Controlling Memory Fragmentation and Higher Order Allocation Failure: Analysis, Observations and Results

Pintu Kumar

pintu.k@samsung.com

CONTENT

- Introduction
- Measuring the memory fragmentation level
- Memory Fragmentation Analysis
- Observations
- Experimentation Results
- Summary
- Some References

INTRODUCTION

- **What is Memory Fragmentation ?**

When a Linux device has been running continuously over a time without reboot and keeps allocating and de-allocating pages, the pages become fragmented. The bigger contiguous free pages become zero and free pages are only available in many smaller pages which are not contiguous. Thus even if we have lots of free memory in smaller units, the page allocation in kernel may fail.

This typical problem is called "External Memory Fragmentation" here after referred to as memory fragmentation.

INTRODUCTION

- **Effect of Memory Fragmentation :**
  - **Memory Fragmentation can cause a system to lose its ability to launch new process.**
  - **Memory fragmentation becomes more of an issue in embedded devices and Linux mobiles.**
    - **DRAM + Flash , Swapless system**
  - **Memory fragmentation can become more critical with high multimedia and graphics activities which requires contiguous higher-order pages.**

MEASURING FRAGMENTATION LEVEL

- **It is important to measure fragmentation level across each zones and for each higher-order allocation in kernel.**
- **We believe by measuring fragmentation level during page allocation we can control higher-order allocation failure in kernel.**
- **We developed kernel utility to measure fragmentation level during runtime without enabling memory COMPACTION.**

MEASURING FRAGMENTATION LEVEL

- **Formula to measure fragmentation level in percentage :**

$$FragLevel(\%) = \frac{TotalFreePages - \sum_{i=j}^{N} (2^i . k_i)}{TotalFreePages} X100$$

TotalFreePages	**=**	**Total number of free pages in each Node**
N	**=**	**MAX_ORDER - 1 ➔ The highest order of allocation**
j	**=**	**the desired order requested**
i	**=**	**page order ➔ 0 to N**
Ki	**=**	**Number of free pages in ith order block**

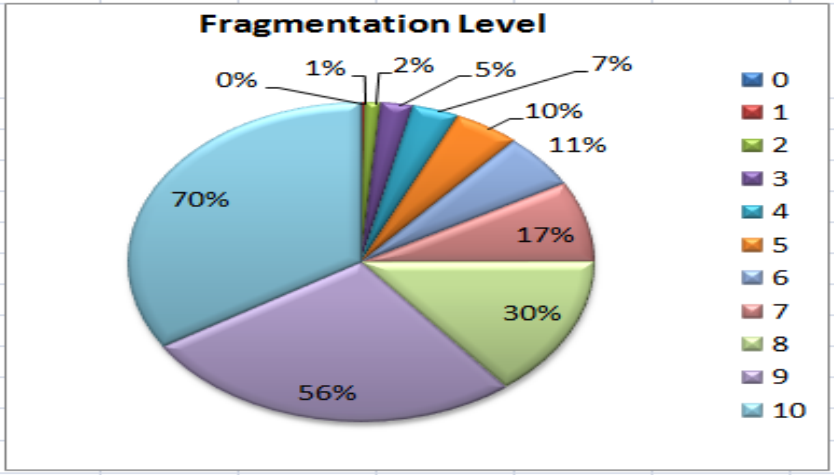
(The above formula derived from Mel Gorman's paper : "***Measuring the Impact of the Linux Memory Manager***")

MEASURING FRAGMENTATION LEVEL

- **SAMPLE OUTPUT : cat /proc/fraglevelinfo**

Fragmentation Level Measurement (cat /proc/fraglevelinfo)

Page Order (i)	Page Block (k)	Free Pages	Movable Pages	Reclaimable Pages	Fragmentation Level [%]
0	1	38	36	1	0%
1	2	59	55	2	1%
2	4	40	32	1	2%
3	8	19	16	0	5%
4	16	12	10	1	7%
5	32	4	2	1	10%
6	64	6	4	1	11%
7	128	7	4	0	17%
8	256	7	0	1	30%
9	512	2	1	0	56%
10	1024	2	2	0	70%
TOTAL		6932	3954	377	19%



$$FragLevel(\%) = \frac{TotalFreePages - \sum_{i=0}^{N}(2^i k_i)}{TotalFreePages} X100$$

Example:

Lets say a NORMAL zone looks like this at some point of time

	2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}
Node 0, zone Normal	3	20	104	13	13	1	1	1	0	0	0

Now lets apply our formula to measure fragmentation level for order 2^5.

