Rečové interaktívne komunikačné systémy

Matúš Pleva, Stanislav Ondáš, Jozef Juhár,
Ján Staš, Daniel Hládek,
Martin Lojka, Peter Viszlay

Ing. Matúš Pleva, PhD.
Katedra elektroniky a multimediálnych telekomunikácií
Fakulta elektrotechniky a informatiky
Technická univerzita v Košiciach
Letná 9, 04200 Košice
Matus.Pleva@tuke.sk

Táto učebnica vznikla s podporou Ministerstvo školstva, vedy, výskumu a športu SR v rámci projektu KEGA 055TUKE-04/2016.

Názov: Rečové interaktívne komunikačné systémy

Autori: Ing. Matúš Pleva, PhD., Ing. Stanislav Ondáš, PhD., prof. Ing. Jozef Juhár, CSc., Ing. Ján Staš, PhD., Ing. Daniel Hládek, PhD., Ing. Martin Lojka, PhD., Ing. Peter Viszlay, PhD.

Vydal: Technická univerzita v Košiciach
Vydanie: prvé

Rukopis neprešiel jazykovou úpravou.

ISBN 978-80-553-2661-0

Obsah

Zoznam obrázkov

Zoznam tabuliek

1

Úvod

Predložená učebnica sa venuje tzv. *rečovým interaktívnym komunikačným systémom*, pričom pod tento pojem spadajú rôzne kategórie systémov s rôznymi atribútmi. Našou snahou bude uviesť čitateľa do tejto problematiky a oboznámiť ho s jednotlivými systémami, ich architektúrou, s ich hlavnými komponentmi a použitými technológiami. Naším cieľom je tiež ponúknuť prehľad technológií, ktoré slúžia na návrh a vývoj interaktívnych komunikačných služieb a aplikácií, pre ktoré je typické, že umožňujú aj interakciu pomocou reči.

Na úplnom začiatku považujeme za nevyhnutné zadefinovať pojem **interaktívny telekomunikačný systém**, ako aj ďalšie pojmy a tiež rozdeliť tieto systémy do jednotlivých kategórií.

Interaktívne telekomunikačné systémy (ITS) sú systémy, ktoré umožňujú interakciu medzi človekom a strojom a sú poskytované cez telekomunikačné siete. Do kategórie ITS systémov spadá veľké množstvo rôznych systémov s rôznymi atribútmi, preto je dôležité ich ďalej kategorizovať.

Podľa počtu modalít, ktoré sú zapojené do výmeny informácií medzi človekom a strojom, môžeme ITS systémy rozdeliť na *unimodálne* a *multimodálne*. V prípade unimodálnych systémov, komunikácia prebieha jedinou modalitou. Typickým predstaviteľom tejto skupiny sú *rečové interaktívne systémy*, ktoré sú založené zvyčajne na rečovom dialógu. Naproti tomu multimodálne interaktívne systémy využívajú minimálne dva druhy vstupných alebo výstupných modalít. Príkladom môžu byť aplikácie pre smartfóny, ktoré umožňujú zadávanie vstupov hlasom alebo cez virtuálnu klávesnicu. Ako výstup takýchto systémov môžu byť použitá syntetizovaná reč spolu s grafikou na displeji. Iným príkladom multimodálnych systémov môžu byť systémy, ktoré slúžia na komunikáciu s robotmi, napr. humanoidmi. Komunikácie s takýmito systémami je typicky multimodálna, kde zväčša humanoidný robot dokáže rozpoznávať nielen ľudskú reč ale aj gestá rúk, mimiku tváre

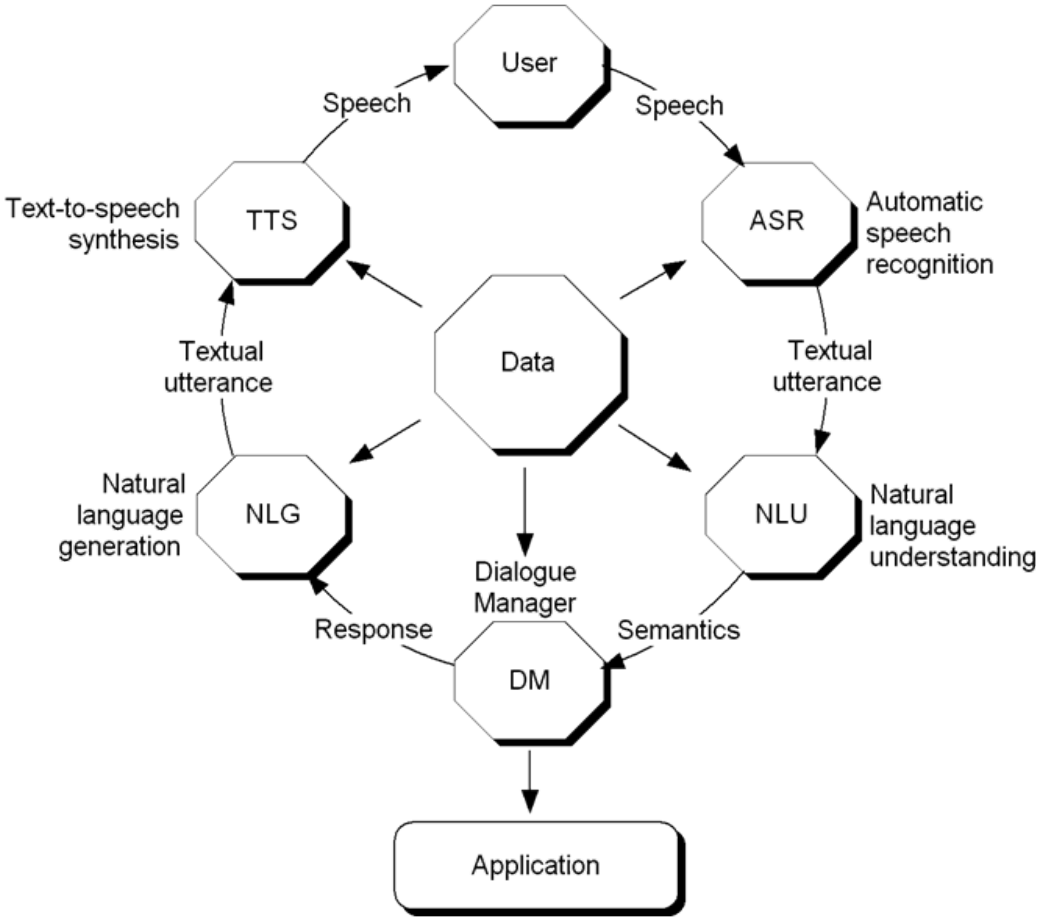
alebo postoj tela. Robot samotný potom, pre vyjadrenie svojej odpovede využíva zvyčajne tiež viacero modalít - syntetizovanú reč, gestá rúk, pohyby hlavy, grafický výstup na displeji (napr. robot Softbank Pepper).

Ďalšie rozdelenie ITS systémov môžeme urobiť podľa typu telekomunikačnej siete. Podľa tohto kritéria môžeme ITS systémy rozdeliť na tie, ktoré poskytujú svoje služby cez telefónnu sieť a ich hlavnou črtou je hlasový dialóg alebo na systémy, ktoré poskytujú svoje služby cez internet a ich súčasťou je aj rečové rozhranie.

Bolo by možné pokračovať v ďalšom delení ITS systémov, avšak to nie je naším cieľom. V rámci tejto učebnice budeme na tieto systémy nazerať ako na rozhrania medzi človekom a strojom (HMI - Human-Machine Interface). Presnejšie, budeme mať na mysli predovšetkým rozhrania medzi človekom a strojom, ktoré ako jednu z komunikačných modalít používajú reč.

Rečové rozhrania medzi človekom a strojom sú inšpirované dialógovou interakciou medzi ľuďmi navzájom a implementujú *komunikačný reťazec medzi človekom a strojom* (human-machine communication chain), ktorý je zobrazený na Obr. 1.1 .

Predstavený komunikačný reťazec medzi človekom a strojom definuje základné moduly, ktoré sú nevyhnutné na výmenu informácií pri takejto komunikácii. Reč od používateľa vstupuje do bloku automatického rozpoznávania reči (ASR - Automatic Speech Recognition), ktorý dekóduje sekvenciu parametrických vektorov reprezentujúcich zachytený úsek reči a generuje na svojom výstupe najpravdepodobnejšiu sekvenciu rozpoznaných slov. Táto sekvencia slov následne vstupuje do bloku porozumenia prirodzeného jazyka (NLU - Natural Language Understanding), kde je transformovaná na sémantickú reprezentáciu vhodnú pre spracovanie jednotkou riadenia dialógu (DM - Dialogue Manager). Dialógový manažér je akýmsi "mozgom"celého systému. Rozhoduje o nasledujúcom kroku v interakcii a konštruuje odpoveď používateľov vo forme sémantickej reprezentácie informácie, ktorá má byť prezentovaná používateľovi. Takáto informácia má často formu atribút-hodnota párov a je potrebné ju transformovať do formy ucelenej vety v prirodzenom jazyku. Tento proces je realizovaný v bloku generovania prirodzeného jazyka (NLG - Natural Language Generation). Po skonštruovaní ucelenej odpovede systému, ju blok syntézy reči z textu (TTS - text-to-speech) transformuje na akustický signál, ktorý je prehraný používateľovi. Popísaný scenár predstavuje základnú interakčnú sľučku, ktorár umožňuje dialógovú výmenu v rečovo-orientovaných rozhraniach medzi človekom a strojom. Rečové dialógové systémy priamo implementuju tento scenár.



Obr. 1.1: Komunikačný reťazec medzi človekom a strojom [63]

1.1 Rečové dialógové systémy

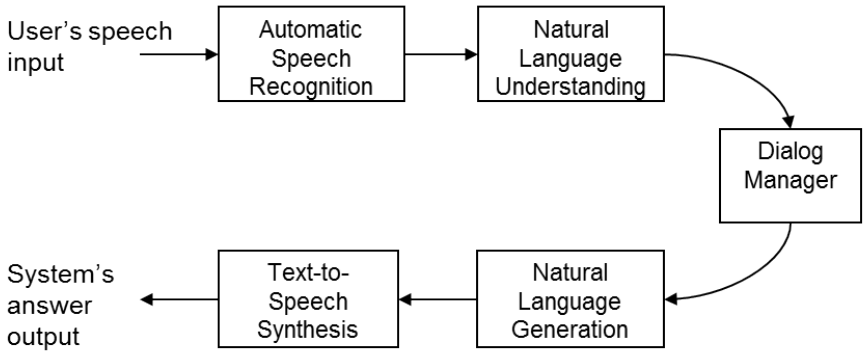
Rečové dialógové systémy (SDS - Spoken Dialogue Systems) sú počítačovými programami, vyvinutými pre realizáciu interakcie s používateľmi s využitím reči na poskytnutie špecifickej automatickej služby [45].

Rečové dialógové systémy patria do skupiny unimodálnych interaktívnych systémov, čo znamená, že akceptujú na svojom vstupe iba jednu modalitu - ľudskú reč a produkujú ako svoj výstup takisto rečový signál, zvyčajne generovaný TTS systémami. Pre zabezpečenie výmeny informácií využívajú dialógovú interakciu, v rámci ktorej získavajú informácie od používateľa potrebné na naplnenie jeho cieľov.

Služby poskytované SDS systémami zvyčajne nazývame *automatizované hlasové služby*. Typickými predstaviteľmi takýchto služieb sú služby predpo-

vede počasia, služby informujúce o odchodoch verejnej dopravy, služby call centier, help deskov, rezervácií hotelových izieb alebo leteniek a pod.

V súčasnosti sú SDS systémy často integrované v komunikačných systémoch automobilov alebo v tzv. personálnych virtuálnych asistentoch, kde sú často rozšírené pre spracovanie aj iných modalít.

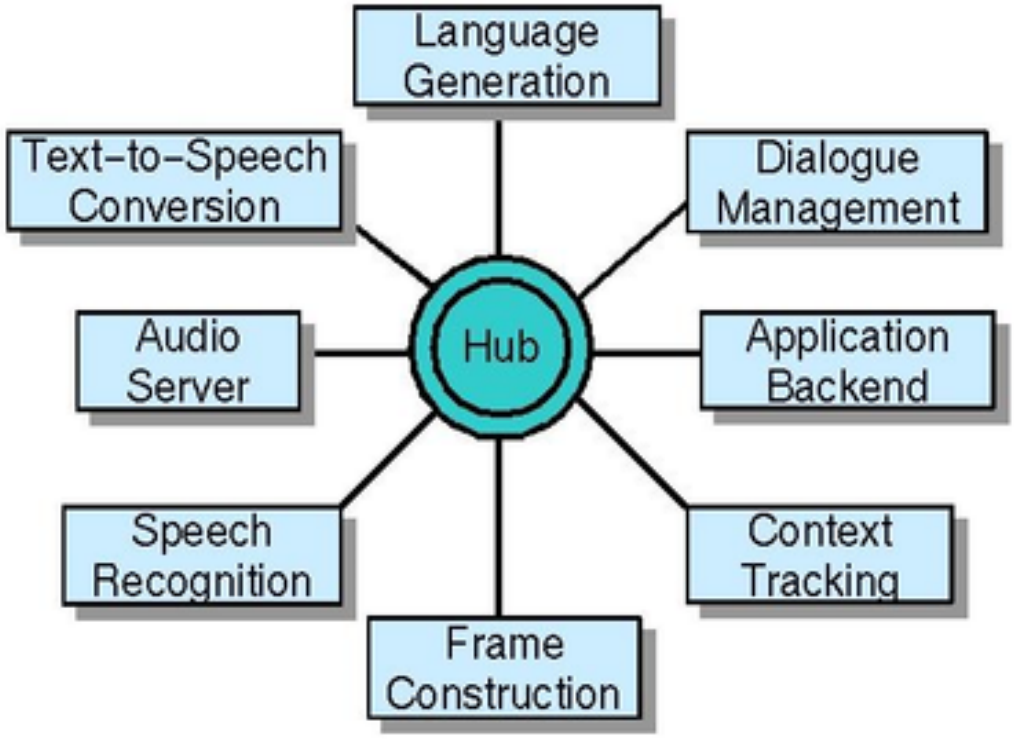


Obr. 1.2: Základná architektúra rečového dialógového systému

Základná architektúra rečového dialógové systému, ktorá priamo implementuje komunikačný reťazec medzi človekom a strojom je zobrazená na Obr. 1.2. Takýto typ architektúry nazývame zreťazená architektúra (pipeline architecture), pretože jej komponenty sú pospájané do série tak, že výstup predošlého modulu je vstupom nasledujúceho modulu. Spracovanie v takejto architektúre je teda synchrónne, z čoho vyplývajú niektoré obmedzenia a nevýhody, ako napr. väčšie oneskorenie systému a neexistujúca spätná väzba medzi komponentmi. Tieto nedostatky viedli k návrhu iných typov architektúr, ktoré umožňujú väčšiu flexibilitu spravovania a väčšiu kooperáciu medzi komponentami. Typickým predstaviteľom takýchto architektúr je **distribuovaná hub-server architektúra**.

Typickým príkladom SDS systému založenom na distribuovanej architektúre typu hub-server je Galaxy Communicator. Hub architektúra samotná bola vyvinutá vrámci projektu financovaného grantovou agentúrou DARPA (Defense Advanced Research Projects Agency), kde bola implementovaná v DARPA komunikátore. DARPA Communicator [93] je voľne dostupný systém, ktorého hlavným komponentom je tzv. "Hub". Hub distribuuje komuni-

káciu medzi službami ostatných komponentov - servermi. Galaxy Communicator Obr. 1.3 bol vybudovaný na tejto hub-server architektúre [71].



Obr. 1.3: Architektúra Galaxy komunikátora

Galaxy Communicator je distribuovaná, na správach založená hub-server softérová infraštruktúra, optimalizovaná pre vývoj rečových dialógových systémov. Pozostáva z hlavných komponentov, ako sú NLG modul (Language Generation), TTS modul (Text-to-Speech Conversion), ASR modul (Speech Recognition), DM modul (Dialogue management, Vstupno-výstupný modul (Audio Server) a pod. Posledne menovaný (Audio server) predstavuje rozhranie systému do telefónnej siete. Sprostredkováva audio dáta od používateľa a k používateľovi a riadi telefónne spojenie. Ďalším komponentom je tzv. Backend server, ktorý zabezpečuje komunikáciu s externými zdrojmi dát (z internetu, databáz).

V oblasti distribuovaných architektúr je možné nájsť aj trochu odlišné systémy v porovnaní s hub-server štruktúrov systému DARPA alebo Galaxy, kde namiesto centrálneho hub, ktorý iba preposiela správy medzi servermi, figuruje centrálny dialógový manažér.

Iným typom architektúry sa vyznačujú napr. agentové systémy, ktoré sú tvorené skupinou agentov s rôznymi úlohami. V takýchto systémoch sa nevyužíva centrálny prvok typu "hub", ale komunikácia prebieha medzi agentmi alebo prostredníctvom iného zdieľaného priestoru (napr. blackboard alebo whiteboard systémy).

1.2 Multimodálne interaktívne systémy

Napriek tomu, že rečové dialógové systémy boli v minulosti veľmi rozšírené a pre mnohé aplikácie sú nápomocné doteraz, najmä vďaka technologickému pokroku ich v mnohých oblastiach nahrádzajú multimodálne interaktívne systémy. Hlavným dôvodom je samozrejme to, že interakcia medzi ľuďmi je prirodzene multimodálna. Ľudia v interakcii s inými osobami používajú reč, gestá, pohľad, čuch, hmat, a tiež iné komunikačné kanály (ako napr. postoj tela), ktoré realizujú výmenu informácií. Mnohé z týchto prvkov je preto výhodné preniesť aj do interakcie so strojmi, resp. s robotickými systémami, nakoľko je využívanie týchto komunikačných kanálov pre človeka prirodzené a ľahko naučiteľné. Špeciálne je multimodálna komunikácia nevyhnutná v interakcii s humanoidnými robotmi, nakoľko ich výzor, napodobňujúci človeka, budí v používateľoch očakávania, podobné ako by mali voči inému človekovi.

Multimodálny dialógový systém (MDS - The Multimodal Dialogue System) je interaktívny systém, ktorý umožňuje interakciu medzi človekom a strojom pomocou viacerých vstupných a/alebo výstupných modalít, akými sú reč, gestá, dotyky a pod.

Všeobecná architektúra multimodálneho systému je zobrazená na Obr. 2.2 neskôr. Multimodálnym interaktívnym systémom sa podrobne venuje nasledujúca kapitola.

1.3 Aplikácie rečových interaktívnych komunikačných systémov

Rečové technológie a rečové interaktívne komunikačné systémy sa postupne stávajú súčasťou nášho každodenného života. V súčasnosti namiesto vyťukávania dlhšieho dotazu na virtuálnej klávesnice smartfónu môžeme, vďaka Google Voice search, jednoducho povedať čo si želáme vyhľadať. Vlastníci iPhonov sa môžu porozprávať s virtuálnym asistentom SIRI. Maličký čierny valec Amazon Echo skrýva virtuálnu asistentku Alexa, ktorá umožní vyhľadávať rôzne informácie, prehrávať hudbu alebo niekomu zavolať - a to všetko bez použitia klasických rozhraní, akými sú klávesnica, myška alebo dotykový

displej. Amazon Echo sa ovláda iba hlasom (OK, dá sa ovládať aj z aplikácie telefónu).

Na začiatku vývoja takýchto rečových interaktívnych systémov a aplikácií bola situácia značne odlišná. Neexistovali smartfóny a internet bol dostupný nanajvýš v internetových kaviarňach na rozmerných stolných počítačoch. V tomto kontexte, rečové rozhrania medzi človekom a strojom mali často podobu tzv. IVR systémov (Interactive Voice Response system), ktoré poskytovanie automatických služieb cez telefón (napr. aj cez analógovú pevnú telefónnu sieť), takým spôsobom, že používateľovi boli prehrávané prednahraté výzvy alebo neskoršie syntetická reč, na ktorú mohol reagovať stlačaním tlačidiel na telefóne (tzv. DTMF voľba). IVR systémy boli často nasadzované v call centrách alebo pre jednoduché aplikácie predpovede počasia, objednávky pizze a pod.

Vďaka neustálemu zlepšovaniu sa technológie rozpoznávania reči sa neskôr IVR systémy transformovali na typické rečové dialógové systémy, ktoré umožňovali poskytovanie komplexnejších služieb, kde používateľ popri voľbe typu DTMF mohol poskytnúť informácie aj pomocou vyslovenia izolovaných slov, číslic alebo krátkych fráz. Použitie takýchto systémov už bolo o čosi komfortnejšie. Moderné rečové dialógové systémy už zväčša umožňujú používateľovi zadávať svoje požiadavky vo forme viet v prirodzenom jazyku, vďaka zlepšeniu sa systémov pre rozpoznávanie plynulej reči s veľkými slovníkmi (LVCSR - Large Vocabulary Continuous Speech Recognition). To umožnilo výrazne skrátiť čas potrebný na získanie požadovanej informácie a interakcia medzi človekom a počítačom sa stala viac podobnou medziľudskej komunikácii.

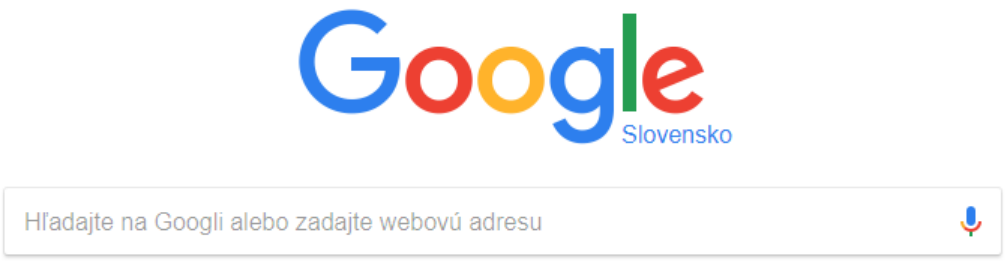
To umožnilo rečovým dialógovým systémom poskytovať komlexnejšie kooperatívne služby ako napríklad služby rezervácie leteniek, rezervácie dovoleniek a ubytovania, tzv. hlasový banking a pod.

Začiatkom milénia sa situácia opäť výrazne mení vďaka výraznemu rozšíreniu sa internetovej konektivity v mobilných zariadeniach a vďaka nástupu smart zariadení s dotykovými displejmi, čo významne ovplyvnilo spôsob interakcie medzi človekom a strojom.

Tieto zmeny posunuli interakciu od zadávania vstupu pomocou klávesnice (klasickej a telefónnej) a myši smerom k používaniu dotykového disleja (virtuálnej klávesnice, dotykov a dotykových gest) a od unimodálneho módu interakcie smerom k multimodálnej forme interakcie.

Príkladom môže byť práve Google vyhľadávací engine (Obr. 1.4), ktorý môže byť vnímaný ako multimodálny systém, nakoľko umožňuje hľadaný text buď napísať pomocou klávesnice alebo jednoducho vysloviť po kliknutí na ikonku mikrofónu. Výsledok vyhľadávania sa zobrazí ako na displeji ako zoznam nájdených výsledkov v podobe textu a grafiky.

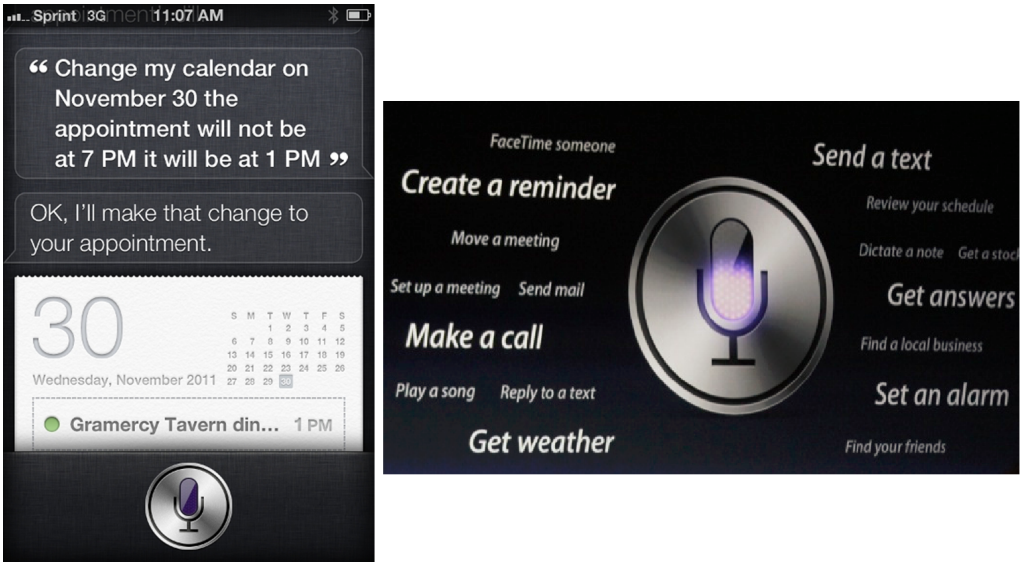
Vďaka postupnému zlepšovaniu technológie automatického rozpoznáva-



Obr. 1.4: Rozhranie Google vyhľadávača

nia reči a syntézy reči z textu, môžeme pozorovať rýchly vývoj pokročilých multimodálnych rozhraní medzi človekom a strojom, ktoré umožňujú rečovú dialógovú interakciu s podobnými atribútmi ako má medziľudský dialóg.

Takéto systémy sú označované ako virtuální asistenti alebo asistenti. Za prvý takýto masovo rozšírený systém môžeme označiť virtuálneho asistenta SIRI od spoločnosti Apple, ktorý bol predstavený v roku 2012. Je možné konštatovať, že sa jednalo o prvého reálne použiteľného a nápomocného virtuálneho asistenta, ktorý je demoštráciou najmodernejších systémov ASR, TTS a NLP (Natural Language Processing). Následne sa objavili ďalší asistenti od technologických gigantov - Google Now (2012), Microsoft Cortana (2014) alebo Samsung Bixby (2017).



Obr. 1.5: Ukážka funkcionalít virtuálneho asistenta Apple SIRI

Obr. 1.5 zobrazuje príklad interakčného scenára s virtuálnym asistentom Apple SIRI.

V oblasti inteligentných asistentov zvlášť rozlišujeme systémy, ktoré využívajú vizualizácie ľudského tela. Pre označenie tejto skupiny agentov sa zaužíval pojem Ëmbodied Conversational Agents"(ECA). Vzhľad podobný človeku dáva týmto asistentom možnosť používať vrámci interakcie napr. gestá, pohyby hlavy a očí a pod. Pomocou týchto modalít môžu jednoduchšie vyjadrovať emócie alebo poskytovať interakčnú spätnú väzbu, čo prispieva k hladkej interakcii.

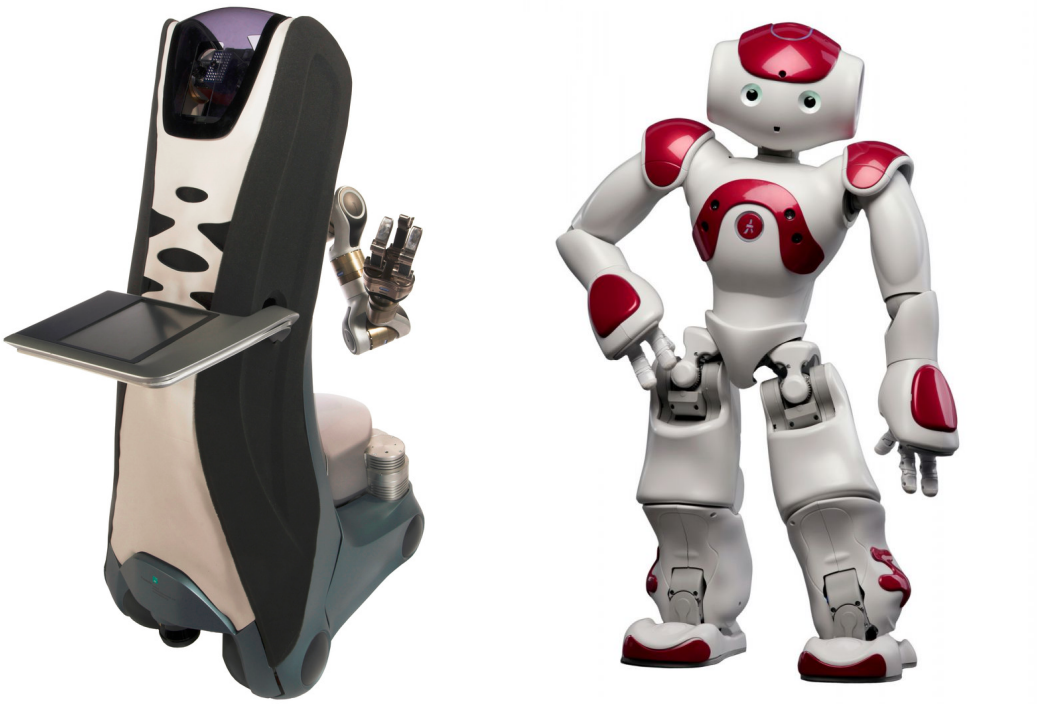
Príkladom takéhoto agenta môže byť systém GRETA vyvinutý na Telecom Paris Tech University (Obr. 1.6).

Obr. 1.6: Embodied Conversational Agent Greta

Od animovaných agentov je to už ïba malý skok"k robotom, ktorí môžu byť takisto vnímaní ako multimodálny interaktívny systém. Robotické systémy, ktoré sa zvyčajne spoliehajú na multimodálnu interakciu s používateľom môžeme ďalej rozdeliť na humanoidné roboty a ostatné robotické systémy (napr. roboti pre zdravotnú starostlivosť), ktoré nemusia mať výzor podobajúci sa na človeka.

Podľa uvedeného delenia je potrebné rozlišovať aj možné vstupné - výstupné modality, ktoré môžu byť použité na interakčné účely. Humanoidné roboty môžu komunikovať pomocou podobných výstupných modalít ako animovaní agenti resp. ako ľudia. V prípade napr. robotov určených pre zdravotnú starostlivosť alebo pre sprevádzanie starších osôb, ktoré často v sebe

integrujú napríklad dotykový displej, druh hardverových rozhraní definuje aj možné komunikačné modality.



Obr. 1.7: Car-o-bot3 (vľavo) a humanoidný robot NAO (vpravo)

Obr. 1.7 zobrazuje robotický systém Car-O-bot 3 (Fraunhofer IPA) navrhnutý pre oblasť zdravotnej starostlivosti a sprevádzania pre ľudí so zdravotným znevýhodnením resp. starších ľudí v porovnaní s humanoidným robotom NAO od spoločnosti Softbank Robotics.

V súčasnosti sa veľmi populárnymi stávajú tzv. rodinné alebo sociálne roboty (family or social robots), ktorých hlavnou doménou je oblasť domácnosti, kde plnia funkciu spoločníkov, ktorí okrem praktických schopností (zapnutie svetiel, nastavenie teploty, vyhľadanie receptov, a pod.) sú vybavení aj schopnosťami slúžiacimi na interaktívnu zábavu (napr. prehrávanie hudby a filmov, čítanie rozprávok, hranie hier, získavanie informácií a pod.). Je zrejmé, že pre túto skupinu robotov bude dôležité disponovať schopnosťami byť účastníkom dialógovej interakcie na veľmi vysokej úrovni. Na Obr. 1.8 je zobrazený rodinný robot Asus Zenbo.

Ďalšou aplikačnou oblasťou rečových interaktívnych systémov sú zariadenia pre domácnosť, ktoré integrujú koncept internetu vecí (Internet of

Obr. 1.8: Rodinný robot Asus Zenbo

Things). Už v súčasnosti je možné si kúpiť zariadenia ako sú Amazon Echo, Google Home alebo Apple HomePod (Obr. 1.9), ktoré okrem iného predstavujú inteligentný hlasom ovládaný hub, pomocou ktorého je možné ovládať zariadenia v domácnosti (napr. svetlá, klimatizáciu, kúrenie a pod.).

Obr. 1.9: Google Home, Amazon Echo, Apple Homepod (zľava doprava)

V pozadí týchto systémov pracujú sofistikované serverové interaktívne systémy, ktoré umožňujú hlasovú interakciu a hlasové vyhľadávanie informácie a ovládania rôznych zariadení.

Koncept internetu vecí v spojení s interaktívnymi systémami je možné považovať za blízku budúcnosť v oblasti domácich zariadení. Pre ilustráciu tohto trendu uvádzame príklad nového typu chladničky od spoločnosti LG, ktorá integruje 29 palcový, na výšku orientovaný displej, operačný systém webOS a virtuálnu asistentku Alexa od spoločnosti Amazon (Obr. 1.10).

Obr. 1.10: Chladnička LG s webOS a virtuálnou asistentkou Amazon Alexa

V neposlednom rade ďalšou aplikáciou rečových technológií a multimodálnych interaktívnych systémov sú infotainmentové a komunikačné systémy vozidiel. Už v súčasnosti je väčšina automobilov vybavená dotykovým displejom, za ktorým sa nachádza informačno-zábavno-komunikačný systém, ktorý je možné ovládať pomocou dotykov, rečových povelov alebo gest.

2

Multimodalita a mobilita v interaktívnych systémoch s rečovým rozhraním

Súčasné informačné systémy majú často k dispozícii väčší počet senzorov a výstupných rozhraní a to je výzvou pre vývojárov aby umožnili využívať viacero vstupných a výstupných kanálov súčasne a poskytovať tak jednoduchšiu, prirodzenejšiu a bohatšiu interakciu. Ak systém umožňuje využívanie viacero vstupných alebo výstupných komunikačných kanálov zároveň voláme ho multimodálny. Zároveň by systém mal v súčasnosti poskytovať služby bez obmedzovania používateľa v jeho polohe (viď Obr. 2.1[1]), teda by mal poskytovať mobilitu.

2.1 Multimodalita

Každý typ komunikačného kanála zabezpečujúci komunikáciu s používateľom je možné nazvať modalitou. Modality môžu byť vstupno-výstupné, ale niektoré sú len vstupné alebo len výstupné a môžu sa líšiť pri komunikácii s robotom, počítačom, mobilným terminálom, avatarom či inteligentným priestorom.

Medzi bežné modality môžeme zaradiť napríklad:

- Reč - rozpoznávanie (vstup); Syntéza reči (výstup)
- Grafické rozhranie - GUI (obraz – výstup)
- Dotyk (prstom, perom, myšou, trackball na obrazovku GUI)

[1] `http://iget2work.com`

Obr. 2.1: Názorná ukážka mobilného informačného systému

- Vstup klávesnicou (väčšinou položka v rámci GUI)
- Gestá (vstup cez kameru prípadne senzory vzdialenosti - napr. Kinect, výstup cez virtuálneho agenta či robota)
- Výstup hmatom (Braillovo písmo)
- Oči - pohľad/smer, poloha hlavy
- Tlakový senzor
- Mozgové vlny (HBI: human-brain interface, EEG)
- iné.

Zmysel multimodálnej komunikácie je v jej väčšej prirodzenosti a presnosti, keďže sleduje viacero doplňujúcich sa vstupov. Hlavnou výhodou môže však byť väčšia univerzálnosť ak systém bude chcieť využiť používateľ s nejakým druhom postihu. Napríklad:

- Zrakovo – využije hlasovú modalitu, Braillovo písmo
- Sluchovo – využije grafickú modalitu a aj grafickú interpretáciu iných výstupov (zvukov, emócie v reči, titulkovanie, meno hovoriaceho, atď)

- Telesne – hlasová modalita, snímanie pohybu očí či mozgových vĺn

Dané prostredie či pracovný proces môže tiež byť pre daný spôsob modality nevhodné. Napríklad:

- reč a veľký hluk
- displej a slnečné svetlo
- šoférovanie a displej
- zásah bezpečnostnej zložky a displej
- prednášajúci a hlasové pokyny
- a podobne.

Využitie viacerých modalít samozrejme so sebou nesie aj možnosť, že rozpoznané vstupy budú niesť protichodnú informáciu alebo budú prichádzať v rôznom čase, čo zvyšuje nároky na vyhodnotenie vstupu od používateľa. Tento rozhodovací či vyhodnocovací proces riadi modul fúzie (viď Obr. 2.2). Pri fúzii modalít môže dojsť v zásade k trom hlavným javom:

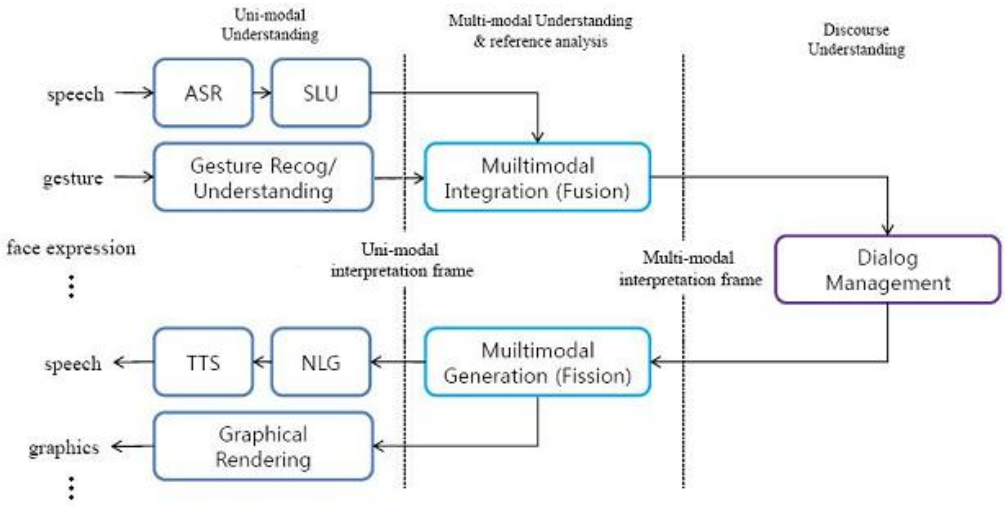
1. Mikro-časová fúzia: vstupy z rôznych modalít prebiehajú synchrónne, výsledky prídu v tesnej náväznosti. Samozrejme v prípadne protichodnej informácie je nutné riešiť vhodný váhovací či rozhodovací mechanizmus. Napríklad pri vstupe rečou a dotykom na obrazovku sa môžeme rozhodnúť že výsledok z dotykovej obrazovky bude mať prednosť pred rozpoznanou rečovou frázou.

2. Makro-časová fúzia: medzi jednotlivými vstupmi je časová medzera ale výsledok spolu súvisí. Tu nepredpokladáme že budú protichodné vstupy, očakáva sa skôr doplnenie či potvrdenie informácie.

3. Kontextová fúzia: systém zbiera informácie postupne a podľa kontextu rozpozná kam vstup patrí - vie že hľadáte letecké spojenie a tak postupne hľadá informáciu o destinácii a dátumoch hľadaných letov, prípadne ďalšie doplňujúce preferencie.

Podobne pri multimodálnom výstupe je potrebné modulom fízie rozdeliť výstupnú informáciu medzi viacero výstupných modalít aby sa spolu doplňali, prípadne informáciu vhodne duplikovali (viď Obr. 2.2[2]).

[2] `http://nlp.postech.ac.kr/research/dialog_system/mdm/`



Obr. 2.2: Architektúra typického multimodálneho systému.

Ako je vidieť na Obr. 2.2 každá vstupná modalita musí byť najskôr zachytená a rozpoznaná. Potom môže byť vykonaná sémantická interpretácia samostatne pre každú vstupnú modalitu alebo rozpoznané vstupy zo všetkých modalít môžu byť interpretované spoločne v bloku multimodálnej integrácie (fúzie). Dátová fúzia alebo multimodálna integrácia integruje informácie prichádzajúce cez niekoľko vstupných modalít na vytvorenie výslednej sémantickej správy pre správcu dialógu (DM).

Podobne môžeme fúziu rozdeliť ešte na *dátovú, parametrovú a rozhodovaciu. Dátová fúzia* sa používa na integráciu dát z rôznych senzorov pre tú istú modalitu - napríklad viackamerový systém pre rozpoznávanie gesta či pohybov tela. *Parametrová fúzia* sa používa na spojenie parametrov získaných z rôznych modalít na zvýšenie robustnosti rozhodovacieho algoritmu - napríklad parametre o pohybe pier spolu s rečovými parametrami na lepšie rozpoznanie reči v zašumenom prostredí. *Rozhodovacia fúzia* sa uplatňuje ak máme výsledky z rôznych modalít ktoré sa ale vzájomne doplňujú, teda ak napríklad gestom či dotykovým perom označíme oblasť kde hľadáme reštauráciu - na čo sa však spýtame hlasom otázkou: Nájdi mi reštaurácie v označenej oblasti.

2.2 Mobilita

Mobilita je definovaná ako vlastnosť siete/aplikácie/zariadenia umožňujúca voľný pohyb koncového účastníka.

Mobilný terminál predstavuje zariadenie, ktoré užívateľ môže premiestniť

alebo ho počas pohybu používať. Môže to byť teda mobilný telefón alebo aj Hands-free, ktoré umožňuje využívať pri komunikácii ruky na inú činnosť – napríklad gestikulovanie, klávesnica, dotykové pero, šoférovanie, atď.

Ak daná aplikácia potrebuje pripojenie na Internet alebo inú dátovú sieť, vyžaduje si mobilita aj využitie mobilného dátového prístupu. Môže byť riešený technológiou WLAN, ale ako záložný je vhodné mať aj dátové služby mobilnej telefónnej siete ako 4G, 3G, GPRS prípadne Bluetooth a podobne. V súčasnosti sa presadzujú aj vysoko-rýchlostné optické siete šírené v mimo viditeľného spektra v miestnosti.

Úplne odlišnou problematikou je mobilita aplikácie či servera.

- Pod mobilitou *aplikácie* si môžeme predstavovať schopnosť aplikácie preniesť údaje na iný terminál bez komplikovaného presunu dát. Takže napríklad na základe mena a hesla či totožnej SIM karty aplikácia rozpozná užívateľa a všetky jeho personalizované údaje prenesie z centrálneho úložiska do terminálu ak je to potrebné. Inými slovami nie je problém pre jedného užívateľa aplikáciu používať na rôznych termináloch, prípadne aj s inými operačnými systémami/platformami (Android, iOS, Linux, Windows Mobile, a podobne).

- Pod mobilitou *servera* máme na mysli schopnosť preniesť aplikačný server z nainštalovaného hardvéru na iný. Znamená to, že existuje štandardizovaný postup ako celý serverový systém nainštalovať fyzicky na iný hardvérový systém. Najjednoduchšie je nainštalovať systém na virtuálny PC, ak server nevyžaduje špecializovaný hardvér ako sú telekomunikačné karty a pripojenie do rôznych telekomunikačných sietí cez pevnú linku, modem, GSM bránu, VoIP bránu a podobne. Ďalším aspektom je schopnosť preniesť IP adresu alebo DNS meno servera tak, aby všetci klienti vždy vedeli kde danú službu kontaktovať. Pričom na rozdelenie záťaže nie je zlé mať viacero paralelných bežiacich systémov so vzájomnou synchronizáciou údajov. Už pri prvej inštalácii servera preto treba mať na mysli otázku ako bude systém schopný byť premiestnený na nový hardvér približne každé dva roky. Prípadne ako bude možné preniesť systém na novšiu verziu operačného systému, kvôli jeho bezpečnosti.

2.3 Rečový dialógový systém pre mobilné zariadenia s podporou multimodality

V súčasnosti sú možnosti mobilných zariadení omnoho širšie ako to bolo pri ich uvedení. Poskytujú vysoký výkon, rýchly bezdrôtový prenos dát, pod-

poru multimédií, kameru, mikrofón, GPS a mnohé iné technológie a služby. Vďaka tomu je možné navrhnúť aplikáciu, ktorá okrem obvyklých vstupno-výstupných modalít poskytne aj možnosť hlasového vstupu či výstupu. Samozrejme tým sa podstatne komplikuje aj riadenie dialógu, keďže môže dôjsť k oneskorenému vstupu na predošlú výzvu, či dvom protichodným vstupom z rôznych modalít.

Táto podkapitola poskytuje stručný pohľad na túto problematiku s názorným príkladom už riešeného problému v rámci projektu MOBILTEL (Mobilné multimodálne telekomunikačné systémy a služby - APVT – 20-029004).

Samozrejme v súčasnosti je možné poskytnúť aplikáciu rozpoznávania reči aj priamo v mobilnom termináli, prezentovaný projekt MOBILTEL však ponúka riešenie pri ktorom služba beží na aplikačnom serveri, kde je možné ju priebežne udržiavať a aktualizovať a klient k nej pristupuje cez bezdrôtový prenos dát, pričom samotná úloha rečového dialógového systému je plne funkčná na strane servera. Tým je zabezpečená plná kontrola poskytovateľa nad stavom služby, aktualizáciami, a vykonanie časovo náročných operácií na vlastnom hardvéri, čím sa služba stane dostupnou širšiemu okruhu používateľov.

Klient tohoto systému potrebuje iba aplikáciu na sprostredkovanie vstupných a výstupných modalít ako reč, grafický výstup, dotyk perom, klávesnicu a v budúcnosti možno aj grafický vstup cez kameru (čiarový kód, gestá, a iné), pohyb terminálom (gesto terminálom cez senzor zrýchlenia umožňuje už dnes identifikovať rôzne povely) či vstup GPS pozície (na zameranie pozície, pre ktorú chce používateľ aplikáciu použiť - najbližšia zastávka, predpoveď počasia, atď.), pričom celá interakcia spolu s pamäťovo a procesorovo náročnými úlohami beží na aplikačnom serveri.

2.3.1 Univerzálne riešenia pre mobilné terminály

Súčasné mobilné zariadenia sú nasadzované s veľkou rozmanitosťou operačných systémov, rozmerov displejov, možných vstupných zariadení a výkonov. V tomto pestrom trhu je ťažké vytvoriť univerzálnu aplikáciu.

Na základe pokusov efektívne vyriešiť prenos hlasu zo smartfónu prostredníctvom IP siete sme dospeli k záveru, že najefektívnejším riešením problému plne duplexného prenosu reči so všetkou potrebnou signalizáciou je využitie hotových VoIP (Voice over IP) knižníc, ktoré existujú už pre všetky operačné systémy mobilných terminálov a sú optimalizované na minimálne HW nároky.

VoIP je služba, ktorá zaznamenala v poslednom čase obrovský rozmach v oblasti telekomunikácií aj na Slovensku. Už je možné využiť aj komerčné VoIP služby slovenských operátorov s nízkymi mesačnými nákladmi a slo-

venským telefónnym číslom, na ktoré je možné sa dovolať ako na klasickú PSTN linku. Využitie lacných zahraničných operátorov zase umožňuje rýchle registrovanie používateľov, ktorých VoIP klienti sa budú z PDA pripájať na VoIP klienta multimodálneho komunikátora. Najefektívnejšie je samozrejme vytvoriť vlastný VoIP server, ktorý by umožnil pripájanie klientov aj bez registrácie.

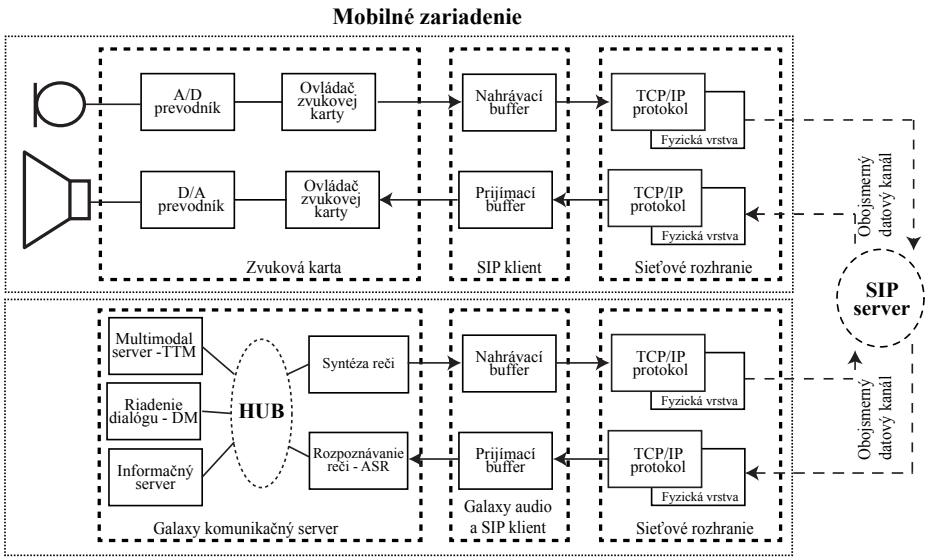
Využitie hotových VoIP štandardov umožňuje preklenúť problémy, ktoré prináša súčasný bezpečný a široko rozvrstvený internet s množstvom firewallov a prekladom IP adries (služba NAT), ktoré sa bežne používa hlavne pri bezdrôtovom pripojení WiFi. Vďaka tomu môže byť služba dostupná z ľubovoľnej WiFi siete, ktorá má internetovú konektivitu, či aspoň konektivitu k VoIP serveru, ktorý službu poskytuje.

VoIP štandardom, ktorý je v maximálnej miere kompatibilný s PDH digitálnymi hierarchiami je protokol H.323. VoIP protokoly riešia nadviazanie a ukončenie spojenia a všetku potrebnú signalizáciu. Na samotný prenos reči využívajú štandardné G.711 a GSM kodeky. Nevýhodou H.323 je to, že sa prenáša v binárnom tvare a je ťažko čitateľný či modifikovateľný, čo malo za následok, že tvorcovia voľne dostupných VoIP softvérov siahli k iným možnostiam.

Alternatívou k protokolu H.323 je protokol SIP (Session Initiation Protocol). Session Initiation Protocol je protokol vyvinutý pracovnou skupinou IETF MMUSIC (the Internet Engineering Task Force, Multiparty Multimedia Sesion Control). SIP je štandardizovaný v IETF RFC 2543 [22], publikovanom v marci 1999, aktuálna verzia 2.0 je definovaná v RCF 3261 [65].

SIP je textový signalizačný protokol na aplikačnej úrovni. Slúži na nadviazanie, modifikáciu a ukončenie multimediálneho spojenia, medzi dvoma alebo viacerými účastníkmi. Oproti H.323, je SIP podstatne odľahčený a jednoduchší. Je to vlastne iba signalizačný protokol, ktorý má za úlohu 'iniciovať spojenie - session'. Zjednodušene povedané zabezpečuje, aby sa dva terminály vzájomne našli a spojili.To je významnou odlišnosťou oproti H.323, ktorý zabezpečuje oveľa viac, napríklad aj sprostredkovávanie výberu vhodných kodekov na oboch stranách.

SIP má syntax podobný protokolom HTTP a SMTP. Na rozdiel od binárne kódovaného H.323 je založený na správach vo forme čistého textu a je transportne nezávislý (môže byť použitý na prakticky ľubovoľnom transportnom protokole). SIP môže vytvárať spojenia pre akýkoľvek multimediálny prenos. Jedná sa hlavne o prenos hlasu, ale i videa. V praxi je SIP najčastejšie využívaný pre IP telefóniu. Pri prenose multimédií sa využívajú aj iné IETF protokoly napríklad RTP (Real-time Transport Protocol). Pre vlastný transport SIP protokolu slúžia transportné protokoly UDP (User Datagram Protocol), TCP (Transmission Control Protocol) a TLS (Transport Layer



Obr. 2.3: Bloková schéma riešenia prenosu reči s využitím SIP telefónnych štandardov

Security). Pre kódovanie správ sa využíva UTF-8 kódovanie.

Na Obr. 2.3 je možné vidieť príklad využitia prenosu zvuku s využitím SIP komunikačného klienta na oboch stranách komunikačného reťazca. Hovor inicializuje mobilné zariadenie a s využitím TCP/IP sa pripojí na SIP server, ktorý uskutoční spojenie na multimodálny komunikačný server, napríklad na báze Galaxy komunikátora [72]. Tento obsahuje SIP klienta implementovaného do Audioservera, ktorý inicializuje interakciu cez dialógový manažér (DM). DM cez blok syntézy reči (TTS) odosiela hlasové správy používateľovi a naopak cez blok rozpoznávania reči (ASR) sa snaží interpretovať hlasové pokyny od používateľa. Ide o zjednodušenú schému a ďalšie bloky budú popísané pri modelovej realizácii multimodálneho systému.

Z obrázka je tiež vidieť, že je potrebné vyriešiť tri podstatné problémy pri implementácii tohoto riešenia:

1. *SIP klient na strane mobilného zariadenia* - ktorý bude v takej forme, že bude možné ho spustiť synchrónne s aplikáciou poskytujúcou ostatné modality (napr. webstránka s aktívnymi prvkami JavaScript, Flash a iné).

2. *SIP server* - ktorý by umožnil jednoduché pripojovanie klientov, prípadne aj bez registrácie a umožňoval tak volať každému terminálu, ktorý si službu spustí bez nejakých obmedzení.

3. *SIP klient na strane Galaxy* architektúry s vyriešením komunikácie s HUB prvkom Galaxy architektúry (detekcia zdvihnutia, položenia hovoru a inicializácie audio kanálov k serverom TTS a ASR).

Najmodernejším trendom je využitie Flash technológie, ktorá je už bežnou súčasťou výbavy webových prehliadačov vo všetkých operačných systémoch a umožňuje prístup webovej aplikácie k mikrofónu, reproduktoru aj kamere daného zariadenia, ak to používateľ povolí. Tým by odpadla nutnosť vytvárania SIP klientov pre rôzne operačné systémy.

Veľmi dôležitým prvkom multimodálneho dialógového systému je komponent zabezpečujúci komunikáciu s inými modalitami, teda jednak prezentáciu informácie od servera, ale zároveň aj oveľa ťažšiu úlohu spájania t.j. fúzie vstupov z rôznych modalít. Systém totiž reaguje povedzme na hlas, ale aj vstup dotykovým perom či klávesnicou. Tento komponent pracuje so všetkými vstupmi, ktoré idú od všetkých modalít (aj reči - teda ASR) a musí rozhodnúť, ktorý z nich, resp. ich kombináciu pošle jednotke dialógového manažéra. Takisto sprostredkováva prezentáciu informácií z dialógového manažéra dostupnými modalitami (reč, obraz, text a iné).

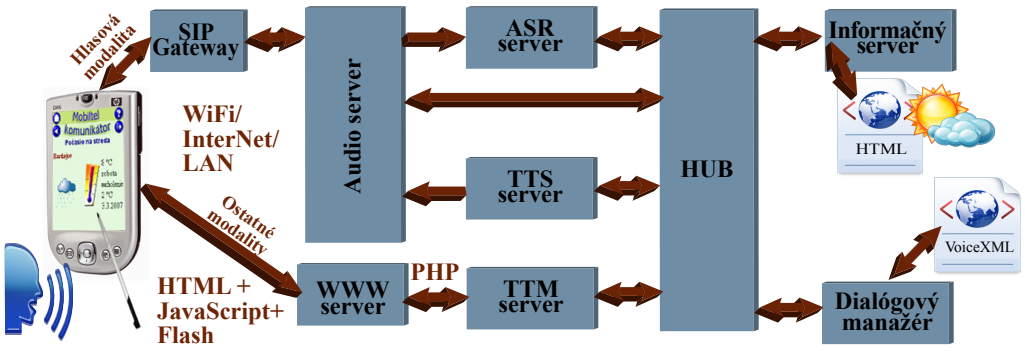
Pre grafickú modalitu je najvýhodnejšie použiť súčasné web technológie a teda vytvoriť interaktívnu webstránku, ktorá bude reagovať na stav dialógu a umožní multimodálnu interakciu.

