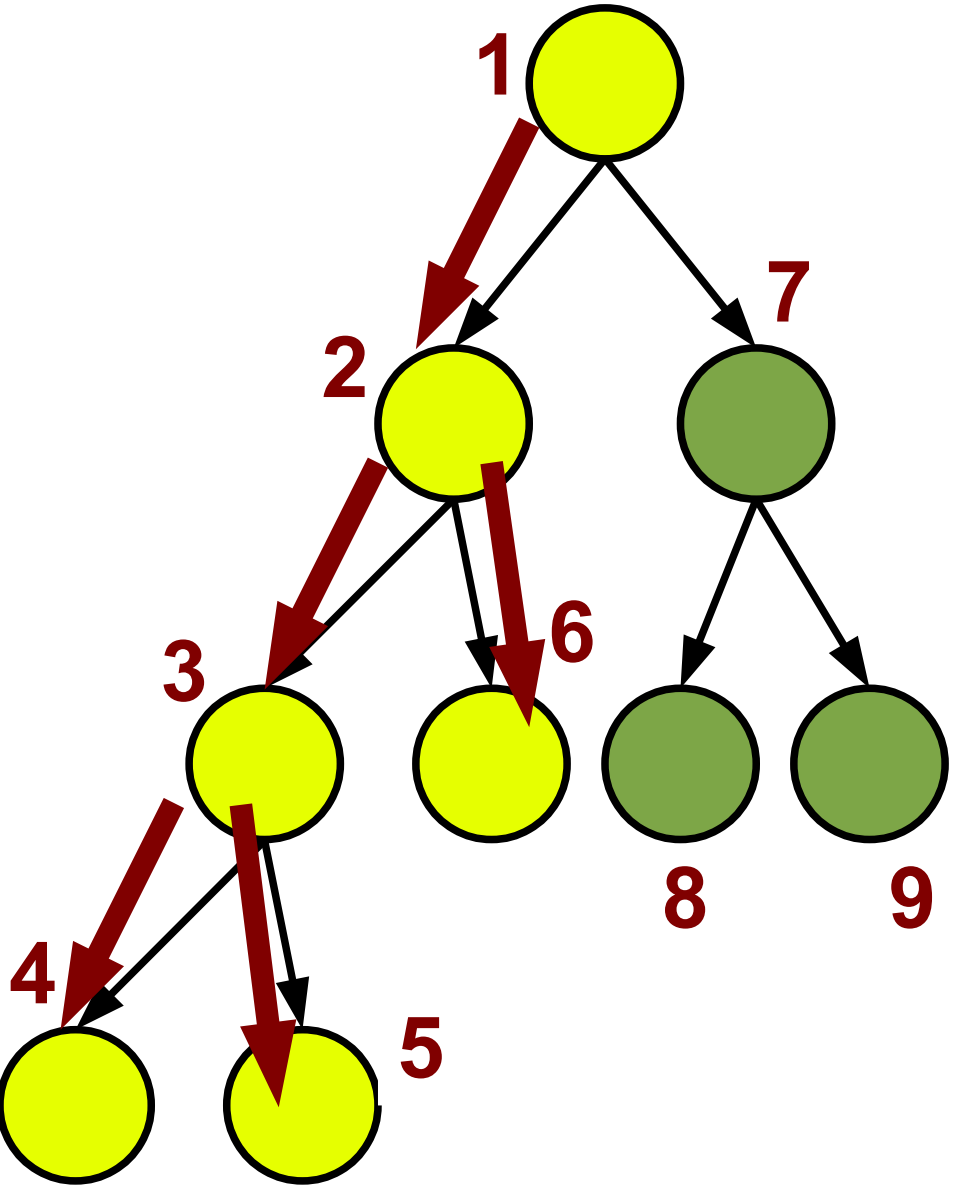
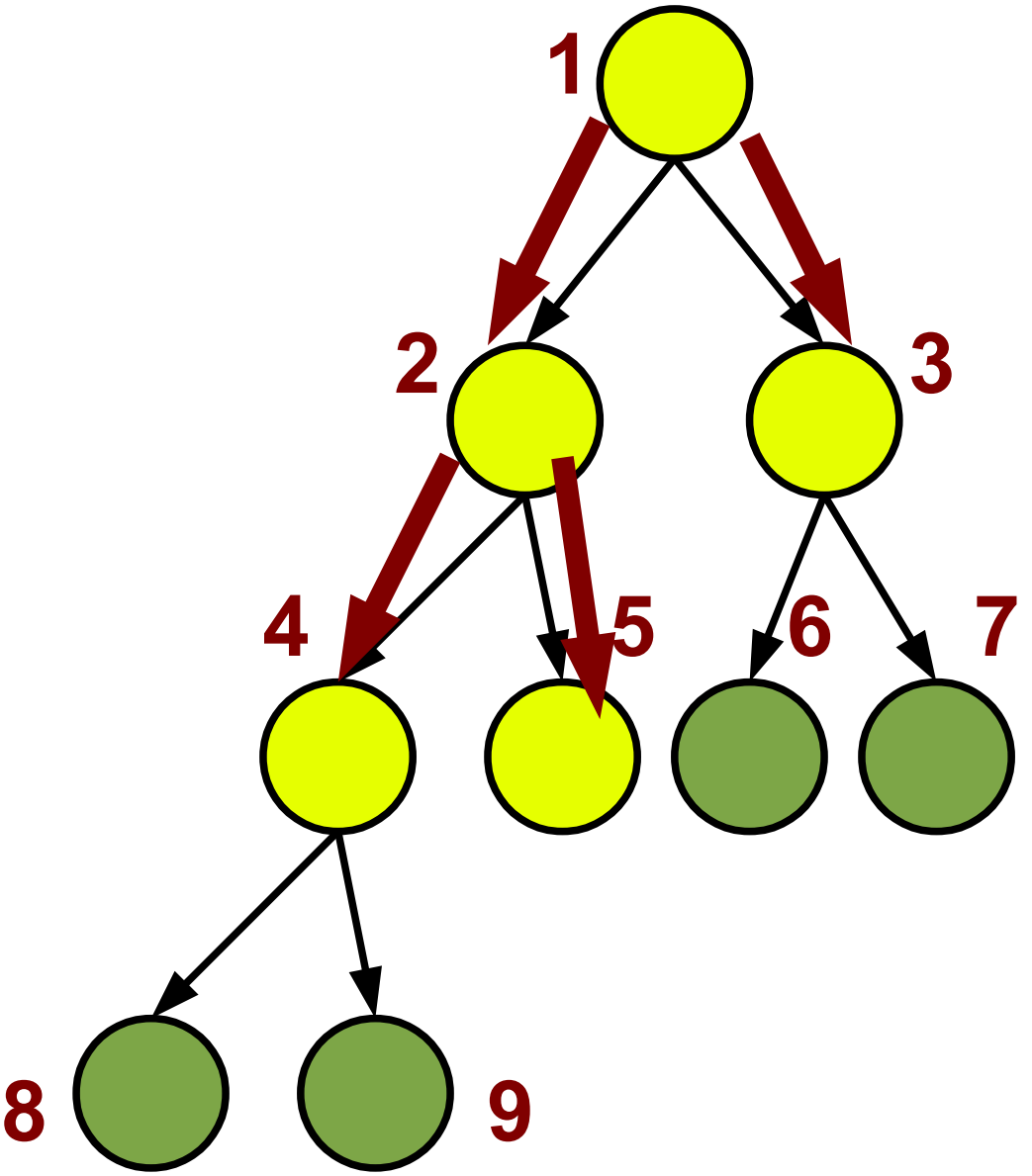
Parcours en largeur

François Schwarzentruber
ENS Cachan – Antenne de Bretagne

Les parcours !

