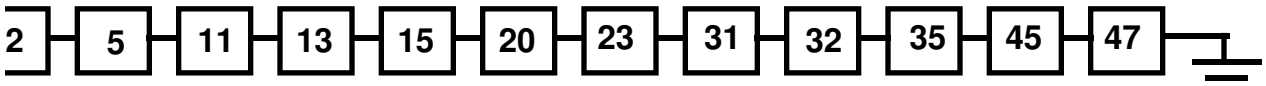
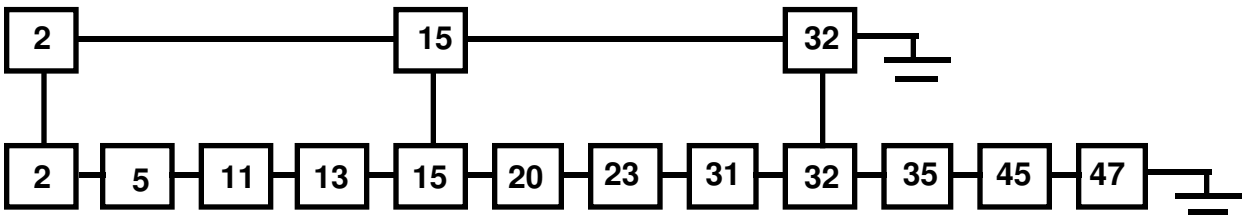
© Itai November 10, 1996

# Skip List - רשימת דילוגים

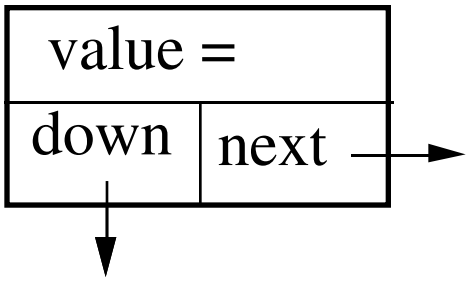
**<u>מטרה</u>: לאפשר חיפוש, הכנסה והוצאה יעילים**
**דוגמא: חיפוש איבר בסוף רשימה ארוכה הוא יקר**




**כדי לזרז את החיפוש נבנה מדריך:**
**תת-רשימה, של כל איבר d (בדוגמא d=4):**




**מבנה צומת**



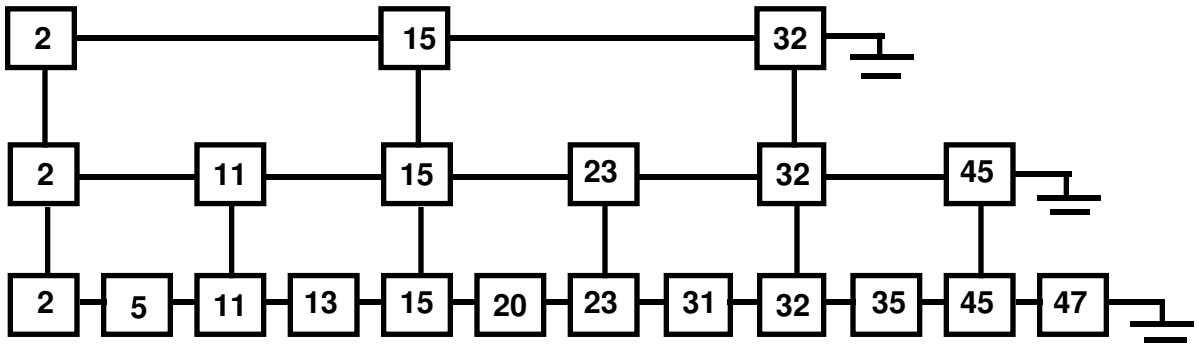

**החיפוש יתחיל ברמה העליונה, וימשיך בתחתונה:**

```
for(p=upper;
     (p->next≠NULL)  &&
           (p->next->value < x);
     p=p->next)
    ;
for(p=p->down;
     (p->next≠NULL)  &&
           (p->next->value < x);
     p=p->next)
    ;
```

