DSA Assignment 5

Suraj S Yadwad
MTech AI 1^{st} year

December 5, 2019

SPHeap Simulation

Objective

To compare memory allocation performance of SPHeap Weighted Buddy System for uniformly and exponentially distributed sequence of requests.

Setup

All memory that needs to be deallocated at current time is freed.

Memory requests are generated using either Uniform distribution between 10000 and 200000 bytes or Exponential distribution with mean 100000 bytes, minimum 10000 bytes and truncated at 200000 bytes.

Time(T) is generated from Uniform distribution between 0 and 10000. A time-stamp is given to each memory that is allocated as (Current Time + T)

The process is repeated for 100000 iterations.

Performance

```
time = 0
nDAll   IntFrag         Nsplits Recomb
   0     15.895145       6       0
time = 1000
nDAll   IntFrag         Nsplits Recomb
  42     19.099515       1448    24
request cannot be fulfilled
requested = 140114 bytes; External fragmentation = 16.787720
time = 1976
nDAll   IntFrag         Nsplits Recomb
 170     19.153918       2765    77
```

Figure 1: Requests from Uniform distribution

```
time = 0
nDAll   IntFrag         Nsplits Recomb
   0     1.671881        6       0
time = 1000
nDAll   IntFrag         Nsplits Recomb
  46     23.085856       1253    17
time = 2000
nDAll   IntFrag         Nsplits Recomb
 207     23.206902       2498    69
request cannot be fulfilled
requested = 200000 bytes; External fragmentation = 9.030151
time = 2379
nDAll   IntFrag         Nsplits Recomb
 296     23.301532       2940    97
```

Figure 2: Requests from Exponential distribution

```
time = 9000
nDAll   IntFrag         Nsplits Recomb
8494     19.019730       9496    8458
time = 10000
nDAll   IntFrag         Nsplits Recomb
9494     19.009068       10481   9491
```

Figure 3: Requests from Uniform distribution, reduced max time stamp to 1000

```
time = 9000
nDAll   IntFrag         Nsplits Recomb
8497     23.664443       8381    7435
time = 10000
nDAll   IntFrag         Nsplits Recomb
9495     23.695103       9281    8313
```

Figure 4: Requests from Exponential distribution, reduced max time stamp to 1000

Comments and notes

Dist	Int Frag	Ext Frag
Uniform	19%	23%
Exponential	23%	9%

- The number of splits and recombinations is higher for uniform distribution

Testing on Polynomial Arithmetic

Objective

To use spHeap, oneBin and malloc on the Polynomial division program from Assignment 2 and comparing the execution times of the programs.

Input

Dividend: $x^{1000} - 1$
Divisor: $x - 1$

Performance

```
Command being timed: "./spHeapPoly.out"
User time (seconds): 5.45
System time (seconds): 0.00
Percent of CPU this job got: 15%
Elapsed (wall clock) time (h:mm:ss or m:ss): 0:34.19
Average shared text size (kbytes): 0
Average unshared data size (kbytes): 0
Average stack size (kbytes): 0
Average total size (kbytes): 0
Maximum resident set size (kbytes): 4380
Average resident set size (kbytes): 0
Major (requiring I/O) page faults: 0
Minor (reclaiming a frame) page faults: 751
Voluntary context switches: 5
Involuntary context switches: 33
Swaps: 0
File system inputs: 0
File system outputs: 0
Socket messages sent: 0
Socket messages received: 0
Signals delivered: 0
Page size (bytes): 4096
Exit status: 0
```

Figure 5: Using SPHeap

```
Command being timed: "./oneBinPoly.out"
User time (seconds): 0.71
System time (seconds): 0.00
Percent of CPU this job got: 2%
Elapsed (wall clock) time (h:mm:ss or m:ss): 0:27.42
Average shared text size (kbytes): 0
Average unshared data size (kbytes): 0
Average stack size (kbytes): 0
Average total size (kbytes): 0
Maximum resident set size (kbytes): 3056
Average resident set size (kbytes): 0
Major (requiring I/O) page faults: 0
Minor (reclaiming a frame) page faults: 413
Voluntary context switches: 5
Involuntary context switches: 1
Swaps: 0
File system inputs: 0
File system outputs: 0
Socket messages sent: 0
Socket messages received: 0
Signals delivered: 0
Page size (bytes): 4096
Exit status: 0
```

Figure 6: Using oneBin

```
User time (seconds): 0.21
System time (seconds): 0.00
Percent of CPU this job got: 1%
Elapsed (wall clock) time (h:mm:ss or m:ss): 0:18.46
Average shared text size (kbytes): 0
Average unshared data size (kbytes): 0
Average stack size (kbytes): 0
Average total size (kbytes): 0
Maximum resident set size (kbytes): 1964
Average resident set size (kbytes): 0
Major (requiring I/O) page faults: 0
Minor (reclaiming a frame) page faults: 139
Voluntary context switches: 4
Involuntary context switches: 1
Swaps: 0
File system inputs: 0
File system outputs: 0
Socket messages sent: 0
Socket messages received: 0
Signals delivered: 0
Page size (bytes): 4096
Exit status: 0
```

Figure 7: Using malloc

Comments and notes

- oneBin is significantly faster than SPHeap.
- Since oneBin is specialized for Polynomial arithmetic and there is no overhead of matching the buddy and coalescing compared to SPHeap, the execution time is low for oneBin.