פרויקט – מבוא לבינה מלאכותית – 67842

מגישים:

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

תוכן

קישור לקוד הפרויקט

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, אף 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] בנושא, נאמר כי מציאת תת קבוצה אופטימלית של

[1] Personalized Course Sequence Recommendations - Jie Xu, Member, IEEE, Tianwei Xing, Student Member, IEEE, and Mihaela van der Schaar, Fellow, IEEE

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:

- **<u>חיפוש מקומי</u>** - (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 $Stochastic\ Beam\ Search, Simulated\ Annealing, Hill\ Climbing$.
  בחרנו בשיטה זו, מכיוון שגודל מרחב המצבים של הבעיה שלנו הוא סופי, ולכן שיערנו לעצמנו שבעזרת חיפוש מקומי מתוחכם, נוכל להתגבר על בעיית המקסימום המקומי, ולמצוא בהסתברות גבוהה את המקסימום הגלובלי.

- **<u>חיפוש בגרף</u> –** 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.
  בחרנו בשיטה זו, מכיוון שבאופן אינטואיטיבי, הרכבת מסלול לימודים נעשית צעד אחר צעד ובונה מסלול בגרף. מכיוון שגרף זה סופי, שיערנו שבעזרת היוריסטיקה טובה נמצא את הפתרון בזמן סביר (תחת מידול נכון של מסלול קצר ביותר).

עבודה קודמת

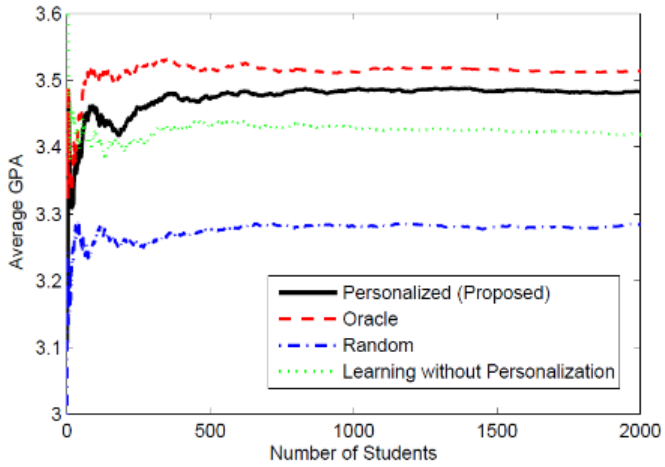
לא מצאנו עבודה שדומה לנושא שבחרנו בפרויקטים משנים קודמות בקורס. עם זאת, כן מצאנו שני מאמרים באינטרנט שמנסים לפתור בעיות דומות לבעיה שלנו:

- המאמר הראשון[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 Solver – 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".



[2] Personalized Course Sequence Recommendations - Jie Xu, Member, IEEE, Tianwei Xing, Student Member, IEEE, and Mihaela van der Schaar, Fellow, IEEE

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] 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 בניסיון 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.

[3] Course Sequence Recommendation with Course Difficulty Index Using Subset Sum Approximation Algorithms - M. Premalatha, V. Viswanathan

מתודולוגיה

בפרויקט שלנו מידלנו את הבעיה בשתי דרכים, אחת עבור בעיית חיפוש מקומי, והשנייה עבור בעיית חיפוש בגרף.

חיפוש מקומי

נגדיר את הבעיה בצורה פורמלית $\langle S, s_0, A, F, Fitness \rangle$:

S היא קבוצת המצבים 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.

המצב ההתחלתי $s_0 \in S_0$ נדגם באקראיות מתוך קבוצת המצבים ההתחלתיים $S_0 \subseteq S$ המכילה תוכניות *semi* חוקיות (ללא אכיפה של הגעה למכסת הקורסים או לקיחת כלל קורסי החובה).

A היא קבוצת הפעולות שניתן לבצע על תוכנית לימודים. כל $a \in A$ היא פעולה שניתן לבצע על מצב – הוספת קורס / הסרת קורס / החלפה של קורס אחד באחר.

$F: D \to S$ היא פונקציית המעברים כאשר $D \subseteq S \times A$ ומכילה את כל הזוגות (s, a) כך שניתן לבצע את הפעולה a על המצב s בצורה חוקית - תוך התחשבות בקורסי קדם, מידת העומס בסמסטר, הסמסטר בו הקורס נלמד, הימנעות מבחירה כפולה של קורסים.

$Fitness: S \to \mathbb{R}$ היא פונקציה המעריכה את "איכות" המצב כאשר המצבים החוקיים יקבלו ערך גבוה יותר מאשר לא חוקיים, וככל שלמצב ממוצע גבוה יותר ומספר רב של קורסים, כך הערך שיתקבל יגדל. הפונקציה בנויה באופן הבא:

$$\forall s \in S \qquad Fitness(s) := \begin{cases} avg^*(s) + 100 & \text{legal degree plan} \\ avg^*(s) & else \end{cases}$$

כאשר:

$$\forall s \in S \qquad avg^*(s) := avg(s) \cdot \frac{points(s)}{\text{Degree Target Points}}$$

בפרט, ככל שתוכנית מכילה פחות נ"ז, כך $avg *$ קטן, וכך הפונקציה מעודדת חוקיות.

חיפוש בגרף

נגדיר את הבעיה בצורה פורמלית $\langle S, s_0, G, A, F, C \rangle$:

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 .

$s_0 \in S$ הוא המצב ההתחלתי והוא תוכנית ריקה מקורסים .

G היא קבוצת כל המצבים המקבלים וכל $g \in G$ מקיים כי סכום הנ"ז של הקורסים ששובצו שווה בדיוק לדרישה במסלול וכן נלקחו כל קורסי החובה.

A היא קבוצת הפעולות שניתן לבצע על תוכנית לימודים. כל $a \in A$ היא פעולה המוסיפה קורס מסוים לתוכנית לימודים .

$F: D \to S$ היא פונקציית המעברים כאשר $D \subseteq S \times A$ ומכיל את כל הזוגות (s, a) כך שניתן לבצע את הפעולה a על המצב s 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.

$C: A \to \mathbb{R}$ היא פונקציית המחיר, כך ש- $C(a) \coloneqq (100 - a.avg) * \frac{a.points}{degree-target-points}$

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- $100 - avg$ כאשר avg הוא ממוצע משוקלל של הקורסים בתוכנית הסופית. לכן, המסלול הקצר ביותר (הזול ביותר) מייצג תוכנית עם ממוצע מקסימלי.

הערכת התוצאות (של שתי שיטות הפתרון)

כדי להעריך את איכות התוצאה של האלגוריתמים השונים, השווינו את התוצאה לחסם מלעיל שחישבנו על הממוצע האפשרי של תכנית לימודים אופטימלית (לא בהכרח ניתן להגיע לחסם זה) – על מנת לחשבו, 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 (של שתי שיטות הפתרון)

- בכל תוכנית לימודים תקינה ישנן מספר נקודות זכות שצריך להגיע בדיוק אליהן.
- בכל תוכנית לימודים יכולים להיות קורסי חובה + בחירה (כל קורס הוא אחד מבין השניים).
- כל קורס נלמד בסמסטר א' ו/או ב' בלבד. (ללא סמסטר קיץ).
- 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.

