PARALLEL PROCESSING

Mohammed Alabdulkareem

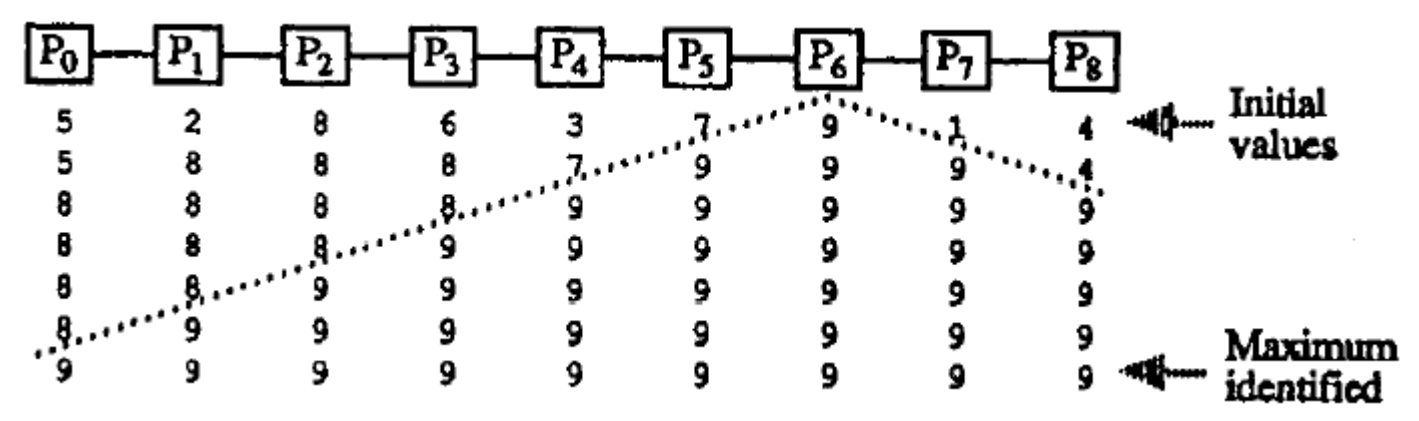
kareem@ksu.edu.sa

Office 2247

4

ALGORITHMS FOR LINEAR ARRAY

- **Semigroup** (an example)
  - At each step each processor sends its MAX to left and right neighbors.
  - Each processor find the MAX =max(left, own, right) and set the local MAX.
  - This takes p-1 communication steps , and the same for comparison (2 (p-1)).

