Erlang Solutions Ltd.

Memory Management Battle Stories

Agenda

- The Problem
- Concepts
- Statistics
- Case 1: Large binaries
- Case 2: Fragmentation
- New Features

The Problem

- Normal OS default allocator:
  - is relatively slow for many small allocations
  - uses same allocation strategy for all data, increased fragmentation
  - no cross platform fine-grained statistics
- Try it on your system, +Mea min
  - Disables erts allocators and uses malloc directly for everything
- With multi-core memory management is even more important (and even more difficult)

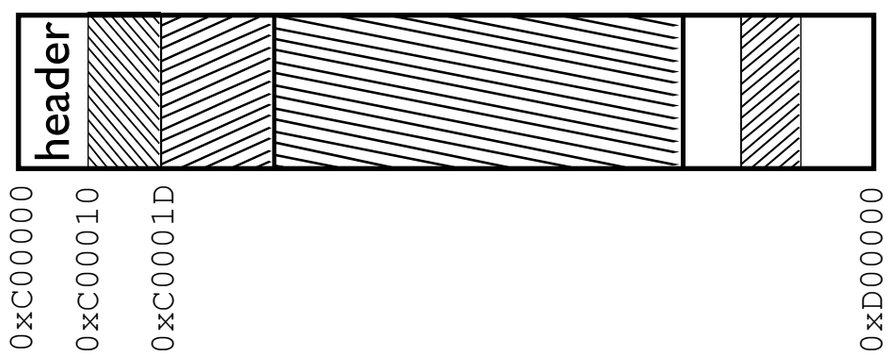
Concepts

- Carriers and Blocks
- Single- vs multi-block carriers
- Multiblock allocators
- Thread specific allocators

Carriers and Blocks

- Block – A piece of memory requested by the VM
- Carrier – A piece of memory that contains one or more blocks

```
ets:insert(Tid,{0,”HELLO”}).
ets:insert(Tid,{1,[0,1,2,....,63]}).
ets:insert(Tid,{2,[0,1,2,...,128]).
ets:delete(Tid,1).
```



Single- vs Multi-block Carriers

- Large blocks are placed in a singleblock carrier (sbc)
  - What is a large block? depends...
    - Control with +M<S>sbct (singleblock carrier threshold)
      - default is 512 kb
- Normally you want most of your data in multiblock carriers (mbc)
  - If you increase sbct you probably want to increase smbcs and lmbcs by an equal %
  - Size of carrier is controlled with +M<S>smbcs, +M<S>lmbcs and +M<S>mbcgs

Allocator types

- Different strategies possible for different types of data
  - eheap, binary, driver, ets
  - temporary, short lived, standard lived, long lived
  - fix size

