Garbage Collection

Garbage not reachable

Before garbage collection

Free list: 10 → 11 → 14 → 13 → 12

Free list | left | right
---|---|---
0 | 10 | ±
1 | 2 | 5
2 | 3 | 4
3 | c | 4
4 | a | 5
5 | 6 | 7
6 | 3 | b
7 | b | 8
8 | 3 | 9
9 | 11 | ±
10 | 14 | ±
11 | ± | ±
12 | ± | ±
13 | 12 | ±
14 | 13 | ±

null
How can you efficiently find garbage so you can put it on the free list?

We can find what program can reach and then rest is garbage.

Simple idea: reference counts

Could we just keep track of # refs to a cell?
If ref count is 0 then cell is garbage and can be placed on free list.

What's the problem?

Some overhead to maintain bigger problem
breadth-first search

depth-first search can both find all vertices reachable from a given source

Before garbage collection

bFS
dFS

application program variable

Source
Run garbage collection alg when there's nothing on free list.

Important to minimize space used by garbage collection alg.