CS 354
Machine Organization and Programming

Lecture 23

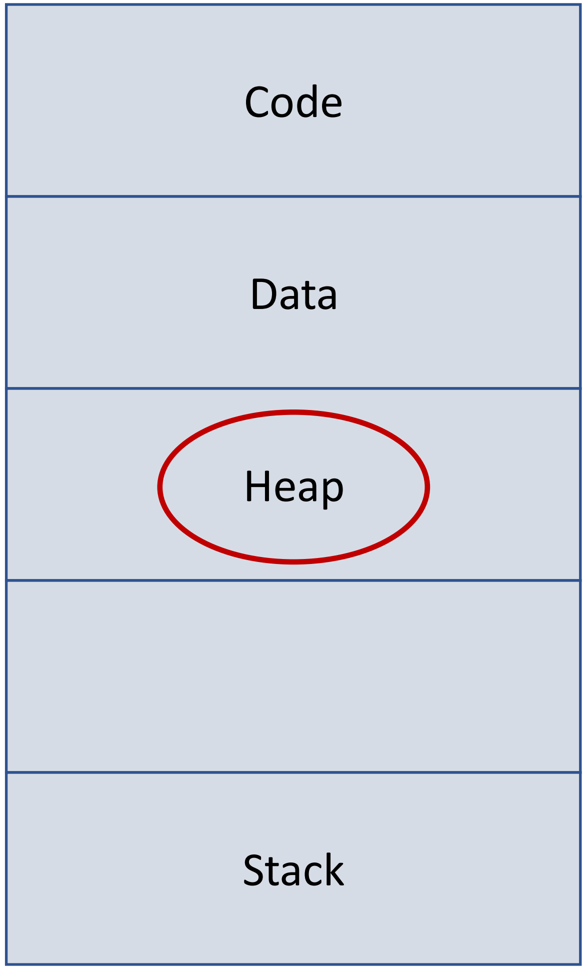
Michael Doescher
Summer 2020

Dynamic Memory Allocation
Part 2

Address Space

Memory Allocator Requirements

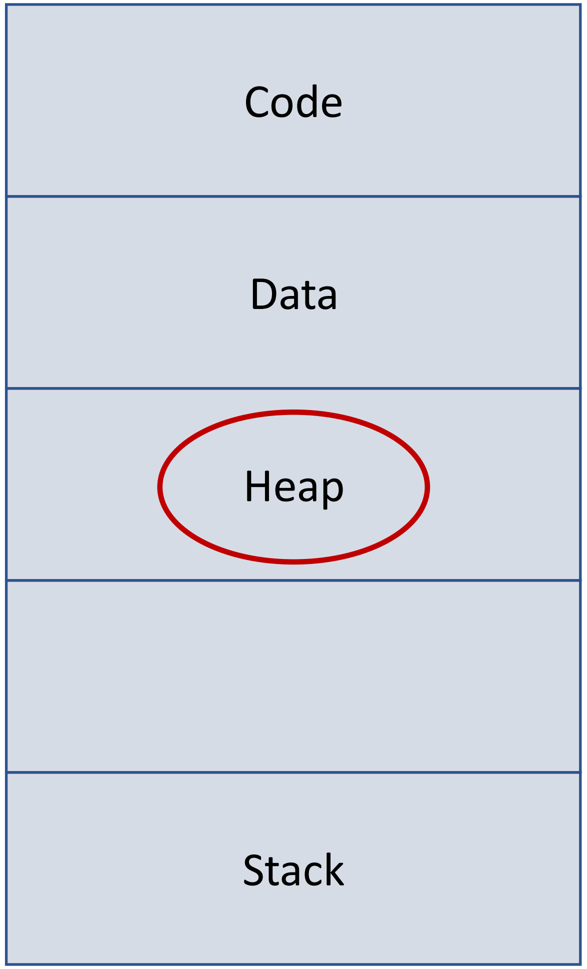
1. Keep track of block sizes for freeing
2. Arbitrary Request Sequences
3. Immediate Response
    1. no buffering
    2. or reordering
4. Use only the heap
5. Block Alignment
6. No modification of allocated blocks



Address Space

Implementation Issues

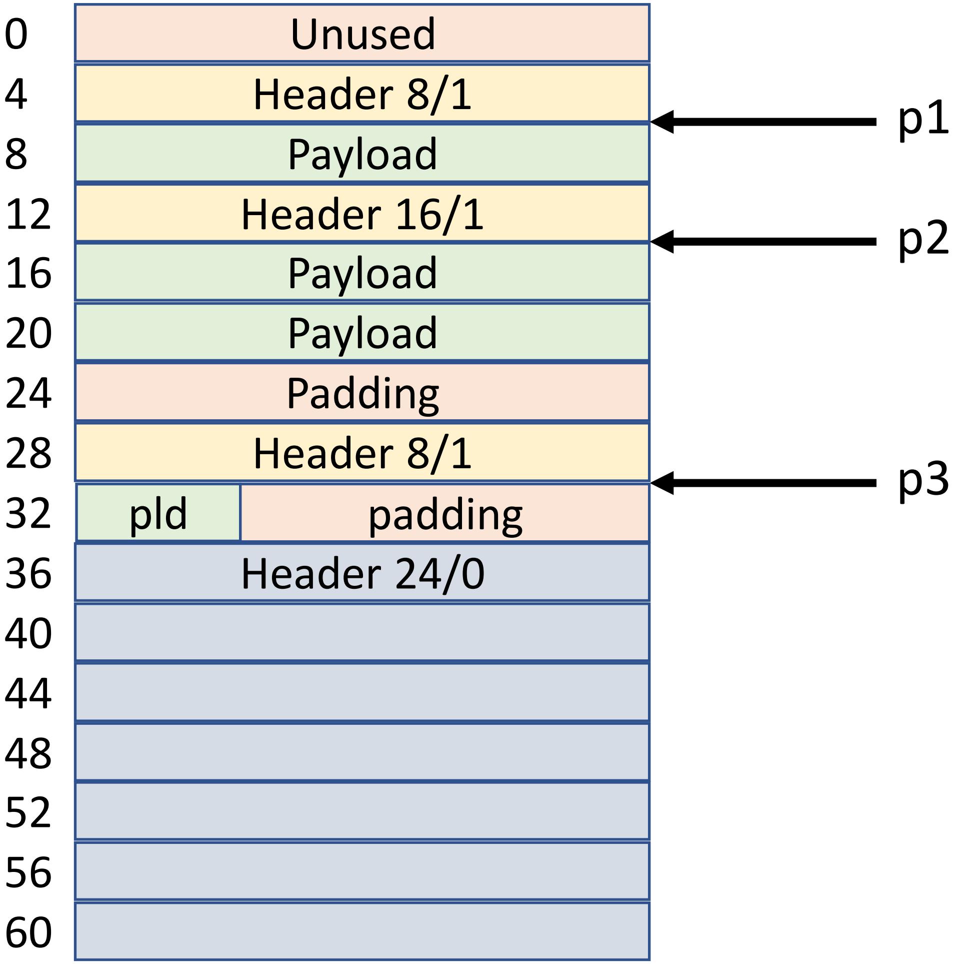
1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing



Address Space

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

Heap



0 Unused

4 Header 8/1

8 Payload

12 Header 16/1

16 Payload

20 Payload

24 Padding

28 Header 8/1

32 pld padding

36 Header 24/0

40

44

48

52

56

60

p1

p2

p3

Implicit Free List

```
int *p1 = malloc(4);
int *p2 = malloc(8);
int *p3 = malloc(1);

free (p2)
```

Heap

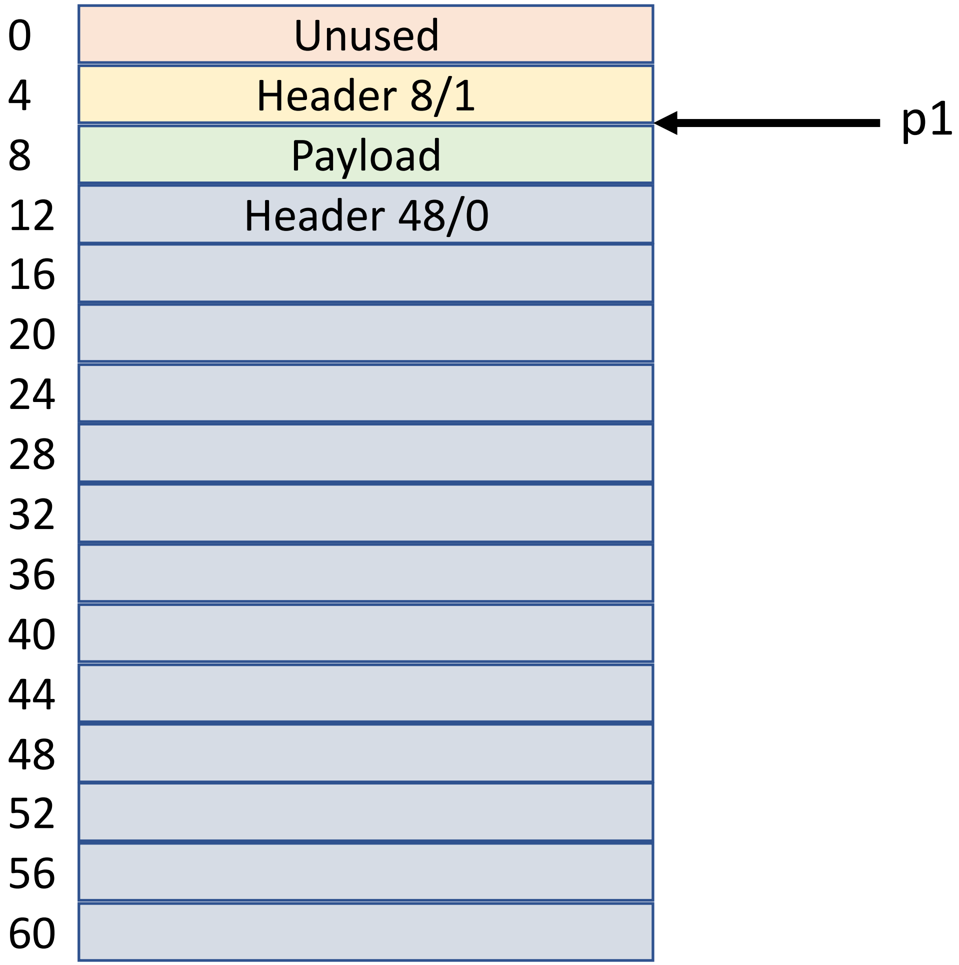
0	Unused
4	Header 56/0
8	
12	
16	
20	
24	
28	
32	
36	
40	
44	
48	
52	
56	
60	

Implicit Free List

```
int *p1 = malloc(4);
int *p2 = malloc(8);
int *p3 = malloc(1);

free (p2)
```

