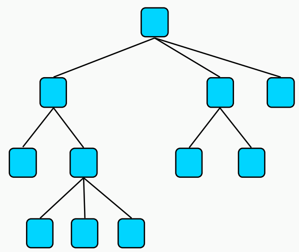
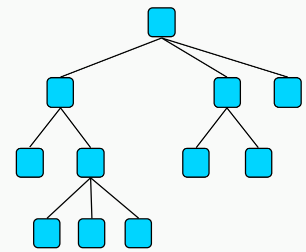
Computations in Trees

Saturation
Election
Minimum Finding
Eccentricity
Center
Ranking



Trees

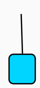
- Acyclic graph
- n entities
- n - 1 links



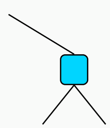
Saturation Technique

- Bidirectional links
- Ordered messages
- Reliability
- Knowledge of the topology
- Distinct identities

Each entity knows if it's a leaf:



Internal node:



S = {available, awake, processing }

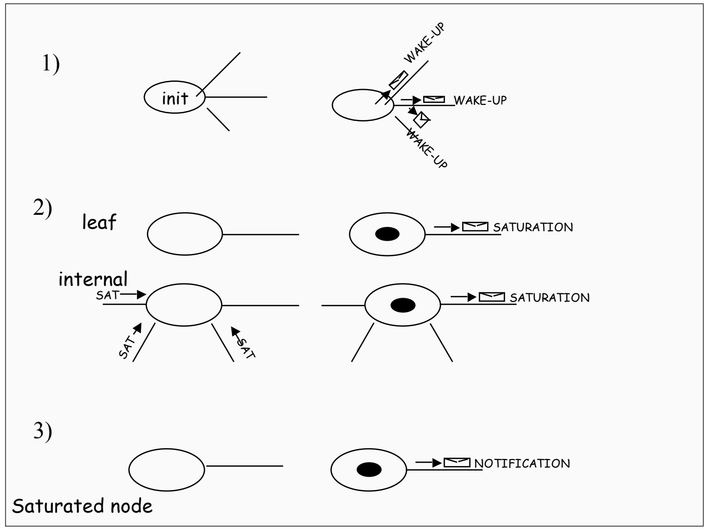
At the beginning, nodes are available

SATURATION: A General Technique

- **Activation phase:**
  all nodes are activated

- **Saturation Phase:**
  a unique pair of neighbours is identified (saturated nodes)

- **Resolution Phase:**
  started by the saturated nodes



S = {AVAILABLE, ACTIVE, PROCESSING
SATURATED}
Sinit = AVAILABLE

AVAILABLE

```
Spontaneously
    send(Activate)  to   N(x);
    Neighbours:= N(x)
    if |Neighbours|=1  then          /* special case if
        M:=("Saturation");              I am a leaf  */
        parent ⇐ Neighbours;
        send(M) to parent;
        become PROCESSING;
    else
        become ACTIVE;
```

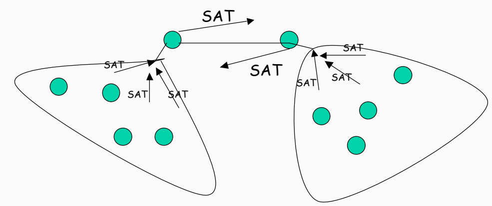
```
Receiving(Activate)
    send(Activate)to N(x)- {sender};
    Neighbours:=  N(x);
    if |Neighbours|=1  then
        M:=("Saturation");
        parent ⇐ Neighbours;
        send(M)  to  parent;
        become PROCESSING;
    else
        become ACTIVE;
```

ACTIVE
Receiving(M)
Neighbours:= Neighbours - {sender};
if |Neighbours|=1 then
M:=("Saturation");
parent ⇐ Neighbours;
send(M) **to parent;**
become PROCESSING;

PROCESSING

receiving(M)
become SATURATED;

become PROCESSING only after sending saturation



become SATURATED only after receiving something in the state PROCESSING

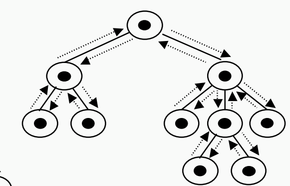
TWO entities become saturated

Which entities become saturated depends on the unpredictable delays

Other Observations and Examples

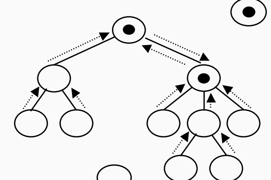
Complexity

Activation: Worst case - n initiators



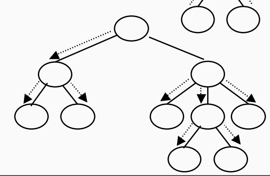
2(n-1)