Here , TotalFreePages = (3x1 + 20x2 + 104x4 + 13x8 + 13x16 + 1x32 + 1x64 + 1x127) = 994

Therefore;
% Fragmentation =
(994 - [(2^5)*1 + (2^6)*1 + (2^7)*1 + (2^8)*0 + (2^9)*0 + (2^10)*0]) * 100) / 994

% Fragmentation = (994 - [32 + 64 + 128]) / 994 = ((994 – 224) * 100) / 994

%Fragmentation = (770 * 100) / 994 = 77.46 % ➔ 77 % (round off)

MEMORY FRAGMENTATION ANALYSIS

- **We developed a sample kernel module and test utility to perform higher-order allocation and doing memory fragmentation analysis before and after the allocation.**
- **Test utility developed and tested for Samsung Linux Platform (kernel2.6.32 and kernel2.6.36)**
- **Test Utility can be shared on Linux Test Project (LTP) after further improvements.**

JUST AFTER PHONE BOOT-UP (Samsung Linux Kernel 2.6.36)

```
/opt/home/root # cat /proc/fraglevelinfo
Node:0, Zone:Normal
Order    FreePages  MovablePages  ReclaimablePages  Fragmentation[%]
   0            2             0                 1               0%
   1            2             1                 1               0%
   2            5             1                 2               0%
   3            3             1                 2               0%
   4            4             1                 2               0%
   5            2             1                 1               0%
   6            6             1                 0               0%
   7            5             1                 1               0%
   8            1             0                 0               1%
   9            1             0                 1               1%
  10          102           101                 0               1%
TotalFreePages: 106414
TotalMovablePages: 103678
TotalReclaimablePages: 731
Overall Fragmentation: 0%

Node:0, Zone:HighMem
Order    FreePages  MovablePages  ReclaimablePages  Fragmentation[%]
   0           15             5                 0               0%
   1           15             2                 0               0%
   2           11             2                 0               0%
   3           10             5                 0               0%
   4           10             6                 0               0%
   5           11             6                 0               0%
   6            4             2                 0               1%
   7            8             5                 0               1%
   8            1             1                 0               3%
   9            1             0                 0               3%
  10           55            54                 0               4%
TotalFreePages: 59049
TotalMovablePages: 56665
TotalReclaimablePages: 0
Overall Fragmentation: 1%
```

```
/opt/home/root # cat /proc/buddyinfo
Node 0, zone   Normal      2      2      5      3      4      2      6      5      1      1    102
Node 0, zone  HighMem     15     15     11     10     10     11      4      8      1      1     55
```

AFTER RUNNING VARIOUS APPLICATIONS (Browser, WiFi Video Share, Camera, Voice Recorder, eBooks, Few Games) for ½ an Hour and then killing All)

```
/opt/pintu # cat /proc/fraglevelinfo
Node:0, Zone:Normal
Order    FreePages  MovablePages  ReclaimablePages  Fragmentation[%]
   0        1139           800                 1               0%
   1         795           540                 7               1%
   2         500           345                 2               3%
   3         315           229                 1               5%
   4         237           168                 0               8%
   5         114            70                 1              13%
   6          67            26                 1              17%
   7          25            15                 1              23%
   8          35            10                 1              26%
   9          16             9                 0              37%
  10          42            36                 0              47%
TotalFreePages: 82333
TotalMovablePages: 57636
TotalReclaimablePages: 511
Overall Fragmentation: 16%

Node:0, Zone:HighMem
Order    FreePages  MovablePages  ReclaimablePages  Fragmentation[%]
   0        2281          1994                 0               0%
   1        1382          1140                 0               9%
   2         861           781                 0              21%
   3         474           430                 0              36%
   4         226           193                 0              52%
   5         106            85                 0              67%
   6          38            26                 0              82%
   7           8             6                 0              92%
   8           3             0                 0              96%
   9           0             0                 0             100%
  10           0             0                 0             100%
TotalFreePages: 23513
TotalMovablePages: 19078
TotalReclaimablePages: 0
Overall Fragmentation: 59%
```

```
/opt/pintu # cat /proc/buddyinfo
Node 0, zone   Normal   1139    795    500    315    237    114     67     25     35     16     42
Node 0, zone  HighMem   2250   1382    861    474    226    106     38      8      3      0      0
```

