בס"ד

**פרויקט מבוא לאופטמיזציה**

**מבוא**

בעיית אופטמיזציה הינה סוג של בעיה המנסה למצוא ערך מקסימלי או מינימלי עבור בעיה עם פונקציית מטרה כלשהי ואילוצים. בעייה נקראת MOO כלומר בעיית אופטימזציה בעלת מספר פונקציות מטרה, כאשר אנו מנסים למצוא ערכים מקסימליים או מינימלים עבור יותר מפונקציה אחת.בעיות כאלה ניתן למצוא בעולם מדעי החיים, מדעי החברה וכמובן ביומיום.

בעבודה הבאה נציג מספר שיטות לטפל בבעיית אופטמיזציה כאשר ישנה יותר מפונקציית מטרה אחת. קושי שנראה שיחזור על עצמו הוא לדאוג שמציאת ערך מקסימלי לפונקציה אחת, לא תבוא על חשבון האחרת.  
  
**שיטת Weighted-sum method**

שיטת זו בעצם לוקחת כל פונקציות המטרה של אותה הבעיה ויוצרת מהם בעיה יחידה ע"י שימוש בוקטור משקלים אחד שיהיה יותר נוח לעבד אותו.  
למרות שהשיטה בעקרון נוחה לעבודה, נוצרות שתי בעיות:

1.בחירת המשקלים לוקטור עבור בעיות עם עוצמות שונות יוצרת BIAS במציאת הפתרון.

2.היא שאם הבעיה (הפונקציה) לא קמורה אז תיווצר בעיה במציאת הפתרון.

**שיטת epsilon method**

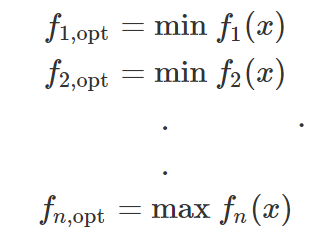
בשביל לפתור את הבעיות הנ''ל האלגוריתם משתמשת בשיטת אפסילון, שבאמצעותה כל פעם ניקח בעיה אחת מוקטור הבעיות ונעשה רק עליה אופטימיזציה, ואז נוסיף את זה לאילוצים, וככה נעשה על כל בעיה בנפרד.ע"י שינוי הסדר שבה ניקח את פונקציות המטרה ונהפוך אותם לאילוצים נקבל פתרונות אופטימליים שונים לבעייה.  
הבעיה החדשה שיכולה להיווצר היא שכנראה יהיו בעיות שלא יוכלו לספק את כל האילוצים כולל האילוצים החדשים החדשים. בעצם מנסים להגיע לאופטימיזציות של הבעיות בוקטור במקביל.

בס"ד

**שיטת Lexicographic method**  
  
בשיטה הלקסיקוגרפית, נפתור את בעיית האופטימיזציה ע"י כך שנתן חשיבות לכל פונקציית מטרה ואז כאשר נפתור את הבעיה, נעבור בעיה אחר בעיה לפי חשיבות כאשר כל פעם הוספנו את הפתרון של כל הבעיות שהגיעו לפני הבעיה הנוכחית וככה נמשיך עד שנגמור לעבור על כל פונקציות המטרה.אם לאחר פונקציה מסויימת מתקבלת תשובה יחידה ולא ניתן לספק יותר פונקציות נעצור שם ונתן את הפתרון הזה. ככה נגיע לפתרון כלשהו, אבל האילוצים הפחות חשובים יקבלו פתרון פחות טוב בשבילם.  
  
בעצם בעית ה-MOO היא בסופו של דבר מגיעה לוקטור הפתרונות האופטימליים לבעיות, שיכול לשלב בעיות min/max.

