Acheminement de messages

Alain Cournier

Master 1 Informatique

Université de Picardie

Contenu du cours

- Commutation
- Congestion et interblocage

La commutation

C’est la stratégie utilisée pour acheminer un message depuis une source vers une destination.

Le chemin emprunté par le message sera défini localement en utilisant la table de routage qui aura été calculé précédemment.

Mais connaitre la route ne donne pas la méthode de transport.

La commutation de circuits

- La commutation de circuits consiste à mettre en relation successivement les différents nœuds intermédiaires afin de propager la donnée du nœud émetteur au nœud récepteur. Dans ce type de scénario, la ligne de communication peut être assimilé à un tuyau dédié à la communication.

- c'est le modèle du téléphone. On réserve une suite de liens par l'envoie d'un en-tête contenant la destination, puis à la réception d'un accusé de réception, la source transmet le message en un coup.

  Conséquence : On peut bloquer beaucoup de liens pour un message court.

La commutation de messages (store-and-forward)

- le message avance pas à pas (nœud par nœud) vers la destination. A chaque étape, le message est effacé d'un nœud dès qu'il est stocké sur le nœud suivant. Un seul nœud à la fois est utilisé comme ressource pour enregistrer le message.
  - Conséquence : Besoin de stocker le message dans chaque routeur, besoin de mémoire (conflit de type nœud).

Commutation de paquets

- Lors d'une transmission de données par commutation de paquets (en anglais packet switching), les données à transmettre sont segmentés en paquets émis indépendamment sur le réseau.
- Les nœuds du réseau sont libres de déterminer la route de chaque paquet individuellement, selon leur table de routage. Les paquets ainsi émis peuvent emprunter des routes différentes et sont réassemblés à l'arrivée par le nœud destinataire.
- Dans ce type de scénario les paquets peuvent arriver dans un ordre différent que l'ordre d'envoi et peuvent éventuellement se perdre. Des mécanismes sont ainsi intégrés dans les paquets pour permettre un réassemblage ordonné et une réémission en cas de perte de paquets.

- Il s'agit du mode de transfert utilisé sur internet, car il comporte les avantages suivants :
  - Résistances aux pannes des nœuds intermédiaires
  - Utilisation rationnelle et efficace des lignes de transmission

Commutation de paquets

On peut voir la commutation par paquet comme :

1. Découper à la source le message M en paquets p_1, p_2, …,p_k
2. Acheminer chacun des p_i de façon indépendante dans le réseau depuis la source vers la destination.
3. Sur le nœud destination, reconstruire le message M à partir des paquets p_1, p_2, …,p_k.

Commutation Wormhole

On peut voir la commutation par wormhole comme :

1. Découper à la source le message M en paquets p_1, p_2, …,p_k
2. Créer un paquet de tête p_t
3. Créer un paquet de queue p_q
4. Acheminer tous les paquets comme un train dont p_t est la locomotive et p_q le wagon de queue, depuis la source vers la destination.
5. Sur le nœud destination, reconstruire le message M à partir des paquets p_1, p_2, …,p_k.

Commutation Wormhole

- Conséquences :
  - il est impossible d'interrompre (sur un nœud) la diffusion d'un message au milieu de sa propagation (seul le paquet de tête connait la destination).
  - Interblocage de type lien.

Coût d'acheminement d'un message

- Soit M un message que nous souhaitons acheminer dans un réseau R nous souhaitons évaluer le temps nécessaire pour rendre le service.

- On note :
  - T_i le temps d'initialisation du message (découpage, mise en forme de M)
  - T_r le temps de reconstruction du message M
  - T_c le temps de transfert d'un bit entre deux liens dans un routeur
  - T_{trans} le temps de transmission d'un bit entre deux nœuds voisins
  - L_M la longueur du message M (en bit)

Calcul du coût le plus simple

- De voisin à voisin :
  - $T_{vàv}$ = 1 le temps de transmission est constant indépendant de la bande passante et de la longueur du message

- Pour une communication à distance d
  - $T_{sàd}$ = d le temps ne dépend que de la distance entre la source et la destination

Calcul du coût le plus compliqué

- De voisin à voisin :
  - $T_{vàv} = T_i + L_M * T_{trans} + T_R$ le temps de transmission est linéairement dépendant de la longueur du message

- Pour une communication à distance d
  - $T_{sàd} = d\ (T_i + L_M * T_{trans} + T_R)$ pour le store-and-forward
  - $T_{sàd} = T_i + d*L_M * T_{trans} + T_R + (d-1) * T_c$ pour le wormhole
  - $T_{sàd} = T_{circuit} + d*L_M * T_{trans}$ pour la commutation de circuits

Optimisations possibles

- En commutation store-and-forward temps linéaire, il devient avantageux d'utiliser l'effet pipeline. Principe : on découpe le message M en p petits messages de taille égale (on suppose L divisible par p). On transmet les messages à la queue-leu-leu comme pour simuler le routage wormhole.

- But : On cherche à optimiser p en fonction de la distance d, et des paramètres du réseau.
- Lorsque p est bien choisi on obtient des temps similaire au wormhole

Optimisations possibles

- Router les messages suivant des routes disjointes (si c'est possible) au lieu d'un seul chemin, ceci afin de d'augmenter le parallélisme.
- Si on a m chemins arêtes-disjoints de longueur au plus d' supérieur ou égal à d, on peut découper le message initial de longueur L en m blocs et router chaque bloc de longueur L/m (supposée de longueur entière pour simplifier) indépendamment sur chacun des m chemins.

- La parallélisation fait que le temps devient le maximum du temps nécessaire pour l'acheminement de chacun des blocs.

Congestions et interblocages

Qu’es aquò ?

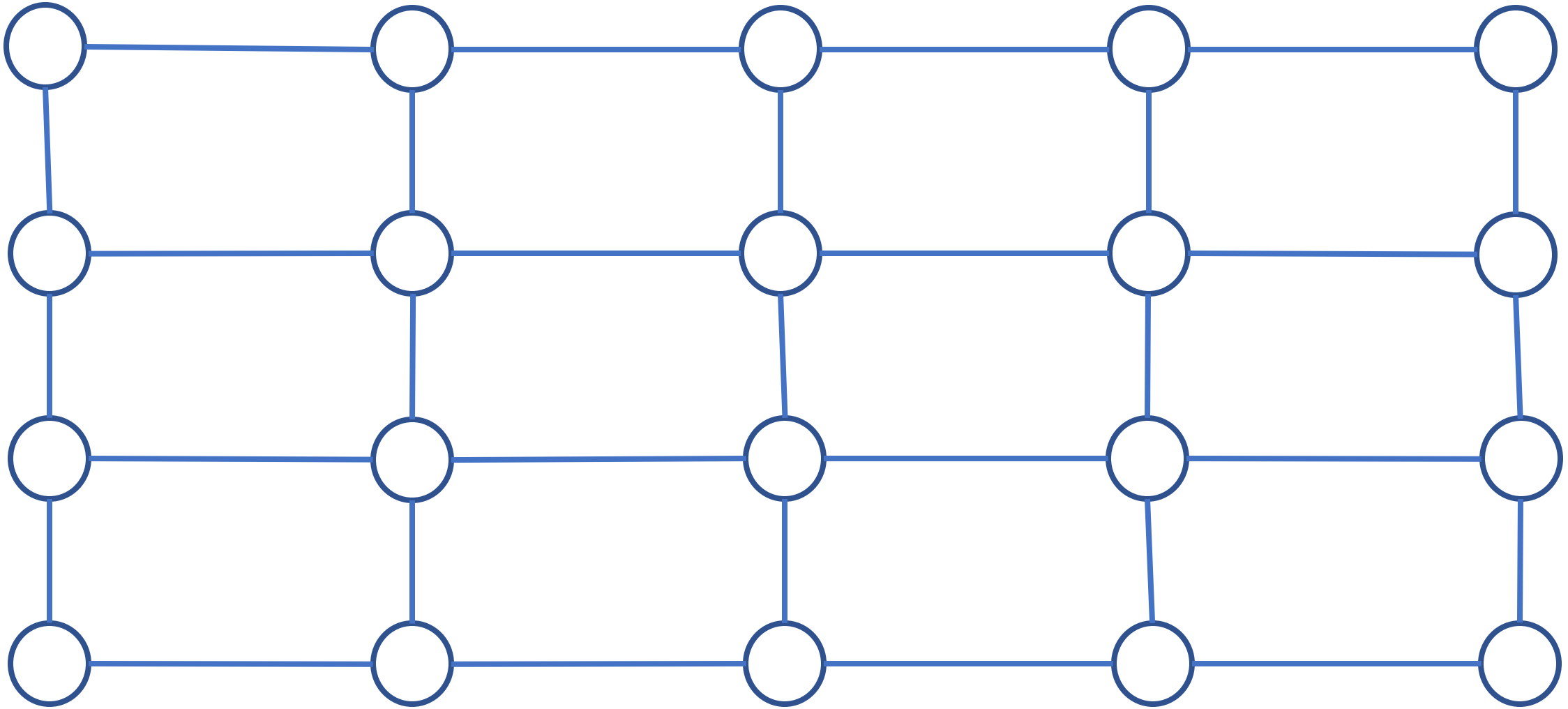
- La congestion dans un réseau consiste à avoir un nombre de messages à traiter supérieur à la capacité d’un nœud ou d’un groupe de nœuds.

- On parle d’interblocage si un ou des nœuds ne sont plus capables de traiter les messages qui doivent les traverser.

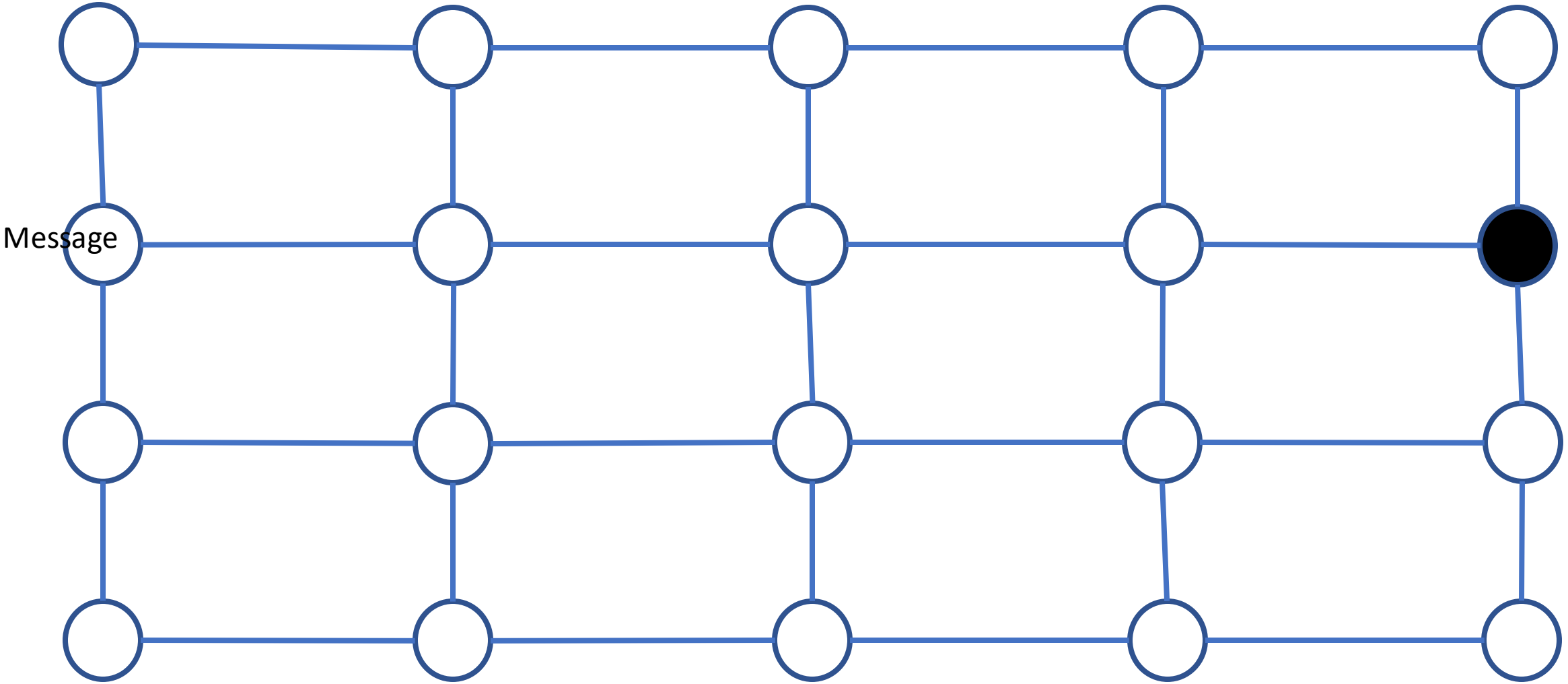
Est-ce possible ?

- Comme nous allons le constater, c'est possible quel que soit la méthode d'acheminement.

