18 אוגוסט 2023

תרגיל מסכם ("פרויקט") בתכנות מקבילי ומבוזר סמסטר קיץ 2023

התרגיל יכתב בעזרת MPI, OpenMP, CUDA.

תאור הבעיה של sequence alignment

התרגיל עוסק בהשוואה בין סדרות (מחרוזות) של אותיות באי"ב האנגלי.

הוא פישוט של אלגוריתמים שמשתמשים בהם בביו-אינפורמטיקה שם המטרה היא למצוא דמיון בין מולקולות ביולוגיות (חלבונים, DNA, RNA). בהקשר זה כל אות מייצגת איזו ישות כימית וסדרה של אותיות מייצגת מבנה של מולקולה.

למשל מולקולת DNA מורכבת מארבעה סוגים של nucleotides שמקובל לסמן אותם באותיות A, T, C, G. ואז המחרוזת TCCGT יכולה לייצג קטע של מולקולת DNA.

הבעיה של השוואת סדרות שנעסוק בה מכונה בביולוגיה בעיית ה- **sequence alignment**.

alignment score.

כשמשווים בין 2 סדרות, נותנים "ציון" לתוצאת ההשוואה. זה ה- alignment score.

ככל שהציון גבוה יותר -- הסדרות נחשבות לדומות יותר.

חישוב ה- alignment score של שתי סדרות שאורכן שווה

כדי לחשב את ה- alignment score של שתי סדרות seq1 ו- seq2 שלשתיהן אותו אורך, יש להשוות את האות הראשונה של seq1 לאות הראשונה של seq2, את האות השנייה של seq1 לאות השנייה של seq2 וכן הלאה. לכל השוואה כזאת ניתן "ציון" שהוא מספר שלם (שעשוי להיות שלילי). נגדיר שסכום הציונים האלו הוא ה- alignment score.

נשתמש בטבלה שנקרא לה "טבלת הציונים" שבה ישמר מיפוי בין כל זוג אותיות לציון שלהם.

לכל אות באי"ב האנגלי יש שורה בטבלה: השורה הראשונה היא עבור האות 'A', השנייה עבור האות 'B', השלישית עבור 'C' וכן הלאה. באופן דומה, לכל אות באי"ב האנגלי יש עמודה: העמודה הראשונה היא עבור האות 'A', השנייה עבור האות 'B' וכן הלאה. בטבלה יהיו בסה"כ 26 שורות ו- 26 עמודות.

הציון של זוג אותיות (אות x בסדרה הראשונה ואות y בסדרה השנייה) ימצא בכניסה בטבלה הציונים שנמצאת בשורה עבור האות x בעמודה עבור האות y.

<u>דוגמא</u>

נניח ש-"טבלת הציונים" היא:

	column A	column B	column C	column D	...
row A	2	4	0	4	
row B	-1	2	7	5	
row C	5	1	4	9	
row D	1	2	3	4	
...					

(כדי לקצר, רשומים כאן רק ארבע עמודות של ארבע השורות הראשונות של הטבלה)

בדוגמא זו:

seq1 = A B B D A B

seq2 = A C A D C A

alignment score = 2 + 7 + (-1) + 4 + 0 + (-1) = 11

<u>חישוב ה- alignment score של 2 סדרות seq1, seq2 כאשר</u>

<u>seq2 קצרה יותר.</u>

רושמים את seq2 מתחת ל- seq1 בהיסט (offset) מסוים

כאשר offset >= 0. אסור שאותיות של seq2 יופיעו מעבר לסוף של seq1.

את ה- alignment score מחשבים תוך התעלמות מהאותיות של הסדרה הארוכה יותר שאין להן אות תואמת.

<u>דוגמא</u>

נניח שהטבלת הציונים כמו בדוגמא הנ"ל.

seq1 = A B B D A B

seq2 = A D C

נשווה בין הסדרות כאשר ה- offset של הסדרה השנייה הוא 2:

seq1 = A B B D A B

seq2 = A D C

alignment score = (-1) + 4 + 0 = 3

בדוגמא זו offset = 0 יניב ציון (alignment score) של 2+5+7 =14

offset = 1 יניב ציון של 7 = 3+5+(-1)

offset = 3 יניב ציון של 12 = 7+4 +1

שימו לב ש- offsets גדולים יותר אינם אפשריים כי אז הסדרה השנייה תמשיך מעבר לסוף הסדרה הראשונה.

בדוגמא זו offset = 0 מניב את ה- alignment score הגבוה ביותר.

<u>הגדרה של Mutant Sequence ("מוטציה")</u>

עבור סדרת אותיות seq נגדיר את ה- Mutant Sequence שיסומן ב- MS(seq, k) (כאשר k = 0, 1, 2 ... strlen(seq)).