**פתרון בעיית ה-MOO מחולק לשתי שיטות עיקריות(השיטות המוצעות לעיל נכנסות תחתיהן):**  
  
**Pareto method**

באופן מתמטי נגדיר את הקלט לבעייה באופן הבא:

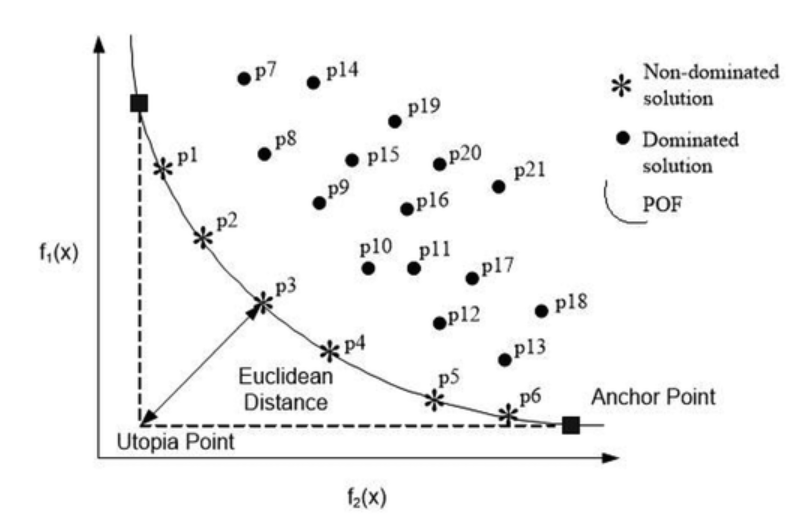
  
השיטה מפצלת את הבעיות באופן בלתי תלוי ומגיעה לפתרונות אופטימליים עבור כל הבעיות.על מנת להגיע לפתרון הדומיננטי(non-dominated solution) נרצה להגיע לפתרון שבו אי אפשר יותר לשפר את הפתרון האופטימלי של הבעיה הנ''ל, מבלי לגרוע מפתרון אופטימלי אחר כלשהו, תנאי הזה נקרא pareto optimal.קבוצת הפתרונות האופטימלי נקרא Pareto optimal solution.

פתרונות שבהם אפשר להמשיך לשפר את הפתרון האופטימלי מבלי לגרוע מפתרון אופטימלי אחר נקראות non-pareto optimal לפתרון כזה גם קוראים dominated solution כלומר קיים פיתרון ש"שולט" עליו.ניתן לפתור את הבעיה באופן מתמטי אם מצאנו כבר פתרון שהוא dominate.

בס"ד

כעת נגדיר שתי מושגים שיעזרו לנו בהמשך:  
**Anchor point** - הנקודה האופטימלית של הפונקציה, נקודת עוגן.  
**Utopia point** - נקודת החיתוך של מינימום\מקסימום של פונקציה אחת ומינימום\מקסימום של פונקציה אחרת. בעצם נקודה שלא נמצאת על הפונקציה והיא נקודת החיתוך של הפתרונות של שתי פונקציות (או יותר) ביחד. חיתוך של הפונקציות היוצאות מנקודות העוגן. ולכן היא מאוד חשובה.

הגרף הבא יעזור להסביר את סוגי הפתרונות השונים ואת המושגים שהגדרנו לעיל:



הכוכביות מייצגות את סט הפתרונות הדומיננטיות(Pareto optimal solution)לעומת הנקודות שהם dominated.הריבועים מייצגים את נקודות הקיצון שלנו שהם הנקודות העוגן. נקודת האוטופיה היא בעצם נקודת חיתוך בים הישרים העוברים דרך נקודת העוגן.  
  
בעצם נבין כי פתרון שהוא non dominated יותר טוב מפתרון dominated אם מתקיימים שני התנאים הבאים:

1. פתרון שהוא non-dominated לא גרוע לעומת הפתרון הdominated

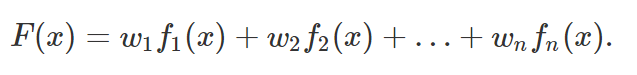
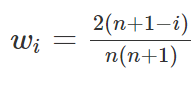
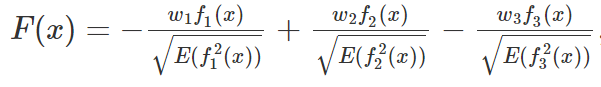
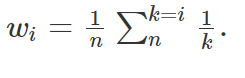
2.שה-non-dominated יהיה טוב בלפחות פונקציית מטרה אחת לעומת הdominated.  
ברגע שמוצאים את נקודת האוטופיה, אז הפתרון האופטימלי מחושב לפי המרחק הקצר ביותר מנקודת האוטופיה לנקודת פתרון.   
בשביל למצוא את ה-non dominated פתרונות, מתבצע כלל עדכון, ואחריו מוצאים את ה-Utopia point ומחשבים את המרחק הקצר ביותר.  
 

