תיאור העבודה אופן פעולת התכנית:

* עיקר התכנית בנוי משני חלקים אשר יחד יוצרים את קבצי הPOSTING השונים והמילון בהתאמה. קבצי הפוסטינג והמילון מהווים יחד את הINVERTED INDEX אשר יעזור בהמשך לאחזר את קבצי המאגר לפי השאילתות שיזין המשתמש.

חלק א- שלבי הקריאה, TEXT OPERATION, פרסור ויצירת הקבצים הזמניים.

כל הפעולות אשר מבוצעות בחלק זה הינן מופרדות אחת מהשניה והינן בלתי תלויות אחת בשנייה. תוצר החלק הנ"ל הינו קבצי פוסטינג זמניים אשר יהוו איחד מהיר ונוח לחלק הבא. המכנה המשותף לכל המחלקות הנ"ל היא שיטת התקשורת בין השלבים, תקשורת Producer Consumer . מחלקות אשר מהותן PRODUCER(יצירת אובייקטים) יקבלו תור אשר אליו יכניסו את האובייקט אותם הם מייצרות ומחלקות אשר מהותן CONSUMER(צריכת אובייקטים) יקחו את האובייקטים מאותו התור. חלק מהשלווים מתנהגים כצרכן עבור השלב הקודם להם וכיצרן לשלב הבא. לצורך חסכון בזמן הריצה וייעול חלק א ככלל כל שלב מתופעל ע"י תרדים (THREADS) הממקבלים את השלבים של חלק זה. על מנת לשלוט על התרדים, לקרוא לכל המחלקות באותו הזמן, לסדר אותם בסדר הנכון ולתת להם את התור הרלוונטי ישנה מחלקת master(תיאור בהמשך) אשר כל יעוד הוא ליצור ולהרוג את השלבים השונים בחלק זה. בכדי לשלוט בכמות המידע הנכנס משלב "קריאת הקבצים" נתנו לכל שלב שני SEMAPHORES אשר מגיבילים את קצב הצריכה מצד אחד(ובנוסף מורידים Busy waiting ) וקצב העברת האובייקטים לשלב הבא מצד שני. מאחר והשתמשנו בthreads נאלצנו להתמודד עם מרוצים בין תרדים השונים. לשם כך, השתמשנו בשיטת הpoisoned pill: כאשר מחלקת המסטר רואה כי היצרן סיים את עבודתו הוא שם בתור אשר קיים אצל הצרכן הרלוונטי אובייקט אשר יש הסכמה עליו כי יפסיק את תהליך הצריכה. באמצעות שיטה זאת מחלקות הצרכן לא יירדמו כאשר ינסו לקחת אובייקטים אשר כבר לא מיוצרים ולא היווצרו מרוצים בין התרדים השונים.

וכעת נתבונן בשלבי חלק א' ומחלקותיהן.

שלב קריאת הקבצים

מחלקת Document Reader

מחלקה זו היא מחלקה מסוג יצרן (Producer) והיא מממשת את ממשק הrunnable בכדי שתהיה אפשרות למקבל אותה. מטרת המחלקה הינה לקחת את הנתיב אשר היא מקבלת, לקרוא את קובץ מהנתיב ולחלק את הקבצים למסמכים. את המסמכים היא שמה בתור לצרכן הרלוונטי, קוראת עוד קובץ וחזור חלילה. המחלקה מסיימת את עבודתה כאשר לא נותרו קבצים לקרוא

שיטות:

**public void** run()

שיטה זו קוראת את הקובץ מהנתיב(התקבל בבנאי) מחלקת את הקובץ למסמכים בעזרת מחלקת XMLREADER, יוצרת אובייקט מסוג DOCUMENT ושמה אותו בתור.

מחלקת XML Reader

מחלקה זו עוזרת למחלקת Document Reader לפצל את המסמך למסמכים לפי תגיות המסמך.

שיטות:

**public** TextOperations.Document getNextDocument()

שיטה זו לוקחת את הקובץ (מקבלת בבנאי) ובעזרת האובייקט ELEMENT מהתוספת JSOUP מגדירה את המסמכים בקובץ. השיטה מפצלת כל אלמנט לפי התגיות של המסמך על מנת לקבל את הטקסט של המסמך, המדינה בה הטקסט נכתב וכו'.. השיטה מחזירה אובייקט מסוג DOCUMENT עם פרטי המסמך.

**public boolean** hasNext()

שיטה המחזירה ערך אמת אם קיים מסמך נוסף בקובץ, אחרת שקר.

