Chapitre 3: Algorithme de Recherche Aveugle

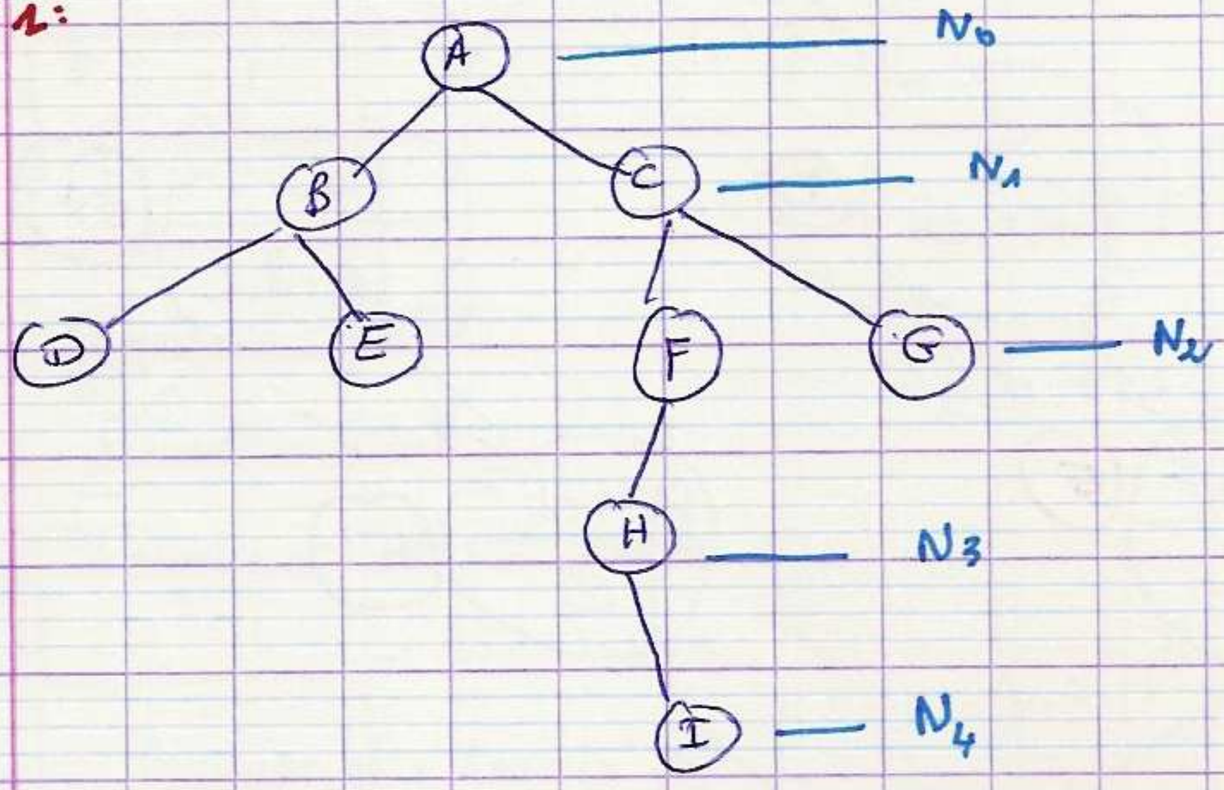
1/ La Méthode de Recherche en largeur d'abord:

Principe:

La recherche en largeur d'abord suit la stratégie d'exploration niveau par niveau (d'une manière horizontale)

L'implémentation se fait à l'aide d'une file d'attente.

exemple 1:



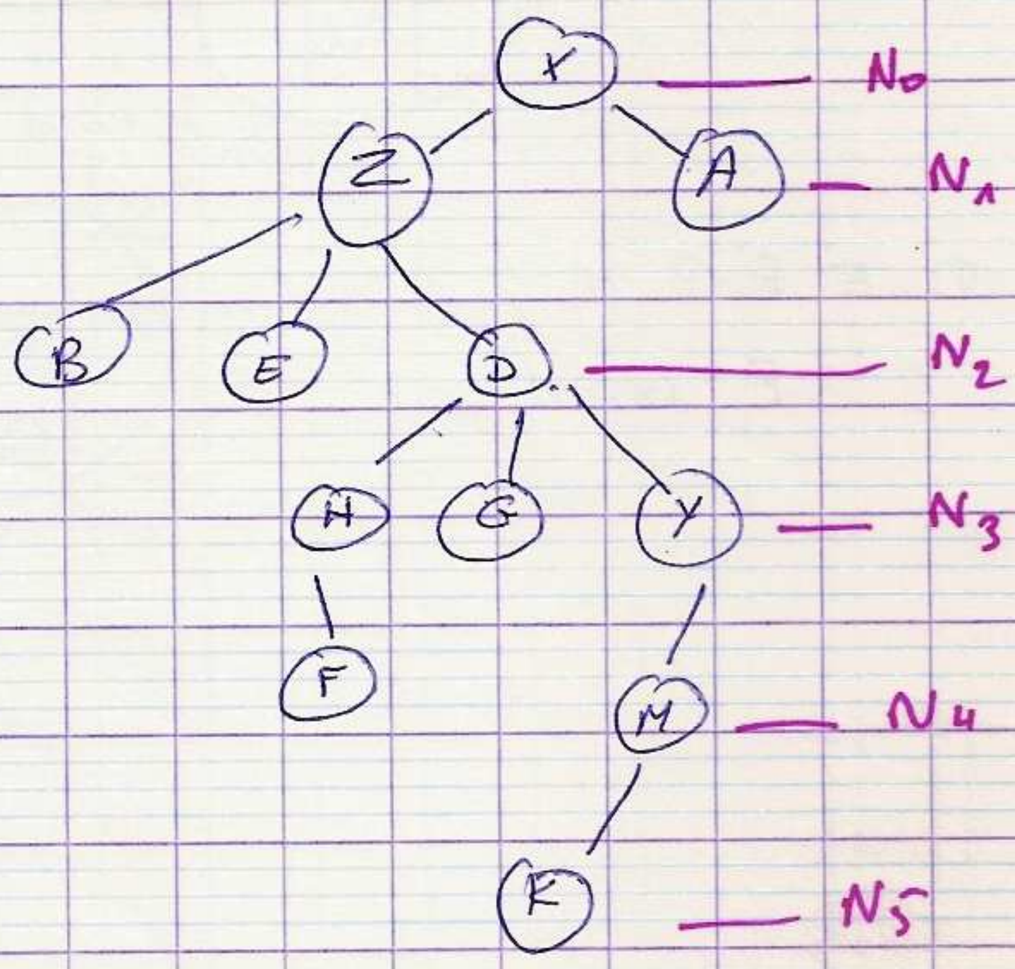
combien de niveau = 4 niveaux

combien de noeuds = 9 noeuds

méthode en Largeur d'abord, ordre de visite des noeuds est comme suit

A_B_C_D_E_F_G_H_I

Exemple 2:



Combien de niveaux ? : 5 niveaux.

Combien de Noeuds ? = 12 noeuds

ordre de visite selon (recherche en largeur) : X_Z_A_B_E_D_H_G_Y

_I_M_K

* But : Y : X_Z_A_B_E_D_H_G_Y

=> 9 noeuds.

Critères d'évaluation de la méthode

* La méthode en largeur d'abord
  - Complete ? oui (si b est fini)
  - Compleauté en temps : $1 + b^1 + b^2 + \dots + b^d = \Theta(b^d)$
  - Compleauté en espace : = $\Theta(b^d)$ tout en gardant chaque noeud en memoire
  - Optimale :
    - pas optimale
    - Oui si chaque étape (niveau) coûte 1 independament du nbre de noeud.

TD (Exercice d'app)

