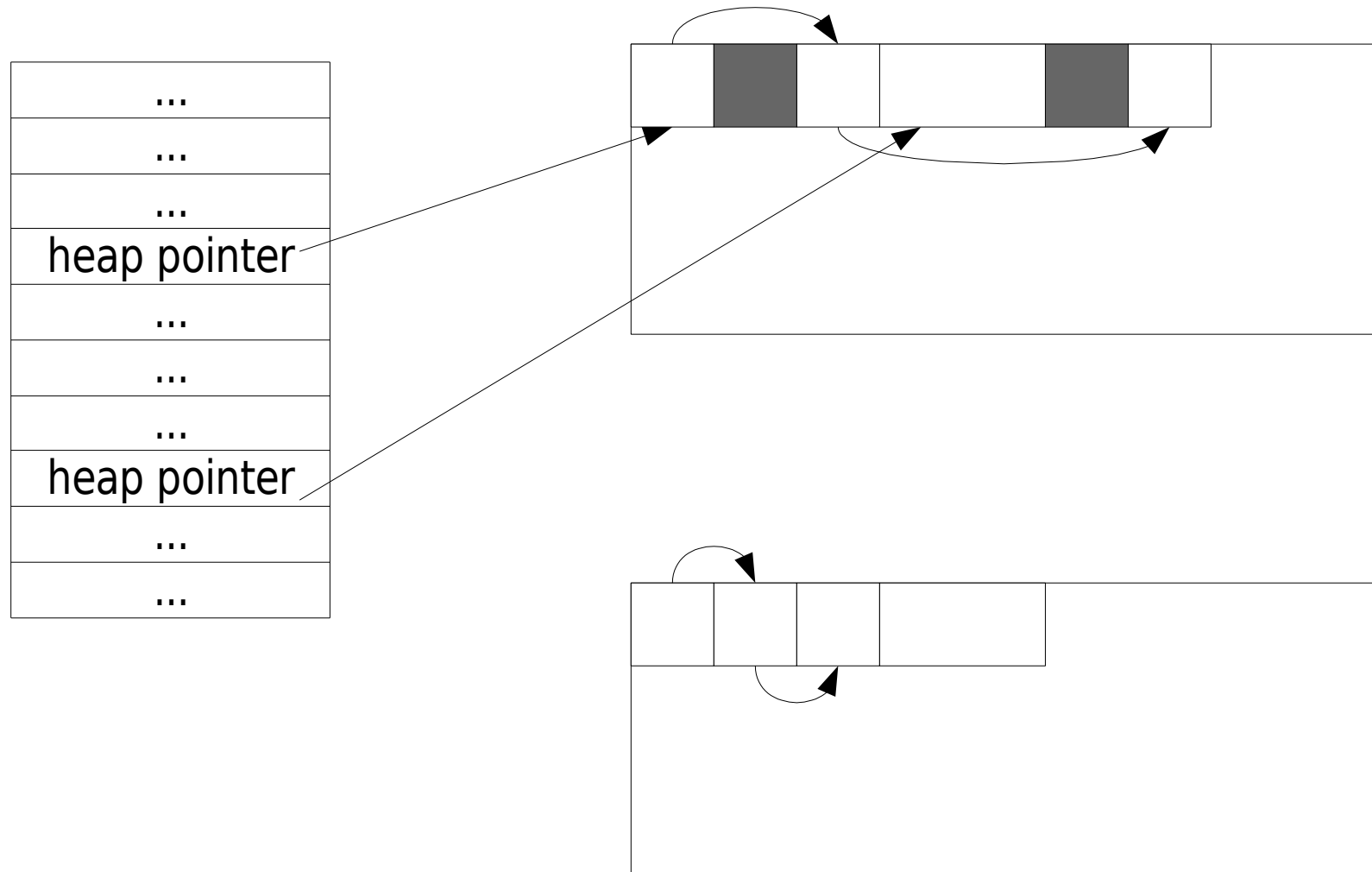


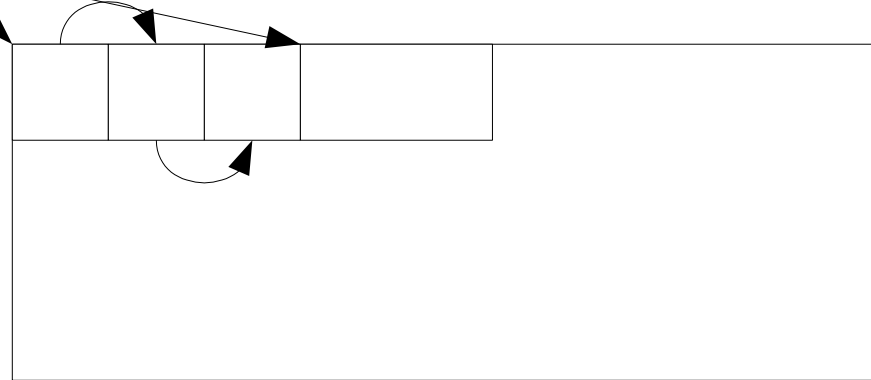
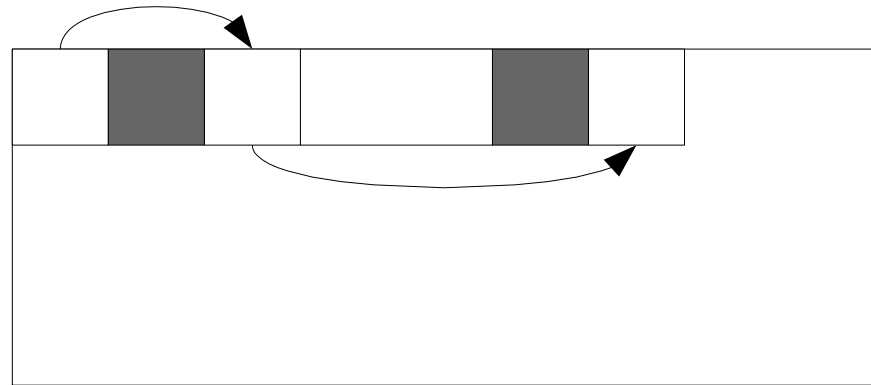
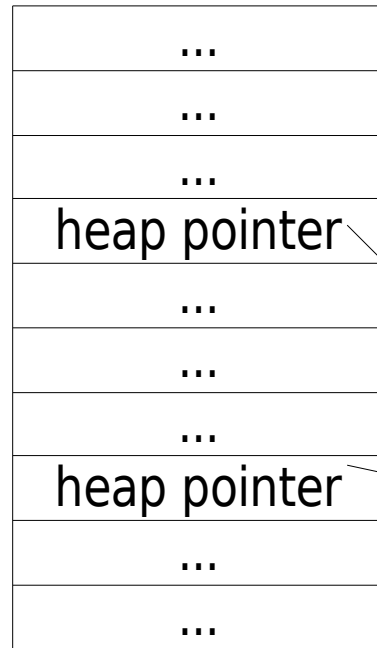
Type-Safe Stack Traversal for Garbage Collection Implementation