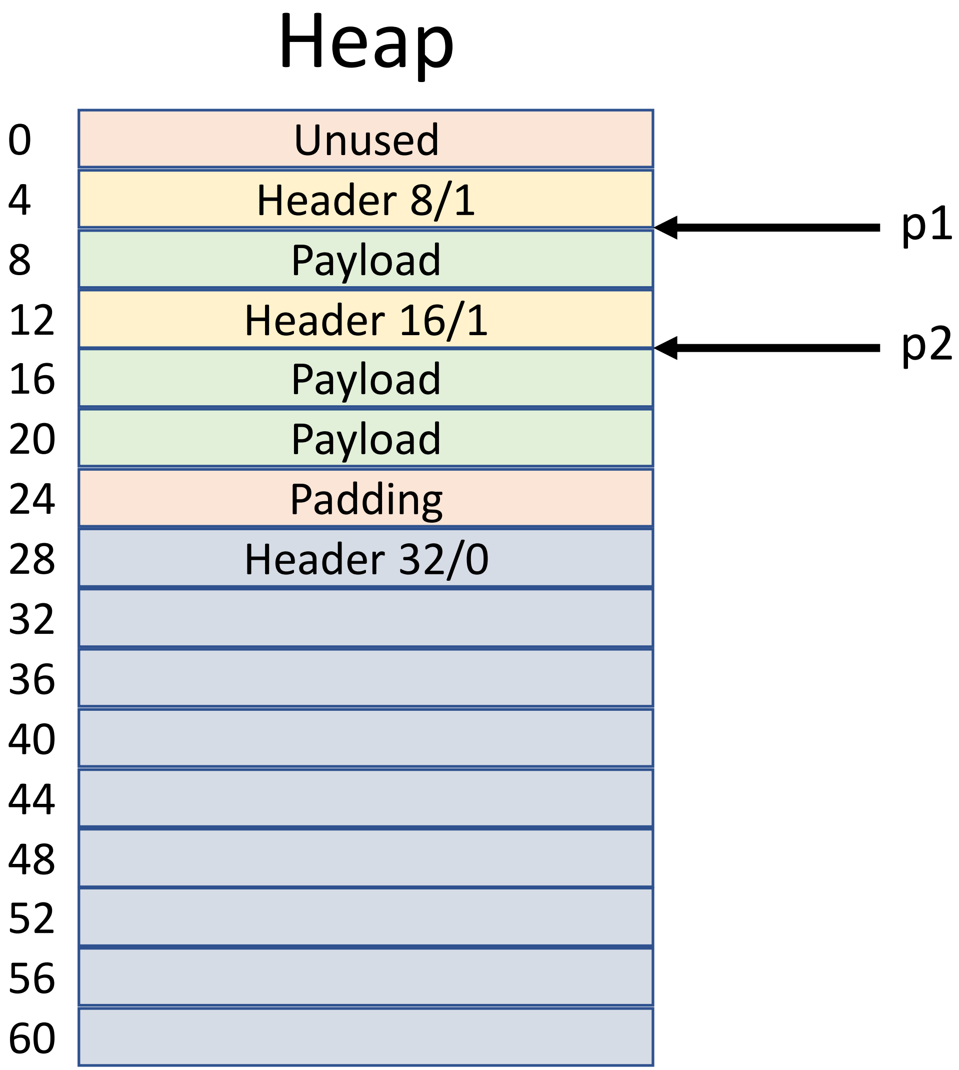
Heap



Implicit Free List

```
int *p1 = malloc(4);
int *p2 = malloc(8);
int *p3 = malloc(1);

free (p2)
```



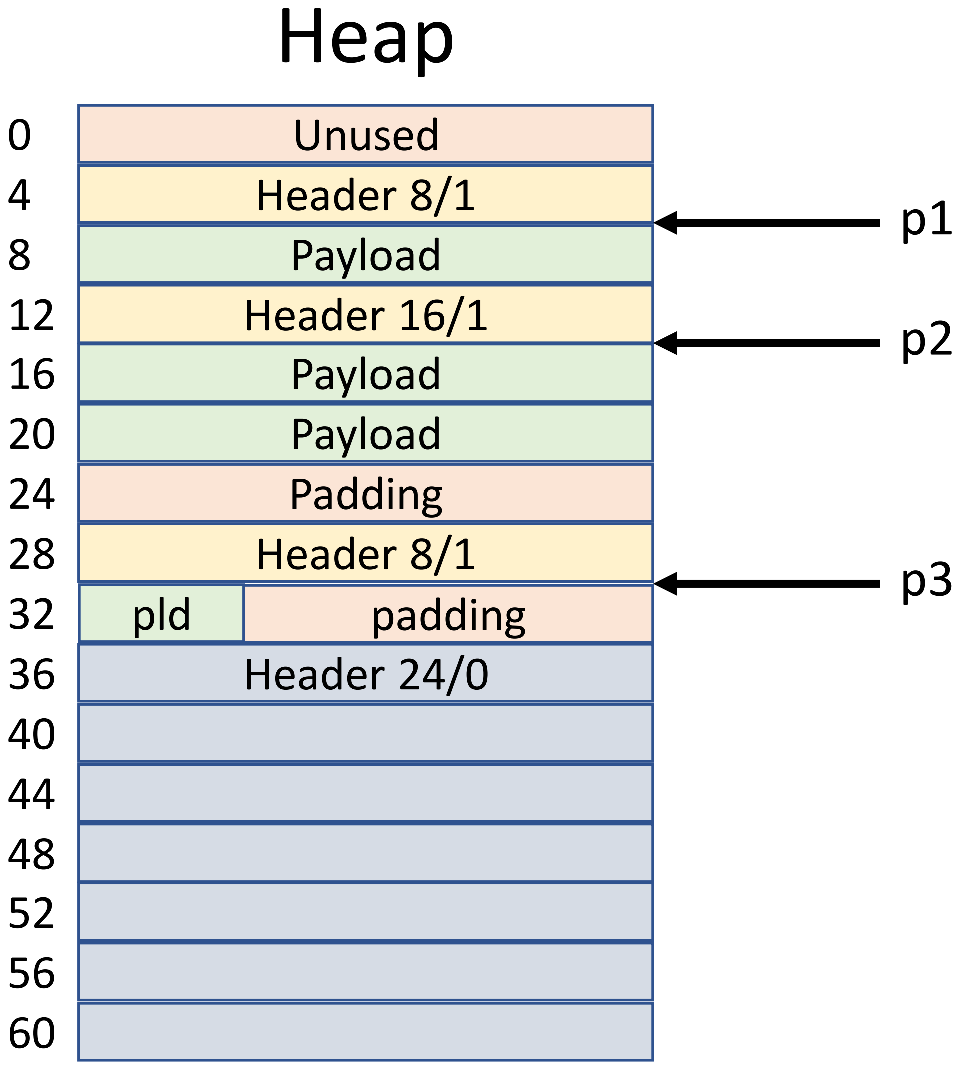
Heap

0	Unused	
4	Header 8/1	
8	Payload	p1
12	Header 16/1	
16	Payload	p2
20	Payload	
24	Padding	
28	Header 32/0	
32		
36		
40		
44		
48		
52		
56		
60		

Implicit Free List

```
int *p1 = malloc(4);
int *p2 = malloc(8);
int *p3 = malloc(1);

free (p2)
```



Heap

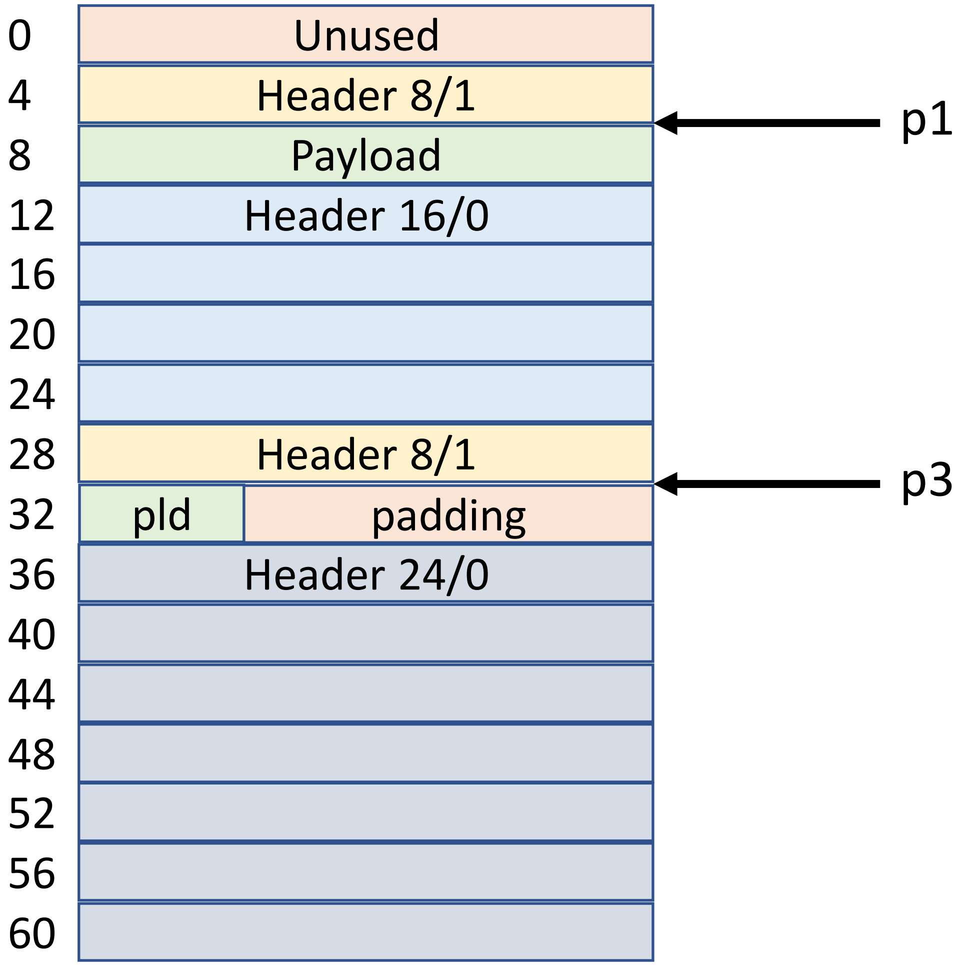
0	Unused		
4	Header 8/1		
8	Payload		p1
12	Header 16/1		
16	Payload		p2
20	Payload		
24	Padding		
28	Header 8/1		
32	pld	padding	p3
36	Header 24/0		
40			
44			
48			
52			
56			
60			

Implicit Free List

```
int *p1 = malloc(4);
int *p2 = malloc(8);
int *p3 = malloc(1);

free (p2)
```

Heap



0	Unused		
4	Header 8/1		
8	Payload		p1
12	Header 16/0		
16			
20			
24			
28	Header 8/1		
32	pld	padding	p3
36	Header 24/0		
40			
44			
48			
52			
56			
60			

