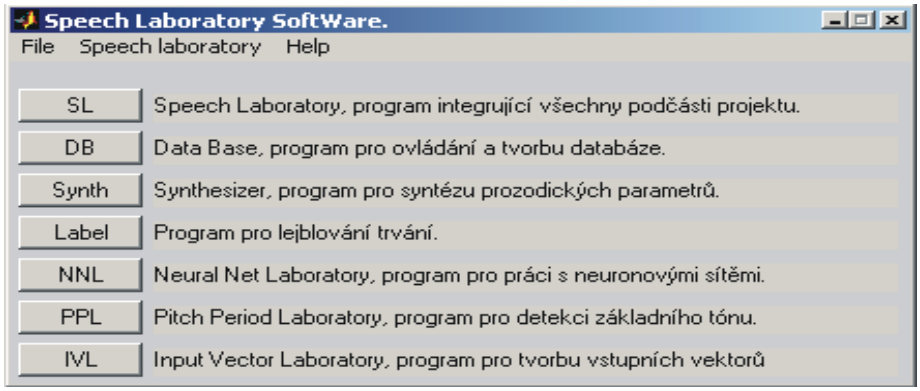
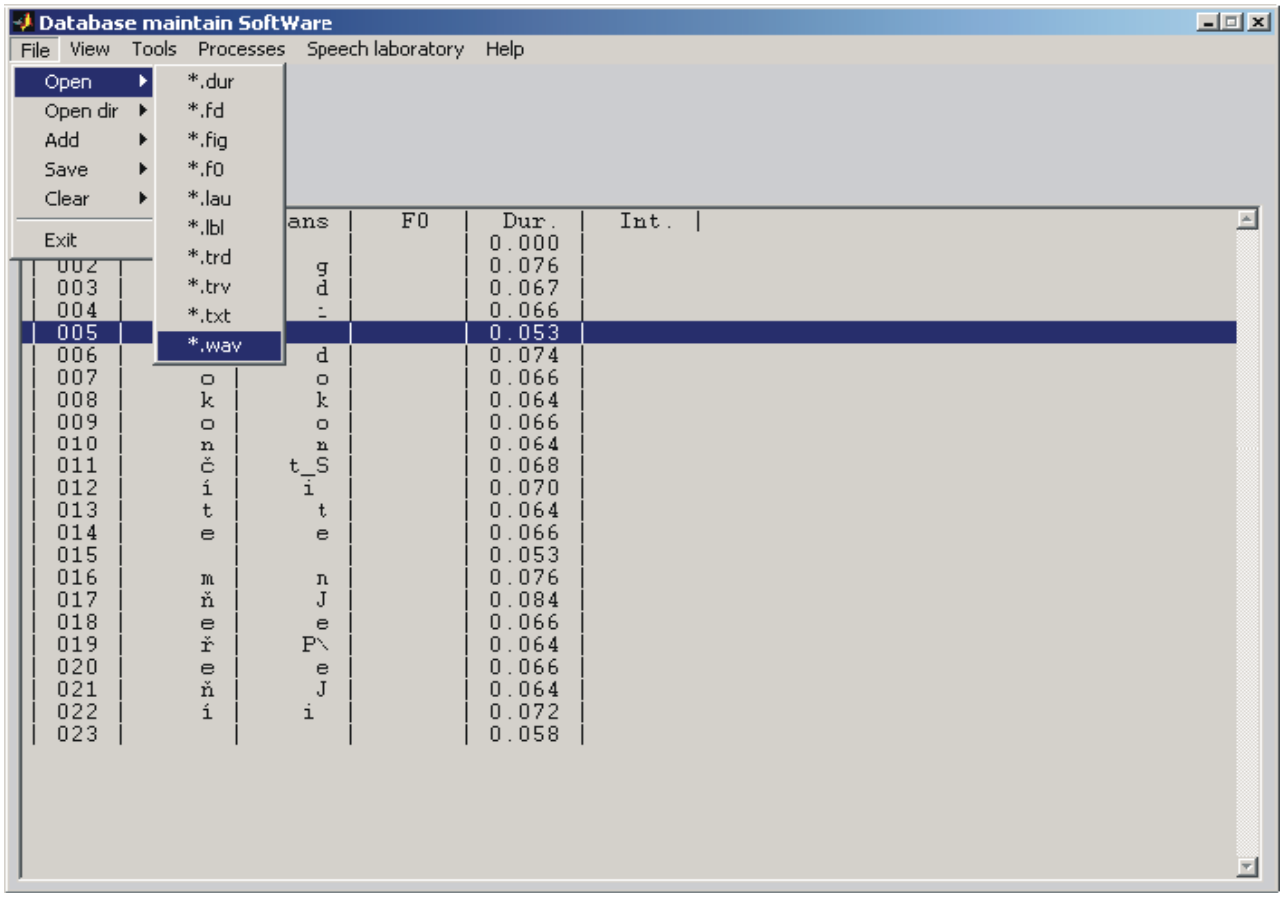
O projektu Speech Laboratory

Součástí disertační práce [1] bylo vytvořit programové prostředky pro TTP (Text-to-Prosody) syntézu. Za tímto účelem byla vytvořena řada programových nástrojů, jejichž úkolem je vhodně zpracovávat data za účelem TTP syntézy. Kolekce těchto nástrojů byla zahrnuta do projektu nazvaného *Speech Laboratory*. Na obrázku je zobrazeno uživatelské rozhraní projektu.



O projektu DB

Během této disertační práce bylo nutné vytvořit a pracovat s velkým množstvím datových standardů. Pro snadnou konverzi mezi těmito standardy a pro jejich případnou úpravu byl vytvořen program *DB*. Příklad uživatelského rozhraní zobrazuje obrázek.

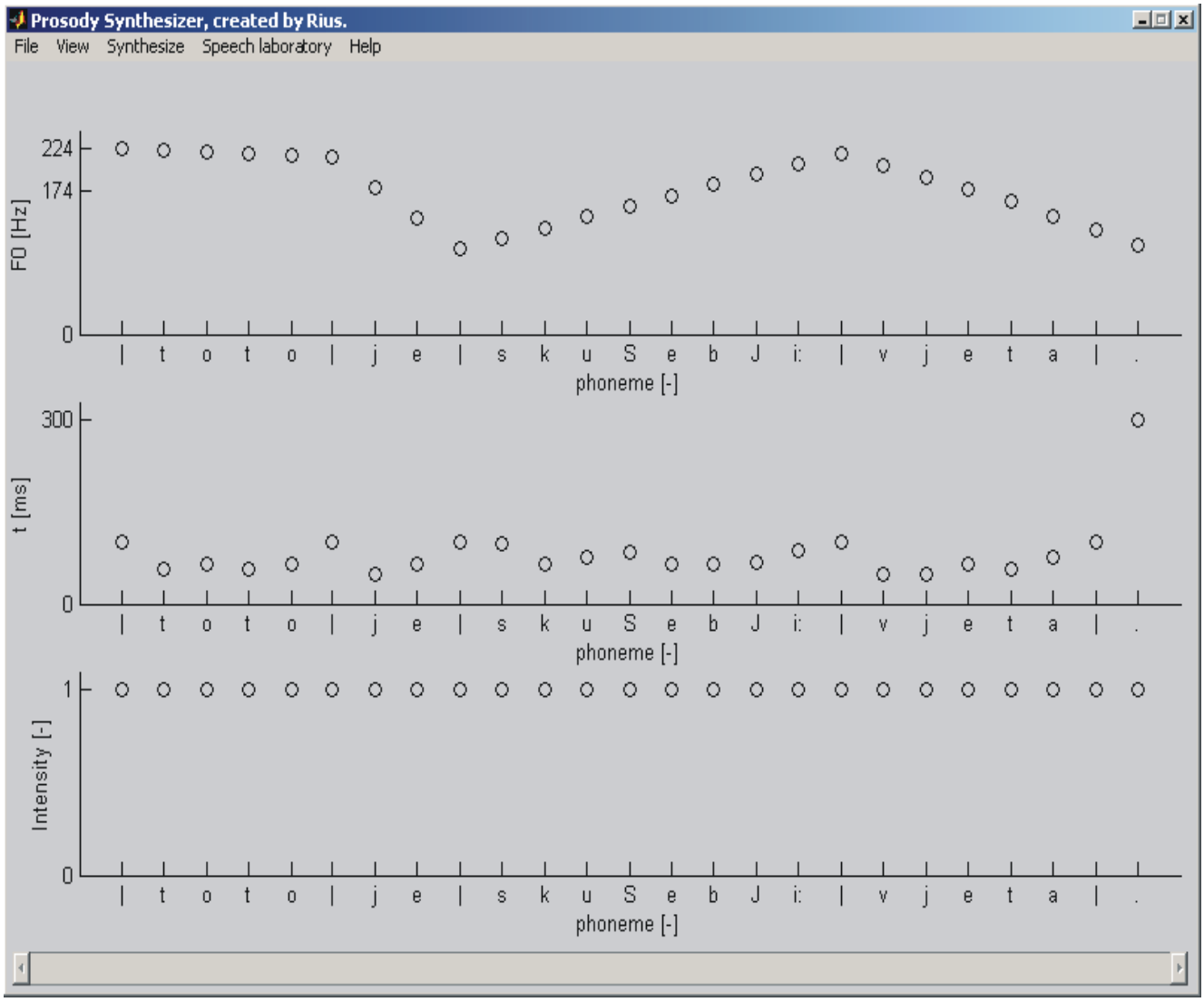


Prvořadým úkolem programu je možnost otevírání různých typů souborů. Data, která jsou v těchto souborech obsažena, jsou nahrávána do paměti počítače. Na tato data se dají aplikovat různé matematické metody, jako je například změna průměrné hodnoty, součet, multiplikace konstantou atd. Zpracovaná data lze pak ukládat do širokého spektra souborových standardů.

