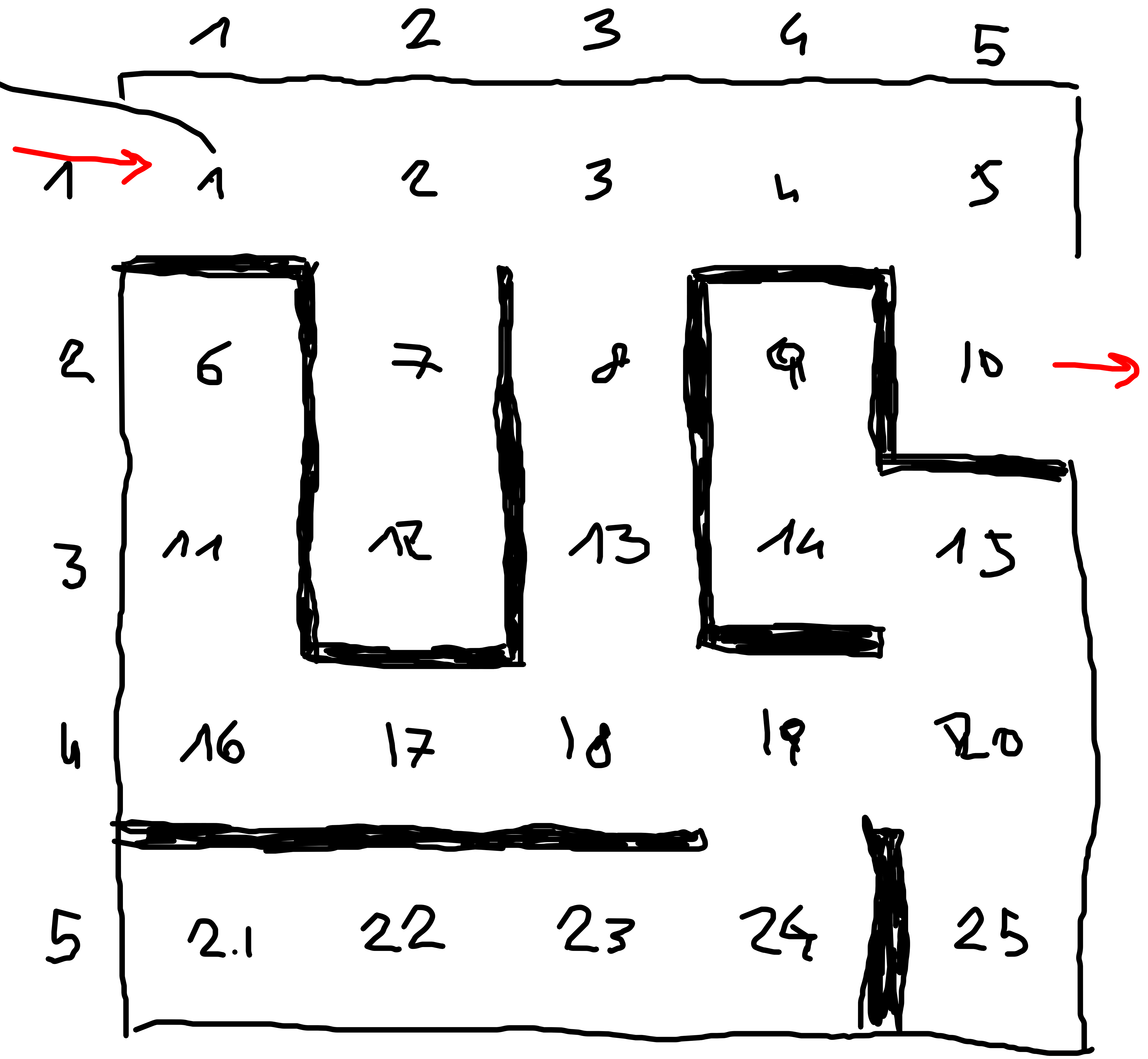
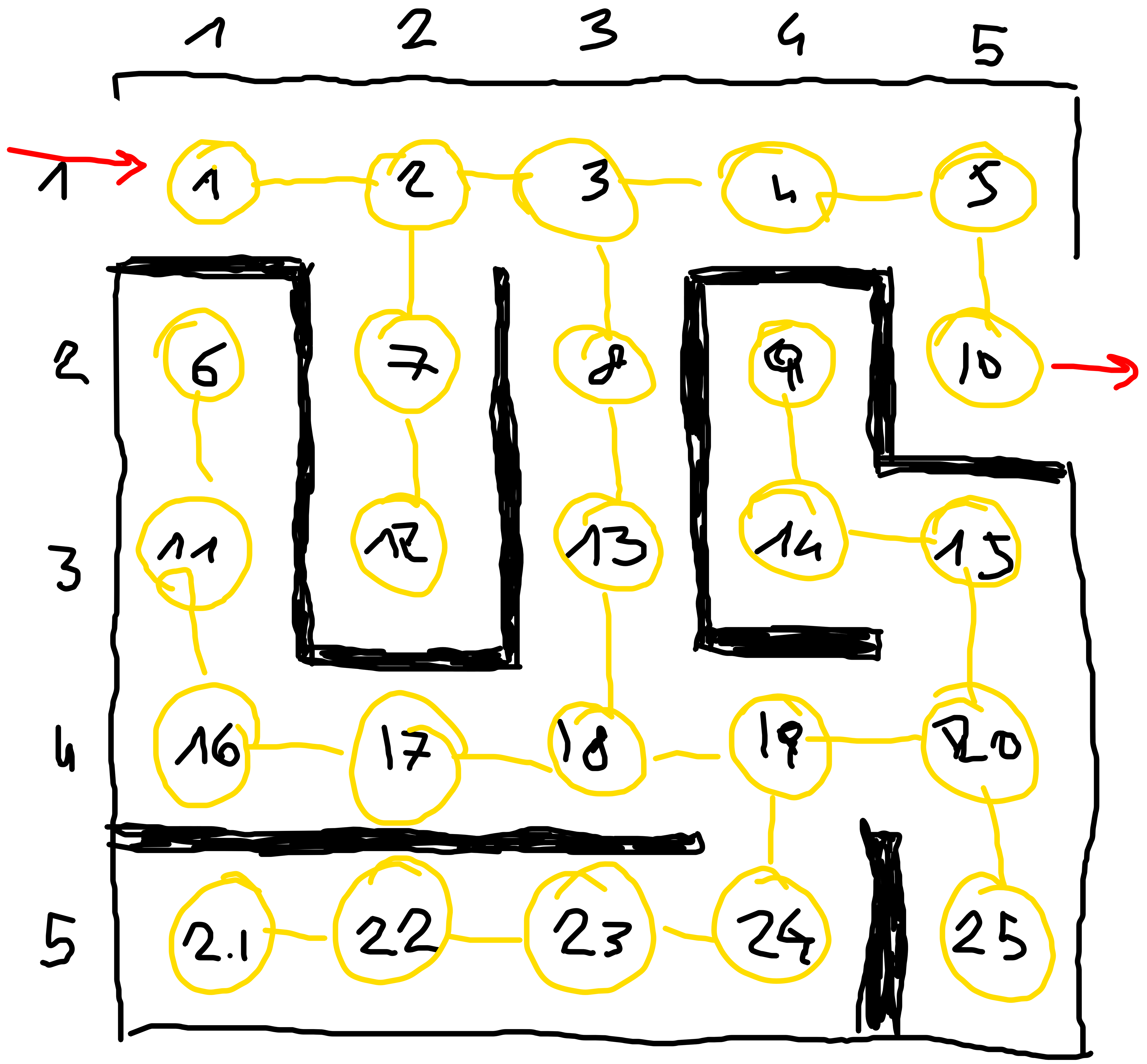
Servono solo a numerare le caselle



