Generational GC

Segregation by age

Why generational garbage collection

- Simple tracing collectors suffer from a # of drawbacks
  - All active data must be marked or copy
    - Delays caused by GC can be obtrusive
  - Deferred RC can be used to smooth out cost of GC
    - But has high CPU overhead and cannot reclaim cycles
  - Spend much time dealing with long-lived objects
    - Repeatedly copies or marks
- Role of GC?
  - To reclaim garbage
  - Improve locality of system
    - Interact well with virtual memory and cache

Weak generational hypothesis

- Lifetime of many objects is short
  - Studies have shown that as high as 98% of objects can become garbage between GC cycles
- Weak generational hypothesis
  - Most objects die young [Ungar, 1984]
- Insights
  - Concentrate efforts on collecting young objects

Weak generational hypothesis

- Benefits:
  - Collect only a part of the heap
  - Pause time diminish
  - GC becomes feasible for interactive systems
    - "Can I garbage collect while tracking the mouse?"
  - Avoid repeatedly processing objects that remain alive
  - Overall effort of GC can be reduced
  - Locality of the collector can be improved

Weak generational hypothesis

- Cost:
  - System must be able to distinguish old from young objects
  - Cost associated with storing in old object pointer to young object can be very expensive

Generational strategy

- Segregate objects by age into 2 or more regions in heap
  - Each is called a generation
  - Number of generations varies with implementation
  - One scheme: vary number of generations dynamically
- Collect different generations at different frequencies
  - Collect young generation most frequently
    - Minor collection
  - Collect older generations least frequently
    - Major collection