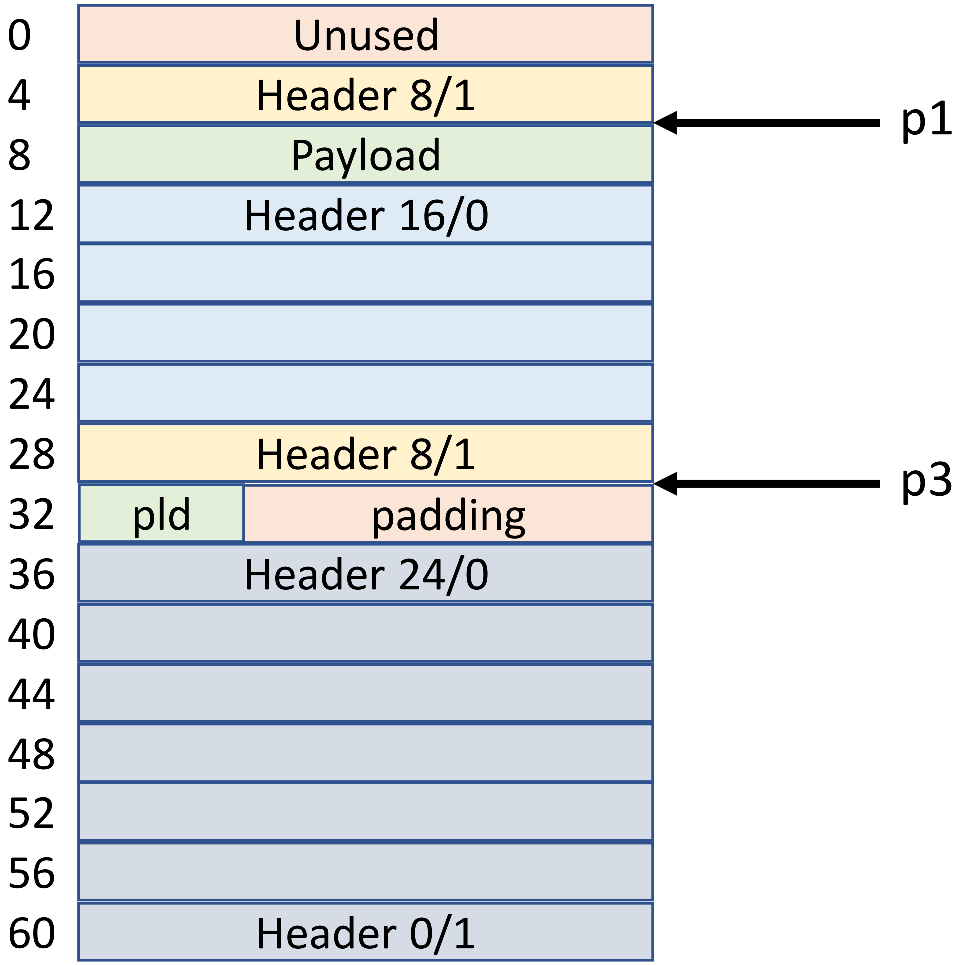
Implicit Free List

```
int *p1 = malloc(4);
int *p2 = malloc(8);
int *p3 = malloc(1);

free (p2)
```

End of the Stack Marker

Heap



0	Unused	
4	Header 8/1	
8	Payload	← p1
12	Header 16/0	
16		
20		
24		
28	Header 8/1	
32	pld | padding	← p3
36	Header 24/0	
40		
44		
48		
52		
56		
60	Header 0/1	

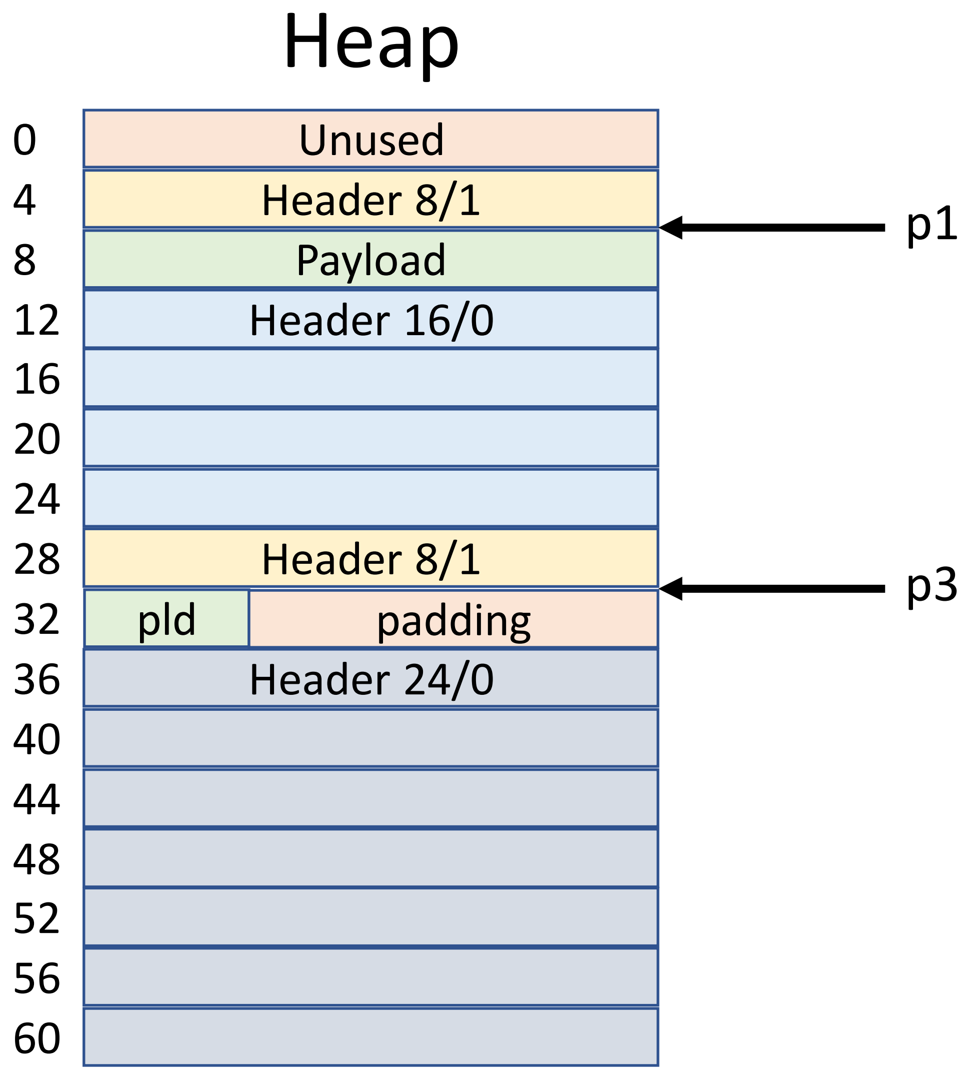
Implicit Free List

<u>How to pack the size of the allocated block and the allocated / free flag into 4 bytes?</u>

The size must be a multiple of 8
0x 0008 : 8 : 0000 1000
0x 0010 : 16 : 0001 0000
0x 0018 : 24 : 0001 1000
0x 0020 : 32 : 0010 0000

<u>Just use the b0 bit for the flag</u>

8/0 -> 0000 1000 = 8
8/1 -> 0000 1001 = 9
16/1 -> 0001 0001 = 17



Heap

0 Unused
4 Header 8/1
8 Payload
12 Header 16/0
16
20
24
28 Header 8/1
32 pld padding
36 Header 24/0
40
44
48
52
56
60

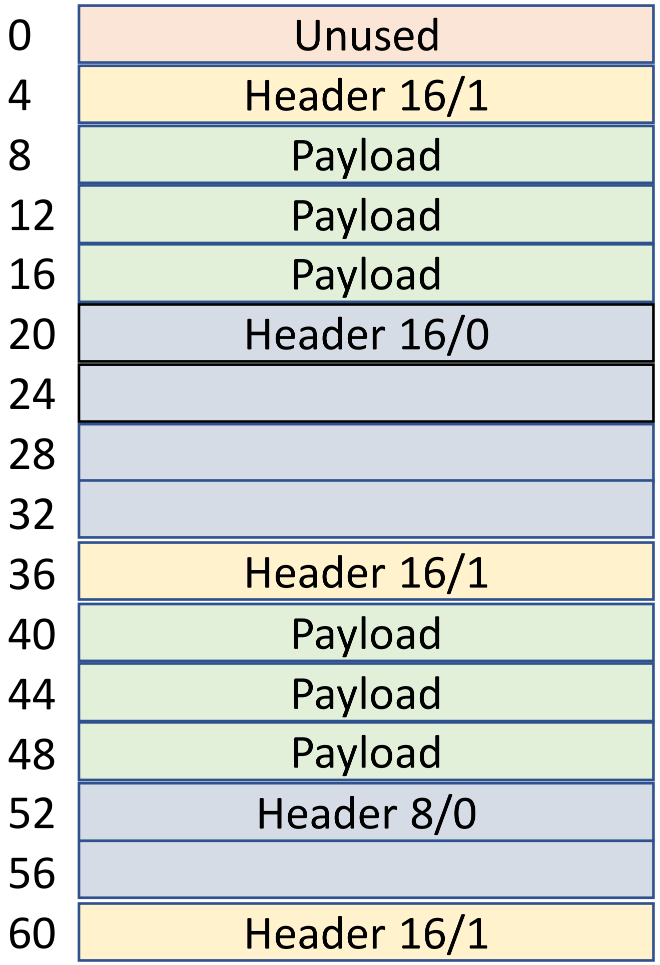
p1
p3

Implicit Free List

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

malloc(4)
malloc(12)

Heap



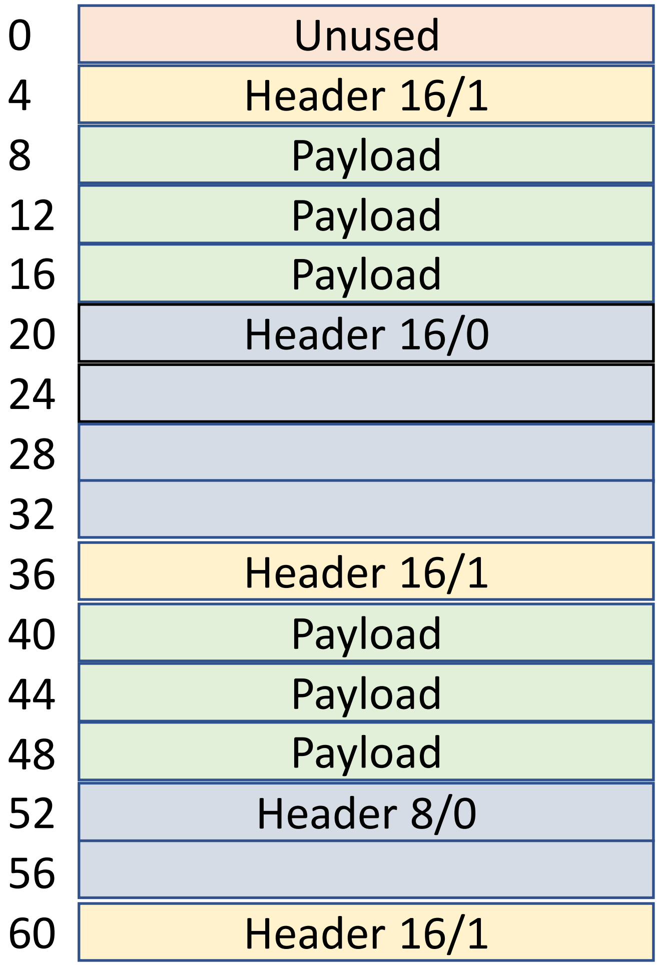
0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 16/0
24	
28	
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

