

## אוגדן בעיות מפורסמות במדעי המחשב

### על בינה מלאכותית ועיבוד רשימות

#### כתבה שרה פולק

#### תיאור הבעיה

האם ניתן להקנות למחשב יכולת חשיבה אנושית? זו הבעיה שמקרת (John McCarty) עסק בה. לאנשים יש יכולת טבעית לשער השערות ולהסיק מסקנות. לדוגמה, כאשר ילדה בת 5 משחקת עם מכונית מפלסטיק, היא דוחפת אותה הלוך ושוב ומצפצפת בצופר. הילדה יודעת שאסור לגלגל את המכונית על השולחן וכשהיא מסיימת לשחק בה, היא שמה את המכונית במקום שהוא מספיק גבוה, כדי שאחיה הקטן לא יגיע למכונית. כך מניחה הילדה, שכשהיא תחזור מבית הספר המכונית תמצא במקום שהיא הניחה אותה. ההיגיון שהנחה את הפעולות והציפיות שלה הם פשוטים, וכל ילד בגילה יכול לבצע אותם. אולם בניגוד לכך, מחשבים בעלי יכולת חישוב גבוה בהרבה משל הילדה ומשל שאר האנשים, לא יכולים לבצע פעולות הקשורות בהיגיון פשוט. למעשה, למחשבים אין ידע על פעולות יום יומיות פשוטות שילד בן 5 למד מהוריו, והמחשב הוא בעצם "מכונה טפשה" חסרת רכיבים שמקנים את היכולת לבצע היסק תוך שימוש בהיגיון פשוט, כפי שאנשים עושים.

#### מה כן מחשב יכול לעשות?

לוגיקה קונבנציונאלית משתמשת בדדוקציה שמאפשרת לנו להסיק שאם הנחת יסוד מתקיימת, אז גם ההיסק תקף. לדוגמה, קיימת הנחת יסוד כי: "כל השחקנים המובטלים עובדים בינתיים כמלצרים" ו-"פרנק הוא שחקן מובטל" ולכן ניתן להסיק ש"פרנק הוא מלצר". אחת התכונות של דדוקציה היא מונוטוניות, כלומר אם למדנו עובדות חדשות שהן לא סותרות את הנחות היסוד אז המסקנה עדיין תקפה. אבל בעוד שרובנו לומדים דדוקציה בבית ספר, אנו בקושי משתמשים בפרקטיקה זו בחיי היום-יום ובדרך כלל אנו משתמשים בהיגיון בריא. למשל הילדה מאמינה שהמכונית תישאר באותו מקום שהיא הניחה אותה, משום שהיא שמה אותו רחוק מהישג ידו של אחיה, אבל יתכן שהיא לא תהיה בטוחה בכך אם היא תראה אותו מטפס על כיסא. היגיון בריא של ילד בן 5 מבוסס על יכולת לבצע השערות וניחושים לוגיים שמשתנים ומתעדכנים במשך הזמן נוכח עובדות חדשות שצצות.

באופן יחסי קל לכתוב תוכנית שתגרום למחשב לבצע דדוקציה כשאר אנו נשענים על מתמטיקה שהיא ברורה ומובנת. אבל קשה מאוד לגרום למחשב לבצע השערה על סמך היגיון בריא עליו כל אדם נשען. כדי שמחשב יהיה מסוגל לעשות זאת, צריך להמציא לוגיקה מתמטית חדשה. זו הייתה המטרה העיקרית של John McCarty - לבנות מחשב עם אינטליגנציה.

בדרך להשגת מטרתו, פיתח McCarty בשנת 1958 את שפת LISP (LISt Processing), שהיא כיום השפה הבסיסית ביישומים מתחום הבינה מלאכותית (Artificial Intelligence) שאפשרה לו להציג ידע סימבולי ולעבד אותו. בפרק זה אנו נציג את תהליך פיתוח שפת LISP והבעיות שעמדו בפני McCarty. לאחר מכן נסביר את המושג בינה מלאכותית ונציג סוגי בעיות שקשורות לתחום זה.

