Programátorská dokumentace ke slovníkovému editoru MorfoEd

David Kolovratník a Leoš Přikryl

2. června 2008

Obsah

Kapitola 1

Editor

1.1 Hádání vzorů

Při zadávání nového slova do slovníku je potřeba určit jeho vzor. Pro zjednodušení práce se program nejdříve pokusí vzor uhodnout.

Po zadání základního tvaru slova se zobrazí nabídka několika vzorů, ke kterým by slovo mohlo patřit. Mezi nabízenými vzory se většinou vyskytuje i ten správný. Hádání samozřejmě není stoprocentně úspěšné (podle testů ale určí správný vzor s více než 96% pravděpodobností).

1.1.1 Jak to funguje?

Program se snaží uhodnout vzory na základě koncovky základního tvaru slova. Celý proces je založený na tom, že již existuje poměrně velký slovník slov s přiřazenými vzory. Hesla z tohoto slovníku jsou použita jako trénovací a testovací data. Editor ve standardní instalaci obsahuje již natrénovaný algoritmus pro hádání vzorů. Většině uživatelů bude pravděpodobně plně vyhovat. Pokud byste chtěli hádání znovu natrénovat (třeba na lepších datech), čtěte dále. Pokud vám dostačuje standardně natrénované hádání vzorů, můžete zbytek kapitoly přeskočit.

1.1.2 Příprava dat

Všechny programy pro trénování hadání vzorů se nacházejí v adresáři `slovnik/guesser`.

Trénovací a testovací data z již existujícího slovníku vytvoříte programem `slovnik/guesser/preparedata.pl`. Jeho použití je následující:

`preparedata.pl` *`poměr train test`*

poměr poměr testovací : trénovací data v % (např. 20 znamená, že 20% dat budou testovací a zbylých 80% trénovací)

train soubor pro trénovací data

test soubor pro testovací data

Program načte data z primárního souboru, který je standardně používán editorem (dáno souborem `.morforc`). Potom je rozdělí v daném poměru a to tak, že lemmata seřadí podle abecedy a pak vždy několik prvních slov vloží do trénovacích dat, dalších několik do testovacích, pak zase několik do trénovacích, atd. (Přičemž „několik“ je voleno tak, aby byl zachován příslušný poměr mezi množstvím trénovacích a testovacích dat). Tak je zajištěna rovnoměrnost rozdělení dat.

Do trénovacích ani testovacích dat nejsou vkládány výjimky, tj. slova se vzory, které jsou uvedeny v seznamu výjimek (soubor uvedený v `.morforc`). Tento seznam můžete libovolně měnit. Standardně jsou v něm vzory 0, zkr, 0abbr, poml, 0n, 0ns. V souboru je na každém řádku uveden jeden vzor, patřící mezi výjimky.

1.1.3 Vytvoření tabulky pro hádání vzorů

Základem pro hádání vzorů je tabulka vzorů, ve které jsou obsaženy možné vzory pro dané koncovky základního tvaru slova. Tato tabulka je sestavena na základě trénovacích dat. Vzhledem k tomu, že trénovací data obsahují chyby, není možné jednoduše vzít koncovky všech slov s daným vzorem a přiřadit je k tomuto vzoru. Také není jasné, jak dlouhou koncovku použít.

Program pro vytvoření tabulky se používá následovně:

`createtable.pl [-d` *`hloubka`*`] [-p` *`počet`*`] [-l` *`poměr`*`] [-f` *`soubor`*`]` *`train`*

hloubka maximální hloubka rekurze (tj. maximální délka koncovky použité při hádání)

[celé číslo, defaultní hodnota 5]

počet při kolika zbývajících vzorech prohlásit koncovku za definitivní (a už nezkoumat delší koncovky se stejným koncem)

[celé číslo, defaultní hodnota 1]

poměr poměr (v %) počtu výskytů daného vzoru oproti součtu výskytů všech vzorů s touto koncovkou. Pokud bude vypočtený poměr menší než tato zadaná konstanta, bude ignorován (odfiltrují se tím řídké případy a chyby)

[reálné číslo, defaultní hodnota 0.03]

soubor soubor pro výsledky

[pokud není zadán, tisknou se výsledky na standardní výstup]

train soubor s trénovacími daty (připravený programem `preparedata.pl`)

Program funguje tak, že postupně vkládá slova pozpátku do trie. V každém vrcholu trie se kromě příslušného písmene ukládájí všechny vzory slov, které mají odpovídající koncovku (odpovídající cestě z kořene do tohoto vrcholu). U každého vzoru je také počet slov s odpovídající koncovkou, která k tomuto vzoru patří.

