Lecture 18 (mini-lecture): Performance Analysis
Announcements

• 3G: due Monday
• 3H: due “Friday”
• 4B: due “Friday”
• 4C: due “Friday”
• “Friday” → not late until June 11th
• And: all work must be turned by June 11th
• And: Project 3 grading may be simplified
Jose's post

4B Suppressed Leaks

According to this SO question, suppressed leaks come from outside the code. Is this acceptable to turn in?

---13274--- HEAP SUMMARY:
---13274--- in use at exit: 34,493 bytes in 424 blocks
---13274--- total heap usage: 552 allocs, 128 frees, 172,941,994 bytes allocated
---13274--- Searching for pointers to 424 not-freed blocks
---13274--- Checked 9,036,960 bytes
---13274--- LEAK SUMMARY:
---13274--- definitely lost: 0 bytes in 0 blocks
---13274--- indirectly lost: 0 bytes in 0 blocks
---13274--- possibly lost: 0 bytes in 0 blocks
---13274--- still reachable: 0 bytes in 0 blocks
---13274--- suppressed: 34,493 bytes in 424 blocks
---13274--- ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 18 from 18)
---13274---
---13274--- used_suppression: 1 OSX10:13-Leak /usr/local/lib/valgrind/default.supp:867 suppressed: 4,896 bytes in 1 blocks
---13274--- used_suppression: 2 OSX10:1-Leak /usr/local/lib/valgrind/default.supp:747 suppressed: 5,936 bytes in 45 blocks
---13274--- used_suppression: 37 OSX10:6-Leak /usr/local/lib/valgrind/default.supp:798 suppressed: 14,930 bytes in 261 blocks
---13274--- used_suppression: 15 OSX10:7-Leak /usr/local/lib/valgrind/default.supp:809 suppressed: 7,154 bytes in 75 blocks
---13274--- used_suppression: 16 OSX10:10-Leak /usr/local/lib/valgrind/default.supp:840 suppressed: 2,115 bytes in 34 blocks
---13274--- used_suppression: 1 OSX10:9-Leak /usr/local/lib/valgrind/default.supp:830 suppressed: 64 bytes in 1 blocks
---13274--- used_suppression: 6 OSX10:5-Leak /usr/local/lib/valgrind/default.supp:788 suppressed: 176 bytes in 6 blocks
---13274--- used_suppression: 1 OSX10:3-Leak /usr/local/lib/valgrind/default.supp:765 suppressed: 16 bytes in 1 blocks
---13274--- ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 18 from 18)

Edit:
This only happens on my personal computer. On ix the suppressed leaks don't occur.

Also, I was getting major errors on ix that I don't get on my own computer when compiling. For example, I got an error that memcpy wasn't declared in the scope and had to include another library for it to work. Why would this happen?
gettimeofday

GETTIMEOFDAY(2)  BSD System Calls Manual  GETTIMEOFDAY(2)

NAME
gettimeofday, settimeofday -- get/set date and time

SYNOPSIS
#include <sys/time.h>

int
gettimeofday(struct timeval *restrict tp, void *restrict tzp);

int
settimeofday(const struct timeval *tp, const struct timezone *tzp);

DESCRIPTION
The system's notion of the current Greenwich time and the current time zone is obtained with the gettimeofday() call, and set with the settimeofday() call. The time is expressed in seconds and microseconds since midnight (0 hour), January 1, 1970. The resolution of the system clock is hardware dependent, and the time may be updated continuously or in "ticks." If tp is NULL and tzp is non-NULL, gettimeofday() will populate the timezone struct in tzp. If tp is non-NULL and tzp is NULL, then only the timeval struct in tp is populated. If both tp and tzp are NULL, nothing is returned.

The structures pointed to by tp and tzp are defined in <sys/time.h> as:

struct timeval {
    time_t    tv_sec; /* seconds since Jan. 1, 1970 */
    suseconds_t tv_usec; /* and microseconds */
};

struct timezone {
    int      tz_minuteswest; /* of Greenwich */
    int      tz_dsttime; /* type of dst correction to apply */
};

The timeval structure specifies a time value in seconds and microseconds. The values in timeval are opaque types whose length may vary on different machines; depending on the architecture, this value is a 32-bit integer, a 64-bit integer, or a structure with both.