Toto riešenie umožňuje skombinovať viacero modalít na ľubovoľnom mobilnom zariadení, ktoré umožňuje použitie SIP klienta (či už špeciálne vyvinutého pre danú aplikáciu alebo iného dostupného pre danú platformu) s prístupom na web (cez webový prehliadač). V minulosti niektoré prehliadače nepodporovali aktívne prvky. Dnes však už s ich podporou nebýva problém a predinštalované prehliadače sú dostatočne vybavené.

2.3.2 Projekt MOBILTEL

Táto podkapitola popisuje staršiu pilotnú aplikáciu, ktorá bola vytvorená v rámci projektu MOBILTEL (Mobilné multimodálne telekomunikačné systémy a služby - kód projektu: APVT – 20-029004) [55, 60, 90], ktorý bol v rokoch 2005 až 2007 podporovaný Agentúrou na podporu vedy a techniky (v čase riešenia projektu transformovaná na Agentúru na podporu výskumu a vývoja). Tento projekt bol v rámci hodnotenia zaradený medzi projekty s najlepšími výsledkami riešenými v tomto období (`mobiltel.tuke.sk`).

Hlavné úlohy projektu boli implementácia jednotlivých modalít do mobilných multimediálnych zariadení, fúzia vstupných modalít, vypracovanie pilotnej (prezentačnej) multimodálnej aplikácie a jej otestovanie v mobilnom termináli využívajúcom bezdrôtové internetové pripojenie.



Obr. 2.4: Bloková schéma projektu MOBILTEL

Architektúra systému vychádzala z projektu IRKR - Inteligentné rečové komunikačné rozhranie [35], ktorý bol podporovaný Ministerstvom školstva SR (júl 2003 až jún 2006), a ktorého výsledkom bol rečový komunikačný systém na báze Galaxy architektúry [72]. Vyvinutý systém umožňuje používateľom rečou cez telefonické spojenie (PSTN, GSM, VoIP: H.323 a Skype) získavať informácie o vlakovom spojení a počasí na Slovensku. Tento systém je naďalej vyvíjaný (vrátane služieb) a je stále dostupný na telefonických číslach zverejnených na webstránke projektu `irkr.tuke.sk`.

Na Obr. 2.4 je možné vidieť výslednú architektúru systému. Zobrazuje jednak klasické prvky ako centrálny Galaxy HUB proces, dialógový manažér, modul syntézy reči, modul rozpoznávania reči, informačný server a zvukový/audio server, ktorý bol postavený hlavne na spoluprácu s telefónnou kartou Intel Dialogic.

Ďalej je možné vidieť nové prvky systému a to SIP Gateway (v čase riešenia projektu nebol vybudovaný nový Galaxy Audioserver na báze SIP klienta), TTM server (Text to Multi-Modality - sprostredkuje prenos správ do webservera a naopak, ale zároveň zabezpečuje fúziu vstupných modalít z rečového a textového vstupu), WWW server (zabezpečuje všetky ostatné modality, okrem rečovej).

V mobilnom zariadení je ešte naprogramovaný SIP klient, ktorý bol v čase projektu vyvinutý len pre operačný systém Windows Mobile, takže časom stratil na aktuálnosti. Tento prvok by bolo vhodné v budúcnosti nahradiť univerzálnejším riešením na báze webaplikácií (napr. Flash SIP klient).

Server pre podporu multimodality - TTM

Server TTM (Text to Multi-Modality) sleduje všetky správy, ktoré vyžadujú zmenu grafického rozhrania či prechádzajúce vstupy a naopak posiela nové

vstupy z rôznych modalít do dialógového manažéra. Jeho úlohou je umožniť externý vstup do pôvodne hlasovej komunikácie a zabezpečiť sledovanie ľubovoľnej komunikácie a stavu dialógu.

Tento server je tiež dôležitým rozhraním medzi Galaxy architektúrou a grafickým rozhraním, ktoré je generované a vytvorené vo forme webaplikácie. Prenos správ je zabezpečený jednoducho - cez spoločné súbory, ktoré sú unikátne pre danú interakciu a slúžia ako zásobníky správ. Webaplikácia pomocou PHP skriptov tieto súbory sleduje a prípadne ich pri vstupe z grafického rozhrania generuje.

Multimodálny server TTM tiež umožňuje zmiešanú interakciu - teda je možné časť informácií zadávať hlasom, perom na dotykovej obrazovke, či prostredníctvom klávesnice, pričom vyššiu prioritu majú informácie získané z grafického rozhrania (GUI). Dôvodom je možné oneskorenie výstupu systému rozpoznávania reči po zadaní príkazu pomocou dotykovej obrazovky. Server musí sledovať a zbierať všetky dostupné vstupy a iba on posiela výsledok dialógovému manažérovi.

Narozdiel od pôvodnej architektúry hlasovej komunikácie (na báze VoiceXML) server TTM odchytáva správy smerované z Hub procesu do modulu syntézy reči - tieto zapisuje do súborov, z ktorých potom PHP skripty extrahujú informáciu o tom, v ktorej fáze sa dialóg nachádza. Táto informácia slúži k zmene aktuálneho zobrazenia na obrazovke. Podobne odchytáva rozpoznané vyjadrenia používateľa z rozpoznávača reči a zapisuje ich do súborov, pričom ak príde aj vstup z inej modality rozhodne, ktorú z nich (alebo ich kombináciu) pošle do dialógového manažéra (fúzia modalít).

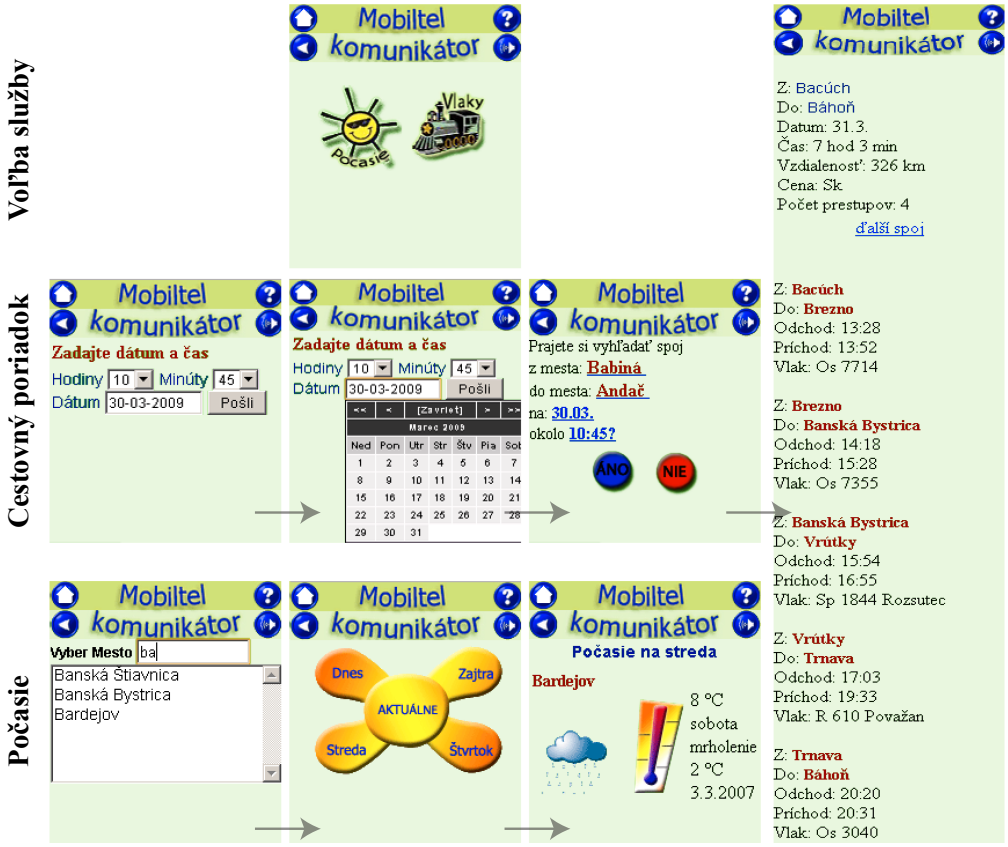
Grafické rozhranie multimodálnej služby - GUI

Pre konkrétnu službu či dialóg je potrebné navrhnúť grafickú reprezentáciu/rozhranie s možnosťou vkladať/zvoliť čo najjednoduchšie a čo najinteraktívnejšie prvky, ktoré používateľ v danej chvíli potrebuje, viď Obr. 2.5.

Vytvoriť synchrónne grafické rozhranie k hlasovému dialógu je však neľahkou úlohou. Každý stav dialógu musí mať svoju jednoznačnú identifikáciu, aby sa nedostalo grafické rozhranie do iného stavu ako je hlasové a prípadné vstupy by zmiatli dialógový manažér.

Návrh grafického rozhrania (grafickej modality) má za úlohu urýchliť hlasovú interakciu tým, že bude možné kliknutím okamžite vybrať z ponúkaných možností. Na druhej strane má za úlohu čo najprehľadnejšie prezentovať získané informácie. Pri hlasovej komunikácii je totiž najväčším problémom čakanie na ukončenie otázky či získanej informácie syntetizátorom reči, a hlavne potvrdzovanie správnosti rozpoznaných údajov.

Využitím JavaScriptu boli maximálne zužitkované možnosti webaplikácií



Obr. 2.5: Príklad grafického rozhrania multimodálnych služieb „Počasie“ a „Cestovný poriadok“.

tej doby. Tento je využitý pri filtrovaní miest podľa už zadaných písmen (výhodné u terminálov s plnohodnotnou klávesnicou - na jednom bol systém testovaný), a takisto pri voľbe dňa služby „Cestovný poriadok“ pomocou automaticky zobrazeného kalendára.

Grafické rozhranie obsahuje tlačidlá, ktoré sú stále prítomné a umožňujú zvoliť kľúčové slová ako pomôž mi, domov alebo späť. Takisto obsahuje logo služby a možnosť vypnúť reproduktor, ak sa človek dostane do prostredia, kde sú mu hlasové výzvy komunikátora príťažou.

3

Parametrizácia rečových a audio signálov

V tejto kapitole sú rozoberané rôzne druhy metód parametrizácie reči, ktorá sa využívajú v systémoch rozpoznávania spojitej reči (CSR - Continuous Speech Recognition) v tichom a zašumenom prostredí. Hlavným cieľom parametrizácie reči je extrakcia príznakov reči zo vstupného rečového signálu, kde tieto príznaky reči sú najčastejšie výsledné kepstrálne koeficienty, ktoré by mali čo najlepšie charakterizovať vstupný rečový signál.

3.1 Predspracovanie

Vstupný rečový signál je v analógovom tvare a pred ďalším spracovaním sa najprv musí analógovo predspracovať a to zosilnením pred mikrofónom a zosilnením pre analógovo/digitálnym prevodníkom (A/D prevodník). Následne je tento zosilnený analógový rečový signál prevedený z analógového do digitálneho tvaru pomocou A/D prevodníka.

Následne sa na tomto digitálnom rečovom signály vykonáva číslicové predspracovanie, ktoré sa skladá z troch krokoch a to preemfázy, segmentácie a váhovania oknovou funkciou.

3.1.1 Preemfáza

Pri preemfáze dochádza k filtrovaniu digitálneho rečového signálu pomocou filtra s konečnou impulzovou odpoveďou (FIR filter), kde pri preemfáze dochádza k zvýrazneniu vyšších frekvencii, pretože veľká časť celkovej energie rečového signálu je vo frekvenčnej oblasti pod hodnotou 300Hz, avšak dôležité informácie pre spracovávanie reči sa nachádzajú nad touto hodnotou

[39].

3.1.2 Segmentácia

Pri segmentácii dochádza k rozdeleniu tohto digitálneho rečového signálu na rámce, ktorých dĺžka sa zvyčajne volí medzi 20-30ms. Na tomto krátkom úseku reči je možné predpokladať, že parametre rečového signálu sú v jednom rámci nemenné, a to vďaka tomu, že rýchlosť zmeny vokálneho traktu človeka je obyčajne dlhšia ako dĺžka rámca reči. Pri segmentácii sa taktiež volí hodnota prekrývania, ktorá sa obyčajne volí na hodnotu 10ms.

3.1.3 Váhovanie oknovou funkciou

Pri váhovaní oknovou funkciou je každý rámec reči vynásobený vybranou oknovou funkciou, kde najčastejšie používanou oknovou funkciou je Hammingová oknová funkcia, ktorá potláča váhu vzoriek na okraji rámcov a najväčšiu váhu dáva vzorkám v strede rámca.

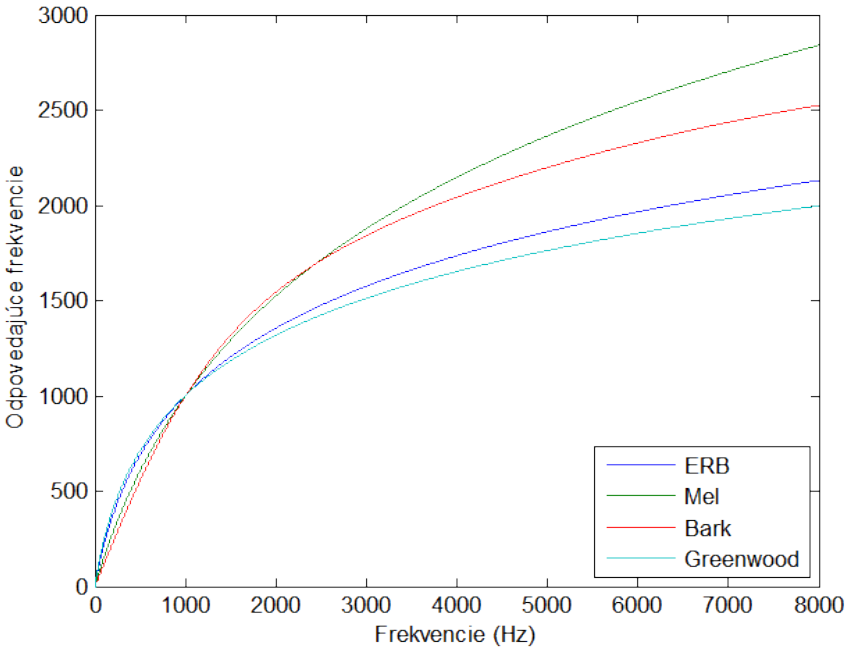
Po číslicovom predspracovaní rečového signálu je následne tento signál transformovaný do spektrálnej oblasti pomocou Diskrétnej Fourierovej Transformácie (DFT).

Predchádzajúci postup výpočtu až do DFT majú všetky parametrizácie zhodný. Navzájom sa od seba líšia až spracovaním v spektrálnej oblasti. Z toho vyplýva, že úspešnosť potlačenia nežiaducich vplyvom, ktoré sa v nahrávkach reči nachádzajú je daná práve spracovaním rečového signálu v spektrálnej oblasti.

3.2 Spracovanie rečového signálu v spektrálnej oblasti

V parametrizáciach reči dochádza v spektrálnej oblasti najprv k filtrovaniu rečového signálu pomocou banky filtrov. Pri filtrácii pomocou banky filtrov je širokopásmový rečový signál rozdelený do frekvenčných subpásiem a pri ďalšom spracovaní sa odhaduje výkon rečového signálu v každom subpásme. Filtre v banke filtrov majú rovnakú šírku pásma a sú lineárne rozložené v stredných frekvenciách na zvolenej frekvenčnej stupnici (Melová, Barková, ERB - equivalent rectangular bandwidth a Greenwood) a zároveň nelineárne rozmiestené na Hertzovej stupnici. Na Obr. 3.1 je znázornená závislosť rozloženia frekvencií Melovej, Barkovej, ERB a Greenwood stupnici na Hertzovej stupnici. Ako vidieť tak všetko sú to logaritmické závislosti, čo je dané tým, že ľudské ucho počuje rozloženie frekvencii skôr logaritmicky ako lineárne.

Zvyčajne sa volí 24 filtrov v banke filtrov, ale v prípade Gammatone banky filtrov je použitých 40 filtrov [62, 34, 98, 14].



Obr. 3.1: Závislosť Melovej, Barkovej, ERB a Greenwood stupnice na Hertzovej stupnici

Po filtrovaní pomocou banky filtrov sa používa v parametrizácia reči buď jedna alebo kombinácia viacerých analýz. Týmito analýzami sú:

- Lineárna Predikčná analýza
- Percepčná Lineárna Predikčná analýza
- Metóda RASTA
- MVDR analýza
- Analýza s odčítaním výkonu šumu
- TRAP analýza
- Fepstrálna analýza

3.2.1 Lineárna predikčná analýza

V Lineárnej Predikčnej analýze dochádza k odhadu n-tej vzorky rečového signálu na základe predikcie predchádzajúcich P vzoriek a budiacej postupnosti. Jej výsledkom sú LP koeficienty a predikčná chyba [39, 34].

3.2.2 Percepčná Lineárna Predikčná analýza

Percepčná Lineárna Predikčná analýza využíva prispôsobenie magnitúdy (či amplitúdy) spektra pomocou kriviek rovnakej hlasitosti (equal loudness curves), vytvorených na základe testov vnímania - percepcie danej frekvencie, a následne dochádza ku kompresii amplitúdy spektra pomocou umocnenia na 1/3 [34, 24].

3.2.3 RASTA metóda

Metóda RASTA využíva lineárnu pásmovú filtráciu pomocou filtra s nekonečnou impulzovou odpoveďou (IIR filtra), ktorý potláča konvolučný šum a rýchle spektrálne zmeny rečového signálu [34, 25].

3.2.4 MVDR analýza

MVDR analýza využíva MVDR spektrum, ktoré predstavuje vyššiu spektrálnu obálku rečového signálu, ktorá je menej citlivá na rýchle zmeny rečového signálu a vypočíta sa z MVDR koeficientov pomocou LP koeficientov a predikčnej chyby, ktoré sa získajú z Lineárnej Predikčnej analýzy v časovej oblasti [95].

3.2.5 Analýza s odčítaním výkonu šumu

Analýza s odčítaním výkonu sumu je založená na skutočnosti, že výkon reči v každom kanáli sa mení oveľa rýchlejšie ako výkon šumu na pozadí v rovnakom kanáli. Preto sa pri výpočte výkonu šumu používa dlhší časový rámec ako pri analýze rečového signálu a jeho dĺžka je 50-120ms. Výkon v tomto dlhšom rámci sa nazýva strednodobý výkon a vypočíta sa z krátkodobého výkonu, ktorý je získaný z každého pôvodného rámca. Po vypočítaní strednodobého výkonu sa vykonáva asymetrické potlačenie šumu (ANS) s časovým maskovaním. Po ANS s časovým maskovaním ešte dochádza k vyhladenie spektra, výkonovej normalizácii a kompresii amplitúdy spektra umocnením na 1/15 [40, 41].

3.2.6 TRAP analýza

V TRAP analýze sa využívajú časové intervaly s dĺžkou 1s, čo predstavuje 101 rámcov, kde prostredný rámec reprezentuje krátkodobé príznaky aktuálneho fonému a ostatných 50 rámcov z každej strany reprezentuje jeho kontextovú závislosť. Základnou vlastnosťou TRAP analýzy je, že využíva mapovanie do tried pomocou viacvrstvovej umelej neurónovej siete (MLP) , ktorej výsledkom sú TRAP príznaky reči [26, 27, 19, 52].

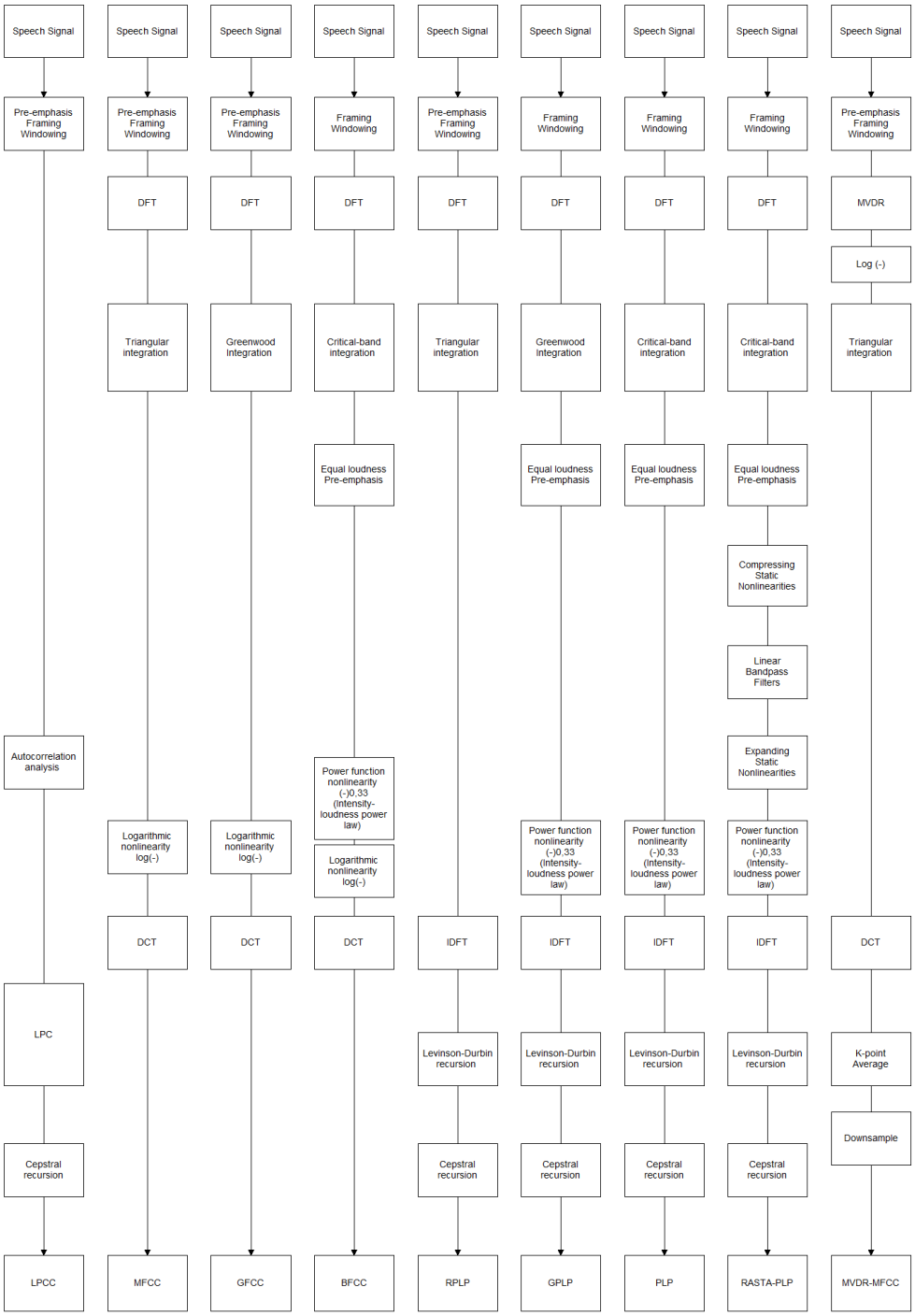
3.2.7 Fepstrálna analýza

Základnou vlastnosťou Fepstrálnej analýzy je, že v jadre výpočtu využíva obálku AM (amplitúdovo modulovaného) rečového signálu v časovej oblasti. Fepstrálna analýza je odlišná oproti ostaným analýzam v tom, že používa 100ms rámce oproti 25,6ms rámcom, ktoré sú bežne používane v metódach parametrizácie reči. Ďalšou odlišnosťou je, že využíva pravouhlú oknovú funkciu namiesto Hammingovej oknovej funkcie. Obálka AM rečového signálu v časovej oblasti sa získa logaritmovaním absolútnych hodnôt filtrovaného rečového signálu po IDFT. Následne je tento AM rečový signál filtrovaný filtrom s pohyblivým priemerom a ešte podvzorkovaný s faktorom 40. Ďalej dochádza k transformácii pomocou DCT, kde sa necháva iba prvých 5 koeficientov s nižším rádom. Následne dochádza k zreťazeniu týchto 5 koeficientov z každého pásma čím vznikne 120 rozmerný sektor. Tento 120 rozmerný vektor ďalej vstupuje do PCA transformácie, ktorej výsledkom sú už výsledne FEPstrálne príznaky reči [91, 89].

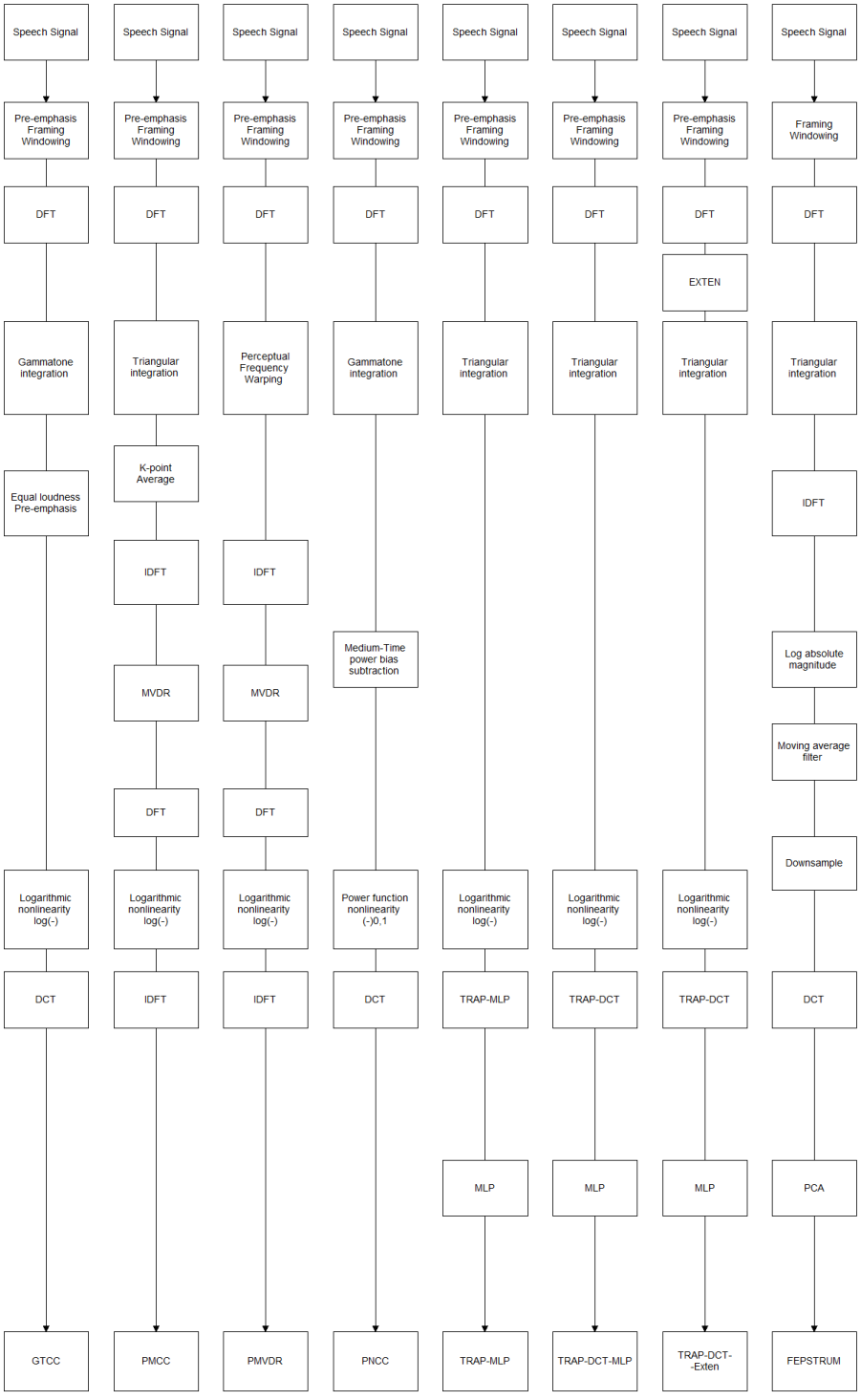
3.2.8 Zhrnutie

Po jednej alebo kombinácii viacerých analýz dochádza k poslednému kroku výpočtu výsledných krátkodobých príznakov reči a to buď IDFT, Levinsonov-Durbinov iteračný algoritmus a kepstrálny rekurzívny vzťah alebo sa použije iba DCT. Pred vykonaním transformácie sa v parametrizáciách, ktoré vo svojom poslednom kroku nevyužívajú kompresiu amplitúdy spektra, najprv používa logaritmovanie pre obmedzenie dynamiky signálu pomocou zmeny amplitúdy spektra. Výsledkom je už 13 základných výsledných koeficientov, z ktorých sa následne vypočítavajú diferenčné a akceleračné koeficienty. Zreťazením základných, diferenčných a akceleračných koeficientov sa získa všetkých 39 výsledných koeficientov.

Na Obr. 3.2 a 3.3 je znázornený vývojový diagram parametrizácií reči, ktoré sú podrobnejšie rozoberané v bakalárskej práci [18]. Na tomto vývojovom diagrame vidieť, ktoré bloky výpočtu majú jednotlivé parametrizácie



Obr. 3.2: Postup výpočtu jednotlivých parametrizácií - časť 1. [18]



Obr. 3.3: Postup výpočtu jednotlivých parametrizácií - časť 2. [18]

spoločné a v ktorých sa naopak líšia. Názvy jednotlivých blokov sú uvádzané v anglickom jazyku a popisujú čo sa v konkrétnom bloku presne vykonáva. Taktiež je ešte dôležité povedať, ktoré parametrizácie využívajú ktorú z hore uvedených analýz:

- Lineárna Predikčná (LPC) analýza: LPCC, MVDR-MFCC, PMCC, PMVDR
- Percepčná Lineárna Predikčná (PLP) analýza: PLP, BFCC, GPLP, RASTA-PLP a iné
- Metóda RASTA: RASTA-PLP
- MVDR analýza: MVDR-MFCC, PMCC, PMVDR
- Analýza s odčítaním výkonu šumu: PNCC
- TRAP analýza: TRAP, TRAP-DCT, TRAP-EXTEN
- Fepstrálna analýza: FEPSTRUM

Parametrizácie RPLP, GFCC sú založené na MFCC parametrizácii, ktorá nevyužíva žiadnu z opísaných analýz ale po filtrovaní pomocou Melovej banky filtrov dochádza už iba k logaritmovaniu a samotnému výpočtu výsledných koeficientov pomocou DCT.

3.2.9 Matlab demo

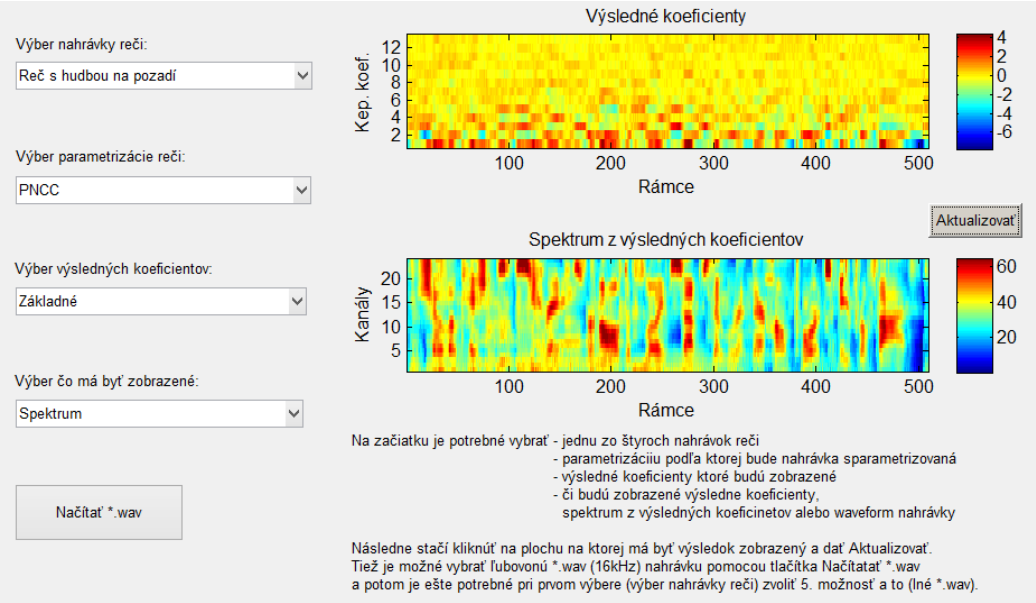
Pre lepšie grafické porovnanie vplyvu vstupného šumu na výsledné koeficienty pri využití rôznych parametrizácii reči boli v programovom prostredí Matlab realizované hore spomenuté parametrizácie a ku nim bolo vytvorené prehľadné užívateľské rozhranie - GUI, ktoré je zobrazené na Obr. 3.4[1].

V tomto GUI je potrebné najprv vybrať jednu zo štyroch predefinovaných nahrávok reči. Ide stále o tú istú nahrávku reči, ale každá je inak zašumená. Prvá nahrávka je nahraná v tichom prostredí, druhá obsahuje biely šum na pozadí, tretia hrajúcu hudbu na pozadí a štvrtá zvuky z ulice na pozadí.

Ďalej je potrebné vybrať jednu z dvanástich implementovaných parametrizácii reči:

- LPCC (Linear Prediction Cepstral Coefficients)
- PLP (Perceptual Linear Prediction)

[1] `http://lrmt.kemt.fei.tuke.sk/ITSS/sk/recove-technologie-lrmt/matlab_GUI%20(2).zip`



Obr. 3.4: MATLAB rozhranie pre demonštráciu rôznych parametrizácií [18]

- RASTA - PLP
- MFCC (Mel-frequency Cepstral Coefficients)
- RPLP (Revised Perceptual Linear Prediction)
- BFCC (Bark Frequency Cepstral Coefficients)
- MVDR-MFCC (MVDR-based MFCC)
- GTCC (Gammatone Cepstral Coefficients)
- GFCC (Greenwood Function Cepstral Coefficients)
- GPLP (Generalized Perceptual Linear Prediction)
- PNCC (Power-Normalized Cepstral Coefficients)
- FEPSTRUM

Následne sa vyberie, ktoré výsledné koeficienty budú vypočítané a zobrazené. Či to bude 13 základných, 13 diferenčných, 13 akceleračných alebo všetkých 39 výsledných koeficientov.

Nakoniec je ešte potrebné vybrať, čo bude zobrazené. Či to budú samotné výsledné koeficienty, alebo spektrum z výsledných koeficientov, alebo

waveform zvolenej nahrávky reči. Je tu aj možnosť načítania ľubovoľnej inej .wav nahrávky pomocou tlačidla (Načítať .wav) avšak musí to byť iba .wav nahrávka so vzorkovacou frekvenciou 16kHz.

Na Obr. 3.4 je vykreslený aj príklad použitia, kde na hornej ploche je zobrazených 13 základných kepstrálnych koeficientov nahrávky reči s tichom na pozadí, ktorá bola sparametrizovaná pomocou PNCC parametrizácie a na spodnej ploche je zobrazené zasa spektrum z týchto koeficientov.

4

Akustické modelovanie pre automatické rozpoznávanie reči

Automatické rozpoznávanie reči je založené okrem iných znalostných zdrojov aj na akustickom modelovaní, ktoré vyjadruje vzťah medzi zvukovou realizáciou slova a samotného textového prepisu. Teda na to, aby bolo možné priradiť textovú podobu slova k jeho zvukovej realizácii je potrebný akustický model. Najjednoduchším spôsobom ako túto reprezentáciu vyjadriť je pomocou samotného zvukového vzoru *daného slova. Napríklad je možné použiť skutočnú nahrávku slova a priradiť k nej jeho textovú podobu. V tomto prípade sa jedná o systém rozpoznávania reči so* vzormi*, ktorý sa hlavne využíva pre systémy na rozpoznávanie izolovaných slov. Pre viac všeobecnejší popis vzťahu textovej a zvukovej podoby slova sa používajú štatistické metódy modelovania založené na skrytých Markovových modeloch (HMM - Hidden Markov Model). Narozdiel od predchádzajúceho spôsobu umožňujú lepšie modelovať variabilitu zvukovej reprezentácie a je možné pomocou nich vytvoriť systém na rozpoznávanie spojitej reči.*

4.1 Princíp akustického modelovania

Cieľom akustického modelovania je teda určiť mieru podobnosti medzi akustickým modelom slova priradeného textovej reprezentácii a vstupnou zvukovou reprezentáciou. V prípade vzorov je to podobnosť vzorovej nahrávky slova a vstupnej nahrávky, ktorú chceme rozpoznať. Ak uvažujeme so štatistickým prístupom je to zase pravdepodobnosť s akou by daný akustický model slova vygeneroval zvukovú reprezentáciu slova vo vstupnej nahrávke. Akustický model si je možné predstaviť aj ako skrinku, do ktorej vstupuje textová reprezentácia slova a zvuková realizácia pričom výstupom je ohodnotenie

podobnosti týchto vstupov. Rozpoznávanie reči je potom možné realizovať pomocou výpočtu podobnosti vstupnej nahrávky so všetkými akustickými modelmi slov a vybrať najpodobnejšie slovo.

Celý postup rozpoznávania reči s akustickým modelom si je možné predstaviť ako reťazec operácií. Prvou z nich je samotná tvorba zvukovej reprezentácie človekom, nasleduje prenosový kanál, teda prostredie v ktorom sa zvuk šíri a nakoniec je to zariadenie, ktorým sa zvuk sníma, teda mikrofón. Nasleduje výpočet podobnosti všetkých modelov slov a vstupného slova a výber najpodobnejšieho. Ideálny akustický model by mal zahŕňať všetky variability, ktoré sa v tomto reťazci nachádzajú:

- **Prízvuk**: Rečníci hovoria nehovoria v materskom jazyku pričom nevyslovujú všetky slová správne alebo hovoria nárečím.
- **Kvalita hlasu**: Každý rečník ma rôzny hlas, pričom variabilita zahŕňa chrapľavosť alebo koktanie.
- **Individuálne vplyvy**: Závisia od osobnosti rečníka, jeho slovnej zásoby, individuálne kladenie dôrazu vo vete, dokonca sám rečník nevysloví rovnaké slovo rovnako.
- **Kvalita kanála**: Šum v pozadí stacionárny alebo nestacionárny, ktorý je krátkodobý (buchnutie dverí, zvuk auta, ventilátor), odrazy miestnosti, v ktorej sa zvuk šíri.
- **Mikrofón**: Typ mikrofónu a jeho umiestnenie, napríklad nahlavným mikrofónom dochádza ku snímaniu aj mimorečových zvukov, ktoré človek vydáva.
- **Rýchlosť prejavu**: Akustické vlastnosti vysloveného slova sa menia s rýchlosťou vyslovenia.

Ďalej na akustické modelovanie je potrebné vyriešiť niekoľko ďalších nezodpovedaných otázok:

- **Trénovanie?**: Akým spôsobom je možné získať vhodné vzory alebo štatistické informácie na modelovanie slov.
- **Evaluácia?**: Akým spôsobom je možné vypočítať podobnosť textovej a zvukovej reprezentácie slov.
- **Dekódovanie?**: Akým spôsobom je možné získať presné časové hranice (časové zarovnanie) slova.

Rôzny typ akustického modelovania bude vyžadovať rôzny typ trénovacieho procesu, evaluácie a dekódovania. Pod pojmom *dekódovanie* budeme neskôr rozumieť aj spôsob využitia dynamického programovania na uľahčenie rozpoznávania reči spájaním modelov rôznych slov.

4.2 Porovnávanie so vzormi

Akustický model je v tomto prípade jednoduchý. *Trénovací* proces vlastne pozostáva v zozbieraní nahrávok slov, ktoré majú byť zaradené do rozpoznávacieho slovníka, teda, ktorým výsledný systém má rozumieť. V prípade jednoduchého systému je možné mať pre každé slovo jeden vzor, ale na zvýšenie úspešnosti systému je možné zahrnúť variabilitu rečníka a prostredia použitím viacerých vzorov vytvorených za rôznych podmienok.

Evaluácia akustického modelu predstavuje pri tomto type modelovania výpočet vzdialenosti medzi vzorom slova a vstupnou nahrávkou. Táto vzdialenosť sa chápe ako súčet vzdialenosti jednotlivých vzoriek signálov, prípadne príznakových vektorov. Keďže jedno slovo vyslovené aj tým istým rečníkom má zakaždým rôznu dĺžku, nie je možné použiť jednoduchú lineárnu metódu, kde odčítame jednotlivé vzorky signálov a sčítame tieto rozdiely. Prvým krokom je najprv tieto signály časovo zarovnať, musíme vedieť ktorú vzorku vstupného signálu s ktorou vzorkou vzoru porovnávať. Na tento účel je vhodná metóda dynamickej transformácie časovej osi (DTW - Dynamic Time Warping).

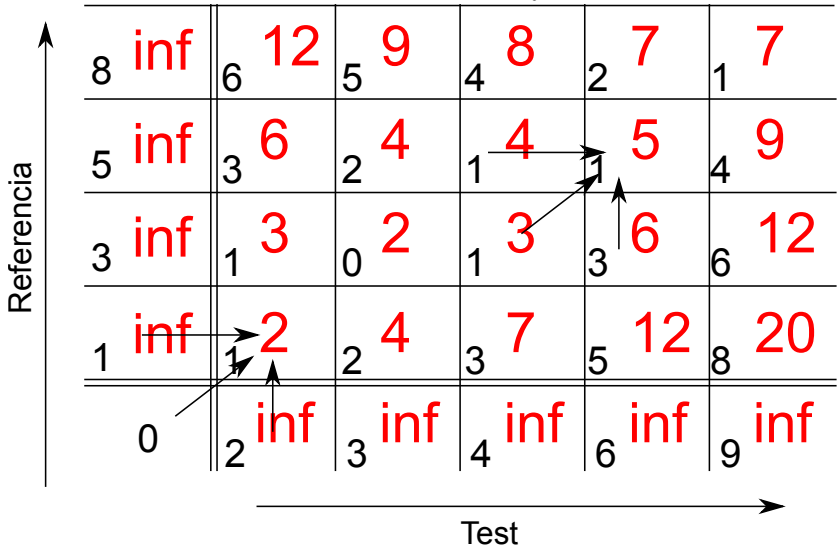
Dekódovaním môžeme rozumieť v tomto prípade hľadanie samotného časového zarovnania vzoru a vstupnej nahrávky. V prípade DTW je to hľadanie najkratšej vzdialenosti.

4.2.1 Dynamická transformácia časovej osi (DTW)

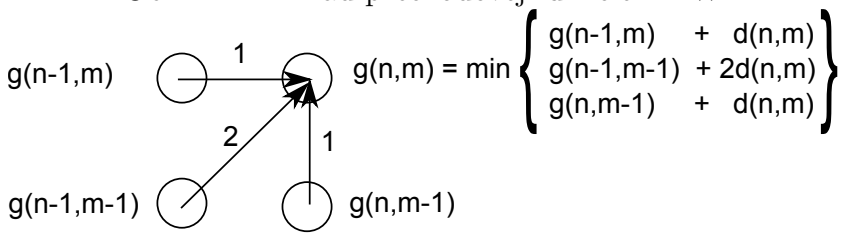
Najjednoduchšia realizácia metódy DTW spočíva vo vytvorení matice, kde na osi Y sa nachádzajú vzorky vzorovej nahrávky slova a na osi X vzorky vstupného slova. Ďalšie prvky matice predstavujú vzdialenosť všetkých vzoriek vzorového slova a vstupného slova. Ak sa jedná o vzorky signálu, v tom jednoduchom prípade to môžu byť rozdiely v ich amplitúdach. Teda prvým krokom je vytvorenie *matice vzdialeností* (d) jednotlivých vzoriek (viď Obr. 4.1 označené čiernou farbou).

Druhým krokom je použitie prechodovej funkcie (viď Obr. 4.2) pomocou ktorej vypočítame prvky tzv. *akumulačnej matice* (g) (viď Obr.4.1 označené červenou farbou). Teda predchádzajúcej hodnote v čase pripočítame váhu

Obr. 4.1: Matica metódy DTW



Obr. 4.2: Príklad prechodovej funkcie DTW



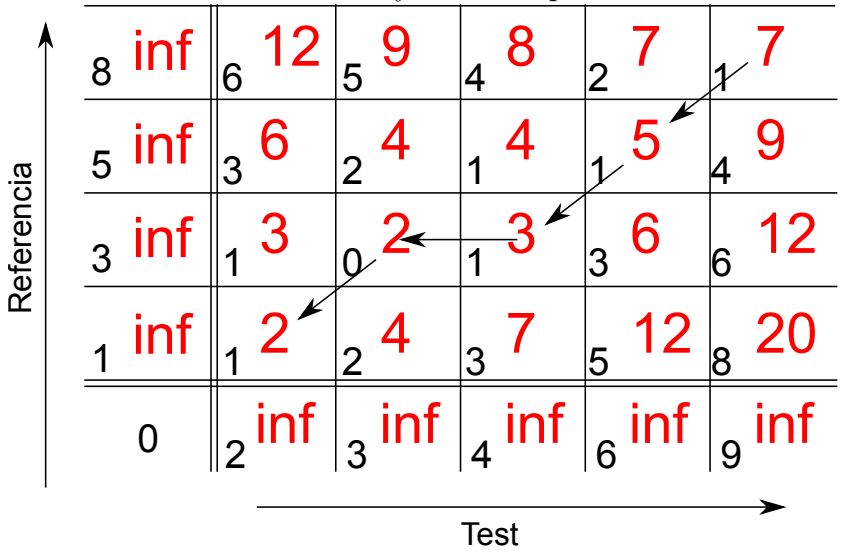
prechodovej funkcie a hodnotu vzdialenosti. Po výbere minima zapíšeme výsledok na príslušnú pozíciu v matici.

Výsledná vzdialenosť medzi vstupnými nahrávkami je určená hodnotou vpravo hore, teda poslednou hodnotou, ktorá predstavuje naakumulovanú vzdialenosť a tiež zodpovedá bodu, kde obidve nahrávky končia. Táto vzdialenosť sa normalizuje pomocou dĺžky cesty, z ktorej táto hodnota vznikla (4.1).

$$vzdialenosť = \frac{1}{N} g(T, R), \tag{4.1}$$

kde N - dĺžka cesty, z ktorej daná akumulovaná vzdialenosť vznikla. Niekedy sa nahradzuje sčítaním dĺžok jednotlivých vstupných slov (T a R), keďže na nájdenie najkratšej cesty je nutné spätne postupovať podľa toho, ktorý prechod mal minimálnu hodnotu v prechodovej funkcii (viď Obr. 4.3).

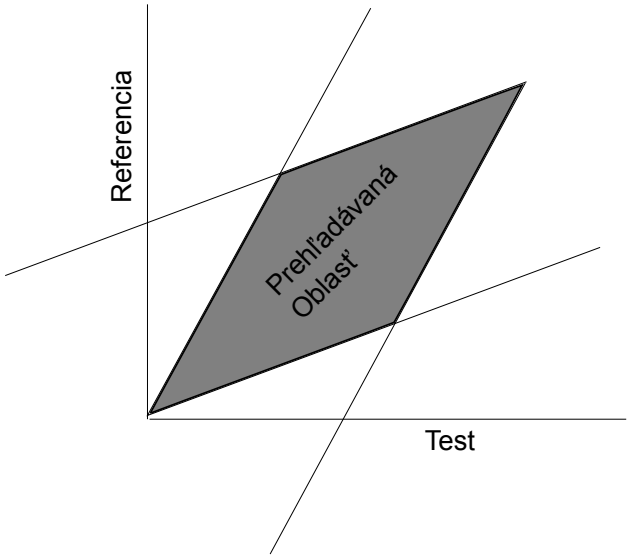
Obr. 4.3: Matica metódy DTW - Spätné trasovanie



4.2.2 Rozpoznávanie reči s DTW

Pomocou DTW (evaluáciou) vieme vypočítať jednotlivé vzdialenosti medzi signálmi. Jeden zo vstupných signálov vyjadruje zvukovú reprezentáciu textového prepisu a ďalší predstavuje slovo, ktoré je potrebné rozpoznať. Výpočtom vzdialenosti so všetkými vzormi a výberom najpodobnejšieho (s najmenšou vzdialenosťou) je možné nájsť rozpoznávané slovo. Pomocou spätného trasovania vieme určiť aj časové hranice, teda zarovnanie jednotlivých signálov. Ak najpravdepdobnejšia cesta vedie diagonálne znamená to, že vzorky sa zhodujú, teda sú časovo na správnom mieste. V prípade cesty horizontálne dochádza k vynechaniu vzorky vstupného signálu alebo nasledujúca vzorka sa priradzuje tej iste vzorke vzorového signálu. Podobná situácia nastáva pri ceste vedúcej horizontálne kde sa následná vzorka vzorového signálu priradzuje tej istej vzorke vstupného.

Ako bolo spomenuté metóda porovnávania so vzormi sa často využíva pri rozpoznávaní izolovaných slov avšak je možné túto metódu rozšíriť na rozpoznávanie spojitých viet ak máme k dispozícii jednotlivé slová. Metóda DTW je ale náročná na výpočet v takomto prípade a pristupuje sa k štatistickým metódam modelovania. Existujú však spôsoby urýchlenia metódy obmedzením prehľadávania, teda vymedzenie pásma výpočtu vzdialeností a akumulovaných hodnôt (viď 4.2.2).



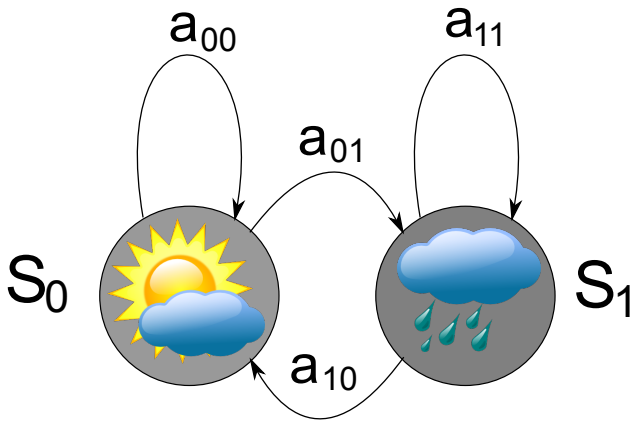
Obr. 4.4: Obmedzenie výpočtu matice DTW

4.3 Štatistické metódy

Namiesto priameho využitia vzorov, je možné popísať tieto vzory štatistickým modelom. Najbežnejšími štatistickými parametrami, ktoré je možné použiť je *stredná hodnota* a *variancia*. Teda ak máme k dispozícii veľké množstvo vzorových nahrávok slova, je možné v každom čase vypočítať strednú hodnotu vzoriek (akustických pozorovaní) a ich varianciu. Takouto reprezentáciou môžeme vytvoriť akýsi stredný model slova a zahrnúť možnú variabilitu, ktorá je viac alebo menej pravdepodobná, no predpokladá sa, že má Gaussové rozdelenie pravdepodobnosti. Teda v každom čase máme slovo vyjadrené strednou hodnotou a varianciou. Tie je možné spojiť pre ľahšie modelovanie do určitých zhlukov reprezentujúcich jeden časový úsek slova, do stavov. Každý stav bude v ideálnom prípade znamenať zmenu signálu, napríklad jedno písmeno slova. Samozrejme je tiež možné využiť iné štatistické parametre alebo iný princíp strojového učenia, no všeobecný rámec týchto metód spočíva v stavoch vychádzajúcich z Markovovho reťazca prvého rádu.

4.3.1 Markovova reťaz

Markovova reťaz je vlastne stavový automat, ktorý sa skladá zo stavov a prechodov medzi nimi. Tento stavový automat vychádza z predpokladu, že stav, v ktorom sa nachádza závisí iba od predchádzajúceho stavu, teda jedná sa o *Markovovú reťaz prvého rádu*. Druhý predpoklad, ktorý je základom evaluácie



Obr. 4.5: Príklad Markovovho reťazca zmeny počasia

modelu je, že pravdepodobnosť, že sa model nachádza v danom stave závisí iba od predchádzajúceho stavu. Príkladom môže byť reťaz zmeny počasia (viď 4.5), ktorá dokáže modelovať pravdepodobnosť postupnosti pozorovaní *"slnečno"* a *"dážď"*. V prípade rozpoznávania reči sú tieto postupnosti postupnosťami akustických príznakov. Teda mame dva stavy pre každú možnosť a prechody medzi nimi, ktoré vyjadrujú s akou pravdepodobnosťou nastane ten, ktorý stav po predchádzajúcom. Prechody medzi stavmi môžeme reprezentovať maticou:

$$A = \begin{vmatrix} 0.8 & 0.2 \\ 0.9 & 0.1 \end{vmatrix} \tag{4.2}$$

Pričom pravdepodobnosť, že začíname práve v danom stave určíme začiatočnými pravdepodobnosťami:

$$\pi = \{0.7 \quad 0.3\} \tag{4.3}$$

Potom ak chceme získať pravdepodobnosť postupnosti

$$S = \{slnečno, slnečno, dážď, slnečno\} \tag{4.4}$$

je potrebné iba sledovať jednotlivé prechody, pričom začíname počiatočnou pravdepodobnosťou π_0. Teda:

$$\begin{aligned} S &= \{s_0, s_0, s_1, s_0\} \\ P(S) &= \pi_0 \times a_{00} \times a_{01} \times a_{10} \\ &= 0.7 \times 0.8 \times 0.2 \times 0.9 \end{aligned}$$

$$= \quad 0.1008$$

Týmto spôsobom bola vykonaná *evaluácia* modelu, teda výpočet pravdepodobnosti vstupnej postupnosti a modelu. *Dekódovaním* sa rozumie časová postupnosť jednotlivých stavov, ktorá je v tomto prípade daná vstupnými príznakmi.

Tento typ Markovovho modelu sa používa na jazykové modelovanie pri rozpoznávaní reči. Na akustické modelovanie je potrebná ďalšia vrstva modelovania.

