# Uživatelská dokumentace k projektu Morfo

David Kolovratník a Leoš Přikryl

27. května 2008

# Obsah

# Kapitola 1

# Instalace

## 1.1 Softwarové požadavky

- gcc 3.4 nebo vyšší,
- Perl 5.8.1 nebo vyšší
- SWIG 1.3 nebo vyšší
- Perl/Tk 804.024 nebo vyšší
- Flex 2.5.4 nebo vyšší (je potřeba jen pro převod starého formátu na XML, všechno ostatní funguje i bez něj)

Program je určen pro operační systémy Linux. K instalaci doporučujeme použít Bash shell.

Před instalací je nutno přidat do proměnné prostředí PERLLIB cestu k souboru `Tk.pm` (součást Perl/Tk).

## 1.2 Rychlá instalace

Nejjednodušší cesta k nainstalování projektu je spuštění příkazů:

```
make
```

```
make install
```

v adresáři, kam byl rozbalen balíček `morfo.tar.bz2`. Projekt se nainstaluje do následujících adresářů:

- datové soubory se nainstalují do `/usr/local/share/morfo`,
- programy se nainstalují do `/usr/local/bin`,
- perlovské moduly se nainstalují do `/usr/local/share/perl5/Morfo`,
- sdílené knihovny se nainstalují do `/usr/local/lib/morfo`.

Tuto instalaci může provést pouze uživatel, který má právo zápisu do těchto adresářů (typicky root).

Před prvním použitím editoru si uživatel musí ve svém domovském adresáři vytvořit soubor `.morforc`, který bude obsahovat alespoň položku author (viz sekce 2.3.2).

Pokud vám rychlá instalace nevyhovuje, můžete si instalaci přizpůsobit svým potřebám - viz další kapitoly.

## 1.3 Změna instalačního adresáře

Jednou z častých změn oproti rychlé instalaci pravděpodobně bude změna instalačního adresáře. Ta se provede změnou makra CONFIG_PREFIX v souboru `common/config.h`. Defaultně je to `/usr/local`. Pokud tuto hodnotu ještě před spuštěním make změníte, nainstaluje se projekt do stejných adresářů jako při rychlé instalaci, jen počáteční adresář `/usr/local` v cestě bude nahrazen vaším zadaným adresářem.

## 1.4 Pokročilá nastavení při instalaci

Instalační adresář není jediné nastavení, které lze změnit. Je možné nastavit mnohem více parametrů. Jedná se převážně o implicitní hodnoty parametrů, které lze uvádět na příkazové řádce nebo v konfiguračním souboru. Nastavení se provádí definicí maker v souboru `common/config.h`.

SEZNAM PARAMETRŮ NASTAVITELNÝCH PŘED KOMPILACÍ

**CONFIG_PREFIX** instalační adresář projektu,

**CONFIG_DATA_DIR** adresář, kam budou uložena platformově nezávislá data

**CONFIG_BIN_DIR** adresář pro programy

**CONFIG_LIB_DIR** adresář pro knihovny

**CONFIG_PERL_MOD_DIR** adresář pro perlovské moduly

**CONFIG_DEFAULT_PRIMARY_FILE** cesta k morfologickému slovníku (makro nemusí být definováno),

**CONFIG_DEFAULT_INDEX_FILE** cesta k indexovému souboru pro morfologický slovník (makro nemusí být definováno),

VAROVÁNÍ

Indexový soubor může být přepsán.

**CONFIG_DEFAULT_TAG_FILE** cesta k seznamu značek (makro nemusí být definováno),

**CONFIG_DEFAULT_PATTERN_FILE** cesta k seznamu ohýbacích vzorů (makro nemusí být definováno),

**CONFIG_DEFAULT_DERIVATION_FILE** cesta k seznamu derivačních vzorů (makro nemusí být definováno),

**CONFIG_DEFAULT_GUESSER_TABLE_FILE** cesta k tabulce pro hádání vzorů (makro nemusí být definováno),

**CONFIG_DEFAULT_SEMANTIC_FLAGS_FILE** cesta k seznamu a popisu sématnických příznaků (makro nemusí být definováno),

**CONFIG_DEFAULT_SYNTACTIC_FLAGS_FILE** cesta k seznamu a popisu syntaktických příznaků (makro nemusí být definováno),

**CONFIG_DEFAULT_STYLE_FLAGS_FILE** cesta k seznamu a popisu stylových příznaků (makro nemusí být definováno),

**CONFIG_DEFAULT_TAG_POSITIONS_FILE** cesta k souboru s popisem jednotlivých pozic v tagu (makro nemusí být definováno),

**CONFIG_DEFAULT_EXCEPTIONS_FILE** cesta k souboru se seznamem výjimek při hádání vzorů (makro nemusí být definováno),

**CONFIG_DEFAULT_DERIVED_FILE** cesta k souboru s rozgenerovaným slovníkem (makro nemusí být definováno),

**CONFIG_DEFAULT_OLD_FORMAT_DEFAULT_FILENAME** jméno souboru, do kterého se při exportu do starého formátu uloží slova, která nemají definovaný původní soubor (makro nemusí být definováno),

**CONFIG_AUTOMAT_LEMMA_FILE** cesta k souboru automatu obsahujícímu lemmata (pokud není makro definováno cesta musí být nastavena konfiguračním souborem),

**CONFIG_AUTOMAT_TAG_FILE** cesta k souboru automatu obsahujícímu tagy (pokud není makro definováno cesta musí být nastavena konfiguračním souborem),

