

דו"ח תרגיל 5 – עיבוד שפות טבעיות

מגיש: שגיא גוילי, ת.ז. 203638804

אתאר את תהליך מימוש האלגוריתמים שהתבקשו בתרגיל; ראשית את אלגוריתם המעבר של דקדוקים חסרי הקשר (CFG- Context Free Grammers) מהצורה הכללית לצורה הנורמלית של חומסקי (CNF – Chomsky Normal Form) לאחר מכן את אלגוריתם מציאת העץ פריסה (Parse Tree) הטוב ביותר עבור משפטי קלט כלשהם, אם קיים כזה (אם לא התוכנית תדפיס לקובץ הפלט את המשפט עם כיתוב not found כפי שהתבקש)

אלגוריתם המעבר מ-CFG ל-CNF

המטרה באלגוריתם זה הוא להפוך כל צורת CFG כללית לצורה הנורמלית של חומסקי (ללא חוקי אפסילון) שנראת כך:

$S \rightarrow AB$

$A \rightarrow a$

בשלב האתחול, האלגוריתם מקבל רשימה של חוקים ושומר אותם במילון (ישמש בחלק השני של התרגיל על מנת למחוק חוקים שנוספו במהלך האלגוריתם כאשר נדפיס את העץ הטוב ביותר) לאחר מכן, בהתאם להוראות התרגיל לפיהן הנון טרמינל בחוק הראשון מהווה את נון טרמינל ההתחלה, האלגוריתם בודק האם נון טרמינל הפתיחה מופיע באיזהו חוק בחלק הימני- אם נוסף חוק $S' \rightarrow S$, כאשר S הוא נון טרמינל הפתיחה המקורי, בכדי שנוכל לדעת תמיד מה החוק הראשון שממנו מתחיל הדקדוק.

השלב הראשון באלגוריתם הוא הסרת חוקי יחידה (לא מעגליים שכן ניתן להניח שבקלט לא יופיעו כאלה), כלומר חוקים המכילים נון טרמינל אחד בלבד בצד ימין, דבר המהווה הפרה של צורה הנורמלית של חומסקי. האלגוריתם עובר בלולאה על כלל החוקים עד אשר לא ימצא בהם עוד חוק יחידה כלשהו, הצורך בלולאה הוא משום שאחרי כל מעבר על החוקים יכול להיווצר חוק יחידה נוסף (אם למשל חוק יחידה גזר חוק יחידה אחר) שמעבר הנוכחי לא נוכל לאתר ולכן במעבר הבא הוא ינוקה. מחיקת חוקי היחידה עובדת בצורה הבאה- כאשר נמצא חוק שהוא חוק יחידה אנו מוחקים אותו מרשימת החוקים ולאחר מכן משאירים את הנון טרמינל השמאלי לגזור את כלל החוקים שגזר הנון טרמינל מצד ימין, דוגמא להמחשה:

$A \rightarrow B$

$B \rightarrow \text{moshe}$

$B \rightarrow \text{yossi}$



$A \rightarrow \text{moshe}$

$A \rightarrow \text{yossi}$

השלב הבא מחיקת חוקים עם יותר משני נון טרמינלים בצד ימין, כמו הסרת חוקי היחידה גם חוק שיש בו יותר משני נון טרמינלים בצד ימין מהווה הפרה של הצורה הנורמלית של חומסקי. הפתרון שלנו למצב זה הוא לעבור על כל הנון טרמינלים בצד ימין של החוק והחל מהנון טרמינל השני להחליף את הנון טרמינל המקורי בנון טרמינל חדש, למחוק את כל שאר הנון טרמינלים שבאים אחריו ולהוסיף אותם לחוק חדש שבו הנון טרמינל החדש יהיה מצד שמאל והנון טרמינלים הנותרים (כאשר הנון טרמינל המקורי הוא הראשון) יהיו בצד ימין. בכל שלב נגדיר מחדש את המשתנה left_side בקוד כך שיהיה הצד השמאלי של החוק אותו כרגע הוספנו בכדי שנוכל להמשיך באותה פעולה עד אשר נגיע לחוק חדש שיש בו רק שני נון טרמינלים מצד ימין. דוגמא להמחשה:

$A \rightarrow BCDE$

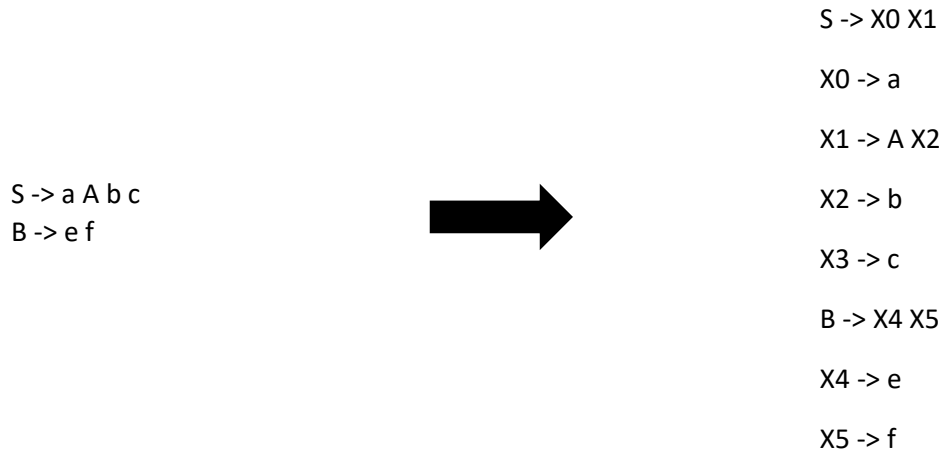


$A \rightarrow B \text{ NewNon0}$

$\text{NewNon0} \rightarrow C \text{ NewNon1}$

$\text{NewNon1} \rightarrow DE$

השלב האחרון הוא מחיקת טרמינלים מחוקים בהם בצד ימין יש נון טרמינל וטרמינל ביחד או יותר מטרמינל אחד. גם דבר זה מהווה הפרה של הצורה הנורמלית של חומסקי. הפתרון למצבים מסוג זה הוא לעבור על כל חוק, לבדוק האם הוא מקיים את ההפרה הנ"ל, אם כן לעבור על כל טרמינל שמופיע בצד ימין, ליצור עבורו נון טרמינל חדש שיגזור אותו, להחליף את המופע של הטרמינל בנון טרמינל החדש שיצרנו ולהוסיף חוק שבו הנון טרמינל החדש גוזר את הטרמינל שזה עתה החלפנו. דוגמא להמחשה:



ישנם מקרים בהם נוצרים לנו בשלב זה חוקי יחידה חדשים, לכן בסוף השלב נפעיל שוב את שלב מחיקת חוקי היחידה.

בסוף כל השלבים יכול להיות מצב בו ישארו לנו חוקים ללא שימוש, נפלט את החוקים הללו ולאחר מכן נדפיס את הדקדוק החדש לתוך קובץ הפלט שמתקבל בארגומנט השני לתוכנית ונשמור את כלל החוקים המילון לחדש בשביל אלגוריתם ה-CYK ההסתברותי שבחלק השני של התרגיל.

אלגוריתם CYK ההסתברותי

המטרה באלגוריתם זה היא לקבל חוקים בצורה הנורמלית של חומסקי בצירוף משפט כלשהוא ולהחזיר את עץ הגזירה הטוב ביותר של המשפט מבחינת ההסתברות שלו. בשלב הראשון מאותחלים 3 טבלאות:


טבלת backpointers - מטרתה לעזור למצוא אילו טרמינלים או נון טרמינלים נגזרים כאשר אנחנו מוצאים חוק גזירה במהלך המעבר על המילים במשפט. הטבלה תאותחל לאיברים ריקים

טבלת Nodes - הטבלה תחיל משתנים מסוג node כאשר Node הוא בעצם עץ שהשורש שלו הוא הנון טרמינל הראשון בחוקים. כל Node מכיל את הנון ימני והנון השמאלי שלו (כלומר את הטרמינלים או נון טרמינלים שהוא גוזר) ובמקרה של טרמינלים גם את הנון טרמינל שגוזר אותם (עוזר בהדפסה של העץ גזירה). הטבלה תאותחל לאיברים ריקים.

