Un algorithme distribué d'énumération des noeuds d'un réseau et application au calcul des distances entre 2 noeuds quelconques et du diamètre d'un réseau

Yves Métivier, John Michael Robson, Akka Zemmari

LaBRI - Université de Bordeaux

AlgoTel 2016
25 Mai 2013

Plan

Plan

Le problème

Etant donné un réseau :

1. attribuer un numéro de 1 à n (la taille du réseau) à chaque noeud,
2. utiliser cette énumération pour :
   - calculer (tous) les plus courts chemins,
   - calculer le diamètre du réseau,
   - autres paramètres : la maille, ...

Modèle

Réseau de communication connexe modélisé par un graphe $G = (V, E)$:

- les noeuds communiquent par passage de messages,
- système synchrone : les noeuds commencent en même temps et opèrent par rondes synchrones
- anonymat : les noeuds n'ont pas des identifiants différents,
- un noeud est dans un état *Leader*.

Modèle

Complexité en temps :

- une ronde (pour un neoud) : 1. envoyer des messages à des voisins, 2. recevoir des messages des voisins, 3. réaliser un calcul local,
- complexité en temps : le nombre de rondes nécessaires pour que *tous* les noeuds terminent.

Complexité en bits :

- dans chaque ronde, chaque noeud peut envoyer/recevoir un bit à/de chaque voisin,
- complexité en bits : le nombre de rondes nécessaires pour que *tous* les noeuds terminent.

Etat de l'art

	Temps	Taille des messages (nombre de bits)	complexité en bits
Almeida et al.	$O(D)$	$O(n \log n)$	$O(Dn \log n)$
Holzer and Wattenhofer (PODC 2012)	$O(n)$	$O(\log n)$	$O(n \log n)$
Peleg et al. (ICALP12)	$O(n)$	$O(\log n)$	$O(n \log n)$
Frischknecht et al. (SODA 2012)		B	$\Omega(n/B)$
Cet article	$O(n)$	$O(1)$	$O(n)$

Calcul du diamètre.

Plan

Enumération distribuée

DEA : un algorithme distribué qui attribue à chaque sommet du graphe un nombre unique dans $\{1, 2, \cdots, n\}$, et tel que :

- la distance entre deux sommets ayant des numéros consécutifs est au plus 3,
- les messages échangés sont de taille $O(1)$,
- la complexité en temps est en $O(n)$.

DEA opère en deux étapes :

1. calcul d'un arbre couvrant BFS de G, dont la racine est le sommet *Leader*,
2. énumération des sommets de G selon une traversée spéciale (Algorithme *Trav*).

Enumération distribuée

DEA : un algorithme distribué qui attribue à chaque sommet du graphe un nombre unique dans $\{1, 2, \cdots, n\}$, et tel que :

- la distance entre deux sommets ayant des numéros consécutifs est au plus 3,
- les messages échangés sont de taille $O(1)$,
- la complexité en temps est en $O(n)$.

DEA opère en deux étapes :

1. calcul d'un arbre couvrant BFS de G, dont la racine est le sommet *Leader*,
2. énumération des sommets de G selon une traversée spéciale (Algorithme *Trav*).

Enumération distribuée

DEA : un algorithme distribué qui attribue à chaque sommet du graphe un nombre unique dans $\{1, 2, \cdots, n\}$, et tel que :

- la distance entre deux sommets ayant des numéros consécutifs est au plus 3,
- les messages échangés sont de taille $O(1)$,
- la complexité en temps est en $O(n)$.

DEA opère en deux étapes :

1. calcul d'un arbre couvrant BFS de G, dont la racine est le sommet *Leader*,
2. énumération des sommets de G selon une traversée spéciale (Algorithme *Trav*).

Enumération distribuée

DEA : un algorithme distribué qui attribue à chaque sommet du graphe un nombre unique dans $\{1, 2, \cdots, n\}$, et tel que :

- la distance entre deux sommets ayant des numéros consécutifs est au plus 3,
- les messages échangés sont de taille $O(1)$,
- la complexité en temps est en $O(n)$.

DEA opère en deux étapes :

1. calcul d'un arbre couvrant BFS de G, dont la racine est le sommet *Leader*,
2. énumération des sommets de G selon une traversée spéciale (Algorithme *Trav*).

Enumération distribuée

DEA : un algorithme distribué qui attribue à chaque sommet du graphe un nombre unique dans $\{1, 2, \cdots, n\}$, et tel que :

- la distance entre deux sommets ayant des numéros consécutifs est au plus 3,
- les messages échangés sont de taille $O(1)$,
- la complexité en temps est en $O(n)$.

DEA opère en deux étapes :

1. calcul d'un arbre couvrant BFS de G, dont la racine est le sommet *Leader*,
2. énumération des sommets de G selon une traversée spéciale (Algorithme *Trav*).

Enumération distribuée

DEA : un algorithme distribué qui attribue à chaque sommet du graphe un nombre unique dans $\{1, 2, \cdots, n\}$, et tel que :

- la distance entre deux sommets ayant des numéros consécutifs est au plus 3,
- les messages échangés sont de taille $O(1)$,
- la complexité en temps est en $O(n)$.

DEA opère en deux étapes :

1. calcul d'un arbre couvrant BFS de G, dont la racine est le sommet *Leader*,
2. énumération des sommets de G selon une traversée spéciale (Algorithme *Trav*).

Enumération distribuée

DEA : un algorithme distribué qui attribue à chaque sommet du graphe un nombre unique dans $\{1, 2, \cdots, n\}$, et tel que :

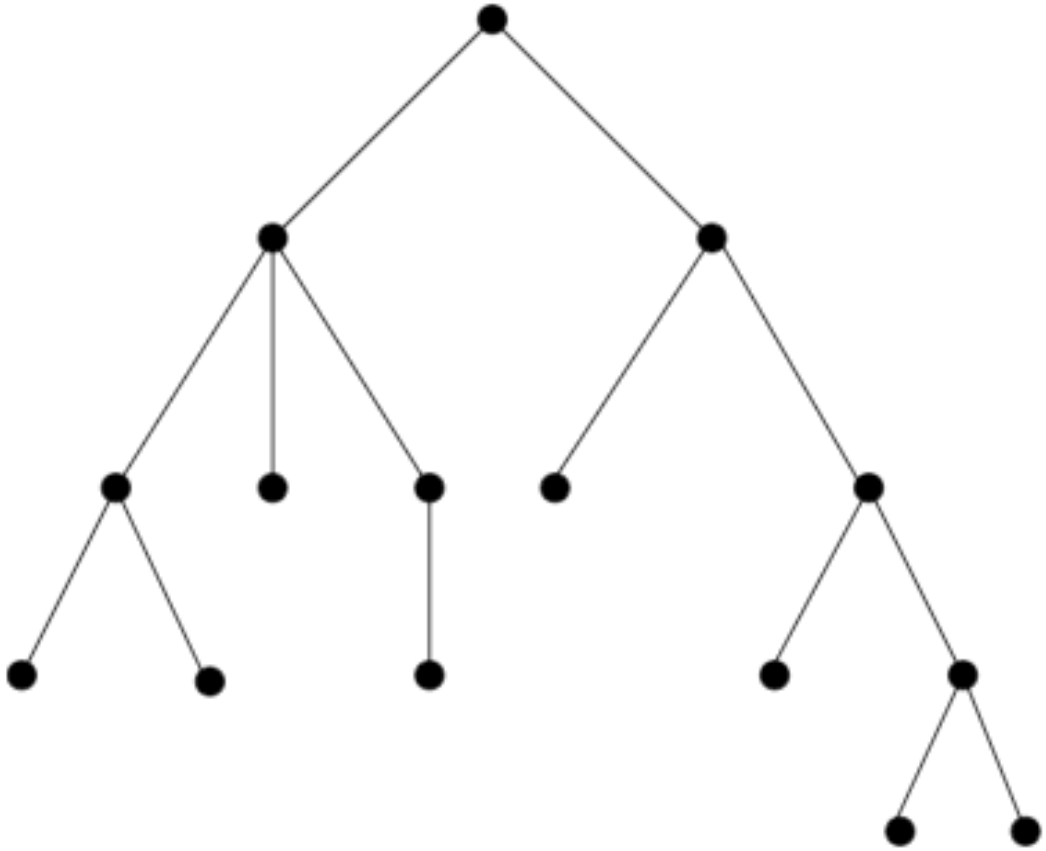
- la distance entre deux sommets ayant des numéros consécutifs est au plus 3,
- les messages échangés sont de taille $O(1)$,
- la complexité en temps est en $O(n)$.

DEA opère en deux étapes :

1. calcul d'un arbre couvrant BFS de G, dont la racine est le sommet *Leader*,
2. énumération des sommets de G selon une traversée spéciale (Algorithme *Trav*).

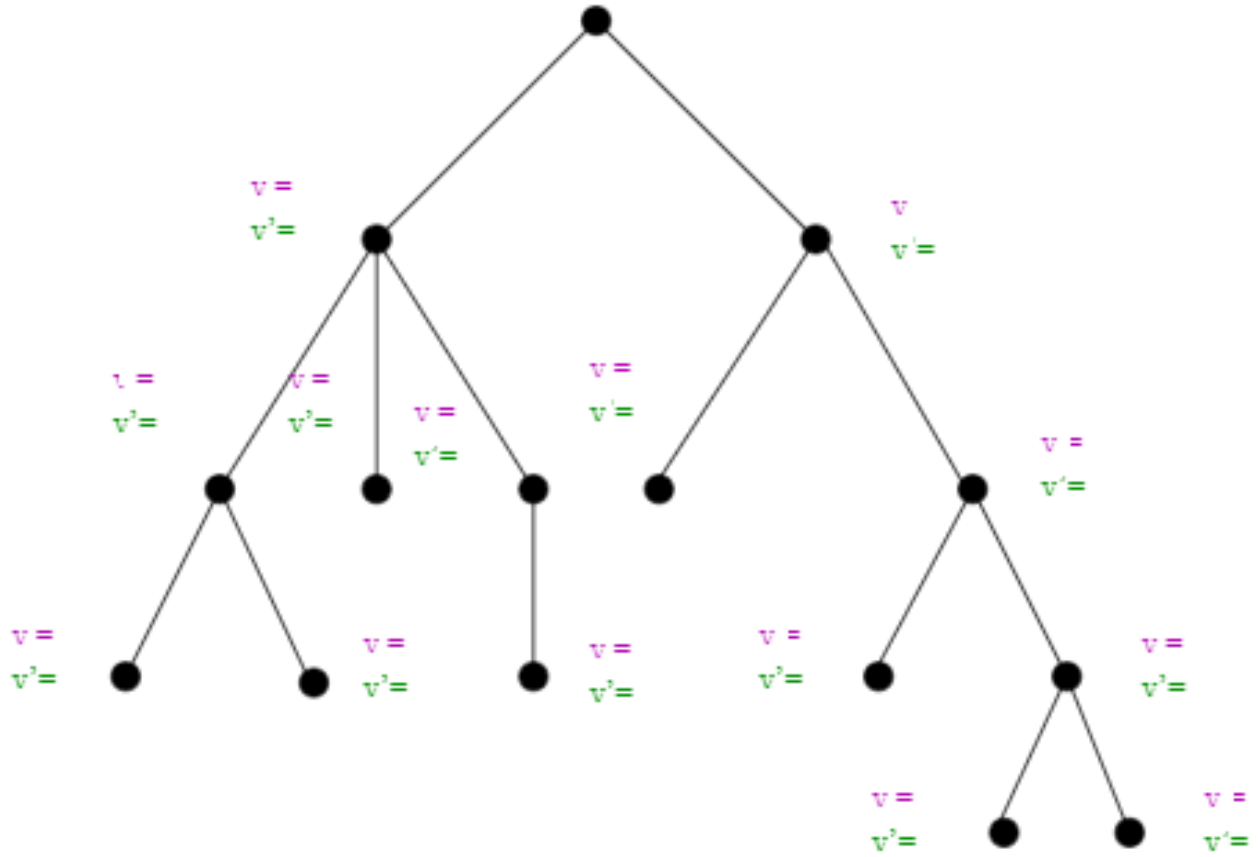
Enumération distribuée

Algorithme *Trav* :



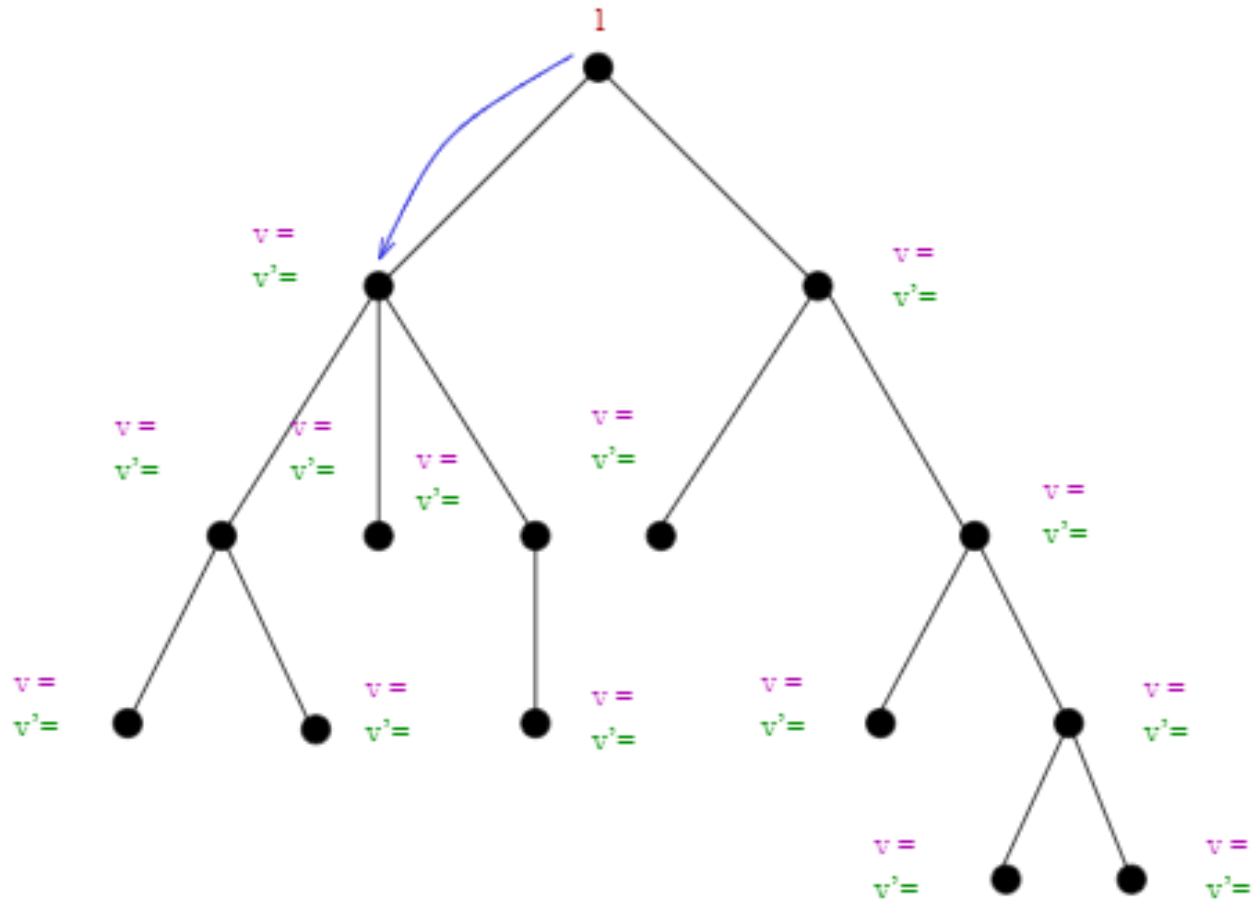
Enumération distribuée

Algorithme *Trav* :



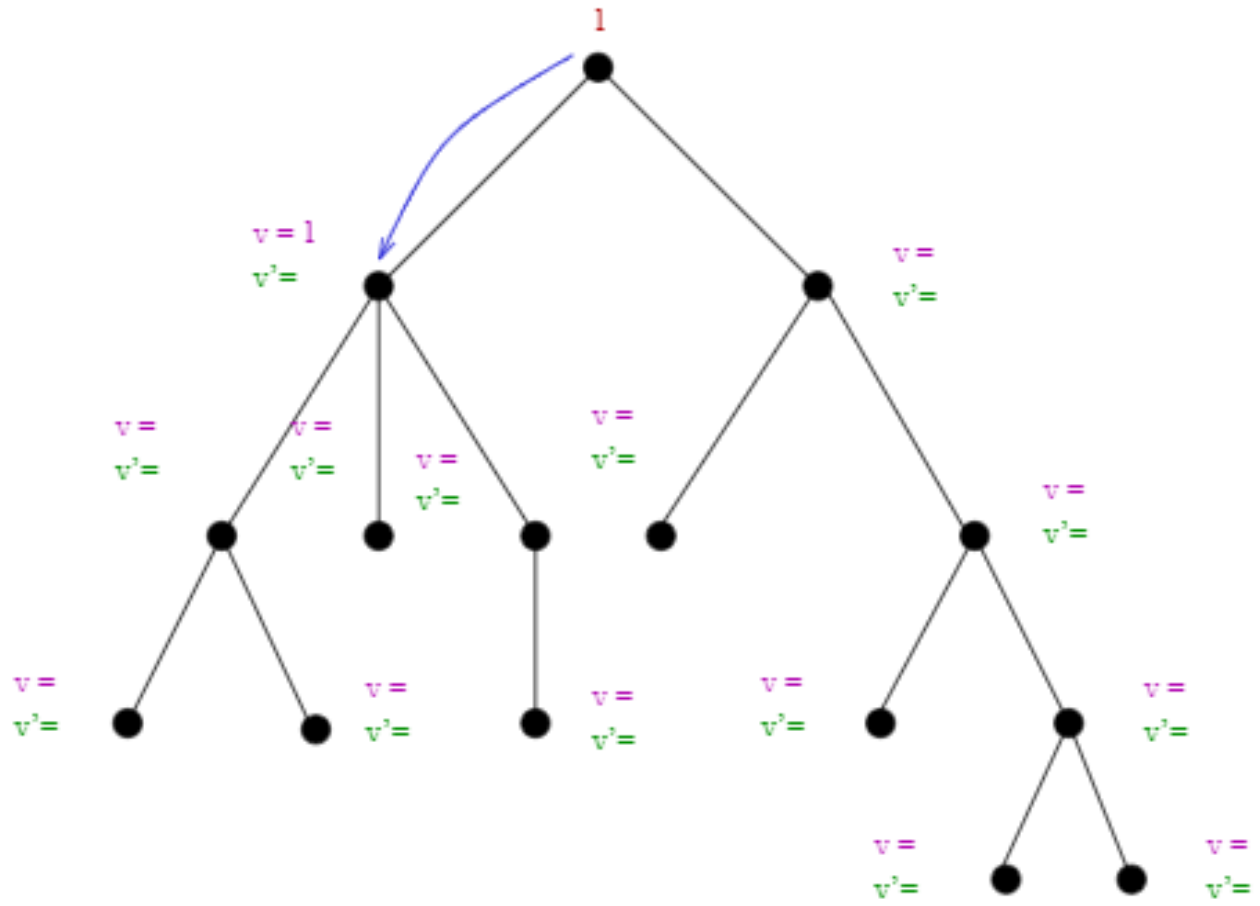
Enumération distribuée

Algorithme *Trav* :



