ערבול

תזכורת: ערבול (hashing)

<u>תזכורת ממבני נתונים:</u>

נתון: זיכרון בגודל M רשומות,

n רשומות בעלות מפתחות בתחום: 0..K−1

$n < M < K$

נבחר פונקצית "צמצום" $h : 0..K-1 \rightarrow 0..M-1$

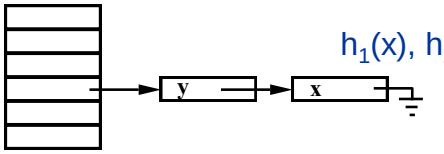
רשומה בעלת מפתח x תושם בזיכרון במקום h(x)

צפיפות (פקטור עומס) $\alpha = \frac{n}{M}$

טיפול בהתנגשויות

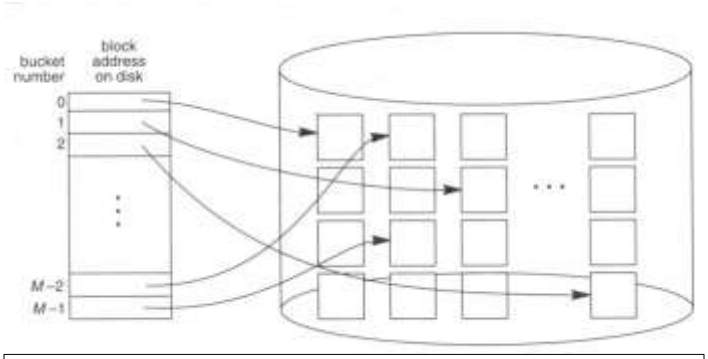
<u>התנגשות:</u> $x \neq y$ אבל $h(x) = h(y)$

- סריקה ליניארית (linear probing):
  שים במקום: $h(x), h(x)+1, \ldots$
- ערבול כפול (double hashing):
  שים במקום: $h(x), h(x) + d(x), h(x) + 2d(x), \ldots$
- ערבול מחדש (rehashing):
  שים במקום: $h_1(x), h_2(x), h_3(x), \ldots$
- שרשראות (chaining)



אינדקס מבוסס על ערבול (hashing)

ערבול: שימוש בפונקציה שממפה מפתחות לכתובות של בלוקים בדיסק

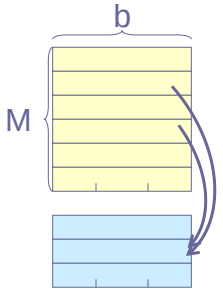


פונקציית ערבול h ממפה מפתחות לדליים וכל דלי מותאם לבלוק או למספר בלוקים בדיסק.

ערבול על הדיסק

טבלת הערבול משמשת כאינדקס, בכל כניסה יש מצביע לדלי

כיון שתמיד קוראים סקטור (לוגי) שלם עם מקום ל-b רשומות,
נשתמש בטבלה עם M דליים (buckets) כל-אחד עם b רשומות



רשומה עם מפתח x תהיה בדלי $i = h(x)$
אם הדלי מלא, נשים את הרשומה **בגרורה** באותו גליל

לרוב, הגרורות כמעט ריקות, ולכן נשתף גרורות בין דליים
אבל רק הגרורה האחרונה של כל דלי תהיה משותפת
שיתוף גרורות לא מגדיל את מספר הגישות בחיפוש.

