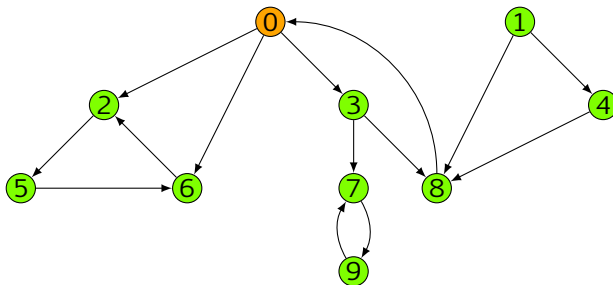


Exemple de parcours d'un graphe

Jean-Pierre Becirspahic
Lycée Marcelin Berthelot

Parcours d'un graphe

à partir du sommet 0



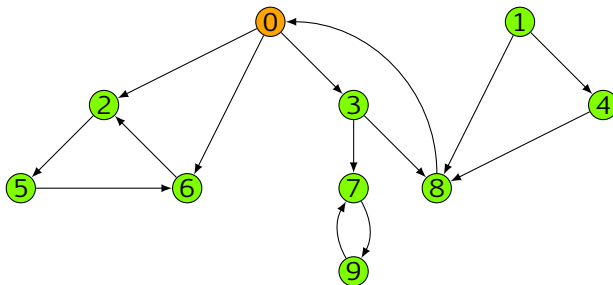
● : non vu ● : vu mais non traité ● : traité

aTraiter = [0]

dejaVu = {0}

Parcours en largeur

FIFO (first in, first out) : le premier entré dans la liste aTraiter est le premier sorti



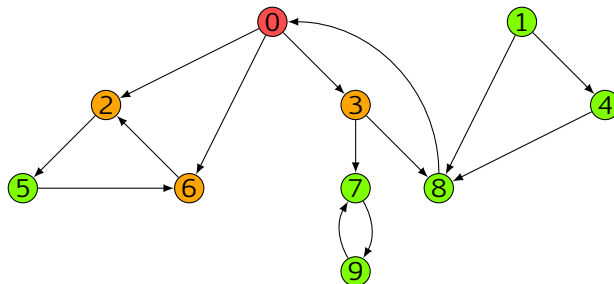
● : non vu ● : vu mais non traité ● : traité

aTraiter = [0]

dejaVu = {0}

Parcours en largeur

FIFO (first in, first out)



● : non vu ● : vu mais non traité ● : traité

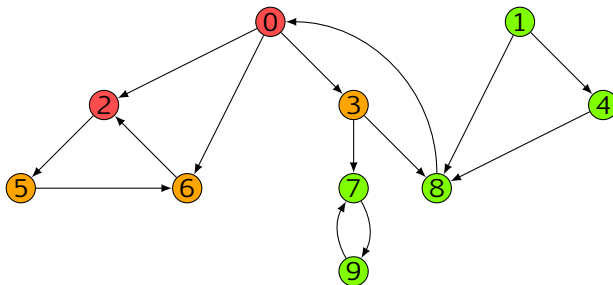
aTraiter = [2, 3, 6]

dejaVu = {0, 2, 3, 6}

0 est traité, ses voisins 2, 3 et 6 sont marqués comme vus et entrent dans la liste à traiter.

Parcours en largeur

FIFO (first in, first out)



● : non vu ● : vu mais non traité ● : traité

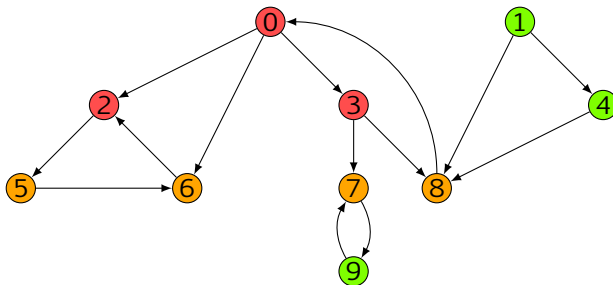
aTraiter = [3, 6, 5]

dejaVu = {0, 2, 3, 5, 6}

2 est traité, son voisin 5 est marqué comme vu et entre dans la liste à traiter.

