אחזור מידע – תרגיל בית 3 – Optimizers Outlaws

שאלה 1: *הרחבות שאילתות ואחזור ממוין*

פתרון:

בואו ניקח בחשבון מקרה חד-ממדי עם שתי מחלקות, ו- (כאשר הוא קבוצה של מסמכים), שבו מסווג Rocchio מקצה תווית שגויה לדגימת אימון.

נניח ש-C1 נמצא במיקום x = 2 ו- נמצא בעמדה x = 6.

מדגם אימון נמצא במיקום x = 4 ומסומן כשייך ל-.

כלומר, (x = 4) הוא חלק ממערך האימון עבור (), ולכן התווית האמיתית היא ().

מסווג Rocchio מקצה דגימה למחלקה שהמרכז שלה קרוב יותר במרחק.

תהליך הסיווג:

- מרחק מ-(x = 4) ל-(): (|4 - 2| = 2).

- מרחק מ-(x = 4) ל-(): (|6 - 4| = 2).

במקרה זה, המרחקים שווים, והמסווג של Rocchio עלול להקצות באופן שגוי את המדגם (x = 4) ל-() במקום (), בהתבסס על כללי קבלת החלטות שרירותיים.

אם (x = 4) מסווג כ-() (בשל מרחקים שווים או עיגול), המסווג סימן באופן שגוי מדגם אימון. אז אנחנו יכולים להסיק שמסווג Rocchio יכול לסווג שגוי של דגימות אימון, במיוחד במקרים של צנטרואידים חופפים או במרחק שווה, מה שמדגיש את הרגישות שלו למיקום מרכז.

שאלה 2: *Link Analysis*

פתרון:

1. A diagram of a diagram

   Description automatically generated
2. נחשב לפי HITS:  
   שלב ראשון:  
    נחשב *Adjacency matrix*

מטריצת הסמיכות עבור הגרף היא:

*כאשר תא מסמנת שיש קשת בין ל- .*

*שלב שני:*

*מאתחלים את הערכים של Authority ו- HUB של כל node ל- 1.*

*שלב שלישי - שלב האיטרציות:*

*לפי הרצאות הנוסחאות של האיטרציות של Authority ו- HUB הן:*

A black symbol with a arrow pointing to the same letter

Description automatically generated with medium confidenceA black symbols with a white background

Description automatically generated with medium confidence

*נכתוב את הנוסחאות בצורת מטריצה:*

*תוצאות האיטרציות יהיו:*

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

*אחרי איטרציה מספר 5, ראינו שהערכים כבר מתכנסים:*

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

*שאלה 3:*

1. ***רשמו טכנולוגיות שונות מעניינות שהשתמשתם בהם בפרויקט.***

*בפרויקט שלנו השתמשנו בשלוש טכנולוגיות מרכזיות אשר תרמו משמעותית לשיפור איכות ומהירות החיפוש:*

1. ***Multithreading Search****:  
   טכנולוגיה זו מאפשרת ביצוע חיפושים מרובים במקביל באמצעות חלוקת המשימות בין מספר תהליכונים (threads). כך, מתקצר זמן העיבוד הכולל ומשתפרת היעילות של מנוע החיפוש, במיוחד עבור שאילתות מורכבות או מסדי נתונים גדולים.*
2. ***Phrase Handling Search****:  
   יישמנו טכניקה שמאפשרת זיהוי והתמודדות עם ביטויים מורכבים במהלך החיפוש. הדבר מאפשר למנוע החיפוש להחזיר תוצאות מדויקות יותר כאשר המשתמשים מחפשים ביטויים מרובי מילים או מונחים מחוברים.*
3. ***Synonyms Recognition****:  
   שילבנו מערכת לזיהוי מילים נרדפות (סינונימים) כדי להרחיב את טווח התוצאות הרלוונטיות. טכנולוגיה זו מסייעת למנוע החיפוש להבין כוונות משתמשים גם כאשר הם מבטאים את חיפושיהם בדרכים שונות או משתמשים במילים דומות עם משמעות דומה.*

*השילוב של שלוש טכנולוגיות אלה שיפר משמעותית את ביצועי מנוע החיפוש שלנו הן מבחינת מהירות והן מבחינת דיוק.*

1. ***כמה זמן רצו השאילתות שלכם?במה זה תלוי? האם לדעתכם ניתן לשפר זמן זה?***

*ברוב המקרים, זמן הרצת השאילתות עמד על טווח של כמה מילישניות עד כמה מאות מילישניות, תלוי במורכבות השאילתה, גודל מאגר הנתונים ומספר התהליכונים (threads) שהוקצו.*

***שיפור זמני החיפוש אפשרי*** *בכמה דרכים עיקריות :*

1. ***שימוש במטמון (Caching)****: שמירת תוצאות של שאילתות נפוצות.*
2. ***אופטימיזציה של אלגוריתם החיפוש****: שימוש במבני נתונים מתקדמים.*
3. ***ניהול Multithreading יעיל****: ניצול מיטבי של תהליכונים.*

***3. האם בדפים שהוחזרו קיימים hubs? Authorities? נמקו.***

*בדפים שלנו יש Related Pages ו-Similar Pages, ולכן ניתן להסיק שהם כוללים Hubs ו-Authorities:*