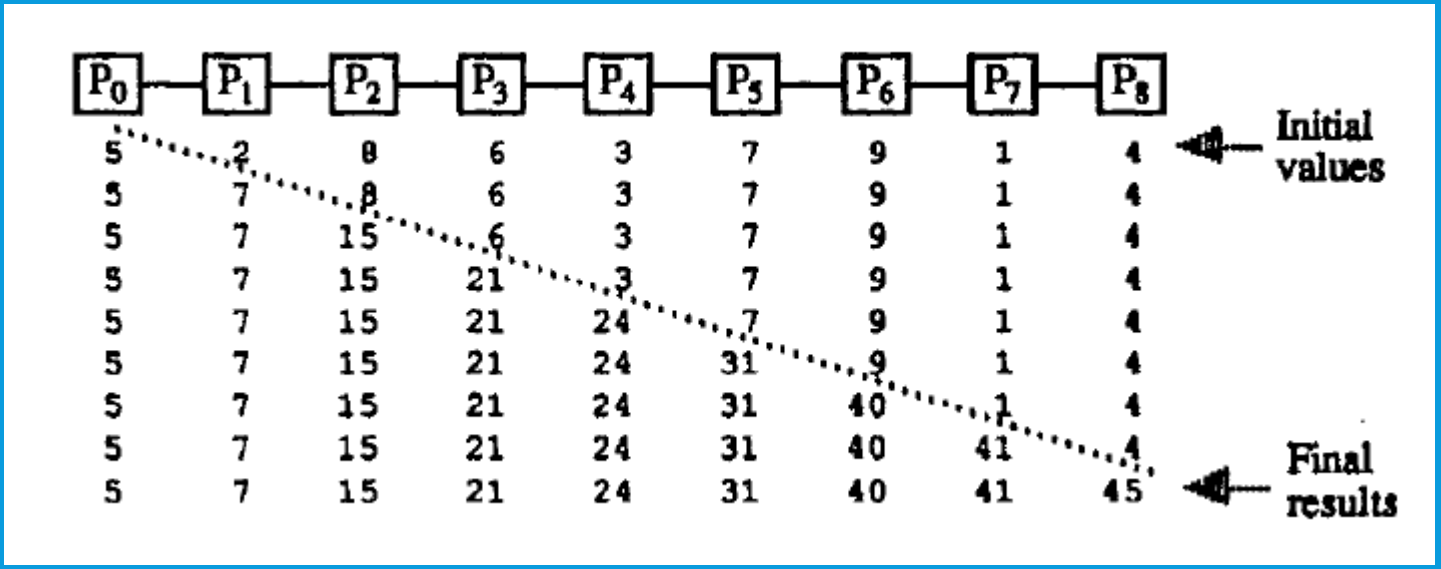


ALGORITHMS FOR LINEAR ARRAY

- For a general semigroup computation, the processor at the left end of the array (the one with no left neighbor) becomes active and sends its data value to the right.
- On receiving a value from its left neighbor, a processor becomes active, applies the semigroup operation $\otimes$ to the value received from the left and its own data value, sends the result to the right, and becomes inactive again.
- This wave of activity propagates to the right, until the rightmost processor obtains the desired result.
- The computation result is then propagated leftward to all processors.
- Total of $2p - 2$ communication steps.

ALGORITHMS FOR LINEAR ARRAY

- Parallel prefix



ALGORITHMS FOR LINEAR ARRAY

- Assume that we want the *i*th prefix result to be obtained at the *i*th processor, $0 \leq i \leq p - 1$.
- The general semigroup algorithm described earlier performs a semigroup computation first and then does a broadcast of the final value to all processors.
- Thus, we already have an algorithm for parallel prefix computation that takes $p - 1$ communication/combining steps.
- A variant of the parallel prefix computation, in which Processor *i* ends up with the prefix result up to the $(i - 1)$th value, is sometimes useful.
- This diminished prefix computation can be performed just as easily if each processor holds onto the value received from the left rather than the one it sends to the right.

ALGORITHMS FOR LINEAR ARRAY

- Broadcast on linear array of p processors
  - Processor *i* wants to broadcast a value (*a*) to all processors
  - Processor *i* sends an rbcast(*a*) message to its right neighbor and
  - lbcast(*a*) message to its left neighbor

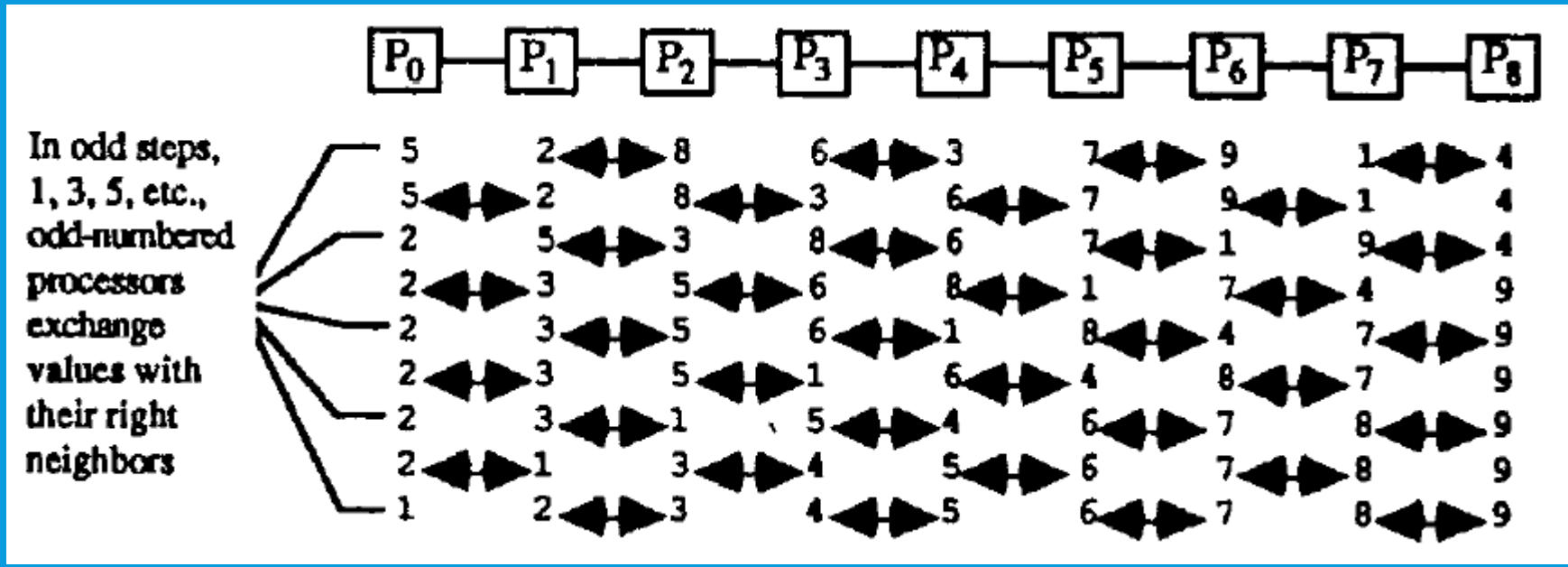
  - Processor receiving an rbcast(a) message copies the value a and forwards the message to its right
  - The same with lbcast(a)

- worst-case number of communication steps for broadcasting is $p - 1$.

