Projekt i IOOPM16

Skräpsamlare

Vi är Humrarna

Adam Inersjö

Simon Pellgård

Henrik Bergendal

Robert Rosborg

Maria Lindqvist

Daniel Ågstrand

Implementation

- GC
- Stack_search
- Alloc_map
- Header

```
struct heap
{
  void *memory;
  alloc_map_t *alloc_map;
  size_t size;
  bool unsafe_stack;
  float gc_threshold;
  size_t number_of_pages;
  page_t *pages[];
};
```

Demonstration av tester

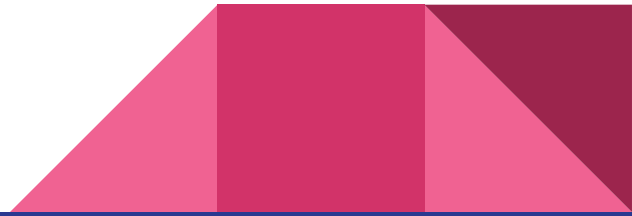
Coverage för gc.c: Lines executed:93.72% of 366

Coverage för header.c: Lines executed:92.86% of 280

Coverage för stack_search.c: Lines executed:100.00% of 10

Coverage för alloc_map.c: Lines executed:100.00% of 29

- Regressions tester



Demonstration av programmet

Demonstration av prestandatester

Running list_bench without filling the heap

Type of test	Runtime (s)
Using Malloc	73.20
Using IOOPM GC	91.88

Trying to insert 100 000 objects with gc_perf_test run h_gc() and search 10 000 times

Type of test	Runtime (s)
Using Malloc	8.82
Using IOOPM GC	88.35

Optimering

```
Each sample counts as 0.01 seconds.
  %   cumulative   self              self  total
 time   seconds   seconds     calls   s/call   s/call  name
 54.32      27.54       27.54 19647249    0.00  0.00  get_ptr_page
 18.70      37.02 9.48  3871512581  0.00  0.00  page_get_start
 14.49      44.36 7.34  3871512581  0.00  0.00  page_get_end
  7.46      48.14 3.78  20004 0.00  0.00  forward_internal_array_ptrs_with_offset
  1.65      48.98 0.84  1           0.84  0.84  create_pages
  0.81      49.39 0.41  40006 0.00  0.00  h_used
```

```c
int
get_ptr_page(heap_t *h, void * ptr)
{
  int number_of_pages = h->number_of_pages;
  for (int i = 0; i < number_of_pages; ++i)
    {
      page_t *current = h->pages[i];
      if(ptr >= page_get_start(current) && ptr < page_get_end(current))
        {
          return i;
        }
    }
    return -1;
}
```

Optimering

```
Each sample counts as 0.01 seconds.
  %   cumulative   self              self  total
 time   seconds   seconds     calls   s/call   s/call  name
 72.77      3.76  3.76        20004 0.00   0.00  forward_internal_array_ptrs_with_offset
  7.93      4.17  0.41        40006 0.00   0.00  h_used
  5.61      4.46  0.29        20000 0.00   0.00  list_contains
  2.71      4.60  0.14  1           0.14   4.13  h_gc_dbg
  1.94      4.70  0.10        15691488    0.00   0.00  find_next_active_page
  1.84      4.80  0.10        75051256    0.00   0.00  page_get_used
```

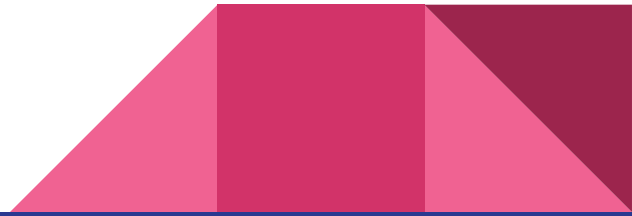
```c
int
get_ptr_page(heap_t *h, void * ptr)
{
  return (int) ((size_t) ptr - (size_t) h->memory) / PAGE_SIZE;
}
```

