

Specyfikacja implementacyjna „gentex”

Maciej Skarbek
Nr albumu: 271088

16 marca 2015

Wprowadzenie

Celem projektu jest stworzenie aplikacji w języku C, która będzie generować teksty wyjściowe na podstawie analizy innych tekstów wykorzystując przy tym łańcuchy Markova. Przy tworzeniu programu korzystać będziemy z systemu kontroli wersji git.

Opis modułów

managment

Rozpoznaje argumenty wywołań i steruje innymi modułami. Zawiera funkcje "main".

store

Przechowuje macierz przejść z podziałem na prefiksy i sufiksy. Implementacja drzewa.

generation

Generuje tekst wynikowy i zapisuje go do pliku.

reading

Odczytuje teksty podane przez użytkownika. Analizuje je dzieląc na prefiksy i sufiksy a następnie przekazuje do modułu "store".

backup

Tworzy i umożliwia wczytanie plików pośrednich z których w przyszłości można generować teksty wynikowe.

stats

Tworzy statystyki łączone dla tekstów bazowych i oddzielną statystykę dla tekstu wynikowego (prawdopodobieństwo wystąpienia pojedynczego słowa, n-gramu z jakich został wygenerowany tekst i wskaźnik PMI).

error

Obsługuje błędy zaistniałe podczas działania.

Opis najważniejszych funkcji i struktur

managment

- void add(char **prefix, char * suffix)
Rozpoznanie argumentów wywołania i przekazanie sterowania do odpowiednich modułów.

store

- Struktury

<i>typedef struct{</i>	<i>typedef structnode{</i>	<i>typedef struct{</i>
<i>char **prefix;</i>	<i>ngram * g;</i>	<i>tree_tt;</i>
<i>char **suffix;</i>	<i>structnode * left,*right;</i>	<i>intnumber_gram;</i>
<i>intsize_s;</i>	<i>}node_t,*tree_t;</i>	<i>intsize;</i>
<i>intn_s;</i>		<i>ngram * *tab;</i>
<i>}ngram;</i>		<i>intn_s_max;</i>
		<i>}store;</i>

Tworzone jest drzewo do przechowywania prefiksów i sufiksów jak również tablica w której będą wskaźniki do komurek drzewa. Tablica będzie używana przy losowaniu prefiksów jak również przy tworzeniu pliku pośredniego (gdybyśmy chcieli stworzyć plik pośredni z drzewa był by on posortowany i przy wczytywaniu zamiast drzewo powstała by nam lista co znacząco wydłużyło by czas pracy programu).

- tree_t insert(tree_t t, char **prefix, char * suffix)
Wstawia do drzewa prefiksy i sufiksy, jeśli prefiks już istnieje to dopisuje tylko sufiks. Dodaje również wskaźnik na "ngram" do "tab".
- void add_from_backup(char **prefix, char **suffix, int n_s)
Dodaje prefiks i całą listę sufiksów do drzewa.
- ngram* rand_prefix()
Losuje prefiks.
- char* rand_suffix(char** prefix)
Losuje sufiks dla podanego prefiksu.

generation

- void generation()
Generuje tekst wynikowy i zapisuje go do pliku.

reading

- void reading(char * name_file)
Czyta podany plik, dzieli na prefiksy i sufiksy a następnie przekazuje do "store" i "stat".

backup

- void backup()
Tworzy plik wczytuje i tworzy plik pośredni.

stats

- void stat_add_word(char * word)
Dodaje pojedynczy wyraz do drzewa ze statystykami (zlicza ilość wystąpień).
- void stat_add_ngram(char** prefix, char * suffix)
Dodaje n-gram do drzewa ze statystykami i zlicza ilość wystąpień.
- double get_probability(tree_stat t, char * word)
Liczy prawdopodobieństwo dla n-gramu.
- long double get_pmi(char* wngam)
Liczy wskaźnik PMI dla n-gramu.
- void write_stat(char * name_file_stat)
Zapisuje statystyki do podanego pliku.

error

- void fatal(int err, const char *msg)
Dostaje wyrażenie logiczne "err" i w przypadku potwierdzenia wystąpienia błędu pokazuje wiadomość "msg" i np. zamyka program.

Testowanie

Użyte narzędzia

time

Polecenie time do pomiaru czasu działania programu aby wybrać optymalny algorytm jaki zaimplementujemy.

valgrind

Narzędzie do debugowania pamięci i wykrywania wycieków pamięci.

