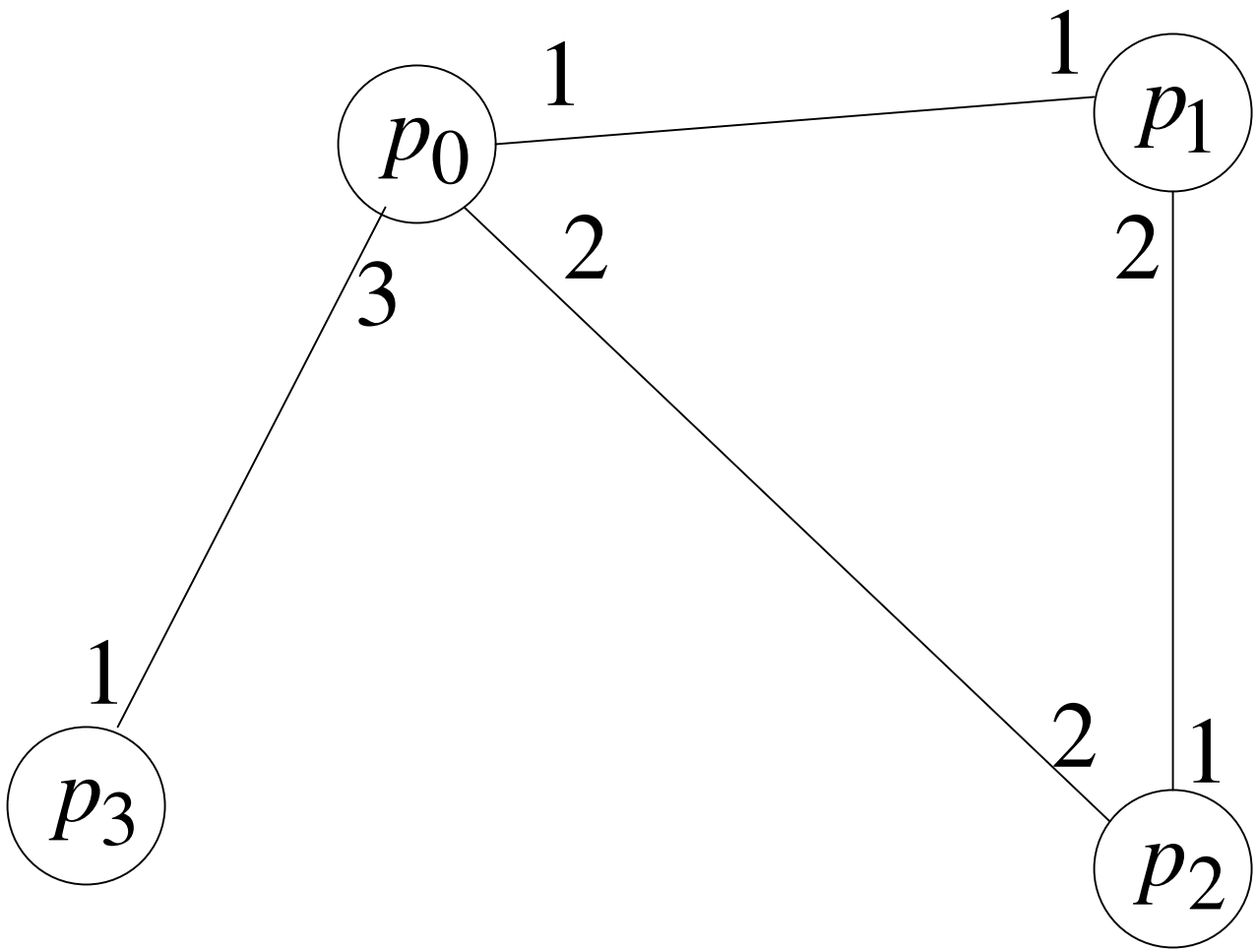
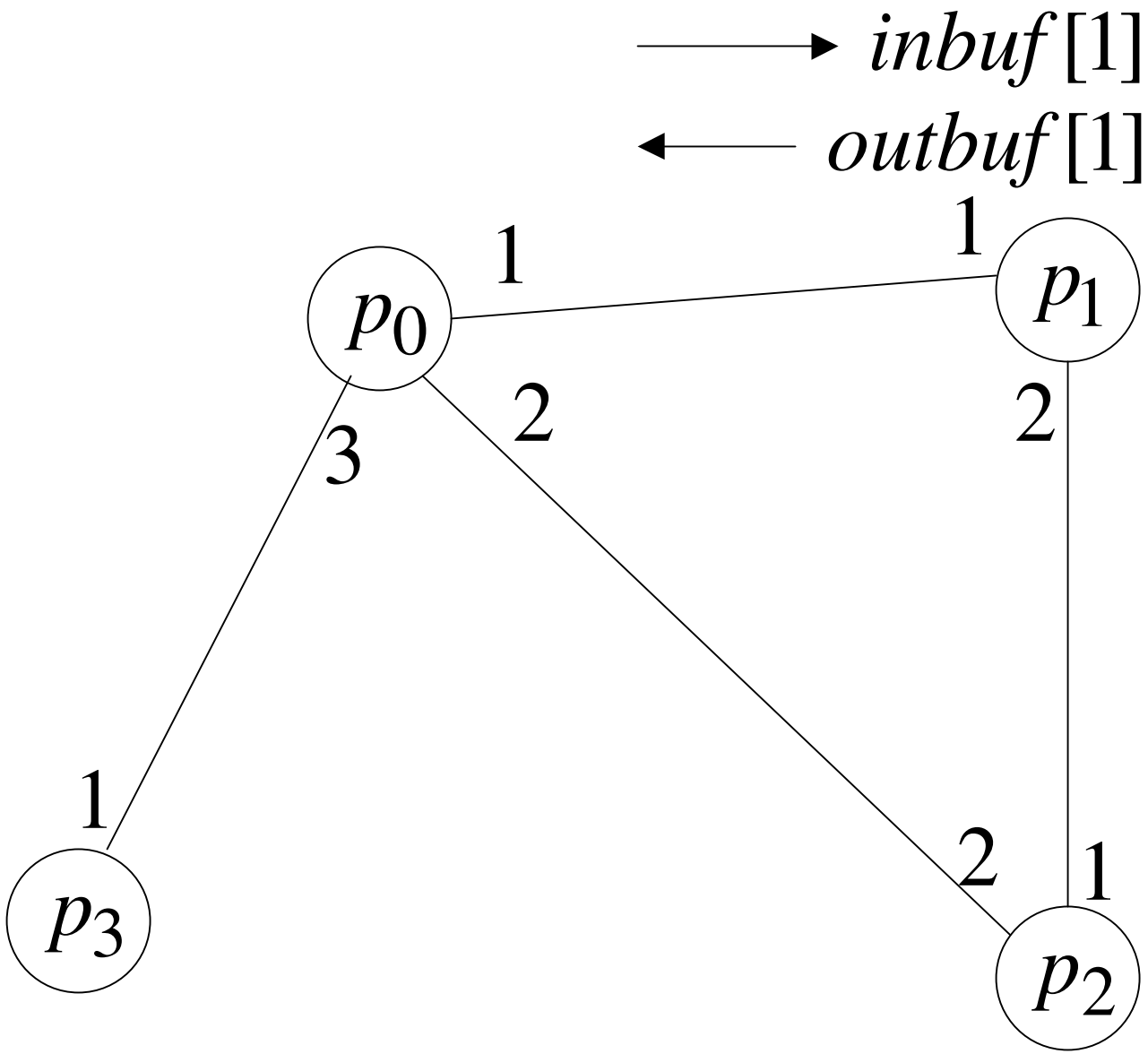
Basic Algorithms
in
Message Passing Systems