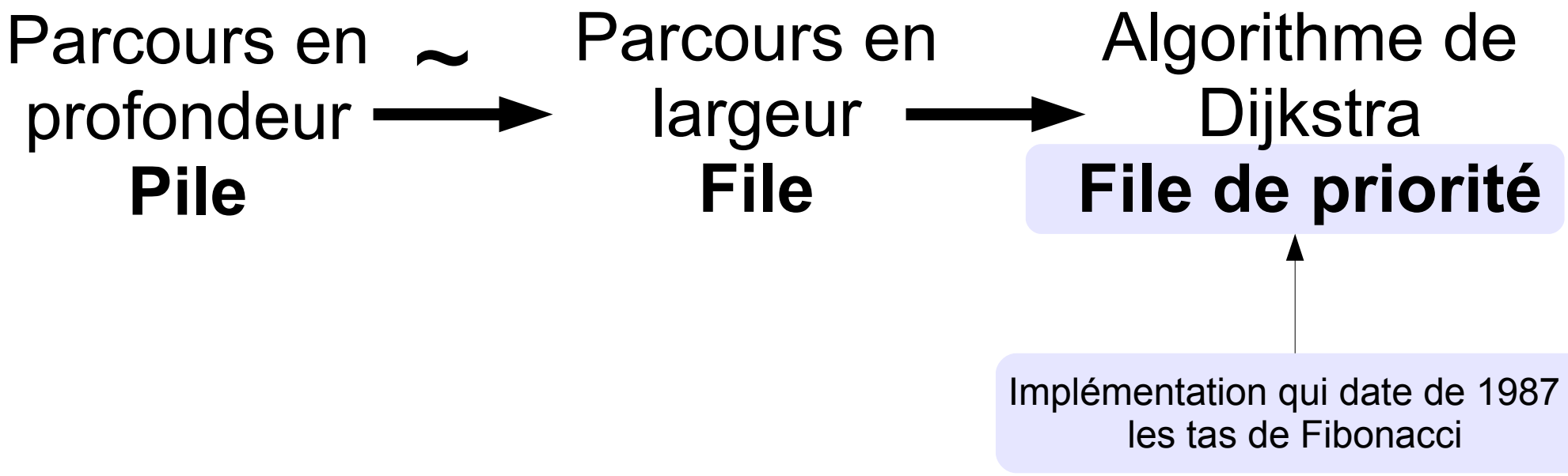




Plan

But : calculer le plus court chemin pour aller de s à t

- Parcours en profondeur est inapproprié
- Transformation ~