תוצאות

את הבעיה פתרנו בעזרת שתי שיטות- חיפוש מקומי וחיפוש בגרף.
את האלגוריתמים בחנו על מספר קלטים של מסלולי לימודים, ועם עומס סמסטריאלי שונה (Semester Load), אשר חלקם נלקחו משנתון האוניברסיטה.

חיפוש מקומי

עבור בעיה זו, השווינו בין האלגוריתמים הבאים: Stochastic Beam Search,Hill Climbing ו-Simulated Annealing. במידול שלנו, בשלושת האלגוריתמים האלו, כל פתרון שיתקבל יהיה Semi חוקי (לא בהכרח יכיל את כל קורסי החובה ולא בהכרח יגיע לסך הנ"ז הנדרש). אנו נותנים לאלגוריתמים "אינטרס" להגיע לחוקיות ואופטימליות (מבחינת ממוצע) בעזרת פונקציית ה-Fitness.

Hill Climbing:

אלגוריתם זה סיים לרוץ בזמן הריצה הנמוך ביותר מבין האלגוריתמים שבדקנו, עם זאת הרוב המוחלט של הפתרונות שקיבלנו בעזרתו לא היו חוקיים. כלומר, התכנס למקסימום מקומי נמוך יחסית.

Simulated Annealing:

לאחר ניסיונות רבים של ערכים שונים להיפר-פרמטרים של אלגוריתם זה, הגענו לאיזון טוב בין ביצועי האלגוריתם לבין זמן הריצה ובין exploration ל-exploitation. האלגוריתם מסיים לרוץ על קבצי הקלט שלנו בזמן סביר (יפורט בגרפים) ונותן אחוז גבוה יותר של פתרונות חוקיים ביחס ל-Hill Climbing.

Stochastic Beam Search:

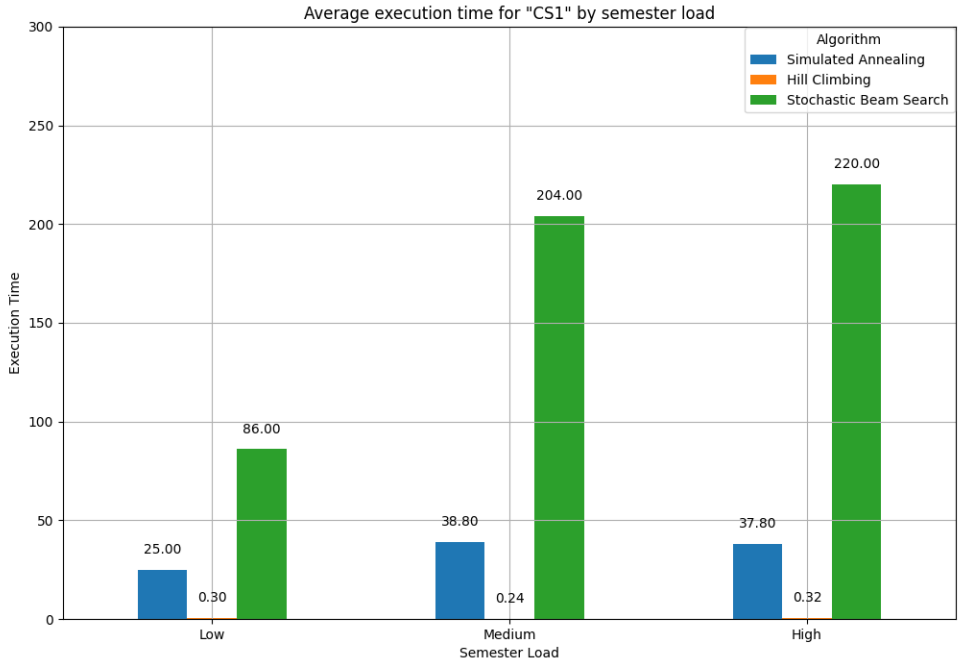
גם באלגוריתם זה ניסינו מספר רב של פרמטרים עד שהגענו לאיזון מבחינת איכות התוצאה וזמן הריצה. בנוסף, באלגוריתם זה הצלחנו להגיע ל-100% פתרונות חוקיים אך עם זמן ריצה גבוה יותר מאשר שאר האלגוריתמים שבדקנו.