Druhou důležitou funkcí programu DB je možnost automatického zpracovávání rozsáhlých databází. Program *DB* disponuje sadou maker, která mají přístup ke všem programům projektu *Speech Laboratory*. Program tak má možnost dálkově spravovat zbylé programy projektu. Této možnosti lze využít například při resyntézách celých databázových jednotek pomocí neuronových sítí.

O projektu Synth

Úkolem programu *Synth* je syntetizovat prozodické parametry. Jedná se tedy o konkrétní TTP syntezátor. Příklad uživatelského rozhraní programu *Synth* zobrazuje obrázek.



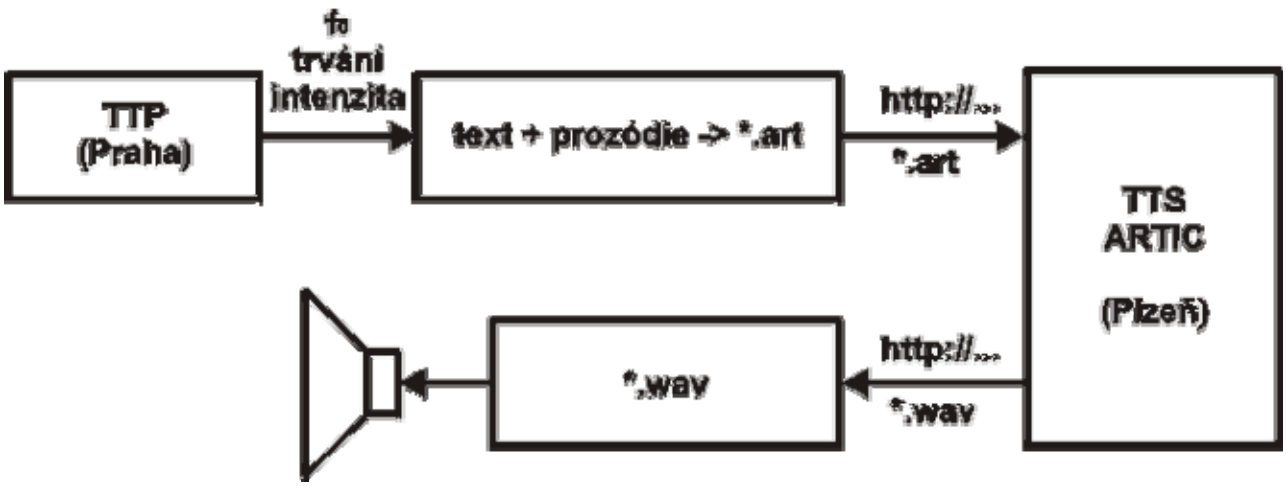
Jednotlivé prozodické parametry lze syntetizovat několikerým způsobem. Program nabízí:

- Načtení dat ze souboru -- kterýkoliv z prozodických parametrů lze načíst z externího souboru. Uživatel má tak možnost využívat externích programových prostředků.
- Syntézu monotónních průběhů -- kterýkoliv z prozodických parametrů lze generovat jako monotónní. V případě trvání jsou generovány tabulkové hodnoty průměrných dob trvání fonémů.
- Syntézu interpolací -- uživatel zadá do interaktivního grafu body, kterými má generovaný prozodický parametr procházet. Těmito body je pak proložena interpolační křivka, která je použitá jako průběh prozodického parametru.
- Syntézu pomocí Fujisakiho modelu -- pro syntézu základního hlasivkového tónu f0.
- Syntézu pomocí neuronové sítě -- dá se použít pro kterýkoliv prozodický parametr. Vstupem je text promluvy, maska vstupních parametrů a neuronová síť.

Výsledky syntézy lze uložit do souborů, graficky zobrazit nebo ozvučit. Ozvučení výsledků je možné dvojím způsobem. První typ převodu je založen na přímé syntéze. Průběh prozodického parametru je syntetizován pomocí sinusovky s amplitudou úměrnou intenzitě, frekvencí f0 a odpovídající dobou trvání.

Výstupem druhého typu syntézy je úplná syntetická promluva s modelovanými prozodickými parametry. Jedná se tedy o plnohodnotnou syntézu řeči. Za tímto účelem byla navázána spolupráce se Západočeskou univerzitou v Plzni, která nám umožnila přístup na TTS systém *Artic* [2].

Přístup k syntezátoru *Artic* nám byl umožněn online, prostřednictvím internetu. Ve spolupráci s Ing. Jindřichem Matouškem, PhD. a Ing. Davidem Tihelkou, PhD. z katedry kybernetiky ZČU v Plzni byl vypracován speciální souborový standard `*.art`. Tento soubor obsahuje informace o syntetizovaném textu a jeho prozodických vlastnostech. Zasláním souboru na speciální portál dojde k provedení syntézy. Syntetická promluva je vrácena ve formě `*.wav` souboru. Princip využívání TTS systému *Artic* zobrazuje následující obrázek.

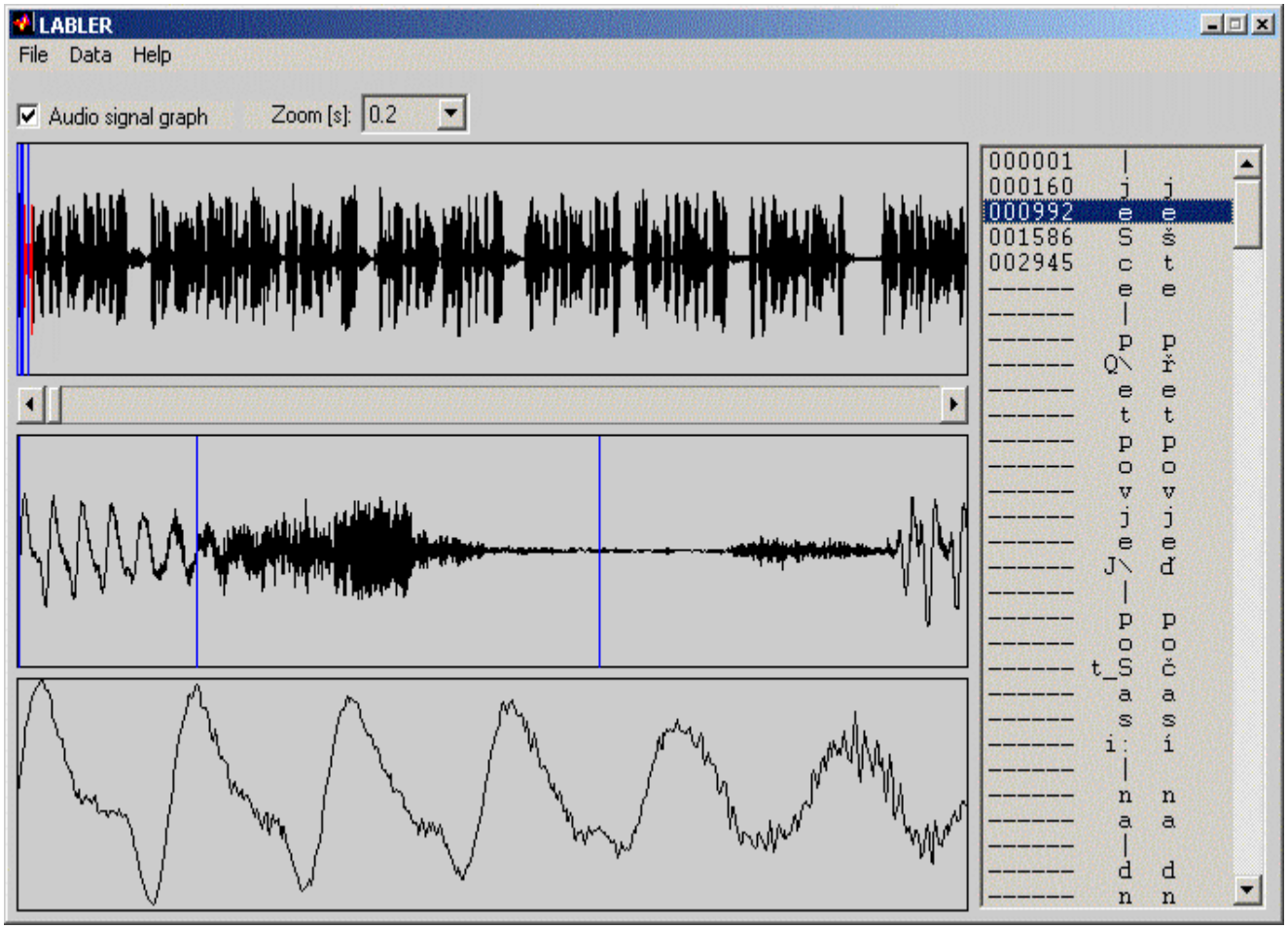


O projektu Label

Program *Label* byl vytvořen za účelem detekce trvání fonémů. Princip detekce trvání fonémů pracuje na základě manuálního vkládání značek začátků a konců fonémů. Příklad uživatelského rozhraní zobrazuje obrázek.

