**ממן 14 – מבוא לתורת החישוביות והסיבוכיות – ברנדס איתי**

**שאלה 1:**

נגדיר .

נראה ש. לשם כך נציע מכונת טיורינג דטרמיניסטית המכריעה את השפה בעזרת שימוש בכמות זיכרון לינארית בגודל הקלט.

במילים אחרות, בהינתן גרף לא-מכוון אנו רוצים לבנות מכונת טיורינג שתבדוק האם יש מעגל פשוט באורך שמתחיל ומסתיים ב כלשהו.

ראשית, נגדיר מכונת-טיורינג עזר ,שתקבל כקלט גרף, צומת מקור ואורך מסלול רצוי n. יקבל אם קיים מסלול פשוט (ללא מעגלים) היוצא מצומת המקור ואורכו שווה לn.

= " על קלט כאשר גרף בלתי-מכוון, צומת ב ו מספר טבעי:

1. אם , *קבל*.
2. סמן את הצומת .
3. לכל קשת :
   1. אם לא צומת מסומן, נסמלץ את מכונת הטיורינג עם הפרמטרים

* 1. אם המכונה תקבל, *קבל*.

1. הסר את הסימון מהצומת .
2. *דחה*.

"

ניתן לממש את מכונת הטיורינג באופן יעיל מבחינת מקום, בו אנו שומרים בכל שלב נתון בסרט העבודה את המסלול שאנו בוחנים באיטרציה הספציפית, יחד עם ערך ה הנוכחי. נוכל לממש את הצעד הרקורסיבי באלגוריתם כך: כאשר מסלול חישוב אחד "יכשל" ונגיע לדחייה, נוכל לקדם את באחד, למחוק את האיבר האחרון ברשימה, ולחשב את המסלול האפשרי הבא (לפי הקשת הבאה בשורה 3, בסדר לקסיקוגרפי).

לפיכך, המכונה ניתנת למימוש במקום לינארי.

(\*) בנוסף, אם נתבונן בסרט העבודה של לאחר סיום הריצה, אז אם קיבלה, נוכל לחלץ מסרט העבודה את המסלול שחושב. במילים אחרות, אפשר להגיד שכאשר מקבל, הוא גם מחזיר כפלט את המסלול שחושב.

כעת נפנה להגדרת מכונת-הטיורינג הראשית , שתקבל כקלט גרף ותחזיר האם יש מעגל פשוט באורך שמתחיל ומסתיים ב כלשהו. במילים אחרות, תכריע את השפה .

= " על קלט כאשר גרף בלתי-מכוון:

1. נציב .
2. לכל צומת :
   1. נסמלץ את מכונת הטיורינג עם הפרמטרים .
   2. אם המכונה תקבל, סימן שיש מסלול פשוט באורך מ לצומת כלשהי. כפלט נקבל גם את המסלול שחושב (\*). נתבונן בצומת האחרון במסלול שחושב, ונבדוק האם קיימת קשת . אם כן, הרי שמצאנו מסלול הימלטוני. *נקבל*.
3. *דחה*.

"

שלב 1 במכונה ידרוש מקום לוגריתמי (עלות האחסון של המספר בזיכרון בבינארית היא ).

בשלב 2 נסמלץ את מכונת הטיורינג שכבר הראנו שניתנת למימוש בזמן לינארי. נצטרך גם לזכור את האינדקס הנוכחי בלולאה, אך האינדקס קטן או שווה לn, ולכן כפי שהראנו כבר, עלות אחסונו היא לוגריתמית.

לסיכום, מכונת הטיורינג הדטרמיניסטית מכריעה את השפה בסיבוכיות מקום **לינארית** בגודל הקלט. *ולכן,* .

**שאלה 2:**

יהי .

נראה רדוקציה בעלת זמן ריצה של השפה לשפה .

הרעיון הכללי הוא להוסיף לכל נוסחת CNF כמתי "יש" () בתחילה עבור כל ליטרל בנוסחה.

נראה את סרט העבודה כסרט המחולק ל2 חלקים:

1. חלק המכיל את כמתי ה- יחד עם הליטרלים שלהם. יהיה ריק בתחילה.
2. חלק המכיל את נוסחת ה המקורית מהקלט.