ALLOCATION SUCCESS CASE - (Samsung Kernel 2.6.32)

```
/opt/pintu # ls -l /dev/pinchar
crw-------    1 root     root       10,  49 Jan 12 13:21 /dev/pinchar
```

```
/opt/pintu # ./app_pinchar.bin
```

	2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}
Node 0, zone DMA	3	1	1	1	0	0	0	1	0	0	0
Node 1, zone DMA	197	129	0	0	0	0	0	0	0	0	0
Node 2, zone DMA	30	12	14	4	5	8	6	4	2	1	27

Enter the page order(in power of 2) :	16	**2^4 order block**	**16 x 4K = 64K bytes**
Enter the number of such block :	5		

State After The Allocation Request Is Successful

	2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}
Node 0, zone DMA	3	1	1	1	0	0	0	1	0	0	0
Node 1, zone DMA	169	143	1	0	0	0	0	0	0	0	0
Node 2, zone DMA	19	12	12	5	2	7	6	4	2	1	27

Explanation :

5 of 2^4 order blocks were requested. These request could only be satisfied by Node 2, Thus Node 2 were selected for allocation. But in Node 2 also, out of 5 (2^4) blocks only 3 could be allocated. Then the other 2 were allocated by splitting the 2^5 order blocks.

5 x 2^4 = [3 x 2^4 + (1 x 2^5)] = [3 x 2^4 + (1 x 2 x 2^4)] = [5 x 2^4]

ALLOCATION FAILURE CASE - (Samsung Kernel 2.6.32)

Buddy State Before Allocation

		2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}
Node 0, zone	DMA	1	0	0	0	1	0	2	0	0	0	0
Node 1, zone	DMA	247	181	15	1	1	2	0	9	10	8	11
Node 2, zone	DMA	19	19	11	3	6	3	4	2	2	2	29

Enter the page order(in power of 2) :	1024	**2^10 order block**	**1024 x 4K = 4096K bytes**
Enter the number of such block :	50	**(this is the highest order)**	

ERROR : ioctl - PINCHAR_ALLOC - Failed, after block num = 48 !!!

Explanation : As you can see the allocation request of 1024 x 50 pages is failed after 47 such allocation. But still there were enough free pages available in lower order.

Buddy State After Allocation FAILED And Other Allocation Freed

		2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}
Node 0, zone	DMA	1	0	18	9	0	0	0	0	0	0	0
Node 1, zone	DMA	88	77	36	25	25	43	23	19	18	11	20
Node 2, zone	DMA	18	14	10	4	5	2	3	3	2	2	29

Explanation : The interesting think to note here is that after requested allocation was failed, kernel tried to arrange that many blocks in desired order so that next similar request can be succeeded.

Observations

- __alloc_pages_nodemask : This is the heart of all allocation in kernel.
- We measure fragmentation level for each higher order here.
- Track higher-order allocation during high fragmentation. Anything above PAGE_ALLOC_COSTLY_ORDER(==3) is considered higher-order allocation and becomes critical.

Observations

- **Direct reclaim does some progress but still could not return any pages during first run.**
- **Similarly direct compact (from kernel2.6.36 onwards) is helpful but still not effective for very higher-order allocation.**
- **May be the other way round (first direct_reclaim then direct_compact) could be more helpful.**
- **A small amount of delay(for GFP_KERNEL) is required after direct_{reclaim,compact} and before retry. Maybe due to lazy buddy allocator.**

Experiments Results

- **We performed some experiments with higher-order allocation and got some results.**
- **We found that whenever we run any application "Xorg" perform 4 or 8 order allocation.**
- **The browser always requires order-4 allocation.**

/opt/pintu # ps ax | grep browser
7159 ? Ssl 0:03 /opt/apps/com.samsung.browser/bin/browser

[3830.215613] [HIGHERORDER_DEBUG] : __alloc_pages_nodemask is called by process <PID = 1168, NAME = Xorg> !!!
[3830.227243] [HIGHERORDER_DEBUG] : ZONE : Normal, NODE : 0, ORDER = 8, Fragmentation Level = 29%
[3830.235645] [HIGHERORDER_DEBUG] : __alloc_pages_nodemask is called by process <PID = 1168, NAME = Xorg> !!!
[3830.244575] [HIGHERORDER_DEBUG] : ZONE : Normal, NODE : 0, ORDER = 4, Fragmentation Level = 13%
(Around 10 times)