4.3.2 Skrytý Markovov Model

Ďalšiu vrstvu modelovania si je možné predstaviť ako neurčitosť pozorovania vyjadrovať stav modelu. V predchádzajúcom prípade boli pozorovania zhodné so stavmi modelu a teda zmena stavov modelu bola vopred jasná. V tomto prípade neviem počasie pozorovať priamo a tak príznaky tvoria pozorovania či ľudia nosia alebo nenosia dáždniky. Problémom ale ostáva, že existujú ľudia, ktorí nosia dáždniky aj v slnečnom počasí a naopak v daždi nie. Teda nie je možné priamo priradiť pozorovanie k stavu modelu. Je možné iba povedať a akou pravdepodobnosťou dané pozorovanie patrí danému stavu. Prakticky to znamená, že jednotlivé stavy môžu obsahovať (generovať) akékoľvek pozorovania s im prislúchajúcou pravdepodobnosťou. Tu sa používajú štatistické parametre, no pre jednoduchosť si tieto pravdepodobnosti vopred definujeme tabuľkou [1]:

dáždnik	Slnečno	Dážď
áno	0.1	0.8
nie	0.9	0.2

Pravdepodonostiam v tejto tabuľke sa hovorí *aposteriórna pravdepodobnosť*. Otázkou ostáva ako *evaluovať* daný model ak vstupná postupnosť pozorovaní bola $O = áno, nie, áno$. Ďalšou otázkou je ako odhaliť postupnosť stavov, teda vykonať *dekódovanie*. Jednou z možností je vytvoriť všetky možné postupnosti stavov S, ktoré zodpovedajú dĺžke vstupných pozorovaní, vypočítať pravdepodobnosti jednotlivých postupnosti a nakoniec tieto pravdepodobnosti sčítať:

$$P(\text{"}slnečno\ slnečno\ slnečno\text{"}|O) \quad = \quad 0.009 \times 0.448 = 0.004032$$

[1] Ak sú pravdepodobnosti dané tabuľkou jedná sa o skrytý Markovov model s diskrétnym rozdelením pravdepodobnosti

$$
\begin{aligned}
P(\text{"}slnečno\ slnečno\ dážď\text{"}|O) &= 0.072 \times 0.112 = 0.008064 \\
P(\text{"}slnečno\ dážď\ slnečno\text{"}|O) &= 0.002 \times 0.126 = 0.000252 \\
P(\text{"}slnečno\ dážď\ dážď\text{"}|O) &= 0.016 \times 0.014 = 0.000224 \\
P(\text{"}dážď\ slnečno\ slnečno\text{"}|O) &= 0.072 \times 0.216 = 0.015552 \\
P(\text{"}dážď\ slnečno\ dážď\text{"}|O) &= 0.576 \times 0.054 = 0.031104 \\
P(\text{"}dážď\ dážď\ slnečno\text{"}|O) &= 0.016 \times 0.027 = 0.000432 \\
P(\text{"}dážď\ dážď\ dážď\text{"}|O) &= 0.128 \times 0.003 = 0.000384
\end{aligned}
\tag{4.5}
$$

Teda výsledná pravdepodobnosť po sčítaní je $P(S|O) = 0,060044$. Keďže každá postupnosť stavov môže generovať akúkoľvek postupnosť pozorovaní nie je možné jednoznačne určiť jednu postupnosť stavov pre úlohu dekódovania. Postupnosť stavov je akoby skrytá, od tohto je odvodený aj názov modelu. Možné je ale určiť najpravdepodobnejšiu postupnosť stavov, napríklad výberom z predchádzajúcich vypočítaných hodnôt. V tomto konkrétnom prípade je to postupnosť $S = \{dážď\ slnečno\ dážď\}$, čo zodpovedá predpokladanému výstupu vzhľadom na vstupné pozorovania.

$$
\begin{aligned}
P(S|O) &= P(O|S)P(S) \\
&= b_1(o_0)\pi_1 \times b_0(o_1)a_{10} \times b_1(o_2)a_{01} \\
&= 0.8 \times 0.3 \times 0.9 \times 0.9 \times 0.8 \times 0.2 \\
&= 0.031104
\end{aligned}
$$

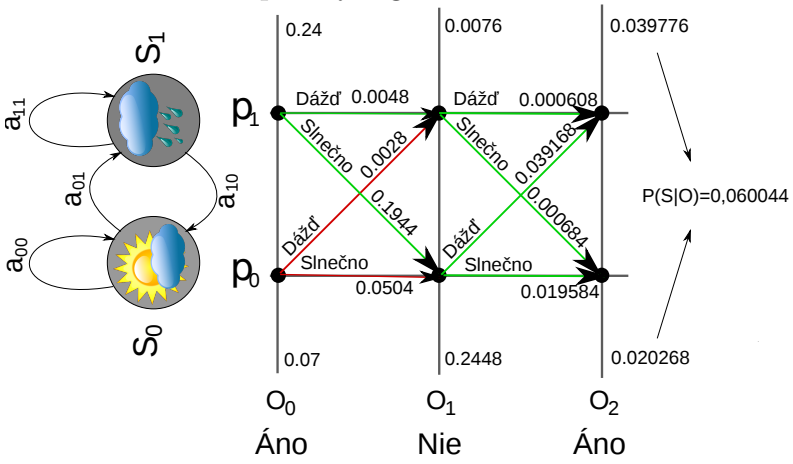
V rozpoznávaní reči sú tieto stavy súčasťou modelu jedného slova a vstupnými pozorovaniami sú akustické príznaky. Teda pre každé slovo existuje podobný model a opäť evaluáciou modelu slova a vstupných príznakov nahrávky, môžeme vypočítať pravdepodobnosť, že tento model patrí danému slovu. Výberom najpravdepodobnejšieho modelu získame rozpoznané slovo. Pomocou dekódovania zase získame najpravdepodobnejšie priradenie jednotlivých stavov k vstupným pozorovaniam.

4.3.3 Algoritmus na evaluáciu HMM

Výpočet všetkých možných kombinácii vstupných stavov a akustických pozorovaní je náročné. Existuje však algoritmus na jeho uľahčenie. Môžeme si definovať premennú $\alpha_t(i)$, ktorá bude udržiavať priebežné hodnoty výpočtu v čase t v stave i:

$$
\alpha_t(i) = P(o_1 o_2 ... o_t | s_1 s_2 ... s_t, s_t = s_i), \tag{4.6}
$$

Obr. 4.6: Dopredný algoritmus evaluácie modelu



potom začíname inicializáciou, teda prijatím prvého pozorovania (jeho pravdepodobnosti $b_i(o_0)$) výpočtu hodnoty $\alpha_i(0)$ pre stavy $i = 1, 2$:

$$\alpha_i(0) = \pi_i b_i(o_0). \tag{4.7}$$

Nasleduje rekurzia, ktorá prechádza všetkými stavmi podľa prechodov a dopĺňuje pre všetky stavy $(i = 1, 2)$ hodnoty $\alpha_i(t)$, pričom používa predchádzajúce hodnoty tejto premennej:

$$\alpha_i(t) = \left[\sum_{j=1}^{N} \alpha_j(t-1)a_{ji}\right] b_i(o_t). \tag{4.8}$$

Výsledná pravdepodobnosť je teda súčtom dočasnej premennej v každom stave:

$$P(S|O) = \sum_{j=1}^{N} \alpha_j(T) \tag{4.9}$$

Príklad dopredného algoritmu je zobrazený na Obr. 4.3.3. Teda hodnotu v každom stave vypočítame tak, že k hodnote v predchádzajúcom čase pripočítame pravdepodobnosť prechodu medzi stavmi vynásobenú pravdepodobnosťou pozorovania. Ako je zrejme výsledok evaluácie je ten istý, ale v tomto prípade sme znížili výpočtovú náročnosť evaluácie.

Podobne vieme postupovať spätne pomocou spätného algoritmu pomocou premennej $\beta_i(t)$:

$$\alpha_t(i) = P(o_T o_T - 1 ... o_t | s_1 s_2 ... s_t, s_t = s_i), \tag{4.10}$$

pričom inicializácia zahŕňa iba vynulovanie premenných:

$$\beta_i(T) = 0. \tag{4.11}$$

Nasleduje rekurzia od času $T-1$ až po začiatok:

$$\beta_i(t) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_j(t+1) \tag{4.12}$$

Posledným krokom je pridanie začiatočných pravdepodobností a spočítanie výsledných pravdepodobnosti pre všetky stavy:

$$P(S|O) = \sum_{i=1}^{N} \pi_0 b_i(o_0) \beta_j(0) \tag{4.13}$$

4.3.4 Algoritmus na dekódovanie HMM

Odhalenie najpravdepodobnejšej postupnosti stavov je podobne možné vypočítať obmenou dopredného algoritmu, tak, že namiesto sčítania hodnôt predchádzajúcich premenných v čase a v každom stave vyberieme maximum. Teda podobne ako v predchádzajúcom prípade si vytvoríme premenú, do ktorej budeme ukladať priebežné hodnoty:

$$V_t(i) = P(o_1 o_2 ... o_t | s_1 s_2 ... s_t, s_t = s_i), \tag{4.14}$$

ktorá vyjadruje hodnotu v stave i v čase t ak sme teda použili o_t pozorovaní. Inicializácia je rovnaká ako v prípade dopredného algoritmu:

$$V_i(0) = \pi_i b_i(o_0), \tag{4.15}$$

nasleduje rekurzia, pri ktorej vyberáme maximálny príspevok k pravdepodobnosti a pritom si zapisujeme odkiaľ, z akého stavu i, sme túto maximálnu hodnotu vypočítali. Na zápis predchádzajúceho stavu použijeme premennú $\psi_i(t)$:

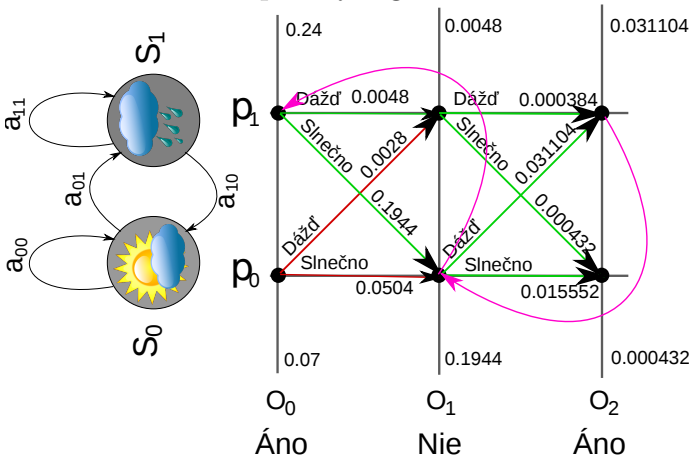
$$\alpha_i(t) = \left[\max_{j=1}^{N} V_j(t-1) a_{ji} \right] b_i(o_t) \tag{4.16}$$

$$\psi_i(t) = \arg \left[\max_{j=1}^{N} V_j(t-1) a_{ji} \right]. \tag{4.17}$$

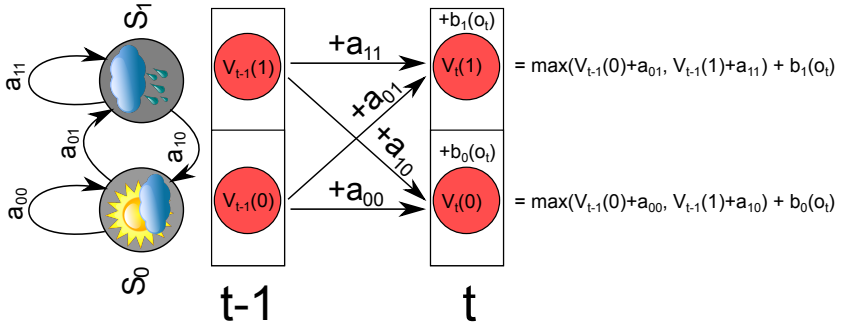
Výsledná pravdepodobnosť najpravdepodobnejšej postupnosti stavov je maximum z vypočítaných hodnôt v každom stave:

$$P(O|S) = \max_{j=1}^{N} V_j(T), \tag{4.18}$$

Obr. 4.7: Dopredný algoritmus evaluácie modelu



Obr. 4.8: Dopredný algoritmus evaluácie modelu



pričom najpravdepodobnejší koncový stav s_T^* obsahuje v premennej $\psi_i(T)$ spätnú cestu, ktorá je najpravdepodobnejšou postupnosťou stavov:

$$s_T^* = \arg\max_{j=1}^{N} V_j(T) \tag{4.19}$$

$$s_{T-1}^* = \psi_{s_T^*}(T) \tag{4.20}$$

Príklad viterbiho dekódovania je zobrazený na Obr: 4.7. Fialovou farbou je zaznačená cesta s maximálnou pravdepodobnosťou. Výsledkom je teda najpravdepodobnejší stav vpravo hore s hodnotou 0.03114, čo odpovedá výpočtu (4.5), ale v tomto prípade sme znížili výpočtové nároky.

Reálna implementácia dekódovacieho algoritmu pozostáva z objektov, ktoré prechádzajú medzi stavmi a hovorí sa im *tokeny*. Tieto tokeny obsahujú dočasnú premennú $V_i(t)$ a tiež premennú na zapisovanie spätnej cesty $\psi_i(t)$. Príklad dekódovania je na Obr: 4.8. Výber najvyššej hodnoty v každom čase môžeme realizovať tak, že obmedzíme počet tokenov na stav, teda dovolíme

iba jeden token na stav. Ak nastane situácie kedy v jednom stave musia byť uložené dva tokeny, vyberiem ten najpravdepodobnejší. Jednoduchá implementácia používa dva zásobníky, jeden pre predchádzajúci čas $(t-1)$ a ďalší pre súčasný (t). Algoritmus sa teda riadi nasledovnými pravidlami:

- Umiestnenie tokenov do každého stavu a jeho počiatočné nastavenie (zásobník $t-1$)
-
  - Vyberanie každého tokenu zo zásobníka $t-1$, prepočítanie jeho hodnoty a uloženie do zásobníka t
  - V prípade vkladania tokenu na miesto do zásobníka t, ktorý už obsahuje token, ponechá sa iba ten s vyššou pravdepodobnosťou
  - Výmazanie zásobníka $t-1$
  - Výmena zásobníkov $t-1$ a t
- Po dokončení dekódovania (spracovania všetkých pozorovaní) výsledný token s maximálnou hodnotou v poslednom zásobníku obsahuje výsledok dekódovania.

4.3.5 Spojité skryté Markovové modely

Doteraz boli pravdepodobnosti jednotlivých pozorovaní určené tabuľkou. Takéto HMM sa nazývajú *diskrétne* HMM akustické modely. Avšak pre lepšie modelovanie je možné použiť štatistické parametre *stredná hodnota* a *varianca*. Teda každé slovo vieme popísať konečným počtom stavov, napríklad písmenami. Každé písmeno vieme popísať *strednou hodnotou* (μ_{jm}) jej akustických príznakov a varianciou (C_{jm}), pričom sa predpokladá Gaussovo rozdelenie pravdepodobnosti. Na modelovanie zložitejších rozdelení pravdepodobnosti, ktorými sa dá dosiahnúť presnejšie modelovanie, sa používajú zmesi Gaussových funkcii. Pravdepodobnosť, že daný stav generoval vstupný akustický príznak bude teda zložená z pravdepodobností viacerých Gaussových hustotných funkcií. Pravdepodobnosť, že daný akustický príznak patrí (alebo je generovaný) daným stavom je:

$$\begin{aligned} b_j(o_t) &= \sum_{m=1}^{M} w_{jm} \aleph(o_t, \mu_{jm}, C_{jm}) \qquad (4.21) \\ &= \sum_{m=1}^{M} w_{jm} \frac{1}{(2\pi)^{n_r} |C_{jm}|} \exp\left(-\frac{1}{2}(o_t - \mu_{jm})^T C_{jm}^{-1}(o_t - \mu_{jm})\right), \end{aligned}$$

kde w_{jm} je váha danej Gaussovej funkcie pre pre stav j modelu Gaussovej funkcie m určenej strednou hodnotou μ_{jm} a varianciou C_{jm}, pričom vstupom je akustický príznak o_t. Po sčítaní jednotlivých výsledkom je výstupom

pravdepodobnosť, ktorá bola v predchádzajúcom prípade určená tabuľkou. Hovoríme jej *aposteriórna* pravdepodobnosť.

Proces *evaluácie* a *dekódovania* je rovnaký ako v prípade diskrétnych HMM s tym rozdielom, že jednotlivé hodnoty pravdepodobností (predtým vyberané z tabuľky) sa prepočítavajú pre všetky vstupné akustické príznaky. Opäť si stačí predstaviť namiesto vstupných pozorovaní akustické príznaky ktoré popisujú vstupnú nahrávku a stavy ako súčasť modelu jedného slova. Stavy vyjadrujú časovú zmenu v slovách ako aj príznaky. Teda sa snažíme zistiť pravdepodobnosť postupnosti stavov a vstupných príznakov. Jeden model slova sa môže skladať z viacerých stavov a takýchto modelov je možné mať pre každé slovo zo slovníka. Pomocou evaluačného dopredného algoritmu je možné vypočítať pravdepodobnosť každého modelu a príznakov a vybrať najpravdepodobnejší model. Slovo, ktorému model patri, je výsledkom rozpoznávania.

Trénovacím procesom zase chápeme nájdenie matice prechodov medzi stavmi a vhodných parametrov Gaussových rozložení pravdepodobnosti z trénovacích dát tvorené slovami, ktoré chceme rozpoznávať. Pri trénovaní sa snažíme nájsť hodnoty všetkých premenných modelu θ tak, aby pravdepodobnosť $P(O|\theta)$, kde O je súbor všetkých vstupných príznakov slov na trénovanie, nadobúdala svoje globálne maximum. Algoritmy, riešiace takýto typ úlohy sa v literatúre označujú ako algoritmy estimácie maximálnej vierohodnosti (z angl. maximum likelihood estimation, skr. MLE). Dva základné typy MLE algoritmov sú *Baumov-Welchov algoritmus* a *Viterbiho algoritmus*. Baumov-Welchov algoritmus je označovaný ako EM (z angl. expectation maximization) algoritmus. Má veľmi dobré konvergentné vlastnosti, vedúce vždy k lokálnemu maximu. Ku globálnemu maximu je možné sa priblížiť niekoľkonásobným opakovaným trénovaním HMM s rôznymi počiatočnými podmienkami a výberom modelu s najvyššou hodnotou $P(O|\theta)$. Tento proces sa nazýva *reestimácia* parametrov HMM.

4.3.6 Subslovné modelovanie

Spojitý rečový signál sa tak vo vedomí človeka rozkladá na elementárne časti, ktorých podobnosť alebo odlišnosť má principiálny význam pre rozpoznávanie reči aj pre zefektívnenie rozpoznávacieho procesu. Z lingvistického hľadiska sú elementárnymi úsekmi reči veta, slovo a slabika. Aby sme dokázali modelovať akékoľvek slovo nielen slová, ktoré sa vyskytovali v trénovacej množine, je možné použiť akustické jednotky menšie ako slovo a ktoré sa pri tvorbe každého slova opakujú. Snažíme sa teda použiť podobné zvuky, z ktorých sa skladajú slová., teda z fonetického hľadiska sú to hláska, fonéma, alofóna a fóna. Najpoužívanejšia jednotka na modelovanie je fonéma.

Fonéma

Najmenšie časové úseky, na ktoré sa súvislý prúd elementárnych rečových zvukov dá rozčleniť, sú hlásky. *Hláska* je foneticky presne určená, (foneticky zatriedená) artikulačno-akustická jednotka reči [42]. Pre pojem *hláska* sa v rozličných jazykoch používajú rozličné označenia. Pojmu hláska, tak ako je chápaný v slovenskej literatúre, napr. [42], zodpovedá anglické označenie *speech sound*. Na bližšie špecifikovanie zvukov reči sa z anglosaskej literatúry prevzala terminológia radu *fonéma - alofóna – fóna* (*phoneme, allophone*).

Fonéma je základnou zvukovou jednotkou reči, nesúcou jazykový obsah reči. Fonému môžeme považovať za ideálnu zvukovú jednotku, charakterizovanú kompletným súborom artikulačných pohybov, ktoré sú potrebné na jej vznik. Z akustického pohľadu fonéma reprezentuje triedu zvukov reči, ktoré majú rovnaký význam. Ak by sme chceli exaktne a konzistentne produkovať reč pozostávajúcu z foném, reč by obsahovala prúd *diskrétnych kódov*. Fonémy jazyka teda zahrňujú minimálny teoretický súbor zvukových jednotiek, ktoré postačujú na vyjadrenie všetkých významov v danom jazyku [42].

Drobné akustické variácie foném sú *alofóny*. Alofóny reprezentujú prípustný stupeň voľnosti pri vyslovovaní foném, pričom táto flexibilita nezávisí iba od fonémy samotnej, ale najmä od jej pozície v rečovom prúde [42]. Závislosť akustickej realizácie fonémy od predchádzajúceho a nasledujúceho zvuku, od tempa a intonácie reči je známa pod označením *koartikulácia*. Koartikulácia súvisí s funkciou artikulačných orgánov, tie sa plynulo pohybujú z jedného artikulačného stavu do druhého, zodpovedajúceho tej ktorej foném v rečovom prúde. Pretože jednotlivé artikulačné orgány sú nezávislé, niektoré sa pohybujú pomalšie, iné rýchlejšie, takže pri niektorých postupnostiach foném v rečovom prúde sa niektorý artikulačný orgán nestihne dostať do príslušnej polohy, preskočí ju a prejde do ďalšej.

Fonetická transkripcia reči

Fonetická transkripcia je súbor znakov (*grafém*), ktoré sa používajú na presný zápis reči v jej zvukovej forme. Cieľom fonetickej transkripcie je zapísať znenie ľubovoľného hovoreného prejavu tak, aby sa čo najpresnejšie a najpodrobnejšie zachytili všetky zvukové javy rečového signálu, aby sa pre každý odlišný zvuk, ktorý sa sluchom zachytil, použil osobitný znak a aby sa pre ten istý zvuk použil vždy ten istý znak. Súbor znakov pre fonetickú transkripciu reči je obyčajne väčší, ako sa používa v pravopisných sústavách.

Jestvuje viac druhov fonetickej transkripcie. Najrozšírenejšia je medzinárodná fonetická transkripcia IPA (*International Phonetic Alphabet*). Medzinárodná fonetická abeceda sa v slovenských fonetických prácach bežne nepou-

žíva. Na fonetický zápis slovenčiny sa používa systém fonetickej transkripcie SPA (*Slovak Phonetic Alphabet*), ktorej východiskom je inventár znakov slovenského pravopisného systému (ortografie) s použitím niektorých znakov IPA [42] (pozri Tab. 4.1). Je to výsledok jednak historických činiteľov ako aj problému technických možností tlačiarní v minulosti. Systematický pokus o podanie stručného obrazu systému slovenských hlások prostredníctvom znakov IPA možno nájsť v [56].

Tak IPA ako aj SPA používajú na zápis zvukov reči znaky, ktoré nekorešpondujú so znakmi, používanými v počítačovom spracovaní textu. Na analýzu a porovnávanie písanej a hovorenej podoby jazyka boli preto v systémoch ARR zavedené iné spôsoby fonetickej transkripcie, ktoré vychádzajú z medzinárodne štandardizovanej *počítačovej abecedy znakov* ASCII. **SAMPA** (*Speech Assessment Methods Phonetic Alphabet*) je fonetická abeceda, ktorá bola navrhnutá medzinárodnou skupinou fonetikov v rámci programu ESPRIT, projekt 1541 - SAM (*Speech Assessment Methods*) v rokoch 1987-89 pre jazyky Európy. SAMPA pre slovenčinu bola prvýkrát navrhnutá na Oddelení analýzy a syntézy reči ÚI SAV a aj prakticky použitá v rámci projektu *INCO-COPERNICUS Project 977017* SPEECHDAT(E) [61]. Iná, trochu odlišná verzia bola navrhnutá v [31]. Obe verzie sú zahrnuté v Tab. 4.1.

Tabuľka 4.1: Inventár foném hovorenej slovenčiny a ich fonetický zápis pomocou SPA [42], SAMPA1 [61] a SAMPA2 [31].

SPA	SAMPA1	SAMPA2	Príklad	SPA	SAMPA1	SAMPA2	Príklad
i	i	I	p**i**vo	e	e	E	m**e**no, p**e**ro
a	a	a	k**a**pitola, p**a**pier	o	o	O	n**o**ha, p**o**p**o**l
u	u	U	b**u**bon, p**u**to	ä	{	{	p**ä**ta, m**ä**so
í	i:	I:	v**í**ťaz, p**í**skať	é	e:	E:	g**é**n, nov**é**
á	a:	a:	poh**á**r, p**á**s	ó	o:	O:	katal**ó**g, p**ó**l
ú	u:	U:	m**ú**r, p**ú**pava	ťa	i_^a	I_^a	p**ia**tok
ťe	i_^e	I_^E	m**ie**r, sp**ie**vať	ťu	i_^u	I_^U	pan**iu**, cudz**iu**
ŭo	u_^o	U_^O	k**ôň**	r	r	r	pa**r**a
r̂	r=	r=	p**r**st, v**r**ch	ŕ̂	r=:	r=:	v**ŕ**ba
l	l	l	ska**l**a, v**l**ak	l̂	l=	l=	v**l**k
Ĺ̂	l=:	l=:	v**Ĺ**ča	l̂̃	L	L	**ľ**ad, **ľ**avý
m	m	m	**m**a**m**a	ŋ	F	F	a**m**fiteáter
n	n	n	ra**n**a, pá**n**sky	ŋ̡	N\	—	Slove**n**sko
ŋ	N	N	ba**n**ka, ce**n**gať	~~ɲ~~	—	—	bro**n**chitída
ň	J	J	va**ň**a	v	v	v	slo**v**o
ŭ	u_^	U_^	ko**v**, pra**v**da	ǐ	i_^	I_^	kra**j**
j	j	j	**j**ama, dva**j**a	p	p	p	**p**o**p**ol
ž	b	b	**ž**aba	t	t	t	va**t**a
ť	c	c	Ma**ť**o, pla**ti**ť	d	d	d	vo**d**a
ď	J\	J\	há**ď**a	k	k	k	pá**k**a
g	g	g	**g**uma, a**g**át	f	f	f	**f**iga, **f**ajka
w	w	w	**v**dova	s	s	s	o**s**a, o**s**em
z	z	z	**z**ima, vá**z**a	š	S	S	**š**ek, košeľa
ž	Z	Z	ve**ž**a	x	x	x	**ch**ata
h	h	h\	Pra**h**a	ɣ	G	G	vr**ch**_hory
c	ts	ts	**c**ena	č	tS	tS	o**č**i, ma**č**ka
ʒ	dz	dz	me**dz**a, há**dz**ať	ǯ	dZ	dZ	**dž**ungľa, **dž**em
>š	—	—	vyšší	>n	—	—	de**nn**ý
>ň	—	—	de**nn**e	—	—	—	—

5

Jazykové modelovanie

Jednou z oblastí výskumu a vývoja systémov interakcie človeka so strojom hovorenou rečou je aj oblasť spracovania prirodzeného jazyka a tvorba štatistických modelov nasaditeľných vo vybraných rečových aplikáciách a použiteľných v praxi. Keďže slovenčina patrí do skupiny vysoko-flektívnych jazykov vyznačujúcich sa bohatou morfológiou a neusporiadanosťou jazyka, vynára sa tu množstvo problémov súvisiacich s tvorbou rozsiahlych korpusov textových dát, ich organizáciou a spracovaním, ako aj s výberom vhodných metód v súčasnosti používaných pri tvorbe štatistických modelov jazyka. Obzvlášť je to v prípade slovenčiny, kde štandardné metódy častokrát vďaka nedostatku jazykových zdrojov nedosahujú očakávané výsledky. Z toho dôvodu, návrh sofistikovanejších riešení opierajúcich sa o morfológiu jazyka je v oblasti tvorby jazykových modelov viac než potrebný.

5.1 Jazykový model

Vo všeobecnosti, úlohou konštrukcie modelu jazyka v systéme na automatické rozpoznávanie reči je poskytnúť čo najpresnejší a najrýchlejší odhad apriórnej pravdepodobnosti $P(W)$ pre ľubovoľnú postupnosť slov W. Jazykový model udáva tiež spôsob formovania postupnosti foném v akustickej reprezentácii jazyka, a tým vnáša určité obmedzenia na postupnosť slov vo vetách, keďže každý jazyk sa od iného líši práve v konkrétnom slovníku a pravidlách skladania slov do viet. Tieto obmedzenia potom delíme na *deterministické* a *stochastické*. V prípade deterministického obmedzenia nemôže byť vyslovené slovo, ktoré nie je zahrnuté v slovníku a nemôže byť vyslovené inak, než udáva správna výslovnosť. Obmedzenie stochastické vychádza z predpokladu, že nie všetky slovné spojenia musia byť vyslovované s rovnakou pravdepodobnosťou.

5.1.1 Deterministický jazykový model

Deterministické metódy v oblasti modelovania jazyka vychádzajú z Chomského teórie formálneho jazyka [37], a sú označované tiež ako *kontextovo-nezávislé gramatiky* (z angl. *context-free grammars*). Jazyk je v tomto prípade modelovaný pomocou jednoduchej *gramatiky* determinujúcej gramatické pravidlá a prípustné vetné štruktúry a *algoritmom syntaktickej analýzy*, ktorý analyzuje vety a porovnáva ich s preddefinovanou gramatickou štruktúrou. Algoritmom sú akceptované len také vetné štruktúry, ktoré spĺňajú pravidlá stanovené gramatikou.

Pri zostavovaní deterministických modelov sa vychádza z dostupných štandardov na zápis rečových gramatík pomocou značkovacích jazykov založených prevažne na formáte XML (z angl. *eXtensible Markup Language*).

Príklad jednoduchej gramatiky pre službu „*Cestovné poriadky*“ v slovenskom rečovom dialógovom systéme IRKR [35] je na Obr. 5.1.

```
<?xml version="1.0" encoding="windows-1250"?>
<grammar root="service" scope="private">
   <rule id="service" scope="public">
     <one-of>
       <item tag="weather">počasie</item>
       <item tag="trains">vlaky</item>
       <item tag="trains">vlakové cestovné poriadky</item>
       <item tag="trains">cestovné poriadky vlakov</item>
       <item tag="mhdke">mestká doprava</item>
       <item tag="mhdke">mestká hromadná doprava</item>
       <item tag="mhdke">mestká doprava v košiciach</item>
       <item tag="mhdke">mestká hromadná doprava v košiciach</item>
       <item tag="mhdke">mhd košice</item>
       <item tag="mhdke">mhd v košiciach</item>
       <item tag="bus">autobusy</item>
       <item tag="bus">autobusové cestovné poriadky</item>
       <item tag="bus">cestovné poriadky autobusov</item>
     </one-of>
  </rule>
</grammar>
```

Obr. 5.1: Príklad jednoduchej kontextovo-nezávislej gramatiky pre službu „*Cestovné poriadky*“

5.1.2 Štatistický jazykový model

Ako už bolo spomenuté, pomocou štatisticky-založeného jazykového modelu je možné stanoviť pre ľubovoľnú postupnosť slov $W = w_1 w_2 \dots w_N$ apriórnu

pravdepodobnosť $P(W)$ tejto postupnosti, a poskytnúť tak bloku prehľadávacej stratégie v procese dekódovania čo najrýchlejší a najpresnejší odhad danej pravdepodobnosti pomocou nasledujúceho vzťahu:

$$P(W) = P(w_1 w_2 \dots w_N) = \prod_{i=1}^{N} P(w_i | w_1 w_2 \dots w_{i-1}), \tag{5.1}$$

kde $P(w_i|w_1 w_2 \dots w_{i-1})$ je pravdepodobnosť výskytu slova w_i, podmienená postupnosťou slov $w_1 w_2 \dots w_{i-1}$, ktorá sa označuje tiež ako *história*. Takýto postup rozkladu umožňuje rozpoznávať postupnosť slov v priebehu jej vyslovovania a určovať pravdepodobnosť $P(W)$ pre účely dekódovania postupne.

V n-gramovom modeli sa predpokladá, že pravdepodobnosť slova je daná práve n za sebou nasledujúcimi slovami v pozorovaní ich náhodného výberu. V praxi sa pre systémy na automatické rozpoznávanie reči najčastejšie využívajú trigramy (ak $n = 3$), kde pre odhad podmienenej pravdepodobnosti nasledujúceho slova môže byť zavedená nasledujúca aproximácia:

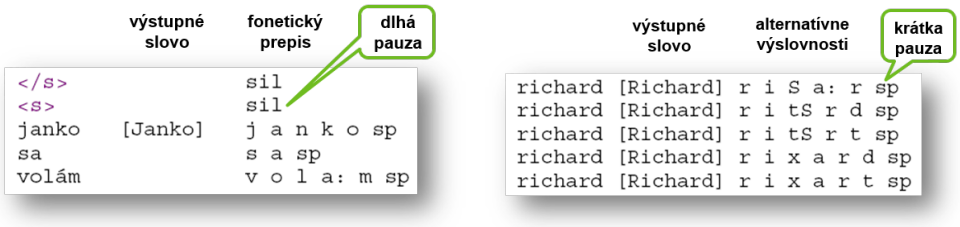
$$P(W) = \prod_{i=1}^{N} P(w_i | w_{i-2} w_{i-1}), \tag{5.2}$$

kedy pravdepodobnosť slova w_i je daná jeho históriou, ktorá je tvorená práve dvoma slovami w_{i-2} a w_{i-1}. Výhodou n-gramových modelov je ich jednoduchý odhad, ktorý je založený na zisťovaní relatívnej početnosti výskytu slov, resp. postupností slov v trénovacích dátach pomocou *metódy maximálnej vierohodnosti* (z angl. *maximum likelihood*) [37].

5.2 Slovník

Slovník v procese rozpoznávania reči slúži na prepojenie akustickej podoby slov s ich písanou podobou a obmedzuje samotný rozpoznávací systém počtom jedinečných tvarov slov, ktoré sú v ňom obsiahnuté. Mal by obsahovať v jazyku čo najviac frekventované slová a v čo najlepšej miere by mal korešpondovať so sémantickým obsahom oblasti, v ktorej rozpoznávanie reči prebieha. Veľkosť slovníka je jedným z kľúčových parametrov pri návrhu každého systému na automatické rozpoznávanie reči. Je závislá od štruktúry daného jazyka a vplýva priamo na výsledok a na výpočtovú a časovú náročnosť procesu rozpoznávania reči.

Slovník vo všeobecnosti obsahuje zoznam najviac frekventovaných a gramaticky správnych tvarov slov v danom jazyku a získava sa väčšinou zo samotného trénovacieho korpusu, napr. *metódou absolútneho výberu slov* alebo



(a) (b)

Obr. 5.2: Príklad slovníka (a) s fonetickým prepisom slov a (b) alternatívnymi výslovnosťami

metódou maximálnej vierohodnosti v prípade doménovo-orientovaných úloh v systéme automatického rozpoznávania reči [36].

Je vhodné pripomenúť, že jedným z kľúčových parametrov udávajúcich kvalitu slovníka je počet *mimoslovníkových slov* (z angl. *out-of-vocabulary*, skr. OOV), ktorý výrazne prispieva k celkovému výsledku rozpoznávania reči.

5.3 Hodnotenie kvality jazykových modelov

Medzi základné hodnotiace miery n-gramových jazykových modelov patrí *perplexita* (z angl. *perplexity*), alebo aj *zložitosť* modelu. Táto je pre postupnosť slov W daná pravdepodobnosťou $P(W)$ tejto postupnosti a normalizovaná počtom N slov nasledovne:

$$PPL_W = P(w_1 w_2 \dots w_N)^{-\frac{1}{N}}. \tag{5.3}$$

Zo vzťahu je možné vidieť, že vysoká podmienená pravdepodobnosť postupnosti slov spôsobuje nízku perplexitu. Potom minimalizácia perplexity je ekvivalentná maximalizácii vierohodnosti funkcie, a tým úspešnosť rozpoznávania narastá. Perplexita modelu vychádza predovšetkým z konštrukcie samotného jazyka a mení sa v závislosti od veľkosti slovníka a pravidiel formovania slov do viet. Táto miera však sama o sebe nemusí garantovať zlepšenie presnosti systému na automatické rozpoznávanie reči, pretože neberie do úvahy akustickú podobu slov, no zvyčajne s takýmto zlepšením koreluje [37].

5.4 Vyhladzovanie modelu

Keďže každý textový korpus je konečný, nemôže obsahovať všetky možné kombinácie slov. Nulový výskyt nepozorovaných javov potom vedie k chybám

v rozpoznávaní, pretože je teoreticky možné, že rečník vysloví aj takú kombináciu slov, ktorá sa v danom trénovacom korpuse, a tým aj v n-gramovom modeli vôbec nevyskytuje. Vyhladzovanie (z angl. *smoothing*) sa preto snaží daný problém riešiť rovnomernejším prerozdelením časti pravdepodobnosti pozorovaných n-gramov v trénovacích dátach medzi nepozorované n-gramy, zvýšením pravdepodobnosti nulových výskytov a výskytov s veľmi malou pravdepodobnosťou a následným znížením pravdepodobnosti pozorovaných javov s vysokou frekvenciou v dôsledku vety o úplnej pravdepodobnosti [12].

Vo všeobecnosti je možné metódy vyhladzovania zadeliť do dvoch základných skupín. K prvej skupine tzv. *základných schém vyhladzovania* patria metódy, ktoré sa snažia zvýšiť výskyt každého n-gramu v jazykovom o vhodné kladné číslo. Ako príklad je možné uviesť metódu aditívneho vyhladzovania „*Add-One*“ (alebo aj Laplaceovo vyhladzovanie), ktorá v prípade n-gramového modelu, zvýši každému n-gramu výskyt o jedna nasledovne:

$$P_{A1}(h|w) = \frac{1 + C(h,w)}{\sum_w [1 + C(h,w)]} = \frac{1 + C(h,w)}{|V| + \sum_w C(h,w)}, \tag{5.4}$$

kde $C(h,w)$ udáva početnosť n-gramu a $|V|$ je veľkosť slovníka. Obdobným spôsobom je možné zvýšiť výskyt n-gramov o menej než 1, v intervale $(0,1)$. Takáto modifikácia sa označuje aj ako „*Add-δ*“ (alebo aj Lindstoneovo) vyhladzovanie, pričom výber koeficientu δ sa uskutočňuje zvyčajne dynamickým spôsobom. K základným schémam vyhladzovania možno zaradiť aj dobre známy *Goodov-Turingov odhad*, diskontný algoritmus, ktorý na odhad celkovej pravdepodobnosti všetkých nepozorovaných javov využíva početnosti tých pozorovaných javov, ktoré sa v celom korpuse vyskytujú práve raz.

trénovací text

<s> Ján číta zaujímavú knihu </s>
<s> túto knihu číta aj Milan </s>

pred vyhladením

$$P(\text{'Milan číta knihu'}) = P(\text{'Milan'}|\text{'<s>'}) \times P(\text{'číta'}|\text{'Milan'}) \times$$
$$\times P(\text{'knihu'}|\text{'číta'}) \times P(\text{'</s>'}|\text{'knihu'}) = \frac{0}{2} \times \frac{0}{1} \times \frac{1}{2} \times \frac{1}{2} = 0$$

po vyhladení

$$P(\text{'Milan číta knihu'}) = P(\text{'Milan'}|\text{'<s>'}) \times P(\text{'číta'}|\text{'Milan'}) \times$$
$$\times P(\text{'knihu'}|\text{'číta'}) \times P(\text{'</s>'}|\text{'knihu'}) = \frac{1}{11} \times \frac{1}{9} \times \frac{2}{9} \times \frac{2}{11} = 0,6061$$

Obr. 5.3: Príklad vyhladzovania modelu pomocou metódy „*Add-One*“

Aj keď Goodov-Turingov odhad dosahuje v praxi často veľmi uspokojivé výsledky, nezahŕňa však kombináciu modelov vyššieho rádu s modelmi niž-

šieho rádu. Takýto spôsob modifikácie parametrov jazykového modelu možno vykonať pomocou tzv. *ústupových* a *interpolačných schém*. Zatiaľčo princíp vyhladzovania v *ústupovej schéme* spočíva v tom, že vyhladené odhady pravdepodobností sú určené buď z pôvodných relatívnych početností n-gramov, alebo z relatívnych početností $(n-1)$-gramov vážených pomocou tzv. *ústupovej váhy* (z angl. *back-off weight*), v prípade že vyššia (pôvodná) úroveň mala nulovú hodnotu, v *interpolačnej schéme* sú pri výpočte odhadov podmienených pravdepodobností vyšších úrovní použité vždy vážené kombinácie všetkých n-gramov nižšej úrovne. Z najbežnejšie používaných algoritmov vyhladzovania obsiahnutých v ústupových a interpolačných schémach možno spomenúť *Katzov diskontný model*, *model s absolútnym a lineárnym diskontom*, *Wittenov-Bellov model*, *Kneserov-Neyov model* a jeho modifikácie [12].

```
\data\
ngram 1=5
ngram 2=4
ngram 3=3

\1-grams:
-2.960382   </s>
-99         <s>      -2.108015
-3.932943   Janko    -1.072028
-3.670839   volám    -1.732694
-3.396382   sa       -1.916983

\2-grams:
-4.263822   <s> volám    -1.802879
-4.638904   Janko </s>   -0.398174
-3.756356   sa Janko     -2.194048
-3.360473   volám sa     -1.735974

\3-grams:
-2.318274   <s> volám sa
-4.535064   sa Janko </s>
-3.190907   volám sa Janko
```

Obr. 5.4: Príklad trigramového jazykového modelu vyhladeného pomocou ústupovej schémy

V prípade slovenského jazyka sa najviac osvedčil Katzov model pri vyhladzovaní modelov trénovaných na veľmi malých textových korpusoch, modifikovaný Kneserov-Neyov algoritmus pri rozsiahlych korpusoch a Wittenov-Bellov model v prípade textov s pevnejšími väzbami medzi slovami [36].

5.5 Spájanie a adaptácia modelov

Hlavným problémom korpusov textových dát, ktoré sú vytvárané zberom textu na sieti Internet je ich tematická rôznorodosť [80]. V úlohách doménovo-orientovaného rozpoznávania reči častokrát tieto dáta vnášajú do procesu rozpoznávania mnoho nejednoznačností, ktoré sú spôsobené nadhodnotením aj takých odhadov pravdepodobnosti n-gramov, ktoré typicky s danou oblasťou rozpoznávania nesúvisia. Z tohto dôvodu je zvyčajne žiaduce znížiť tieto *mimodoménové* (z angl. *out-of-domain*) odhady pravdepodobnosti na úroveň, ktorá zároveň posilní štatistiky *vnútrodoménových* (z angl. *in-domain*) slov, ale aj rozšíri štatistiky o také slovné spojenia, ktoré sa v rámci domény môžu vyskytovať len s veľmi malou pravdepodobnosťou alebo vôbec. Takýto spôsob modifikácie pôvodných odhadov pravdepodobností jednotlivých početností v procese trénovania jazykového modelu alebo na úrovni modelu, možno uskutočniť kombináciou viacerých čiastkových tematicky-zameraných modelov s následnou adaptáciou na vybranú tému [80].

Jedna z prvotných metód kombinácie jazykových modelov je založená na spájaní resp. vážení početností n-gramov (z angl. *count merging*, alebo aj *n-gram weighting*) v modeli jazyka na úrovni ich frekvenčného výskytu, ktorá vychádza z kritéria MAP (z angl. *maximum a posteriori*) a je definovaná [29]:

$$P_{CM}(w|h) = \frac{\epsilon C_A(h,w) + C_B(h,w)}{\epsilon C_A(h) + C_B(h)}, \tag{5.5}$$

kde $C_A(h,w)$ a $C_B(h,w)$ vyjadrujú frekvenčný výskyt slov podmienených ich históriou h a ϵ je parameter udávajúci silu (váhu) adaptácie prvého modelu.

Medzi najviac používané metódy kombinácie modelov, ktoré sa osvedčili aj pri tvorbe modelov slovenského jazyka patria metódy vychádzajúce z *lineárnej interpolácie* viacerých tematicky-zameraných modelov s následnou adaptáciou využívajúcu metódy minimalizácie perplexity modelu na *množine odložených dát* napr. pomocou algoritmu EM (z angl. *expectation-maximization*) [37, 82]. Formálne je možné takýto model interpolácie opísať:

$$P_{LI}(w|h) = \sum_i \lambda_i P_i(w|h), \text{ pričom } \lambda_i \geq 0, \text{ a zároveň } \sum_i \lambda_i = 1, \tag{5.6}$$

kde λ_i udáva interpolačnú váhu čiastkového modelu.

Okrem štandardne zvažovanej lineárnej interpolácie je možné kombinovať modely aj v logaritmickej oblasti použitím *log-lineárnej interpolácie* alebo pomocou *zovšeobecnenej lineárnej interpolácie*, ktorá spája výhody spájania početností a interpolácie modelov v jednom algoritme. Okrem interpolačných schém existuje aj rad diskriminatívnych metód založených na maximálnej entropii (z angl. *maximum entropy*) [66, 29].

5.6 Prerezávanie modelu

Keďže jazykové modely v systémoch na rozpoznávanie plynulej reči sú trénované typicky na stovkách miliónov, resp. miliárd slov, nekomprimovaný jazykový model v praxi nadobúda porovnateľnú veľkosť s množstvom dát, na ktorých je trénovaný. Táto kladie vysoké pamäťové nároky na jeho uchovávanie a proces dekódovania sa tak stáva časovo príliš náročným. Cieľom techník *prerezávania* (z angl. *pruning*) je redukovať veľkosť n-gramového modelu odstránením explicitných odhadov pravdepodobností z modelu, s ohľadom na čo najmenšiu degradáciu v perplexite modelu a presnosti systému na automatické rozpoznávanie reči. Technika prerezávania môže byť tak použitá pri návrhu kompaktného a účinného jazykového modelu [75].

Z najznámejších algoritmov prerezávania modelu možno spomenúť *metódu absolútneho prahu početnosti* (z angl. *count cut-off*), ktorá vychádza z predpokladu, že v jazykovom modeli sa vyskytuje veľké množstvo javov, ktoré sa pri trénovaní vyskytovali v textovom korpuse len raz alebo dvakrát a nemajú výrazný vplyv na celkovú kvalitu modelu jazyka. Vylúčením takýchto n-gramov z modelu môžeme významne redukovať pamäťové nároky výslednej aplikácie. Avšak kvôli bohatej morfológii a neusporiadanosti slovenského jazyka, daný spôsob prerezávania nebol príliš vhodný, preto bolo potrebné sa zamerať na oveľa sofistikovanejšie riešenia. K týmto možno zaradiť *metódu váženého rozdielu* (z angl. *weighted difference method*) a *prerezávanie založené na relatívnej entropii* (z angl. *relative entropy-based pruning*) [84]. V oboch prípadoch sa vychádza z predpokladu, že ak sa určitý n-gram v jazykovom modeli nevyskytuje, model tento odhad pravdepodobnosti nahrádza odhadom pravdepodobnosti vyhladeného $(n-1)$-gramu namiesto pôvodného odhadu. Ak takýto odhad vyhladenej pravdepodobnosti je veľmi blízky pôvodnému odhadu, nie je potrebné tento pôvodný odhad uchovávať. Ako vhodné kritérium na určenie takejto odchýlky sa javí napr. minimalizácia vzdialenosti pomocou *Kullbackovej-Leiblerovej vzdialenosti* použitá v metóde založenej na relatívnej entropii [75, 36].

Je vhodné poznamenať, že po aplikácii jednej z techník prerezávania vyhladeného jazykového modelu, je nutné prepočítať ústupové váhy a odhady pravdepodobností n-gramov v celom jazykovom modeli.

5.7 Pokročilé metódy modelovania jazyka

Jedným z hlavných nedostatkov n-gramových jazykových modelov, ktorý vychádza práve z ich podstaty je skutočnosť, že pri výpočte podmienených pravdepodobností neberú do úvahy celú svoju históriu, ale táto je obmedzená na

$(n-1)$ slov. S tým súvisí aj druhý nedostatok, a to tzv. *vnútorný rozptyl*, resp. *nedostatočnosť trénovacích dát*, ktorý narastá práve s použitým rádom n-gramového modelu. Pokročilé metódy, ktoré zohľadňujú ďalšie dodatočné závislosti medzi slovami a snažia sa tieto nedostatky takýmto spôsobom eliminovať budú opísané v nasledujúcich častiach.

5.7.1 Modely založené na triedach slov

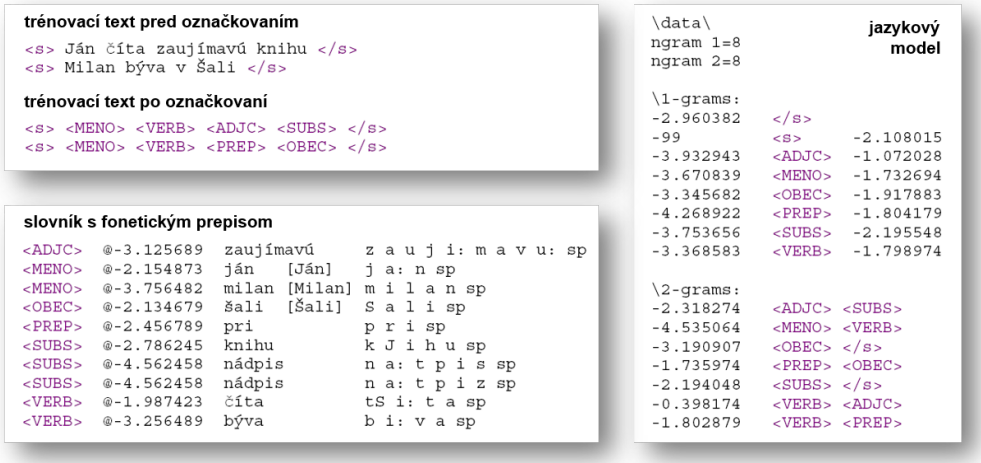
Modely založené na triedach slov (z angl. *class-based models*) sú pomerne široko uplatňované v rôznych aplikáciách zameraných na rozpoznávanie reči a spracovanie prirodzeného jazyka. Narozdiel od štandardných n-gramových modelov, na triedach založené modely eliminujú problém nedostatočnosti trénovacích dát zhlukovaním slov do slovných tried, a tým redukujú celkový počet parametrov modelu s cieľom efektívne vyhladiť a redukovať veľkosť jazykového modelu. Potom všetky slová umiestnené v jednej triede zdieľajú rovnaké štatistické závislosti vzhľadom k svojmu okoliu.

Existuje niekoľko rôznych spôsobov ako zhlukovať slov do tried. Prvotné techniky zvažovali *syntakticky-*, *sémanticky-* alebo *morfologicky-založené triedy* alebo *dátami-riadené prístupy zhlukovania* slov do tried [62]. Tieto boli odvodené zvyčajne od slovných lém a morfologických značiek, ktoré boli priradené slovám pomocou tzv. *morfologického analyzátora*, navrhnutého alebo prispôsobeného pre daný jazyk. V súčasnosti sa však využívajú oveľa priamočiarejšie prístupy, ktoré nevyužívajú komplexné algoritmy zjednoznačnenia spôsobu priradenia morfologickej značky, ale *štatistické metódy* identifikujúce *kmene*, či *koncovky slov* reprezentujúce *morfologicky-založený prístup zhlukovania slov do tried*.

Vo všeobecnosti modely založené na slovných triedach zhlukujú slová do ekvivalentných tried a modelujú jazyk ako súčin dvoch podmienených pravdepodobností nasledovne [37]:

$$P_{CL}(w|h) = \prod_{c \in C} P(w_i|c_i)P(c_i|c_{i-n+1} \dots c_{i-1}), \tag{5.7}$$

kde $P(w_i|c_i)$ je pravdepodobnosť slova w_i v triede c_i a $P(c_i|c_{i-n+1} \dots c_{i-1})$ je podmienená n-gramová pravdepodobnosť medzi triedami slov. Použitím tohto prístupu môžeme efektívne redukovať počet parametrov v jazykovom modeli (počet n-gramov, veľkosť modelu a pod.), pretože počet tried je vždy menší než celkový počet slov v slovníku a pravdepodobnosť slov, ktoré nesú podobné vlastnosti alebo zákonitosti v jazyku, je akumulovaná v rámci príslúchajúcej triedy [76]. Ukážka predspracovania textového korpusu, slovníka výslovnosti s fonetickým prepisom a vytvoreného jazykového modelu odvodeného od slovných tried je zobrazená na Obr. 5.5.



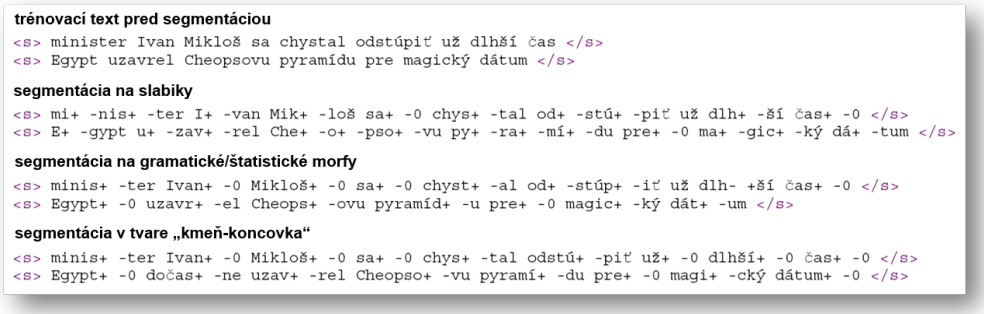
Obr. 5.5: Príklad modelovania jazyka pomocou slovných tried

Experimentálne výsledky modelovania slovenského jazyka pomocou slovných tried odvodených od slovných kmeňov a koncoviek opísané v [76] ukazujú mierne zlepšenie presnosti rozpoznávania reči oproti štandardným modelom približne o 5% v relatívnej miere a sú určitým kompromisom medzi modelmi založenými na slovných druhoch a lemách a štandardnými modelmi, čo do počtu tried, percenta tvarov OOV slov, či perplexity modelu.

5.7.2 Morfémové modely

Morfémové modely (z angl. *morph-based models* alebo *subword-based models*) riešia problém nedostatočnosti trénovacích dát dekompozíciou slov do menších subslovných jednotiek reči. Takýto spôsob reprezentácie slov výrazne redukuje veľkosť slovníka a počet mimoslovníkových tvarov slov, keďže počet všetkých možných morf je vždy menší než počet všetkých možných slov v danom jazyku a priemerný výskyt týchto subslovných jednotiek v trénovacom korpuse je vždy väčší než priemerný výskyt slov v tom istom korpuse. Vo všeobecnosti, morfy môžu byť tvorené slabikami, predponami, slovnými kmeňmi, koreňmi, príponami, alebo koncovkami slov, príp. inými plnovýznamovými časťami slova. Výber vhodných subslovných jednotiek pri modelovaní jazyka potom záleží od viacerých parametrov, napr. od priemernej dĺžky slov v jazyku, hraníc ohybu a spôsobu tvorenia slov, od ich výslovnosti, či vnútroslovného resp. medzislovného pokrytia [79].

