**פרויקט – מבוא לבינה מלאכותית – 67842**

מגישים:

ניר בלוקה (206760811) | רועי מאיר (211442843) | רון בנימין (207837089)

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*תוכן עניינים\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. **מבוא**  
   הבעיה שבחרנו לפתור היא בעיית אופטימיזציה.  
   הבעיה שנבחרה היא מציאת תוכנית לימודים חוקית ואופטימלית לתואר כך שהממוצע של התכנית המוצעת יהיה מקסימלי. זאת על מנת לתת מענה לסטודנטים לתכנון של מסלול לימודים המאפשר להם להשיג את הממוצע הטוב ביותר.  
   כלומר, עבור שנתון של מסלול לימודים מסוים, המכיל את כמות נקודות הזכות הדרושות למסלול, רשימת הקורסים המוצעים (שם, מספר, נ"ז, דרישות קדם, ממוצע, סמסטר), ועבור דרגת עומס לכל סמסטר (מבחינת כמות נ"ז), נרצה למצוא תוכנית לימודים העומדת בכל הדרישות וממקסמת את הממוצע הכללי בתואר.  
     
   מצאנו את בעיה זו מעניינת משתי סיבות עיקריות:

* מדובר בבעיה שאנחנו וכלל הסטודנטים מכירים מקרוב, כולנו משקיעים זמן רב בתכנון תוכנית הלימודים של התואר, וכולנו יודעים כמה זה מורכב לקחת את כלל השיקולים בחשבון כדי לסיים את התואר עם ממוצע טוב ככל הניתן.
* מבחינה חישובית הבעיה היא קשה מאוד – ישנם המון סידורים של תוכניות לימודים חוקיות (מספר אקספוננציאלי במספר הקורסים המוצעים), ומציאת התוכנית שממקסמת את הממוצע עלולה לגרום לבדיקת כלל האפשרויות.  
  הבעיה שאותה אנו מנסים לפתור דומה למספר בעיות קשות.  
  נשים לב שפתרון לבעיה שלנו הוא בפרט בחירה של קורסים שסכום הנ"ז הכולל שלהם שווה בדיוק למספר הנקודות הדרוש במסלול הלימודים. בעיה זו, ללא ההתייחסות למקסום הממוצע או חוקיות התכנית, היא קשה בפני עצמה, שכן היא דומה לבעיית הסכומים החלקיים (Subset Sum Problem) שהיא בעיה NP-Complete.  
  בנוסף, בבואנו למקסם את ממוצע הקורסים הנלקחים אף ללא התחשבות בחוקיות התכנית, פתרון הבעיה דומה לפתרון בעיית התרמיל השלם (0-1 Knapsack), שהיא בעיה NP-Hard. בדומה לבחירתם של קורסים שיניבו את הממוצע המקסימלי, בבעיית התרמיל נבחר פריטים כך שערכם הכולל יניב את הערך המקסימלי.  
  בנוסף, באחד המאמרים שקראנו[[1]](#footnote-1) בנושא, נאמר כי מציאת תת קבוצה אופטימלית של קורסים עם דרישות קדם זו בעיה NP-Hard וקל להיווכח שזוהי תת בעיה של הבעיה שאנו מנסים לפתור ולכן בפרט הבעיה שלנו היא NP-Hard בעצמה.

בחרנו לנסות לפתור את בעיה זו באמצעות שתי שיטות שנלמדו בקורס:

* **חיפוש בגרף –** תכננו למדל את הבעיה כבעיית חיפוש בגרף בה מציאת מסלול קצר ביותר בגרף החיפוש שקול לפתרון הבעיה. כלומר, בעזרת המסלול הקצר ביותר בגרף זה, נוכל לחלץ את תכנית חוקית ואופטימלית למסלול הלימודים.

בפרט, תכננו להשתמש באלגוריתם A\* על מנת להשיג פתרון בזמן ריצה מתקבל על הדעת.

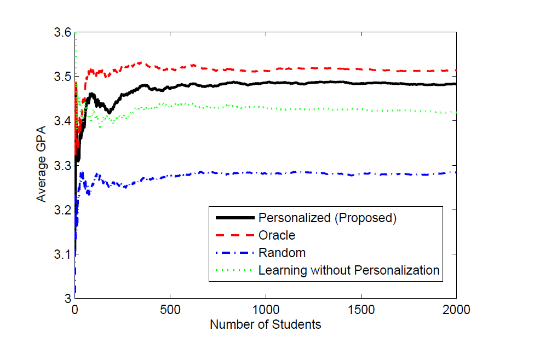
* **חיפוש מקומי** - (Local Search) – תכננו למדל את הבעיה כבעיית חיפוש מקומי בה מתחילים מתוכנית מסוימת, ואותה מנסים לשפר בהדרגתיות בעזרת אלגוריתמי חיפוש שונים הנלמדו בקורס. בפרט, תכננו להשתמש באלגוריתמים Stochastic Beam Search, Simulated Annealing, Hill Climbing.

