УДК 004.63

ЕФЕКТИВНЕ ВИКОРИСТАННЯ УТИЛІТ LINUX ДЛЯ ПОШУКУ ДУБЛІКАТІВ ФАЙЛІВ

Сокорчук І.П.
ihor.sokorchuk@nure.ua
Харківський національний університет радіоелектроніки, кат. ПІ
Харків, Україна

This paper explores efficient methods for detecting duplicate files in the Linux file system, including the use of specialized utilities (fdupes, rdfind), standard commands (find combined with xargs, parallel, md5sum, sha256sum, sha512sum), Python scripts, and built-in file system capabilities (btrfs). The advantages and disadvantages of each approach are analyzed, along with their efficiency depending on data volume and hardware resources. Special attention is given to performance and parallel processing capabilities. Recommendations are provided for selecting appropriate methods for different application scenarios.

Пошук дублікатів файлів є важливим завданням при адмініструванні комп'ютерних систем, оптимізації сховищ даних і підвищенні ефективності використання дискового простору. Одним із найбільш гнучких та ефективних підходів є використання командного інтерпретатора Bash [1] у поєднанні з утилітами Linux: find [2], xargs [3], md5sum [4], sha256sum, sha512sum, sort та uniq.

Для пошуку дублікатів файлів у файловій системі можуть бути використані спеціалізовані утиліти fdupes або rdfind. Утиліта fdupes дозволяє здійснювати пошук дублікатів на основі контрольних сум та побайтового порівняння, тоді як rdfind використовує гешування та аналіз бітового вмісту. Окрім цих утиліт, доцільно використовувати стандартні програмні засоби Linux, такі як find у поєднанні з md5sum або sha256sum, що дозволяє ефективно знаходити дублікати без встановлення додаткового програмного забезпечення. Використання утиліт xargs та parallel [5] забезпечує можливість розпаралелювання процесу порівняння файлів, що особливо важливо для великих файлових систем. Зокрема утиліта xargs дозволяє виконувати команди для великої кількості аргументів, які вона отримує з потоку введення. Перевагами цієї утиліти є підтримка розпаралелювання через параметр -P. Вона дозволяє передавати аргументи у пакетному режимі, що знижує кількість викликів команд. Команда md5sum використовується для генерації MD5 контрольних сум файлів, що дозволяє порівнювати вміст файлів і знаходити дублікати. Команда sort сортує вхідні дані, а команда uniq дозволяє знайти унікальні текстові рядки або навпаки рядки, які повністю або частково повторюються. Приклад фрагменту bash скрипта, який знаходить дублікати файлів у вказаній у

змінній chkdir директорії: find "$chkdir" -type f -print0 | xargs -0 -P $(nproc) md5sum -b | sort | uniq -w 33 -D

Утиліта xargs використовується тут для пакетної обробки аргументів, переданих через стандартний вхідний потік. Вона дозволяє уникати перевищення обмежень на довжину командного рядка, розбиваючи аргументи на групи. Ця утиліта підтримує обмежене розпаралелювання за допомогою опції -P. У наведеному скрипті в команді xargs використано ключ -P $(nproc) для розпаралелювання обробки даних на усі доступні на платформі ядра, що дозволяє підвищити ефективність обробки для багатоядерних процесорів. Перевагою утиліти xargs є те, що вона входить у стандартний набір утиліт GNU, що забезпечує її доступність на будь-якій системі Linux. Вона ефективно використовує пам’ять, оскільки працює з пакетами даних. Ця утиліта добре інтегрується з іншими утилітами як, наприклад, find, grep, awk. Недоліком є обмежена підтримка у ній розпаралелювання (ключ -P дозволяє запустити кілька процесів, але не забезпечує гнучке управління ними). Вона не дозволяє керувати розподілом процесів між ядрами процесора. У ній відсутні механізми балансування навантаження між процесами.