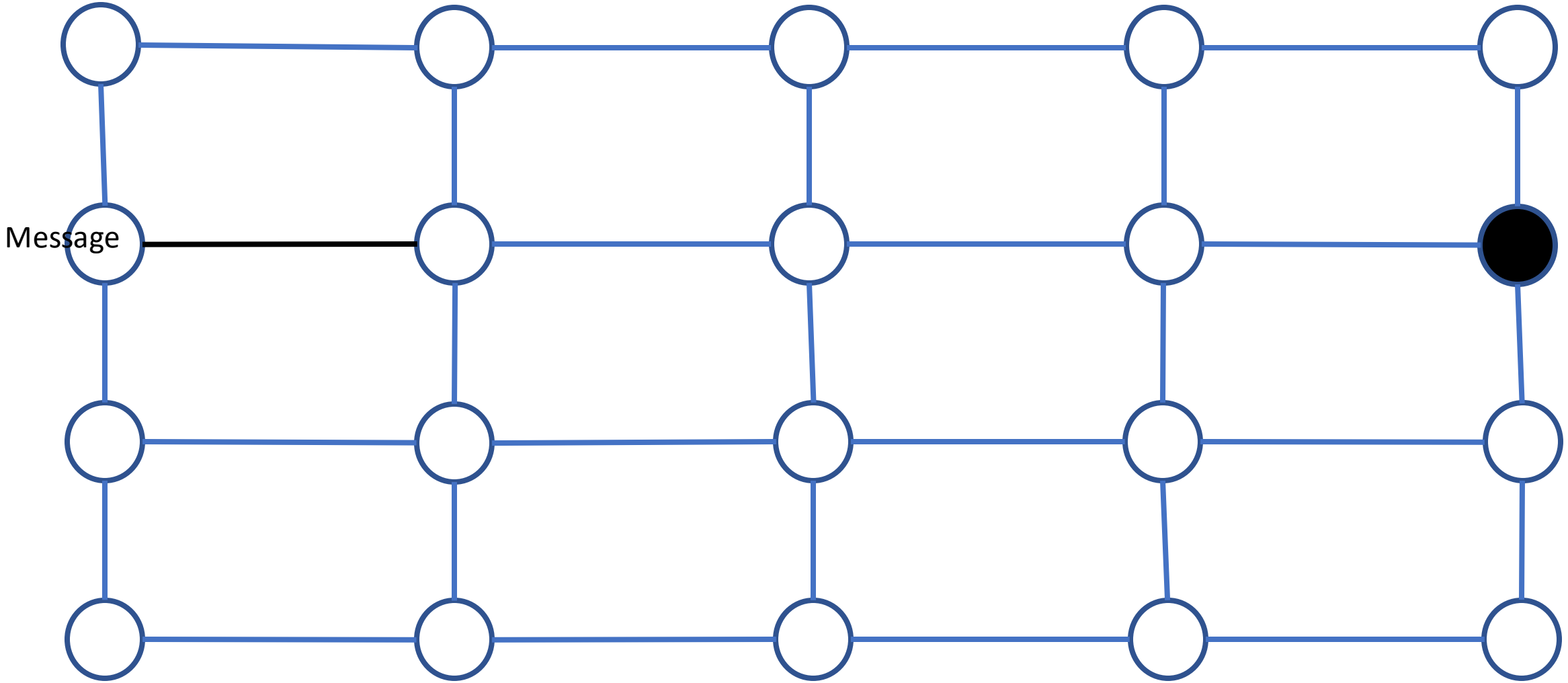
Commutation de circuits



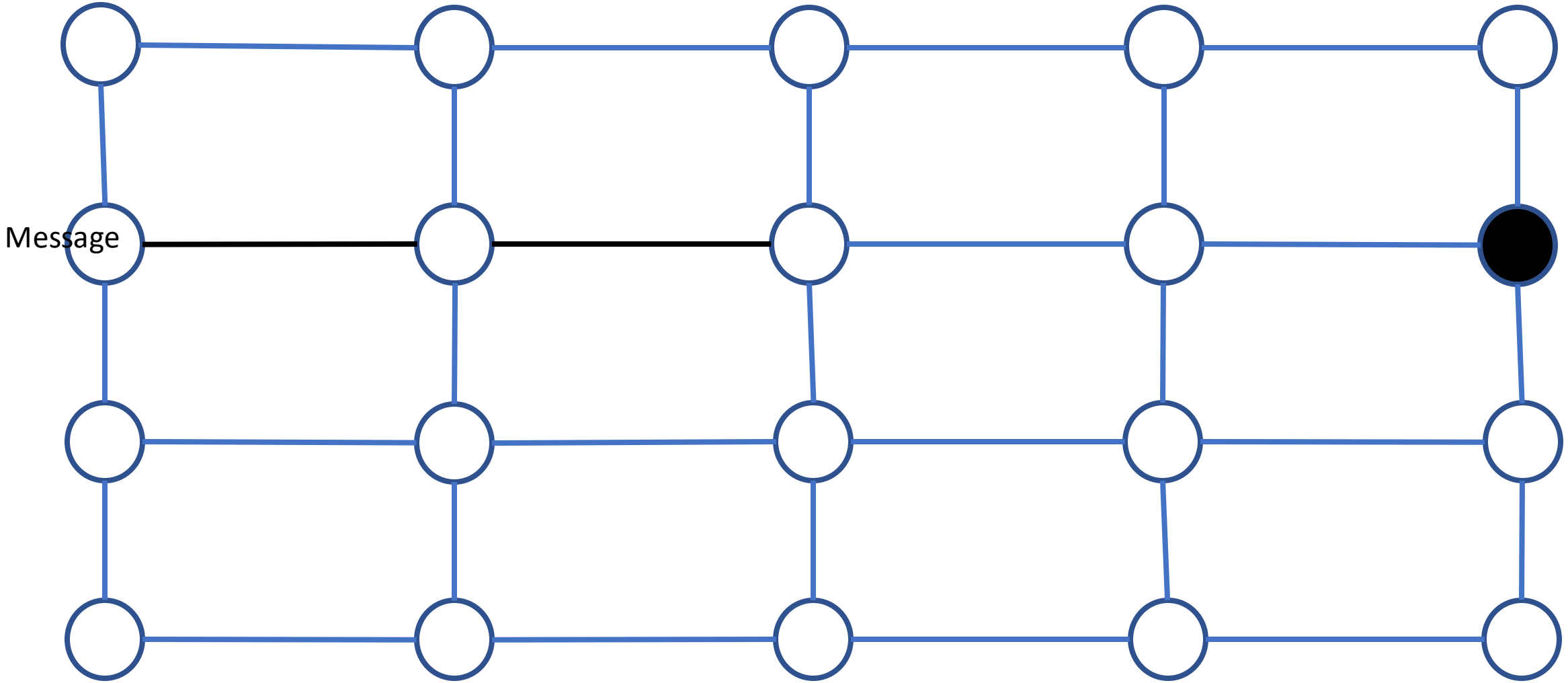
Comment ça marche



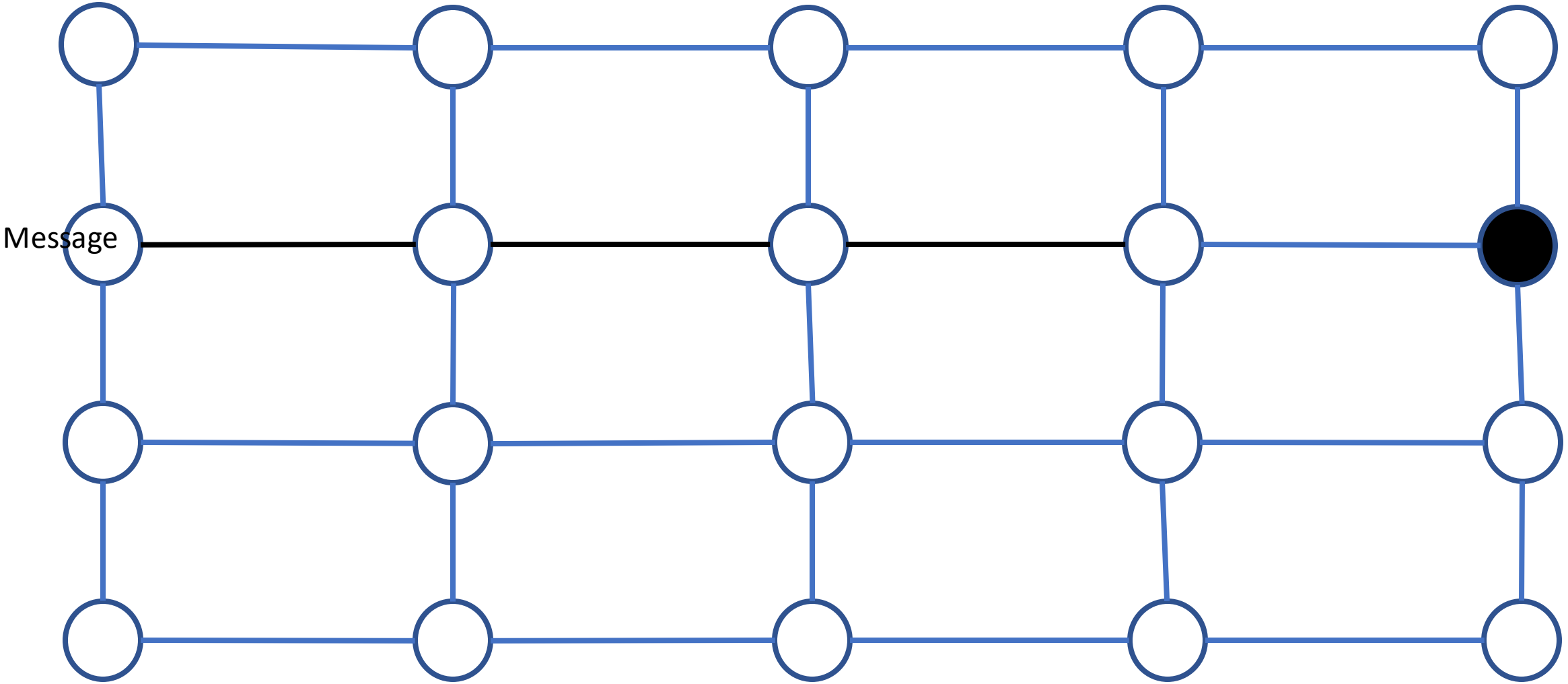
Comment ça marche



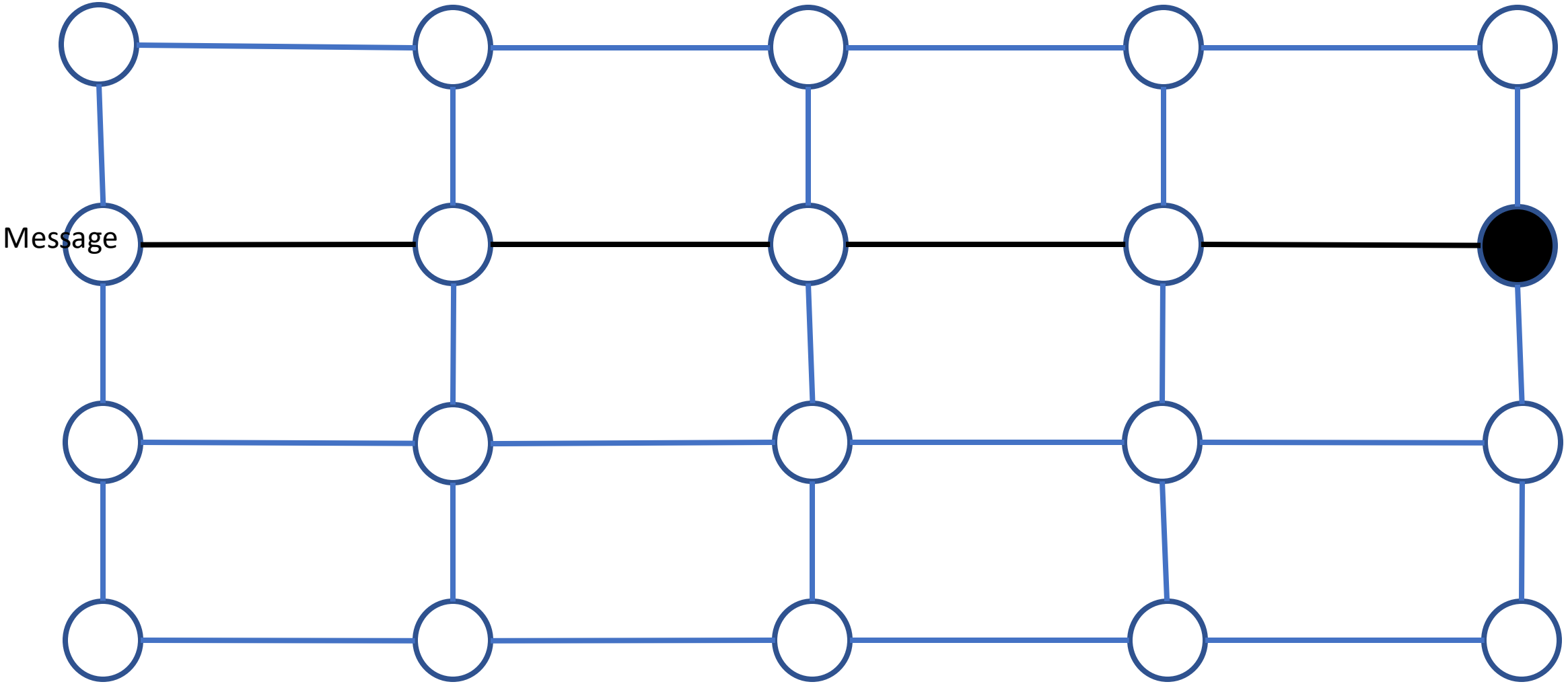
Comment ça marche



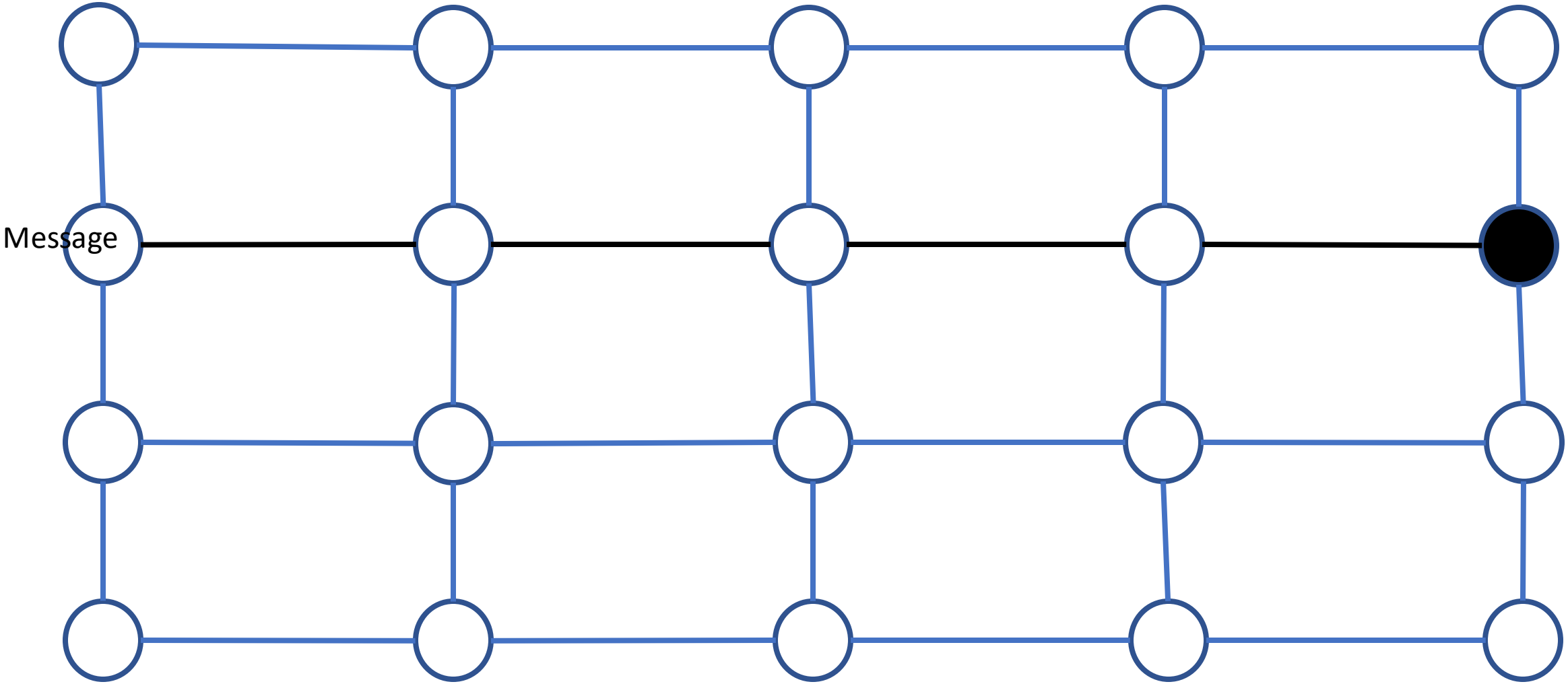