Förenkling/avvikelser

- SPARC
- Dumpa register
- h_used()
- Alokeringskarta
- Höga adresser
- Code coverage
- BDW

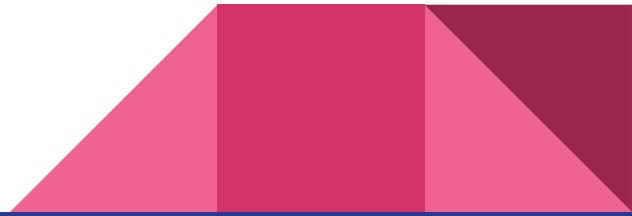
Tre svåraste

- Förstå stacken
- SPARC
- Hitta aktiva pekare på heapen



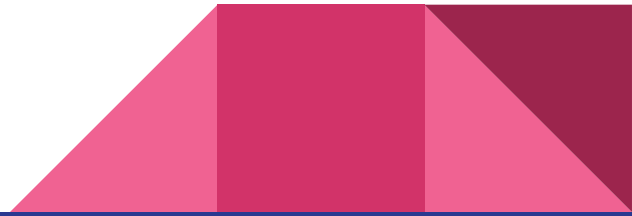
Mest nöjd med

- Bara en malloc
- Tester
- Att det fungerar
- Optimering av programmet



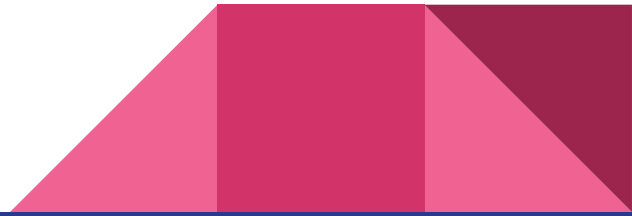
Fulhack

- __built_in_stackframe på SPARC
- Höga adresser



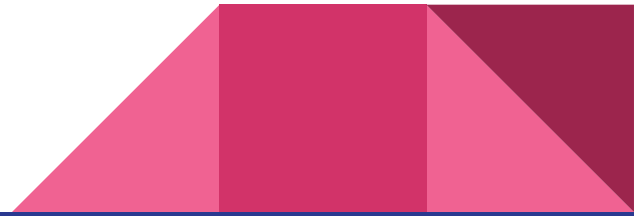
Organiserat arbetet

- Trello
- Slack
- Sprintar
- Parprogramering
- TDD



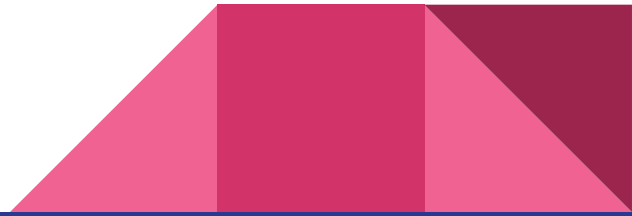
Hur har arbetet tillsammans gått?

Har parprogrammering fungerar bra?



Lösningar på mjuka problem?

- Formaterade om grupper
- Arbetat runt det



Tre saker som fungerat sämst?

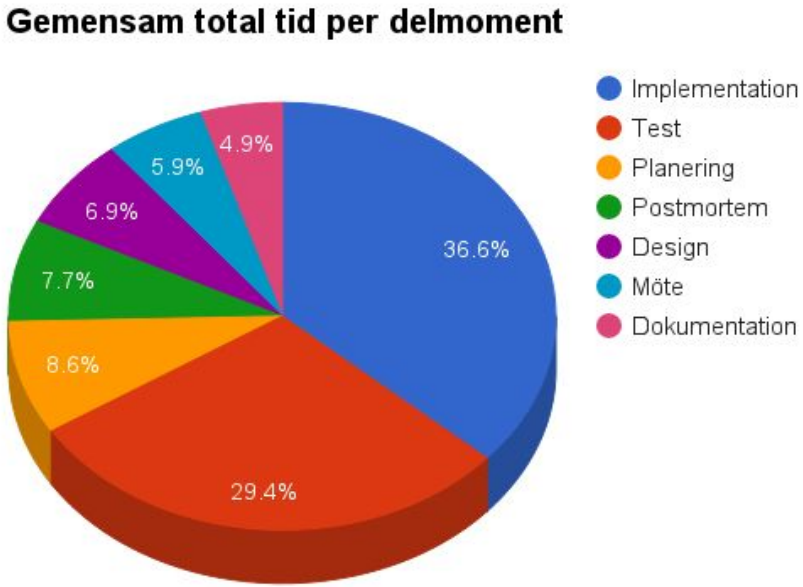
- Tider
- Olika scheman
- Lagt ner tid på omöjliga problem