גישות לדיסק – חישוב גס

- במקרה הטוב גישה לרשומה מצריכה גישת דיסק אחת בלבד
- דלי יכיל לפחות גזרה אחת
- דליים קטנים:
  - אינדקס גדול יותר
  - ייתכן שצריך לאחסן אותו בדיסק
- דליים גדולים:
  - אינדקס קטן יותר
  - סיכוי קטן יותר לגרורה
  - קריאת כל הדלי ארוכה יותר
- רשומה בגרורה דורשת גישת דיסק נוספת
- נשתדל להקצות מקום לגרורות באותו גליל כמו הדלי שלהן

הסתברות גלישה במבנה ערבול

הנחת פיזור אחיד: h ממפה את המפתחות לדליים בהתפלגות אחידה.
כלומר, עבור טבלה עם M דליים, ומפתח x: $P(h(x) = i) = 1/M$

תהיינה: b רשומות בדלי
n רשומות בקובץ
T = גודל הטבלה (ברשומות)

פקטור עומס $\alpha = \frac{n}{T} = \frac{n}{Mb}$

ההסתברות שרשומה אקראית תמופה לדלי i: $p = \frac{1}{M} = \frac{\alpha b}{n}$

המספר הצפוי של רשומות בדלי i: $\lambda = np = \alpha b$

הסתברות למיפוי j רשומות לדלי אחד

$$P(j) = \binom{n}{j} p^j (1-p)^{n-j} = \frac{1}{j!}\frac{n!}{(n-j)!}\left(\frac{p}{1-p}\right)^j (1-p)^n$$

$$\cong \frac{1}{j!}\frac{n!}{(n-j)!}\left(\frac{\alpha b}{n}\right)^j \left(1-\frac{\alpha b}{n}\right)^{-j} e^{-\alpha b}$$

$$(1-p)^n = \left(1-\frac{\alpha b}{n}\right)^n = \left(\left(1-\frac{\alpha b}{n}\right)^{n/\alpha b}\right)^{\alpha b} \cong e^{-\alpha b}$$

$(1-1/x)^x$ היא פונקציה עולה
השואפת ל- e^{-1} (כאשר x גדל)

הסתברות למיפוי j רשומות לדלי אחד

$$P(j)=\binom{n}{j}p^j(1-p)^{n-j}=\frac{1}{j!}\frac{n!}{(n-j)!}\left(\frac{p}{1-p}\right)^j(1-p)^n$$

$$\cong\frac{1}{j!}\frac{n!}{(n-j)!}\left(\frac{\alpha b}{n}\right)^j\left(1-\frac{\alpha b}{n}\right)^{-j}e^{-\alpha b}$$

$$\frac{n!}{(n-j)!}=n(n-1)\ldots(n-j+1)<n^j$$

הסתברות למיפוי j רשומות לדלי אחד

$$P(j)=\binom{n}{j}p^j(1-p)^{n-j}=\frac{1}{j!}\frac{n!}{(n-j)!}\left(\frac{p}{1-p}\right)^j(1-p)^n$$

$$\cong\frac{1}{j!}\frac{n!}{(n-j)!}\left(\frac{\alpha b}{n}\right)^j\left(1-\frac{\alpha b}{n}\right)^{-j}e^{-\alpha b}<\frac{1}{j!}n^j\left(\frac{\alpha b}{n}\right)^j\left(1-\frac{\alpha b}{n}\right)^{-j}e^{-\alpha b}$$

$$\frac{n!}{(n-j)!}=n(n-1)\ldots(n-j+1)<n^j$$

הסתברות למיפוי j רשומות לדלי אחד

$$P(j)=\binom{n}{j}p^j(1-p)^{n-j}=\frac{1}{j!}\frac{n!}{(n-j)!}\left(\frac{p}{1-p}\right)^j(1-p)^n$$

$$\cong\frac{1}{j!}\frac{n!}{(n-j)!}\left(\frac{\alpha b}{n}\right)^j\left(1-\frac{\alpha b}{n}\right)^{-j}e^{-\alpha b}<\frac{1}{j!}n^j\left(\frac{\alpha b}{n}\right)^j\left(1-\frac{\alpha b}{n}\right)^{-j}e^{-\alpha b}$$

$$\cong\frac{1}{j!}n^j\left(\frac{\alpha b}{n}\right)^j e^{-\alpha b}\cong\frac{(\alpha b)^j}{j!}e^{-\alpha b}$$