Oblasť modelovania jazyka pomocou subslovných jednotiek je v súčasnosti zameraná na využitie troch základných princípov reprezentácie morf v



Obr. 5.6: Príklad automatickej segmentácie slov v slovenskom jazyku

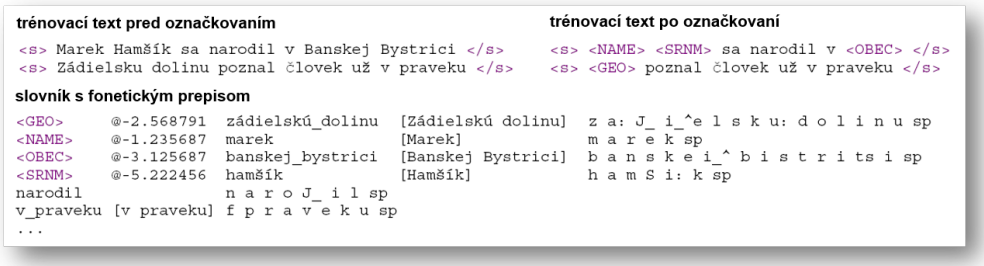
jazykovom modeli, a to: *modely založené na slabikách*, *modely založené na gramatických* alebo *štatistických morfách* a *modely v tvare slovného kmeňa a koncovky*, ktoré dekomponujú slová len na dve časti a sú pomerne často využívané pri modelovaní flektívnych jazykov. Rozkladom slov len na dve časti sa výrazne zvýši aj prediktívna schopnosť jazykového modelu oproti predchádzajúcim prístupom, pri ktorých dĺžka histórie morf v rámci jedného slova nie je pre n-gramový model všeobecne nikdy známa a ten môže ich zreťazením vygenerovať množstvo gramaticky nesprávnych tvarov. Naviac sú určitým kompromisom medzi veľmi krátkymi segmentami slova, akými sú slabiky a medzi celými slovami. Krátke segmenty tiež vyžadujú n-gramy vyššieho rádu, s ktorými vzrastá aj veľkosť a perplexita výsledného modelu [37, 79]. Ukážka automatickej segmentácie slov na slabiky, gramatické, resp. štatistické morfy a slovné kmene a koncovky je zobrazená na Obr. 5.6.

Modely v tvare slovného kmeňa a koncovky boli neskôr použité aj pri modelovaní slovenského jazyka [79]. V rámci tejto úlohy bol vytvorený aj vlastný algoritmus na automatickú segmentáciu slov na kmene a koncovky, ktorý zvažuje delenie slov nielen na morfematických, ale aj na fonetických (tzv. morfo-fonologických) hraniciach slov. Experimentálne výsledky modelovania slovenského jazyka pomocou morfémových modelov ukazujú výraznú redukciu počtu OOV tvarov a perplexity modelov, približne o tretinu s miernou degradáciou presnosti rozpoznávania reči (relatívne 2 až 3%). Avšak pri použití malých slovníkov (do 25 tisíc) sa ukazuje výhoda použitia morfémových modelov najmä pre pamäťovo-nenáročné aplikácie rozpoznávacieho systému s porovnateľnými výsledkami.

5.7.3 Modely s viacslovnými výrazmi

Pomerne častým a nežiaducim javom v systémoch na automatické rozpoznávanie reči je nesprávne rozpoznanie krátkych jednoslabičných slov. Tieto sú často ignorované (rozpoznané ako šum) alebo zlúčené so slovom, ktoré im predchádza alebo za nimi nasleduje. Podobný prípad nastáva aj pri slovách susedných, ktoré pri vyslovovaní môžu byť zlúčené do jedného slova, alebo naopak jedno slovo môže byť rozdelené na dve alebo viacero foneticky príbuzných slov. Z toho dôvodu je vhodné pri štatistickom modelovaní jazyka upriať pozornosť na tzv. *viacslovné výrazy* (z angl. *multiword expressions*), v jazykovede označované tiež ako ustálené slovné spojenia.

Jednou z výhod použitia viacslovných výrazov pri modelovaní jazyka je aj skutočnosť, že ich začlenením do slovníka, a tým aj do jazykového modelu, vzrastá rád n-gramu. Takýmto spôsobom je možné jednoducho pokryť aj okolie väčšie než okolie jedného slova. Ďalším dôvodom vhodnosti použitia viacslovných výrazov je aj jav zvaný *vzájomná medzislovná výslovnosť* (z angl. *cross-word pronunciation*), kedy niektoré priľahlé slová sú vyslovované inak v odlišnom kontexte, najmä v prípade spodobovania resp. zdvojovania hlások na hranici slov. Potom zavedením viacslovných výrazov do slovníka dokážeme jednoducho eliminovať aj celkový počet alternatívnych výslovností, a tým eliminovať chyby spôsobené zlou výslovnosťou rečníka [78].



Obr. 5.7: Spôsob reprezentácie viacslovných výrazov pri modelovaní jazyka

Viacslovné výrazy je možné použiť aj pri tvorbe *modelov názvoslovných pomenovaní* (z angl. *named entities*), akými sú napr. názvy miest a obcí, ulíc miest, názvy organizácií a inštitúcií alebo pri modelovaní cudzojazyčných odborných termínov, či rôznych iných ustálených slovných spojení často vyskytujúcich sa v jazyku. Takýto spôsob reprezentácie viacslovných výrazov v trénovacom korpuse a slovníku výslovnosti je zobrazený na Obr. 5.7.

5.7.4 Faktorované modely

Faktorované modely (z angl. *factored models*) predstavujú zovšeobecnený prípad morfologicky- a morfémovo-založených modelov [37]. Vo faktorovanom modeli sú slová reprezentované formou vektora, ktorý obsahuje $k = 1 : K$ nezávislých faktorov, čiže platí, že $w_i = f_i^1, f_i^2, \ldots, f_i^K$. Jednotlivé faktory tak môžu obsahovať rôzne črty slova, napr. morfologické značky, lemy, kmene, či korene slov a iné syntaktické znalosti, resp. závislosti. Vo všeobecnosti, postupnosť slov dĺžky N môže byť transformovaná na postupnosť K paralelných faktorov, označených $f_1^{1:K}, f_2^{1:K}, \ldots, f_N^{1:K}$. Potom trigramový jazykový model môže byť aproximovaný pomocou reprezentácie tejto submnožiny faktorov:

$$P_{FM}(w|h) \approx \prod_{i=3}^{N} P(f_i^{1:K}|f_{i-1}^{1:K}, f_{i-2}^{1:K}). \tag{5.8}$$

Každé slovo teraz závisí nielen od svojej histórie, ale aj na dodatočných paralelných vlastnostiach jednotlivých faktorov. Takáto reprezentácia môže byť neskôr použitá pri efektívnom vyhladzovaní n-gramov pomocou tzv. *zovšeobecnenej paralelnej ústupovej schémy* a to najmä v prípade, ak ich pôvodný odhad pravdepodobnosti je príliš malý.

Je vhodné poznamenať, že prípad dvojfaktorových modelov si možno predstaviť ako štandardný na triedach založený jazykový model.

Aj keď faktorované modely disponujú viacerými výhodami obzvlášť pri modelovaní vysoko-flektívnych jazykov, ich aplikácia v rečovo-založených systémoch prebiehajúcich v reálnom čase vzhľadom na ich extrémne vysoké pamäťové a časové nároky v procese trénovania jazykového modelu, ako aj v procese dekódovania vyslovenej postupnosti slov, je častokrát obmedzená.

5.7.5 Dynamické modely

Pomocou *dynamických modelov* je možné podchytiť do pozorovania históriou vzdialenejšie závislosti medzi slovami, ktoré n-gramové modely nezahrňujú.

Príkladom môže byť prípad dvojice slov, ktoré spolu úzko súvisia, pričom nenasledujú hneď za sebou, ale s určitým nepravidelným odstupom. Potom možno sledovať ich závislosť a priamo počas rozpoznávania zvyšovať pravdepodobnosť takýchto dvojíc. V tomto prípade hovoríme o *modeloch založených na spúšťačoch* (z angl. *trigger models*) [37].

Špeciálnym prípadom modelov založených na spúšťačoch sú *modely s krátkodobou pamäťou* (z angl. *cache models*), ktoré vychádzajú z predpokladu, že slová, ktoré sa vyskytli v nedávnej minulosti, sa môžu vyskytnúť so zvýšenou pravdepodobnosťou v blízkej budúcnosti opäť [37]. Takéto modely uchovávajú v pamäti slová s cieľom vytvorenia tzv. „lokálneho“ jazykového modelu,

ktorý je možné použiť napr. pri často sa vyskytujúcich konverzačných javoch alebo pri korekcii textu počas samotného rozpoznávania [62].

Osobitným prípadom dynamických modelov sú tzv. *adaptívne modely*, ktoré dokážu dynamicky meniť svoje parametre, t.j. veľkosť slovníka, konverzačnú tému a pod. vzhľadom na prebiehajúcu konverzáciu [62].

5.7.6 Tematicky-zamerané a zmiešané modely

V praxi sa adaptívne modely realizujú pomocou viacerých *tematicky-založených modelov* (z angl. *topic-based models*). Avšak niektoré konverzačné témy môžu spadať do viacerých tematických oblastí. Potom je nutné takto vytvorené modely spájať do tzv. *tematicky-zmiešaných modelov* (z angl. *topic-mixture models*). Pri tvorbe tematických modelov sa potom často využívajú metódy na zhlukovanie textových dokumentov založené prevažne na *skrytej Dirichletovej alokácii* (z angl. *latent Dirichlet allocation*) alebo na *skrytej sémantickej indexácii* (z angl. *latent semantic indexing*) [82].

Tematicke modely sa často používajú aj v doménovo-orientovaných úlohách rozpoznávania reči, napr. pri návrhu systému na prepis lekárskych nálezov, či súdnych pojednávaní. V tomto prípade je vhodné využiť celú trénovaciu množinu textových dát a parametre jazykového modelu prispôsobiť na určitú úzko-špecifickú oblasť rozpoznávania reči (na množinu odložených dát) pomocou jednej z metód určených na adaptáciu jazykových modelov [80].

5.7.7 Možnostné modely

Najnovšie prístupy v oblasti modelovania jazyka, ktoré sa snažia vysporiadať s problémom nedostatočnosti trénovacích dát bez zbytočného uchovávania rozsiahlych korpusov textových dát, sú zamerané na *rozširovanie štatistík* (z angl. *data augmentation*) n-gramových modelov buď na úrovni trénovacích dát, alebo priamo na úrovni modelu, aktualizáciou hodnôt odhadu pravdepodobnosti slov počas samotného trénovania. Na rozširovanie štatistík jazykových modelov sa v súčasnosti často využívajú rôzne *vyhľadávače* (z angl. *search-engines*) umožňujúce získať okrem samotného n-gramu aj jeho frekvenčný výskyt (pozri Obr. 5.8) priamo na sieti Internet (napr. Google, Yahoo, Bing a pod.). Modely, ktoré využívajú takýto spôsob aktualizácie hodnôt odhadov pravdepodobnosti málopočetných či mimodoménových n-gramov sa nazývajú *možnostné modely* (z angl. *possibilistic models*) [81].

Obr. 5.8: Porovnanie frekvenčných výskytov spojenia „*Cheopsova pyramída*“

5.7.8 Modely pre vložené pauzy a dysfluentné javy

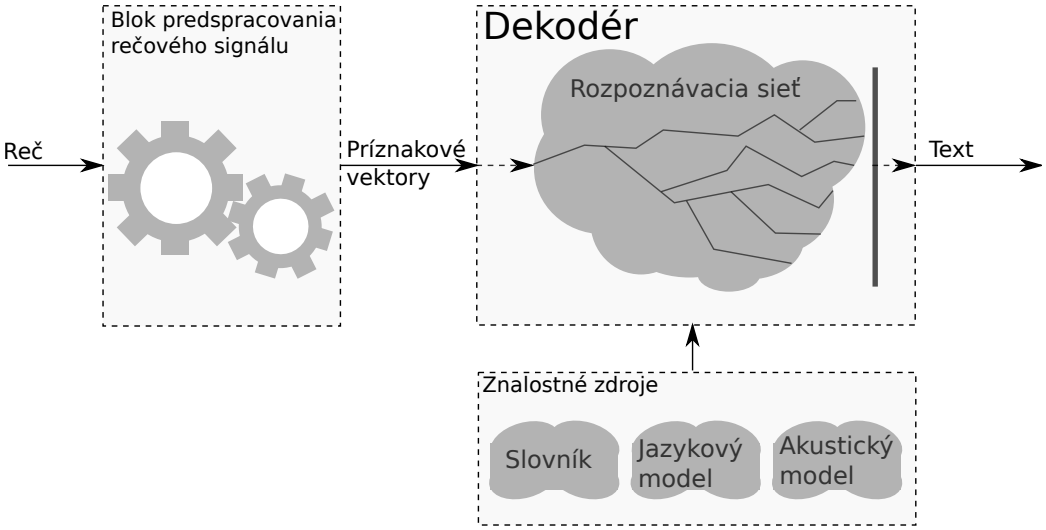
Pri tvorbe jazykových modelov použiteľných v interaktívnych rečovo-založených systémoch prebiehajúcich v reálnom čase je často nutné sa vysporiadať aj s rôznymi mimorečovými prejavmi, ktoré pochádzajú priamo od rečníka a sú spôsobené najmä zlou výslovnosťou, nevhodnou artikuláciou, či nedokonalosťou rečového prejavu.

Modely dysfluentných javov (z angl. *models with disfluent speech*) sa preto snažia podchytiť a zahrnúť do jazykového modelu rôzne suprasegmentálne javy obsiahnuté v rečovom prejave, ako napr. zaváhanie, zajakávanie, opakovanie fráz, skomolenie slov, či časté používanie rôznych typov *vložených páuz* (z angl. *filled pauses*) [77]. Tieto dysfluencie často vo vysokej miere vplývajú aj na celkovú chybovosť rozpoznávacieho systému ako celku. Odhad relatívnej pravdepodobnosti týchto javov sa potom získava zväčša z prepisov rečových nahrávok, ktoré sa používajú pri trénovaní akustického modelu. Vhodným výberom a správnou reprezentáciou vybraných dysfluentných javov v slovníku výslovnosti a jazykovom modeli je potom možné dosiahnuť výrazné zlepšenie presnosti rozpoznávania reči, ako to bolo ukázané v [77] pri modelovaní slovenského jazyka v systéme na automatický prepis parlamentných debát.

6

Automatické rozpoznávanie reči

Automatické rozpoznávanie reči (ARR) je proces, ktorý umožňuje počítaču identifikovať slová vyslovené človekom. Takýto systém ARR sa dá využiť ako základ systémov na automatický prepis zvukových nahrávok, na konštrukciu diktovacích systémov, prípadne systémov na riadenie pomocou hlasových povelov. Systém ARR je možné rozdeliť na dva základné bloky (viď Obr. 6.1).



Obr. 6.1: Základná štruktúra rozpoznávača reči

Prvým z nich je blok predspracovania rečového signálu, ktorý ho transformuje na sled príznakových vektorov. Tento sled príznakových vektorov čo najvernejšie predstavuje vstupný rečový signál a zvyšuje celkovú robustnosť systému ARR voči externému rušeniu [30].

Druhým je blok, ktorý vykonáva skutočné rozpoznávanie reči, teda transformáciu rečového signálu na text. Problém rozpoznávania reči je možné definovať ako odhad postupnosti slov, ktorú vyslovil rečník. Rečník *zakódoval* postupnosť slov do akustických pozorovaní, ktoré sú po predspracovaní signálu predstavované príznakovými vektormi a cieľom je *dekódovanie* originálnej postupnosti slov aj napriek externému rušeniu alebo variácií v *kódovaní*. Tento blok sa nazýva *dekodér* a táto kapitola sa zaoberá dekódovacími technikami v ARR.

Prvé systémy ARR boli založené na intuitívnom riešení *porovnávaním so vzormi* (v angl. Template Matching) [70]. Významným predstaviteľom tejto techniky je metóda *dynamického deformovania času* (v angl. Dynamic Time Warping, skr. DTW). Základom tejto metódy sú vzory slov (príkazov), s ktorými sa porovnávajú vstupné dáta a hľadá sa najväčšia zhoda. Tento systém ARR sa však ťažko rozširuje na rozpoznávanie plynulej reči a dlhých viet.

Riešenie dekódovania reči v oblastiach rozpoznávania plynulej reči s veľkým slovníkom[1] (v angl. Large Vocabulary Continuous Speech Recognition, skr. LVCSR) prichádza v aplikovaní stochastických metód [32]. Táto technika používa stochastické modely na reprezentáciu akustickej a jazykovej stránky reči, ktorú rozpoznáva. Tieto modely patria medzi *znalostné zdroje* ARR o rozpoznávanej reči. Znalostné zdroje sa využívajú často v podobe slovníka, akustického a jazykového modelu. Úlohu dekódovania reči si je možné predstaviť ako hľadanie takej postupnosti slov, u ktorých spojenie akustických a jazykových modelov najlepšie charakterizuje vstupnú postupnosť akustických pozorovaní. Nech W je postupnosť slov a nech O je postupnosť vstupných akustických pozorovaní, prislúchajúcich postupnosti vyslovených slov, potom rozpoznávanie reči je možné definovať ako

$$\hat{W} = \underset{W}{\operatorname{argmax}}\, P(W|O,\Theta), \tag{6.1}$$

kde $P(W|O,\Theta)$ je pravdepodobnosť postupnosti slov $W = \{w_1, w_2, ..., w_n\}$, ak bolo prijatých T akustických pozorovaní $O = \{o_1, o_2, ...o_T\}$, pričom parametre ARR boli Θ. Priamy výpočet rovnice (6.1) nie je možný pre LVCSR, ale pomocou Bayesovho pravidla túto rovnicu môžeme prepísať nasledovne

$$\hat{W} = \underset{W}{\operatorname{argmax}} \left\{ \frac{P(O|W,\Theta_a)P(W|\Theta_l)}{P(O|\Theta)} \right\}. \tag{6.2}$$

Pretože $P(O|\Theta)$ je konštanta, vzhľadom na maximalizáciu vzťahu (6.2) je ju možné zanedbať:

$$\hat{W} = \underset{W}{\operatorname{argmax}}\, P(O|W,\Theta_a)P(W|\Theta_l), \tag{6.3}$$

[1] Viac ako 100k slov.

kde $P(W|\Theta_l)$ je apriórna pravdepodobnosť, že bola vyslovená postupnosť slov W, pričom parametre Θ_l sú parametrami jazykového modelu. $P(O|W,\Theta_a)$ je podmienená pravdepodobnosť generovania akustických pozorovaní O, ak bola vyslovená postupnosť slov W, pričom Θ_a sú parametre akustického modelu. Vyhľadávanie správneho spojenia modelov jednotlivých slov sa vykonáva pomocou prehľadávacích (dekódovacích) algoritmov a na základe prehľadávacieho priestoru založeného na konečných stavových metódach. Prehľadávací priestor je vo väčšine systémov ARR tvorený akustickým, jazykovým modelom a slovníkom. Jednotlivé znalostné zdroje tvoria vrstvy prehľadávacieho priestoru, ktoré sú podľa výberu prehľadávacieho algoritmu viac alebo menej prepojené. Tvoria hierarchické usporiadanie, čo znamená, že každé slovo v jazykovom modeli je možné nahradiť postupnosťou sub-slovných jednotiek akustického modelu na základe slovníka. Teda jazykový model obsahuje slová a im priradené pravdepodobnosti výskytu, akustický model jednotlivé subslovné akustické jednotky a s nimi súvisiace pravdepodobnostné ohodnotenie. Nakoniec slovník obsahuje zoznam slov a ich fonetický prepis, teda prepojenie medzi jazykovým a akustickým modelom. Slovník môže obsahovať aj viac alternatívnych prepisov pre jedno slovo.

Dekodér predstavuje vhodné spojenie reprezentácie prehľadávacieho algoritmu a prehľadávacieho priestoru. Ideálny dekodér by mal spĺňať niekoľko podmienok:

- *Efektívnosť* – Výpočtové nároky dekodéra by mali byť čo najmenšie v porovnaní s úlohou, ktorú vykonáva. Napríklad pri diktovacích úlohách by mal byť dekodér schopný pracovať v lepšom ako reálnom čase, aby nezaostával za rečníkom v prepise reči.

- *Presnosť* – Ideálny dekodér nájde vždy bezchybnú výstupnú postupnosť slov na základe vstupnej reči. Vo väčšine prípadov sa to ale zaručiť nedá a pri samotnom dekódovaní (prehľadávaní priestoru) nastávajú chyby. Napriek tomu by mal dekodér zabezpečiť čo najmenší výskyt týchto chýb.

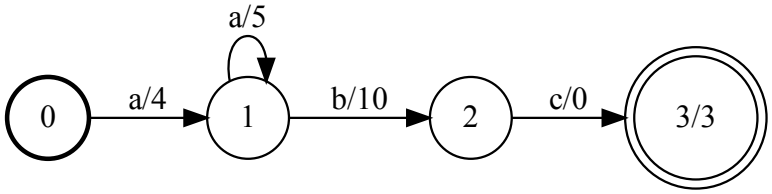
- *Škálovateľnosť* – Nárast výpočtových nárokov po pridaní zložitejších znalostných zdrojov by mal byť ekvivalentný s nárastom presnosti systému. Napríklad pri vložení nových slov do slovníka by mal dekodér zabezpečiť ich správne rozpoznávanie.

- *Univerzálnosť* – Ideálny dekodér by mal umožňovať pridávanie rôznych obmedzení a znalostných zdrojov priamo do prehľadávacieho algoritmu bez väčšieho obmedzenia jeho efektívnosti.

[1]Je dostupný ihneď po ukončení prejavu rečníka.

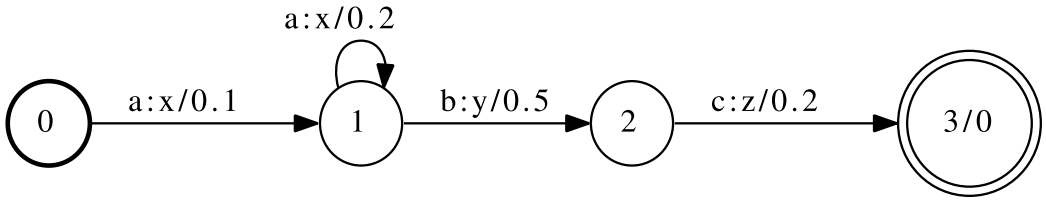
6.1 Konečná stavová reprezentácia prehľadávacieho priestoru

Súčasný stav v metódach ARR tvoria metódy založené na vážených konečných stavových metódach, teda na konečných stavových automatoch (v angl. Weighted Finite State Machine, skr. WFSM). Spojením znalostných zdrojov do celého rozpoznávacieho (stavového) priestoru získame rozpoznávaciu sieť, na ktorú je potom možné aplikovať prehľadávací algoritmus na nájdenie najpravdepodobnejšej postupnosti slov. Klasickým príkladom WFSM je konečný stavový akceptor (Obr. 6.2) (v angl. Weighted Finite-State Acceptor, skr. WFSA). Ide o automat s množinou stavov prepojených prechodmi, na ktorých sa môžu nachádzať symboly, ktoré automat akceptuje a váhy, ktoré jednotlivým symbolom priraďuje. Taktiež symboly môžu byť tvorené GMM v prípade reprezentácie akustického modelu.



Obr. 6.2: Príklad váženého konečného stavového akceptora (WFSA)

Čoraz väčšiu popularitu získavajú konečné vážené konečné stavové transducery (Obr. 6.3) (v angl. Weighted Finite-State Transducer, skr. WFST), ktoré zovšeobecňujú funkciu WFSA tým, že obsahujú dvojicu symbolov; vstupnú, ktorú akceptujú a výstupnú, ktorú generujú. [28, 3, 37].



Obr. 6.3: Príklad váženého konečného stavového transducera (WFST)

Keďže rýchlosť prehľadávacieho (dekódovacieho) algoritmu závisí od veľkosti výsledného prehľadávacieho priestoru, snažíme sa o vytvorenie optimalizovanej rozpoznávacej siete. Čím väčší je stavový automat jednotlivých

zdrojov tým väčší bude výsledný prehľadávací priestor. Reprezentácie prehľadávacieho priestoru sú rozdelené podľa spôsobu vytvárania.

- *Statická* rozpoznávacia sieť – Priestor je pripravený na použitie ihneď, nie je nutná jeho konštrukcia počas behu rozpoznávania reči, možnosť použitia optimalizačných techník na jej zmenšenie a tak urýchlenie následného dekódovacieho procesu. Nevýhodou sú výššie nároky na pamäť počas jej konštrukcie pretože zahŕňa všetky informácie zo znalostných zdrojov a je nutné vybudovať kompletnú štruktúru prehľadávacieho priestoru.

- *Dynamická* rozpoznávacia sieť – Nižšie pamäťové nároky, ktoré sú ale nahradené vyšším nárokom na výšší výpočtový výkon pre dekódovanie keďže potrebné časti rozpoznávacej siete sa vytvárajú počas tohto procesu.

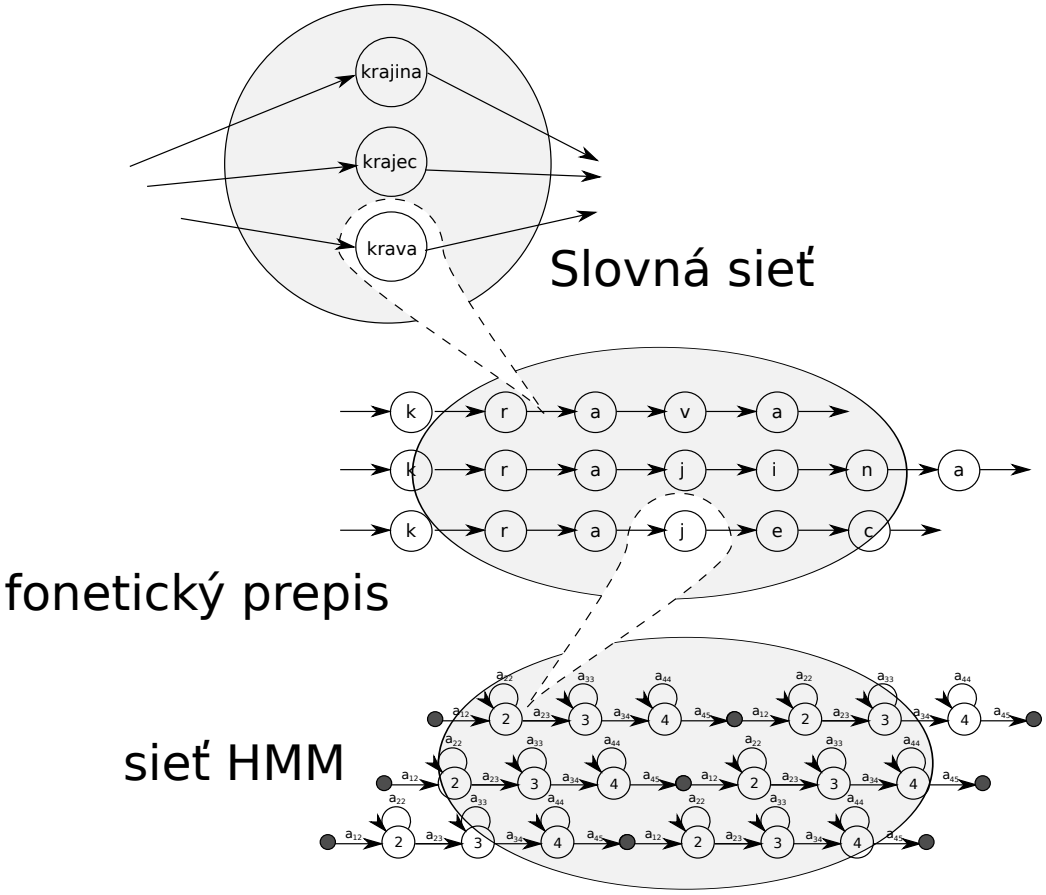
6.1.1 Prehľadávací priestor s WFSA

Používa sa substitučná metóda kde slovo z jazykového modelu je nahradzované jeho výslovnosťou podľa slovníka, ktorá je následne rozšírená o medzislovné - trifónové jednotky, ktoré sú ďalej predstavované stavmi HMM (Obr. 6.4).

Výsledný prehľadávací priestor je možné zmenšiť zdieľaním rovnakých začiatkov slov slovníka. Preto často prehľadávací priestor pri metóde WFSA nadobúda tvar *lexikálneho stromu* (viď Obr. 6.5). Jednotlivé váhy vyplývajúce z aplikovania jazykového modelu sú potom rozprestreté touto štruktúrou. Tejto metóde hovoríme *technika nazerania dopredu* (z angl. Look-Ahead). Význam tejto techniky je v tom, že znižuje chybovosť systému pri aplikovaní heuristických metód počas prehľadávania [53, 54].

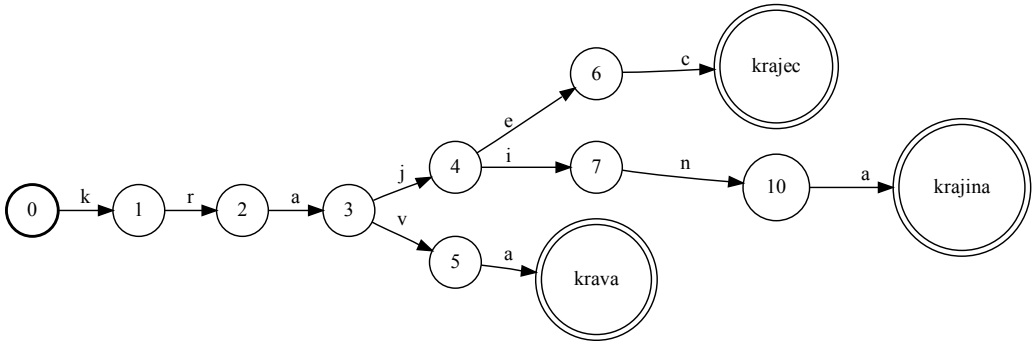
Reprezentácia statického prehľadávacieho priestoru môže byť reprezentovaná pomocou *statických lexikálnych stromov*. Rozpoznávacia sieť je tvorená hlavne štruktúrou jazykového modelu, kde napríklad pre bigramový model (viď Obr. 6.6) sa prehľadávací priestor skladá z unigramovej časti tvoreného kompletným lexikálnym stromom a bigramových častí tvorených redukovanými stromami iba pre danú históriu slov. Ostatná história sa modeluje pomocou ústupovej váhy [4].

Dynamická reprezentácia priestoru môže zodpovedať vytváraniu nového lexikálneho stromu v pamäti pre každé ukončené slovo z predchádzajúceho stromu (viď Obr. 6.7). Ide o *lexikálne stromy podmienené históriou slov*, kde je nutné vytvárať pre každé nové začaté slovo nový kompletný lexikálny strom.

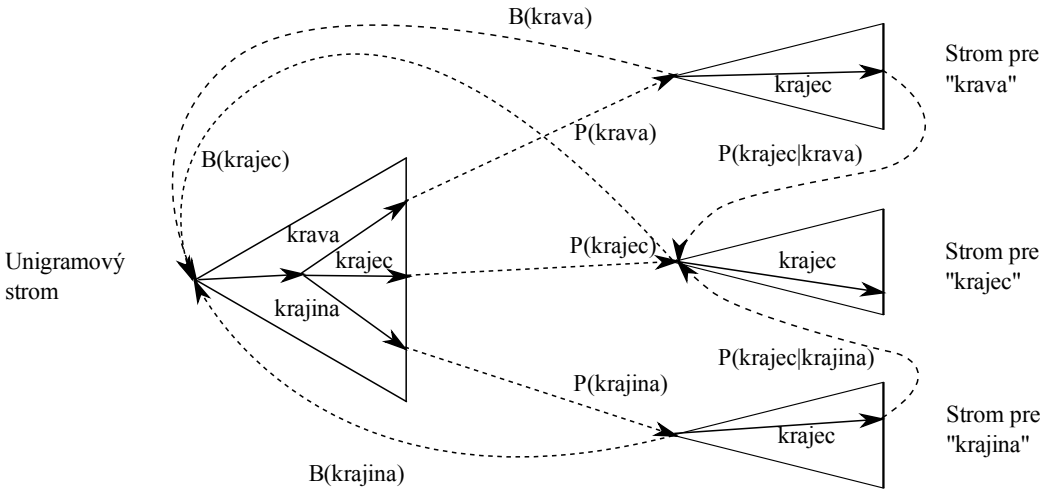


Obr. 6.4: Rozpoznávacia sieť

Vhodnou úpravou predchádzajúcej metódy je *reentrantný lexikálny strom*, ktorý je tvorený len jedným kompletným lexikálnym stromom. Ak prehľadávanie dorazí na koniec lexikálneho stromu vstupuje opäť na jeho začiatok. Pri prehľadávaní takéhoto reentrantného priestoru nastáva stret rôznych hypotéz slov (histórií slov) v jednom stave priestoru. Preto je nutné, aby každý stav priestoru obsahoval zásobník udržiavajúci prehľadávanie pre jednotlivé histórie slov. Na obmedzenie veľkosti týchto zásobníkov sa používajú rôzne heuristické metódy. Nevýhodou je, že nie je možné použiť techniku nazerania dopredu na hodnotenie jazykovým modelom, keďže jednotlivé stavy môžu predstavovať rôznu históriu slov jazykového modelu. Väčšinou sa teda používa na rozprestretie pravdepodobnosti aspoň unigramová pravdepodobnosť.



Obr. 6.5: Príklad lexikálneho stromu

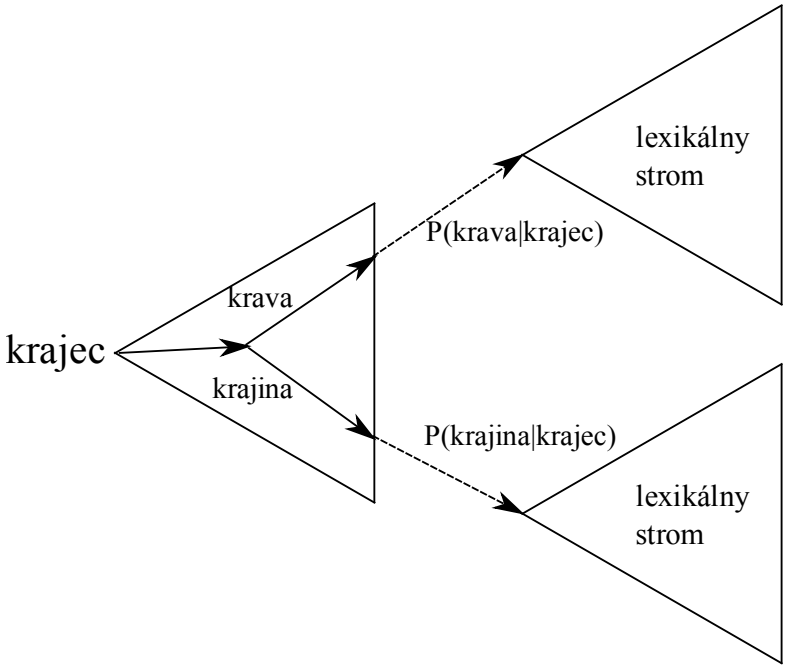


Obr. 6.6: Príklad prehľadávania pomocou statických lexikálnych stromov

6.1.2 Prehľadávací priestor s WFST

Všetky znalostné zdroje je možné reprezentovať pomocou vážených konečných stavových transducerov [59, 48, 11]. Keďže vážené transducery sa dajú doslova preložiť i ako prekladové automaty, reprezentácia jednotlivých znalostných zdrojov bude predstavovať článok rozpoznávania reči, ktorý je vlastne prekladovým automatom medzi určitými úrovňami, od akustických pozorovaní až ku postupnosti rozpoznaných slov. Či už ide o dynamickú alebo statickú konštrukciu rozpoznávacej siete, je ju možno chápať ako kaskádu jednotlivých transducerov znalostných zdrojov (viď Obr. 6.8), Táto postupnosť jednotlivých prekladov sa nazýva *rozpoznávacia kaskáda* [49, 58].

Jednotlivé znalostné zdroje je nutné reprezentovať pomocou WFST transducera. *Transducer akustického modelu* (H) je ekvivalentný skrytému Marko-

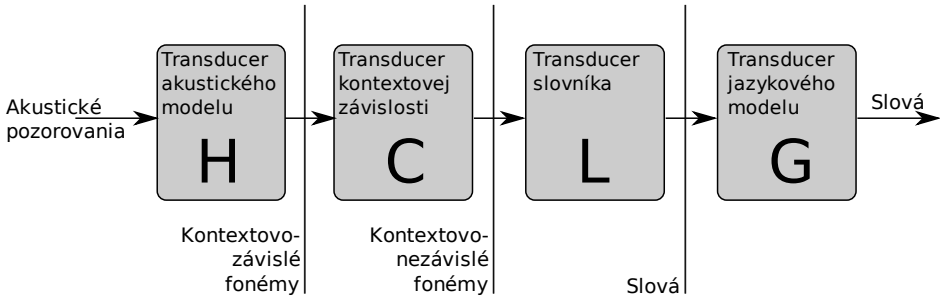


Obr. 6.7: Príklad prehľadávania pomocou lexikálnych stromov podmienených históriou slov

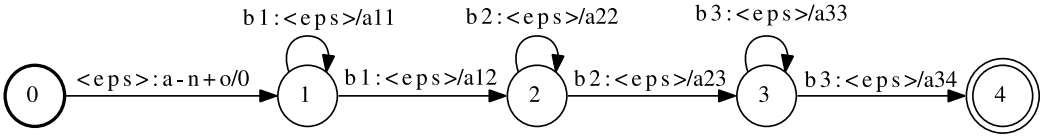
vovému modelu. Jeho vstupnými symbolmi sú jednotlivé GMM a výstupnými symbolmi kontextovo závislé akustické jednotky (trifóny) (Obr. 6.9). V prípade ak prechod neakceptuje alebo negeneruje žiaden symbol, ale nim iba prechádza sa používa prázdny symbol $< eps >$.

Na preloženie kontextovo závislých foném na kontextovo nezávislé (teda z trifón na fonémy) potrebujeme *transducer kontextovej závislosti C* (viď príklad na Obr. 6.10) [51]. Teda vstupné symboly sú trifóny a výstupné symboly sú tvorené fonémami. Ďalej kontextovo nezávislé fonémy je možné pomocou *transducera slovníka* (L) preložiť sekvencie foném na slová (Obr. 6.11). Teda tu budú vstupnými symbolmi fonémy a vystupnými slová. Keďže je týmto automatom akceptovaná postupnosť viacerých symbolov a generovaný je iba jeden symbol (slovo), vyžaduje si tento automat prázdne $< eps >$ výstupné symboly na svojich prechodoch. Podledným je *transducer jazykového modelu*, ktorý prekladá slová opäť na slová avšak dôležité sú jeho váhy na prechodoch, ktoré obsahuje jazykový model (viď Obr. 6.12)(G).

Takýmto postupným prekladom pomocou jednotlivých transducerov je možné rozpoznávanie reči. Avšak existujú dobre definované operácie, ktoré umožňujú spojenie týchto transducerov do jedného kompaktného, ktorý bude



Obr. 6.8: Kaskáda transducerov rozpoznávania reči



Obr. 6.9: Skrytý Markovov model (HMM)

vykonávať tento preklad v jednom kroku [49]:

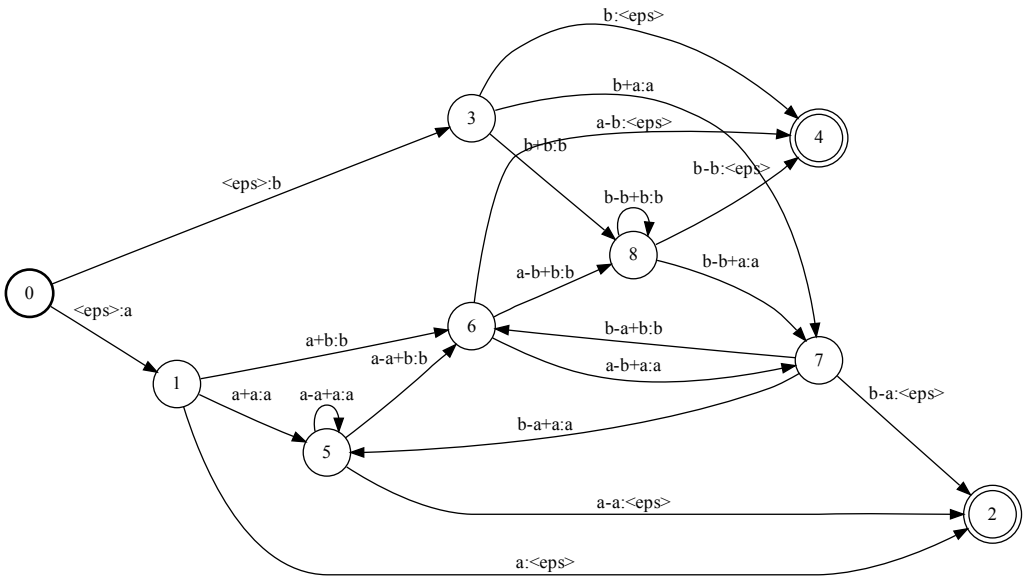
$$N = H \circ C \circ L \circ G, \tag{6.4}$$

kde N predstavuje výslednú rozpoznávaciu sieť. Takto zostavenú rozpoznávaciu sieť je potom možné následne optimalizovať pomocou nasledujúcich operácií:

- *Determinizácia* (*det*) – deterministický automat obsahuje pre danú vstupnú postupnosť symbolov práve jednu úspešnú cestu. Je si to možno predstaviť ako spájanie ciest s rovnakými akceptovanými symbolmi ako v prípade lexikálnych stromov.

- *Minimalizácia* (*min*) – dochádza k zlučovaniu *ekvivalentných* stavov [47]. Minimalizovať je možné deterministický automat. Dva stavy deterministického automatu sú *ekvivalentné*, ak majú prechody ku koncovému stavu označené presne tou istou postupnosťou symbolov a zároveň majú to isté ohodnotenie každej postupnosti vrátane koncovej váhy koncového stavu. Jedná sa o zdieľanie koncov ciest s rovnakými symbolmi.

- *Odstránenie prázdnych symbolov* (Π_ϵ) – zahŕňa odstránenie všetkých prechodov s prázdnym symbolom. Operácia odstránenia prechodov s prázdnymi symbolmi sa používa hlavne pre odstránenie redundancie a taktiež hlavnej príčiny nedeterminizmu automatu. Má vplyv na rýchlosť dekódovacieho procesu.



Obr. 6.10: Transducer kontextovej závislosti pre dve kontextovo nezávislé fonémy 'a' a 'b'

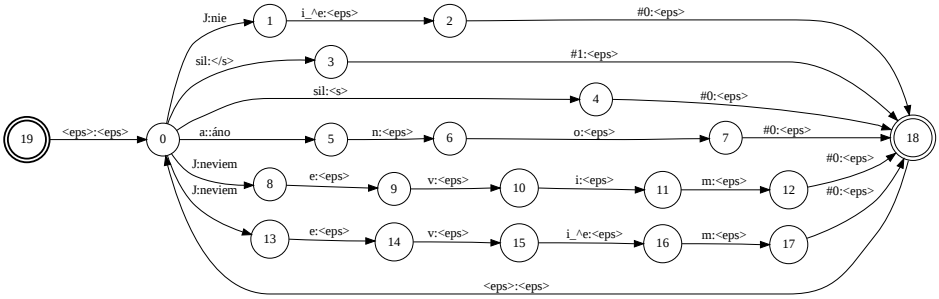
Po aplikácii všetkých operácii teda výsledná rozpoznávacia sieť je daná [57, 50, 2]:

$$N = \Pi_\epsilon(\min(\det(H \circ \det(C \circ \det(L \circ G))))). \tag{6.5}$$

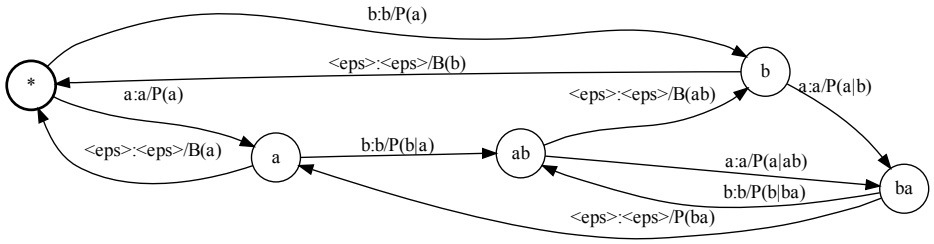
6.2 Prehľadávacie algoritmy

Úlohu hľadania správneho spojenia modelov jednotlivých slov charakterizujúcich vstupné akustické pozorovania uľahčujú metódy prevzaté z umelej inteligencie [68]. Ide o vytvorenie *prehľadávacieho priestoru* tvoreného znalostnými zdrojmi a obsahujúceho všetky prípustné kombinácie modelov postupnosti slov. Priestor sa tiež nazýva *stavovým*, pretože je tvorený stavmi a prechodmi medzi nimi. Ide o konečnú stavovú reprezentáciu prehľadávacieho priestoru. Priestor je prehľadávaný vhodne zvoleným *prehľadávacím (dekódovacím)* algoritmom, ktorý postupuje na základe vstupných akustických pozorovaní a odoziev jednotlivých modelov. Základnými technikami sú *prehľadávanie do šírky* (v angl. Depth-first search) a *prehľadávanie do hĺbky* (v angl. Breath-first search). Tieto techniky patria medzi skupinu tzv. *slepých* prehľadávaní.

Pre systémy LVCSR môžu byť modely rozsiahle a rovnako aj prehľadávací priestor. Algoritmy prehľadávania založené na hrubej sile (v angl. brute-



Obr. 6.11: Transducer slovníka



Obr. 6.12: Transducer jazykového modelu pre dve slová 'a' a 'b'

force) prehľadávajú celý priestor, a preto nie sú vhodné pre LVCSR. V tomto prípade sa na obmedzenie prehľadávania použijú rôzne heuristické metódy. Jedná sa o *informované* prehľadávanie, medzi ktoré patrí prehľadávanie typu *prvý-najlepší* (v angl. Best-first search). Úlohou je obmedziť počet stavov, ktoré sa následne navštívia využitím heuristickej informácie, pričom funkcia je daná

$$f(q) = g(q) + h(q), \tag{6.6}$$

kde $f(q)$ je hodnota hodnotiacej funkcie v stave q prehľadávacieho priestoru, $g(q)$ je doterajšia hodnota cesty k stavu q, $h(q)$ je heuristická funkcia a vyjadruje cenu cesty ku koncovému stavu. Ak najlepšia cesta je cesta s minimálnou cenou alebo ohodnotením, potom sa budú navštevovať stavy s minimálnou hodnotou heuristickej hodnotiacej funkcie. Ceny doterajšej a zostatkovej cesty nie sú známe a používajú sa len ich odhady $\hat{g}(q)$ a $\hat{h}(q)$. V prípade ak použitá heuristická funkcia vedie k optimálnemu riešeniu, nájdeniu tej najpravdepodobnejšej postupnosti stavov, jedná sa potom o *prípustnú* heuristickú informáciu a o $A*$ (v angl. A-star) prehľadávanie. Aby použitie heuristickej funkcie viedlo k optimálnemu riešeniu je nutné splnenie podmienky

$$\hat{h}(q) \leq h(q), \tag{6.7}$$

teda odhad hodnoty zostatkovej cesty musí byť menší ako je skutočný v prí-

pade ak uvažujeme, že najlepšia cesta je tá s minimálnym ohodnotením.

Obľúbeným spôsobom implementácie jednotlivých techník je použitie zásobníka, do ktorého sa ukladajú nenavštívené stavy prehľadávacieho priestoru. Pri prehľadávaní sa potom stavy navštevujú v poradí, v akom sú uložené v zásobníku. Jednotlivé techniky prehľadávania sa líšia spôsobom ukladania a triedenia stavov v zásobníku. Pri prehľadávaní do hĺbky sa nenavštívené stavy ukladajú na začiatok zásobníka, a teda pri prehľadávaní sa vždy sleduje aktuálna cesta. Pri prehľadávaní do šírky sa stavy ukladajú na koniec zásobníka, a teda prehľadávací algoritmus predtým ako postúpi ďalej, najprv navštívi všetky stavy v rovnakej hĺbke. Pri použití heuristických metód ide o usporiadanie zásobníka tak, aby bola sledovaná cesta s najväčšou pravdepodobnosťou úspechu.

Tak ako bolo uvedené, jednotlivé znalostné zdroje sú založené na konečnej stavovej reprezentácii a spolu vytvárajú stavový prehľadávací priestor. Z pohľadu vrstiev prehľadávacieho priestoru je možné aj dekódovanie rozdeliť a definovať osobitne na vrstve akustického a na vrstve jazykového modelu. Na základe takto definovaného dekódovania existujú dva typy dekódovacích techník, *časovo-synchrónne* a *časovo-asynchrónne*. Pre každú techniku existujú metódy pre ich urýchlenie využitím rôznych prípustných ale aj neprípustných heuristických techník.

6.2.1 Dekódovanie na úrovni akustického modelu

Pri dekódovaní na úrovni akustického modelu ide o výpočet jeho ohodnotenia pre danú postupnosť slov W, teda výpočet pravdepodobnosti $P(O|W,\Theta_a)$. Na tento výpočet sa používajú dve hodnotiace kritéria. Prvým z nich je MAP kritérium (v angl. Maximum A Posteriori), čo možno zapísať nasledovne

$$P(O|W,\Theta_a) = \sum_Q P(O,Q|W,\Theta_a), \tag{6.8}$$

kde Q je jednou z možných postupnosti stavov HMM, ktoré súčasne reprezentujú postupnosť slov W. Výpočet pravdepodobnosti na základe vzťahu (6.8) je najčastejšie aproximovaný *Viterbiho kritériom*

$$P(O|W,\Theta_a) \approx \max_Q P(O,Q|W,\Theta_a), \tag{6.9}$$

kde suma je nahradená operátorom max. Zásadný rozdiel medzi MAP a Viterbiho kritériom vo výpočte pravdepodobnosti $P(O|W,\Theta_a)$ je v uvažovaní postupnosti stavov. MAP kritérium uvažuje so všetkými možnými postupnosťami stavov tvoriace postupnosť slov W. Viterbiho kritérium uvažuje iba s najpravdepodobnejšou postupnosťou stavov.

6.2.2 Dekódovanie na úrovni jazykového modelu

Dekódovanie na úrovni jazykového modelu je definované ako nájdenie najpravdepodobnejšej postupnosti slov. Tak ako v prípade dekódovania na úrovni akustického modelu aj tu sa využívajú dve kritéria dekódovania a to MAP a Viterbiho kritérium. V prípade dekódovania pomocou kritéria MAP podľa vzťahu (6.10) je nutné vyšetrovať všetky možné usporiadania stavov akustického modelu reprezentujúcich tú istú postupnosť slov W a vykonať ich súčet. Teda ide o súčet tých istých pravdepodobností postupnosti slov cez všetky možné časové hranice vo vstupnom akustickom signále. Potom sa vyberie najpravdepodobnejšia postupnosť slov zo všetkých možných postupností. V prípade použitia Viterbiho kritéria podľa vzťahu (6.11) sa využívajú zlučovacie uzly WFSA reprezentácie jazykového modelu, v ktorých sa vyberá doterajšia najpravdepodobnejšia postupnosť slov. Výsledkom je teda hneď najpravdepodobnejšia postupnosť slov.

$$\begin{aligned}
\hat{W} &= \underset{W}{\operatorname{argmax}} \{P(O|W,\Theta_a)P(W|\Theta_l)\} \\
&= \underset{W}{\operatorname{argmax}} \left\{ \left[\sum_{Q} P(O,Q|W,\Theta_a) \right] P(W|\Theta_l) \right\} \qquad (6.10) \\
&\approx \underset{W}{\operatorname{argmax}} \left\{ \left[\max_{Q} P(O,Q|W,\Theta_a) \right] P(W|\Theta_l) \right\}. \qquad (6.11)
\end{aligned}$$

6.2.3 Časovo-synchrónne dekódovanie

Pri časovo-synchrónnom prehľadávaní prehľadávacieho priestoru sú vrstvy akustického modelu, jazykového modelu a slovníka úzko prepojené. Jednotlivé stavy jazykového modelu sú prepojené so stavmi akustického modelu pomocou slovníka do jednej komplexnej *rozpoznávacej siete*. Ako kritérium prehľadávania na úrovní akustického a jazykového modelu sa často používa Viterbiho kritérium, ktoré odhaľuje najpravdepodobnejšiu postupnosť slov podľa vzťahu (6.11). Teda s výhodou sa využívajú zlučovacie uzly (stavy) akustického a jazykového modelu a uplatňujú sa princípy *dynamického programovania*. Viterbiho prehľadávanie je časovo-synchrónny proces, čo znamená, že najprv ukončí spracovanie vstupných príznakových vektorov v čase t predtým ako postúpi na spracovanie vektorov v čase $t+1$. Podľa Viterbiho algoritmu sa na konci dekódovania vykoná na spätné odsledovanie najpravdepodobnejšej postupnosti stavov akustického modelu, z ktorých sa vyvodí najpravdepodobnejšia postupnosť slov. Viterbiho prehľadávanie je považované za prehľadávanie priestoru do šírky, s využitím dynamického

programovania, teda využitím zlučovacích stavov priestoru, v ktorých sa vyberá doterajšia najpravdepodobnejšia cesta. Pri Viterbiho dekódovaní ide o aproximáciu najpravdepodobnejšej postupnosti slov (postupnosti stavov jazykového modelu) najpravdepodobnejšou postupnosťou stavov akustického modelu, pričom jednotlivé hypotézy slov sa vyšetrujú paralelne.

Časovo-synchrónne prehľadávanie je tiež možné použiť na konštrukciu viacprechodového dekodéra, kde prvý prechod s jednoduchšími modelmi vytvorí možnosti výslednej postupnosti slov. Druhý prechod s adaptovanými akustickými modelmi alebo s detailnejším jazykovým modelom vykoná výber tej najpravdepodobnejšej postupnosti slov [5, 21].

V aplikáciách rozpoznávania reči pomocou Viterbiho aproximácie podľa vzťahu (6.11) sa s výhodou využíva algoritmus *podávania žetónov* (v angl. token-passing) [96]. Žetón je objekt, ktorý sa nachádza v stave prehľadávacieho priestoru a jeho úlohou je prechádzať medzi jednotlivými stavmi, pričom si zapamätáva cestu a cenu doterajšej prejdenej cesty. Cena žetónu je maximálna doterajšia pravdepodobnosť zložená z hodnotenia postupnosti HMM stavov končiaca v danom stave a čase a hodnotenia jazykovým modelom. Hlavnou časťou algoritmu cestovania žetónov je prechod žetónu z aktuálneho stavu do nasledujúcich stavov s každým prijatým vstupným akustickým pozorovaním. Ak dôjde ku stretu dvoch žetónov v tom istom stave a s tou istou históriou slov, ponechá sa iba žetón s najlepším ocenením, ostatné žetóny sa odstránia. Žetón v poslednom stave HMM obsahuje záznam o najpravdepodobnejšej postupnosti stavov, ktoré definujú zároveň výstupnú postupnosť rozpoznaných slov.

6.2.4 Časovo-asynchrónne dekódovanie

Pri časovo-asynchrónnom prehľadávaní sú jednotlivé vrstvy prehľadávacieho priestoru izolované. Hlavnou časťou prehľadávania je prehľadávanie na úrovni jazykového modelu a teda nejde o aproximáciu najpravdepodobnejšou postupnosťou stavov akustického modelu ako pri časovo-synchrónnom prehľadávaní. Na úrovni jazykového modelu je možné použiť dekódovacie kritérium MAP alebo Viterbi, ktorý berie do úvahy zlučovacie stavy jazykového modelu. Na úrovni akustického modelu sa využíva presnejšie MAP kritérium, pričom výhodou je tiež použitie dopredného algoritmu výpočtu ohodnotenia akustickým modelom.