Tre saker som fungerat bra?

- Bra sammanhållning i gruppen
- Alla var fas med vad vi gjorde
- Bra kommunikation via bland annat Slack

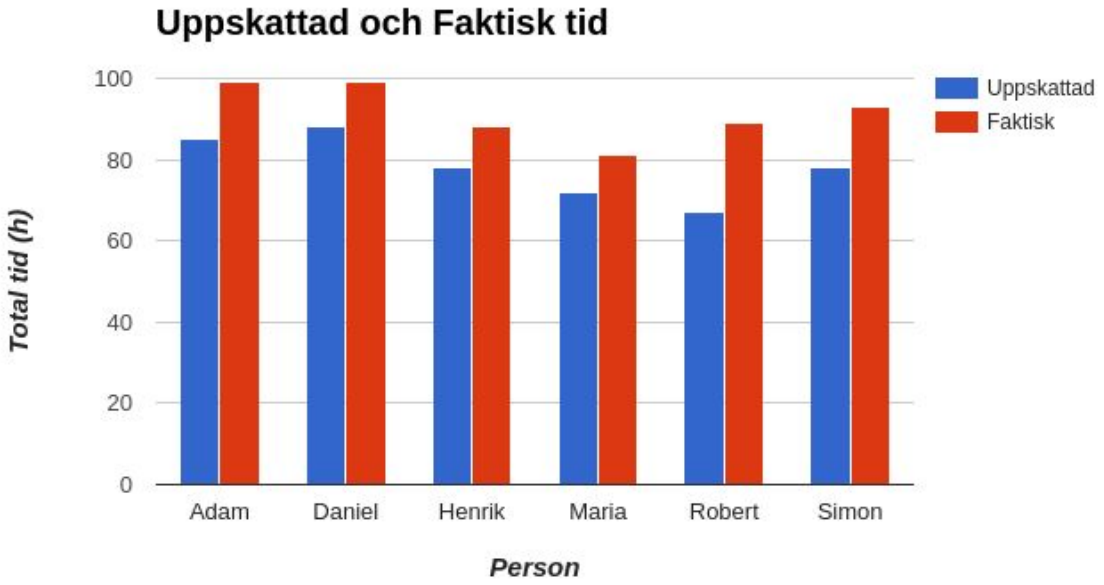
Tid

- Hur mycket tid har vi lagt **Total tid: 550+ timmar**
- Hur var tidsfördelningen per moment?



Tid

- Gick det att uppskatta tiden

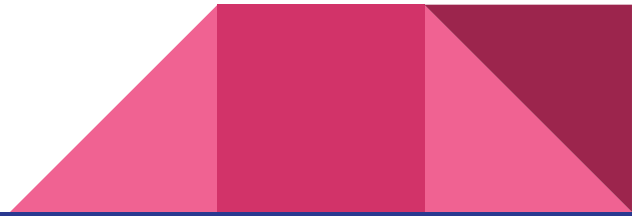


Planeringen

- Hur gick planeringen?
- Har planeringen hållit?
- Vad har vi gjort när planeringen inte hållit

Finns det någon plan att implementera det som saknas?

- SPARC
- Bitmap
- Performance tester



Hur lång tid skulle det ta att färdigställa?

- 50 ~ 250+ arbetstimmar