עבור $n>>b$ (ולכן $n>>\alpha b$)

$$\left(1-\frac{\alpha b}{n}\right)^{-j}\cong 1$$

הסתברות למיפוי j רשומות לדלי אחד

$$P(j)=\binom{n}{j}p^j(1-p)^{n-j}=\frac{1}{j!}\frac{n!}{(n-j)!}\left(\frac{p}{1-p}\right)^j(1-p)^n$$

$$\cong\frac{1}{j!}\frac{n!}{(n-j)!}\left(\frac{\alpha b}{n}\right)^j\left(1-\frac{\alpha b}{n}\right)^{-j}e^{-\alpha b}<\frac{1}{j!}n^j\left(\frac{\alpha b}{n}\right)^j\left(1-\frac{\alpha b}{n}\right)^{-j}e^{-\alpha b}$$

$$\cong\frac{1}{j!}n^j\left(\frac{\alpha b}{n}\right)^j e^{-\alpha b}\cong\frac{(\alpha b)^j}{j!}e^{-\alpha b}$$

קיבלנו התפלגות פואסון עם פרמטר αb

הסתברות למיפוי j רשומות לדלי אחד

אם בניסוי המספר הצפוי של הצלחות הוא λ, ההסתברות שיהיו בדיוק k הצלחות היא **התפלגות פואסון** עם פרמטר λ

$$P(k;\lambda)=\frac{\lambda^k e^{-\lambda}}{k!}$$

במקרה שלנו, $\lambda=\alpha b$

ולכן ההסתברות ש-j רשומות ימופו לדלי מסוים היא

$$P(j;\lambda)\cong\frac{(\alpha b)^j e^{-\alpha b}}{j!}$$

ההסתברות לגלישה

יש גלישה כאשר $b<j$, ולכן נשתמש בנוסחה הקודמת:

$$P(\text{גלישה})=\sum_{j=b+1}^{n}P(j)<\sum_{j=b+1}^{\infty}\frac{(\alpha b)^j}{j!}e^{-\alpha b}=e^{-\alpha b}\sum_{j=b+1}^{\infty}\frac{(\alpha b)^j}{j!}$$

$$<e^{-\alpha b}\frac{(\alpha b)^b}{b!}\sum_{j=b+1}^{\infty}\alpha^{j-b}=e^{-\alpha b}\frac{(\alpha b)^b}{b!}\frac{\alpha}{1-\alpha}$$

$$\frac{b^j}{j!}=\frac{b\times b\times\cdots\times b\times\cdots\times b}{1\times 2\times\cdots\times b\times\cdots\times j}<\frac{b\times b\times\cdots\times b}{1\times 2\times\cdots\times b}=\frac{b^b}{b!}$$

ההסתברות לגלישה

עם דלי בגודל b=10

הסתברות לשתי גרורות	הסתברות לגרורה בודדת		α
	ניתוח אחר	נוסחה	
1.5×10^{-5}	0.20	0.20	0.7
	0.30	0.31	0.75
	0.43	0.48	0.8
4.4×10^{-4}	0.77	1.29	0.9

איך ייתכן שההסתברות גדולה מ-1?

$$P(\text{גרורות } 2) < e^{-\alpha b} \sum_{j=2b+1}^{\infty} \frac{(\alpha b)^j}{j!}$$

מסקנות:

1. אפשר להניח שלכל דלי יש לכל היותר גרורה אחת
2. ההסתברות לגלישה < 30% (גם עם דליים קטנים).