Allocator types

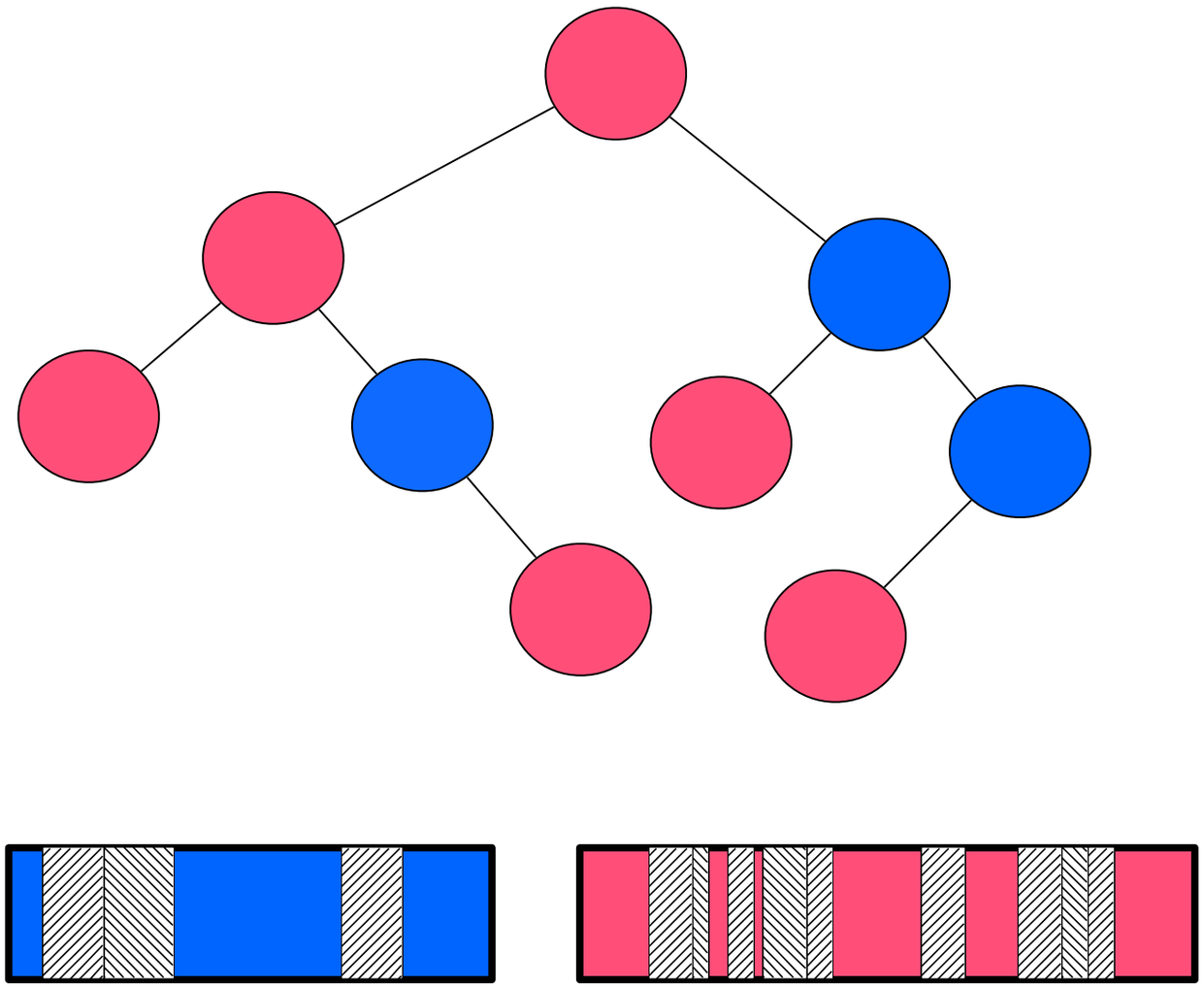
- temporary
  - C function scope
  - temp gc rootset
  - dist msg decode
- short
  - ets match specs
  - short timers
  - fd select list
- standard
  - links
  - monitors
- long
  - code
  - atoms
- fixed
  - process control block
  - port control block