בס"ד

**Scalarization method**

לוקח את כל הפונקציות מטרה ומחבר אותן לפונקציית מטרה אחת גדולה, כאשר לכל אחת מהפונקציות מטרה הקטנות יש משקל לפי החשיבות שלה, נתינת משקל גבוה מעיד על חשיבות לעומת זאת נתינת משקל נמוך מעיד חשיבות נמוכה.

פונקציית המטרה החדשה נוצרת באופן הבא:

  
יש שלוש שיטות לקביעת המשקלים:  
1. משקלים שווים, כל פונקציית מטרה מקבלת משקל שווה ביחס לכמות פונקציות.  
2.rank order centroid כל פונקציה מקבלת משקל לפי הפונקציה הבאה:  
Rank sum .3 כל פונקציה מקבלת משקל לפי הפונקציה הבאה:   
  
קביעה של האם מדובר בפונקציית מינימום או מקסימום תיוצג על ידי +\- במשוואה.כפי שמוצג באופן הבא:

בשביל לייצר הוגנות בין כל פונקציות המטרה נבצע גם נורמליזציה עליהן.

בס"ד

הגדרת הבעיה שנדגים עליה MOO:

נגדיר בעיה פשוטה שנשתמש בה על מנת להשוות בין שיטות שונות לפתירת בעיות בעלות פונקציות מטרה רבות:

ישנה חברה בשם CHOCHO-MOO המייצרת מוצר שוקו מאוד פופולרי בקרב ילדים בגילאי 3-10.

לשוקו יש 2 מתכונים מיוחדים של שוקו בננה ושוקו תות.

שוקו בננה מוגדר בתור x1 ושוקו תות בתור x2.דני מאוד אוהב שוקו מחברת CHOCO-MOO. כל שוקו בננה שדני שותה מוסיפים לו 2 יחידות של הנאה וכל שוקו תות שהוא שותה מוסיפים 3 יחידות של הנאה.

מכיוון שהשוקו כ"כ טעים ודני שותה הרבה אמא שלו רוצה שיהיה לו כמה שיותר בריא. כל שוקו בננה מוסיף 4 יחידות בריאות וכל שוקו תות מוריד 2 יחידות בריאות.

אמא של דני שמאוד אוהבת לקנות מהחברה CHOCO-MOO הציבה לעצמה שתי מטרות עיקריות למקסם את רמת הבריאות וההנאה של דני. כמובן בהתאם לאילוצים של משרד הבריאות על הכמויות המומלצות לתת לילדים.

פונקציות מטרה:

אילוצים:

בס"ד

בקוד בלינק של הגיט הנמצא כאן:

מימשנו 2 גישות לפתירת בעיות MOO הראשונה לפי שיטת הpareto שם מימשנו את הsmall-epsilon method שבו פותרים לפי פונקצייה מסויימת ואז מוסיפים לבעיה אילוץ לפי הפיתרון שקיבלנו ועוברים לפונקציית המטרה הבאה.

איך מריצים?

בוחרים בעיה כלשהי(הקבצים שם הם לפי הבעיה שהגדרנו לעיל)

נותנים לתכנית קובץ עם כל פונקציות המטרה והאילוצים באופן הבא:

2 -מספר פונקציות המטרה

4,-2;2,3 - פונקציות המטרה מופרדות ע"י נקודה פסיק(;)

1,1,10;2,1,15 - האילוצים מופרדות ע"י נקודה פסיק(;) כך שהאחרון הוא ההגבלה על האילוץ

באופן שמימשנו הנחנו שהאילוצים הם קטנים שווים ניתן להמיר לפני בעת הצורך

שורת הפקודה:

python small\_epsilon\_method.py problem\_details\_file.txt

השיטה השניה היא לפי שיטת הscalarization:

אופן קבלת הארגומנטים הוא זהה.

בשיטה זו אנו לוקחים את כל פונקציות המטרה ונותנים לכל אחת משקל מחברים אותם כפי שתואר לעיל ואז פותרים לפי פונקציית המטרה החדשה הגדולה שקיבלנו.

ישנם שיטות שונות לנתינת משקלים לפונקציות אנו מראים 2 שיטות שיטה אחת של נתינת משקלים זהים ושיטה אחרת שבה