זמן חיפוש כושל

יהי t זמן קריאת דלי (= סקטור) t = seek + rotation-delay + transfer

אם המפתח לא נמצא במבנה, חייבים לקרוא גרורות. תוספת הזמן היא:

△ = rotation-delay + transfer

זמן חיפוש למפתח שאינו במבנה (עם הסתברות זניחה ליותר מגרורה אחת)

T_{unsucc} = P(אין גלישה) t + P(יש גלישה) (t + △) + … (זניח)

≈ t + P(יש גלישה) (rotation-delay + transfer)

עם 450 סקטורים למסילה ו- P(גלישה) = 30%

$t = seek + \frac{1}{2} rotation + \frac{1}{450} rotation = 3.4ms + 2.008ms = 5.408ms$

$T_{unsucc} = 5.408 \text{ ms} + 0.30 \times 2.008 \text{ ms} \approx 6.01 \text{ ms}$

זמן חיפוש מוצלח

אם המפתח נמצא במבנה, נקרא גרורה רק אם הרשומה לא בדלי העיקרי

זמן חיפוש ממוצע של מפתח שנמצא במבנה:

T_{succ} = P(נמצא בעיקרי)·t + P(נמצא בגרורה)·(t + △)

= t + P(נמצא בגרורה) · △

= t + P(יש גלישה | נמצא בגרורה) · P(יש גלישה) · △

≤ t + P(יש גלישה) · 0.5 · △

= t + P(יש גלישה) · 0.5 ·(rotation-delay + transfer)

אם יש גלישה, המפתח נמצא בגרורה בהסתברות חצי

ובהצבת ערכים מספריים כמו קודם:

$T_{succ} \le$ 5.408 ms + 0.30 · 0.5 · 2.008 ms ≈ 5.71 ms

הכנסה והוצאה של רשומה x

1. חפש "כאילו" x לא נמצא
2. עדכן חוצץ
3. כתוב

$T_{insert} = T_{unsuccessful\text{-}find}$ + rotation = 6.01ms + 4ms = 10.01ms

- מה קורה אם הרשומה החדשה גרמה לגלישה?
- מה ההסתברות לכך?

הוצאת רשומה ב**שיטת המצבה** (רק מסמנים שהרשומה לא רלוונטית):

$T_{delete} = T_{successful\text{-}find}$ + rotation = 5.71ms + 4ms = 9.71ms

מעבר סדרתי על כל הקובץ

רוצים לקרוא את כל הרשומות בסדר עולה

במבנה ערבול אין הצבעה לרשומה הבאה, אבל גם אם לכל רשומה היה שדה עם כתובת המפתח הבא:

עבור $n=10^6$ רשומות, כל אחת בגודל סקטור (גודל הקובץ בערך ½ GB):

$T_{scan} = n \cdot T_{succ} = 10^6 \cdot 5.71\ ms = 5{,}710\ sec = 1{:}35{:}10\ hours$

<u>אלטרנטיבה:</u> למיין את הקובץ.

כל מיון דורש כ-22 דקות (בשני דיסקים) או כ-51 דקות (בדיסק אחד)

– החישוב הזה לא מביא בחשבון שהמבנה לא לגמרי מלא. למעשה, המעבר הראשון של המיון קצת יותר ארוך.

- אחרי עדכון צריך להחזיר למבנה המקורי ⇐ למיין לפי המפתח המעורבל

עדכון batch של מבנה ערבול

בקובץ U יש k פעולות עדכון לקובץ מקור F במבנה טבלת ערבול

- **<u>שיטה נאיבית:</u>** לכל פעולה ב-U גש ל-F ועדכן

  הזמן הנחוץ בשיטה זו הוא $k \times T_{insert}$

  לכל עדכון, seek ממוצע והשהיית סיבוב ⇐ k תזוזות זרוע (כל-אחת למרחק ממוצע)

- **<u>עדכון ממוין:</u>** מיין את U לפי h(u), ומזג את הקובץ שהתקבל עם הקובץ F מעבר על F לפי הסדר הפיזי, אפשר לדלג על מסילות/גלילים שלא מעדכנים

  לכל היותר $\min\{C,k\}$ תזוזות זרוע (אם הקובץ תופס C גלילים), ורובן למרחק קצר

פעולות על יחסים באמצעות ערבול: רעיון בסיסי

- ביצוע רב הפעולות במעבר יחיד דורש זיכרון כגודל הנתונים
- לולואות מקוננות דורשות מעברים רבים על הנתונים
  - לא את כל הפעולות ניתן לבצע באמצעות לולאות!

