System Programming Project 3

담당 교수 : 김영재 교수님

이름 : 조다민

학번 : 20181297

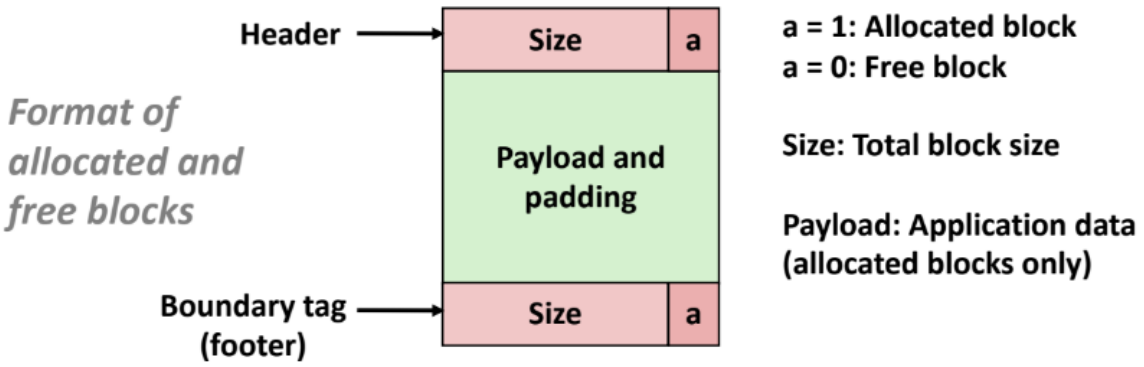
1. **개발 목표**

이번 프로젝트를 통해 dynamic memory allocator의 작동에 대해서 이해하고, implicit list 방식으로 malloc, free, realloc을 구현했다. 추가로 explicit free list 방식에 대한 이론을 살펴보고 explicit free list를 이용한 malloc package를 위해서 추가로 필요한 내용에 대해서도 고민하는 시간을 가질 수 있었다.

2. **개발 범위 및 내용**

1) 사용한 자료구조

- heap list 구조: heap list는 아래 figure1과 같이 header와 footer가 있는 block의 형태를 이용하고, header와 footer에 쓰여진 size 값을 통해서 이전 block 또는 다음 block으로 접근할 수 있다.



2) 선언한 변수 및 macro

- 각종 macro: block pointer에 쉽게 접근하기 위해 GET, PUT, GET_SIZE, GET_ALLOC, HDRP, FTRP, NEXT_BLKP, PREV_BLKP 등의 macro를 작성하였고 각각에 대한 설명은 mm.c code 주석으로 작성하였다.

- static char* heap_listp를 이용해 heap 내부의 첫번째 block을 가리킬 수 있도록 한다.

3) 선언한 함수

- static void* coalesce(void* bp): coalesce 함수에서는 heap list에 block이 추가될 때, 앞 뒤 block이 allocate 상태인지 free 상태인지에 따라 처리 방식에 차이가 있는데, 각각에 대한 처리를 coalesce 함수에서 다룬다. case는 4개로 나누어 앞뒤 모두 allocate 상태일때, 앞의 block이 free 상태일 때, 뒤의 block이 free 상태일 때, 앞 뒤 모두 free 상태일 때로 나눠서 free인 block이 있을 때는 현재 block과 free된 block을 합치는 작업을 이 함수에서 실행한다.

- static void* extend_heap(size_t words): extend_heap 함수에서는 heap_listp 안에 적절한 size를 가진 free 상태의 block이 없을 때, 해당 size를 가진 block을 추가하는 함수이다.

- static void place(void* bp, size_t asize): place 함수에서는 사용가능한 블록의 시작에 asize bytes의 블록을 배치하고, 남은 블록이 크기 이상일 경우에는 분할을 한다.

- static void* find_fit(size_t asize): find_fit함수에서는 heap_listp 안의 free된 block 중 원하는 size에 맞는 block을 찾는 함수이다. heap_listp 안에 있는 block들의 맨 앞부터 살피면서 제일 먼저 나오는 조건을 만족하는 block을 return 한다. 만약 조건을 만족하는 block이 없다면 NULL을 return해 find_fit을 호출한 함수에서 heap list에 새로운 block을 추가할 수 있도록 한다.

- int mm_init(void): malloc, free, realloc등의 malloc package 안의 함수를 부를 때 heap list가 초기화 되어있지 않으면 mm_init함수를 불러 heap_listp를 초기화한다. 초기화를 할 때는 heap_listp에 allocated 되어 있는 block 하나를 추가하고 추가된 만큼 heap을 확장 시킨다.

- void * mm_malloc(size_t size): size 크기의 block을 allocate 하기 위해서 함수를 사용한다. 기본적인 크기를 DSIZE로 하기 때문에, 혹시 size가 DSIZE보다 작다면 size를 조정해 padding을 만들어주고 block의 size를 조정해 asize 만큼의 free 상태의 block이 있는지 find_fit 함수를 이용해 찾고 만약 없다면 extend_heap 함수를 이용해 heap을 확장한다.

- void mm_free(void* ptr): block을 free 상태로 표시하고 free 된 block에 대하여 coalesce 함수를 실행한다.
- void mm_realloc(void *ptr, size_t size): ptr block을 주어진 size로 realloc하는 함수이다. size가 0이면 ptr을 free하고, ptr이 원래 NULL인 pointer라면 mm_malloc(size)를 실행해 그대로 return 한다. 그 외의 경우에는 먼저 새로운 포인터를 mm_malloc(size)를 통해 malloc한 뒤 memcpy를 통해 ptr의 내용을 size만큼 복사한다. 그리고 ptr을 free 시킨 뒤 새로운 포인터를 return한다.

3. 구현 결과

1) 성능 분석

- find_fit과 checkheap을 제외한 모든 함수가 시간 복잡도가 O(1)을 가진다.
- find_fit과 checkheap은 block 수에 따라 for문을 탐색하기 때문에 O(n)의 시간 복잡도를 가진다.
- mdriver를 이용한 성능 분석 값은 space utilization 44, throughput 10으로 총 54의 결과값이 나왔다.
- find_fit 함수 등에서 allocated 상태의 block까지 살펴보기 때문에 시간적 효율성이 떨어진다.

2) Explicit free list의 이용

- implicit list를 사용할 경우의 단점을 극복하기 위해 Explict free list 방식을 이용할 수 있다. 이 경우 free_listp 를 추가로 선언해 free된 block을 모아 관리하는 pointer를 만든다.