שיבוץ קורסים במערכת שעות אקדמאית

מגישות:

יובל רפפורט 208892471

סתו דודו 209339472

תוכן עניינים

מבוא קצר

שיבוץ קורסים במערכת שעות אקדמאית במחלקות גדולות באוניברסיטה הוא בעיה קשה אשר בהרבה אוניברסיטאות בארץ ובעולם עדיין נפתרת על ידי עבודה ידנית סיזיפית.
בעבודה זו נתמודד עם בעיה זו בעזרת כלים מתחום האלגוריתמים האבולוציוניים אותם למדנו בכיתה במסגרת הקורס וכן הרחבנו את הידע שלנו באופן עצמאי לאורך הפרויקט.

הרעיון הכללי של הפרויקט הוא יצירת מערכת שעות שבועית (ראשון – חמישי בשעות 18-08 בחלונות של שעתיים להרצאה) בה ישובצו כל הקורסים הנדרשים בהתאם למידע הנתון : מרצים אפשריים, שעות שבועיות, ומספר סטודנטים בקורס.

על פי עקרונות האבולוציה, נייצר תחילה כמות גדולה של מערכות שעות אקראיות ללא התחשבות בהגבלות שיבוץ ולאחר מכן נשפר את המערכות מדור לדור עד להגעה למערכת שעות אופטימלית.

בפרויקט זה ננסה לפתור את הבעיה בשפת פייתון באמצעות שימוש בחבילת ECKity.

תיאור הבעיה

הנחות העבודה

- מספר חלונות הלימוד בשבוע הוא לפחות כמספר שעות ההרצאה לשיבוץ
- מספר הכיתות הניתן מאפשר שיבוץ חוקי של ההרצאות לשיבוץ
- אין העדפה של ימים/שעות לקורסים ולמרצים
- לכל קורס לשיבוץ קבוצת הרצאה אחת בלבד
- אין שני מרצים עם אותו שם
- אין שני חדרים עם אותו מספר
- בקורסים עם כמה הרצאות בשבוע יוכלו ללמד מספר מרצים שונים מתוך מרצי הקורס באותו השבוע

הגבלות העבודה

- בכיתה יש יותר מקומות ישיבה מכמות סטודנטים מקסימלית בקורס המשובץ בכיתה
- מרצה לא יכול ללמד שני קורסים במקביל
- לא ניתן לשבץ שני קורסים באותה כיתה במקביל

<u>קלט הבעיה</u>: קובץ אקסל, "Data", עם שני גיליונות:

- "Courses" - גיליון קורסים בו יפורטו לכל קורס שם, מרצים, כמות סטודנטים, מספר הרצאות בשבוע.
- "Rooms" - גיליון חדרים בו יפורט לכל חדר אפשרי מספר מקומות בחדר

דוגמה לקלט:

name	teachers	max_students	windows
Introduction to CS	Dr. Shemesh, Dr. Keisar, Dr. Kudish	100	2
Data Structures	Proff. Carmi, Dr. Shemesh	100	2
Discrete Structures	Dr. Stein, Dr. Ben-Daniel, Dr. Rubin, Dr. Neiman	100	2
Computational Models	Dr. Fismanm, Dr. Ben-Daniel	100	2
SPL	Dr. Kogen, Dr. Adler	80	2
PPL	Dr. Adler, Dr. Elhadad, Dr. Gonen	80	2
Algorithms	Dr. Stein, Dr. Ben-Daniel	100	2
Numerical Analysis	Dr. Sapir	80	2
Comipler Principles	Dr. Goldberg	60	2
Principles of CS	Dr. Neiman	60	2
Machine Learning	Dr. Sabto	60	2
Natural Language Processing	Dr. Adler	40	1
Computer Graphics	Dr. Sharff	40	1
AI	Dr. Shimoni	40	2
Animation	Dr. Grossinger	40	1
Information Security	Dr. Orlov	30	1
Web Programming	Dr. Ganaim	30	2
Logics	Dr. Shkop, Dr. Kojman, Dr. Smith	100	2
Cryptographi	Dr. Beimel	35	2

number	capacity
7	100
114	50
35	40
1	100
2	80
42	100
10	40
12	35
55	90
34	80

<u>פלט הבעיה</u>: תדפיס המתאר את שיבוץ המערכת השבועית האופטימלית כאשר השאיפה היא לייצר מערכת שעות אשר כל הגבלות העבודה מתקיימות בה. במידה ולא ניתן ליצור מערכת כזו במסגרת מספר הדורות שהוגדר לאלגוריתם, תוחזר מערכת השעות עם מספר השגיאות המינימלי בנוסף לטבלה המתארת את שגיאות השיבוץ.