Základom je použitie zásobníka (alebo Fibonacciho haldy), ktorý obsahuje hypotézy postupnosti slov zoradené podľa toho, ktorá hypotéza má najväčšiu pravdepodobnosť úspechu. Najpravdepodobnejšia hypotéza je následne vybraná zo zásobníka a rozšírená o ďalšie slovo a vložená späť. Zásobník je opäť preusporiadaný. Keďže sa vždy rozširuje hypotéza alebo skupina

hypotéz, ktorá je najpravdepodobnejšia, je prehľadávanie menej výpočtovo náročné ako pri časovo-synchrónnom prehľadávaní. Zásobník ale z rovnakého dôvodu obsahuje hypotézy s rôznou dĺžkou a teda aj im prislúcha rôzny počet spracovaných vstupných akustických pozorovaní. Z toho vyplýva, že jednotlivé hypotézy nemusia končiť v rovnakom čase ako to bolo v prípade časovo-synchrónneho prehľadávania. Použitie časovo-asynchrónneho prehľadávania má výhodu v použití aj všeobecnejších jazykových modelov a oddeleniu prehľadávania na úrovni jazykového a akustického modelu. Nevýhodou je ale obtiažné nájdenie vhodnej hodnotiacej funkcie na zoraďovanie zásobníka.

Zásobník je zoraďovaný pomocou *heuristickej hodnotiacej funkcie*. Keďže ide o porovnávanie nekompletných hypotéz postupnosti slov a z dôvodu zabezpečenia porovnateľnosti sa používa nasledovné zloženie heuristickej funkcie

$$\hat{f}(W) = \hat{g}(W_0^t) + \hat{h}(W_t^T), \tag{6.12}$$

kde $\hat{g}(W_0^t)$ predstavuje dopredné ohodnotenie hypotézy slov W od času 0 do času t a $\hat{h}(W_t^T)$ vyjadruje predpokladané ohodnotenie pre jej pokračovanie od času t do konca rečníkovho prejavu T. $\hat{h}(W_t^T)$ tvorí heuristickú informáciu. Ak je využitá heuristická informácia prípustná, teda nevyradí z prehľadávania najpravdepodobnejšiu postupnosť slov, potom sa jedna o A* prehľadávanie. Teda pri použití záporných logaritmov pravdepodobností platí, že ak $\hat{h}(W_t^T) < h(W_t^T)$ odhad zostatkovej ceny je menší ako skutočná cena do času T. V skutočnosti mnoho zásobníkových dekódovacích techník používa heuristické informácie, ktoré nie sú prípustné z dôvodu aplikovania rozpoznávania v reálnom čase.

Posledným spôsobom je viacprechodové prehľadávanie, kedy sa vykoná časovo-synchrónne prehľadávanie priestoru Viterbiho algoritmom pri použití jednoduchších akustických a jazykových modelov. Potom sa vykoná na získaných výsledkoch asynchrónne prehľadávanie [74].

6.2.5 Metódy urýchlenia dekódovania

Jednotlivé techniky dekódovania sú výpočtovo náročné pokiaľ sa jedná o LVCSR. Pri časovo-synchrónnom Viterbiho prehľadávaní je nutné použitie heuristických metód na obmedzenie počtu vyšetrovaných hypotéz postupnosti slov, či už na úrovni stavov HMM akustického modelu alebo na úrovni jazykového modelu. Pri časovo-asynchrónnom prehľadávaní ide o obmedzenie počtu hypotéz v zásobníku a taktiež o obmedzenie kandidátov slov na rozšírenie vybranej hypotézy (Ak slovník obsahuje V slov potom v každom časovom okamžiku vzniká V nových hypotéz). Aplikácia heuristických metód na obmedzenie prehľadávania sa nazýva *prerezávanie*. Často sa využívajú

nasledovné prerezávacie metódy.

Beam prerezávanie Niekedy je ťažké nájsť vhodnú heuristickú funkciu, pre ktorú platí spomínaná podmienka prípustnosti vyjadrená vzťahom (6.7), hlavne ak existuje iba málo alebo vôbec žiadna informácia o zostávajúcej ceste ku koncovému stavu. Prehľadávanie typu beam je jednoduchým riešením tohto problému, pričom ale nejde o prípustnú heuristickú informáciu a teda nevedie k optimálnemu riešeniu [30]. Úlohou prehľadávania typu beam je použitie doterajšieho ohodnotenia jednotlivých ciest, pričom ďalšie prehľadávanie sa bude vykonávať iba zo stavov q, ktoré v danom čase spĺňajú nasledujúcu podmienku

$$\hat{g}(q) = \hat{g}(q_{best}) + T \text{ kde } T > 0, \tag{6.13}$$

čo znamená, že z prehľadávania budú odstránené stavy, ktorých doterajšie ohodnotenie bude väčšie ako hodnotenie doterajšej najlepšej cesty končiacej v danom čase a stave q_{best}. Existuje niekoľko implementácii beam prerezávania, ktoré sa líšia od miesta ich aplikovania. Základnými sú:

- Prerezávanie na úrovni akustického modelu, kde ide o priame deaktivovanie stavov akustického modelu, ktorým vedie cesta pod hranicou beam.
- Prerezávanie na úrovni modelu fonémy, čo vychádza z podobného princípu ako na úrovni stavov s tým rozdielom, že sa vykonáva pri prechode z jedného modelu na ďalší.
- Prerezávanie na úrovni slov kde sa beam aplikuje pri prechode na ďalšie slovo.

Histogramové prerezávanie Beam prerezávanie udržuje prehľadávanie v úzkom zväzku. Napriek tomu dochádza ku zvýšeniu nárokov prehľadávania na dvojnásobok hlavne pri nerečových zvukoch alebo pri zaváhaní. V týchto prípadoch má viac ciest rozpoznávacím priestorom približne rovnaké ohodnotenie a teda spadajú do prahu beam prerezávania. Jednoduchým riešením je nastavenie hornej hranice aktívnych ciest (hypotéz). Ide teda o nastavenie prahu prerezávania tak, aby výsledkom bolo N_{best} najlepších ciest (hypotéz). Použitie histogramového prerezávania s dobre definovanou hranicou nevplýva na presnosť rozpoznávania reči [44].

Prerezávanie Fast-Match Táto technika sa využíva pri časovo-synchrónnom a aj časovo-asynchrónnom prehľadávaní. Základ tejto prerezávacej techniky súvisí s použitím aproximovaného akustického modelu, ktorý vyžaduje nižšie výpočtové nároky na výpočet ohodnotenia akustických pozorovaní. Pri časovo-synchrónnom prehľadávaní sa často používa v usporiadaní viacnásobného dekódovania, kde prvým prechodom cez dekodér sa vytvoria najpravdepodobnejšie hypotézy rozpoznaných slov, pričom sa použije aproximovaný alebo jednoduchší akustický model, často v spojitosti s jazykovým modelom. Pri ďalšom prechode dekodérom sa použijú detailnejšie alebo adaptované akustické modely. Pri časovo-asynchrónnom dekódovaní ide hlavne o obmedzenie kandidátov slov rozširujúce doterajšie hypotézy postupnosti slov [5, 21].

Prerezávanie deaktiváciou foném Pri tomto prerezávaní sa používa odhad pravdepodobnosti, že daná fonéma existuje v danom čase vo vstupnom signáli. Ak je táto pravdepodobnosť nižšia ako stanovená hranica, potom všetky slová, ktoré túto fonému obsahujú nebudú uvažované pri prehľadávaní stavového priestoru. Hranica sa vo väčšine prípadov nastavuje experimentálne. Odhad pravdepodobnosti výskytu fonémy v danom čase je možné odvodiť od klasického výpočtu ohodnotenia akustickým modelom, no používajú sa na tento účel hybridné systémy, ktoré zahrňujú akustické modely na báze HMM a neurónových sietí [64].

7

Spracovanie prirodzeného jazyka

S príchodom výpočtovej techniky sa stala potreba počítačového spracovania prirodzeného jazyka aktuálnou celosvetovou témou. Vedci sa po celom svete snažia podchytiť charakter takmer každého jazyka s cieľom zjednodušiť interakciu medzi ľuďmi a strojmi a komunikáciu medzi ľudmi samotnými. Oblasť spracovania prirodzeného jazyka zahŕňa širokú škálu disciplín, ako napr. vyhľadávanie informácií, štatistické modelovanie jazyka, strojový preklad, automatické porozumenie reči a podobne. Jednotlivé disciplíny však vo väčšine prípadov úzko súvisia, dopĺňajú sa, a pomocou nich je možné ľuďom uľahčiť prácu, štúdium, komunikáciu, či zábavu.

Počítače nám môžu lepšie porozumieť a pomáhať rozumieť našim emailom, našej práci a našej reči. Najväčšie firmy na svete vznikli vďaka spracovaniu prirodzeného jazyka a ďalšie rastú.

Nie je možné podať presnú definíciu spracovania prirodzeného jazyka. Z vedeckého hľadiska ide o kombináciu viacerých techník z oblasti:

- Teórie formálnych jazykov,
- Štatistiky,
- Umelej inteligencie,
- Lingvistiky,
- Psychológie.

Schopnosť strojovo spracovať texty vytvorené človekom otvára nové možnosti. Medzi typické úlohy riešené systémami pre spracovanie prirodzeného jazyka patrí:

- Monitoring médií: Čo všetko sa napíše alebo odvysiela v televízii alebo v rádiu o mojej spoločnosti?
- Získavanie informácií: Vyhľadávanie na webe napr. Vyhľadanie stránok súvisiacich s "natural language processing".
- Dolovanie v dátach: Aká je priemerná cena hamburgeru v Južnej Amerike?
- Porozumenie otázke: Vyhľadávací asisteni a chatboty.
- Fulltextové vyhľadávanie na internete alebo v textových databázach.
- Ohodnotenie sentimentu a detekcia nevhodných alebo podozrivých textových príspevkov: Facebook, LinkedIn
- Strojový preklad medzi rôznymi jazykmi.
- Detekcia spamu (nevyžiadanej pošty alebo správ).
- Cielenie reklamy na základe obsahu webstránky a identity používateľa.
- Dopĺňanie diakritiky: Častým javom pri komunikácii medzi ľudmi prebiehajúcej na sieti Internet je vysoký výskyt preklepov a chýbajúca diakritika. Hoci človeku to väčšinou pri porozumení správy nerobí problém, pri počítačovom spracovaní prirodzeného jazyka je potrebné nájsť vhodný spôsob pre rozlíšenie významu nejednoznačných zápisov na základe okolitého kontextu.

7.1 Neurčitosť v prirodzenom jazyku

Ľudský jazyk obsahuje veľa neurčitosti. To isté vieme povedať rôznymi spôsobmi a jedna výpoveď môže mať veľa významov, ktoré často nie je možné rozlíšiť ani podľa kontextu. Často pri komunikácii prenášame aj neverbálnu informáciu pomocou prízvuky a štýlu reči, giest, alebo výrazu tváre.

Príklady neurčitosti v jazyku sú:

- Homonymá: Práve sedím v škole. Nevyznám sa v občianskom práve. To auto stojí 10000 eúr. Auto stojí na kraji cesty.
- Synonymá: Išiel som do Bratislavy. Išiel som do Blavy.
- Neurčité poradie slov vo vete: Dnes je pekný deň. Pekný deň je dnes. Deň je dnes pekný.

- Neurčitý význam slov: Po tráve sa nechodí, po tráve sa smeje
- Novotvary a slangové výrazy: Vyggogli si to a potom to buchni na fejsbúk.
- Emócie a spoločenské konvencie: Pane! Pekne ste sa doriadil!
- Preklepy a brepty.

Z príkladov je zrejmé, že to isté slovo môže mať v rôznom kontexte rôzny význam a že jeden význam možno zapísať viacerými spôsobmi.

Počítačový jazyk je jednoznačný. Pre prácu s neurčitosťou potrebujeme špecializované metódy, ktoré sú schopné vyjadriť a redukovať nejednoznačnosť. Vhodným postupom je možné zmeniť prirodzený ľudský jazyk do podoby, ktorá je zrozumiteľná počítaču a vhodná pre automatické spracovanie. Tým je možné dosiahnuť, že počítač "porozumie" človeku a je schopný s ním komunikovať spôsobom, ktorý je človeku blízky. Proces odstránenia neurčitosti nazývame aj "disambiguácia". Z množiny možných spôsobov porozumenia algoritmus na základe kontextu vyberie ten, ktorý sa zdá najvhodnejší.

7.2 Príprava dát na spracovanie

Prvým krokom spracovania prirodzeného jazyka je vhodná metóda predspracovania. Textové dáta nie je možné spracovávať v podobe v akej boli vytvorené. Po získaní a uložení dát je potrebné transformovať ich do vhodnej podoby odstránením nepodstatných časti a zjednotením rôznych zápisov tej istej významovej jednotky.

Štatistické modelovanie si často vyžaduje manuálne vyznačenie sledovaných javov v databáze textov. Model zostavený na základe trénovacieho korpusu je potom schopný aj neznámym kontextom priradiť najpravdepodobnejší jav. Príkladom zaujímavého javu je význam slova vo vete, jeho gramatická funkcia alebo či slovo označuje konkrétnu vec alebo osobu (tzv. "pomenovaná entita").

Príprava na spracovanie dát v prirodzenom jazyku prebieha v týchto troch krokoch:

- Získanie textových dát: Textové dáta je potrebné zozbierať a konvertovať do vhodného formátu.
- Príprava textových dát: V texte sú identifikované významové jednotky (tzv. "tokeny"), ktorým je priradený jednotný zápis. Nepodstatné časti sú odstránené.

- Príprava trénovacieho korpusu: Označenie zaujívavých javov v texte a výber testovacej množiny.

7.2.1 Získanie textu

Kvalitný štatistický model si vyžaduje veľké množstvo dát. Niekedy sú textové dáta priamo dostupné v existujúcej databáze. Druhým spôsobom získania textových dát pre spracovanie je agent pre dolovanie textu, ktorý prechádza webové stránky a extrahuje z nich text a ostatné zaujímavé informácie. Získané dokumenty sú uložené v databáze.

Proces získavania textu pomocou agenta pozostáva z týchto krokov:

1. Návšteva zdroja, napr. webovej stránky.
2. Extrakcia textu a informácií o dokumente (napr. čas vytvorenia, autor, alebo zdroj).
3. Identifikácia duplicitných a nepodstatných častí dokumentu.
4. Identifikácia a uloženie odkazov na ďalšie zdroje.
5. Vloženie textu do databázy.

Výsledkom činnosti agenta pre získavanie textu je rozsiahla databáza textových dokumentov.

7.2.2 Príprava textu

Najdôležitejšou časťou predspracovania textu je jeho *tokenizácia*. Jej cieľom je identifikácia významových jednotiek jednotlivých slov a vetných hraníc. V tomto kroku sa tiež snažíme zjednotiť spôsob zápisu číslic, diakritiky, interpunkcie, akronymov, symbolov, skratiek a iných významových jednotiek.

Tokenizácia sa zvyčajne vykonáva postupnou aplikáciou vhodne zapísaných regulárnych výrazov, ktoré obsahujú pravidlá pre identifikáciu textových jednotiek, dôležitých pre ďalšie spracovanie. Nepodstatné časti textu, ktoré nie sú pokryté pravidlami, sú z textu vynechané. Identifikácia vetných hraníc je ďalej vykonávaná pomocou rozlíšenia významu bodky, jej disambiguáciou. Napr. v slovenských textoch môže byť bodka súčasťou označenia číselného poradia, skratky alebo e-mailovej alebo webovej adresy.

Na začiatku procesu identifikácie významových častí je vstupný reťazec porovnaný so všetkými pravidlami v zozname regulárnych výrazov. Pravidlo, ktoré vyhovuje najdlhšiemu textu, je vybraté a jeho zodpovedajúci text je

prepísaný podľa požiadaviek. Tento text je potom odstránený zo vstupného reťazca. Ak nevyhovuje žiadne pravidlo, vstupný reťazec je skrátený o jeden znak a prehľadávanie bázy pravidiel pokračuje. Výsledkom tokenizácie je text, kde sú textové jednotky oddelené medzerou a vety novým riadkom.

7.2.3 Príprava trénovacieho korpusu

Štatistické algoritmy napodobňujú činnosť človeka, ktorý je schopný na základe kontextu rozlíšiť neurčitý význam vety alebo dokumentu. Zaujímavé javy je preto potrebná vyznačiť ručne v trénovacej databáze. Trénovací korpus sa využije na zostavenie štatistického modelu. Vstupné dáta by mali mať v kontexte vyznačené javy, ktoré sú predmetom modelovania. Model je potom schopný rozlíšiť prítomnosť sledovaného javu na základe kontextu aj v prípade, že sa kontext nenachádzal v trénovacom korpuse.

V trénovacom korpuse automaticky alebo manuálne vyznačujeme javy, ktoré nás zaujímajú. Podľa riešenej ulohy môže ísť o:

- slovné druhy,
- vetné členy,
- vlastné podstatné mená (pomenované entity),
- významy slov,
- sentiment vo vete alebo v dokumente,
- kategória a téma dokumentu a iné.

Podľa spôsobu prípravy rozlišujeme tri druhy trénovacích korpusov.

- *Neoznačné korpusy*: Neoznačený korpus pozostáva z textov, ktoré obsahujú iba informáciu o hraniciah slov, viet a dokumentov. Tento druh korpusu je najjednoduchší na prípravu, ale zvyčajne je potrebné pripraviť veľké množstvo textu.
- *Automaticky označené korpusy*: Automaticky označené korpusy využívajú existujúci štatistický alebo pravidlový systém pre pridanie dodatočnej informácie ktoré bude predmetom modelovania v ďalšom kroku.
- *Manuálne označené korpusy*: Do textu je manuálne pridaná informácia. Tento druh korpusu je najzložitejší na prípravu, lebo si vyžaduje prácu experta, ktorý je zaškolený na správne označovanie sledovaného javu.

7.3 Štatistické modely neurčitosti

Pre prácu s neurčitosťou v spracovaní prirodzeného jazyka môžeme využiť, ľudskú znalosť vo forme pravidiel, štatistické informácie z trénovacích korpusov alebo kombináciu oboch prístupov.

Pravidlové systémy vedia na základe človekom vloženej informácie vedia vo forme slovníkov, formálnej gramatiky alebo regulárnych výrazov rozlíšiť neurčité prípady.

Tam kde to je možné, využívame pre anotáciu textu prístupy založené na štatistickom modelovaní.

7.3.1 Klasifikácia kontextov

V trénovacej databáze sú zvyčajne tokenom, vetám alebo dokumentom priradené značky ktoré hovoria o výskyte sledovaného javu (význam slova, téma dokumentu a pod.). Štatistický klasifikátor analyzuje trénovací korpus a je schopný priradiť najpravdepodobnejšiu značku aj takým kontextom, ktoré sa v trénovacej databáze nevyskytujú. Slovenčina sa vyznačuje relatívne voľným poradím slov vo vetách, vysokým počtom morfologických tvarov slov a gramatických výnimiek. Počet možných kontextov tak môže byť veľmi vysoký, a to sťažuje úlohu natrénovania čo možno najpresnejšieho štatistického klasifikátora.

Na zostavenie štatistického modelu kontextov sa najčastejšie využívajú modely založené na:

- maximálnej entropii,
- skryté Markovove modely,
- umelé neurónové siete,
- n-gramové modely,
- SVM (Support Vector Machine),
- CRF (Condidtional Random Fields).

Vstupom do trénovania štatistického klasifikátora je jeden kontext (napr. časť vety) a značka, označujúca sledovaný jav. Kontext je transformovaný na príznaky. Klasifikátor priradí spoločnému výskytu príznaku a značky určitú váhu. Množina váh potom tvorí model. Pri klasifikácii vie klasifikátor priradiť ľubovoľnému kontextu najpravdepodobnejšiu značku.

Morfologické značky sú jedným z najdôležitejších príznakov v spracovaní prirodzeného jazyka. Klasifikátor je založený na skrytom Markovovom modeli (z angl. "*hidden Markov model*", skr. HMM) druhého rádu a najpravdepodobnejšia postuponosť morfologických značiek je vyhľadávaná Viterbiho algoritmom.

Nami HMM klasifikátor pre morfologickú analýzu flektívnych jazykov sa skladá z nasledujúcich štyroch častí:

1. lexikón, ktorý navrhuje množinu možných značiek na základe slova alebo jeho koncovky;
2. model prechodov, ktorý vyjadruje pravdepodobnosť nasledujúcej značky na základe dvoch predchádzajúcich,
3. model pozorovaní, ktorý vyjadruje pravdepodobnosť slova na základe možnej značky;
4. a v prípade, že skúmané slovo sa nenachádza v trénovacej databáze, využije sa dodatočný model pozorovaní, ktorý vyjadruje pravdepodobnosť stavu na základe koncovky daného slova.

7.3.2 Modelovanie dokumentov

S narastajúcim množstvom textových dokumentov stiahnutých zo siete Internet a potrebou vytvárať čoraz presnejšie doménovo orientované interaktívne rečovo-založené systémy a rozhrania, sa vynorila otázka kategorizovať textové dáta nielen podľa adresy (URL) zdroja textu, odkiaľ daný textový dokument pochádza, ale aj na úrovni jeho obsahu. Navyše webová adresa zdroja textu nemusí byť hneď jednoznačným identifikátorom obsahu dokumentu, vychádzajúc tiež z predpokladu, že jeden dokument môže pojednávať o viacerých témach.

Pre modelovanie celých dokumentov za účelom vyhľadávania (information retrieval) sa najčastejšie využívajú metódy ktoré transformujú dokument do vektorového priestoru, ako napríklad:

- TF-IDF (term frequency-inverse dopcument frequency),
- LSI (latent semantic indexing),
- LDA (latent Dirichlet allocation).

Pri vyhľadávaní sa identifikujú dokumenty, ktoré ležia vo vektrovom priestore blízko zadanému dopytu. Blízkosť vo vektorovom priestore môže indikovať sémantickú podobnosť. Skupiny sémanticky blízkych dokumentom (dokumentov s príbuznou témou) je možné identifikovať metódami zhlukovania (nekontrolovaného učenia).

Na rozdiel od *metód zhlukovania*, kde dokumenty s využitím štatistických prístupov spájame do určitého počtu zhlukov, v ktorých tému vopred nepoznáme, pri *kategorizácii dokumentov* sa snažíme zadeliť dokumenty do dvoch alebo viacerých tried na základe ich minimálnej vzdialenosti, resp. sémantickej podobnosti, udávajúcej prienik slov alebo celých fráz medzi dokumentami. Kategorizácia textu má preto veľký význam pri návrhu a tvorbe robustných doménovo orientovaných systémov na automatické rozpoznávanie reči, ale aj v iných úlohách využívajúcich textové dáta ako zdroj informácií, napr. pri návrhu a vývoji interaktívnych rečovo-založených rozhraní. V oboch prípadoch je nutné identifikovať tému v danom zhluku, resp. triede, a to buď pomocou prístupov založených na extrakcii kľúčových slov, pravdepodobnostných prístupov založených na výpočte podobnosti dokumentov pomocou dištančných metrík, alebo ich kombináciou.

8

Riadenie dialógu

Technológia riadenia dialógu umožňuje počítačovým systémom byť účastníkmi dialógových interakcií a reagovať na rečový alebo multimodálny vstup od používateľa na základe aktuálneho stavu dialógu, histórie dialógu a na základe dialógových modelov. Jednotka riadenia dialógu je zväčša centrálnym prvkom rozhraní medzi človekom a strojom, ktoré sú založené na rečovej komunikácií.

Hlavnou úlohou systému riadenia dialógu je riadiť dialógovú interakciu, teda rozhodovať o nasledujúcom kroku, ktorým bude systém reagovať na vstupy od používateľa. Riadenie dialógu zahŕňa tiež riadenie ostatných komponentom interaktívneho systému, ako sú systém rozpoznávania reči, syntézy reči z textu a pod., čo vedie k realizácii dialógovej interakcie.

8.1 Dialóg ako spôsob interakcie medzi človekom a strojom

Pre rečové dialógové systémy (RDS) je charakteristická interakcia používateľa so systémom vo forme hlasového dialógu. Dialóg je prirodzeným médiom pre výmenu informácií medzi jeho účastníkmi a umožňuje riešiť kooperatívne úlohy vo vzájomnej interakcii.

V rečových dialógových systémoch, ako to bolo spomenuté v predchádzajúcej časti, je za dialógové funkcie zodpovedný blok riadenia dialógu (dialógový manažér - DM). Dialógový manažér riadi celkovú interakciu medzi systémom a používateľom a môžeme ho považovať za kľúčový prvok RDS. Hlavnou úlohou DM je nájsť vhodnú reakciu systému na používateľský vstup, ktorá korešponduje s aktuálnym stavom systému a históriou interakcie. Na tento proces sa môžeme pozerať ako na zobrazenie akcie používateľa do akcie systému, alebo ako na zobrazenie jedného stavu systému do iného stavu

systému.

Vo všeobecnosti sa na manažment dialógu môžeme pozerať ako na systém pozostávajúci z dvoch častí: samotného riadenia dialógu a modelovania (kontextu) dialógu. Prvá časť sa zaoberá riadením toku dialógu. Druhá časť modeluje informácie, ktoré dialógový manažér používa na interpretáciu vstupu používateľa a informovanie o rozhodnutiach samotného riadiaceho komponentu [33], na modelovanie správania sa používateľa a systému a správu histórie interakcie.

Problematika riadenia dialógu je v súčasnosti pomerne dobre prepracovanou oblasťou výskumu [6, 7, 9, 13, 17]. Doposiaľ bolo navrhnutých viacero sofistikovaných, hoci nie veľmi úspešných, textovo-orientovaných systémov. Hoci komunikácia hovorenou rečou je veľmi odlišná od textovej, mnohé princípy z týchto systémov možno použiť aj v rečovo-orientovaných systémoch. Nie je však realistické predpokladať, že textovo-orientované dialógové systémy možno pridaním automatického rozpoznávania a syntézy reči jednoducho konvertovať na rečovo-orientované systémy.

V nasledujúcom texte budú prezentované dva hlavné aspekty riadenia dialógu – stratégia a iniciatíva v dialógu a mechanizmy striedania iniciatívy.

8.1.1 Stratégia a iniciatíva v dialógu

Jedným z kľúčových aspektov manažmentu dialógu je spôsob správy iniciatívy. Stratégie manažmentu dialógu sa vo všeobecnosti môžu rozdeľovať na stratégie [46, 88]:

- s iniciatívou na strane systému
- s iniciatívou na strane používateľa
- so zmiešanou iniciatívou

Stratégia riadenia dialógu s iniciatívou na strane systému

Dialóg s iniciatívou na strane systému prebieha tak, že systém kladie otázky používateľovi, ktorý poskytne informácie požadované systémom. Po získaní potrebných informácií dôjde k zmene stavu dialógu a k vygenerovaniu príslúchajúcej akcie blokom riadenia dialógu. Ako najdôležitejšie sa v procese návrhu dialógu javí návrh stratégie manažmentu opravy chýb v dialógu a návrh správnych otázok, ktoré sú kladené používateľovi. Žiadna z týchto úloh nie je triviálna.

Táto stratégia riadenia dialógu je vhodná pre dobre predikovateľné dialógy s jednoznačným tokom dialógu, čo umožňuje používať napr. kontextovonezávislé rečové gramatiky pre rozpoznávanie vyjadrení používateľa. Každý stav dialógu potom môže mať prispôsobenú gramatiku, ktorá zúži rozpoznávací priestor a tým prispeje k účinnejšiemu rozpoznávaniu.

Ďalšou výhodou stratégie dialógu s iniciatívou na strane systému je, že systém vedie používateľa, aby mu pomohol dosiahnuť jeho cieľ. Pretože otázky kladie systém, používateľ si môže byť istý, že budú dodržané všetky potrebné kroky. To vyvoláva u používateľa pocit komfortu a zamedzuje dezorientácii. Tento prístup je zvlášť vhodný pre začínajúcich používateľov, ktorí nemajú poznatky o režimoch prevádzky systému.

Nevýhodou tejto stratégie je ťažkopádnosť interakcie so skúsenými používateľmi. Zvlášť ak systém predpokladá, že v každom úseku dialógu je vymenená iba jedna položka informácie. Dialóg potom trvá dlho a napreduje pomaly. To sa dá kompenzovať prístupom, kedy je systém schopný extrahovať viacero informácií z jedného vyjadrenia používateľa. Týmto spôsobom môže skúsený používateľ vynechávať niektoré úseky dialógu použitím zložitejších vyjadrení. Zvyčajne to nekladie zvýšené nároky na zložitosť manažmentu dialógu, ale zložitejšími sú rečové gramatiky.

Stratégia s iniciatívou na strane systému nie je vhodná pre všetky typy úloh. Najvhodnejšia je pre dobre definované úlohy s primeranou sekvenčnosťou, kde systém potrebuje poznať určité čiastkové informácie, aby vytvoril požiadavku pre databázu, resp. podobnú vyhľadávaciu úlohu. Typickým príkladom sú požiadavky do databázy cestovných poriadkov vlakov a autobusov a pod. Úlohy s viac otvoreným koncom, akými sú e-mailové aplikácie, je obtiažne modelovať ako sekvenciu úloh a preto pre nich nie je vhodné použiť túto stratégiu.

Stratégia riadenia dialógu s iniciatívou na strane používateľa

Táto stratégia predpokladá, že používateľ vie čo má robiť a ako naviazať interakciu so systémom. V tejto všeobecnej forme systém čaká na vstup od používateľa a následne reaguje primeranou akciou. Systémy s iniciatívou na strane používateľa sú často označované ako systémy príkazov a riadenia, aj keď použitý jazyk môže byť viac sofistikovaný. V každom prípade je používateľ aktívnym účastníkom v týchto systémoch, čo sa týka iniciatívy v dialógu. Výhodou systémov s iniciatívou na strane používateľa je, že skúsený používateľ je schopný používať systém voľne, bez spôsobu daného systémom. Je to prirodzené v tzv. úlohách s otvoreným koncom, ktoré majú nezávislé podúlohy.

Bežným argumentom, ktorý favorizuje systémy s iniciatívou na strane po-

užívateľa je, že ak schopnosti systému porozumieť prirodzenému jazyku sú na vyššej úrovni, systém môže porozumieť vyjadreniam používateľa v bežnom jazyku, čo približuje ich vzájomnú interakciu k interakcii medzi ľuďmi navzájom. V skutočnosti má tento model obmedzené reálne použitie, pretože používanie neobmedzeného rozsahu jazyka vedie k problémom s ich rozpoznaním a interpretáciou. Ďalším aspektom, ktorý limituje reálne nasadenie takýchto systémov sú vysoké nároky na kognitívne schopnosti používateľa a jeho znalosť riešenej úlohy, ako aj možností systému.

Stratégia riadenia dialógu so zmiešanou iniciatívou

Stratégia dialógu so zmiešanou iniciatívou predpokladá, že iniciatíva môže byť na oboch stranách (na strane používateľa aj na strane systému). Používateľ má slobodu pri preberaní iniciatívy, ale v prípade vyskytnutia sa problému pri komunikácii, alebo ak to úloha vyžaduje, systém môže prebrať iniciatívu a viesť interakciu.

Aplikácie môžu využívať stratégiu zmiešanej iniciatívy rôznymi spôsobmi. Napr. úloha môže formovať hierarchiu, v ktorej podúlohy môžu používať odlišné stratégie dialógu. Alternatívne môže systém adaptovať štýl interakcie, aby vyhovoval jednotlivým používateľom alebo situácii. To sa môže realizovať napr. na začiatku voľbou stratégie s iniciatívou systému a neskôr môže používateľ preberať väčšiu iniciatívu v závislosti ako sa naučí spolupracovať so systémom. Podobne, ak používateľ mal problém so stratégiou s iniciatívou na strane používateľa, systém prevezme iniciatívu až kým interakcia nespĺňa očakávania.

Stratégia zmiešanej iniciatívy môže byť zložitou čo sa týka modelovania aj použitia. Tento prístup kladie zvlášť veľké nároky na bloky rozpoznávania reči a interpretácie významu. Pre úspešné dekódovanie vyjadrení systému je nutné použiť zložitejšie jazykové modely a algoritmy extrakcie významu.

Často je táto stratégia riadenia dialógu chápaná limitovane, tak, že pod pojmom iniciatíva používateľa sa myslí, len možnosť poskytnutia viacerých informácií v jednom vyjadrení používateľa, ktoré navyše neboli explicitne očakávané v danom kroku dialógu. Toto zjednodušenie je prijímané s ohľadom na skutočnú zložitosť systémov so skutočnou zmiešanou iniciatívou, kde používateľ môže taktiež klásť otázky systému, alebo odpovedať otázkou.

8.2 Prístupy k riadenie dialógu

Vzhľadom na to, že riadenie dialógu je zložitou a komplexnou úlohou, môžeme konštatovať, že existuje veľké množstvo prístupov a metód manažovania

interakcie. Podľa [94], je možné rozdeliť metódy riadenia dialógu do nasledujúcich skupín:

- systémy založené na konečno-stavovom automate resp. na tzv. dialógovej gramatike
- prístupy založené na realizácií tzv. plánu
- kolaboratívne prístupy
- prístupy založené na štatistickom modelovaní dialógu

K základnému rozdeleniu je ešte potrebné doplniť, z nášho pohľadu, dva veľmi dôležité prístupy k riadeniu dialógu, ktoré boli základom kvázi priemyselných štandardov v tejto oblasti:

- formularovo-orientované systémy
- systémy na báze jazykov na popis dialógu - DDL

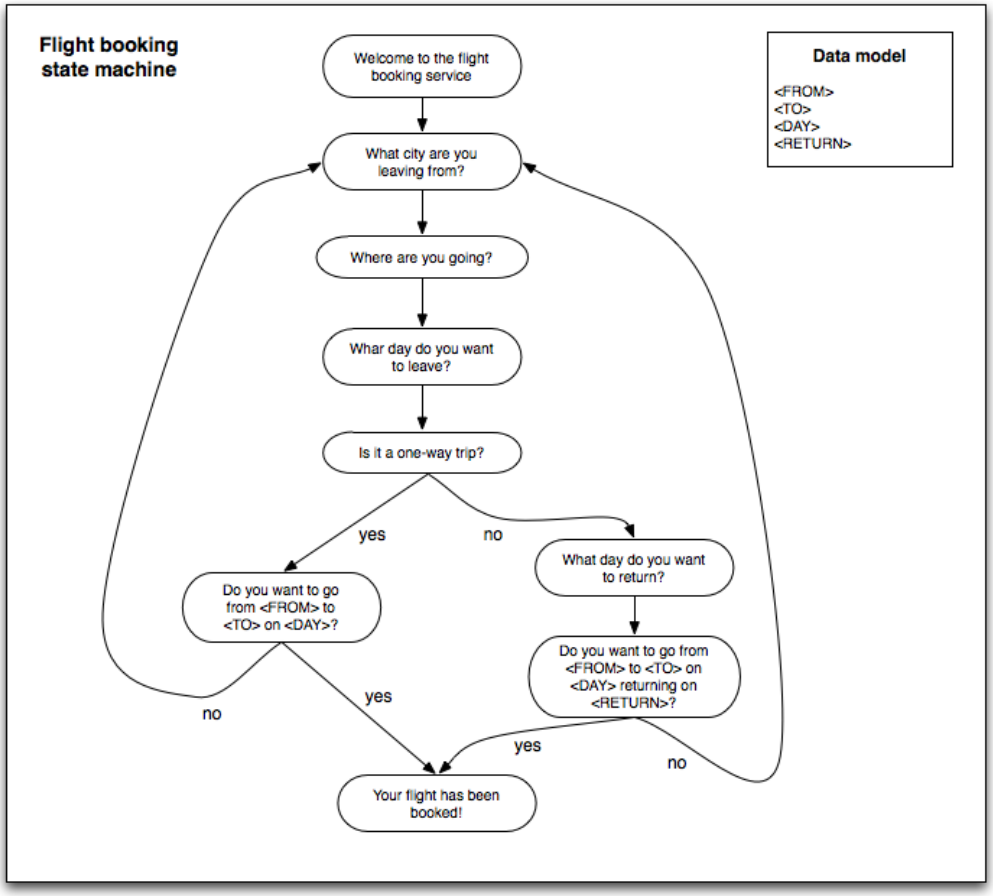
Jednotlivé skupiny prístupov budú podrobnejšie popísané v nasledujúcich podkapitolách.

8.2.1 Dialógové systémy založené na konečno - stavovom automate

V systéme založenom na konečnom počte stavov je štruktúra dialógu reprezentovaná sieťou prechodov medzi stavmi, pričom uzly predstavujú otázky kladené systémom a prechody medzi uzlami určujú všetky možné cesty cez sieť, čím sú špecifikované všetky legálne dialógy. Každý uzol siete zároveň reprezentuje informačný stav, v ktorom je informácia získavaná od používateľa alebo ním potvrdzovaná. V základnej štruktúre môžu byť použité poddialógy , ktoré zjednodušujú sieť prechodov a zabezpečujú štandardný prístup k jednotlivým, opakujúcim sa dialógovým výmenám medzi systémom a používateľom.

Hlavnou výhodou modelu založenom na konečnom počte stavov je jeho jednoduchosť. Z hľadiska perspektívy vývoja štruktúr prechodov stavov sú výhodné zvlášť pre modelovanie toku dialógu v dobre štruktúrovaných úlohách. Rozsah reakcií používateľa je obmedzený a tým sú na komponenty systému kladené menšie požiadavky, zvlášť na systém rozpoznávania reči. Chýbajúca flexibilita a prirodzenosť je vyvážená zmenšenými technologickými nárokmi. Z týchto dôvodov, najčastejšie dostupné komerčné systémy používajú určité formy modelov založených na konečných stavoch.

Príkladom dobre štrukturovaného dialógu, avšak príliš zložitého na modelovanie systémom s konečným počtom stavov môže byť rezervačný systém letov. Aj keď úloha rezervácie je dobre štruktúrovaná, pozostáva zo série zoradených podúloh, medzi ktorými je zložitá závislosť (napr. medzi zľavneným cestovným a dostupnosťou letu, spiatočným letom a časmi odletov a pod.)



Obr. 8.1: Konečno-stavová reprezentácia dialógu

8.2.2 Formularovo-orientované systémy

Vhodnejšie než vystavanie dialógu podľa vopred predurčených postupností otázok, organizovaných vo forme automatu s konečným počtom stavov, môže byť zhromažďovanie vopred definovaných informácií pomocou formulárovo orientovaných dialógov.

V tomto prípade tzv. formulár (frame), alebo šablóna plní úlohu modelu dialógu, ktorý obsahuje súpis položiek, pre ktoré systém požaduje informácie od používateľa. Prirodzene, model obsahuje tiež otázky, ktoré však nie sú kladené v konkrétnej postupnosti. Jednotka riadenia dialógu rozhoduje, ktorú zo všetkých daných otázok v aktuálnom stave dialógu položí. Konkrétna otázka sa položí vtedy, ak jej prislúchajúca podmienka má logickú hodnotu "true", pričom otázky nemusia byť uvádzané v chronologickom poradí. Ak táto podmienka je splnená pre viacero otázok naraz, môžu byť pri výbere použité aj iné faktory, ktorými sa rozlišuje priorita poradia.

Formulárovo orientovaný prístup má oproti systému založenom na konečnom počte stavov niekoľko výhod, tak z pohľadu používateľa ako aj vývojára. Čo sa používateľa týka, je viac flexibilný. Je zrejmé, že môže byť obtiažne vynútiť si od používateľa práve takú odpoveď akú požaduje systém. Schopnosť používať prirodzený jazyk a možnosť viacnásobného prístupu pre vyplňovania jednej položky ponúka systému možnosť spracovania odpovedí používateľa, v ktorých sa môže vyskytovať nadbytočná informácia vzhľadom k položenej otázke. Tato skutočnosť môže redukovať nevyhnutný čas na realizáciu dialógu a komunikácia sa môže stať prirodzenejšou. Z vývojárskeho pohľadu je implementácia takého stupňa flexibility v grafovo-orientovaných systémoch resp. v systémoch založených na konečnom počte stavov značne obtiažne ak nie nemožné.

Formulárovo orientovaný systém môže byť špecifikovaný deklaratívne podobne ako expertný systém. Podobne ako systémy založené na konečnom počte stavov aj formulárovo orientované systémy sú vhodné pre dobre a presne definované úlohy, v ktorých systém preberá iniciatívu v dialógu a získava informácie od používateľa za účelom splnenia poslania, ktorým môže byť sprístupnenie požadovanej položky databázy. Formulárovo orientovaný systém ponúka väčšiu flexibilitu dialógu, ktorý nie je rigidne určený, ale modifikovaný udalosťami. Avšak, súvislosť systému, ktorá prispieva k určeniu jeho nasledujúcej činnosti je značne limitovaná a býva redukovaná hlavne na analýzu predchádzajúceho výroku používateľa vzhľadom k vyplneniu určitej položky vo formulári resp. šablóne. Zložitejšie transakcie nemôžu byť modelované použitím tohto prístupu z nasledujúcich dôvodov:

- rozličnı́ používatelia sa môžu líšiť v úrovni poznania a tak je pre systém potrebná široká škála odpovedí,
- stav v danej aplikačnej doméne sa môže dynamicky meniť a tak nemusí byť možné špecifikovať všetky možné konfigurácie v predstihu.

8.2.3 Prístupy založené na realizácií tzv. plánu

V prístupoch riadenia dialógu orientovaných na plánovanie sú výroky používateľa spracovávané rovnakým spôsobom ako akcie v plánovacom systéme, ktoré sú vykonávané v takom poradí aby bolo dosiahnuté určité poslanie [15]. Úlohou výroku používateľa môže byť dosiahnutie určitého fyzikálneho stavu, ako napr. „piť pivo pri bare“. V danom prípade je požiadavka na pivo zahrnutá do plánu, pričom sú tiež nevyhnutné určité akcie, akými sú zaplatenie a podanie piva barmanom. Protikladom toho sú výroky, ktorých poslaním je prenos informácií, meniacich myšlienkový stav počúvajúceho. Kľúčovým prvkom tohto prístupu bolo modelovanie výrokov ako rečových aktov (speech act), ktoré ako operátori akcií (úkonov) pri plánovaní, predstavovali úlohy, podmienky, obmedzenia a efekty.

V prístupe orientovanom na plánovanie existuje niekoľko problémov. Pre odvodenie plánu hovoriaceho, by mal byť počúvajúci agent schopný rozpoznať akty komunikácie predkladané vo výrokoch hovoriaceho a vymedziť ich v príslušnej schéme plánu. Ak je akt komunikácie rozpoznaný nesprávne, vedie to k nesprávnej identifikácii plánu hovoriaceho. Vyžaduje to prinajmenšom ďalší mechanizmus na opravu nesprávneho označenia zámeru hovoriaceho. Navyše je tu problém, že proces rozpoznania plánu a plánovania, v ktorom je obsiahnuté zreťazenie počiatočných podmienok plánov do akcii, môže byť v zložitejších prípadoch kombinatoricky nerealizovateľný. Keď algoritmy plánovania vyžadujú argumenty z oblasti základných princípov, potom je najvhodnejšie pracovať vo vymedzených oblastiach, v ktorých sú argumenty udržateľné v kontrolovateľných medziach.

8.2.4 Kolaboratívne prístupy

Štruktúra dialógu je konštruovaná dynamicky ako výsledok uplatnenia princípov racionálnej kooperatívnej interakcie. Proces dialógu môže byť reprezentovaný vo forme plánov, zámerov a cieľov agentov, ktoré sú súčasťou dialógu. Plány samé o sebe nedefinujú dopredu-určené schémy akcií, ale sú odvádzané deduktívne z racionálnych princípov. Pretože dialóg je spoločná aktivita dvoch agentov, každý agent dialógu má záujem na tom aby bol porozumený [15]. Teoretický rámec pre racionálne pôsobenie agenta je založený na sade logických axióm, ktoré formalizujú základné princípy racionálnej činnosti a kooperatívnej komunikácie. Tento teoretický rámec bol definovaný v [16] a rozšírený v [69].

Princípy racionálnosti opisujú uvažovanie agenta o činnostiach, domnienkach a plánoch. Napr. prostredníctvom akcií, ktoré zahŕňajú akty komunikácie, meniace rozumový stav účastníka interakcie. Princípy spolupráce vyjad-

rujú motiváciu agenta správať sa kooperatívne vzhľadom k inému účastníkovi interakcie. Agent v dialógu akceptuje minimálne záväzky, napr. aktívne sa zúčastňovať v konverzácii, aby sa pokúsil porozumieť zámery iného agenta a generovať odpovede pre tieto záležitosti. Tieto princípy boli formalizované podobným spôsobom ako princípy racionálnosti.

8.2.5 Prístupy založené na štatistickom modelovaní rečového dialógu

Systémy s využitím štatistického prístupu k riadeniu dialógu sú alternatívou k štandardným prístupom, pri ktorých je dialóg vygenerovaný na základe ručne zapísaných pravidiel. Účelom hľadania alternatívnych prístupov je riadenie a modelovanie dialógu tak, aby pomocou neho bolo možné úspešne vykonať transakcie požadované používateľom a zároveň, aby tento proces bol čo najefektívnejší. Úspešnosť interakcie je často v konflikte s jej efektívnosťou [46]. Nevýhodou štandardných prístupov, ktoré využívajú ručne zapísané pravidlá je ich neflexibilita a neschopnosť zachytiť všetky možné stavy dialógu. Štatistické modelovanie umožňuje tento jav značne eliminovať.

V [46] rozdeľujú tieto systémy na systémy využívajúce:

- kontrolované učenie - supervised learning (učenie s učiteľom),
- maximalizáciu odmeny - utility maximization,
- učenie odmenou a trestom - reinforcement learning - RL (učenie bez učiteľa).

Prístup s kontrolovaným učením môže byť použitý v prípade, že je k dispozícií korpus dialógov, ktorý obsahuje príklady optimálnych dialógov. Prístupy založené na maximalizácií odmeny resp. na učení odmenou a trestom sa snažia identifikovať priority systému vo forme tzv. funkcií vhodnosti. Proces optimalizácie potom rozhoduje o vhodnej akcii systému v danom stave, na základe maximalizácie funkcie vhodnosti. Pri metóde maximalizácie odmeny sa vyberá činnosť (akcia) systému s maximálnou okamžitou „odmenou“. Naproti tomu, pri prístupe založenom na učení odmenou a trestom systém vyberá tú akciu, ktorá maximalizuje sumu „odmien“ v aktuálnom stave.

8.2.6 Systémy na báze jazykov na popis dialógu - DDL

Systémy na báze jazykov na popis dialógov (DDL – Dialogue Description Languages) možno považovať za osobitnú skupinu prístupov k modelovaniu riadenia dialógov, hoci často používajú aj také modely riadenia dialógu, ako

sú prístupy založené na konečnom počte stavov a formulárovo orientované prístupy. V súčasnosti najznámejšími jazykmi na popis dialógu sú jazyky VoiceXML (Voice eXtensible Markup Language) a SALT (Speech Application Language Tags).

Jazyk VoiceXML patrí do skupiny jazykov známych pod označením W3C Speech Interface Framework. Táto skupina jazykov tvorí komplexný rámec, ktorý umožňuje návrh a interpretáciu rečových aplikácií. Jazyku VoiceXML ako aj celému frameworku sa podrobne venujeme v kap. 10.

Jazyk SALT je samostatným jazykom, ktorý tvorí rozšírenie pre existujúce jazyky na zápis webových stránok (napr. HTML). Toto rozšírenie umožňuje pridávať hlasovú modalitu do existujúcich webových stránok.

9

Generovanie reči z textu

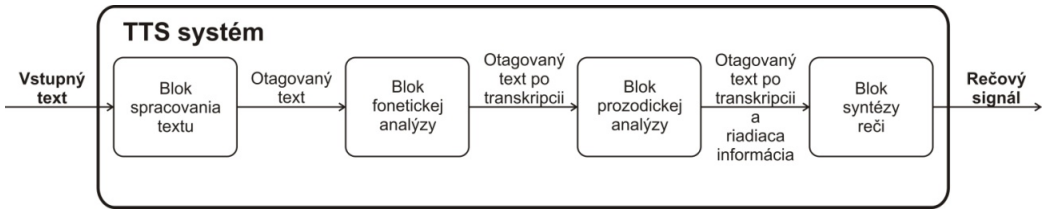
Technológia syntézy reči z texty je neoddeliteľnou súčasťou súčasťou rečových rozhraní medzi človekom a strojom a umožňuje počítačovým systémom poskytovať informácie používateľovi vo forme umelo generovanej reči. Nemalý význam má táto technológia napr. pre nevidiacich, ktorým umožňuje ovládať aj systémy, pôvodne navrhnuté iba s grafickým používateľským rozhraním.

9.1 Princípy syntézy reči z textu

Generovanie syntetickej reči, inými slovami syntéza reči z textu (TTS – Text-to-Speech) je technológia nevyhnutná pre umožnenie rečovej komunikácie medzi človekom a počítačovými resp. robotickými systémami. Syntéza reči z textu predstavuje neoddeliteľnú súčasť problematiky spracovania rečového signálu. Moderné metódy generovania syntetickej reči sú v dnešnej dobe realizované výlučne počítačovými TTS systémami, ktoré sa typicky skladajú z dvoch hlavných častí - **blok spracovania prirodzeného jazyka** a **blok formovanie akustického signálu - teda samotnej syntézy reči**. Blok spracovania prirodzeného jazyka zabezpečuje spracovanie vstupného textu do vhodnej podoby pre samotný proces syntézy reči. Blok syntézy reči generuje rečový signál zodpovedajúcou metódou na základe vstupných informácií, akými sú postupnosť foném, ktoré zodpovedajú syntetizovanému textu, informácia o prozódií danej frázy, prípadne ďalšie doplňujúce informácie. Jednotlivé bloky sa skladajú z modulov, ktoré zabezpečujú čiastkové úlohy.

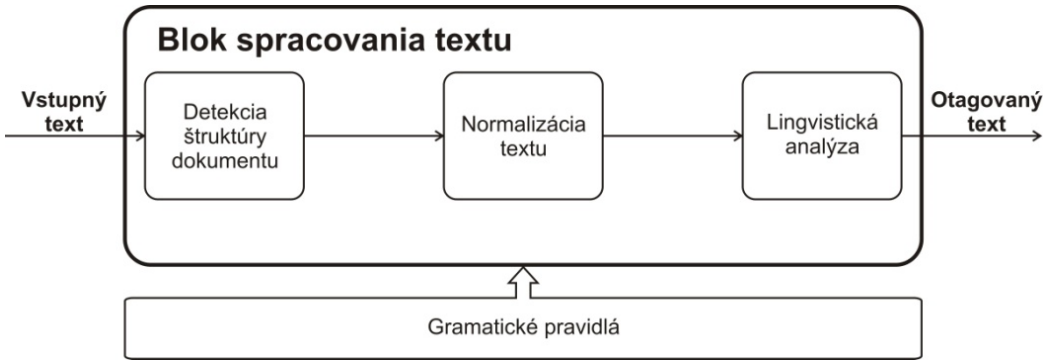
Všeobecná bloková schéma TTS systému je zobrazená na Obr. 9.1

Blok spracovania prirodzeného jazyka (Obr. 9.2) môže mať rôznu zložitosť v závislosti od aplikácie. Zväčša vykonáva analýzu vstupného textu a fonetickú a prozodickú analýzu. Vstupný text najprv vstupuje do modulu detekcie štruktúry dokumentu, kde sa vykoná detekcia typu textu (napr. čistý



Obr. 9.1: Všeobecná bloková schéma TTS systémov

vs. text s XML značkami), rozdelenie textu podľa jeho štruktúry (na odseky a kapitoly) a filtrovanie špecifických symbolov. Primárnou úlohou tohto bloku je ale rozdelenie textu na vety, ktoré sú základnými jednotkami, s ktorými TTS systém pracuje.



Obr. 9.2: Blok spracovania textu v TTS systémoch

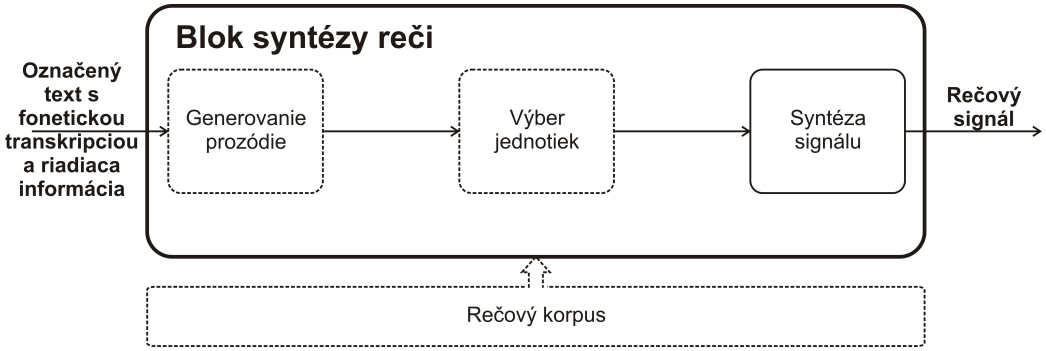
Ďalšou dôležitou fázou predprípravy textu na syntézu je jeho **normalizácia**, ktorá spočíva v prepise vstupného textu do jeho plnej slovnej formy s ohľadom na správny gramatický tvar. Ide najmä o prepis čísloviek, časových údajov, skratiek a symbolov do ich správnej textovej formy.

Súčasťou predspracovania textu pre proces syntézy je aj lingvistická analýza, ktorá realizuje syntaktické a sémantické označenie textu príslušnými značkami. Priradzujú sa sémantické vlastnosti jednotlivým slovám, frázam a vetám, ktoré sú dôležité pre voľbu správnej výslovnosti a prozódie v nasledujúcich krokoch spracovania textu. Proces je obvykle riadený gramatickými pravidlami jazyka a slovníkom.

Výstupom predchádzajúcich krokov je normalizovaný text, rozdelený do viet, ktorého súčasťou sú značky pre zabezpečenie správnych parametrov výslednej syntetickej reči. Takto označený text vstupuje do blokov fonetickej a prozodickej analýzy. Výstupom je postupnosť foném, ktoré reprezentujú požadovanú akustickú podobu syntetizovaného textu spolu so značkami, ktoré

upravujú prozódiu (hlasitosť, základný tón, trvanie) výslednej vygenerovanej reči.

Blok syntézy reči predstavuje v TTS systémoch časť zodpovednú za generovanie rečového signálu na základe výstupných informácií z bloku spracovania prirodzeného jazyka. Táto časť celého procesu generovania syntetickej reči je závislá na použitej metóde formovania výstupného signálu, avšak celý proces je možné zovšeobecniť a zhrnúť do niekoľkých kľúčových modulov (Obr. 9.3).



Obr. 9.3: Blok syntézy reči

Prvým modulom je **modul generovania prozódie**. Jeho základnou úlohou je generovanie príslušných prozodických vlastností na základe ich popisu, ktorý bol získaný v procese prozodickej analýzy. Hlavnou úlohou tohto modulu je vygenerovať priebehy intonácie, intenzity a dĺžky trvania pre jednotlivé časti vstupného textu. Na generovanie melódie rečového prejavu je možné použiť viacero modelov intonácie. Tieto modely je možné rozdeliť podľa oblasti, v ktorej je výsledná intonácia formovaná, na *akustické modely*, *lingvistické modely* a *percepčné modely intonácie*.