[3831.355884] [HIGHERORDER_DEBUG] : __alloc_pages_nodemask is called by process <PID = 7159, NAME = browser> !!!
[3831.364649] [HIGHERORDER_DEBUG] : ZONE : Normal, NODE : 0, ORDER = 4, Fragmentation Level = 13%
[3831.373484] [HIGHERORDER_DEBUG] : __alloc_pages_nodemask is called by process <PID = 7159, NAME = browser> !!!
[3831.383134] [HIGHERORDER_DEBUG] : ZONE : Normal, NODE : 0, ORDER = 4, Fragmentation Level = 13%
(Around 26 times)

RESULT #1 - (Samsung Kernel 2.6.32)

	2^0	2^1	2^2	2^3	2^4	2^5	2^6	2^7	2^8	2^9	2^{10}
Node 0, zone DMA	685	104	1	1	0	0	0	1	0	0	0
Node 1, zone DMA	33	19	31	18	9	1	1	0	0	0	0
Node 2, zone DMA	11	50	9	5	5	3	2	1	0	0	0

Enter the page order(in power of 2) :	1024	**2^10 order block**	**1024 x 4K = 4096K bytes**
Enter the number of such block :	10	**(this is the highest order)**	

[24768.550017] [HIGHERORDER_DEBUG] : __alloc_pages_nodemask is called by process <PID = 2289, NAME = app_pinchar.bin> !!!
[24768.559578] [HIGHERORDER_DEBUG] : ZONE : DMA, NODE : 0, ORDER = 10, Fragmentation Level = 100%
[24768.568020] [HIGHERORDER_DEBUG] : __alloc_pages_nodemask is called by process <PID = 2289, NAME = app_pinchar.bin> !!!
[24768.578686] [HIGHERORDER_DEBUG] : ZONE : DMA, NODE : 1, ORDER = 10, Fragmentation Level = 100%
[24768.587251] [HIGHERORDER_DEBUG] : __alloc_pages_nodemask is called by process <PID = 2289, NAME = app_pinchar.bin> !!!
[24768.597919] [HIGHERORDER_DEBUG] : ZONE : DMA, NODE : 2, ORDER = 10, Fragmentation Level = 100%
[24768.606486] [HIGHERORDER_DEBUG] : __alloc_pages_nodemask : Allocation going via - slowpath !!!
[24768.615141] app_pinchar.bin: page allocation failure. order:10, mode:0x4020
← ---------- Wait for 2 seconds and retry allocation ---------- →
[24770.669441] [HIGHERORDER_DEBUG] : Trying - Final time !!!!!!!!!!
[24770.686688] <PINCHAR> : PINCHAR_ALLOCATE - Success(index = 0) !

Explanation : As you can see here, due to 100% fragmentation, page allocation request was failing, even after direct reclaim (slow path). But after a delay and retrying allocation request again, all subsequent allocation were successful.
This delay indicates something needs to be done after direct reclaim. Maybe wait till lazy buddy allocator arranges free pages in the subsequent free areas.

RESULT #2 - (Samsung Kernel 2.6.36) [Without COMPACTION]

Initial Fragmentation Level

```
/opt/pintu # cat /proc/fraglevelinfo ; cat /proc/buddyinfo
Node:0, Zone:Normal
Order     FreePages        MovablePages     ReclaimablePages         Fragmentation[%]
  0         271                25                0                        0%
  1         171                 5                0                        2%
  2         143                 1                1                        4%
  3         100                 1               18                        9%
  4         154                 1               61                       15%
  5          79                 2               27                       33%
  6          26                 1                6                       53%
  7           7                 3                3                       65%
  8          10                 0                0                       72%
  9           0                 0                0                       92%
 10           1                 0                0                       92%
TotalFreePages: 13117
TotalMovablePages: 575
TotalReclaimablePages: 2756
Overall Fragmentation: 39%

Node:0, Zone:HighMem
Order     FreePages        MovablePages     ReclaimablePages         Fragmentation[%]
  0        6850              6837                0                        0%
  1        8826              8803                0                       26%
  2         261               255                0                       95%
  3           9                 1                0                       99%
  4           5                 5                0                       99%
  5           0                 0                0                      100%
  6           0                 0                0                      100%
  7           0                 0                0                      100%
  8           0                 0                0                      100%
  9           0                 0                0                      100%
 10           0                 0                0                      100%
TotalFreePages: 25698
TotalMovablePages: 25551
TotalReclaimablePages: 0
Overall Fragmentation: 83%

Node 0, zone   Normal    271    171    143    100    154     79     26      7     10      0      1
Node 0, zone  HighMem   6850   8826    261      9      5      0      0      0      0      0      0
```