Implicit Free List

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

Linked List

Heap



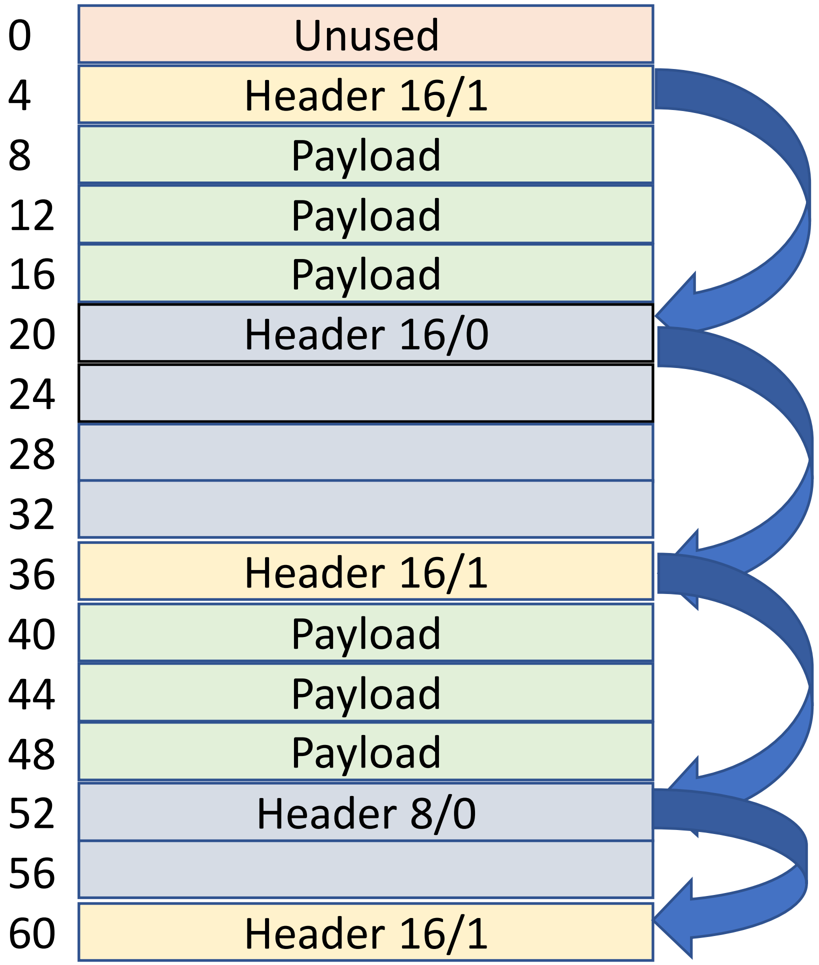
0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 16/0
24	
28	
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

Implicit Free List

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

Linked List

Heap



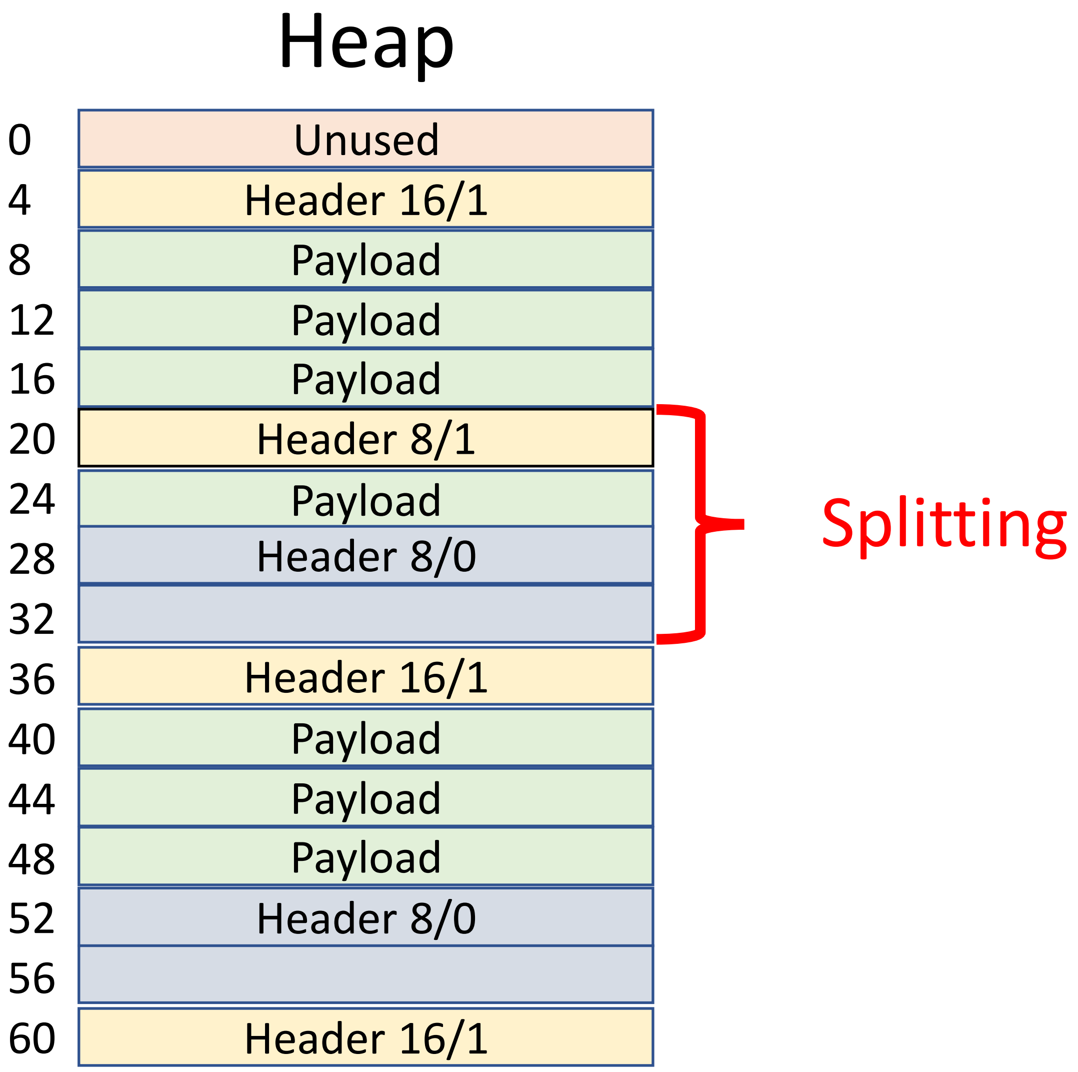
0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 16/0
24	
28	
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

Placement Policies

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

First Fit

- malloc(4) -> success
- malloc(12)



Heap

0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/1
24	Payload
28	Header 8/0
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

Placement Policies

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

First Fit

- malloc(4) -> success
- malloc(12) -> fails

Heap

0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/1
24	Payload
28	Header 8/0
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

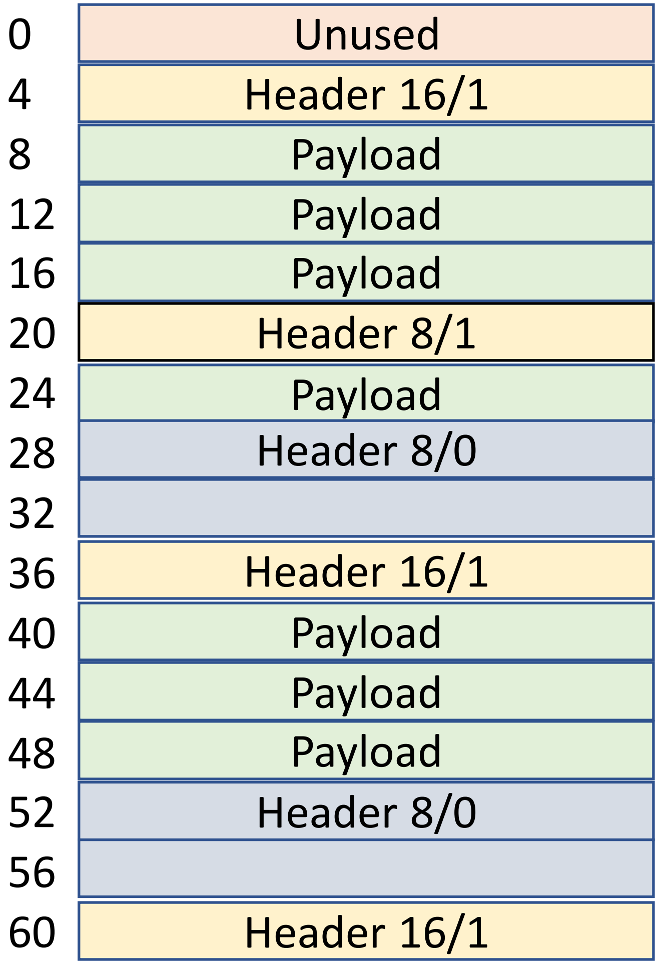
Placement Policies

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

First Fit

- retains large blocks at the end of the list
- but fragments the beginning of the list
- increases search time

Heap



0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/1
24	Payload
28	Header 8/0
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

Placement Policies

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

First Fit

Next Fit

- begin searching for the next available block from the last found.

