Linked Lists ctd..

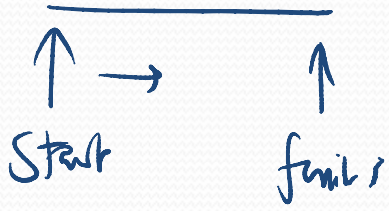
15-123

Systems Skills in C and Unix

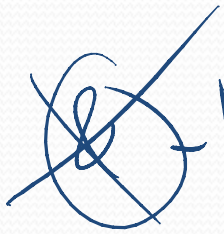
Checking the integrity of a LL

```
int  isSegment(node* start, node* finish){
        assert(start != NULL && finish != NULL);
        if (start==finish) return 1;
        while (start != finish) {
                if (start->next == NULL) return 0;
                start = start->next;
        }
        return 1;
}
```

boolean

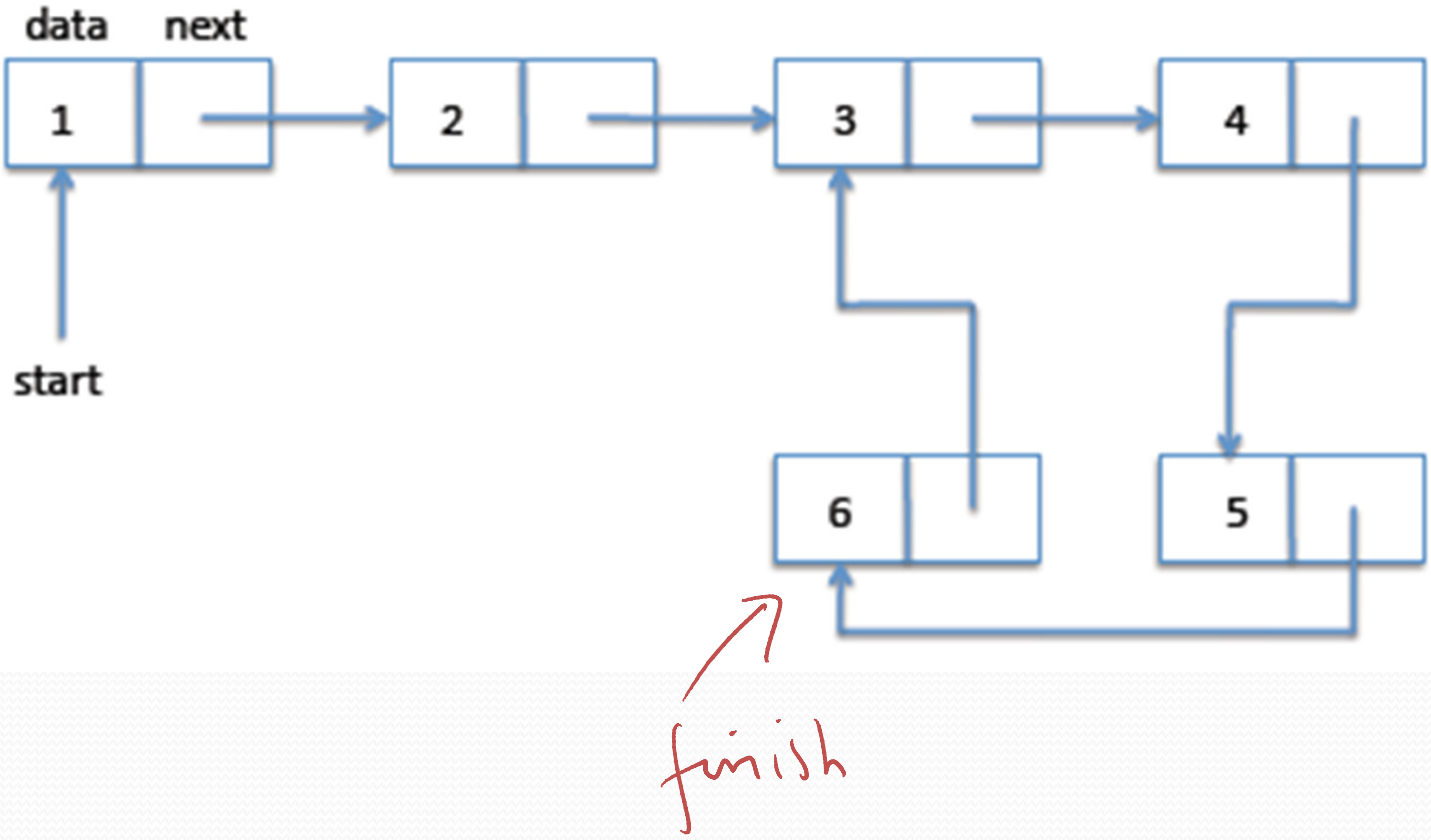


1 & 2 → 0



2 = 0010
&
1 = 0001
