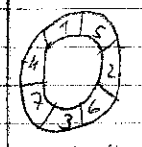
הרעיון של יוניקס ולינוקס הוא ליצור רשימה של כל הכוחות, משם ע"י הפיצול מתאים מול הרכיב עדמו. הרעיון מעניין את ומאפשר הבנה טובה של המערכת גם מבלי שנדע כדי לבנות שנגיה (נניח שהרכיב לא קיים) אפשר רק בסיס שרשרת של הקריאות.

SSD בדרך הוא לא יותר מהיר מכונני דיסקים. בגלל כל"י, למרות שזה חדש ניתן להביט לשמירות יותר מהירות בלי הפלת. תלוי מעיד עליהם. נשים ב התשיני של התפתחות הדיסקים אינה פעולת מלא תוך "מאיר". הסיבה לכך הוא של הדיסק הוא הכוח אנלוגי.

הסיבה שמיקום בזיכרון בינתית המת הנית מתשב הוא סיבה כלכלית (לא מתמטית ונכונה וכו') כי גם הסיבה שבדיסקים כל הרושם רגיל לאותו כיוון (כמקשה אחת).

תשובות בלוק הוא היחידה המינימלית שמעבירים ביחד בדיסק (כמו) לחלק גם בכ אחד אם נקבל משיקולי יעילות). הגיון המקסימלי וכו' לגת כל הדיסק. הגיון הטוב נקבל מי סטטיסטיקת של פעולות דיסק.

כשהר חלקים פקריא מן הדיסק 2 בלוקים רציפים, (תקן) בבעיה שנה שהתקבלה הקודמת לקרוא את הבלוק הבא, דרוש לחכות סיבוב שלם כדי הראש כבר עבר את הבלוק. כדי לפתור את הבעיה נשתמש בפיתרון של interleave (סריגה) כך, הראש יהיה במקום המתאים ולא נצטרך לחכות סיבוב שלם.



סיבוב של הדיסק

בקשות read ו- write אורכות ל interrupt. בן לא מאפשרים כל פעולות אלה בשתשבים מחכו לפעולת וגם אלו תלוי הקונפיגורציה) אלא עם לגמרי הן מעדיפות להשאר בחלון הבקשה מחרות בין מעדיף בקלות לבין הקריאה בדיסק לדיסק.

הבקשות הנכנסות מגיעות לתקיעות שונים בדיסק. השאלה בעיה סדר לבצעם את התורך מולנו מעדיפים שזו יעילות הכנות. השיטה הראשונה היא מרמר (FIFO). כי שיטה מעדיף הוגנת אך גם מעיד לא יעילה.

שיטה נוספת היא SSTF (לקוחר = קצר בישר הקר הראשון). בשיטה כי אינו מתחרים את הבקשה שרבו קרובה אינו מבריות יעילות השיטה הזו טובה מאוד אך כי גם בישה מעיד לא הוגנת ויכולה להגיע למצב של STARVATION, כלומר עם נגיע אל בקשתו אל בעיה.

בדרך מנסים להגיע לאמצע בשרה הדומה למעלה, כלומר, הראש כל הזמן רץ ימינה, אם הגיע בסוף הבקשה, הוא ישרת אותה, אחרת ימשיך הלאה, בסוף יחזיר להתחלה.

יש מספר וריאציות של שיטת המעלית scan מבדלות סדרת עד הסוף וחוזרה בשיטת הבקודה מבדלת כשמגיעה לסוף תוכנת להתחלה ובדרך מבדלת בקשות גם בחזרה (C-scan) פעם אחת בדרך אחת בהגיעו לסוף הקיצוני (הגרוע מאוד) להתחלה ואז מבדלת בקודות בדרך חזרה.

look - בדומה ל scan אך בינתיים להמשיך עד הסוף ממשיך עד אבל הסקודה הקיימת כלומר עד הבקשה ביותר שיש מבוקשת בקשה. C-look - בדומה דומה ל look ו C-scan.