**איך נדע אם x נמצא במבנה?**

**עבור רשימה באורך n מהו הערך האופטימלי של d?**

**אפשר יותר משתי רמות:**

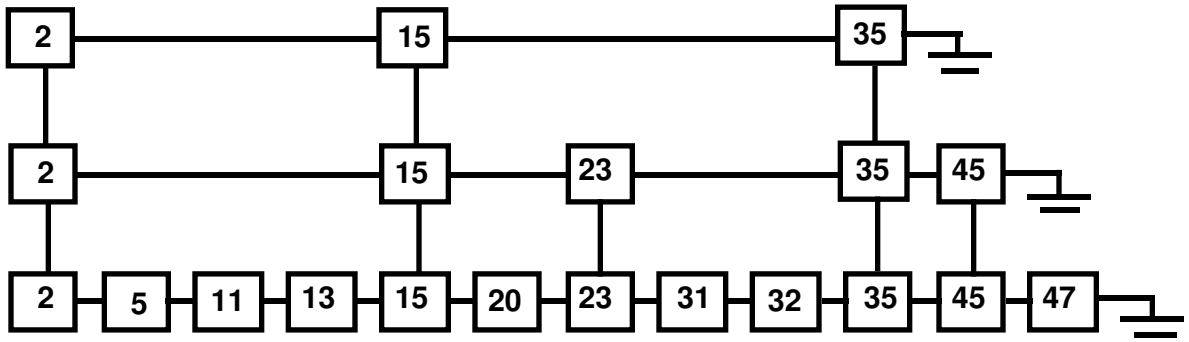



**עבור רווח d, נבנה $\log_d n$ רמות. (בדוגמא d=2).**

**<u>אלגוריתם החיפוש</u> (נניח שהרשימה אינה ריקה)**

```
NODE *search(KEY x, NODE *p){
    for(;;){
      for(;
            (p->next ≠ NULL)
            && (p->next->value<x);
          p=p->next)
          ;
      if(p->down == NULL)
        return p;
      p=p->down;
    }
}
```

**הערה:**
**אלגוריתם החיפוש נכון גם אם מספר הצמתים של רמה i בין כל שני צמתים ברמה i-1 אינו קבוע.**




**(הסיבוכיות עלולה להפגע).**

## הכנסה

**נדרוש שלכל רשימה תהיה כותרת:**
**צומת בעל ערך $-\infty$**

**נתחיל את ההכנסה ברמה הנמוכה ביותר, h.**
**בהסתברות $\frac{1}{2}$ נוסיף צומת גם ברמה מעליה, h-1.**
**וכך-הלאה, ...**
**אם הוספנו צומת לרמה העליונה,**
**בהסתברות $\frac{1}{2}$ נוסיף עוד רמה.**

**הפרוצדורה insert1 מחזירה את הצומת שהוכנס ברמה העליונה ביותר,**
**או NULL אם לא הוכנס צומת ברמה זו.**

**הפרוצדורה היא רקורסיבית,**
**עומק הרקורסיה = גובה המבנה.**

**ברמה התחתונה ביותר:**
**נוסיף צומת חדש בעל ערך x, ונחזיר מחוון אליו.**

**ברמה i:**
**נבצע חיפוש ברמה i, עד לצומת p שלפני הערך x.**
**נקרא רקורסיבית מ-p, ונקבל מחוון deeper.**
**אם deeper == NULL**
**(לא הוספנו צומת ברמה i+1)**
**אל תוסיף גם ברמה i והחזר NULL.**
**אחרת: בהסתברות 1/2:**
**הוסף צומת new_node (בעל ערך x)**
**והחזר את new_node.**
**בהסתברות 1/2:**
**אל תוסיף צומת והחזר NULL.**

**<u>פרוצדורות עזר:</u>**
**add_after(x,p): מוסיפה צומת חדש**
**בעל ערך x אחרי p, ומחזירה מחוון אליו.**
**toss(): מחזירה 0 או 1 בהסתברות שווה.**

```
NODE *insert1(KEY x, NODE *p){
    NODE *deeper, *new_node;

    for(;
        p->next≠NULL &&
                p->next->value<x;
        p=p->next)
      ;

    if(p->down == NULL)
      return add_after(x,p);

    deeper = insert1(x,p->down);
    if((deeper == NULL)||toss())
      return NULL;
    new_node = add_after(x,p);
    new_node->down = deeper;
    return new_node;

}
```