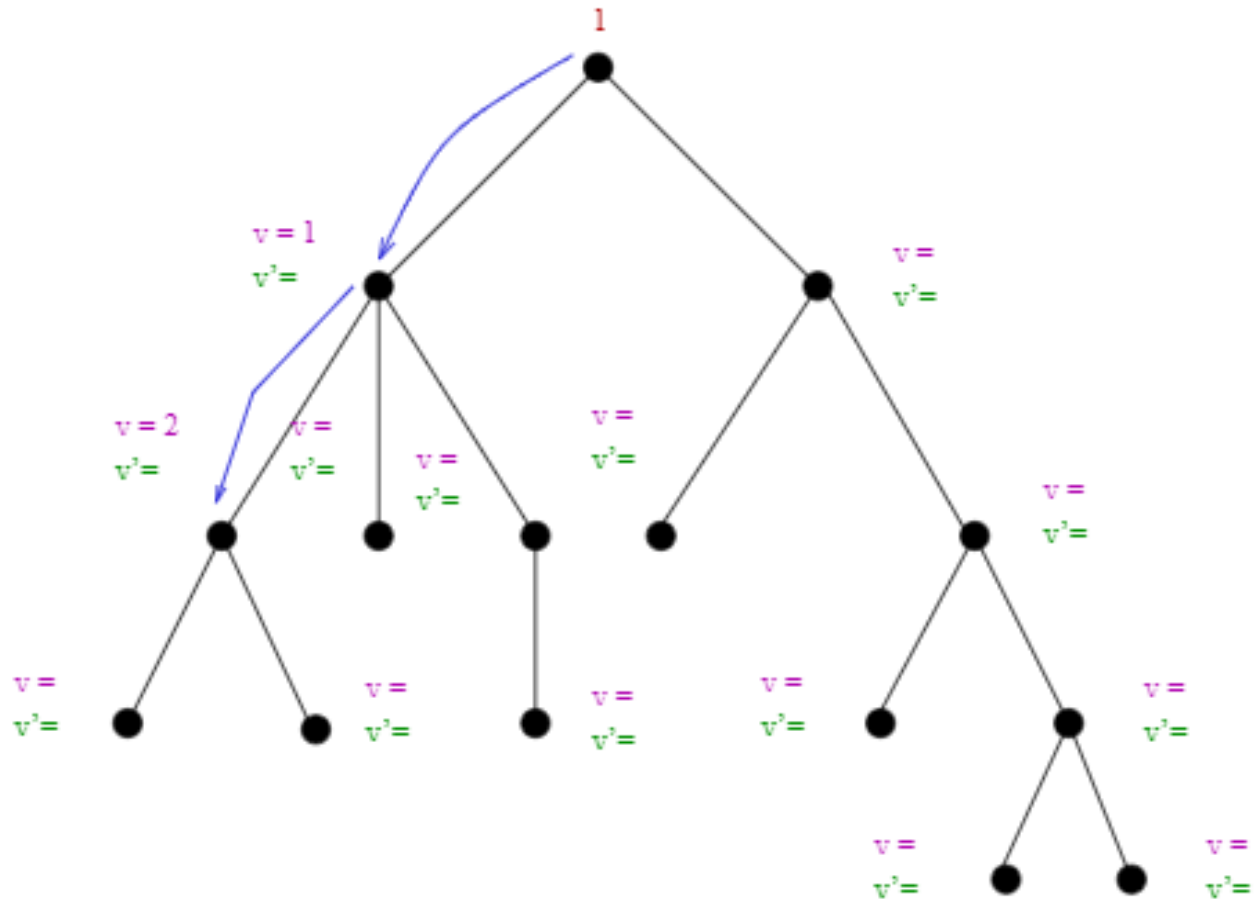
Enumération distribuée

Algorithme *Trav* :



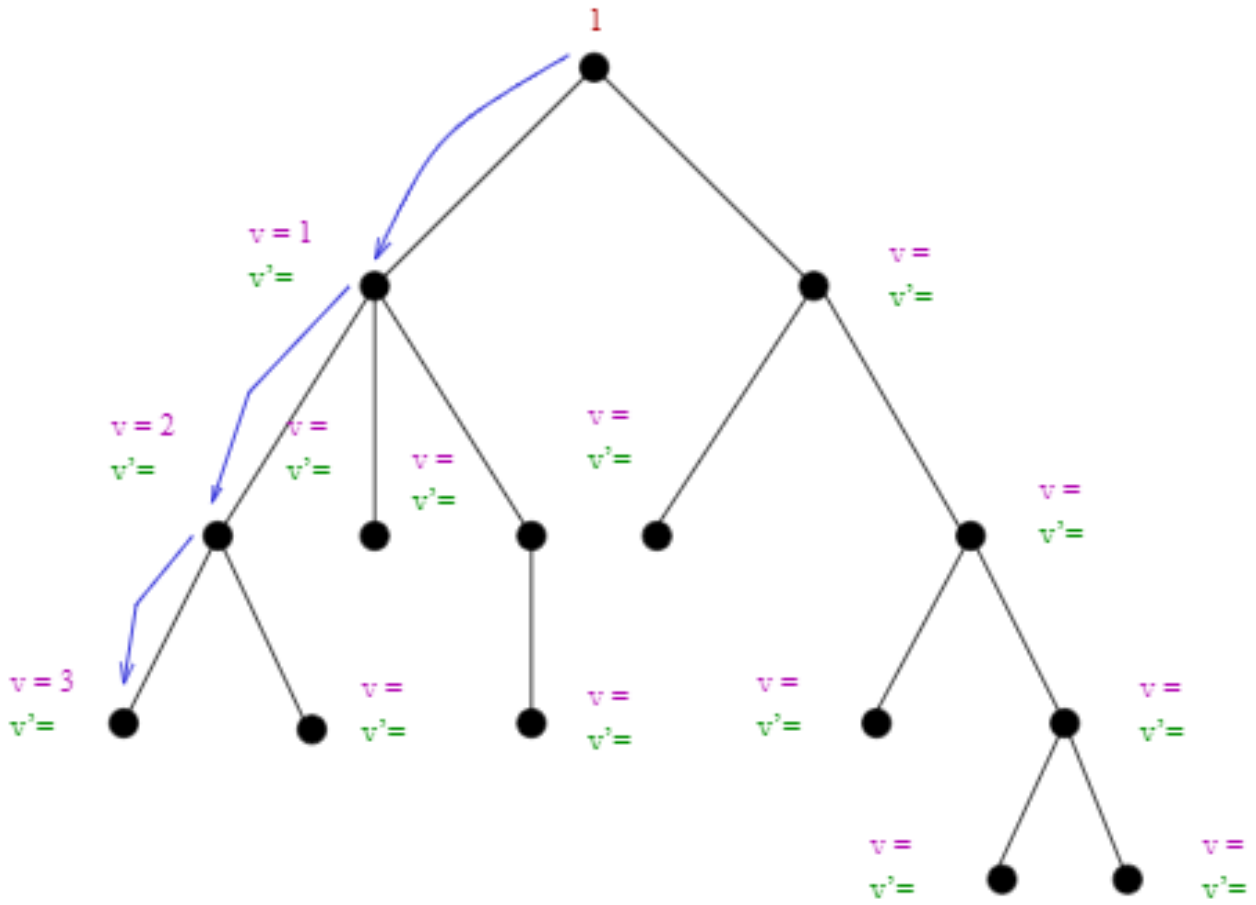
Enumération distribuée

Algorithme *Trav* :



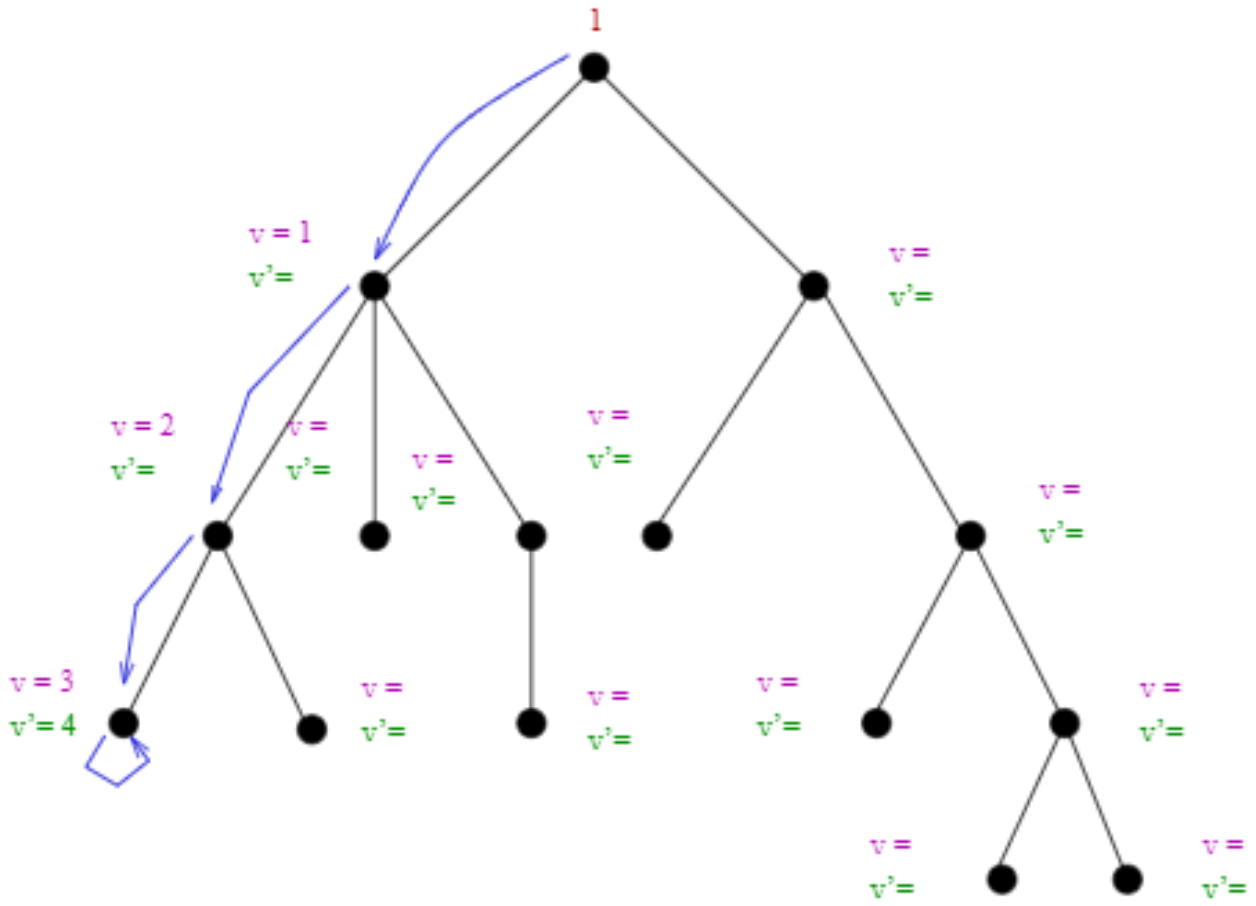
Enumération distribuée

Algorithme *Trav* :



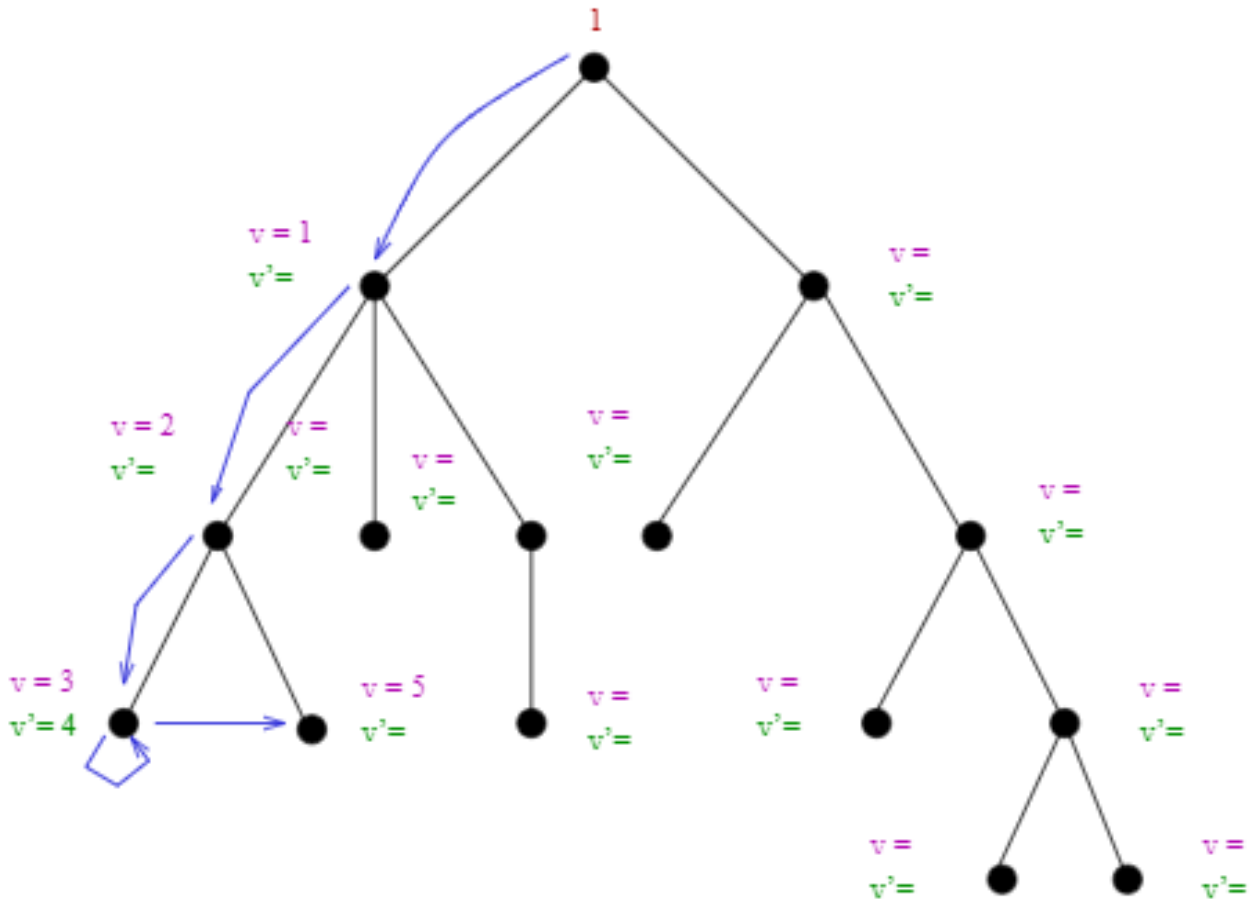
Enumération distribuée

Algorithme *Trav* :



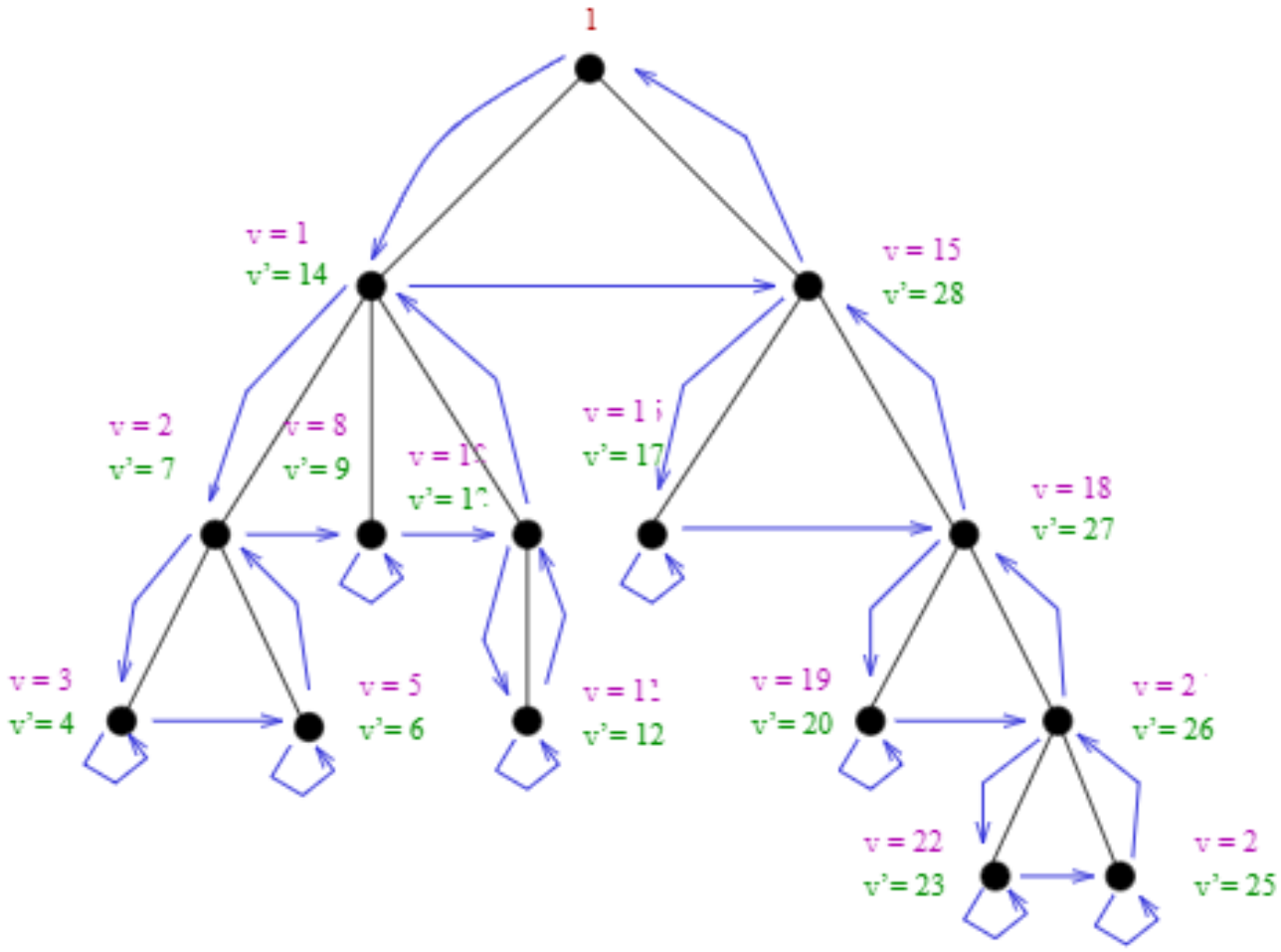
Enumération distribuée

Algorithme *Trav* :



Enumération distribuée

Algorithme *Trav* :



Enumération distribuée

Algorithme *Trav* :

Lemme.

