משימה 4- חלק א'

מגישים: נוי משט 205835275
בנימין פרסצקי 310727771

שאלה 1:

n – מספר הצמתים (בשני העצים)

2t-1 – מס' הבלוקים בכל צומת בB-tree (צמתים מלאים)

D – גודל כל בלוק

2t – מספר המצביעים מכל צומת

המקום הדרוש לאחסון כל הצמתים בעץ - הוא גודל כל בלוק, כפול מס' הבלוקים בכל צומת, כפול מס' הצמתים בעץ.

$$n * (2t - 1) * D$$

המקום הדרוש לאחסון כל המצביעים - הוא גודל כל מצביע (בייט 1), כפול מס' המצביעים בכל צומת, כפול מס' הצמתים בעץ.

$$1 * n * 2t$$

המקום הדרוש לאחסון B-tree הוא המקום הדרוש לאחסון הבלוקים ועוד המקום הדרוש לאחסון כל המצביעים:

$$n * (2t - 1) * D + n * 2t$$

המקום הדרוש לאחסון כל צומת בMBT - גודל כל צומת (בכל צומת מאוחסנת החתימה של הצומת שגודלה הוא גודל פלט הפונקציה SHA1- 20 בייט), כפול מס' הצמתים בעץ.

$$20 * n$$

המקום הדרוש לאחסון כל המצביעים ב MBT - כמו בB-tree.

$$n * 2t$$

המקום הדרוש לאחסון MBT הוא המקום הדרוש לאחסון הצמתים ועוד המקום הדרוש לאחסון כל המצביעים:

$$20 * n + 2t * n$$

היחס בין המקום הדרוש לאחסון B-tree למקום הדרוש לאחסון MBT הנגזר ממנו:

$$\frac{n * D * (2t - 1) + n * 2t}{20 * n + 2t * n} = \frac{D * (2t - 1) + 2t}{2t + 20}$$

שאלה 2:

עדכון הB-Tree ע"י הכנסה או מחיקה של בלוק לא מחייב חישוב מחדש לכל צמתי הMBT.

על מנת לנסח מחדש אלגוריתמי הכנסה ומחיקה נוסיף לכל צומת שדה בו נשמור את החתימה שלו.

אלגוריתם הכנסה:

במהלך ההכנסה עוברים מהשורש עד לצומת בו צריך להכניס את הבלוק החדש. בכל צומת שנעבור החל מהשורש, נאפס את ערך החתימה.
כאשר נגיע לצומת עם 2t-1 בלוקים נצטרך לבצע פיצול. במקרה זה, נבדוק לאיזה מהבנים שנוצרו בפיצול נצטרך להמשיך לרדת בתת העץ שלו, עד לצומת שבו נצטרך להוסיף את הבלוק החדש.
לבן זה, נאפס את החתימה. לבן השני נחשב מיד את החתימה החדשה שלו.
כך נמשיך עד שנגיע לצומת שבו נכניס את הבלוק החדש. מכיוון שווידאנו שבכל צומת יש פחות מ2t-1 בלוקים, נוכל לבצע את ההכנסה. נחשב את החתימה החדשה של הצומת עם הבלוק החדש.
כעת, קיבלנו ענף בעץ שבו מהשורש ועד לצומת בו ביצענו את ההכנסה, כל החתימות של הצמתים מאופסות.
נרצה לחשב מחדש את כל החתימות של הצמתים שאיפסנו.
חתימת כל צומת תלויה בחתימות הבנים שלה, ומכיוון שעד ההכנסה איפסנו לכל צומת שעברנו בה בן אחד, נצטרך לחשב מחדש את כל החתימות המאופסות.
על מנת לעשות זאת נתחיל מהשורש. נרצה לחשב מחדש את החתימה שלו, אך מכיוון שלאחד הבנים שלו אין חתימה עדכנית, נצטרך לחשב גם אותה.
כך, נבצע בצורה רקורסיבית חישוב מחדש של כל חתימה מאופסת עד שלכל הבנים של השורש תהיה חתימה עדכנית, ואז נוכל לחשב גם את החתימה שלו.
מכיוון שחישבנו את החתימה של הצומת המכילה את הבלוק החדש, החישוב הרקורסיבי יעצור כאשר נגיע לצומת שלכל הבנים שלו יהיו חתימות, כלומר לאבא של הצומת בו הוכנס הבלוק החדש.