1. **עבודה קודמת**

לא מצאנו עבודה שמספיק דומה לנושא שבחרנו בפרויקטים משנים קודמות בקורס. עם זאת, כן מצאנו שני מאמרים באינטרנט שמנסים לפתור בעיות דומות לבעיה שלנו:

* המאמר הראשון[[2]](#footnote-2) מתייחס לבניית מערכת מותאמת אישית ומשתנה עבור כל סטודנט, שמשיגה איזון בין סיום מהיר של התואר לבין מקסום הממוצע. זאת תוך התחשבות באילוצים דומים לבעיה שלנו אך בתוספת התייחסות למספר פרמטרים על הסטודנט עצמו לדוגמה ציון ה-SAT שלו.   
  כותבי המאמר השתמשו בשני אלגוריתמים עיקריים כדי לפתור בעיה זו:
  + Forward-Search Backward-Induction Algorithm – אלגוריתם שלא למדנו ומנסה לשבץ קורסים לפי דרישות הקדם שלהם בעזרת תכנון דינאמי.
  + Multi-Armed Bandit Algorithm – אלגוריתם שלא למדנו ומתמקד במציאת המלצות לקורסים למיקסום הממוצע ומזעור הזמן לסיום התואר, בהסתמך על נתונים של סטודנטים קודמים עם פרופיל דומה לסטודנט שמנסה לבנות את המערכת.

במאמר זה, החוקרים השוו את תוצאותיהם אל מול מאגר ציונים של אוניברסיטת UCLA כדי למדוד את איכות התוצאות שקיבלו. בפרט הם מדדו את איכות הפתרון שלהם למול מודלים שונים. הראשון הינו מודל "אורקל" אשר מכיר סטטיסטיקות של ציוני מסלולים על בסיס העבר ועל כן מהווה חסם עליון לפתרון, והשני הוא מודל רנדומלי ועל כן מהווה חסם תחתון לכל פתרון סביר. ניתן לראות שהתוצאות שלהם קרובות למודל ה-"אורקל".



ההנחות במאמר הן שקיימים מסלולי תואר בהם קורסים עם דרישות קדם חובה, כאשר כל קורס יכול להילמד בסמסטר אחד או יותר בשנה (אצלם יש רבעונים), בנוסף מניחים קיום מאגר כנ"ל.

* המאמר השני[[3]](#footnote-3) מתייחס לבניית מערכת קורסים חוקית תוך פיזור העומס ורמת הקושי בין הסמסטרים. כותבי המאמר השתמשו בשלושה אלגוריתמים על מנת לפתור את הבעיה:
  + Maximum Prerequisite Weightage (MPW) Algorithm, אשר מעדיף לשבץ בתחילה קורסים שהם קורסי קדם של קורסים רבים תוך התחשבות בנסיון לאזן את הקושי של הקורס.
  + Difficulty Approximation (DA) Algorithm, אשר מתמקד באיזון דרגת העומס והקושי בין הסמסטרים השונים ומכוון למזער פונקציית הפסד (Loss) מעל דרגת הקושי בין סמסטרים.
  + Adaptive Genetic Algorithm (AGA), אלגוריתם עם גישה היוריסטית שבאה לידי ביטוי בפונקציית ה- Fitness, בה נבחרת סדרת קורסים תוך התחשבות לקורסי קדם וקושי, אשר מתפתחת לאורך איטרציות בריצת האלגוריתם.

במאמר זה, השוו החוקרים את הפתרונות של האלגוריתמים השונים וכן של אלגוריתם רנדומלי, והתמקדו במדידת והשוואת פיזור הקושי בין הסמסטרים. בנוסף, השוו בין מספר הנ"ז שנלקחו בכל סמסטר.

ההנחות במאמר הן שישנם מסלולי תואר עם קורסים, לכל קורס ישנם קורסי קדם חובה, וכן שישנו אינדקס של קושי לקורסים וכל קורס מכיל ערך סטטי המייצג את הקושי שלו.