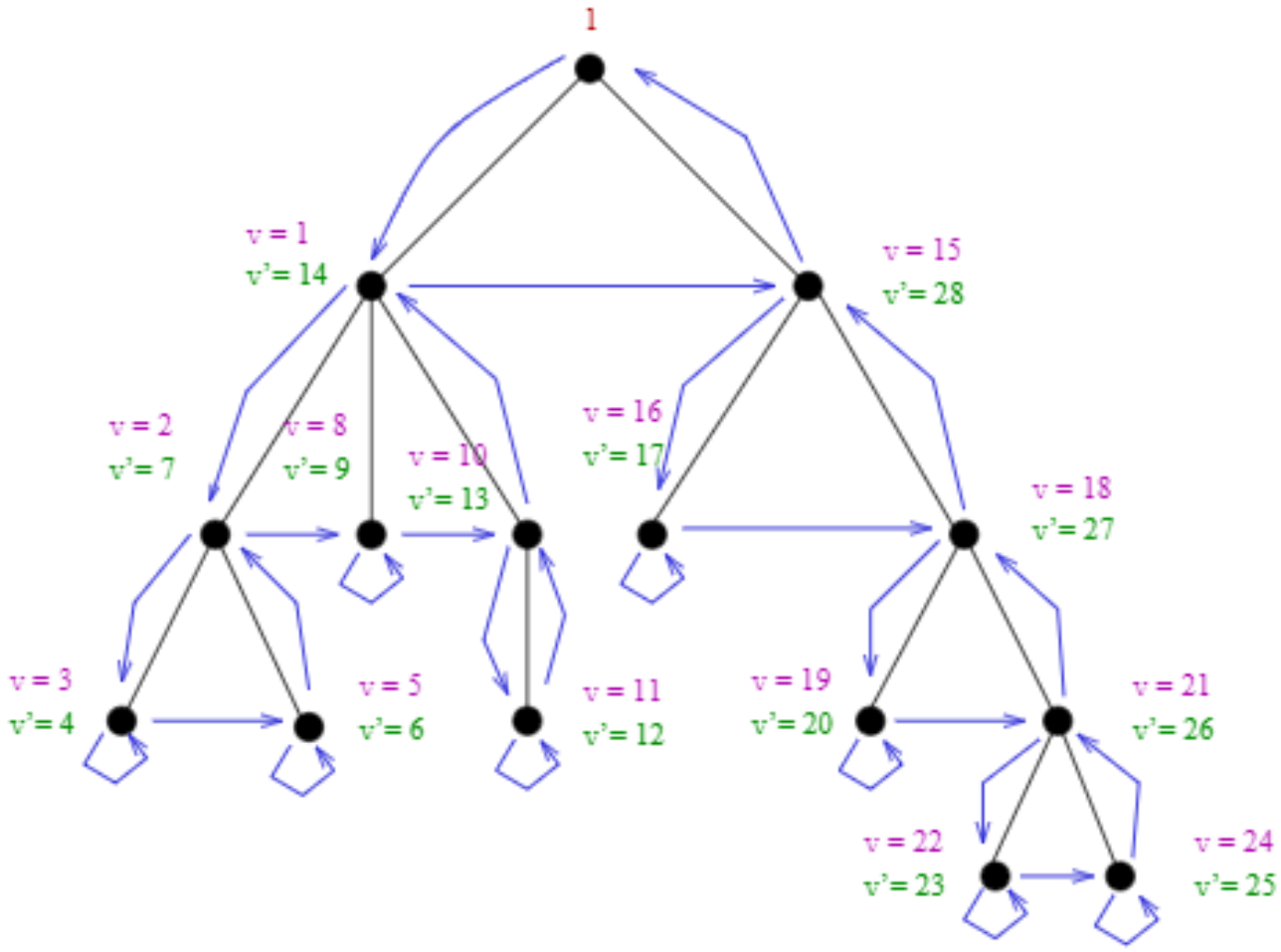
Pour tout sommet u, ν_u est pair ssi u se trouve à un niveau pair et ν'_u est pair ssi u se trouve à un niveau impair.

Corollaire.

Pour tout sommet u, si ν_u est pair (resp. impair) alors ν'_u est impair (resp. pair).

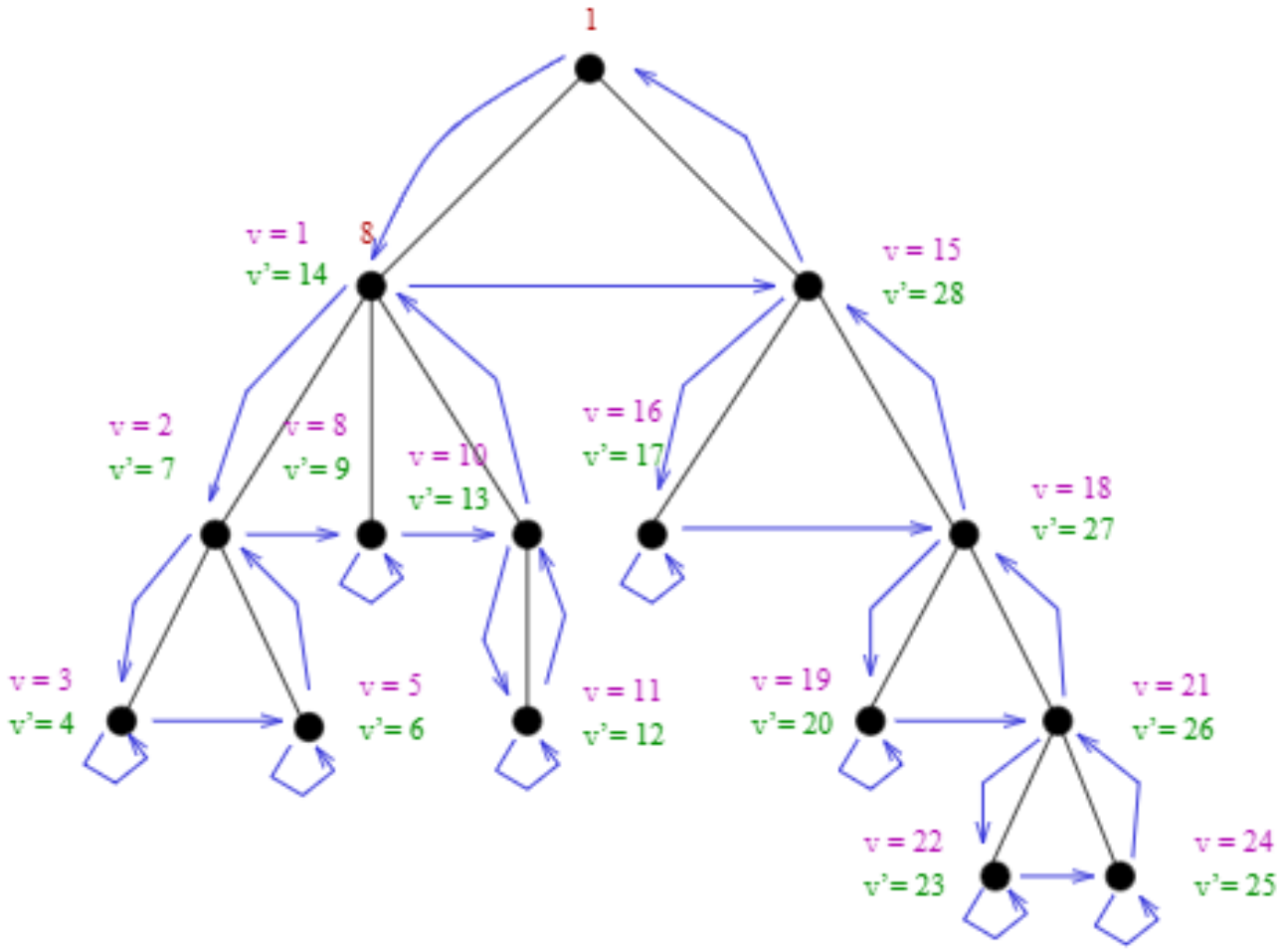
Enumération distribuée

Algorithme *Trav* :



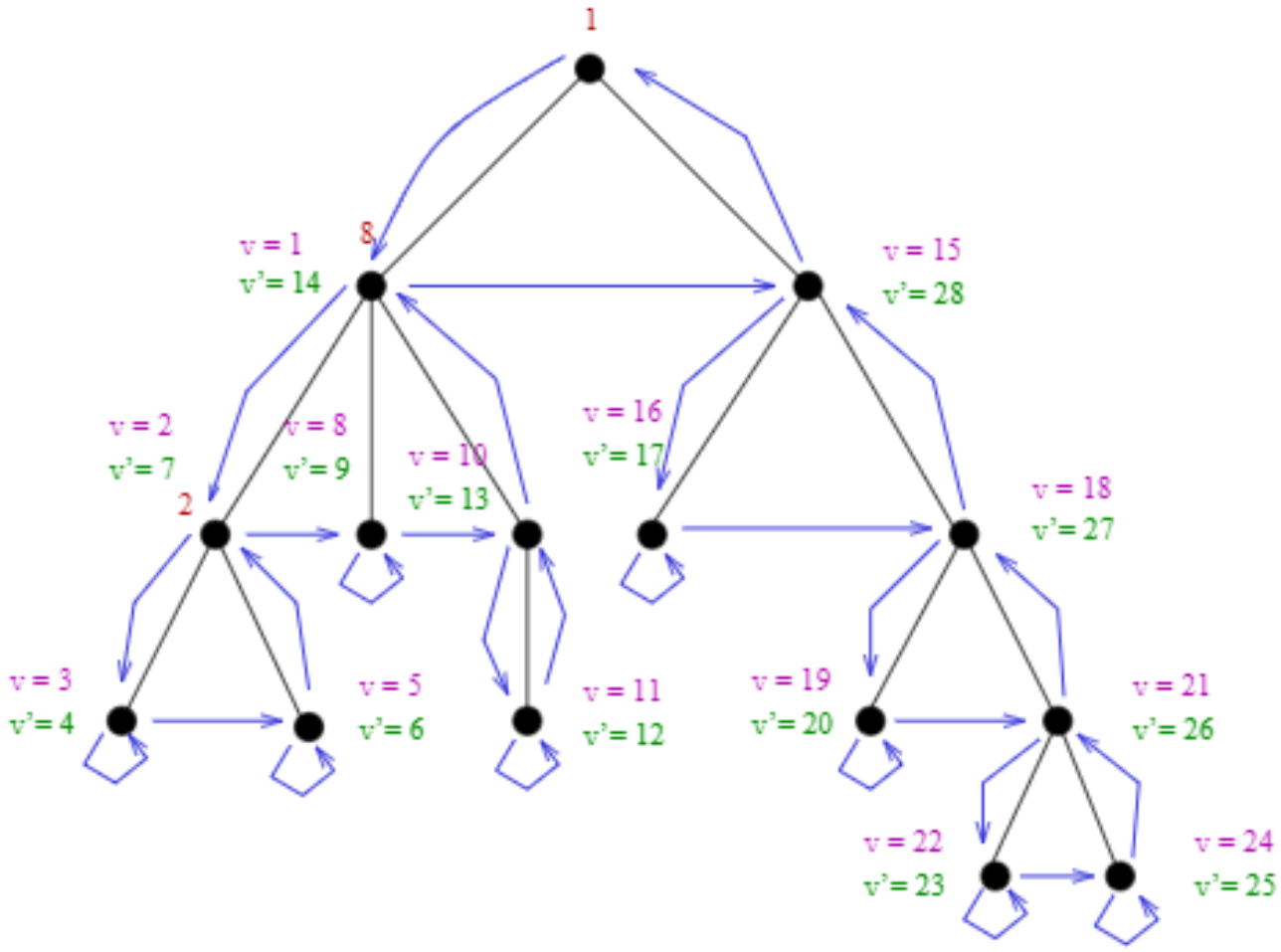
Enumération distribuée

Algorithme *Trav* :



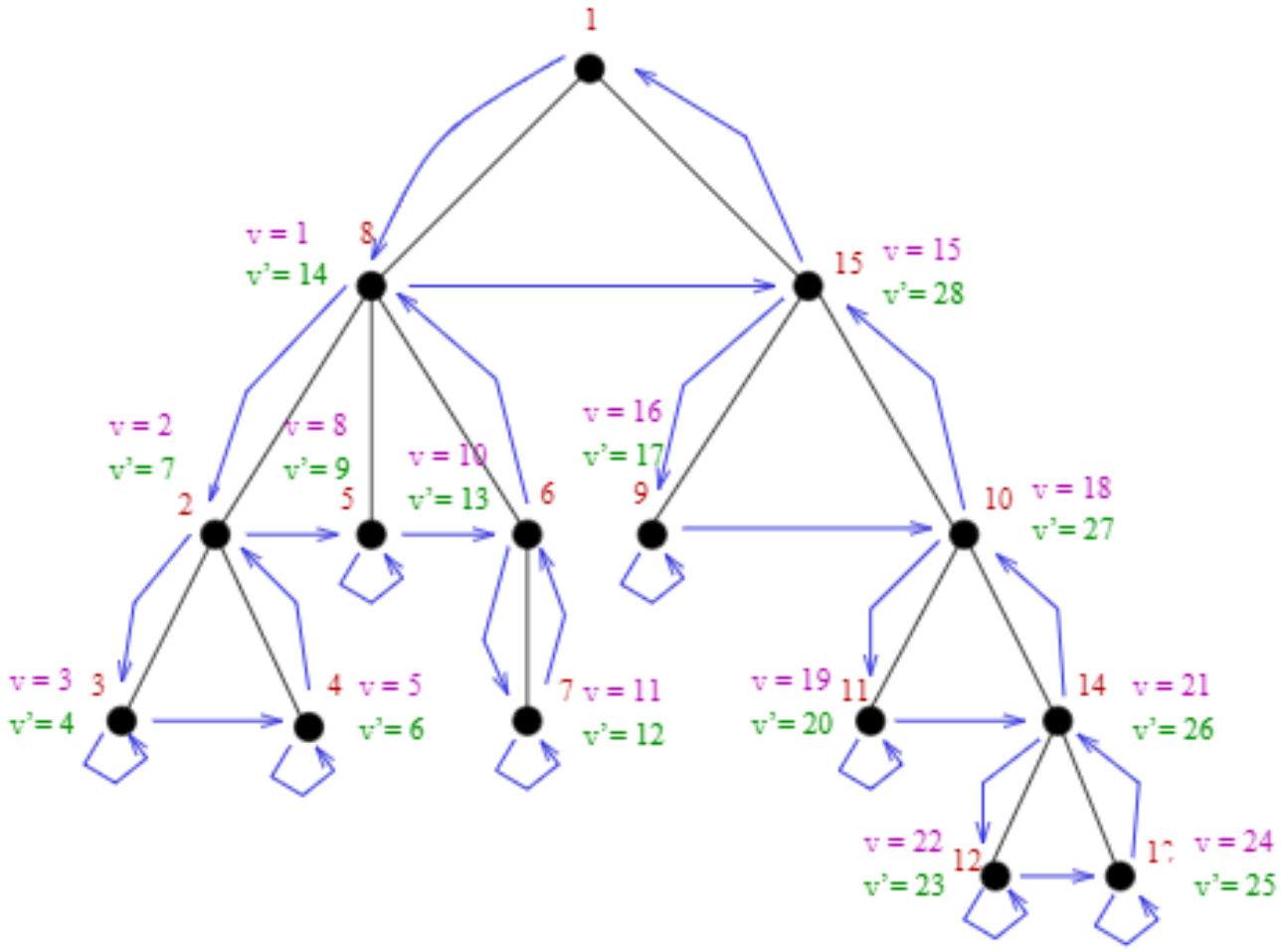
Enumération distribuée

Algorithme *Trav* :



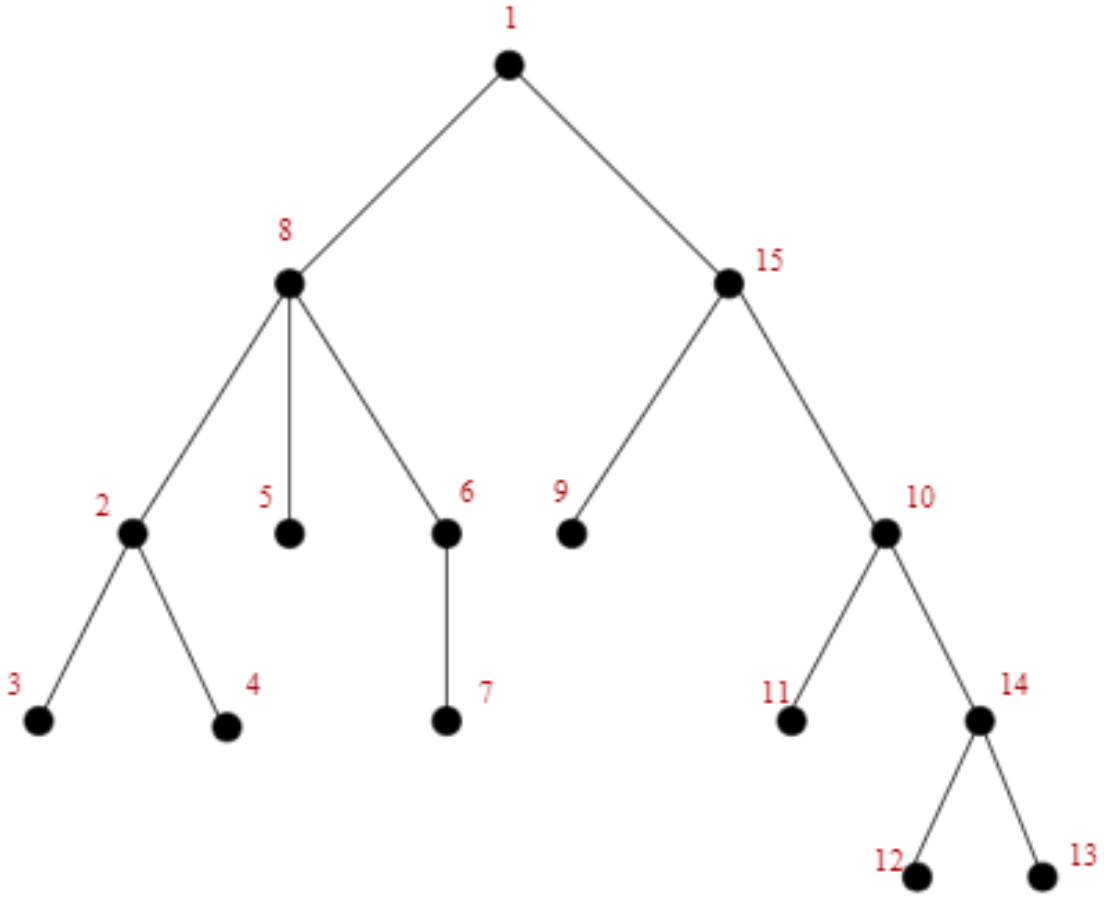
Enumération distribuée

Algorithme *Trav* :



Enumération distribuée

Algorithme *Trav* :