* *Hubs: הדפים כוללים קישורים ל-Related Pages, אשר מפנים למקורות נוספים ורלוונטיים לנושא. בכך הם מתפקדים כמרכזים שמחברים למידע נוסף.*
* *Authorities: הדפים כוללים גם Similar Pages, המשקפים תוכן סמכותי שאוסף הפניות מדפים אחרים ומעיד על מהימנותם בנושאים ספציפיים.*

*השילוב של שני האלמנטים הללו מעיד על קיומם של Hubs ו-Authorities בדפים שלנו.*

1. ***5 הדפים שהזחלן החזיר וחישוב ה- pagerank עבורם:***

|  |  |
| --- | --- |
|  |  |
|  | <https://store.steampowered.com/> |
|  | <https://store.steampowered.com/app/2507950/Delta_Force/> |
|  | <https://store.steampowered.com/app/1149460/ICARUS/> |
|  | <https://store.steampowered.com/app/730/CounterStrike_2/> |
|  | <https://store.steampowered.com/app/271590/Grand_Theft_Auto_V/> |

*A computer screen shot of a diagram

Description automatically generated*

*החישוב נעשה על ידי הפונקציות:*

def calculate\_new\_pagerank(current\_ranks, links):

    new\_ranks = {}

    # Calculate new PageRank for each page

    for page in current\_ranks:

        # Find who links to this page

        incoming\_links = [p for p, outgoing in links.items() if page in outgoing]

        # Sum up PageRank contributions

        rank\_sum = 0

        for source\_page in incoming\_links:

            # Get number of outgoing links from source page

            num\_outgoing = len(links[source\_page])

            # Add contribution from this source page

            rank\_sum += current\_ranks[source\_page] / num\_outgoing

        new\_ranks[page] = rank\_sum

    return new\_ranks

def print\_ranks(ranks):

    print("-" \* 35)

    print("Page  |  PageRank Value")

    print("-" \* 35)

    for page, rank in ranks.items():

        print(f"  {page}   |     {rank:.3f}")

    print("-" \* 35)

def page\_rank():

    # Define the web structure

    links = {

        'https://store.steampowered.com/': ['https://store.steampowered.com/app/2507950/Delta\_Force/', 'https://store.steampowered.com/app/1149460/ICARUS/', 'https://store.steampowered.com/app/730/CounterStrike\_2/', 'https://store.steampowered.com/app/271590/Grand\_Theft\_Auto\_V/'],

        'https://store.steampowered.com/app/2507950/Delta\_Force/': ['https://store.steampowered.com/', 'https://store.steampowered.com/app/730/CounterStrike\_2/'],

        'https://store.steampowered.com/app/1149460/ICARUS/': ['https://store.steampowered.com/'],

        'https://store.steampowered.com/app/730/CounterStrike\_2/': ['https://store.steampowered.com/','https://store.steampowered.com/app/2507950/Delta\_Force/'],

        'https://store.steampowered.com/app/271590/Grand\_Theft\_Auto\_V/': ['https://store.steampowered.com/', 'https://store.steampowered.com/app/2507950/Delta\_Force/']

    }

    # Initialize PageRank values (1/5 for each page)

    pages = ['https://store.steampowered.com/', 'https://store.steampowered.com/app/2507950/Delta\_Force/', 'https://store.steampowered.com/app/1149460/ICARUS/', 'https://store.steampowered.com/app/730/CounterStrike\_2/', 'https://store.steampowered.com/app/271590/Grand\_Theft\_Auto\_V/']

    current\_ranks = {page: 1/len(pages) for page in pages}

    print("Web structure:")

    for page, outlinks in links.items():

        print(f"Page {page} links to: {', '.join(outlinks)}")

    # Print initial values

    print("\nInitial PageRank Values:")

    print\_ranks(current\_ranks)

    for iteration in range(10):

        new\_ranks = calculate\_new\_pagerank(current\_ranks, links)

        current\_ranks = new\_ranks

    # print values after iterations

    print("\PageRank Values:")

    print\_ranks(current\_ranks)

    # Find highest PageRank after second iteration

    highest\_page = max(current\_ranks.items(), key=lambda x: x[1])

    print(f"\nHighest PageRank after second iteration:")

    print(f"Page {highest\_page[0]} with PageRank value of {highest\_page[1]:.3f}")

    return current\_ranks

1. Relevance Feedback ניתן על ידי שני סטודנטים אשר סיווגו את התוצאות הרלוונטיות של הזחלן בצורה הבאה:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 5 | 4 | 3 | 2 | 1 |  |
| R | - | R | R | R | Student 1 |
| - | R | R | - | R | Student 2 |

*הערה: באישור מנועמי התאפשר לנו לעשות את חישוב ה-pagerank בשיטת hard coded כי היה לנו בעיה לחשב את ה-pagerank ב- steam כי הזחלן לא היה מוציא את הקישורים לדפים אחרים כמו הצורך.*

קישור לתקיה של גיט: [GitHub Repository For HW3](https://github.com/oRABiiA/Data-Retrieval)