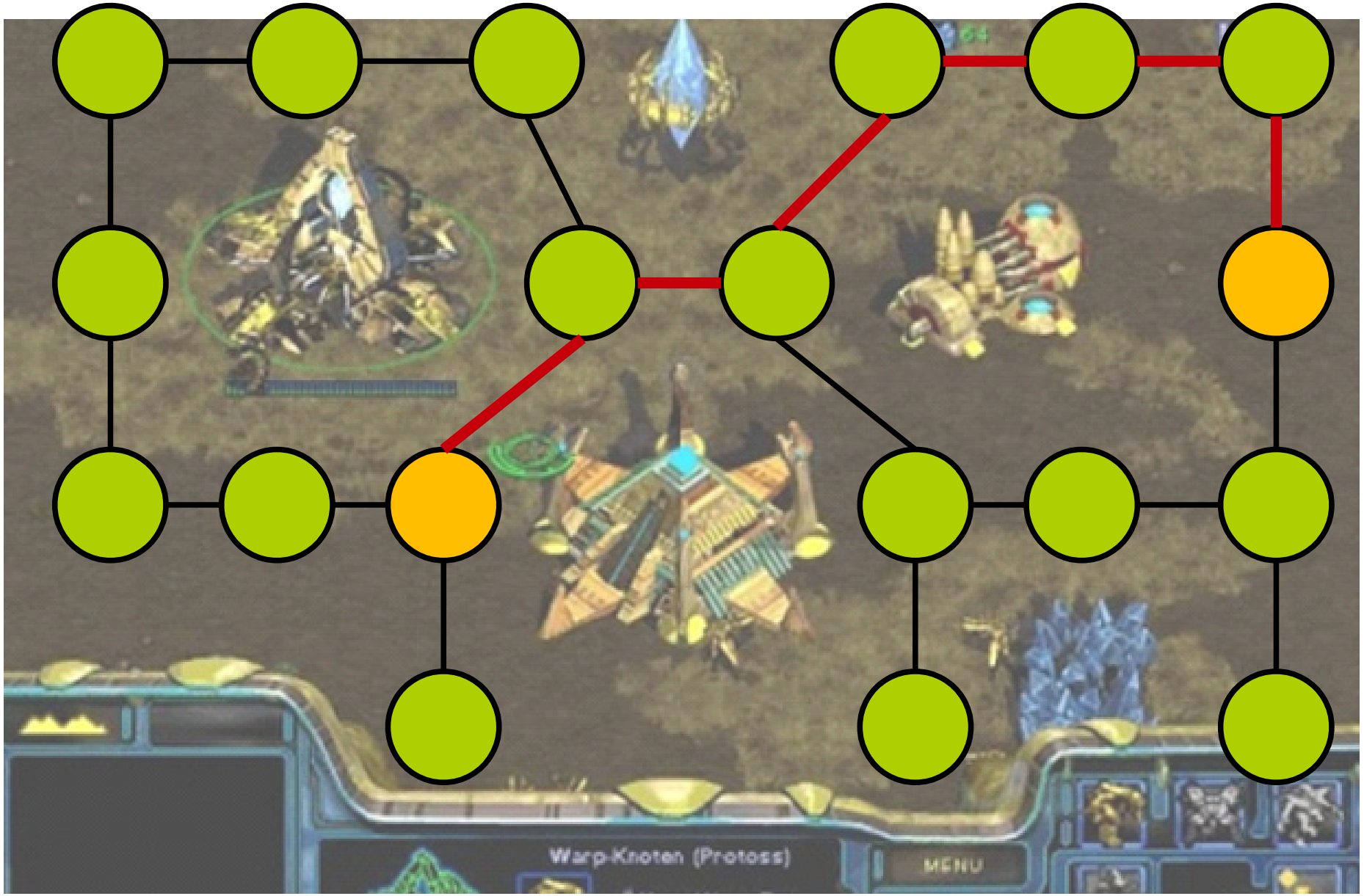


Plus court chemin

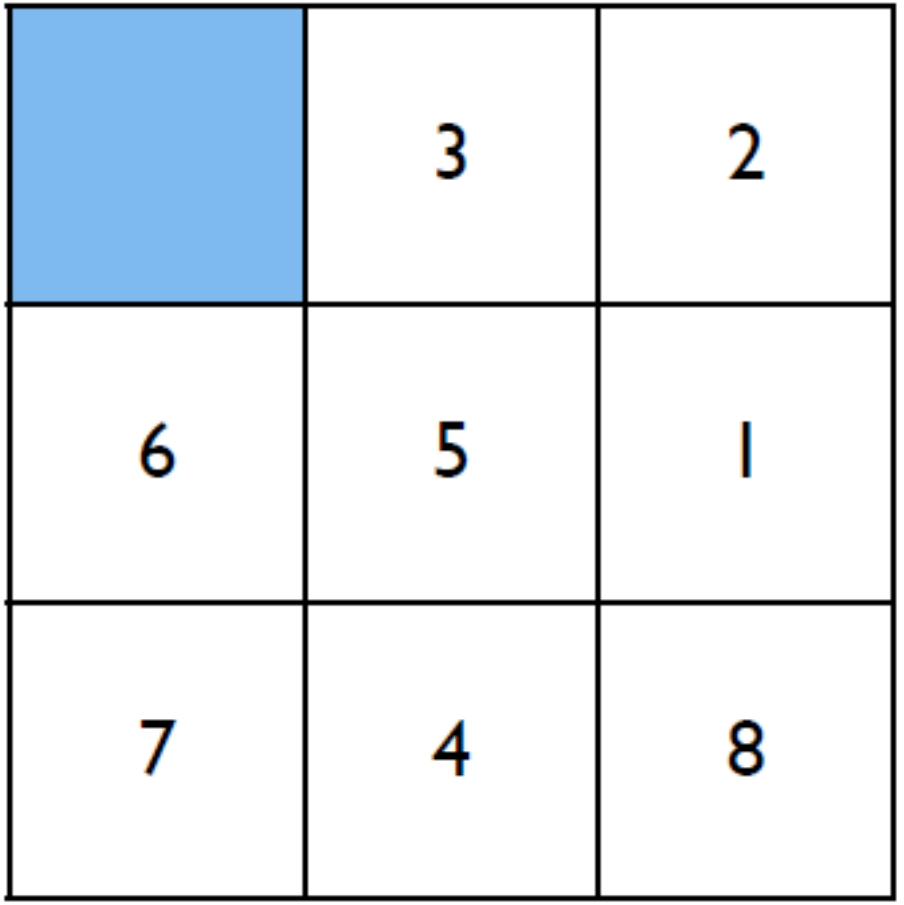
Ceci n'est pas un plus court chemin

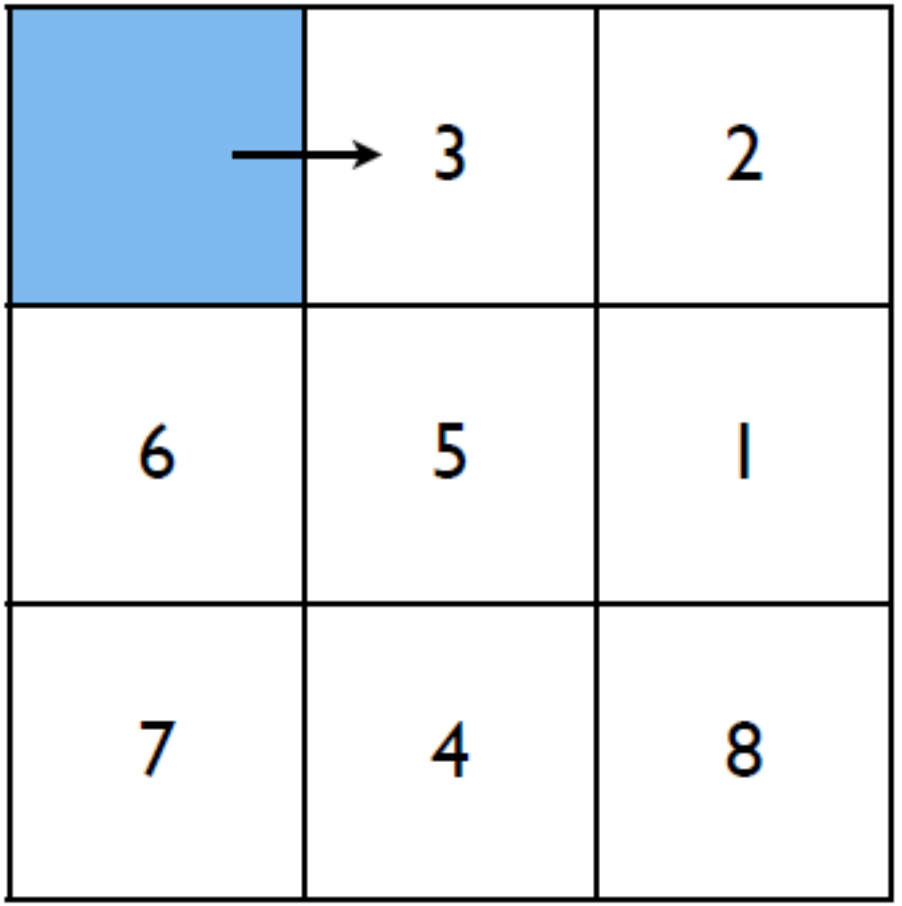
Plus court chemin