שמורה

```
v->down ≠ NULL
⇒
v->value == v->down->value
```

**:BUG**

**הפרוצדורה insert1 לא מוסיפה רמות.**
**כדי לתקן זאת, בהסתברות 1/2 נוסיף רמה אם**
**הפרוצדורה insert1 מחזירה ערך ≠ NULL.**

**הפרוצדורה insert מחזירה מחוון לרשימת**
**הדילוגים שהתקבלה כתוצאה מההכנסה.**

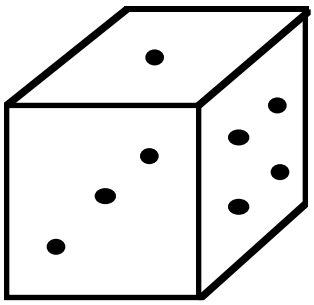
```
NODE *insert(KEY x,
             NODE *old_list) {
    NODE *deeper;

    deeper = insert1(x,old_list);
    if(deeper == NULL)||toss())
      return old_list;
    return new_node(-INFINITY,
                    NULL, old_list);
}
```

**הפרוצדורה new_node(x,n,d) מחזירה מחוון**
**לצומת חדש v כך ש-**
**v->value=x; v->next=n; v->down=d**

**<u>שאלה</u>: איך נראית הרמה העליונה?**
**<u>הערה</u>: צריך לבדוק ש-malloc לא נכשל.**

**בנינו אלגוריתם שזמן הריצה שלו תלוי בתוצאות של הגרלות שעורך המחשב.**
**אלגוריתם שעושה הגרלות נקרא**
**<u>אלגוריתם רנדומאלי</u>.**

# אנליזה הסתברותית של רשימות דילוגים



## מבוא להסתברות

**כשמטילים קוביה נקבל מספר בין 1 ל-6.**
**מהי התוחלת (הממוצע)?**

**ההסתברות שהתוצאה היא $i$ :** $p_i = \frac{1}{6}$

**התוחלת:**

$$E = p_1 \cdot 1 + p_2 \cdot 2 + \ldots p_6 \cdot 6$$
$$= \frac{1}{6} \cdot (1 + 2 + \ldots + 6) = \frac{1}{6} \cdot 21 = 3\frac{1}{2}$$

**אם ההסתברויות אינן שוות,**
**למשל** $p_1 = p_2 = \frac{1}{4} \quad p_3 = p_4 = p_5 = p_6 = \frac{1}{8}$

$$E = p_1 \cdot 1 + p_2 \cdot 2 + \ldots + p_6 \cdot 6$$
$$= \frac{1}{4}(1+2) + \frac{1}{8}(3 + \ldots + 6) = \frac{3}{4} + \frac{18}{8} = 3.$$

**עבור קוביה עם אינסוף פאות:**

$$E = \sum_{i=0}^{\infty} p_i \cdot i = \sum_{i=1}^{\infty} p_i \cdot i$$

## בעיה

**נניח שנטיל מטבע עד שיצא "ראש"**
**מה המספר הצפוי של הטלות?**

## פתרון

$$p_0 = 0$$
$$p_1 = 1/2$$
$$p_i = 1/2^i$$

**התוחלת**

$$E = \sum_{i=0}^{\infty} p_i \cdot i = \sum_{i=1}^{\infty} (1/2)^i \cdot i = \frac{1}{2} \sum_{i=1}^{\infty} (1/2)^{i-1} \cdot i$$