Colin Gordon
ScB '08
Honors Thesis Presentation

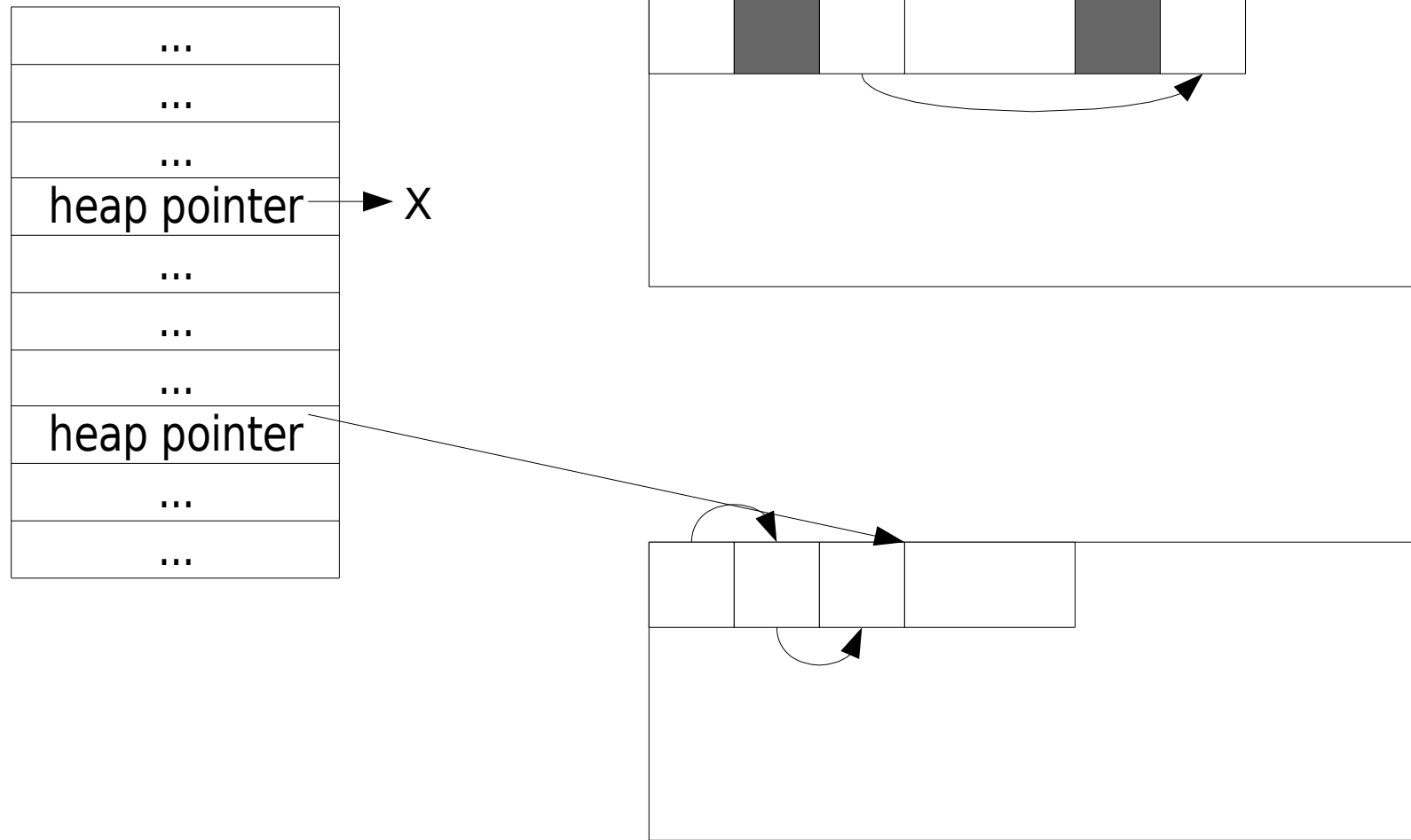
Copying Collector



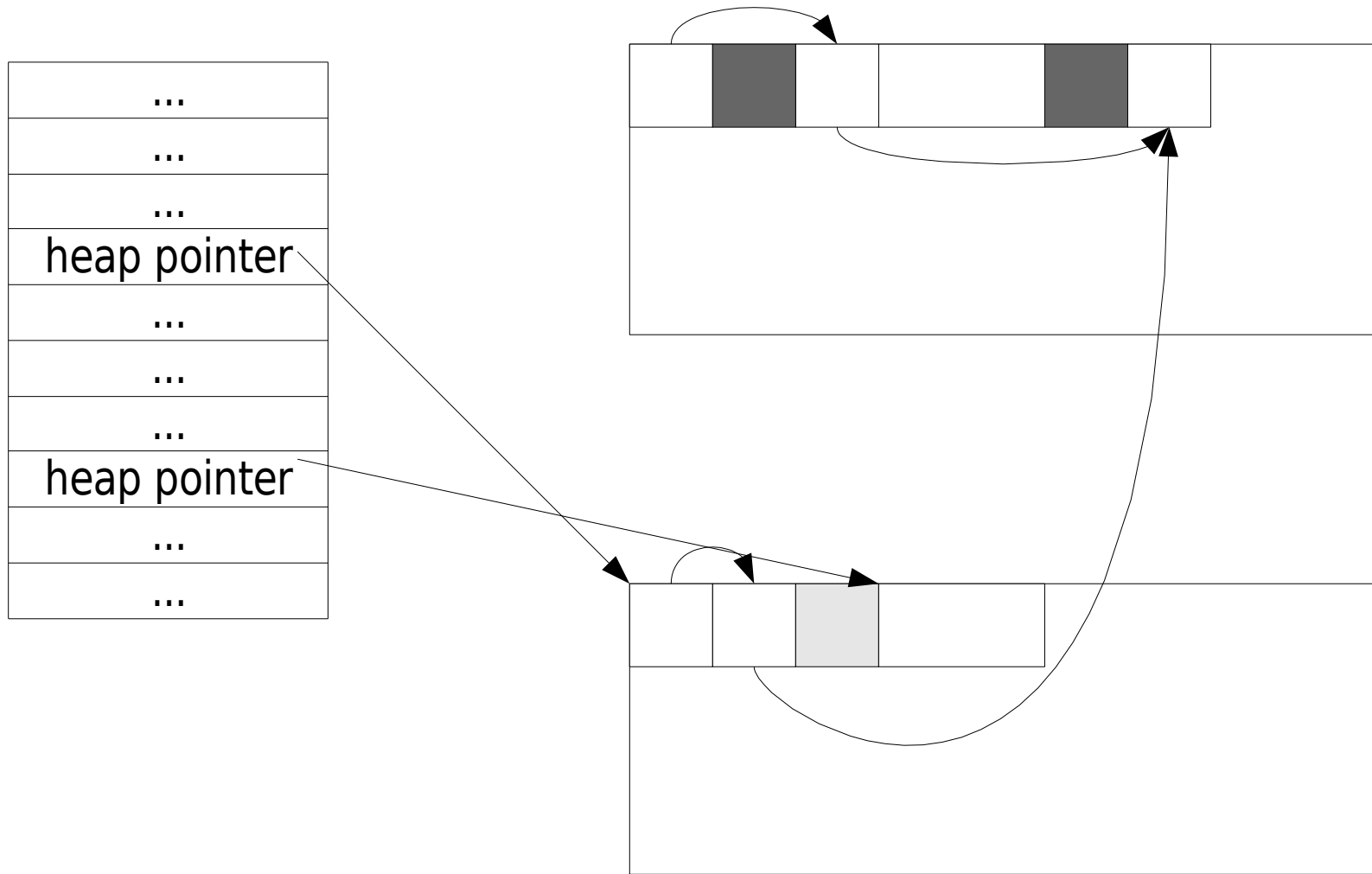