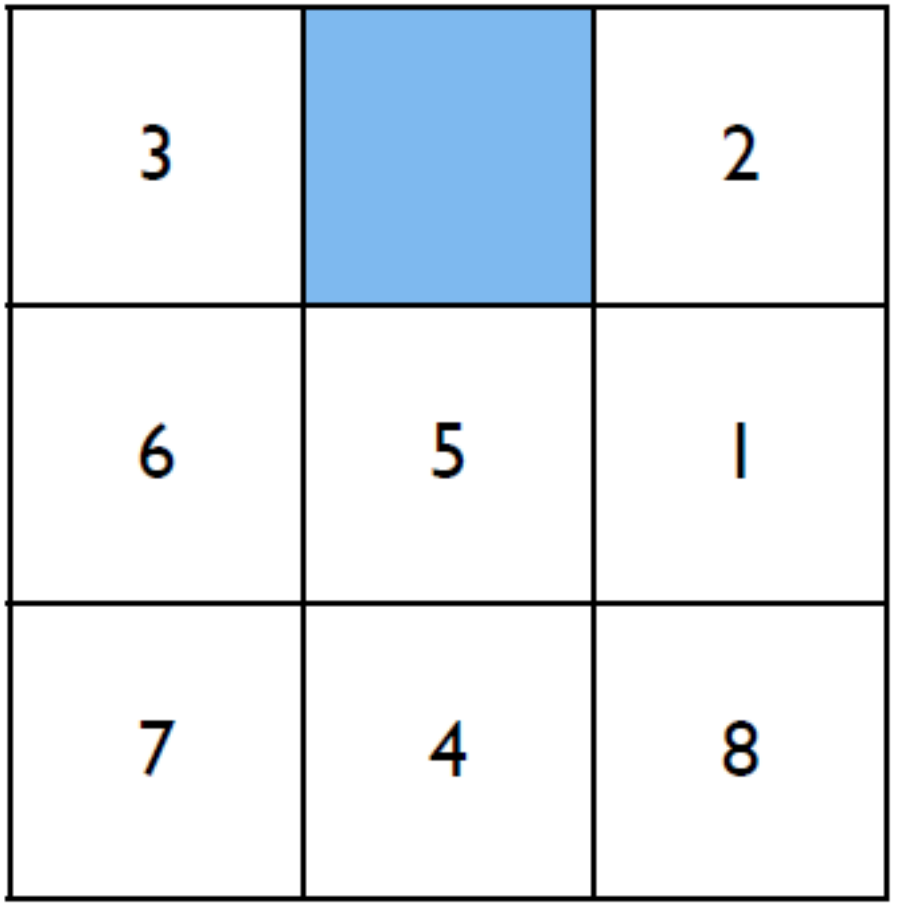
Plus court chemin

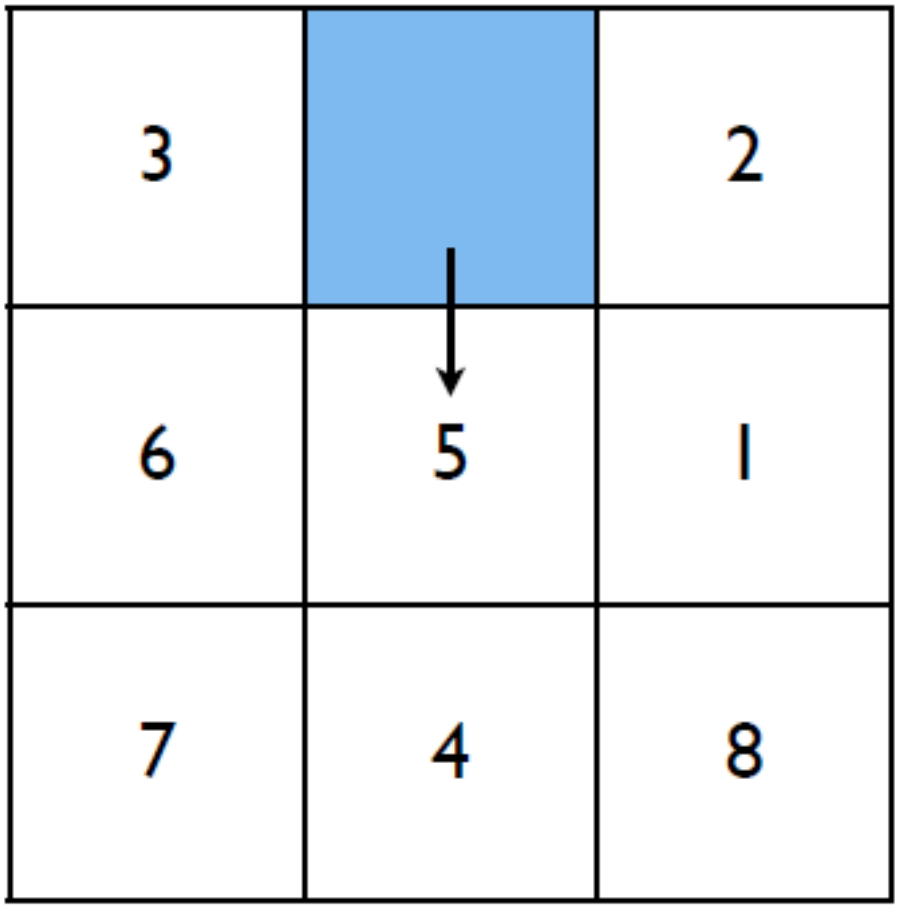


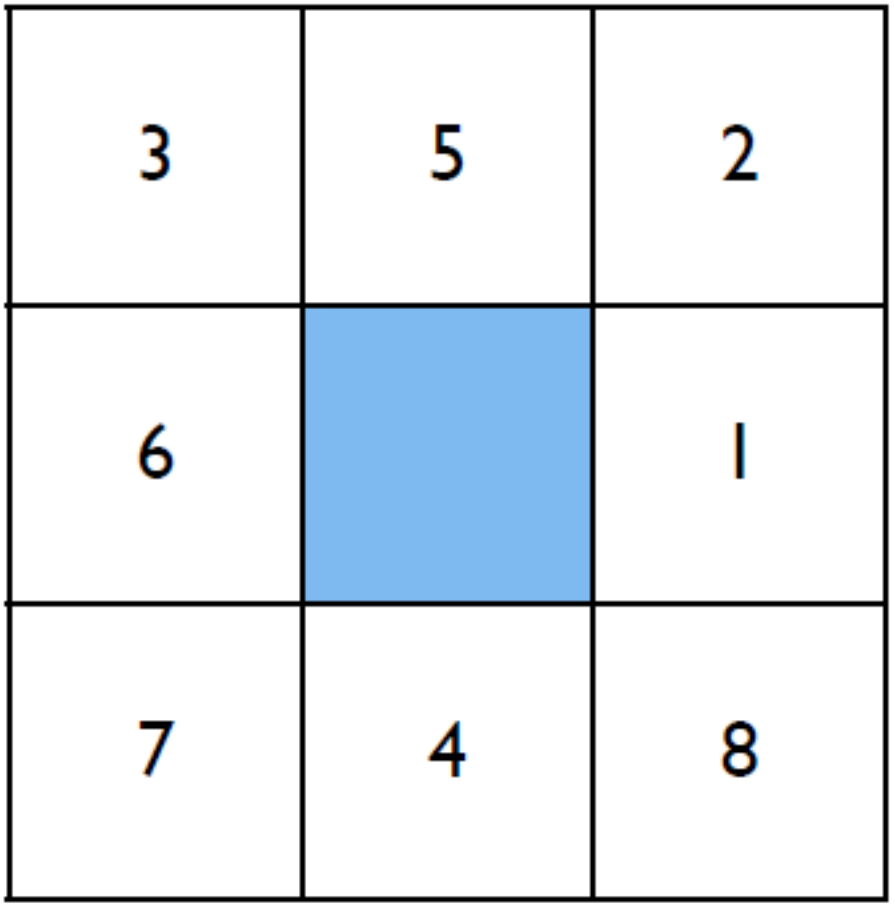
Autre application : jeu du Taquin (par exemple)