0000

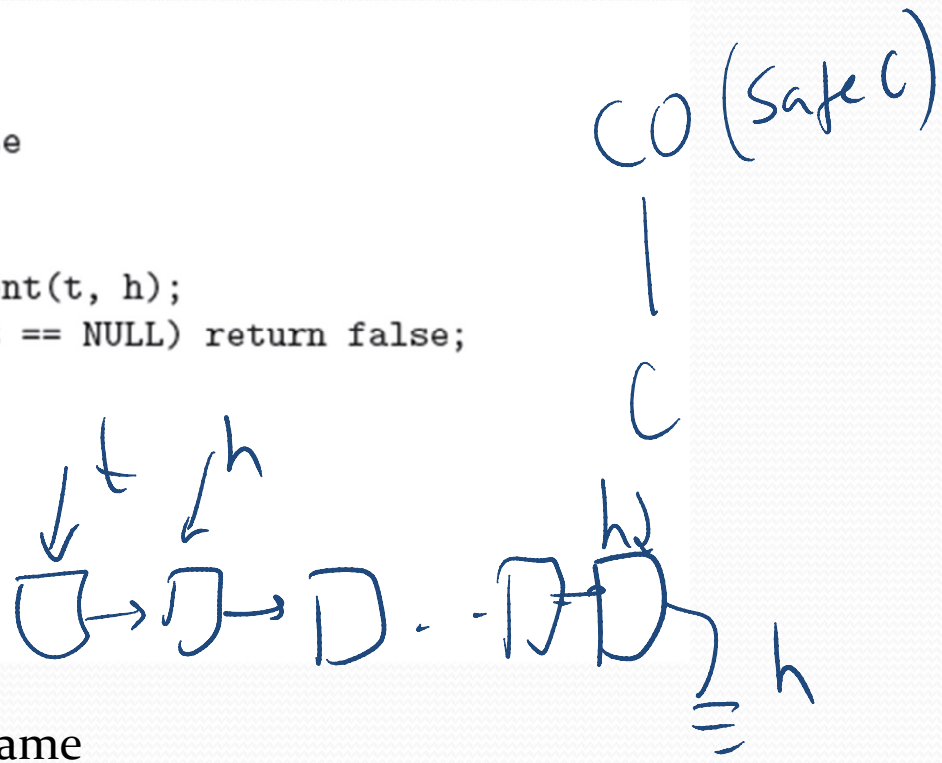
Detecting bad lists



Tortoise and hare algorithm

- Ideal for detecting cycles or circular LL's

```
bool is_circular(list l)
{ if (l == NULL) return false;
  { list t = l;       // tortoise
    list h = l->next; // hare
    while (t != h)
      //@loop_invariant is_segment(t, h);
      { if (h == NULL || h->next == NULL) return false;
        t = t->next;
        h = h->next->next;
      }
    return true;
  }
}
```



This is co code. But idea is the same

Reversing a list in O(n)?