Copying Collector



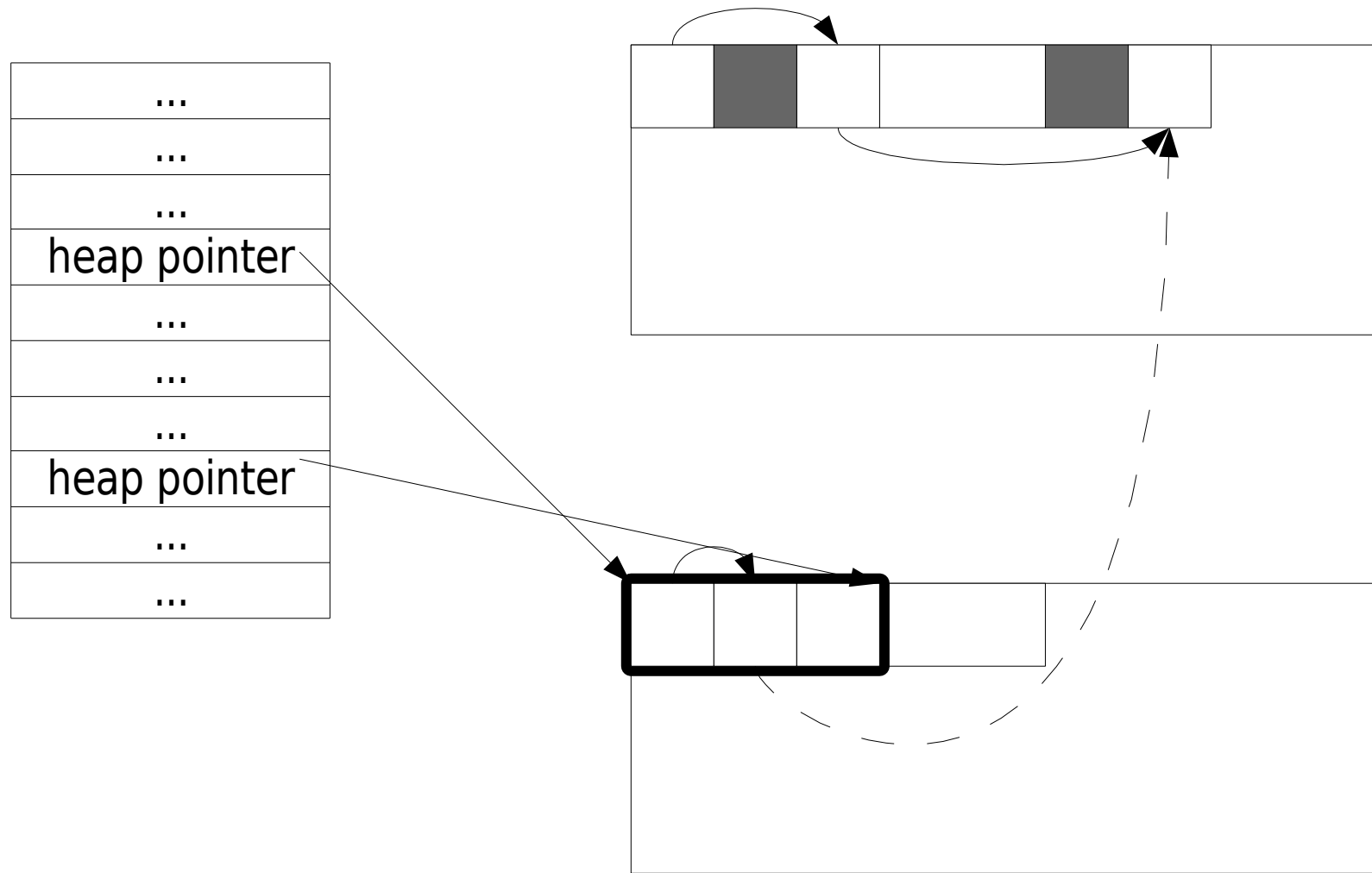
Copying Collector Bugs



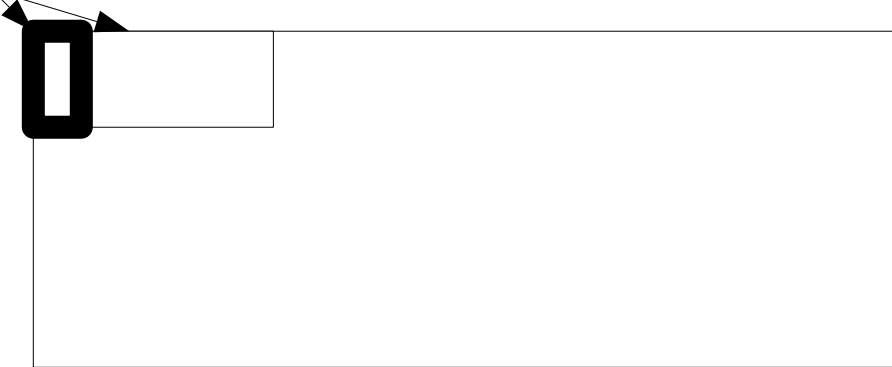
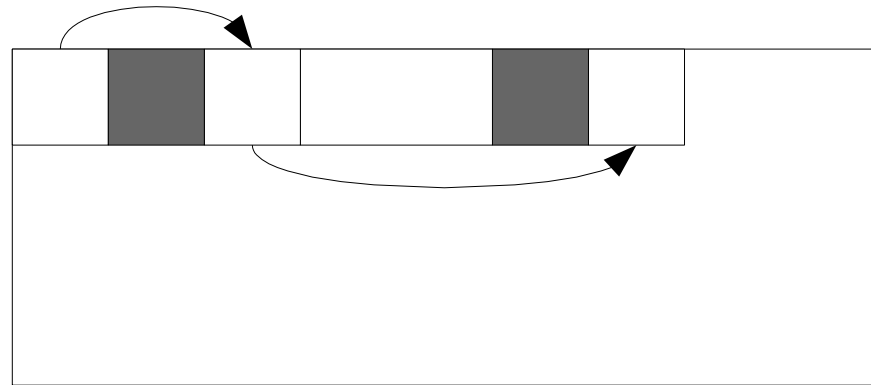
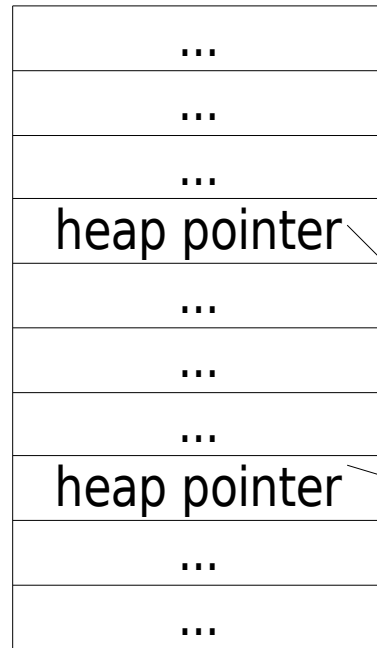
Copying Collector Bugs



Copying Collector Bugs



Copying Collector Bugs



Initial Goal:

Statically Checked
Garbage Collector

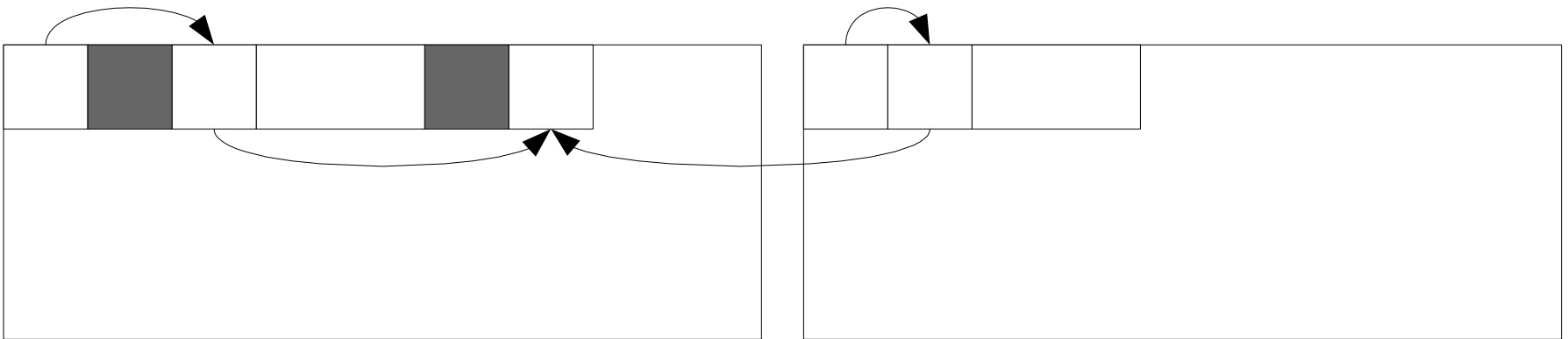
Regions

Wang and Appel '01

```
fun gc[src] roots[src] =  
  let region dest in  
    let val roots' = copy[src,dest] roots  
    in  
      only dest in  
        resume_program roots'
```

Wang and Appel '01

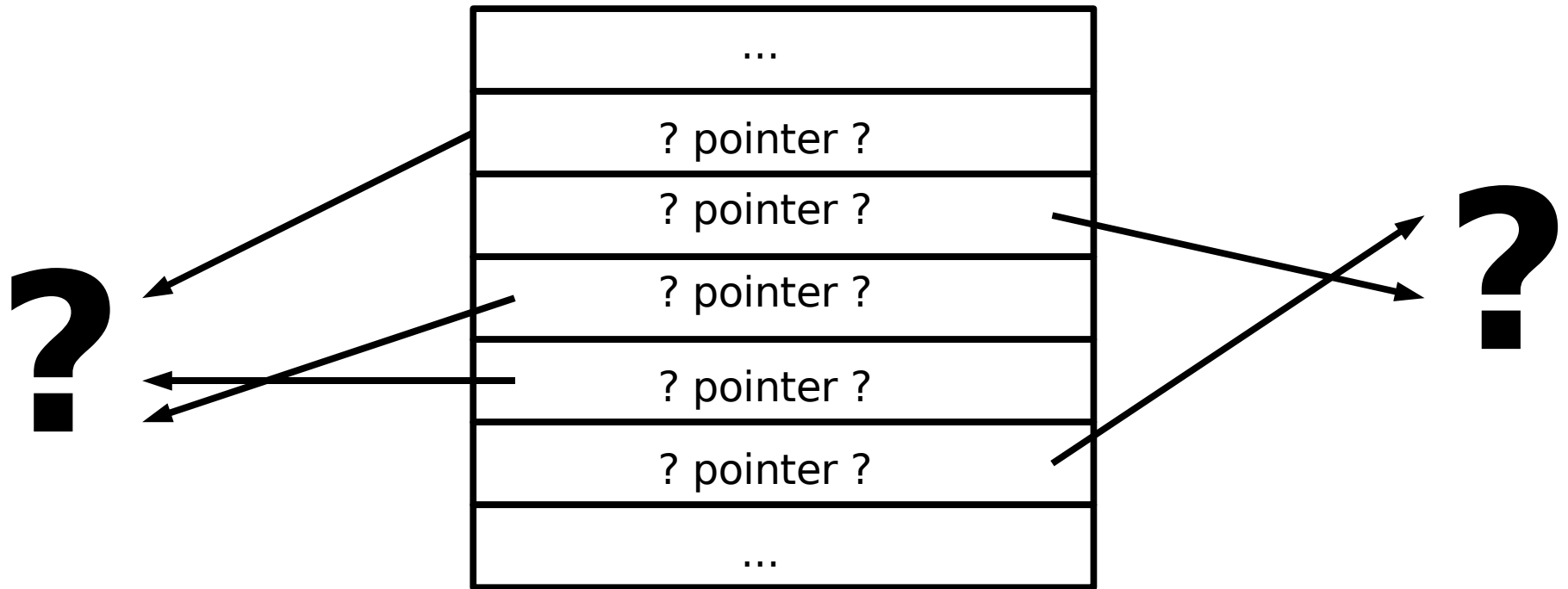
```
fun gc[src] roots[src] =  
  let region dest in  
    let val roots' = copy[src,dest] roots  
    in  
      only dest in  
        resume_program roots'
```