Jeu de taquin 3x3

Etat initial

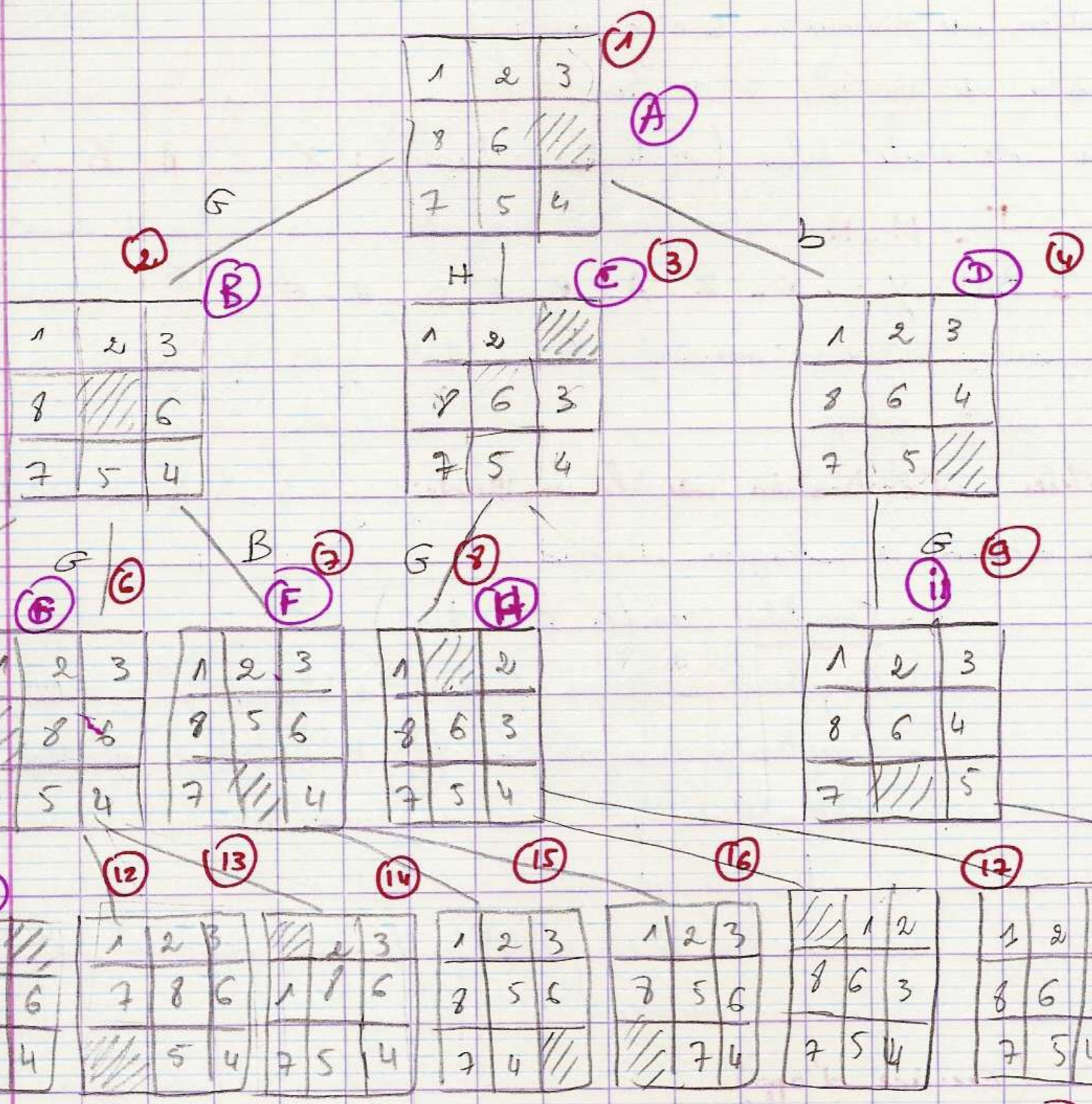
8	6	
7	5	4

Etat Final.

8		4
7	6	5

App la recherche Aveugle en largeur d'abord.

Correction:

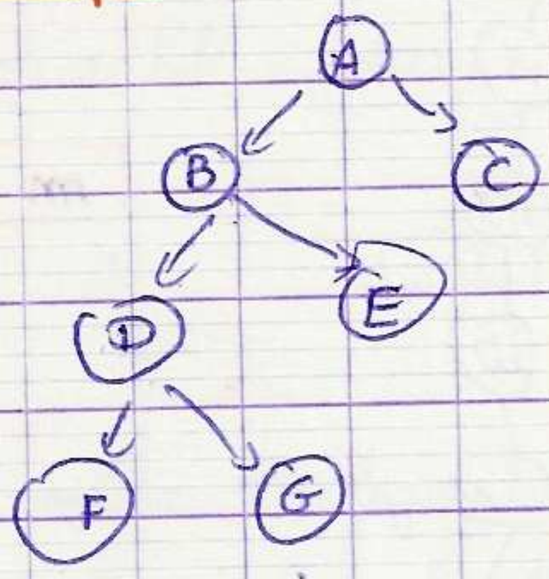


2. La Méthode de recherche en profondeur d'abord:

Stratégie: etudier le nœud le plus profond

Implémentation: insertion des successeurs en tête d'une liste (pile)

Exemple:



Ordre de visite (en profondeur d'abord)

A - B - D - F - G - E - C

* Etat initial B → Etat Final G.

  chemin optimal B - D - G

  chemin en largeur d'abord B - D - E - F - G.