erl_alloc.types

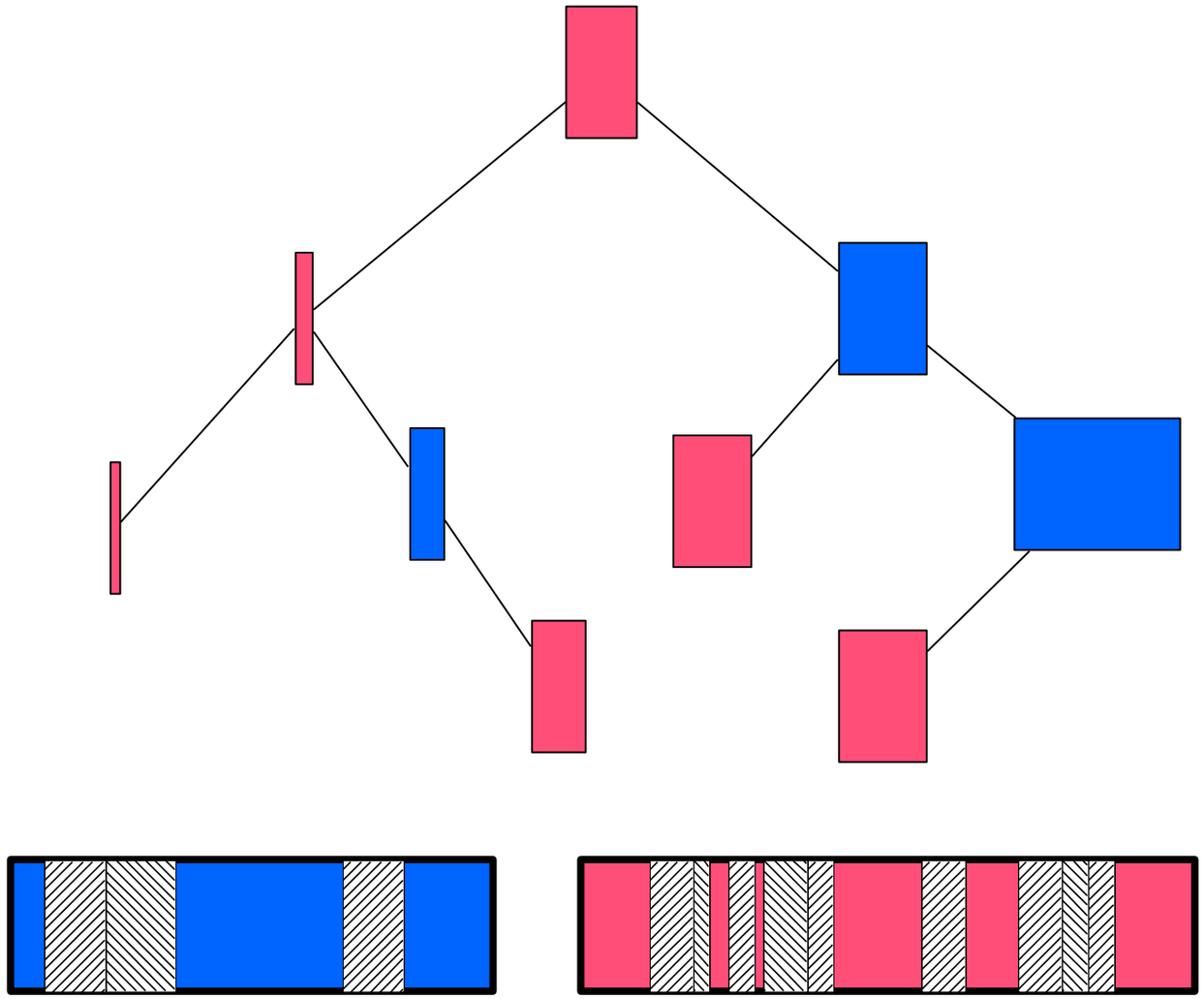
Multiblock allocator strategies

- Block oriented
  - best fit
  - address order best fit
  - address order first fit
  - good fit
  - a fit
- Carrier oriented
  - address order first fit carrier best fit
  - address order first fit carrier address order best fit