Vstupní data tvoří akustický signál. Ten je požadován v souboru typu `*.wav`. Jedná se o standardní typ souboru, který uchovává akustická data v nekomprimované podobě. Výstupní data jsou ukládána do textového souboru ve formě matice dat. Uložená data udávají doby trvání fonémů v sekundách.

Ovládání programu má intuitivní charakter. Program disponuje několika interaktivními grafy. Hranice začátků a konců fonémů se vkládají pomocí myši, a to kliknutím na zvolenou pozici v grafu. Zvolený foném lze pak pozorovat jak v jeho grafické podobě, tak v jeho akustické reprezentaci.



Uživatelské rozhraní

Uživatelské rozhraní programu Label je patrné z výše uvedeného obrázku. V levé části okna jsou tři grafické náhledy signálu. Jsou to postupně:

- **Graf celého audio-signálu** - zobrazuje náhled celého audio-signálu. Tento náhled lze zakázat pomocí "Checkboxu" v levé hodní části okna, nazvaného "Audio signal graph".
- **Graf výběru audio-signálu** - zobrazuje náhled vybrané části audio-signálu. Délka výběru se volí pomocí "Popup" menu v horní části okna, nazvaného "Zoom [s]".
- **Graf audio-signálu aktuálního fonému** - zobrazuje náhled audio-signálu, aktuálního fonému. To, který foném je aktuální, je dáno "listem" fonémů.

V pravé části okna se nachází **list fonémů**. Vybraný foném je zvolen jako aktuální a průběh jeho audio-signálu je znázorněn v **grafu audio-signálu aktuálního fonému**. Výběrem fonému v tomto listu vybíráme zároveň aktuální foném. K tomuto fonému se pak vztahuje většina dále popisovaných funkcí. Struktura výpisu fonému je následující:

pozice foném text (např. 00022 Q/ ř)

Hodnota ***pozice*** udává pozici vzorku audio-signálu, na kterém začíná daný `foném`. Pro snazší práci s programem je navíc uveden textový reprezentant `fonému`. (Z příkladu tedy vyplývá, že foném `Q/`, který odpovídá hlásce `ř`, začíná na `22` vzorku audio-signálu).

Uživatelské funkce jsou dostupné pomocí "drop-down" menu v horní liště okna. K dispozici jsou následující volby:

- **File** - funkce pro správu souborů.
  - **Open** - nabídka pro otevření souboru. K dispozici jsou *.wav (audio soubor) *.trd (soubor dat transkripce) a *.lbl (soubor programu Label). Zvolený soubor se nahraje z disku počítače do programu.
  - **Save** - nabídka pro uložení dat do souboru. K dispozici jsou *.trd (soubor dat transkripce) a *.lbl (soubor programu Label). Data se z programu uloží do zvoleného souboru. Doporučujeme práci průběžně ukládat. Předejdete tak nebezpečí ztráty dat.
  - **Close** - zvolením této položky smažete všechna data programu. Tato položka slouží k vymazání paměti počítače před novou prací. POZOR!!! Smazaná data nelze obnovit. Je proto důležité je před smazáním uložit pomocí volby **Save**
  - **Exit** - zvolením této položky ukončíte program Label. Data programu se však nesmažou a zůstanou v operační paměti MATLAB-u. Opětovným spuštěním programu Label jsou data automaticky nahrána.
- **Data** - funkce pro správu dat.
  - **Insert phoneme** - vloží nový foném na aktuální pozici.
  - **Delete phoneme** - smaže aktuální foném.
  - **Change phoneme** - změní aktuální foném.
  - **Insert next position** - vloží novou pozici, a to o hodnotu dále, než je pozice aktuálního fonému. Tato volba je výhodná, je-li zapotřebí přidat minimální šířku fonému (např. nulová pauza mezi slovy).
  - **Change position** - změní pozici aktuálního fonému.
  - **Translate** - vytvoří nový překlad fonémů v listu fonémů. Tato volba se používá při desynchronizaci fonémů a jejich textových protějšků při změnách fonémů, ...
  - **Play signal all** - ozvučí celý audio-signál.

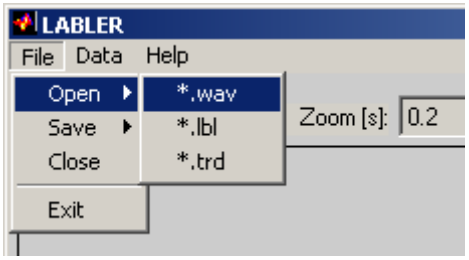
    - **Play signal selection** - ozvučí část audio-signálu, který je zobrazen v okně výběru signálu.
    - **Play signal phonem** - ozvučí aktuální foném.
- **Help** - funkce nápovědy.
    - **Help** - zobrazí tento HTML dokument.
    - **About** - zobrazí informační okno programu.

K dispozici je ještě jedno kontextové menu. Vyvolá se kliknutím pravého tlačítka myši nad **listem fonémů**. Jeho položky mají stejný význam, jako položky hlavního "drop-down" menu **Data**. Položky kontextového menu jsou:

- **Insert phoneme**
- **Delete phoneme**
- **Change phoneme**
- **Insert next position**
- **Change position**
- **Translate**

Ovládání programu Label

Prvním krokem po spuštění programu Label je nahrání audio-signálu (ze souboru *.wav). Docílíme toho zvolením položky **File/Open/*.vaw**.



Pokud je zvolena položka "check boxu" **Audio signal graph**, pak se v oblasti **grafu celého audio-signálu** zobrazí černý graf audio-signálu. Zároveň se v tomto grafu červeně zvýrazní vybraná část signálu. V **grafu výběru audio-signálu** se zobrazí vybraná část signálu. Kliknutím myší do **grafu celého audio-signálu** se vždy vybere zvolená část signálu. Velikost výběru závisí na hodnotě položky **Zoom [s]**. Ta udává velikost výběru v sekundách. Změnou této hodnoty se okamžitě mění i velikost výběru. Zvolením položky **Data/Play signal all** nebo **Data/Play signal selection** můžete ozvučit celý signál nebo jeho výběr.

o signál jsme úspěšně nahráli. Nyní musíme nahrát fonémovou transkripci. Ta je obsažena v souboru typu *.trd (TRanscription Data). Položka pro otevření fonémové transkripce je

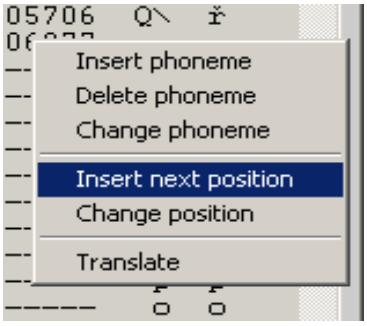
File/Open/*.trd.

Do **listu fonémů** se automaticky nahraje celá fonémová struktura textu. Vedle značky fonému se zobrazí jeho textová reprezentace (hláska). Ta slouží jen pro snazší orientaci v textu. Jako první pozice se automaticky nastaví první vzorek signálu (00001).

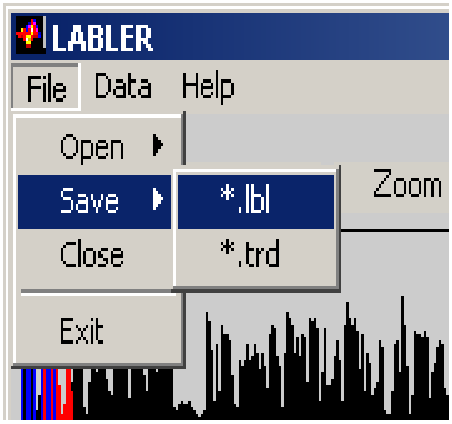
Všechna potřebná data jsou nyní nahrána, můžeme tedy přistoupit k vlastnímu "labelování". Značku (lejbl) vložíme do signálu tak, že klikneme levým tlačítkem myši do **grafu výběru audio-signálu**. Značka se projeví svislým modrým pruhem v obou grafech audio-signálu. V **listu fonémů** se u aktuálního fonému objeví číslo reprezentující počáteční polohu fonému v audio-signálu. Opětovným kliknutím na vytvořenou značku ji odstraníme.

