**מחשוב מקבילי ומבוזר פרויקט סיום Perceptron**

מגישה: דוראל שושני 204059968

**MPI + OpenMP + CUDA**

**הגדרת הבעיה:**

נתונות n נקודות, לכל נקודה יש: K קורדינטות, ו answerהמהווה מיקום ביחס לקן הישר: y=mx+b.

כאשר נקודה נמצאת מעל הקו הישר: answer =+1 וכאשר הנקודה מתחת לקו הישר: answer=-1.

הימצאות נקודה מעל הקו או מתחת לקו: מהווה סיווג בין 2 קבוצות: למשל אוסף A ואוסף B.

נרצה למצוא את משוואת קו הישר עצמו, "לצייר אותו" על בסיס הנתונים שבידנו ולסווג נקודות.

פרספטרון הוא אלגוריתם למידה "און ליין", משמע הוא לומד תוך כדי ריצה מדגימות שאבחן בזמן פעולתו. הוא אלגוריתם סיווג העובר נקודה נקודה ו"מנחש" אם היא מעל קו הישר או מתחת.

כל פעם שטעה ב"ניחוש", אנו נתקן אותו ואת ווקטור המשקלים (ווקטור המשקלים מהווה כיוון למשוואת הישר) – כך שככל שהאלגוריתם מנוסה יותר (המשקלים מדויקים יותר מאיטרציה לאיטרציה) הוא משיג תוצאות מדויקות יותר.

**חישוב הניחוש** יבוצע על ידי מכפלה סקלרית של ווקטור המשקלים W (שמאותחל לאפס) עם קורדינטות הנקודה. תוצאת המכפלה הסקלרית (sum כלשהו) יעבור לפונקציה sign: אם sum גדול מאפס, האלגו ניחש שהנקודה מעל הקו, אם קטן האלגו ניחש שהנקודה מתחת לאפס.

**החישוב יבוצע כך:**

i=0,1………… k

Sum = w[0]\*P[0] + w[1]\*P[1] +..........w[k-1]\*P[k-1]

לאחר מכן, האלגו משווה את התוצאה שלו עם התוצאה האמתית (זו שנתונה בקלט answer): אם התוצאה לא נכונה, **האלגו יתקן את ווקטור המשקלים** וירוץ מההתחלה על הנקודות מהנקודה הראשונה שקיבל (כלומר חשוב שסדר הנקודות ישמר).

**תיקון ווקטור המשקלים יבוצע כך:**

i=0,1………… k

W[i]+= alpha\*error\*P[i]

כחלק מהקלט: נקבל qc – איכות מקסימלית שלפיה נפעל, אלפא0 ואלפא מקסימלית.

נרצה למצוא את האלפא המינימלית והq המינימלי (שקטן מQC).

נממש את האלגוריתם Perceptron כך שנמצא את האלפא המינימלית מבין התהליכים עבורם האיכות שתתקבל תהיה נמוכה מהאיכות המקסימלית.

**חלוקת התפקידים:**

* **MPI** – הMASTER מחלק את הטווח בין אלפא אפס לאלפא מקס בין ה – SLAVES (כולל לעצמו). כך שכל Process מבצע את האלגוריתם פרספטרון בטווחי אלפאות שונים ומסכמים את תוצאותיהם בסטרקטים. בסוף המאסטר לוקח את האלפא המינימלית שנמצאה.
* **OpenMP** – כל חישוב שהמאסטר מבצע לבדו, למשל מציאת אלפא מינימלית מבין כל התהליכים (כולל עצמו), לולאה שבה המאסטר שולח לכל התהליכים את מערך הנקודות וגודלו.
* **CUDA** – מבצע את החישוב האחרון באלגוריתם פרספטרון – שבו אנו משווים בין ניחוש לבין התשובה האמתית – כדי לדעת מה איכות התוצאה שלנו. הוא "מנחש" לכל נקודה אם היא מעל או מתחת לקו הישר ומחזיר מערך בגודל N של החישובים. ההשוואה עצמה נעשית על ידי openMP – בCPU ולא בGPU.

**פירוט הפתרון:**

* צד ה – MASTER:

1. ה – MASTER קורא מקובץ את הנתונים והנקודות ומכניס אותם לתוך מערך.
2. ה – MASTER שולח ל – SLAVES את ה-limit, qc, alpha0 , alphaMax על ידי MPI\_Bcast.
3. ה – MASTER שולח ל - SLAVES את מספר הנקודות(שימוש בOpenMP).

ומערך הנקודות, טווחי האלפות שכל אחד צריך לחשב (אלפא מינימלית ואלפא מקסימלית), ועל ידי MPI\_Scatter כל salve מקבל struct ספציפי שבו ישמור את המידע הסופי שיצא לו בחישוב (אלפא מינימלית, q שקטן מqc ו-ווקטור משקלים w).

1. ה – MASTER מחכה לקבלת הנתונים מה – SLAVES: על ידי MPI\_Gather ואוסף אותם לתוך מערך outputs בגודל numofPro.
2. ה – MASTER מחפש מבין כל האלפות את האלפא המינימלית (שימוש בOpenMP).
3. ה – MASTER יכתוב לקובץ את האלפא המינימלית, q כלומר האיכות המינימלית שמצא ואת ווקטור המשקלים המתאים שחושב.

* צד ה – SLAVES:

1. מחכים לקבל מה- :MASTER LIMIT qc, k, alpha0,.
2. מחכים לקבלת הנתונים מה – MASTER: מספר הנקודות, מערך הנקודות וטווח אלפאות לחישוב.
3. מבצעים את אלגוריתם פרספטרון, התוצאה נשמרת בתוך struct .
4. המידע מוחזר לMASTER על ידי MPI\_Gather.

**סיבוכיות:**

המאסטר שולח לכל הסלבים נתונים: o(1) בגלל הפרגמה.

כול פרוסס מאתחל לקודה ושומר מערך נקודות: o(1) כי זה קורה במקביל על n טראדים.

נסתכל על הפונקציה Perceptron:

אתחול למשקלים: מכיוון שיש פרגמה וk עצמו מוגבל עד 20, הוא רץ בo(1).

מכילה לולאת while ראשית עם q וטווחי אלפות:

בתוכה יש לולאת for -> שרץ עד limit. O(limit)

בתוכו יש לולאת for -> שרץ עד n. בתוך לולאת הfor הזו יש קריאה לf: שרצה עד k : o(limit\*n\*k)

לאחר מכן ישנה אפשרות לתיקון המשקלים: ששוב רץ עד k: o(limit\*n\*k + k)= o(n\*k)

סה"כ קיבלנו: o(limit\*n\*k).

חישוב nMiss מהקודה מתרחש במקביל: o(1).

מכיוון שכל פרוסס מבצע את האלגוריתם הנ"ל (כל פרוסס מקבל טווח של אלפאות) נקבל:

סה"כ קיבלנו: **o( (limit\*n\*k) / numofpro) )**

**במידה והlimit זניח, K זניח ו-numofpro: נקבל o(n).**