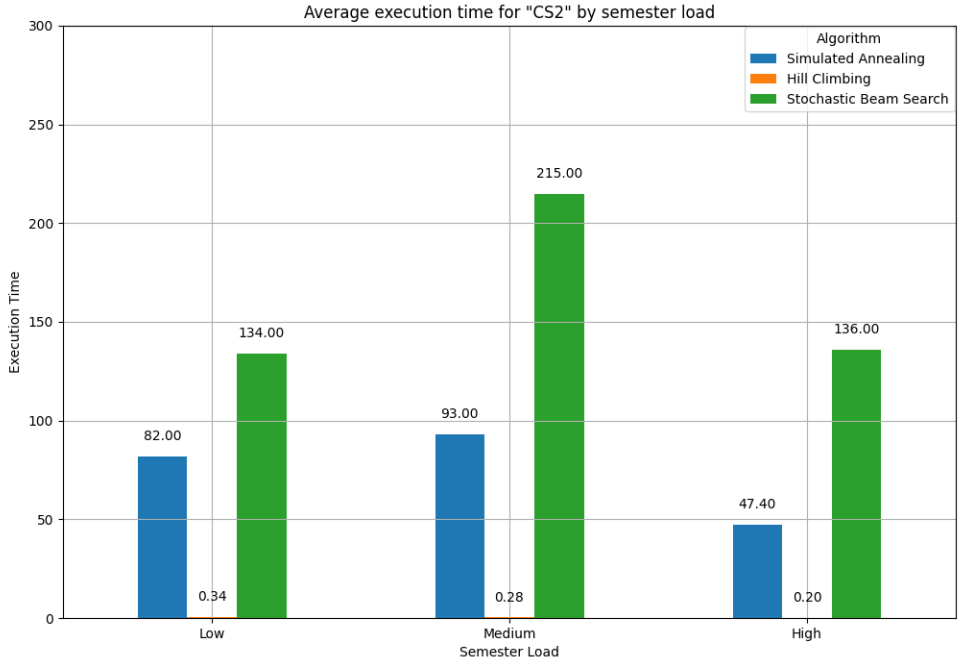
השוואות זמני הריצה

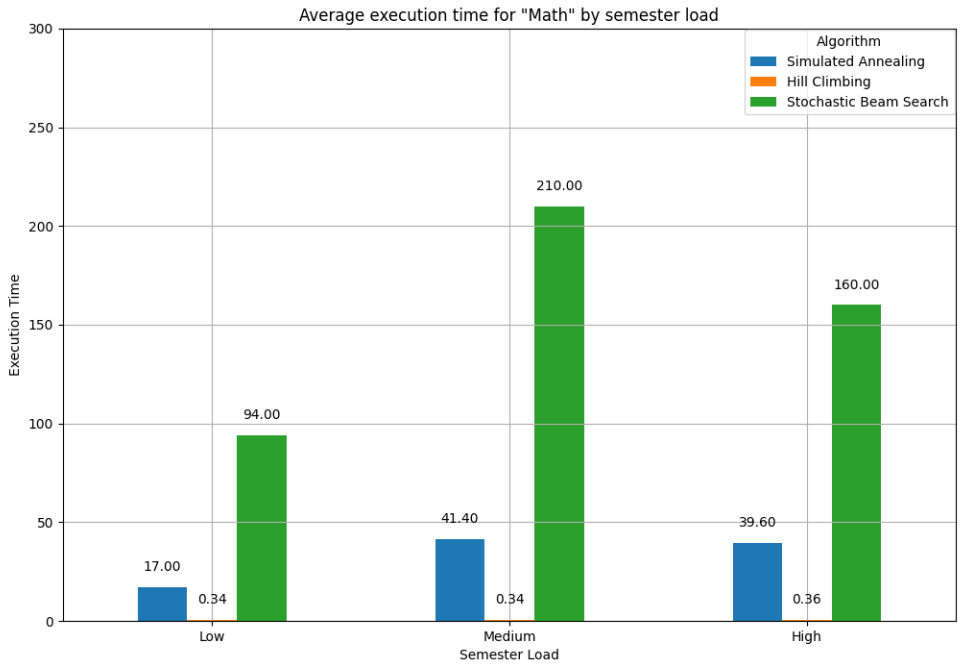
חישבנו את זמני הריצה הממוצעים על מספר ריצות של כל האלגוריתם:

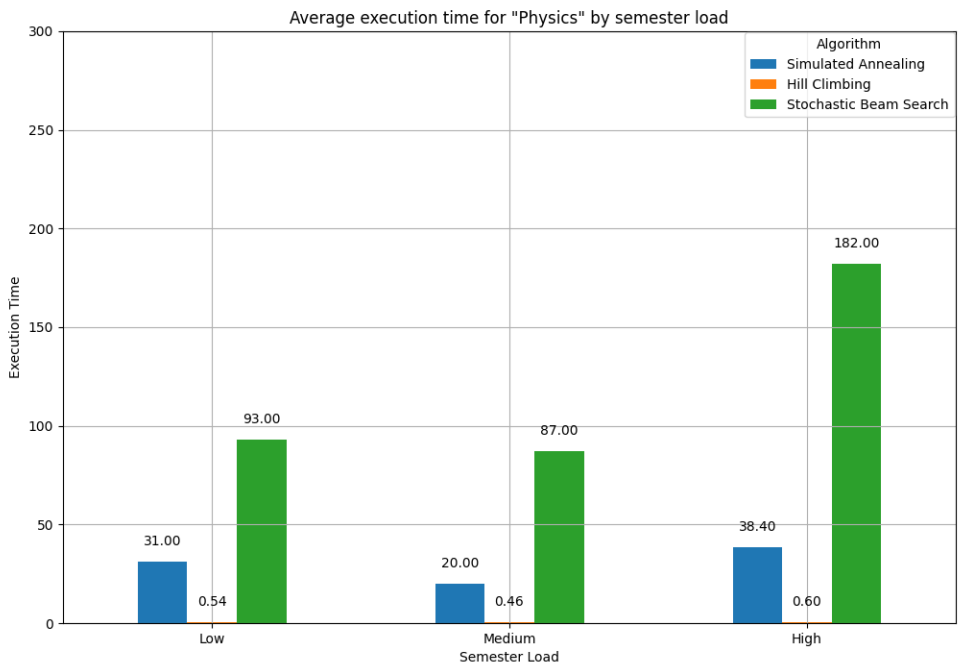
למעט אלגוריתם $Beam\ Search$ שנבדק על 10 ריצות שונות (מפאת זמן ריצה גבוה ביחס לאחרים), שני האלגוריתמים האחרים נבדקו על 50 ריצות שונות.

להלן ממוצעי זמני הריצה בעבור הקלטים השונים:









ניתן לראות בבירור מהגרפים שעבור זמני הריצה מתקיים:

$$Hill\ Climbing\ <\ Simulated\ Annealing\ <\ Stochastic\ Beam\ Search$$

בנוסף נבחין שזמן הריצה של $Hill\ Climbing$ נמוך בצורה משמעותית מאשר שני האלגוריתמים הנוספים (ולכן קשה להבחין בו בגרפים).

השוואות הממוצעים ויחס החוקיות

נציין שבחישוב הממוצע התחשבנו רק בערכים של תוכניות שלמות וחוקיות. כדי להעריך כמה הממוצע טוב, השתמשנו בחסם מלעיל שתואר קודם לכן (חסם שאינו בהכרח בר השגה).

להלן התוצאות:

Distance From Upper Bound			**Average Grade**			**Input**	
Hill Climbing	Beam Search	Simulated Annealing	Hill Climbing	Beam Search	Simulated Annealing	Dataset	Load
0.99	0.75	0.98	79.16	79.4	79.17	CS1	Low
7.325	6.335	6.685	74	74.99	74.64	CS2	Low
1.293	0.723	1.213	78.94	79.51	79.02	Math	Low
2.228	1.808	2.158	76.59	77.01	76.66	Physics	Low
0.83	2.64	0.79	79.32	77.51	79.36	CS1	Medium
6.955	6.715	6.795	74.37	74.61	74.53	CS2	Medium
1.243	0.853	1.133	78.99	79.38	79.1	Math	Medium
2.088	1.998	1.918	76.73	76.82	76.9	Physics	Medium
1.12	3.94	0.7	79.03	76.21	79.45	CS1	High
7.185	8.705	6.235	74.14	72.62	75.09	CS2	High
0.843	4.173	1.323	79.39	76.06	78.91	Math	High
1.698	2.978	1.518	77.12	75.84	77.3	Physics	High