Saturation:



n

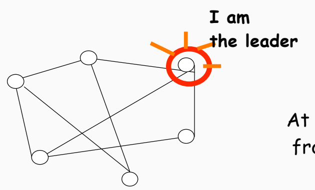
Notification:



n - 2

Tot: 2n -2+n+n-2=4n-4

Election

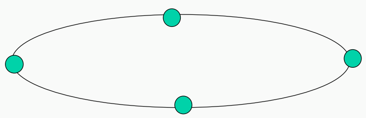


Initially: everybody in the same state

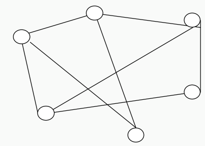
At the end: one entity different from the others

Election is in general impossible if the entities do not have distinct Identifiers

Ex.



Election

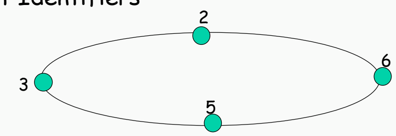


Initially: everybody in the same state

At the end: one entity different from the others

Election is in general impossible if the entities do not have distinct Identifiers

Ex.

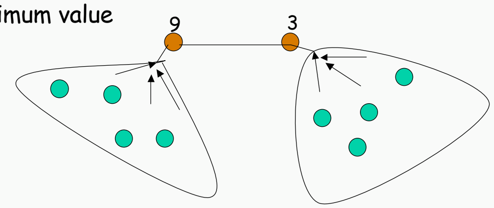


Election in the Tree

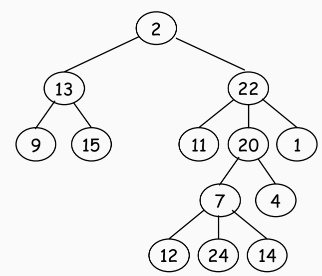
To each node x is associated a distinct identifier v(x)

1) Execute the saturation technique,

2) Choose the saturated node holding the minimum value



Minimum Finding



Put information in the saturation message

States S {AVAILABLE, ACTIVE, PROCESSING, SATURATED} Sinit = AVAILABLE

AVAILABLE

Spontaneously

```
send(Activate)  to   N(x);
min := v(x);
Neighbours:= N(x)
if |Neighbours|=1  then
        M:=("Saturation", min);
        parent ⇐ Neighbours;
        send(M) to parent;
        become PROCESSING;
else  become ACTIVE;
```

Receiving(Activate)

```
send(Activate) to N(x) – {sender};
min:=v(x);
Neighbours:=  N(x);
if |Neighbours|=1  then
    M:=("Saturation", min);
    parent ⇐ Neighbours;
    send(M)  to  parent;
    become PROCESSING;
else  become ACTIVE;
```

ACTIVE

Receiving(M)

```
min:= MIN{min, M}
Neighbours:= Neighbours - {sender}};
if |Neighbours|=1 then
        M:=("Saturation", min);
        parent ⇐ Neighbours;
        send(M) to parent;
        become PROCESSING;
```

PROCESSING

receiving(M)

```
min:= MIN{min, M}
Notification:= ("Resolution", min)
send (Notification)  to N(x) -parent
if v(x)=min then
        become MINIMUM
else
        become LARGE
```

receiving(Notification)

```
send(Notification)  to N(x) -parent
if v(x)=Received_Value  then
        become MINIMUM;
```

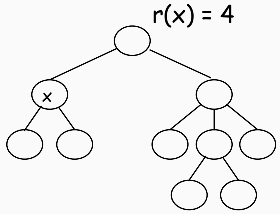
Finding Eccentricities

d(x,y) = distance between x and y

$\max_{y}\{d(x,y)\} = r(x)$ eccentricity of x

Ex: r(x) ?

r(x) = 4



Idea:

Every node broadcasts a request, the leaves send up a message that will collect the distances.