Comment ça marche



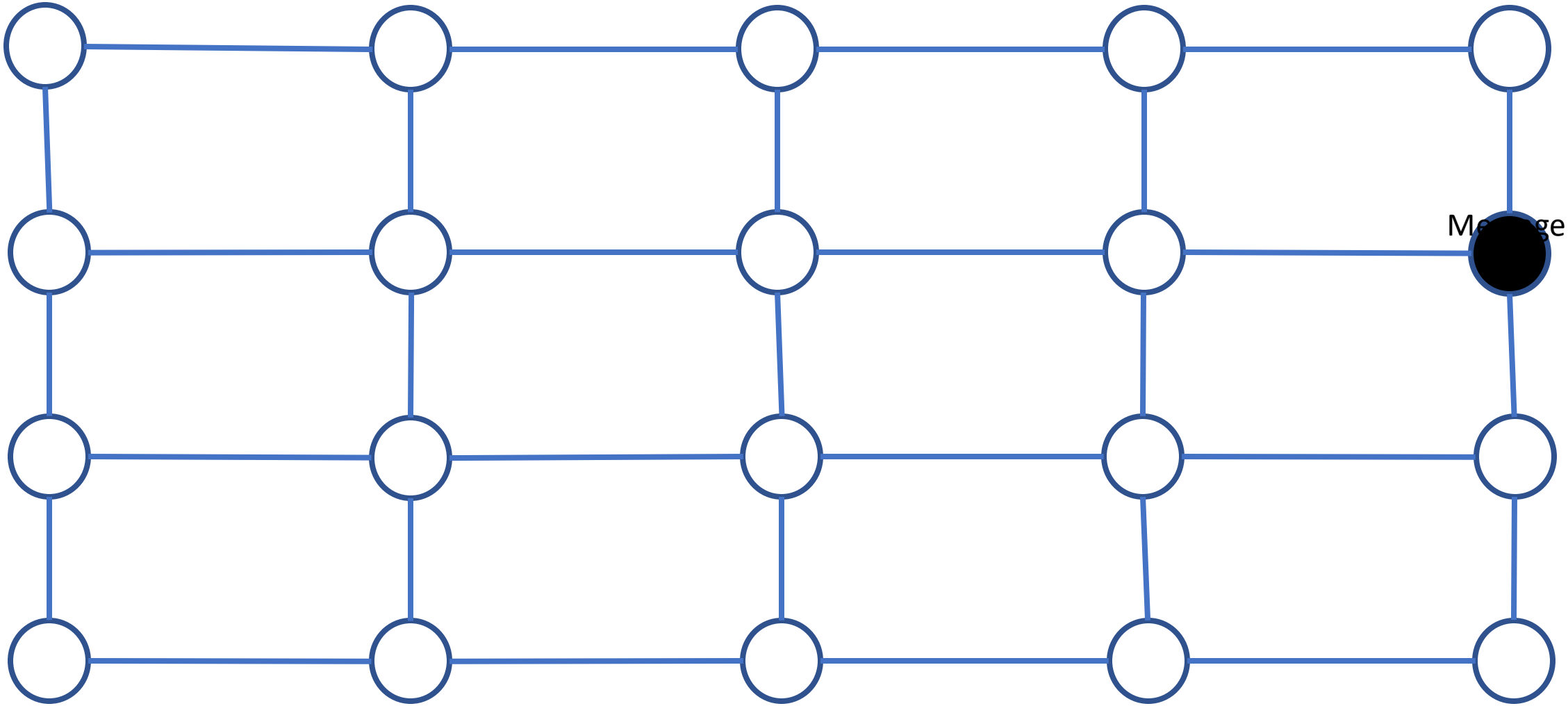
Comment ça marche



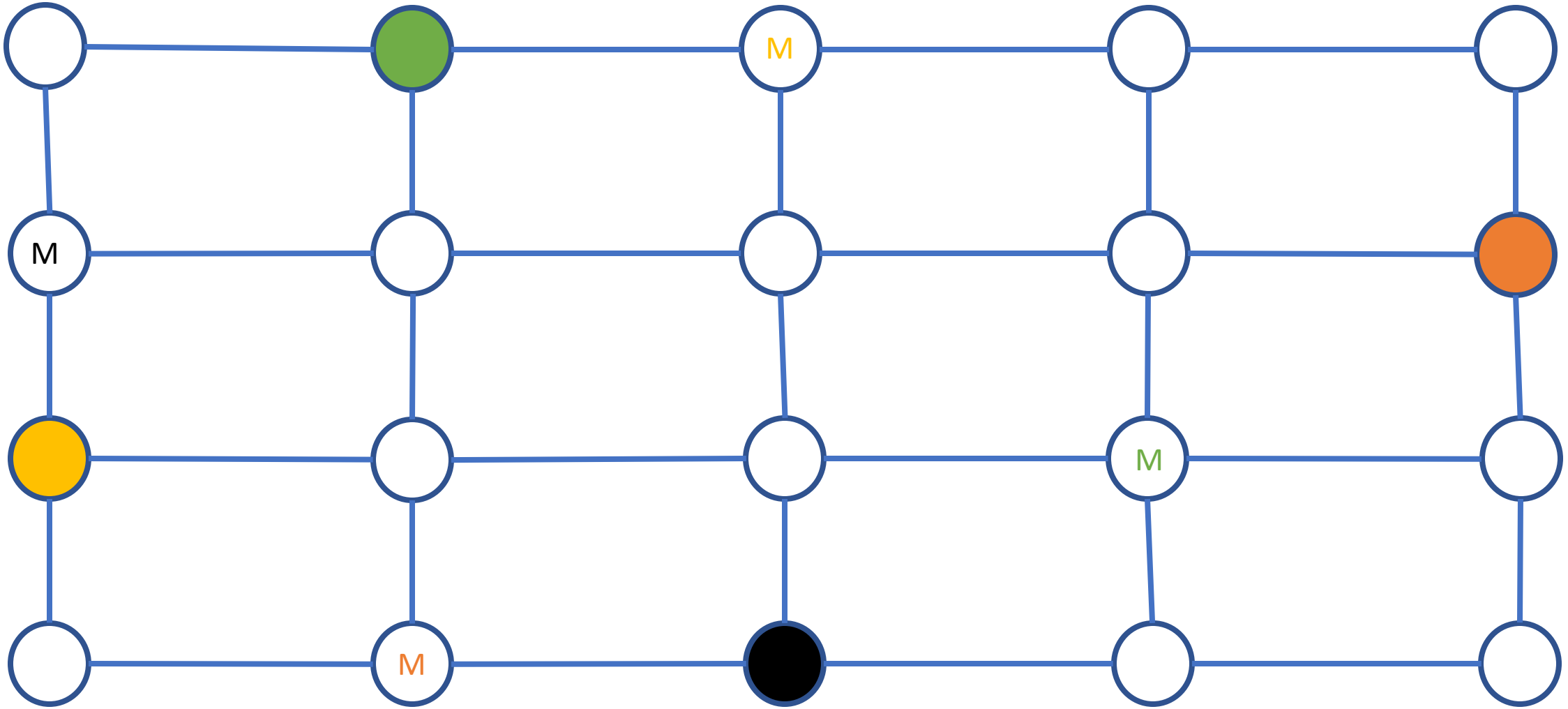
Comment ça marche



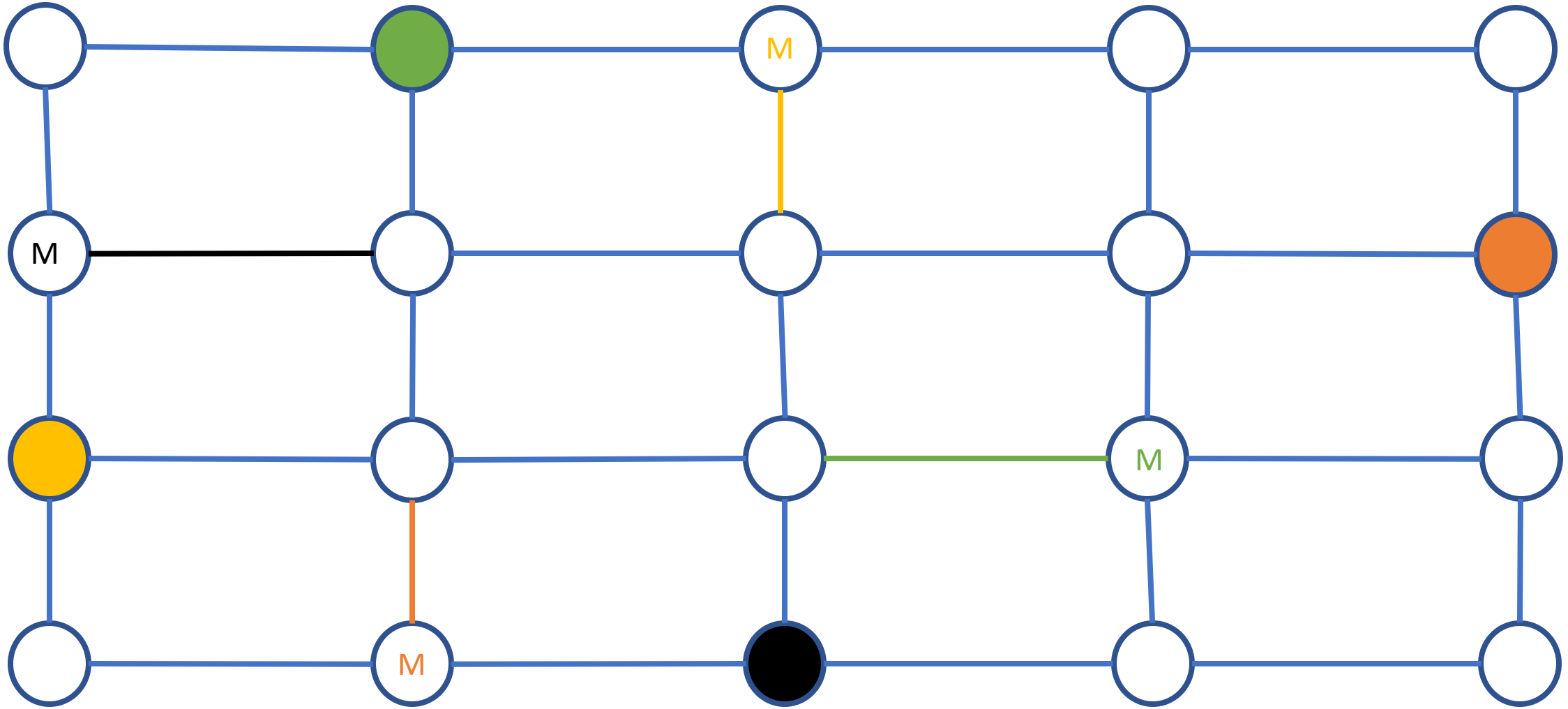
Comment ça marche



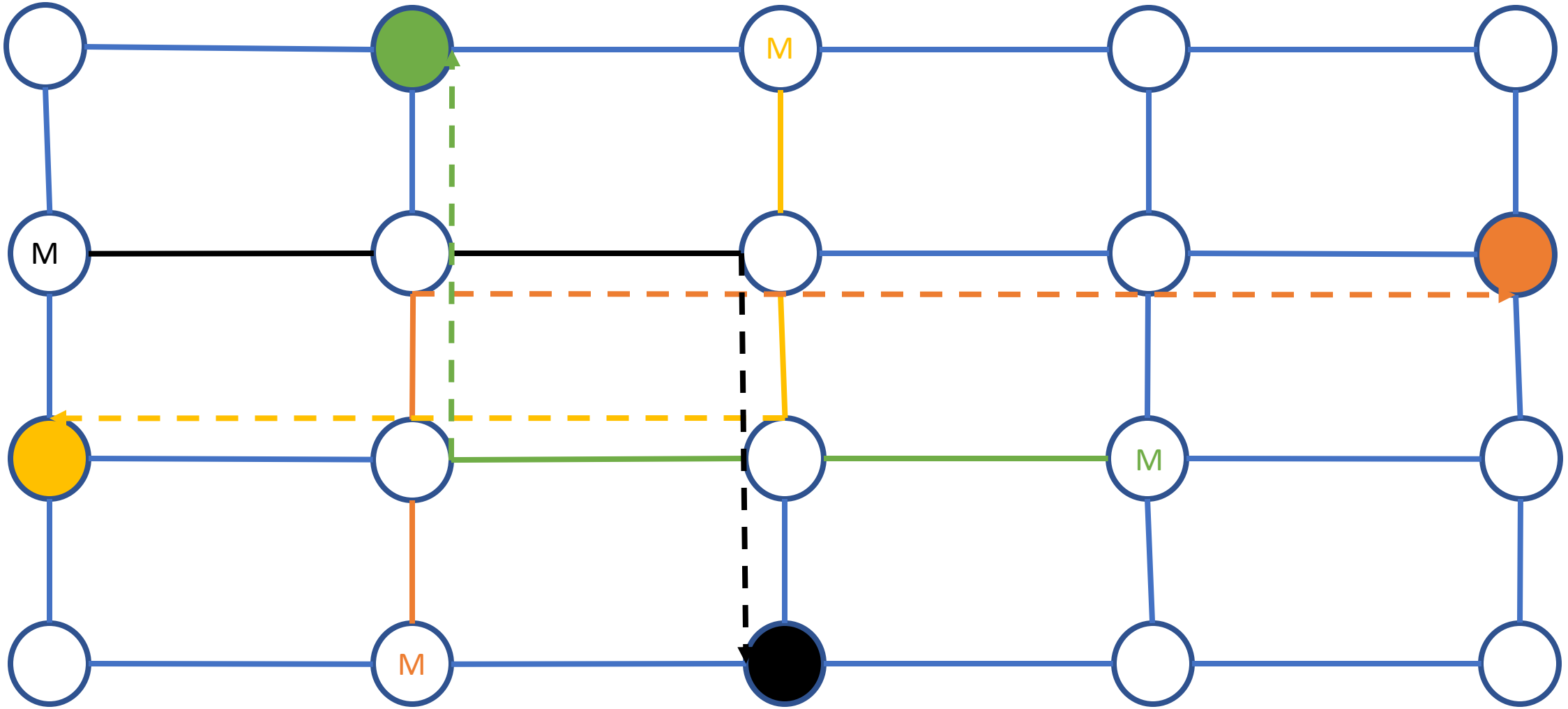
Comment ça marche



Comment ça marche



Comment ça marche PAS !