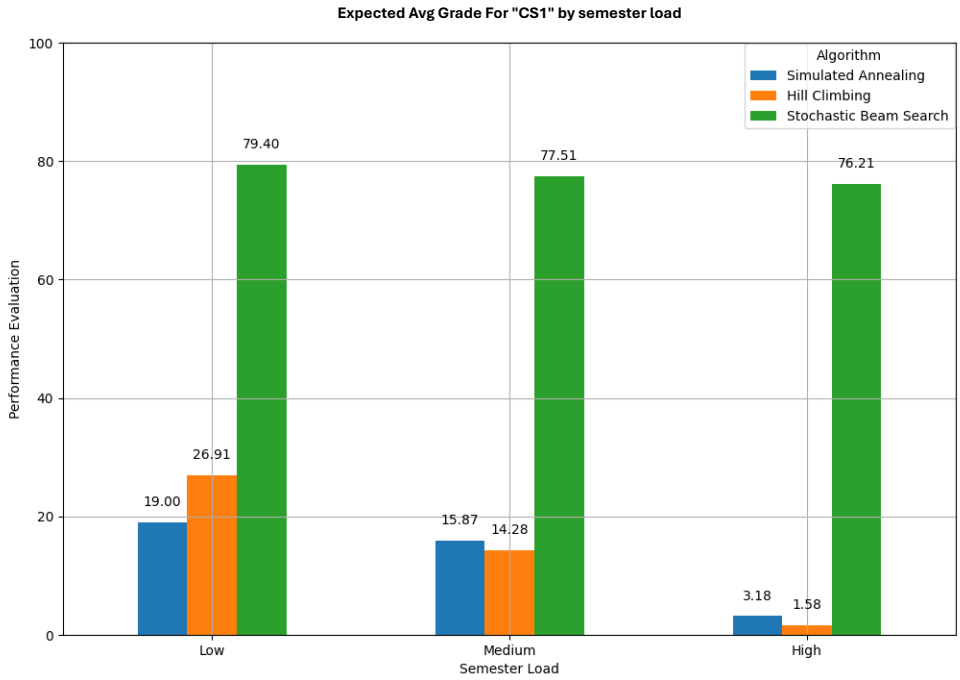
Legal Result Ratio			**Input**	
Hill Climbing	Beam Search	Simulated Annealing	Dataset	Load
34%	100%	24%	CS1	Low
48%	100%	92%	CS2	Low
64%	100%	54%	Math	Low
56%	100%	68%	Physics	Low
18%	100%	20%	CS1	Medium
26%	100%	58%	CS2	Medium
42%	100%	60%	Math	Medium
34%	100%	54%	Physics	Medium
2%	100%	4%	CS1	High
14%	100%	38%	CS2	High
38%	100%	32%	Math	High
6%	100%	10%	Physics	High

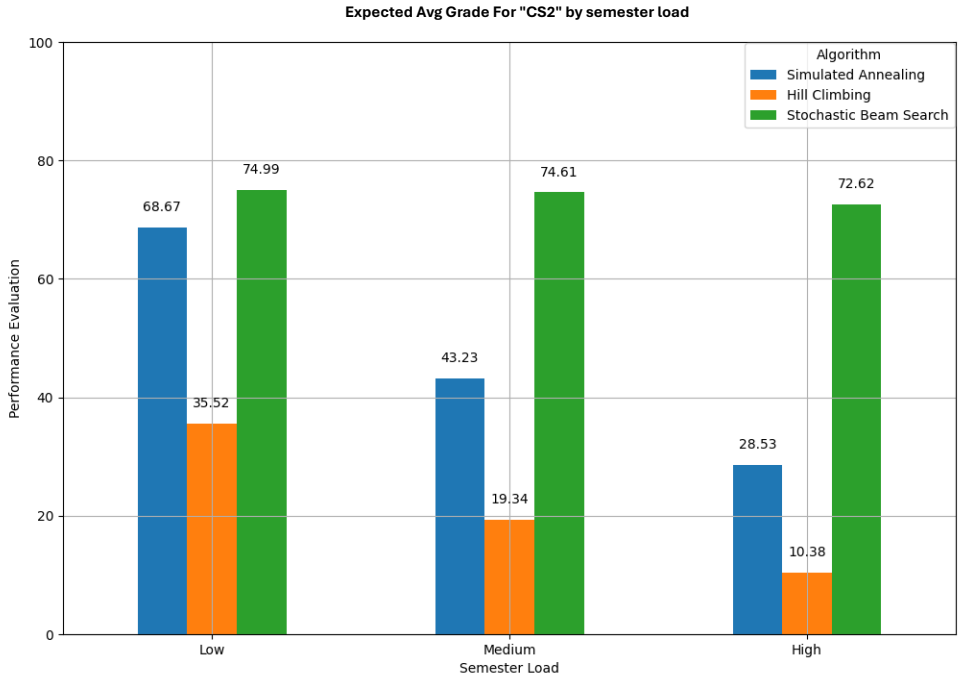
ניתן להסיק מהטבלאות, שהתוכניות שהתקבלו משלושת האלגוריתמים הביאו ממוצעים קרובים יחסית אל החסם מלעיל, מלבד קובץ קלט אחד (CS2) שהמרחק עליו היה גבוה יותר – וזאת מפני ששונות הציונים בו גבוהה משמעותית מאשר שאר הקלטים.

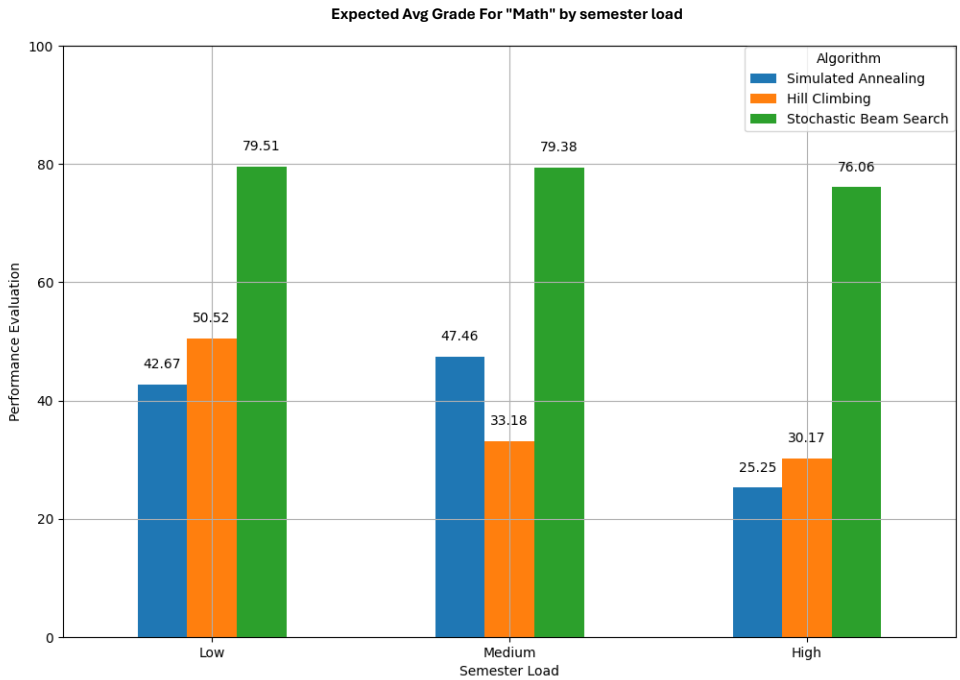
כדי להמחיש את איכות תוצאות האלגוריתמים (ממוצע למול יחס תוכניות חוקיות), נעריך את הביצועים בעזרת תוחלת הממוצע שמשיגים האלגוריתמים (בהנחה שתוכניות לא חוקית מביאה ממוצע 0).