1. **מתודולוגיה**בפרויקט שלנו מידלנו את הבעיה בשתי דרכים, אחת עבור בעיית חיפוש בגרף, והשנייה עבור בעיית חיפוש מקומי.  
     
   **חיפוש בגרף**   
   נגדיר את הבעיה בצורה פורמלית :  
   כאשר כל s ∈ S הוא מצב המייצג תוכנית לימודים (לאו דווקא שלמה/חוקית). כל תוכנית לימודים מכילה בתוכה את רשימת הקורסים ששובצו ובאילו סמסטרים נלקחו.   
    הוא המצב ההתחלתי והוא תוכנית ריקה מקורסים.   
    היא קבוצת כל המצבים המקבלים וכל מקיים כי סכום הנ”ז של הקורסים ששובצו שווה בדיוק לדרישה במסלול וכן נלקחו כל קורסי החובה.  
    היא קבוצת הפעולות שניתן לבצע על תוכנית לימודים. כל היא פעולה המוסיפה קורס מסוים לתוכנית לימודים.   
    היא פונקציית המעברים כאשר ומכיל את כל הזוגות   
   כך שניתן לבצע את הפעולה על המצב בצורה חוקית - תוך התחשבות בקורסי קדם, מידת העומס בסמסטר, הסמסטר בו הקורס נלמד, הימנעות מבחירה כפולה של קורסים, מבלי לחרוג מכמות הנ”ז הדרושים במסלול. בפרט המצבים המקבלים יקיימו חוקיות זו.  
    היא פונקציית המחיר, כך ש- כלומר, המחיר של הוספת קורס, הינו כמות הנקודות שהתוכנית מפסידה מהממוצע הכולל, בלקיחת הקורס. כלומר, בהינתן מסלול ממצב התחלתי עד למצב מקבל, סכום כל הצלעות במסלול שווה בדיוק ל- כאשר הוא ממוצע משוקלל של הקורסים בתוכנית הסופית. לכן, המסלול הקצר ביותר (הזול ביותר) מייצג תוכנית עם ממוצע מקסימלי.

**חיפוש מקומי**

במידול זה, קבוצת המצבים היא קבוצת תוכניות הלימודים כפי שהגדרנו בחיפוש בגרף.  
המצב ההתחלתי  *נדגם באקראיות מתוך קבוצת המצבים ההתחלתיים המכילה תוכניות semi חוקיות (ללא אכיפה של הגעה למכסת הקורסים או לקיחת כלל קורסי החובה).* היא קבוצת הפעולות שניתן לבצע על תוכנית לימודים. כל היא פעולה שניתן לבצע על מצב – הוספת קורס / הסרת קורס / החלפה של קורס אחד באחר.   
 היא פונקציית המעברים כאשר ומכיל את כל הזוגות כך שניתן לבצע את הפעולה על המצב בצורה חוקית - תוך התחשבות בקורסי קדם, מידת העומס בסמסטר, הסמסטר בו הקורס נלמד, הימנעות מבחירה כפולה של קורסים.  
 היא פונקציית המחיר, כך ש- כלומר, \*\*\*

כדי להעריך את איכות התוצאה של האלגוריתמים השונים (בשתי השיטות), השוונו את התוצאה לחסם מלעיל שחישבנו על הממוצע האפשרי של תכנית לימודים אופטימלית (לא בהכרח ניתן להגיע לחסם זה) – על מנת להגיע לחסם זה, ביצענו רלקסציה לבעיה תוך התעלמות מכלל המגבלות / אילוצים של הבעיה, מלבד לקיחת כל קורסי החובה והגעה למספר הנ"ז הדרוש למסלול. השיטה שבחרנו לחשב את החסם הינה ההדוקה ביותר שהצלחנו לממש.

בנוסף, עבור קלטים קטנים יחסית, אותם ניתן לפתור בעזרת אלגוריתם UCS, המבטיח

אופטימליות – ניתן להשוות את תוצאותינו לתוצאה האופטימלית.  
   