Parcours en largeur

FIFO (first in, first out)



● : non vu ● : vu mais non traité ● : traité

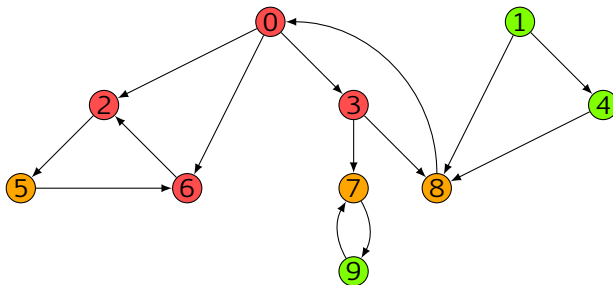
aTraiter = [6, 5, 7, 8]

dejaVu = {0, 2, 3, 5, 6, 7, 8}

3 est traité, ses voisins 7 et 8 sont marqués comme vus et entrent dans la liste à traiter.

Parcours en largeur

FIFO (first in, first out)



● : non vu ● : vu mais non traité ● : traité

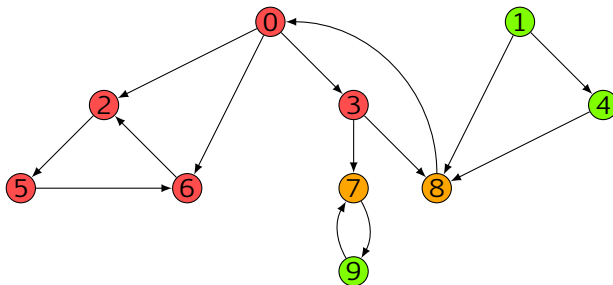
aTraiter = [5, 7, 8]

dejaVu = {0, 2, 3, 5, 6, 7, 8}

6 est traité, son voisin 2 a déjà été vu donc ne rentre pas de nouveau dans la liste à traiter.

Parcours en largeur

FIFO (first in, first out)



● : non vu ● : vu mais non traité ● : traité

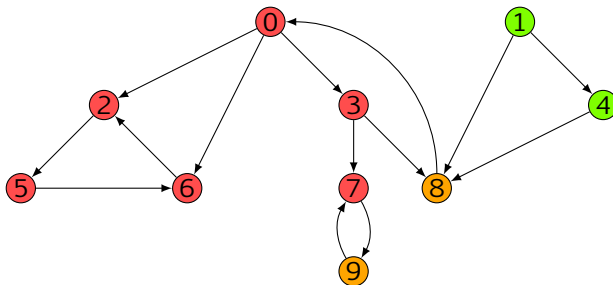
aTraiter = [7, 8]

dejaVu = {0, 2, 3, 5, 6, 7, 8}

5 est traité, son voisin 6 a déjà été vu donc ne rentre pas de nouveau dans la liste à traiter.

Parcours en largeur

FIFO (first in, first out)



● : non vu ● : vu mais non traité ● : traité

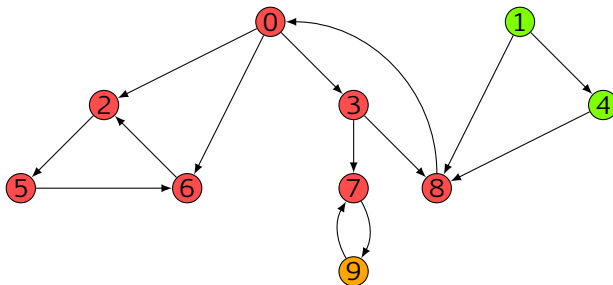
aTraiter = [8, 9]

dejaVu = {0, 2, 3, 5, 6, 7, 8, 9}

7 est traité, son voisin 9 est marqué comme vu et entre dans la liste à traiter.

Parcours en largeur

FIFO (first in, first out)



● : non vu ● : vu mais non traité ● : traité

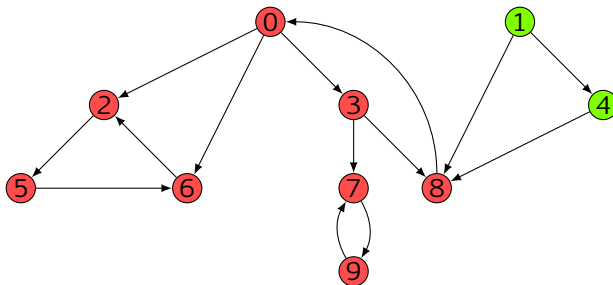
aTraiter = [9]

dejaVu = {0, 2, 3, 5, 6, 7, 8, 9}

8 est traité, son voisin 0 a déjà été vu donc ne rentre pas de nouveau dans la liste à traiter.