Complexity: $O(n^2)$

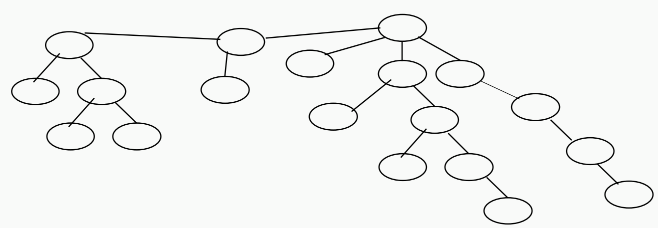
Other Idea:

based on the saturation technique:

1) Find the eccentricity of the two saturated nodes
2) Propagate the needed info so that the other nodes can find their eccentricity (in the notification phase)

Complexity = saturation

Observations and Examples



States S {AVAILABLE, ACTIVE, PROCESSING, SATURATED} Sinit = AVAILABLE

```
                    define Distance[ ]

AVAILABLE

Spontaneously
        send(Activate) to N(x);
        Distance[x]:= 0;
        Neighbours:=N(x)
        if |Neighbours|=1  then
                maxdist:= 1+ Max{Distance[*]}

                M:=("Saturation", maxdist);
                parent ⇐ Neighbours;
                send(M) to parent;
                become PROCESSING;
        else  become ACTIVE;
```

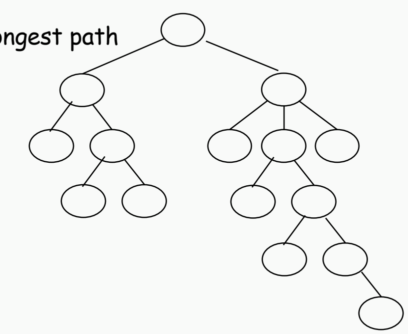
```
Receiving(Activate)
        send(Activate)to N(x) – {sender};
        Distance[x]:= 0;
        Neighbours:=  N(x);
        if |Neighbours|=1  then
                maxdist:= 1+ Max{Distance[*]}
                M:=("Saturation", maxdist);
                parent ⇐ Neighbours;
                send(M)  to parent;
                become PROCESSING;
        else  become ACTIVE;
```

```
ACTIVE
Receiving(M)
        Distance[{sender}]:= Received_distance;
        Neighbours:= Neighbours - {sender}};
        if |Neighbours|=1 then
                maxdist:= 1+ Max{Distance[*]}
                M:=("Saturation", maxdist);
                parent ⇐ Neighbours;
                send(M) to parent;
PROCESSING      become PROCESSING;
receiving(M)
        Distance[{ sender}]:= Received_distance;
        r(x):= Max { Distance[z]: z ∈ N(x) }
        for all  y ∈ N(x)-{parent} do
                maxdist:= 1+ Max{Distance[z]:
                                   z∈ N(x)- {y}
                send("Resolution", maxdist)  to y
        endfor
```

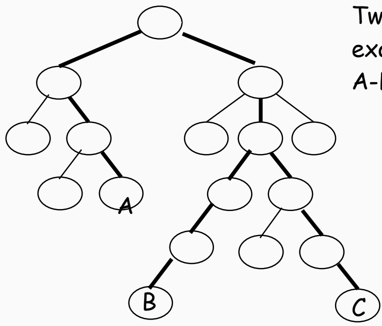
Center Finding

c is the center if $r(c) \leq r(x)$ for all x belonging to V. Max distance is minimized.

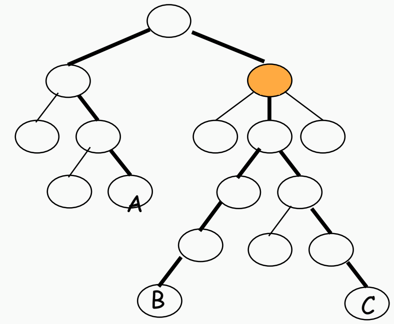
Diametral path: Longest path



Two diameters in this example:
A-B, and A-C



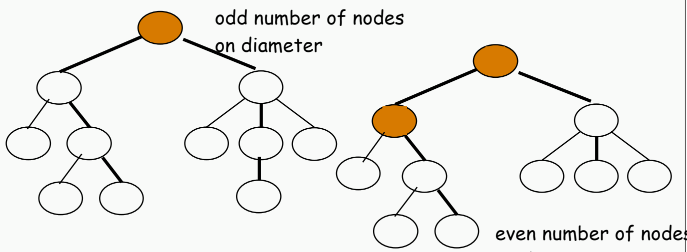
The center is the node with smallest eccentricity



One Idea:

1) Find all the eccentricities
2) Find the smallest

Proposition 1: There is only one center, or there are two centers (neighbours)



Proposition 2: Centers are on diametral paths.