  " " profondeur " B - D - F - G

* Etat initial A → Etat final E

  chemin optimal A - B - E

  chemin en largeur d'abord : A B C D E

  chemin en profondeur d'abord: A - B - D - F - G - E

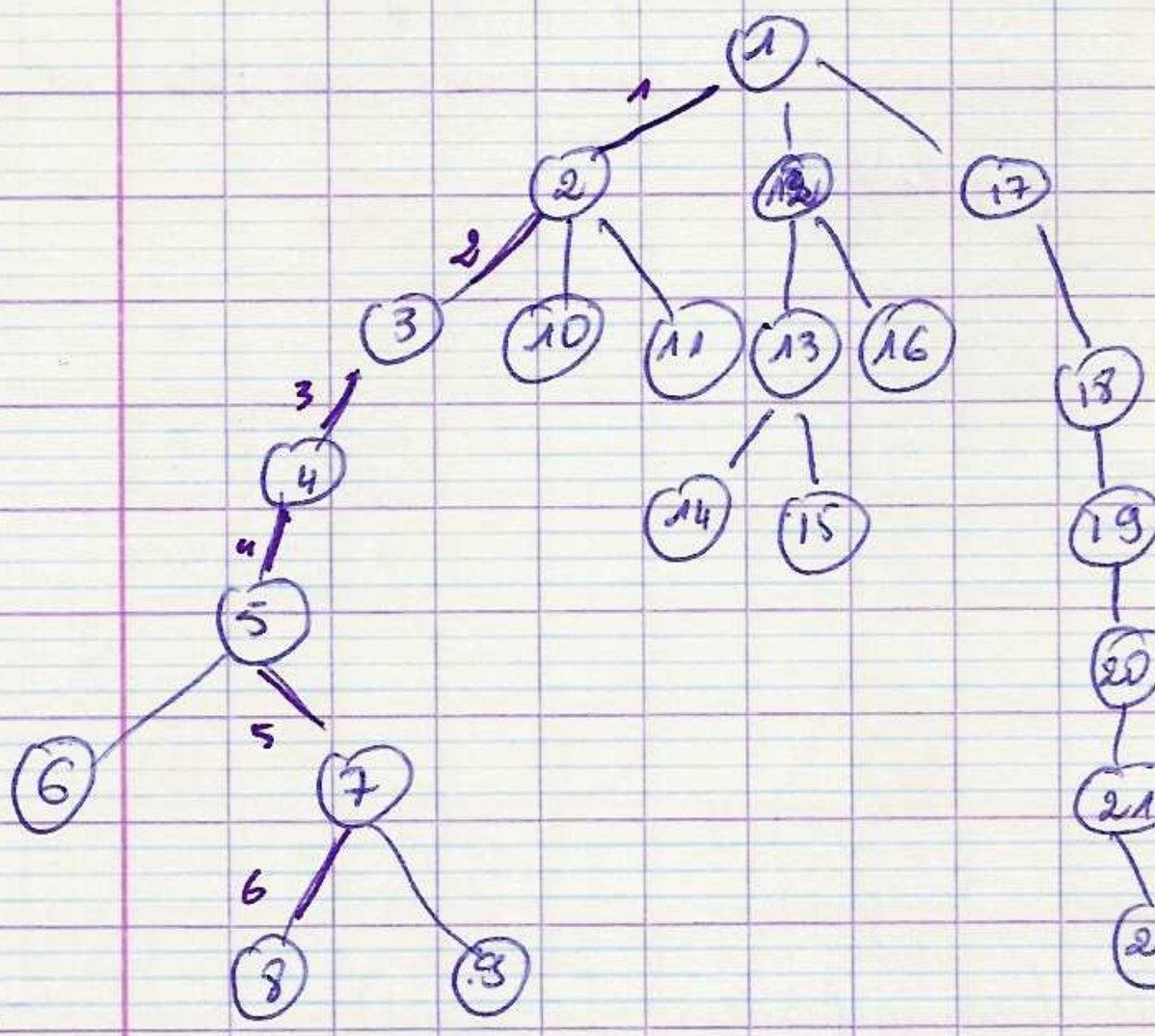
Remarque:

1/ il faut un espace de recherche fini et sans cycle

⚠ . Attention! au cycles infinis

2/ il est necessaire d'éliminer les noeuds déja rencontrés.