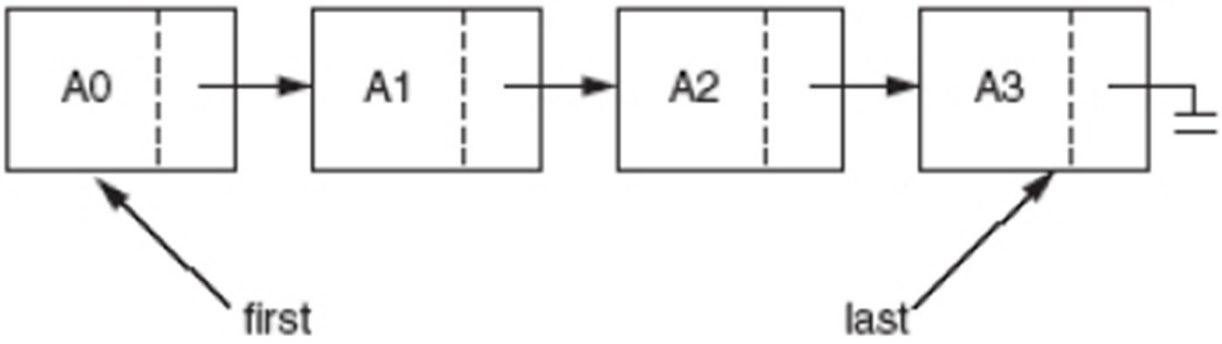
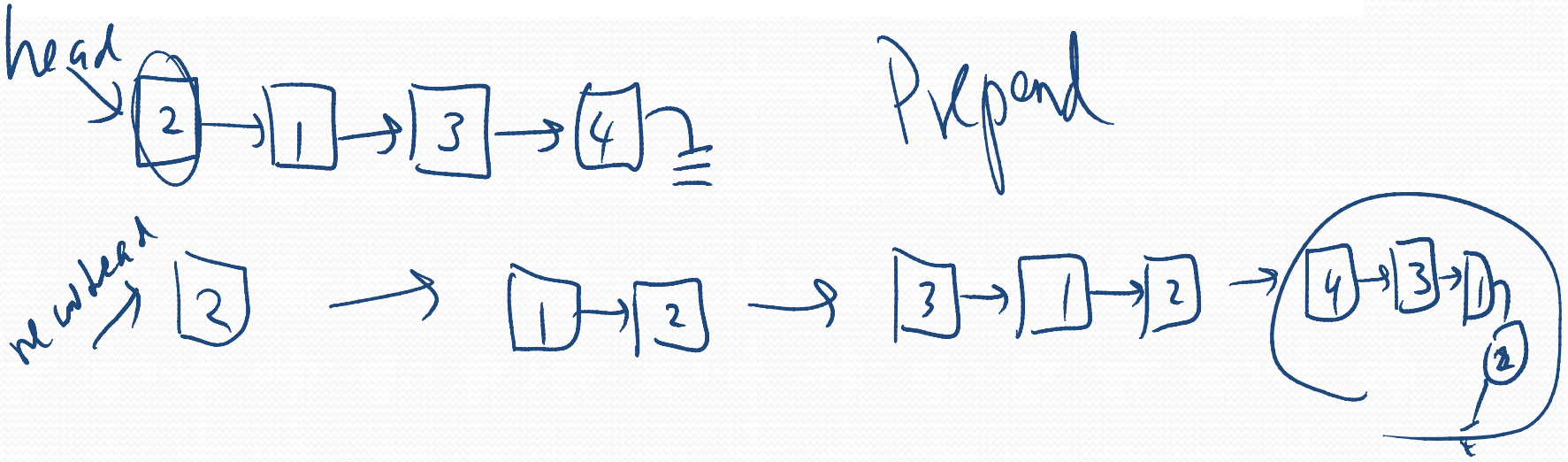