נוכל להפריד בין החלקים ע"י תו @ לדוגמה.

= "על קלט כאשר היא נוסחת :

1. בראשית, הראש הקורא-כותב מצביע על התחלת הקלט. נסיט את כל הקלט במקום אחד ימינה, ונציב @ בראש הסרט (ע"מ ליצור חלוקה בין 2 החלקים).
2. נזיז את הראש הקורא-כותב ימינה עד שנגיע לסוף הקלט (סימן רווח). בכל תזוזה ימינה נבדוק האם נתקלנו בליטרל (ז"א, בתו שאיננו בקבוצה ). אם כן:
   1. נסמן אותו.
   2. נזיז את כל החלק השני (מהתו @ עד לתו רווח) ב2 מקומות ימינה. את התזוזה נבצע מהסוף להתחלה.
   3. ב2 מקומות שלפני התו @ נכתוב כאשר הוא הליטרל המסומן.
   4. נחזיר לליטרל המסומן.
   5. נמחק את הסימון ונעבור לתו הבא.
3. נחזיר את תוכן סרט העבודה. "

נכונות:

הרדוקציה מוסיפה לנוסחת ה כמתי "יש" () בתחילה עבור כל ליטרל בנוסחה, ז"א בהינתן נוסחה הרדוקציה תייצר .

ברור מההגדרה שהנוסחה ספיקה אמ"מ הנוסחה ספיקה: *נוסחת ספיקה אמ"מ קיימת השמה כלשהי של ליטרלים שתספק את הנוסחה. זוהי בדיוק ההגדרה של המשפט הבוליאני שבנינו.*

המכונה לעיל עלולה ליצור כמתים עודפים, שכן יתכן שבנוסחה המקורית, ליטרל ספציפי מופיע בכמה פסוקיות שונות, אך כמובן שזוהי איננה טעות ועדיין מייצר משפט בוליאני אמיתי תקין.

ניתוח סיבוכיות זמן:

יהי .

שורה 1 רצה בזמן לינארי: יש להסיט תווים ימינה. כל הסטה מתבצעת בזמן קבוע.

בשורה 2 אנו רצים בלולאה על תווי קלט, כאשר על כל אחד מהם, במקרה הגרוע, אנו מבצעים הסטה של כל הקבוצה השנייה (בעלות לינארית), כותבים תווים בקבוצה הראשונה (בסיבוכיות לינארית, כשלוקחים בחשבון את התזוזה של הראש הקורא-כותב על כל הקלט) וחוזרים לתו המסומן (בסיבוכיות לינארית גם כן) ואז מבצעים עבודה קבועה.

בסה"כ בשורה 2 אנו במקרה הגרוע רצים n פעמים ומבצעים פעולות בכל פעם. לכן, זמן הריצה הכולל הוא . **לכן, סיבוכיות זמן הריצה היא** , כנדרש.

**ובכך הראנו רדוקציה בעלת זמן ריצה של השפה לשפה .**

**שאלה 3:**

נתונה השפה:

**סעיף א':**

נראה ש:

בהוכחת משפט 5.9 (ע"מ 223 בספר הלימוד) מופיע האלגוריתם המכריע את . האלגוריתם מסמלץ את המכונה על הקלט למשך צעדים, או עד שהיא עוצרת. אם המכונה עצרה תוך צעדים, האלגוריתם מחזיר את הערך שהחזירה. אחרת, האלגוריתם דוחה. בספר הוכחה נכונות האלגוריתם.

לצורך הסימולציה, נצטרך לשמור בסרט העבודה את:

* המיקום הראש הקורא-כותב על הסרט שאנו מסמלצים. ולכן כמות הזיכרון הדרושה למיקום הראש היא לוגריתמית במקרה הגרוע (עלות האחסון של המספר בזיכרון בבינארית היא ).
* המצב שאנו נמצאים בו כרגע – בעלות מקום של .
* מונה הצעדים – בעלות מקום של . נבחין שמתקיים:

כמו כן נבחין שהקלט שלנו איננו רק אורך המילה w (שהיא n בהוכחה שלי), אלא גם אורך הייצוג הטקסטואלי של האוטומט החסום-לינארית , שמכיל בתוכו גם את המצבים ואת התווים באלפבית.