טבלת הסתברויות - מטרת טבלה זו היא לשמור את ההסתברות של כל טרמינל או נון טרמינל שנמצא במהלך הריצה של ה-CYK ולפי מה שנלמד בהרצאה יש צורך להשתמש בהסתברויות של החוקים שגזרו כל טרמינל או נון טרמינל לכן במהלך הריצה נעזר בהסתברויות הללו כדי לחשב הסתברויות של חוקים שנמצא תוך כדי הרצת האלגוריתם. הטבלה תאותחל עם 0 בכל איבר.

בשלב הבא, האלכסון הראשי בכל טבלה מאותחל עם הערכים המתאימים של הטרמינלים, כלומר עם המילים במשפט; באלכסון של טבלת ה-backpointers נשמרים הנון טרמינלים שגוזרים כל אחת המילים, באלכסון של טבלת ה-Nodes נוצר Node לכל טרמינל שמכיל את ערך הטרמינל ואת הנון טרמינל שגוזר אותו (ללא בנים כלומר Nodes אלה יהיו העלים בעץ) ובאלכסון של טבלת ההסתברויות נשמרת ההסתברות של כל טרמינל, ההסתברות מחושבת באופן שתואר בפורום- 😊 סכום ההסתברויות לכל החוקים עם אותו

סמל שמאלי תהיה אחידה ותסתכם ל-1, כלומר התפלגות אחידה בין מספר הטרמינלים ונון טרמינלים בצד ימין של כל חוק שאנחנו מקבלים במהלך האלגוריתם.

בשלב שלאחר מכן, נרוץ על כל תא של ייצוג של מילה בטבלאות, בדומה לדרך שהוצגה בהרצאה, כלומר עבור המילה הראשונה לא נעשה דבר (כי היא על האלכסון והוא כבר אותחל) עבור כל מילה שתבוא לאחר מכן נחשב את הסיכוי של המילה להופיעה בין מהמקום ה-0 עד המקום ה- i בו היא נמצאת, נמשיך לבדוק מהמקום ה- j עד i לכל $j < i$. בכל פעם שנמצא חוק שגוזר מילה או שתי מילים (לא יותר כי הצורה של חומסקי כל חוק גוזר שני נוןטרמינלים או טרמינל אחד) נחשב את ההסתברות של החוק על ידי הנוסחה שראינו בהרצאה- הסתברויות הנון טרמינלים או טרמינלים שנגזרים מהחוק כפול ההסתברות של החוק עצמו, שהיא מתפלגת אחיד ביחס למספר החוקים שהצד השמאלי של החוק (בדומה למתואר ב-).

נבדוק האם ההסתברות α שקיבלנו באיטרציה הנוכחית גדול מההסתברות שקיימת כרגע בתא ה- ij בטבלת ההסתברויות- אם כן נחליף את הערכים בתאים המתאימים בכל הטבלאות, כאשר בטבלת ההסתברויות נשים את ההסתברות α .

בסוף התהליך נקבל אם התא ה- ij בטבלת backpointers לא ריק (או שונה מהנון טרמינל הראשון של החוקים שקיבלנו בקלט), כלומר נדפיס את העץ המתקבל. אם לא אז נציג את משפט הקלט בצירוף הודעה .not found

כפי שתואר בהרצאה, העץ המתקבל יהיה העץ הטוב ביותר שכן בכל איטרציה שבה נמצאה גזירה המתאימה לחוקים שמרנו את ההסתברות המקסימלית מבין כלל הגזירות האפשריות עבור אותם נון טרמינלים או טרמינלים.

השלב האחרון באלגוריתם הוא הדפסת עץ הגזירה בפורמט שתואר בהוראות לתרגיל, אנחנו מתחילים מהשורש וברקורסיה עוברים על כל Node שהשורש פורש, כאשר בכל איטרציה מדפיסים נון טרמינלים או טרמינלים. הנון טרמינלים נבדקים על ידי המילון שהוכן בתחילת האלגוריתם בחלק הראשון של התרגיל- אם הם היו קיימים בחוקים המקוריים (לפי המעבר ל-CNF) מדפיסים אותם אחרת מחליפים אותם בנון טרמינלים שהם החליפו במהלך הריצה של שני החלקים.