Parcours en largeur

FIFO (first in, first out)



● : non vu ● : vu mais non traité ● : traité

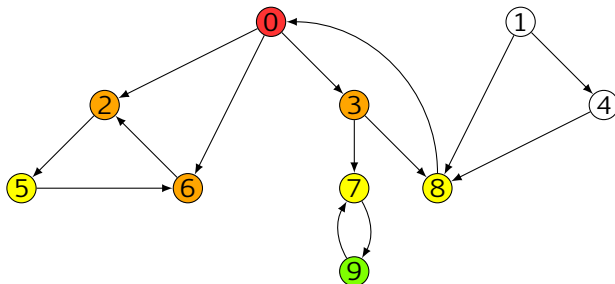
aTraiter = []

dejaVu = {0, 2, 3, 5, 6, 7, 8, 9}

9 est traité, son voisin 7 a déjà été vu donc ne rentre pas de nouveau dans la liste à traiter. Cette liste est désormais vide, l'algorithme est terminé.

Parcours en largeur

Intérêt du parcours en largeur



● : dist = 0 ● : dist = 1 ● : dist = 2 ● : dist = 3

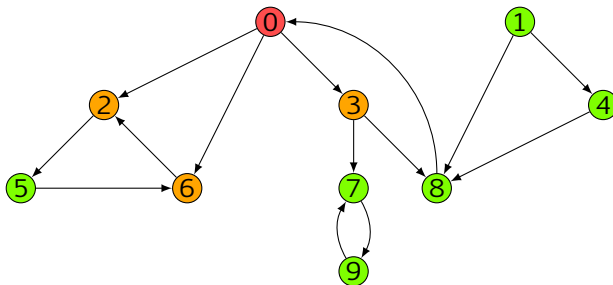
Ordre du traitement lors d'un parcours en largeur :

(0) - (2 - 3 - 6) - (5 - 7 - 8) - (9)

Les sommets sont traités par distance à l'origine du parcours croissante.

Pseudo parcours en profondeur

LIFO(last in, first out) : le dernier entré dans la liste aTraiter est le premier sorti



● : non vu ● : vu mais non traité ● : traité

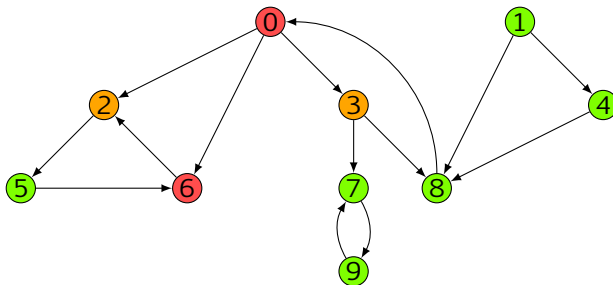
aTraiter = [2, 3, 6]

dejaVu = {0, 2, 3, 6}

0 est traité, ses voisins 2, 3 et 6 sont marqués comme vus et entrent dans la liste à traiter.

Pseudo parcours en profondeur

LIFO (last in, first out)



● : non vu ● : vu mais non traité ● : traité

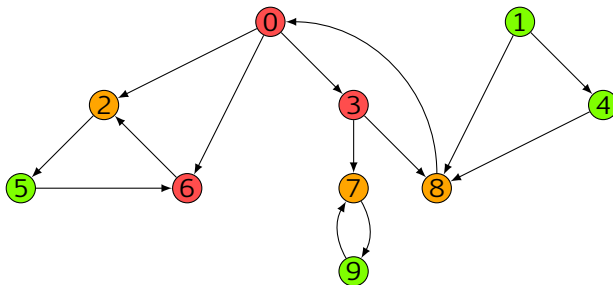
aTraiter = [2, 3]

dejaVu = {0, 2, 3, 6}

6 est traité, son voisin 2 a déjà été vu donc ne rentre pas de nouveau dans la liste à traiter.

Pseudo parcours en profondeur

LIFO (last in, first out)



● : non vu ● : vu mais non traité ● : traité

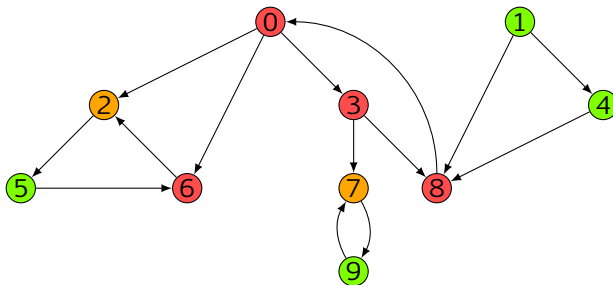
aTraiter = [2, 7, 8]

dejaVu = {0, 2, 3, 6, 7, 8}

3 est traité, ses voisins 7 et 8 n'ont pas encore été vu donc sont marqués comme tels et rentrent dans la liste à traiter.