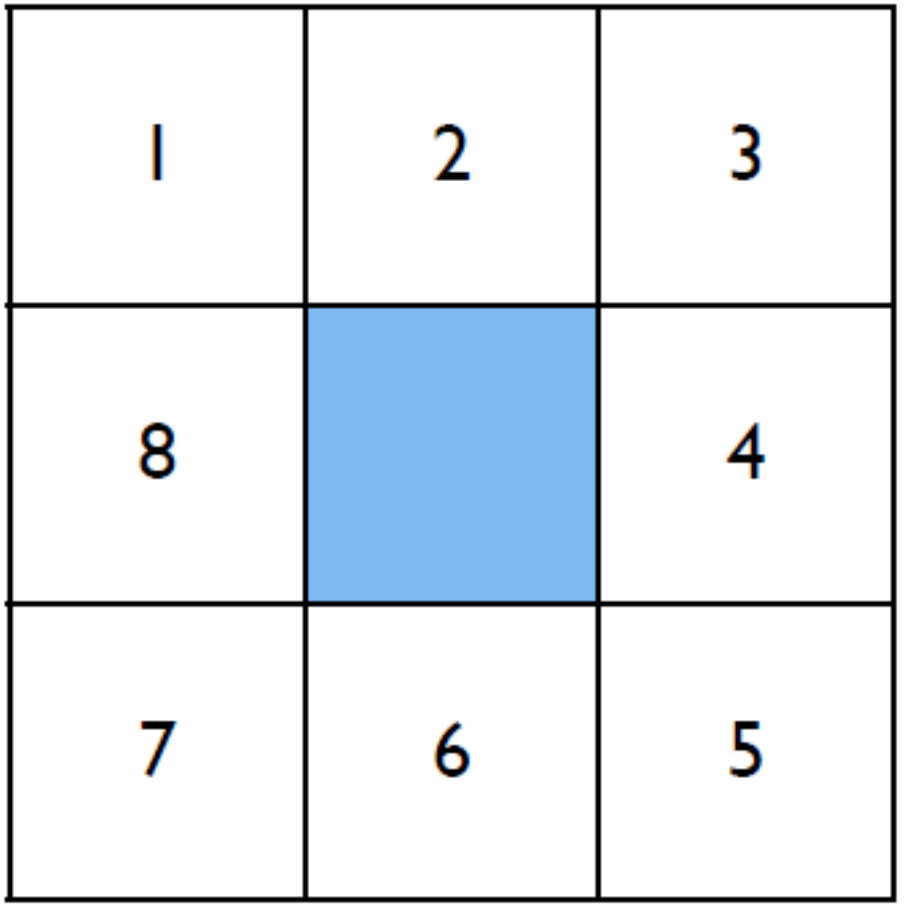

3
2
6
5
1
7
4
8


3
2
6
5
1
7
4
8

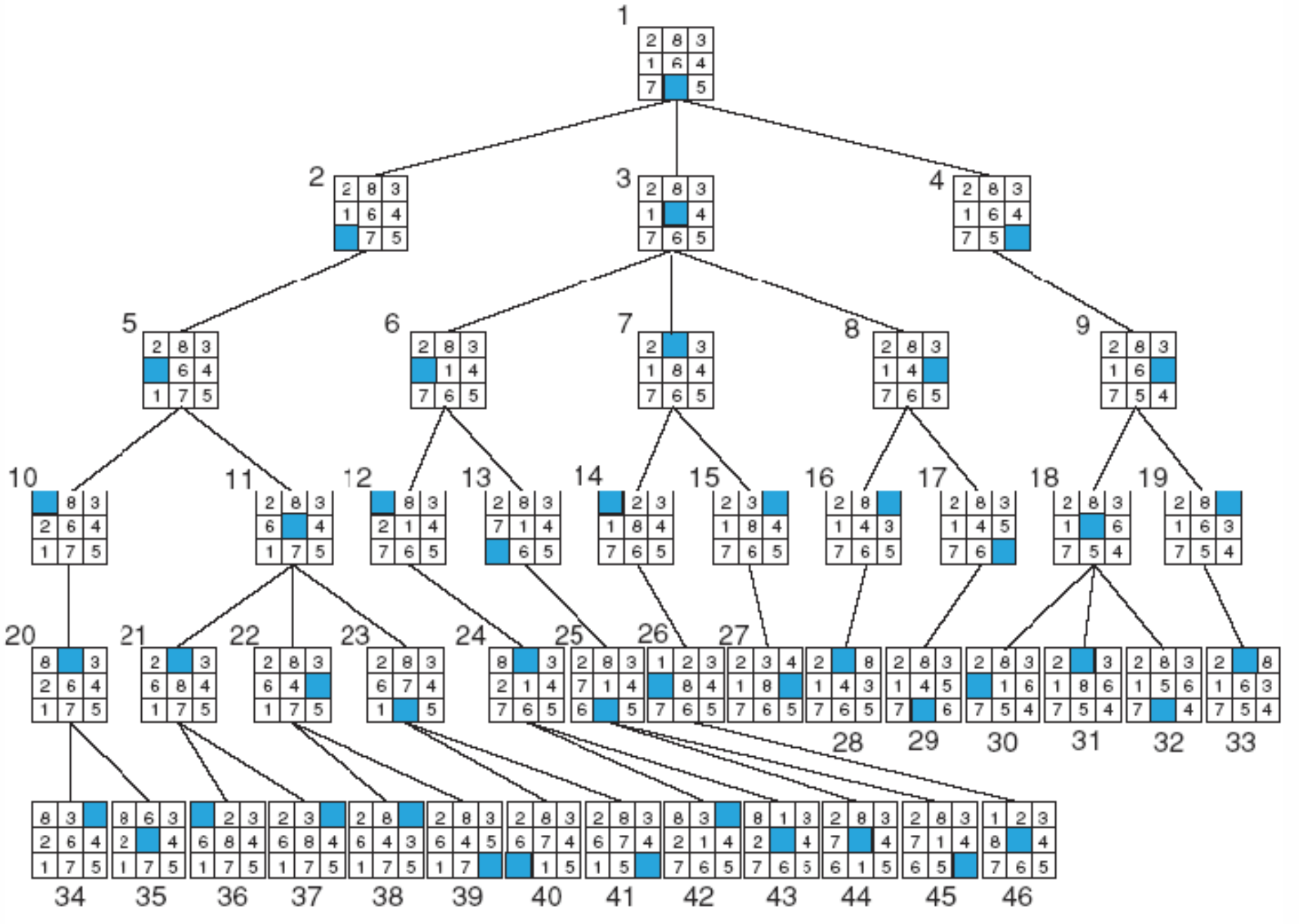

3
2
6
5
1
7
4
8


3
5
2
6
1
7
4
8

But : arriver à cet état en le moins de coups possibles



Parcours en largeur



Parcours en profondeur

Fonction visite(s)

Pour tout s → t
Si (t)
visite(t)

Procedure Parcours en profondeur(G) :
Pour tout s
Si (s)
Visite(s)

Pile d'appel

Fonction visite(s)

previsite(s)
Pour tout s → t
  Si t
    visite(t)

Fonction visite(s)
  P ← pile_creer([s])

Tant que P.nonvide
  s' ← P.depiler
  Si s'

    previsite(s')
    Pour tout s' → t
      P.empiler(t)

L'idée y est...

Fonction visite(s)
    P ← pile_creer([s])

Tant que P.nonvide
    s' ← P.depiler
    Si 
        
        previsite(s')
        Pour tout s' → t
            P.empiler(t)

Fonction visite(s)
    F ← file_creer([s])

Tant que F.nonvide
    s' ← F.defiler
    Si 
        
        previsite(s')
        Pour tout s' → t
            F.enfiler(t)

Vers un algorithme `plus joli' du parcours en largeur

Fonction visite(s)
F ← file_creer([s])

Tant que F.nonvide
s' ← F.defiler
Si (s')

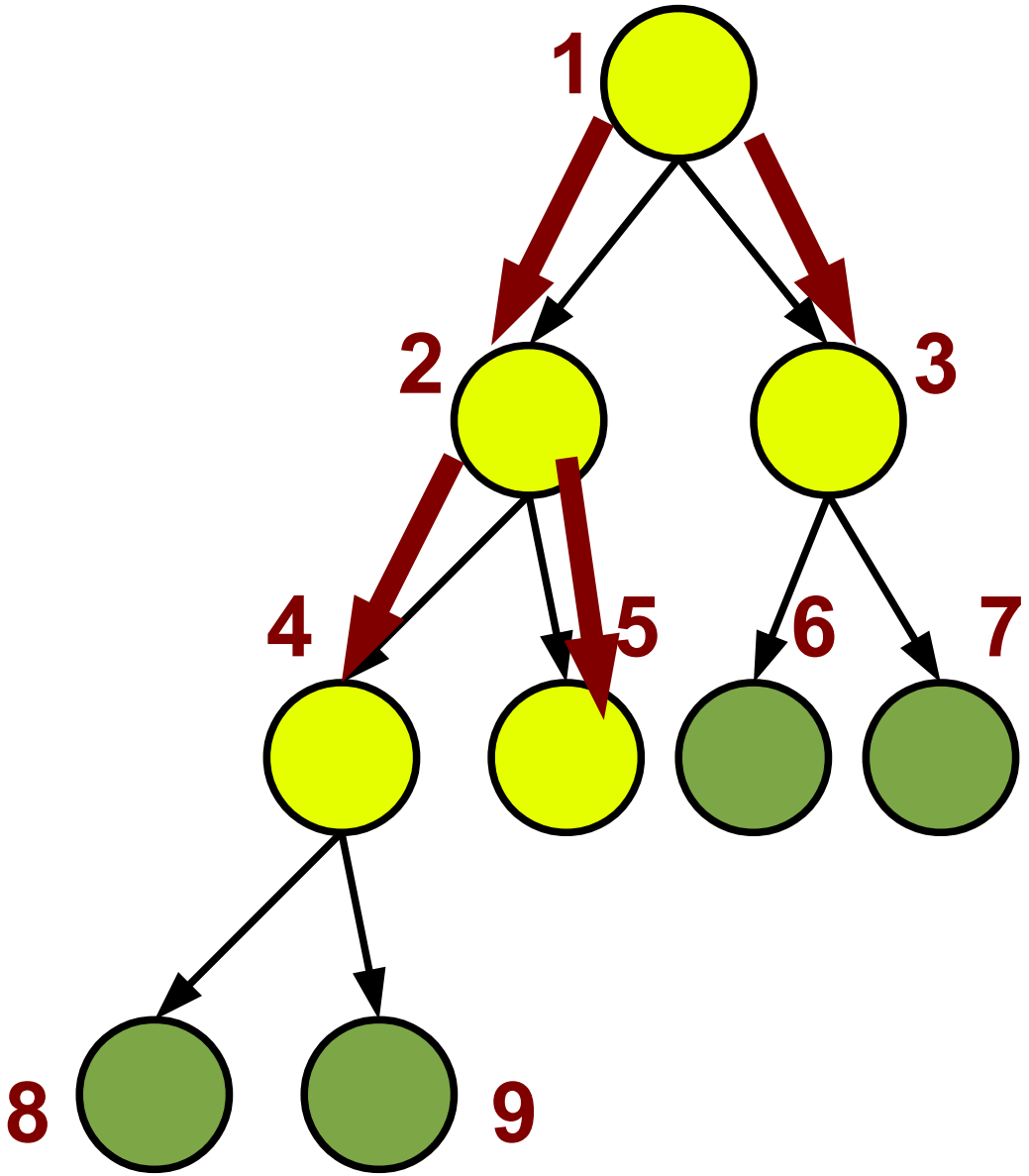
previsite(s')
Pour tout s' → t
F.enfiler(t)