זו סדרת האותיות המתקבלת ע"י כך שכל אות בסדרה seq מלבד k האותיות הראשונות, מוחלפת באות העוקבת לה:

A מוחלף ב- B, B מוחלף ב- C וכן הלאה (Z מוחלף ב- A).

שימו לב שבתור מקרה קצה, k יכול להיות strlen(seq). במקרה כזה אף אות לא מוחלפת ונשארים עם הסדרה המקורית. כלומר כל סדרה תחשב גם כ-"מוטציה" של עצמה.

למספר k ב- MS(seq, k) נקרא <u>"פרמטר המוטציה"</u>.

דוגמא: עבור סדרת האותיות ACZT יהיו חמישה mutant sequences:

MS(ACZT, 0) = BDAU

MS(ACZT, 1) = ADAU

MS(ACZT, 2) = ACAU

MS(ACZT, 3) = ACZU

MS(ACZT, 4) = ACZT

"המוטציה" האחרונה היא בעצם הסדרה המקורית ACZT עצמה.

<u>תאור הבעיה</u>

עבור מחרוזות נתונות seq1, seq2 כאשר seq2 היא הקצרה יותר

יש למצוא את ה- offset ואת פרמטר המוטציה k של seq2 עבורם יתקבל

ה- alignment score המקסימלי כשמשווים את המוטציה של seq2 ל- seq1.

<u>דוגמא</u>

seq1 = A B B D A B

seq2 = A C Z

ל- seq2 יש ארבע מוטציות אפשריות (פרמטר המוטציה k יכול להיות 0 או 1 או 2 או 3).

יש שלושה offsets אפשריים לכל אחת מהמוטציות. לכן יש כאן בסה"כ

4 * 3 = 12 צרופים אפשריים של ה- offset ו- k (פרמטר המוטציה) .

<u>השאלה היא: איזה מהם יניב את ה- alignment score הגבוה ביותר?</u>

למען הפשטות נניח ש-"טבלת הציונים" אומרת שהציון של השוואת שתי אותיות זהות (למשל a עם a, b עם b ...) הוא 1 והציונים של השוואת אותיות שאינן זהות הוא 0.

אז ניתן לראות שעבור offset = 2, k = 0 יתקבל ה- alignment score המקסימלי:

המוטציה של seq2 שתניב ציון מקסימלי היא: MS(ACZ, 0) = B D A

ואז כשנשווה את המוטציה הזאת עם offset = 2 ל- seq1 נקבל:

seq1:	A	B	B	D	A	B
MS(seq2, 0), offset=2:			B	D	A	
alignment score =			1 +	1 +	1 =	3

וניתן לראות שעם "טבלת הציונים" שהנחנו -- זה יהיה ה- alignment score המקסימלי.

<u>התכנית שעליכם לכתוב</u>

בקלט יופיעו מספר סדרות של אותיות. יש להשוות את הסדרה הראשונה (נסמנה Seq1 ונכנה אותה "הסדרה הראשית") לכל אחת מהסדרות המופיעות בהמשך הקלט.

עבור כל אחת מסדרות אלו יש למצוא את ה- offset ואת פרמטר המוטציה (k) עבורן יתקבל ה- alignment score המקסימלי כשמשווים את המוטציה ל- Seq1.

הקלט לתוכנית יופיע ב- standard input. הפלט יכתב ל- standard output.

רק אחד מתהליכי ה- MPI (תהליך 0) יקרא את הקלט וידאג להעביר את המידע הנחוץ לתהליכים האחרים. אותו תהליך יכתוב גם את הפלט.

מספר תהליכי ה- MPI יקבע בעת הרצת התכנית. אין להניח שיהיו רק 2 תהליכים.

גם מספר ה- threads של OpenMP לא צריך להיות קבוע בקוד של התכנית

(אין להגדיר משהו כמו #define NTHREADS 4).

<u>הפורמט של הקלט</u>

בשורה הראשונה של הקלט תופיע הסדרה הראשית (Seq1) (לא יותר מ- 3000 אותיות)

בשורה הבאה יופיע מספר שלם שנכנה אותו כאן number_of_sequences.

זה מספר הסדרות שיש להשוות לסדרה הראשית (Seq1).

ב- number_of_sequences השורות הבאות יופיעו הסדרות אותן יש להשוות

ל- Seq1 כאשר כל סדרה כזאת תופיע בשורה נפרדת. האורך של כל סדרה כזאת לא יעלה על 2000 אותיות ואורכה יהיה קטן מהאורך של Seq1.

כל סדרה תופיע בקלט כסדרת אותיות שאין ביניהם white space למשל

ABGFA (לא A B G F A).

הקלט אינו case sensitive כלומר אין הבחנה בין אותיות קטנות וגדולות. למשל הסדרה aBCd, תחשב זהה לסדרה ABcD.