**CONFIG_AUTOMAT_TRIE_FILE** cesta k souboru automatu obsahujícímu trie (pokud není makro definováno cesta musí být nastavena konfiguračním souborem),

**CONFIG_AUTOMAT_ENDNODE_FILE** cesta k souboru automatu obsahujícímu koncové uzly (pokud není makro definováno cesta musí být nastavena konfiguračním souborem),

**CONFIG_USERS_CONFIG_FILE** jméno konfiguračního souboru v domácím adresáři užitavele.

**CONFIG_READ_SYSTEM_WIDE_PRIOR_USERS_CONFIGURATION** Makro ovlivňuje pořadí hledání konfiguračního souboru. Je-li definováno, zkusí se zpracovat systémový konfigurační soubor a poté uživatelský. Není-li makro definováno, zpracování začne uživatelským souborem. Systémový bude zpracován jen v případě, že uživatelský neexistuje (selhalo otevření). (makro nemusí být definováno)

Cesty k adresářům se zadávají bez lomítka na konci. Všechna makra začínající CONFIG_DEFAULT určují výchozí nastavení. Tato nastavení může každý uživatel změnit pomocí konfiguračního souboru .morforc nebo u některých programů pomocí parametrů na příkazové řádce.

V konfiguraci lze využít makro CONFIG_MORFO_SRC_DIR. To je překladovým systémem nastaveno na absolutní cestu ke kompilačnímu adresáři.

## 1.5 Seznam nainstalovaných programů

Instalace vytvoří v adresáři `bin` následující programy:

**morfoEd** editor slovníku, popis viz XrefId[?kapitola Editor?]

**morfo-wib** program pro stavbu indexu, popis viz kapitola Indexace

**morfo-export** program pro export slovníku do XML a starého formátu, popis viz kapitola Slévání, export

**analyzeFiles** program pro morfologickou analýzu souborů v CSTS formátu, popis viz XrefId[?kapitola Analýza csts souborů?]

**analyzeWord** program pro morfologickou analýzu jednoho slovního tvaru, popis viz XrefId[?kapitola Morfologická analýza jednoho slova?]

**derive_aut** program pro generování automatu, popis viz XrefId[?kapitola Generování pro potřeby morfologické analýzy?]

**derive_cmd** program pro rozgenerování celého slovníku, popis viz XrefId[?kapitola Rozgenerování slovníku?]

**derive_word_cmd** program pro derivaci jednoho slova, popis viz XrefId[?kapitola Rozgenerování jednoho slova?]

**derive_file_cmd** program pro derivování seznamu slov, uloženého v souboru, popis viz XrefId[?kapitola Rozgenerování více slov ze souboru?]

## 1.6 Patch na Perl/Tk 804.024

Perl/Tk 804.024 má chybu, který způsobuje, že v některých window managerech se nezobrazí správně české znaky v titulku okna. V adresáři `slovnik` najdete patch, který tuto chybu řeší (`Tk804.027patch`). Patch mění přímo zdrojový kód Perl/Tk, takže po jeho aplikace je potřeba celý Perl/Tk překompilovat. Pache aplikujete příkazem

```
patch pTk/mTk/unix/tkUnixWm.c < MORFO_DIR/slovnik/Tk804.027patch
```

spuštěným v adresáři Tk-804.027 (adresář, kde je rozbalen instalační balíček Perl/Tk). MORFO_DIR nahraďte cestou k instalačnímu adresáři Morfa.

# Kapitola 2

# Všeobecné informace

## 2.1 Autorská práva

Autoři studentského projektu Morfo souhlasí s využitím výsledků své práce v souladu s licencí *zlib/libpng* <http://www.gzip.org/zlib/zlib_license.html>. Tato licence byla nadací Free Software Foundation <http://www.fsf.org> uznána za free software licenci, iniciativou Open Source Initiative <http://www.opensource.org> za open source licenci. Je kompatibilní s GPL.

> This software is provided 'as-is', without any express or implied warranty. In no event will the authors be held liable for any damages arising from the use of this software.
>
> Permission is granted to anyone to use this software for any purpose, including commercial applications, and to alter it and redistribute it freely, subject to the following restrictions:
>
> 1 The origin of this software must not be misrepresented; you must not claim that you wrote the original software. If you use this software in a product, an acknowledgment in the product documentation would be appreciated but is not required.
>
> 2 Altered source versions must be plainly marked as such, and must not be misrepresented as being the original software.
>
> 3 This notice may not be removed or altered from any source distribution.

VZTAH K DATŮM

Data Ústavu formální a aplikované lingvistiky (zejména morfologický slovník) nejsou výsledkem této práce, nepodléhají vůli autorů studentského projektu a nejsou dostupná pod licencí studentského projektu Morfo.

## 2.2 Terminologie

V dokumentaci se objevují slova a slovní spojení ve specifickém významu - termíny. Pro správné pochopení definujeme následující.

POZNÁMKA

Pro lepší pochopení termínů *slovníkové heslo*, *společné lemma* a *alternativní lemma* doporučuji příklad 3.3.1.

**slovníkové heslo** Všechny slovní tvary, které mají alespoň jedno lemma shodné. Součástí slovníkového hesla jsou také všechny atributy příslušející k těmto tvarům. Editor zobrazuje vždy právě jedno slovníkové heslo. Slovníkové heslo se v programátorské části někdy také nazývá XrefId[?struktura pro lemma?].

**společné lemma** Každé slovníkové heslo má jedno lemma, které je společné všem jeho tvarům. Toto lemma nazýváme společné lemma. Viz též struktura slovníku (6.3).

**alternativní lemma** Kromě společného lemmatu může mít každý slovní tvar jedno další lemma - alternativní lemma. Viz též struktura slovníku (6.3).

