**יבש 2 – NLP**

מגישים:

אמיר בלדר – 204179659

רועי גנץ – 204506349

**שאלה 1**

אם נבחן את המשפט ואת סט החוקים שקיבלנו, נשים לב כי ישנן רק 2 אפשרויות ל-parse trees נכונים. שמנו לב לכך תוך כדי הבנייה של העצים bottom-up, כשבעצם התחלנו מהעלים. סט החוקים הנתון הינו כזה שלא מאפשר בחירות מרובות. אנחנו מניחים שהסיבה לכך היא הכמות הלא כל כך גדולה של חוקים במקרה הזה, והעובדה שהמשפט עצמו הינו מאוד קצר. לפי הגדרת השאלה עלינו להסתכל על עצים שההסתברות לקבלם היא C כלשהו, חיובי ממש, ועל כן, בדקנו את שתי האפשרויות שלנו.

העץ הראשון:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S | | | | | | | | | |
| NP | VP | | | | | | | | |
| Vt | NP | | | | | | | |
| NP | | | | | PP | | |
| NP | | CC | NP | | IN | NP | |
| DT | NN | DT | NN | DT | NN |
| John | saw | The | cat | And | The | Dog | with | the | Mouse |

נחשב את ההסתברות של העץ הזה:

העץ השני:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S | | | | | | | | | |
| NP | VP | | | | | | | | |
| Vt | NP | | | | | | | |
| NP | | CC | NP | | | | |
| NP | | PP | | |
| IN | NP | |
| DT | NN | DT | NN | DT | NN |
| John | saw | The | cat | and | The | Dog | with | The | Mouse |

עתה נחשב את ההסתברות של העץ הזה:

אם כן, ישנם רק שני עצים אפשריים ולשניהם יש את אותה ההסתברות, . מש"ל.

**שאלה 2**

נסתכל על הנתון במקרה הזה, לפיו האורך של כל ענף ימני בעץ הינו בהכרח 1, כפי שהוגדר בשאלה, "directly into a terminal symbol". על כן, אם נסתכל על משפט, ועל המילה ה-j במשפט.  
הנוסחא המקורית הינה:



עתה נשתמש בנתון של השאלה, ונבין כי הביטוי שלנו בעצם ישתנה. אנו יודעים תמיד כי הענף הימני עתיד להיות terminal symbol. אם כן, אם קודם הסתכלנו על ביטוי כללי , עתה אנו יודעים בהכרח שמה שיבחר הוא terminal symbol עבור הענף הימני, לכן בעצם נוכל להסתכל על אוטומטית. אם נסמן את המילה ה-j במשפט ב- , אז בעצם מדובר . עתה נציב זאת בתוך המשוואה המקורית ונקבל:

אם כן, זוהי בעצם הנוסחא הרקורסיבית החדשה שלנו, כאשר המעבר במקרה הזה בין השורה הראשונה לשנייה, ובין השורה השנייה לשלישית, נובע ישירות מההסבר לעיל.

**שאלה 3**

במקרה הזה, סט החוקים שקיבלנו איננו זהה לסט החוקים הרגיל של Chromsky, זאת בשל החוק של , שחורג ההגדרה של Chromsky, לפיה כל חוק מתפצל לשני חוקים, מלבד terminal symbol.   
ההגדרה השונה הזו של חוקים גורמת בעצם לשינוי באלגוריתם הרקורסיבי שלנו.  
בעצם בהנתן יהיה עלינו להבין אם לשנות את התיוג של המילה המדוברת יוכל לשפר את עץ ה-parse שלנו, או לא. גם טבלת ה-bp שלנו תצטרך להשתנות ועתה להכיל אופציה נוספת עבור .  
אם כן, כך יראה חוק העדכון הרקורסיבי החדש שלנו:

עתה נתאר את ריצת האלגוריתם, יחד עם כלל העדכון החדש שלנו.

אתחול –

כלומר בחירת שההסברות עבורם איננה אפס.

המקרים שהיו מקבלים הסתברות גדולה מאפס גם במדיניות Chromsky:

ובעזרת החוק החדש נקבל עוד כמה אפשרויות:

*עתה, נציג את האיטרציות השונות של האלגוריתם, כאשר בכל איטרציה נציג אח ורק את המקרים שעבורם לא נקבל הסתברות אפס:*

*עבור , כל תתי-המשפטים באורך 2:*

*עבור , כל תתי-המשפטים באורך 3:*

*עבור , כל תתי-המשפטים באורך 4:*

*עבור , כל תתי-המשפטים באורך 5:*

במשפט המקורי ישנן חמש מילים, ועל כן אין יותר איטרציות.