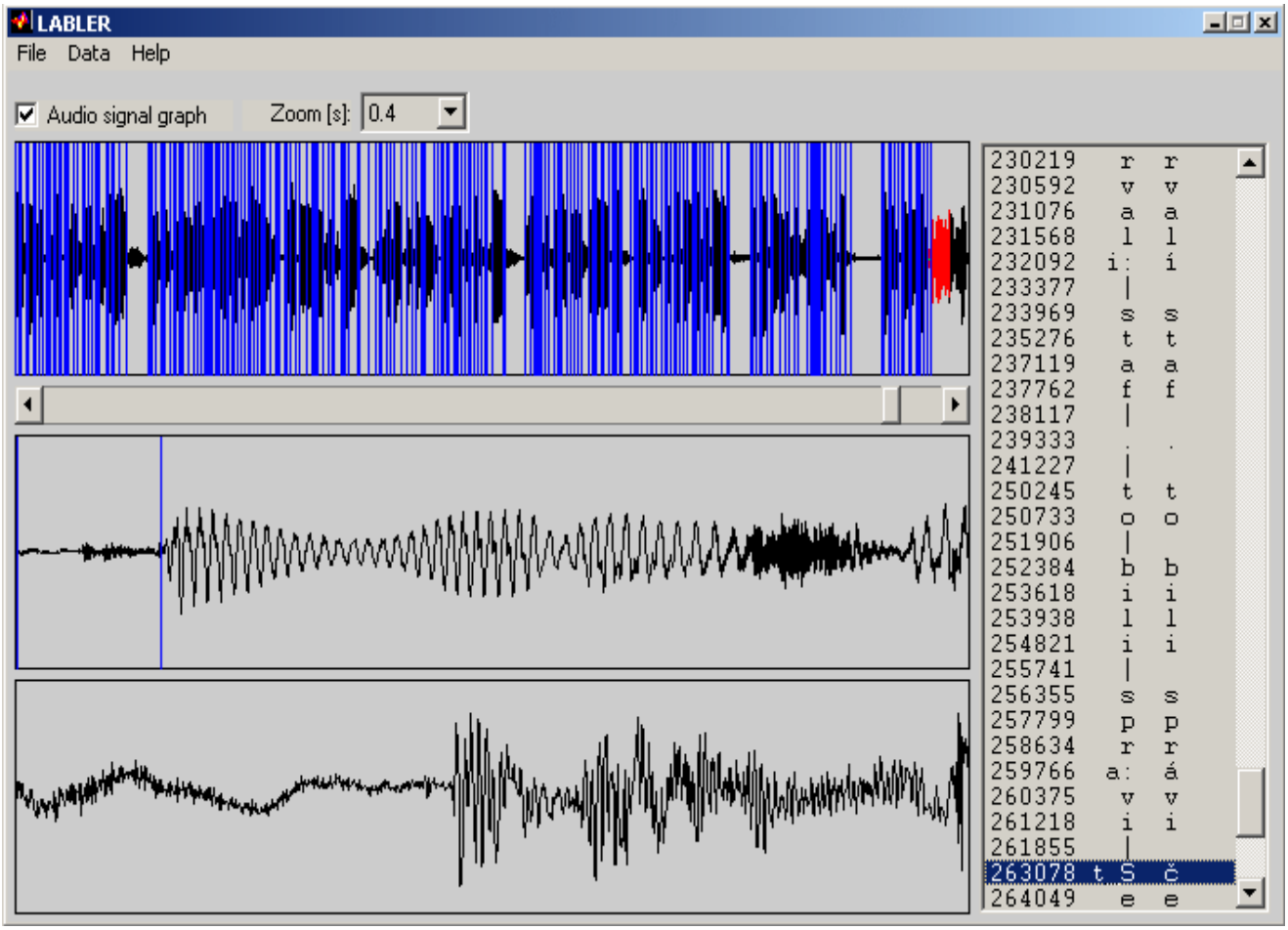
Pro usnadnění labelování je k dispozici kontextové menu **listu fonémů**. Kontextové menu vyvoláme kliknutím pravého tlačítka myši nad aktuálním fonémem. (V některých instalacích MATLAB-u nefunguje kontextové menu. Z toho důvodu jsou tyto nabídky zdvojeny v hlavním "drop-down" menu v položce **Data**.) K dispozici jsou následující volby:

- **Insert phoneme** - vloží na aktuální pozici nový foném.
- **Delete phoneme** - smaže aktuální foném.
- **Change phoneme** - změní aktuální foném.
- **Insert next position** - vloží novou značku na pozici o vzorek dále. Tato volba je vhodná pro označování minimálních vzdáleností. (Např. mezi slovy, která na sebe navazují.)
- **Change position** - změní polohu aktuálního fonému.
- **Translate** - vytvoří nový překlad. Tato volba se používá, pokud dojde k desynchronizaci fonémů a jejich textových reprezentantů.



Během práce v programu Label doporučujeme v pravidelných intervalech zálohovat data na disk. K tomu slouží položka **File/Save/*.lbl**. V souvoru *.lbl se ukládají zároveň pozice i kompletní transkripce věty. Při opětovném otevření souboru *.lbl, volbou **File/Open/*.lbl**, pak již není třeba otevírat transkribční soubor *.trd.





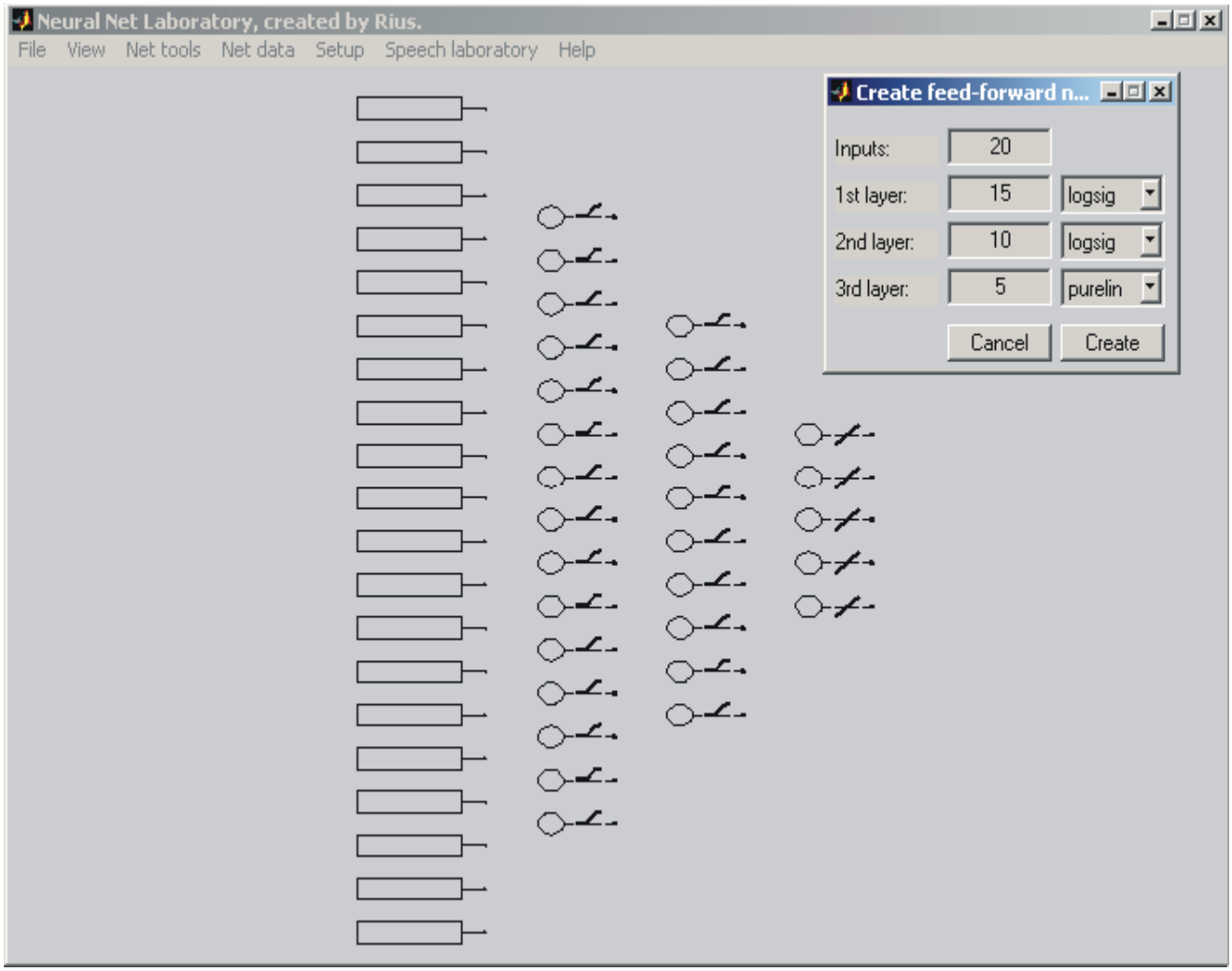
O projektu Neural Net Laboratory

Program *NNL* slouží k usnadnění práce a rozšíření možností při aplikaci neuronových sítí. V projektu jsou implementovány metody pro vytvoření, archivaci, implementaci, trénování a analýzu neuronových sítí. Příklad uživatelského rozhraní zobrazuje následující obrázek.