Heap

0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/1
24	Payload
28	Header 8/0
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

Placement Policies

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

First Fit

Next Fit

- begin searching for the next available block from the last found.
- Faster than First Fit
- Worse Memory Utilization

Heap

0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/1
24	Payload
28	Header 8/0
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

Placement Policies

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

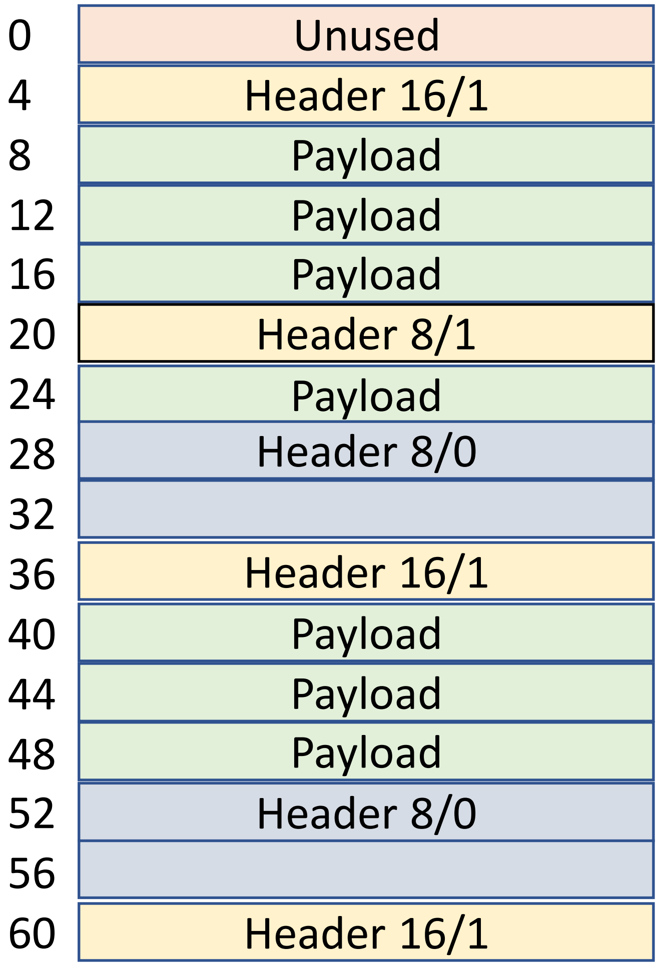
First Fit

Next Fit

Best Fit

- Search the entire memory for the smallest block that will work
- Slow
- Good memory utilization
- Small Fragments

Heap



0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/1
24	Payload
28	Header 8/0
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

Placement Policies

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

First Fit

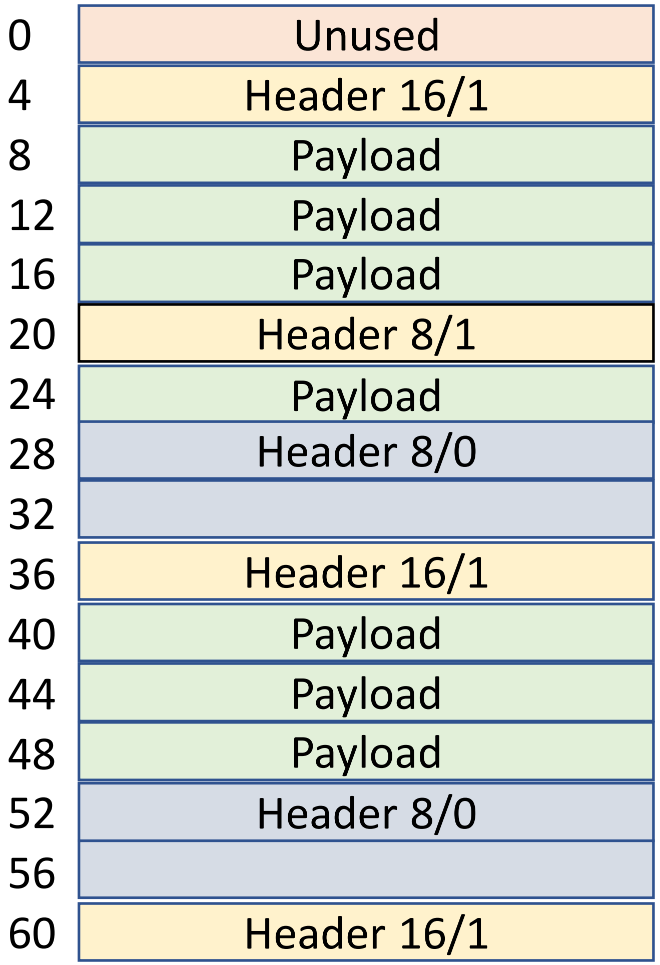
Next Fit

Best Fit

Worst Fit

- Allocate from the largest block
- Doesn't leave small fragments behind
- "Maximizes the chance that the next allocation will fit"

Heap



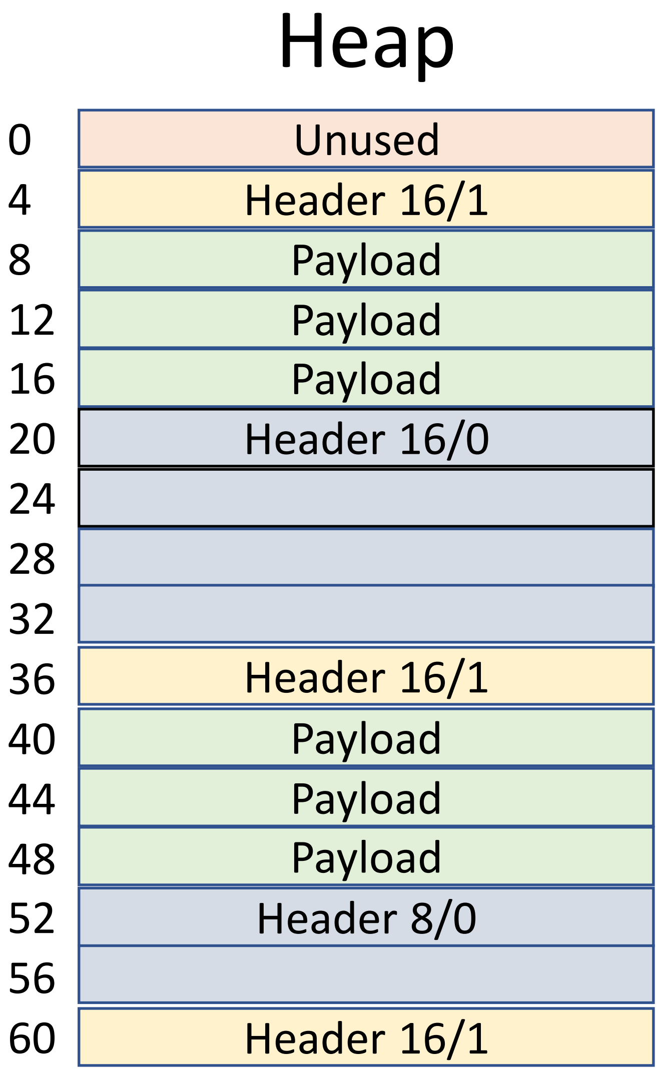
0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/1
24	Payload
28	Header 8/0
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

Splitting

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

Allocate the entire block

Split the block



Heap

0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 16/0
24	
28	
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

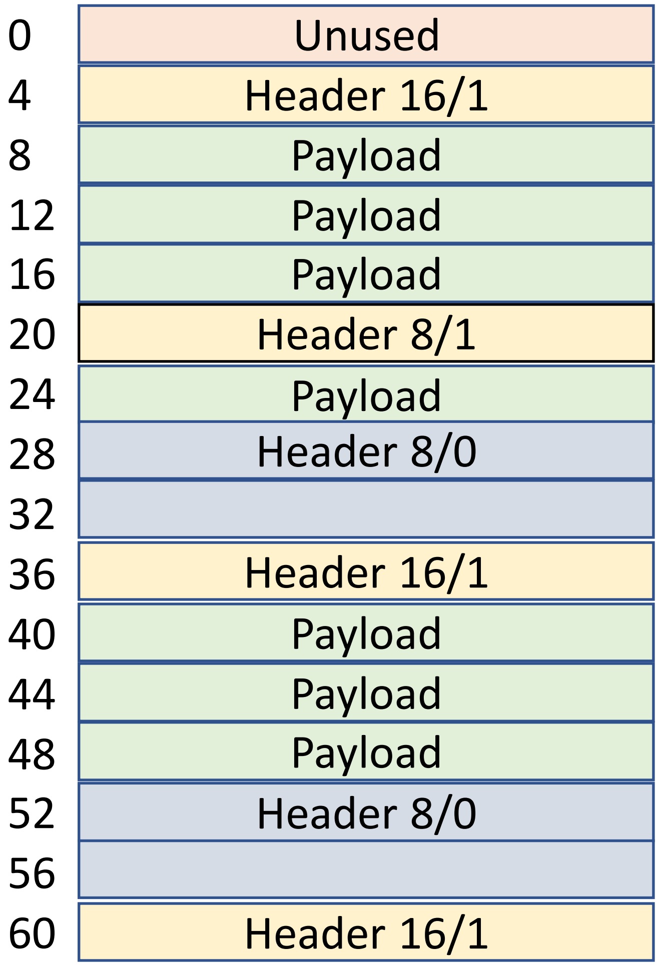
Coalescing

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

Immediate

Deferred

Heap

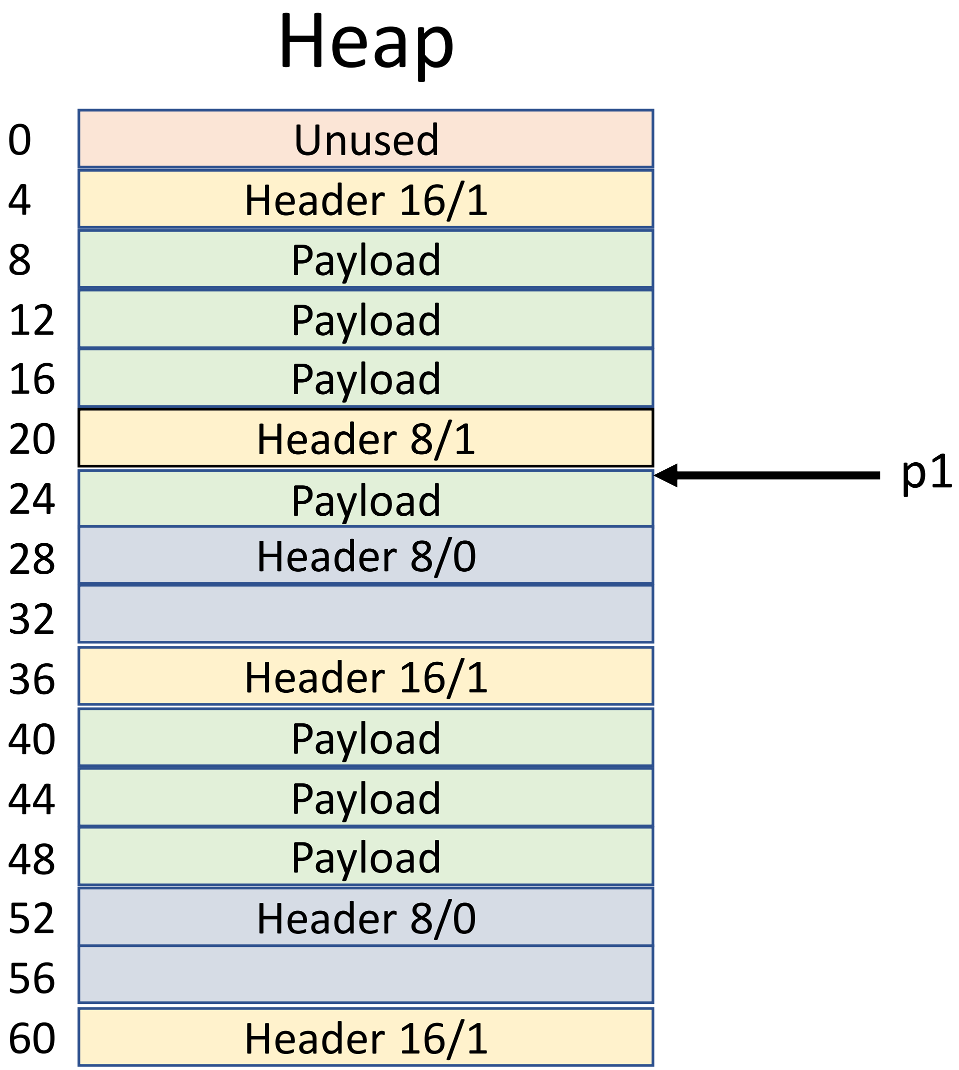


0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/1
24	Payload
28	Header 8/0
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