העץ הסופי שיתקבל הינו:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S | | | | |
| NP | VP | | | |
| NNP | VP | | PP | |
| Vt | NP | IN | NP |
| NNP | NNP |
| John | saw | Mary | with | Sally |

נשים לב שהוא מעט שונה מעץ הזהב שנתון בשאלה.

תחילה נבחן במה צדקנו ובמה טעינו:

|  |  |
| --- | --- |
| Rules used in gold-standard | Rules found by our algorithm |
| S🡪NP VP | S🡪NP VP |
| NP🡪NNP (John) | NP 🡪 NNP (John) |
| NNP🡪John | NNP🡪John |
| VP🡪Vt NP | VP 🡪 VP PP |
| Vt🡪saw | VP 🡪 Vt NP |
| NP🡪NP PP | Vt 🡪 saw |
| NP🡪NNP (Mary) | NP 🡪 NNP (Mary) |

באדום, המקומות שטעינו בהם לעומת עץ הזהב.

עתה, נחשב את ה-precision ואת ה-recall שלנו בעזרת בחינת התוצאות כפי שתואר בהרצאה ובתרגול.

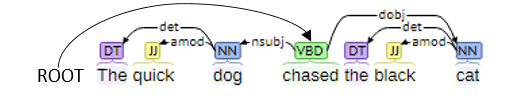
בעצם קיבלנו 12 חוקים בסה"כ, עתה, נמשיך בחישוב:

ועל כן:

ו-F שנקבל הינו:

**שאלה 4**

המשפט וה- parse tree הינם כדלקמן: זה די שונה מהרפרנס – אשמח אם תוודא. אני עבדתי בstack אבל נראה שהוא בין הstack לbuffer.



להלן המצבים השונים ב-Malt Parser:

|  |  |
| --- | --- |
|  | Step 0 - init |
|  | Step 1 - shift |
|  | Step 2 - shift |
|  | Step 3 - shift |
|  | Step 4 – left arc |
|  | Step 5 – left arc |
|  | Step 6 – shift |
|  | Step 7 – left arc |
|  | Step 6 - shift |
|  | Step 6 – shift |
|  | Step 7 - shift |
|  | Step 7 – left arch |
|  | Step 8 – left arch |
|  | Step 9 – right arc |
|  | Step 9 – right arc |

**שאלה 5**

1. ראשית, נציין את אופן פעולת האלגוריתם : מדובר באלגוריתם רקורסיבי אשר מקבל בין היתר את קבוצת הצמתים והקשתות ומחזיר עץ פורש. בכל קריאה רקורסיבית, האלגוריתם מוסיף את הקשת הטובה ביותר עבור כל צומת. במידה וקיבלנו עץ פורש, נחזיר אותו. אחרת, יש לנו מעגל – נקבץ את צמתי המעגל לצומת יחיד (contract) ונמצא עץ פורש עבור הגרף החדש, ע"י קריאה רקורסיבית. לאחר מכן, נפרוש את המעגל שקובץ ונסיר ממנו את הקשתות שיצרו את המעגל (expand), כך שעדיין יוותר עץ פורש.

הגרף שלנו הינו גרף סופי שכן קבוצת הקשתות והצמתים סופיים. בכל קריאה רקורסיבית, אנו מצמצמים את קבוצת הקשתות והצמתים. אי לכך, מובטח כי לאחר מספר סופי של קריאות רקורסיביות האלגוריתם יסתיים - ניוותר עם גרף המכיל צומת מקובץ יחיד (במקרה הגרוע ביותר), המהווה עץ פורש, או שמא נמצא עץ פורש קודם לכן. כך או כך, מדובר באלגוריתם סופי.

1. במקרה הגרוע, תהיינה קריאות רקורסיביות, שכן ייתכנו מעגלים (במידה ובשלב הראשון ייווצרו מעגלים בעלי 2 צמתים). עפ"י ההנחה חיפוש המעגלים יעלה לנו . אולם, עלינו להריץ אלגוריתם למציאת מעגלים פעמים ולכן, הסיבוכיות הכוללת הינה -

כאשר המעבר האחרון נובע מההנחה לפיה .

נציין כי עבור מקרה זה, סיבוכיות הפעולות של הקריאה הרקורסיבית עצמה הינן .