Una regola per uscire da un labirinto è:
"Segui sempre il muro alla tua destra".

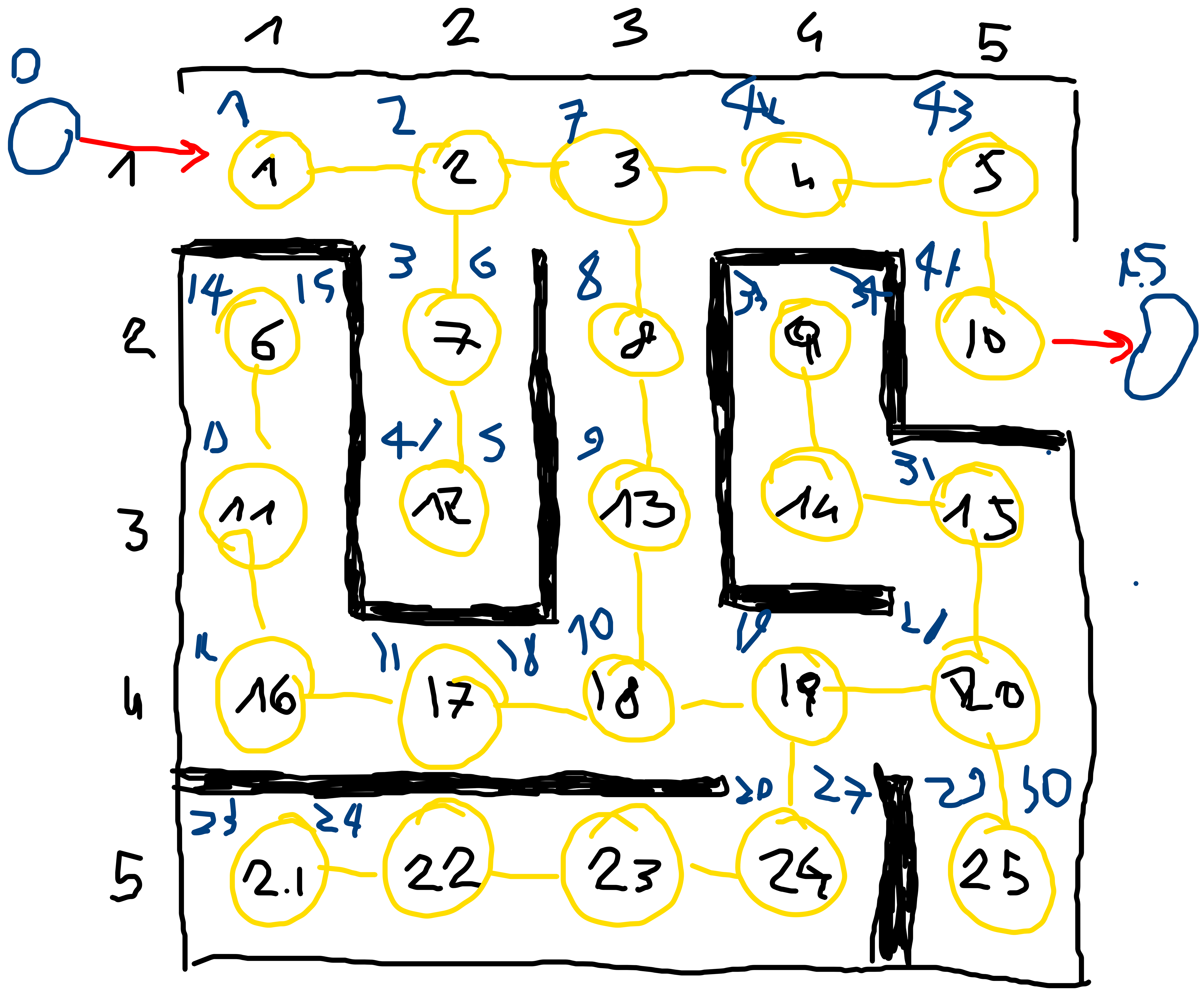
- Modella il labirinto come un grafo
- Trova l'uscita in almeno 2 modi
- Discuti le differenze.