ALGORITHMS FOR LINEAR ARRAY

- Odd–even transposition sort
  - Assume each processor holds one number
  - In an odd-numbered step, odd-numbered processors compare values with their even-numbered right neighbors. Exchange values if needed
  - in an even-numbered step, even-numbered processors compare (and exchange as needed) values with their right neighbors
- Total number of steps is p

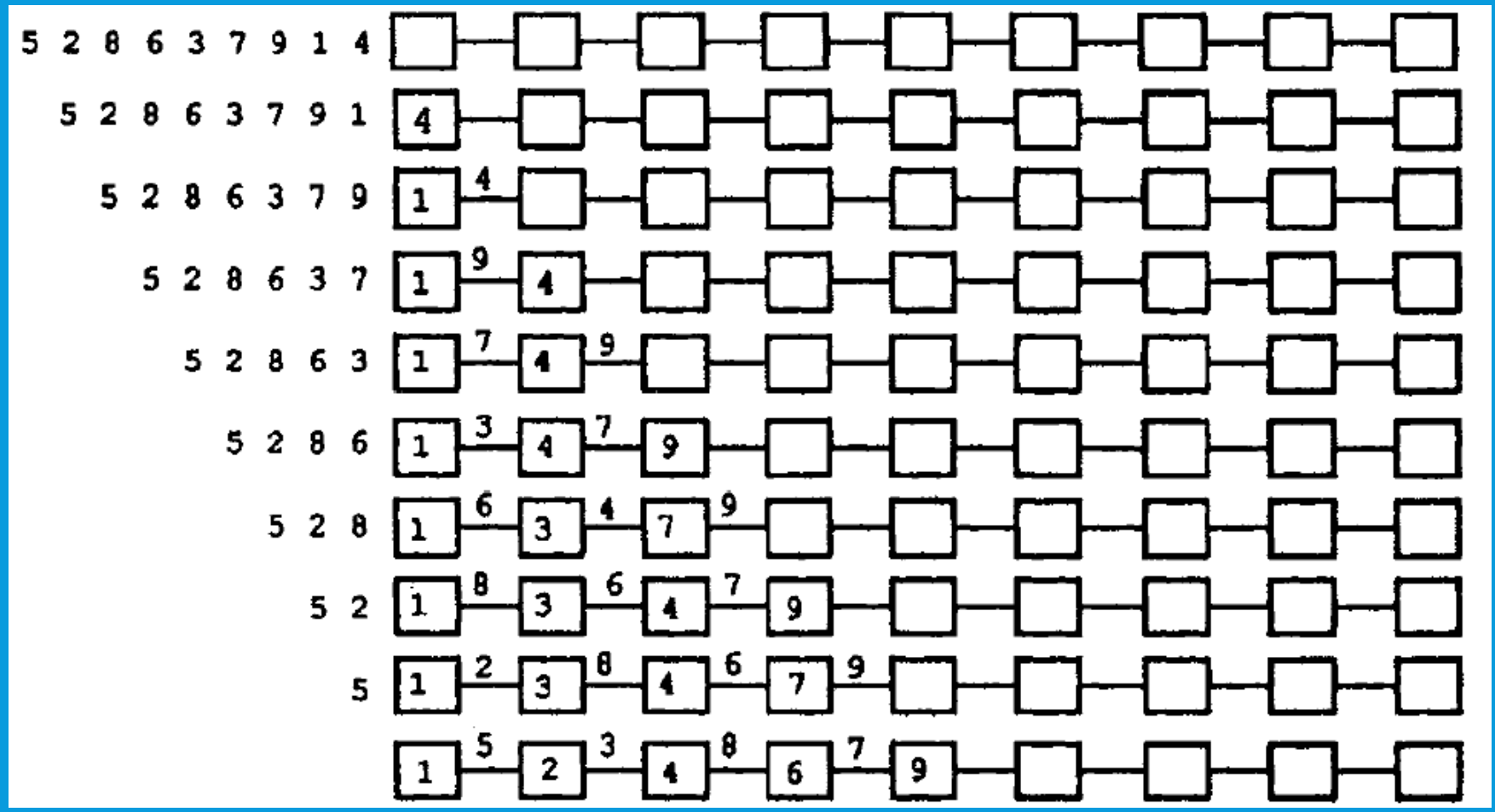
ALGORITHMS FOR LINEAR ARRAY



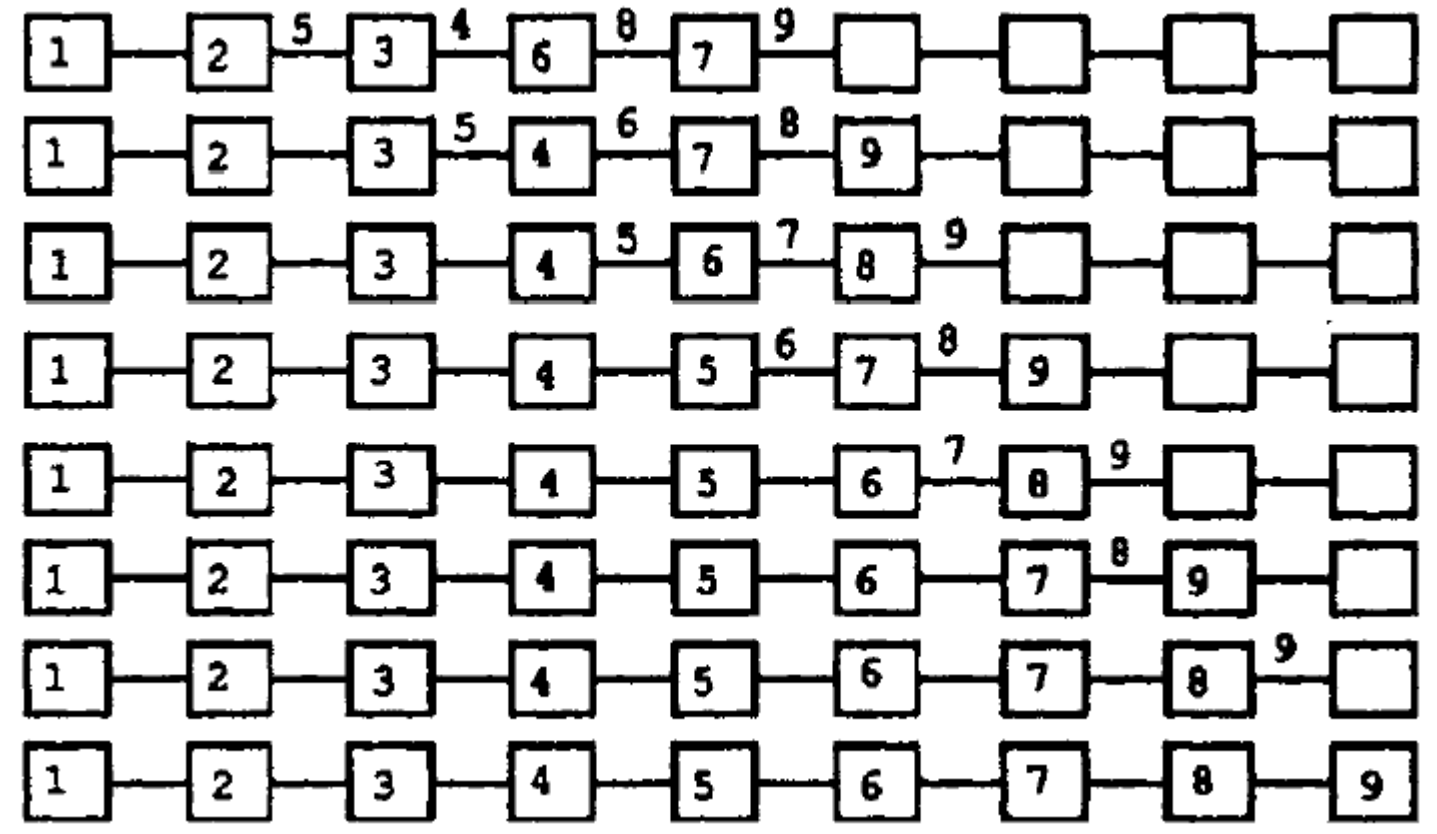
ALGORITHMS FOR LINEAR ARRAY

- Sorting with the keys input sequentially from the left
  - *p* keys are input one at a time from the left
  - At each step processor *i* compares the received value with the value stored in its local register the smaller will be kept the larger is passed to the right neighbor
  - Once all *p* inputs have been received, we must allow $p - 1$ additional communication cycles for the key values that are in transit to settle into their respective positions.