Best fit example



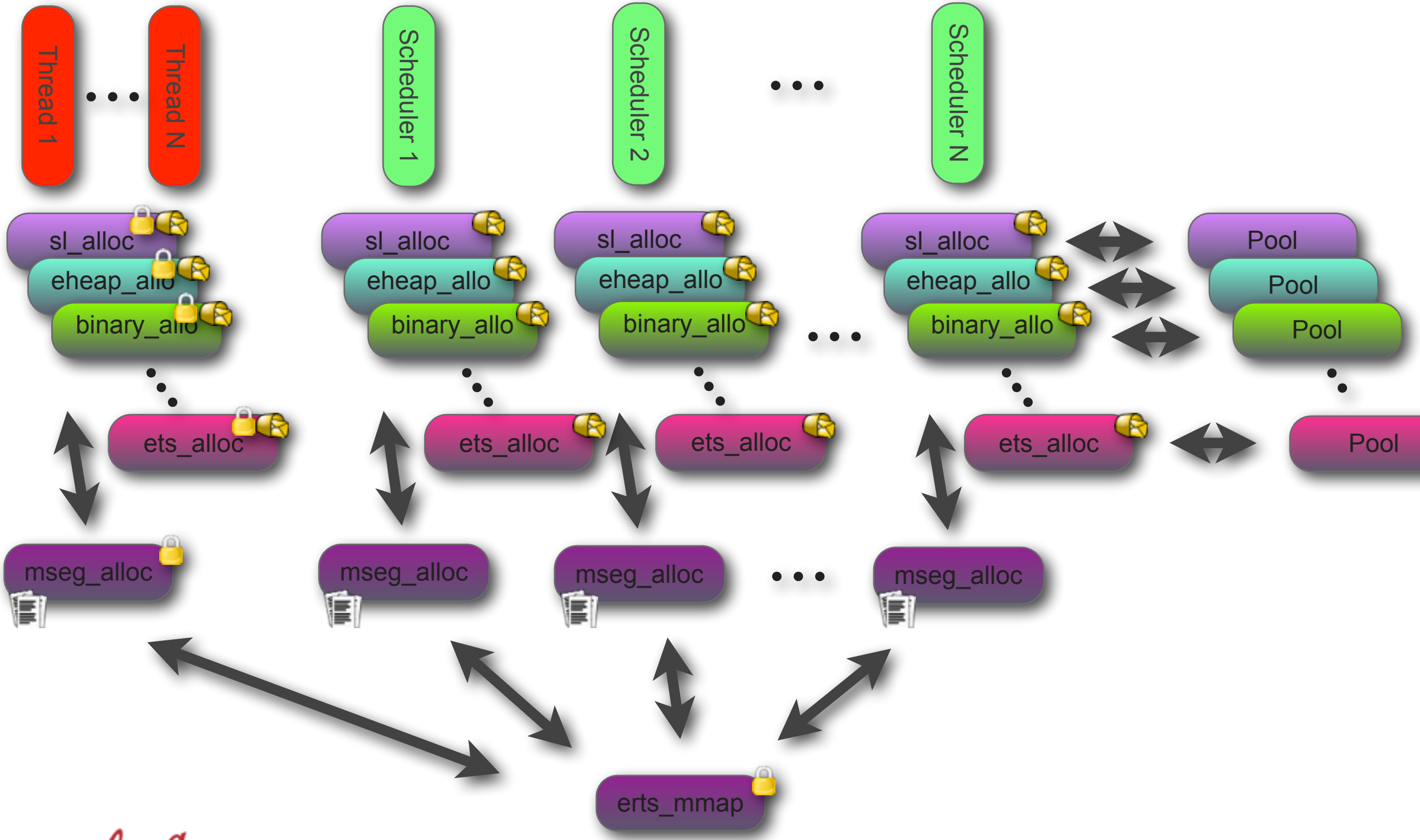
Best fit example



Carrier Allocators

- mseg alloc
  - uses /dev/zero and mmap,munmap,mremap
  - caches freed carriers
- sys alloc
  - maps to malloc,free (or posix_memalign/free)
  - carriers are a multiple of +Muycs to help avoid fragmentation
  - used for main carrier allocation +M<S>mmbcs

Memory architecture



Statistics: erlang:system_info(allocator)

- Allocator types
  - sys_alloc mseg_alloc
  - eheap_alloc ets_alloc binary_alloc driver_alloc
  - temp_alloc sl_alloc std_alloc ll_alloc fix_alloc
- Settings
  - sbc threashold, mbc allocation strategy, etc
- Features
  - aligned alloc, lock physical, etc

Statistics: erlang:system_info({allocator,Type})

```
[{instance,0,
    [{versions, ... },
     {options, ... },
     {mbcs, ... },
     {sbcs, ... },
     {calls, ... }]},
{instance,1, ... },
{instance,S+1, ... }]
```

Statistics: mbcs / sbcs

```
[{blocks,1066675,1068988,1811013},
 {blocks_size,
860267920,862367120,3546346384},
 {carriers,455,455,455},
 {carriers_size,
3763863552,3763863552,3763863552},
```

Statistics: example mbc

	Current	Max (Last)	Max
blocks	1066675	1068988	1811013
blocks_size	860267920	860267920	3546346384
carriers	455	455	455
carriers_size	3763863552	3763863552	3763863552

Statistics: example mbc

	Current	Max (Last)	Max
blocks	1066675	1068988	1811013
blocks_size	820 MB	820 MB	3382 MB
carriers	455	455	455
carriers_size	3590 MB	3590 MB	3590 MB

Statistics: example sbc

	Current	Max (Last)	Max
blocks	6	6	21
blocks_size	6 MB	6 MB	20 MB
carriers	6	6	21
carriers_size	7.5 MB	7.5 MB	25 MB

Statistics: calls

	alloc	free	realloc
binary	28379577160	28378510479	985494638
mseg	24186	23725	6839
sys	0	0	0

Statistics: calls

	alloc	free	realloc
binary	28380 MC	28379 MC	985 MC
mseg	24186	23725	6839
sys	0	0	0

Statistics: mseg

cached segments	2
cache hits	424
segments	12
segments_size	12136448
segments_watermark	4
mseg alloc	464
mseg dealloc	452
mseg create	40
mseg destroy	32

Case studies

- Case 1: Large binaries
- Case 2: Fragmentation

Case 1: Large binaries

- Symptoms
  - used strace to find that many more malloc than mmap were made

Case 1: Large binaries

calls binary_alloc	321 MC
calls mseg_alloc	0.4 MC
calls sys_alloc	1.4 MC
mbcs carrier_size	2.4 GB
sbcs carrier_size	11 GB
avg sbc block size	1.68 MB

Case 1: Large binaries

- +MBsbct 2147483648
  - Put binaries that are > 2 MB to mbcs
- +MBlmbcs 20480 +MBsmbcs 1024
  - Increase average mbc size to fit the new larger blocks that will be put there

Case 2: Fragmentation

- Symptoms
  - erlang:memory(total) = about 7GB
  - top showed process at about 15 GB
  - Crash dump was written to: erl_crash.dump. ets_alloc: Cannot allocate XYZ bytes of memory. Abnormal termination

Case 2: Fragmentation

	Current	Max
blocks	2161022	4346598
blocks_size	1647 MB	6823 MB
carriers	934	936
carriers_size	7262 MB	7271 MB
avg block sz	799 Bytes/Block	1645 Bytes/Block
avg carrier sz	7.8 MB/Carrier	7.7 MB/Carrier
block sz / carrier sz	22,7%	93,8%

Case 2: Fragmented binaries

- +MBas aobf
  - Strive to allocate binaries in address order when there are ties
- +MBlmbcs 512
  - Decreasing largest mbc size will make more carriers and hopefully be able to free them

New features

- Migration of carriers of same type
  - Added in R16B01, default in 17.0
  - Requires carrier oriented allocation strategy
- Super carrier
  - Added in R16B03

Questions?