アプリケーション説明書

提出日：2021 年 12 月 24 日

日本大学 文理学部 情報科学科
5419100 宮田章裕

1. アプリケーション名

Document Analyzer

2. 想定シーン

大量のドキュメントをダウンロードしながら，並行して，ダウンロード済みのドキュメントを分析するシーン。

3. 並行して行う処理とその理由

3.1. ダウンロード処理と分析処理

ドキュメントのダウンロードする処理と，ダウンロード済みのドキュメントの分析処理を並行して行う。なぜならば，ダウンロード処理ではアプリケーション外部に起因する待ち時間（ダウンロード元のサーバの処理時間やネットワーク遅延）が発生するため，この待ち時間の間に分析処理行うことで合計処理時間を短縮できるからである。

3.2. 分析処理

分析処理を複数のスレッドで並行して行う。なぜならば，ダウンロード処理よりも分析処理の方が時間がかかるため，分析処理を単一スレッドで行うと未分析ドキュメントが蓄積してしまうが，複数スレッドで行えばこの蓄積が起こらずに合計処理時間を短縮できるからである。

4. 実装上の制約

本アプリケーションでは，一部の処理の実装を行っていない。本来の実装と，代用の実装の関係は次のとおりである。

本来の実装	代用の実装
大量のドキュメント	1つのテキストファイル
1つのドキュメント	上記テキストファイルの1行
ドキュメントのダウンロード	ローカルディスクからのファイル読み込み
ドキュメントの分析	行中の単語数のカウント
時間がかかる処理	sleep

5. クラス・メソッドの一覧と説明

5.1. クラス一覧

クラス名	説明
Main	アプリケーション全体を制御するクラス。
DocDownloader	サーバから，大量のドキュメントをダウンロードするクラス。
WordCounter	ドキュメントを分析するクラス。
Result	結果集計を行うクラス。

5.2. Main クラスのメソッド

メソッド名	説明
main	引数：コマンドライン引数 返り値：なし DocDownloader クラスのインスタンスと，WordCounter クラスのインスタンス群を作成し，それぞれ別スレッドとして起動する。

5.3. DocDownloader クラスのメソッド

メソッド名	説明
run	引数：なし 返り値：なし ドキュメントをダウンロードする処理を，ローカルファイルを 1 行ずつ読み込む処理でシミュレートする。
getLine	引数：なし 返り値：String deque に格納されている読み込み済みの行群のうち，最初に読み込んだものを返して，その行を deque から削除する。deque が空の場合は null を返す。
isFinished	引数：なし 返り値：boolean ローカルファイルの読み込みが完了し，かつ，deque が空であるか否かを返す。

5.4. WordCounter クラスのメソッド

メソッド名	説明
countWord	引数：String 返り値：int 与えられた文字列を構成する単語数を返す。文字列は各単語がスペースで区切られた英文等を前提としている。
run	引数：なし 返り値：なし DocDownloader の読み込み処理が完了，かつ，deque が空になるまで，deque から読み込み済みの行を 1 件ずつ取得し，その行に含まれる単語数を countWord()を用いて求め，Result に結果を登録する。

5.5. Result クラスのメソッド

メソッド名	説明
addWordCount	引数：int 返り値：なし 与えられた単語数を全単語数に加算する。
getWordCount	引数：なし 返り値：int 全単語数を返す。

6. 並行処理を実現するために工夫した点

- DocDownloader において，ファイルの読み込みが完了，かつ，deque が空であるか否かを判定する isFinished メソッドを設け，WordCounter が分析／待機の判断を容易に行えるようにした点。
- DocDownloader の getLine メソッドにおいて，複数の WordCounter スレッドが同時に同じ要素を取得しようとする競合が起きないよう，排他制御を行った点。