## פיתוח שפת LISP:

בשנות ה-1950 המאוחרות, מספר מדענים (Newell, Shaw & Simon) תיארו את הגרסה השנייה של השפה לעיבוד מידע (Information Processing Language – IPL2). שלושת המדענים פיתחו תכנה שנקראה Logic Theory Machine, תכנה שיכול להוכיח טענות על סמך לוגיקה בסיסית ולשחק משחקים. לשם כך הם היו צריכים שפת תכנות שיכולה לעשות מניפולציות על סימבולים על עצמים כמו כלים בשחמט, פעולות שהיו שונות לחלוטין מעיבוד מתמטי על מספרים.

כדי להדגים את השימוש ברשימה לעיבוד סימבולי נשתמש בדוגמה מהספר "עליסה בארץ הפלאות". נניח כי החתול צ'שייר (Cheshire) אומר לעליסה " או שאני משוגע או שהכובען משוגע". נסמן את הכובען ב-H, את החתול ב-C ואת עליסה ב-A. נוכל לרשום את טענתו של החתול במנה של רשימה בצורה הבאה:

$$(or C H)$$

אחר כך הוסיף החתול ואמר: " או שאת או שהכובען משוגעים". ניתן להוסיף טענה זו לטענה הקודמת ולרשום:

$$(and (or C H) (or A H))$$

לבסוף אמר החתול " רק אחד משלושתנו הוא משוגע". כלומר לפחות שניים מבין השלושה הם לא משוגעים. את כל המידע הזה ניתן לרשום בצורה הבאה:

$$(and (or C H) (or A H) (or (and(not A) (not C)) (and (not A) (not H)) (and (not C) (not H))))$$

אחרי שהגדרנו את המידע ברשימה, אנו יכולים לכתוב חוקים לעיבוד רשימה כמו

$$(and (or X Y) (or Z Y)) = (or (and X Z) Y)$$

שהמשמעות היא: אם אחד משנים או X או Y וגם אחד משניים Z ו Y מתקיים, אז או ש-X ו-Z מתקיימים או Y מתקיים.

יישום של חוק זה וחוקים נוספים מאפשרים לנו להסיק ש-

$$(and H (not C) (not A))$$

ובהתאם לחתול, ניתן להסיק כי רק הכובען הוא משוגע.

שימוש בטכניקה של עיבוד רשימות להיסק לוגי הוא מבריק משום שרשימות יכולות לגדול ולהצטמצם ולעדכן את עצמם מחדש. יתרה מזאת, ניתן לייצג באותה מבנה (רשימה) גם חוקים וגם מידע. זה היה אחד ההישגים בכנס בדרטמוט. ב-1956, John Backus וצוות מ-IBM הציג את שפת התכנות העילית הראשונה- שפת פורטרן. פורטרן פותחה כדי להקל על ביצוע חישובים אריתמטיים, שביצועם באסמבלר היה קשה ומסובך. עד היום היא נחשבת פורטרן כשפת תכנות המתאימה לחישוב הנדסי ומדעי. FLPL היתה הניסיון הראשון להרחיב את שפת פורטרן לעיבוד של סמלים. בקיץ 1958 כשמקרתית עבד ב-IBM הוא השתמש בה כדי לכתוב תוכניות על רשימות לביצוע משימה שהוא למד עליה בתיכון: גזירה של ביטויים אלגבריים. הרעיון המיידני שנדרש היה שימוש ביטויים רקורסיביים, אותן לא היה ניתן לבצע בפורטרן, כפי שמקרתית מעיד: "אם פורטרן היתה מאפשרת רקורסיה, אני לא הייתי ממשיך לפתח את

PLPL. אני אפילו חקרתי את השאלה איך ניתן להוסיף רקורסיה לפורטרן , אבל זה היה יותר מידי מסובך".