דוגמה לפלט:

Course Name	Teacher	Time	Room
Introduction to CS	Dr. Kudish	SUN 12-14	40
Introduction to CS	Dr. Shemesh	TUE 16-18	1
Data Structures	Proff. Carmi	SUN 16-18	10
Data Structures	Dr. Shemesh	THU 14-16	90
Discrete Structures	Dr. Rubin	THU 10-12	1
Discrete Structures	Dr. Rubin	WED 12-14	114
Computational Models	Dr. Ben-Daniel	SUN 16-18	40
Computational Models	Dr. Ben-Daniel	MON 14-16	2
SPL	Dr. Kogen	TUE 14-16	90
SPL	Dr. Adler	WED 16-18	55
PPL	Dr. Adler	THU 12-14	114
PPL	Dr. Gonen	MON 12-14	34
Algorithms	Dr. Stein	THU 16-18	7
Algorithms	Dr. Ben-Daniel	WED 16-18	114
Numerical Analysis	Dr. Sapir	WED 10-12	42
Numerical Analysis	Dr. Sapir	WED 08-10	2
Comipler Principles	Dr. Goldberg	MON 12-14	90
Comipler Principles	Dr. Goldberg	THU 16-18	90
Principles of CS	Dr. Neiman	WED 14-16	10
Principles of CS	Dr. Neiman	THU 14-16	2
Machine Learning	Dr. Sabto	MON 12-14	40
Machine Learning	Dr. Sabto	THU 08-10	2
Natural Language Processing	Dr. Adler	SUN 14-16	34
Computer Graphics	Dr. Sharff	TUE 16-18	114
AI	Dr. Shimoni	TUE 10-12	42
AI	Dr. Shimoni	WED 14-16	55
Animation	Dr. Grossinger	MON 12-14	2
Information Security	Dr. Orlov	THU 14-16	12
Web Programming	Dr. Ganaim	WED 12-14	35
Web Programming	Dr. Ganaim	TUE 12-14	55
Logics	Dr. Kojman	MON 10-12	90
Logics	Dr. Smith	MON 08-10	40

bugs:

Lecture	Bug
['OOP', 'Dr. Meri', 'TUE 08-10', 'room: 42']	bug: room double booked
['Algebra 1', 'Dr. Efrat', 'THU 10-12', 'room: 35']	bug: capacity

תיאור הפתרון

מהלך ריצת האלגוריתם ופתרון הבעיה

- בקובץ הרצת התוכנית, EvolutionarySchedules.py, נבנה אובייקט מטיפוס SimpleEvolution אשר אחראי על ניהול והרצת האלגוריתם האבולוציוני.
- האובייקט מורכב מתת אוכלוסייה אחת בפרויקט שלנו אשר לה creators, evaluators, size, genetic operators, selection methods כמו כן לאובייקט יש breeder and a termination checker.
- האלגוריתם מופעל על ידי קריאה למתודת evolve ממחלקת SimpleEvolution אשר מתחילה את תהליך האבולוציה המוכתב בחבילת ECKity.
- התהליך מפתח דורות בהתאם להתנהגות אלגוריתמים אבולוציוניים כפי שלמדנו בכיתה כאשר המטרה בפרויקט שלנו היא להגיע לפרט (מערכת שעות) בעלת fitness נמוך ביותר עם כמה שפחות שגיאות בשיבוץ ההרצאות.
- בסוף התהליך, בהגעה למספר דורות מקסימלי שהוגדר או הצלחה במשימה, האלגוריתם ידפיס את מערכת השעות בעלת ה-fitness הנמוך ביותר שהתקבל בנוסף לטבלה המציגה את שגיאות השיבוץ במערכת השעות.

Genetic Algorithm

מבנה הפרט – מערכת שעות המיוצגת ע"י אובייקט מסוג Schedule המומש בפרויקט.
אובייקט זה מכיל שדה של רשימה של אובייקטים מסוג Lecture כאשר כל הרצאה כזו מורכבת מקורס, מרצה, חלון זמנים וכיתה. בנוסף מכיל שדה של רשימת שגיאות בשיבוץ לוח השעות.

מבנה האוכלוסייה – רשימה באורך n של אובייקטים מטיפוס Schedule שנבנו באופן רנדומלי על ידי ScheduleCreator שיצרנו אשר מרחיב את ה-Creator של ECKity.

האבולוציה –

- **Fitness** – בתחילת כל דור מתבצעת הערכה של הפרטים באוכלוסייה על ידי ה – ScheduleEvaluator המרחיב את SimpleIndividualEvaluator של ECKity.
הערכה מוגדרת על ידי הגבלות העבודה כך שכל הגבלה שלא מתקיימת, כלומר שגיאה בשיבוץ, נוסיף 1 לערך ה – Fitness של הפרט.
- **Selection** – השתמשנו ב – Tournament Selection הממומש ב – ECKity עם tournament size = 3 כאשר higher is better = false. בנוסף, ביצענו בחירת אליטיזם של פרט יחיד מתוך האוכלוסייה בעל ה- fitness הנמוך ביותר (הטוב ביותר).
- **Mutation** – בתחילת כל דור תתבצע מוטציה על פרטי האוכלוסייה בהסתברות P, אשר נקבעת בבניית אובייקט SimpleEvolution. המוטציה מתבצעת באמצעות אופרטור MutateSchedule שמימשנו אשר מרחיב את GeneticOprator של ECKity.
הרחבה נוספת על מהות המוטציה בסעיף "מבט על של התוכנה".

- **Crossover** – בתחילת כל דור תתבצע פעולת crossover בין הפרטים באוכלוסייה בהסתברות P, אשר נקבעת בבניית אובייקט SimpleEvolution. הפעולה מתבצעת באמצעות אופרטור ScheduleCrossover שמימשנו אשר מרחיב את GeneticOperator של ECKity.
הרחבה נוספת על הפעולה בסעיף "מבט על של התוכנה".