Fonction visite(s)
F ← file_creer([s])

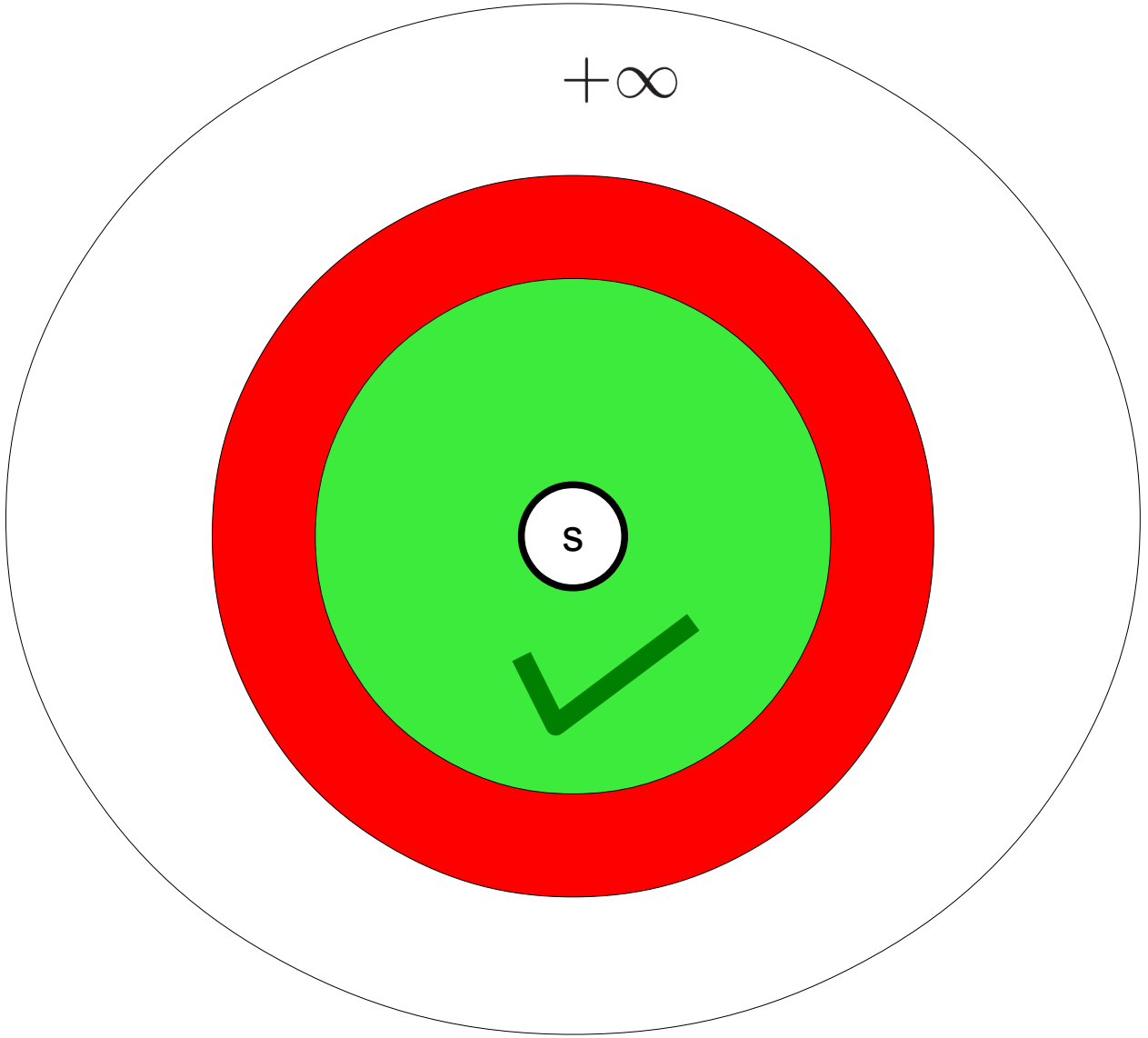
Tant que F.nonvide
s' ← F.defiler
Pour tout s' → t
Si (t)

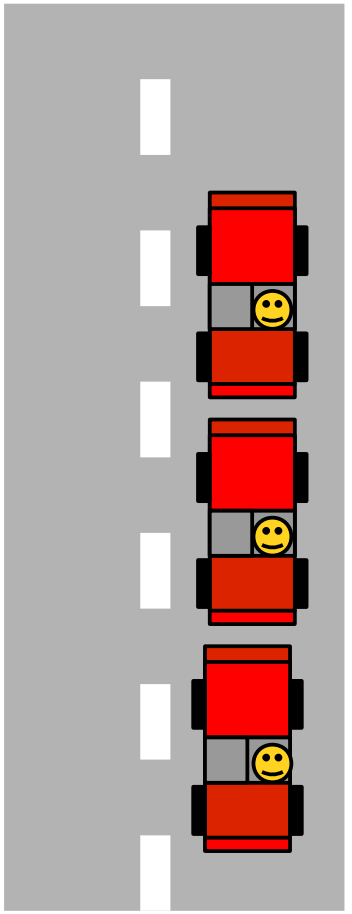
F.enfiler(t)