Enumération distribuée

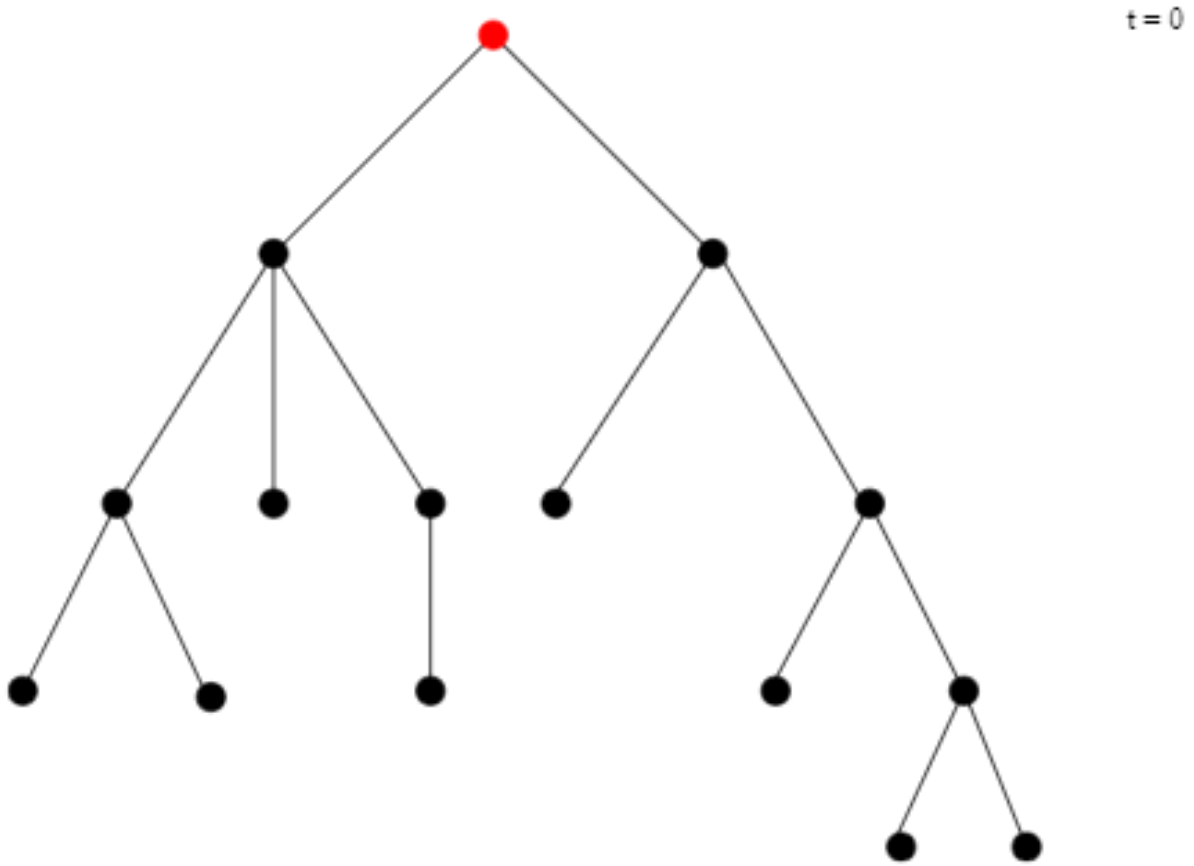
Lemme.

Soit G un graphe connexe. La distance, dans G, entre deux sommets ayant des numéros consécutifs est d'au plus 3.

$\Rightarrow$ l'algorithme est optimal : il n'existe pas d'énumération des noeuds telle que deux noeuds ayant des numéros consécutifs sont à distance au plus 2.

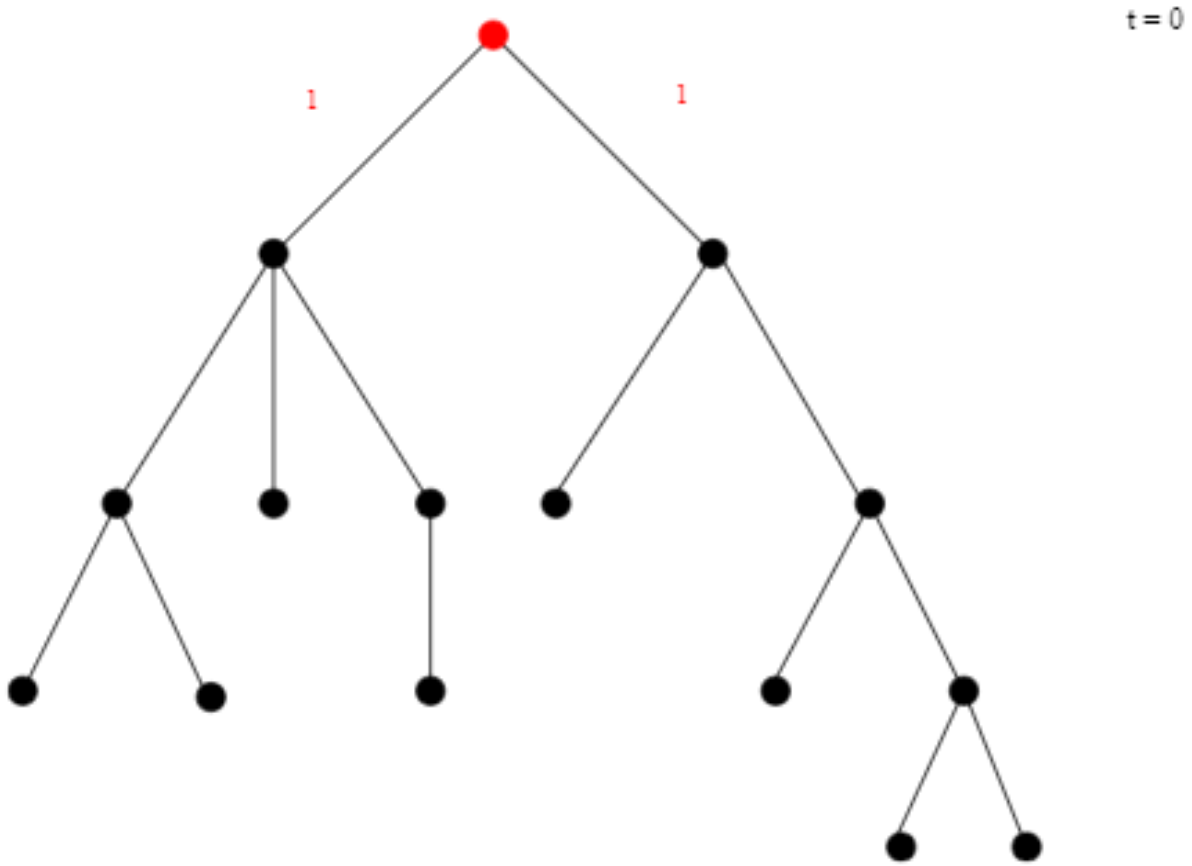
Application 1 : Calcul des plus courts chemins

1. Algorithme *DistCal* : Calcul des distance du leader aux autres sommets :



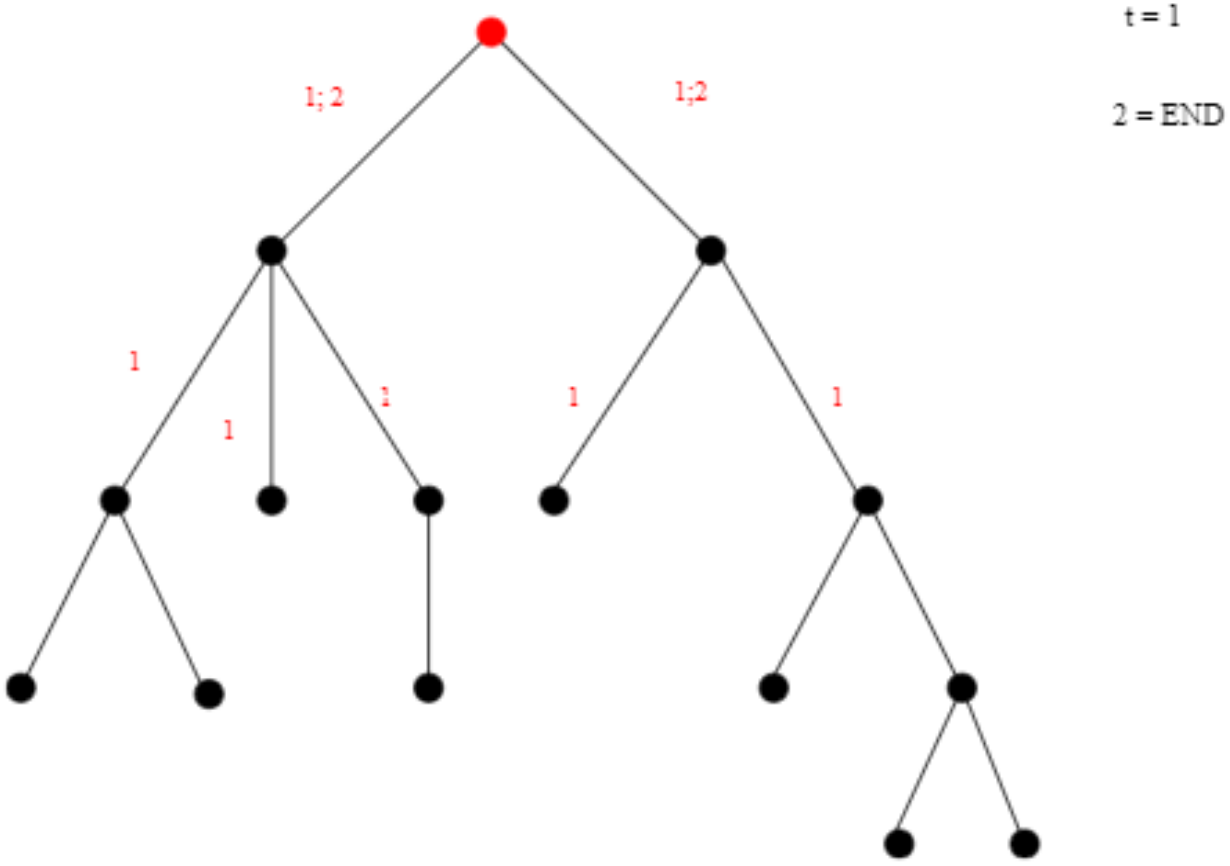
Application 1 : Calcul des plus courts chemins

1. Algorithme *DistCal* : Calcul des distance du leader aux autres sommets :



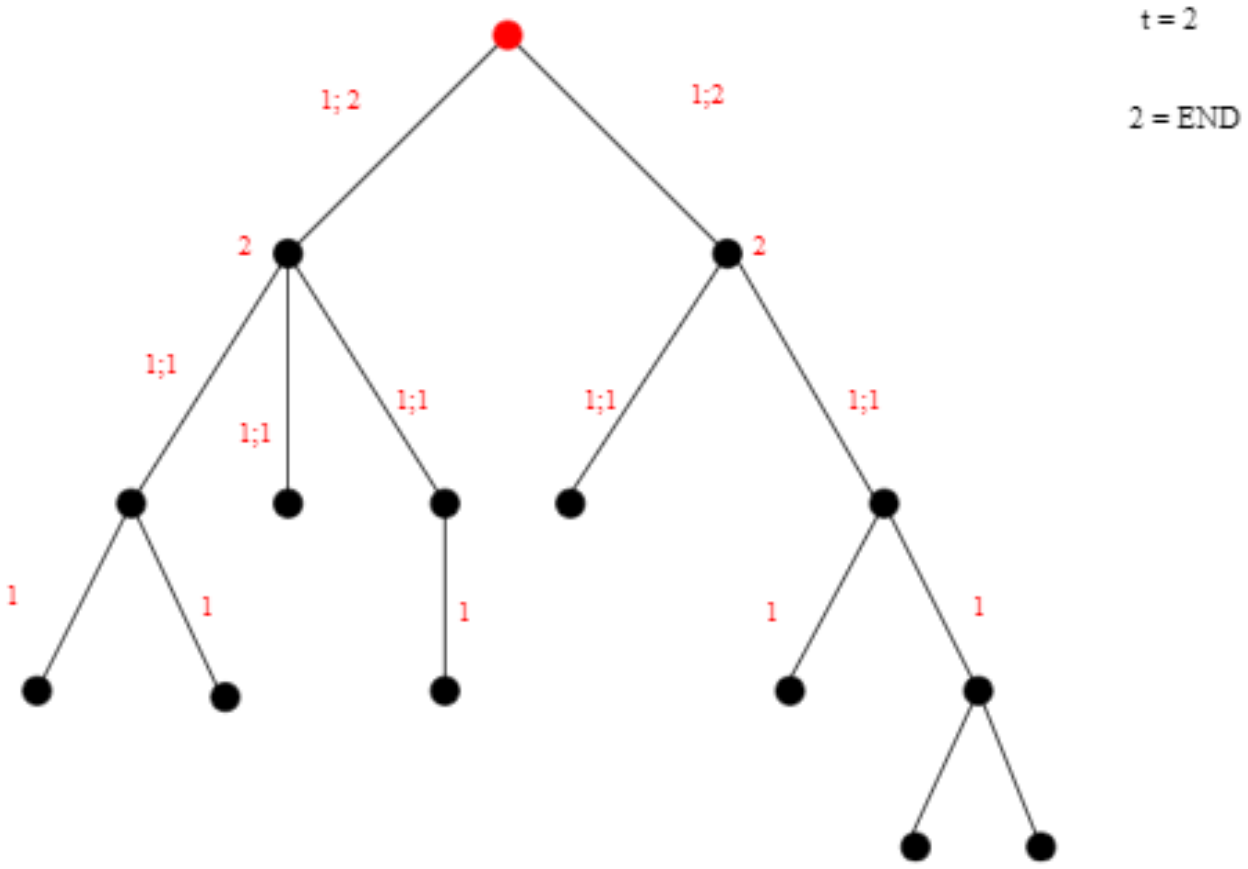
Application 1 : Calcul des plus courts chemins

1. Algorithme *DistCal* : Calcul des distance du leader aux autres sommets :



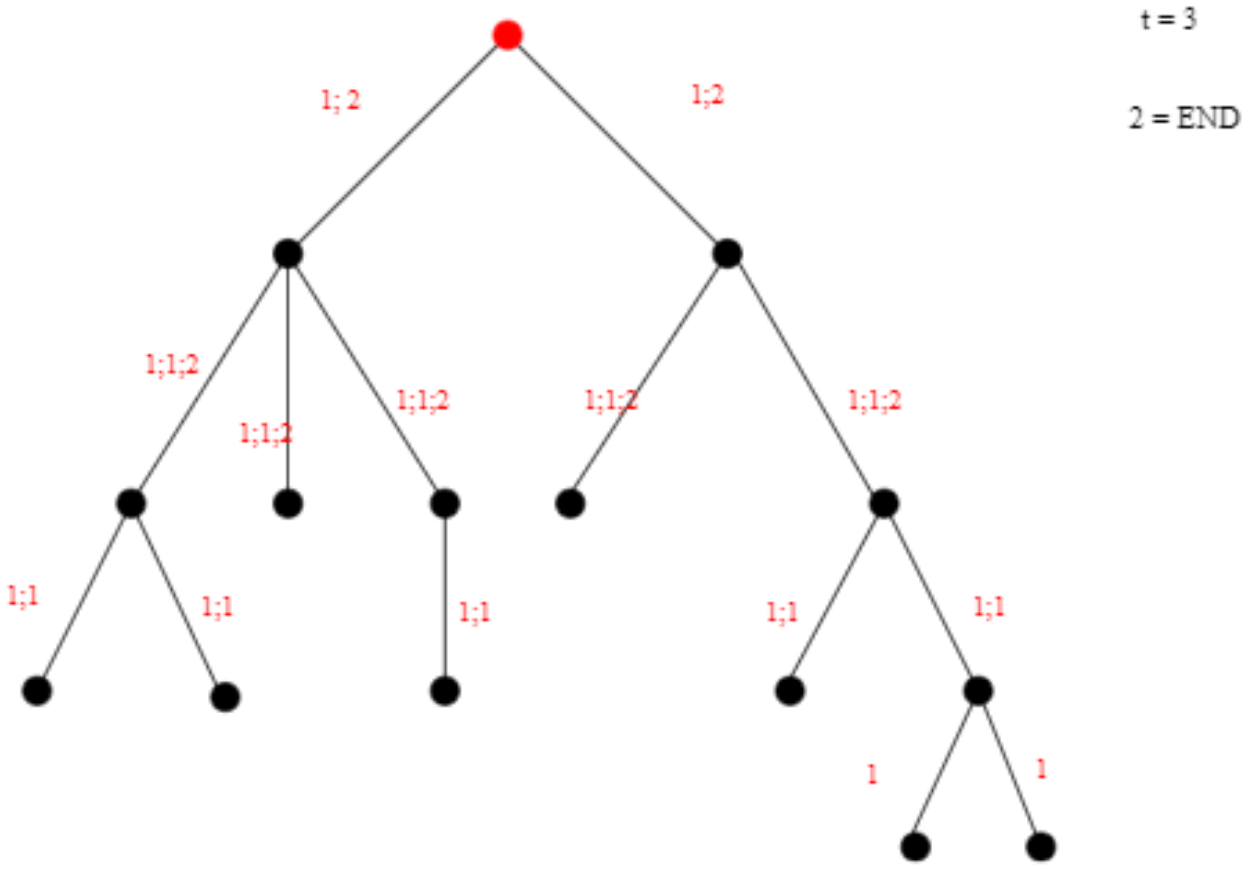
Application 1 : Calcul des plus courts chemins