מבט על של התוכנה

- **EssentialClasses** – בקובץ זה ממומשים כל האובייקטים החיוניים לריצת האלגוריתם בפרויקט שלנו.
לכל אובייקט בפרויקט getters and setters רלוונטיים על פי שדותיו.

פירוט המחלקות:
Schedule – האובייקט המרכזי בפרויקט אשר מייצג מערכת שעות שהיא פרט באוכלוסייה שלנו.
מערכת שעות מורכבת משדה Lectures, שהוא רשימה של אובייקטים מטיפוס הרצאה.
בנוסף, לאובייקט שדה bug_list, שהוא רשימה של שגיאות שנמצאו בשיבוץ לוח השעות.
מלבד בנאי לאובייקט ישנה שיטת Initialize_random אשר יוצרת Schedule חדש באופן אקראי ע"י מעבר על כל הקורסים הנדרשים לשיבוץ לפי ה- Data (הרחבה בהמשך) תוך בניית אובייקטים של Lectures כך: לכל קורס ייבחרו באופן אקראי מורה מתוך רשימת המורים המלמדים את הקורס, חדר מתוך רשימת החדרים וחלון זמנים מתוך רשימת חלונות הזמנים. כל זאת ללא התחשבות בהגבלות עבודה. כל הרצאה כזו נוספת לרשימת ההרצאות של לוח השעות הנ"ל.

Lecture – אובייקט המייצג הרצאה בודדת אשר מורכבת מקורס, מורה, חדר וחלון זמנים.

Course – אובייקט המייצג קורס לשיבוץ במערכת אשר מורכב משם הקורס, רשימת מורים מלמדים ומספר סטודנטים מקסימלי בקורס.

Teacher – מורה מיוצג על ידי string.

Room – אובייקט המייצג חדר אשר מורכב ממספר חדר ותפוסה מקסימלית.

CourseWindow – אובייקט המייצג חלון זמנים אשר מורכב מיום ושעות לשיבוץ כאשר חלונות השיבוץ הינם בני שעתיים.

Data – אובייקט גלובלי המחזיק את המידע הנדרש לשיבוץ מערכת שעות אשר מוגדר ע"י המשתמש בתוכנית. במחלקה זו מתבצעת קריאה של המידע מתוך קובץ אקסל מהמשתמש המתואר בקלט הבעיה.
בנוסף נבנים אובייקטים של חדרים וקורסים לשיבוץ בהתאם למידע המתקבל ושל חלונות זמן לפי ימות השבוע וזמנים רלוונטיים.

- **EvolutionarySchedules** – זהו קובץ ההרצה של הפרויקט בו בונים אובייקט מסוג SimpleEvolution שמקבל את הפרמטרים הבאים:
  - אובייקט חדש מטיפוס subpopulation עם הפרמטרים:
    - Creator מסוג ScheduleCreator שמומש בפרויקט.
    - Population_size – 300/500/1000.
    - Evaluator מסוג scheduleEvaluator שמומש בפרויקט.
    - Higher_is_better=false כיוון שמדובר בבעיית מינימיזציה.
    - Elitism_rate שמשתנה בהתאם לגודל האוכלוסייה, אנחנו בחרנו פרט אחד מהאוכלוסייה.
    - Operator_sequence: מערך המקבל את האופרטורים הגנטיים ScheduleCrossover שמימשנו בפרויקט ומבצע crossover בהסתברות של 0.8, MutateSchedule שמימשנו בפרויקט ומבצע מוטציה בהסתברות של 0.2.
    - Selection_methods – TournamentSelection כאשר tournament_size=3
  - Breeder – SimpleBreeder
  - Max_workers = 4
  - Max_generations = 100
  - Termination_checker – ThresholdFromTargetTerminationChecker שמקבל כפרמטרים optimal = 0, threshold = 0.0

תהליך הפתרון מופעל באמצעות קריאה למתודה evolve.

- **ScheduleCreator**: מלבד לבנאי מימשנו את המתודה create_individuals שמקבלת כפרמטר את גודל האוכלוסייה, n, ומייצרת בהתאם מערכות שעות ריקות ומפעילה על כל schedule את המתודה initialize_random ולבסוף מכניסה את n מערכות השעות האקראיות למערך ומחזירה אותו.

- **ScheduleEvaluator**: מלבד לבנאי מימשנו את המתודה evaluate_individual_ שמקבלת כפרמטר Schedule להערכה ופועלת באופן הבא:
במתודה נבנה שני מילונים המשמשים לאיתור שגיאות בלוח השעות – מילון הממפה בין מורה לחלונות זמן בהם שובץ ללמד ומילון הממפה בין חדר לחלונות זמן בהם שובץ.
בנוסף נאתחל רשימת באגים בשיבוץ ומונה באגים.
עבור כל הרצאה בלוח השעות נבדוק:
  - אם מספר התלמידים המקסימלי בקורס גדול מתפוסת הכיתה ששובצה נמנה שגיאה ונוסיפה לרשימת השגיאות.
  - עבור חלון הזמן ששובץ למורה בהרצאה זו נבדוק ייתכנות. בדיקה זו מתבצעת באמצעות המילון שצוין לעיל כך שאם חלון הזמנים הנוכחי לא שובץ בעבר למורה זה נוסיפהו למילון, אחרת נמנה שגיאה ונוסיפה לרשימת השגיאות.
  - עבור חלון הזמן ששובץ לחדר בהרצאה זו נבדוק ייתכנות. בדיקה זו מתבצעת באמצעות המילון שצוין לעיל כך שאם חלון הזמנים בנוכחי לא שובץ בעבר לחדר זה נוסיפהו למילון, אחרת נמנה שגיאה ונוסיפה לרשימת השגיאות.