Coalescing

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

Example 1: Free block at 24



Heap

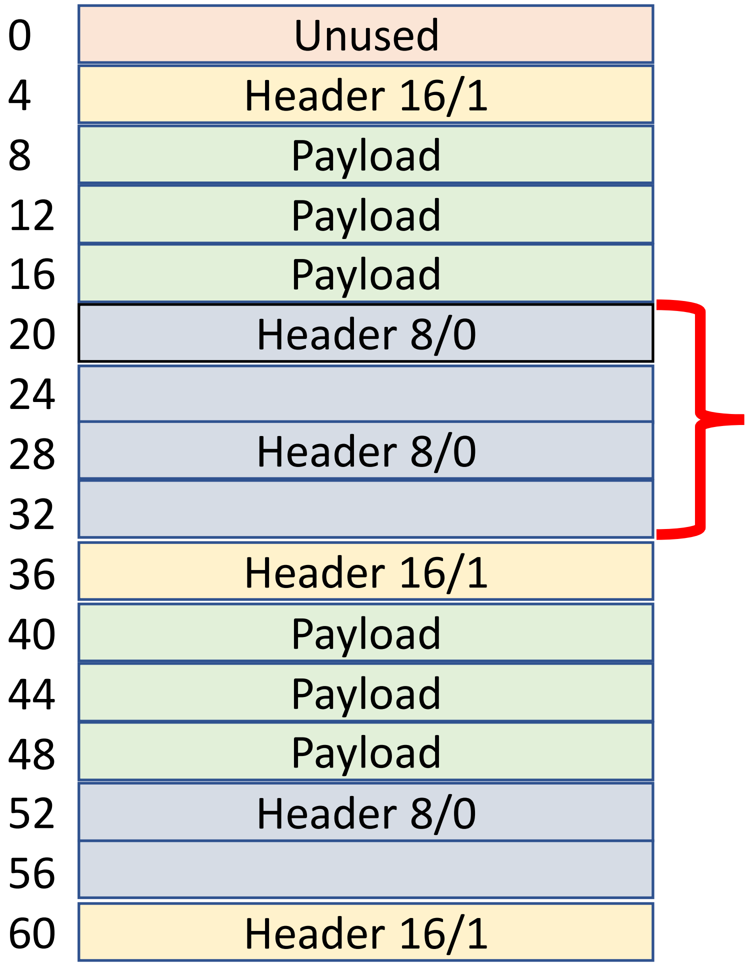
0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/1
24	Payload
28	Header 8/0
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

Coalescing

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

Example 1: Free block at 24

Heap



0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/0
24	
28	Header 8/0
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

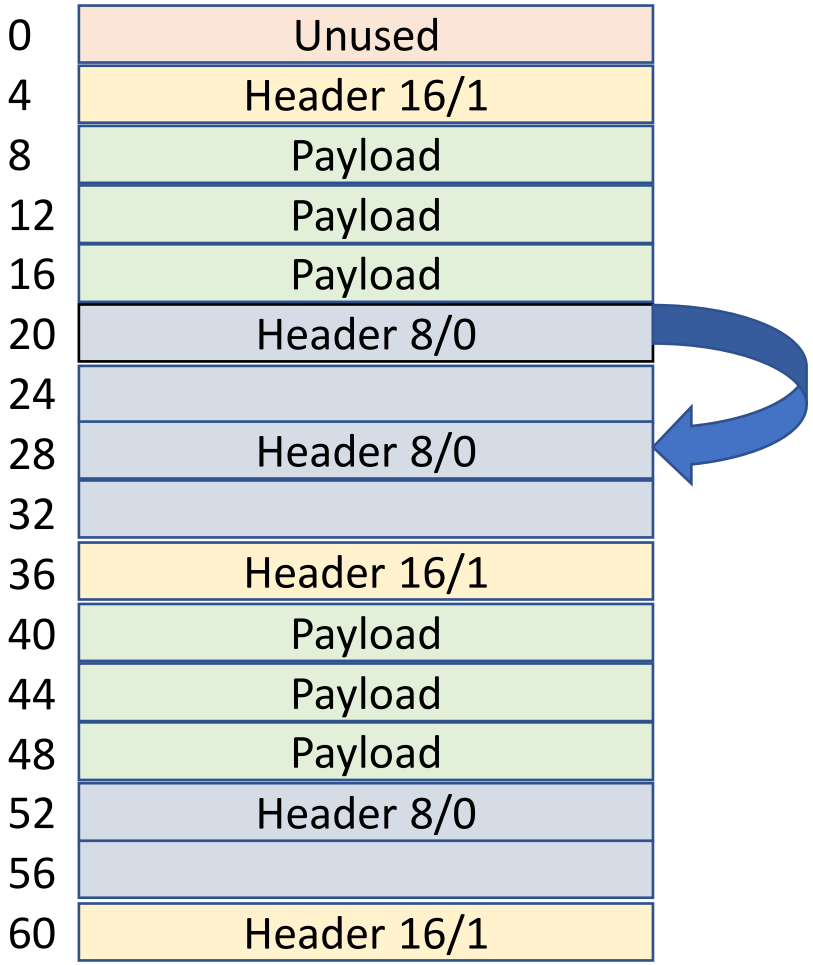
False Fragmentation

Coalescing

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

Example 1: Free block at 24

Heap



0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/0
24	
28	Header 8/0
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