Po vložení všech slov se trie prořezává. Tento proces má za úkol eliminovat chyby v trénovacích datech a některé velmi řídké případy. V každém uzlu jsou vymazány vzory, které se vyskytují u méně než daného počtu slov. Tento počet je určen jako procentuální část z celkového počtu slov obsažených v tomto uzlu (tzn. slov, které mají koncovku odpovídající cestě z kořene do tohoto vrcholu). Použitý procentuální poměr lze nastavit pomocí parametru *`-l`*.

Délku koncovek použitých při hádání ovlivňují dva parametry. Přímo ji ovlivňuje parametr *`-d`*. Ten určuje maximální délku koncovky použité pro hádání. Žádná použitá koncovka tedy nebude delší než hodnota tohoto parametru. Dále ji nepřímo ovlivňuje parametr *`-p`*. Ten říká, že pokud pro nějakou kratší koncovku existuje už jenom p nebo méně možných vzorů, použije se pro hádání tato kratší koncovka a delší už se nezkoumají. Koncovky použité pro hádání tedy nemusí mít všechny stejnou délku.

1.1.4 Měření úspěšnosti

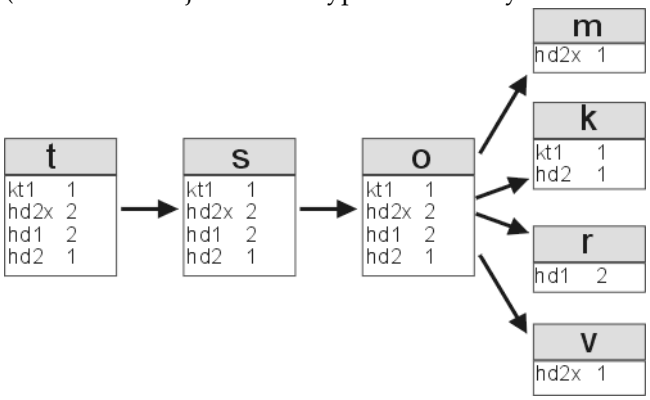
Úspěšnost hádání při použití určité tabulky vzorů lze otestovat programem `tester.pl`. Jeho použití je následující:

Příklad 1.1.1 Příklad

Použité parametry -d 5 -p 4 -l 0.1
Vkládaná slova: kost (vzor kt1), most (hd2x), porost (hd1), samorost (hd1), kakost (hd2), chvost (hd2x)
Po vložení těchto slov do trie, bude tabulka vzorů ve vrcholu odpovídajícím koncovce -ost vypadat následovně:

vzor	počet
kt1	1
hd2x	2
hd1	2
hd2	1

(Poznámka: Stejně budou vypadat i tabulky ve vrcholech pro koncovky -st a -t)



Nyní zkusíme v tomto vrcholu prořezávat. Celkový počet výskytů vzorů v tomto vrcholu je 1+2+2+1=6. Zahodit bychom tedy měli vzory, které se vyskytují v méně než 0.1/100*6=0.006 případech (dělení 100 je tam kvůli tomu, že parametr -l je udávaný v procentech). Žádný takový vzor se v tomto vrcholu nevyskytuje.
Spočítáme počet různých vzorů v tomto vrcholu. Dostáváme 4, což je rovno parametru -p. Takže do tabulky vzorů přidáme záznam „ost kt1 hd2x hd1 hd2“ a delší koncovky (jako například -kost, -vost, -orost,...) už nezkoumáme a do tabulky nezařazujeme, i když jsou kratší nebo stejně dlouhé jako určený limit `-d`.

`tester.pl [-b chyby] tabulka test`

chyby soubor pro uložení slov, která byla určena špatně

[defaultní hodnota `/dev/null`]

tabulka soubor s tabulkou vzorů (dvojice <koncovka, seznam možných vzorů>)

test soubor s testovacími daty (vytvořený programem `preparedata.pl`)

Výsledkem programu jsou dvě čísla. Jedno udává procentuální úspěšnost, tj. v kolika případech byl správný vzor mezi nabízenými. Druhé je průměrný počet nabízených vzorů. Toto číslo je také velice důležité, protože ideální je nabízet uživateli co nejméně nesprávných vzorů a šetřit tak jeho čas při prohlížení nabízených tvarů. Hledáme takovou tabulku vzorů, která má co největší úspěšnost a co nejmenší počet nabízených vzorů. Tyto dva požadavky jdou proti sobě (pokud by nabízený počet vzorů byl maximální, je úspěšnost automaticky 100% a naopak), takže výsledkem je kompromis.