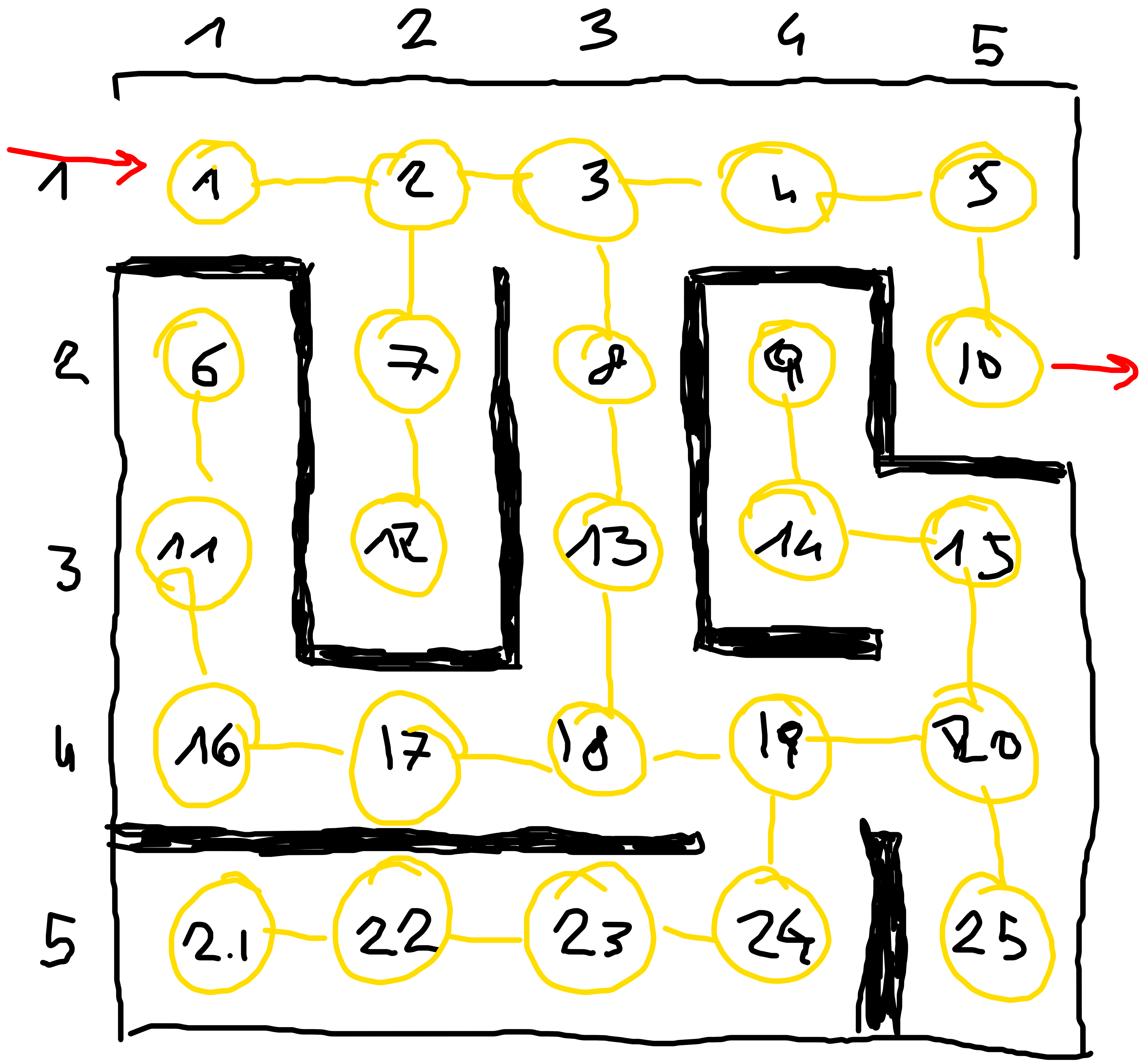
Una regola per uscire da un labirinto è:
"Segui sempre il muro alla tua destra".

- Modella il labirinto come un grafo
- Trova l'uscita in almeno 2 modi
- Discuti le differenze.