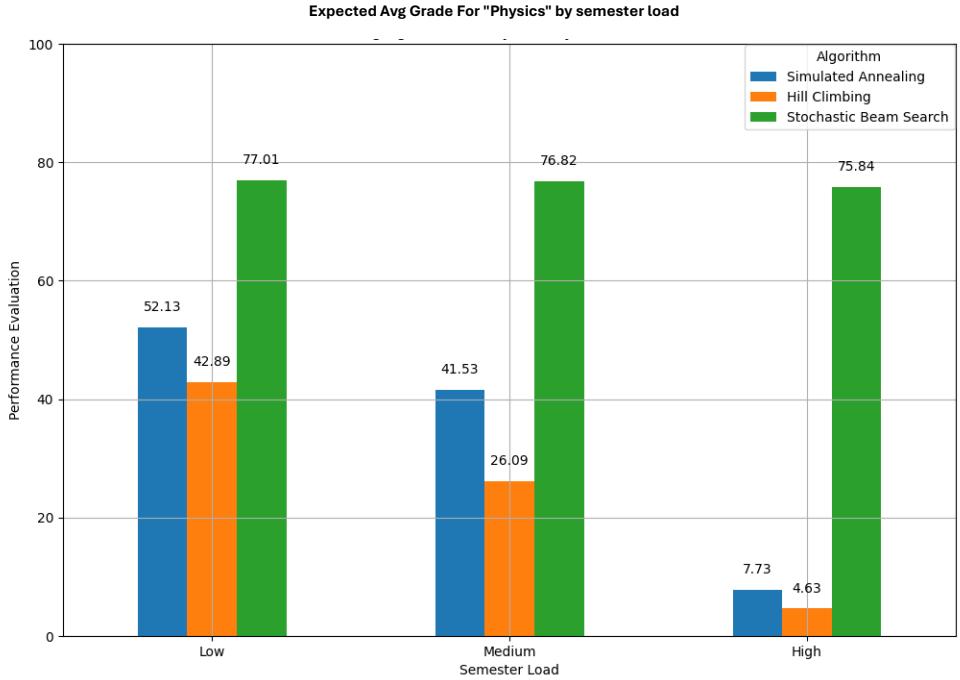
ערך כל תוחלת כנ"ל נקבע על ידי הנוסחה: $E = Legal\ Result\ Ratio\ *\ Average\ Grade$

להלן התוצאות:









מגרפים אלו ניתן להבחין שמבחינת תוחלת הממוצע של האלגוריתמים, ברוב המקרים מתקיים:

$$Hill\ Climbing\ <\ Simulated\ Annealing\ <\ Stochastic\ Beam\ Search$$

תוצאות אלו מתיישבות עם התיאוריה שכן $Hill\ Climbing$ מתכנס מהר למינימום מקומי קרוב, לעומת זאת $Simulated\ Annealing$ חוקר סביבה גדולה יותר ולכן סביר שימצא מינימום לוקלי טוב לפחות כמו $Hill\ Climbing$. בנוסף $Beam\ Search$ משתמש בסוכנים רבים העוזרים לו לחקור את המרחב בצורה נרחבת יותר.

חיפוש בגרף

עבור בעיה זו, האלגוריתם A* Search היה האלגוריתם בו בחרנו להשתמש, ואותו השווינו לשני אלגוריתמי בסיסיים - DFS, אשר מוצא תוכנית לימודים חוקית שרירותית שאינה בהכרח אופטימלית, ו- UCS, אשר מוצא תוכנית לימודים חוקית ואופטימלית.

DFS:

אלגוריתם זה סיים והחזיר פתרון חוקי לכלל הקלטים- לפי איך שהבעיה מוגדרת, אם הוא מחזיר פתרון אז 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 זה 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.

A* Search:

לאחר ניסיונות רבים לבניית היוריסטיקה עבור אלגוריתם זה, הגענו להיוריסטיקה אדמיסבילית אך לא קונסיסטנטית, לכן לא מחזירה בהכרח פתרון אופטימלי, שכן הבעיה שלנו מודלה כבעיית חיפוש בגרף ולא בעץ (שכן ישנן מספר דרכים שאיתן ניתן להגיע לאותה תכנית לימודים).
בחרנו בהיוריסטיקה זו, מכיוון שהביאה לממוצע הטוב ביותר, ביחס לזמן הריצה. היוריסטיקה מבצעת הערכת חסר לכמות הנקודות שתוכנית נתונה עלולה להפסיד מהממוצע הסופי, ממצב מסוים (תכנית חלקית) ועד בניית תוכנית לימודים שלמה. כלומר, היא 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 לא מובטח לנו פתרון אופטימלי.

השוואות זמני הריצה והקודקודים שנחקרים

להלן טבלה בה ניתן לראות את זמני הריצה השונים ומספר הקודקודים שנחקרו בריצת אלגוריתמי Graph Search השונים:

Time (Sec)			**Expanded**			**Input**	
DFS	A*	UCS	DFS	A*	UCS	Dataset	Semester Load
0.01	1300	4H+	63	1701048	4M+	CS1	Low
0.01	140	4H+	63	315619	4M+	CS2	Low
0.01	75	4H+	65	100324	4M+	Math	Low
0.01	4H+	4H+	43	4M+	4M+	Physics	Low
0.005	10	150	24	24259	1260027	Mini CS	Low
0.001	14	300	64	10021	1700926	Mini Math	Low
0.09	2	280	921	363	957805	Mini Physics	Low
0.05	220	4H+	381	546530	4M+	CS1	Medium
0.05	15	4H+	381	48811	4M+	CS2	Medium
0.01	10	4H+	94	17010	4M+	Math	Medium
0.06	4H+	4H+	363	4M+	4M+	Physics	Medium
0.15	1	13	2105	3538	135912	Mini CS	Medium
0.8	55	100	10532	192605	1134098	Mini Math	Medium
0.5	0.02	120	10466	206	957805	Mini Physics	Medium
0.03	115	4H+	139	353657	4M+	CS1	High
0.03	20	4H+	139	51729	4M+	CS2	High
10	5	4H+	65958	10405	4M+	Math	High
21	4H+	4H+	105683	4M+	4M+	Physics	High
0.007	2	2	72	10092	21960	Mini CS	High
0.01	1	10	163	3327	52509	Mini Math	High
X	X	X	X	X	X	Mini Physics	High