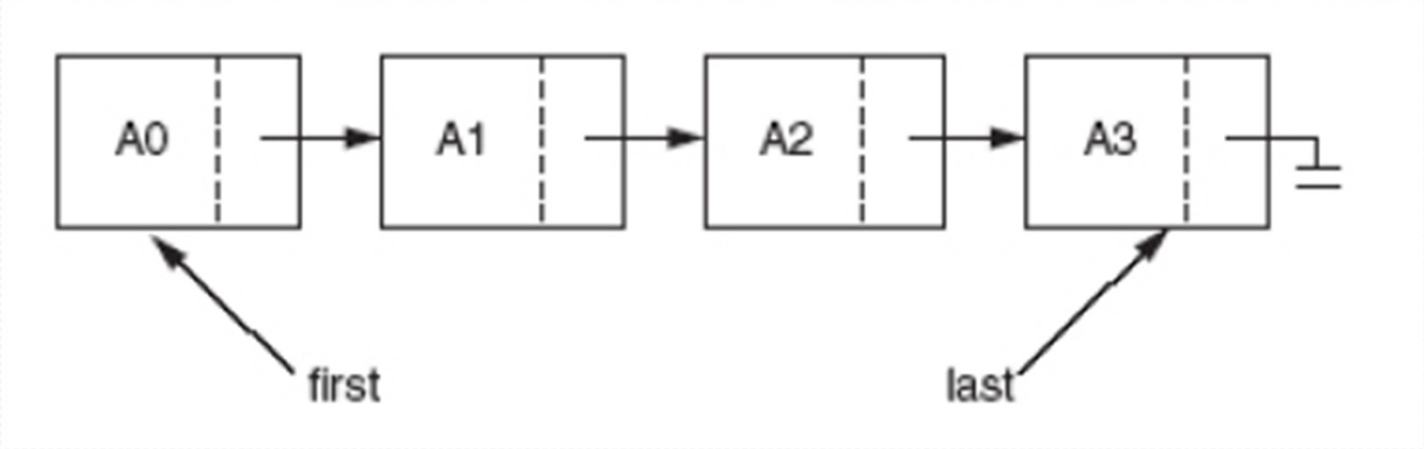