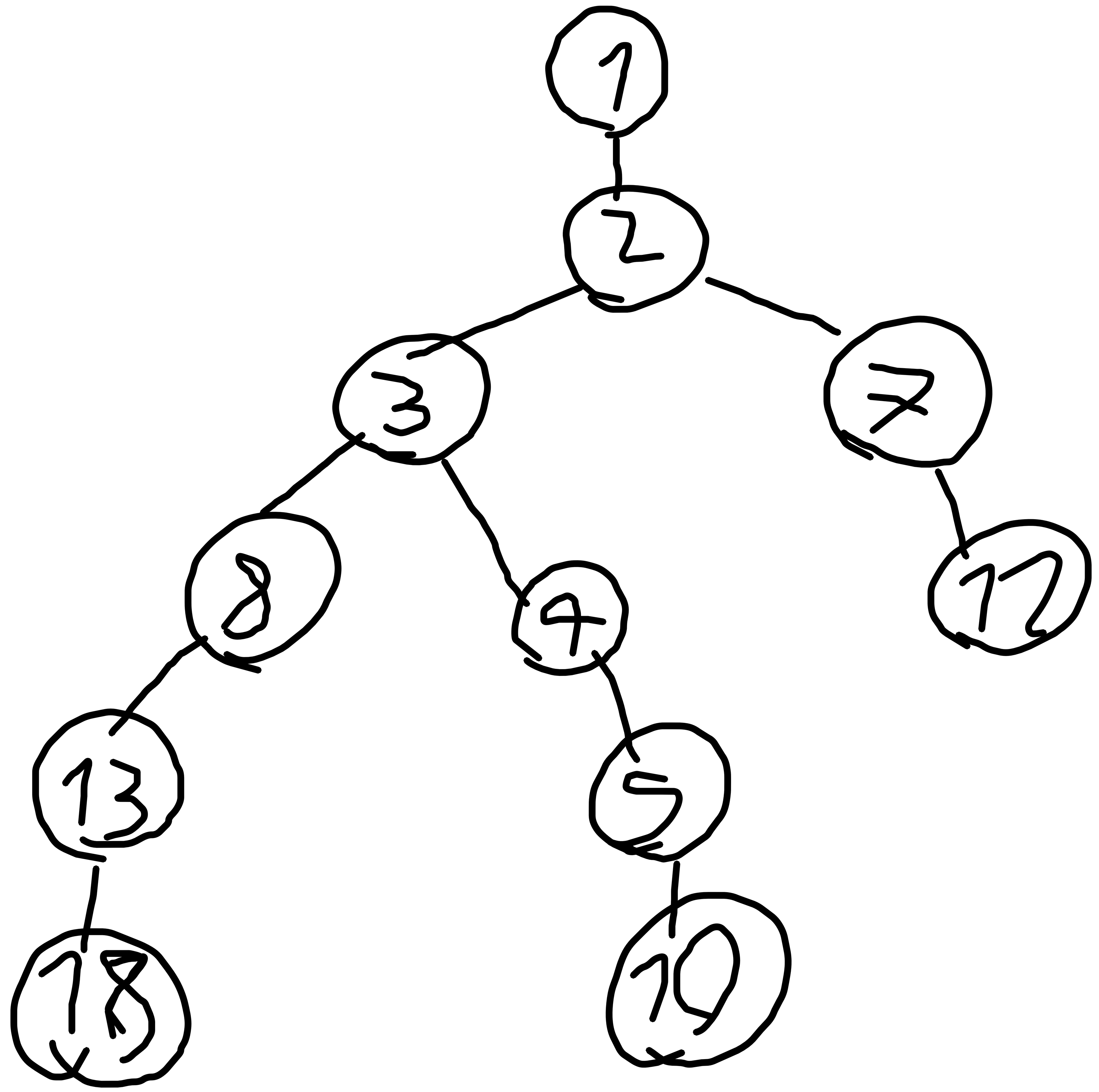
Usiamo DFS



Usiamo BFS

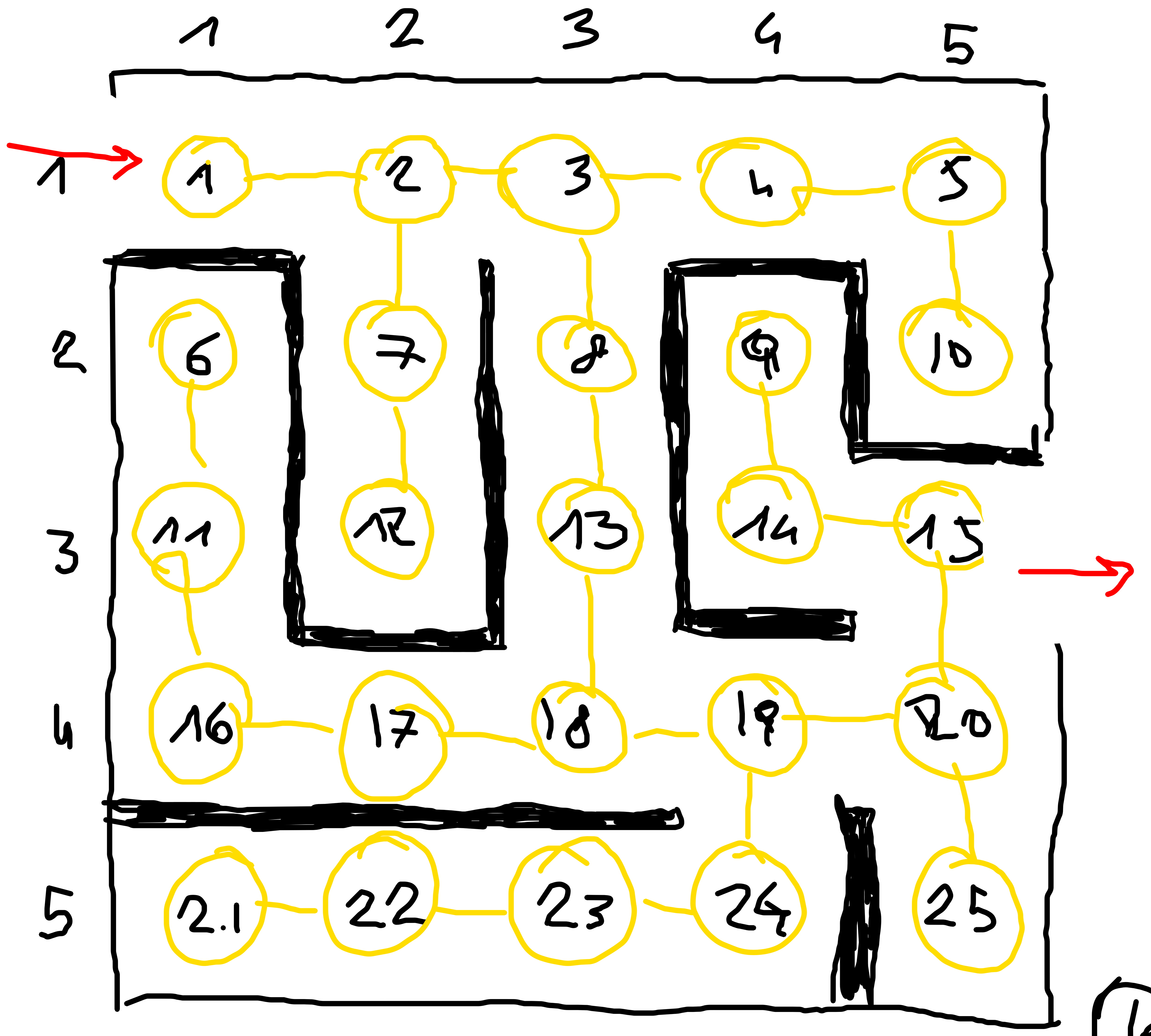