מחלקת Document

מחלקה זו מייצגת את המסמך בעזרת רשימות של STRING לפי פיצול התגיות.

שיטות:

**public** String getPath()

שיטה זו מחזירה את נתיב המסמך

**public** String getID()

שיטה זו מחזירה את קוד הזיהוי של המסמך

**public** String getHeader()

שיטה זו מחזירה את הכותרת של המסמך

**public** String getDate()

שיטה זו מחזירה את תאריך יצירת המסמך

**public** String getText()

שיטה זו מחזירה את תוכן המסמך

שלב ה TEXT OPERATION

מחלקת Text Operation

מחלקה זו הינה מסוג צרכן של אובייקטים מסוג Documents. בנוסף לכך, המחלקה הינה מסוג יצרן של אובייקטים מסוג Tokenized Document. המחלקה מממשת את ממשק הrunnable בכדי שתהיה אפשרות למקבל אותה. מטרת המחלקה היא להוריד את חלק מה STOP WORDS וליצור TOKEN מכל מילה.

שיטות:

**public void** run()

השיטה לוקחת אובייקטים מסוג document מהתור , מעבירה את האובייקט דרך פילטר אשר מוריד מילים לא נחוצות. לאחר מכן השיטה יוצרת אובייקט מסוג Tokenized Document ודוחפת אותו לתור.

שלב הפירסור

מחלקת Master Parser

מחלקה זו הינה מסוג צרכן של אובייקטים מסוג Tokenized Document וגם מסוג יצרן של אובייקטים מסוג HashMap. המחלקה מממשת את ממשק הrunnable בכדי שתהיה אפשרות למקבל אותה. המחלקה יוצרת מפה של<String,TermDocumentInfo> כאשר המפתח הינו שם הTerm והערך הינו המידע על הTerm במסמך יחיד. המחלקה קוראת לכל הפרסרים השונים ונותנת להם לפרסר לפי חוקיהם ולהכניס את הערכים לאותה המפה.

שיטות:

**public void** run()

השיטה לוקחת אובייקטים מסוג Tokenized Document מהתור, יוצרת מפה ושולחת את שני האובייקטים הנ"ל למחלקות פרסור. בסוף התהליך השיטה דוחפת את המפה לתור.

ממשק Info

ממשק להחזרת מידע בצורת String

שיטות:

String toString()

השיטה מחזירה את המידע הרלוונטי על האובייקט בצורת String

ממשק ItermDocumentInfo

ממשק המתאר פונקציונליות של מידע על Term במסמך Document מסויים.

שיטות:

**int** getFrequency()

מחזיר את התדירות של המילה במסמך

Term getTerm()

מחזיר את הTerm

String getDocumentID()

מחזיר את השם המזהה של המסמך

**void** addToFrequency(**int** frequency)

השיטה מוסיפה לערך התדירות של המילה את המספר שקיבלה

**void** setFrequency(**int** frequency)

השיטה קובעת את ערך התדירות של המילה לפי המספר שקיבלה

מחלקת Abstract Parser

המחלקה מייצגת אובייקט המסוגל לפרסר אובייקטים מסוג TokenizedDocument על פי חוקים כלשהם.

שיטות:

Token get(**int** i)

השיטה מחזירה את הערך ה-i ברשימת הטוקנים

**abstract void** manipulate()

השיטה הינה שיטה מופשטת מאחר וכל Parser היורש ממחלקה זו יממש את הפרסור בצורה שונה עם חוקים שונים. השיטה תעבור על רשימת הטוקנים ותכניס את המילים (term) המתאימים לחוקי הפרסר למפה.

**void** putInMap(String s)

השיטה לוקחת את הString שיש להכניס למילון ובודקת האם הוא כבר קיים. אם קיים, המפה תעדכן את הערכים של AbstractTermDocumentInfo אחרת תיצור AbstractTermDocumentInfo חדש ותכניס את הערכים למפה כך שs הוא המפתח.

**int** getTxtSize()

השיטה מחזירה את גודל רשימת הטוקנים

**boolean** mapContains(String s)

השיטה מחזירה ערך אמת אם ערך המילה s קיים במילון, אחרת שקר.

**void** mapRemove(String s)

השיטה מעבירה את ערכי s.toUpperCase() לערך s ומוחקת את ערכי s.toUpperCase() מהמפה.

מחלקת CountryParser

המחלקה יורשת ממחלקת Abstract Parser . המחלקה מפרסרת לפי החוק הבא: עבור כל עיר המופיעה בטקסט וגם בתגית FP 104