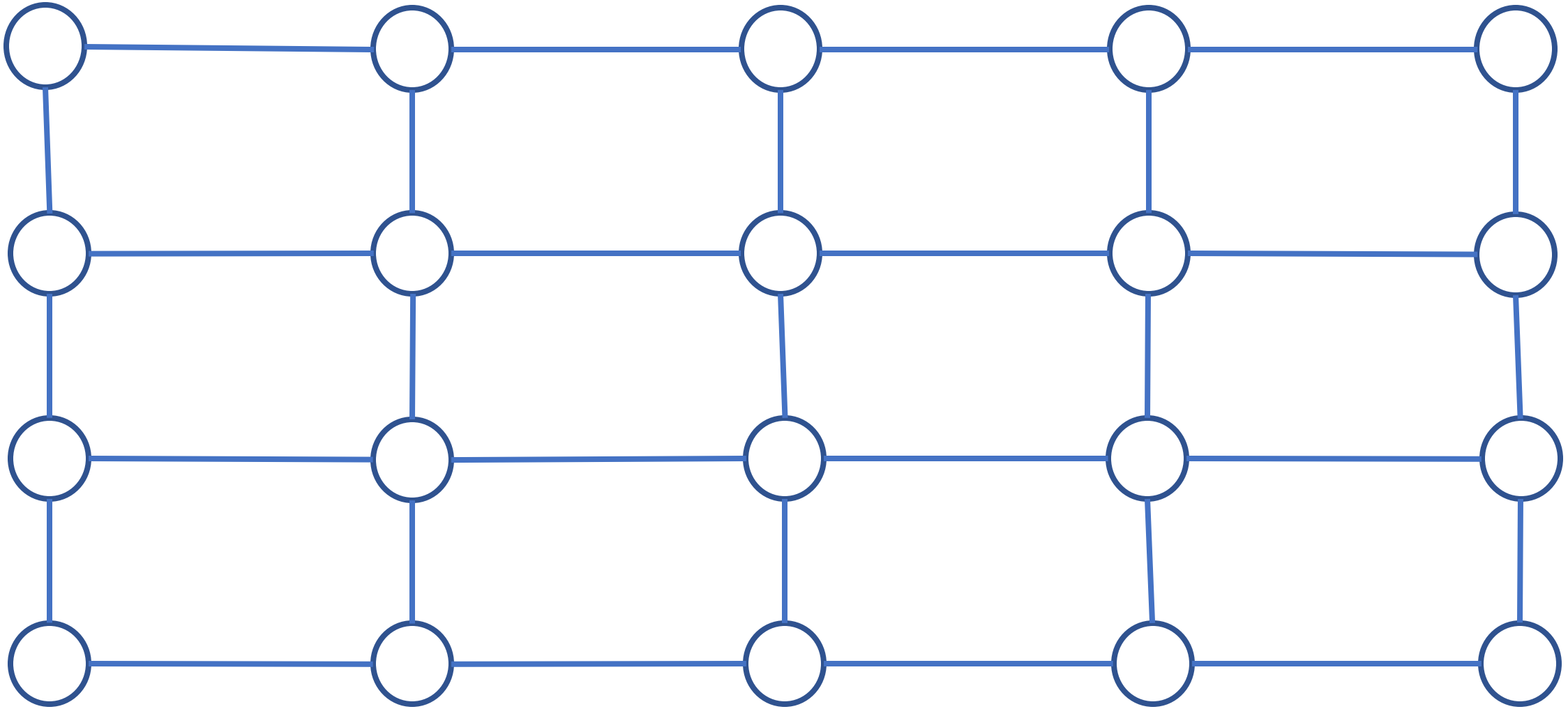


Pourquoi ?

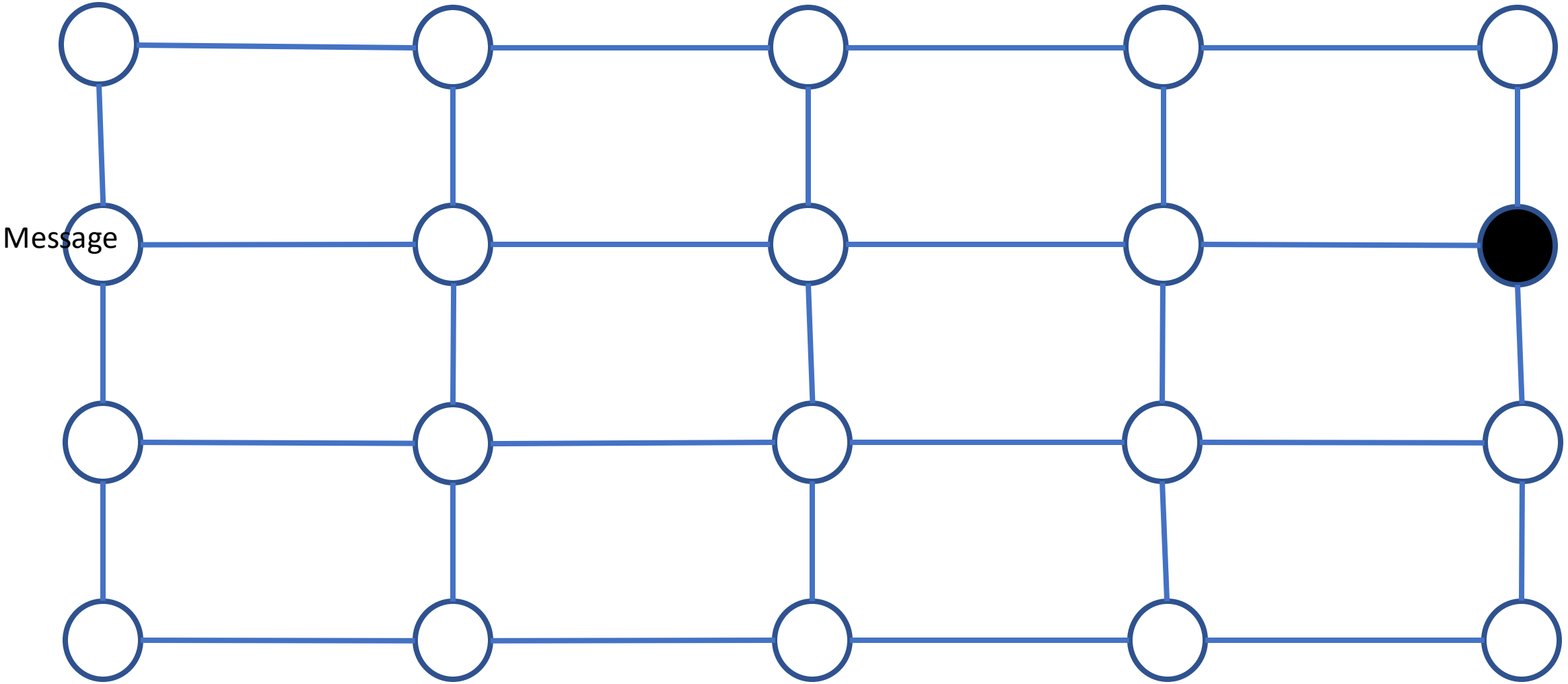
- M monopolise les liens (0,2,E) et (1,2,E) et attend le lien (2,2,S)
- M monopolise les liens (1,0,N) et (1,1,N) et attend le lien (1,2,E)
- M monopolise les liens (3,1,W) et (2,1,W) et attend le lien (1,1,N)
- M monopolise les liens (2,3,S) et (2,2,S) et attend le lien (2,1,W)

- C'est un bête problème de gestion de ressources comme vu en système d'exploitation.
- Est-ce la faute de la méthode ?