Akustické modely intonácie pracujú na báze reprezentácie intonácie pomocou priebehu kontúry základného tónu v čase. Medzi najpoužívanejšie modely tohto typu patria Fujisakiho model [20], modely pracujúce s transkripčným systémom ToBI a intonačný model Tilt [8]. V súčasnej dobe patrí táto skupina modelov medzi najpoužívanejšie, pričom väčšina TTS systémov využíva práve model *Tilt*.

Lingvistické modely intonácie vychádzajú z lingvistickej reprezentácie prozódie, a teda patria medzi najzložitejšie. Poslednou skupinou modelov sú **percepčné modely**, ktoré využívajú princíp percepcie intonácie z hľadiska jej vnímania užívateľom sýtemu. Reprezentantom tejto skupiny je napríklad model *IPO* [23].

Ďalším z nepovinných modulov, ktorý figuruje v bloku syntézy reči, je

modul výberu jednotiek. Jeho použitie je potrebné najmä v prípade *konkatenačnej syntézy* (princípom je zreťazenie akustických jednotiek), pričom modul je zodpovedný za výber najvhodnejších jednotiek pre syntézu reči. Pri korpusovo-orientovaných systémoch je celý proces odvodený od rečového korpusu, kde je nevyhnutné vhodne zvoliť dĺžku jednotlivých časti (jednotiek) rečového prejavu (fonémy, difóny, slová alebo jednotky s premennou dĺžkou) z hľadiska maximalizácie zrozumiteľnosti a prirodzenosti prejavu na výstupe.

Posledným blokom systému syntézy reči je **syntéza signálu**. Tento modul sa môže líšiť v závislosti od použitej metódy a obvykle je tvorený potupnosťou viacerých krokov, ktorých výstupom je rečových signál. Jednotlivé metódy syntézy reči budú bližšie opísané v nasledujúcej časti.

9.2 Metódy syntézy reči z textu

Metódy syntézy reči je možné rozdeliť podľa viacerých hľadísk, avšak najdôležitejším je delenie podľa použitého prístupu ku generovaniu signálu. Z tohto hľadiska môžeme metódy syntézy reči rozdeliť nasledovne [86]:

- Syntéza reči na báze modelovania vokálneho traktu
  - Artikulačná syntéza
  - Formantová syntéza
- Konkatenačná syntéza
  - Difónová syntéza
  - Syntéza s predvýberom jednotiek (angl. unit-selection)
- Štatistická parametrická syntéza
  - Syntéza s využitím HMM modelov
  - Syntéza s využitím neurónových sietí

9.2.1 Artikulačná syntéza

Princíp artikulačnej syntézy spočíva v modelovaní procesu produkcie reči človekom, pričom principiálne patrí táto metóda medzi tie najprirodzenejšie [67]. Keďže celý proces je popísaný pomocou vopred definovaných pravidiel, niekedy sa zvykne táto metóda nazývať aj ako pravidlami riadená. Produkcia syntetickej reči prebieha pomocou vopred navrhnutého matematicko-fyzikálneho modelu stavby rečových organov človeka, pričom výstup celej

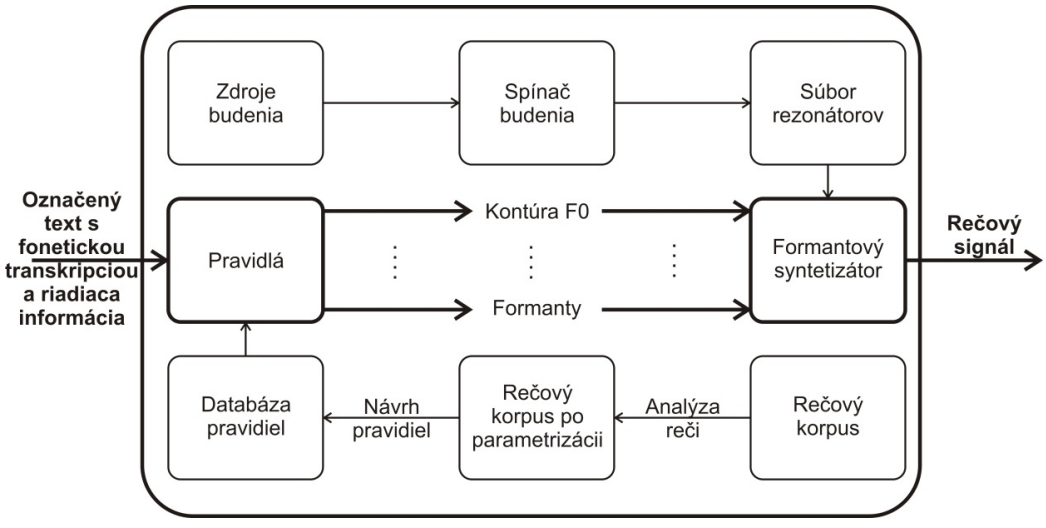
tejto sústavy tvorí signál získaný matematickou simuláciou vydychovaného vzduchu a činnosti jednotlivých modelov.

Keďže fyziologický proces tvorby reči u človeka je pomerne zložitý, táto zložitosť sa automaticky prenáša aj na model artikulačnej syntézy reči. Naproti tomu, výhodou tohto modelu je pomerne jednoduché vytváranie nových hlasov pomocou zmeny jeho parametrov. Dnes sa tieto syntetizátory využívajú najmä v oblasti fonetického výskumu, keďže umožňujú relatívne presne skúmať celý proces vytvárania reči a dôsledky zmeny parametrov rečových orgánov na zmenu celkového rečového prejavu.

Kvalita, ktorú sa darí dosiahnuť zaostáva za ostatnými metódami, a to zvlášť preto, že je zložité vytvárať dlhšie rečové úseky. Ďalším odvetvím, kde sa uplatnila táto metóda je audiovizuálna syntéza, teda napríklad oblasť vytvárania audiovizuálneho 3D modelu ľudskej hlavy [83].

9.2.2 Formantová syntéza

Formantová syntéza predstavuje druhú metódu z kategórie pravidlami riadených prístupov k syntézy reči. Z historického hľadiska sa dá považovať za veľmi úspešnú, aj keď v dnešnej dobe je už na ústupe a jej využitie v praxi je zriedkavé. Princíp metódy spočíva vo využití teórie zdroja a filtra, a teda na modelovaní zdroja budenia signálu a lineárneho filtra, ktorý je zodpovedný za popis vokálneho traktu. Schematický náčrt formantového syntetizátora riadeného pravidlami je zobrazený na Obr. 9.4.



Obr. 9.4: Bloková schéma formantového syntetizátora

Tento typ syntetizátora sa skladá z dvoch základných blokov, ktorými sú

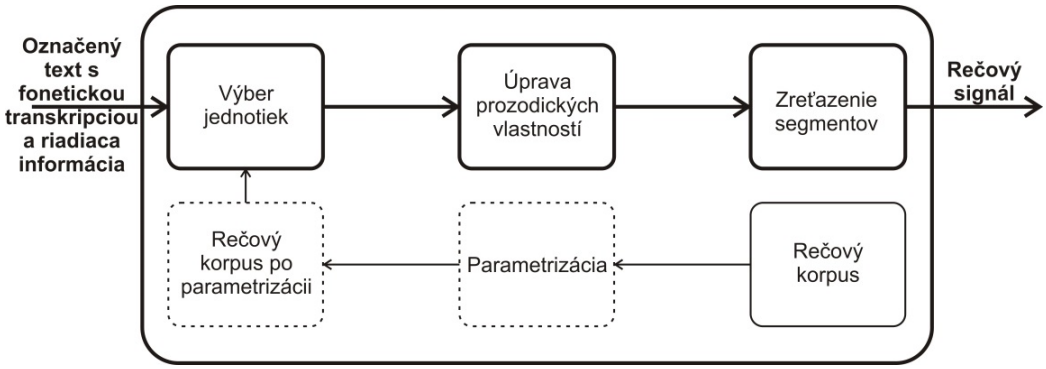
navrhnuté pravidlá a formantový syntetizátor. Pravidlá sa obvykle získavajú z rečového korpusu, pričom dáta v korpuse je nutné parametrizovať, čo má za následok oddelenie časti budenia od časti vokálneho traktu. Na takto upravených dátach sa následne zostavujú pravidlá pre nastavenie syntetizátora. Ich návrhom je obvykle poverený expert z oblasti fonetiky, fonológie a akustiky, avšak existujú aj prístupy, kedy sa pravidlá navrhujú automaticky priamo z rečovej databázy napríklad s využitím HMM modelov [1]. Pravidlá následne vstupujú do bloku formantovej syntézy, ktorý je zostavený zo zdroja budenia (zdroj periodického budenia a zdroj šumu), spínača jednotlivých zdrojov a série rezonátorov. Formantové syntetizátory ponúkajú veľmi dobrú zrozumiteľnosť na výstupe, avšak prirodzenosť ich prejavu je nepostačujúca. Je to spôsobené najmä pomerne jednoduchou implementáciou zdroja a celkového modelu vokálneho traktu, ktorý nereprezentuje verne dynamiku reči. Dôvodom prečo sa používanie tejto metódy v poslednej dobe ustupuje, je zložitý návrh presných pravidiel, s ktorými je možné dosiahnuť primeranú kvalitu systému.

9.2.3 Konkatenatívna syntéza

Konkatenačné metódy patria medzi metódy druhej generácie TTS systémov. Princípom metódy je, že rečový signál na výstupe vznikne pospájaním – konkatenáciou jednotlivých častí prejavu za pomoci rečových segmentov zo špecifického korpusu. V porovnaní s prvou skupinou TTS systémov (založených na pravidlách) je základnou výhodou absencia manuálneho definovania pravidiel, pričom každý segment prejavu je priamo odvodený z rečového korpusu, a teda z prirodzenej reči. Pomerne zložitými sa stavajú v tomto prípade javy, akým je napríklad koartikulácia, ktorá musí byť zachytená v jednotlivých pospájaných segmentoch pre dosiahnutie čo najlepšej prirodzenosti reči na výstupe. Schematický náčrt tejto metódy je zobrazený na Obr. 9.5.

Difónová konkatenácia

TTS systém, ktorý pracuje na báze difónovej konkatenácie je zložený z troch základných častí. Vstupné dáta získané z procesu spracovania prirodzeného jazyka sú v prvom kroku spracovávané blokom zodpovedným za výber jednotiek, ktorý realizuje výber najvhodnejších jednotiek (difón) z predpripraveného rečového korpusu. Rečový korpus predstavujú v tomto prípade špeciálne pripravené akustické dáta, v ktorých sú zachytené podľa možností všetky difóny jazyka, pre ktorý je systém určený. Difóny sú obvykle obsiahnuté v nosných slovách alebo vetách tak, aby sa podarilo čo najviac priblížiť k ich reálnej reprezentácii v jazyku. Rečové korpusy môžu byť ďalej buď paramet-



Obr. 9.5: Bloková schéma difónovej konkatenačnej sytnézy reči

rizované a rečová syntéza následne prebieha pomocou rekonštrukcie týchto parametrov na výstupe, alebo sa môže k nim pristupovať aj priamo bez parametrizácie. Blok výberu jednotiek podľa časových hraníc jednotlivých difón v nosných jednotkách vyberie potrebné difóny a tie následne pokračujú do druhého bloku, ktorý je zodpovedný za úpravu a prispôsobenie prozodických vlastností, s cieľom čo najviac sa priblížiť požadovanému kontextu syntetickej reči. Medzi základné algoritmy pre modifikáciu periódy základného tónu a trvania jednotlivých segmentov patria [86]:

- **PSOLA** – (angl. *Pitch Synchronous OverLap and Add*). Táto technika separuje vstupný rečový signál do rámcov synchrónne s periódou základného hlasivkového tónu a vykonáva modifikáciu pomocou prekrývania a pridávania týchto rámcov tak, aby sa výsledný prejav zhodoval s očakávanou špecifikáciou syntézy na výstupe. Algoritmus pracuje buď v časovej oblasti (angl. TD-PSOLA – Time Domain PSOLA), alebo v spojení s niektorou z metód parametrizácie (angl. FD-PSOLA – Frequency Domain PSOLA, LP-PSOLA – Linear Predictive PSOLA). Vo všeobecnosti je pre túto metódu príznačná vysoká kvalita výstupnej reči.

- **RELP** – (angl. *Residual-Excited Linear Prediction*). Táto metóda využíva lineárnu predikciu budenej chybovým signálom. Pracuje na báze extrakcie LPC koeficientov a ich reziduií, ktoré sú následne modifikované obdobným spôsobom, ako to bolo pri metóde PSOLA. Modifikácia rezíduí sa však musí realizovať spoločne s modifikáciou prenosovej funkcie filtra, ktorý popisuje vokálny trakt, inak dochádza k degradácii výsledného signálu.

- **Sinusoidálny model** – využíva harmonický model a dekompozíciu

každého rámca na sadu harmonických zložiek na základe estimácie frekvencie základného tónu. Parametre modelu následne tvoria amplitúdy a fázy jednotlivých harmonických zložiek, pomocou ktorých je možné zmeniť hodnotu základného tónu, zatiaľ čo spektrálna obálka signálu zostáva nezmenená.

- **Harmonický a šumový model** – (angl. HNM - *Harmonic plus Noise Model*) predstavuje rozšírenie pôvodného sinusoidálneho modelu o ďalší šumový komponent, ktorý umožňuje presnejšie modelovanie vyšších frekvenčných zložiek znelých segmentov a všetkých častí neznelých segmentov.

- **MBROLA** – (angl. *MultiBand Resynthesis OverLap-Add*) predstavuje ďalšiu z techník založených na princípe PSOLA. Táto metóda analyzuje všetky rečové rámce pomocou algoritmu multipásmového budenia MBE (angl. Multi-Band Excitation), ktorého úlohou je vypočítať parametre hybridného harmonicko-stochastického modelu reči. Fáza jednotlivých znelých úsekov sa následne nastaví na konštantnú hodnotu a amplitúda je nastavená tak, aby hodnota frekvencie základného tónu vo všetkých segmentoch bola rovná jeho priemernej hodnote získanej z rečových nahrávok obsiahnutých vo vstupnom rečovom korpuse. Výsledný rečový signál sa následne získa poskladaním jednotlivých upravených segmentov.

Všetky vyššie spomenuté techniky poskytujú porovnateľnú kvalitu, a tak voľba najvhodnejšej techniky je závislá najmä od kritérií, akými sú rýchlosť výpočtu a množstvo diskového priestoru, ktorý je potrebný pri jeho vykonávaní. Posledným z blokov pri difónovej syntéze je zreťazenie jednotlivých upravených segmentov do výsledného rečového signálu. Využitie difón predstavuje v prípade difónovej syntézy určitý kompromis medzi ich početnosťou a maximálnou možnou dosahovanou kvalitou. Ich základnou vlastnosťou, ktorá ich predurčuje na použitie v TTS systémoch je ich konštrukcia. Difóny sú tvorené oblasťou od stredu prvej hlásky po stred nasledujúcej, čo umožňuje zachytiť prechod medzi jednotlivými hláskami, a tým aj koartikuláčné javy. Kvalita tejto metódy je všeobecne považovaná za veľmi dobrú. Pre slovenský jazyk existuje viacero implementácii difónovej syntézy, pričom všetky vykazovali dobrú kvalita výstupnej reči [85]. Medzi základné nevýhody patrí napríklad veľmi problematická transformácia hlasov. Difónové syntetizátory sú závislé na rečníkovi, ktorý nahovoril rečový korpus, a pokiaľ je nutné vybudovať TTS systém s novým hlasom, je nutné aj pomerne časovo náročné nahrávanie nového rečového korpusu. Existujú však aj metódy, pomocou kto-

rých je možné do určitej miery pozmeniť parametre existujúcich hlasov, avšak sú výpočtovo veľmi náročné a ich použitie často speje ku jeho degradácii [38].

Syntéza s predvýberom jednotiek (unit-selection)

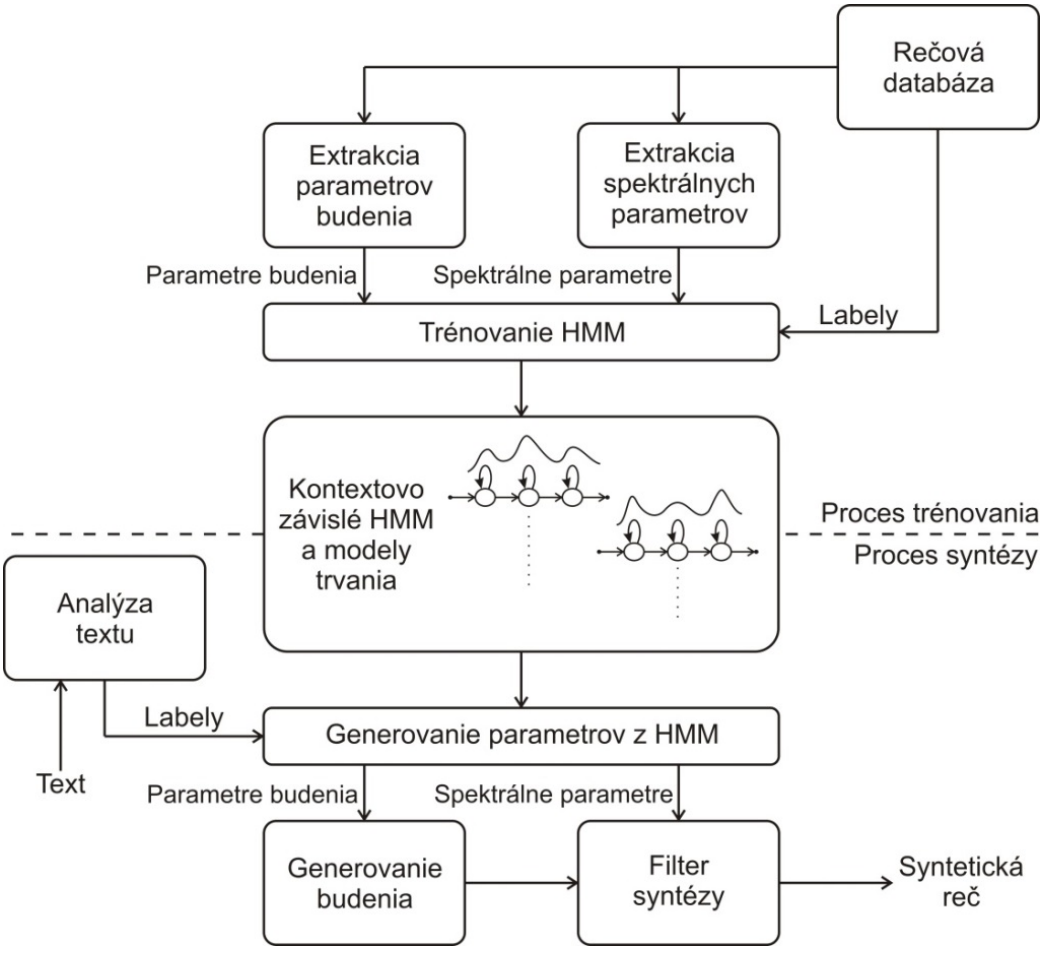
Ďalším zástupcom konkatenačnej syntézy je syntéza s predvýberom jednotiek. Táto metóda predstavuje technicky najpokročilejší prístup v oblasti konkatenačnej syntézy. Základnou podstatou unit-selection metódy je výber najvhodnejšej jednotky pre reprezentáciu vstupného textu z veľkého rečového korpusu prirodzeného jazyka. Rozhodujúcim rozdielom oproti difónovej syntéze je, že rečový korpus nie je tvorený nosnými slovami a vetami, ale vetami, ktorých každá časť môže predstavovať jednotku, z ktorej bude výsledky rečový prejav zložený. Menší počet miest, v ktorých sú jednotky tým pádom spojené, má za následok zvýšenie prirodzenosti syntézy na výstupe. Zvýšená pozornosť je tu venovaná najmä oblasti práce s rôznymi typmi jednotiek v rozsiahlych korpusoch, ich korektnej konkatenácii, rozšírenejšej práci s prozódiou a potlačeniu negatívnych vplyvov spracovania signálu na finálnu syntézu reči.

Vstup do systému pre syntézu reči na báze unit-selection tvorí označený text spoločne so zodpovedajúcim fonetickým prepisom a riadiacou informáciou, ktorá spresňuje informácie o výslednej prozódii vety. Do procesu syntézy ďalej vstupuje rečový korpus, ktorý je pomocou vhodnej implementácie upravený na databázu všetkých jednotiek, ktoré sa v danom korpuse nachádzajú. Korpus obsahuje vety v danom jazyku, pričom je nutné, aby sa v ňom nachádzalo dostatočné množstvo dát (rádovo hodiny akustických nahrávok). Je tiež potrebné, aby v korpuse bolo podľa možností viacero realizácií jednotlivých akustických jednotiek. Vstupné dáta sú následne spracované pomocou algoritmu výberu jednotiek, ktorého úlohou je nájsť v databáze najvhodnejšieho zástupcu rečových jednotiek a ich vyhladením vytvoriť výslednú reč. Základnou úlohou je teda nájsť optimálnu postupnosť rečových jednotiek na základe požiadavky definovanej vstupným textom, ktorý má byť syntetizovaný. Kvalita výslednej reči je definovaná práve blízkosťou nájdenej postupnosti jednotiek k požadovanej podobe syntetickej reči, pretože čím je táto blízkosť vyššia, tým menej dodatočných modifikácií signálu je potrebné vykonávať. Po nájdení najlepšej možnej postupnosti je vykonaná úprava prozodických parametrov a jednotlivé segmenty sa pre získanie rečového signálu zreťazia.

9.2.4 Štatistická parametrická syntéza

Štatistická parametrická syntéza reči na báze skrytých Markovových modelov (angl. HMM – Hidden Markov Model) patrí v posledných rokoch medzi naj-

viac rozvíjajúce sa metódy v tejto oblasti. Základný princíp spočíva vo využití kontextovo závislých HMM modelov, natrénovaných pomocou rečovej databázy, ako generatívnych modelov pre proces syntézy reči. Rečovú databázu reprezentujú v tomto prípade akustické modely, ktorých veľkosť je rádovo niekoľko MB, čo oproti veľmi rozsiahlym korpusom v prípade unit-selection syntézy predstavuje markantnú úsporu úložného priestoru a v konečnom dôsledku aj výpočtových nárokov, keďže sa zmenšujú nároky na prehľadávanie veľkého množstva dát. Obr. 9.6 zobrazuje blokovú schému syntézy reči na báze HMM.



Obr. 9.6: Bloková schéma syntézy reči na báze HMM

Systém pre syntézu reči metódou, ktorá využíva HMM modely, je rozdelený na dve časti, a to na trénovaciu časť a na časť syntézy reči [97]. Proces trénovania je obdobný, aký sa používa pri systémoch pre rozpoznávanie reči. Hlavným rozdielom je, že ako spektrálne parametre, tak aj parametre

budenia, sú extrahované priamo z akustickej databázy a sú modelované kontextovo závislými HMM modelmi, ktoré zohľadňujú fonetický, lingvistický a prozodický kontext. Trénovanie spektrálnych parametrov, kontúry základného tónu a dĺžky trvania segmentov sa uskutočňuje pomocou rozhodiacich stromov. Proces syntézy reči spočíva v analýze vstupného textu pomocou modulu pre spracovanie prirodzeného jazyka (výstupom tohto kroku je ortoepicky prepis s príslušnými značkami), jeho následným spracovaním formou zreťazovania príslušných HMM modelov a generovania jeho parametrov, výsledkom čoho je produkcia syntetickej reči.

10

Technológie návrhu rečových interaktívnych komunikačných systémov

Súčasné technológie návrhu rečových interaktívnych komunikačných systémov je možné rozdeliť do viacerých skupín, podľa ich určenia:

- **Jazyky** - zvyčajne slúžia ako riadiace skripty pre definovanie a riadenie jednotlivých činností/modulov v rozhraniach človek-stroj.
- **Rozhrania** - definujú spôsob komunikácie medzi jednotlivými modulmi rečových interaktívnych systémov, ako napr. ASR, TTS a pod.
- **Platformy** - predstavujú zväčša komplexnejšie riešenia pre poskytovanie rečových interaktívnych služieb.
- **Nástroje** - umožňujú jednoducho pripraviť alebo použiť predpripravené moduly, systémy alebo služby.
- **Dáta (databázy) a modely** - slúžia pre vytvorenie modelov pre jednotlivé rečové technológie, resp. sa jedná o samotné modely.

V predchádzajúcom období, bolo badať významnú snahu o štandardizáciu vo všetkých uvedených skupinách, ako odpoveď na problémy s kompatibilitou a prenositeľnosťou. Kompatibilita a prenositeľnosť vyvinutých riešení (modulov, modelov a služieb) sú dôležité pre redukciu nákladov a časovej náročnosti pri vývoji služieb založených na rečových technológiách.

Proces štandardizácie nie je možné považovať za ukončený, najmä vďaka neustálemu pokroku a vývoju v samotných technológiách, ako sú rozpoznávanie a syntéza reči, porozumenie a generovanie prirodzeného jazyka alebo

riadenie dialógu a tiež vďaka novým konceptom v oblasti používania rečových technológií a vďaka novým typom zariadení a rozhraní.

Výsledkom procesu štandardizácie sú viaceré štandardy resp. odporúčania, ktoré definujú jazyky, rozhrania alebo spôsob reprezentácie dát, ktoré je možné považovať za úspešné, vzhľadom na širokú akceptáciu naprieč akademickou a tiež priemyselnou komunitou.

Hlavní predstavitelia jednotlivých typov technológií a štandardov budú predstavení v nasledujúcich podkapitolách.

10.1 Jazyky pre návrh rečových interaktívnych komunikačných systémov a služieb

V oblasti jazykov pre návrh rečových interaktívnych systémov a služieb je možné konštatovať, že dominantnou skupinou jazykov sa stali jazyky navrhnuté pod záštitou World Wide Web konzorcia (W3C). Konkrétne sa jedná o dve skupiny jazykov, pričom prvá z nich, s názvom W3C Speech Interface Framework vznikla pre štandardizáciu unimodálnych rečových systémov, a druhá skupina W3C Multimodal Interactive Framework ponúka štandardy pre multimodálne systémy.

10.1.1 W3C Speech Interface Framework

Jazyky združené pod označeném W3C Speech Interface Framework (SIF) môžeme označiť za najdôležitejšiu skupinu štadardov jazykov, ktorá slúži pre vytváranie rečových interaktívnych systémov a služieb. Tieto jazyky vznikli ako výsledok práce pracovnej skupiny Voice Browser Working Group (VBWG), ktorá bola vytvorená vrámci organizácie World Wide Web Consortium (W3C) v roku 1999. W3C je medzinárodná komunita, ktorá vyvíja otvorené štandardy pre dlhodobý rast webu [https://www.w3.org/].

Pracovná skupina VBWG bola zameraná na vývoj štandardov pre tzv. "hlasový prehliadač"(Voice Browser). Prvotnou ideou pri vzniku tejto skupiny bolo pripraviť štandardy pre tzv. hlasový prehliadač, ktorý by umožňoval prehliadať internet pomocou rečovej komunikácie cez telefónnu sieť. Hlasový prehliadač bol navrhovaný ako systém, ktorý umožní ľuďom sprístupniť obsah webu použitím syntézy reči z textu a automatického rozpoznávania reči cez telefón, avšak počas existencie VBWG sa situácia rapídne zmenila. Konkrétne sa jedná o dve zásadne zmeny, ktoré ovplyvnili vývoj a následné využitie spomínaných štandardov.

Prvou významnou zmenou bol zásadný posun v dostupnosti internetu priamo v mobilných telefónoch, ktorý sa udial počas existencie VBWG. Za-

tiaľčo v dobe vzniku skupiny (r. 1999) nebol internet bežne dostupný v mobilných telefónoch, ktoré na to navyše ani neboli pripravené, po roku 2000 sa situácia začala výrazne meniť. Dostupnosť internetu v mobilných telefónoch sa začala zvyšovať vďaka ponuke mobilných operátorov a tiež vďaka rapídnemu rozšíreniu WIFI sietí. Základná myšlienka tzv. hlasového browsera, ktorý by sprístupňoval obsah webu cez telefonický hovor pomocou rečových technológií prestávala mať zmysel, ak bolo možné prehliadať internetový obsah priamo na displeji telefónu cez webový prehliadač.

Druhou významnou zmenou bol nástup smartfónov s dotykovými displejmi, ktoré úplne zmenili spôsob interakcie s takýmito zariadeniami. Používať webový prehliadač sa stalo omnoho jednoduchším jednak pre možnosť dotykového ovládania a tiež vďaka väčšiemu displeju.

S príchodom smatfónov so systémom Android sa tiež začala éra malých aplikácií napojených na internet, ktoré sa stali často komfortnejšou alternatívou voči získaniu informácií cez telefonický hovor.

Dobrým príkladom je vyhľadávanie informácií o počasí. Zatiaľčo koncepcia hlasového prehliadača predpokladala, že informáciu o počasí používateľ získa zavolaním na tzv. hlasový portál, kde získanie informácie prebehne pomocou hlasových príkazov a odpovedí v rámci dialógu, v dobe webových aplikácií stačí kliknúť na ikonku s aplikáciou o počasí a zobrazí sa aktuálne počasie a predpoveď počasia pre ďalšie dni na displeji smartfónu.

Napriek týmto zmenám, ktoré menia mieru použiteľnosti konceptu hlasového prehliadača, koncept ako taký a tiež jednotlivé vyvinuté jazyky boli úspešne implementované v reálnom živote a akceptované v akademickej aj vedeckej sfére. Dôvodom je fakt, že koncept hlasového prehliadača vychádza a implementuje koncept a technológie pre rečovú dialógovú interakciu medzi človekom a strojom (Human-machine interaction), ktorý je stále aktuálny a je využívaný pre komunikáciu s osobnými asistentnmi (napr. Apple SIRI, Google Now, Microsoft Cortana, resp. Amazon Echo), robotickými zariadeniami, humanoidnými robotmi a v neposlednom rade v koncepte ïnternet vecí"(Internet of Things).

W3C Speech Interface Framework je skupina štandardov, ktoré sú združené okolo VoiceXML štandardu. VoiceXML je tzv. jazyk na popis dialógu (Dialoge Description Language - DDL). Framework pozostáva z nasledujúcich štandardov (recommnedations):

- VoiceXML – jazyk pre vytváranie rečových dialógov. Umožňuje písať skripty, ktoré definujú tok dialógu, jeho obsah a ostatné atribúty rečových dialógových interakcií.
- SRGS (Speech Recognition Grammar Specification) – jazyk pre zápis rečových gramatík. Definuje formát zápisu rečových gramatík pre

systémy automatického rozpoznávania reči. Rečové gramatiky sú dokumenty, ktoré definujú slová a frázy, ktoré má akceptovať rozpoznávač reči v aktuálnom stave dialógu.

- SISR (Semantic Interpretation for Speech Recognition) – jazyk pre zápis značiek pre sémantickú interpretáciu. Definuje formát pre pridávanie sémantických značiek do rečových gramatík vo formáte podľa SRGS. Sémantické značky umožňujú extrahovať význam z výsledku automatického rozpoznávania reči.

- SSML (Speech Synthesis Markup Language) – jazyk pre rečovú syntézu. Definuje spôsob ako manažovať výstup systému v podobe syntetickej reči, prednahradej reči alebo hudby. Umožňuje vytvárať inštrukcie, ktoré menia vlastnosti produkovanej syntetickej reči.

- PLS (Pronunciation Lexicon Specification) – jazyk pre reprezentáciu fonetickej informácie pre systémy automatického rozpoznávania a syntézy reči.

- CCXML (Call Control XML) – jazyk, ktorý umožňuje písať pravidlá pre komunikáciu medzi telefónnou platformou a hlasovým portálom (platformou).

- SCXML (State Chart XML) – jazyk, ktorý umožňuje pripravovať konečno - stavové automaty pre riadenie relácií s používateľom, dialógov a iných procesov.

Nasledujúce podkapitoly prinášajú stručný popis jednotlivých štandardov.

VoiceXML

Jazyk VoiceXML (Voice eXtensible Markup Language) je XML jazyk navrhnutý pre vytváranie rečových aplikácií. Ide o skriptovací jazyk, ktorý slúži na riadenia rečovej dialógovej interakcie medzi človekom a počítačovým systémom, resp. rečovou platformou (Voice Platform). Takéto skripty obsahujú inštrukcie pre tok dialógu, obsah dialógu a spôsob jeho prezentácie. VoiceXML jazyk sa stal veľmi obľúbeným a rozšíreným vďaka jednoduchému syntaxu, ktorý je založený na XML jazyku, čo vedie k dobrej čitateľnosti a spravovateľnosti vytvorených rečových aplikácií. Široké akceptovanie jazyka VoiceXML ako priemyselného štandardu prinieslo ďalšie výhody v podobe dobrej prenositeľnosti služieb na iné platformy.

Príklad VoiceXML kódu je možné vidieť na Obr. 10.1.

```
<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0" xmlns="http://www.w3.org/2001/vxml">

<form id="start">
  <block name="answer">
    Welcome to Weather forecast service.
  </block>
  <goto next="#inputs"/>
</form>

<form id="inputs">
  <field name="city">
    <noinput> I did't catch your answer. </noinput>
    <nomatch> Please say city for which
              do you want to obtain weather forecast. </nomatch>


    <prompt> For which city do you wish to look for weather forecast? </prompt>
    <grammar src="city.grxml"/>

    <filled>
      <assign name="Forecast_city" expr="city"/>
      <goto next="#confirm_city"/>
    </filled>
  </field>
</form>

</vxml>
```

Obr. 10.1: Príklad zdrojového kódu VoiceXML dokumentu

VoiceXML jazyk je v súčasnosti oficiálnym ödporúčaním"W3C organizácie, čiže je defakto štandardom. Začiatky jeho vývoja ale neboli pod záštitou W3C. V roku 1999 štvorica spoločností AT&T, IBM, Lucent a Motorola založili tzv. "VoiceXML fórum"[8] pre návrh jazyka, ktorý by prispel k zlepšeniu vývoja rečových aplikácií. Prvá verzia jazyka bola predstavená v auguste 1999. Prvá oficiálna verzia jazyka (VoiceXML 1.0), pripravená VoiceXML fórom, bola prezentovaná v Marci 2000. Po uvedení tejto verzie, organizácia W3C prevzala záštitu nad vývojom VoiceXML a začal sa vývoj ďalšej verzie jazyka.

Jazyk VoiceXML 1.0 obsahoval, okrem tagov (značiek) pre popis dialógu, tiež značky pre manažment hovorov, rečové gramatiky a rečovú syntézu. Nakoľko pre tieto jednotlivé činnosti vznikla potreba ich oddeliť aj z dôvodu prehľadnejšieho návrhu resp. väčších nárokov na popis jednotlivých činností, nová verzia jazyka VoiceXML (2.0) bola od začiatku vyvíjaná tak, že sa zameriavala iba na popis samotného dialógu. Značky pre manažment hovorov, rečové gramatiky a syntézu reči sa stali základom pre nové jazyky v rámci W3C Speech Interface Frameworku - CCXML, SRGS a SSML.

Jazyk VoiceXML verzie 2.0 bol oficiálne vydaný ako W3C recommendation v marci 2004. Táto verzia jazyka sa stala kvázi priemyselným štandardom v oblasti rečových služieb [9]. V júni 2007 bola predstavená ďalšia verzia jazyka - VoiceXML 2.1, ktorá nie je kompletne novým štandardom,

iba rozšírením predošlej verzie o malú skupinu pridaných atribútov.

Následne bol odštartovaný vývoj novej verzie jazyka (VoiceXML 3.0), ktorý mal byť postavený na novom koncepte troch vrstiev - dáta, tok dialógu a prezentácia. Práce na tejto verzii jazyka však neboli doteraz ukončené. Posledná pracovná verzia jazyka bola publikovaná v roku 2010.

VoiceXML jazyk je možné označiť za najdôležitejší a hlavný jazyk zo skupiny W3C SIF jazykov, pričom ostatné jazyky ho doplňajú najmä SRGS, SISR, SSML a CCXML). Tomuto jazyku sa budeme venovať aj neskôr v samostatnej kapitole.

Odporúčanie pre zápis gramatík - W3C SRGS

Ako bolo spomenuté v predchádzajúcej kapitole, časť značiek používaných v jazyku VoiceXML na popis rečových gramatík bola použitá ako základ novej špecifikácie pre zápis gramatík pre systémy automatického rozpoznávania reči (SRGS - Speech Recognition Grammar Specification (SRGS). V skutočnosti avšak tieto značky pôvodne vznikli na základe syntaxe definovanej Java Speech Grammar Format (JSGF). SRGS špecifikácia definuje jazyk, ktorý umožňuje zapisovať kontextovo-nezávislé deterministické gramatiky pre rečový vstup od používateľa a tiež vstup vo forme DTMF voľby.

Podľa SRGS gramatika môže byť zapísaná buď v XML formáte alebo podľa tzv. ABNF (Augmented Backus-Naur form) syntaxe. Práca na tomto štandarde (SRGS) bola začatá v roku 1999. V marci 2004 bola sfinalizovaná prvá verzia W3C odporúčania - SRGS 1.0, ktorá bola akceptovaná ako štandard pre zápis deterministických rečových gramatík. [10].

Hlavnou výhodou gramatík podľa SRGS odporúčania, v porovnaní s ostatnými formami zápisu, je ich veľmi dobrá čitateľnosť tak pre človeka ako aj pre počítačové systémy. Príklad jednoduchej rečovej gramatiky v XML formáte, ktorá umožňuje rozpoznať jeden z dní v týždni, je možné vidieť na Obr. 10.2.

Sila SRGS odporúčania spočíva tiež v kooperácií s ostatnými W3C odporúčaniami, najmä s odporúčaním pre sémantickú interpretáciu (SISR - Semantic Intepretation for Speech Recognition) [11]. Pridaním sémantických značiek podľa SISR odporúčania, je možné realizovať proces sémantickej interpretácie veľmi efektívne priamo na základe rečovej gramatiky.

W3C SRGS forma zápisu gramatík sa stala široko akceptovaným štandardom, čo dosvečuje veľké množstvo platforiem a systémov, ktoré podporujú takéto gramatiky.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<grammar root="main" version="1.0" xml:lang="en">

  <rule id="main" scope="public">
    <one-of>
      <item> Sunday </item>
      <item> Monday </item>
      <item> Tuesday </item>
      <item> Wednesday </item>
      <item> Thursday </item>
      <item> Friday </item>
      <item> Saturday </item>
    </one-of>
  </rule>
</grammar>
```

Obr. 10.2: Príklad zápisu rečovej gramatiky podľa SRGS odporúčania v XML forme.

Odporúčanie pre sémantickú interpretáciu - W3C SISR

Ďalším "jazykom"v skupine W3C SIF je špecifikácia pre sémantickú interpretáciu, ktorá definuje elementy a atribúty, ktoré môžu byť vložené do rečových gramatík pre následnú extrakciu sémantického výsledku. Týmto spôsobom je možné efektívnejšie a komplexnejšie extrahovať význam prehovoru používateľa, ktorý je doručený späť do VoiceXML aplikácie. Príklad rečovej gramatiky, ktorá obsahuje SISR sémantické značky je zobrazený na Obr. 10.3.

Práce na tomto odporúčaní začali v apríli 2003 a v apríli 2007 sa špecifikácia stala oficiálnym W3C odporúčaním.

10.1.2 W3C Multimodal Interaction

Vzhľadom na rapídny posun od unimodálnej komunikácie medzi človekom a strojom k multimodálnej interakcii, čo je podporované dostupnosťou nových typov zariadení, existuje aj v tejto oblasti potreba štandardizácie. Pre tento účel bola vytvorená samostatná pracovná skupina s názvom W3C Multomodal Interaction Working Group (MIWG), ktorá vyvíja odporúčania pre multimodálnu interakciu.

Vzhľadom na komplexnosť multimodálnych systémov, tieto zahŕňajú veľké množstvo rôznych technológií, ktoré využívajú veľkú skupinu jazykov, štandardov a odporúčaní. Tieto definujú napríklad spôsob riadenia pohybu avatarov resp. robotov, získanie vstupu z modalít a jeho interpretáciu, fú-

```
<?xml version="1.0" encoding="UTF-8"?>
<grammar root="main" version="1.0" xml:lang="en">

  <rule id="main" scope="public">
    <one-of>
      <item><ruleref uri="#yes"/></item>
      <item><ruleref uri="#no"/></item>
    </one-of>
  </rule>
  <rule id="yes">
    <one-of>
      <item>yes</item>
      <item>yeah <tag>yes</tag></item>
      <item>sure <tag>yes</tag></item>
    </one-of>
  </rule>
  <rule id="no">
    <one-of>
      <item>no</item>
      <item>not <tag>no</tag></item>
      <item>nope <tag>no</tag></item>
    </one-of>
  </rule>
</grammar>
```

SISR content

Obr. 10.3: Príklad zápisu rečovej gramatiky s vloženými sémantickými značkami podľa SISR odporúčania.

ziu modalít a tiež distribúciu informácie do výstupných modalít alebo tiež manažment správania sa takýchto systémov.

Odporúčania, ktoré boli vyvinuté pod záštitou W3C Multimodal Interaction Group, sú združené pod spoločným názvom W3C Multimodal Interactive Framework. Jedná sa o nasledujúce odporúčania:

- Multimodal Architecture and interfaces
- Extensible Multi-Modal Annotations (EMMA)
- Emotion Markup Language (EmotionML) 1.0
- InkML - an XML language for digital ink traces

V ďalšom texte sa pozrieme podrobnejšie na prvé tri odporúčania, ktoré je možné považovať za najdôležitejšie.

W3C Multimodal Architecture and interfaces

Odporúčanie W3C Multimodal Architecture and Interfaces je možné považovať za hlavné odporúčanie tejto skupiny. Popisuje architektúru multimodálneho systému a navrhuje rozhrania medzi jeho komponentmi. Hlavnou úlohou tejto špecifikácie bolo poskytnúť všeobecný a flexibilný framework, ktorý zlepší interoperabilitu komponentov multimodálnych systémov a ich prenositeľnosť. Autori sa zamerali na definovanie základnej infraštruktúry pre riadenie aplikácií a platformových služieb. Odporúčanie tiež definuje zoznam základných komponentov architektúry multimodálnych systémov a tzv. run-time diagram architktúry. Nasledujúce komponenty sú identifikované ako základné stavebné prvky takejto architektúry:

- **Manažér interakcie**, ktorý koordinuje rôzne modality.
- **Dátový komponent**, ktorý poskytuje dátové modely
- **Komponenty modalít**, ktoré poskytujú interakčné schopnosti špecifické k daným modalitám.
- **Runtime framework**, ktorý poskytuje základnú infraštruktúru plus umožňuje komunikáciu medzi komponentmi.

Vymenované komponenty sú usporiadané v run-time architektúre, ktorá je zobrazená na Obr. 10.4

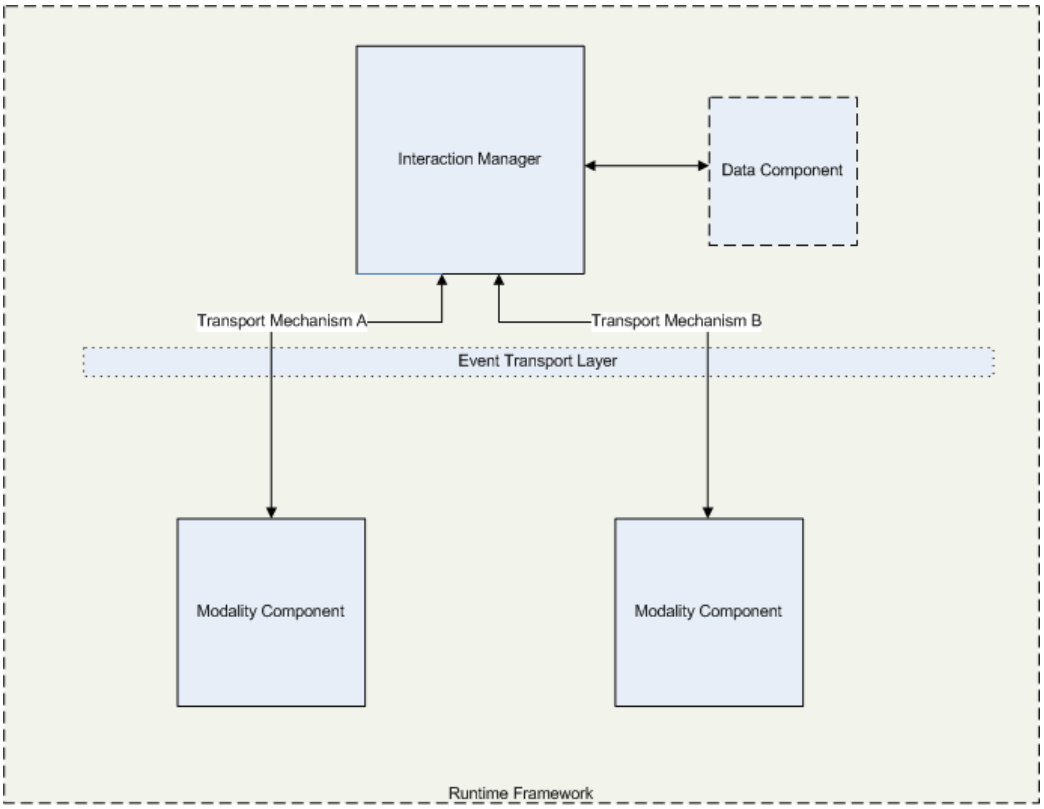
Ako je možné pozorovať, stupeň abstrakcie je v prípade tejto architektúry vyšší, ako tomu bolo pri architektúre hlasového prehliadača. To je spôsobené vyššou variabilitou takýchto architektúr, ktorá je spôsobená použitím rôznych modalít, interakčných scenárov alebo zamýšľaných platforiem a zariadení.

Konkrétnejším príkladom multimodálneho systému, môže byť nasledujúci systém (Obr. 10.5) , ktorý kombinuje webový a hlasový prehliadač.

W3C Extensible Multi-Modal Annotations (EMMA)

Ďalšou špecifikáciou v rámci W3C Multimodal Interactive Framework je EMMA - Extensible Multi-modal Annotations. EMMA je značkovací jazyk, ktorý definuje formát pre výmenu dát, čím vlastne vytvára rozhranie medzi komponentmi spracovania vstupov a blokmi interakčného manažmentu.

Definuje skupinu elementov a atribútov, ktoré umožňujú presnú reprezentáciu anotácie interpretácie vstupu. Predpokladá sa, že tieto značky budú použité primárne ako štandardný formát pre výmenu dát (data interchange format) medzi komponentmi multimodálneho systému. Spravidla bude tento

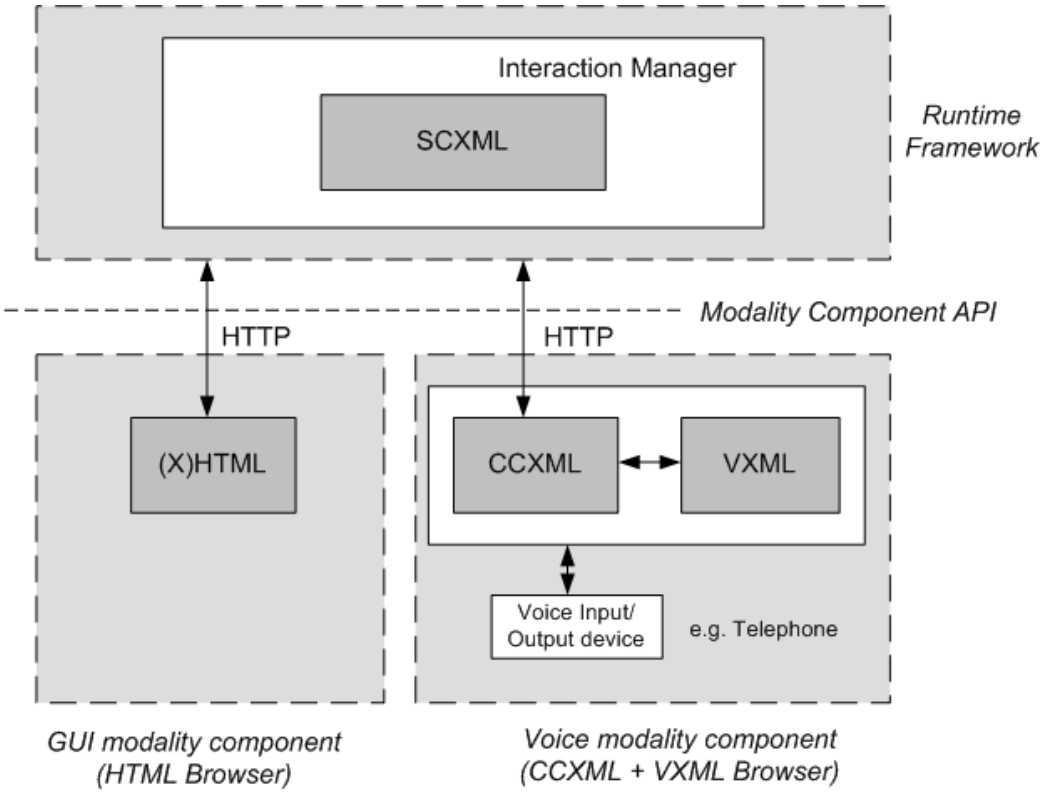


Obr. 10.4: Run-time architektúra multimodálneho interaktívneho systému

formát automatický generovaný interpretačnými komponentmi na reprezentáciu významu používateľského vstupu (NLU, ASR, ...).

EMMA 1.0 je W3C odporúčaním od roku 2009. Príklad EMMA dokumentu je zobrazený na Obr. 10.6

Takýto XML kód môže byť generovaný blokom porozumenia prirodzeného jazyka (NLU) ako popis výsledku rozpoznávania reči a jeho významu. Ako môžeme vidieť, dvojica <one-of> </one-of> značiek (riadky 8 až 21) obsahuje dve hypotézy interpretácie prijatého vstupu. Každá interpretačná hypotéza je zapuzdrená v <emma:interpretation> elemente (riadky 11 až 14 a 16 až 19). Analyzované vstupné vyjadrenie "from Košice to Budapešť"sa nachádza v atribúte *emma:tokens*. Extrahovaná informácia o význame sa nachádza v dvojici XML elementov - <origin> a <destination>. Interpretačné bloky obsahujú tiež informáciu o dôveryhodnosti oboch interpretácií (atribút *emma:confidence*



Obr. 10.5: Architektúra multimodálneho interaktívneho systému kombinujúceho hlasový a webový browser

```
<emma:emma version="1.0"
    xmlns:emma="http://www.w3.org/2003/04/emma"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.w3.org/2003/04/emma
     http://www.w3.org/TR/2009/REC-emma-20090210/emma.xsd"
    xmlns="http://www.example.com/example">

  <emma:one-of id="r1" emma:start="3432342" emma:end="3438342"
     emma:medium="acoustic" emma:mode="voice">

    <emma:interpretation id="int1" emma:confidence="0.75" emma:tokens="from Košice to Budapešť">
      <origin>Košice</origin>
      <destination>Budapešť</destination>
    </emma:interpretation>

    <emma:interpretation id="int2" emma:confidence="0.59" emma:tokens="from Košeca to Budapešť">
      <origin>Košeca</origin>
      <destination>Budapešť</destination>
    </emma:interpretation>

  </emma:one-of>
</emma:emma>
```

Obr. 10.6: Príklad zápisu EMMA dokumentu

W3C Emotion Markup Language (EmotionML)

Vzhľadom na rozširovanie sa rozhraní medzi človekom a strojom (HMI - Human-machine interface), ktoré sa čím ďalej tým viac podobajú ľuďom alebo sa prejavujú podobným správaním ako ľudia, aj naše očakávania voči ním stúpajú. Čoraz častejšie očakávame od počítačových systémov správanie podobné človeku, čo zahrňa aj rozpoznávanie a porozumenie emócií a tiež prejavovanie emócií strojmi.

Pre zefektívnenie práce s emóciami v HMI bolo potrebné navrhnúť jednotný jazyk pre popis a interpretáciu emócií. W3C odporúčanie Emotion Markup Language (EmotionML) 1.0 reflektuje na túto potrebu [20]. Umožňuje štandardizovaným spôsobom reprezentovať emócie pre HMI aplikácie, kde je určené pre manuálnu anotáciu dát, automatické rozpoznávanie emócií a tiež generovanie emočne-relevantného správania sa systému.