<u>הפלט</u>

בפלט יופיעו number_of_sequences שורות, שורה עבור כל סדרה שהופיעה בקלט והושוותה לסדרה הראשית Seq1. בשורה ירשם מה ה- alignment score

המקסימלי שנמצא עבור אותה סדרה ומה ה- offset ופרמטר המוטציה (k) שמניבים את ה- score המקסימלי. כל שורה תהיה בפורמט

highest alignment score = … offset = … k = …

הסדר של השורות בפלט תואם את סדר הופעת הסדרות בקלט.

command line argument עבור "טבלת הציונים"

לתכנית יהיה command line argument אחד אופציונלי: שם הקובץ שבו מופיעה "טבלת הציונים". בקובץ זה יופיעו מספרים שלמים (חיוביים או אפס או שליליים) מופרדים ע"י white space. מספרים אלו הם התוכן של "טבלת הציונים"

מסודר לפי שורות: בהתחלה מופיעים המספרים של השורה של A, אחריהם

המספרים של השורה של B וכך הלאה. (בכל שורה מופיע קודם המספר בעמודה A,

אחריו המספר בעמודה B וכך הלאה).

כדי לפרט את תוכן הטבלה כולה יהיה צורך ב- 676 = 26*26 מספרים.

אם בקובץ מופיעים פחות מ- 676 מספרים, אז נסכים שהמספרים החסרים כולם 0.

כאמור, ה- command line argument המפרט את שם הקובץ בו מופיעה טבלת הציונים הוא אופציונלי. אם הוא חסר אז ברירת המחדל עבור טבלת הציונים היא טבלה שבה כל הכניסות הן אפס מלבד האלמנטים באלכסון שערכם 1.

הנה חלק מהטבלה שהיא ברירת המחדל:

	column A	column B	column C	column D	...
row A	1	0	0	0	
row B	0	1	0	0	
row C	0	0	1	0	
row D	0	0	0	1	
...					

במילים אחרות, השוואה של אותיות זהות תקבל ציון 1. השוואה של אותיות שאינן זהות תיתן ציון 0.

רק תהליך 0 של MPI יגש ל- command line argument והוא יפיץ את טבלת הציונים לכל שאר התהליכים.

הנחיות

ההגשה דרך מודל ביחידים. מותר להתייעץ עם חברים אבל את הקוד יש לכתוב לבד.

אם משתמשים בקוד שהורד מהאינטרנט יש לציין את מקורו.

יש לצרף תיעוד שמסביר את האלגוריתם ואת מבני הנתונים בהם השתמשתם.

התיעוד יהיה קצר (לא יותר מעמוד אחד).

כל סטודנט או סטודנטית "יגנו" על העבודה בפגישת zoom בה הם יריצו את התרגיל ויסבירו מה עשו. יש לדעת להסביר כל שורה בקוד.

יש לדעת לענות על שאלות כמו למשל:

מה חלוקת העבודה בין תהליכי ה- MPI?

מה חלוקת העבודה בין ה- threads ב- OpenMP ?

מה חלוקת העבודה בין ה- CUDA threads ?

התוכנית צריכה לרוץ מהר יותר מגרסה סדרתית שלה.

לצורך כך יש להכין גם גרסה סדרתית של התכנית ולהציג זמני ריצה של הגרסה הסדרתית מול הגרסה המקבילה.

במדידת זמנים של הגרסה המקבילה אין לקחת בחשבון את הזמן שלוקח להפיץ את טבלת הציונים בין תהליכי ה- MPI.

הניקוד:

15 נקודות יורדו אם אין שימוש ב- CUDA.

עבור כל יום איחור בהגשה (כולל ימים שאינם "ימי עסקים") תורד נקודה מהציון.

האלגוריתם, הצורה שבה התכנית מוקבלה ואיכות הקוד ילקחו בחשבון בעת מתן הציון.

טיפים:

עדיף לא ליצור עותק חדש של כל סדרה עבור כל מוטציה שלה כי העתקה

של הרבה מאוד מחרוזות עלולה לקחת זמן רב. כשמחשבים alignment score של מוטציה ניתן להשתמש בעותק המקורי של הסדרה ממנה היא נגזרת כאשר במהלך ההשוואה יודעים שבמקום חלק מהאותיות יש לקחת בחשבון את האותיות העוקבות.

אל תקראו (באופן ישיר או עקיף) ל-kernel של CUDA מתוך אזור מקבילי של OpenMP אלא אם יש לכם סיבה טובה לעשות את זה.

אל תקראו (באופן ישיר או עקיף) לפונקציות MPI מתוך אזור מקבילי של OpenMP אלא אם יש לכם סיבה טובה לעשות את זה.

בהצלחה!