- ערבול מאפשר חלוקה של הטבלה לחלקים **בלתי תלויים**
  - **רשומות בתת טבלה א' לא רלוונטיות לחישוב על תת טבלה ב'**
  - הפעולה מבוצעת על כל תת-טבלה בנפרד
  - תתי הטבלאות קטנות ונכנסות לזיכרון

פעולות על יחסים באמצעות ערבול: רעיון בסיסי

- במעבר מקדים מחלקים את כל אחד מהיחסים לדליים באמצעות פונקצית ערבול מתאימה
- כל דלי יכיל מספר בלוקים
- כאשר החוצץ של דלי מלא, הוא נכתב לדיסק

- במעבר העיקרי, בכל שלב מביאים לזיכרון את הדליים המתאימים של היחסים, ומחשבים את הפלט על סמך הדליים האלה

δ בעזרת ערבול

הקובץ המקורי: 53,42,73,81,22,53,91,61,53,11,92,91
מעבר ראשון: נחלק את הקובץ לדליים (רשומות זהות ממופות לאותו דלי).
1: 81,91,61,11,91
2: 42,22,92
3: 53,73,53,53

מעבר שני: נעבור על כל הדליים. לכל אחד:
נקרא את כל הדלי לזיכרון
נבצע δ על הדלי (נשמור עותק אחד מכל רשומה):
1: 81,91,61,11,91
2: 42,22,92
3: 53,73,53,53

למה צריך יותר ממעבר אחד?

למה לא לסלק את הכפולים כאשר מנסים להכניסם לדלי?

מעבר ראשון: מספר הדליים בחלוקה

במעבר הראשון, מחלקים את הקובץ לדליים.
לשם כך, צריך להשתמש בחוצצים: לקלט ולכל אחד מדליי הפלט

צריך שני חוצצים לקלט ושני חוצצים לכל דלי
⇦ $M \geq 2k + 2$ (גודל הזיכרון)
ז"א $k \leq M/2 - 1$

δ בעזרת ערבול: מעבר שני

B(R) > M (מספר הבלוקים בקובץ), אחרת מספיק מעבר אחד

מעבר שני : נעבור על כל הדליים.
קרא דלי שלם לזיכרון, בצע δ על הדלי (כתוב העתק אחד מכל רשומה).

כל הדלי צריך להיכנס לזיכרון ⇦ גודל הדלי ≥ M
אם $B(R)/k \leq M-2$, אז שני מעברים מספיקים
⇦ $B(R) \leq kM = M\left(\frac{M}{2}-1\right) \approx \frac{M^2}{2}$

ובסך הכל, 3B(R) גישות דיסק (לא מחייבים על כתיבת התוצאה)

δ בעזרת ערבול: אם הדלי לא נכנס לזיכרון

אם גודל הדלי גדול מ- M
- במעבר הראשון נחלק ל- k = M/2-1 דליים,
- במעבר השני, נחלק כל דלי ל- k דליים,
- ...
- במעבר ה-h נחלק כל דלי ל- k דליים שגודלם אינו עולה על M.

מספר הדליים k^h, וגודל כל דלי (בערך) $\frac{B(R)}{k^h}$

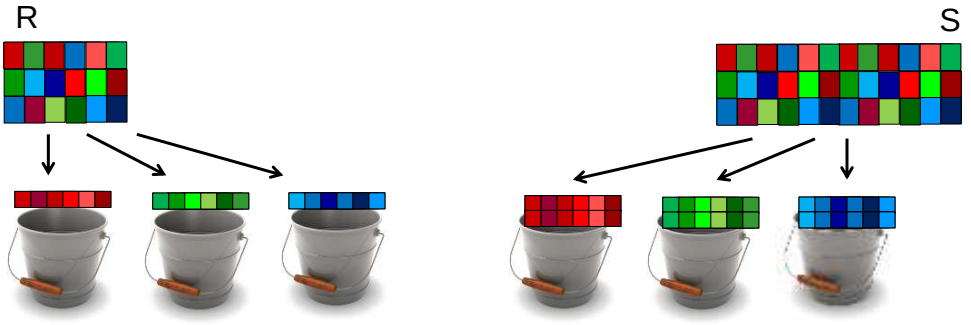
מספר המעברים: $h = \lceil \log_k B(R)/M \rceil$