p_0
p_1
p_2
p_3
1
1
2
2
3
1
2
1


inbuf [1]
outbuf [1]
1
1
p_0
p_1
2
2
3
1
2
1
p_3
p_2

State of processor p_i : q_i

A function of

- previous state
- message received

Result: a new message to send

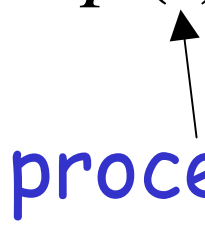
Time 0: Initial state

Configuration: $(q_0, q_1, \ldots, q_{n-1})$

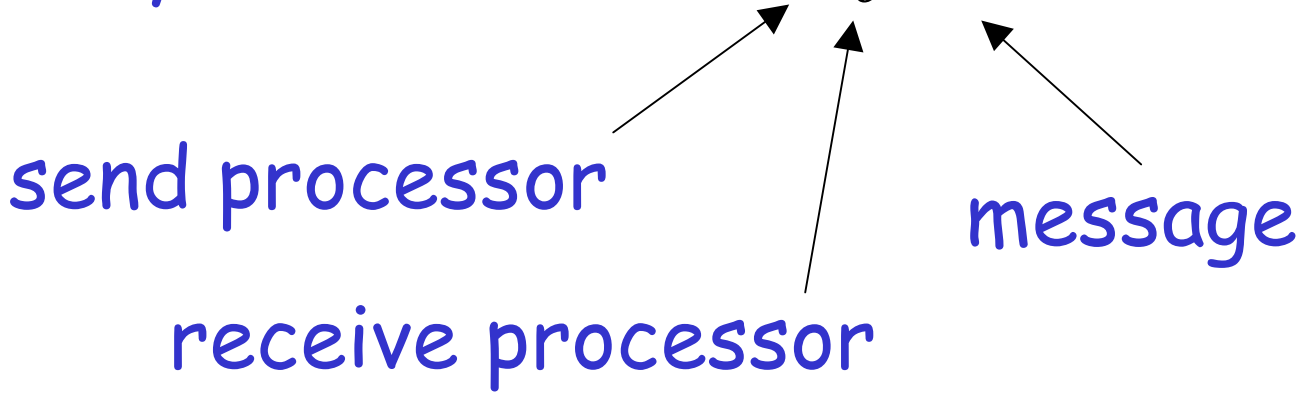
states

Events:

- Computation event: $comp(i)$



- Delivery event: $del(i, j, m)$



Execution: Any sequence of events that satisfy

- Safety conditions:

  Always satisfied

  Example: a message is received after it is sent

- Liveness conditions:

  Satisfied sometimes

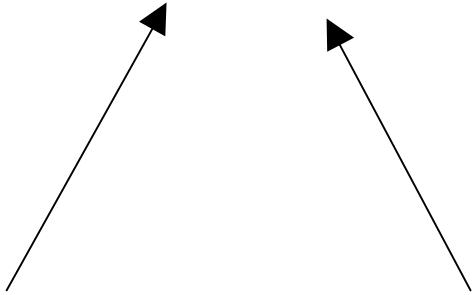
  Example: program must terminate

Asynchronous Systems

No upper bound on message delivery time

No upper bound on time between two processor steps

Execution: $C_0\ \phi_1\ C_1\ \phi_2\ C_2\ \phi_2 \cdots$



Configuration Event

$$C_0\ \phi_1\ C_1\ \phi_2\ C_2\ \phi_2 \cdots$$

Event

Delivery event: Deliver a message from an Outbuf to an Inbuf

Computation event: Execute a step in a processor

Synchronous Systems

Processors execute in lock steps

Execution is partitioned into rounds

At each round:

Every processor computes according to the message received at the previous round

Every processor sends a message

Remark:

In synchronous deterministic systems
the execution depends only
on the initial configuration

Not true for asynchronous systems

Complexity measures

- Number of messages
- Amount of time

Terminating state:

When a processor enters a terminating state it never leaves the state

The system terminates:

All processors enter terminating states

Message complexity:

Maximum number of messages over all possible executions

Time complexity:

Maximum time until execution terminates

Message delay:

Synchronous systems: one round

Asynchronous systems:
one time step
(normalized)