```
<emma:emma version="1.0" xmlns:emma="http://www.w3.org/2003/04/emma"
        xmlns="http://www.w3.org/2009/10/emotionml">
    <emma:interpretation emma:start="1245790094000" emma:end="1245790095000"
                                            emma:mode="voice" emma:verbal="false">

        <emotion category-set="http://www.w3.org/TR/emotion-voc/xml#everyday-categories">
            <category name="bored" value="0.1" confidence="0.1"/>
        </emotion>

    </emma:interpretation>
</emma:emma>
```

Obr. 10.7: Príklad zápisu EMMA dokumentu s popisom emócií podľa EmotionML odporúčania

```
<?xml version="1.0"?>
<speak version="1.1" xmlns="http://www.w3.org/2001/10/synthesis"
         xmlns:emo="http://www.w3.org/2009/10/emotionml"
         xml:lang="en-US">

    <s>
        <emo:emotion category-set="http://www.w3.org/TR/emotion-voc/xml#everyday-categories">
            <emo:category name="worried" value="0.4"/>
        </emo:emotion>

        Do you need help?
    </s>

</speak>
```

Obr. 10.8: Príklad zápisu SSML dokumentu s popisom emócií podľa EmotionML odporúčania

Obr. 10.7 ponúka príklad popisu emócií pomocou EmotionML jazyka, ktorý je vložený v EMMA dokumente, kde je skombinovaná významová analýza vstupu od používateľa s informáciou o emócii, ktorá bola detegovaná z tohto vstupu. V tomto príklade bola analyzovaná neverbálna vokalizácia

(emma:mode=”voice” emma:verbal=”false”), teda vyjadrenie emócie pomocou neverbálneho výrazu realizovaného hlasom (napríklad zívanie). Emócia, ktorá bola vyjadrená je popísaná párom <emotion> a </emotion> elementov v riadkoch 6 až 8. Atribúty detegovanej emócie sú popísané atribútmi <category> elementu. V tomto prípade bol detegovaný emočný typ ünudený"s úrovňou intenzity 0,1 a mierou spoľahlivosti na úrovni 0,1.

EmotionML jazyk môže byť tiež použitý s jazykom SSML. V tomto prípade môže slúžiť pre ovplyvnenie výslednej emócie, prezentovanej pomocou syntetického hlasu. Jednoduchý príklad takéhoto použitie sa nachádza na Obr. 10.8

10.2 Rozhrania komponentov v rečových interaktívnych komunikačných systémoch

Definovanie rozhraní medzi komponentmi interaktívnych systémov je nemenej dôležité v porovnaní s ostatnými technológiami. Dôležitosť štandardizácie inter-komponentových rozhraní spočíva v komplexnosti takýchto rozhraní, pričom je vyžadovaná zväčša spolupráca viacerých komplexných technológií. Navyše, jednotlivé komponenty sú vyvíjané množstvom výrobcov, resp. sú výsledkami práce rôznych výskumných tímov. Aby teda bolo možné ich spojiť do jedného interaktívneho systému, musia tieto komponenty implementovať spoločné aplikačné rozhranie (API).

Aj keď sme skupinu technológií pre návrh rečových interaktívnych systémov rozdelili okrem iného na jazyky a rozhrania, niektoré jazyky z predchádzajúcej podkapitoly je možné vnímať ako rozhrania medzi jednotlivými komponentmi interaktívnych systémov. Typickým príkladom je jazyk pre TTS systémy - SSML, kde SSML dokumenty môžu tvoriť rozhranie medzi blokom systému generovania prirodzeného jazyka a blokom samotnej syntézy reči (TTS).

Na druhej strane, môžeme identifikovať skupinu technológií, ktoré označujeme skôr ako rozhrania. O nich bude reč v nasledujúcich podkapitolách, aj keď hneď prvý predstaviteľ tejto skupiny bude vlastne výnimkou.

10.2.1 Java Speech API

Java Speech API (JSAPI) vyvinula spoločnosť Sun Microsystems, Inc. Pomocou JSAPI je možné začleniť rečové technológie do používateľského rozhrania k appletom a aplikáciám, ktoré sú založené na Java technológiách. Java Speech API definuje spôsob podpory príkazov a riadenia systémov rozpoznávania a syntézy reči. [JSAPI08-dizert]

Java Speech API pozostáva z dvoch jazykov:

Java Speech API gramatický formát (JSGF)

Java Speech Grammar Format (JSGF) je platformovo-nezávislá textová reprezentácia gramatík pre použitie pri rozpoznávaní reči. Gramatiky determinujú na čo sa má systém rozpoznávania reči zamerať, a teda popisujú čo môže používateľ povedať. JSGF preberá štýl a konvencie Java programovacieho jazyka a využíva tradičný zápis gramatík.

Java Speech API Markup Language (JSML)

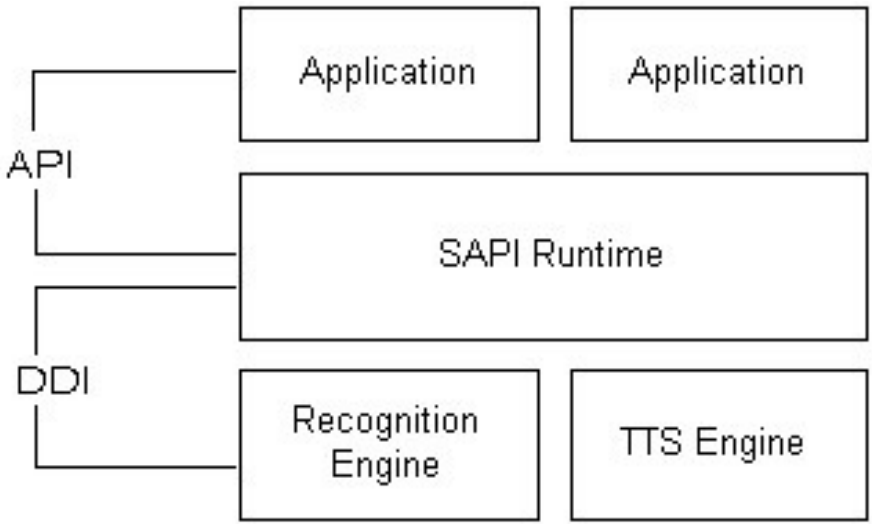
Java Speech API Markup Language je textový formát používaný aplikáciami na anotovanie textového vstupu rečových syntetizátorov. JSML definuje elementy, ktoré popisujú štruktúru dokumentu, poskytujú výslovností slov a fráz, indikujú frázovanie, dôraz (prízvuk), výšku tónu a rýchlosť reči, a riadia ostatné dôležité aspekty reči. Jazyk SSML, definovaný W3C konzorciom vznikol na základe JSML značiek, ktoré boli súčasťou prvej verzie jazyka VoiceXML.

10.2.2 Microsoft Speech API

Pri vymenúvaní rôznych typov rozhraní pre rečové technológie nesmieme zabudnúť na aktivitu spoločnosti Microsoft v tejto oblasti. Prirodzenou snahou Microsoftu bolo umožniť používanie technológií automatického rozpoznávania reči a syntézy reči z textu vo svojom operačnom systéme Windows. Pre jednoduchú implementáciu rečových technológií a následne tvorby aplikácií Microsoft predstavil v roku 1995 prvú verziu tohto rozhrania (SAPI 1), ktoré bolo určené pre MS Windows 95 a NT 3.51. Čo sa týka ďalšieho vývoja tohto rozhrania, je možné nasledujúce verzie rozdeliť do dvoch skupín. Prvá skupina verzií 1-4 sú si navzájom podobné, pričom každá novšia verzia ponúkala prídavné funkcie. Do druhej skupiny radíme verzie s označením 5 (5.1 - 5.4), ktoré predstavujú v porovnaní s predchádzajúcou skupinou, úplne nové rozhranie.

Podstatou MS SAPI rozhrania je oddelenie tzv. "low-level enginovöd samotných aplikácií. Hlavnou myšlienkou je jednoduchá tvorba rečových aplikácií, kde ich tvorca nemusí poznať detaily samotných systémov rozpoznávania a syntézy reči (enginov) na to, aby systémy použil v navrhovanej aplikácií. Jediné čo musí poznať je samotné rozhranie. Tento princíp ilustruje aj nasledujúci Obr. 10.9

MS SAPI oddeľuje aplikácie a rečové ënginy"pomocou rozhrania (SAPI Runtime). To umožňuje, aby boli aplikácie nezávislé na použitých rečových technológiach do tej miery, že tieto môžu byť nahradené inými, bez toho, aby bolo potrebné meniť kód aplikácií. Na druhej strane, všetky rečové enginy (rozpoznávače reči a syntetizátory reči), ktoré implementujú MS SAPI roz-



Obr. 10.9: Architektúra MS SAPI rozhrania

hranie môžu byť jednoducho integrované do prostredia operačného systému Windows a môžu byť používané aj všetkými aplikáciami, ktoré využívajú MS SAPI rozhranie.

Medzi ďalšie výhody využívania MS SAPI je možné zaradiť aj to, že v rámci operačného systému Windows poskytujú aj vstavané ënginy"pre automatické rozpoznávanie reči a syntézu reči z textu pre niekoľko najviac rozšírených svetových jazykov.

Microsoft Speech API (MS SAPI) je niekedy vnímané medzi developermi aj v negatívnom slova zmysle, vzhľadom na nekompatibilitu medzi jeho verziami a tiež pre komplikovanosť samotného rozhrania.

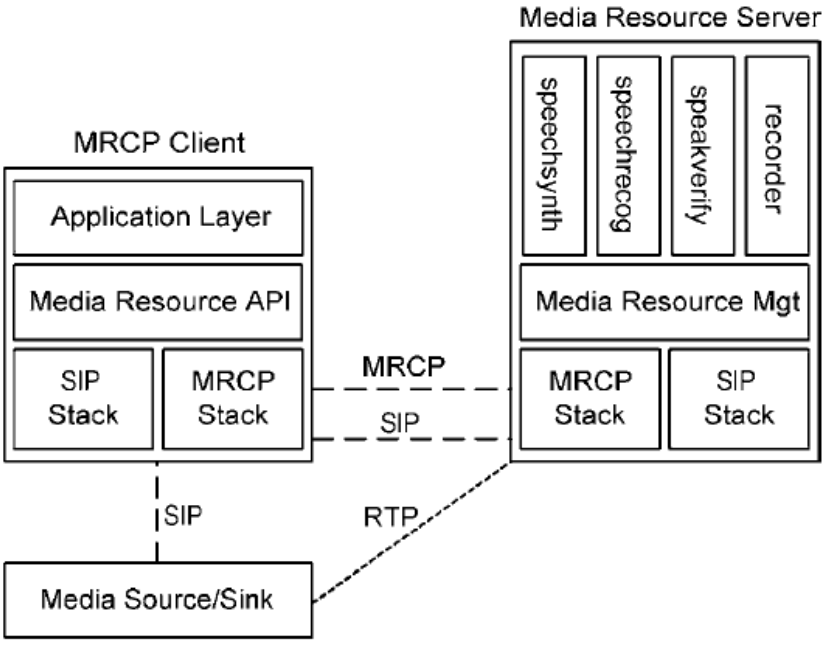
10.2.3 Media Resource Control Protocol

Media Resource Control Protocol (MRCP) je protokol navrhnutý pre komunikáciu medzi komponentmi, kde definuje výmenu dát a komunikáciu vo vnútri distribuovaných platforiem, ktoré poskytujú hlasové služby.Umožňuje riadiť zariadenia ako sú syntetizátory reči, systémy automatického rozpoznávania reči a iných cez telekomunikačnú sieť. Pre tento účel integruje dva dôležité protokoly - RTSP (Real Time Streaming Protocol) a SIP (Session Initiation Protocol), ktoré umožňujú riadiť spojenie a streamovanie z/do externých zariadení.

MRCP taktiež umožňuje implementovať distribuované platformy rečových dialógových systémov, ktoré využívajú VoiceXML. Protokol definuje požiadavky, odpovede a udalosti, ktoré riadia manažment zdrojov. Umožňuje

definovať stavový automat pre každé zariadenie a prechody medzi stavmi pre každú požiadavku.

Aktuálnou verziou rozhrania MRCP je verzia 2.0. Všeobecná architektúra klient-server MRCP komunikácie je zobrazená na Obr. 10.10



Obr. 10.10: Architektúra klient-server MRCP komunikácie

MRCP bolo vyvinuté spoločnosťami Cisco Systems, Inc., Nuance Communications a Speechworks, Inc a stalo sa industriálnym štandardom pre medzikomponentovú komunikáciu v rámci rečových platforiem. Je implementované vo viacerých komerčných systémoch ako sú: IBM WebSphere Voice Server, Microsoft Speech Server, LumenVox Speech Engine alebo Nuance Recognizer.

10.2.4 Web Speech API

Vo februári 2013 Google priniesol novú verziu Google Chrome web prehliadača verzie 25 (v.25) s podporou nového Web Speech API rozhrania.

Web Speech API je JavaScriptové rozhranie, ktoré umožňuje integrovať rečové technológie do webových stránok a aplikácií. Web Speech API bolo predstavené skupinou W3C Web Speech API Community Group v roku 2012. Umožňuje webovým developerom integrovať do webových stránok a aplikácií

rečové technológie, konkrétne poskytnutie rečového vstupu a syntézu reči z textu (TTS - text-to-speech) vo webovom prehliadači.

Web Speech API spolu s HTML 5 umožňujú využívať rečové technológie od Google priamo vo webovom prehliadači. Spojením týchto technológií sa stalo používanie rečového rozhrania omnoho jednoduchším a praktickejším. Jedným z dôvodov je aj fakt, že Google rozpoznávač umožňuje rozpoznávanie reči pre veľkú skupinu jazykov (viac ako 100) v porovnaní s iným poskytovateľmi rečových služieb.Navyše, rozpoznávanie od Googlu, vďaka použitiu neurónových sietí, poskytuje vysokú presnosť rozpoznávania reči.

Aby bolo možné takto presné rozpoznávanie reči, v rôznych zariadeniach, napr. v smartfónoch, kde je limitovaný výkon a aj spotreba energie, je využité riešenie typu klient-server. To znamená, že zatiaľčo je audio signál zaznamenaný v používateľovom zariadení (PC, notebook, tablet, smartfón, a pod.), samotné rozpoznávanie reči sa deje na vzdialenom serveri a výsledok rozpoznávania je doručený späť na zariadenie používateľa (cez web). Praktické skúsenosti ukazujú, že sa jedná o veľmi dobrý prístup, ktorý prináša aj ďalsie výhody, ako je možnosť neustále zlepšovať technológie bez toho, aby tým bol obmedzovaný samotný používateľ (napr. nutnosťou aktualizovať softvér vo svojom zariadnení.)

10.3 Nástroje a platformy

Vzhľadom na vysokú komplexnosť jednotlivých rečových technológií, často nie je možné navrhnúť a realizovať tieto technológie öd podlahy". Navyše, samotné algoritmy sú jazykovo-nezávislé a preto je výhodné využívať už pripravené technológie, ktoré často stačí naplniť jazykovo-špecifickými dátami v podobe modelov, slovníkov a pod.

Cieľom tejto kapitoly je poskytnúť prehľad nástrojov, ktoré môžu byť užitočné pri vývoji rečových rozhraní a pre implementáciu jednotlivých rečových technológií. Tieto sme rozdelili do nasledujúcich kategórií:

- **Nástroje pre rozpoznávanie reči**
- **Nástroje pre syntézu reči z textu**
- **Nástroje na modelovanie jazyka**
- **Nástroje na riadenie dialógu**
- **VoIP softvérové ústredne**

10.3.1 Nástroje pre rozpoznávanie reči

Systémy automatického rozpoznávania reči (ARR) sú komplexnými systémami, ktoré vyžadujú špecializované modely pre svoju činnosť. Nástroje na prípravu ARR systémov rozdelíme do dvoch kategórií. Prvú kategóriu budú predstavovať open-source a free nástroje, ktoré umožňujú pripravovať korpusy dát, trénovať modely a tiež pripraviť kompletný ARR systém. Navyše ich zdrojové kódy bude možné modifikovať pre ich optimalizáciu alebo adaptovanie pre konkrétne riešenia. Z tejto kategórie sa budeme venovať trom systémom - HTK/ATK, Julius toolkit a systému Kaldi.

Druhú kategóriu tvoria systémy, ktoré je možné použiť iba ako celok, tak ako sú. Zväčša sa jedná o komerčné systémy, ktoré môžu byť použité zadarmo pre nekomerčné použitie, resp. umožňujú limitované použitie bez poplatkov. Prevažne ide o systémy, ktoré pracujú v server-klient režime, kde ARR systém je lokalizovaný na serveroch poskytovateľa a používateľ môže poslať svoj hlas ako audio nahrávku alebo stream cez preddefinované rozhranie a získať späť textový ekvivalent. V rámci nasledujúcej kapitoly si stručne popíšeme systém od Googlu a od spoločnosti Microsoft.

HTK/ATK

The Hidden Markov Model Toolkit (HTK) s An Application Toolkit (ATK) predstavuje sadu nástrojov, ktoré umožňujú realizovať komplexný proces prípravy a skrytých Markovových modelov (HMM), vybudovať kompletný systém rozpoznávania reči a jeho aplikácie.

Bol vyvinutý v Machine Intelligence Laboratory na Cambridge University (Engineering Department). Neskôr v roku 1999 získal HTK Microsoft pričom licenciu stále vlastní pôvodné laboratórium.

HTK ponúka skupinu modulov a nástrojov vo forme knižníc napísaných v jazyku C. Tieto knižnice vykonávajú analýzu reči, trénovanie HMM modelov, testovanie a analýzu výsledkov. HTK obsahuje tiež dva rozpoznávacie systémy (enginy) HVite a Hdecode, ktoré sú primárne určené na testovanie.

V porovnaní s iným toolkitmi, HTK poskytuje obsiahlú veľmi nápomocnú dokumentáciu a mnoho príkladov. Úspech HTK nástrojov demonštruje veľké množstvo rozpoznávačov reči, ktoré sú postavené na báze HTK.

Okrem základných nástrojov HTK je možné použiť aj aplikačné rozhranie ATK (Real-Time API for HTK). ATK umožňuje budovať experimentálne aplikácie na báze HTK. Pridáva C++ vrstvu k HTK knižniciam, ktorá umožňuje jednoduchšie vytvoriť funčný systém s funkcionalitou rozpoznávania reči.

Viac informácií ako aj oba toolkity je možné stiahnúť z nasledujúcej ad-

resy:

http://htk.eng.cam.ac.uk/develop/atk.shtml

Julius toolkit

Ďalším veľmi populárnym open-source nástrojom pre rozpoznávanie reči je Julius toolkit (`http://julius.osdn.jp/en_index.php`).

Julius je ucelený systém pre rozpoznávanie plynulej reči s veľkým slovníkom (LVSCR - Large Vocabulary Continous Speech Recognition Engine). Vývoj systému Julius začal v roku 1997 a aktuálne prebieha pod záštitou *Interactive Speech Technology konzorcia* (ISTC).

Julius predstavuje vysoko výkonný dekóder, ktorý pracuje v reálnom čase. Ponúka rozpoznávanie reči založené na kontextovo-závislých HMM akustických modeloch a na slovných N-gramových jazykových modeloch.

Veľkou výhodou systému Julius je, že pracuje so štandardnými formátmi, čo ho robí kompatibilným s iným nástrojmi, ako napr. HTK. Nemenej dôležité je to, že je distribuovaný pod voľnou licenciou spolu so zdrojovými kódmi.

Kaldi

Ďalším dôležitým a významný voľne dostupným toolkitom na rozpoznávanie reči je Kaldi (http://kaldi-asr.org/). Principiálne je podobný systému HTK avšak jeho výhodou je že implementuje novší a viac efektívny princíp, využívaný na rozpoznávanie reči, a síce hlboké neurónové siete (Deep Neural Networks - DNN).

Kaldi je vyvíjaný od roku 2009 na Johns Hopkins University. Je napísaný v jazyku C++ a je poskytovaný pod Apache Licence v 2.0.

Aj napriek tomu, že ide o najmladší systém zo zatiaľ vymenovaných, veľmi rýchlo sa stal jedným z favorizovaných a obľúbených najmä z dôvodu, že dosahuje lepšie výsledky na úlohe LVCSR.

Windows Speech recognizer

Do úplne inej skupiny patria rozpoznávací systém od spoločnosti Microsoft, pomenovaný ako *Windows Speech Recognition system*.

Microsoft ponúka svoj rozpoznávací systém ako súčasť svojho operačného systému (ver.8.0 - Vista, Win 7,8). Systém teda nie je voľne distribuovaný, napriek tomu stojí za zmienku, nakoľko vďaka MS SAPI rozhraniu môže byť pomerne jednoducho integrovaný do aplikácií vyvinutých pre operačný systém Windows, resp. tento rozpoznávač používajú aplikácie od spoločnosti Microsoft (napr. MS Office).

ASR systém od Microsoftu aktuálne podporuje niekoľko základných jazykov (angličtina, francúzština, španielčina, nemčina, japončina, zjednodušená čínština a tradičná čínština). Podpora konrétneho jazyka súvisí s korešpondujúcou lokalizáciou daného Windowsu.

Pre jednoduchšie využitie svojho rozpoznávača reči Microsoft poskytuje nástroje pre integrovanie jeho funkcií vo vývojovom prostredí Microsoft Visual Studio.

Google serverový ASR systém

Najvýznamnejším poskytovateľom funkcionality rozpoznávania reči v súčasnosti je bezpochyby spoločnosť Google.

Google poskytuje túto funkcionalitu cez webové rozhranie, ktoré využíva JavaScriptové Web Speeh API rozhranie v rámci HTML5 štandardu.

Rozpoznávací systém od spoločnosti Google je postavený na koncepte server-klient, kde ASR systém je lokalizovaný na serveroch Googlu a používatelia ho používajú v móde klienta, čo znamená, že ich zariadenia iba posielajú rečový signál do tejto vzdialenej služby a spätne dostávajú výsledok rozpoznávacieho procesu. Toto riešenie umožňuje veľmi kvalitné rozpoznávanie reči nezávisle na výkonnosti hárveru používateľa, čo je veľmi dôležité v súčasnom svete smart zariadení (smartfónov, tabletov, IoT zariadení), ktorých výkon je nedostatočný pre výkonné LVCSR systémy.

10.3.2 Nástroje na modelovanie jazyka

Deterministické jazykové modely vo forme rečových gramatík je možné relatívne jednoducho vytvárať manuálne v textových editoroch alebo v špeciálnych "grammar builderoch".

Inak je to však v prípade stochastických jazykových modelov, ktoré vyžadujú fázu trénovania často na veľmi veľkých objemoch dát a preto sa nezaobídu bez špecializovaných trénovacích nástrojov.

Existuje viacero trénovacích nástrojov, ktoré môžu byť použité. Nasledujúce tri však môžeme považovať za najčastejšie používané a teda najpopulárnejšie:

- SRI Language Modeling Toolkit (SRILM)
- MIT Language Modeling Toolkit (MITLM)
- IRSTLM Toolkit

Najpopulárnejšou skupinov nástrojov pre štatistické modelovanie jazyka je SRI Language Modeling Toolkit, ktorý obsahuje nástroje pre trénovanie, estimáciu, evaluáciu, kombinovanie a adaptáciu rozličných jazykových modelov (N-gramových modelov). Tent toolkit tiež poskytuje nástroje pre segmentáciu a označovanie dát v textových korpusoch.

Ďalším populárnym nástrojom je MIT LM Toolkit, ktorý v porovnaním so SRILM poskytuje tiež iné adaptačné a kombinačné techniky.

IRSTLM toolkit pridáva navyše iba jednu prerezávaciu techniku založenú váhovanej rozdielovej metóde.

Všetky tri toolkity podporujú štandardný ARPA formát pre jazykové modely.

10.3.3 Nástroje pre syntézu reči z textu

TTS systémy môžeme rozdeliť do dvoch skupín na komerčné systémy a voľne dostupné.

Komerčné TTS systémy sa zameriavajú hlavne na väčšie svetové jazyky, ako sú angličtina, francúzština, taliančina, španielčina, a iné, avšak v poslednom období sa zlepšila aj podpora menších jazykov, vďaka Googlu. Pre komerčné systémy je tiež typická vysoká kvalita výstupnej reči, čo sa nie vždy dá povedať o voľne dostupných systémoch. TTS systémy IVONA, CereProc alebo NeoSpeech môžeme považovať za reprezentntov vysoko kvalitných komerčných TTS systémov.

Na druhej strane je skupina voľne dostupných TTS systémov a toolkitov, ktoré môžu byť použité na prípravu syntetických hlasov alebo aj komplétnych TTS systémov. Takéto systémy sú zvyčajne dostupné pod licenciami, ktoré umožňujú ich použitie pre nekomerčné účely. Tieto systémy sú najčastejšie využívané v akademickej komunite pre výskum a vývoj v oblasti syntézy reči z textu.

Do tejtot skupiny najviac populárnych TTS tooklitov patria: Festival, HTS, MARY TTS a MBROLA.

Festival

Festival toolkit je skupina nástrojov, ktoré tvoria framwerok pre vytváranie systémov syntézy reči z textu a na prípravu syntetických hlasov. Bol vyvinutý v Center for Speech Technology Research na Univerzite v Edinburgu. Jeho obľúbenosť podčiarkuje fakt, že pomocou neho bolo vytvorených množstvo TTS systémov pre viaceré jazyky.

Festival môže byť použitý v troch používateľských scenároch:

- priame použitie Festival TTS systému a jeho voľne dostupných hlasov (anglický a španielský) na generovanie syntetickej reči
- použitie Festivalu na vytvorenie TTS systému pre vlastnú aplikáciu
- použitie Festivalu na výskum a vývoj nových metód rečovej syntézy a na prípravu nových hlasov.

Festival podporuje tri metódy syntézy reči - difónovú konkatenatívnu syntézu, korpusovú syntézu a HMM syntézu (štatistický parametrický prístup). Obsahuje taktiež nástroj **Fextvox** pre vytváranie nových syntetických hlasov. TTS systém a hlasy, pripravené pomocou nástroja Festival môžu byť použité tiež vo viac kompaktnom a rýchlejšom nástroji **Flite**.

HMM-based Speech Synthesis System - HTS

Ďalším veľmi populárnym nástrojom je systém HTS - the HMM-based Speech Synthesis System (HTS). Jedná sa o toolkit, ktorý ponúka viacero dôležitých nástrojov pre trénovanie modelov pre prípravu syntetických hlasov a pre samotný TTS systém, ktorý je založený na princípe parametrickej rečovej syntézy s použitím HMM modelov.

HTS vyvinula skupina HTS working group. Toolkit bol implementovaný ako modifikovaná verzia HTK toolkitu pre trénovanie akustických modelov pre rozpoznávanie reči. HTS umožňuje trénovať modely pre parametrickú rečovú syntézu. Trénovací proces môže byť chápaný ako proces nastavovania parametrov kontextovo-závislých modelov (HMM modelov) podľa označených rečových nahrávok, ktoré vstupujú do tohto procesu. Následne sú tieto modely použité na generovanie parametrov rečového syntetizátora.

HTS toolkit neobsahuje nástroje pre analýzu vstupného textu a jeho spracovanie, preto musí na túto úlohu využívať iné systémy, napr. Festival, MARI TTS, Flite a pod.

MARY TTS

Systém MaryTTS je tradičný nástroj ktorý umožňuje vytvárať TTS systémy. Tento nástroj bol vyvinutý v laboratóriu *Language Technology Lab na DKFI* (Das Deutsche Forschungszentrum für Künstliche Intelligenz GmbH/German Research Institute of Artificial Intelligence) a v inštitúte *Institute of Phonetics at Saarland University.* Následne sa práce na tejto platforme presunuli pod pracovnú skupinu *Multimodal Speech Processing Group in the Cluster of Excellence MMCI and DFKI.*

MaryTTS ponúka TTS enginy, ktoroé podporujú viacero jazykov (nemčinu, angličtinu, francúzštinu, taliančinu, švédčinu, ruštinu...). Naviac MaryTTS ponúka toolkit, ktorý umožňuje jednotuch pridávať nové jazyky a vybudovať TTS systém založený na metóde výberu jednotiek alebo na báze HMM modelov. MaryTTS systém bol integrofaný do veľkého množstva rozhraní medzi človekom a strojom, cez možnosť vytvorenia tzv. konektora (rozhrania) na MaryTTS engine.

10.3.4 Nástroje na riadenie dialógu

Situácia s dostupnými nástrojmi na riadenie dialógu a modulmi je horšia v porovnaní s dostupnosťou toolkitov pre rozpoznávanie alebo syntézu reči.

Jednotky riadenia dialógu sú často integrálnou časťou celého rečové dialógového systému alebo platformy. Existuje iba niekoľko dialógových manažérov, ktoré je možné relatívne jednoducho použiť vo vlastnej aplikácií alebo systéme. Jedným z dôvodov tejto situácie je aj fakt, že vzhľadom na komplexnosť danej problematiky, stále neexistuje konsenzus o najvhodnejšej metóde pre riadenie dialógu.

Hoci sa jazyk VoiceXML stal akýmsi priemyselným štandardom pre riadenie interakcie a bol použitý v mnohých komerčných platformách, predsalen nepredstavuje dostatočne všeobecný prístup k samotnej problematike riadenia dialógu a môže byť použitý iba limitovane, najmä v aplikáciach s dobre definovanou štruktúrov doménovo-špecifického dialógu. VoiceXML sa navyše absolútne nehodí pre neúlohovo-orientované dialógy resp. konverzácie.

Veľmi často teda pozorujeme použitie platformovo-špecifických prístupov, v prípade, že sa dá očakávať komplexnejšia dialógová interakcia. Vzhľadom na predchádzajúce skutočnosti sme vybrali štyroch reprezentantov rôznych prístupov, ktorí sú voľne dostupní.

JVoiceXML

JVoiceXML interpreter je jeden z mála voľne dostupných nekomerčných VoiceXML interpreterov. Bol vyvinutý tímom okolo Dirka Schnelle-Walka. JVoiceXML je napísaný v Jave a poskytuje otvorenú architektúru. Impelementuje Java rozhrania Java Speech API (JSAPI) a Java Telephone API (JTAPI). JVoiceXML podporuje odporúčania VoiceXML 2.0 a 2.1.

RavenClaw

RavenClaw dialógový manažér reprezentuje iný prístup k riadeniu dialógu v porovnaní s DDL (Dialog Description Languages) jazykmi. V tomto prípade

sa jedná o systém založený na plánovaní resp. ägende", ktorý využíva dve dátové štruktúry - *strom úloh* a *agendu*.

Strom úloh reprezentuje plán pre vykonanie doménovo-špecifických úloh. Agenda má formu usporiadaného zoznamu agentov, ktorí slúžia na spojenie vstupu s vhodným agentom v stromovej štruktúre úloh.

Činnosť RavenClaw systému je riadená vstupmi (sémantickou reprezentáciou prichádzajúceho vstupu). Keď sa objaví konkrétny vstup, jeho agent je aktivovaný, a vykoná jeho spracovanie. RavenClaw môže byť použitý aj na riadenie dialógovej interakcie v multimodálnych systémech, pretože nie je závislý na type vstupných dát. Spolieha sa na sémantické koncepty. Systém bol vytvorený dvojicou autorov Rudnicky a Bohus (pozri [10]).

RavenClaw dialógový manažér je integrálnou časťou architektúry systému RavenClaw/Olympus, ktorý predstavuje kompletný rečový/multimodálny interaktívny systém vyvinutý na Carnegie Mellon University (CMU).

TrindiKit

TrindiKit toolkit je nástroj na vytvorenie dialógovej interakcie, založený na myšlienke tzv. ïnformačných stavov"navrhnutej Traumom a kol. v [87] a [43]. TridiKit bol vyvinutý v rámci troch projektov - TRINDI, SIRIDUS a TALK v Centre for Language Technology (CLT) na Univerzite v Gothenburgu.

TrindiKit systém má dva hlavné komponenty - *reprezentáciu informačných stavov* (Information State Representation) a *modul dialógových zmien* (DME - Dialogue Move Engine), ktorý mení informačné stavy podľa pozorovaného progresu dialógu.

Vhodná zmena v dialógu (dialog move) je generovaná ako výstup sýstemu. Informačný stav je tu chápaný ako štruktúra, kde agent ukladá informácie potrebné pre dokončenie úlohy. Progres dialógu závisí od zmeny informačného stavu.

TrindiKit sa nesnaží byť kompletnou architektúrou, skôr špecifikovať formáty pre definíciu informačných stavov, pravidlá pre ich zmenu, dialógový progres a asociované algoritmy.

Na vybudovanie modulu dialógových zmien je potrebné definovať pravidlá pre update ifnormačných stavov, definovať prechody medzi stavmi a algoritmy, ako aj internú štruktúru informačných stavov.

DialogFlow

V súčasnosti sme svedkami príchodu nových zariadení, akými sú virtuálni asistenti v podobe aplikácií (Apple SIRI, Microsoft Cortana, Google Now a pod.) resp. v podobe inteligentných zariadení, ako napr. Amazon Echo alebo

Google Home, ktoré prinášajú nové výzvy do oblasti riadenia dialógu. V súčasnosti je možné konštatovať, že dialógová interakcia s týmito zariadeniami má charakter izolovaných výmen otázok a odpovedí (question-answer) avšak snahou je priniesť zákazníkom väčší zážitok z interakcií a teda presunúť sa skôr ku konverzáciam, podobne ako v prípade chatbotov.

Jedným zo spôsobov, ako vyvinúť takúto dialógovú interakciu pre interaktívne zariadenia a aplikácie ako sú Google Assistant, Amazon Alexa alebo Facebook Messenger je využitie nástroja **DialogFlow** od Googlu. Pôvodným tvorcom systému DialogFlow je skupina Api.ai, ktorú kúpil Google v roko 2016.

Výhodou tohto systému je výborná dostupnosť dokumentácie a podpory pre vytvorenie virtuálnych asistentov, resp. chatovacích robotov (chatbotov). Systém je založený na strojovom učení.

10.3.5 VoIP softvérové ústredne

V začiatkoch rečových interaktívnych služieb, boli tieto poskytované cez telefónne siete, nakoľko sa zrodili ešte v ére analógovej telefónie. Navyše v 90tých rokoch neexistovali smartfóny a nebolo bežné mať internet v mobile, čo nahrávalo myšlienke rozvíjanej W3C Voice Browser Working Group, ktorá sa snažila o vývoj tzv. hlasového prehliadača, ktorý by poskytoval informácie dostupné na internete cez rečový dialóg vedený po telefóne. Tento koncept avšak časom ustúpil s príchodom smartzariadení a s rastúcou dostupnosťou internetu v nich. Pre poskytovanie automatických rečových dialógových služieb cez telefónnu sieť bolo nutné využiť ústredne, ktoré pripájajú celý rečový dialógový systém (RDS) do telefónnej siete. V dobe analógovej telefónie sa jednalo o hardverové ústredne, ktoré boli prepojené so servermi, na ktorých RDS bežali. Neskôr s nástupom VoIP technológie, teda internetovej telefónie boli tieto hardverové ústredne nahradené softvérovými VoIP ústredňami.

Najznámejšou a najpoužívanejšou voľne dostupnou (open source) softvérovou VoIP ústredňou je **Asterisk**. Asterisk je VoIP brána, ktorá umožňuje budovať aplikácie s telefónnym rozhraním.

Asterisk umožňuje vytvárať IP PBX systémy, VoIP brány, konferenčné servery a iné. Výborne sa hodí pre IVR (Interactive Voice Response) riešenia, pretože podporuje prehrávanie a nahrávanie audia, získanie DTMF voľby, prácu s databázami a web službami a tiež rozpoznávanie a syntézu reči.

11

Návrh rečových aplikácií v jazyku VoiceXML

Táto kapitola je zameraná na proces návrhu hlasových služieb založených na jazyku VoiceXML. V úvode sú diskutované základné princípy návrhu služieb, ktoré prispievajú k zlepšeniu použiteľnosti a vnímaniu kvality hlasovej služby. V dalšej časti je uvedený krátky úvod do návrhu a prípravy VoiceXML aplikácií a rečových gramatík podla SRGS a SISR odporúčaní W3C konzorcia. V závere kapitoly je opísaný príklad VoiceXML aplikácie spolu s jej zdrojovým kódom.

11.1 Základné princípy návrhu

Pri návrhu aplikácií a služieb v jazyku VoiceXML je veľmi dôležitá otázka „*Čo znamená navrhnúť hlasovú službu*?" Na základe tejto otázky môžeme zadefinovať, že samotný návrh znamená navrhnúť:

- štruktúru a tok dialógu/dialógov,
- výzvy (otázky) systému a očakávané reakcie (odpovede) používateľa,
- rečové gramatiky, ktoré vymedzujú prípustné výpovede používateľa,
- a prístup do informačných databáz a iných zdrojov.

Je potrebné poznamenať, že každá vymenovaná položka (krok) má svoje vlastné pravidlá, ktoré by mali byť zohľadnené v procese návrhu aplikácie.

11.1.1 Štruktúra a tok dialógu

Štruktúra dialógu a tok dialógu sú determinované najmä na základe informácií, ktoré musia byť získané prostredníctvom dialógu a na základe štruktúry informácií, ktoré majú byť poskytnuté používateľovi.

Prvým krokom v procese návrhu by mala byť analýza používateľských scenárov a úlohy, ktorá by mala byť dokončená dialógovou interakciou. Klient, ktorý plánuje poskytovať navrhnutú hlasovú službu, by mal **formulovať hlavný cieľ** požadovanej služby. Následne by sa mali identifikovať **hodnoty**, ktoré je potrebné **získať od užívateľa** v dialógovej interakcii, spolu s typom prezentovaných informácií. Dialóg by mal byť navrhnutý takým spôsobom, aby bolo možné získať všetky požadované hodnoty potrebné pre vykonanie a dokončenie úlohy.

V tejto fáze je potrebné vykonať viacero rozhodnutí, ktoré sa týkajú stratégie dialógu, opravy chýb alebo stratégie potvrdzovania.

Stratégie dialógu

Existujú tri rôzne typy **stratégie dialógu**:

- stratégia s iniciatívou na strane systému (v angl. *system-initiative dialog*),
- stratégia na strane používateľa (v angl. *user-initiative dialog*),
- stratégia so zmiešanou iniciatívou (v angl. *mixed-initiative dialog*).

Stratégiu s iniciatívou na strane systému je možné charakterizovať ako interakciu, v ktorej systém kladie otázky a používateľ na nich odpovedá. Takáto stratégia dialógu obyčajne vedie k úspešnému dokončeniu úloh bez zvýšeného kognitívneho zaťaženia používateľa, avšak často je dialóg v takomto prípade príliš dlhý a únavný. Túto stratégiu je možné považovať za najprimitívnejší prípad interakcie, ktorý má najnižšie požiadavky na moduly systému, akými sú ASR, TTS, NLU a vedie k dobrej spoľahlivosti služby. Interakcia v dialógu s iniciatívou na strane systému môže vyzerať nasledovne:

Systém: Vitajte v službe Predpoveď počasia.
S: Pre ktoré mesto si želáte vyhľadať Predpoveď počasia?
Používateľ: Košice.
S: Na ktorý deň?
P: Piatok.
S: Zvolili ste mesto Košice a deň piatok?
P: Áno.
S: ...

Opačný scenár je vykonaný v prípade interakcie s iniciatívou na strane používateľa, kedy používateľ kladie otázky a systém na nich odpovedá (ako je uvedené v príklade nižšie). Takýto scenár interakcie kladie vysoké nároky na komponenty systému (najmä na jednotky ASR a NLU) kvôli vysokej variabilite možných výpovedí používateľa. Tento fakt je tiež dôvodom, prečo sú dialógy s iniciatívou na strane používateľa menej úspešné v reálnych aplikáciách.

Systém: Vitajte v službe Predpoveď počasia.
Používateľ: Dobrý deň, chcel by som zistiť predpoveď počasia pre Košice.
S: Rozumiem.
P: Na piatok.
S: Predpoveď pre Košice na piatok.
P: Áno, správne.

Dialógy so zmiešanou iniciatívou najlepšie modelujú dialóg medzi ľuďmi. V takomto prípade sa dialógová iniciatíva presúva od jedného účastníka k druhému, podľa potreby interakcie. Jazyk VoiceXML podporuje dialógy so zmiešanou iniciatívou v špecifickej, obmedzenej verzii, ktorá kladie aj obmedzené požiadavky na moduly hlasovej platformy. Stratégia zmiešanej iniciatívy jazyka VoiceXML umožňuje systému položiť používateľovi otázku typu „*Ako Vám môžem pomôcť*?", na ktorú môže používateľ odpovedať prirodzeným spôsobom (prirodzenou výpoveďou). Používateľ môže vo svojej odpovedi zhrnúť všetky informácie požadované pre vykonanie/dokončenie úlohy. Ak niektoré hodnoty chýbajú, systém preberie iniciatívu a vyzve používateľa, aby doplnil chýbajúce informácie. Takýto dialóg so zmiešanou iniciatívou môže vyzerať nasledovne:

Systém: Vitajte v službe Predpoveď počasia.
S: Ako Vám môžem pomôcť?.
Používateľ: Dobrý deň, potrebujem zistiť predpoveď počasia pre Košice, na piatok.
S: Predpoveď pre Košice na piatok?
P: Áno, správne.

Jazyk VoiceXML umožňuje vytvárať len dialógy s iniciatívou na strane systému a obmedzenú verziu dialógov so zmiešanou iniciatívou, ako bolo spomenuté vyššie. Z toho dôvodu je dôležité na začiatku návrhu hlasovej služby rozhodnúť o type iniciatívy v dialógu, pretože to súvisí s možnosťami platformy a návrhom výziev systému a gramatík.

Stratégie opravy chýb

Nasledujúca časť pojednáva o stratégiách opravy chýb v dialógu. Proces opravy chýb a „zotavenia" systému po ich výskyte môže využívať viacero stratégií. V prípade jazyka VoiceXML je situácia jednoduchšia a chyby, ktoré sa môžu vyskytnúť, sú zhrnuté do nasledovných kategórií:

- chyba (udalosť) typu **Nomatch** - reprezentuje situácie, v ktorých používateľ poskytne vstupnú výpoveď, ktorá sa nezhoduje s aktívnou gramatikou v konkrétnom dialógu;
- chyba (udalosť) typu **Noinput** - reprezentuje situácie, v ktorých používateľ neposkytne žiaden hlasový vstup v stanovenom časovom intervale;
- **Nepsrávne rozpoznanie** bez výskytu udalosti Nomatch - reprezentuje situácie, v ktorých bol hlasový vstup používateľa nesprávne rozpoznaný, ale zhoduje sa s aktívnou gramatikou a tiež miera dôveryhodnosti bola vysoká;
- **Iné systémové chyby** - reprezentujú iné chyby systému, ktoré sa môžu vyskytnúť v dialógu počas interakcie.

Udalosti *Nomatch* a *Noinput* patria do kategórie štandardných udalostí definovaných v jazyku VoiceXML a poskytujú riadiace mechanizmy na zotavenie systému zo spomínaných chýb. Okrem toho existuje aj všeobecný spôsob zachytávania chýb pomocou elementu `<catch>`.

V prípade udalostí alebo chýb typu *Nomatch* a *Noinput*, jedna z najjednoduchších stratégií zotavenia je zopakovanie poslednej výzvy systému, ktorý opakovanie vyzve používateľa na zadanie hlasového vstupu. Na tento účel

môže byť využitý element `<repeat>`. Lepšie riešenie môže využívať stratégiu s viacerými vrstvami s nasledujúcimi krokmi [73]:

1. Oznám používateľovi, čo sa stalo (napr. „*Prepáčte, systém nezachytil Vašu odpoveď.*").

2. Potom, povedz používateľovi, čo má robiť (napr. „*Zopakujte prosím svoju odpoveď.*").

3. Poskytni používateľovi viac informácií o tom, čo má ďalej robiť (poskytni príklad očakávanej odpovede, napr. „*Zadajte prosím mesto a štát, napríklad: L.A. a California.*").

4. Ak je to potrebné, oboznám používateľa s pomocnými položkami (napr. „*Povedzte prosím -pomoc- pre viac informácií o povolených vstupoch.*").

V prípade nesprávneho rozpoznania používateľského vstupu bez zachytenia tejto udalosti systémom môže nastať nepriaznivá situácia. V tomto prípade by mal používateľ iniciovať proces zotavenia sa z chyby, čo môže priniesť systému náročnú situáciu. V takýchto prípadoch by mal byť používateľ navigovaný na zadanie jedného z univerzálnych navigačných príkazov, ktoré mu môžu pomôcť v návrate k neprávne rozpoznanému vstupu a umožniť mu ho opraviť. Medzi typické univerzálne navigačné príkazy patria [73]:

- *zopakovať (repeat)*,
- *zrušiť (cancel)*,
- *späť (back)*,
- *podpora (backup)*,
- *pomoc (help)*,
- *ukončiť (quit)*,
- *koniec (exit)*.

Podpora týchto príkazov je platformovo závislá, avšak jazyk VoiceXML definuje príkazy *pomoc* a *koniec*, reprezentované elementami `<help>` a `<exit>`, ktoré vyžadujú gramatiku štandardnej platformy s príslušnými príkazmi, aktívnu počas celej dialógovej interakcie.

Potvrdenie vstupu

Potvrdenie vstupu (v angl. *input confirmation*) je ďalšia dôležitá vlastnosť alebo schopnosť systému vyžadovaná v dialógovej interakcii. Podobne, ako v prípade dialógu typu človek-človek, aj v prípade dialógu typu človek-stroj, sa môžu vyskytnúť situácie, kedy získaná informácia musí byť potvrdená, pretože tu môže byť nejaká úroveň neurčitosti. Stratégie potvrdzovania môžu byť rozdelené na *implicitné* a *explicitné* metódy.

Stratégia implicitného potvrdenia (viď nasledujúci príklad) zahŕňa akt potvrdenia priamo do nasledujúcej výzvy systému, čo predstavuje ďalšiu funkciu komunikácie. Táto metóda uchováva trvanie dialógu a môže byť vnímaná ako viac prirodzená, avšak riziko chyby je vyššie, pretože rečové gramatiky schopné zachytiť takéto výpovede - vstupy používateľa, sú výrazne komplexnejšie.

Systém: Pre ktoré mesto si želáte získať predpoveď počasia?
Používateľ: Košice.
S: Pre ktorý deň si želáte predpoveď pre Košice?
a) P: Nie Košice, Viedeň.
b) P: Na piatok.

Na druhej strane, *stratégia explicitného potvrdenia* poskytuje spoľahlivejší spôsob potvrdenia, kde potvrdenie je vykonané prostredníctvom novej výzvy systému bez ďalšej funkcie komunikácie (viď nasledujúci príklad). Má výrazne nižšie požiadavky na rečové gramatiky, ale zväčšuje dialóg a môže byť vnímaný ako nudný alebo únavný.

Systém: Pre ktoré mesto si želáte získať predpoveď počasia?
Používateľ: Pre Košice.
S: Zvolili ste Košice?
P: Áno.
S: Pre ktorý deň?
P: Piatok.
S: Zvolili ste piatok?
P: Áno.

Podmienené potvrdenie (v angl. *conditional confirmation*) predstavuje prístup, ktorý môže pomôcť eliminovať nevýhody stratégie explicitného potvrdenia. Táto metóda zahŕňa *skóre dôveryhodnosti* (v angl. *confidence score*) rozpoznaných používateľových výpovedí (vstupov) na rozhodnutie o nutnosti potvrdenia. Ak dané skóre prekročí určitý prah (obyčajne vyšší ako 90%), daná výpoveď môže byť vyhodnotená ako dostatočne dôveryhodná a potvr-

denie nemusí byť vykonané.

Ďalšie vylepšenie môže byť dosiahnuté spojením potvrdenia niekoľkých vstupných položiek do jednej explicitnej potvrdzujúcej výzvy. Namiesto žiadania používateľa o potvrdenie po získaní každej vstupnej hodnoty, potvrdenie môže byť vykonané po zhromaždení niekoľkých položiek.

Vlastnosti vstupu a výstupu

V iniciálnom návrhu toku a štruktúry dialógu by mali byť vykonané aj ďalšie rozhodnutia. Je potrebné rozhodnúť o type systémového výstupu, ktorý môže byť vo forme syntetizovanej reči alebo prednahratých audio súborov. Zatiaľčo využitie TTS systému ponúka väčšiu flexibilitu, výstupná kvalita umelej reči stále nie je perfektná. Na druhej strane, prednahratá reč s príjemným hlasom môže poskytnúť štúdiovú kvalitu, tomuto prístupu však chýba flexibilita. Pre určité jazyky, použitie audio nahrávok je jediná možná cesta riešenia systémového výstupu.

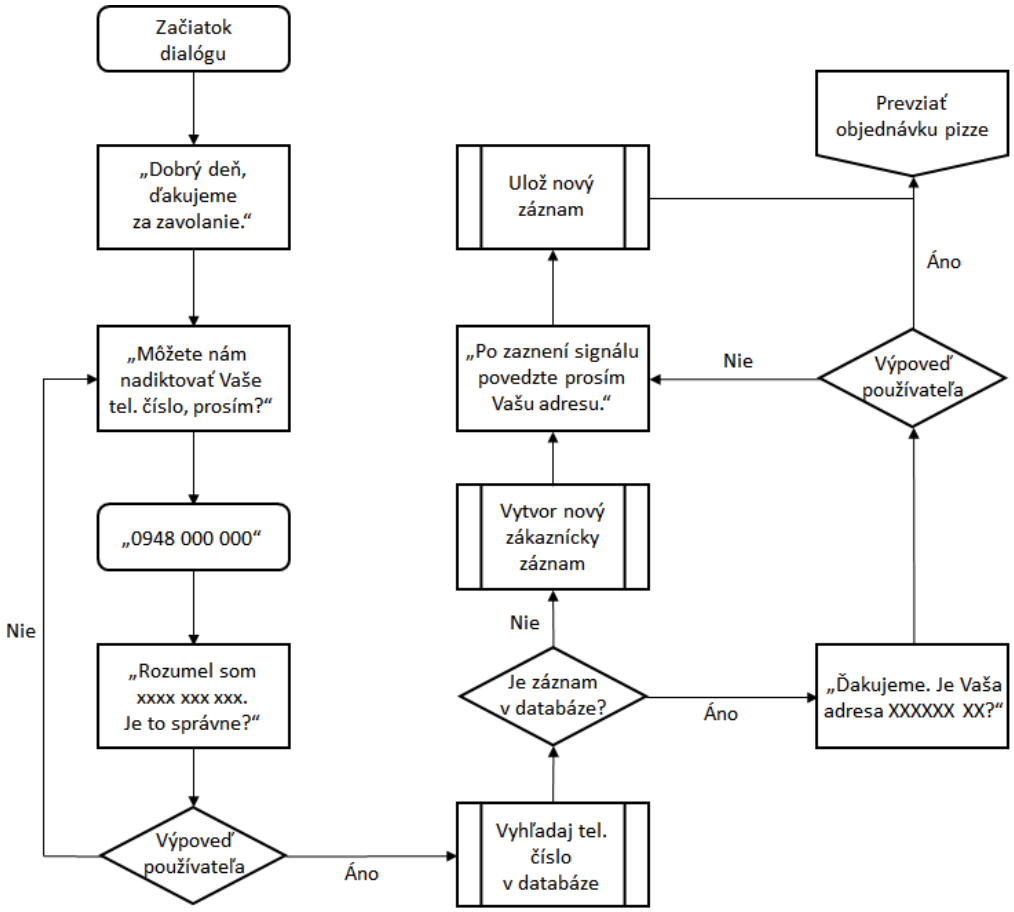
Ak sa bližšie pozrieme na vstupy systému, je zrejmé, že nie každá vstupná informácia môže byť systému doručená hlasom. V prípade PIN kódov alebo iných identifikačných alebo osobných informácií, musí byť zvolený iný **typ vstupu**. Použitie tónov **DTMF (Dual-Tone Multi-Frequency)** alebo inej vstupnej modality (virtuálna klávesnica) by malo byť zvážené pre vyššiu bezpečnosť. DTMF tóny, ktoré sa vzťahujú na tlačidlá telefónu, môžu byť nápomocné tiež v situáciách, kedy je hovorená komunikácia znehodnotená alebo narušená vplyvom šumu alebo iných nepriaznivých podmienok.

Ďalšia voľba súvisí s možnosťou prerušenia výzvy systému užívateľom. Táto možnosť sa voľne prekladá ako „skákanie do reči" (v angl. *barge-in*). Prerušenie iného účastníka dialógu patrí do skupiny často používaných interakčných vzorov, ktoré môžu významne zrýchliť dialógovú interakciu. VoiceXML platformy by mali podporovať funkciu barge-in, ktorá môže byť aktivovaná/deaktivovaná pomocou atribútu elementu `<prompt>`.

Po vyriešení všetkých otázok, diagram dialógu môže byť načrtnutý vo forme **diagram toku dialógu**. Uzly takéhoto diagramu reprezentujú otázky systému a prechody reprezentujú možné reakcie používateľa. Príklad diagramu toku dialógu pre rečovú dialógovú službu donášky pizze je uvedený na Obr. 11.1.

11.1.2 Písanie výziev a gramatík

Na písanie systémových výziev môže byť formulovaných niekoľko odporúčaní, ktoré prispejú ku spoľahlivejším a ergonomickejším hlasovým službám.



Obr. 11.1: Príklad diagramu toku dialógu pre dialógovú službu donášky pizze

Vhodná formulácia systémových výziev významne ovplyvňuje odpovede používateľa ako aj rečové gramatiky, ktoré by im mali tiež zodpovedať.

Úroveň „otvorenosti” systémových výziev determinuje, ako veľmi bude systémová otázka (výzva) obmedzujúca. *Otvorené výzvy* sú menej obmedzujúce a umožňujú používateľovi použiť voľnejší jazyk. Typickým príkladom otvorenej výzvy je výzva typu:

„*Ako Vám môžem pomôcť*?”,

ktorá môže byť zodpovedaná akýmkoľvek vyjadrením hovoreného jazyka. Je zrejmé, že napísať vhodnú deterministickú rečovú gramatiku pre takýto typ vstupov bude náročná úloha. Príprava štatistického jazykového modelu môže byť vnímaná ako vhodné riešenie, avšak nie spolu s jazykom VoiceXML, pretože je navrhnutý prevažne na použitie s deterministickými gramatikami napísanými podľa SRGS a SISR odporúčaní. Samozrejme, spomenuté odpo-

rúčania ponúkajú riešenia tiež vo forme *garbage* modelov a výplní (*fillers*), avšak ich podpora závisí od hlasovej platformy (alebo SDS). Namiesto opísaných obtiažností, jazyk VoiceXML umožňuje používať otvorené výzvy v režime so zmiešanou iniciatívou, ale stále vyžaduje len vyplnenie preddefinovanej množiny hodnôt.

Uzavreté výzvy ostávajú na druhej strane. Tie priamo obmedzujú užívateľa len na výber z malej množiny možností. Typická uzavretá výzva môže vyzerať nasledovne:

„*Vyberte si jednu z možností, prosím: šport, filmy alebo hudba.*"

Takmer vo všetkých situáciách sa výzvy konštruujú niekde v rozmedzí od otvorených po uzavreté výzvy. Pri návrhu a písaní výziev systému by mali byť uvažované nasledovné odporúčania:

- *Navrhovať výzvy s vhodnou dĺžkou.* Dĺžka výziev je dôležitá vlastnosť. Výzvy by nemali byť príliš dlhé, pretože pozornosť používateľa sa môže znížiť. Príliš dlhé výzvy tiež predlžujú dialóg, ktorý potom môže byť pre používateľa nepohodlný. Na druhej strane, príliš krátke výzvy nemusia byť dostatočne jasné a vysvetľujúce.

- *Formulovať zdvorilé výzvy, ktoré sú príjemné pre používateľa.*

- *Zvážiť možnosti TTS systému* počas konštrukcie výziev. Občas sa môže stať, že niektorá kombinácia difón (alebo nejakých slov) môže byť nesprávne syntetizovaná v porovnaní s ostatnými. Syntetizované výzvy by mali byť vypočuté ešte pred tým, než bude dialógová aplikácia prezentovaná používateľovi, za účelom zaručenia zrozumiteľnosti a dostatočnej úrovne kvality služby.

- Konštruovať výzvy, ktoré sú dostatočne nápomocné a vedú používateľa k poskytnutiu informácie, ktorá je očakávaná systémom.

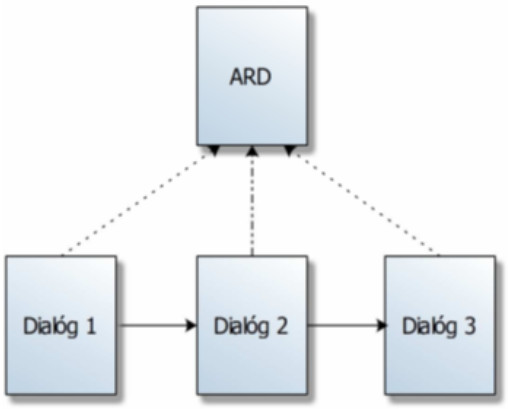
Iniciálny návrh dialógu, vrátane skonštruovaných výziev, môže byť vyhodnotený s využitím metódy **Wizard-of-Oz**, ktorá môže byť veľmi nápomocná pri vytváraní vhodných rečových gramatík pre partikulárne výzvy. Metóda *Wizard-of-Oz* je výskumná metóda, pri ktorej testovacie subjekty interagujú s počítačovým systémom, mysliac si, že daný systém je autonómny, avšak to nie je pravda. V tejto metóde je systém ovládaný alebo čiastočne ovládaný neviditeľnou osobou. Trénovaný operátor v experimente Wizard-of-Oz nahrádza dialógový manažér a manažuje interakciu s testovacími subjektami s cieľom získania informácie o správaní používateľa a jazyku, ktorý používateľ zvyčajne používa. Získaná odpoveď používateľa môže pomôcť dizajnérovi

služby navrhnúť rečové gramatiky, ktoré pokrývajú výpovede zvyčajne vyslovené používateľmi ako odpoveď na uvažované otázky (výzvy).

Iný, často používaný prístup k **príprave rečových gramatík** pre želaný dialóg je metóda brainstormingu, kedy sa dizajnér pokúša zhromaždiť všetky možné odpovede na navrhnuté systémové výzvy. Potom sa dizajnér snaží odhaliť logickú štruktúru takýchto odpovedí a následne transformovať túto logiku do pravidiel rečových gramatík.

11.2 Písanie VoiceXML aplikácií

VoiceXML aplikácie pozostávajú z jedného alebo viacerých VoiceXML dokumentov. Tieto súbory VoiceXML dokumentov sa vyznačujú príponou súborov s označením „.vxml". Hoci každý dokument môže byť spustený ako samostatná aplikácia, často môžu byť dokumenty združené spolu s využitím tzv. koreňového ARD dokumentu (z angl. *Application Root Document*). Tento dokument umožňuje nastaviť globálne premenné a nastavenia a tiež umožňuje zdieľať iné premenné, definované v tomto rozsahu aplikácie dizajnérom. Typická štruktúra VoiceXML aplikácie je uvedená na Obr. 11.2.



Obr. 11.2: Štruktúra typickej VoiceXML aplikácie

Všetky VoiceXML dokumenty začínajú s XML deklaráciou na prvom riadku:

```
<?xml version="1.0"?>
```

Obsah dokumentu je oddelený párom elementov `<vxml>` a `</vxml>`. V rámci tohto elementu (tagu) sa vyžaduje atribút verzie, ktorý určuje verziu VoiceXML a má byť použitý nasledovne:

```
<vxml version="2.0">
      obsah dokumentu ...
</vxml>
```

Každý VoiceXML dokument pozostáva z hlavičky a jedného alebo viacerých dialógov, ktoré môžu byť napísané cez elementy `<form>` alebo `<menu>`. Hlavička obyčajne obsahuje metadáta a deklarácie premenných spolu s rozsahom dokumentu (z angl. *document scope*). Na včlenenie metadát VoiceXML ponúka elementy `<metadata>` alebo `<meta>`. V nasledujúcom príklade na Obr. 11.3 sú zahrnuté metadáta typu *author* s ich hodnotou. Premenná `welcome_mess` je deklarovaná s iniciálnou hodnotou `Welcome`. Typ premennej vo VoiceXML je nastavený podľa prvej zahrnutej hodnoty. V ukázanom príklade je typ premennej *text*.