Parcours en largeur

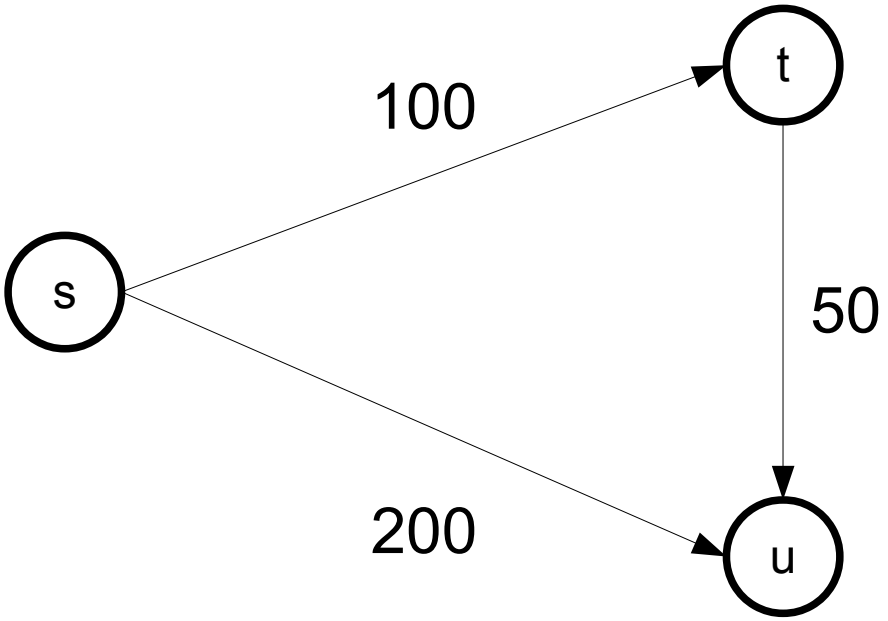


Invariant du parcours en largeur

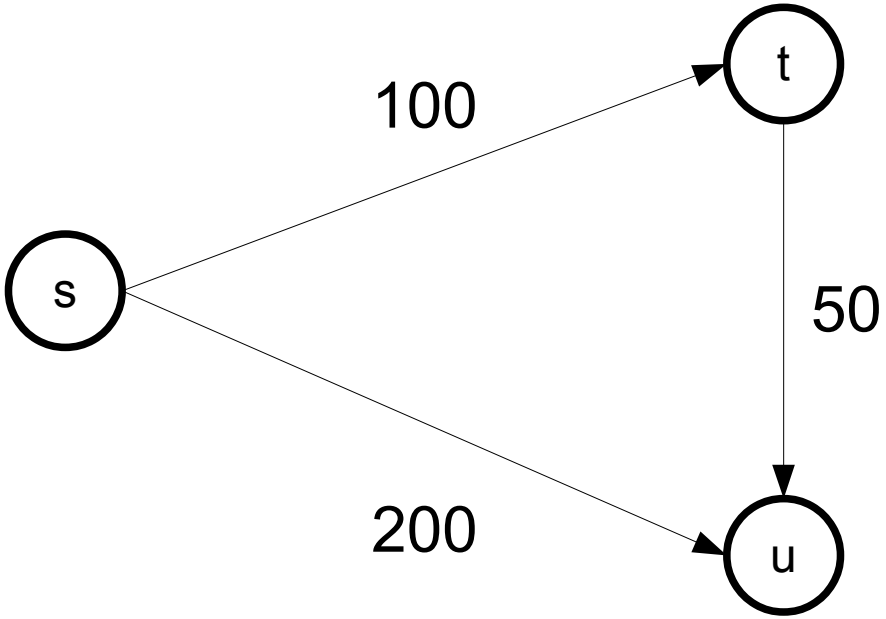


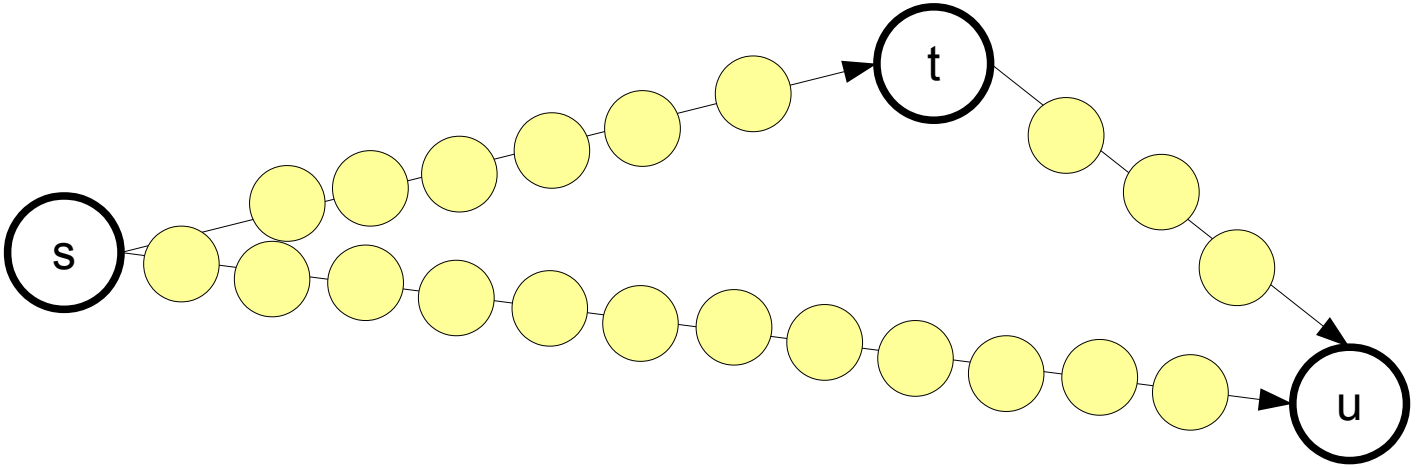


Graphe pondéré positivement

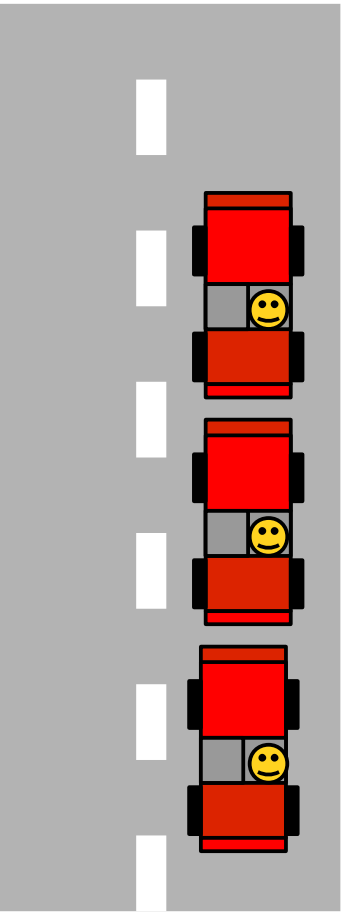


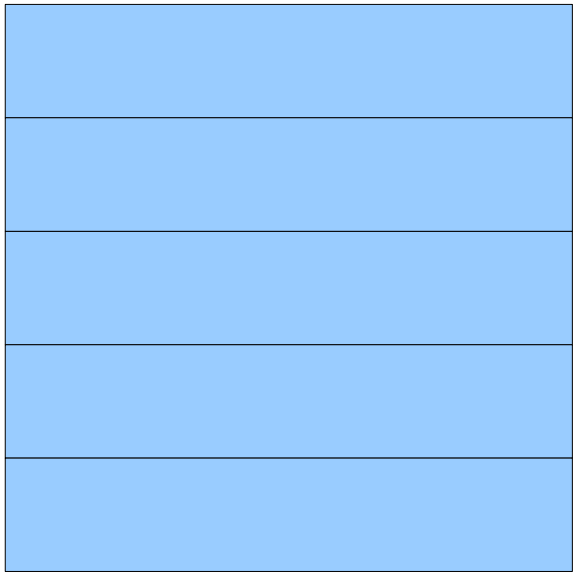
Transformation ?... Non !





File (FIFO : first in, first out)

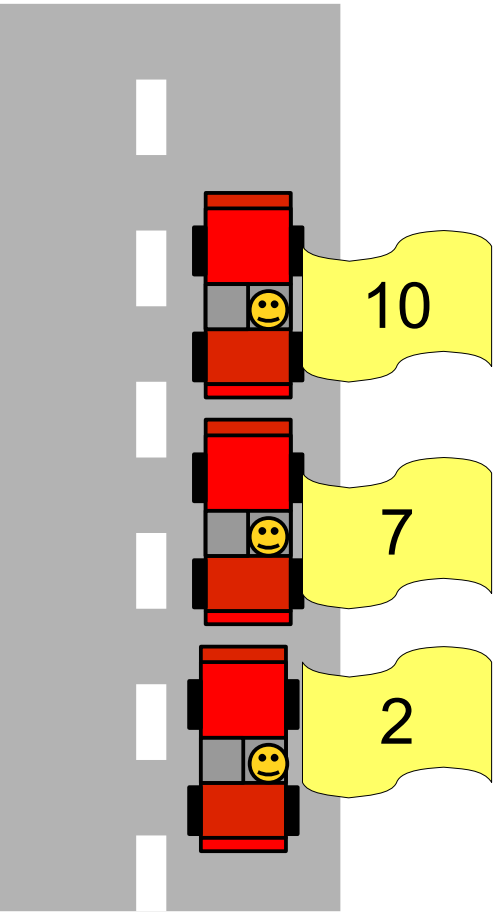


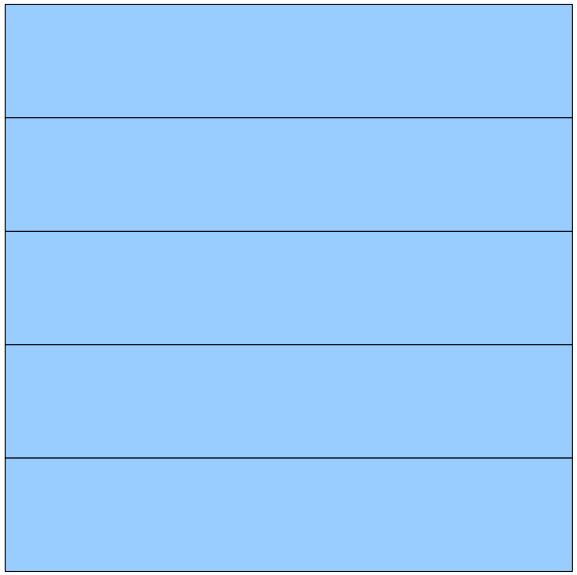


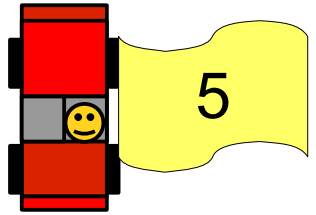
enfiler

défiler

Structure de données abstraite « File de priorité »