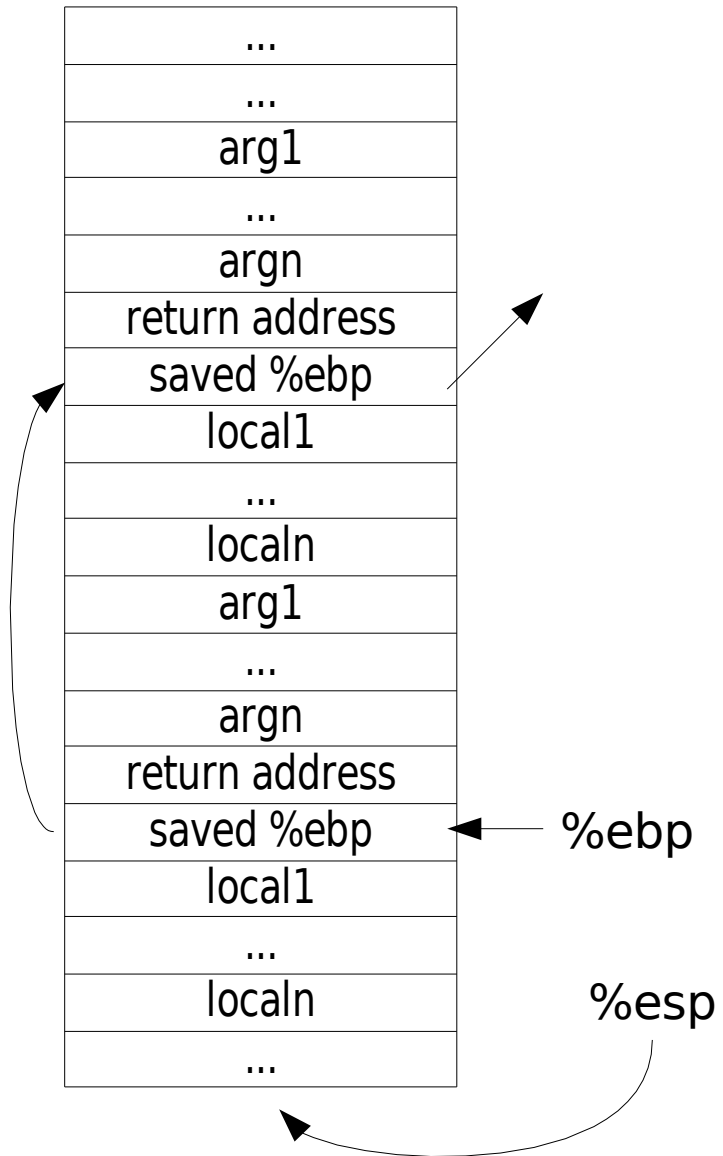
Wang and Appel '01

```
fun gc[src] roots[src] =  
  let region dest in  
    let val roots' = copy[src,dest] roots  
    in  
      only dest in  
        resume_program roots'
```