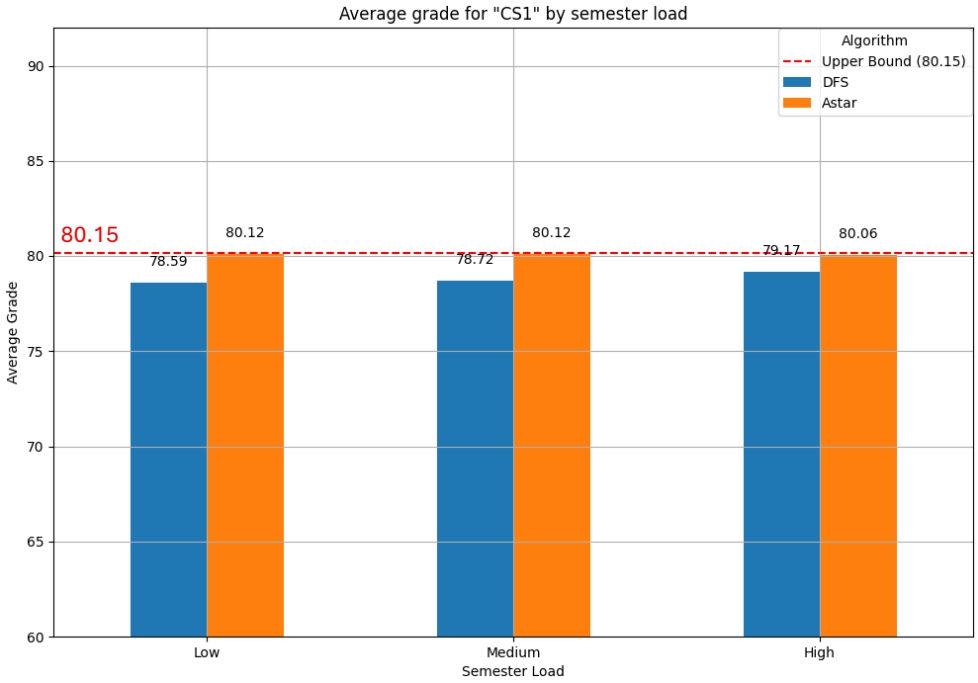
ניתן לראות בבירור, שברוב המוחלט של המקרים, מתקיים עבור זמני הריצה והקודקודים שנחקרו: DFS < A* < UCS.

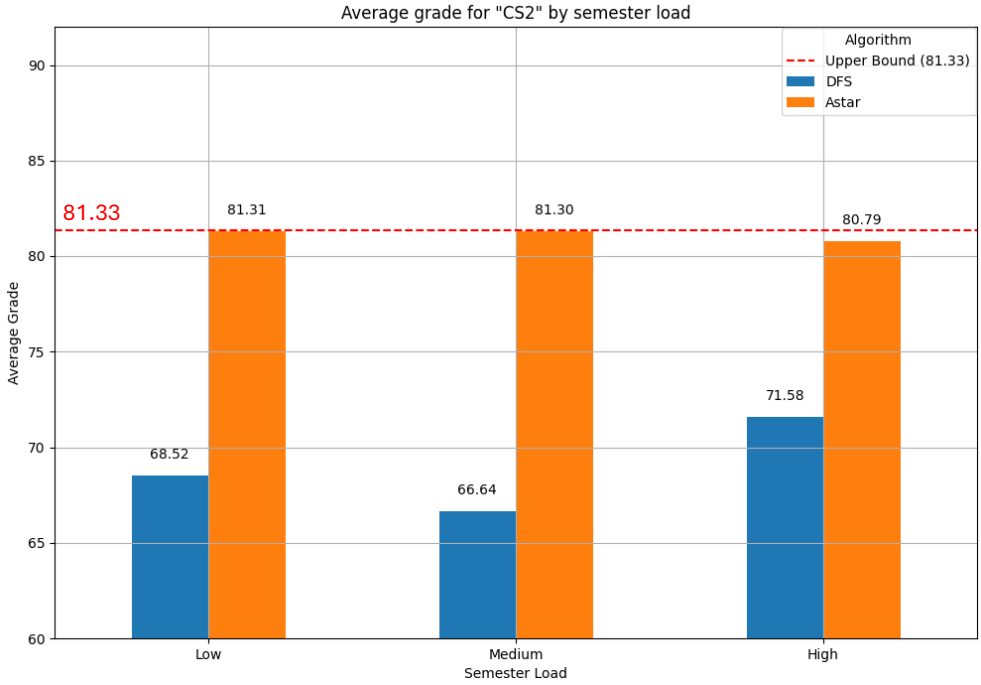
בנוסף, נבחין שזמני הריצה של A* משמעותית טובים יותר מאשר של UCS.

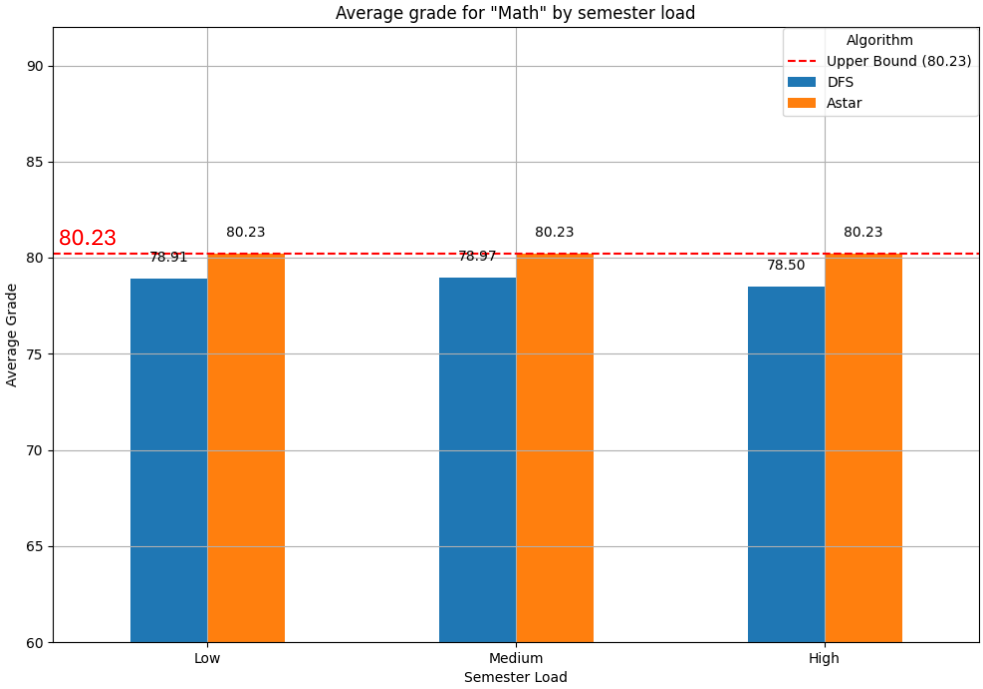
עוד נציין, שעל קובץ הקלט "Mini Physics" האלגוריתמים החזירו שלא קיים פתרון כאשר נבחר עומס גבוה בסמסטר שכן לא קיים פתרון בתנאים אלו עבור קלט זה.

השוואת הממוצעים המתקבלים

ראשית נציג את התוצאות עבור קבצי הקלט הגדולים בהסתכלות על הממוצע של תכנית הלימודים שמתקבלת, כפי שהסברנו קודם, עבור קלטים אלו, UCS לא עצר ולכן את איכות התוצאה השווינו לחסם מלעיל שחישבנו (שאינו בהכרח בר השגה בתוכנית חוקית).