Program je vnitřně koncipován jako sada nástrojů, které pracují se společnými daty, která reprezentují vlastní neuronové sítě. V projektu *NNL* může být současně zpracováváno několik neuronových sítí najednou. Tato vlastnost dává uživateli možnost porovnávat vlastnosti různých typů neuronových sítí. Zároveň se dá tímto způsobem sledovat závislost výsledků simulací na různých typech vstupních parametrů.

V programu *NNL* byla implementována sada nástrojů pro analýzu neuronových sítí. Byly implementovány následující analyzační metody:

- Graf časových průběhů výstupních vektorů neuronové sítě -- tato analyzační metoda zobrazuje časové průběhy výstupního vektoru a vektoru požadovaných hodnot. Ve stejném grafu je zobrazena i chybová funkce.
- Graf závislosti výstupního vektoru a vektoru požadovaných hodnot -- tato analyzační metoda zobrazuje graficky závislost hodnot výstupního vektoru a vektoru požadovaných hodnot.
- Analýza pracovních bodů neuronů neuronové sítě -- tato analyzační metoda analyzuje polohy pracovních bodů neuronu a odchylky od těchto hodnot.



- Analýza energetického toku neuronovou sítí -- tato metoda sleduje energetické charakteristiky signálů, které jsou nositelem informací mezi jednotlivými neurony.
- Citlivostní analýza výstupů neuronů na hodnoty vstupního vektoru -- tato metoda sleduje citlivost výstupů neuronů na hodnotách vstupního vektoru neuronové sítě.
- Citlivostní analýza výstupů neuronů na vahách neuronové sítě -- tato metoda sleduje citlivost výstupů neuronů na vahách neuronové sítě.
- Citlivostní analýza výstupů neuronů na prazích neuronové sítě -- tato metoda sleduje citlivosti výstupů neuronů na prazích neuronů neuronové sítě.
- Citlivostní analýza výstupů neuronů na výstupech neuronů neuronové sítě -- tato metoda sleduje citlivosti výstupů neuronů na zbylých neuronech neuronové sítě.

File/	View/	Net Tools/	Net Data/	Setup/	Speech Laboratory/	Help/
Open	Data	Create	Refresh Data	Extended logsig	Projekty	Help
Save		Initialize				About
Export		Train				
Import		Simulate				
Exit		Analysis				

Program *NNL* podporuje celou sadu grafických výstupů pro vizualizaci topologie neuronové sítě, klasifikaci vstupních a výstupních dat, atd. V následujících řádcích budou popsány podrobněji jednotlivé položky menu.

Položka menu: Setup/Extended logsig

Tato položka slouží pro nastavení parametrů rozšiřující funkce ***elogsig*** *(extended logsig).* Tato funkce je inovací logické sigmoidy, která umožňuje nastavení nového sklonu, rozsahu a dynamiky této funkce.

Položka menu: Net tools/Train

Pomocí této položky spustíte proces trénování NS. K dispozici je zatím (v dubnu 2006) **beach mode typ trénování** (trainFcn). Před započetním trénování je zapotřebí nahrát ***input*** a ***target*** vektor. K tomu slouží položky menu File/Open.

Položka menu: Net tools/Simulate

Tato položka spouští proces simulace NS. Jako vstupní data pro simulaci slouží **input** vektor, který se do paměti nahrává pomocí volby menu ***File/Open/Input vectore (*.ivd).*** Výsledek simulace je uložen do proměnné ***TrainData.Output,*** který se dá uložit na disk pomocí volby menu ***File/Save/Output vectore (*.out).***

Položka menu: Net tools/Create

Tato položka umožňuje vytvořit novou NS v paměti MATLABu. Pokud se v paměti již nachází nějaká jiná NS, bude novou NS přemazána. Původní data NS budou ztracena. K dispozici jsou následující volby:

Feed-forward
: Vytvoří dopřednou NS požadované topologie.

General
: Vytvoří obecný typ prázdné NS. Velikost a jednotlivé architektury sítě se musí zadat manuálně pomocí struktury uložené v položce menu **Net data**.

Položka menu: Net tools/Analysis

Tato položka obsahuje volby pro analýzu NS. K dispozici je několik typů analýz:

Out=f(Tar)
: Zobrazí graf, ve kterém jsou vyneseny hodnoty target vektoru verzus output vektoru.

Tar,Out,Err=f(Time)
: Zobrazí časové průběhy terget vektoru, output vektoru a průběhu chyby (rozdíl target a output vektoru).

Operation point
: Analýza pracovních bodů neuronů v NS.

Energy flow
: Analýza energetického toku v NS.

Sensitivity

Citlivostní analýza.

Položka menu: View/Data

Tato položka umožňuje zobrazit vnitřní datovou strukturu projektu NNL. Možnost zobrazit datovou strukturu byla vytvořena pro potřeby programátorů přídavných modulů projektu NNL (pro rozšiřování programu). Protože je datová struktura příliš rozsáhlá, byla rozdělena do následujících celků:

TrainData

Proměnná ***TrainData*** je strukturní proměnná, která obsahuje hodnoty input vektoru, target vektoru, ... Jinými slovy se dá říct, že obsahuje všechna obslužná data, která jsou potřeba pro procesy simulace a trénování. Struktura ***TrainData*** obsahuje následující položky:

TrainData.Input
Matice, která obsahuje input vektory pro simulaci a trénování NS.
TrainData.InputTest
Matice, která obsahuje input test vektor pro trénování NS. Společně s ***TargetTest*** vektorem tovoří data testovací množiny pro proces trénování.
TrainData.Target
Matice, která obsahuje target vektory pro trénování NS.
TrainData.TargetTest
Matice, která obsahuje target test vektor pro trénování NS. Společně s ***InputTest*** vektorem tovoří data testovací množiny pro proces trénování.
TrainData.Output
Matice, která obsahuje output data vzniklá procesem simulace input vektoru pomocí NS.
TrainData.TrainingRecord
Struktura, která obsahue záznam průběhu procesu trénování. Struktura záznamu je dána implementací MATLABu**.**
TrainData . TrainFcns
Matice (cell) která obsahuje seznam použitelných trénovacích funkcí. Z této proměnné se určuje, které typy trénovacích funkcí budou použity. Proměnná je inicializovaná při startu NNL.

Net

Proměnná, která nese strukturu NS. Struktura objektu je dána implementací MatLab-u. Podrobné prohlížení jednotlivých částí struktury je možné pomocí položky menu **Net data**. Struktura je nositelem kompletní informace o NS.

Položka menu: File/Save

Položka obsahuje volby pro uložení dat projektu NNL do souborů. K dispozici jsou následujcí typy souborů:

Neural net (*.net)

Uloží data NS do souboru ****.net.*** V souboru jsou uloženy kompletní informace o NS.

Input vectore (*.ivd)

Uloží input vektor do souboru ***.ivd***. Input vektor je chápán jako vstupní data NS, která se používají pro proces simulace nebo trénování.

Target vectore (*.tar)

Uloží target vektor do souboru ***.tar***. Target vektor je chápán jako vzorová výstupní data NS, která se používají pro proces trénování.

Output vectore (*.out)

Uloží output vektor do souboru ***.out.*** Output vektor je chápán jako data, která vzniknou procesem simulování *vstupního vektoru*.

Položka menu: File/Open

Položka obsahuje volby pro nahrání dat ze souborů do projektu NNL. K dispozici jsou následující typy souborů:

Neural net (*.net)

Nahraje NS, která je uložena v souboru ***.net.*** Soubor obsahuje kompletní informace o síti.

Input vectore (*.ivd)

Nahraje input vektor, který je uložený v souboru ***.ivd***. Input vektor slouží jako vstupní data pro simulaci nebo trénování.

Target vectore (*.tar)

Nahraje target vektor, který je uložený v souboru ***.tar.*** Target vektor slouží jako vzorová data pro proces trénování.

Output vectore (*.out)

Nahraje výstupní vektor, který je uložený v souboru ***.out***. Output vektor je výsledkem procesu simulace. (Možnost nahrání výsledku simulace byla zachována pro snažší porovnání výsledků simulace různých sítí.)

Input test vectore (*.ivd)

Mahraje input test vektor, který je uložený v souboru ***.ivd.*** Je-li v paměti nahrán input test vektor, je během procesu trénování zobrazovaná nejen chyba trénování, ale i chyba testovacího souboru dat. Chyba je jen zobrazována, nemá žádný vliv na proces trénování.

Target test vectore (*.ivd)

Mahraje target test vektor, který je uložený v souboru ***.tar.*** Target test vektor je součástí testovací sady dat. Viz **Input test vektor**.