Numéroter cet arbre de recherche selon l'ordre de visite par profondeur d'abord:



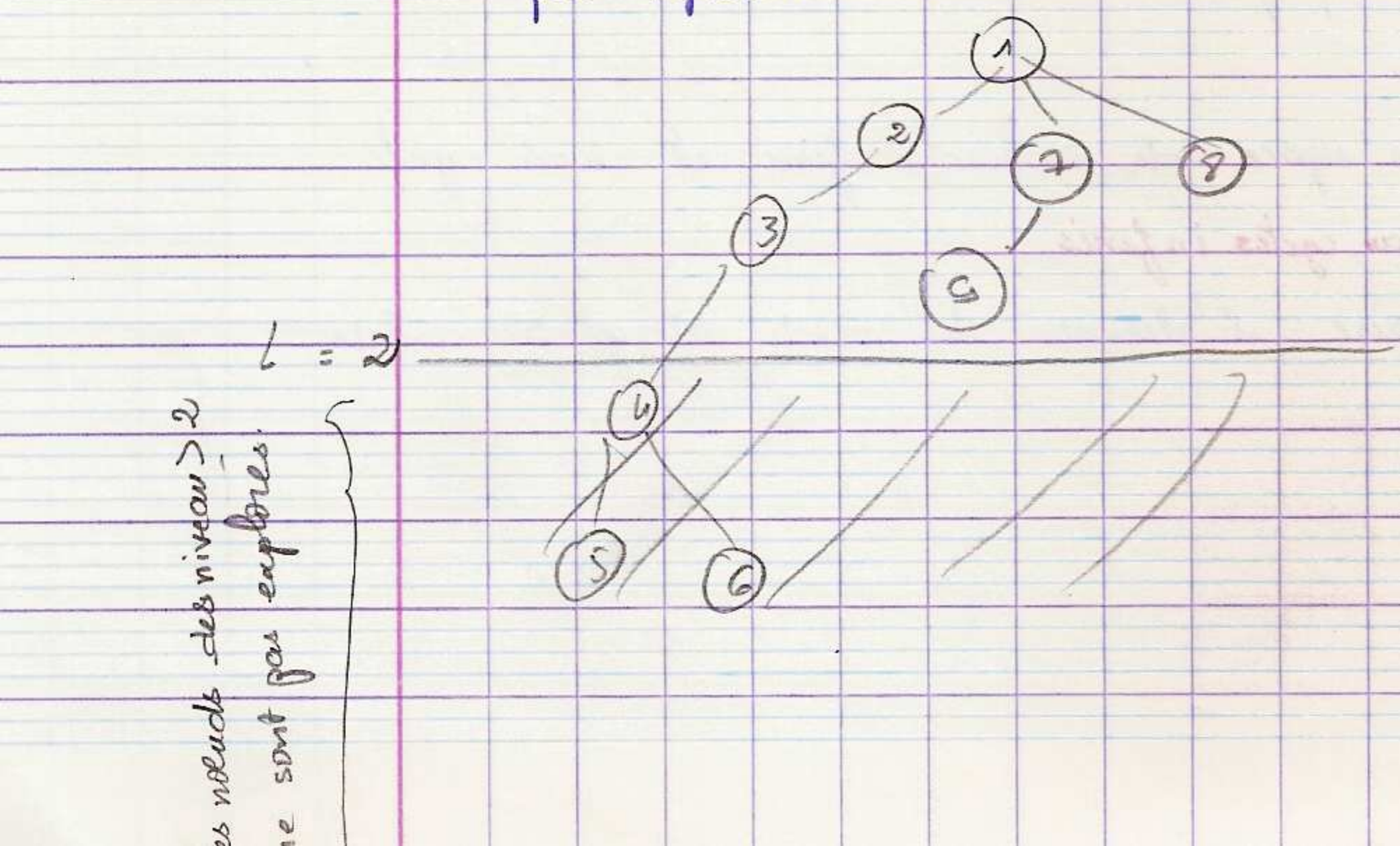
m = 6

* Complétude : Non (echoue dans les espaces infinis ou avec cycle (ou complète si l'espace est fini sans cycle))
* Complexité en espace : $O(b \times m)$
* Complexité en temps : $O(b^m)$. terrible si m beaucoup plus grande que d.
* Optimalité : Non

3. La Methode en profondeur limitée:

Stratégie: Suit l'algorithme de recherche en profondeur avec une limité d'exploration L.

Implémentation: les successeurs des noeuds de profondeur L ne sont pas explorés.



- Completude → oui (si L ≥ d)
  → Non. (sinon)

- Complexité en temps $\theta(b^L)$
- Complexité en espace $\theta(b * L)$
- Optimalité : Non

T.D (Jeu de taquin 3x3)

Résoudre la même Configuration du jeu de taquin 3 x 3 en appliquant la recherche en profondeur limitée L = 3