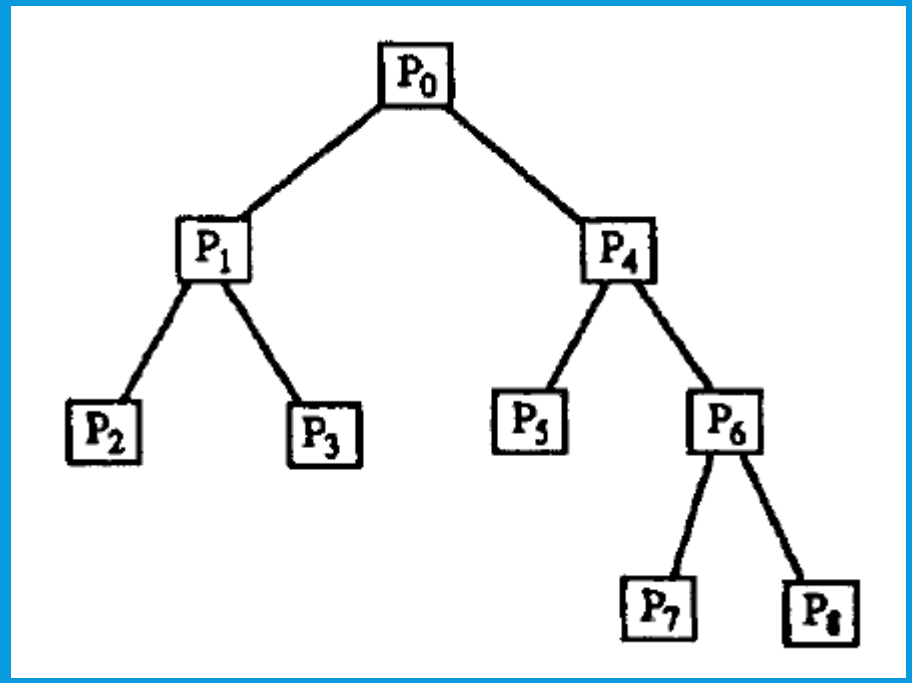
ALGORITHMS FOR LINEAR ARRAY



ALGORITHMS FOR LINEAR ARRAY



ALGORITHMS FOR BINARY TREE

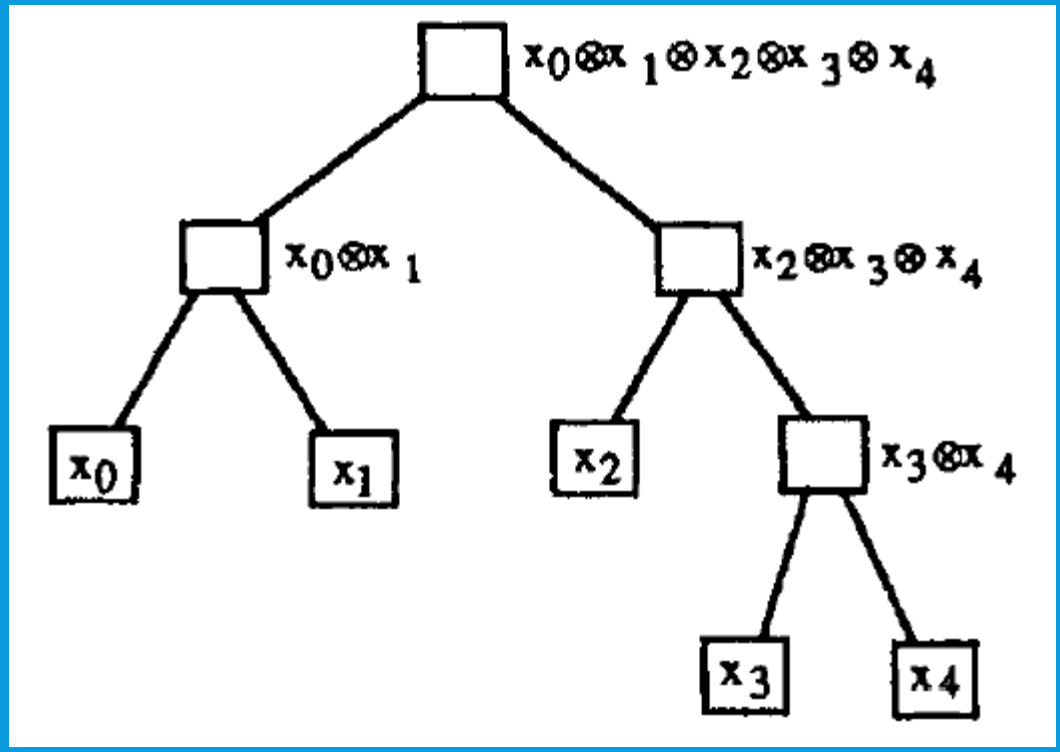


ALGORITHMS FOR BINARY TREE

جامعة الملك سعود
King Saud University
1957

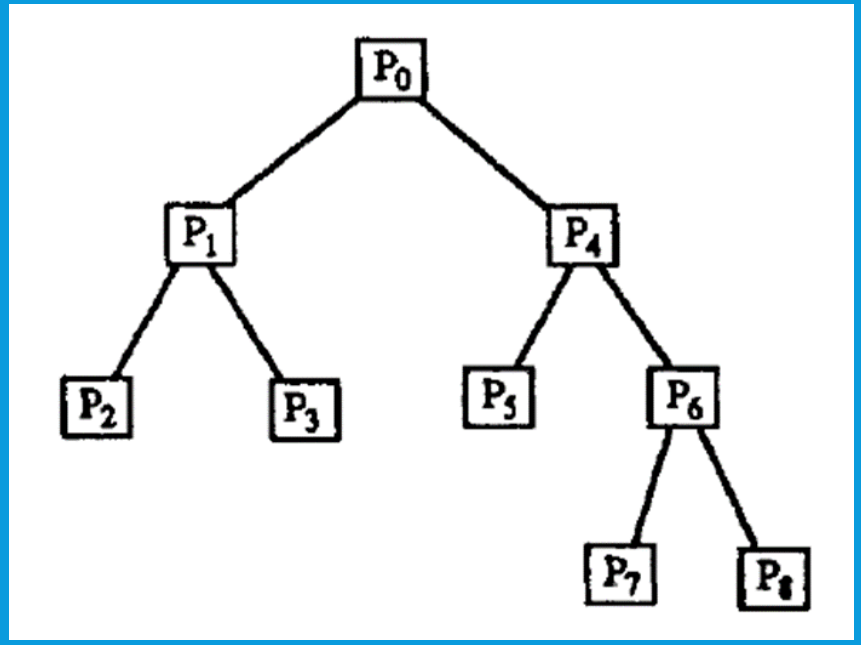
- Semigroup algorithm
  - Assume data elements are at the leafs.
  - Each inner node will receive two values from its children.
  - Do computation and pass the results upward.
  - After $\lceil \log_2 p \rceil$ steps the root will have the computation result
  - The result can be broadcasted from the root to the other nodes in $\lceil \log_2 p \rceil$ steps

ALGORITHMS FOR BINARY TREE



ALGORITHMS FOR BINARY TREE

- Routing algorithm
  - The numbering is as in the figure
  - Each node knows its number (self)
  - Each node knows largest node in its left (maxl) and right (maxr)