After Higher order allocation request

./app_pinchar.bin 1024 25

[17949.789934] [HIGHERORDER_DEBUG] : __alloc_pages_nodemask is called by process <PID = 27713, NAME = app_pinchar.bin> !!!
[17949.801633] [HIGHERORDER_DEBUG] : ZONE : Normal, NODE : 0, ORDER = 10, Fragmentation Level = 92%
[17949.811073] [HIGHERORDER_DEBUG] : __alloc_pages_nodemask : Allocation going via - slowpath !!!
[17949.831793] [HIGHERORDER_DEBUG] : did_some_progress = 151
[17949.844090] [HIGHERORDER_DEBUG] : NO pages.....even after direct reclaim
[17949.859104] app_pinchar.bin: page allocation failure. order:10, mode:0x40d0
←---------- Wait for 2 seconds and retry allocation ----------→
[17951.879156] [HIGHERORDER_DEBUG] : Trying - Final time !!!!!!!!!!
[17951.893248] <PINCHAR> : PINCHAR_ALLOCATE - Success(index = 0)
[17960.189583] [HIGHERORDER_DEBUG] : __alloc_pages_nodemask is called by process <PID = 27713, NAME = app_pinchar.bin> !!!
[17960.201128] [HIGHERORDER_DEBUG] : ZONE : Normal, NODE : 0, ORDER = 10, Fragmentation Level = 98%
[17960.210269] [HIGHERORDER_DEBUG] : __alloc_pages_nodemask : Allocation going via - slowpath !!!
[17960.335044] [HIGHERORDER_DEBUG] : did_some_progress = 887
[17960.339918] [HIGHERORDER_DEBUG] : Got some pages after direct reclaim .....
[17960.368939] <PINCHAR> : PINCHAR_ALLOCATE - Success(index = 4) !
[17964.518845] [HIGHERORDER_DEBUG] : __alloc_pages_nodemask is called by process <PID = 27713, NAME = app_pinchar.bin> !!!
[17964.530629] [HIGHERORDER_DEBUG] : ZONE : Normal, NODE : 0, ORDER = 10, Fragmentation Level = 83%
[17964.547138] <PINCHAR> : PINCHAR_ALLOCATE - Success(index = 8) !
[17965.552976] [HIGHERORDER_DEBUG] : __alloc_pages_nodemask is called by process <PID = 27713, NAME = app_pinchar.bin> !!!
[17965.564319] [HIGHERORDER_DEBUG] : ZONE : Normal, NODE : 0, ORDER = 10, Fragmentation Level = 84%
[17965.580823] <PINCHAR> : PINCHAR_ALLOCATE - Success(index = 9) !
[17966.586440] [HIGHERORDER_DEBUG] : __alloc_pages_nodemask is called by process <PID = 27713, NAME = app_pinchar.bin> !!!
[17966.597175] [HIGHERORDER_DEBUG] : ZONE : Normal, NODE : 0, ORDER = 10, Fragmentation Level = 85%
[17966.613424] <PINCHAR> : PINCHAR_ALLOCATE - Success(index = 10) !

Allocation failed directly during the first attempt itself even after direct reclaim. But after introducing a delay and retrying, all further allocation succeeded. May be Kswapd takes sometime to clear up dirty pages and buddy adding it back to free area.

Final Fragmentation Level

```
/opt/pintu # cat /proc/fraglevelinfo ; cat /proc/buddyinfo
Node:0, Zone:Normal
Order     FreePages         MovablePages     ReclaimablePages         Fragmentation[%]
  0         864                465               135                      0%
  1         863                583               118                      0%
  2         868                626               107                      2%
  3         659                493                74                      5%
  4         566                434                41                     10%
  5         358                280                28                     18%
  6         222                184                16                     28%
  7          90                 83                 6                     41%
  8          78                 67                 0                     51%
  9          13                 13                 0                     69%
 10          26                  0                 0                     75%
TotalFreePages: 110818
TotalMovablePages: 70191
TotalReclaimablePages: 4735
Overall Fragmentation: 27%

Node:0, Zone:HighMem
Order     FreePages         MovablePages     ReclaimablePages         Fragmentation[%]
  0        4579               4523                0                      0%
  1        8239               8202                0                      7%
  2        4789               4756                0                     34%
  3        1760               1734                0                     65%
  4         362                355                0                     88%
  5          35                 32                0                     97%
  6           4                  4                0                     99%
  7           0                  0                0                    100%
  8           0                  0                0                    100%
  9           0                  0                0                    100%
 10           0                  0                0                    100%
TotalFreePages: 61461
TotalMovablePages: 60783
TotalReclaimablePages: 0
Overall Fragmentation: 71%

Node 0, zone    Normal    864    862    868    659    566    358    222     90     78     13     26
Node 0, zone   HighMem   4579   8239   4789   1760    362     35      4      0      0      0      0
```

Here you can see lots of movable pages after lots of direct reclaim. Thus direct compact might be helpful after direct reclaim and not before.