Testovací program může ukládat slova, která byla uhodnuta chybně (mezi uhodnutými vzory nebyl ten správný). Je pravděpodobné, že některá takto špatně určená slova nejsou ve skutečnosti určená špatně, ale jedná se o chybu v trénovacích datech a tím pádem i ve slovníku (ke slovu je přiřazen nesprávný vzor). Stojí tedy určitě za pokus projít tento seznam a ručně ho zkontrolovat.

1.1.5 Hledání optimálních hodnot parametrů

Při trénování defaultní tabulky pro hádání byly použity následující parametry: -d 5 -p 4 -l 0.06.

Jejich hodnoty byly nalezeny experimentálně, postupným trénováním s různými parametry a porovnáváním výsledků. Vybrané parametry jsou kompromisem mezi procentuální úspěšností hádání a počtem nabízených vzorů.

K automatickému generování tabulek a jejich následnému testování lze použít program `automatic.pl`.

`automatic.pl [-d` *`mind-maxd`*`] [-p` *`minp-maxp`*`] [-l` *`minl-maxl`*`] [-s` *`krok`*`]` *`train test výsledky`*

mind-maxd čísla omezující maximální hloubku rekurze (tj. maximální délka koncovky použité při hádání)

[celá čísla, defaultní hodnota 3-5]

minp-maxp při kolika zbývajících vzorech prohlásit koncovku za definitivní (a už nezkoumat delší koncovky se stejným koncem)

[celá čísla, defaultní hodnota 1-4]

minl-maxl poměr (v %) počtu výskytů daného vzoru oproti součtu výskytů všech vzorů s touto koncovkou. Pokud bude vypočtený poměr menší než tato zadaná konstanta, bude ignorován (odfiltrují se tím řídké případy a chyby)

[reálná čísla, defaultní hodnota 0-0.1]

krok krok, po kterém se mění procentuální poměr

[reálné číslo, defaultní 0.01]

train soubor s trénovacími daty (vytvořený programem `preparedata.pl`)

test soubor s testovacími daty (vytvořený programem `preparedata.pl`)

results soubor pro zapsání výsledků v HTML formátu (pokud existuje bude přepsán!)

Tento program postupně generuje různé tabulky vzorů podle zadaných parametrů a pro každou spustí `tester.pl`. Výsledkem je tabulka (v HTML) s výsledky programu `tester.pl` pro jednotlivé hodnoty parametrů. Parametry probíhají od zadaného minima po maximum a vyzkouší se všechny kombinace v daných mezích. Vyhodnocení, která tabulka je nejlepší, je nutno provádět manuálně, protože, jak píšu výše, je potřeba nalézt kompromis mezi úspěšností a počtem nabízených vzorů.

1.2 Rozhraní Perl/C

1.2.1 Komunikace mezi C a Perlem

Vzhledem k tomu, že vstupní a výstupní funkce jsou naprogramovány v C a grafické rozhraní editoru pomocí Perl Tk, je potřeba komunikovat mezi jazykem C a Perlem. K tomu je použit program SWIG <http://www.swig.org>.

Hlavním souborem pro tento program je `slovnik/editor/MorfoIOWrap_swig.i`. Ten obsahuje hlavičky C funkcí a definice struktur, které jsou používány v Perlu. Podle tohoto souboru vytvoří SWIG perlovský modul (`slovnik/lib/MorfoIOWrap.pm`) a sdílenou C knihovnu (`slovnik/lib/MorfoIOWrap.so`), které slouží ke komunikaci mezi Perlem a C.

Perlovský modul vytvoříme příkazem:

swig -perl5 MorfoIOWrap_swig.i

spouštěným v adresáři `slovnik/editor/`. Tento příkaz zároveň vytvoří zdrojový soubor `MorfoIOWrap_swig_wrap.c`, který tvoří základ pro sdílenou knihovnu. Překlad řídí Makefile vytvořený pomocí perlovské knihovny `ExtUtils::MakeMaker`.

Všechny výše uvedené úkony včetně překladu za vás provede Makefile.

Druhým směrem (tj. použití perlovské funkce v C) komunikace neprobíhá.

1.2.2 MorfoIO.pm

Funkce uvedené v souboru `MorfoIOWrap_swig.i` lze již v Perlu přímo používat (zavolají se jako `MorfoIOWrap::název_funkce(parametry)`). Návratové hodnoty ale nejsou úplně bez problémů použitelné perlovské proměnné, takže kromě automaticky vygenerovaného obalujícího modul jsem přidal ještě druhý modul `MorfoIO.pm`. Všechny požadavky z editoru jdou přes tento modul, žádný modul nevyužívá přímo `MorfoIOWrap.pm`. Tento modul se postará o všechny potřebné vstupní a výstupní operace. Jedinou výjimkou je modul `MorfoConst.pm`, který využívá přímo funkcí z `MorfoIOWrap.pm` k načítání parametrů.