אלגוריתם מחיקה:

אלגוריתם זה דומה ברובו לאלגוריתם ההכנסה מבחינת אופן הביצוע שלו, בכל צומת שנעבור בו החל מהשורש ועד לצומת המיועד למחיקת הבלוק, נאפס את החתימה.
כאשר נגיע לצומת עם t-1 בלוקים נבצע אחת משתי הפעולות הבאות:
1. shift- כאשר לאח השמאלי או לאח הימני יש יותר מ t-1 בלוקים. במקרה זה לאחר הshift נעדכן מיד את החתימה של האח. את החתימה של הבן נאפס ונמשיך לרדת בתת העץ שלו עד לצומת בו נמצא הבלוק שצריך למחוק.
2. merge- כאשר גם לאחים יש פחות מt בלוקים, נצטרך לבצע מיזוג עם האבא והאח השמאלי, ואם הוא לא קיים אז עם האח הימני. במקרה זה, נאפס את החתימה של הצומת החדש ונמשיך לרדת בתת העץ שלו.
כשנגיע לצומת בו נמצא הבלוק המיועד למחיקה,
אם הוא עלה – מכיוון שווידאנו שיש לו יותר מt-1 בלוקים נוכל לבצע את המחיקה ולחשב החתימה שלו מחדש.
אם הוא לא עלה – נרצה לשנות את הבלוק המיועד למחיקה עם הבלוק של העוקב או הקודם אליו.
במקרה זה, נאפס את כל הצמתים שנעבור בהם עד לעוקב או הקודם, ואחרי השינוי נמחק את הבלוק העוקב או הקודם ונחשב את החתימה של הצומת שלו מחדש. אם לשני הילדים אין מספיק בלוקים נבצע איתם מיזוג, ואז אחרי מחיקת הבלוק נחשב מחדש את החתימה של הצומת. בשני המקרים נשאר לנו ענף של צמתים שחתימתם מאופסת.
כעת באותו אופן כמו בהכנסה, נחשב מחדש בצורה רקורסיבית את החתימות של כל הצמתים שאיפסנו להם את החתימה.

חישוב סיבוכיות זמן ריצה:

לאלגוריתמים אותו זמן ריצה.
במקרה הגרוע, על מנת למצוא את הצומת שבה תתבצע ההכנסה/מחיקה, נעבור בכל צומת על O(t) בלוקים.
אם הצומת המבוקש הוא עלה נרד כגובה העץ $O(\log_t n)$.
חישוב מחדש של חתימה של כל צומת יהיה O(t) לפי מה שלמדנו בהרצאה על פונקציות גיבוב.
ולכן, זמן הריצה הדרוש בכל אלגוריתם הוא $O(t * \log_t n)$.

חישוב סיבוכיות מקום:

במקרה הגרוע ביותר, בהכנסה- בכל צומת יש מקסימום בלוקים ובמחיקה- בכל צומת יש מינימום בלוקים, או שניצור מצביע חדש או שנמחק מצביע קיים. מכיוון שבמקרה הגרוע ביותר נעבור על גובה העץ, ונשנה מצביע אחד, סיבוכיות המקום של כל אלגוריתם היא $O(\log_t n)$.

<u>שאלה 3:</u>

הסיבה שהמתכנתים החליטו לאתחל את המשתנים למשתנים קבועים היא כדאי שלפונקצית הגיבוב SHA1 יהיה גיבוב אחיד ושלכל קלט יהיה פלט קבוע. הסיבה שהמתכנתים החליטו לפרסם את המשתנים היא כדי להראות אמינות לציבור המשתמשים בפונקציה. כביכול, אם המספרים לא היו מפורסמים לציבור אז היה עולה חשש שהמתכנת שכתב את הפונקציה יכול ליצור התנגשויות מכוונות. כאשר הם פרסמו את המספרים של הפונקציה, כוונתם הייתה לעורר אמינות בפוקנציה כך שהם מראים שהמשתנים הם קבועים ורנדומליים וגם להם אין את הדרך לעורר התנגשויות או לבצע פגיעה במשתמשים בפונקציה.