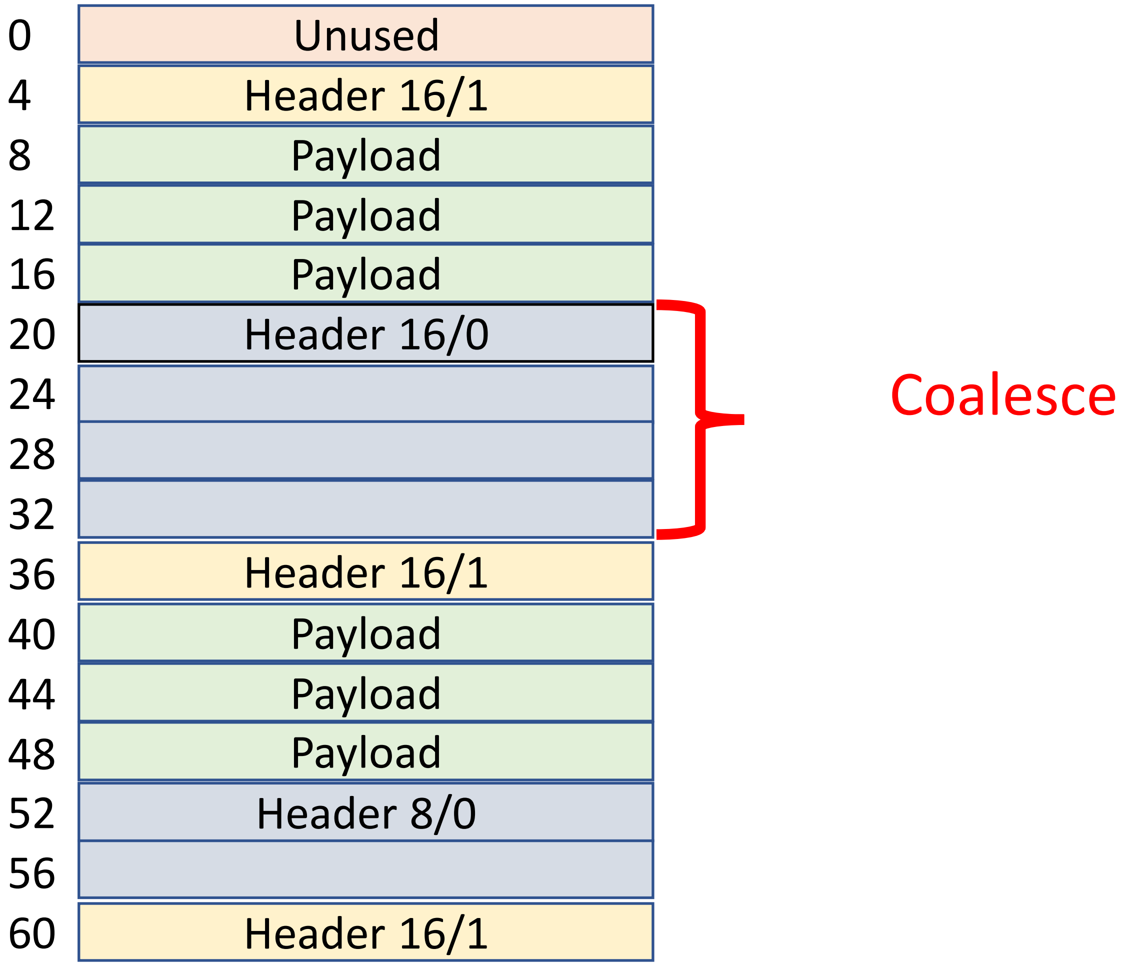
Coalescing

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

Example 1: Free block at 24

Immediate

Heap



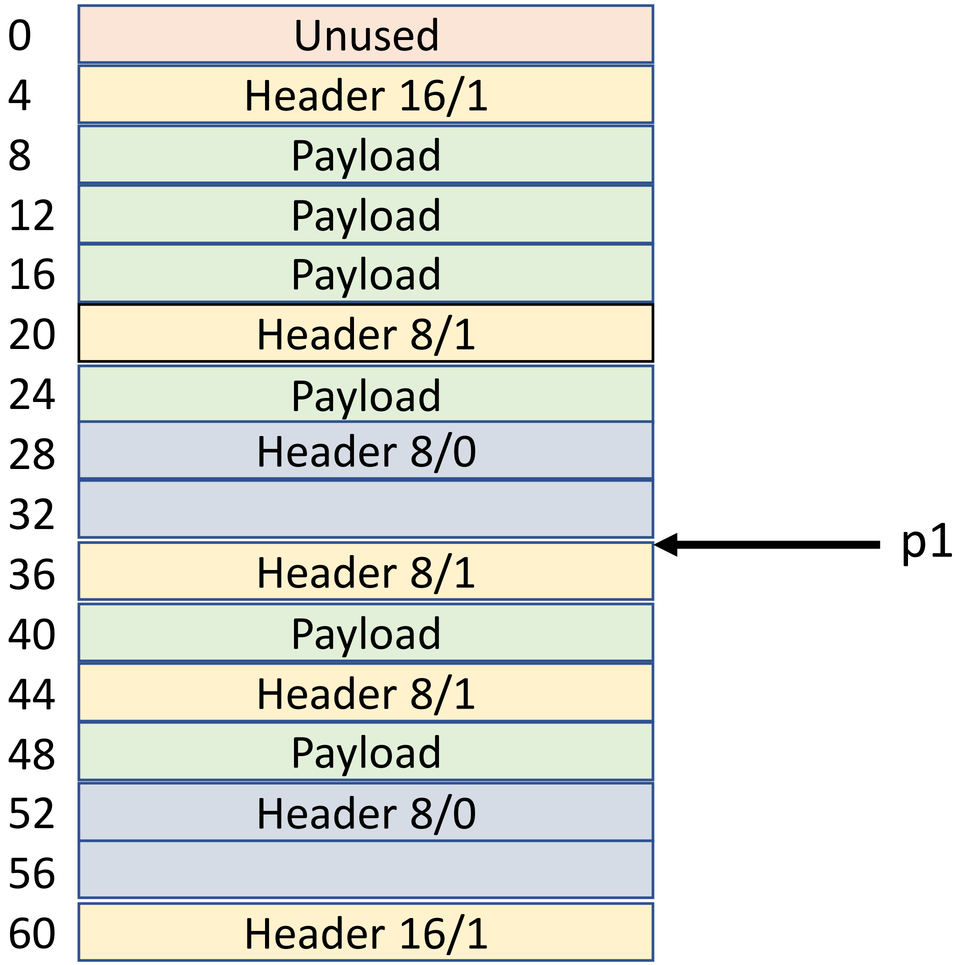
0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 16/0
24	
28	
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

Coalescing

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

Example 2: Free block at 36

Heap



0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/1
24	Payload
28	Header 8/0
32	
36	Header 8/1
40	Payload
44	Header 8/1
48	Payload
52	Header 8/0
56	
60	Header 16/1

Coalescing

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

Example 2: Free block at 36

Heap

0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/1
24	Payload
28	Header 8/0
32	
36	Header 8/0
40	
44	Header 8/1
48	Payload
52	Header 8/0
56	
60	Header 16/1

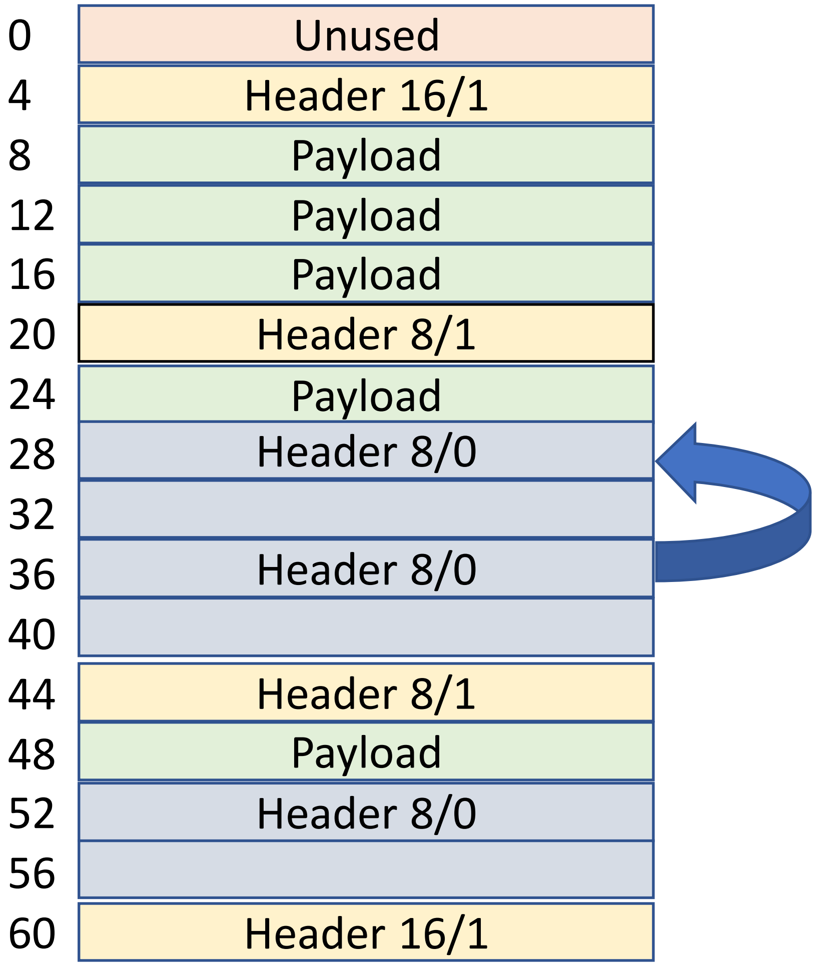
Coalescing

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

Example 2: Free block at 36

No pointer backward

Heap



0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/1
24	Payload
28	Header 8/0
32	
36	Header 8/0
40	
44	Header 8/1
48	Payload
52	Header 8/0
56	
60	Header 16/1

Coalescing

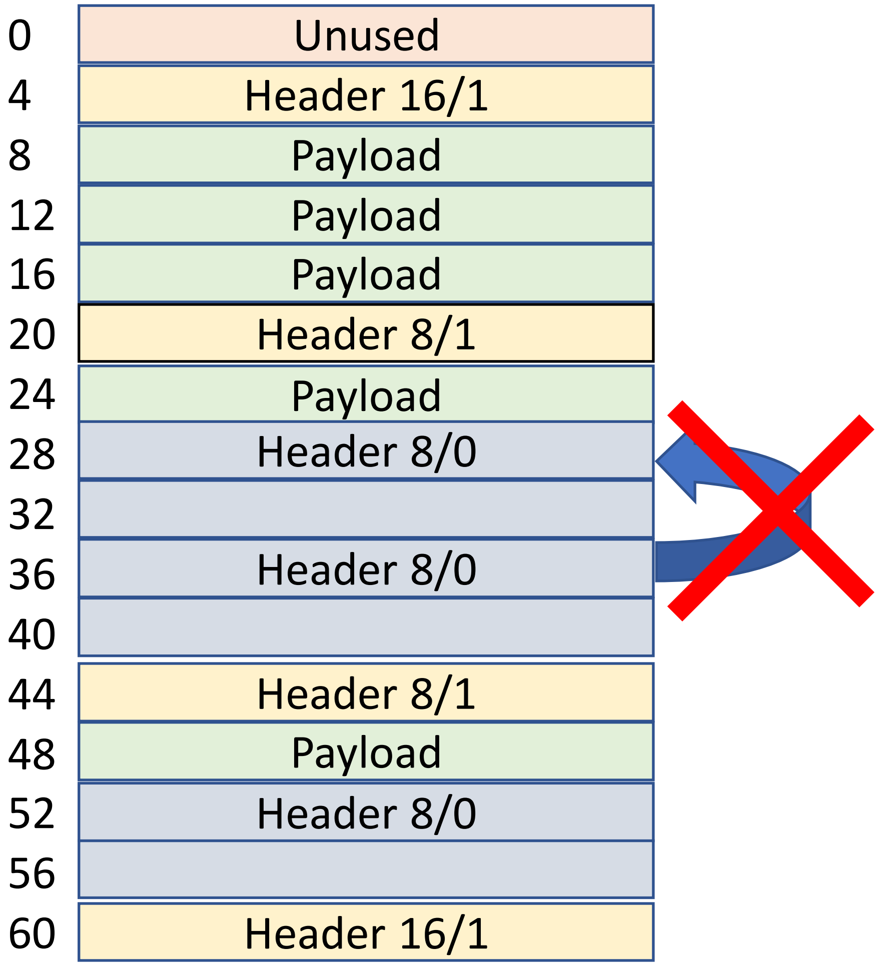
1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

Example 2: Free block at 36

Need to traverse the entire list

Deferred

Heap



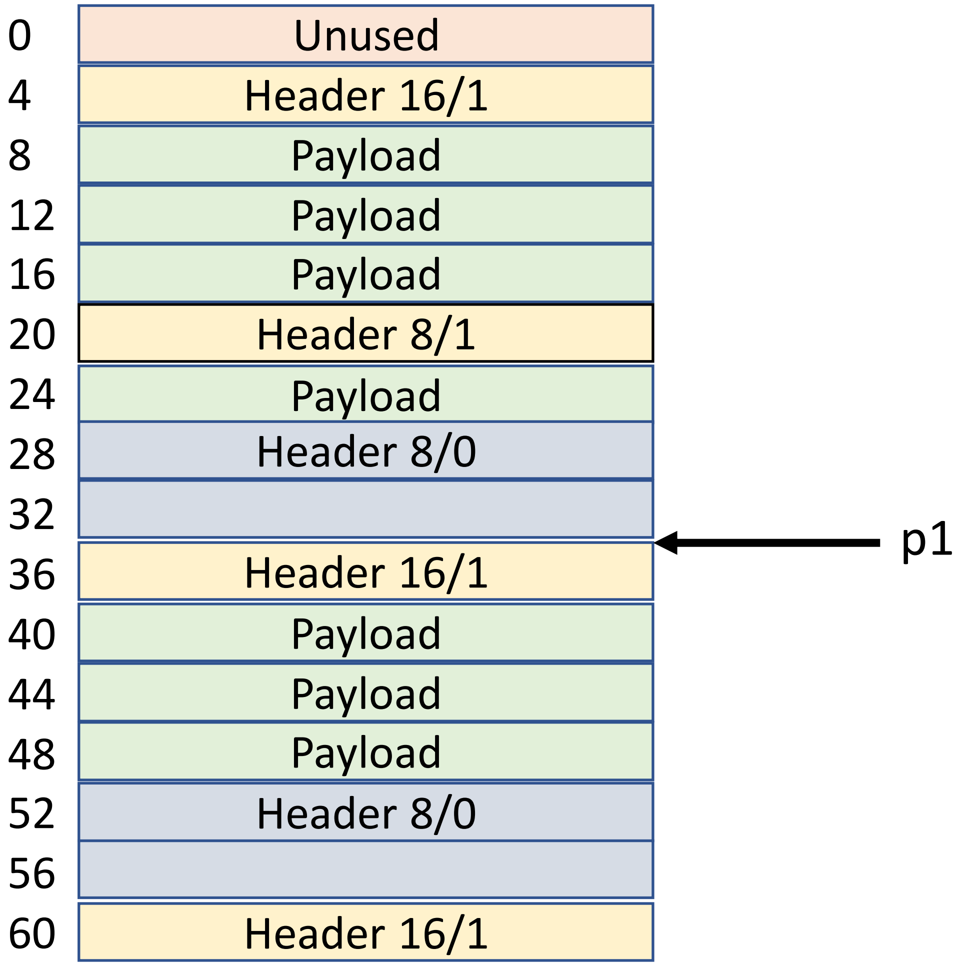
0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/1
24	Payload
28	Header 8/0
32	
36	Header 8/0
40	
44	Header 8/1
48	Payload
52	Header 8/0
56	
60	Header 16/1