באותו זמן ב- IBM איבדו את העניין ב- AI בעיקר מסיבות פוליטיות. חלק מהלקוחות של IBM חשבו שהם יאבדו את עבודתם כשמכונות אינטליגנטיות יכנסו. בתגובה פרסה IBM הודעה שהמחשב היא מכונה טיפשה והיא עושה מה שאומרים לה, לא פחות ולא יותר. קשיים אלו גרמו למקקרת להחליט שלא להשתמש בפורטרן, אלא לפתח שפה חדשה LISP - שפה לעיבוד רשימות ובה כל המידע מוצג כרשימות. רשימות אלה נרשמות בתוך ( ). לדוגמה: (אלון מלמד דנה) הוא הייצוג של המשפט "אלון מלמד את דנה". במשפט זה וברשימה המייצגת אותו הסדר חשוב. לעומת זאת ברשימה "חלב, לחם, ריבה" המתארת רשימת מכולת ובייצוג שלה (חלב, לחם, ריבה) אין חשיבות לסדר האיברים. בשתי הדוגמאות הרשימות מכילות אטומים שהם האלמנטים של הרשימה. כמובן רשימה יכולה להכיל רשימות. רשימות יכולות לייצג כל מבנה מתמטי סטנדרטי, לדוגמה: (times 6 (plus x y), שהמשמעות שלו היא  $6 \times (x+y)$ . כך ניסה מקקרת להרחיב את השפה כדי שהיא תוכל לבצע כל עיבוד מתמטי, ולכן הוא הוסיף את הפעולה eval שמאפשר לתוכנית להגדיר פונקציה או פרוצדורה חדשה ולבצע אותה כחלק מהתוכנית. רוב השפות בהן מוגדר פונקציה חדשה, יש לעצור את התוכנית ולהדרה מחדש, זאת לפני שהתכנית תוכל להשתמש בפונקציה החדשה. לעומת זאת, ב- LISP, הפונקציה eval יכולה להשתמש בכל פונקציה ולבצע אותה, וכך היא מדמה "מכונת טיורינג אוניברסלית".

נדגים שימוש בפעולה eval ביישום עסקי בתחום תיווך דירות. לדוגמה, דמי תיווך של בתים להשכרה מבצע את החישובים 24 שעות ביממה, 7 ימים בשבוע בגלל הפעילות של שווקים פיננסיים בינלאומיים. נניח שמישהו כתב תוכנית לניתוח נתוני בורסה. ברורים ירצו להשתמש בתוכנית מיד אבל רק אם הם יוכלו להשתמש בה מבלי לאבד את השימוש בתכניות שבהן הם משתמשים. תכונה eval מאפשרת זאת. הרעיונות ששולבו ב- LISP יושמו בשפות אחרות, לדוגמה שפת Algol היתה השפה הראשונה, אך בהמשך גם פסקל, C, עדה ושאר שפות התכנות המודרניות, שאפשרה שימוש ברקורסיה וביטויים מותנים. עד לאחרונה השפות העיקריות לא כללו את eval משום שמתכנני השפה פחדו לתת למתכנתים את היכולת לשים פעולות חדשות בתוכנית מורצת. אבל בגלל סוג היישומים שרובם מורצים ברציפות ongoing האפשרות הזאת נכללת בשפות חדשות.

LISP היא השפה השלטת ביישומי בינה מלאכותית. מקקרת לא ציפה שהיא תשרוד ואפילו הציע לשנותה ולעשות אותה דומה ל- ALGOL. אבל עד היום מתכנתים ב- AI מעדיפים את הגרסה המקורית של LISP. ממקרת ראה ב- LISP אמצעי להגיע למטרה הגדולה שלו – "לבנות מכונות שתהיינה אינטליגנטיות כמו אנשים".