```
<?xml version="1.0" encoding="UTF-8"?>
<vxml xmlns="http://www.w3.org/2001/vxml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/vxml
   http://www.w3.org/TR/voicexml20/vxml.xsd"
   version="2.0">

 <meta name="author" content="Stanislav Ondas"/>

 <var name="welcome_mess" expr="'Welcome'"/>

   ...

 </vxml>
```

Obr. 11.3: Príklad hlavičky VoiceXML dokumentu

Hlavnými stavebnými jednotkami VoiceXML aplikácie sú formy a menu elementy, ktoré reprezentujú partikulárny dialóg. Každá forma má jedinečný názov a obsahuje inštrukcie na vykonanie daného dialógu. Formy sú reprezentované párom elementov `<form>` a `</form>`. Každá forma obsahuje povinný atribút „id", ktorý definuje názov danej formy. Definovanie názvu formy umožňuje odkazovanie sa na ňu z iného miesta aplikácie (v rámci rovnakého VoiceXML dokumentu alebo z iného dokumentu tej istej aplikácie) alebo inou aplikáciou. Formy môžu obsahovať rôzne elementy, ktoré realizujú úlohy požadované na vykonanie daného dialógu. Obr. 11.4 poskytuje príklad jednoduchej formy, ktorá sa pozdraví používateľovi vo forme pozdravu „Hello!".

Uvedený príklad je VoiceXML dokument len s jednou formou, ktorá obsahuje blok vykonávateľného obsahu zapuzdreného v elementoch `<block>` a

```xml
<?xml version="1.0" encoding="UTF-8"?>
<vxml xmlns="http://www.w3.org/2001/vxml"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.w3.org/2001/vxml
   http://www.w3.org/TR/voicexml20/vxml.xsd"
   version="2.0">

 <meta name="author" content="Stanislav Ondas"/>

 <form id="hello">
    <block>
       <prompt> Hello! </prompt>
    </block>
 </form>

 </vxml>
```

Obr. 11.4: Jednoduchá „Hello" aplikácia VoiceXML

`</block>`. Element `<prompt>` vnútri bloku slúži na vyzývanie používateľa s určitou vetou (výzvou), ktorá môže byť syntetizovaná s využitím TTS systému.

VoiceXML kód uvedený na Obr. 11.4 nie je možné použiť na získanie vstupu používateľa. Kód je schopný len povedať používateľovi „Hello!". Na prípravu reálne interaktívnej hlasovej služby musia byť použité ďalšie položky formy vnútri tela formy, zahŕňajúc tzv. *vstupné polia* (v angl. *input fields*), ktoré sú navrhnuté na zber používateľského vstupu.

Položky formy predstavujú skupinu elementov, ktoré môžu byť použité v rámci elementu `<form>` na realizáciu úloh spojených s vykonaním dialógu. Tieto položky môžu byť rozdelené do dvoch hlavných kategórií: *položky poľa* (v angl. *field items*) a *položky ovládania* (v angl. *control items*). Položky poľa zhromažďujú informácie od používateľa s cieľom naplniť premenné (premenná vstupného poľa, v angl. *field input variable*). Tie môžu obsahovať výzvy usmerňujúce používateľa, čo má povedať, ďalej gramatiky, ktoré definujú interpretáciu toho, čo je povedané a môžu tiež obsahovať manipulátory udalostí. Položky ovládania, t.j. bloky, uzatvárajú určitý vykonávateľný obsah, ale nemôžu byť použité na zhromažďovanie vstupu od používateľa.

Formy môžu obsahovať nasledovné **vstupné polia** (v angl. *input fields, field items*):

`<field>` zhromažďuje vstup od používateľa prostredníctvom reči alebo automatického DTMF rozpoznávania;

`<record>` umožňuje nahrať hlasový vstup používateľa ako audio nahrávku;

<transfer> umožňuje presmerovať hovor používateľa na iné telefónne číslo;

<object> vyvolá platformovo-špecifický objekt, ktorý môže zhromažďovať ďalšie typy používateľského vstupu a vracia výsledok vo forme ECMAScript objektu;

<subdialog> presmeruje dialóg do ďalšieho/iného dialógu alebo dokumentu a vracia výsledok vo forme ECMAScript objektu.

Položky ovládania je možné rozdeliť na dva typy:

<block> môže byť použitý na definovanie vykonávateľného obsahu a na prehratie nejakej správy používateľovi;

<initial> slúži na vytvorenie interakcie s používateľom so zmiešanou iniciatívou.

Každá položka formy má priradenú premennú položky formy, ktorá je prednastavená na hodnotu „undefined", keď sa interpretácia presunie do danej formy. Táto premenná položky formy slúži na ukladanie výsledku pri interpretácii formy. V prípade vstupných položiek je táto premenná nazývaná ako *premenná vstupnej položky* a uchováva hodnotu prijatú od používateľa. Premennej položky formy môže byť priradený názov (meno) prostredníctvom atribútu „name" alebo môže byť tiež ponechaná bez mena (v tom prípade je pre ňu platformou vygenerované interné meno).

Každá položka formy má tiež „**strážiacu podmienku**" (v angl. *guard condition*), ktorá je testovaná za účelom rozhodnutia, či môže byť daná položka formy zvolená na interpretáciu interpretačným algoritmom formy (skr. FIA, v angl. *form interpretation algorithm*). Prednastavená podmienka len testuje, či premenná položky formy má nejakú hodnotu. Ak je táto podmienka pravdivá, do takejto položky formy sa nebude pristupovať.

Najviac preferovaný spôsob, ako zhromažďovať vstup od používateľa, je vytvorenie jednoduchej formy s jedným elementom vstupného poľa `<field>`. Obr. 11.5 znázorňuje jednoduchú VoiceXML aplikáciu, ktorá vyzýva používateľa, aby povedal svoj obľúbený deň v týždni. Uvedený VoiceXML kód umožňuje dva nasledovné scenáre interakcie (viď Obr. 11.6).

Ukázaný VoiceXML kód pozostáva z jednej formy (elementy `<form>` a `</form>`), v ktorej je vnorené vstupné pole (element `<field>`) reprezentované na riadkoch 5 až 29. Toto vstupné pole má definovaný atribút „name", ktorý definuje meno jeho premennej vstupnej položky pod názvom „favorite_day". V tomto poli je zapuzdrená jednotka dialógu, ktorá je definovaná výzvou, rečovou gramatikou a časťou kódu vymedzenou elementami

```xml
<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0">

  <form id="hello">
    <field name="favorite_day">
      <prompt> What's your favorite day? </prompt>
      <grammar root="main">
        <rule id="main" scope="public">
          <one-of>
            <item>Sunday</item>
            <item>Monday</item>
            <item>Tuesday</item>
            <item>Wednesday</item>
            <item>Thursday</item>
            <item>Friday</item>
            <item>Saturday</item>
          </one-of>
        </rule>
      </grammar>

      <filled>
        <if cond="favorite_day=='Sunday'">
          Nice. Me too.
        <else/>
          <prompt>You said <value expr="favorite_day"/>.</prompt>
          <prompt>I do not like that.</prompt>
        </if>
      </filled>
    </field>
  </form>
</vxml>
```

Obr. 11.5: Jednoduchá „Favorite Day" aplikácia VoiceXML

```
Scenario 1:

System> What's your favorite day?
User> Sunday
System> Nice. Me too.

Scenario 2:

System> What's your favorite day?
User> Monday
System> You said Monday. I do not like that.
```

Obr. 11.6: Scenáre interakcie v jednoduchej „Favorite Day" aplikácii VoiceXML

`<filled>` a `</filled>`, ktorá bude vykonaná po zhromaždení vstupu od používateľa.

Vstupná výzva je špecifikovaná elementom `<prompt>` a bude prehraná používateľovi potom, čo sa interpretácia presunie do tohto poľa. Vstup do poľa tiež znamená, že proces rozpoznávania reči je tiež aktivovaný a po vyzvaní používateľa systémovou výzvou, systém bude čakať na prejav používateľa.

Všetky možné vstupy od používateľa sú definované rečovou gramatikou, ktorá môže byť špecifikovaná pomocou VoiceXML elementu `<grammar>`. Rečové gramatiky vo VoiceXML aplikáciách sú písané podľa W3C SRGS špecifikácie, ktorá definuje dva formáty gramatiky: XML a ABNF. V uvedenom príklade je gramatika napísaná vo formáte XML špecifikácie SRGS (riadky 7 až 19) a priamo vo VoiceXML dokumente, preto sa jedná o tzv. internú gramatiku (v angl. *internal grammar*). Z dôvodu komplexnosti rečových gramatík, najmä v reálnych aplikáciách, gramatiky sú častejšie konštruované ako separátne dokumenty, s príponou „*.grxml*". V takom prípade nazývame gramatiku externou gramatikou (v angl. *external grammar*). Obr. 11.7 poskytuje príklad na odkazovanie sa na súbor esternej gramatiky, ktorý má rovnaký obsah ako jej interná verzia v zdrojovom kóde na Obr. 11.5.

Ak používateľ poskytne jeho odpoveď na systémovú výzvu, jeho výpoveď je skonvertovaná do textu, ktorý je porovnaný s aktívnou gramatikou. Ak sa jeho výpoveď zhoduje s gramatikou, premenná vstupného poľa je naplnená jeho výpoveďou alebo jej sémantickou hodnotou. V našom príklade, premenná `favorite_day`, ak uvažujeme Scenár 1 z Obr. 11.6, bude naplnená textovým reťazcom „Sunday".

Následne sa spracovanie vstupného poľa presunie do elementu `<filled>`, ktorý je vykonaný po zhromaždení používateľského vstupu. V tejto časti je umiestnený element podmienenej logiky (riadky 22 až 27), ktorá rozhoduje o ďalšom toku dialógu podľa podmienky definovanej v atribúte `cond`. Ak je táto podmienka pravdivá, vykoná sa obsah podmienky `<if>`. V opačnom prípade je vykonaný obsah vetvy `<else>`.

Ak uvažujeme zdrojový kód z predošlého príkladu (Obr. 11.5), v tomto bode je interakcia zastavená, pretože nie je špecifikovaný žiadny prechod do inej formy alebo dokumentu. Na špecifikovanie prechodu do inej položky dialógu, iného dialógu alebo dokumentu, môže byť využitý element `<goto>`, ako je ukázané na príklade v zdrojovom kóde na Obr. 11.7. Element `<goto>` je použitý vnútri elementu `<filled>` (riadok 21), ktorý presmerováva dialóg do ďalšej formy. Môže využívať jeden z atribútov: `next`, `nextitem`, `expr` alebo `expritem` na definovanie cieľa, kde by mal dialóg pokračovať. Atribúty `next` a `nextitem` môžu využívať identifikátory (`id`) alebo mená nasledujúcej položky formy, formu (s využitím prefixu `#`) alebo VoiceXML dokument. Atribúty `expr` a `expritem` umožňujú definovať výraz, ktorý je vyhodnotený na získanie názvu položky, do ktorej by mal dialóg prejsť.

V nasledujúcom príklade, element `<goto>` obsahuje atribút `next` s me-

```
<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0">

  <form id="hello">
    <field name="favorite_day">
      <noinput>I didn't hear you.</noinput>
      <nomatch>Please, say one of the week day. </nomatch>
      <help>Days of the week are: Sunday, Monday,
      Tuesday, Wednesday, Thursday, Friday and Saturday.</help>

      <prompt> What's your favorite day? </prompt>
      <grammar src="days.grxml" type="application/srgs+xml"/>

      <filled>
        <if cond="favorite_day=='Sunday'">
          Nice. Me too.
        <else/>
          <prompt>You said <value expr="favorite_day"/>.</prompt>
          <prompt>I do not like that.</prompt>
        </if>
        <goto next="#goodbye"/>
    </filled>
    </field>
  </form>

  <form id="goodbye">
    <block>
      Goodbye my friend!
    </block>
  </form>
</vxml>
```

Obr. 11.7: VoiceXML aplikácia na získanie obľúbeného dňa používateľa s externými gramatikami a manipulátormi udalostí

nom nasledujúceho dialógu - `goodbye`. Odkazovanie sa na nasledujúcu formu vyžaduje znak prefixu `#`. V tomto kroku sa interpretácia formy s identifikátorom `hello` zastaví a interpretér vstupuje do dialógu `goodbye`, ktorý spôsobí prehratie výzvy „goodbye" používateľovi.

Interakcia, ktorá môže byť vykonaná ako výsledok interpretácie predošlého VoiceXML dokumentu, je ukázaná na Obr. 11.8. Scenár 1 ukazuje hladký a bezchybný dialóg. Úplne odlišná situácia je ilustrovaná v Scenári 2, kde sa vyskytli určité nedorozumenia. Prvý problém sa vyskytol, keď používateľ nezodpovedal na prvú systémovú výzvu v definovanom časovom intervale (riadky 10 a 11). Druhý problém nastal, keď používateľ poskytol neočakávanú odpoveď (riadok 13). Našťastie, na záver používateľ povedal pomocný príkaz „help", ktorý mu pomohol ísť v dialógu ďalej.

Ako môžeme vidieť v Scenári 2, nastali tri typy udalostí: *Noinput*, *Nomatch* a *Help*. VoiceXML poskytuje niekoľko spôsobov, ako zachytiť nielen

```
1 Scenario 1:
2
3 System> What's your favorite day?
4 User> Sunday
5 System> Nice. Me too.
6 System> Goodbye my friend!
7
8 Scenario 2:
9
10 System>  What's your favorite day?
11 User>    (user did not answer in defined time slot) NOINPUT EVENT!
12 System>  I didn't hear you.
13 User>    Yesterday                                  NOMATCH EVENT!
14 System>  Please, say one of the week days.
15 User>    help                                       HELP EVENT!
16 System>  Days of the week are: Sunday, Monday,
17          Tuesday, Wednesday, Thursday, Friday and Saturday.
18 User>    Saturday
19 System>  You said Saturday. I do not like that.
20 System>  Goodbye my friend!
```

Obr. 11.8: Scenáre interakcie vo „Favorite Day" aplikácii VoiceXML s prechodom k dialógu „goodbye" a manipulátormi udalostí

tieto udalosti a ako ich spracovať. Pre tieto tri typy udalostí sú vo VoiceXML špecifikácii definované tri typy špeciálnych elementov: `<noinput>`, `<nomatch>` a `<help>`. Je dostupný tiež element `<error>` na zachytenie iných typov chýb, zvyčajne generovaných systémom. Jazyk VoiceXML poskytuje tiež všeobecný mechanizmus na zaobchádzanie (manipulovanie) s udalosťami iného typu, ktoré sa môžu vyskytnúť počas dialógovej interakcie. Element `<catch>` môže byť použitý na zachytenie a spracovanie ďalších, platformovo-špecifických udalostí.

Elementy na zaobchádzanie s udalosťami môžu byť definované na jednej z aplikačných úrovní, ktoré definujú ich rozsah. Ak sú tieto elementy definované v koreňovom ARD dokumente, sú schopné zachytiť udalosti v každom mieste výskytu v aplikácii. Elementy na zaobchádzanie s udalosťami môžu byť definované na úrovni dokumentu, v hlavičke dokumentu. V tomto prípade môžu elementy zachytiť udalosti, ktoré sa vyskytujú v rovnakom dokumente. Manipulátory udalostí (v angl. *event handlers*) definované na úrovni dialógu (vnútri elementu `<form>`) spracovávajú udalosti v rámci tohto dialógu. V našom príklade na Obr. 11.7, manipulátory udalostí (riadky 7 až 9) sú definované na úrovni alebo v rozsahu „anonymous" čo znamená, že môžu zachytiť len udalosti, ktoré sa vyskytnú vnútri rovnakej položky formy (v našom prípade v elemente `<field>`, ktorý začína na riadku 5). Všetky tri manipulátory udalostí sú vytvorené najjednoduchším spôsobom, kedy prehrajú používateľovi príslušnú správu, ktorá by mu mala pomôcť poskytnúť vhodný vstup. Po

prehratí definovanej správy sa do elementu `<field>` opäť vstúpi bez prehratia výzvy definovanej vnútri (na opätovné prehratie výzvy vstupného poľa môže byť do manipulátora udalosti zahrnutý element `<reprompt>`) a bude sa ešte raz čakať na vstup používateľa.

Poskytnutý návod môže byť považovaný za minimum, aby bol človek schopný napísať jednoduché dialógy, ktoré sú produkované interpretáciou statických VoiceXML dokumentov. Viac informácií o VoiceXML je možné nájsť priamo vo W3C odporúčaní [92] alebo v mnohých návodoch dostupných na Internete.

Doteraz sme sa nezmienili o naviazaní VoiceXML aplikácií na zdroje dát, akými sú databázy alebo Internet. Táto téma sa vzťahuje tiež na VoiceXML aplikácie s „dynamickým" obsahom, čo znamená, že nie všetky VoiceXML dokumenty, z ktorých aplikácia pozostáva, sú napísané pred spustením aplikácie, ale jedna časť aplikácie môže byť generovaná dynamicky určitým webovým serverom alebo dokumentovým serverom (v angl. *document server*), ako je to ukázané na Obr. 11.9. Hlavnou myšlienkou je to, že po získaní požadovaných dát od používateľa, ktoré sú potrebné na vytvorenie dopytu pre zdroj dát (databáza, Internet), VoiceXML interpretér pošle požiadavku na dokumentový server so skupinou párov atribút - hodnota, ktoré uchovávajú získané dáta. Následne dokumentový server vykoná vyhľadanie dopytu a získa informácie, ktoré by mali byť doručené používateľovi a vygeneruje nový VoiceXML dokument, ktorý bude prezentovať dané informácie.

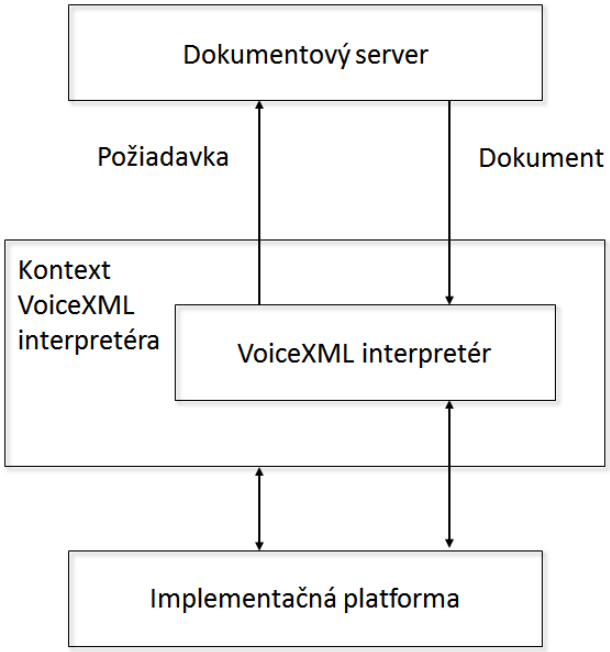
Vygenerovanie takéhoto VoiceXML dokumentu môže byť vykonané prostredníctvom webovej stránky s dynamickým obsahom napísanej v PHP alebo JSP. Vygenerovaný dokument je doručený späť VoiceXML interpretéru a dialóg pokračuje interpretáciou dynamicky generovaného dokumentu.

Takýto scenár je realizovaný prostredníctvom elementu `<submit>`, ktorý umožňuje odovzdať informáciu webovému serveru, kde sa vytvorí VoiceXML dokument s dynamicky generovaným obsahom. Element tiež vykonáva prechod do vráteného dokumentu, kde dialóg pokračuje jeho interpretáciou.

Element `<submit>` ponúka dôležitú možnosť poslania zoznamu premenných dokumentovému serveru cez HTTP `get` alebo `post` požiadavku. Hodnoty týchto premenných sú potom využité pri databázovom dopyte ako jeho parametre.

Napríklad si predstavme službu predpovede počasia, v ktorej sa nachádzajú dve dôležité hodnoty, ktoré je nutné získať od používateľa - mesto (city) a deň (day). Ak predpokladáme rovnaké názvy premenných, tieto hodnoty môžu byť odovzdané webovému serveru, ako je to ukázané v časti zdrojového kódu na Obr. 11.10.

Po dopytovaní služby predpovede počasia, webovým serverom je vygenerovaný nový VoiceXML dokument s využitím niektorého skriptovacieho



Obr. 11.9: Model dynamickej VoiceXML architektúry

```
<submit next="http://weatherforecast.sk/obtain_weather.php"
        namelist="city day" method="post"/>
```

Obr. 11.10: Odosielanie premenných „city" a „day" webovému serveru

jazyka (PHP, JSP, ASP). Vygenerovaný dokument je potom vrátený späť platforme VoiceXML. Tento dokument môže vyzerať ako dokument ukázaný na Obr. 11.11.

```
<?xml version="1.0" encoding="UTF-8"?>
<vxml version="2.0">

  <form id="weather_forecast">
    <block>
      <prompt>Weather forecast for city Poprad for Friday is:
        sunny, 19 degrees Celsius.  </prompt>
      <goto next="what_next.vxml"/>
    </block>
  </form>
</vxml>
```

Obr. 11.11: Dynamicky generovaný VoiceXML dokument s požadovanými informáciami

12

Webové aplikácie s rečovým rozhraním

Webové aplikácie predstavujú nový trend v oblasti interaktívnych služieb. Hlavná myšlienka webových aplikácií spočíva v poskytovaní služieb, ktoré sú prístupné prostredníctvom rôznych zariadení bez špecifických požiadaviek. Tento fakt predstavuje nezávislosť hardvéru a softvéru. Je to prirodzená požiadavka, pretože ľudia používajú niekoľko osobných zariadení - osobné počítače, notebooky, netbooky, tablety, smartfóny a pod. Tie vyžadujú prístup k ich obľúbeným každodenným službám zo všetkých zariadení. Ak sú tieto služby (ako e-mail, Facebook atď.) poskytované prostredníctvom jednotného rozhrania, môžu byť považované za užívateľsky priateľské a ich používanie je preto pohodlnejšie. Aby bolo možné poskytovať takéto služby, je nutné nájsť vhodnú platformu, ktorá so sebou prinesie nezávislosť hardvéru a softvéru. Webové prehliadače boli vytvorené ako takáto platforma. Kombinácia jazyka HTML5, jazyka JavaScript a nového jazyka CSS3 umožňuje napísať moderné webové aplikácie, ktoré sú vhodné aj na streamovanie multimédií a na prístup k iným technológiám ako ASR (automatické rozpoznávanie reči) alebo TTS (syntéza reči z textu). V prípade rozpoznávania reči, spojenie s uvedeným konceptom s technológiou rozpoznávania reči spoločnosti Google predstavuje možnosť vytvoriť webové aplikácie s rečovým rozhraním.

V nasledujúcom texte sa zameriame na technológie pre tvorbu webových aplikácií s rečovým rozhraním.

Obr. 12.1: Štandardy HTML5 + CSS3 na tvorbu rôznych HMI.

12.1 Technológie na tvorbu webových aplikácií

12.1.1 HTML5

Jazyk HTML5 je novou verziou HTML štandardu (publikovaná ako štandard W3C v októbri 2014), ktorý bol vyvinutý na podporu najnovších multimédií. HTML5 rozširuje a zlepšuje značky, ktoré možno použiť na písanie webu, a to hlavne na jednoduché pridávanie a spracovanie multimediálneho obsahu. Prináša aj aplikačné rozhrania (application interface, API), ktoré umožňujú tvorbu komplexných webových aplikácií. HTML5 bol tiež navrhnutý s ohľadom na jeho použitie na zariadeniach s nízkym výkonom (smartfóny, tablety). Vďaka svojim vlastnostiam je kandidátom na mobilné aplikácie na viacerých platformách. W3C tiež vydalo nové logo pre tento štandard, ako je znázornené na Obr. 12.2.

Nové aplikačné rozhrania robia z webového prehliadača novú platformu, ktorá môže poskytovať rôzne služby. Napríklad webové prehliadače, vďaka

Obr. 12.2: HTML5 logo

HTML5 môžu ukladať dáta a môžu byť potom spustené aj v režime offline. Samotný formát HTML5 nemožno použiť pre animácie ani pre interaktivitu. Na tento účel je nutné využiť štandardy JavaScript a CSS3.

12.1.2 Jazyk JavaScript

Programovací jazyk JavaScript sa vyžaduje v prípade, ak je potrebné vytvoriť dynamickú webovú stránku, čo znamená, že bude reagovať na udalosti vyvolané používateľom. Existuje aj iná možnosť, ako napísať dynamické stránky (Adobe Flash), ale vyžaduje to inštaláciu zásuvných modulov (pluginov). Avšak takéto moduly nemôžu byť k dispozícii pre všetky operačné systémy alebo prehliadače. Jazyk JavaScript a rozhrania API sa stali dôležitou technológiou v novej štandarde HTML5. Zatiaľ čo HTML5 definuje obsah webovej stránky, JavaScript určuje správanie webových stránok a CSS, ako tretia dôležitá technológia, definuje spôsob prezentácie. Každá moderná webová stránka obsahuje aj časť kódu v jazyku JavaScript a všetky moderné webové prehliadače ho podporujú (vrátane prehliadačov na herných konzolách, tabletoch, smartfónoch, inteligentných televízoroch atď.). JavaScript môže byť spojený s dokumentmi HTML pomocou jedného z jeho rozhraní, nazývaného DOM (Document Object Model). Tieto modely reprezentujú dokument ako hierarchiu uzlov, ktoré sú usporiadané do stromu, kde koreňový uzol má typ dokumentu. Uzly v takomto modeli môžu byť pridané, zmenené, prehliadané alebo odstránené. DOM predstavuje platformovo a jazykovo nezávislé rozhranie, ktoré umožňuje dynamický prístup k dokumentu HTML.

Jazyk JavaScript spolu s modelom DOM umožňuje pridanie dynamického správania na webovú stránku, čo znamená, že bude schopná reagovať na udalosti vyvolané používateľom. Tieto reakcie majú podobu obsahu a vzhľadu takejto stránky. JavaScript umožňuje pridanie iných aplikačných rozhraní

na webové stránky napísané v HTML5. Pomocou tohto prístupu sa webové stránky menia na webové aplikácie, ktoré ponúkajú väčšiu interaktivitu.

12.1.3 Technológie poskytujúce vstupnú rečovú modalitu vo webových aplikáciách

V tejto časti budú opísané rozhrania HTML5, ktoré sú dôležité pre pridanie funkcií ovládania hlasu do webových aplikácií. Najjednoduchším spôsobom, ako pridať funkciu rozpoznávania reči do webovej aplikácie, je využitie rozpoznávania reči spoločnosti Google dostupného cez Internet. Je možné to vykonať pomocou prehliadača Chrome spoločnosti Google a jedného z rozhraní Speech Input API alebo Web Speech API. V tejto časti budú tiež uvedené rozhrania HTML5, ktoré umožňujú získať prístup k mikrofónu v reálnom čase. Dátový tok získaný prostredníctvom webovej stránky môže byť odoslaný na vzdialený server s rozpoznávačom prostredníctvom Internetu. Výsledný text je možné odoslať naspäť do webovej aplikácie.

Prehliadač Google Chrome a ovládanie modulu reči

Začiatkom roka 2011 bolo do prehliadača Chrome integrované nové rozhranie Speech Input API od spoločnosti Google. Toto rozhranie bolo nahradené v polovici roka 2014 novou verziou s názvom Web Speech API. Uvedené rozhrania umožňujú zaslanie rečového vstupu používateľa do systému Google na rozpoznávanie hlasu na diaľku. Prvotne zmienené rozhranie Speech Input API priamo rozšírilo HTML značku `<input>` o funkciu podpory rečového vstupu. Druhé rozhranie, Web Speech API je rozhranie založené na jazyku JavaScript, ktoré pridáva podporu rečového vstupu užívateľa pomocou JavaScript kódu.

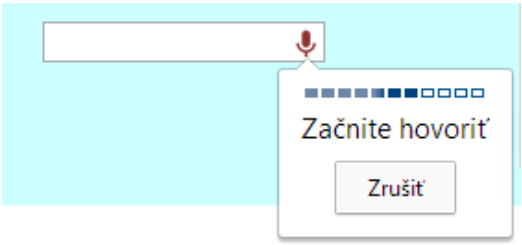
Rozhranie Speech Input API

Spoločnosť Google predstavila svoju verziu rozhrania API na začiatku roka 2011 a implementovala toto rozhranie aj do vlastného prehliadača Chrome. Rozhranie Speech Input API rozširuje element HTML <input> o atribút "speech", s prefixom predajcu, ako je znázornené nasledujúcim kódom:

```
<input type = "text"x-webkit-speech />
```

Použitím atribútu `x-webkit-speech` sa objavila ikona mikrofónu na pravej strane vstupného poľa (ako je zobrazené na Obr. 12.3. Po kliknutí na ikonu sa objavilo okno s textom "Začať hovoriť". Po vyslovení vyjadrenia

(prejavu) sa údaje získané z mikrofónu konvertujú z reči na text pomocou webovej služby a rozpoznaný text sa doručí späť na webovú stránku. Tento text je možné získať vďaka modelu DOM ako hodnota vstupného elementu HTML. Potom môže byť priradená iným premenným a ďalej spracovaný.



Obr. 12.3: Vstupné pole HTML umožňujúce naplnenie rečovým vstupom užívateľa pomocou rozhrania x-webkit-speech.

Rozhranie Speech Input API, vďaka vzdialenému nástroju rozpoznávania reči spoločnosti Google, podporuje mnoho jazykov (vrátane slovenského jazyka). Jazyk prehliadača je tiež predvolený jazyk pre rozpoznávanie reči. Môže to byť zmenené pomocou atribútu `lang`, ako je uvedené nižšie:

```
<input type="text"lang="en" x-webkit-speech />
```

Hoci rozhranie Speech Input API ponúkalo veľmi jednoduché a pohodlné rozhranie, malo aj niekoľko obmedzení:

- Proces rozpoznávania sa zastaví po krátkej pauze, čo nie je vhodné na zadávanie dlhších textov (napr. diktovanie);
- Každé nové rozpoznávanie prepíše predtým rozpoznaný text do rovnakého vstupného poľa;
- Rozpoznávanie sa môže vykonať iba v HTML elemente `<input>`.

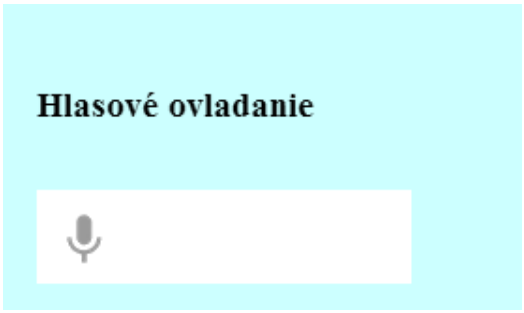
Rozhranie Speech Input API bolo veľmi vhodné pre hlasové vyhľadávanie, riadenie navigácie na webových stránkach alebo pre jednoduché webové hry. Nanešťastie rozhranie bolo úplne nahradené rozhraním Web Speech API v polovici roka 2014.

Rozhranie Web Speech API

Na základe predchádzajúcich špecifikácií a záverečnej správy skupiny W3C HTML Speech Incubator Group, spoločnosť Google, na konci roka 2011, zverejnila rozšírenú verziu Speech JavaScript API špecifikácie, ktorá bola neskôr premenovaná na rozhranie Web Speech API. Špecifikácia rozhrania Web Speech API bola dokončená v októbri 2012 a bola prvýkrát implementovaná vo verzii 25 prehliadača Google Chrome. Toto rozhranie je založené na jazyku JavaScript. Na povolenie používania technológie rozpoznávania reči, musí byť vytvorený nový JavaScript objekt `webkitSpeechRecognition()` nasledovne:

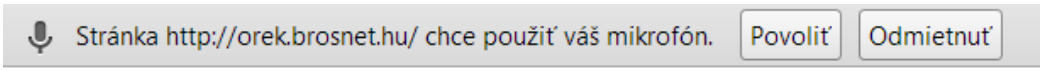
```
var reco = new webkitSpeechRecognition();
```

Rozhranie Web Speech API definuje niekoľko funkcií na odosielanie udalostí súvisiacich s procesom rozpoznávania reči. Tieto udalosti sú zahodené, keď sa proces rozpoznávania spustí alebo sa zastaví, keď sa zdeteguje reč, alebo sú k dispozícii výsledky alebo sa vyskytne nejaká chyba.



Obr. 12.4: Rečové rozhranie vytvorené pomocou rozhranie Web Speech API.

V porovnaní s rozhraním Speech Input API, vytváranie webovej aplikácie s rečovým rozhraním vyžaduje komplexnejší zdrojový kód. Rozhranie Web Speech API rieši aj bezpečnostné problémy, ku ktorým dochádza v prípade rozhrania Speech Input API. V prípade použitia štandardného protokolu HTTP sa zobrazí potvrdzovacie okno (Obr. 12.5), ktoré požiada používateľa o povolenie používania mikrofónu. V prípade protokolu HTTPS je prechod považovaný za bezpečný a potvrdenie použitia mikrofónu sa nevyžaduje.



Obr. 12.5: Vyskakovacie (pop-up) okno na umožnenie použitia mikrofónu.

Možnosť záznamu plynulej reči od používateľa (obmedzená na 60 sekúnd) je hlavným rozšírením v porovnaní s rozhraním Speech Input API. Ďalším dôležitým rozšírením je možnosť zobraziť aj čiastkové výsledky rozpoznávania (hypotézy), ale s nižšou pravdepodobnosťou. Rozhranie Web Speech API sa teda dá použiť na diktovanie textu alebo písanie e-mailov. Web Speech API je podporované niekoľkými webovými prehliadačmi, ktoré sú určené pre mobilné zariadenia. Prehliadač Google Chrome pre systém Android (4.0) podporuje rozhranie Web Speech API od verzie 32.0. V systéme iOS je toto rozhranie podporované prehliadačom Safari od siedmej verzie.

Literatúra

[1] Acero, A.: Formant analysis and synthesis using hidden markov models. In: Sixth European Conference on Speech Communication and Technology (1999)

[2] Allauzen, C., Mohri, M.: Generalized optimization algorithm for speech recognition transducers. In: Proc. of the IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing, ICASSP'03, vol. 1, pp. 352–355 (2003)

[3] Anderson, J.A.: Automata Theory with Modern Applications. Cambridge University Press, New York, NY, USA (2006)

[4] Antoniol, G., Brugnara, F., Cettolo, M., Federico, M.: Language model estimations and representations for real-time continuous speech recognition. In: Proc. of the 3rd Intl. Conf. on Spoken Language Processing, ICSLP'94. Yokohama, Japan (1994)

[5] Aubert, X., Dugast, C., Ney, H., Steinbiss, V.: Large vocabulary continuous speech recognition of wall street journal data. In: Proc. of the IEEE Intl. Conf. on Acoustics, Speech, and Signal Processing, ICASSP'04, vol. 2, pp. 129–132. Adelaide (1994)

[6] Baekgaard, A.: Dialogue management in a generic dialogue system. In: Dialogue Management in a Generic Dialogue System, pp. 123–132 (1996)

[7] Bernsen, N.O., Dybkjær, H., Dybkjær, L.: Designing interactive speech systems: From first ideas to user testing. Springer Science & Business Media (2012)

[8] Black, A.W., Dusterhoff, K.E., Taylor, P.A.: Using the tilt intonation model: A data-driven approach. In: Data-Driven Techniques in Speech Synthesis, pp. 199–214. Springer (2001)

[9] Black, W., Thompson, P., Funk, A., Conroy, A.: Learning to classify utterances in a task-oriented dialogue. In: Proc. Workshop on Dialogue

Systems: Interaction, Adaptation and Styles of Management. 10th Conf. of the EACL (2003)

[10] Bohus, D., Rudnicky, A.I.: Ravenclaw: Dialog management using hierarchical task decomposition and an expectation agenda. In: Proceedings of Eurospeech-2003, Geneva, Switzerland (2003)

[11] Caseiro, D., Trancoso, I.: A decoder for finite-state structured search spaces. In: ISCA ITRW Intl. Workshop on Automatic Speech Recognition: Challenges for the Next Millennium, pp. 18–20. Paris, France (2000)

[12] Chen, S., Goodman, J.: An empirical study of smoothing techniques for language modeling. Tech. Rep. TR-10-98, Computer Science Group, Cambridge (1998). URL `https://people.eecs.berkeley.edu/~klein/cs294-5/chen_goodman.pdf`

[13] Chu-Carroll, J.: Form-based reasoning for mixed-initiative dialogue management in information-query systems. In: Sixth European Conference on Speech Communication and Technology (1999)

[14] Clemins, P.J., Trawicki, M.B., Adi, K., Tao, J., Johnson, M.T.: Generalized perceptual features for vocalization analysis across multiple species. In: Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on, vol. 1, pp. I–I. IEEE (2006)

[15] Cohen, P.R.: Models of dialogue. In: Proc. of the 4th NEC Research Symposium: Cognitive Processing for Vision and Voice, p. 245–274 (1994)

[16] Cohen, P.R., H., L.: Rational interaction as the basis for communication. In: P. R. Cohen, J. Morgan & M. E. Pollack (Eds.): Intentions in Communication, p. 221–256 (1990)

[17] Dahlbäck, N., Jönsson, A.: Knowledge sources in spoken dialogue systems. In: EUROSPEECH'99, pp. 1523–1526 (1999)

[18] Ečegi, M.: Metódy parametrizácie reči. FEI TUKE (2013). PhD thesis

[19] Fousek, P.: Extraction of features for automatic recognition of speech based on spectral dynamics. České vysoké učení technické v Praze (2007). PhD thesis

[20] Fujisaki, H.: The role of quantitative modeling in the study of intonation. In: Proceedings of the International Symposium on Japanese Prosody, pp. 163–174 (1992)

[21] Gauvain, J.L., Lamel, L.: Fast decoding for indexation of broadcast data. In: Proc. of the 6th Intl. Conf. on Spoken Language Processing, ICSLP'2000). Beijing, China (2000)

[22] Handley, M., Schulzrinne, H., Schooler, E., Rosenberg, J.: SIP: Session Initiation Protocol. The Internet Society: Network Working Group (1999). URL `http://tools.ietf.org/html/rfc2543`

[23] Hart, J., Collier, R., Cohen, A.: A perceptual study of intonation: an experimental-phonetic approach to speech melody. Cambridge University Press (2006)

[24] Hermansky, H.: Perceptual linear predictive (PLP) analysis of speech. the Journal of the Acoustical Society of America **87**(4), 1738–1752 (1990)

[25] Hermansky, H., Morgan, N.: Rasta processing of speech. IEEE transactions on speech and audio processing **2**(4), 578–589 (1994)

[26] Hermansky, H., Sharma, S.: Traps-classifiers of temporal patterns. In: ICSLP, pp. 1003–1006 (1998)

[27] Hermansky, H., Sharma, S.: Temporal patterns (traps) in asr of noisy speech. In: Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on, vol. 1, pp. 289–292. IEEE (1999)

[28] Hopcroft, J.E., Motwani, R., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation (3rd Edition). Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (2006)

[29] Hsu, B.: Language modeling for limited-data domains. Ph.D. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology (2009)

[30] Huang, X., Acero, A., Hon, H.W.: Spoken Language Processing: A Guide to Theory, Algorithm, and System Developement. Prentice Hall, New Jersey (2001)

[31] Ivanecký, J., Nábělková, M.: Fonetická transkripcia SAMPA a slovenčina. Jazykovedný časopis **53**(2), 81–95 (2002)

[32] Jelinek, F.: Statistical Methods for Speech Recognition. MIT Press, Cambridge, MA, USA (1998)

[33] Jokinen, K., McTear, M.: Spoken dialogue systems. Synthesis Lectures on Human Language Technologies **2**(1), 1–151 (2009)

[34] Juhár, J., kolektív: Rečové technológie v telekomunikačných a informačných systémoch. Equilibria, Košice (2011)

[35] Juhár, J., Ondáš, S., Čižmár, A., Rusko, M., Rozinaj, G., Jarina, R.: Development of Slovak GALAXY/VoiceXML based spoken language dialogue system to retrieve information from the Internet. In: Proc. of INTERSPEECH 2006, pp. 485–488. Pittsburg, Pennsylvania, USA (2006)

[36] Juhár, J., Staš, J., Hládek, D.: Recent progress in development of language model for Slovak large vocabulary continuous speech recognition. In: C. Volonescu (ed.) New Technologies: Trends, Innovations and Research, pp. 261–276. InTech Open, Rijeka, Croatia (2012)

[37] Jurafsky, D., Martin, J.H.: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition (2nd Edition). Prentice Hall, Pearson Education, New Jersey (2009)

[38] Kain, A., Macon, M.W.: Design and evaluation of a voice conversion algorithm based on spectral envelope mapping and residual prediction. In: Acoustics, Speech, and Signal Processing, 2001. Proceedings.(ICASSP'01). 2001 IEEE International Conference on, vol. 2, pp. 813–816. IEEE (2001)

[39] Kesarkar, M.P.: Feature extraction for speech recognition. In: M. Tech. Credit Seminar Report, Electronic Systems Group, EE. Dept, IIT Bombay, pp. 1–12 (2003)

[40] Kim, C., Stern, R.M.: Power-normalized cepstral coefficients (pncc) for robust speech recognition. In: Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on, pp. 4101–4104. IEEE (2012)

[41] Kim, C., Stern, R.M.: Power-normalized cepstral coefficients (pncc) for robust speech recognition. IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP) **24**(7), 1315–1329 (2016)

[42] Kráľ, A., Sabol, S.: Fonetika a fonológia. Slovenské pedagogické nakladateľstvo, Bratislava (1989). (in Slovak)

[43] Larsson, S., Traum, D.R.: Information state and dialogue management in the trindi dialogue move engine toolkit. Natural language engineering **6**(3-4), 323–340 (2000)

[44] Llorens, D., Casacuberta, F.: An experimental study of histogram prunning in speech. In: Proc. of the 8th Simposium Nacional de Reconocimiento de Formas y Anàlisis de Imàgenes, pp. 5–6. AERFAI, Bilbao (1999)

[45] Lopez, R., Delgado, C., Araki, M.: Spoken, Multilingual and Multimodal Dialogue Systems, Development and Assessment. Wiley (2005)

[46] McTear, M.F.: Spoken dialogue technology: enabling the conversational user interface. ACM Computing Surveys (CSUR) **34**(1), 90–169 (2002)

[47] Mohri, M.: Minimization algorithms for sequential transducers. Theor. Comput. Sci. **234**, 177–201 (2000)

[48] Mohri, M.: Statistical natural language processing. In: M. Lothaire (Ed.): Applied Combinatorics on Words, pp. 199–255. Cambridge University Press (2005)

[49] Mohri, M., Riley, M.: Integrated context-dependent networks in very large vocabulary speech recognition. In: Proc. of the 6th European Conf. on Speech Communication and Technology, EUROSPEECH'99. Budapest, Hungary (1999)

[50] Mohri, M., Riley, M.: Network optimizations for large vocabulary speech recognition. Speech Communication **25** (1999)

[51] Mohri, M., Riley, M., Hindle, D., Ljolje, A., Pereira, F.: Full expansion of context-dependent networks in large vocabulary speech recognition. Computational Linguistics **23**, 269–311 (1997)

[52] Ondráček, V.: Robustní parametrizace řeči na bázi časovỳch trajektorií. České vysoké učení technické v Praze (2012)

[53] Ortmanns, S., Ney, H.: Look-ahead techniques for fast beam search. Computer Speech & Language **14**(1), 15–32 (2000)

[54] Ortmanns, S., Ney, H., Eiden, A.: Language-model look-ahead for large vocabulary speech recognition. In: Proc. of the 4th Intl. Conf. on Spoken Language Processing, ICSLP'96, pp. 2095–2098. Philadelphia, PA, USA (1996)

[55] Papaj, J., Pleva, M., Čižmár, A., Ľ. Doboš, Juhár, J., Ondáš, S.: MOBILTEL – mobile multimodal telecommunication systems and services. In: Proc. of the 17th Intl. Conf. on RADIOELECTRONICS'07, pp. 517–520. Brno, Czech Republic (2007)

[56] Pavlík, R.: Slovenské hlásky a medzinárodná fonetická abeceda. Jazykovedný časopis **55**(2), 87–109 (2004)

[57] Pereira, F., Riley, M.: Speech recognition by composition of weighted finite automata. In: Finite-State Language Processing, pp. 431–453. MIT Press (1997)

[58] Pereira, F.C.N., Mohri, M., Riley, M.: Weighted automata in text and speech processing. In: Proc. of the 12th Biennial European Conf. on Artificial Intelligence, ECAI'96. Budapest, Hungary (1996)

[59] Picheny, M., Nahamoo, D.: Towards superhuman speech recognition. J. Benesty et al. (Eds.): Springer Handbook of Speech Processing pp. 597–616 (2008)

[60] Pleva, M., Papaj, J., Čiămár, A., Ľ. Doboš, Juhár, J., Ondáš, S., Mirilovič, M.: Towards Mobile Multimodal Telecommunications Systems and Services, *Lecture Notes in Computer Science*, vol. 4775 LNAI. Springer-Verlag (2007)

[61] Pollák, P., Černocký, J., Boudy, J., Choukri, K., van den Huevel, H., k. Vicsi, Virag, A., Siemund, R., Majewski, W., Sadowski, J., Staroniewicz, P., Tropf, H., j. Kochanina, Ostroukhov, A., Rusko, M., Trnka, M.: SpeechDat(E) – eastern european telephone speech databases. In: Proc. LREC'2000 Satellite Workshop XLDB – Very Large Telephone Speech Databases, pp. 20–25. Athens, Greece (2011)

[62] Psutka, J., Müller, L., Matoušek, J., Radová, V.: Mluvíme s počítačem česky. Academia, Praha, Česká republika (2006)

[63] Rabiner, L.R., Schafer, R.W.: Digital Processing of Speech Signals. Prentice Hall, Englewood Cliffs, NJ, USA (1978)

[64] Renals, S.: Phone deactivation pruning in large vocabulary continuous speech recognition. IEEE Signal Processing Letters **3**(1), 4–6 (1996)

[65] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., Schooler, E.: SIP: Session Initiation Protocol. The Internet Society: Network Working Group (2002). URL `http://tools.ietf.org/html/rfc3261`

[66] Rosenfeld, R.: A maximum entropy approach to adaptive statistical language modeling. Computer Speech & Language **10**(3), 187–228 (1996)

[67] Rubin, P., Baer, T., Mermelstein, P.: An articulatory synthesizer for perceptual research. The Journal of the Acoustical Society of America **70**(2), 321–328 (1981)

[68] Russell, S.J., Norvig, P.: Artificial Intelligence: A Modern Approach. Pearson Education (2003)

[69] Sadek D. M., d.M.R.: Dialog systems. In: Spoken Dialogues with Computers, p. 523–561 (1998)

[70] Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. In: A. Waibel and K.-F. Lee (Eds.): Readings in Speech Recognition, pp. 159–165. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (1990)

[71] Seneff, S., Hurley, E., Lau, R., Pao, C., Schmid, P., Zue, V.: Galaxy-ii: A reference architecture for conversational system development. In: Fifth International Conference on Spoken Language Processing (1998)

[72] Seneff, S., Lau, R., Polifroni, J.: Organization, communication, and control in the GALAXY-II conversational system. In: Proc. of the 6th European Conf. on Speech Communication and Technology, EUROSPEECH'99, pp. 1271–1274. Budapest, Hungary (1999)

[73] Sharma, C., Kunnins, J.: VoiceXML: Strategies and Techniques for Effective Voice Application Development with VoiceXML 2.0 (Illustrated edition). John Wiley & Sons (2002)

[74] Soong, F.K., Huang, E.F.: A tree-trellis based fast search for finding the n-best sentence hypotheses in continuous speech recognition. In: Proc. of the IEEE Intl. Conf. Acoustics, Speech, and Signal Processing, ICASSP'91, pp. 705–708. IEEE Computer Society, Washington, DC, USA (1991)

[75] Staš, J., Hládek, D., Juhár, J.: Language model size reduction by quantization and pruning. Journal of Electrical and Electronics Engineering **3**(1), 205–208 (2010)

[76] Staš, J., Hládek, D., Juhár, J.: Morphologically motivated language modeling for Slovak continuous speech recognition. Journal of Electrical and Electronics Engineering **5**(1) (2012)

[77] Staš, J., Hládek, D., Juhár, J.: Adding filled pauses and disfluent events into language models for speech recognition. In: Proc. of CogInfoCom 2016, pp. 133–137. Wroclaw, Poland (2016)

[78] Staš, J., Hládek, D., Juhár, J., Ološtiak, M.: Automatic extraction of multiword units from slovak text corpora. In: Proc. of the 7th International Conference on Natural Language Processing, Corpus Linguistics, E-learning, SLOVKO 2013, pp. 228–237. Bratislava, Slovakia (2013)

[79] Staš, J., Hládek, D., Juhár, J., Zlacký, D.: Analysis of morph-based language modeling and speech recognition in Slovak. Advances in Electrical and Electronic Engineering **10**(4) (2012)

[80] Staš, J., Juhár, J., Hládek, D.: Classification of heterogeneous text data for robust domain-specific language modeling. EURASIP Journal on Audio, Speech and Music Processing **2014**(14), 1–12 (2014)

[81] Staš, J., Petráš, M., Juhár, J.: Využitie webových zdrojov v štatistickom modelovaní slovenského jazyka. In: Electrical Engineering and Informatics 4: Proceedings of the Faculty of Electrical Engineering and Informatics of the Technical University of Košice, pp. 524–529. Košice, Slovakia (2013)

[82] Staš, J., Zlacký, D., Hládek, D.: Semantically similar document retrieval framework for language model speaker adaptation. In: Proc. of RADIOELEKTRONIKA 2016, pp. 403–407. Košice, Slovakia (2016)

[83] Steiner, I., Ouni, S.: Artimate: an articulatory animation framework for audiovisual speech synthesis. arXiv preprint arXiv:1203.3574 (2012)

[84] Stolcke, A.: Entropy-based pruning of backoff language models. In: Proc. of DARPA Broadcast News Transcription and Understanding Workshop, pp. 270–274. Lansdowne, Pennsylvania, USA (1998)

[85] Sulír, M., Juhár, J.: Diphone text-to-speech synthesis and its evaluatio. In: Proceedings of International Conference on Applied Electrical Engineering and Informatics, pp. 53–56 (2012)

[86] Taylor, P.: Text-to-speech synthesis. Cambridge university press (2009)

[87] Traum, D., Bos, J., Cooper, R., Larsson, S., Lewin, I., Matheson, C., Poesio, M.: A model of dialogue moves and information state revision. Tech. rep., Tech. rept. Deliverable (1999)

[88] Turunen, M.: Jaspis-a spoken dialogue architecture and its applications. Tampereen yliopisto (2004)

[89] Tyagi, V., Wellekens, C.: Fepstrum representation of speech signal. In: Automatic Speech Recognition and Understanding, 2005 IEEE Workshop on, pp. 11–16. IEEE (2005)

[90] Čižmár, A., Pleva, M., Papaj, J., Ľ. Doboš, Juhár, J.: MobilTel – mobile multimodal telecommunications dialogue system based on VoIP telephony. Journal of Electrical and Electronics Engineering **2**(2), 134–137 (2009)

[91] Viszlay, P., Muchanic, M., Juhar, J.: Fepstrálna analýza v systéme rozpoznávania reči. In: Electrical Engineering and Informatics. FEI TUKE (2012)

[92] VoiceXML 2.0: W3C Recommendation (2004). URL `http://www.w3.org/TR/voicexml20/`

[93] Walker, M.A., Passonneau, R., Boland, J.E.: Quantitative and qualitative evaluation of darpa communicator spoken dialogue systems. In: Proceedings of the 39th Annual Meeting on Association for Computational Linguistics, pp. 515–522. Association for Computational Linguistics (2001)

[94] Wilks, Y., Catizone, R., Worgan, S., Turunen, M.: Some background on dialogue management and conversational speech for dialogue systems. Computer Speech & Language **25**(2), 128 (2010)

[95] Yapanel, U.H., Hansen, J.H.: A new perceptually motivated mvdr-based acoustic front-end (pmvdr) for robust automatic speech recognition. Speech Communication **50**(2), 142–152 (2008)

[96] Young, S.J., Russel, N.H., Thornton, J.H.S.: Token passing: a simple conceptual model for connected speech recognition systems. Tech. Rep. CUED/F-INFENG/TR38, Cambridge University (1989)

[97] Zen, H., Nose, T., Yamagishi, J., Sako, S., Masuko, T., Black, A.W., Tokuda, K.: The hmm-based speech synthesis system (hts) version 2.0. In: SSW, pp. 294–299 (2007)

[98] Zhang, Y., Abdulla, W.H.: Gammatone auditory filterbank and independent component analysis for speaker identification. In: Ninth International Conference on Spoken Language Processing (2006)

ISBN 978-80-553-2661-0