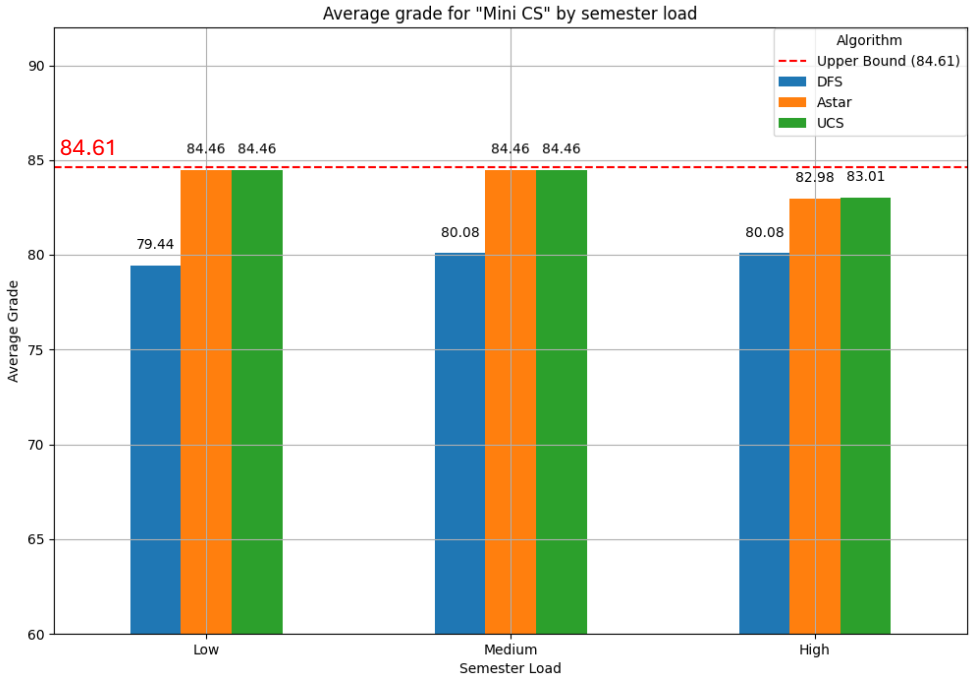


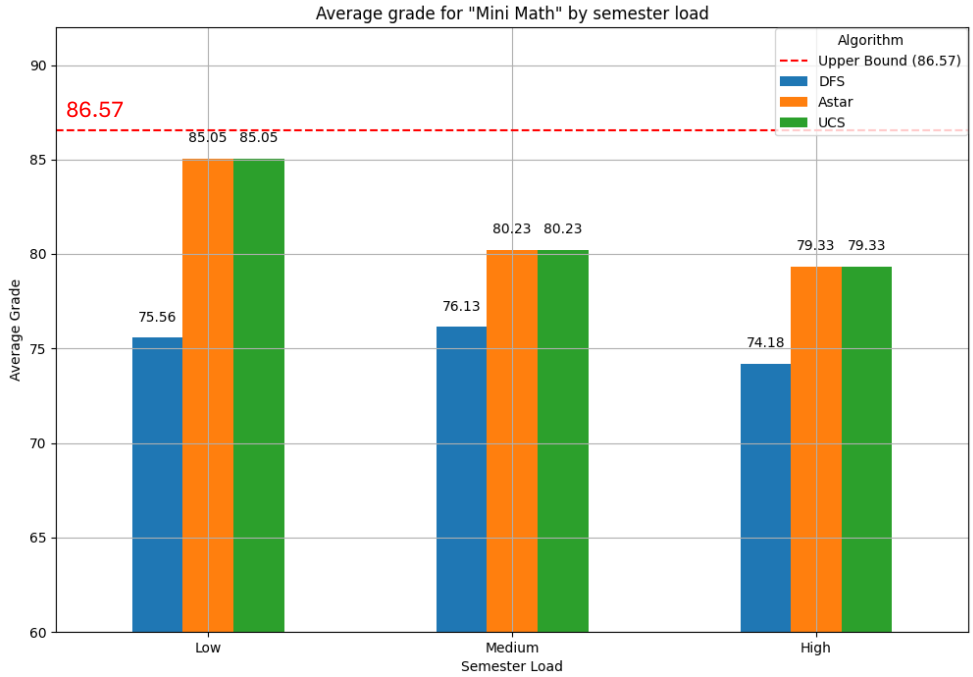
נבחין שבשלושת קבצי הקלט הנ"ל, התוצאה שמתקבלת על ידי A* טובה יותר מאשר האחת שמתקבלת על ידי DFS. בנוסף, התוצאות של A* מגיעות קרוב מאוד לחסם מלעיל ובקובץ קלט אחד אפילו מגיעות אליו ממש – כלומר מתקבל פתרון אופטימלי. נציין שמכיוון שהחסם מלעיל לא בהכרח בר השגה, ייתכן כי תוצאות ריצה נוספות של A* הגיעו לפתרון אופטימלי.

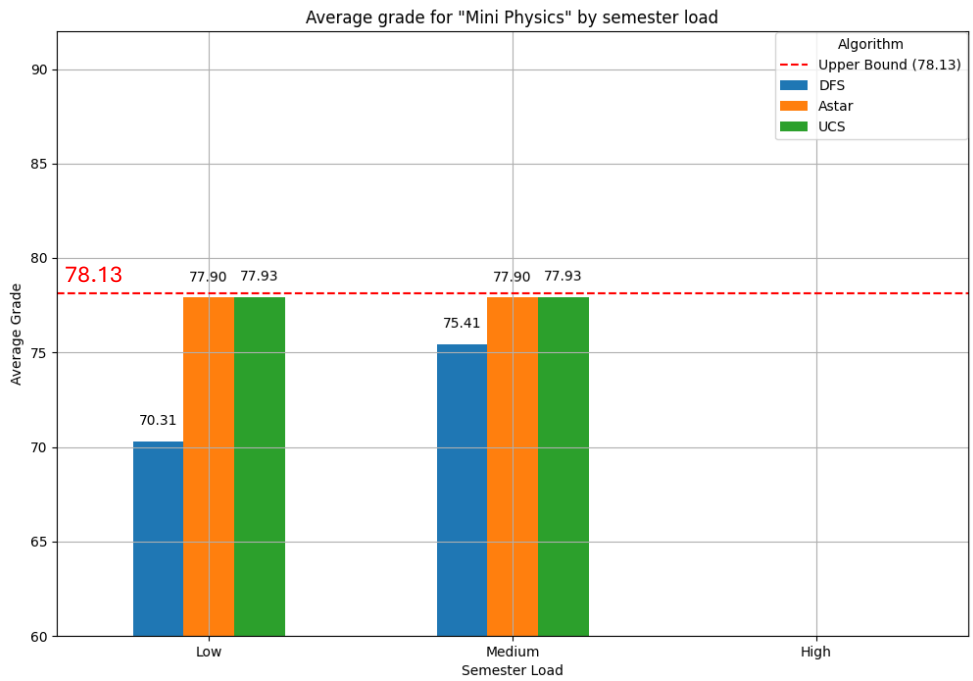
נשים לב שעל קובץ הקלט "CS2" התקבלו ההפרשים הגדולים ביותר, וזאת מפני שהשונות של הציונים שם גבוהה מאוד ביחס לשאר הקלטים, ולכן השפעה גדולה יותר לאלגוריתם שמתחשב במשקלי הצלעות (בניגוד לקבצים עם שונות נמוכה שבהן כל תכנית לימודים חוקית תהיה בטווח ציונים מצומצם יחסית).