מספר גישות דיסק: $(2h+1)B(R)$

(לא מחייבים על כתיבת התוצאה)

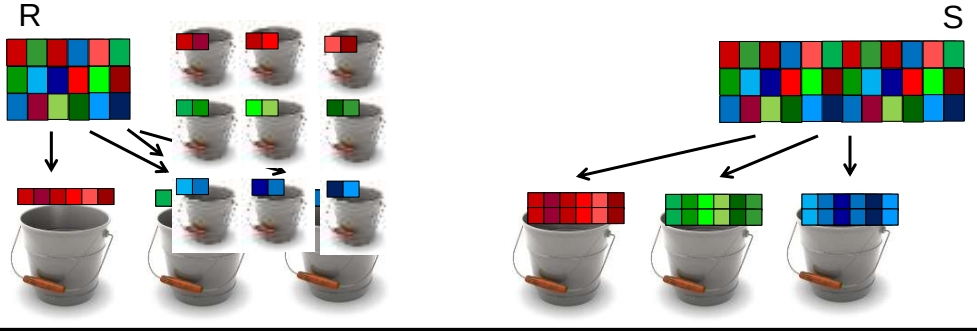
פעולות בינאריות באמצעות ערבול

- מחלקים את שתי הטבלאות R ו-S לתתי טבלאות כמו שראינו
- **רשומות בתת טבלה א' של R רלוונטיות רק לחישוב על רשומות בתת טבלה א' של S!**
  - פונקציית הערבול צריכה לשלוח רשומות מתאימות לדליים מתאימים
  - ←פונקציה זהה לשתי הטבלאות (מופעלת על עמודות משותפות)



פעולות בינאריות באמצעות ערבול

- איך נבחר את k?
  - חלוקת הטבלאות בלתי תלויה - נשתמש בכל הזיכרון לכל טבלה בנפרד
- כמה מעברי ערבול דרושים?
  - בביצוע הפעולה על תתי הטבלאות, מספיק שתת הטבלה הקטנה תיכנס לזיכרון
  - האם מספיק לחלק שוב רק את הטבלה הקטנה?



חיתוך בעזרת ערבול

מניחים שלכל i, הדליים $B_S[i]$ או $B_R[i]$ נכנסים לזיכרון

```
for all s ∈ S
 write s to bucket B_s[h(s)];
for all r ∈ R
 write r to bucket B_R[h(r)];

for i=1 to number_of_buckets {
 find the intersection of B_s[i] and B_r[i]
 using the single-pass algorithm
 & produce the sub-relation T_i
 output T_i
}
```

חלוקת S לדליים	2B(S)
חלוקת R לדליים	2B(R)
קריאת דליי S	B(S)
קריאת דליי R	B(R)
זמן כולל	**3B(S) + 3B(R)**

צירוף בעזרת ערבול

תהי A קבוצת התכונות המשותפות

- חלק את S לדליים $S_1,\ldots S_k$ ע"פ h(s.A)
  כך שלכל i: $B(S_i) \le M$
- חלק את R לדליים $R_1,\ldots R_k$ ע"י h(r.A)
- עבור i=1,..,k חשב $S_i \bowtie R_i$ (כיצד?)