לפיכך, כל אחד מהמשתנים שעלינו לשמור בסרט העבודה גודלו פולינומיאלי בגודל הקלט.

**ולכן, .**

**סעיף ב':**

יהי . נראה רדוקציה חשיבה פולינומיאלית של ל-.

ולכן קיימת מכונת טיורינג דטרמיניסטית המכריעה את השפה בסיבוכיות מקום של עבור טבעי כלשהו.

הרעיון הכללי הוא "לנפח" את מילת הקלט למילה באורך עם ערכים שהמכונה תתעלם מהם, ואז מכונת הטיורינג (\*), שמכריעה את השפה בסיבוכיות מקום של , תהיה למעשה אוטומט חסום-לינארית, ואז יכריע את השפה בהצלחה.

נגדיר הרדוקציה הבאה:

*" על קלט כאשר מכונת טיורינג המתוארת ב(\*), ו w מילה:*

1. *נגדיר את באופן זהה ל מלבד השינויים הבאים:*
   1. *נוסיף לאלפבית את הסמל @ (אם הוא כבר קיים, נוסיף סמל אחר שאיננו קיים).*
   2. *בפונקציית המעברים , לכל התייחסות לתו רווח, נוסיף גם התייחסות לתו @.*
2. *נגדיר את להיות w בתוספת ריפודי סמלי @ מימין עד שגודל המילה יהיה .*
3. *נחזיר את .*

*אם , אז נבחין שהמכונה תקבל את , שכן השינוי היחיד שלה מM המקורית הוא שהיא מתייחסת גם לסימני @. אך היא מתייחסת אליהם כסימני רווח (ז"א, סוף קלט), לכן גם הוספת סימני @ בסוף המילה לא ישנו את תשובת עליה. לכן, אם מקבל את , אז הוא יקבל גם את . המכונה משתמשת בסרט עבודה בגודל לינארי בגודל הקלט שלה, ולכן היא אוטומט חסום-לינארית. לפיכך, מתקיים .*

*אם אז בפרט , ו*סיבוכיות המקום היא אכן  *(במקרה זה היא אף לינארית), כנדרש.*

זמן הריצה של שלב 1 הוא לינארי בגודל הקלט (אנו יוצרים העתק של המכונה M ואז יוצרים מעברים כפולים – במקרה הגרוע אנו מכפילים את מספר המעברים ב2. בכל אופן זמן הריצה נשאר לינארי).

זמן הריצה של שלב 2 הוא , כאשר קבוע טבעי. נבחין ש מכיל בתוכו את אורך וגם את הקידוד של המכונה . לכן יהיה נכון להגיד שזמן הריצה של שלב 2 מקיים .

ולכן, זמן הריצה של הרדוקציה הוא , ולכן חשיבה בזמן פולינומיאלי, כנדרש.

בכך הראנו רדוקציה חשיבה פולינומיאלית מתאימה של ל-. **לפיכך, מתקיים .**

**סעיף ג':**

בסעיף א' הראנו ש . בסעיף ב' הראנו שכל שפה ב ניתנת לרדוקציה בזמן פולינומיאלי ל. לפיכך, לפי ההגדרה, היא שפה -שלמה.

**שאלה 4:**

נראה שהמחלקה סגורה לפעולת השרשור (concatenation).

נניח שיש לנו 2 מכונות טיורינג שמכריעות 2 שפות בL. נוכל להוכיח את הטענה ע"י כך שניצור מכונה שתנסה "לשבור" את מילת הקלט ל2 חלקים כלשהם, ותיתן כל חלק למכונת טיורינג אחרת. אם שתיהן יקבלו את חלק הקלט, הרי שהקלט למעשה קיים בשפת השרשור של 2 השפות. ניתן לממש מכונה זו בסיבוכיות מקום לוגריתמית, שכן 2 המכונות המקוריות זקוקות לסיבוכיות מקום לוגריתמית.

תהי ו שתי שפות ב המוכרעות בסיבוכיות זמן לוגריתמית ע"י המכונות ו-, בהתאמה.