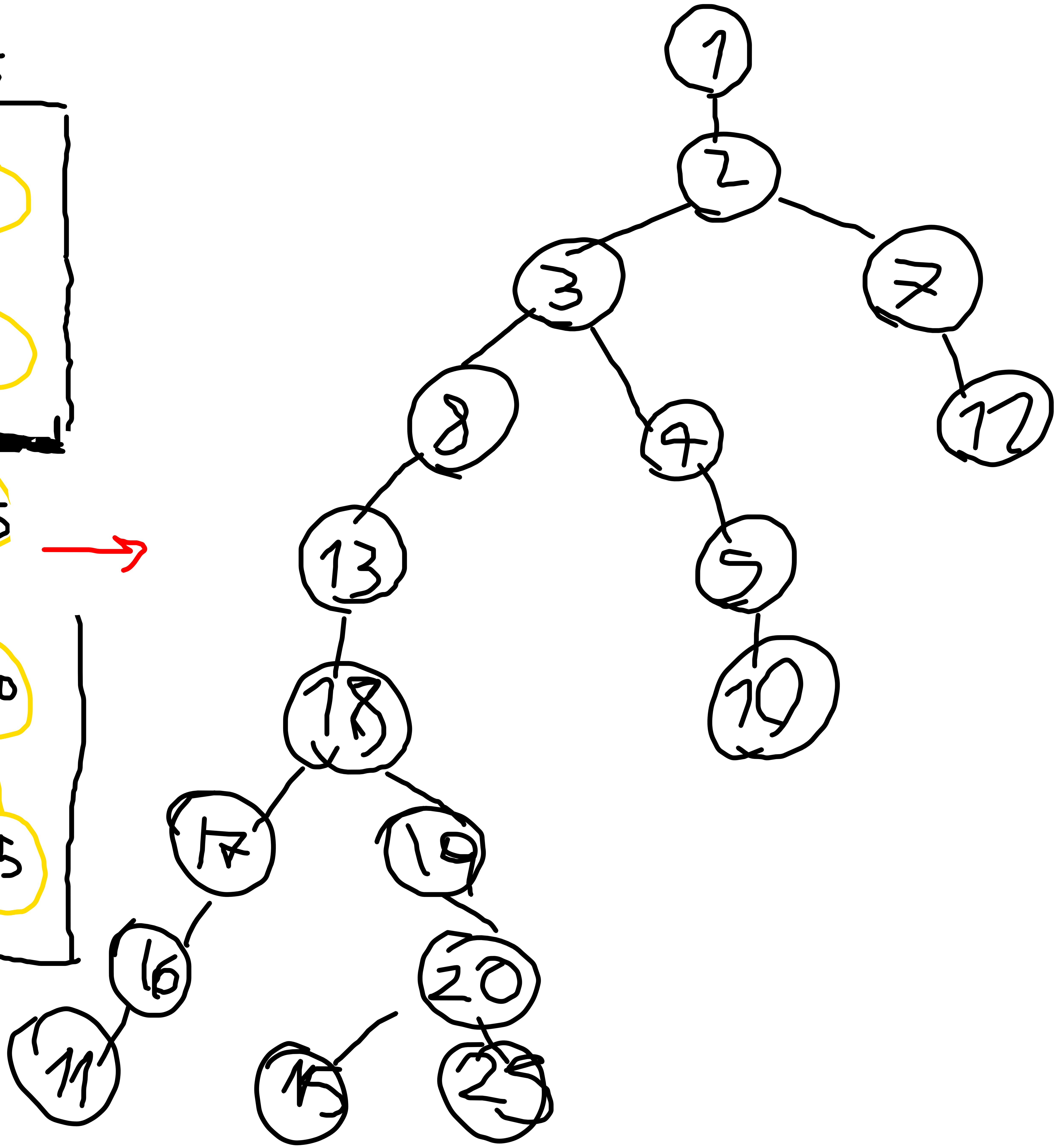
1
2
3
4
5
1
2
3
4
5
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