גישות לדיסק

2B(S)	
2B(R)	
$\Sigma_i(B(S_i)+B(R_i))$	
3B(S)+3B(R)	סה"כ

אחסון מבוזר על קצה המזלג

- המטרה: שיפור ביצועי המערכת על ידי שימוש בשרתים מרובים
  - כל שרת הוא מכונה נפרדת עם מעבד, זיכרון ראשי וזיכרון משני
- נפח האחסון הזמין גדול יותר
- גישה לנתונים מתפזרת בין השרתים ומבוצעת במקביל

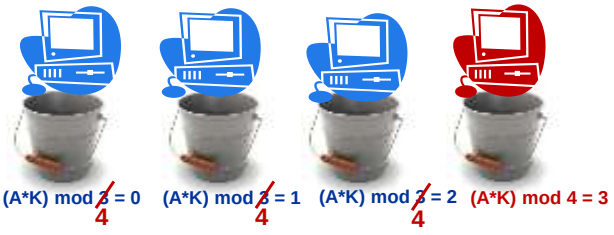
- האתגרים
  - על נפח האחסון במערכת לגדול בהתאמה לגידול בנתונים (scalability)
  - התפלגות הגישה לנתונים לא ידועה מראש ועלולים להיווצר צווארי בקבוק
  - **תמיד** יש נפילות
  - תלות בשרת מרכזי (למשל לצורך אינדקס) עלולה לעצור את המערכת במקרה של נפילה, עומס זמני, או גידול

טבלת ערבול מבוזרת: העקרון

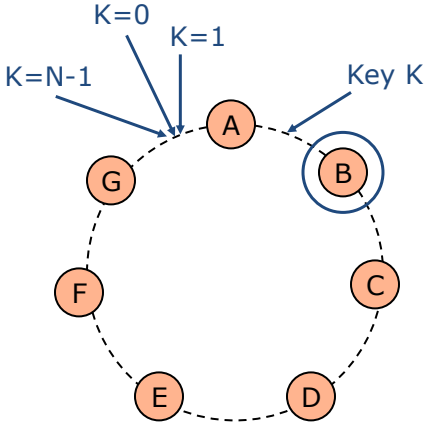
- הנתונים נשמרים בדליים
- הדליים ממוקמים פיזית על שרתים שונים
- הנחות:
  - מאגר מפתח-ערך שכל פעולה בו ניגשת לנתונים בודדים
  - סדר גודל המערכת הוא של מאות שרתים (חסם עליון)
  - כל השרתים "מכירים" אחד את השני
  - תקשורת יעילה בין שרתים
- דרישות:
  - קל למצוא את השרת עליו נמצא כל אובייקט
  - עומס האחסון והשירות מחולק בצורה אחידה בין השרתים

טבלת ערבול מבוזרת: מימוש נאיבי

- בוחרים מספר אקראי A
- במערכת ישנם N שרתים
- אובייקט עם מפתח K נמצא על שרת (A*K) mod N
  - ✓קל למצוא את השרת עליו נמצא כל אובייקט
  - ✓עומס האחסון והשירות מחולק בצורה אחידה בין השרתים (בזכות האקראיות)
- מה קורה כאשר N משתנה?
  - צריך למקם מחדש את כל האובייקטים!

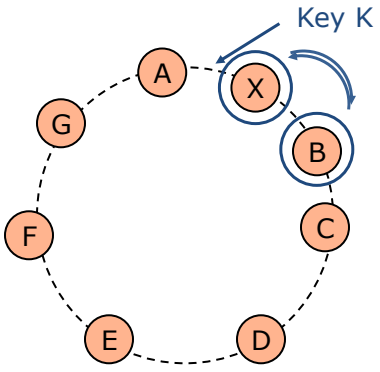


ערבול עקבי (Consistent Hashing)