לבסוף, נעדכן את שדה רשימת השגיאות של ה-Schedule עם רשימת השגיאות שנוצרה כעת ונחזיר את מונה השגיאות שהוא למעשה מדד ה-Fitness של ה-Schedule.

- **ScheduleCrossover** – הרעיון של ה-Crossover בפרויקט שלנו מיישם את הרעיון של VectorKPoints של ECKity. כלומר, עבור שני Schedules נקבל מספר נקודות, K, לחלוקה של וקטור ההרצאות ונחליף בין מקטעים של וקטורי ההרצאות של שני מערכות השעות לסירוגין.
מלבד לבנאי מימשנו את המתודה apply שמקבלת שני Schedules לביצוע crossover ופועלת באופן הבא: נבחר באופן אקראי K נקודות בהתאם לגודל וקטור ההרצאות של מערכת השעות.
נחלק את וקטורי ההרצאות של מערכות השעות ונחליף בין המקטעים של המערכות השונות לסירוגין.
נחזיר את מערכות השעות לאחר השינויים.

- **MutateSchedule** – מלבד לבנאי מימשנו את המתודה apply המקבלת כפרמטר Schedule עליו נרצה לבצע מוטציה הפועלת באופן הבא: נייצר Schedule רנדומלי חדש בעזרת המתודה initialize_random. עבור כל הרצאה של מערכת השעות המקורית בהסתברות של 0.3 נחליף בין הרצאה זו להרצאה ממערכת השעות הרנדומלית.

תוצאות והדגמות ריצה

את כלל הניסויים בצענו על דאטא הנמצא בקובץ Data.
את כל המידע לקחנו מקובץ הקורסים האוניברסיטאי של אוניברסיטת בן גוריון. הקורסים הינם מהמחלקות למדעי המחשב, מתמטיקה והנדסת חשמל.
ניתן לערוך את קובץ הדאטא בהתאם לצרכים לביצוע ניסויים שונים.

בחירת פרמטרים לניסויים

פרמטרים קבועים:

- מספר דורות בריצה – בחרנו להריץ כל ניסוי למשך 100 דורות מאחר שמצאנו כי מספר זה מאפשר להגיע לפתרון במידה וניתן לעשות זאת בזמן סביר ובמידה ופתרון לא יתקבל בזמן סביר הוא לא יתקבל ב-100 הדורות הראשונים.
- שיעור אליטיזם – לאחר בחירת גדלים לאוכלוסייה בצענו ניסיונות רבים להגיע לפתרון עם שיעורי אליטיזם שונים: 1/3/5 פרטים מתוך האוכלוסייה, 1/5 אחוזים מתוך האוכלוסייה. מניסיונות אלו עלה כי השיעור בעל התוצאות הטובות ביותר ביחס לגדלי האוכלוסייה הנבחרים הוא שיעור של פרט אחד מתוך האוכלוסייה.
- הסתברות ל-crossover ו-mutation – לאחר בחירת גדלים לאוכלוסייה בצענו ניסיונות רבים להגיע לפתרון עם הסתברויות שונות: 0.7/0.8/0.9 ל-crossover ו-0.3/0.2/0.1 ל-mutation בהתאמה. מצאנו כי ההסתברויות בעלות התוצאות הטובות ביותר ביחס לגדלי האוכלוסייה הנבחרים הן 0.8 ל-crossover ו-0.2 ל-mutation.

פרמטרים משתנים:

- גודל האוכלוסייה – בחרנו להריץ את הניסויים על אוכלוסייה בשלושה גדלים שונים: 300, 500 ו-1000. לאחר הרצה של ניסיונות רבים לבחירה של גדלי אוכלוסייה מצאנו כי גדלים אלו משקפים בצורה טובה את ההבדלים בתוצאות הניסויים וביכולת להגיע לפתרון לבעיה.
- מספר נקודות חיתוך ב-crossover – בחרנו להריץ את הניסויים עם crossover במספר נקודות חיתוך שונה: 1/2/3. מצאנו כי מספרים אלה של נקודות מיזוג משקפים בצורה טובה את ההבדלים בתוצאות הניסויים וביכולת להגיע לפתרון לבעיה.