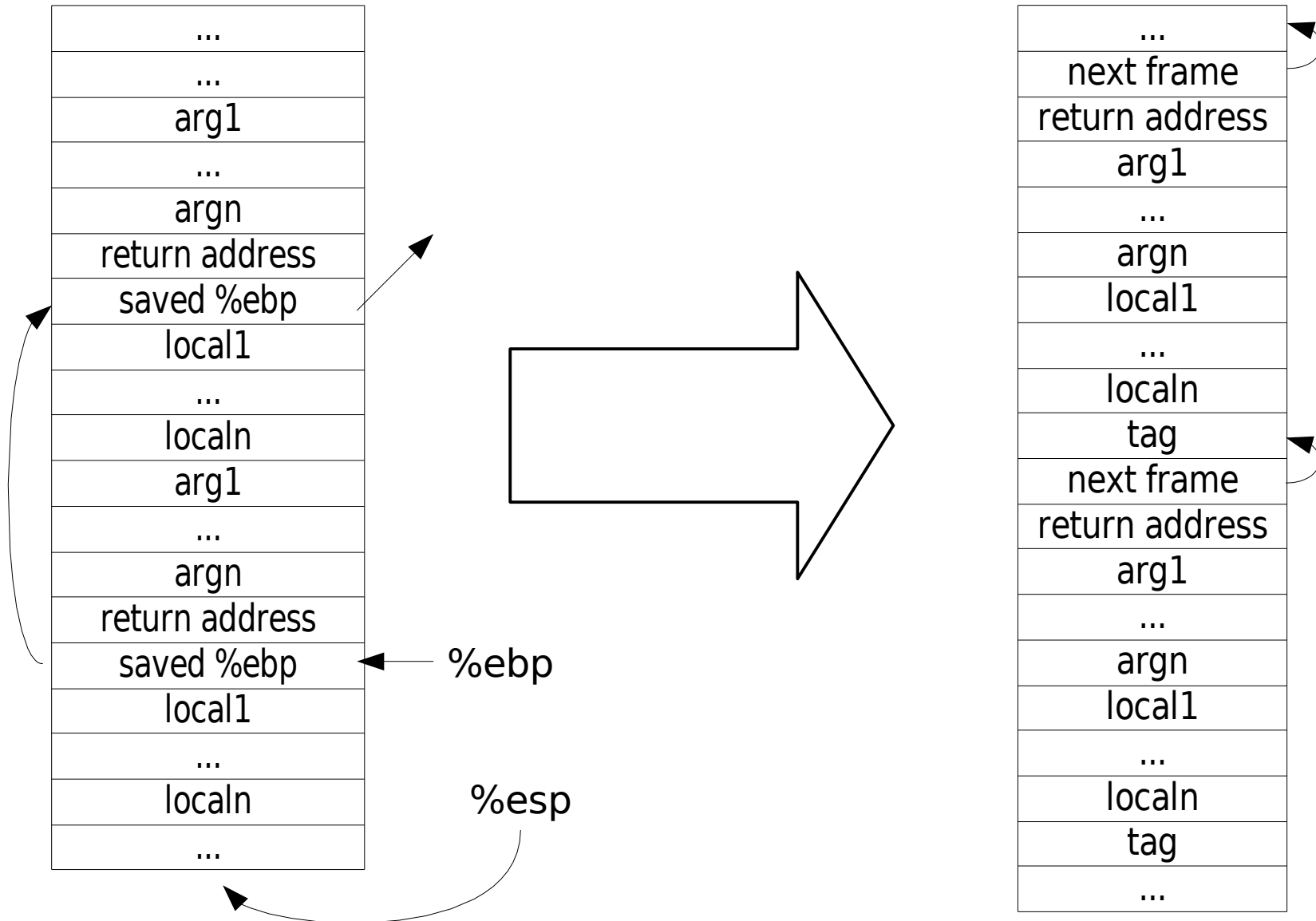
The Problem



Intuition



Intuition



Explicit Frame Structures

...
return address
arg 1
...
local 1: garbage
local 2: garbage

Explicit Frame Structures

...
return address
arg 1
...
local 1: garbage
local 2: garbage

...
return address
arg 1
...
local 1: garbage
local 2: garbage
body ptr
tag1
next frame
return address
arg 1
...

Explicit Frame Structures

...
return address
arg 1
...
local 1: garbage
local 2: garbage

...
return address
arg 1
...
local 1: garbage
local 2: garbage
body ptr
tag1
next frame
return address
arg 1
...

...
return address
arg 1
...
local 1: int list
local 2: garbage

Explicit Frame Structures

...
return address
arg 1
...
local 1: garbage
local 2: garbage

...
return address
arg 1
...
local 1: garbage
local 2: garbage
body ptr
tag1
next frame
return address
arg 1
...

...
return address
arg 1
...
local 1: int list
local 2: garbage

...
return address
arg 1
...
local 1: int list
local 2: garbage
body ptr
tag2
next frame
return address
arg 1
...

New Goal:

Type Safe Garbage Collector
Using the Real Stack

The ML Kit Compiler

GC Frame Types

```
datatype stackframe =
```

```
  ...  
  | frame_makingfirstcall of  
    word * word * int * word * stackframe  
  | frame_makingsecondcall of  
    word * int list * int * word * stackframe  
  ...  
  | lastframe
```

```
    ...  
  [parent tag]  
  [next ptr]  
  [ret addr]  
  [arg]  
  [garbage]  
  [garbage]
```

```
    ...  
  [parent tag]  
  [next ptr]  
  [ret addr]  
  [arg]  
  [garbage]  
  [garbage]  
  [body ptr]  
  [first call tag]  
  [next ptr]
```

```
    ...  
  [parent tag]  
  [next ptr]  
  [ret addr]  
  [arg]  
  [int list ptr]  
  [garbage]  
  [body ptr]  
  [second call tag]  
  [next ptr]
```

...

...

GC Frame Types

```
datatype stackframe =
```

```
  ...  
  | frame_makingfirstcall of  
    word * word * int * word * stackframe  
  | frame_makingsecondcall of  
    word * int list * int * word * stackframe  
  ...  
  | lastframe
```

```
    ...  
  [parent tag]  
  [next ptr]  
  [ret addr]  
  [arg]  
  [garbage]  
  [garbage]
```

```
    ...  
  [parent tag]  
  stackframe  
  word  
  int  
  word  
  word  
  [body ptr]  
  [first call tag]  
  [next ptr]
```

```
    ...  
  [parent tag]  
  stackframe  
  word  
  int  
  int list  
  word  
  [body ptr]  
  [second call tag]  
  [next ptr]
```

...

...

Implementation Challenges

- Propagating Types
- Variant Names
- Live Variables
- Manipulating Region Inference

Recap

- Garbage Collector Bugs
- Prior Work (Regions + Garbage Collection)
- Stack Structure
- Intuition & Explicit Stack Structure
- Implementation