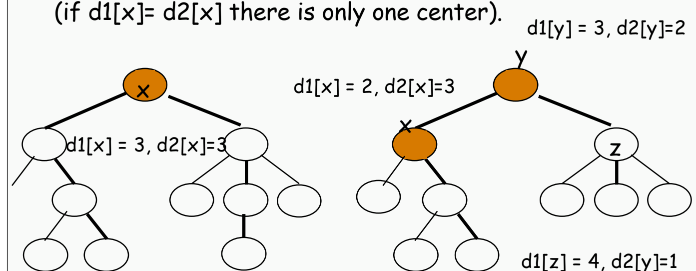
d1[x] = max dist   d2[x] = second max dist

(through different neighbours)

Proposition 3: A node x is a center iff

d1[x] – d2[x] ≤ 1

(if d1[x]= d2[x] there is only one center).



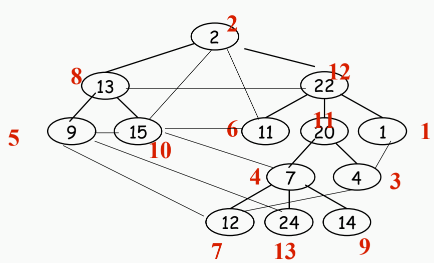
Another Idea:
1) Find all the eccentricities
2) Each node can find out locally whether
 it is the center or not

Yet another Idea:

1) Find the eccentricities of the
 saturated nodes
2) Check if I am the center (checking largest
 and second largest)
3) If I am NOT the center, propagate the
 distance info ONLY in the direction of
 the center
 How do I know the direction of the center ?

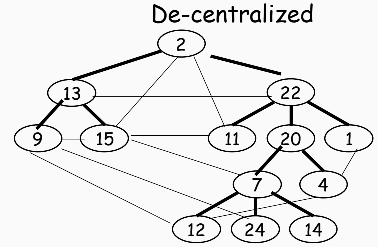
Examples ....

Ranking

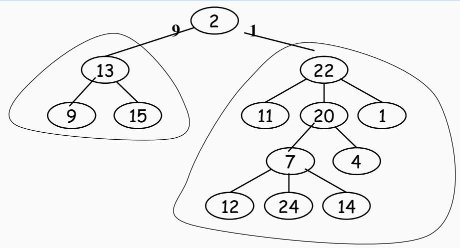


In an arbitrary network:

1) Find a spanning tree
2) Use saturation+ minimum finding
 to find a starting node
3) Do ranking
Phase 3 can be: Centralized
 De-centralized



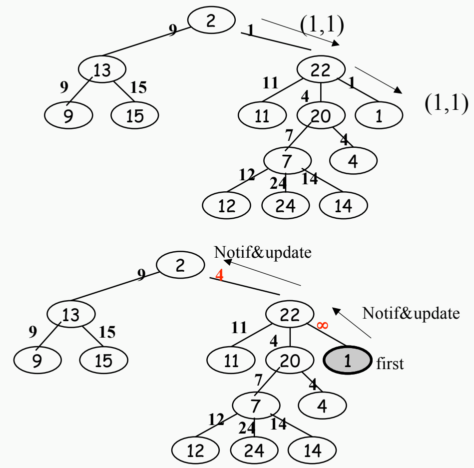
Centralized Ranking

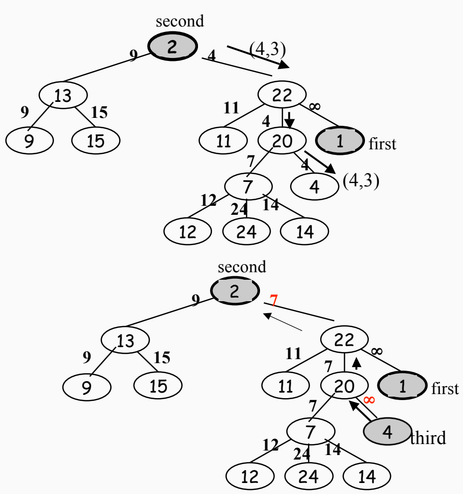


The leader knows the minimum, it sends in that direction a ranking message

Every node knows the minimum in its subtrees, they can then forward the ranking message in the right direction

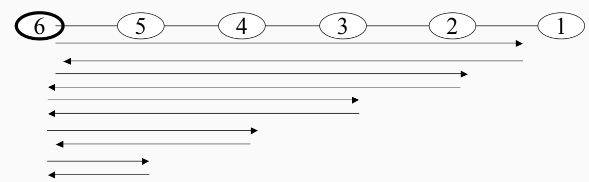
When the node to be ranked receives the message It sends up a notification&update message that will travel up to the leader