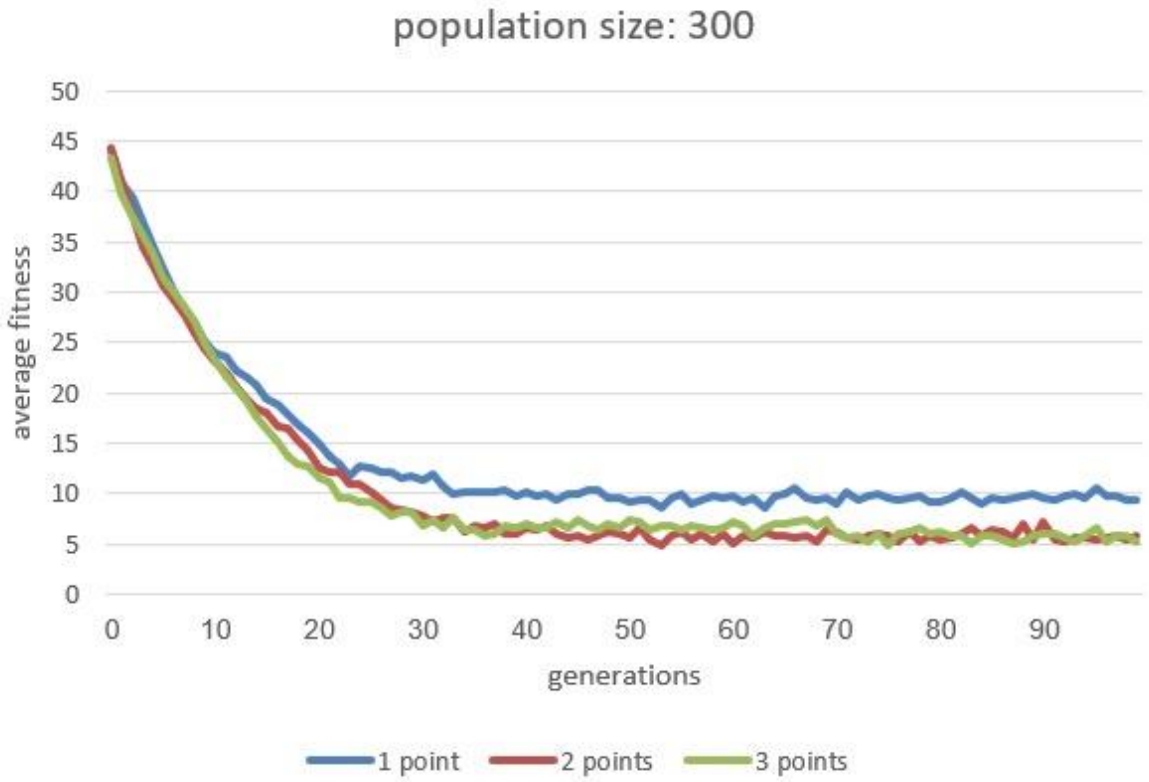
לאחר ביצוע 100 ניסויים לכל צירוף פרמטרים אפשרי, בחרנו להציג את הריצות שמהוות את החציון בקטגוריה שלהן.

בנוסף, בצענו ניסוי בו אנחנו קיבלנו את אותו דאטא כמו האלגוריתם וניסינו לשבץ מערכות שעות כנ"ל באופן ידני במקביל להרצת האלגוריתם עם הפרמטרים הטובים ביותר שנמצאו – אוכלוסיות גדולות ונקודות חיתוך מרובות ל-crossover.
תוצאות הניסוי היו שריצות האלגוריתם הצליחו באופן מהיר משמעותית מאתנו. האלגוריתם הגיע לפתרון תוך ממוצע של 2-3 דקות לעומת השיבוץ הידני שלנו שלקח ממוצע של 25 דקות.

להלן תוצאות וגרפים של ריצות החציון בניסויים:

גודל אוכלוסייה - 300

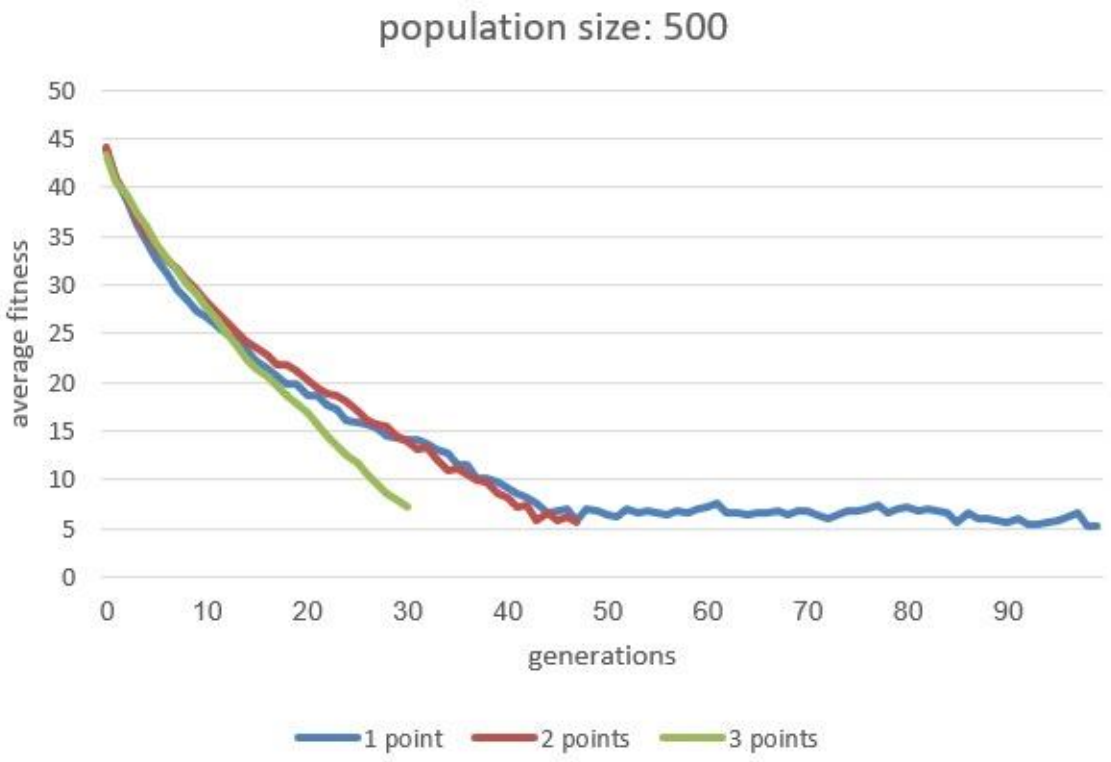
gen.	1 point	2 points	3 points	gen.	1 point	2 points	3 points	gen.	1 point	2 points	3 points	gen.	1 point	2 points	3 points
0	44.17	44.25	43.41	25	12.49	10.1	9.21	50	9.23	5.55	7.37	75	9.58	5.75	4.93
1	40.94	41.15	39.83	26	12.15	9.37	8.63	51	9.37	6.61	7.23	76	9.33	5.14	5.92
2	39.29	37.55	37.46	27	12.23	8.65	7.76	52	9.35	5.46	6.34	77	9.51	6.17	6.14
3	36.98	34.51	35.56	28	11.58	8.47	8.27	53	8.55	4.94	6.86	78	9.84	5.3	6.63
4	34.54	32.72	33.92	29	11.76	8.24	8.12	54	9.5	5.89	6.78	79	9.15	5.73	5.96
5	32.41	30.72	31.55	30	11.34	7.86	6.75	55	9.87	6.12	6.34	80	9.21	5.34	6.21
6	29.94	29.16	29.81	31	11.89	7.16	7.43	56	9.01	5.48	6.85	81	9.51	5.54	5.83
7	28.05	27.63	28.58	32	10.78	7.56	6.7	57	9.3	5.99	6.56	82	10.1	5.92	5.85
8	26.88	25.94	27.04	33	10.05	7.61	7.66	58	9.78	5.27	6.44	83	9.57	6.7	5.1
9	25.17	24.39	25	34	10.23	6.17	6.32	59	9.51	6.1	6.56	84	9.03	5.75	5.8
10	24.02	23.16	23.31	35	10.19	6.87	6.38	60	9.7	5.07	7.27	85	9.47	6.48	5.88
11	23.58	22.08	21.51	36	10.06	6.64	5.79	61	9.1	5.91	6.72	86	9.33	6.19	5.5
12	22.18	20.59	20.48	37	10.06	6.97	5.98	62	9.6	5.65	5.81	87	9.6	5.63	5.12
13	21.65	19.48	19.15	38	10.45	6.1	6.72	63	8.65	6.18	6.61	88	9.84	6.95	5.19
14	20.73	18.46	17.61	39	9.7	6.01	6.68	64	9.76	5.77	7.04	89	10.05	5.43	5.86
15	19.36	18.13	16.56	40	10.1	6.64	6.98	65	10.05	5.72	6.99	90	9.53	7.16	6.08
16	18.79	16.6	15.09	41	9.79	6.41	6.64	66	10.47	5.61	7.12	91	9.39	5.45	5.92
17	17.78	16.5	13.72	42	9.94	6.89	6.7	67	9.65	5.91	7.49	92	9.71	5.21	5.56
18	16.83	15.26	12.91	43	9.33	5.92	7.12	68	9.45	5.31	6.76	93	10.03	5.61	5.3
19	16.03	14.26	12.81	44	9.98	5.66	6.7	69	9.5	6.45	7.35	94	9.64	5.71	5.72
20	14.93	12.61	11.52	45	9.98	5.77	7.35	70	8.91	6.26	6.02	95	10.46	5.47	6.64
21	13.65	12.19	11.15	46	10.32	5.36	6.71	71	10.13	5.56	5.64	96	9.7	5.57	5.2
22	12.93	12.17	9.61	47	10.37	5.86	6.43	72	9.46	5.46	5.83	97	9.79	5.74	5.87
23	11.71	11.04	9.63	48	9.63	6.12	6.91	73	9.77	5.83	5.2	98	9.27	5.5	5.88
24	12.63	10.86	9.08	49	9.64	6.09	6.63	74	10.02	6.09	6.11	99	9.39	5.74	5.32



מסקנות מהגרפים: עבור גודל אוכלוסייה 300 הניסויים לא צלחו ולא הגענו לפתרון מושלם. ניתן לראות הבדל ברור ומשמעותי בתוצאות הניסוי בין שימוש ב-crossover עם נקודת חיתוך אחת לבין שתיים ושלוש נקודות, כאשר התוצאות עם שתיים ושלוש נקודות טובות יותר. בין שתיים לשלוש נקודות אין הבדל משמעותי בתוצאות הניסוי.