enfiler

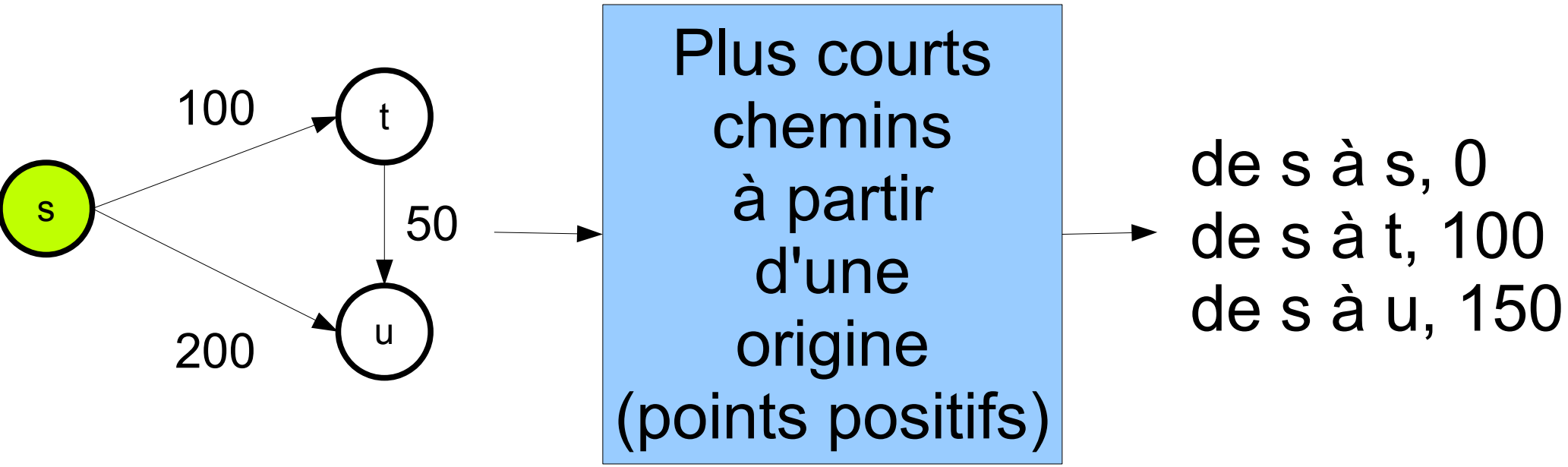
défiler max

min

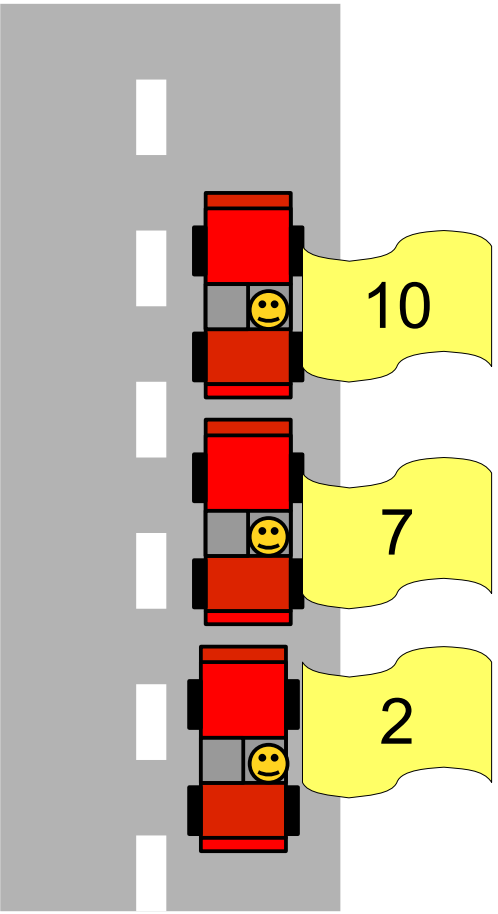
Questions...

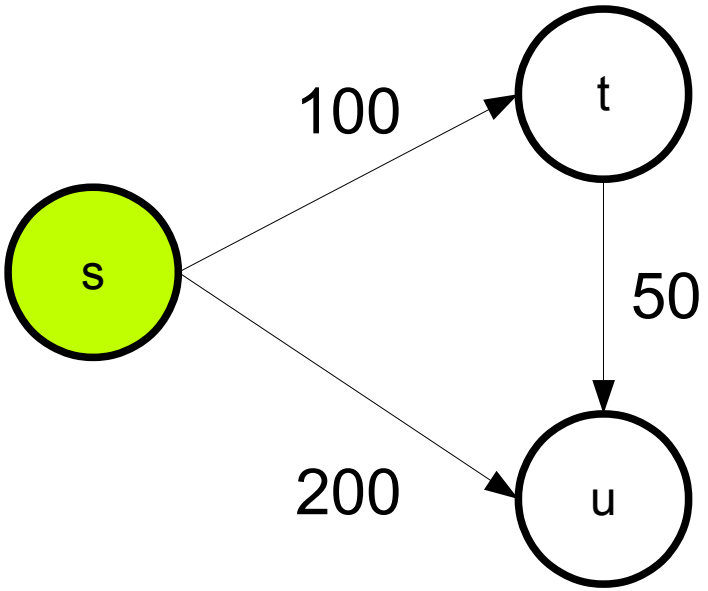
- Et si les points sont négatifs ?
- Et si on cherche le plus long chemin ?

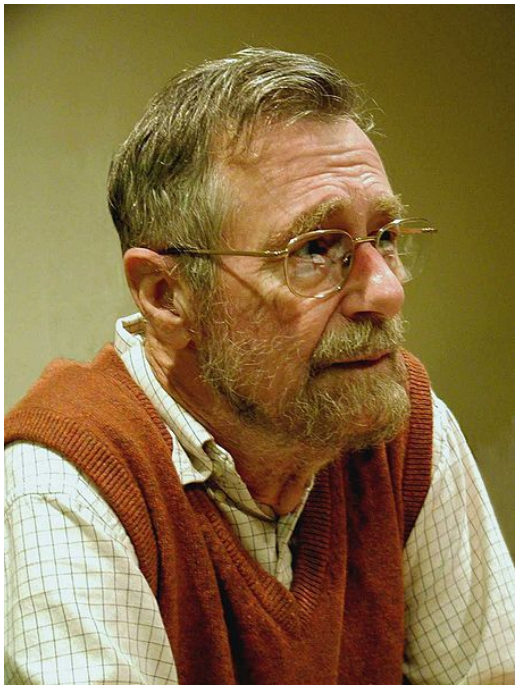
Problème des plus courts chemins à partir d'une origine



Algorithme de Dijkstra







Invariant