**נשתמש בשוויון**

$$\sum_{i>0} i \cdot x^{i-1} = \frac{1}{(1-x)^2} \qquad : |x| < 1$$

**כדי לקבל**

$$E = \frac{1}{2}\left(\frac{1}{1-(1/2)}\right)^2 = 2$$

## גודלה הצפוי של רשימת דילוגים בת $n$ איברים

**ההסתברות שאיבר מיוצג ע"י $i$ צמתים:**

$$p_i = 1/2^i$$

**לכן, המספר הצפוי של צמתים שנוספים בגין הוספת איבר לרשימת הדילוגים הוא**

$$E(\text{צמתים}) = E(\text{הטלות}) = 2$$

**יהי $s_i$ מספר הצמתים שנוספו בגין האיבר ה-$i$**

**גודל המבנה**

$$SIZE_n = s_1 + s_2 + \ldots + s_n$$

**התוחלת**

$$E(SIZE_n) = E(s_1) + E(s_2) + \ldots + E(s_n) = 2n$$

**<u>משפט</u>: ההסתברות שהגובה של רשימת דילוגים בת $n$ איברים $h \geq c\log n$**

$$P(h \geq c\log n) \leq \frac{1}{n^{c-1}}$$

**<u>מסקנה</u>: תוחלת מספר הרמות ברשימת דילוגים בת $n$ מפתחות הוא $O(\log_2 n)$.**

**<u>משפט</u>: תוחלת מספר הצמתים ברמה i+1 הנמצאים בין שני צמתים סמוכים ברמה i הוא 2.**

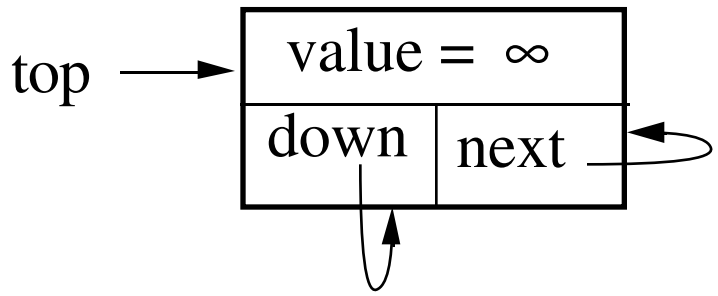
**<u>מסקנה</u>: הזמן הצפוי של חיפוש הוא $O(\log_2 n)$.**

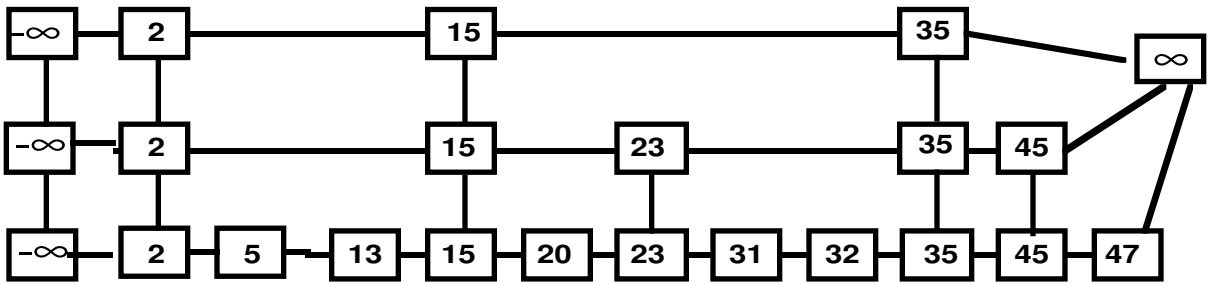
## הוצאות

**כדי לסלק ערך x מהמבנה,**
**יש להוציא את x מכל הרמות בהן הוא נמצא.**

**כיוון שכל צומת משוכפל בממוצע פעמיים,**
**זמן ההוצאה צפוי = זמן חיפוש + $O(1)$.**

**הערה:**
**כדי להמנע מלבדוק `p->next≠NULL` בחיפושים,**
**מוסיפים צומת `top` בעל ערך ∞ ושדה `next`**
**שמצביע לעצמו.**




**האם זה משנה את הסיבוכיות?**

**כמו-כן, בכל רמה נוסיף צומת עם ערך $-\infty$ .**
**צמתים אלו לא יוצאו אף פעם.**