חשוב לציין שלא צירפנו גרף עבור קובץ קלט גדול נוסף שבדקנו (Physics), שכן אלגוריתם A* לא עצר בזמן סביר עבורו – דבר זה אפשרי שכן האלגוריתם לא מבטיח קיצור זמנים ביחס ל-UCS עבור כל קלט.

להלן התוצאות עבור קבצי הקלט הקטנים בהסתכלות על הממוצע של תכנית הלימודים שמתקבלת, כפי שהסברנו קודם, עבור קלטים אלו, UCS עצר ולכן ניתן למדוד את איכות התוצאה של A* לתוצאות שמתקבלות על ידיו.







נבחין שבשלושת קבצי הקלט הנ"ל, התוצאה שמתקבלת על ידי *A טובה יותר מאשר האחת שמתקבלת על ידי DFS. בנוסף, התוצאות של *A מגיעות קרוב מאוד לתוצאות של UCS וברוב הפעמים אף מגיעות אליו ממש – כלומר מתקבל פתרון אופטימלי.
כפי שהוסבר קודם, על קובץ הקלט "Mini Physics" לא התקבלו תוצאות עבור אף אלגוריתם כאשר נבחר עומס גבוה בסמסטר שכן לא קיים פתרון בתנאים אלו עבור קלט זה.

למרות שההיוריסטיקה שלנו אינה קונסיסטנטית, ניתן לראות מהתוצאות שהצלחנו להגיע לממוצעים טובים ואף אופטימליים וכן קיצרנו את זמני הריצה באופן משמעותי – ולכן נסיק שהיוריסטיקה שבחרנו נותנת הערכה טובה.

השוואה בין שתי השיטות

נראה שאלגוריתם *A הגיע לתוצאות טובות בהרבה מבחינת הממוצעים שהתקבלו לעומת אלגוריתמי Local Search שבדקנו. בנוסף, לפי צורת המידול שבחרנו בה, חיפוש בגרף מבטיח לנו פתרון חוקי אם קיים בעוד שחיפוש מקומי אינו מבטיח זאת.
עם זאת, עבור קלטים רבים אלגוריתמי Local Search סיימו מהר יותר את ריצתם לעומת *A (מלבד Beam Search) וכן החזירו פתרון עבור כל הקלטים בניגוד ל-*A (שלא סיים עם Physics).

סיכום

הבעיה שבחרנו לפתור היא בעיית אופטימיזציה של תכנון לימודים אקדמיים. המטרה היא למצוא תוכנית לימודים חוקית ואופטימלית לתואר, שתמקסם את הממוצע הכללי. זאת כדי לסייע לסטודנטים בתכנון מסלול לימודים שיאפשר להם להשיג את הממוצע הטוב ביותר.

הנתונים כוללים:

1. כמות נקודות הזכות הדרושות למסלול.
2. רשימת הקורסים המוצעים עם פרטיהם (שם, מספר, נ"ז, דרישות קדם, ממוצע, סמסטר, חובה/בחירה).
3. דרגת עומס מוגדרת לכל סמסטר (מבחינת כמות נ"ז).

האתגר הוא למצוא תוכנית לימודים שעומדת בכל הדרישות האקדמיות והלוגיסטיות, תוך מקסום הממוצע הכללי בתואר שכן זוהי בעיה נפוצה בקרב סטודנטים ובו בזמן מסווגת כ- NP-Hard.

את הבעיה ניסינו לפתור בצורה יעילה שתביא לתוצאות טובות באמצעות כלים היוריסטיים שלמדנו בקורס – תחילה עם אלגוריתמי חיפוש מקומי שונים:

$Stochastic\ Beam\ Search, Simulated\ Annealing, Hill\ Climbing$,

שמתחילים עם תוכנית לימודים רנדומלית ועוברים בין שכנים באמצעות שינויים קטנים לעבר פתרון טוב ככל הניתן.

לאחר מכן, ניסינו לפתור את הבעיה באמצעות חיפוש בגרף (A^*, DFS, UCS). בו אנו מתחילים מתוכנית לימודים ריקה ובכל שלב מנסים להוסיף קורס חדש למערכת בצורה חוקית ועד להגעה לתוכנית מלאה. במידול זה, המסלול הזול ביותר מייצג את תוכנית הלימודים עם הממוצע הטוב ביותר.

לאחר השוואה בין שתי השיטות, ניתן לראות שחיפוש מקומי הביא לרוב לזמני ריצה קצרים יותר מאשר A* (מלבד $Beam\ Search$), עם זאת A* הביא לתוצאות טובות יותר מבחינת חוקיות וממוצע ואף ברוב המקרים הגיע לתוצאה האופטימלית (או קרוב מאוד אליה).

ניתן להסביר זאת מהעובדה שבמידול שבחרנו לחיפוש מקומי, פונקציית ה-$Fitness$ הייתה אחראית על יותר ממקסום הממוצע אלא גם על חוקיות התוכנית, בעוד שב-A* ההיוריסטיקה התבססה אך ורק על מקסום הממוצע. דבר זה ככל הנראה פגע ביכולת של אלגוריתמי החיפוש המקומי למקסם את הממוצע. בנוסף, ניתן להבחין שבבעיה זו ישנו מספר רב של מקסימום מקומיים דבר שאלגוריתמי Local Search רגישים אליו יותר מאשר חיפוש בגרף.

כאשר הורדנו את הדרישה לקביעת עומס בסמסטר, ראינו שזמן הריצה של A* גדל משמעותית ברוב המקרים, מכיוון שכמות המצבים האפשריים גדלה מאוד והיו הרבה יותר מסלולים לחקור.

בנוסף, באוניברסיטה בפועל ניתן לחרוג מכמות הנ"ז הדרושה לזכאות לתואר, אך אם נוריד את הדרישה אצלנו, נצטרך להגדיר פתרון אופטימלי בצורה נרחבת יותר- האם כלל הנ"ז שנלמדו יכנסו לשקלול הממוצע, התייחסות לקורסי החובה לעומת הבחירה, וכן איך חישוב פונקציית ה- Cost יחושב בצורה שונה שכן בדרך שבה מידלנו את פונקציית ה- Cost, החישוב שלה משתמש בערך הנ"ז של המערכת השלמה, שבמקרה זה לא יהיה ניתן לחזות מראש.

שיטות נוספת לפתרון הבעיה שלנו, שלא נבחרו, יכולות להיות Genetic Algorithm שראינו בעבודות קודמות שנעשו בנושא, וכן Reinforcement Learning.

את הבעיה ניתן להכליל למקסום כל מאפיין שניתן לייצג בטווח [0, 100].
כך לדוגמה, על ידי דירוג מרצים, נוכל למצוא תוכנית לימודים עם המרצים הטובים ביותר, וכן על ידי דירוג רמת עניין, נוכל למצוא תוכנית לימודים עם הקורסים הכי מעניינים וכן הלאה.