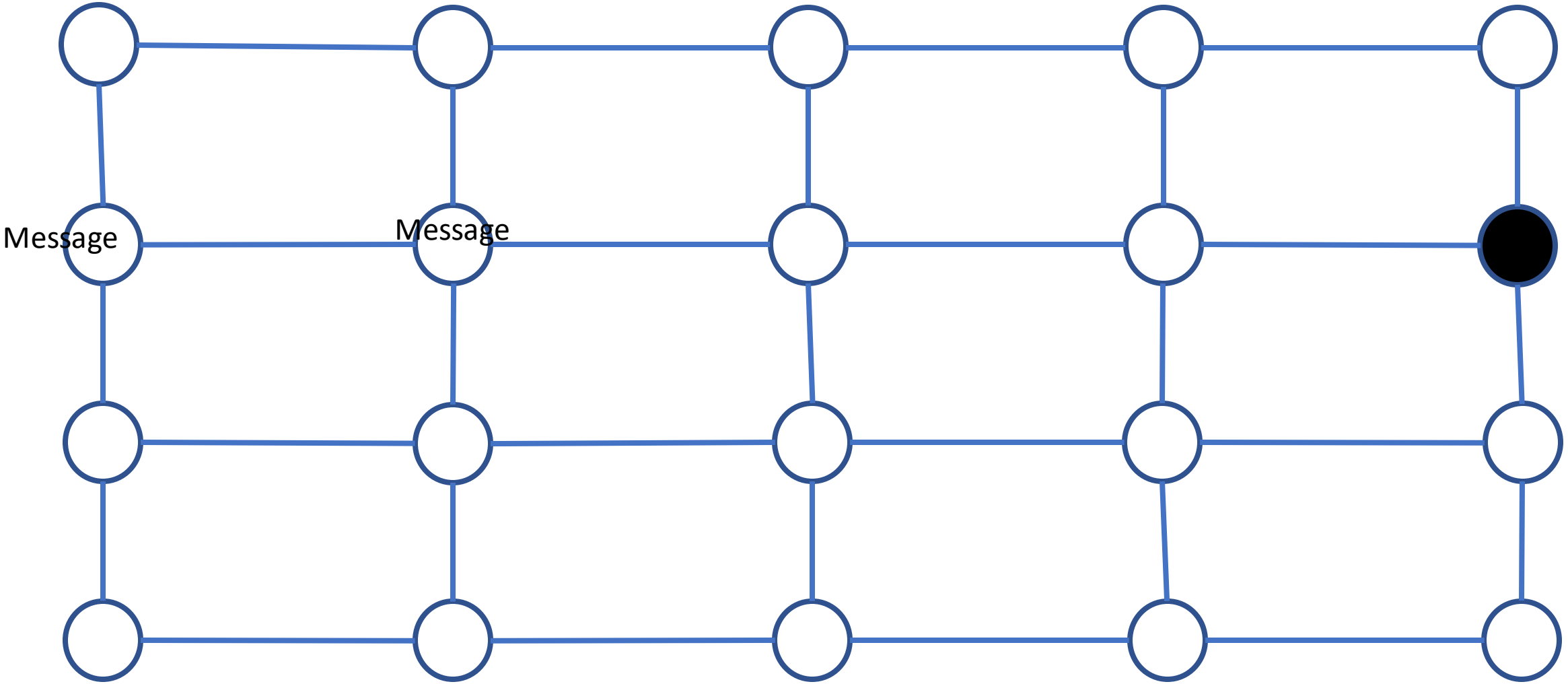
Commutation Store-and-forward



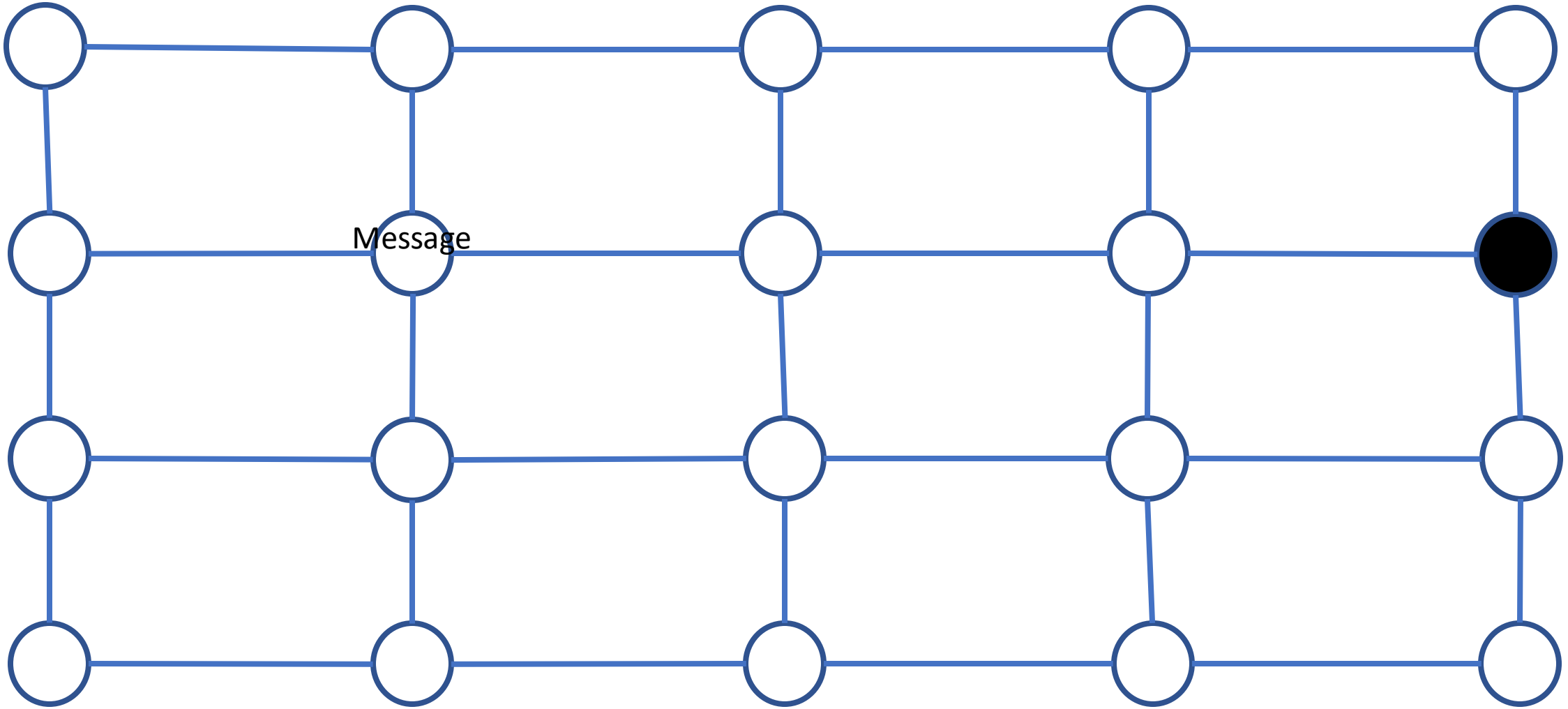
Comment ça marche



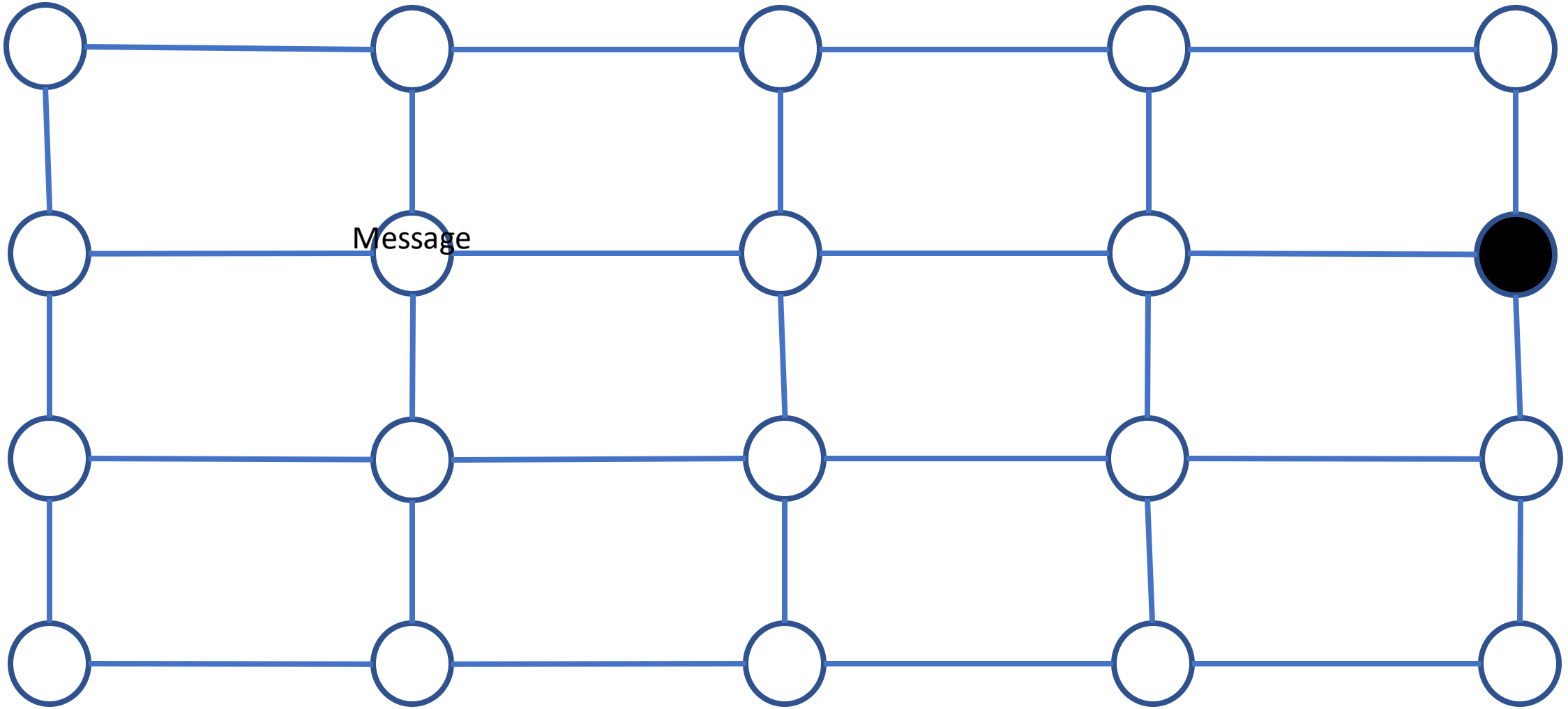
Comment ça marche



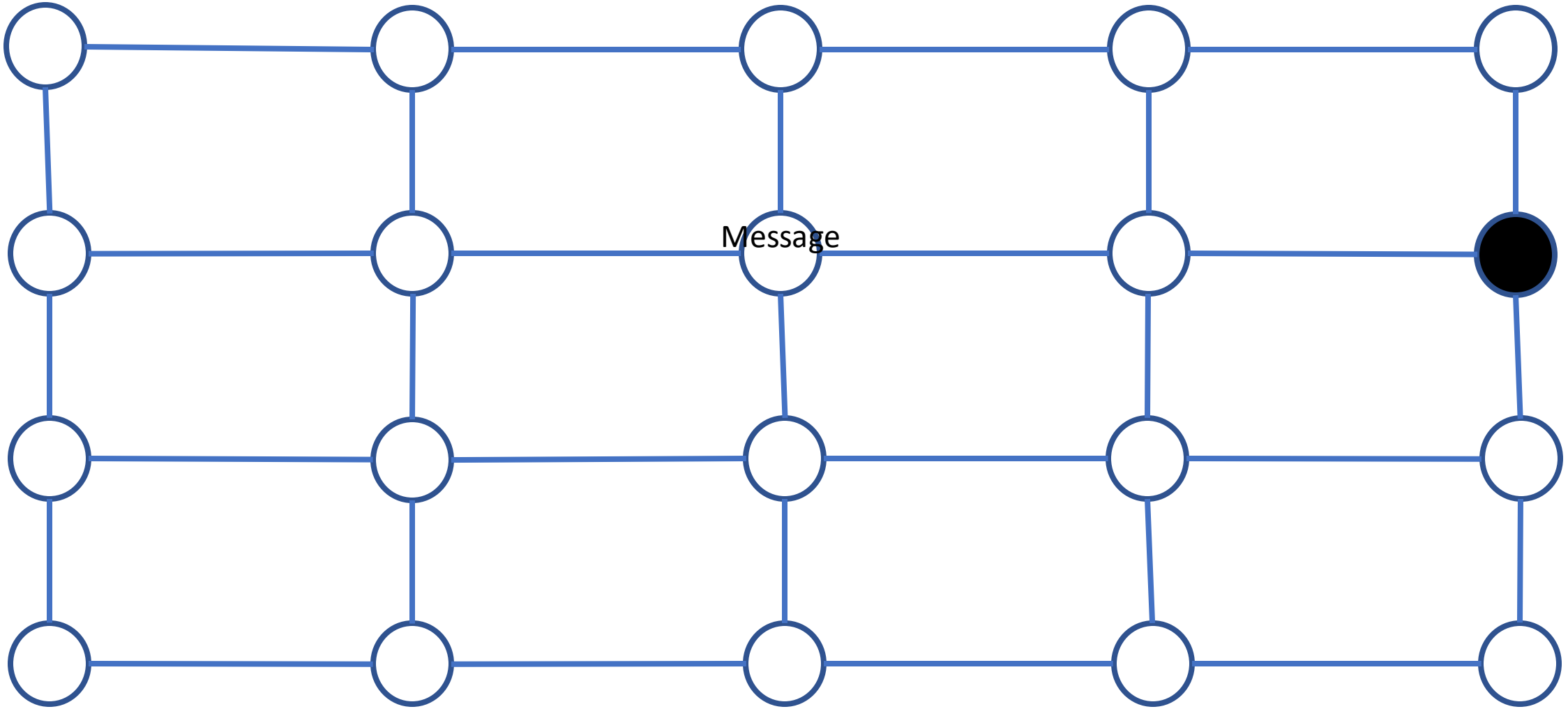
Comment ça marche



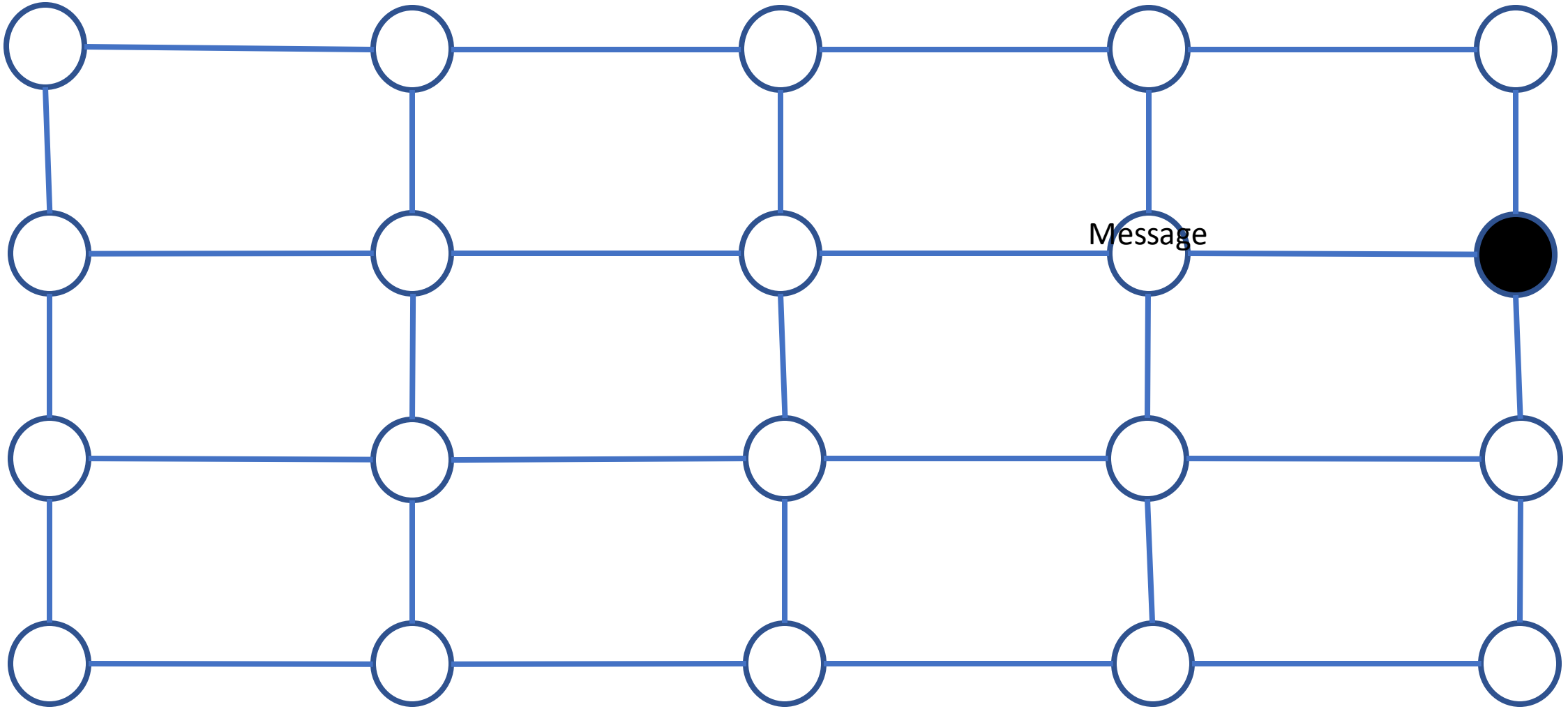
Comment ça marche



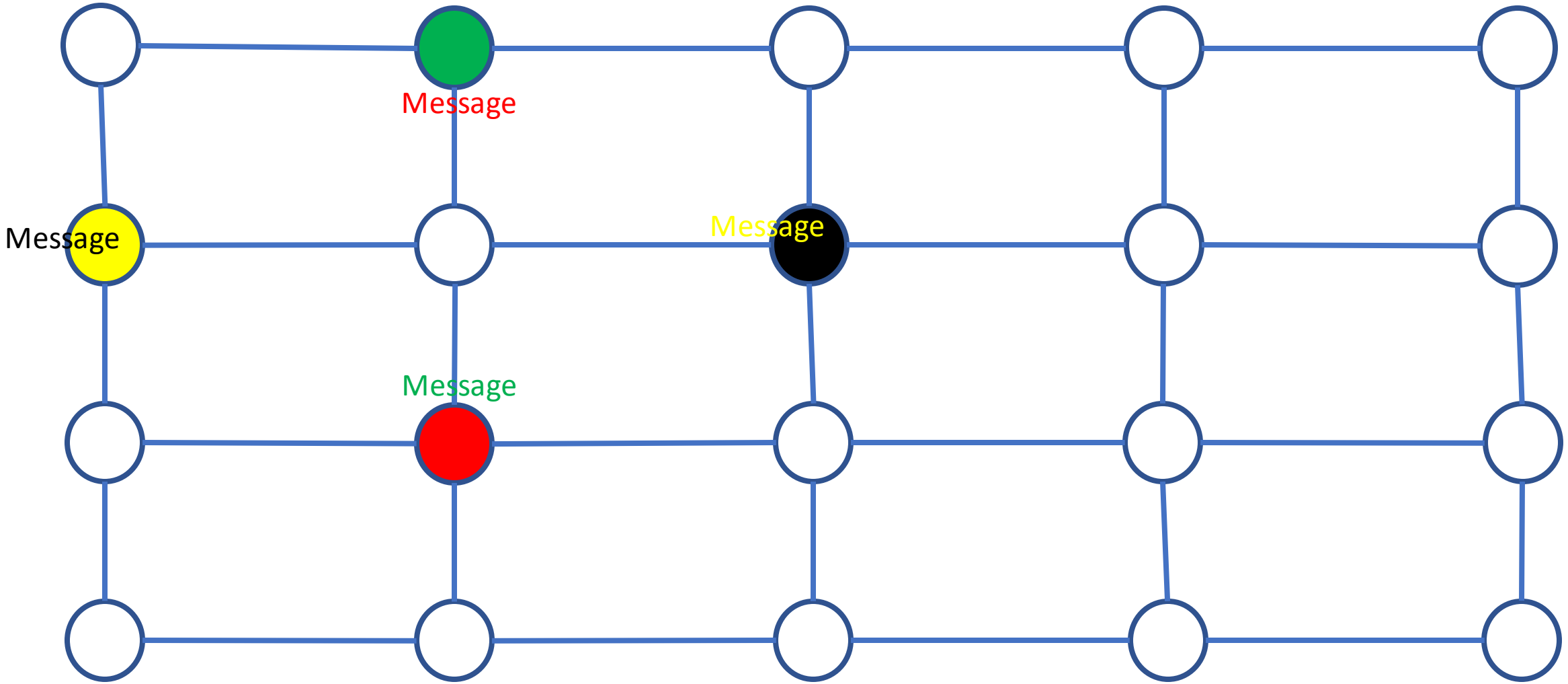
Comment ça marche



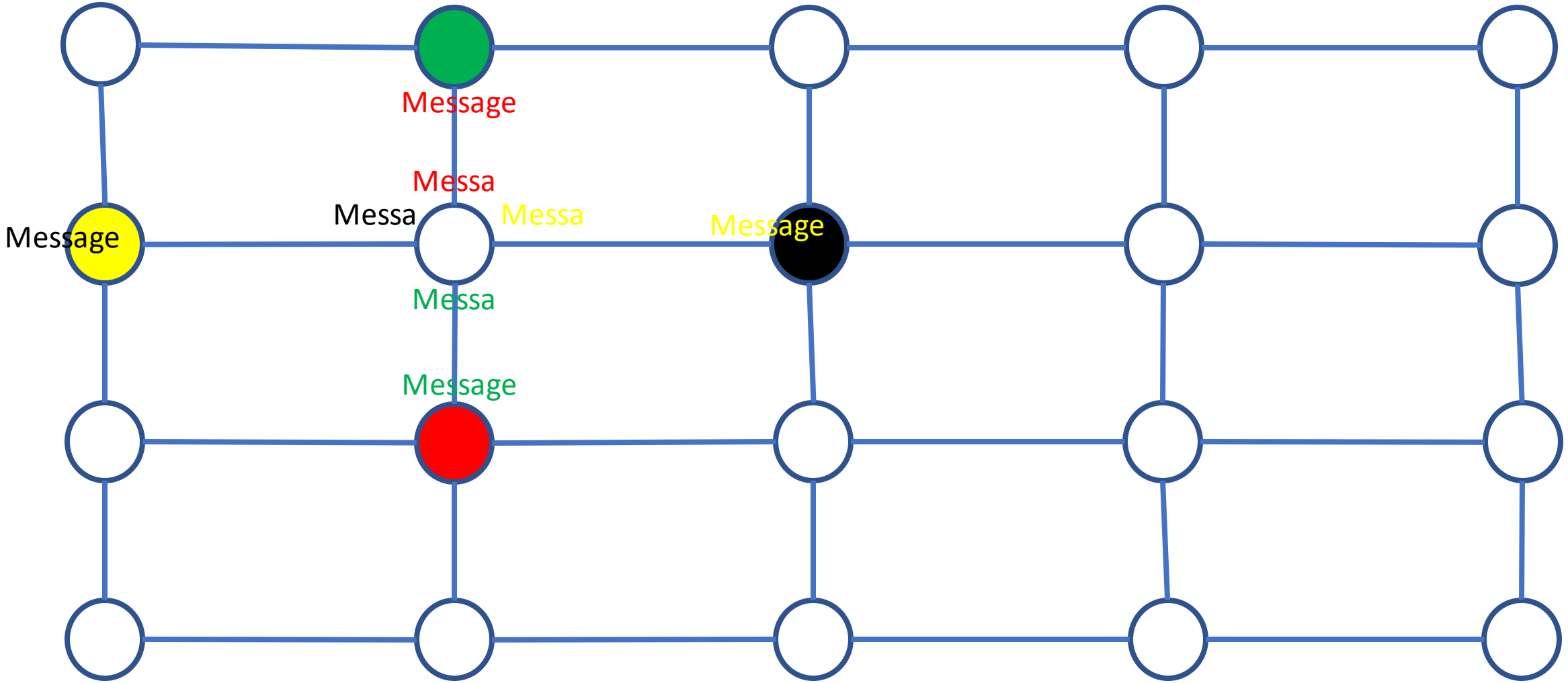
Comment ça marche