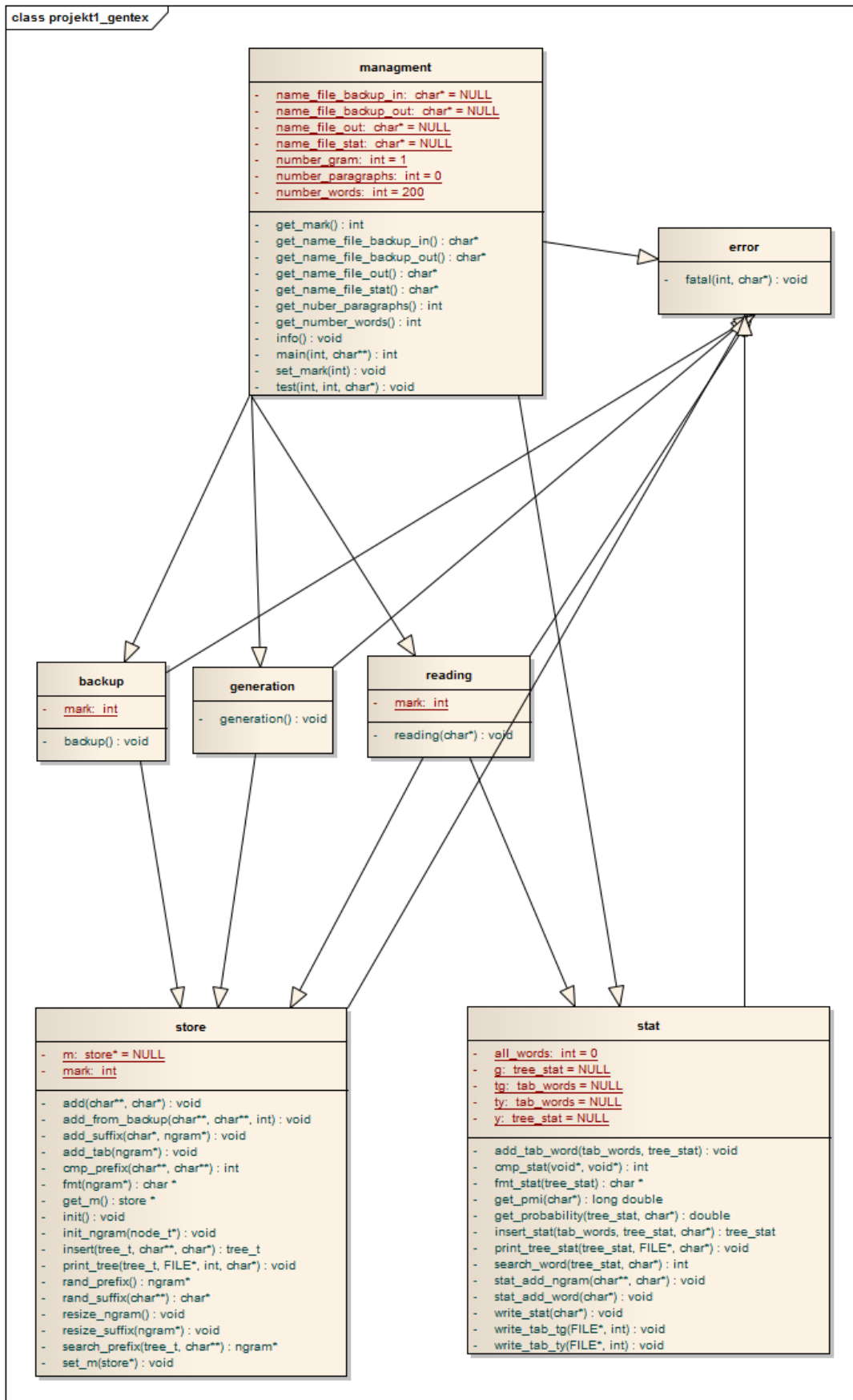
Sposób testów

Testy będą wykonywane dla pojedynczych funkcjonalności programu, dla całych modułów a następnie po przyłączeniu kolejnego modułu zostaną wykonane kompleksowe testy całego programu. Szczególną uwagę należy zwrócić na punkty krytyczne w których możemy spodziewać się błędów.

Punkty krytyczne

- Bardzo duże pliki wejściowe
- Puste pliki wejściowe
- Bardzo małe pliki wejściowe (np 2 słowa)
- Błędne wywołanie programu
- Podanie pliku do zapisu który już istnieje
- Mała ilość pamięci urządzenia na którym zostanie uruchomiony program
- Wycieki pamięci

Diagram modułów



Rysunek 1: