

המחלקה להנדסת תוכנה

### פרוייקט גמר בהנדסת תוכנה – תשע"ה

**הצעת פרויקט**

**BWA Optimizer**

**יעול חיפוש מיקומם של מקטעי DNA על פני הגנום האנושי.**

|  |
| --- |
| **נ ת ו נ י ם א י ש י י ם** |
| **אבי טרנר**  **ת.ז: 039991708**  **דוא"ל:** [**avi.turner111@gmail.com**](mailto:avi.turner111@gmail.com) |
| **ארגון**  **עזריאלי – המככלה האקדמית בירושלים** |
| **מנחה אקדמי**  **ד"ר יהודה חסין** |

# תיאור מסגרת הפרויקט

הפרויקט הוא פרויקט מחקרי, על מנת לאפשר הבנה טובה יותר של הבעיה והפתרון המוצע, נפתח בתיאור קצר של הרקע ושל מונחים הקשורים לפרויקט.

**DNA**: הDNA היא מולקולת ענק שמצויה בכל אחד ואחד מתאי הגוף שלנו ובה מצוי כל המידע התורשתי לבניית החלבונים בתא אצל כל האורגניזמים הידועים, מחיידקים ועד לבני אדם.

המבנה של הDNA בנוי כמעיין "סולם" שמסתלסל סביב עצמו, כאשר ה"שלבים בסולם" מורכבים, כל אחד, מזוג בסיסים המתחברים זה לזה ומסומנים באותיות הלטיניות A, G, T, C.

בכל "שלב" מתחברים הבסיסים עם בן זוג קבוע – A עם T ו C עם G, כך שאם ידוע לנו רק צד אחד של ה"סולם" אנו יכולים לשחזר ממנו במדויק גם את הצד השני.

המדהים הוא שכ99.9% מהDNA של כל בני האדם משותף למרות אבני הבניין המועטות והפשוטות– וזוהי תכונה קרדינלית לפרויקט זה.

אם כן, את המידע המגולם בDNA ניתן לייצג כמחרוזת של הבסיסים המרכיבים אותו, וכך נתייחס אליו בפרויקט זה (לדוג': TGACCGTCAG....)

הDNA מורכב מכ שלבים. במונחים דיגיטלים, המידע הגלום בDNA שווה ערך למעט יותר מ 1.6 GB (או כ2 דיסקים...)

כמו כן, DNA יכול לעבור מוטציה, שינוי. רוב המוטציות אינן מזיקות אך אם הן מופיעות במקומות מסוימים על גבי רצף הDNA הן יכולות לגרום לבעיות גנטיות וביניהן לנטיה למחלות גנטיות ובפרט לסרטן.

כיום כאשר חולה מגיע לאבחון לראות האם הוא חולה בסרטן, תהליך הבדיקה הוא ארוך ומסורבל:

1. לקיחת דגימת DNA.
2. השוואת הDNA לDNA של אדם בריא.
3. איתור מוטציות בDNA.
4. חיפוש ידני במאגרי מידע האם המוטציות הן במקום שידוע כגורם לסרטן.
5. חיפוש ידני האם קיימת תרופה שעוזרת לסוג המסוים של הסרטן הנ"ל.

בין הבעיות בתהליך זה ניתן לציין את תהליך ההשוואה (2) שלוקח זמן ארוך במיוחד. כיום, בתי החולים שוכרים חוות שרתים ע"מ ליעל את החישוב הארוך של השוואה.

עם זאת, גם כיום שלב זה לוקח כיום שלם – פרויקט זה יתמקד בייעול שלב זה.

## הגדרות – שפה משותפת לפרויקט

* הגנום שנשתמש בו להשוואה לכל הקריאות (ראה להלן).
* – base pair - זוג בסיסים המהווים שלב ב"סולם הDNA". משמש כמידת אורך של קריאה (ראה להלן).
* – קריאה – קטע מהDNA של החולה שנדגם לצורך בדיקה.
* - אורך של קריאה. באופן טיפוסי הקריאות הן באורך של bp 100- bp 200 אך ה"לקוח" הספציפי של תוצר הפרויקט, משתמש בקריאות באורך bp35.
* – הקריאה שהחלה במקום ה ב X
* – אורכו של X ביחידות bp -

# תיאור הבעיה

נתאר מעט את תהליך ההשוואה כדי לעמוד על הקושי הכרוך בו.

על מנת להשוות את הDNA לDNA של אדם בריא, דגימה מהDNA של החולה נדגמת ונחתכת למספר רב של חתיכות, להלן "קריאות", קצרות יחסית, באורך של כbp 100 (תהליך זה נקרא NGS - Next-generation sequencing).

באמצעות אלגוריתם BWA מוצאים את המיקום המתאים של הקריאה על גבי הרצף של הDNA של האדם הבריא.

השימוש באלגוריתם לחיפוש יעיל הכרחי מכיוון, שכזכור, בDNA ישנן באופן טיפוסי מוטציות שאינן נמצאות על הDNA המושווה (בנוסף לחלקים היחודיים לכל אדם ואדם).

נעמוד על הקושי שבהשוואה שכזו על ידי ניתוח של אלגוריתם השוואה נאיבי של קריאה (מחרוזת) שיתכן שנפלו בה 0-2 שגיאות במקום לא ידוע:

נניח שאורך של קריאה הוא 100.

אם לא נפלה אף שגיאה – ישנה מחרוזת 1 להשוואה.

אם נפלה שגיאה אחת – ישנם 100 מחרוזות להשוואה.

אם נפלו 2 שגיאות – ישנם 9,900 מחרוזות להשוואה.

סה"כ, בהנחה של עד 2 שגיאות, עברנו מקריאה אחת באורך 100 ל10,001 מחרוזות באורך 100 שנצטרך להשוות. מכיוון שמלכתחילה יש לנו כ3,000,000,000 מחרוזות כאלו, ברור שחיפוש שכזה אינו ישים עבור מידע מסדר גודל של הגנום האנושי.

יש לציין שבתאור זה הנחנו שטעויות באות לידי ביטוי בהחלפת אות אחת באות אחרת בעוד שלמעשה יתכנו טעויות של החלפת מיקומים של אותיות \ קטעים, וכן טעויות בקריאה של המכונה הדוגמת.

על מנת להתגבר על בעיות כגון אלו, בשלב הדגימה לוקחים המון דגימות – בכמות כזו שסטטיסיטית כל מקטע של הDNA נדגם מספר פעמים. דבר זה עוזר כדי לוודא שאכן כל הDNA נדגם (בסבירות גבוהה) וגם מחפה על טעויות בקריאה (לא סביר שתהיה טעות באותו המקום בכל הדגימות).

נעמוד על פרונות אפשריים לבעיה זו ע"י שימוש באלגוריתמים קיימים:

1. האלגוריתם הנאיבי, שכבר הראנו שאינו רלוונטי ויעילותו:
2. KMP – אלגוריתם המנצל את מבנה התבנית על מנת ליעל את החיפוש ויעילותו היא , למרות שאלגוריתם זה הוא יעיל למדי מפני שn שלנו גדול מאוד, ומפני שיש צורך לבצע אלגוריתם זה עבור כ"א מהקריאות (סדר גודל של כמה מליונים), ולכן גם הוא לא טוב מספיק.
3. BWA – אלגוריתם יעיל לחיפוש מהסוג שלנו שפותח בדיוק למטרה זו, ויתואר ביתר הרחבה בהמשך. יעילותו (לא תלוי באורך באורך הגנום!).

בפועל, כיום תהליך זה לוקח זמן רב (כיום שלם) למרות השימוש באלגוריתם זה מכייון שמריצים את האלגוריתם עבור כל קריאה.

1. BWA ממוקבל – אותו האלגוריתם כמו BWA אך ממומש באמצעות מיקבול האלגוריתם על גבי מאיץ גרפי כך ש הפעולות לא מתבצעות באופן סדרתי - מטרת הפרויקט היא לממש אלגוריתם זה.

# תיאור הפתרון

על מנת להתגבר על בעית זמן הריצה, פותח האלגוריתם BWA. בבסיסו, האלגוריתם מחולק לשלושה שלבים:

1. Index - אינדוקס של הגנום – שמירת הגנום בצורה שניתן לגשת אליו בצורה יעילה. זו פעולה שיש לעשות אותה פעם אחת בלבד (ולא בכל בדיקה של חולה...).
2. Alignment – מציאת המיקום של הקריאות על פני הגנום.
3. Pairing – כחלק מקריאת הDNA של החולה, הדגימות נחתכות ל- 2 ויש צורך למצוא התאמה בין 2 חלקי הדגימה.

שלב הAlignment, שבו יתמקד פרויקט זה, מיועל ע"י ישום של "עץ סיפות" השומר את כל הסיומות האפשריות של הגנום (זהו האינדקס משלב 1), ומימוש של אלגוריתם המחפש ב"צורה חכמה" עבור כל קריאה לאיזה סיומת היא מתאימה בהינתן מספר טעויות כלשהו (זהו פרמטר הניתן לשינוי).

יתרונו הגדול של ה BWA הוא שבזכות השימוש ב"עץ סיפות", שגורם לכל מחרוזות שחוזרות על עצמן בגנום להתבטא בענף אחד של העץ, ולכן זמן הריצה של האלגוריתם תלוי באורך הקריאה ולא באורך הגנום.

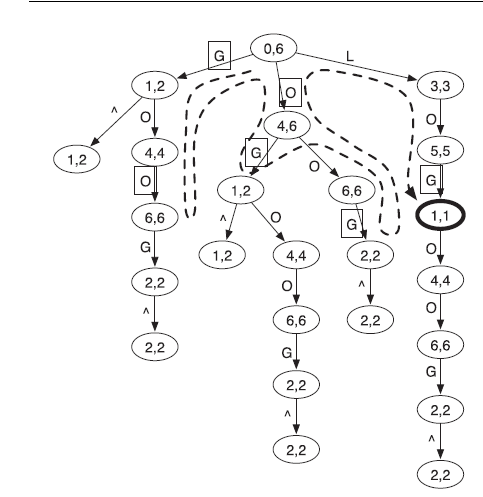
אבחנה זו גם מרמזת שייתכן וניתן לייעל עוד יותר את האלגוריתם ע"י מיקבול הפעולות של מציאת מיקום הקריאות השונות.

כאמור, אלגוריתם זה כבר נמצא בשימוש ע"י בתי החולים ששוכרים חוות שרתים לביצוע חישוב זה, ועדיין התהליך לוקח כיום.

בפרויקט זה ננסה ליעל את את האלגוריתם ע"י יישומו בעזרת תכנות מקבילי על מעבדים של מאיצים גרפיים.

## המחשה ראשונית של שלבים האלגוריתם

**עץ רישות** של המחרוזת "Googol". הסמל ∧ מסמן את תחילתה של המחרוזת.

****שני המספרים בכל צומת הם האינטרוול ב SA (מערך הסיפות) של המחרוזת המיוצגת על ידי

הצומת.

החלק המעניין בתרשים זה הוא הקו המקווקו המראה את המסלול של האלגוריתם בחיפוש אחר המחרוזת 'LOL', תוך אפשור חוסר התאמה אחד. נשים לב ש:

1. הקו לא יורד לכל עומק העץ.
2. רישות שלהן יש רישה משותפת, נמצאים על אותו מסלול של האלגוריתם (חסכון בזמן ריצה).

האותיות על הצלעות המוקפות ריבוע,מסמנות חוסר התאמה ("שגיאה") לשאילתא בחיפוש. ההתאמה היחידה לחיפוש היא הצומת המודגשת [1,1] המייצגת את המחרוזת 'GOL' מחרוזת.

## בניית האינדקס בעזרת מערך סייפות:

## יהי X = googol$

## נבצע הזזה מחזורית של X, ונשמור תוצאה של כל הזזה כרשומה.

## לאחר מכן נמיין את הרשומות מיון לקסוגרפי.

## לאחר המיון, התו הראשון בכל רשומה מהוות את מערך הסיומת S(i) (6,3,0,5,2,4,1)

## השרשור של התוים האחרונים של הרשומות נותן לנו את "מחרוזות BWT" b[i]- lo$oogg.

וזהו למעשה האינדקס שנשתמש בו.

(ניתן לעיין בנוגע ל טרנספורם BWT – זהו נושא שלם בפני עצמו).

## תיכון ראשוני:

# לשם יישום הפרויקט נשתמש בשפות הבאות:

# C# - לבניית אב טיפוס, הוכחת התכנות, בדיקות יחידה וניסויי שיפור באלגוריתם.

# C++ - שימוש במעטפת קיימת של אלגורית BWA, קריאות לקוד Cuda.

# Cuda – מימוש החלק המקבילי בפרויקט ע"ג מאיץ גרפי.

# במהלך הפיתוח נשתמש במערכות ההפעלה הבאות:

# Linux – מערכת ההפעלה שעליה פרויקט הBWA הנכחי יכול לרוץ

# Windows – לבניית אב טיפוס וייתכן שהמוצר הסופי יעבוד על גבי מערכת הפעלה זו (תלוי בהיקף העבודה שידרש להמרת קוד קיים כך שיוכל לרוץ במערכת הפעלה זו).

## שימוש ברכיבים קיימים:

* הפרויקט מבוסס על יעול של אלגוריתם קיים. לכן, אנו נשתמש ברכיבי הפרויקט של BWA[[1]](#endnote-1).
* עבור כלי בדיקה, נשתמש בnunit.

## מהי המערכת

המערכת היא תכנה שמבצעת חיפוש של מחרוזת קטנה בתוך מחרוזת ארוכה (מאוד), כאשר החיפוש יניב תוצאות גם אם נפלו שגיאות במחרוזת הקצרה, כך שהיא לא זהה במדויק למקטע במחרוזת האורכה.

## תהליכים ונתוני המערכת

במערכת ישנם שלושה תהליכים:

1. Index - אינדוקס של הגנום – שמירת הגנום בצורה שניתן לגשת אליו בצורה יעילה. זו פעולה שיש לעשות אותה פעם אחת בלבד (ולא בכל בדיקה של חולה...).
2. Alignment – מציאת המיקום של הקריאות על פני הגנום.
3. Pairing – כחלק מקריאת הDNA של החולה, הדגימות נחתכות ל- 2 ויש צורך למצוא התאמה בין 2 חלקי הדגימה.

בפרויקט זה נממש את שלב ה Alignment.

המידע שקיים במערכת הוא:

1. הגנום האנושי המאונדקס (מידע סטטי).
2. קטעי דגימות DNA (הקלט של המערכת).

## תיאור הכלים המשמשים לפתרון

ע"מ לייעל את מימוש האלגורית BWA נשתמש במאיץ גרפי ובשפת C for Cuda בסבביבת עובדה של Visual Studio / Eclipse.

התכנות בסביבת CUDA, ובעיקר בשפת התכנות C for CUDA, מורכב משני אלמנטי תוכנה עיקריים[[2]](#endnote-2):

1. הקוד המתמטי-מדעי, הנכתב בשפת C for CUDA ומכיל את הגרעין (Kernels) החישובי.
2. קוד המעטפת (בדרך כלל בשפת C) אשר אחראי להקצאות ושחרור זיכרון, העתקות יחידות מידע אל ההתקן ומן ההתקן חזרה אל המחשב המארח, והפעלת גרעיני החישוב.

סביבת הפיתוח כוללת את המרכיבים הבאים:

1. מהדר (Compiler)
2. ממשק תוכנה (סט קובצי C header files) וספריות תוכנה.
3. מנהל התקן (driver)
4. סביבת הרצה: CUDA תומכת במערכות ההפעלה: Microsoft Windows, לינוקס ו-Mac OS.

# השוואה לפתרונות בספרות

פרויקט הקוד הפתוח BarraCUDA[[3]](#endnote-3) יישם גם כן את אלגוריתם BWA בעזרת תכנות מקבילי. בשלב זה אנו נמנעים מלהסתכל בצורת המימוש שלו ע"מ לשמור על ראש פתוח לרעיונות ייעול משלנו.