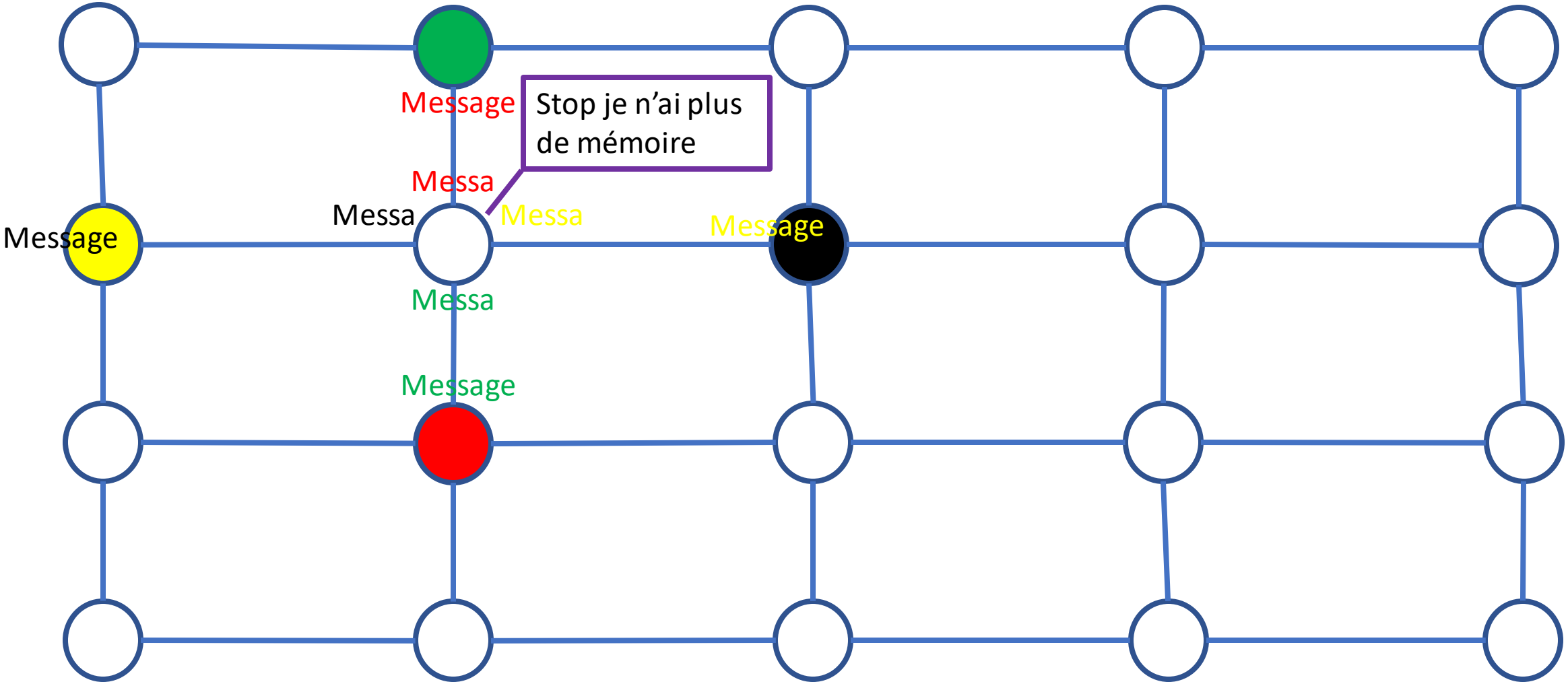
Comment ça marche



Comment ça marche



Comment ça marche pas !

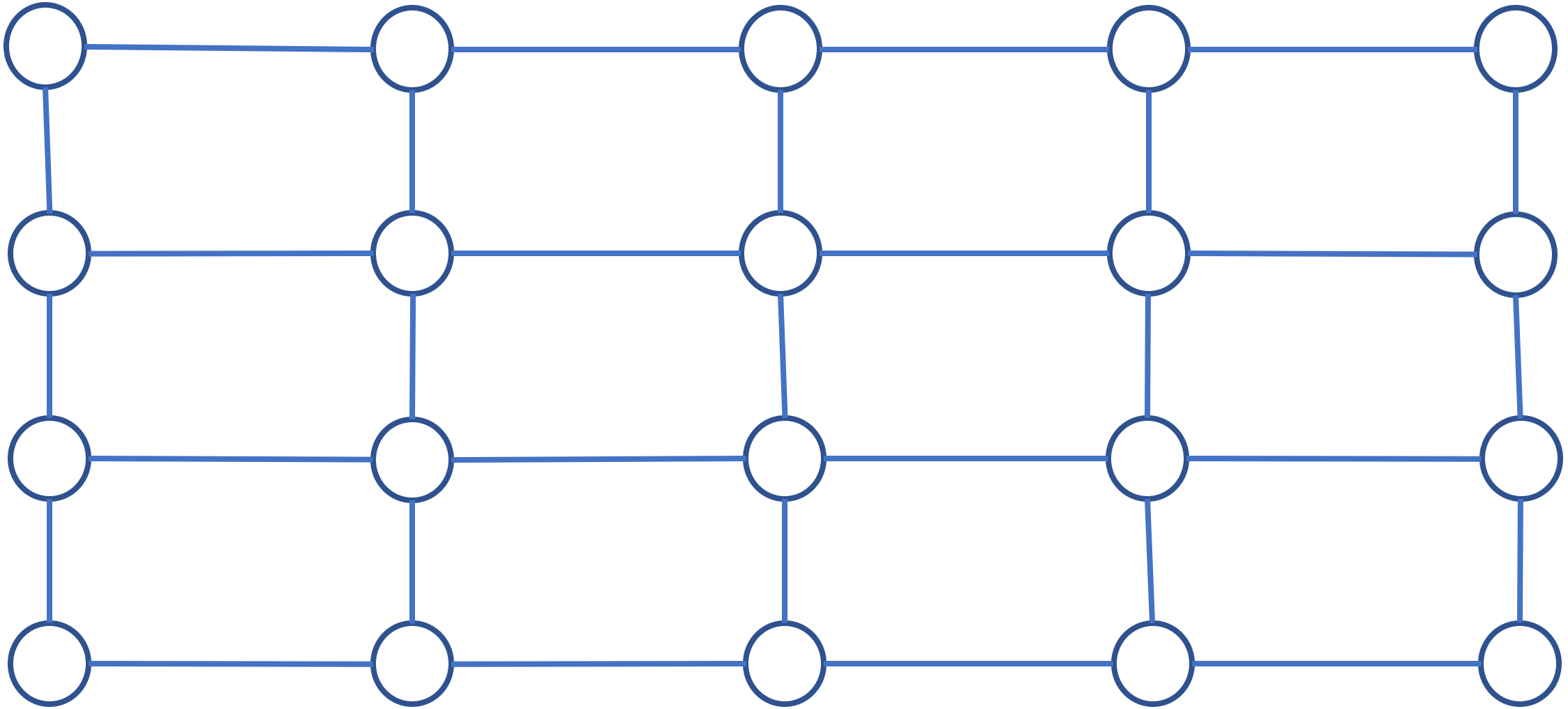


Pourquoi ?

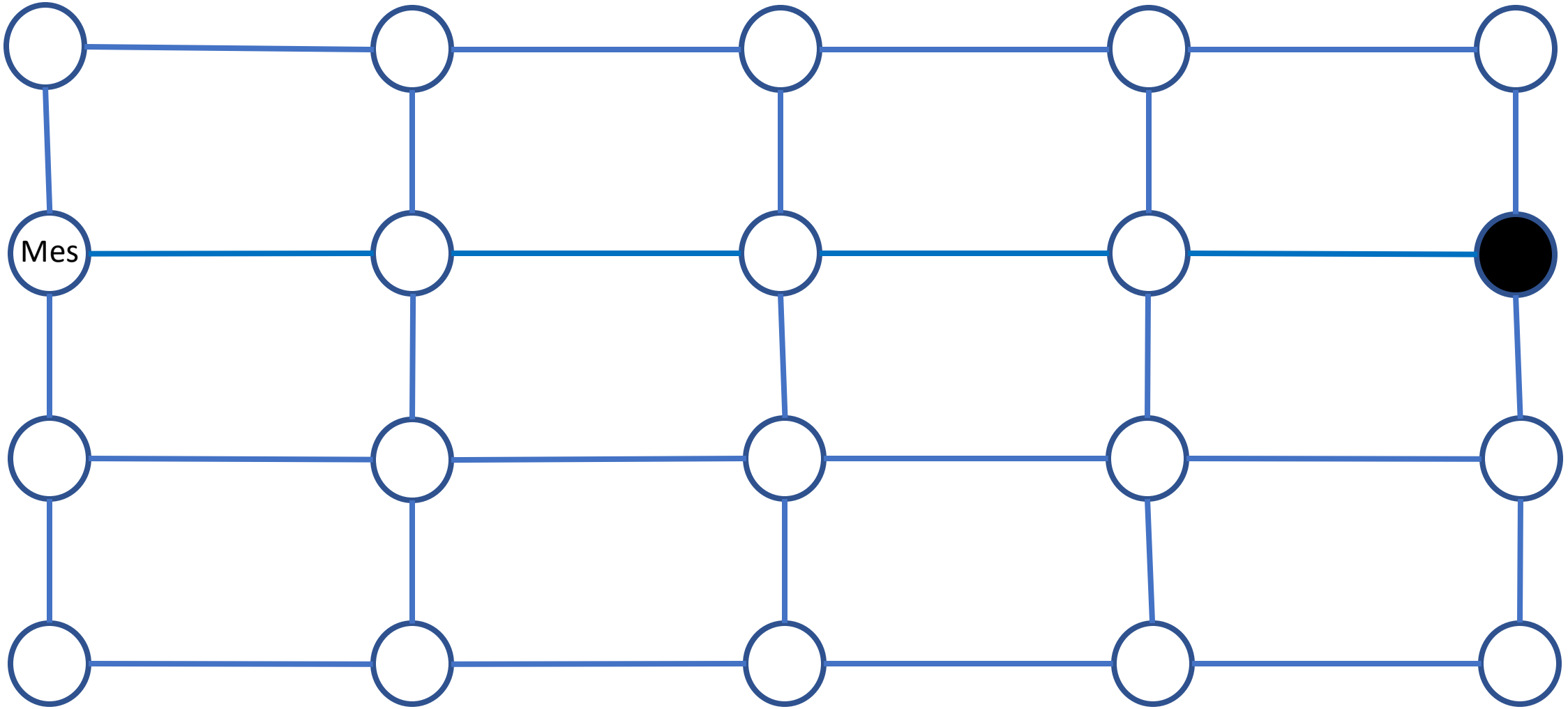
- M monopolise 5 unités de mémoire sur le nœud (1,2)
- M monopolise 5 unités de mémoire sur le nœud (1,2)
- M monopolise 5 unités de mémoire sur le nœud (1,2)
- M monopolise 5 unités de mémoire sur le nœud (1,2)
- Si le nœud (1,2) n'a que 20 unités de mémoire, BLOCAGE !

