

cs140 – algorithms
prof. yi chen

september 5, 2013

9/5/13

insertion sort

```
insertion-sort(A)
1. for j = 2 to length of A
2.   key = A[j]
3.   // insert key into sorted sequence A[1:j-1]
4.   i = j-1
5.   while ((i>0) and (A[i]>key))
6.     A[i+1] = A[i]
7.     i = i-1
8.   endwhile
9.   A[i+1] = key
10. endfor
```

9/3/13

correctness – loop invariants

- ▶ loop invariant
- ▶ initialization
- ▶ maintenance
- ▶ termination

```
insertion-sort(A)
1. for j = 2 to length of A
2.   key = A[j]
3.   // insert key into sorted sequence A[1:j-1]
4.   i = j-1
5.   while ((i>0) and (A[i]>key))
6.     A[i+1] = A[i]
7.     i = i-1
8.   endwhile
9.   A[i+1] = key
10. endfor
```

9/5/13

efficiency – RAM

- ▶ what is the model?

- ▶ how well does it approximate reality?

- ▶ what does it say about insertion sort?

- ▶ best case?
- ▶ worst case?

```
insertion-sort(A)
1. for j = 2 to length of A
2.   key = A[j]
3.   // insert key into sorted sequence A[1:j-1]
4.   i = j-1
5.   while ((i>0) and (A[i]>key))
6.     A[i+1] = A[i]
7.     i = i-1
8.   endwhile
9.   A[i+1] = key
10. endfor
```

9/5/13

Divide-and-conquer

- ▶ divide the problem into a number of subproblems.
- ▶ conquer by solving the subproblems (do this recursively)
- ▶ combine the solutions to the subproblems to give a solution to the overall problem.

9/5/13

Mergesort

```

mergesort(A,p,r)
  if (p ≥ r)
    return
  else
    q = floor((r+p)/2)
    mergesort(A,p,q)
    mergesort(A,q+1,r)
    merge(A,p,q,r)
  endif

```

```

merge(A,p,q,r)
  // create temp arrays L[q-p+1], R[r-q]
  L = A[p:q]
  R = A[q+1:r]
  i=1
  j=1
  for k = p:r
    if (L[i] < R[j]) // + checks for end of array
      A[k] = L[i]
      i = i+1
    else
      A[k] = R[j]
      j = j+1
    endif
  endfor

```

9/5/13

correctness

- ▶ merge subroutine

```

for k = p:r
  if ( L[i] < R[j] )
    A[k] = L[i]
    i = i+1
  else
    A[k] = R[j]
    j = j+1
  endif
endfor

```

- ▶ overall mergesort algorithm

```

if (p ≥ r)
  return
else
  q = floor((r+p)/2)
  mergesort(A,p,q)
  mergesort(A,q+1,r)
  merge(A,p,q,r)
endif

```

9/5/13

efficiency

▶ recurrences

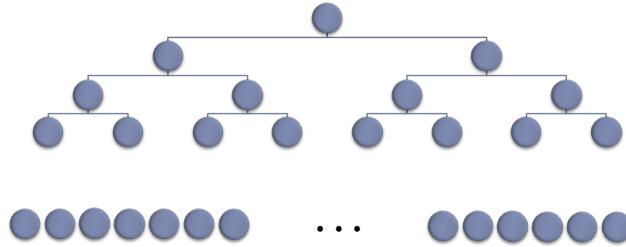
- ▶ recursive case
- ▶ base case

▶ observations

- ▶ floors/ceilings

9/5/13

recursion tree analysis of mergesort



9/5/13