# תכנון הפרויקט

## לוח זמנים מתוכנן

|  |  |
| --- | --- |
| 26.11.14 | הגשת הצעת הפרויקט |
| 15.12.14 | אב טיפוס מוכן. |
| 22.12.14 | הבנת פרמטרים מפרויקט BWA :   1. ערך ברירת מחדל של מספר השגיאות המותר. 2. בדיקה על אב טיפוס - תלותת של זמן ריצה ב:    1. אורך הקלט.    2. מספר שגיאות מותר. |
| 16.1.15 | הרצת קוד BWA ממוקבל מהפרויקט הקיים. |
| 23.1.15 | איתור הקוד שנמקבל. |
| 20.3.15 | מיקבול הקוד. |
| 15.4.15 | סיום פיתוח. |
| 28.6.15 | הגשת הפרויקט |
| 28.7.15 | הצגת הפרויקט |

**הערה:** בגלל אופיו המחקרי של הפרויקט, נפח העבודה במונחים של כמות קוד היא קטנה ולכן הן Waterfall והן agile לא נמצאו מתאימים כמתודלוגיית עבודה. צווארי הבקבוק שאובחנו הם:

1. הכנת אבטיפוס.
2. מיקבול הקוד.

## טבלת סיכונים

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **הסיכון** | **חומרה** | **סבירות** | **מדד סיכון (סבירות X חומרה)** | **מענה אפשרי** |
| 1 | המיקבול יתברר כלא אפשרי. | 5 | 1 | 5 | ננסה ליעל את האלגוריתם עצמו |
| 2 | המיקבול יהיה פחות יעיל ממקבול קיים. | 3 | 3 | 9 | נמקבל מה שאפשרי, וננסה ליעל את האלגוריתם. |
| 3 | חוסר הבנה של האלגוריתם. | 4 | 2 | 8 | נבצע בדיקות של זמן ריצה ותוצאות בסיוע של בדיקות יידה לוודא נכונות גם בלי הבנה מעמיקה. |

# רשימת ספרות

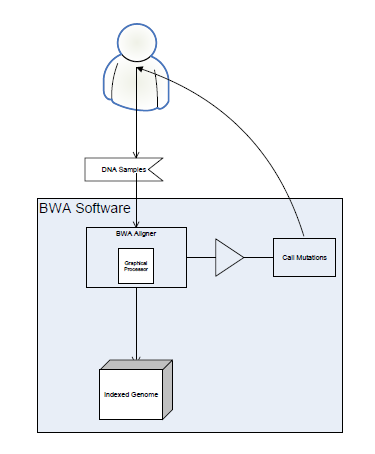
1. להכנת מסמך זה נעזרתי
   1. בויקיפדיה:
      1. Cuda: <http://he.wikipedia.org/wiki/CUDA>
   2. Short-Read Sequence Alignment: <http://en.wikipedia.org/wiki/List_of_sequence_alignment_software#Short-Read_Sequence_Alignment>
   3. האיציקלופדיה הקטנה של המדעים מאת רן לוי.
2. ספרות עם חומר שרלוונטי לפרויקט:
   1. הרצאות בביואינפורמטיקה פרקים 17-21. <https://www.cs.cmu.edu/~ckingsf/bioinfo-lectures/>.
   2. שאלות נפוצות בקשר ל BWA: <http://bio-bwa.sourceforge.net/>
   3. מסמך המתאר את צורת העבודה של האלגוריתם: <http://www.math.pku.edu.cn/teachers/xirb/Courses/biostatistics2013/Bioinformatics-2009-Li-1754-60.pdf>.
   4. השוואה בין שיטות NGS: <http://en.wikipedia.org/wiki/DNA_sequencing#Next-generation_methods>

# נספחים

**יומן הפרויקט:** <http://projects.jce.ac.il/moodle/mod/wiki/view.php?id=6>

**מאגר קוד:** <https://github.com/turner11/BWA-Final_Project>

**דיאגרמת רכיבים \ הפצה (UML)**



**טבלת דרישות (User Requirement Document)**

|  |  |
| --- | --- |
| מס' דרישה | תיאור |
| 1 | GPU – מאיץ גרפי מתוצרת Nvidia |
| 2 | דגימות DNA מסוג NGS – עבור קלט למערכת. |
| 3 | מחשב עם התכנה מותכנת. |

1. <http://sourceforge.net/projects/bio-bwa/?source=navbar> [↑](#endnote-ref-1)
2. <http://he.wikipedia.org/wiki/CUDA> [↑](#endnote-ref-2)
3. <http://seqbarracuda.sourceforge.net/> [↑](#endnote-ref-3)