1. Algorithme *DistCal* : Calcul des distance du leader aux autres sommets :



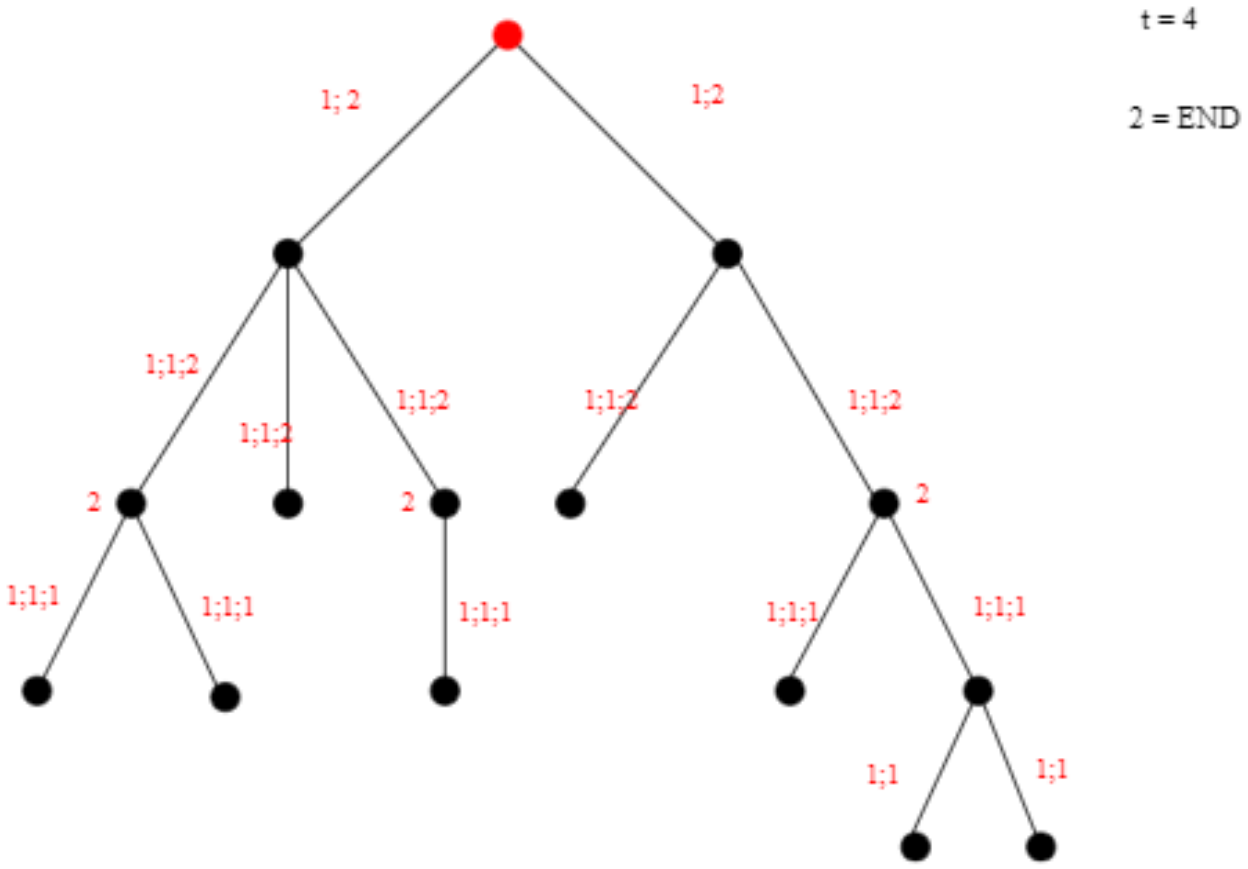
Application 1 : Calcul des plus courts chemins

1. Algorithme *DistCal* : Calcul des distance du leader aux autres sommets :



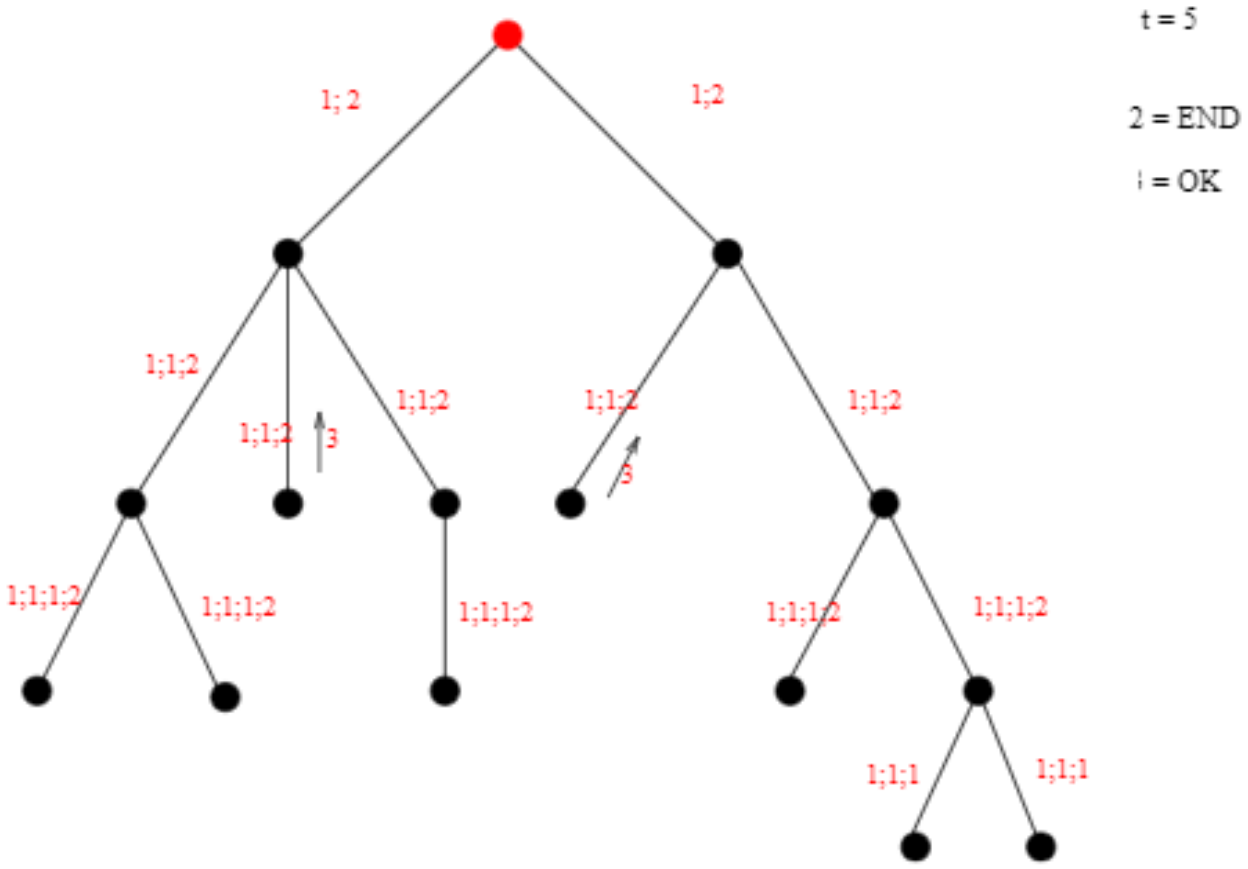
Application 1 : Calcul des plus courts chemins

1. Algorithme *DistCal* : Calcul des distance du leader aux autres sommets :



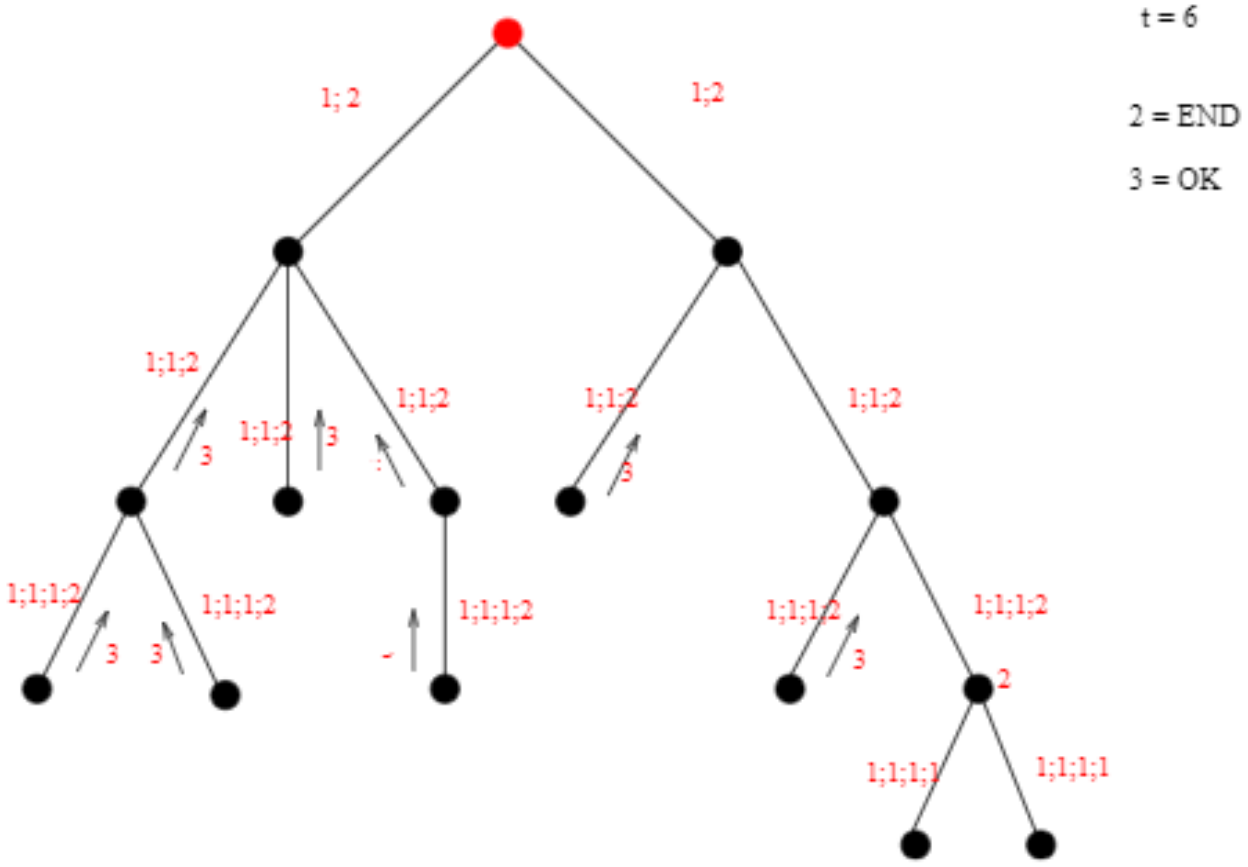
Application 1 : Calcul des plus courts chemins

1. Algorithme *DistCal* : Calcul des distance du leader aux autres sommets :



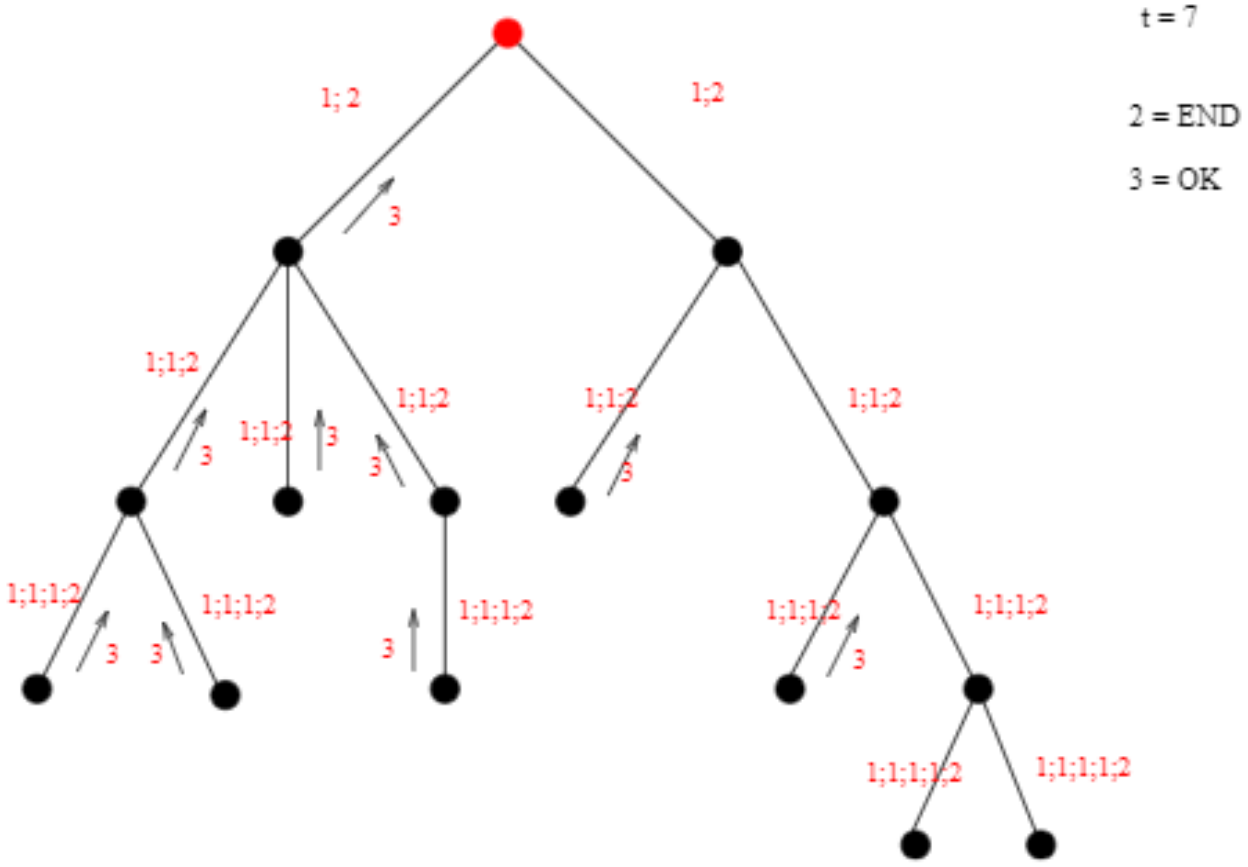
Application 1 : Calcul des plus courts chemins

1. Algorithme *DistCal* : Calcul des distance du leader aux autres sommets :



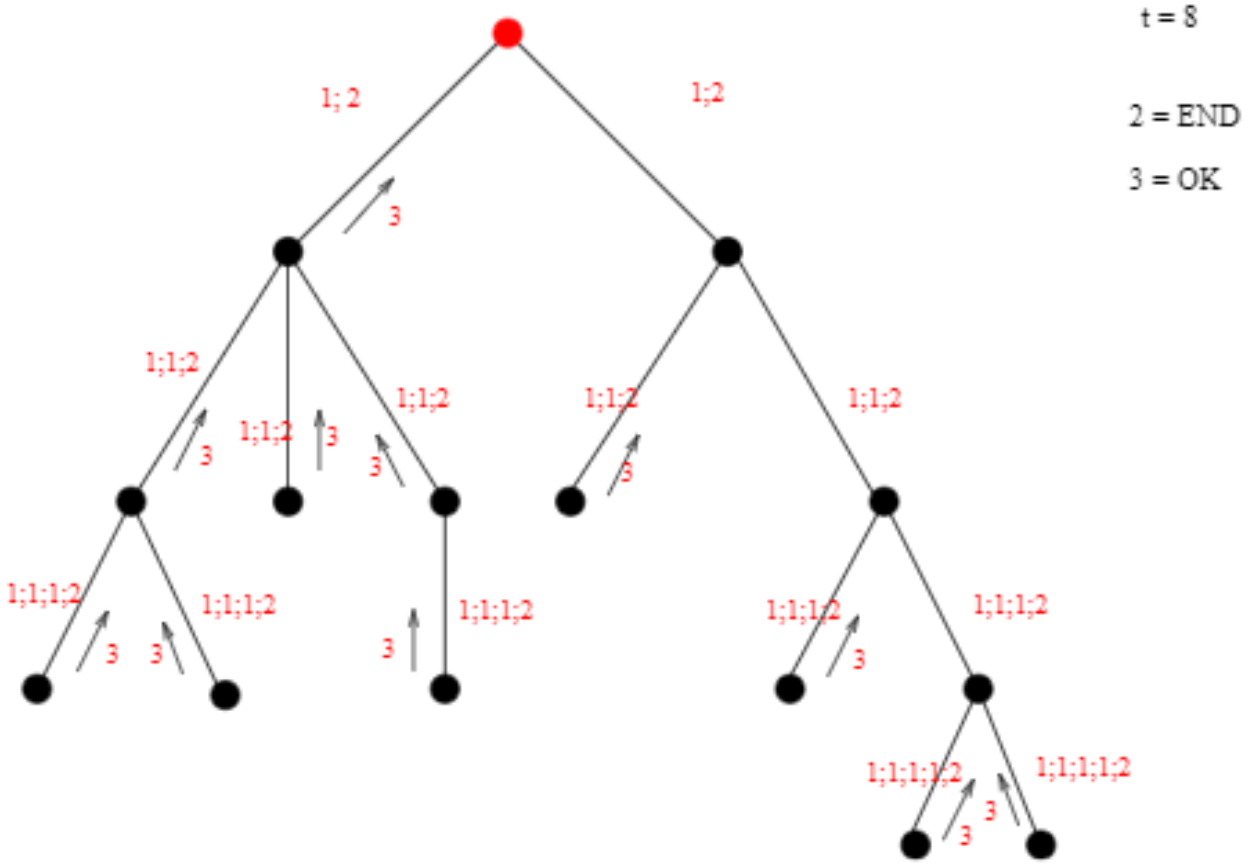
Application 1 : Calcul des plus courts chemins

1. Algorithme *DistCal* : Calcul des distance du leader aux autres sommets :



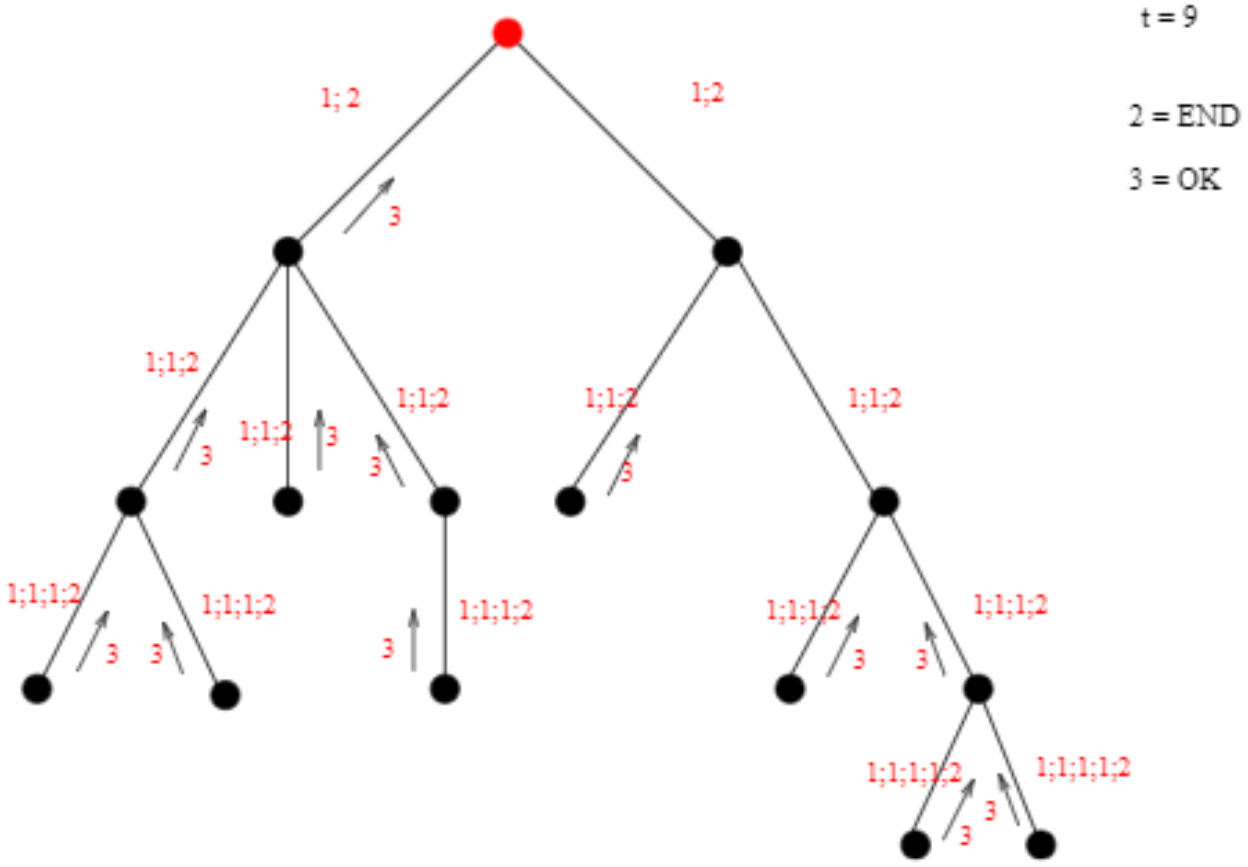
Application 1 : Calcul des plus courts chemins