*נגדיר מכונת טיורינג דטרמיניסטית שמכריעה את שפת השרשור של השפות ו-:*

*M = "על קלט כאשר מילה:*

1. *נציב .*
2. *לכל :*
   1. *נסמלץ ריצה על עם הקלט .*
   2. *נסמלץ ריצה על עם הקלט .*
   3. *אם 2 המכונות קיבלו,* קבל*.*
3. דחה*.*

לצורך הסימולציות, נצטרך לשמור בסרט העבודה את:

* אחסון אורך המילה, , היא לינארית, ולכן כמות הזיכרון הדרושה למיקום הראש היא לוגריתמית (עלות האחסון של המספר בזיכרון בבינארית היא ).
* אחסון המונה בלולאה גם הוא דורש מקום לוגריתמי.
* את מיקום הראש הקורא-כותב על הסרט שאנו מסמלצים (ב או ).

נבחין שאין צורך בסרט נוסף, או להעתיק את הקלט לחלק אחר, אלא ניתן להציב תו @ בין ל , וכך נוכל בסימולציה של להבחין מתי נגמר הקלט, ובסימולציה של נוכל להבין מתי מתחיל הקלט (אם התו @ איננו נמצא באלפבית, אם כן, נבחר בתו אחר).

לפיכך, נוכל להשתמש בסרט המקורי, ולא נצטרך להגדיר חדשים.

כל השלבים ניתנים לחישוב בסיבוכיות מקום לוגריתמית, ולכן ניתן לממש את המכונה בסיבוכיות מקום לוגריתמית, ומתקבל ש . בנוסף לכך, הראנו בהסבר הקודם שהמכונה מכריעה את שפת השרשור של השפות ו-, שאלה הן שפות כלשהן ב.

**לפיכך, שהמחלקה סגורה לפעולת השרשור (concatenation).**

**שאלה 5:**

נראה שניתן להכריע את השפה בשימוש בכמות זיכרון לוגריתמית. כשנראה זאת, נוכל להגיד ש .

נניח שבסרט העבודה, קיים התו "#" לסימון תחילת הקלט, לאחריו מילה , ולאחריהם סימני רווח שיסמנו את סימון סוף הקלט.

נמנה מספר כללים שיקיימו שייכות של המילה לשפה :

1. *האלפבית הוא .*
2. *מספר תווי ה"(" ומספר תווי ה")" שווה.*
3. *בכל התת מחרוזות כאשר מתקיים שמספר תווי ה")" גדול או שווה למספר תווי ה"(".*
4. *לא קיים הצירוף "()".*
5. *לכל אחד מהתווים אסור שיהיה במקום הקודם לו אחד מן התווים (# הוא תו תחילת הקלט).*
6. *לכל אחד מהתווים אסור שיהיה במקום העוקב לו אחד מן התווים (רווח הוא תו סיום הקלט).*
7. *לפני תו ")" אסור שיהיה מספר.*
8. *לאחר תו "(" אסור שיהיה מספר.*

*כלל 1 מוודא שהאלפבית תקין. כללים 2 ו3 מוודאים תקינות סוגריים, ז"א, שלכל תו ")" ניתן למצוא תו "(" חוקי. בכלל 2 אנו מוודאים שיש מספר תווי "(" ו")" זהים, ובכלל 3 אנו מוודאים שלא הגענו למצב שיש תו "(" שלא ניתן להתאים לו תו ")" (משום שכל תווי ה")" הקיימים כבר תפוסים במצב זה).*

*כלל 4 מוודא שלא קיים הצירוף הלא תקין "()", וכללים 5-6 מוודאים שאין אופרטור מתמטי כלשהו שעטוף באופרטור מתמטי אחר, בסוגריים, או בתו התחלת/סוף קלט, שכן גם זהו מצב לא תקין. כללים 7-8 מוודאים שאין מספרים לפני/אחרי סוגריים.*

*כל שאר המחרוזות הן מחרוזות המבטאות ביטויים אריתמטיים של מספרים שלמים אי-שליליים שבהם הסוגריים תקינות.*