### ג'ון מקרטי (John McCarty)

נולד בבוסטון 1927 להורים שהיו חברי המפלגה הקומוניסטית ולכן נאלצו לעבור ממקום למקום תכופות. אביו האירי עבד כנגר, דייג וחבר פעיל במפלגה ואמו היהודייה עבדה כעיתונאית. משפחתו נדדה ועברה מבוסטון לניו-יורק ולבסוף ללוס אנג'לס. הרקע הפוליטי של משפחתו השפיע על מקקרת והוא התעניין

בתרומה של המדע: " הייתה אמונה שטכנולוגיה היא משהו טוב לאנושות. אני זוכר שכילד קראתי ספר שנקרא "Whys 100000", ספר פופולארי על טכנולוגיה סובייטית שנכתב ע"י M. Ilin בשנות ה-1930 המוקדמות. אני לא זוכר שום ספר אמריקאי מאותו סוג."

ב-1944 הוא למד במחלקה המתמטית ב-California Institute of Technology ושם סיים ב-1948. באותה שנה הוא נכח בסימפוזיון בו השתתף Von Neumann מתמטיקאי ומתכנן של ארכיטקטורת המחשב בו אנו משתמשים כיום. Von Neumann הציג מאמר על Self-replicating automata – מכונה שיכולה ליצור עותקים של עצמה, ולמרות שבכנס זה לא התייחסו ל-AI, ההרצאה של וון ניומן הציתה את הסקרנות של ממקרתי. ב-1949 הוא התחיל החל בלימודי דוקטורט במחלקה למתמטיקה בפרינסטון. שם הוא הציג את הניסיון הראשון שלו למודל של AI על מכונה: "התייחסתי לבניה כאל אוטומט סופי המקושר לסביבה שהיא אוטומטציה סופית. קבעתי פגישה עם וון ניומן. הוא עודד אותי. הוא אמר 'כתוב את זה'. אבל אני לא כתבתי כי לא הרגשתי שזה ממש טוב". הוא תיאר מודלים אוטומטיים של מכונות הנעות ממצב למצב. לדוגמה כאש נהג מניע מכונת יהיה מעבר ממצב "off" למצב של "in-neutral-but-on", וכאשר הוא מעביר להילוך ראשון, יהיה מעבר למצב "בהילוך ראשון וסע". מקקרתי זנח את הניסיון הראשון לשימוש באוטומטה למודל של אינטליגנציה אנושית. ב-1952 פגש מקקרתי בוגר אחר של פרינסטון- Jerry Rayna. גפרי הציע שמקקרתי ינסה להקים צוות של חוקרים שמעוניינים בנושא של מכונות אינטליגנטיות. ואכן גובשה של קבוצה שאספה מאמרים בנושא, ובקיץ 1956 אורגן כנס ראשון ב-Dartmouth שעסק בנושא שממקרתי כינה לראשונה בשם Artificial Intelligence - AI. כך החל לקרום עור וגידים תחום חדש במדעי המחשב והוא הבינה המלאכותית. מקקרתי במחקרו התמקד בקשר שבין שפות ואינטליגנציה כשהוא מקווה שמחשב יוכל "לשחק משחקים היטיב ולעשות עוד דברים". בעוד שהכנס לא כל כך הצליח לפתור את הקשיים והבעיות ביצירת מכונה אינטליגנטית, הוא אפשר למשתתפים בו לבסס מטרות, תיאוריות וטכניקות שהובילו אחר להכרה ש-AI הוא תחום נפרד במדעי המחשב. על תרומתו לתחום בינה מלאכותית קיבל מקקרתי פרס טיורינג 1971.

## מקורות

מאמר שכתב מקקרתי על שפת LISP בשם "History of Lisp", נמצא באתר :

<http://www-formal.stanford.edu/jmc/history/lisp/lisp.html>

מאמר זה נמצא באתר של מקקרתי הכולל פרסומים רבים :

<http://www-formal.stanford.edu/jmc/>

מקור נוסף הוא הספר

Denis Shasha and Cathy Lazere. Out of Their Minds: The Lives and Discoveries of 15 Great Computer Scientists, NY: Copernicus, 1995, pp. 21-38.