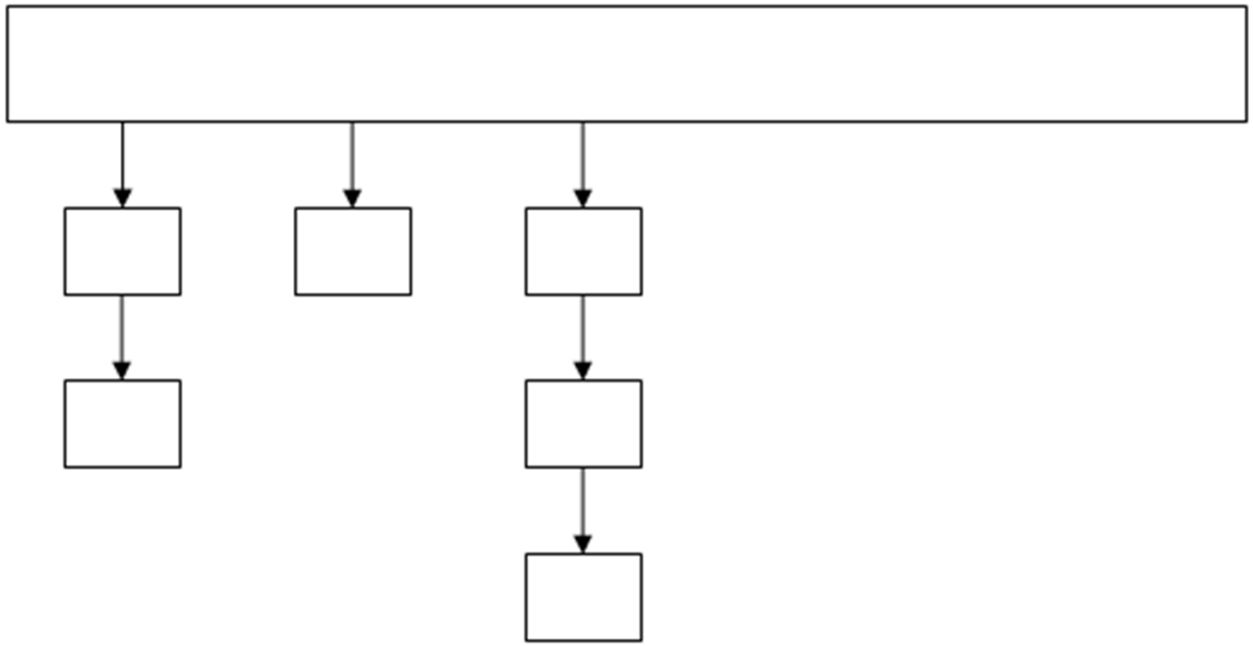
image source: Weiss Data Structures



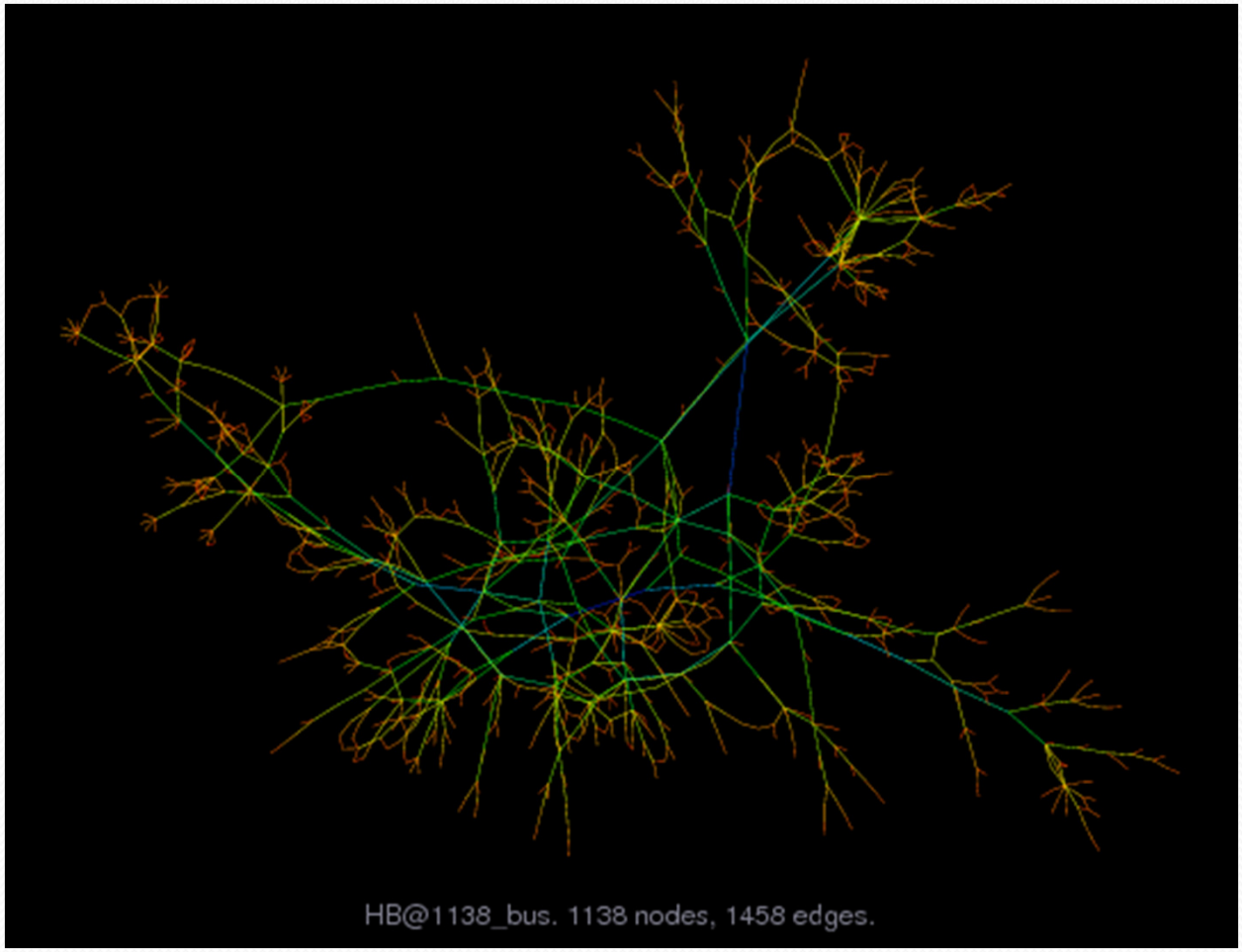
Insert in order



An Array of Linked Lists

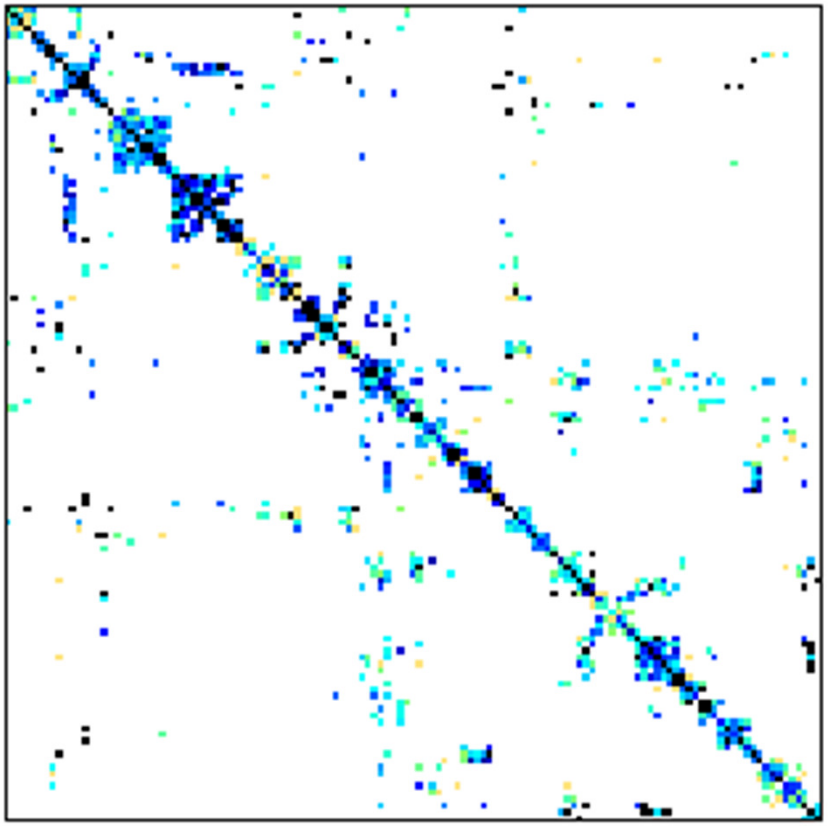


Sparse Matrices



Description: S ADMITTANCE MATRIX 1138 BUS POWER SYSTEM, D.J.TYLAVSKY, JULY 1985.

The matrix



Description: S ADMITTANCE MATRIX 1138 BUS POWER SYSTEM, D.J.TYLAVSKY, JULY 1985.

Storing Sparse Matrices

Structs used

```
typedef struct node {
        int row, column,
        double value;
        struct node*  rowPtr;
        struct node*  colPtr;
} node;
```

Structs used

```
typedef struct matrix {
        node**  rowList;
        node** columnList;
        int  rows, columns;
  } matrix;
```

Coding Examples