**tag** V tomto textu znamená tag patnáctipoziční morfologickou značku, která se používá například v ČNK. Jednotlivé pozice kódují různé morfologické informace o slovním tvaru (slovní druh, rod, číslo, atd.). Popis najdete v [1].

**starý formát** Poziční formát slovníku a slovníčků značek, derivačních a ohýbacích vzorů používaný na ÚFALu. Od tohoto formátu se postupně ustupuje a měl by jej nahradit nový XML formát.

**kofix** Tímto termínem je myšlen základ slova, který je spolu s lemmatem uveden ve slovníku lemmat. Někdy se označuje i jako kořen, ale kořen ve smyslu české gramatiky to není, proto raději tento termín nebudeme používat. Kofix je vlastně taková část slova, ke které se pak při ohýbání slov přidávají koncovky, aby se tak vytvořily jednotlivé tvary slov. Je to nejdelší možný řetězec, ve kterém se všechny výsledné tvary slova shodují (v rámci jednoho ohýbacího vzoru, tj. jednoho vyderivovaného slova). V případě, že se lemmata nederivují (mají ve slovníku uveden už rovnou výsledný tvar slova), kofix odpovídá právě výslednému tvaru slova. Pro každé jedno entry ve slovníku existuje právě jeden kofix. Při derivování se z původního kofixu ze slovníku tvoří nový kofix pro každé jedno vyderivované lemma.

Příklad: pro lemma „kocour“ máme ve slovníku například kofix „kocou“. Je patrné, že to není kořen tak, jak ho známe z české gramatiky. Ale má svůj smysl. K tomuto kofixu se dají rovnou přidávat koncovky, spolu s nimiž pak kofix vytvoří výsledné slovo. Můžeme tedy pouhým přiřetězením koncovky vytvořit slova jako „kocourovi“, „kocouří“, „kocourem“, ... V případě slov, která mohou tvořit i negaci a třetí stupeň, je možné ještě před kofix předřetězit předpony „ne“, „nej“ či „nejne“. Je však patrné, že pro všechna výsledná slova vzniklá z určitého kofixu je tento kofix nejdelším řetězcem obsaženým zároveň ve všech slovech.

**primární soubor** Soubor obsahující celý slovník v XML formátu. Během práce s editorem se tento soubor nemění. Všechny změny jsou uloženy do souboru změn.

**soubor změn** Soubor, do kterého se při práci s editorem ukládají všechny změny. Tento soubor je mnohem menší než celý slovník (primární soubor) a jde tak snadno přenášet mezi počítači. Soubor změn také umožňuje práci více uživatelů nad jediným primárním souborem.

**prefix** V tomto textu neznamená prefix předponu v klasickém lingvistickém smyslu, ale jakýkoliv počáteční úsek nějakého řetězce (včetně celého řetězce).

**sufix** V tomto textu neznamená sufix koncovku v klasickém lingvistickém smyslu, ale jakýkoliv koncový úsek nějakého řetězce (včetně celého řetězce).

**koncovka** Viz sufix.

**derivační vzor** Vzor, který je uveden u slov ve slovníku. Tento vzor odkazuje na jeden nebo více vzorů ve slovníčku ohýbacích vzorů, ze kterých jsou pak odvozeny jednotlivé slovní tvary. Vyskytne-li se v textu pouze slovo vzor bez bližšího určení, rozumí se tím vzor derivační.

**ohýbací vzor** Vzory, na které ukazují derivační vzory. Každému ohýbacímu vzoru přísluší několik konců slov a příslušné tagy. Přidáním konce slova ke kofixu vznikne slovní tvar s příslušným tagem.

**záznam** Záznam referuje k nejmenší ucelené části databáze morfologického slovníku. Více viz interpretace záznamu (6.4.2). Jeho realizace v konkrétním datovém formátu je označována entry.

**entry** Realizace záznamu v konkrétním datovém formátu. Pro ilustraci poslouží příklad z morfologického slovníku (3.2.4).

## 2.3 Konfigurační soubor

### 2.3.1 Umístění a zpracování konfiguračních souborů

Umístění a zpracování konfiguračních souborů lze upravit před kompilací nastaveními v souboru `common/config.h`. Úplný popis nastavení je uveden v kapitole 1.4, Pokročilá nastavení při instalaci.

Jméno uživatelského konfiguračního souboru definuje makro CONFIG_USERS_CONFIG_FILE (defaultně nastaveno na `.morforc`). To se připojí před cestu zjištěnou z proměnné prostředí `HOME`. Není-li nastavena, uživatelský konfigurační soubor nebude zpracován. Systémový konfigurační soubor se hledá na cestě dané makrem CONFIG_SYSTEM_WIDE_CONFIG_FILE, je-li definováno.

Je-li definováno makro CONFIG_READ_SYSTEM_WIDE_PRIOR_USERS_CONFIGURATION, zkusí se zpracovat systémový konfigurační soubor a poté uživatelský. Není-li makro definováno, zpracování začne uživatelským souborem. Systémový bude zpracován jen v případě, že uživatelský neexistuje (selhalo otevření). Hodnoty nastavovaných parametrů lze přebít parametry předanými na příkazové řádce.

### 2.3.2 Formát konfiguračního souboru

Konfigurační soubor je posloupnost dvojic <klíč, hodnota>. Záznam začíná klíčem. Klíč je ukončen jedním nebo více tabulátory (*key-value separator*). Klíč musí být neprázdný. Následuje hodnota. Ta je ukončena znakem nového řádku, pokud jej nepředchází znak \ (*backslash*).