  - Routing from node *i* to (dest) and the packet currently on node (self)



ALGORITHMS FOR BINARY TREE

- Routing algorithm for a tree

```
if dest = self
then remove the packet {done}
else if dest < self or dest > maxr
    then route upward
    else if dest ≤ maxl
        then route leftward
        else route rightward
        endif
    endif
endif
```

ALGORITHMS FOR BINARY TREE

- Broadcast algorithm
- Processor *i* sends the desired data upwards to the root processor in $\lfloor \log_2 p \rfloor$ steps.
- then broadcasts the data from root downwards to all processors in $\lfloor \log_2 p \rfloor$ steps.

ALGORITHMS FOR BINARY TREE

- Sorting algorithm

if you have 2 items

then do nothing

else if you have 1 item that came from the left

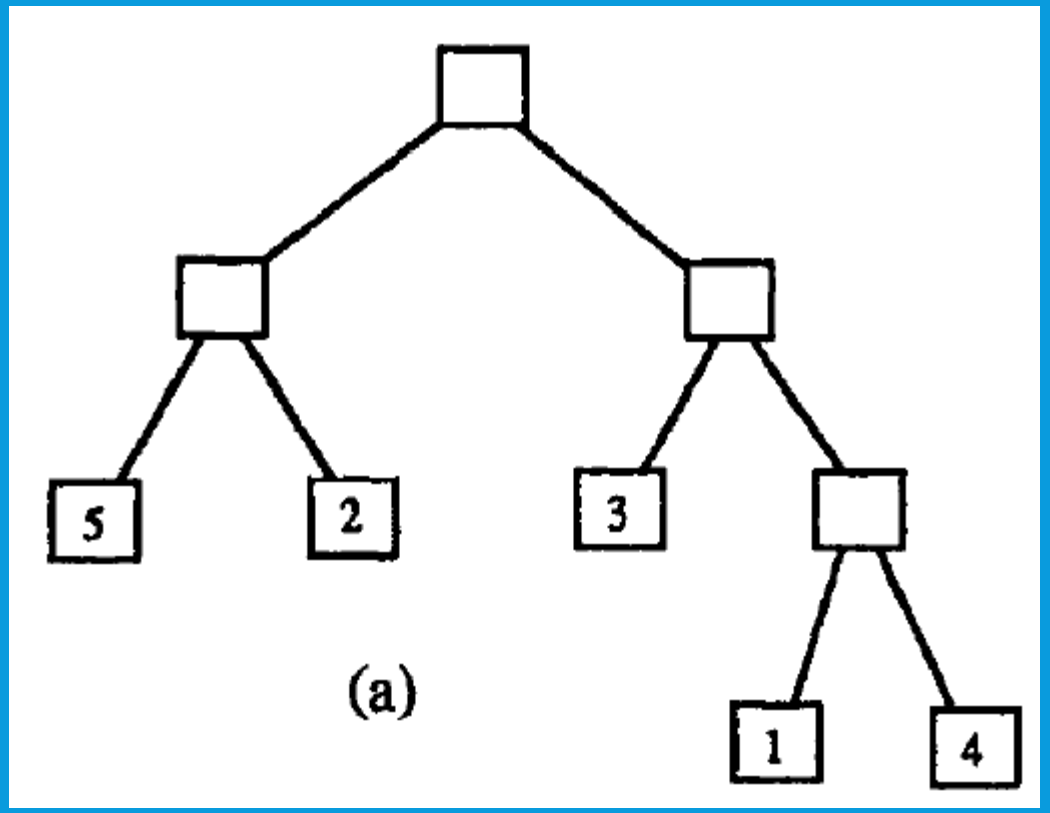
then get the smaller item from the right child