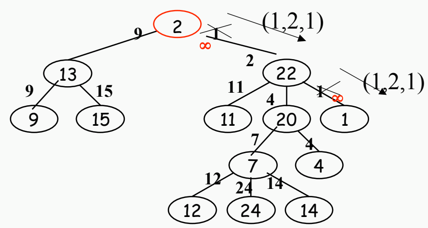


Etc…

Complexity: worst case



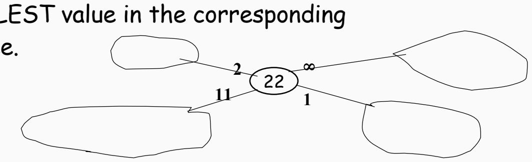
Decentralized Ranking



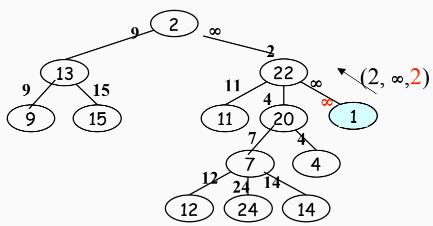
The starter node send a ranking message of the form:
(**first**, **second**,**rank**) in the direction of first.

first: smallest value
second: second smallest known **SO FAR**
(this is a guess on the value that has to be

The value on a link indicates the SMALLEST value in the corresponding subtree.



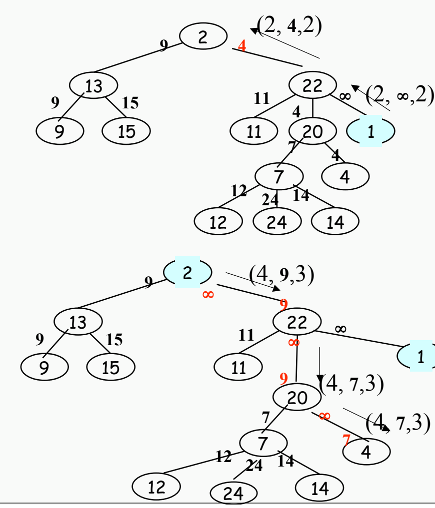
If no value is indicated (or the value is ∞) it means that the smallest in the corresponding subtree is unknown (for the moment)

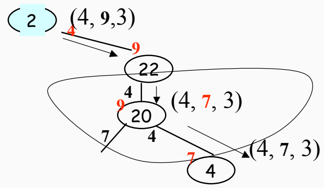


The ranked node attempts to send a ranking message to the next node to be ranked

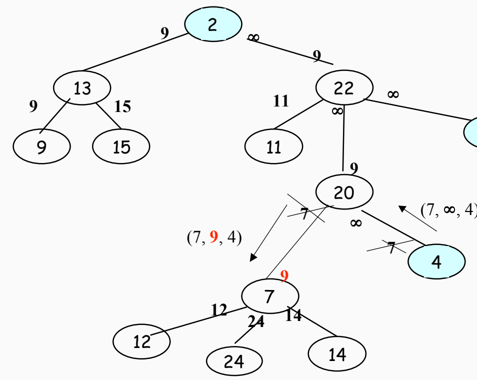
Second might now be unknown, in this case the value ∞ is used

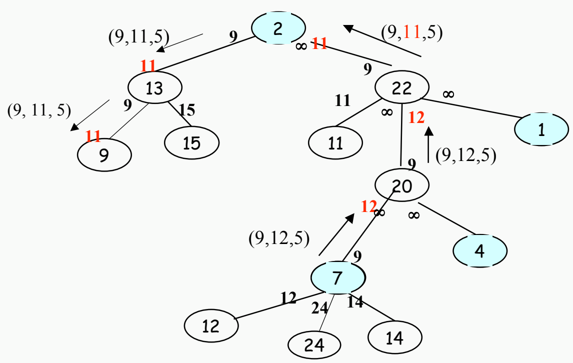
The second variable of the rank message is updated during its travel and the minimum values on the links of the tree are also updates

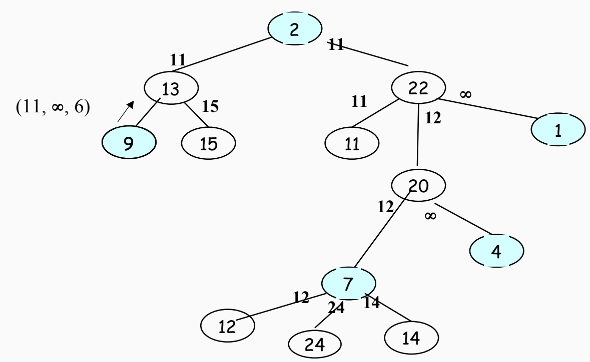




Notice the update







Complexity: worst case