Položka obsahuje volby pro exportování (uložení) dat projektu NNL do souborů. Tato položka slouží pro potřeby separátního uložení vah a prahů NS do dílčích souborů. K dispozici jsou následující možnosti:

Input weights (*.wxx)

Uloží vstupní váhy do souborů ***.wxx***. Jako vstupní váhy jsou chápány váhy, které jsou mezi vstupním vektorem a NS. (1. skrytá vrstva) Značení vyplývá z konvencí MATLABu.

Layer weights (*.wxx)

Uloží váhy zbylých vrstev do souborů ***.wxx***. Jako zbylé vrstvy jsou chápány všechny kromě vstupní. (viz položka výše) Značení vyplývá z konvencí MATLABu.

Biases (*.bxx)

Uloží prahy všech vrstev do souborů ***.bxx***. Jsou uloženy postupně prahy všech vrstev NS.

Položka menu: File/Export

Položka obsahuje volby pro exportování (uložení) dat projektu NNL do souborů. Tato položka slouží pro potřeby separátního uložení vah a prahů NS do dílčích souborů. K dispozici jsou následující možnosti:

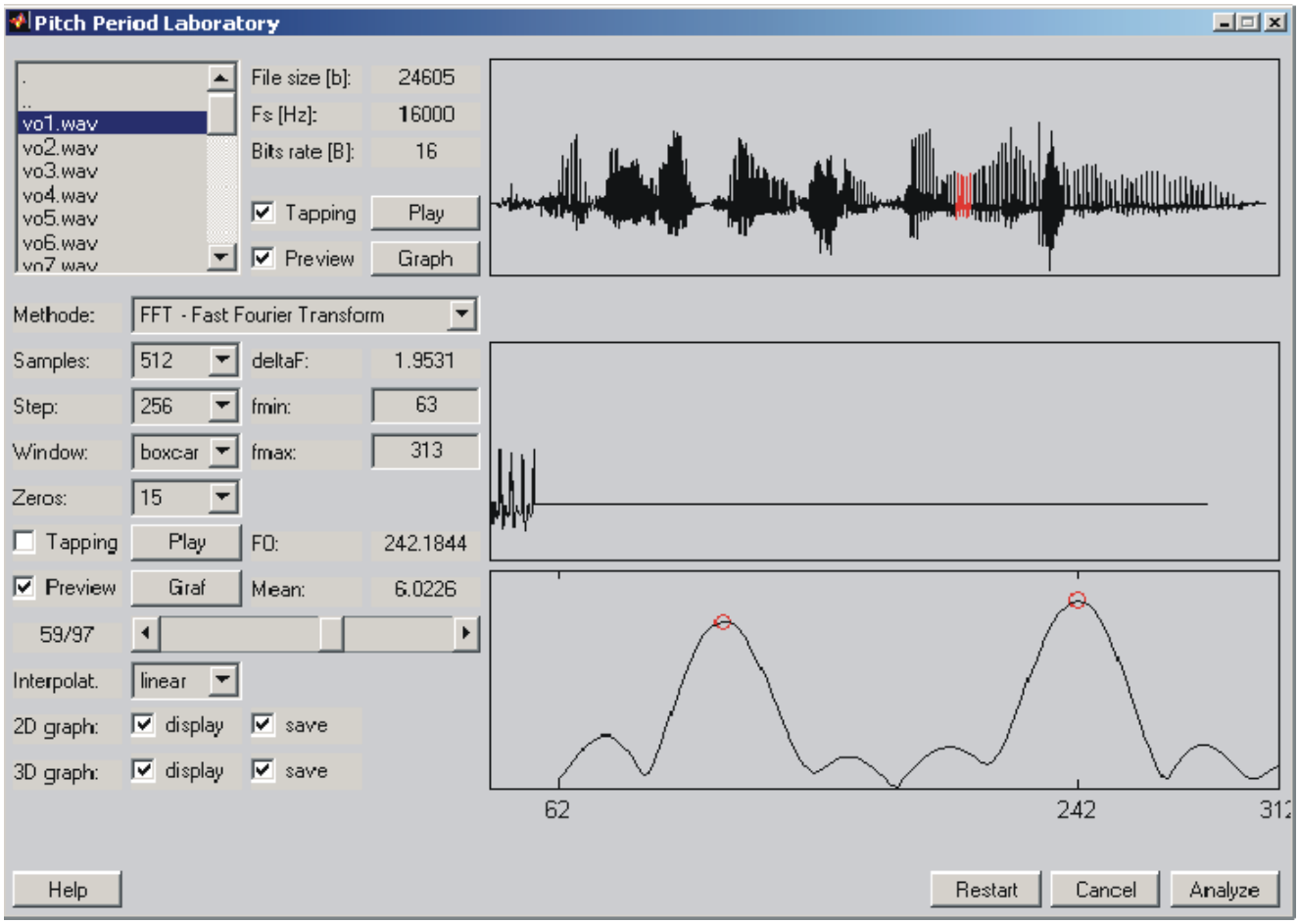
Input weights (*.wxx)
: Uloží vstupní váhy do souborů ***.wxx.*** Jako vstupní váhy jsou chápány váhy, které jsou mezi vstupním vektorem a NS. (1. skrytá vrstva) Značení vyplývá z konvencí MATLABu.

Layer weights (*.wxx)
: Uloží váhy zbylých vrstev do souborů ***.wxx.*** Jako zbylé vrstvy jsou chápány všechny kromě vstupní. (viz položka výše) Značení vyplývá z konvencí MATLABu.

Biases (*.bxx)
: Uloží prahy všech vrstev do souborů ***.bxx.*** Jsou uloženy postupně prahy všech vrstev NS.

O projektu Pitch Period Laboratory

Projekt *{PPL}* byl vytvořen pro extrakci základního hlasivkového tónu z řečového signálu. Cílem projektu bylo vytvořit prostředí, ve kterém by se snadno zpracovávaly nejen jednotlivé věty, ale i celé databáze vět. Základním požadavkem byla plně automatická extrakce f0. Ukázku uživatelského rozhraní programu *{PPL}* zobrazuje obrázek.

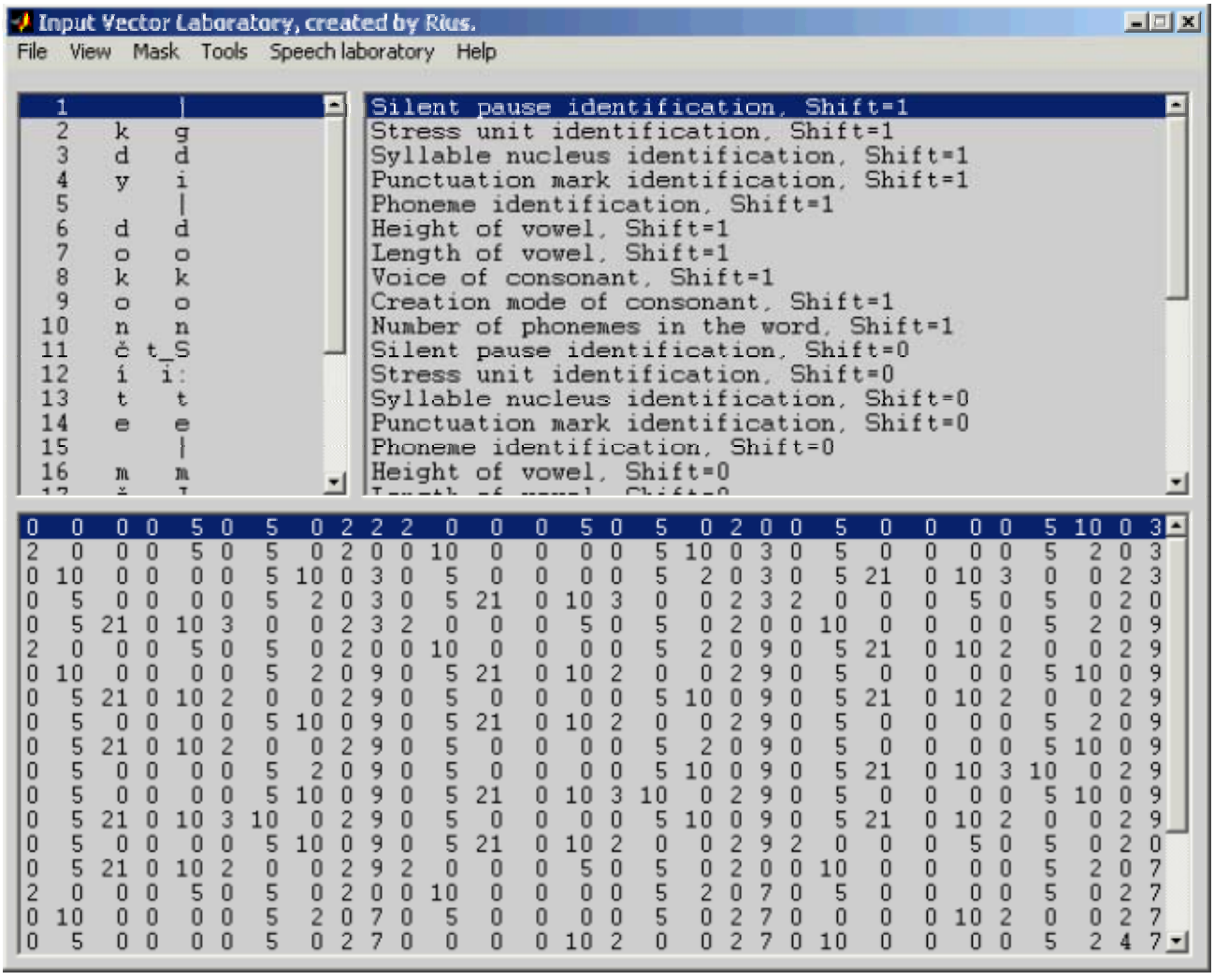


Jako vstupní data slouží data řečového signálu, která musí být uložena v souboru typu `*.wav`. Jedná se o standardní typ souboru, který archivuje data akustického signálu v nekomprimované podobě. Výstupní data reprezentuje posloupnost detekovaných hodnot f_0. Data jsou uložena v textovém souboru ve formě sloupcového vektoru. Program nabízí i velké množství grafických výstupů.