else get the smaller item from each child

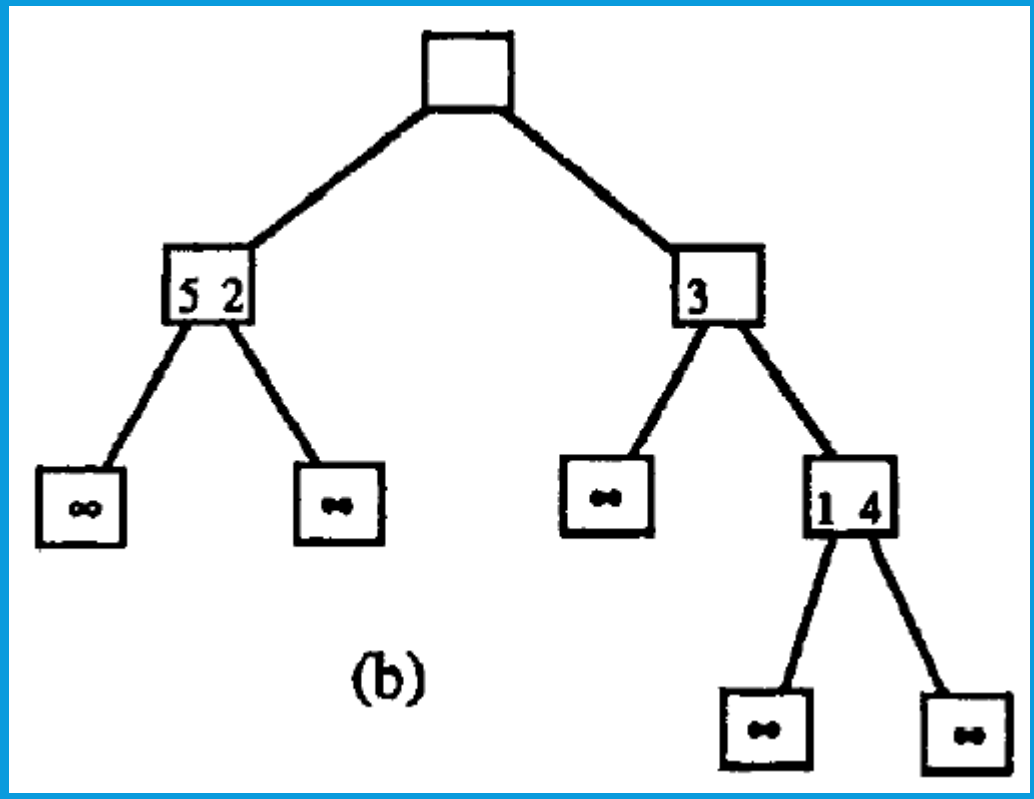
endif

endif

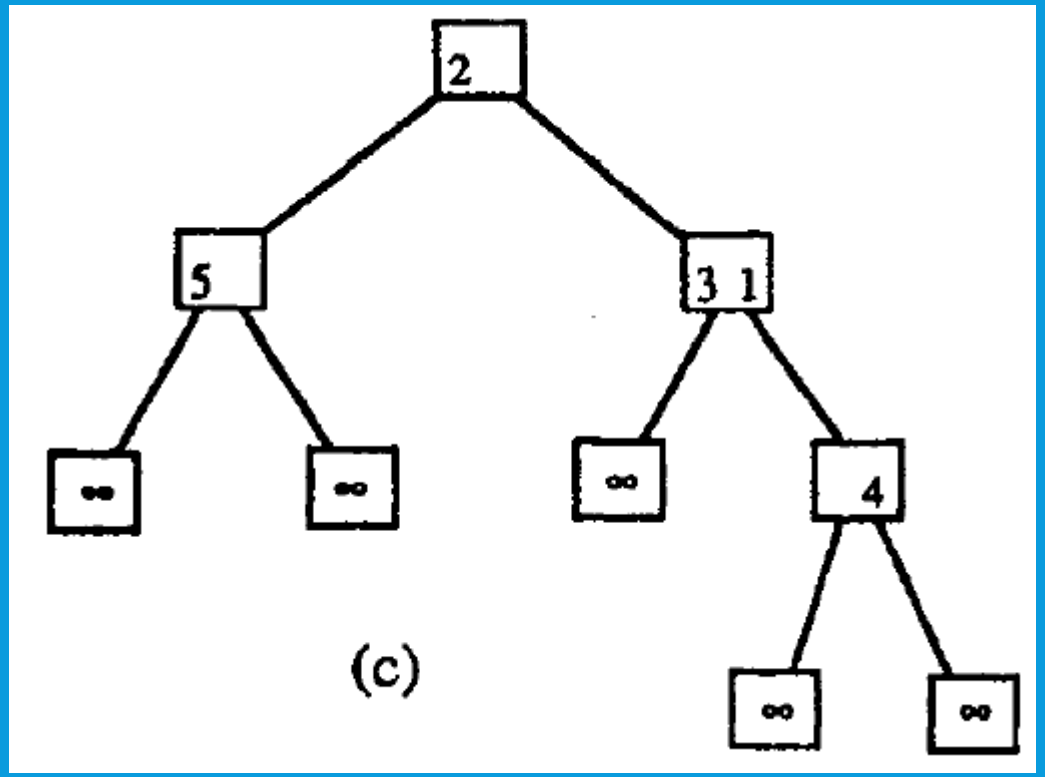
ALGORITHMS FOR BINARY TREE



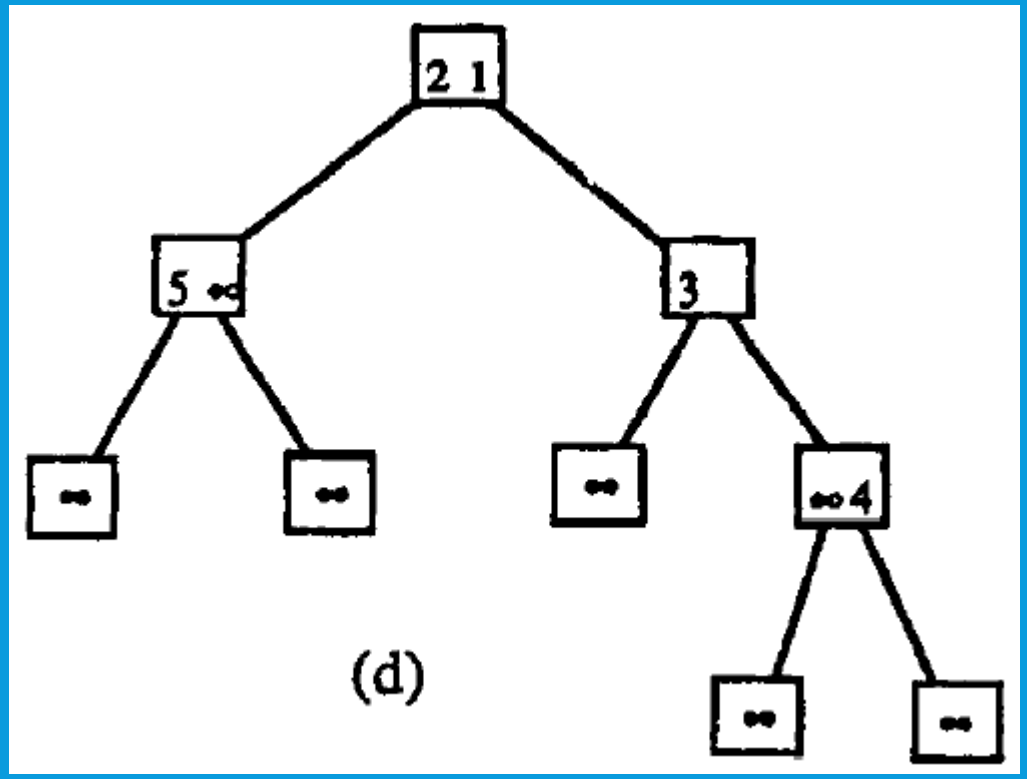
ALGORITHMS FOR BINARY TREE



ALGORITHMS FOR BINARY TREE



ALGORITHMS FOR BINARY TREE



ALGORITHMS FOR BINARY TREE

Each node knows the number of leaf nodes in its left subtree.

If the rank order of the element received from above (kept in a local counter) does not exceed the number of leaf nodes to the left, then the data item is sent to the left.

Otherwise, it is sent to the right.

ALGORITHMS FOR BINARY TREE

$2\lfloor \log_2 p \rfloor$