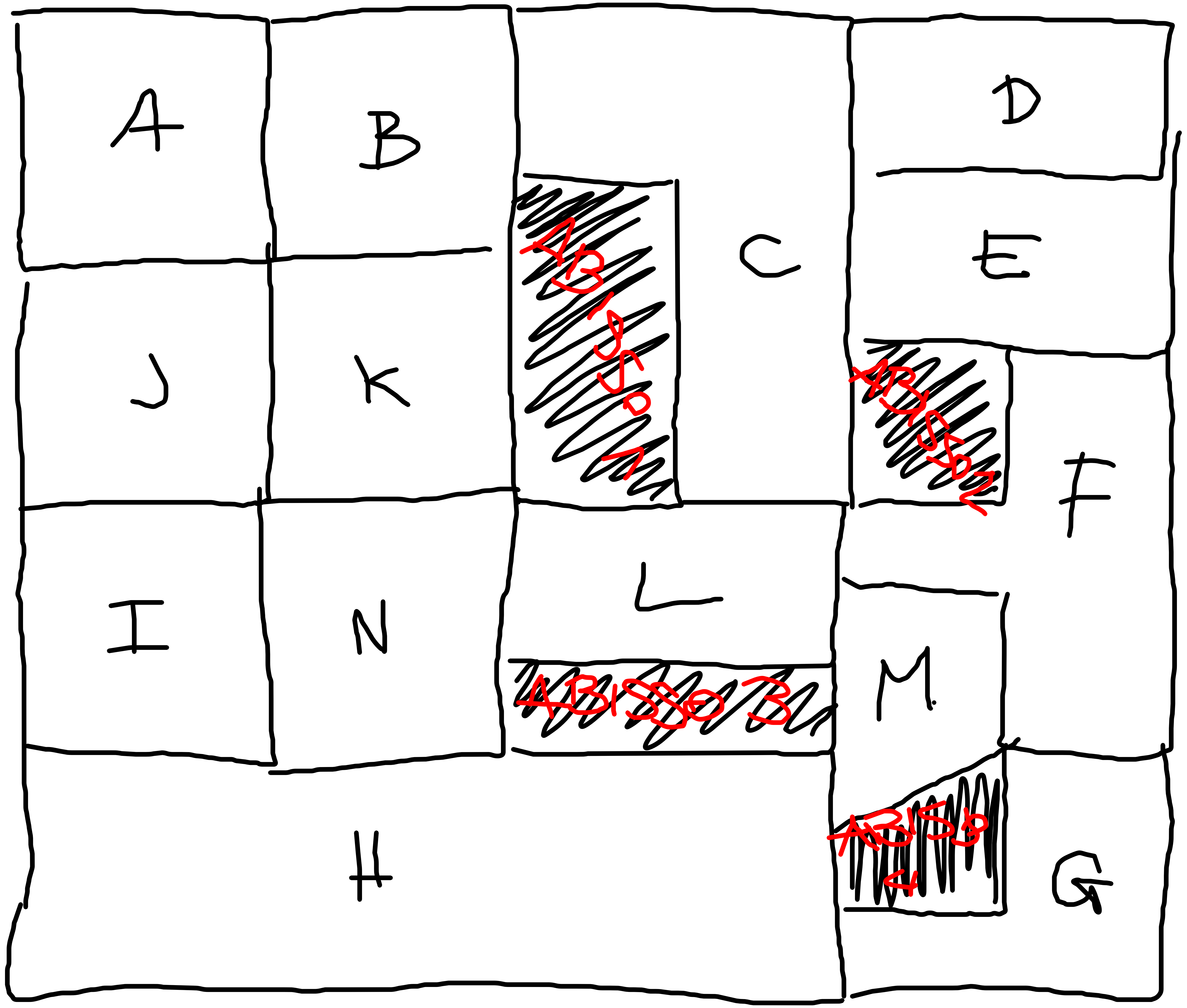


1
2
3
7
8
4
12
13
5
18
10

Usiamo BFS


1
2
3
4
5
1
2
3
4
5
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25

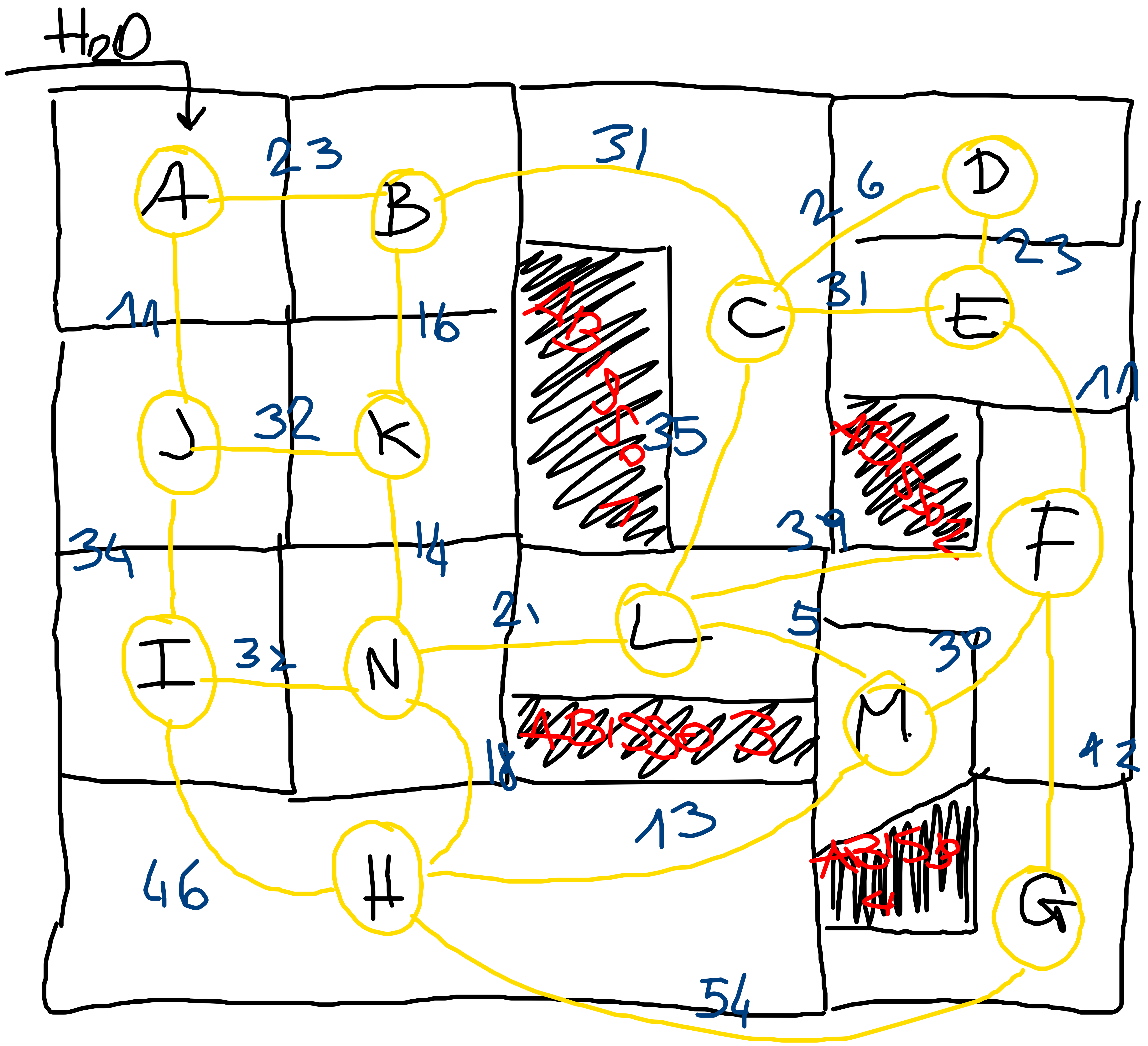

1