*נגדיר מכונת טיורינג דטרמיניסטית שמכריעה את :*

*M = "על קלט כאשר מילה:*

1. *נגדיר (מונה ה"עומק" הנוכחי, שיוגדר כמספר תווי ה")" פחות מספר תווי ה"(" שנקראו עד כה).*
2. *נסרוק את סרט הקלט מהתחלתו עד לסופו (עד לקריאת רווח, המסמל את סוף הקלט), ולכל תו שנקרא:*
   1. *נבדוק האם אם לא,* נדחה*. (כלל 1)*
   2. *אם , נגדיל את באחד. (כללים 2 ו-3)*
   3. *אם , נפחית את באחד. אם ,* נדחה*. (כללים 2 ו-3)*
   4. *אם , נבדוק האם התו שבא אחריו בסרט הקלט הוא "(". אם כן,* נדחה*. (כלל 4)*
   5. *אם , נבדוק האם התו הקודם לו נמצאים ב . אם כן,* נדחה*. (כלל 5)*
   6. *אם , נבדוק האם התו העוקב לו נמצאים ב . אם כן,* נדחה*. (כלל 6)*
   7. *אם , נבדוק האם התו שלפניו נמצא ב. אם כן,* נדחה*.*
   8. *אם , נבדוק האם התו שאחריו נמצא ב. אם כן,* נדחה*.*
3. *אם ,* נקבל*. אחרת,* נדחה*. (כלל 2)*

*נכונות:*

*בתחילת הפיתרון הצגנו מדוע כללים אלה אכן מגדירים שייכות לשפה .*

*בפתרון אנו משתמשים במונה ע"מ לספור את ה"עומק" הנוכחי, שמאותחל ל0. כשאנו נתקלים ב) אנחנו מגדילים את העומק ב1, וכשנתקלים ב( מפחיתים ב1, תוך כדי בדיקה שהעומק לא נהיה שלילי (ואז כלל 3 נשבר). בשורה 3 אנו מוודאים שבסוף הריצה העומק הסופי הוא אכן 0, שכן אחרת פירוש הדבר שיש תו ) שאין לו תו ( מתאים.*

*חישוב סיבוכיות מקום:*

נצטרך לשמור בסרט העבודה את:

* המונה , שגודלו במקרה הגרוע לינארי, ולכן כמות הזיכרון הדרושה לשמירתו היא לוגריתמית (עלות האחסון של המספר בזיכרון בבינארית היא ).
* עבור אנו זקוקים למקום קבוע. לשמירת התו העוקב/הקודם בכללים 4-8 אנו זקוקים גם לסיבוכיות קבועה.
* עבור קבוצות התווים מולם עלינו לבדוק את כללים 1 ו5-8 ניתן לשמור בכמות זיכרון קבועה. כמו כן, גם את האינדקס הדרוש ללולאה עליהם ניתן לשמור בכמות זיכרון קבועה.

*לכן, למימוש המכונה אנו זקוקים לכמות זיכרון לוגריתמית בגודל הקלט.*

***לפיכך, מתקיים , כנדרש.***

**שאלה 6:**

יהי .

עלינו להראות ש היא שפה -שלמה. לשם כך עלינו להראות ש וגם שכל שפה ב ניתנת לרדוקציית מקום לוגריתמית אל .

נבחין שבמשפט 8.25 בספר הלימוד (ע"מ 353) הוכח ש היא שפה -שלמה, ולכן בפרט כל שפה ב ניתנת לרדוקציית מקום לוגריתמית *אליה.*

*לכן אם נראה שהשפה PATH ניתנת לרדוקציית מקום לוגריתמית אל , לפי ההדרכה, נקבל ודאי שכל שפה ב ניתנת* לרדוקציית מקום לוגריתמית אל .

**נראה ש:**

*נגדיר מכונת טיורינג א-דטרמיניסטית שמכריעה את ע"י סימולציה:*

*M = "על קלט כאשר אוטומט סופי א-דטרמיניסטי ו מילה:*

1. *נציב .*
2. *נגדיר את כמצב הסימולציה הנוכחי. נאתחל אותו בתור של האוטומט .*
3. *נגדיר את כמספר התו שראש האוטומט נמצא עליו. נאתחל אותו להיות 1.*
4. *כל עוד :*
   1. *נגדיר .*
   2. *נבחר באופן א-דטרמיניסטי מעבר מן האפשרויות בפונקציית המעברים המתקבלות בקריאת או ממצב . אין מעבר כזה,* דחה*.*
   3. *לפי התשובה שחוזרת מפונקציית המעברים:*
      1. *נגדיר את להיות המצב החדש.*
      2. *אם קראנו את התו , נקדם את באחד.*
5. *אם נמצא בקבוצת המצבים המקבלים של האוטומט ,* קבל*. אחרת,* דחה*.*