#### 2.3.2.1 Komentář

Znak # uvozuje komentář. Komentář je ignorován. Končí znakem nového řádku.

Prázdné řádky (řádky odpovídající regulárnímu výrazu `/^$/`) se ignorují.

#### 2.3.2.2 Zpracování klíčů a hodnot

Dvojice znaků `\n` se převede na znak nového řádku. Dvojice znaků `\t` se převede na znak tabulátoru. Znak `\` ruší speciální chování znaků `\\` a `#`. Před jinými znaky se chová jako ostatní znaky, nemá speciální význam, stává se součástí textu.

Vícenásobné uvedení téhož klíče není chybou. Hodnoty se ke klíči řadí do seznamu v uvedeném pořadí nebo přepisují původní hodnotu (záleží na klíči).

Tabulátory na začátku pokračování předchozí řádky (ukončené znakem `\`) se ignorují.

### 2.3.3 Nastavitelné parametry

V konfiguračních souborech (uživatelském i systémovém) lze nastavit tyto parametry:

**`primary_file`** cesta k morfologickému slovníku,

**`index_file`** cesta k indexovému souboru pro morfologický slovník,

**`tag_file`** cesta k seznamu značek

**`pattern_file`** cesta k seznamu ohýbacích vzorů,

**`derivation_file`** cesta k seznamu derivačních vzorů,

**`guesser_table_file`** cesta k souboru s tabulkou pro hádání vzorů,

**`syntactic_flags_file`** cesta k souboru se seznamem syntaktických příznaků,

**`semantic_flags_file`** cesta k souboru se seznamem sémantických příznaků,

**`tag_positions_file`** cesta k souboru se seznamem možných hodnot na jednotlivých pozicích v tagu,

**`exceptions_file`** cesta k souboru s vyjímkami při hádání vzorů,

**`old_format_default_filename`** jméno souboru, do kterého se při exportu do starého formátu uloží slova, která nemají definovaný original file,

**`author`** autor, pracující s editorem (signatura, která se uloží do created a modified). Tento parametr je povinný (nemá zakompilovanou hodnotu) a nejde bez něj spustit editor.

**`automat_lemma_file`** Jméno souboru, ve kterém má automat pro morfologickou analýzu uložená lemmata. Při generování se vytváří soubor s tímto názvem a analýza předpokládá jeho existenci.

**`automat_tag_file`** Jméno souboru, ve kterém má automat pro morfologickou analýzu uložené tagy. Při generování se vytváří soubor s tímto názvem a analýza předpokládá jeho existenci.

**`automat_trie_file`** Jméno souboru, ve kterém je uložen vlastní automat pro morfologickou analýzu. Při generování se vytváří soubor s tímto názvem a analýza předpokládá jeho existenci.

**automat_endNode_file** Jméno souboru, ve kterém má automat pro morfologickou analýzu uložené koncové uzly. Při generování se vytváří soubor s tímto názvem a analýza předpokládá jeho existenci.

Uživatelská nastavení mají přednost před systémovými, nastavení v konfiguračním souboru mají přednost před zakompilovanými.

## 2.4 Společné parametry z příkazové řádky

Většina programů potřebuje ke své práci pět zdrojů. Ty lze zadat z příkazové řádky následujícími parametry. Tyto parametry lze použít pro všechny programy kromě editoru. Ten načítá nastavení vždy z konfiguračního souboru.

SPOLEČNÉ PARAMETRY Z PŘÍKAZOVÉ ŘÁDKY

***-P*, *--primary-file* soubor** cesta k morfologickému slovníku,

***-I*, *--index-file* soubor** cesta k indexovému souboru pro morfologický slovník,

***-T*, *--tag-file* soubor** cesta ke slovníčku značek,

***-A*, *--pattern-file* soubor** cesta ke slovníčku ohýbacích vzorů,

***-D*, *--derivation-file* soubor** cesta ke slovníčku derivačních vzorů.

***--no-compiled-paths*** Zakazuje použití zakompilovaných cest k výše jmenovaným zdrojům.

# Kapitola 3

# Vstup/Výstup

## 3.1 Datové zdroje

Modul V/V připravuje informace uložené v souborech ke zpracování do paměti. Ze zvolené perzistentní reprezentace konstruuje reprezentaci vhodnou k výpočtu. Pro data, která se mění, zajišťuje možnost odpovídající úpravy původního zdroje.

Modul V/V obstarává vstup těchto typů zdrojů:

- seznam pozičních značek,
- seznam derivačních vzorů,
- seznam ohýbacích vzorů,
- morfologický slovník,
- konfigurační soubor.

Modul V/V obstarává aktualizaci těchto typů zdrojů:

- morfologický slovník,
- index morfologického slovníku.

## 3.2 Formát datových zdrojů

Data jsou uložena ve strukturovaných textových souborech. Struktura je vyznačena značkami XML standardu. Soubory jsou dobře tvořenými (*well formed*) XML dokumenty a lze je zpracovávat dostupnými nástroji. Ke všem zdrojům existují DTD popisy.

XML standard a DTD dokumenty nepopisují všechny požadavky, kterým musí data vyhovět, aby mohla být úspěšně zpracována. Některé vyplývají z omezení parseru, jiné z logiky věci.

### 3.2.1 Omezení parseru

Vestavěný parser není XML parser. Neimplementuje všechny vlastnosti popsané XML standardem. Zejména se jedná o následující zjednodušení.

- Parser podporuje pouze kódování znaků UTF-8.
- Nečte DTD (jednodušší umí přeskočit), důsledkem je, že nenahrazuje entity.
- Neprovádí normalizaci atributů.

