CSCI 2132
Software Development

Lecture 30:

Merge Sort with Linked Lists

Instructor: Vlado Keselj

Faculty of Computer Science

Dalhousie University

Previous Lecture

- Heap (Free Store) continued
  - Efficient use of heap
  - Additional allocation functions
- Linked lists
- list.c example (student database) started

Merge Sort with Linked Lists

- Previous code showed typical operations on a linked list
- Let us consider sorting a linked list
- Remember **mergesort**:
  - fast on linked lists (even when compared to quicksort)
- We will modify program to include sorting option

Main Steps in Mergesort

1. Divide: Divide the n-element linked list to be sorted into two sub-lists of n/2 elements each
2. Conquer: Sort the two sub-lists recursively using mergesort
3. Combine: Merge the two sorted sub-lists to produce the sorted answer

Approaches to Divide List in Two

- First approach:
  - Count elements of the list
  - Restart iterating and find half after $n/2$ steps
- Second approach:
  - Iterate two pointers in the same time
  - One moves twice faster than another
  - When faster pointer reaches the end, the first pointer is at half of the list
- We will use the second approach

Mergesort Code for a Linked List

- Available at:

  `~prof2132/public/sortlist.c-blanks`

- Much code shared with `list.c`
- We will focus on sort-related parts