הנחות:

* בכל תוכנית לימודים תקינה ישנן מספר נקודות זכות שצריך להגיע אליהן.
* לזכאות לתואר יש להגיע בדיוק לכמות הנ"ז הנדרשות מבלי לחרוג.
* בכל תוכנית לימודים יכולים להיות קורסי חובה + בחירה (כל קורס הוא אחד מבין השניים).
* כל קורס נלמד בסמסטר א' ו/או ב' בלבד.
* לקורס יכולים להיות דרישות קדם ולא ניתן לקחת קורס ללא עמידה בדרישות – אין צורך לעבור עם ציון מסוים.
* מספר נ"ז של קורס מסוים הוא שלם אי שלילי.
* לכל קורס שנלמד בסמסטר מסוים, יש ממוצע ציונים (למשל משנים קודמות).
* קורס זהה בסמסטרים שונים יכיל מספר קורס זהה.
* ניתן לקחת כל קורס בכל שלב כל עוד תנאי הקדם שלו מולאו.
* בכל קלט של מסלול כלשהו, יש מספיק קורסים על מנת לעמוד בדרישות.

1. **תוצאות**

את הבעיה פתרנו בעזרת שתי שיטות- חיפוש בגרף וחיפוש מקומי.  
את האלגוריתמים בחנו על מספר קלטים של מסלולי לימודים שלמים, שנלקחו משנתון האוניברסיטה, וכן קלטים מצומצמים.

**חיפוש בגרף-**

עבור בעיה זו, האלגוריתם A\* Search היה האלגוריתם בו בחרנו להשתמש, ואותו השוונו לשני אלגוריתמי בסיסיים - DFS, אשר מוצא תוכנית לימודים חוקית אך לא בהכרח

אופטימלית, ו- UCS, אשר מוצא תוכנית לימודים חוקית ואופטימלית.

**A\* Search:**  
לאחר ניסיונות רבים לבניית היוריסטיקה עבור אלגוריתם A\*, הגענו להיוריסטיקה אדמיסבילית אך לא קונסיסטנטית. זאת מכיוון שזו הביאה לממוצע הטוב ביותר, ביחס לזמן הריצה. היוריסטיקה שבחרנו, מבצעת הערכת חסר לכמות הנקודות שתוכנית נתונה עלולה להפסיד מהממוצע הסופי מאותה נקודה ועד בניית תוכנית לימודים שלמה. כלומר, היוריסטיקה מנסה להעריך את המסלול הטוב ביותר שהתוכנית יכולה להשיג, תוך ביצוע רלקסציה למגבלות החוקיות של התוכנית. לכן, בבעיה שלנו שמודלה כבעיית חיפוש בגרף ולא בעץ (שכן ישנן מספר דרכים שאיתן ניתן להגיע לאותה תכנית לימודים),   
לא מובטח לנו פתרון אופטימלי.

**DFS:**

אלגוריתם DFS סיים והחזיר פתרון חוקי לכלל הקלטים- לפי איך שהבעיה מוגדרת, אם הוא מחזיר פתרון הוא בהכרח חוקי. עם זאת, מכיוון שהאלגוריתם אינו מתייחס כלל למשקלי הצלעות, הממוצע של התוכנית המוחזרת לא היה טוב מספיק עבור הקלטים שנבדקו.

**UCS:**

אלגוריתם UCS לא עצר על קבצי הקלט הגדולים, גם אחרי מספר שעות, ואחרי מיליוני פעולות expand. האלגוריתם מבטיח פתרון אופטימלי ולכן הוחלט לבנות ולבדוק קבצי קלט מצומצמים יותר, אשר בעזרתם נוכל להעריך בצורה טובה את התוצאות של DFS ו- A\*.

כדי להעריך את טיבו של הממוצע המתקבל מקבצי הקלט הגדולים, השווינו את התוצאות עם החסם מלעיל שתיארנו בסעיף המתודולוגיה.

להלן טבלה בה ניתן לראות את זמני הריצה השונים ומספר הקודקודים שנחקרו בריצת אלגוריתמי חיפוש בגרף השונים:

|  |  |  |  |
| --- | --- | --- | --- |
| **Algorithm** | **Data set** | **Expanded** | **Time (sec)** |
| Astar | Mini CS | 24259 | 10 |
| DFS | Mini CS | 24 | 0.005 |
| UCS | Mini CS | 1260027 | 150 |

1. Personalized Course Sequence Recommendations - Jie Xu, Member, IEEE, Tianwei Xing, Student Member, IEEE, and Mihaela van der Schaar, Fellow, IEEE [↑](#footnote-ref-1)
2. Personalized Course Sequence Recommendations - Jie Xu, Member, IEEE, Tianwei Xing, Student Member, IEEE, and Mihaela van der Schaar, Fellow, IEEE [↑](#footnote-ref-2)
3. Course Sequence Recommendation with Course Difficulty Index Using Subset Sum Approximation Algorithms - M. Premalatha, V. Viswanathan [↑](#footnote-ref-3)