גודל אוכלוסייה - 500

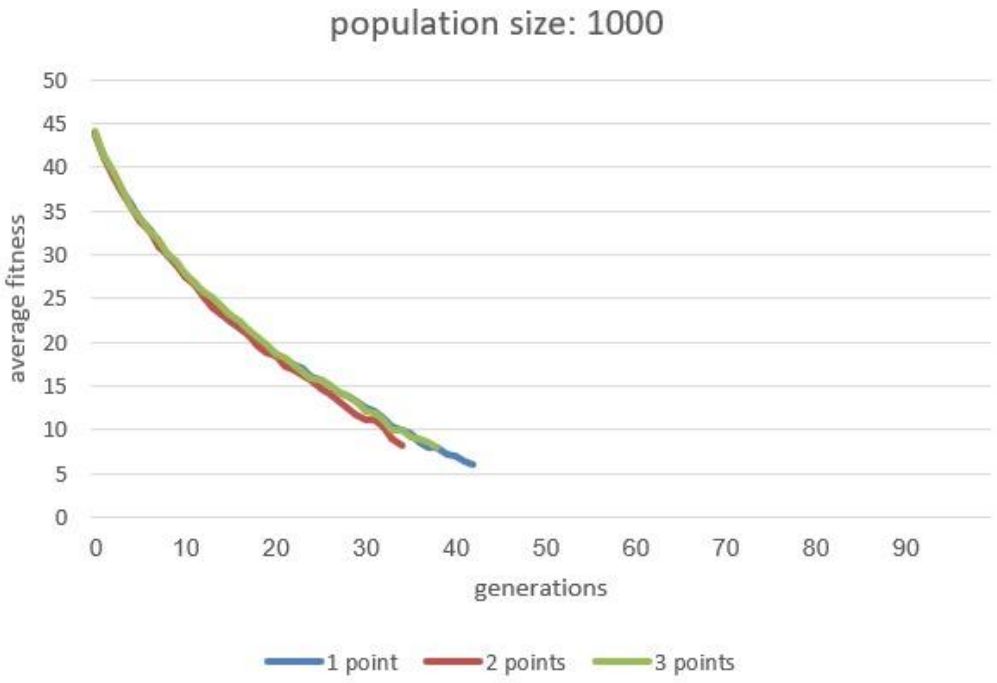
gen.	1 point	2 points	3 points	gen.	1 point	2 points	3 points	gen.	1 point	2 points	3 points	gen.	1 point	2 points	3 points
0	43.88	44.09	43.44	25	15.85	17.04	11.76	50	6.49			75	6.9		
1	40.97	41.05	40.54	26	15.74	16.16	10.5	51	6.21			76	6.93		
2	38.59	39.01	39.18	27	15.21	15.66	9.54	52	7.07			77	7.35		
3	36.25	36.81	37.35	28	14.52	15.41	8.51	53	6.54			78	6.52		
4	34.36	35.14	35.93	29	14.4	14.59	8.05	54	6.9			79	7.04		
5	32.67	33.82	34.23	30	14.02	13.99	7.19	55	6.7			80	7.19		
6	31.06	32.5	32.74	31	14.2	13.13		56	6.48			81	6.79		
7	29.45	31.67	31.5	32	13.63	13.3		57	6.78			82	6.93		
8	28.44	30.45	30.06	33	13.1	12.05		58	6.59			83	6.86		
9	27.43	29.47	29.09	34	12.79	11.02		59	7			84	6.56		
10	26.71	28.26	27.76	35	11.63	11.09		60	7.22			85	5.69		
11	25.74	27.22	26.38	36	11.57	10.64		61	7.51			86	6.58		
12	24.88	26.1	24.97	37	10.07	9.92		62	6.68			87	5.95		
13	24.35	25.14	23.69	38	10.21	9.76		63	6.57			88	6.03		
14	23.45	24.19	22.42	39	9.83	8.54		64	6.39			89	5.78		
15	22.21	23.52	21.49	40	9.09	8.12		65	6.62			90	5.69		
16	21.47	22.77	20.66	41	8.61	7.26		66	6.56			91	6.01		
17	20.62	21.8	19.72	42	8.25	7.3		67	6.79			92	5.37		
18	19.85	21.82	18.57	43	7.51	5.9		68	6.43			93	5.33		
19	19.79	21.19	17.9	44	6.6	6.68		69	6.84			94	5.53		
20	18.73	20.31	16.82	45	6.82	5.82		70	6.81			95	5.84		
21	18.6	19.36	15.78	46	7.03	6.26		71	6.36			96	6.22		
22	17.71	18.9	14.45	47	5.85	5.62		72	6.11			97	6.53		
23	17.19	18.68	13.47	48	7			73	6.38			98	5.31		
24	16.17	17.99	12.49	49	6.86			74	6.73			99	5.29		



מסקנות מהגרפים: עבור גודל אוכלוסייה 500 ניתן לראות כי הריצות המייצגות שנבחרו הגיעו לפתרון מושלם עבור שימוש ב-2/3 נקודות crossover (סוף הגרפים הירוק והאדום). בנוסף ניתן לראות הבדלים משמעותיים בתוצאות בין נקודה אחת, שתי נקודות ושלוש נקודות כאשר יש מגמת שיפור ככל שמספר נקודות החיתוך גדול יותר.