### 3.2.2 Slovníček pozičních značek

Slovníček pozičních značek jednak definuje množinu přípustných pozičních značek a jednak jejich vzájemně jednoznačné zobrazení na zkratky. Z toho vyplývá, že každá značka (poziční i zkratka) se ve slovníčku vyskytuje pouze jednou.

**Příklad 3.2.1** Příklad seznamu pozičních značek

```
<?xml version="1.0"?>
<!DOCTYPE tagcatalog SYSTEM "tag.dtd">
<tagcatalog>
  ...
  <tagentry><tag>NNFS1-----@----</tag><shorthand>NFS1@</shorthand></tagentry>
  ...
</tagcatalog>
```

### 3.2.3 Slovníček ohýbacích vzorů

Slovníček ohýbacích vzorů definuje ohýbací vzory. Musí platit následující omezení.

- Jméno vzoru (hodnota atributu name elementu `patternentry`) je jednoznačné.
- Obsah elementu `tag` je poziční značka, která bude v době zpracování definovaná.

**Příklad 3.2.2** Příklad seznamu ohýbacích vzorů

```
<?xml version="1.0"?>
<!DOCTYPE patterncatalog SYSTEM "pattern.dtd">
<patterncatalog>
  ...
  <patternentry name="ccm" neg="no">
    <form>
      <ending>0</ending>
      <tag>Cv----------</tag>
    </form>
    <form>
      <ending>e</ending>
      <tag>Cv----------1</tag>
    </form>
  </patternentry>
  ...
</patterncatalog>
```

### 3.2.4 Slovníček derivačních vzorů

Slovníček derivačních vzorů definuje derivační vzory. Musí platit následující omezení.

- Jméno vzoru (hodnota atributu name elementu `derivationentry`) je jednoznačné.
- Obsah elementu `pattern` je jméno ohýbacího vzoru, které bude v době zpracování definované.
- Obsah elementů `derivation` a `refererderivation` je malé přirozené číslo volitelně prefixované písmenem `r`.

POZNÁMKA

Je-li obsah elementů `kofixending`, `lemmaending` a `refererlemmaending` roven `0`, je převeden na prázdný řetězec.

**Příklad 3.2.3** Příklad seznamu derivačních vzorů

```
<?xml version="1.0"?>
<!DOCTYPE derivationcatalog SYSTEM "derivation.dtd">
<derivationcatalog>
  ...
  <derivationentry name="cc">
    ...
    <rule createreferer="no">
      <kofixending>0</kofixending>
      <pattern>ccz</pattern>
      <derivation>0</derivation>
      <lemmaending>0</lemmaending>
      <refererderivation>0</refererderivation>
      <refererlemmaending>0</refererlemmaending>
    </rule>
    ...
    <rule createreferer="yes">
      <kofixending>krát</kofixending>
      <pattern>ccm</pattern>
      <derivation>r0</derivation>
      <lemmaending>krát</lemmaending>
      <refererderivation>0</refererderivation>
      <refererlemmaending>0</refererlemmaending>
    </rule>
  </derivationentry>
  ...
</derivationcatalog>
```

### 3.2.5 Morfologický slovník

Morfologický slovník je databáze, která koncentruje komputačně-lingvistické informace o morfologii slov. Lze z ní odvodit přes derivační a ohýbací vzory tvary českých slov s dodatečnými informaceni.

Struktura databáze byla přejata od zadavatele. Záznam odpovídá lemmatu a, podle typu, buď jeho konkrétnímu tvaru nebo pravidlu pro jeho expanzi derivačním vzorem. Jednotlivé záznamy týkající se jednoho lemmatu k sobě nejsou formálně sdruženy.

V databázi musejí být splněna následující omezení.

- Verze záznamu (hodnota atributu ver elementu `entry`), je-li uvedena, je celé číslo reprezentovatelné v číselném typu int.
- Obsah elementu `pat` je jméno derivačního vzoru, které bude v době zpracování definované.
- Oddělovač `<S/>` v seznamech `lstag`, `lslem`, `lssty`, `lssyn`, `lssem`, `lscom` a `lsder` odděluje neprázdné části; seznamy jsou neprázdné.
- Prvky seznamu `lstag` jsou poziční značky, které budou v době zpracování definované.
- Seznam `lslem` má délku nejvýše dva.

**Příklad 3.2.4** Příklad záznamů k lemmatu „coca“ v morfologickém slovníku

```
<?xml version="1.0"?>
<!DOCTYPE dictroot SYSTEM "dict.dtd">
<dictroot>
  ...
  <entry>
    <stat val="A"/>
    <rt>coca</rt>
    <pat>poml</pat>
    <lslem cnt="1">coca</lslem>
    <lssyn cnt="1">N</lssyn>
    <origfile>sup9703a</origfile>
    <created uid="JH">1105531573</created>
    <modified uid="JH">1105531573</modified>
  </entry>
  ...
  <entry>
    <stat val="A"/>
    <rt>coca</rt>
    <pat>0</pat>
    <lstag cnt="1">AXXX1A</lstag>
    <lslem cnt="1">coca</lslem>
    <lssyn cnt="1">A</lssyn>
    <lssem cnt="1">L</lssem>
    <lssty cnt="1">t</lssty>
    <origfile>supf001</origfile>
    <created uid="JH">1105531573</created>
    <modified uid="JH">1105531573</modified>
  </entry>
  ...
</dictroot>
```

První záznam k lemmatu přiřazuje vzor „poml“, druhý uvádí konkrétní tvar slova s morfologickou značkou.