Coalescing

1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

Example 3: Free block at 36

Heap



0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/1
24	Payload
28	Header 8/0
32	
36	Header 16/1
40	Payload
44	Payload
48	Payload
52	Header 8/0
56	
60	Header 16/1

Coalescing

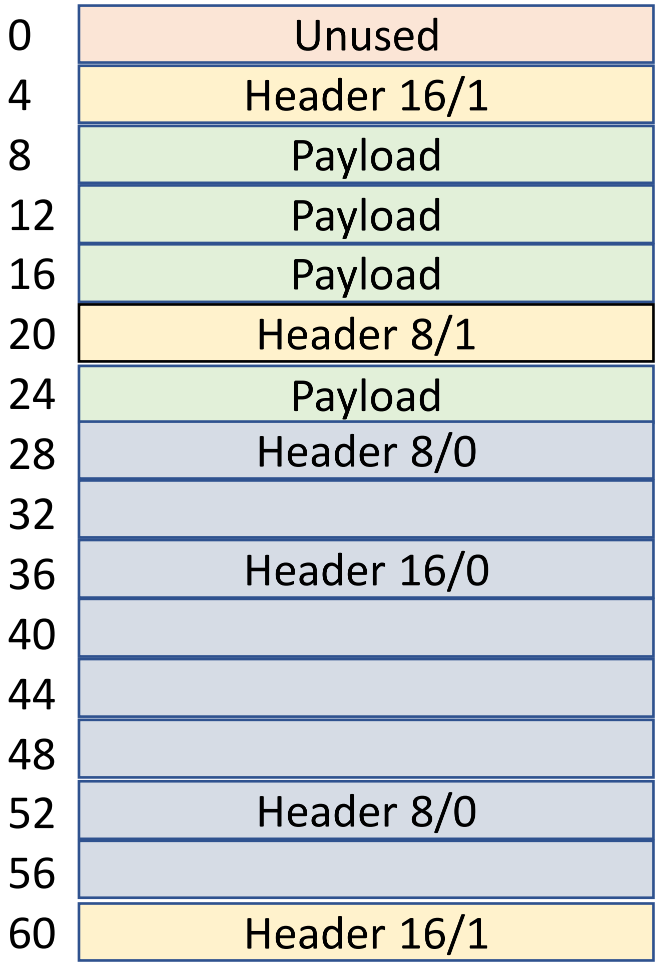
1. Tracking Free Blocks – Free List
2. Placement Policy
3. Splitting
4. Coalescing

Example 3: Free block at 36

One free block ahead

One free block behind

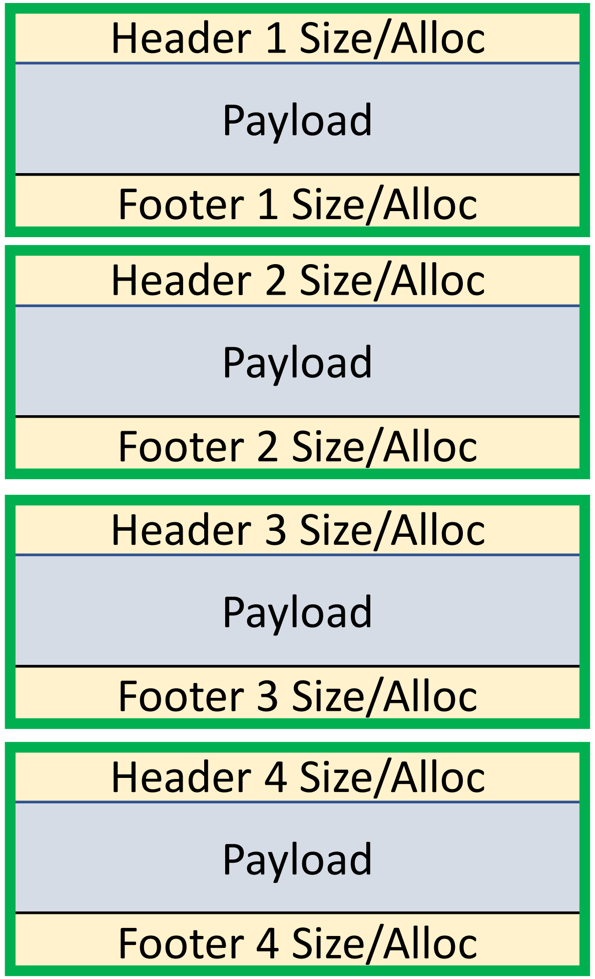
Heap



0	Unused
4	Header 16/1
8	Payload
12	Payload
16	Payload
20	Header 8/1
24	Payload
28	Header 8/0
32	
36	Header 16/0
40	
44	
48	
52	Header 8/0
56	
60	Header 16/1

Coalescing Strategies – Boundary Tags

Strategy: Duplicate the header as the last block



Header 1 Size/Alloc

Payload

Footer 1 Size/Alloc

Header 2 Size/Alloc

Payload

Footer 2 Size/Alloc

Header 3 Size/Alloc

Payload

Footer 3 Size/Alloc

Header 4 Size/Alloc

Payload

Footer 4 Size/Alloc

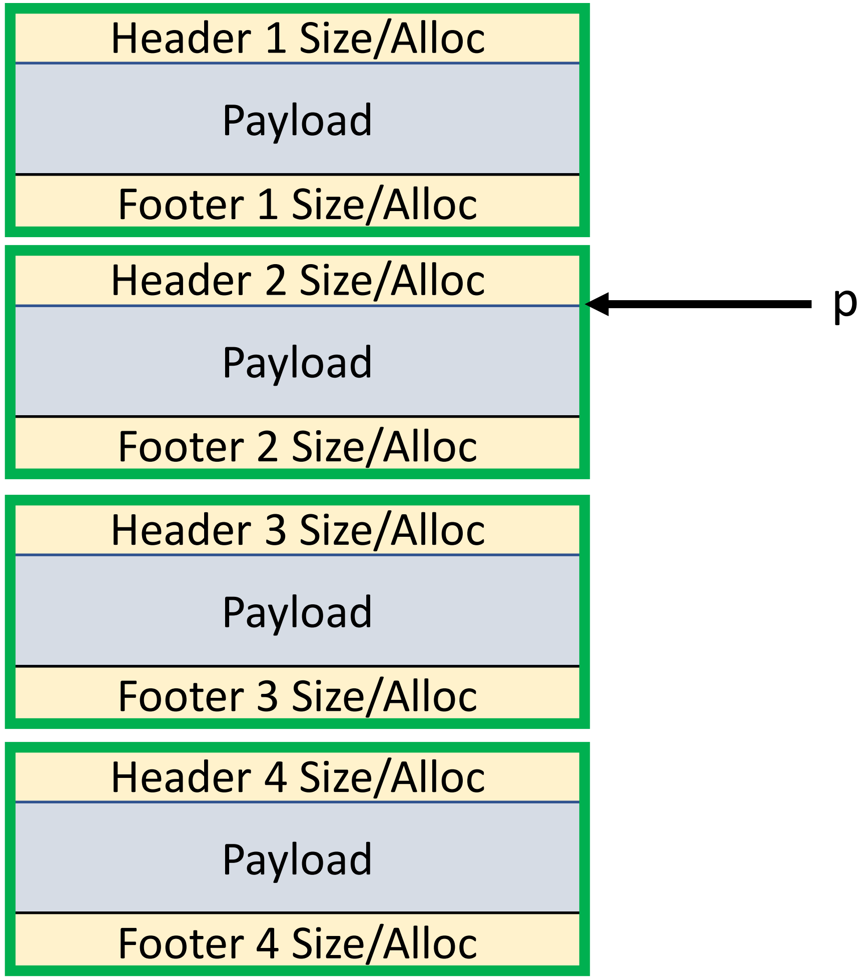
Coalescing Strategies – Boundary Tags

Strategy: Duplicate the header as the last block

free (p)

- byte before p holds the size of this block
- 2 bytes before holds the size/alloc of the previous block

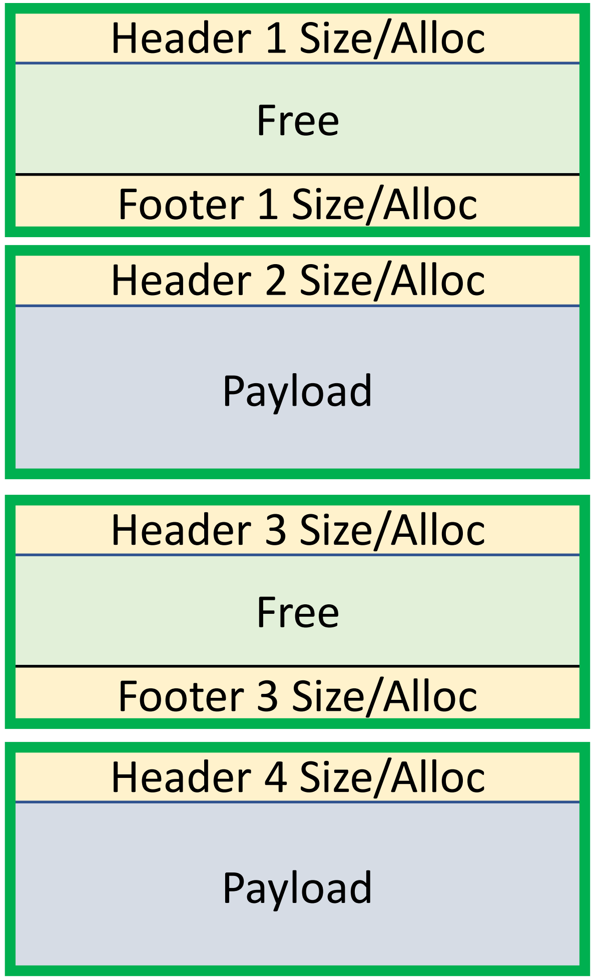
- Decreases memory utilization – especially for applications that use many small blocks (linked lists)
- Fast coalescing of adjacent blocks



Coalescing Strategies – Improved Boundary Tags

- Only free blocks need footers
- Headers are multiple of 8 so we get three bits that must be 0 for the size
- We're already using b0 for the **alloc** bit
- Use b1 for the **previous alloc** bit

- Slightly more complicated
- Still constant time coalescing
- Better memory utilization


Header 1 Size/Alloc
Free
Footer 1 Size/Alloc
Header 2 Size/Alloc
Payload
Header 3 Size/Alloc
Free
Footer 3 Size/Alloc
Header 4 Size/Alloc
Payload