EXPERIMENTATION DATA

Page Order	Block Used	Available Blocks	No of Blocks Requested	Current Fragmentation Level	No of Blocks Allocated	Pass Rate
10	1024	0	20	100%	20	100%
9	512	11	20	94%	20	100%
8	256	4	20	90%	20	100%
8	256	0	50	100%	50	100%
9	512	1	30	97%	30	100%
10	1024	28	40	10%	40	100%
10	1024	0	50	100%	46	92%

DATA COLLECTED ON :

Samsung Mobile Target With Kernel 2.6.32

SUMMARY

- **Measuring fragmentation level and tracking higher-order is important at least for low memory notifier.**
- **It was observed that allocation takes slowpath whenever fragmentation level is above 90%.**
- **The delay introduced here is only for experimental purpose.**
  - **Delay could be because, dirty pages has to be written to the disk before it is marked freed.**
  - **May be the real thing could be to wait till lazy buddy allocator rearranges the free pages.**
  - **This is valid only for GFP_KERNEL where a sleep is allowed.**

- **For fragmentation > 90%, introduce temporary kernel thread to do direct reclaim/compact in background.**
  - **Buy the time you come back for next request, pages will be ready for you.**
  - **Not enough data to share. Further experiments in progress.**
- **From kernel2.6.35 COMPACTION contains its own fragmentation level measurement.**
  - **/sys/kernel/debug/extfrag/unusable_index**
  - **But this requires COMPACTION and HUGETLB to be enabled.**
  - **May be we can utilize this from kernel2.6.35 onwards.**
  - **Difficult to back port compaction to lower kernel version.**
  - **Mostly helpful for large system and may not be useful for small embedded products.**

- **Can we introduce something like system wide fragmentation level ???**
- **Reboot is not a good choice for end users even for small system.**
- **May be introduce something like “Reset Physical memory state”.**
  - **Bring back memory to original state without reboot.**
  - **Not enough data.**
  - **May be develop system utility to shrink physical memory using “shrink_all_memory” used during snapshot image creation in Hibernation.**

- **Reserving memory during boot time can reduce fragmentation to some extent.**
  - Good only if you have bigger RAM.
  - Tracking higher-order fragmentation level can help decide which memory to reserve in future.
  - May be something like dynamic reserving based on past performance could be better.
- **Contiguous Memory Allocator (CMA) and Virtual Contiguous Memory (VCM) can help fight fragmentation.**
  - CMA : same like reserving memory during boot but transparently allows the memory to be reused and latter migrate pages to create similar chunk.
  - Can be used for frame buffers and other memory hungry multimedia devices.

- **But CMA requires pages to be movable and may now use compaction, again not guaranteed because most kernel pages are not movable.**
  - **May be we have to limit the reuse of CMA region.**
  - **Or share CMA region only for very high order.**
- **Problem easily reproducible in DRAM + Flash swapless embedded system without HighMem.**
- **Further combination of investigation is in progress to derive a solution which does not requires reboot.**

Some References

1. **Wikipedia, "*Buddy Memory Allocation*".** http://en.wikipedia.org/wiki/Buddy_memory_allocation.
2. **Jonathan Corbet. (2010), "*Memory Compaction*"** http://lwn.net/Articles/368869/
3. **Lifting The Earth (2011) "*Linux Page Allocation Failure*",** http://www.linuxsmiths.com/blog/.
4. **Mark S. Johnstone and Paul R. Wilson (1997), "*The Memory Fragmentation Problem – Solved?*"**
5. **Mel Gorman and Patrick Healy (2005) "*Measuring the Impact of the Linux Memory Manager*"** http://thomas.enix.org/pub/rmll2005/rmll2005-gorman.pdf
6. **Corbet (2004), "*Kswapd and higher-order allocations*"** http://lwn.net/Articles/101230/

Thank You !!!