Projekt für die Veranstaltung Text-Indexierung im WS 2024/25

Ziel des Projektes ist die Implementierung mehrerer Trie-Varianten. In der Vorlesung sind fünf verschiedene Varianten vorgestellt worden: Arrays variabler Größe, Arrays fester Größe, Hashtabellen, balancierte Suchbäume, gewichtsbalancierte Suchbäume und 2 Level mit Arrays fester Größe und gewichtsbalancierten Suchbäumen.

Aufgabe ist es mindestens drei der Trie-Varianten zu implementieren. Hierbei dürfen für die Hashtabelle und die (gewichts)balancierten Bäume externe Bibliotheken verwendet werden.

Ein- und Ausgabe

Das Programm muss per Kommandozeile steuerbar sein. Die Eingabe hierfür hat das folgende Format. Bitte achten Sie darauf, dass der Name der auszuführenden Datei korrekt ist.

```
ti_programm -variante=<1-3> eingabe_datei query_datei
```

Bei der Eingabe-Datei handelt es sich immer um eine Liste von Wörter über einem Byte-Alphabet. Ein einzelnes Zeichen ist somit immer ein Byte groß. In jeder Zeile kommt exakt ein Wort vor. Wörter können aus beliebigen Zeichen a-Z, A-Z und 0-9 bestehen und enden immer auf ein 0-Byte.

Beispiel: (Das 0-Byte wird durch ein $ symbolisiert.)
```
battery$
car$
train$
gas$
```

Bei der Query-Datei handelt es sich ebenfalls um eine Liste von Wörtern. Diese Wörter erfüllen die gleichen Eigenschaften, wie die Wörter in der Eingabe-Datei. Jede Zeile ist zudem gefolgt vom Typ des Queries: `c`ontains, `d`elete, `i`nsert. Jedes Query gibt `true` oder `false` zurück, abhängig davon, ob die Operation erfolgreich war (delete/insert) oder ob das Wort im Trie enthalten ist (contains). Die Ergebnisse der Anfragen werden zeilenweise in eine Datei mit dem Namen `result_<eingabe_datei>.txt` geschrieben.

Beispiel:
```
carbon$ i
car$ c
car$ d
car$ c
apple$ d
```
Dieses Beispiel sollte (mit der Beispieleingabe oben) die Ausgabe `true, true, true, false, false` erzeugen. Aus Platzgründen ist die Ausgabe in diesem Beispiel in eine Zeile geschrieben worden.

Darüber hinaus gibt das Programm noch folgende Ausgabe per Kommandozeile aus (Leerzeichen sind nur zwischen Parametern erlaubt):
```
RESULT name=<your-name> trie_variant=<variant name> trie_construction_time=<construction time (ms)>
trie_construction_memory=<memory peak (MiB)> query_time=<query time (ms)>
```

Die Laufzeiten für die Queries beziehen sich auf die Gesamtlaufzeit. Die Zeitmessungen beginnen jeweils, wenn die Eingabedateien in den Speicher geladen worden sind und Enden, sobald die letzte Operation für die Konstruktion/Anfragen abgeschlossen wurde.

Minimalanforderungen

Das Projekt darf in C, C++, Rust, oder Python umgesetzt werden. Dem Projekt sollte eine detaillierte Anleitung beiliegen, die beschreibt, was zum kompilieren/ausführen benötigt wird und wie genau das Programm ausgeführt werden kann. Das Programm muss das oben beschrieben Ein- und Ausgabeformat unterstützen.

Wichtig: **Es dürfen keine externen Bibliotheken verwendet werden**, die nicht von Florian Kurpicz genehmigt wurden. Bitte per Mail (`kurpicz@kit.edu`) nachfragen, bevor externe Bibliotheken eingebunden werden. Die Standardbibliothek der jeweiligen Programmiersprache kann allerdings ohne Fragen verwendet werden.

Dokumentation, Evaluation und Präsentation

Der Code muss so dokumentiert sein, dass dem Leser klar wird, was an welcher Stelle was genau passiert. Hierfür sollte darauf geachtet werden, dass die Dokumentation auch erklärt *warum* etwas gemacht wird und nicht nur *was* gemacht wird.

Die Evaluation ist Teil der Präsentation. Hier können unter anderem die Laufzeiten des eigenen Ansatzes für unterschiedliche Eingabe gezeigt werden. Ein Vergleich mit anderen Implementierungen ist auch möglich. (Hinweis: Das Ausgabeformat kann direkt zum Erstellen von Diagrammen genutzt werden, siehe `https://github.com/bingmann/sqlplot-tools/`.)

Die Ergebnisse sollen dann in einer ca. fünfminütigen Präsentation vorgestellt werden. In der Präsentation soll neben der Evaluation der Implementierung darauf eingegangen werden, was implementiert wurde, wie es implementiert wurde und was eventuell besonders an der Implementierung ist.

Wettbewerb

Des Weiteren gibt es noch einen kleinen Wettbewerb, an dem jedes eingereichte Projekt automatisch teilnimmt. Das Abschneiden im Wettbewerb hat keinen Einfluss auf die Note des Projekts! Bei dem Wettbewerb werden alle Laufzeiten und der Speicherplatzbedarf gewichtet bepunktet. Die Gewichtung ist: 25 % Konstruktionszeit, 50 % Queryzeit und 25 % Speicherplatz.

Deadline

Das Projekt muss bis zum 27.01.2024 um 23:59 Uhr deutscher Zeit per Mail an `kurpicz@kit.edu` gesendet werden (gerne als Link zu einem Repository). Spätere Abgaben können leider nicht berücksichtigt werden. Die Vorträge werden am 03.02.2024 stattfinden.