Jako vlastní detekční metody byly použity *ACF* (Auto Correlation Function) a *DFT* (Discrete Fourier Transformation).

O projektu Input Vector Laboratory

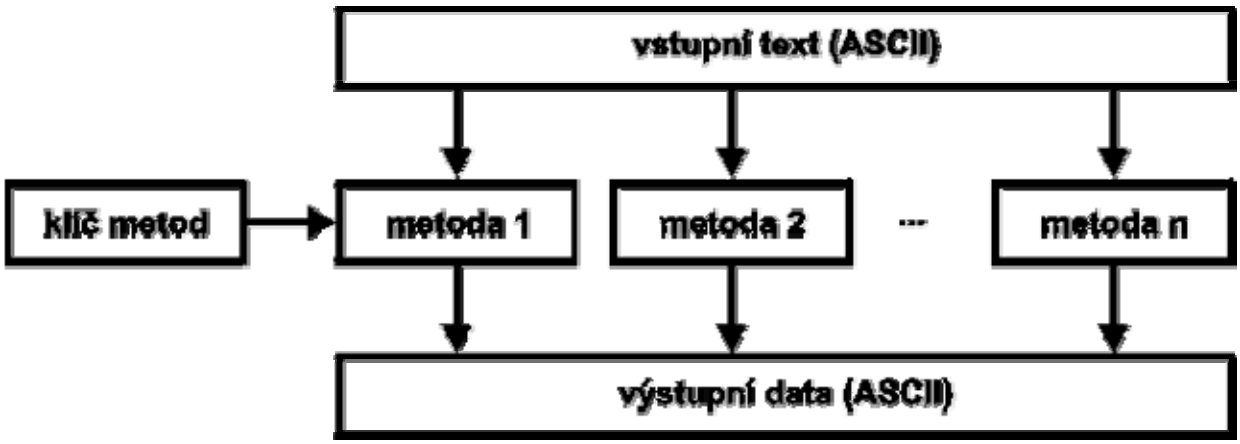
Program *IVL* slouží pro snadnou detekci vstupního textu. Vstupem je prostý text, výstupem pak matice, která se skládá z vektorů charakterizujících jednotlivé fonémy vstupního textu. Příklad uživatelského rozhraní zobrazuje obrázek.



Vnitřní struktura projektu je patrná z následujícího diagramu. Vstupními daty je prostý text, bez formátovacích značek. Tento text je zpracováván pomocí metod, z nichž každá metoda detekuje jeden typ textového parametru. Výstupem metod jsou vektory hodnot, které reprezentují hledané parametry. Složením těchto vektorů vzniknou výstupní data, reprezentující vlastnosti vstupního textu. Pořadí a typy parametrů (tzn. posloupnost a typy

použitých metod) jsou dány klíčem metod. V programu jsou implementovány metody pro detekci následujících parametrů:

- typ fonému -- vokál, konsonant, jiný znak
- typ vokálu -- nízký, střední, vysoký
- délka vokálu -- krátký, dlouhý
- typ konsonantu -- plozívy, afrikáty, frikativy
- znělost konsonantu -- znělý, neznělý, sonorní
- typ pauzy -- mezi slovy, mezi větami, po interpunkčním znaménku
- typ interpunkčního znaménka
- pozice ve slově -- relativní nebo absolutní pozice parametru ve slově
- pozice ve větě/souvětí -- relativní nebo absolutní pozice parametru ve větě/souvětí
- počet fonémů -- ve slově, větě
- detekce slovního přízvuku -- jedná se o detekci začátku slova
- detekce jádra slabiky -- detekuje vybrané hlásky ve slabice

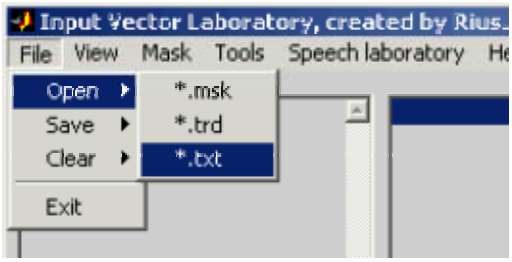


Jednotlivé metody byly implementovány tak, aby se jejich počet dal jednoduchým způsobem rozšířit a umožnit tak detekci nových typů parametrů. Každá z metod má navíc možnost detekce fonému jak v místě fokusu, tak v libovolné poloze před i za ním.

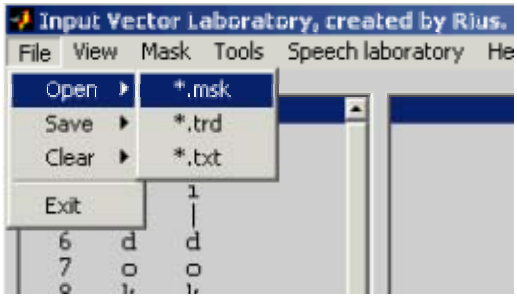
Způsob ovládání programu IVL

V následujících bodech jsou popsány kroky, které jsou nutné k vytvoření vstupního vektoru pomocí programu *IVL*.

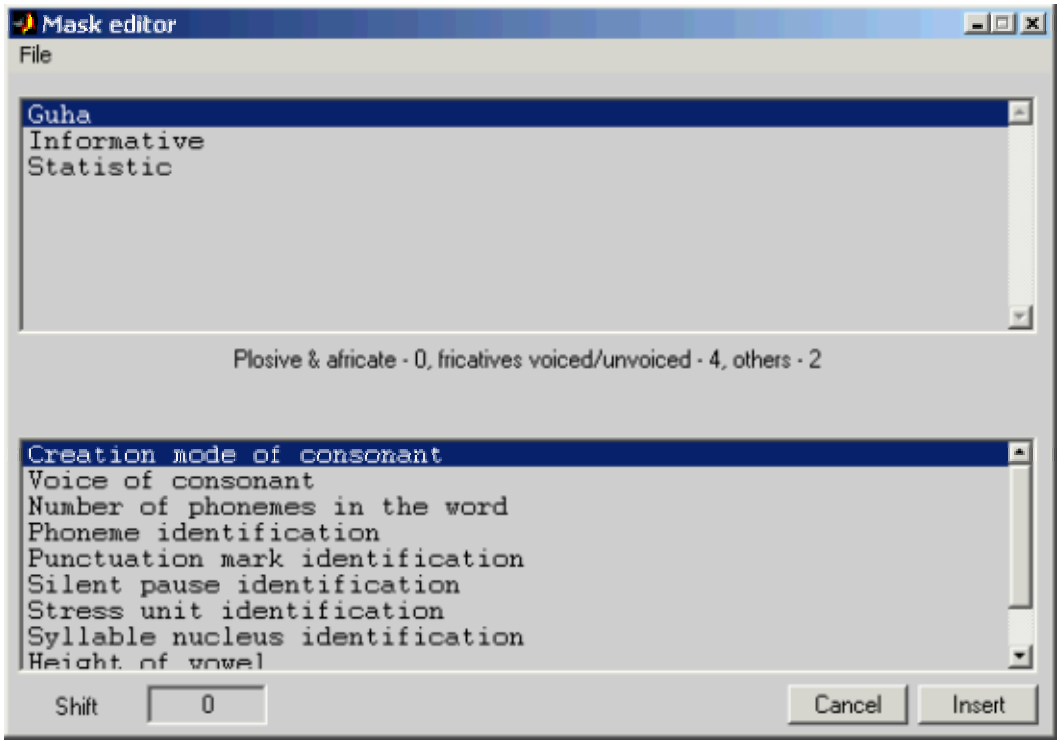
- Spusťte základní rozhraní programu. Jsou-li správně nastavené cesty, postačí napsat **`ivl`** na příkazovou řádku *MATLABu*. Program se dá jednoduše spustit z rozhraní projektu *Speech Laboratory*.
- Nahrajte vstupní text, jehož parametry se mají detekovat. Text se dá nahrát z textového souboru **(`*.txt`)** nebo z transkripčního souboru **(`*.trd`)**.



- Nahrajte nebo vytvořte masku metod. Tato maska určuje pořadí a typy parametrů, které se budou generovat. Máte-li již masku uloženu někde na disku, pak použijte volbu pro otevření masky. Maska je standardně uložená v souboru typu **`*.msk`.**



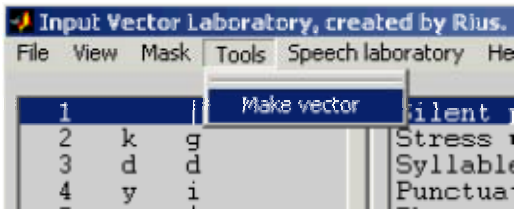
Nemáte-li masku uloženou, musíte ji vytvořit. Z hlavního menu zvolte položku **`Mask/Insert Method`**. Zobrazí se vám následující okno.



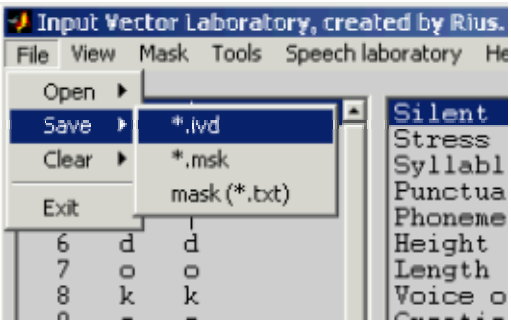
V horní části okna si zvolte okruh požadovaných metod. Ve spodní části se zobrazí dostupné metody z daného okruhu. Zvolte si požadovanou metodu. V informačním poli (střední část okna) se zobrazí stručný popis metody. Tlačítkem `Insert` vložíte metodu do masky. Metoda se vloží na aktuální pozici, což je pozice, která je zvýrazněná řádkem jiné barvy. Tuto pozici lze libovolně měnit, a to kliknutím myši na

libovolný řádek masky. Chcete-li zařadit metodu na konec masky, klikněte na první prázdný řádek za výpisem masky.

- Následuje vygenerování vektoru hodnot. K tomuto účelu slouží volba **`Tools/Make Vector`** z hlavního menu. V závislosti na množství fonémů vstupního textu a počtu použitých metod to může trvat i několik minut. Po dobu generování dat se na obrazovce zobrazí **bargraf**, který procentuálně ukazuje průběh generování dat.



- Nyní je vstupní vektor vytvořen. Nachází se ale jen v paměti počítače. K uložení vstupního vektoru do souboru **`*.ivd`** slouží položka **`File/Save/*.ivd`** z hlavního menu.



- Na disku máme nyní uložená data vstupního vektoru. Pokud jsme měnili nebo vytvářeli novou masku metod, máme možnost ji také uložit. K tomu slouží položka **`File/Save/*.msk`** z hlavního menu.

Zmena parametrizace vstupního vektoru

Změna parametrizace vstupního vektoru vyžaduje programátorský zásah do detekčních rutin. Každý parametr je vždy detekován jednou detekční rutinou. Detekční rutiny jsou uloženy v adresáři:

…/IVL/methodes/*.*

Způsob úpravy parametrizace ukážeme na konkrétním příkladu. Následující text je výpisem metody pro detekci typu hlásky:

```
function Data = guhaphonid(Shift)
% Input Vector Laboratory
% Created for Czech Technical University in Prague
% made by Ing. Jiri Santarius
% dep. of Circuit Theory
% CVUT-FEL
```

```
% All rights reserved.
%
% Methode:
%
```

První část je tvořená jen definicí funkce a poznámkami, které se zobrazí při vyvolání ***help*** funkce. V následující části

```
%Variables
global IVL
```

se zpřístupní globální proměnná *IVL* pro použití. Následuje část programu, která obsahuje informace o metodě. Z této části metody se projekt IVL dozvídá informace o názvu detekční metody a jejím popisu:

```
%Identification
if nargin == 0
Data.Fcn = 'guhaphonid';
Data.Shift = 0;
Data.Name = 'Phoneme identification';
Data.Group = 'Guha';
Data.Description = 'Vowel - 10, consonant - 0, others - 5';
return;
end
```

Tato část programu je jen informační a neovlivňuje průběh detekčních algoritmů. Jde jen o textové informace, které se zobrazí uživateli při používání programu IVL. Vlastní dekódovací algoritmus obsahuje následující kód:

```
%Decoding
Data={}; i=1;
while i <= size(IVL.Text,2)
if ((i-Shift) < 1) | ((i-Shift) > size(IVL.Text,2))
Data{i} = '5';
else
switch lower(IVL.Text{i-Shift})
case {'a','e','#','i','y','o','u','au','ou'}
Data{i} = '10';
case {'á','é','í','ý','ó','ú','u'}
Data{i} = '10';
case {'h','ch','k','r','d','t','n'}
Data{i} = '0';
case {'ž','š','c','r','c','j','d','t','n'}
Data{i} = '0';
case {'b','f','l','m','p','s','v','z'}
Data{i} = '0';
case {'e'}
Data{i} = '0';
otherwise
Data{i} = '5';
```

```
end
end
i=i+1;
end
```

Na pátém řádku je hodnota, která se má vložit na pozici, která je mimo detekovatelný text. Tato hodnota se vkládá v případě, že je použitý nenulový posuv **(*shift*).** Od osmého řádku jsou kombinace vyjmenovaných znaků (za příkazem ***case***) a hodnot, které se k těmto znakům přiřadí. V případě, že v těchto příkazech není zvolená hláska, je použita hodnota za příkazem ***otherwise*** (v tomto případě 5).
Při přepisování hodnot, které se mají vkládat na detekované pozice je přitom důležité, aby hodnoty byly vepisovány ve formě textu. Chceme-li, aby se na pozici samohlásek 'á','é','í','ý','ó','ú','u' zapisovala hodnota 12, pak v rutině upravíme kód následovně:

```
case {'á','é','í','ý','ó','ú','u'}
Data{i} = '12';
```

Jednotlivé metody jsou uloženy pod názvem, ze kterého se nedá snadno zjistit, o jakou metodu se jedná. Následující tabulka proto udává názvy souboru a metod, které se v nich nacházejí.

guhaconscreation Creation mode of consonant
guhaconsvoice Voice of consonant
guhafoncount Number of phonemes in the word
guhaphonid Phoneme identification
guhapunctmark Punctuation mark identification
guhasilpausid Silent pause identification
guhastresunit Stress unit identification
guhasyllnucleus Syllable nucleus identification
guhavowheight Height of vowel
guhavowlength Length of vowel
index Position index
letter Actual letter
phonem Actual phonem
positioninfirstsubsentence Relative position in first subsentence
positioninlastsubsentence Relative position in last subsentence
positioninsentence Relative position in complex sentence
positioninsubsentence Relative position in subsentence
positioninword Relative position in word

Tvorba vstupního vektoru programem IVL

V následujících bodech bude popsáno, jakým způsobem se vytvářejí vstupní vektory neuronových sítí, pomocí programu IVL.

- Nastavte aktuální adresář MATLABu do složky ***…/sl/*.**
- Příkazem ***sl*** spusťte hlavní okno projektu ***Speech Laboratory*.**
- Zvolte program ***IVL* (*Input Vector Laboratory*).**

Tím jste spustili program pro tvorbu vstupních vektorů. V dalších bodech již budete pracovat s tímto programem.

- Načtete masku metod ze souboru ***.msk.*** Maska se otevírá pomocí volby ***File/Open/*.msk*** z hlavního menu. Pokud maska vstupního vektoru nebyla ještě vytvořena, pak ji vytvořte pomocí volby ***Mask/Insert Method***. Vytvořenou masku lze uložit na disk pomocí volby ***File/Save/*.msk***.
Maska tvoří klíč, podle kterého se pak budou generovat jednotlivé vstupní vektory. Určuje pořadí a typ generovaných dat.

- Načtěte vstupní text, ke kterému chcete vytvořit vstupní datový vektor pro neuronovou síť. K načtení použijte volbu:

a) ***File/Open/*.trd*** pro načtení fonetického překladu vstupního textu,
b) ***File/Open/*.txt*** pro načtení textu v jeho originální podobě.

- Nyní máte nahrána všechna potřebná data a můžete spustit samotný překlad. Překlad zahájíte pomocí volby ***Tools/Make Vector***. V paměti počítače se vytvoří požadovaný vektor. Podle složitosti masky metod a velikosti vstupního textu může vlastní překlad trvat i několik minut.
- Vytvořená data uložíte na disk pomocí volby ***File/Save/*.ivd.*** V souboru, který jste v předchozím bodě zvolili, se pak nachází požadovaná data.
- Chcete-li překládat další vstupní text, opakujte od bodu, ve kterém načítáte vstupní text.

Literatura:

[1] Santarius, J.: Systémy pro modelování prozódie syntetické řeči. Disertační práce, FEL ČVUT v Praze, 2005.

[2] http://artin.zcu.cz/projects/tts/index.htm