תרגיל תכנות 2 במחשוב מקבילי ומבוזר (סמסטר קיץ 2020)

יש להשתמש ב- MPI בתרגיל זה. הגשה בזוגות.

התוכנית תקרא מהקלט סדרה של ערכים ותמיין אותם בסדר עולה (או ליתר דיוק בסדר לא יורד כדי לאפשר גם ערכים שווים).

השתמשו ב- ShearSort כדי למיין. אלגוריתם זה מחייב למיין שורות ועמודות. לצורך כך השתמשו באלגוריתם המיון odd even transposition sort. תיאור של שני האלגוריתמים מופיע בהמשך.

מספר הערכים בקלט צריך להיות רבוע של מספר שלם כדי שניתן יהיה לסדר את הערכים במטריצה בה מספר השורות שווה למספר העמודות. למשל אם יש בקלט 25 ערכים אז ניתן יהיה לסדר אותם במטריצה של 5 שורות ו- 5 עמודות. אם מספר הערכים בקלט אינו עומד בתנאי אז יש להוציא הודעת שגיאה ולסיים.

מספר התהליכים יהיה כמספר הערכים בקלט. כל תהליך יאחסן ערך אחד. (תהליכים יחליפו ביניהם ערכים). התהליכים יהיו מסודרים בטופולוגיה וירטואלית קרטזית. אם בקלט יהיו למשל 25 ערכים אז התהליכים יהיו מסודרים בחמש שורות וחמש עמודות.

קלט ופלט

כל ערך בקלט מורכב משלושה מספרים שלמים מופרדים ע"י white space (רווחים, טאבים, newlines). אחרי כל שלשה יופיע הסימן $. דוגמא לקלט (כאן יש 3 שלשות):

```
15 14 8 $ 1 2 3 $

7 8

 9$
```

הקלט יקרא מקובץ ששמו יופיע כ- `command line argument`. אם אין `command line argument` יש לקרוא את הקלט מה- `standard input`.

התהליך עם rank אפס יקרא את הקלט מהקובץ ויפיץ אותו לשאר התהליכים כנדרש.

את הפלט (סדרת הערכים ממוינת בסדר לא יורד) יכתוב התהליך עם rank אפס ל- `standard output`. בפלט כל שלשה של מספרים תופיע בשורה נפרדת.

שלשות של מספרים

הסדר בין השלשות הוא סדר לקסיקוגרפי (סדר מילוני). כלומר הסדר ביניהן נקבע לפי הסדר בין הרכיבים הראשונים (משמאל) בהם הם נבדלים.

למשל `(7,8,9) < (7,10,2)` כי `8 < 10`.

כדי להקל על משלוח שלשות בין תהליכים יש להגדיר data type בעזרת MPI_Type_struct.

<u>הנה תאור של שני האלגוריתמים למיון</u>

<u>אלגוריתם ShearSort</u>

הקלט הוא מטריצה לא ממוינת של ערכים בגודל $n \times n$ (מספר הערכים שיש למיין הוא n*n).

הפלט הוא מטריצה ממוינת בסדר "דומה לנחש".

הכוונה שכדי לקרוא את הערכים בסדר ממוין יש לקרוא את השורה הראשונה משמאל לימין, את השורה השנייה מימין לשמאל, את השורה השלישית שוב משמאל לימין וכן הלאה.

הנה האלגוריתם: (הניסוח הזה מבוסס על התאור ב-

(https://www.inf.hs-flensburg.de/lang/algorithmen/sortieren/twodim/shear/shearsorten.htm)

1. יש לחזור על הצעדים הבאים log(n) פעמים:

   מיין את השורות בכיוונים מתחלפים. כלומר שורות 0, 2, 4, ... ימוינו כך שהערכים יגדלו <u>משמאל לימין</u>, שורות 1, 3, 5, 7 ... ימוינו כך שהערכים יגדלו <u>מימין לשמאל</u>.

   מיין את העמודות (כך שהערכים יגדלו <u>מלמעלה למטה</u>)

2. מיין את השורות בכיוונים מתחלפים.

(הערה: אם log(n) אינו מספר שלם אז עגלו אותו למעלה).

ניתן לראות דוגמא בעמוד 43 במצגת של פרק 9 "אלגוריתמים למיון" לפי ספר של Wilkinson & Allen. המצגת נמצאת ב- moodle (בתיקיה "שונות").

<u>אלגוריתם Odd Even Transposition Sort</u>

נסמן את התהליכים ב- P0, P1, P2 ... . התהליכים מסודרים בשורה וכל אחד מהם מחזיק בערך בודד. המטרה היא למיין את הערכים המוחזקים ע"י התהליכים.

בצעד הראשון כל התהליכים שמספרם <u>זוגי</u> עושים compare-and-exchange עם שכנם מימין (P0 עם P1, P2 עם P3 וכן הלאה). בצעד השני כל התהליכים שמספרם <u>אי זוגי</u> עושים compare-and-exchange עם שכנם מימין (P1 עם P2, P3 עם P4 וכן הלאה).

בצעד הבא שוב הזוגיים עושים compare-and-exchange עם השכן מימין. בצעד לאחריו האי זוגיים מתקשרים עם השכן מימין וכן הלאה.

בסך הכל יש לעשות n צעדים כאשר n מספר הערכים שיש למיין.

בסופו של דבר הערכים x0, x1, x2 ... (המוחזקים בהתאמה ע"י P0. P1, P2 ...) יהיו ממוינים.

כאשר שני תהליכים עושים compare-and-exchange הם משווים בין המספרים שהם מחזיקים

ובמקרה הצורך מחליפים ביניהם את המספרים כך שיופיעו בסדר המבוקש. לדוגמא נניח ש- P4 מחזיק במספר 20 ושכנו מימין P5 מחזיק במספר 10 ואנו מעונינים למיין את המספרים כך שיגדלו משמאל לימין. אז פעולת compare-and-exchange בין P4 ו- P5 תסתיים בכך ש- P4

יחזיק ב- 10 ו- P5 יחזיק ב- 20. אילו לפני ביצוע ה- compare-and-exchange P4 היה מחזיק ב- 10 ו- P5 היה מחזיק ב- 20 אז שני התהליכים היו ממשיכים להחזיק באותם הערכים

גם אחרי ה- compare-and-exchange.

ניתן לראות דוגמא בעמוד 16 במצגת של פרק 9 "אלגוריתמים למיון" לפי ספר של Wilkinson & Allen . המצגת נמצאת ב- moodle (בתיקיה "שונות") .