1. Algorithme *DistCal* : Calcul des distance du leader aux autres sommets :



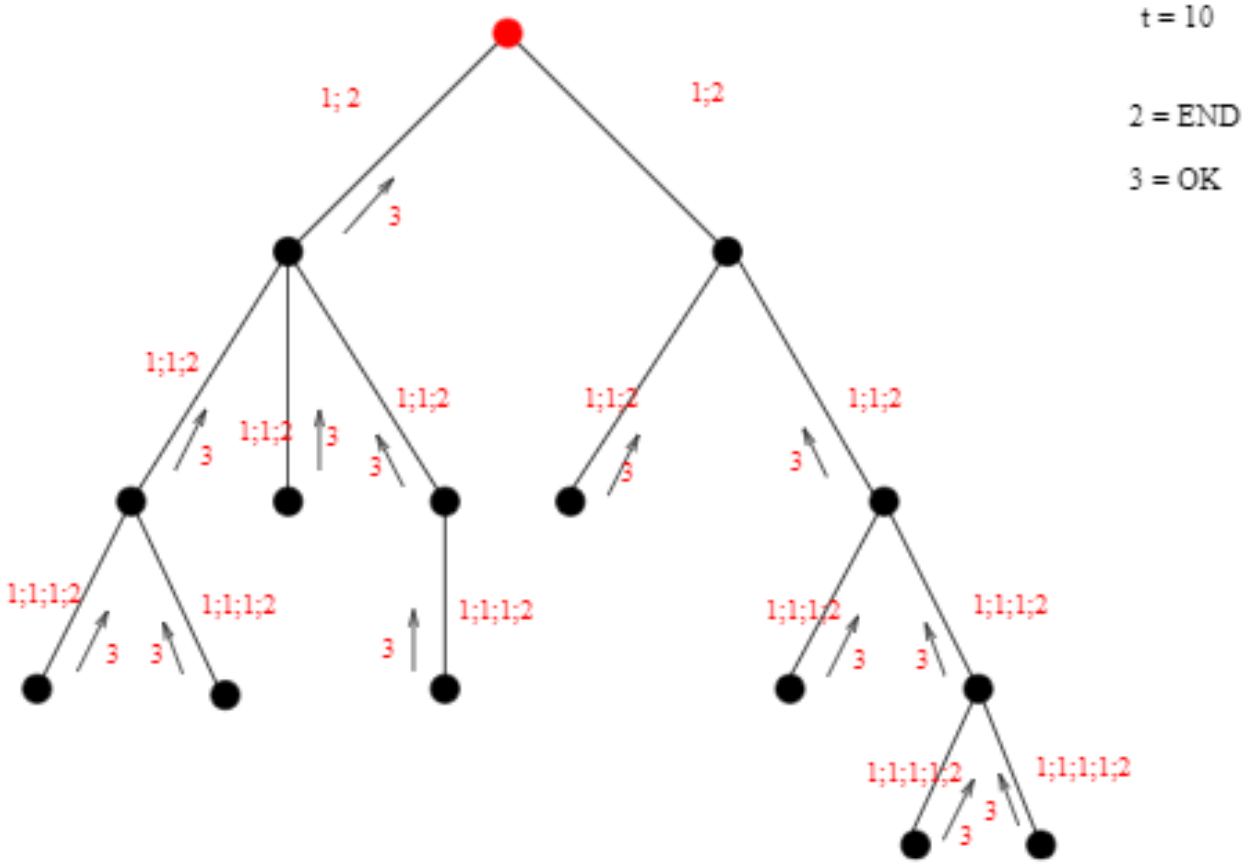
Application 1 : Calcul des plus courts chemins

1. Algorithme *DistCal* : Calcul des distance du leader aux autres sommets :



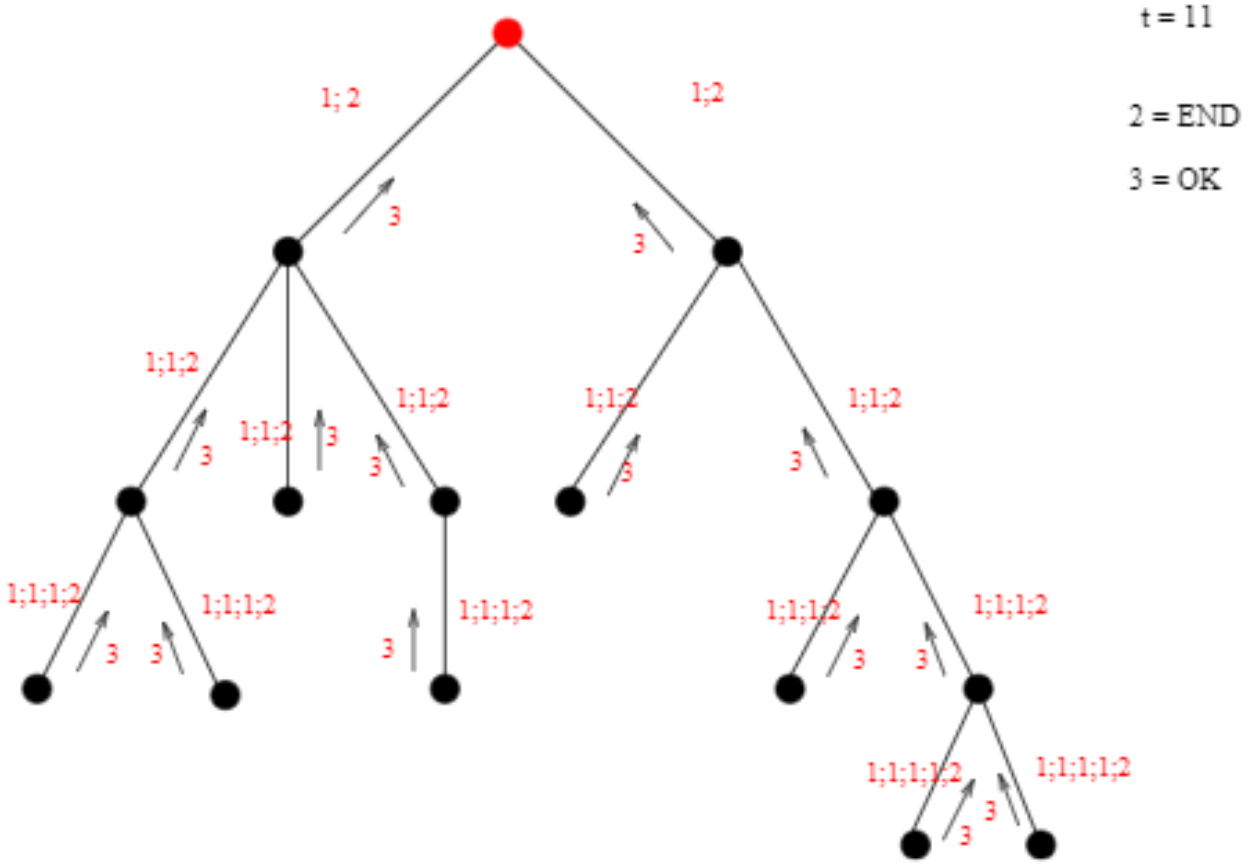
Application 1 : Calcul des plus courts chemins

1. Algorithme *DistCal* : Calcul des distance du leader aux autres sommets :



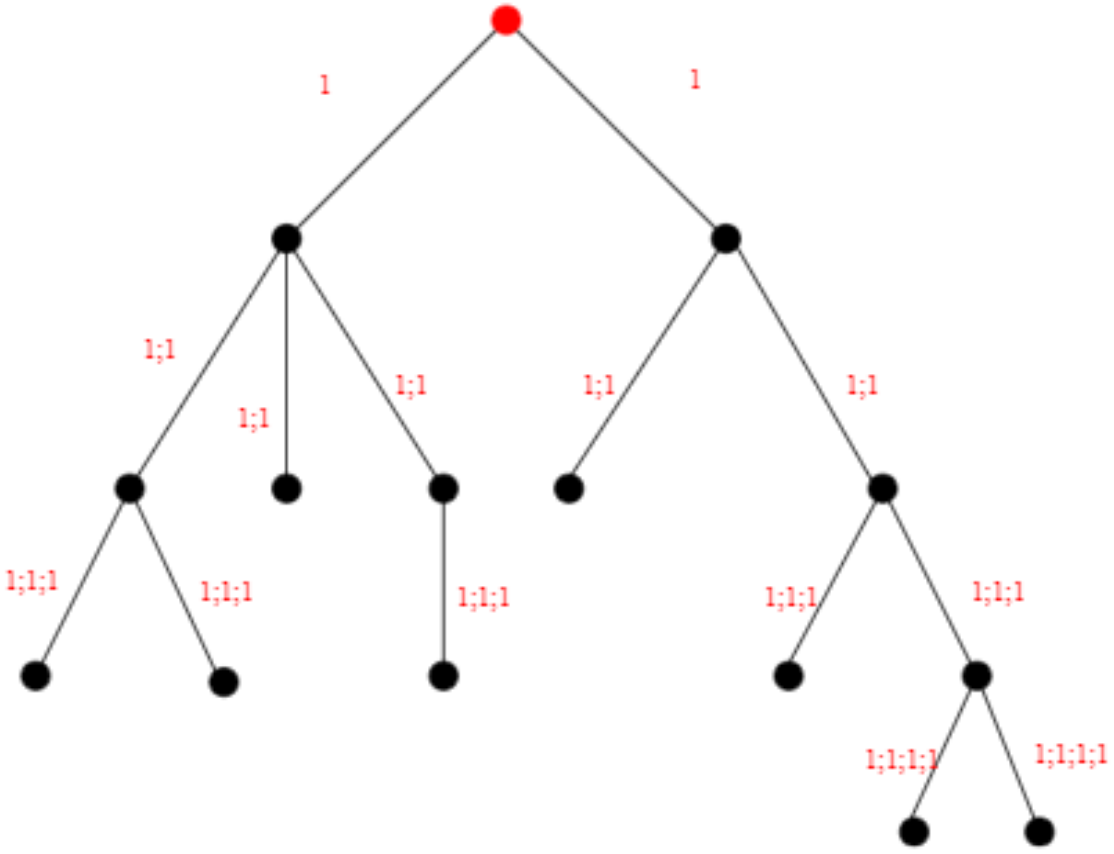
Application 1 : Calcul des plus courts chemins

1. Algorithme *DistCal* : Calcul des distance du leader aux autres sommets :



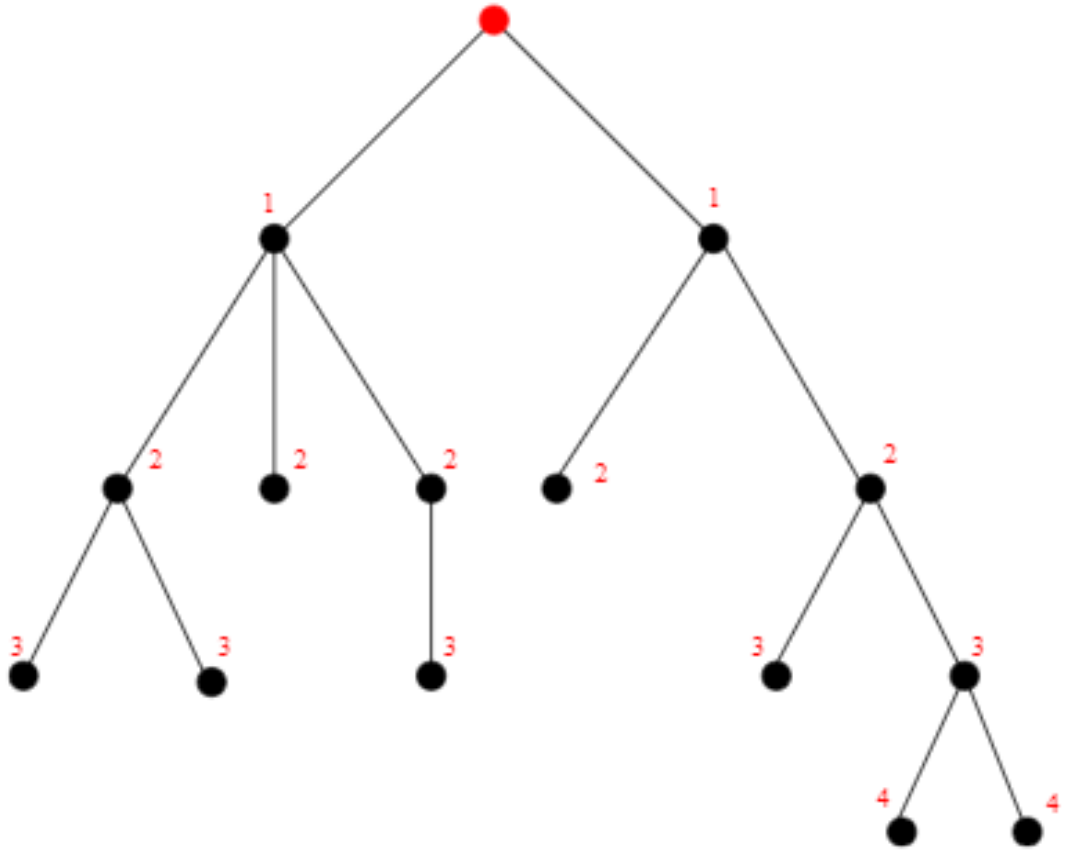
Application 1 : Calcul des plus courts chemins

1. Algorithme *DistCal* : Calcul des distance du leader aux autres sommets :



Application 1 : Calcul des plus courts chemins

1. Algorithme *DistCal* : Calcul des distance du leader aux autres sommets :



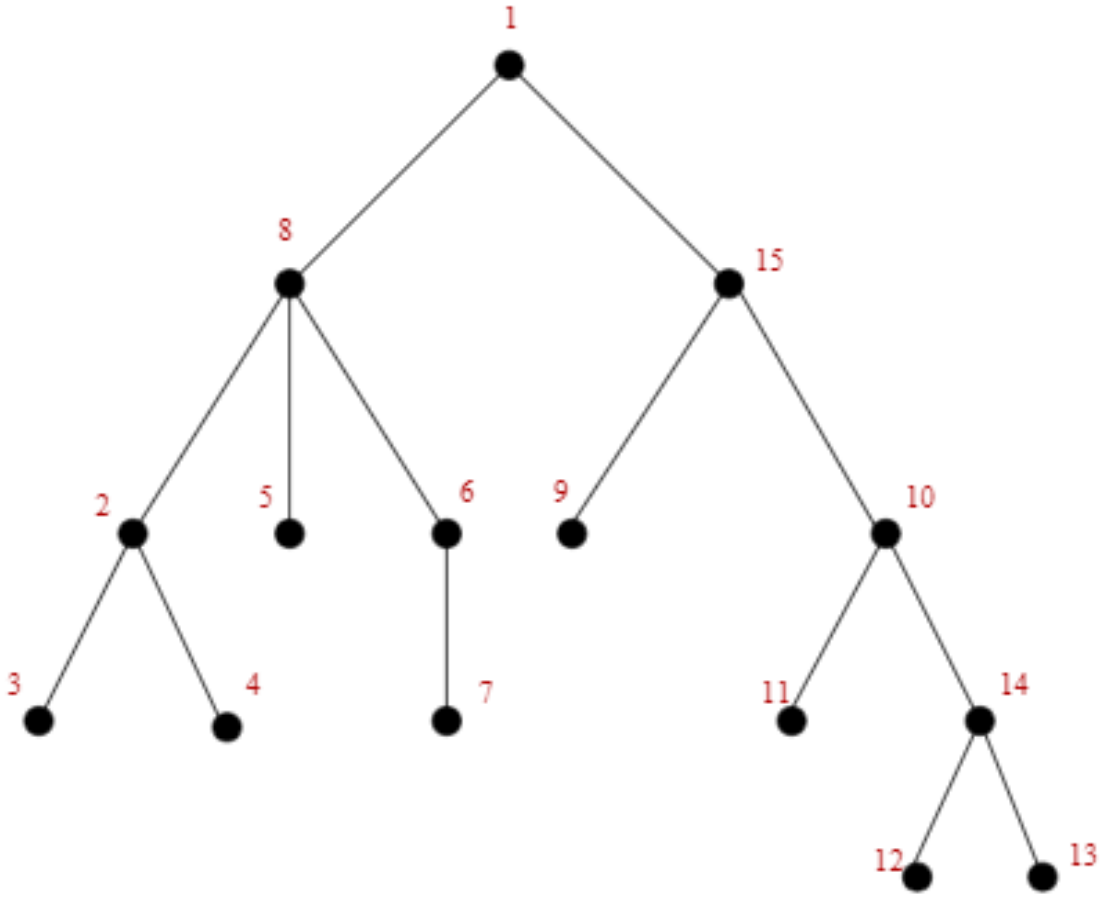
Application 1 : Calcul des plus courts chemins

Lemme.

DistCal permet à chaque sommet de connaître sa distance au Leader. Sa complexité en temps et sa complexité en bits sont $O(n)$.

