Copy Collection

- Copy collection is also called **defragment collection**.
- When using copy collection, the **heap is divided into two separate regions**.
- At any point in time, all dynamically **allocated blocks reside in only one of the two regions**. We call this region **active region**. The other region is unoccupied, which is called **inactive region**.
- When the **memory in the active region is exhausted**, the program is suspended and **garbage collection is invoked**.
- Copy collection algorithm **copies** the memory block of all **active references from the active region to the inactive region**.
- **As** each memory block is **copied**, the reference is updated to the new location.
- **After** the copying is completed, the active and inactive region exchange their roles.
- Since the copy collection algorithm **copies only the memory block of active references**, the garbage blocks are left behind.
- The **memory space occupied by the garbage is reclaimed all at once when the active region becomes inactive**.
- **As** the copy collection algorithm **copies blocks from active region to inactive region**, it **stores these blocks in contiguous memory locations**. Therefore, this algorithm automatically defragments the heap.

![Copy Collection Diagram](image-url)
The copy collection algorithm can detect inaccessible rings of nodes and can defragment. However, if an application program has a large memory footprint, the time required to copy all objects can be quite significant. Besides, it requires twice as much memory as the program actually uses and need to suspend the current running program while execution.