenšují. To se pak následně může promítnout i do detekční schopnosti a počtu falešných poplachů.

Nakonec ještě zmiňme vlastnost indexu s WFST, která možná není na první pohled tak patrná, ale v reálném nasazení může představovat problém. Takto konstruovaný index je totiž velmi těžkopádně rozšiřitelný o nově přidaná data. Možnost přidávání a odebírání záznamů z řečového archivu je sice samozřejmostí, avšak pro vyhledávání to znamená jeho kompletní přegenerování. Řešení spočívá např. v rozdělení indexu na menší podsoubory (což je ostatně nezbytné i z hlediska jeho spočitatelnosti), ke kterým je možno operací sjednocení transducerů [5] nově napočítané mřížky přidávat. Takový automat ovšem nebude nedeterministický a minimální, proto by bylo nutné občas index tak jako tak celý přegenerovat. Jiná možnost je dívat se na problém ne jako na indexaci promluv (obsahujících slova nebo i celé věty) v rámci celých videonahrávek, ale jako na index menších jednotek, tj. slov nebo jejich částí, nad všemi nahrávkami v celém archivu. Takováto struktura by sice zabírala v podstatě stejné množství datového prostoru, ale umožňovala by patrně ještě rychlejší vyhledávání a především efektivnější aktualizaci indexu.

Po vyřešení všech výše popsaných problémů předpokládáme v budoucnu integraci nástrojů s WFST do knihovny Voiar (angl.: Voice Archive), vyvinuté na Katedře kybernetiky ZČU pro práci s rozsáhlými řečovými archivy. Zde by měla popisovaný systém indexace i zcela nahradit.

Reference

[1] Psutka, Josef; Švec, Jan; Psutka, Josef; Vaněk, Jan; Pražák, Aleš; Šmídl, Luboš; *Fast Phonetic/Lexical Searching in the Archives of the Czech Holocaust Testimonies: Advancing Towards the MALACH Project Visions.* Lecture Notes in Computer Science, 2010, roč. 2010, č. 6231, s.385-391.

[2] Psutka, Josef; Švec, Jan; Psutka, Josef; Vaněk, Jan; Pražák, Aleš; Šmídl, Luboš; Ircing Pavel; *System for fast lexical and phonetic spoken term detection in a Czech cultural heritage archive*, EURASIP Journal on Audio, Speech, and Music Processing, 2011.

[3] Can, Dogan; Saraclar, Murat. *Timed Indexation of Weighted Automata - Application to Spoken Term Detection*, IEEE Transactions on Audio, Speech and Language processing, vol. 18, No. 8, November 2010

[4] Can, Dogan; *Indexation, Retrieval & Decision techniques for Spoken Term Detection*, Master thesis, Bogaziçi University, 2006

[5] Allauzen, Cyril; Riley, Michael; Schalkwyk, Johan; Skut, Wojciech; Mohri, Mehryar; *OpenFst: A General and Efficient Weighted Finite-State Transducer Library*, Lecture Notes in Computer Sciance, vol. 4783. Springer, 2007, pp. 11-23. www.openfst.org

[6] Psutka, Josef; Müller, Luděk; Matoušek, Jindřich; Radová, Vlasta; *Mluvíme s počítačem česky*, ACADEMIA, Praha 2006

[7] Čada, Roman; Kaiser, Tomáš; Ryjáček, Zdeněk; *Diskrétní matematika*, FAV ZČU 2004

[8] Šmídl, L.; *Metody rychlé detekce klíčových slov*, Disertační práce, FAV ZČU 2005.

[9] Psutka, Josef; Kepka, Jiří; *Strukturální metody rozpoznávání: umělá inteligence*, ZČU Plzeň, 1993

[10] Povey, D.; Hannemann, M.; Boulianne, G.; Burget, L.; Ghoshal, A.; Janda, M.; Karafiát, M.; Kombrink, S.; Motlíček, P.; Qian, Y.; Riedhammer, K.; Veselý, K.; Thang Vu, N.; *Generating exact lattices in the WFST framework*, ICASSP 2012;

[11] Wessel, Frank; Schlüter, Ralf; Mecherey, Klaus; Ney, Hermann; *Confidence Measures for Large Vocabulary Continuous Speech Recognition*, IEEE Transactions on Speech and Audio processing, vol. 9, No. 3, March 2001;

[12] Kuich, W.; Salomaa, A.; *Smirings, automata, languages.* Springer Verlag, 1986.

[13] Mohri, M.; Pereira F.; Riley, M.; *Weighted Finite-State transducers in speech recognition*, Computer Speech & Language, vol. 16, No. 1, pp. 69-88, 2002.

[14] Mohri, M.; *Semiring Frameworks and Algorithms for Shortest-Distance Problems*, Journal of Automata, Languages and Combinatorics, vol. 7, No. 3, pp. 321-350, 2002.

[15] Blumer, A.; Blumer, J.; Ehrenfeucht, A.; Haussler, D.; Chen, M.T.; Seiferas, J.; *The smallest automaton recognising the subwords of a text*, Theoretical Computer Science, vol. 40, pp. 31-55, 1985

[16] Crochemore, M.; *Transducers and repetitions*, Theoretical Computer Science, vol. 45, No. 1, pp. 63-86, 1986

[17] Mohri, M.; Moreno, P.; Weinstein, E. *General suffix automaton construction algorithm and space bounds*, Theoretical Computer Science, vol. 410, No. 37, pp. 3553-3562, 2009

[18] Mohri, M.; *Finite-State Transducers in Language and Speech Processing*, Computational Linguistics, vol. 23, No. 2, pp. 269-311, 1997.

[19] Young, S. et al.; *The HTK Book (for HTK Version 3.4)*, Cambridge University Engineering Department 2006.