Pseudo parcours en profondeur

LIFO (last in, first out)



● : non vu ● : vu mais non traité ● : traité

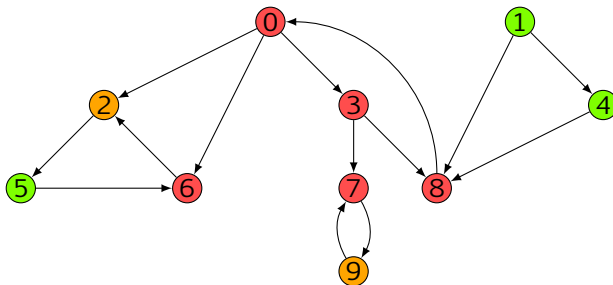
aTraiter = [2, 7]

dejaVu = {0, 2, 3, 6, 7, 8}

8 est traité, son voisin 0 a déjà été vu donc ne rentre pas de nouveau dans la liste à traiter.

Pseudo parcours en profondeur

LIFO (last in, first out)



● : non vu ● : vu mais non traité ● : traité

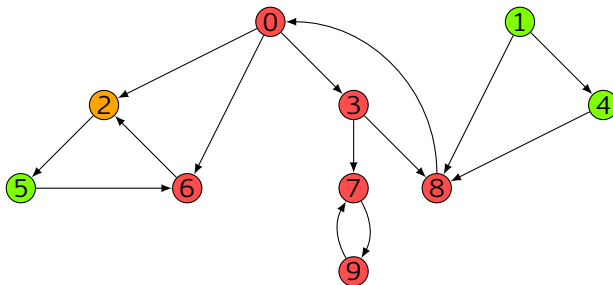
aTraiter = [2, 9]

dejaVu = {0, 2, 3, 6, 7, 8, 9}

7 est traité, son voisin 9 n'a pas encore été vu donc est marqué comme tel et entre dans la liste à traiter.

Pseudo parcours en profondeur

LIFO (last in, first out)



● : non vu ● : vu mais non traité ● : traité

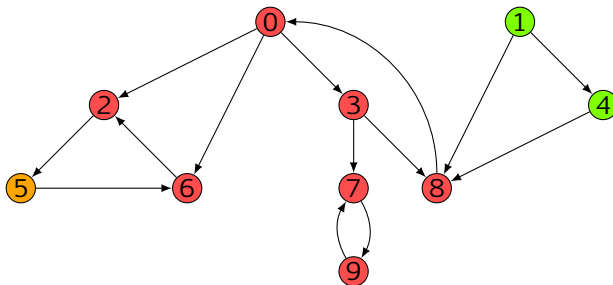
aTraiter = [2]

dejaVu = {0, 2, 3, 6, 7, 8, 9}

9 est traité, son voisin 7 a déjà été vu donc ne rentre pas de nouveau dans la liste à traiter.

Pseudo parcours en profondeur

LIFO (last in, first out)



● : non vu ● : vu mais non traité ● : traité

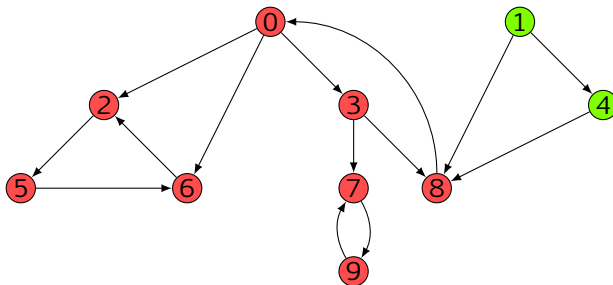
aTraiter = [5]

dejaVu = {0, 2, 3, 5, 6, 7, 8, 9}

2 est traité, son voisin 5 n'a pas encore été vu donc est marqué comme tel et entre dans la liste à traiter.

Pseudo parcours en profondeur

LIFO (last in, first out)



● : non vu ● : vu mais non traité ● : traité

aTraiter = []

dejaVu = {0, 2, 3, 5, 6, 7, 8, 9}

5 est traité, son voisin 6 a déjà été vu donc ne rentre pas de nouveau dans la liste à traiter.

La liste aTraiter est vide, l'algorithme est terminé.