- Seznam `lstag` smí být uveden jedině v kombinaci s některým ze vzorů „0“, „0n“, „0ns“, „zkr“ nebo „poml“. Všechny tyto vzory musí být vždy definovány. Je-li seznam uveden v kombinaci s jiným vzorem, je ignorován.

- Obsah elementů `created` a `modified` je přirozené číslo reprezentovatelné v typu type_t odpovídající UNIXovému času.

- Musí být splněna podmínka na strukturu slovníku.

POZNÁMKA

Atribut cnt elementů `lstag`, `lslem`, `lssty`, `lssyn`, `lssem`, `lscom` a `lsder` je ignorován.

## 3.3 Struktura slovníku

Seznam záznamů obohacujících morfologii lemmat je jeden možný pohled na slovník. Nabízí se i pohled jiný - pohled po lemmatech. Pro lemma lze sestavit seznam záznamů, které se jej týkají. S tím úzce souvisí interpretace seznamu `lslem` (připouští se délka buď jedna nebo dva). Ten říká, ke kterým lemmatům záznam patří.

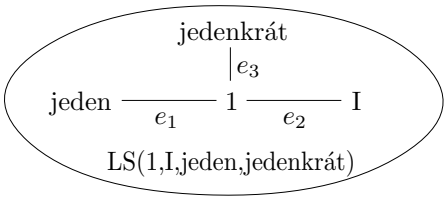
Pohled po lemmatech (slovníkových heslech) by měl přinést ucelený obraz o morfologii lemmatu. Předpokládá se, že společné lemma a alternativní lemma k sobě mají blízko, náleží do jednoho pohledu. Aby se s pohledem dalo rozumně pracovat, je stanoveno následující omezení formulované v termínech teorie grafů.

Definuji graf, jehož vrcholy jsou lemmata. Hrana mezi lemmaty vede právě tehdy, jsou-li v nějakém společném seznamu `lslem`. Většina vrcholů v grafu je izolovaných. Graf má mnoho komponent. Pro každou komponentu musí platit, že je stromem o n vrcholech, kde právě jeden vrchol má stupeň (n-1) a ostatních (n-1) vrcholů má stupeň 1. Komponentu si lze představit jako hvězdu. Centrální vrchol hvězdy označujeme společné lemma, listy stromu alternativní lemmata.

Pravidlo vychází z dat, které jsme měli k dispozici. Zajišťuje rozumné chování aktualizací slovníku. K pohledu na lemma patří záznamy tvořící komponentu. Graf byl sice definován na lemmatech, ale za uvedených podmínek je zobrazení na záznamy jasné.

Omezení si lze představit jako třídy ekvivalence na lemmatech. Vrcholy každé komponenty grafu tvoří jednu třídu. Propojení hranami pozbývá významu.

**Příklad 3.3.1** LS pro lemmata 1,I,jeden,jedenkrát




Lemma „1“ je společné všem záznamům. Lemmata „I“, „jeden“, „jedenkrát“ jsou alternativní.

## 3.4 Zpracování morfologického slovníku

### 3.4.1 Indexace

K rychlému zpracování slovníku je třeba provést jeho indexaci. K tomu slouží program **morfo-wib**. Některé programy si index postaví v paměti samy, není-li zadán.

**Parametry programu morfo-wib.** Program akceptuje všechny společné parametry. Navíc lze zadat následující parametry.

SPECIFICKÉ PARAMETRY PROGRAMU **MORFO-WIB**

***-b, --build*** Vynutí přestavění indexu i v případě, že systémová data primárního souboru a indexu napovídají, že je aktuální.

***--help*** Vypíše stručný přehled parametrů.

### 3.4.2 Interpretace záznamu

Záznam se interpretuje jako nějaká morfologická informace pro lemma. Z toho vychází uspořádání indexu - je organizován po lemmatech. Jsou možné dvě interpretace, podle toho, zda je přítomen seznam `lstag` (výčet tagů).

- Je-li seznam uveden, záznam říká, že pro lemmata ze seznamu `lslem` existuje tvar obsažený v elementu `rt`. Jeho morfologickou platnost vymezují značky ze seznamu `lstag`.
- Není-li seznam uveden, záznam říká, že se k lemmatům ze seznamu `lslem` mají z řetězce v elementu `rt` vygenerovat podle derivačního vzoru v elementu `pat` odvozené záznamy.

Záznamy tvary a možnosti odvození obohacují, nikdy neomezují.

### 3.4.3 Interpretace verzí

Záznamy v primárním souboru jsou volně řazeny za sebou. Nepočítá se s žádným odhadem velikosti. Za takových podmínek nelze efektivně realizovat aktualizace přímo na místě.

Problém je řešen verzováním záznamů. Verze záznamu je celé číslo uvedené v atributu ver elementu `entry`. Není-li atribut uveden, je nula.

Při běžném zpracování se uvažují pro dané lemma jen záznamy s verzí v absolutní hodnotě největší. Záporná verze má přednost před kladnou (-5 a 5). Nezáporná verze znamená přítomnost, záporná smazání záznamu.

Nezáleží na pořadí záznamů v souboru, zpracování probíhá podle verzí. To umožňuje snadno realizovat aktualizace jak v původním souboru, tak v odděleném souboru změn. Všechny záznamy konkrétní verze patřící k jednomu slovníkovému heslu musí být v jednom souboru.

Dodrží-li se XML struktura, mohou se soubory slučovat jednoduchým spojením za sebe.