- C'est un bête problème de gestion de ressources comme vu en système d'exploitation.
- Est-ce la faute de la méthode ?

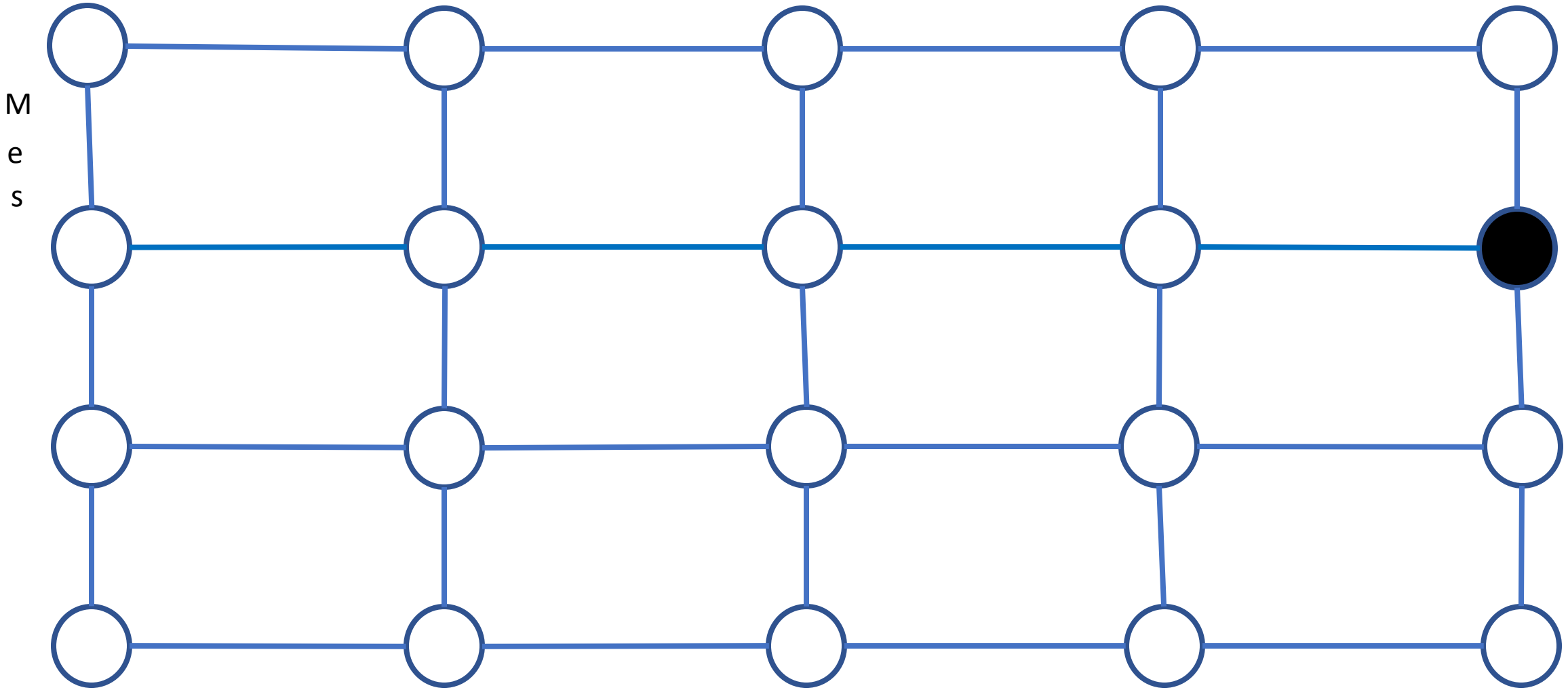
Commutation de paquets



Comment ça marche

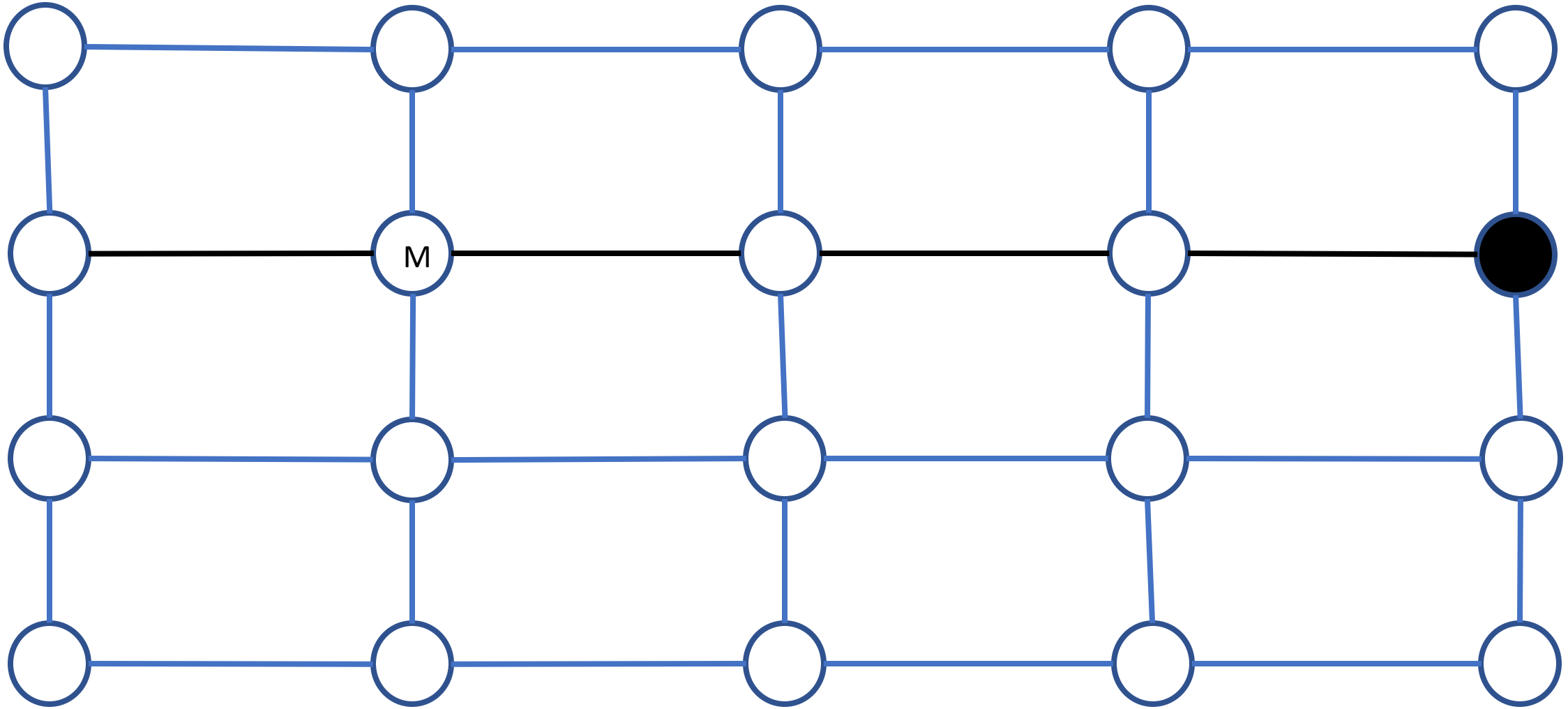


Comment ça marche

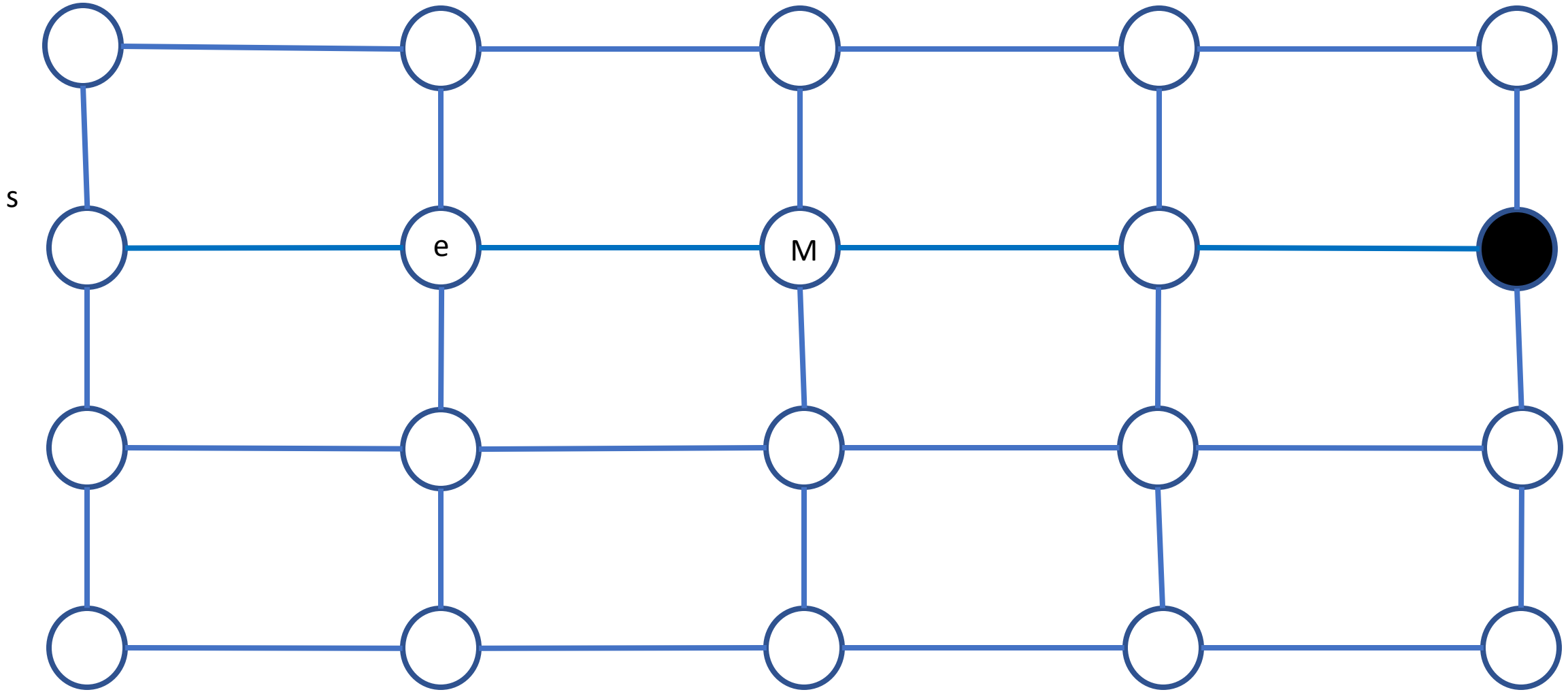