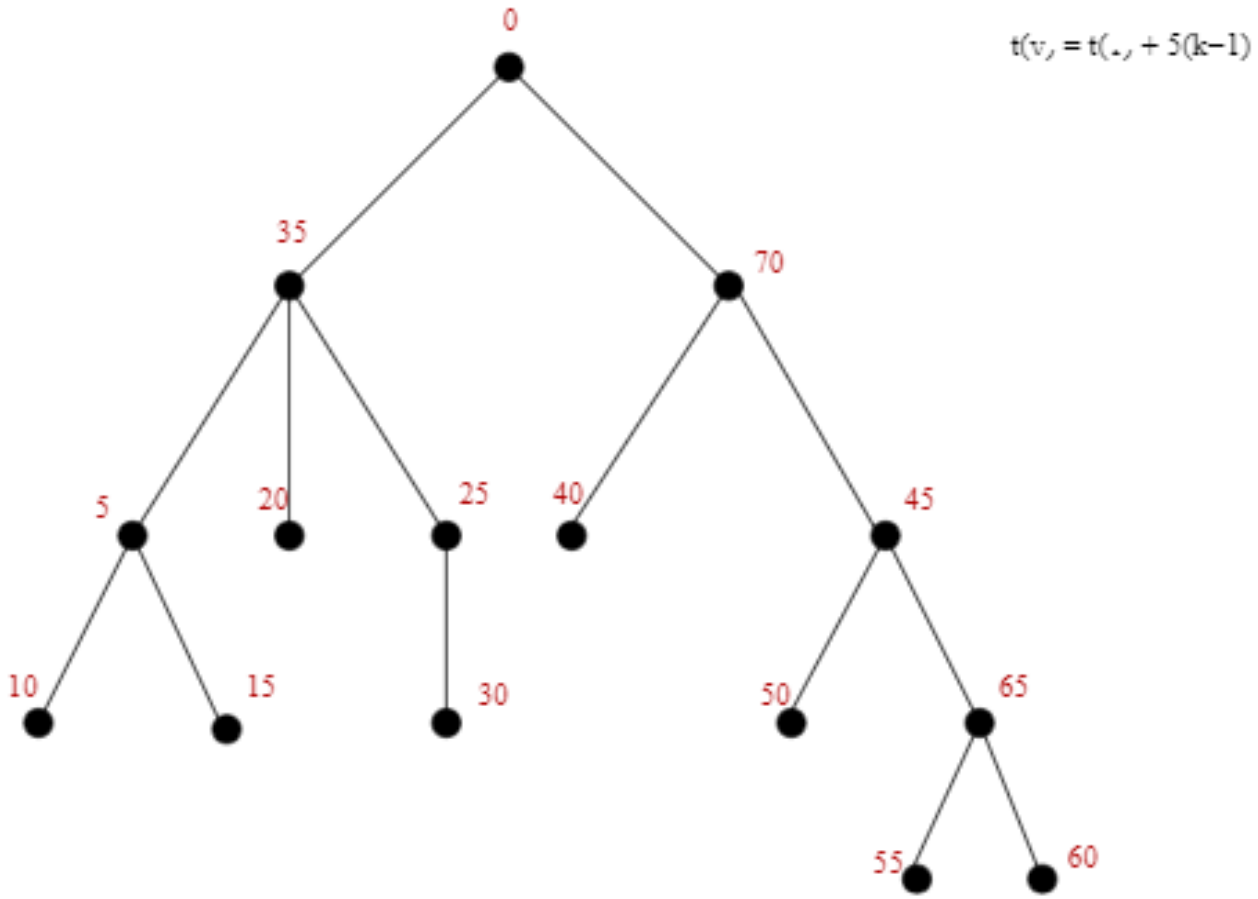
Application 1 : Calcul des plus courts chemins

2. APSP : Calcul des plus courts chemins :



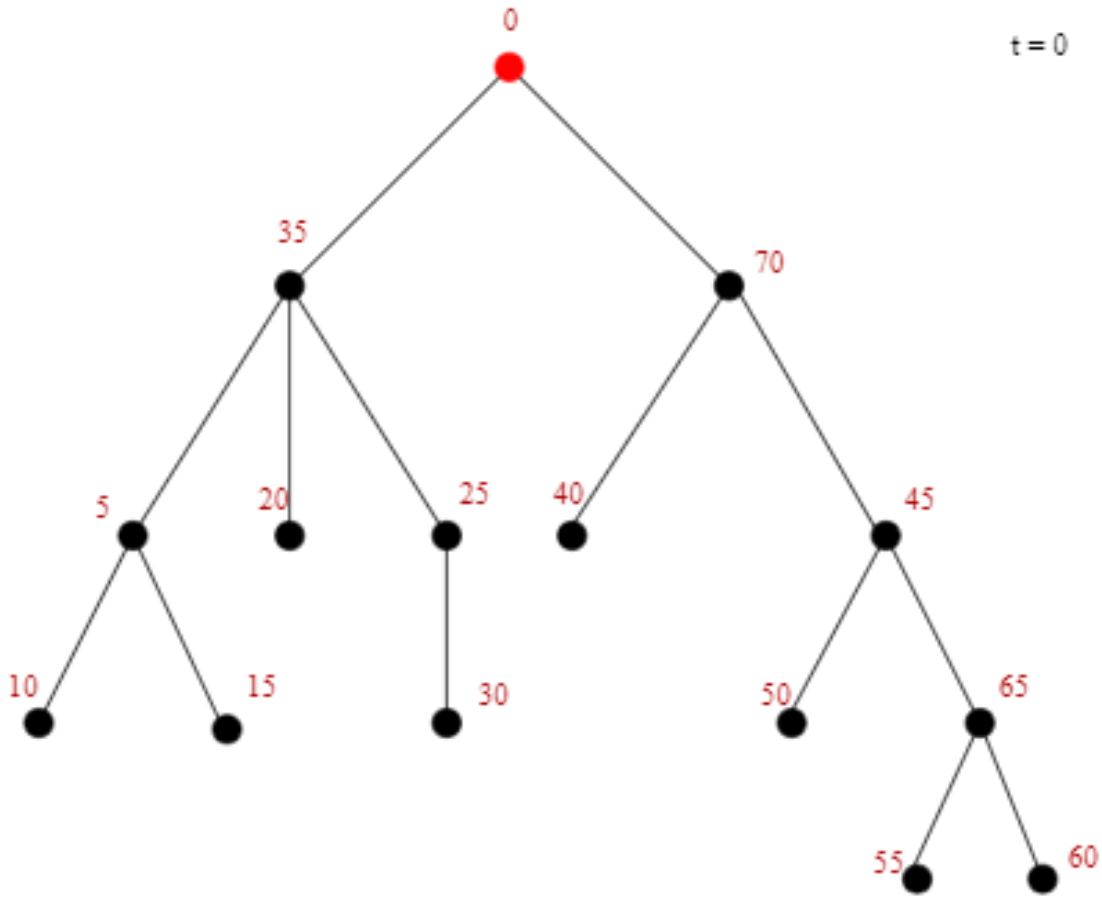
Application 1 : Calcul des plus courts chemins

2. APSP : Calcul des plus courts chemins :



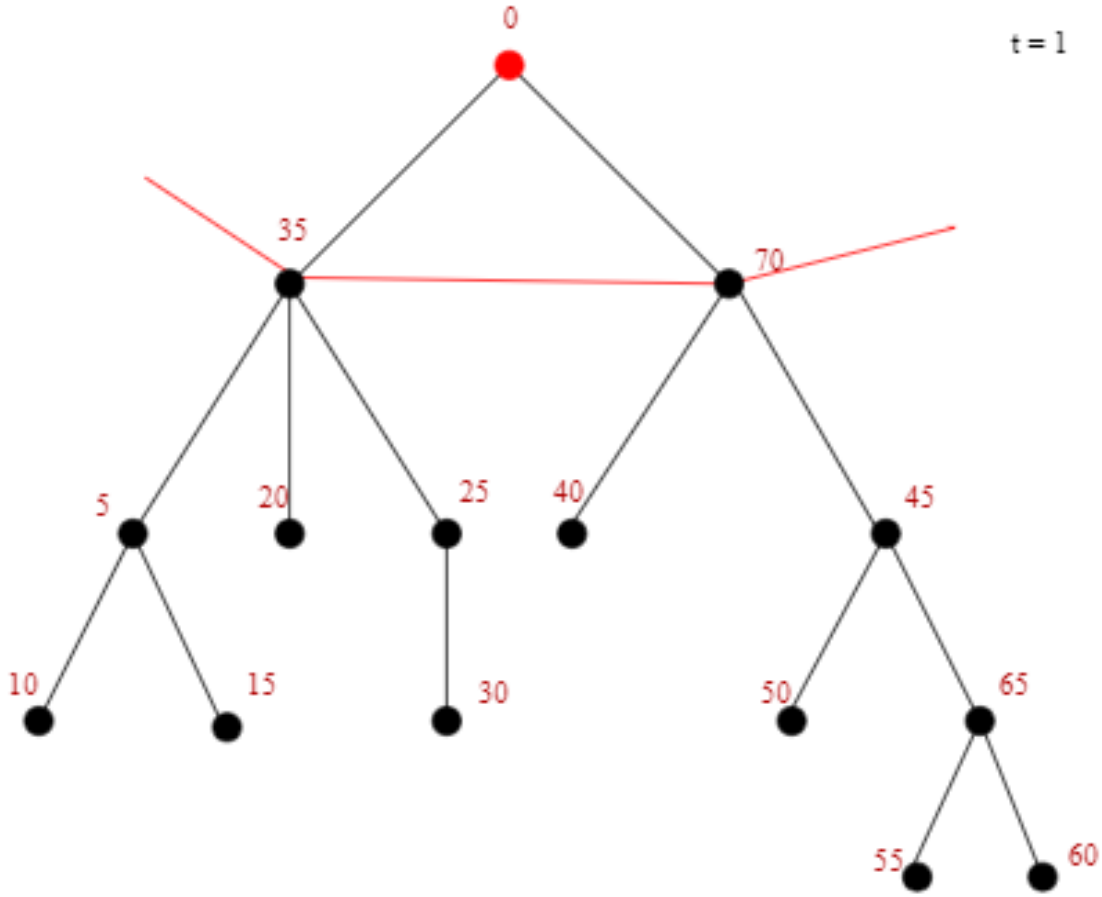
Application 1 : Calcul des plus courts chemins

2. APSP : Calcul des plus courts chemins :



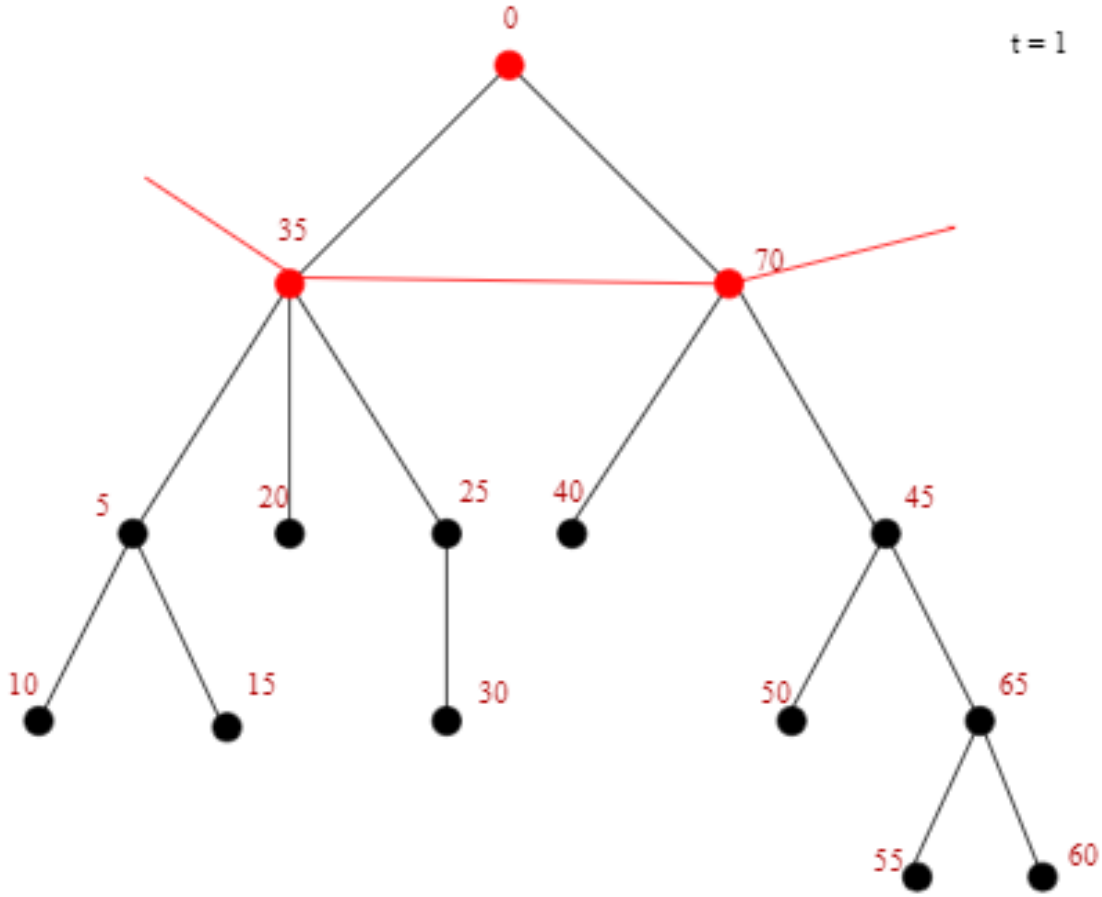
Application 1 : Calcul des plus courts chemins

2. APSP : Calcul des plus courts chemins :



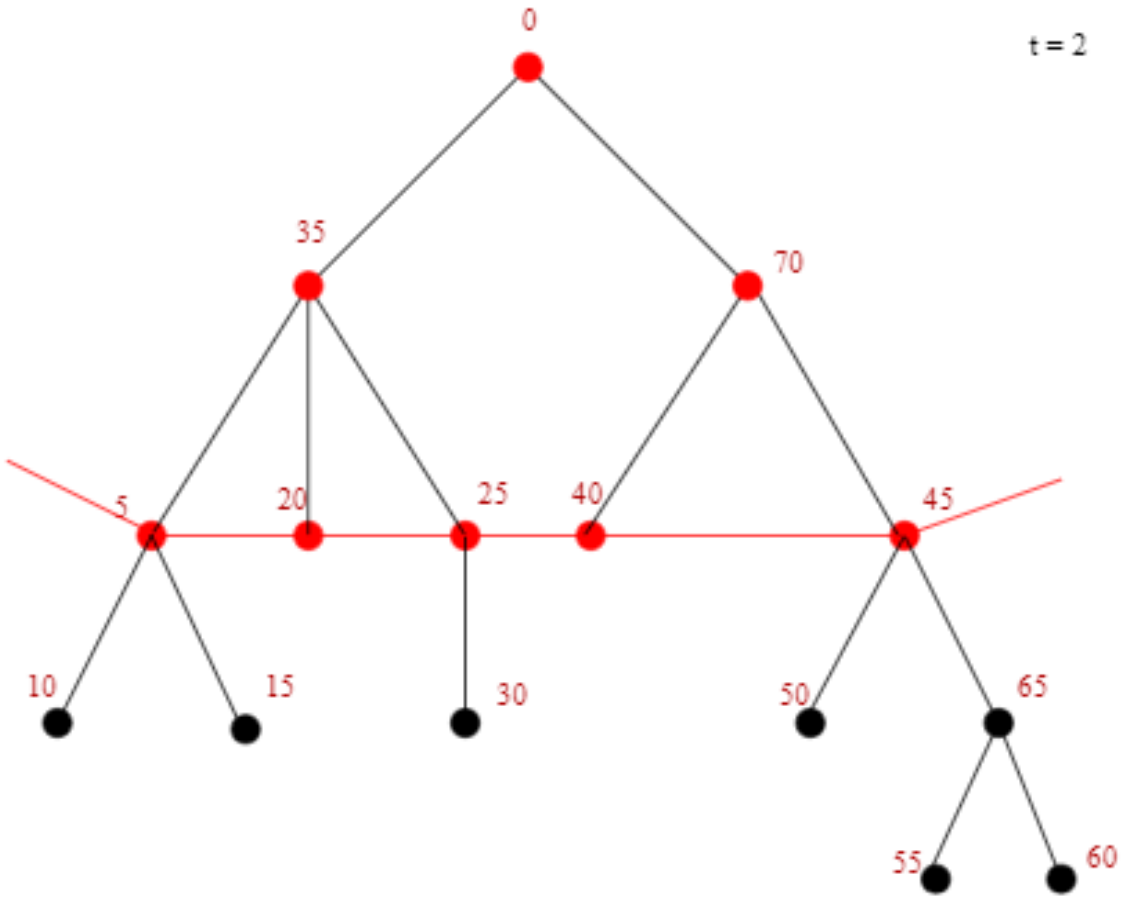
Application 1 : Calcul des plus courts chemins

2. APSP : Calcul des plus courts chemins :



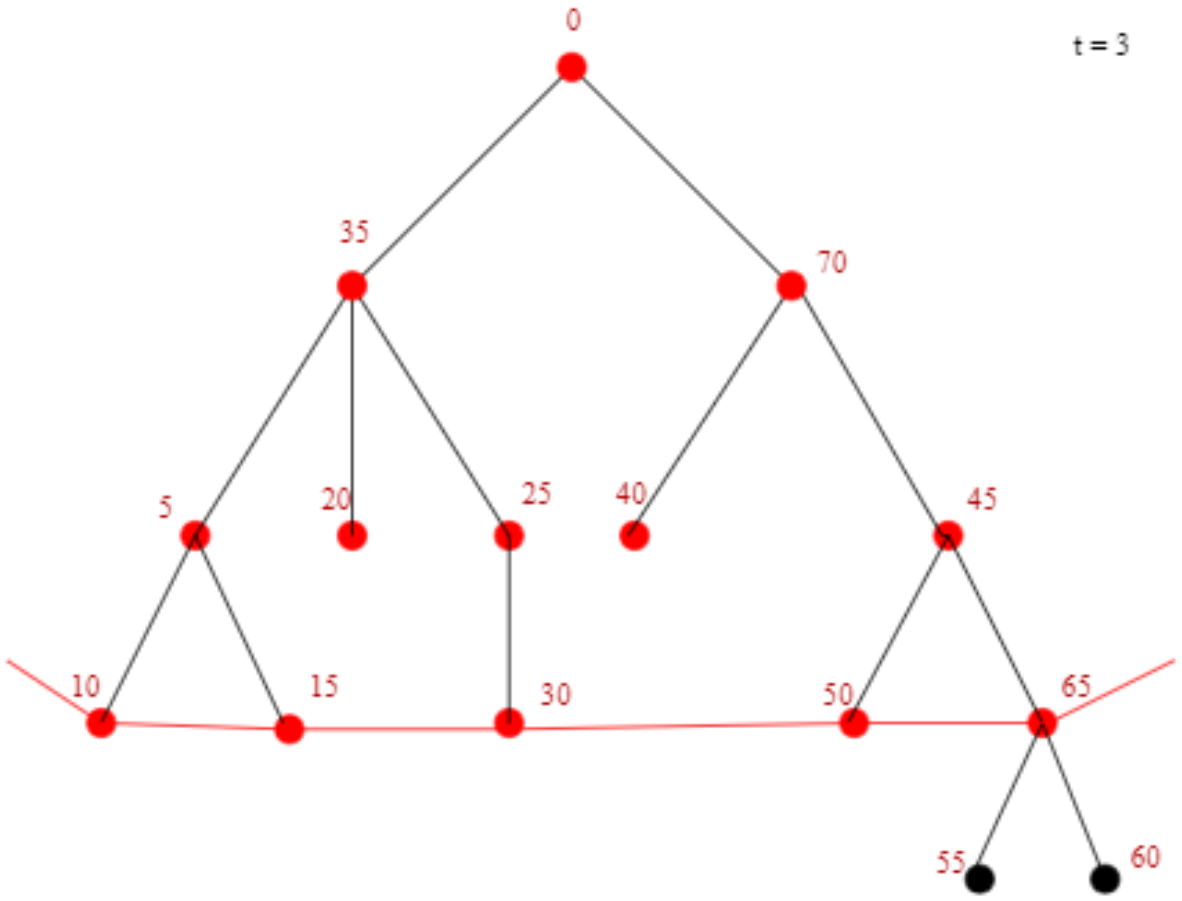
Application 1 : Calcul des plus courts chemins