לצורך הסימולציות, נצטרך לשמור בסרט העבודה את:

* אחסון אורך המילה, , היא לינארית, ולכן כמות הזיכרון הדרושה למיקום הראש היא לוגריתמית (עלות האחסון של המספר בזיכרון בבינארית היא ).
* אחסון מספר המצב יתבצע בסיבוכיות מקום של .
* אחסון המונה בלולאה גם הוא דורש מקום לוגריתמי.
* עבור אנו זקוקים למקום קבוע.

נבחין ש מכיל בתוכו את אורך וגם את הקידוד של המכונה . לכן, המקום הדרוש בסרט העבודה של הוא ודאי לוגריתמי בגודל הקלט. כמו כן, ברור ש, שכן אנו מסמלצים את האוטומט על הקלט .

לפיכך, .

**נראה ש:**

יהי .

נגדיר הרדוקציה הבאה:

*" על קלט כאשר גרף מכוון, ו צמתים:*

1. נגדיר אוטומט סופי א-דטרמיניסטי כאשר:
   1. אלפבית האוטומט יהיה .
   2. לכל צומת בגרף ניצור מצב מתאים באוטומט.
   3. המצב ההתחלתי יהיה (המצב הנוצר מהצומת s).
   4. קבוצת המצבים המקבלים תהיה (המצב הנוצר מהצומת t).
   5. את פונקציית המעברים נגדיר כך:
      1. לכל , נגדיר מעבר ממצב למצב בקריאת 0.
      2. בנוסף, במצב המקבל, , נגדיר גם מעבר אל עצמו בקריאת 0.
2. *נגדיר את המילה להיות .*
3. *נחזיר את .*

*"*

נכונות:

אם , אז קיים מסלול מכוון מ-s ל-t בגרף . נבחין שקיים מסלול שכזה ללא מעגלים (אם הוא מכיל מעגלים ניתן למחוק את המעגל ולקבל מסלול ללא מעגלים). לכן אורך המסלול אינו גדול מ, כאשר (מספר הצמתים). לכל קשת בגרף קיים מעביר מתאים בטבלת המעברים ב, והאוטומט מתחיל מהמצב המתאים לצומת s ומצבו המקבל היחיד הוא המצב המתאים לצומת t. מתקבל ש יקבל את המילה , שכן אכן קיימת שרשרת מעברים המתחילה מ ונגמרת ב באורך קטן או שווה ל. נבחין שקיים לנו מצב מלכודת מקבל במצב המקבל, ולכן גם אם אורך המסלול קטן ממש מ, עדיין נשאר במצב מקבל (ברגע שהצלחנו להגיע מs לt תוך צעדים, האוטומט מקבל). ולכן, האוטומט מסיים במצב מקבל, ומתקבל .

אם אז המילה מביאה את האוטומט מהמצב ההתחלתי למצב המקבל . לכן, לפי אופן הבניה, קיים בגרף מסלול מכוון מ-s ל-t. ולכן, .

סיבוכיות מקום:

לצורך הרדוקציה, נצטרך לשמור בסרט העבודה את:

* מצבי , בסיבוכיות מקום של .
* המצב ההתחלתי, המצבים המקבלים, והאלפבית של , בזמן קבוע.
* פונקציית המעברים , שהיא העתקה של הקשתות של (בתוספת קשת אחת) ולכן סיבוכיות המקום היא .
* לבניית המילה נצטרך לספור את הצמתים , בסיבוכיות מקום של .
* בנוסף, נצטרך אינדקס ע"מ לעבור על הצמתים השונים, הן בשלב 1 והן בשלב 2, וגם עבורו נזדקק לסיבוכיות מקום של .

סיבוכיות המקום היא לוגריתמית לגודל הקלט.

כמו כן, ברור שהפונקציה חשיבה, שכן כל הצעדים בה סופיים ואין לולאות אינסופיות.

בכך הראנו רדוקציית מקום לוגריתמי חשיבה של ל-. **לפיכך, מתקיים .**

**ולכן, היא שפה -שלמה.**