The timezone structure indicates that Daylight Saving time adjustment is in effect. The field tz_minuteswest indicates the number of minutes to be added to or subtracted from Greenwich, and a flag that, if nonzero, indicates that Daylight Saving time adjustment is in effect. The field tz_dsttime indicates the type of DST correction to apply.

Only the super-user may set the time and time zone. Only the super-user may call settimeofday() with a non-NULL timezone argument and a pointer to a timezone structure which contains a non-zero tz_dsttime field, and a non-zero tz_minuteswest field (other than 1 (see init(8)), the time may only be advanced. This limitation is imposed to prevent manipulation of system time by system administrators in order to alter time stamps on files. The system time can still be adjusted backwards using the regular system call chrt(8) and other ways that do not alter time stamps on files.

RETURN
A 0 return value indicates that the call succeeded. A -1 return value indicates an error occurred, and in this case an error code is stored into the global variable errno.

(there are lots of Unix system calls, which do lots of different things)
gettimeofday example

fawcett:330 child$ cat timings.C
#include <sys/time.h>
#include <stdio.h>

int main()
{
    int num_iterations = 100000000;
    int count = 0;
    struct timeval startTime;
    gettimeofday(&startTime, 0);
    gettimeofday(&endTime, 0);
    double seconds = double(endTime.tv_sec - startTime.tv_sec) +
                     double(endTime.tv_usec - startTime.tv_usec) / 1000000.;
    printf("done executing, took %f\n", seconds);
}
gettimeofday example

```c
fawcett:330 child$ cat timings.C
#include <sys/time.h>
#include <stdio.h>

int main()
{
    int num_iterations = 100000000;
    int count = 0;
    struct timeval startTime;
    gettimeofday(&startTime, NULL);
    gettimeofday(&endTime, NULL);
    double seconds = double(endTime.tv_sec - startTime.tv_sec) +
                     double(endTime.tv_usec - startTime.tv_usec) / 1000000.;
    printf("done executing, took %f\n", seconds);
}
```

```
fawcett:330 child$ g++ -O2 timings.C
fawcett:330 child$ ./a.out
done executing, took 0.000000
```
gettimeofday example

fawcett:330 child$ cat timings.C
#include <sys/time.h>
#include <stdio.h>

int main()
{
    int num_iterations = 100000000;
    int count = 0;
    struct timeval startTime;
    gettimeofday(&startTime, 0);
    for (int i = 0; i < num_iterations; i++)
        count += i;
    printf("Count was \%d\n", count);  /* NEW LINE OF CODE */
    struct timeval endTime;
    gettimeofday(&endTime, 0);
    double seconds = double(endTime.tv_sec - startTime.tv_sec) +
        double(endTime.tv_usec - startTime.tv_usec) / 1000000.;
    printf("done executing, took \%f\n", seconds);
}
gettimeofday example

fawcett:330 childs$ cat timings2.C
#include <sys/time.h>
#include <stdio.h>

int LoopFunction(int iteration, int &count)
{
    count += iteration;
}

int main()
{
    int num_iterations = 100000000;
    int count = 0;
    struct timeval startTime;
    gettimeofday(&startTime, 0);
    gettimeofday(&startTime, 0);
    for (int i = 0 ; i < num_iterations ; i++)
    {
        LoopFunction(i, count);
        /* No longer need this: printf("Count was %d\n", count); */
        struct timeval endTime;
        gettimeofday(&endTime, 0);
        double seconds = double(endTime.tv_sec - startTime.tv_sec) +
                        double(endTime.tv_usec - startTime.tv_usec) / 1000000.;
        printf("done executing, took %f\n", seconds);
    }
    printf("Done!");

    fawcett:330 childs$ g++ -O2 timings2.C
    fawcett:330 childs$ ./a.out
done executing, took 0.213101
More performance analysis

• gprof: old program ... I’m struggling to get it to work
• PAPI: library used widely to capture things like L1 cache misses, stalls, etc
• TAU: full performance analysis infrastructure – made right here at UO
Project 4C

• will involve gettimeofday on your Project 3 code
• still figuring what more to do (PAPI?)