### 3.4.4 Operace (Undo)

Základní funkcí entries je tvořit slovníkové heslo. Zápornými verzemi lze slovníkové heslo označit za smazané. Entry lze použít ještě jedním způzobem. Jedná se o možnost zneplatnění starších (v absolutní hodnotě nižších) verzí.

Funkci entry určuje atribut opetation. Implicitní hodnota (atribut není explicitně uveden) odpovídá základní funkci, tedy prezenci. Mazání se spouští zápornou verzí nebo uvedením atributu opetation s hodnotou delete. Operace zneplatnění se zadává hodnotou undo.

Je-li v databázi entry s operací undo, pohlíží se na starší záznamy (náležící k danému lemmatu), jako by v databázi nebyly (lze odčinit smazání).

**Příklad 3.4.1** Verzování

Příklad ukazuje, který záznam platí, vyskytují-li se v databázy entries s různými verzemi a operacemi. Poznamenejme, že na pořadí entries nezáleží.

| db1 | | db2 | | použitá |
|---|---|---|---|---|
| verze | operace | verze | operace | verze |
| 0 | present | | | db1-0 |
| | | 1 | present | db2-1 |
| | | 1 | delete | smazáno |
| | | -1 | undo | db1-0 |
| | | 2 | present | db2-2 |

### 3.4.5 Slévání, export

Editace databáze probíhá inkrementálním způsobem, změny se často ukládají do samostatného souboru představujícího jakousi podobu žurnálu. Čas od času může být žádoucí slít více souborů do jednoho a provést všechny naznačené operace. To umí (z prostředí příkazové řádky) program **morfo-export**. Funkci lze vyvolat také z XrefId[?editoru?].

**Parametry programu morfo-export.** Program akceptuje všechny společné parametry. Primární soubor ani index nehledá na zakompilovaných cestách ani na cestách zadaných v konfiguračním souboru. Navíc lze zadat následující parametry.

SPECIFICKÉ PARAMETRY PROGRAMU **MORFO-EXPORT**

***-o, --output-file* soubor** výstupní soubor (implicitně je nastaven standardní výstup).

***-f, --original-format* soubor** Zapíná export do původního formátu, nastavuje název souboru, do nějž budou uloženy záznamy, které nemají specifikový soubor. Parametr *-o* se stává povinným. Specifikuje adresář, v němž budou tvořeny výstupní soubory. Existující soubory budou přepsány.

***-r, --reset-versions*** Nuluje verzi vypisovaných entries.

***--help*** Vypíše stručný přehled parametrů.

**Zadání primárního souboru a indexu.** Parametry pro zadání primárního souboru a indexu se mohou opakovat. Index patří k poslednímu (z hlediska pořadí) zadanému primárnímu souboru. Není-li soubor indexu zadán, je index pro potřeby slévání postaven v paměti.

**Příklad 3.4.2** parametry programu **morfo-export**

```
morfo-export -P p1.xml -I p1.idx -P p2.xml -P p3.xml -I p3.idx -o merged.xml
```

Pro zpracování primárniho souboru `p1.xml` je použit index ze souboru `p1.idx`, index k souboru `p2.xml` je postaven. K souboru `p3.xml` je zadán index `p3.idx`. Export bude uložen do souboru `merged.xml`.

# Kapitola 4

# XGenerování

## 4.1 XGenerování

Slovník představuje kompaktní formu uložení seznamu slovních tvarů. Kompaktnosti je dosaženo využitím pravidelností slovotvorby a skloňování. Generování interpretuje záznamy ve slovníku a derivační a ohýhací vzory a tvoří tak slovní formy doplněné morfologickými informacemi.

Fáze slovotvorny se označuje jako derivování. Ta probíhá jako první. Následuje ohýbání.

### 4.1.1 Derivování

### 4.1.2 Ohýbání

## 4.2 Program morfo-inflex

Program morfo-inflex představuje rozhraní ke generování pro příkazovou řádku.

**Parametry programu morfo-inflex.** Program akceptuje všechny společné parametry. Primární soubor a index hledá na zakompilovaných cestách v případě, že nebyly zadány na příkazová řádce pomocí parametrů. Navíc lze zadat parametry popsaná níže.

**Schema paramertů.**

```
morfo-inflex [-P pfile [-I ifile]...] [-T tagfile] [-A patfile] [-D
    derfile] [--no-compiled-paths] [-o file] [-l file] [-a] [-n] [-s] [-b]
    [-e] [-t regexp] [-m] lemma...
```

SPECIFICKÉ PARAMETRY PROGRAMU **MORFO-INFLEX**

***-o*, *--output-file* soubor** výstupní soubor (implicitně je nastaven standardní výstup).

***-l*, *--lemma-list* soubor** soubor se seznamem lemmat ke generování (co řádek, to lemma). Čtení standardního vstupu se zadává pomlčkou −.

***-a*, *--all-dict*** zapíná rozgenerování celého slovníku.

***-n*, *--no-negation*** vypíná odvozování negetivních forem (předpona ne-).

***-s*, *--no-superlatives*** vypíná odvozování superlativů (předpona nej-).

***-b, --basic*** omezuje zpracování pouze na lemmata shodná se zadaným. Jiná lemmata získaná derivováním jsou potlačena.

***-e, --no-expand*** zapíná úsporný výpis kombinující předpony nej- a ne- (jsou-li ohýbacím vzorem předepsány) a výstup značek se zástupnými symboly pro stupňování (#) a negaci (@).