2
3
7
8
4
12
13
5
18
10
17
19
16
20
11
15
25



Un contadino vuole irrigare i suoi campi utilizzando un sistema che costi il meno possibile.

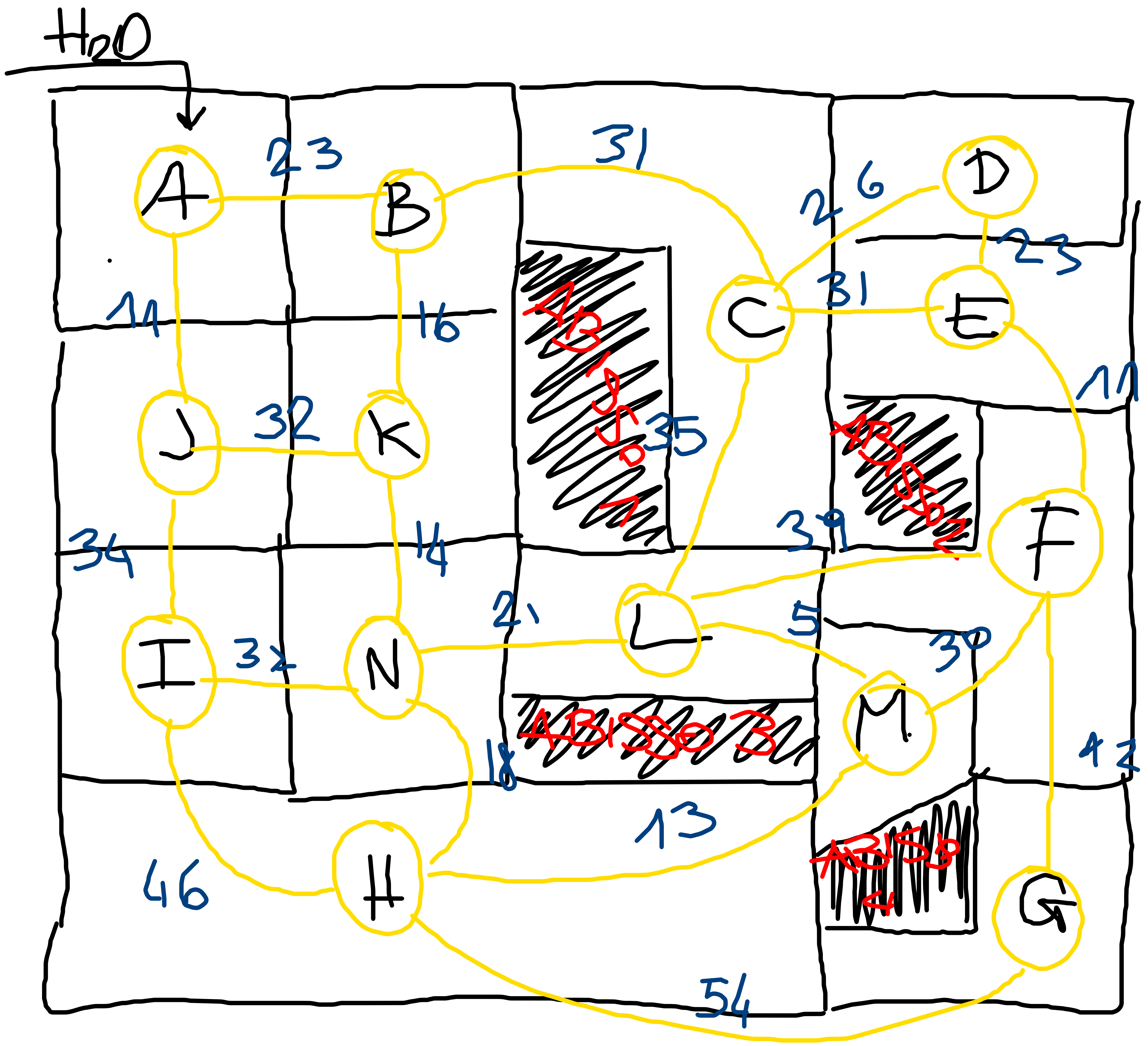
Possiamo aiutarlo?