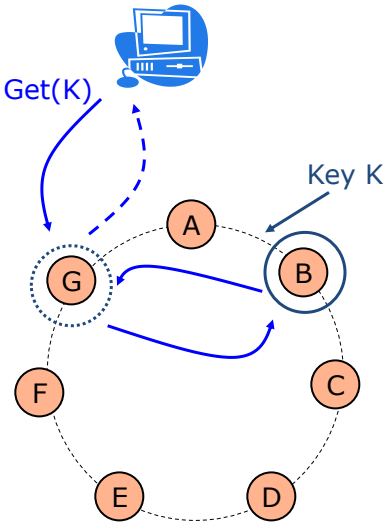
- פונקציית הערבול מתאימה מפתח לכל אובייקט
- טווח המפתחות מיוצג כמעגל
- כל שרת "מגריל" מפתח שמגדיר את מיקומו במעגל
- אובייקט עם מפתח K נמצא בשרת הראשון שהמפתח שלו גדול מ-K
  - ✓קל למצוא אובייקט
  - ✓העומס מחולק בצורה אחידה

הוספת/הוצאת שרת



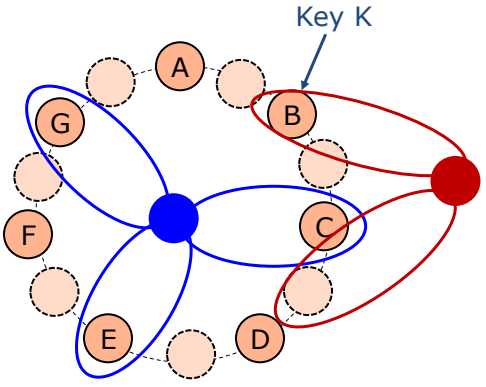
- בהוספת שרת חדש מגרילים עבורו מפתח ומודיעים לשאר השרתים
- חלק מן האובייקטים של השכן העוקב עוברים לשרת החדש
- בהוצאת שרת קיים מעבירים את האובייקטים שלו לשכן העוקב
  - ✓קל למצוא אובייקט (כמו קודם)
  - ✓העומס מחולק בצורה אחידה (כמו קודם)
  - ✓שינוי המיקום מוגבל לאובייקטים שנמצאים אצל שרת אחד בלבד (אופטימלי)

גישה לאובייקט



- בקשת גישה לאובייקט מגיעה לטיפול שרת אקראי
  - על פי עומס, מיקום גאוגרפי וכו'
- השרת המטפל מנתב את הבקשה לשרת המחזיק באובייקט
  - בכל שרת נשמרת טופולוגיית הטבעת: כתובות כל השרתים והמפתחות שלהם
    - ✓סדר גודל מספר השרתים חסום
    - ✓אין שרת מרכזי

חלוקת עומס (Load Distribution)



- הטווחים מוגרלים ואינם אחידים
- לא לכל השרתים אותה קיבולת
  - כמות דיסקים, מעבדים, גיל
- לא כל האובייקטים מבוקשים באותה תדירות

← ייתכן שבשרתים מסויימים יהיה עומס בעוד ששרתים אחרים פנויים

← כל שרת מגריל מספר מפתחות
- ניתן להגריל על פי הקיבולת
- ניתן להחליף תפקידים

שיקולים נוספים (על קצה המזלג)

- כל אובייקט משוכפל על מספר שרתים
- טיפול יעיל בנפילות שרתים
  - בנפילה זמנית או אובדן תקשורת ניתן לגשת לעותק של האובייקט
  - בנפילת שרת סופית משתמשים בעותקים כדי לשחזר את האובייקט
- טיפול יעיל בצווארי בקבוק
  - בזמן עומס על אובייקטים הבקשות יופנו לשרתים המחזיקים עותקים
- בהמשך הקורס נדבר על ההשלכות של שכפול

- לקריאה נוספת:

- DeCandia, G., Hastorun, D., Jampani, M., Kakulapati, G., Lakshman, A., Pilchin, A., Sivasubramanian, S., Vosshall, P., and Vogels, W. Dynamo: Amazon's Highly Available Key-value Store. *SOSP 2007.*

MapReduce על מאגר נתונים מבוזר

- במקרים רבים הקלט לשאילתה מפוזר מלכתחילה בין שרתים
- Map() מתבצע מקומית על השרת שמאחסן כל חלק של הקלט