Finding maximum element

Max-elt(A) =

$$\begin{cases} \uparrow & \text{if } |A| = 0 \\ A[1] & \text{if } |A| = 1 \\ A[1] & \text{if } A[1] > \textit{Max-elt}(A[2 \,..\, |A|]) \\ \textit{Max-elt}\,(A[2 \,..\, |A|]) & \text{otherwise} \end{cases}$$

$$T_{\textit{Max-elt}}(n) = \begin{cases} O(1) & \text{if } n = 0,1 \\ O(1) + 2T_{\textit{Max-elt}}\,(n-1) & \text{otherwise} \end{cases}$$

Solution: $O(n)$

Converting associative binary operators to associative *n*-ary operators

- If $\oplus$ is an associative binary operator on values in set S (stored as an array), then f: $S^n \rightarrow S$ is the corresponding operator on arrays of S, where

$f\,(A)$ =

$$\begin{cases} \uparrow & \text{if } |A| = 0 \\ A[1] & \text{if } |A| = 1 \\ A[1] \oplus f\,(A[2 \,..\, |A|]) & \text{otherwise} \end{cases}$$

Algorithm analysis using recurrences

1. Define algorithm iteratively in pseudocode
2. Define recurrence relation that specifies function computed by algorithm
3. Based on (2), define recurrence that relates running time of algorithm to size of input
4. Solve recurrence relation in O, Ω, θ notation

Example problem: linear search

1. Iterative version:

```
Search(A, key)
i ← 1
While i ≤ |A|
    if A[ i ] = key return true
    i ← i + 1
Return false
```

2. Recursive definition:

$$Search(A, key) = \begin{cases} False & \text{if } |A| = 0 \\ True & \text{if } A[1] = key \\ Search(A[2 .. |A|], key) & \text{otherwise} \end{cases}$$

3. Time recurrence
   - It takes one step to test size of array,
   - plus one to compare $A[1]$ to *key*,
   - plus the number of steps to search the rest of the array, starting at 2

$$T_{Linear}(n) = \begin{cases} 3 & \text{if } n \leq 1 \\ 1 + T_{Linear}(n-1) & \text{otherwise} \end{cases}$$

4. Solution to time recurrence

   $O(n)$

Recurrences and time analysis

- Recurrences are function definitions
- Their form is different from pseudocode
  - There are curly braces in recurrences
  - In recurrences, the return value goes on the left and the IF goes on the right.
- In running-time recurrences, the solution is a big-O expression.