Sì, modelliamo questo problema come un grafo



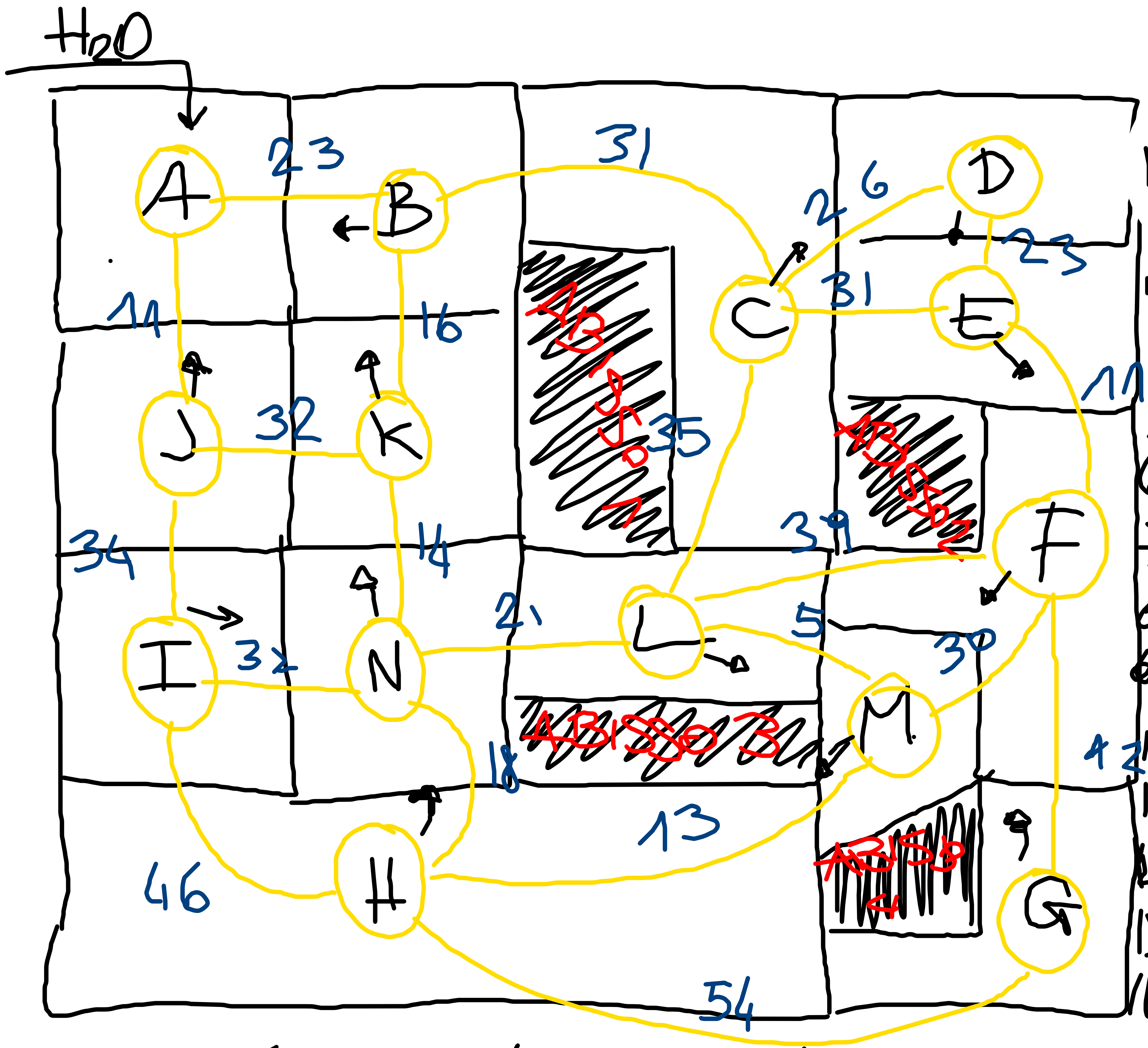
Troviamo il grafo, determiniamo il costo dei tubi e gli facciamo un preventivo 250€.

Il contadino è dubbioso. Ha ragione?



- Prim implementa una coda di min priorità in cui ad ogni nodo è associata una chiave
- Dopo init ($Key = \infty \; \forall u \in V - \{v\}$, $v.key = 0$) estraiamo il nodo con la chiave minore e updatiamo la chiave di $Adj(v)$ in base a $w(v,u)$
- Finiamo dopo aver VISITATO tutta la coda

H_2O



APPLICHIAMO PRIM

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1)	0	∞	∞	·	·	·	·	·	·	·	·	—	·	∞
2)		23								11				
3)		23							342		32			
4)			31						34		16			
5)			31						34					4
6)								18	32			21		
7)							54		32			21 13		
8)			31			30	54		32			5		
9)			31			30	54		32					
10)			31		11				32					
11)			31	23			12		32					
12)			31	23					32					
13)			26						32					
14)									32					

(N, I, 32) ✓

(A, J, 11) (A, B, 23) (B, K, 16) (K, N, 14)
(N, H, 18) (H, M, 13) (M, L, 5) (M, F, 30) (F, E, 11), (F, G, 12), (E, D, 23), (D, C, 26)

23.2.1

Supponiamo che il grafo G(V,E) sia rappresentato come una matrice di adiacenza sviluppati una implementazione dell'alg. di Prim con tempo $O(V^2)$

- Devo eliminare dall'algoritmo le operazioni inerenti la coda di MIN PRIORITÀ

- Posso utilizzare una matrice A di supporto

$$A = \begin{array}{c|ccccccc} u & 1 & 2 & \dots & & i & \dots & n \\ \hline v & & & & & & & \\ w_{uv} & & & & & & & \\ \pi u & & & & & & & \end{array}$$

v → vertice alla fine dell'arco di peso minimo uscente da u

w_{uv} → peso dell'arco u,v

πu → padre del nodo u

```
FOR i in 1 .. |V|
    A[1,i] = NIL
    A[2,i] = ∞
    A[3,i] = NIL
S = {r}
FOR i in 1 .. |V|
    IF Adj(r,i) != 0
        A[1,i] = r
        A[2,i] = Adj[r,i]
FOR each u in {V - S}
    k = min_i (A[2,i])   → Prendo il nodo associato al peso più minimo
    S = S ∪ {k}
    A[3,k] = A[1,k]
FOR i in 1 .. |V|
    IF Adj(k,i) != 0 & Adj[k,i] < A[2,i]
        A[2,i] = Adj[k,i]
        A[1,i] = K
```