גודל אוכלוסייה - 1000

gen.	1 point	2 points	3 points	gen.	1 point	2 points	3 points
0	43.84	43.98	44.2	25	15.72	14.72	15.76
1	41.34	41.12	41.43	26	15.1	14.03	15.03
2	39.26	38.82	39.31	27	14.38	13.31	14.31
3	37.26	36.97	37.36	28	13.95	12.54	13.92
4	35.83	35.49	35.44	29	13.38	11.69	13.28
5	34.14	33.77	34.17	30	12.61	11.11	12.19
6	32.79	32.7	32.75	31	12.12	11.18	11.89
7	31.27	30.98	31.61	32	11.28	10.29	11.2
8	29.87	30.11	30.14	33	10.27	8.98	9.99
9	28.84	28.84	29.32	34	9.95	8.16	9.96
10	27.73	27.5	27.88	35	9.53		9.24
11	26.79	26.48	26.68	36	8.51		8.91
12	25.42	25.19	25.69	37	8.05		8.53
13	24.39	24.06	25.07	38	8.04		7.95
14	23.34	23.27	24.21	39	7.18		
15	22.48	22.58	23.22	40	6.95		
16	21.67	21.7	22.33	41	6.35		
17	21.05	20.72	21.44	42	6.12		
18	20.21	19.6	20.68	43			
19	19.39	18.85	19.92	44			
20	18.6	18.45	18.59	45			
21	17.9	17.26	18.3	46			
22	17.49	16.82	17.5	47			
23	17.13	16.34	16.72	48			
24	16.13	15.63	15.97	49			



מסקנות מהגרפים: עבור גודל אוכלוסייה 1000 כל הריצות המייצגות שנבחרו הגיעו לפתרון מושלם תוך מספר דורות מועט יחסית. בנוסף, לא נראו הבדלים משמעותיים בתוצאות עבור מספר נקודות חיתוך שונה ל-crossover.

סיכום ומסקנות

מסקנות מכלל הניסויים:

- מצאנו כי גודל האוכלוסייה הינו גורם משפיע יותר על כמות הניסויים המצליחים לעומת הכושלים ועל כמות הדורות הנדרשת להגעה לפתרון לעומת מספר נקודות החיתוך לביצוע ה-crossover. זאת למרות שכן נראו הבדלים בניסויים עם מספר נקודות חיתוך שונה אך גודל האוכלוסייה גבר על כך.
- מצאנו כי באופן חד משמעי הניסויים שהגיעו להצלחה בפתרון הבעיה עשו זאת לאחר מגמת ירידה מונוטונית כמעט לחלוטין ב-average fitness. לעומת זאת, הניסויים שלא הגיעו להצלחה לאחר 100 דורות הגיעו לשלב מסוים בו ה-average fitness "נתקע" בצורה מחזורית, כלומר חווה עלייה וירידה סביב מספר מסוים באופן קבוע ללא ירידה מתחת למינימום מסוים בריצה וללא שיפור.
  מסקנה משמעותית שגזרנו מכך הינה שה-average fitness המינימלי עצמו בין הריצות המצליחות לעומת הכושלות לא בהכרח נמוך יותר, אלא מגמת הירידה המתמידה לעומת הכניסה למחזוריות היא הגורם המשפיע יותר.
- כיוון שיצירת הפרטים באוכלוסייה רנדומלית לחלוטין לא ניתן לקבוע באילו ריצות מדד ה-average fitness יכנס למחזוריות, ולכן ככל שהאוכלוסייה גדולה יותר החלק היחסי של הפרטים באוכלוסייה שלא יתקעו גדול יותר והסיכויים להצלחה גדולים יותר.
- מצאנו כי בכלל הניסויים שהגיעו לנקודת מחזוריות ב-fitness זה קרה לאחר סדר גודל של כ-40 דורות.
  בנוסף, בריצות שהגיעו לפתרון לבעיה זה קרה לאחר סדר גודל של <u>עד</u> כ-40 ריצות.

סיכום:

מצאנו כי בעיית שיבוץ מערכת שעות אקדמאית הינה פתירה בעזרת האלגוריתם האבולוציוני שמומש בפרויקט כתלות בגודל האוכלוסייה. כלומר, בהינתן דאטא מסוים לשיבוץ, העומד בהנחות העבודה של הפרויקט (שצוינו בתיאור הבעיה), קיים גודל אוכלוסייה עבורו ניתן ליצור מערכת שעות משובצת ללא שגיאות בהרצת האלגוריתם.

בנוסף בהתייחס לניסוי השיבוץ הידני לעומת השיבוץ באמצעות האלגוריתם האבולוציוני, ברור כי השיבוץ הממוחשב יעיל באופן משמעותי. הערכתנו היא שככל שגודל הדאטא לשיבוץ גדול יותר, הפערים בין שיבוץ ידני לבין שימוש באלגוריתם האבולוציוני לשיבוץ גדלים משמעותית.

<u>README to run project -</u>

- Make sure the "Data" excel file is saved in same source folder as the code
- In runtime environment/ cmd/ terminal from source folder run <python EvolutionarySchedules.py>
- To change data, edit the given "Data" excel with new parameters. Make sure not to rename the file, sheets and fields.