**LMBench**

**Basics**

The tool measures the most commonly found performance bottlenecks in a wide range of system applications.

The benchmarks focus on **latency** and **bandwidth** because performance issues are usually caused by latency problems, bandwidth problems or some combination of the two.

**The tool only measures a system's ability to transfer data between processor, cache, memory, network and disc.**

**Sizing the benchmark**

1. All of the benchmarks that could be affected by the cache size are run in a loop with increasing sizes (usually power of 2) until some maximum size is reached. The results are then plotted to see where the benchmark no longer fits in the cache.
2. Verify that there is sufficient memory to run all of the benchmarks in the main memory. If any memory reference takes more than a few microseconds then the page is no longer in memory.
3. **Code that benchmarks clock speed and context switch times must be compiled without any optimization to produce correct results.** If possible use the gcc compiler always to run your benchmarking code
4. Don't enable any other optimization flags – This enables you to see results that would commonly be seen by most application writers.
5. Use shared library linking
6. **Do not pin a process to a particular processor** as this leads to better cache reuse and defeats the purpose of the Multi-processor scheduler

**Timing issue**

1. The tool measures elapsed time by reading the system clock via the gettimeofday interface
2. On some systems this interface has a resolution of 10 milliseconds
3. To compensate for the coarse clock resolution we execute an operation in a small loop which is sometimes unrolled if the operation is exceedingly fast and then we divide the loop time by the loop count
4. **If the benchmark expects the data to be in cache, the benchmark is typically run several times; only the last result is recorded**
5. If the benchmark does not want to measure cache performance it sets the size parameter larger than the cache
6. We try to defeat only the hardware cache not the file or the process page cache
7. **Run the benchmark as the only user of the system without other resource intensive or unpredictable processes or daemons**