Timed executions:

the time that the event happens

Non-decreasing

Strictly increasing for each processor

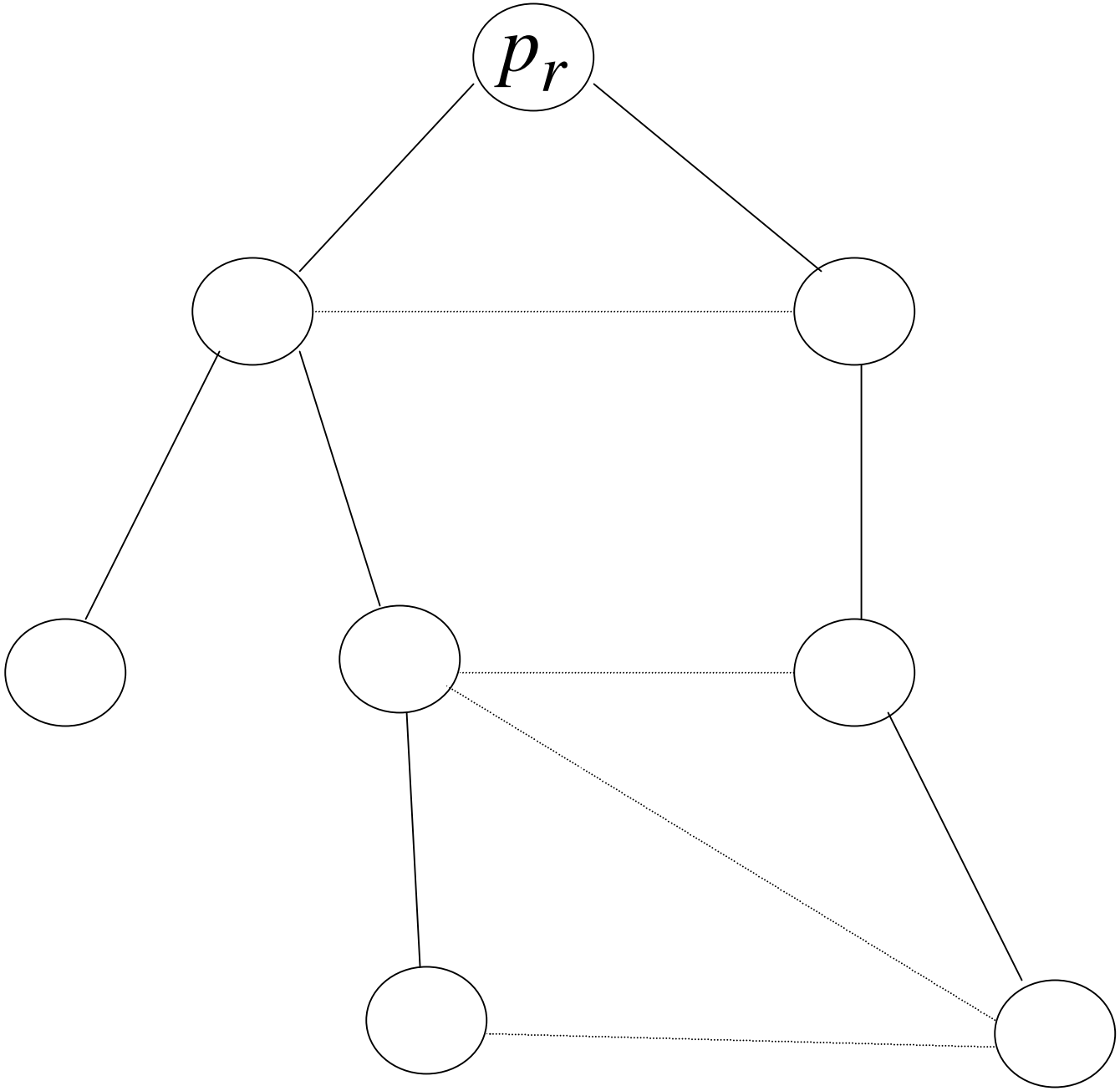
Broadcast

The problem:

A processor p_r wants to broadcast a message in the network

We are given:

a spanning tree with root p_r


p_r

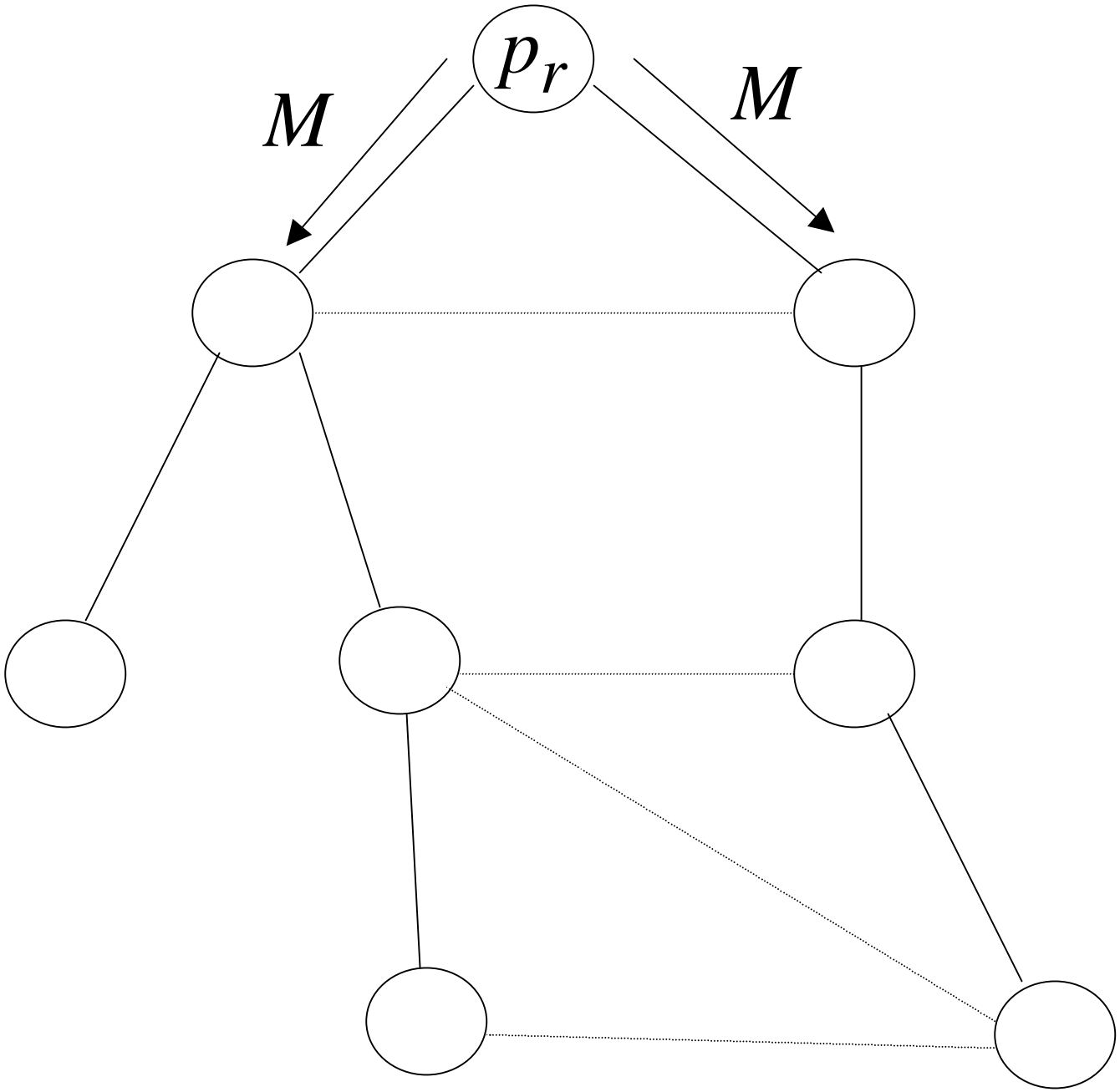
The Broadcast Algorithm:

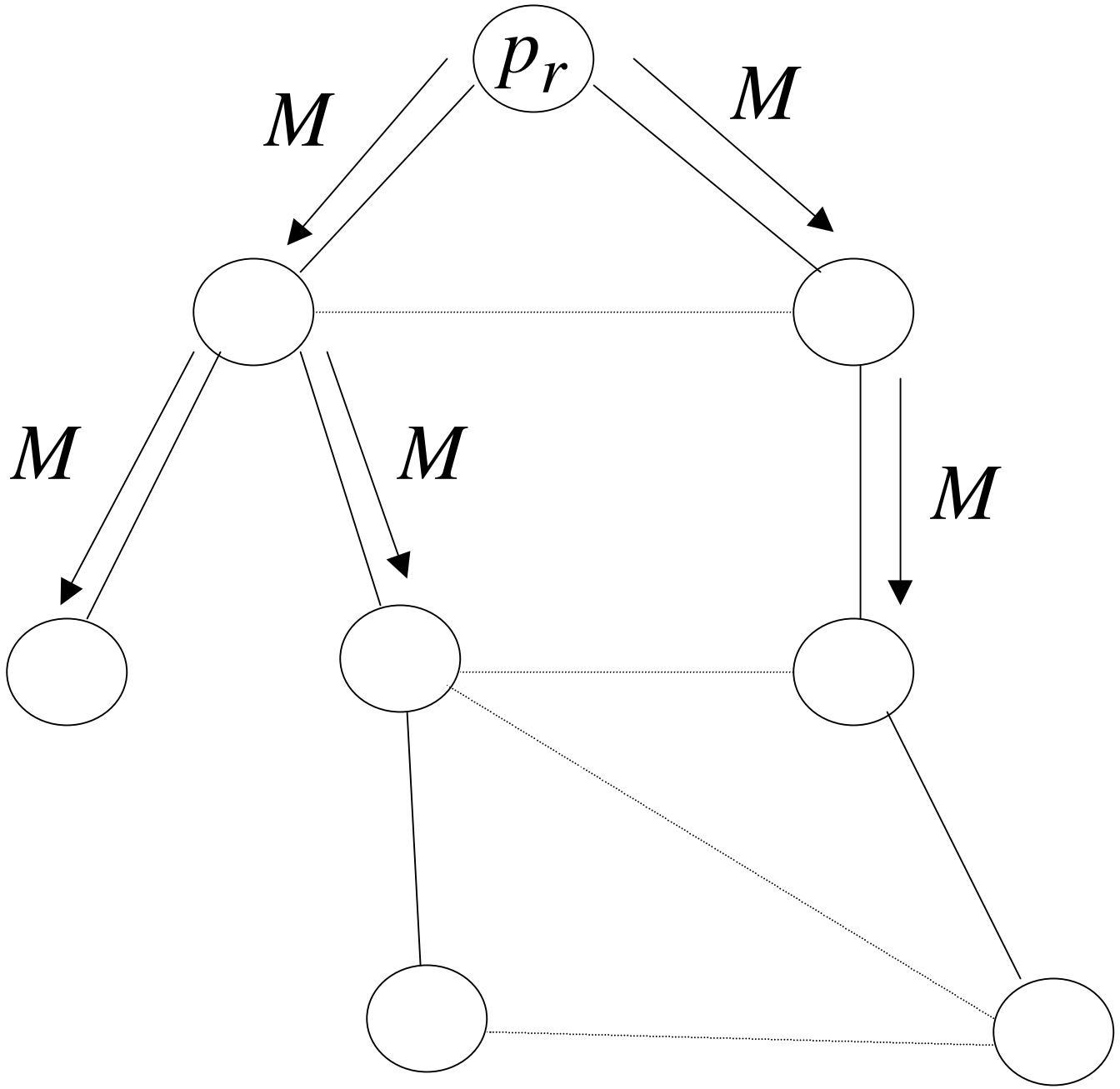
Root:

1. Send Message to children
2. terminate

Other Node:

1. Upon receiving message
2. Send message to children
3. terminate


p_r
M
M


p_r
M
M
M
M
M

Message Complexity

Network: n nodes

Spanning tree: $n-1$ edges

Total messages: $n-1$

Time Complexity

Spanning tree with depth d

Synchronous system: d rounds

Asynchronous system: d time steps