An Overview of the Bonnie++ Benchmarking System

- Gary Miller

Key Features

The main purpose of the bonnie++ system is to test the performance of hard drives and file system operations. It allows users to receive a detailed report of the degree to which certain operations are used and makes it possible to analyze the overall efficiency by reporting values of throughput and processing time. The system's readme document outlines the main features of the system to be associated with testing input and output throughput similar to the way databased applications would conduct these operations and test the cost associated with the creation, reading, and deletion of files. The system does this by using six difference file IO tests the fall into three different categories.

Sequential Output

The first category of tests focuses on sequential output and consists of three tests. The first sequential test measures CPU overhead of per-character operations and is computed by adding the time of output code plus space allocation. The next test is similar to the first but focuses on using a block, which eliminates the overhead associated with output code and is only affected by space allocation. The final sequential output test focuses on rewriting data. This is useful should testing the effectiveness of data caching and the speed of transferring data.

Sequential Input

The next category of test focuses on sequential input and consists of two tests. the first test is per-character input and is similar to the per-character test of sequential output except the overhead is measured by the cost of input code execution and inputing the data. The second test focuses on sequential input using blocks. This test simply measure the cost of reading in data and therefore is a very accurate description of the cost on read operations.

Random Seeks

This consists of only one test and focuses on the effectiveness of seeking for specific locations. This is done by randomly selecting a location for by identified, this location is then read. In 10% of the cases, the location is written with new data. This test does a good job at describing how well a system can locate certain objects.

Parameters

Bonnie++ accepts several different parameters that can help make the system run in a more effective fashion, the list of these parameters are outlined in the system's manual. Each parameter is used by typing its flag followed by the value of the parameter.

- -d allows the user to specify the directory to be tested
- -s specifies the size of the files, in megabytes, that will be tested
- -n specifies the number of files, in multiples of 1024, used in file creation
- -r allows user to specify RAM size
- -x allows the user to specify the number of test runs to be conducted
- -u specifies the user-id, which identifies who is conducting the tests
- -g specifies a group-id, used to specify compatibility with other programs
- -q enables quiet mode, suppress some of the extra output
- -f size-for-char-Io specifies the number of tests ran on per-char IO test, helps run program faster
- -b enables no write buffering, will call `fsync()` after every write
- -p used to synchronize multiple bonnie++ processes on a single execution
- -y s|p performs synchronization before each test
- -D uses direct IO for the bulk IO tests
- -z seed allows user to specify the random number seed, allows the same test to be repeated
- -Z random-file allows user to specify file containing random data in network byte order
- bon_csv2html outputs test results in html format
- bon_csv2txt outputs test results in txt format