***-t, --filter-tag*** sám nevím

***-m, --silent-missing*** vypíná hlášení o zadaných lemmatech, která nebyla nalezena ve slovníku.

***--help*** Vypíše stručný přehled parametrů.

**Zadání primárního souboru a indexu.** Parametry pro zadání primárního souboru a indexu se mohou opakovat. Index patří k poslednímu (z hlediska pořadí zleva doprava) zadanému primárnímu souboru. Není-li soubor indexu zadán, je index pro potřeby slévání postaven v paměti.

Není-li zadán žádný primární soubor, hledá se v konfiguračním souboru případně zakompilovaných cestách (nebyly-li deaktivovány parametrem `--no-compiled-paths`). K takto (implicitně) zadanému primárnímu souboru nelze zadat soubor indexu pomocí parametrů příkazová řádky (může však být zadám implicitně v konfiguračním souboru nebo zakompilovaných cestách).

**Příklad 4.2.1** parametry programu **morfo-inflex**

```
morfo-inflex -P p1.xml -I p1.idx -P p2.xml -P p3.xml -I p3.idx \
        -o forms -l lemma-list kousek
```

Pro zpracování primárniho souboru `p1.xml` je použit index ze souboru `p1.idx`, index k souboru `p2.xml` je postaven. K souboru `p3.xml` je zadán index `p3.idx`. Export bude uložen do souboru `forms`. Vstupem jsou lemmata ze souboru lemma-list a lemma „kousek“.

# Dodatek A

# DTD používaných dokumentů

**DTD seznamu značek.**

```
  <!ELEMENT tagcatalog     (tagentry)+>
<!ELEMENT tagentry      (tag, shorthand)>
<!ELEMENT tag         (#PCDATA)>
<!ELEMENT shorthand     (#PCDATA)>
```

**DTD seznamu ohýbacích vzorů.**

```
  <!ELEMENT patterncatalog (patternentry)+>
<!ELEMENT patternentry     (form)+>
<!ELEMENT form        (ending, tag+)>
<!ELEMENT ending      (#PCDATA)>
<!ELEMENT tag         (#PCDATA)>

<!ATTLIST patternentry     name     CDATA    #REQUIRED
            neg       (yes|no) #REQUIRED>
```

**DTD seznamu derivačních vzorů.**

```
  <!ELEMENT derivationcatalog (derivationentry)+>
<!ELEMENT derivationentry  (rule)+>
<!ELEMENT rule        (kofixending, pattern, derivation, lemmaending,
             refererderivation, refererlemmaending)>
<!ELEMENT kofixending      (#PCDATA)>
<!ELEMENT pattern     (#PCDATA)>
<!ELEMENT derivation     (#PCDATA)>
<!ELEMENT lemmaending      (#PCDATA)>
<!ELEMENT refererderivation   (#PCDATA)>
<!ELEMENT refererlemmaending  (#PCDATA)>

<!ATTLIST derivationentry  name     CDATA    #REQUIRED>
<!ATTLIST rule        createreferer  (yes|no) #REQUIRED>
```

**DTD morfologického slovníku.**

```
  <!ENTITY  % list      "#PCDATA | S">
<!ENTITY  % list_attr      "cnt     CDATA    #IMPLIED">

<!ELEMENT dictroot      (entry)*>
<!ELEMENT entry         (stat, rt, pat, lstag?, lslem,
            lsder?, lssyn?, lssem?, lssty?, lscom?,
            origfile, created, modified?)+>
<!ELEMENT stat       EMPTY>
<!ELEMENT rt         (#PCDATA)>
<!ELEMENT pat        (#PCDATA)>
<!ELEMENT origfile      (#PCDATA)>
<!ELEMENT created    (#PCDATA)>
<!ELEMENT modified      (#PCDATA)>
<!ELEMENT lstag         (%list;)*>
<!ELEMENT lslem         (%list;)*>
<!ELEMENT lssty         (%list;)*>
<!ELEMENT lssyn         (%list;)*>
<!ELEMENT lssem         (%list;)*>
<!ELEMENT lscom         (%list;)*>
<!ELEMENT lsder         (%list;)*>

<!ELEMENT S       EMPTY>
<!ATTLIST entry         ver      CDATA    #IMPLIED
           operation   (delete|undo)  #IMPLIED>
<!ATTLIST stat       val      (A)      #REQUIRED>
<!ATTLIST created    uid      CDATA    #REQUIRED>
<!ATTLIST modified      uid      CDATA    #REQUIRED>
<!ATTLIST lstag         %list_attr;>
<!ATTLIST lslem         %list_attr;>
<!ATTLIST lssty         %list_attr;>
<!ATTLIST lssyn         %list_attr;>
<!ATTLIST lssem         %list_attr;>
<!ATTLIST lscom         %list_attr;>
<!ATTLIST lsder         %list_attr;>
```

# Doporučená literatura


[1] *Manual for Morphological Annotation*, Jiří Hana a Hana Hanová, CKL Technical Report TR-2002-14, Charles University, Czech Republic , 2002, dostupné v HTML <http://ufal.mff.cuni.cz/pdt/Corpora/PDT_1.0/References/mman.html>.

[2] *Disambiguation of Rich Inflection*, Jan Hajič, Karolinum, Prague, Czech Republic , 2001.

[3] *Manuál ke starší verzi morfologického editoru (MCLASS)*, Jan Hajič, dostupné na instalačním cd, v souboru doc/mclass.il2.

[4] *Finite State Morphology*, Kenneth R. Beesley a Lauri Karttunen, CSLI Publications, 2003, dostupné v HTML <http://www.fsmbook.com>.

[5] *Algorithms and Data Structures Research and Reference Material*, L. Allison, Monash University, Australia , dostupné v HTML <http://www.csse.monash.edu.au/~lloyd/tildeAlgDS/>.