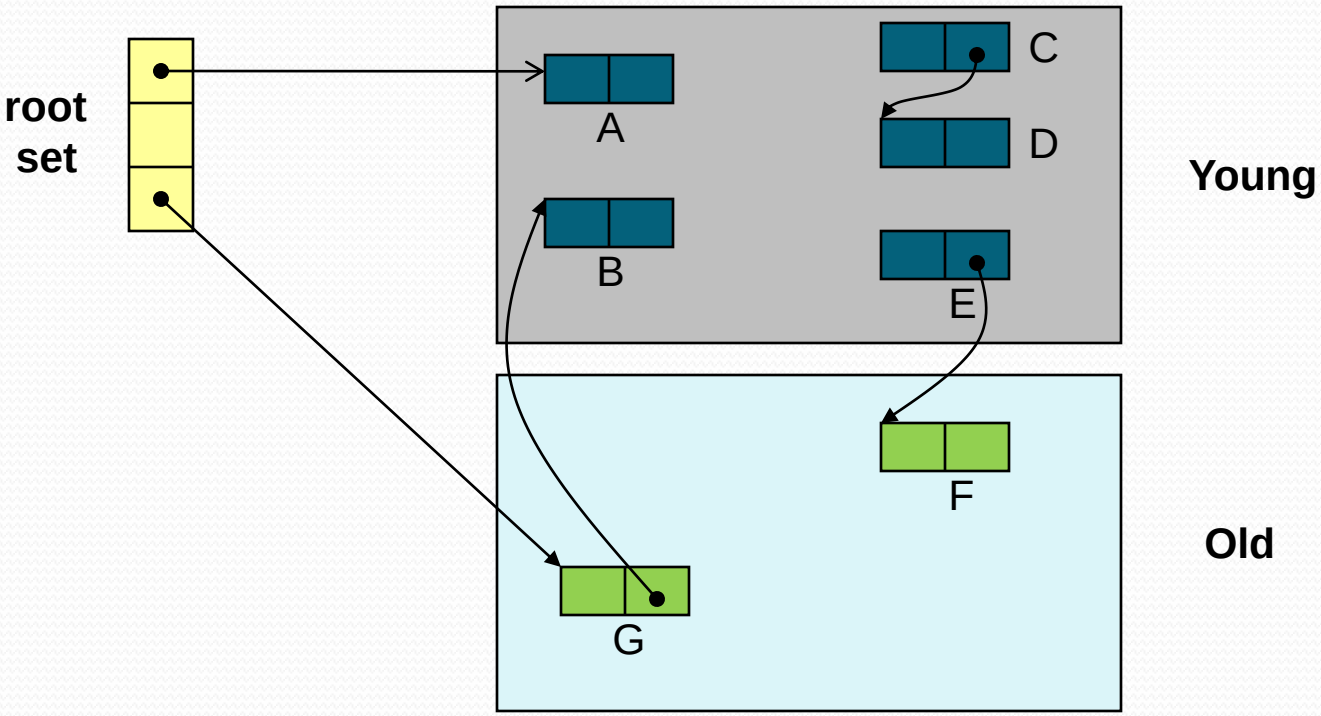
Impact of generational GC

- Often used with incremental collection schemes
- Generational techniques have been very successful
  - Use is widespread
    - All commercial Lisps
    - Modula-3, Glasgow Haskell, commercial SmallTalk systems
    - For many applications today is collection system of choice

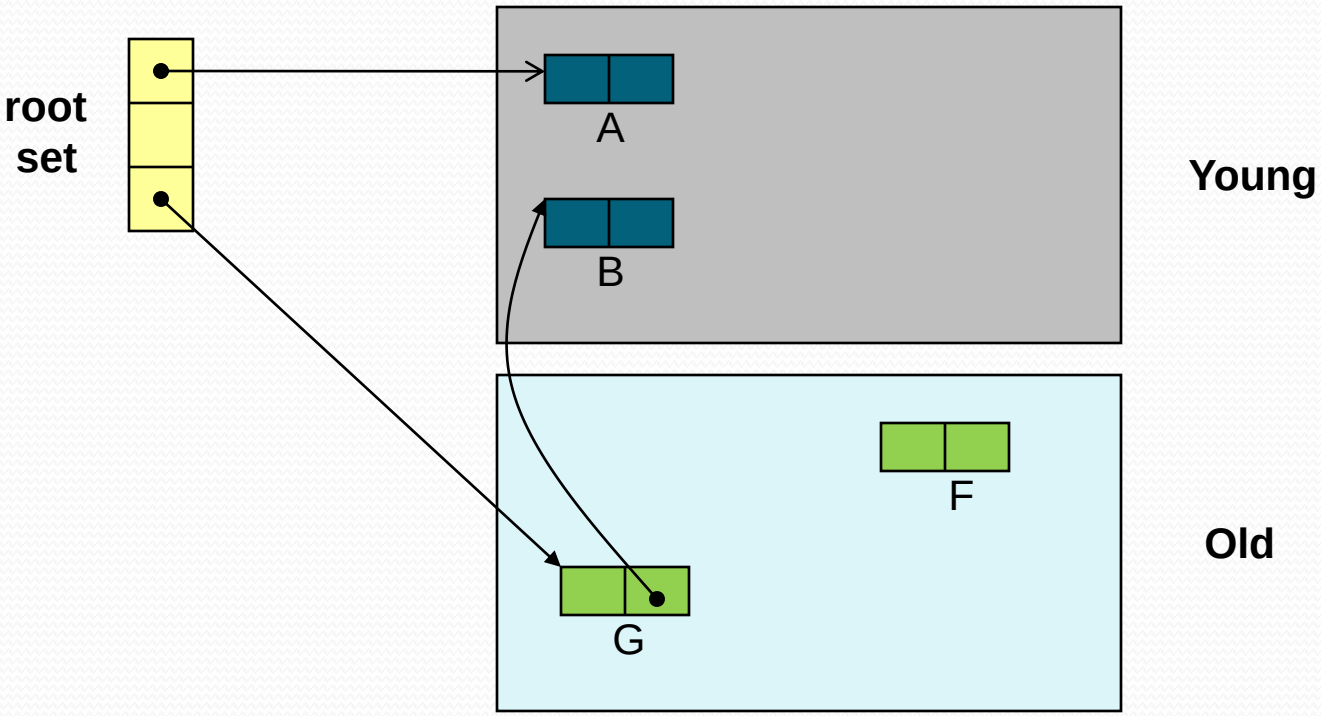
How does generational GC work?