1.3 Perlovské moduly

Všechny perlovské moduly jsou v adresáři `slovnik/lib`. Následuje stručný popis jednotlivých modulů. Podrobnější popis jednotlivých modulů a funkcí naleznete v XrefId[?dokumentaci NaturalDocs?].

1.3.1 MorfoIOConst.pm

Tento modul se stará o načítání parametrů ze souborů `.morforc` a `.morfoinfo`. Hned po spuštění ukládá všechny parametry do globálních promměných, které pak lze kdykoliv použít. Při ukončení editoru uloží všechny změněné parametry ze souboru `.morfoinfo` zpět do tohoto souboru. Informace o názvech a významu globálních proměnných a o jejich umístění v souborech `.morforc` a `.morfoinfo` naleznete v popisu tohoto modulu v XrefId[?dokumentaci NaturalDocs?].

1.3.2 TagEntry.pm

Obsahuje funkce pro vytvoření speciální komponenty pro zobrazení a editaci tagu. Ke správnému fungování potřebuje soubor se seznamem a popisem všech možných hodnot na jednotlivých pozicích v tagu. Ten je standardně v `data/tag_positions`. Tento soubor musí obsahovat popis všech 15 pozic tagu. Popis jedné pozice vypada takto:

```
:Pozice číslo - Popis pozice
znak význam
```

Dvojtečka před slovem „Pozice“ je důležitá a musí být na začátku řádku. Číslo je pořadové číslo pozice od 1 do 15 psáno číslicemi. Pomlčka před popisem může být z obou stran oddělena libovolným počtem mezer, které jsou při načítání ignorovány. Popis pozice končí koncem řádku.

Na dalších řádcích pak následuje vždy znak, libovolný počet mezer, které jsou ignorovány, a význam tohoto znaku na této pozici v tagu. Popis končí koncem řádku.

Prázdné řádky jsou ignorovány, soubor neumožňuje žádné komentáře.

1.3.3 MorfoBasics.pm

Tento modul obsahuje elementární funkce používané ve všech modulech editoru. Více viz popis tohoto modulu v XrefId[?dokumentaci NaturalDocs?].

1.3.4 SearchListbox.pm

Modul, umožňující svázat `Tk::Listbox` a `Tk::Entry` a přidat k listboxu inkrementální vyhledávání na základě obsahu entry. Při inkrementálním vyhledávání používá porovnávací funkci `MorfoIO::cmpWithDash`, takže data v listboxu musí být touto funkcí setříděna.

1.3.5 MorfoIO.pm

Soubor starající se o vstupy a výstupy. Jeho hlavním úkolem je obalit funkce z MorfoIOWrap.pm tak, aby šly použít v editoru. Kdyby měl být editor v budoucnu předělán na jiný formát slovníku, byl by tento soubor to hlavní, co by bylo potřeba změnit.

1.3.6 MorfoIOWrap.pm

Tento modul je automaticky vygenerován pomocí programu SWIG a slouží pro komunikaci mezi Perlem a C.

1.3.7 MorfoGuesser.pm

Obsahuje všechny funkce pro hádání vzorů. Ke své práci potřebuje natrénovanou tabulku vzorů. O jejím vytvoření viz hádání vzorů.

1.4 Popis datových souborů

Popis většiny datových souborů, nutných ke spuštění editoru najdete v Uživatelské dokumentaci, části XrefId[?Formát datových zdrojů?]. Soubory jsou defaultně umístěné v adresáři, kam byla instalována data.

Soubory s příznaky (syntaktickými, sémantickými a stylovými) mají následující formát:

- Na každém řádku je popis jedné možné hodnoty.
- Řádek začíná možnou hodnotu (jeden znak), následuje libovolný počet mezer a pak až do konce řádku popis významu dané hodnoty.
- Prázdné řádky jsou ignorovány, řádky začínající znakem # jsou považovány za komentáře a také ignorovány.

Defaultní názvy souborů jsou `syntactic_flags`, `semantic_flags` a `style_flags`.

Soubor výjimek je popsán v části Hádání vzorů.

Soubor s popisem jednotlivých pozic v tagu je popsán v části TagEntry.pm.

Index primárního souboru je vytvářen automaticky a je to binární soubor.

Soubor s tabulkou pro hádání vzorů je také vytvářen automaticky. Na každém jeho řádku je konec slova, mezera a pak mezerami oddělený seznam všech možných derivačních vzorů, které k tomuto konci slova mohou příslušet (podle natrénování hádání vzorů).