RAID (מדגי = מערך דיסקים ע"י יתר) - הרעיון הוא לקחת מספר דיסקים ולהתיחס אליהם בהתחלה אחד. בגלל מגיע מרווחים מקבילות

בוא נקח את הרעיון של ה RAID (מערך כונני) והשתמש במקבילו מנוע של משפחות גדולות.

נשים לב שהסכנה בשימוש ב RAID הוא במקרה ויש הרבה דיסקים מתקלקל
במקרה כזה אנחנו בבעיה כי אם נאבד את כל המידע בדיסק אחד (כל אחת בדיסק) אז אם נאבד דיסק ספציפי, יכול לנו לאבד הכל ה-0 כל המידע
יש דרך להתגבר על כך בעזרת ה- R

כל RAID-ים יש הרבה מאפשרים יש לבדוק בכל שלמה בכלל.

RAID 0 - בשיטה זו מחלקים כל בלוק לדיסק אחר (הבלוק הראשון בדיסק הראשון הבלוק השני הדיסק השני וכו' ...).

במקרה בו קיימים 4 דיסקים ניתן לקרוא את הבלוקים 0,1,2,3 במקביל. את בלוק 4 לא ניתן לקרוא יחד עם 0. ניתן לקרוא לבלוק "בדיעבד".

RAID 1 - מבנה שיכפול של כל המידע. כמות המידע שניתן לשמור הוא יותר מאשר הכמות ניתנת בדיסק.

RAID 3 - בשיטה זו מחלקים דיסק אחד לשמירת בלוקים. בשיטה זו אינו מגדיל את הכמות ביחס ל RAID0 ובנוסף מאפשרים שמירת מידע בכדי יותר מכונני האמצע. רק אם אם דיסק אחד נופל (נניח יודעים איזה) ניתן לשחזר את המידע שאבד. (שאר הדיסקים ב RAID3 לא יפגעו בשיטת הלמידה).

RAID 5 - דומה ל RAID3 אך במקום לשמור את הבלוקים בדיסק אחד, שומרים אותו מחולקים בכל דיסק. בשיטה זו אינו מגבילים את המקביליות תוך שמירה על אמינות. (דוגמה לשיפור מהתרשים בשקף שבמרכז היא במצב של כיבוי אזור 1 - 6 ו - 14).

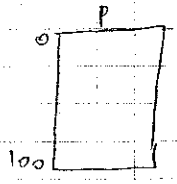
נשים לב שגם במקרה הגרוע RAID5 יהיה יותר טוב מ RAID3.
נשים לב שב - RAID5 כדי לכתוב אנחנו נדרשים לקרוא קודם את מידע ישן. ב RAID3 התכנון מיועד לרציפות לכן ניתן לבצע יותר את הכתיבה.

דיסק אופטי - שומר את המידע בצורה כזו שכל פיט שיגדר לא להתעוות אור (לא בשלב המאגנט).

סרטים מגנטים יש משהו לא מעולה שניתן למגנט. מיועד בעיקר לגיבויים של אינטרנט ולא לשמש עמיד שיטפה (נמוך בישה ברחבים).

GPU - יחידה גרפית (מחשב לא רבה). היחידה הגרפית (בשלב כלשהו) בקיר מחשוב של פיקסלים. בכל פיקסל יש מידע על עוצמת הצבע של כל אחד מרכיבי ה- RGB.

נתון לימודי
קרוא, מערכת ההפעלה דרושה בישה לעמדן, נדרה השתמש בדיסק בתור התחבת לדיו.



הפרדה בין כתובת פיזית לכתובת לוגית (ווירטואלית) - כל תוכנית מתחילה בכתובת 0. גודל כתובת וירטואלית הממופה לכתובת פיזית בזיכרון (כדי) אפשר לכל התוכנית להתחיל מ 0 מבלי לדרוס מידע של תוכנית אחרת.
אומנם המיפוי מסופק ע"י מערכת ההפעלה.