- Objects first allocated in youngest generation
- Objects ***promoted*** to older generation if they survive long enough
  - Youngest generation collected most frequently
    - Weak generational hypothesis
    - Promote objects to older generation
  - After # of minor collections collect older generation
    - Eliminate tenured garbage
    - Collect younger generation when you collect older generation
    - If more than 2 generations, promote objects to even older generation

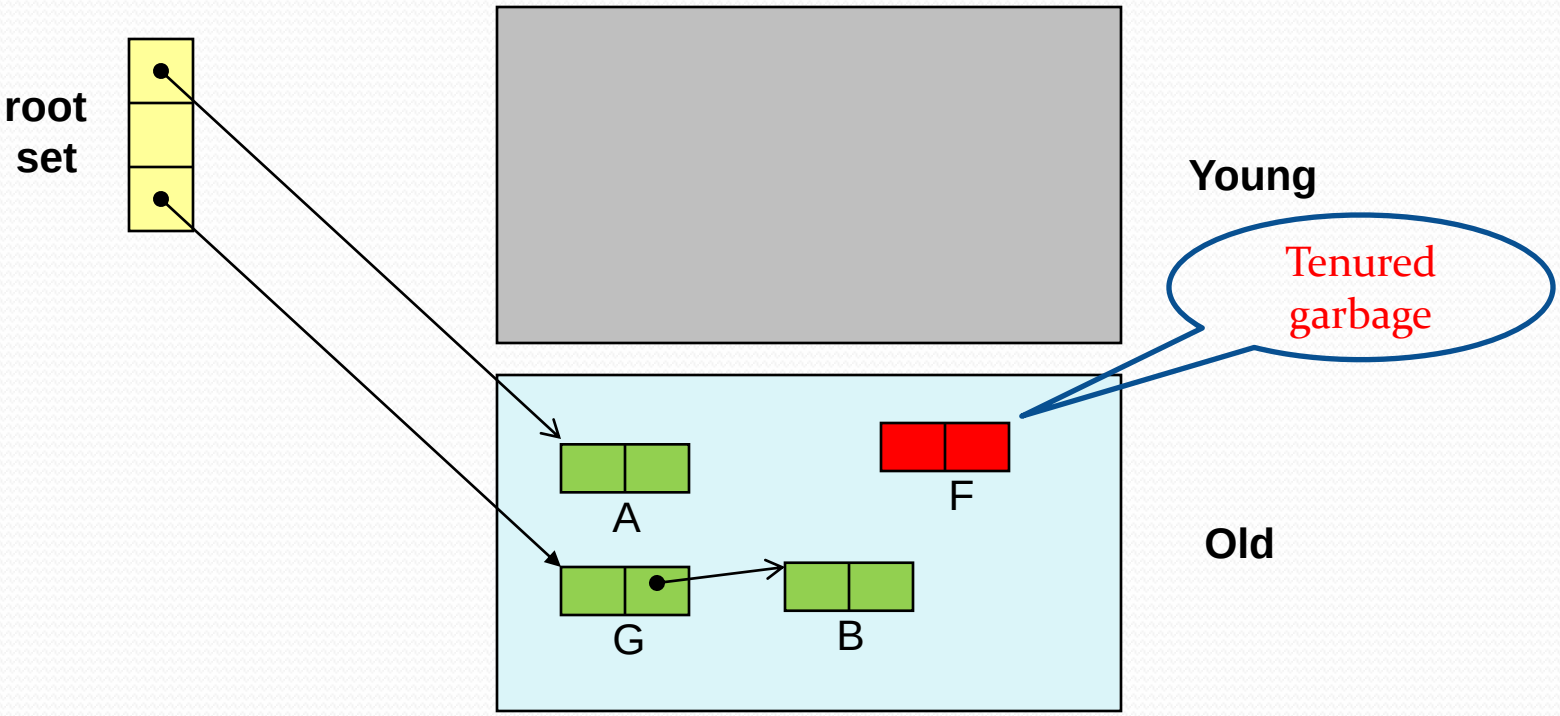
A simple example



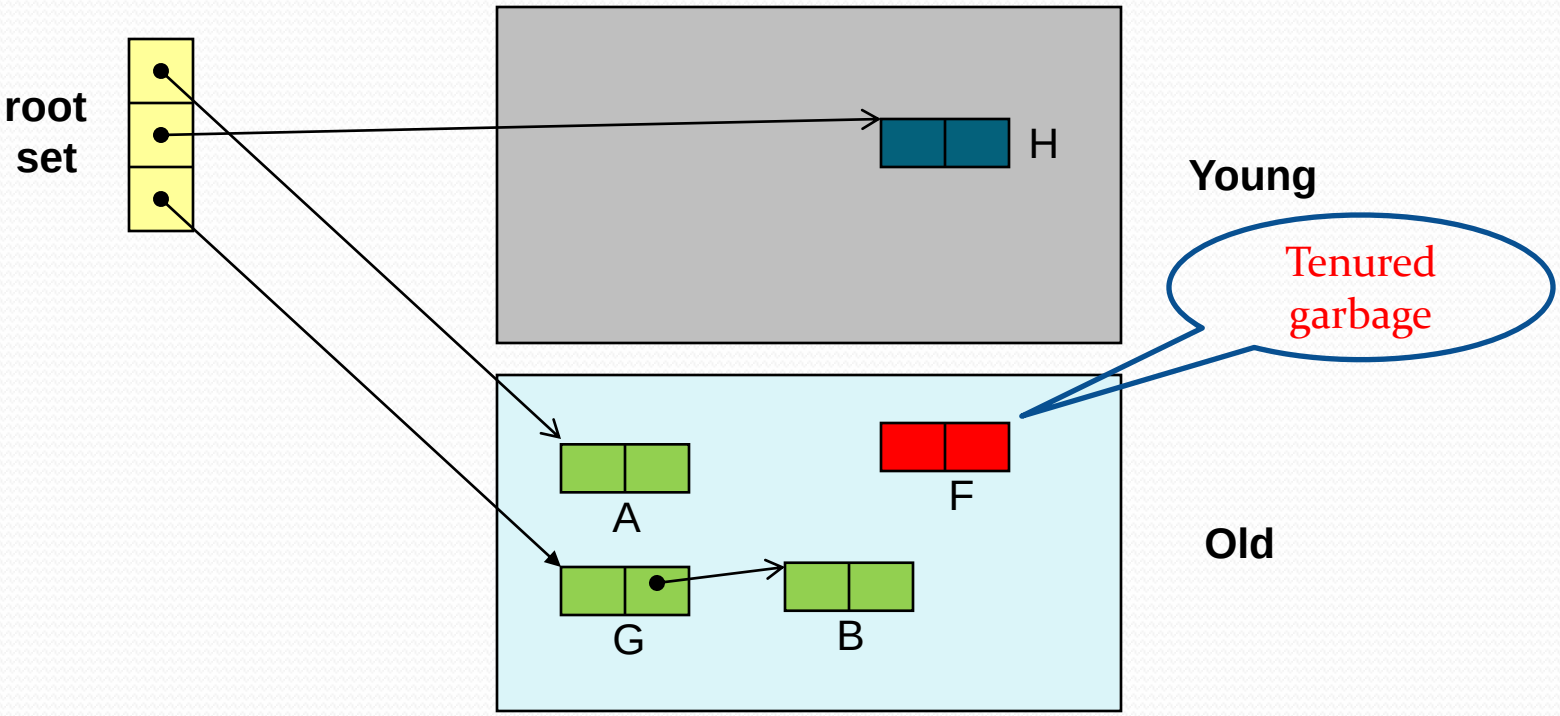
Collect young generation



Promote survivors to old generation



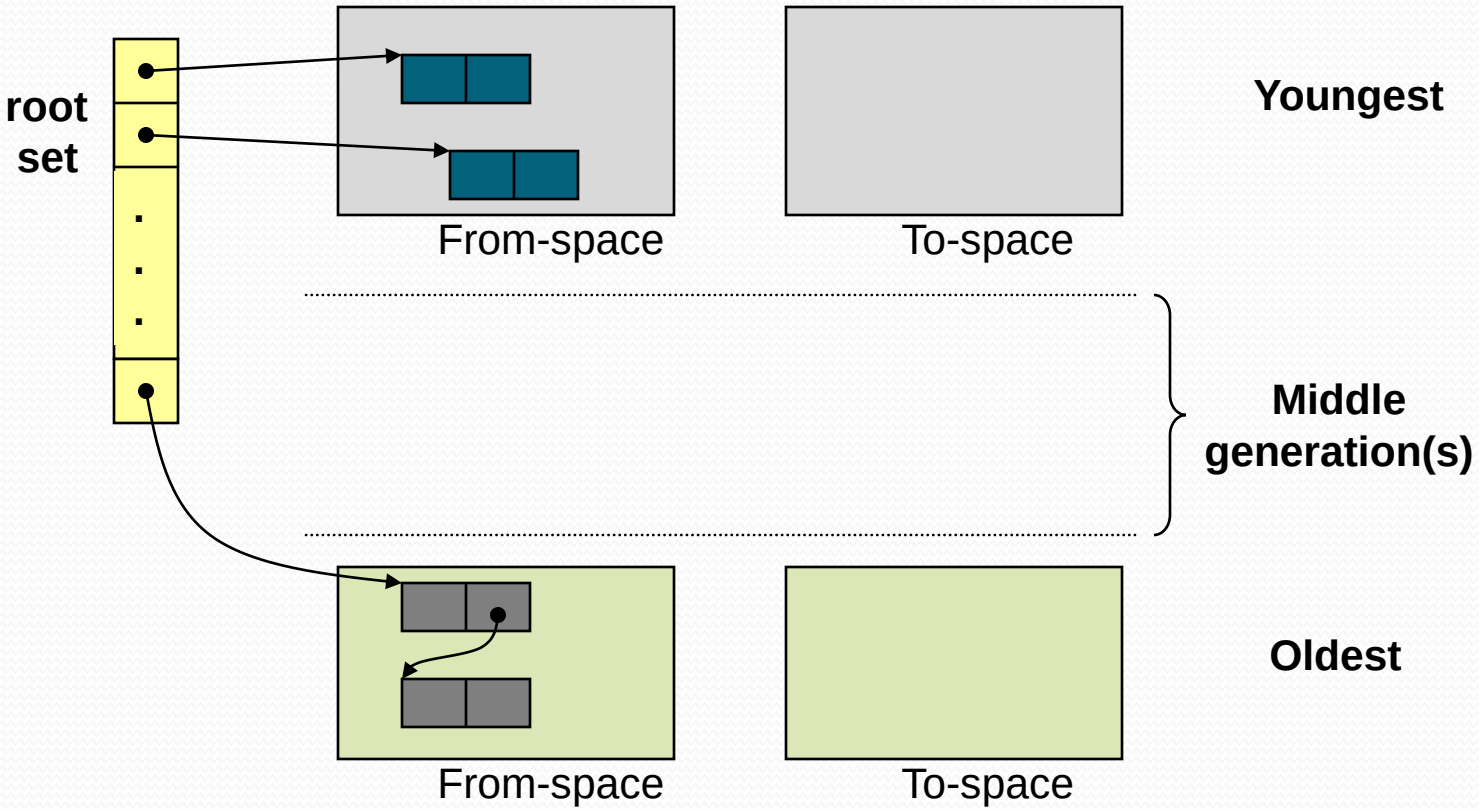
Allocate new object in young gen.



Properties of generational GC

- It is possible to collect younger generations without collecting older generations
- Young objects that survive # of minor collections promoted to older generation
- Minor collection successfully collect all short-lived objects in graph
- Inter-generational pointer (from G to B)
  - G treated as part of root set for minor collection
- Garbage in older generation (***tenured garbage***) cannot be reclaimed by minor collection

Generational copy collector



Other generational observations

- Can determine objects' age by wall-clock-time or by growth rate due to allocation
- Strong generational hypothesis
  - The older an object is the less likely it is to die
    - Not generally true
- Advantages:
  - Pauses for GC are shorter
  - Less data to trace or copy at each collection
  - Total volume of data moved throughout entire program is smaller
  - Effective with short-lived objects

Inter-generational pointers

- Created in 2 ways
  - storing pointers in object (assignment)
  - Object containing pointers promoted to older gen.
- Burden on mutator or collector to track
  - Promotion: can be easily tracked by collector
  - Assignment: need write barrier to trap and record
    - Recall most stores are in local variables
    - Only need to record old-young pointers, Why?
      - They are rare
      - They become roots for minor collection