2. APSP : Calcul des plus courts chemins :



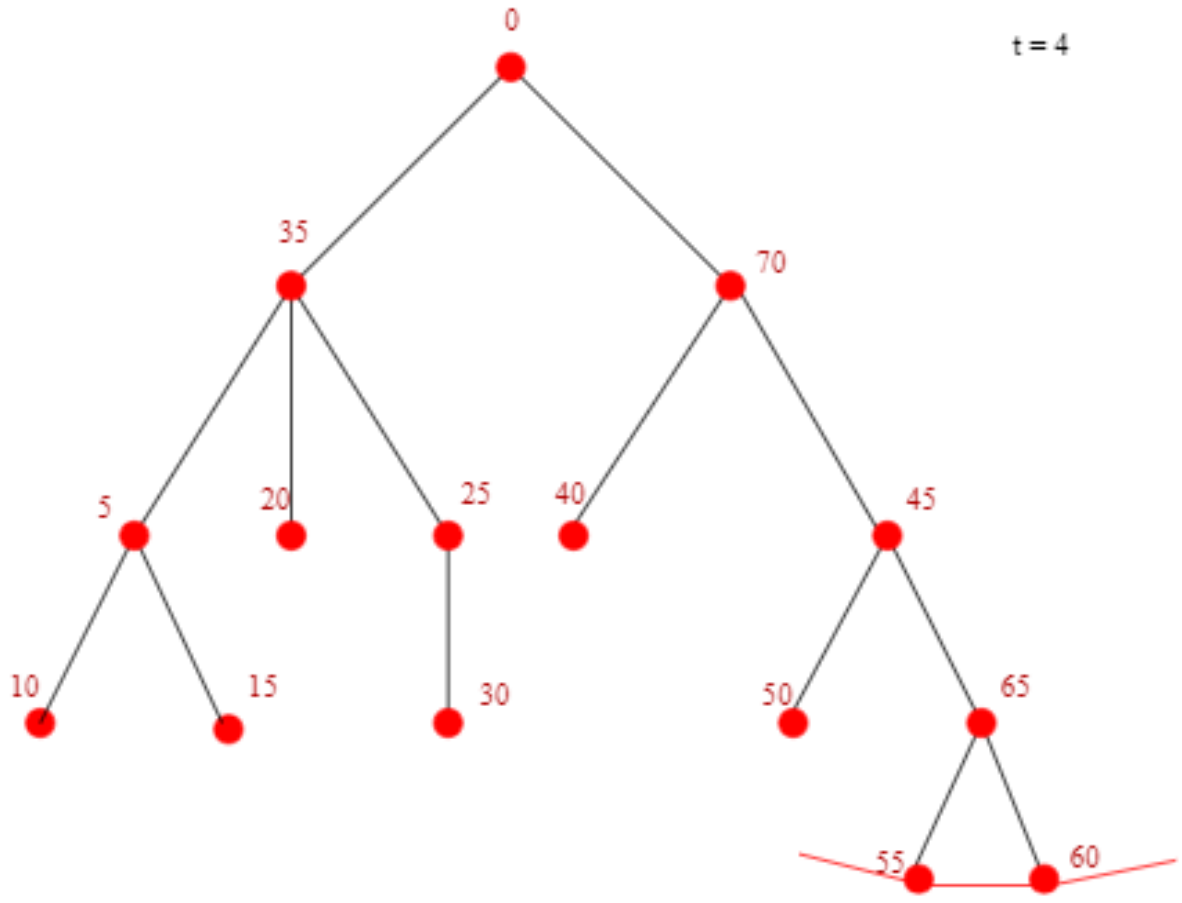
Application 1 : Calcul des plus courts chemins

2. APSP : Calcul des plus courts chemins :



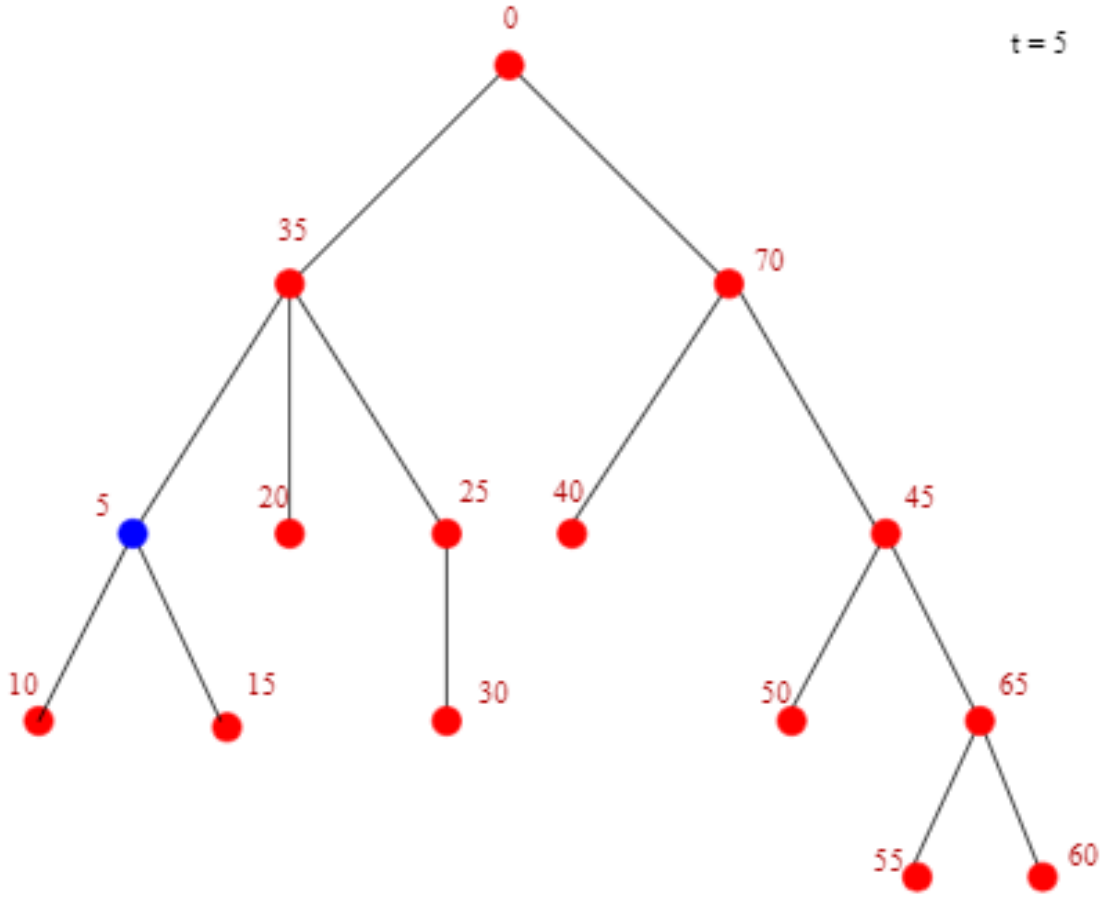
Application 1 : Calcul des plus courts chemins

2. APSP : Calcul des plus courts chemins :



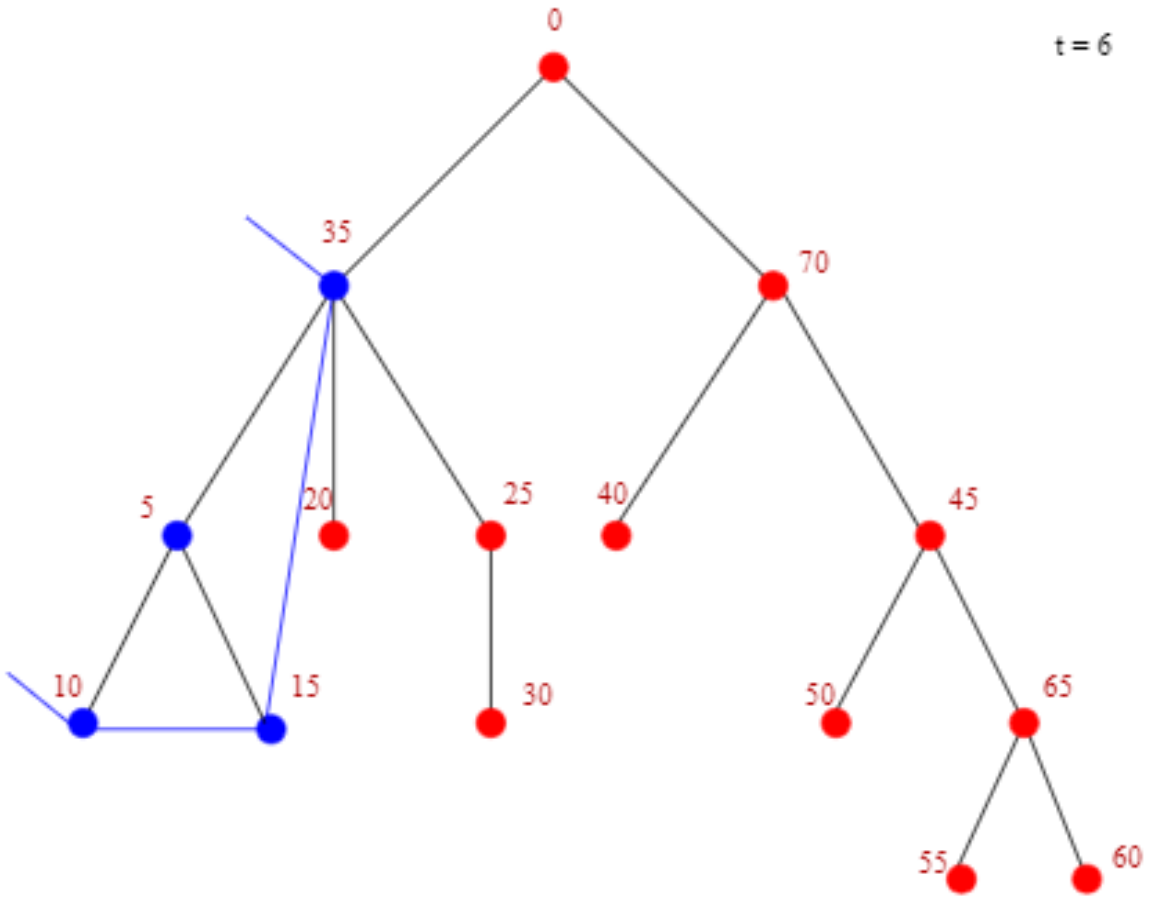
Application 1 : Calcul des plus courts chemins

2. APSP : Calcul des plus courts chemins :



Application 1 : Calcul des plus courts chemins

2. APSP : Calcul des plus courts chemins :



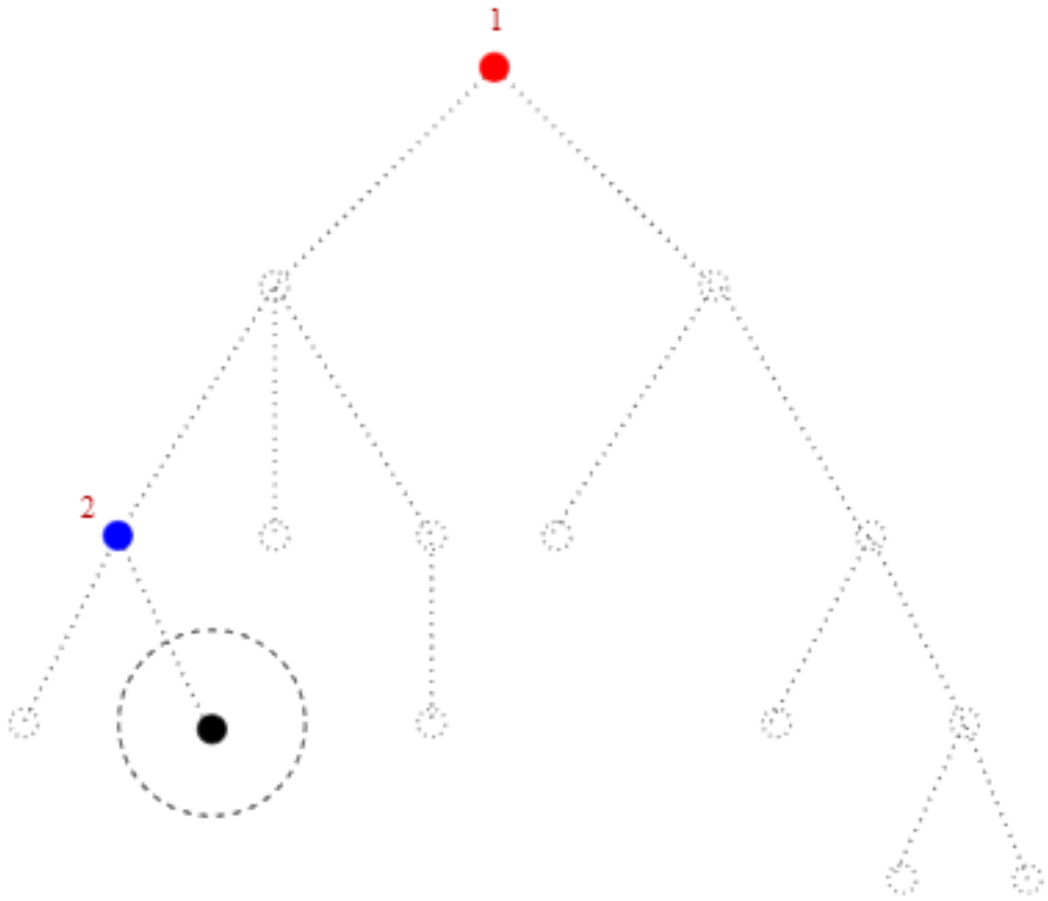
Application 1 : Calcul des plus courts chemins

Propriété. *Si v initialise une vague alors tout sommet w à distance d de v la reçoit à l'instant d et (éventuellement) à l'instant $d + 1$.*

Lemme. *Les vagues commencées par deux sommets consécutifs n'entrent jamais en collision.*

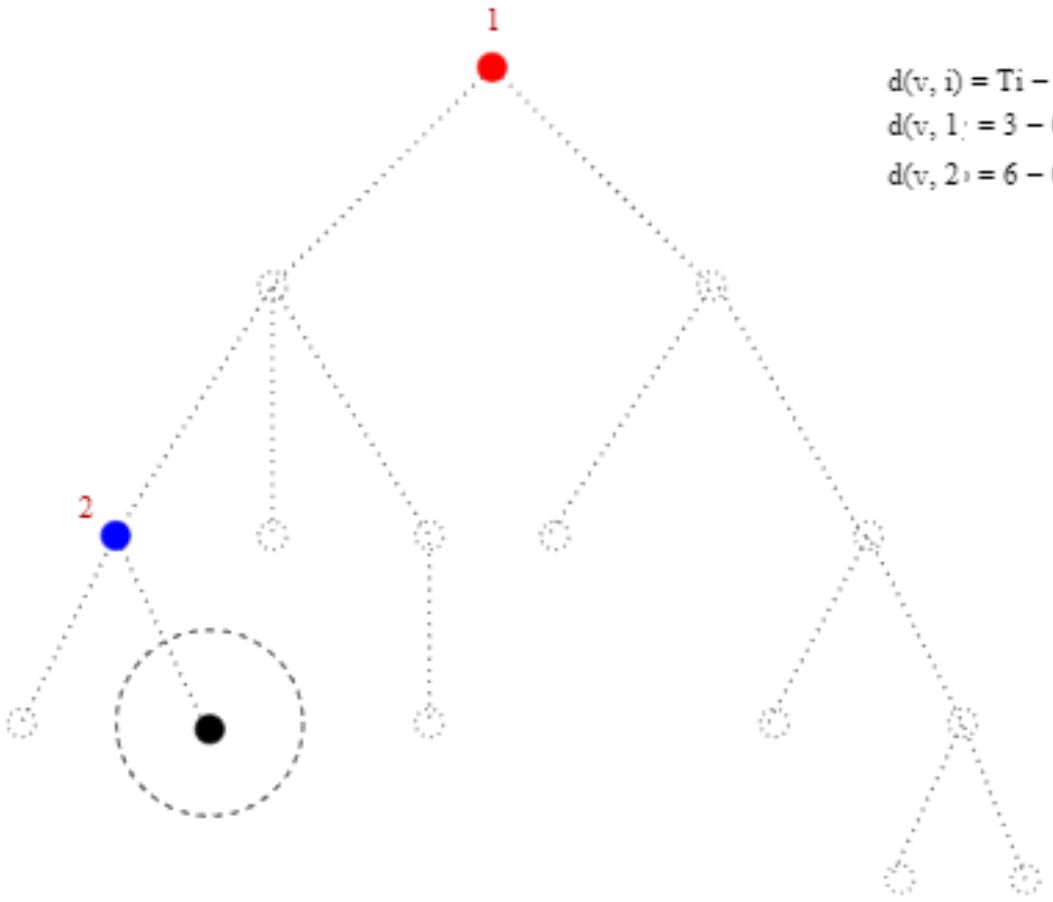
Application 1 : Calcul des plus courts chemins

2. APSP : Calcul des plus courts chemins :



Application 1 : Calcul des plus courts chemins

2. APSP : Calcul des plus courts chemins :



$d(v, i) = Ti - t1 - 5(i-1)$
$d(v, 1) = 3 - 0 - 5(1-1) = 3$
$d(v, 2) = 6 - 0 - 5(2-1) = 1$

Application 1 : Calcul des plus courts chemins

Théorème

Soit G un graphe de taille n avec un sommet distingué. Il existe un algorithme distribué synchrone calculant les plus cours chemins en $O(n)$ rondes et ayant une complexité en bits égale à $O(n)$.

Application 2 : Calcul du diamètre

Principe :

1. chaque sommet calcule le maximum des distances (des sommets) de ses sous arbres au *Leader*, et l'envoie à son père;
2. Le *Leader* "centralise" le calcul;
3. Le *Leader* envoie le maximum à tous les sommets de l'arbre.

Application 2 : Calcul du diamètre

Théorème.

Soit G un graphe de taille n avec un sommet distingué. Il existe un algorithme distribué synchrone permettant de calculer le diamètre D de G en $O(n)$ rondes avec une complexité en bits égale à $O(n)$. Chaque sommet connaît la valeur de D à la fin de l'algorithme.

Plan

Conclusion

La même énumération peut être utilisée pour :

- calculer la maille du graphe,
- déterminer les isthmes,
- déterminer les points d'articulation.

Plan

Bibliographie

1. Y. Métivier, J. M. Robson and A. Zemmari. *A distributed enumeration algorithm and applications to all pairs shortest paths, diameter...* Inf. Comput. 247: 141-151 (2016)
2. P. S. Almeida, C. Baquero, and A. Cunha. *Fast distributed computation of distances in networks.* CoRR, abs/1111.6087, 2011.
3. D. Peleg, L. Roditty, and E. Tal. *Distributed algorithms for network diameter and girth.* In ICALP (2), pages 660 - 672, 2012.
4. S. Holzer and R. Wattenhofer. *Optimal distributed all pairs shortest paths and applications.* In PODC, pages 355 - 364, 2012.
5. S. Frischknecht, S. Holzer and R. Wattenhofer. *Networks cannot compute their diameter in sublinear time.* In SODA 2012, pages 1150-1162, 2012.

MERCI