Інші альтернативним методом пошуку дублікатів файлів може бути використання утиліти find та утиліти parallel: find "$chkdir" -type f -print0 | parallel -q0 md5sum | sort | uniq -w 33 -D

Утиліта GNU parallel є утилітою для паралельного виконання команд із використанням багатоядерних процесорів. Вона дозволяє автоматично розподіляти навантаження між доступними ядрами. Ця утиліта дозволяє задавати кількість одночасних процесів (–jobs або -j). Вона дозволяє запускати команди не тільки у межах одного сервера, але й паралельно на кількох серверах у мережі, Результати обробки з допомогою цієї утиліти виводяться завжди у правильній послідовності. Перевагами цієї утиліти є висока ефективність у використанні багатоядерних систем, гнучке управління параметрами розпаралелювання, підтримка складних шаблонів введення та виведення, інтерактивний режим з можливістю перегляду проміжних результатів. Утиліта parallel забезпечує ефективніше управління розпаралелюванням у порівнянні з xargs. Недоліком є те, що ця утиліта не входить у стандартний набір утиліт і потребує окремого встановлення (sudo apt install parallel). Вона має вищі витрати ресурсів порівняно з xargs, що може бути критичним на слабких системах. Деякі команди, які викликаються на виконання з допомогою цієї утиліти, можуть мати проблеми з обробкою великого обсягу даних, якщо ці команди були неправильно оптимізовані. У таблиці 1 наведено порівняння утиліт xargs та parallel. Утиліту xargs доцільно використовувати для простих операцій із невеликим рівнем паралельності. Утиліта parallel є кращою для високонавантажених задач, які потребують ефективного використання всіх ядер процесора або виконання цих задач на кількох серверах у мережі.

У разі потреби мінімальних витрат пам’яті та забезпечення сумісності, доцільно використовувати xargs. Якщо пріоритетом є швидкість обробки великих масивів даних, то кращим вибором може бути утиліта parallel.

Таблиця 1 – Порівняння утиліт xargs та parallel

Параметр	**xargs**	**parallel**
Розпаралелювання	обмежене (-P)	гнучке (–jobs)
Балансування навантаженням	немає	є
Доступність	входить в GNU	потребує встановлення
Оптимізація	низька	висока
Гнучкість	обмежена	висока
Розподіл між ядрами	відсутній	є автоматичний
Витрати ресурсів	низькі	вищі
Виконання завдань на кількох серверах	немає	є

Для створення контрольних сум файлів для подальшого пошуку однакових файлів, крім утиліти md5sum, можуть бути застосовані утиліти sha256sum або sha512sum. Приклад використання цих утиліт:

```
find /path/to/directory -type f -exec sha256sum {} + | sort | uniq -w 64 -D
```

Перевагою є те, що ці утиліти є стандартними програмними засобами. Використання sha256sum або sha512sum замість md5sum підвищує надійність при порівнянні файлів. Недоліком є їхня повільніша робота на великих наборах даних.

Python-скрипти на основі бібліотеки hashlib дозволяють створювати більш гнучкі рішення для пошуку дублікатів, особливо у випадках складної фільтрації чи додаткової обробки результатів. Вбудовані механізми файлових систем, такі як дедуплікація в btrfs, забезпечують автоматичне управління дублікатами на рівні файлової системи без необхідності ручного пошуку.

Список використаних джерел:

1. Офіційна сторінка проєкту GNU Bash Manual : вебсайт. URL: https://www.gnu.org/software/bash/manual (дата звернення: 03.03.2025).

2. Офіційна сторінка проєкту GNU Findutils Manual : вебсайт. URL: https://www.gnu.org/software/findutils/manual/ (дата звернення: 03.03.2025).

3. Офіційна сторінка проєкту Linux Man Pages Xargs Manual : вебсайт. URL: https://man7.org/linux/man-pages/man1/xargs.1.html (дата звернення: 03.03.2025).

4. Офіційна сторінка проєкту GNU Coreutils Manual : вебсайт. URL: https://www.gnu.org/software/coreutils/manual/ (дата звернення: 03.03.2025).

5. Офіційна сторінка проєкту GNU Parallel Manual : вебсайт. URL: https://www.gnu.org/software/parallel/ (дата звернення: 03.03.2025).