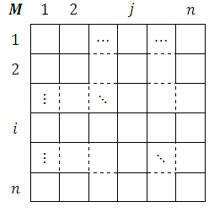
**נושאים תרגול 8**

**BFS breadth-first search - חיפוש לרוחב**

1. הכרת האלגוריתם + פסאודוקוד
2. דוגמא
3. האם הגרף קשיר?
4. מציאת מסלול קצר ביותר בין 2 קודקודים
5. חישוב מספר רכיבי הקשירות
6. מציאת הקודקודים בכל רכיב קשירות
7. חישוב קוטר הגרף (הרצת פעמיים BFS)

**ייצוג/מימוש גרפים בתכנות**

על מנת שנוכל לנתח את סיבוכיות הזמן והמקום של אלגוריתמים בתורת הגרפים, נראה כיצד אנו מייצגים גרפים בזיכרון מחשב.

1. **שימוש באובייקטים, יצירת מחלקות: Graph, Vertex, Edge**
2. **מטריצת שכנויות/סמיכויות  :(adjacency matrix)**

כאשר נרצה לייצג גרפים פשוטים (מכוונים או לא מכוונים) במחשב, נוכל לייצגם בעזרת מטריצת שכנויות בגודל |V|x|V|. במקום ה- (*i*, *j*) במטריצה יופיע 1 אם יש קשת בין הצמתים ה- *i*וה-*j*. אחרת יהיה במקום זה 0. כאשר נייצג גרף לא מכוון, המטריצה תהיה מטריצה סימטרית.

**גרף דליל** הוא גרף שאין בו מספר רב של קשתות. גרף דליל מוגדר כגרף שבו מספר הקשתות קטן ממספר הקודקודים בריבוע.

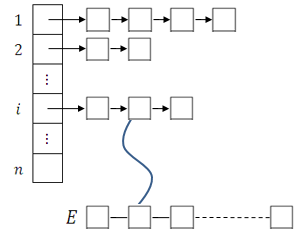
חסרונות:

* ייצוג לא יעיל (בזבזני במקום ובזמן) עבור גרפים דלילים ().
* פעולות בסיסיות כמו מעבר על כל השכנים של צומת לוקח זמן רב (Θ()).

יתרונות:

* ייצוג פשוט מאוד.
* תכונות אלגבריות של מטריצה מאפשרות להציג מידע רב על גרף (כגון לאתר מידע על מספר מסלולים מקודקוד אחד לשני באורך כלשהו).
* מחיקת קשת - Ο(). מציאת קשת - Ο().

1. **רשימת סמיכויות (שכנויות):**

רשימת הסמיכויות מורכבת ממערך צמתי הגרף.

* תא המתאים לצומת v במערך מכיל את רשימת כל הקשתות (שכנים) אשר מחוברות לצומת v.
* מבנה הנתונים המקובל לצורך זה הוא מערך של רשימות מקושרות (linked list).

בנוסף, המבנה מכיל רשימת קשתות.

* לכל קשת e∈E נשמור את שני צמתי הקצה שלה.

יתרונות:

* סיבוכיות הזכרון (המקום) היא (Θ(+|E|)) בניגוד ל עבור מטריצת שכנויות.
* פעולות רבות על הגרף ניתנות לביצוע יעיל יותר, למשל מעבר על השכנים לוקח Θ(

נא לראות קובץ בשם BFS algorithm.docx