Comment ça marche

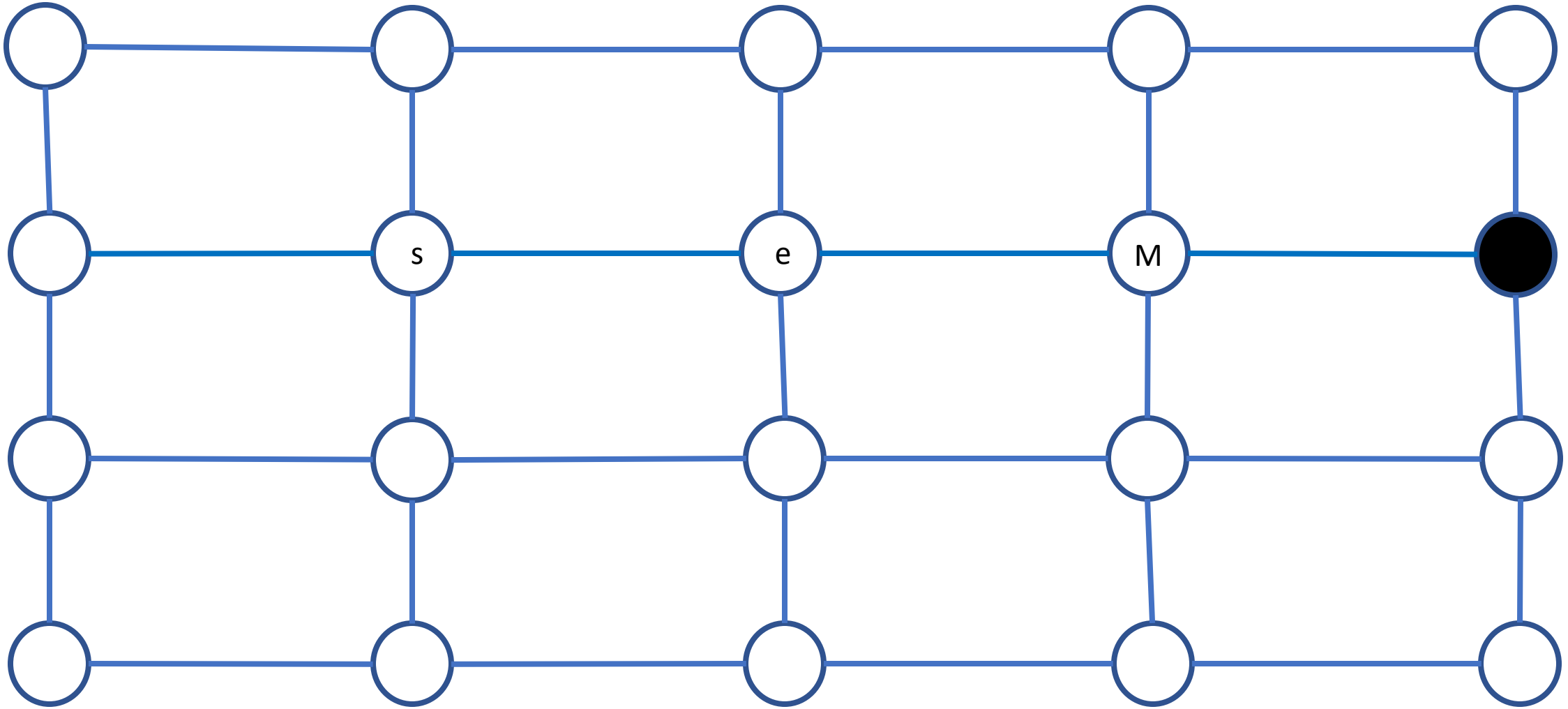
e
s



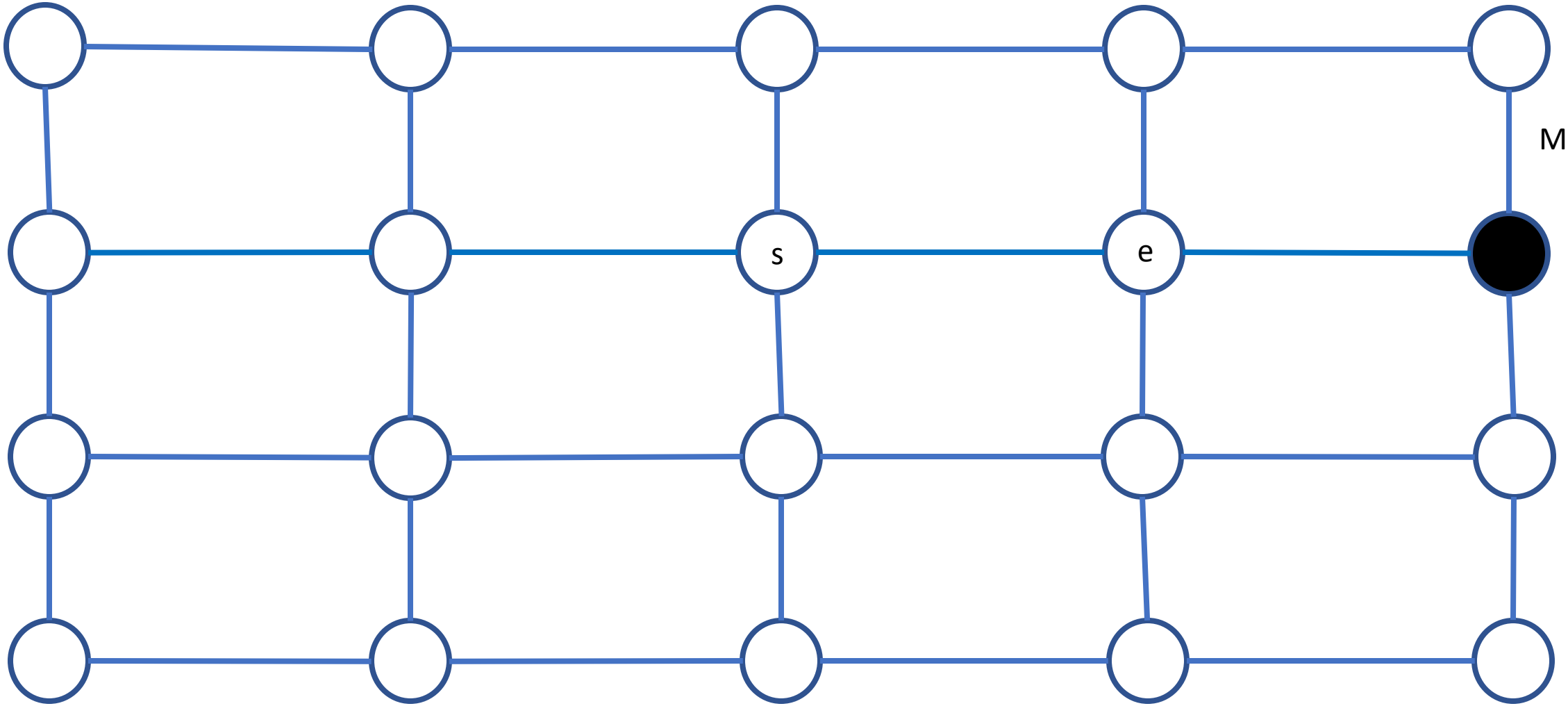
Comment ça marche



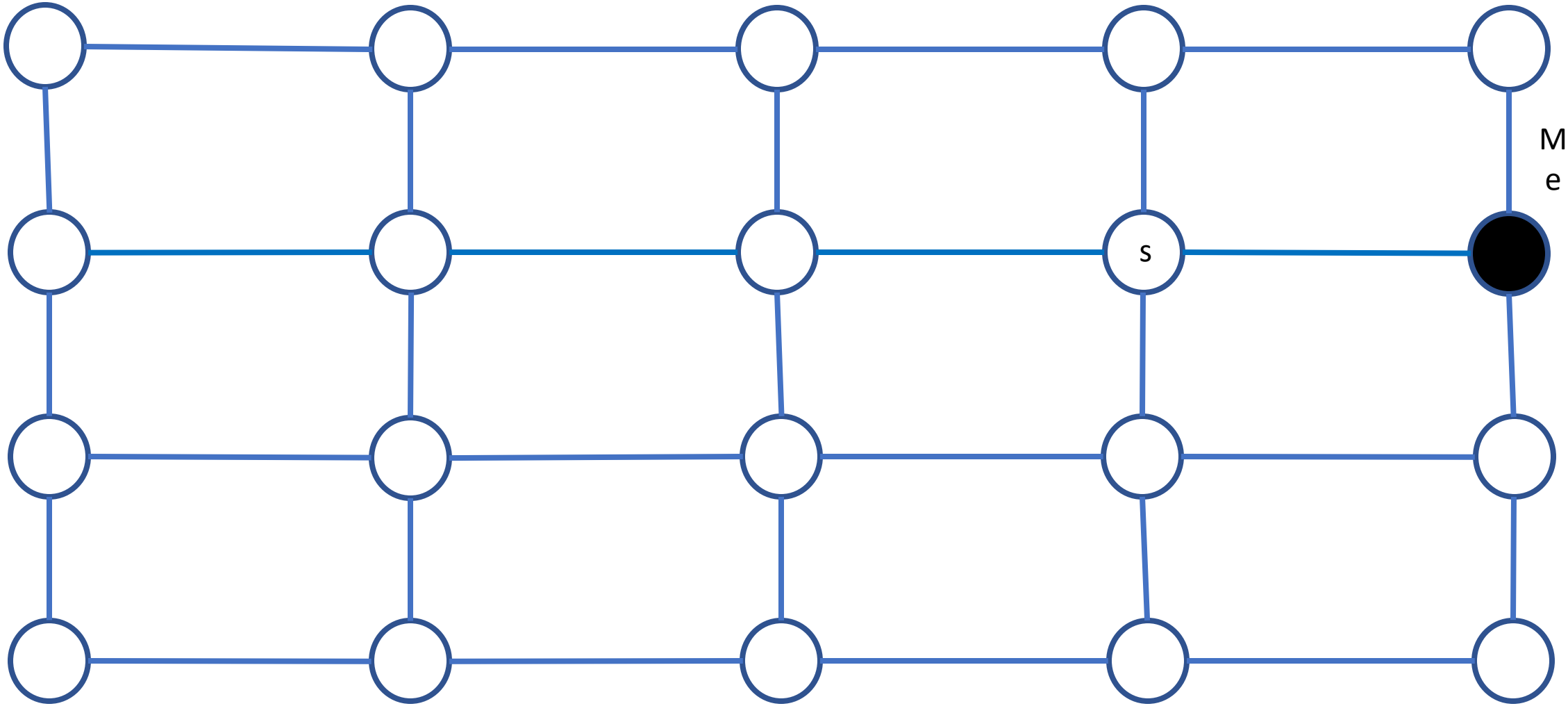
Comment ça marche



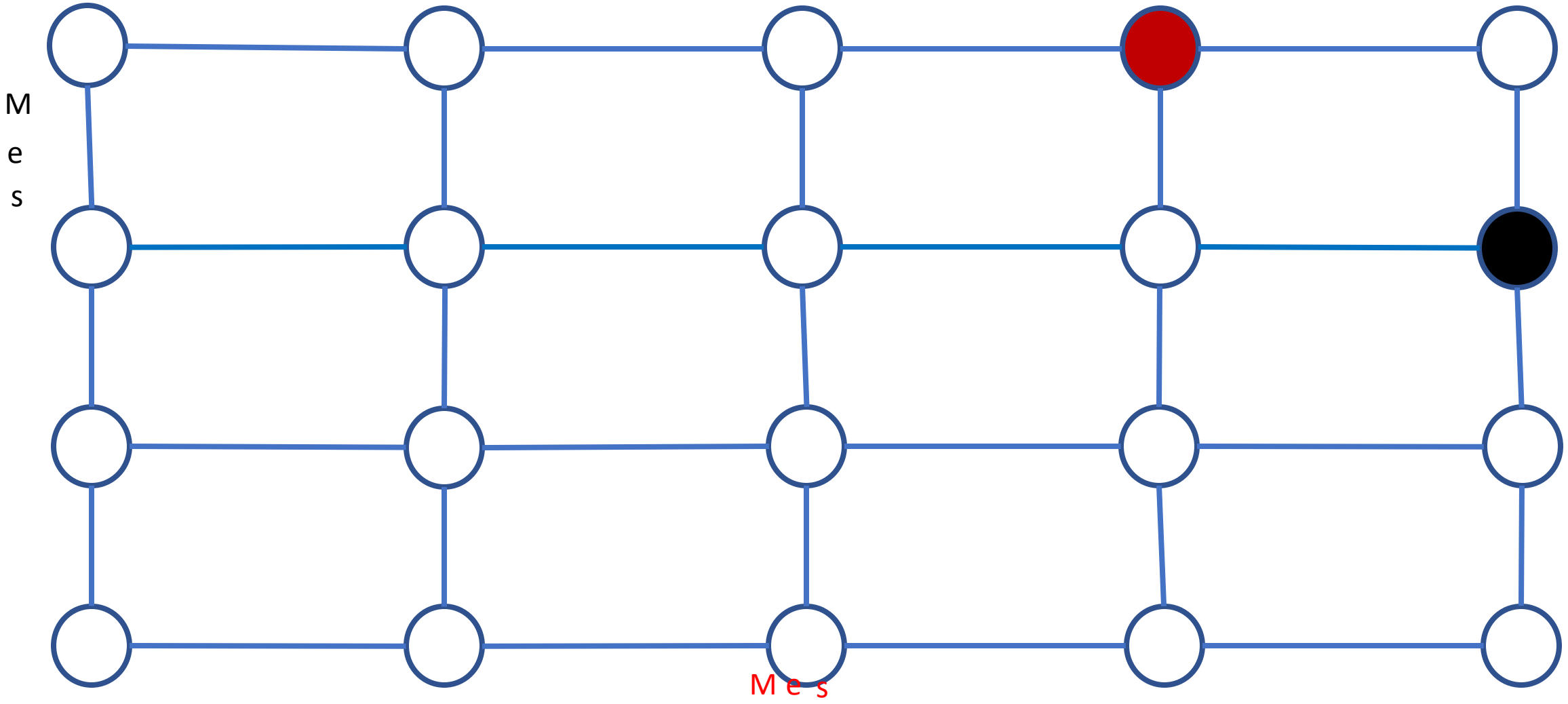
Comment ça marche



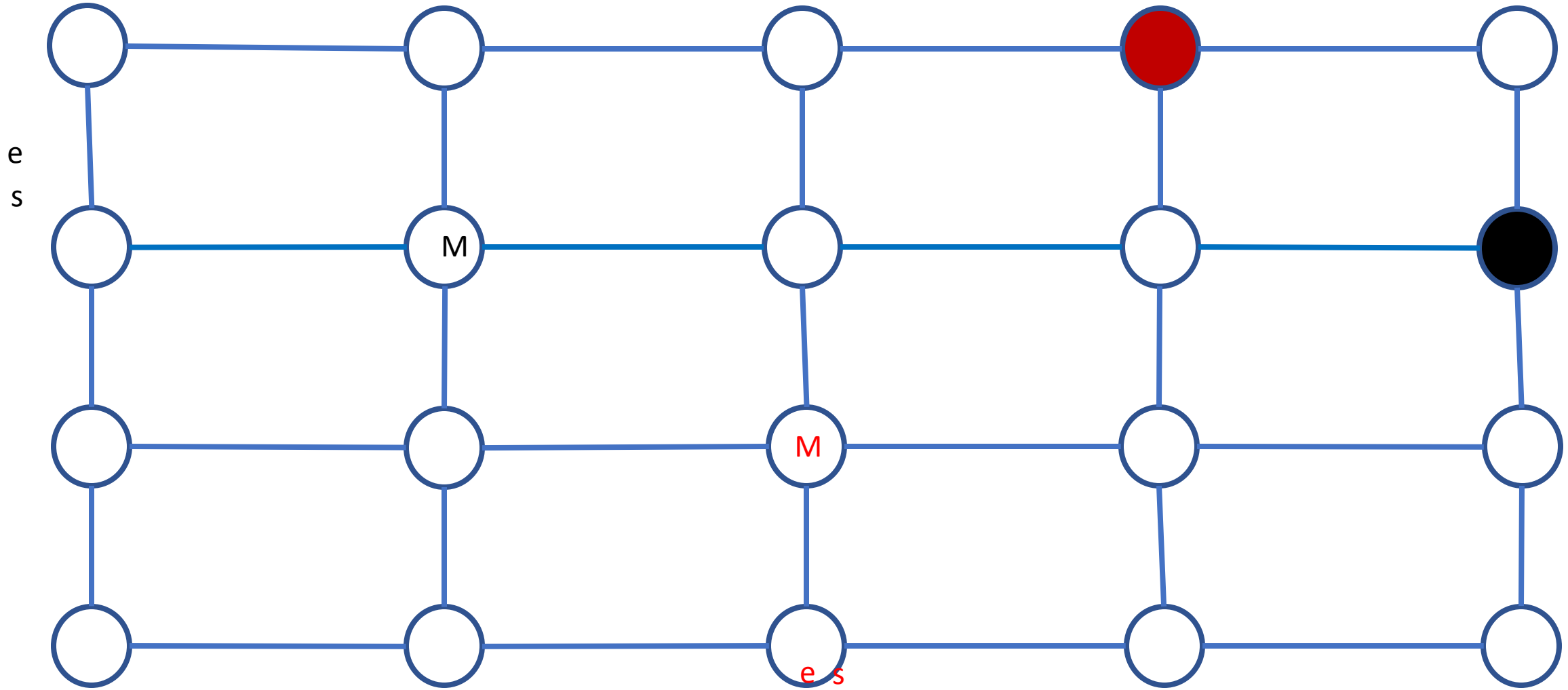
Comment ça marche



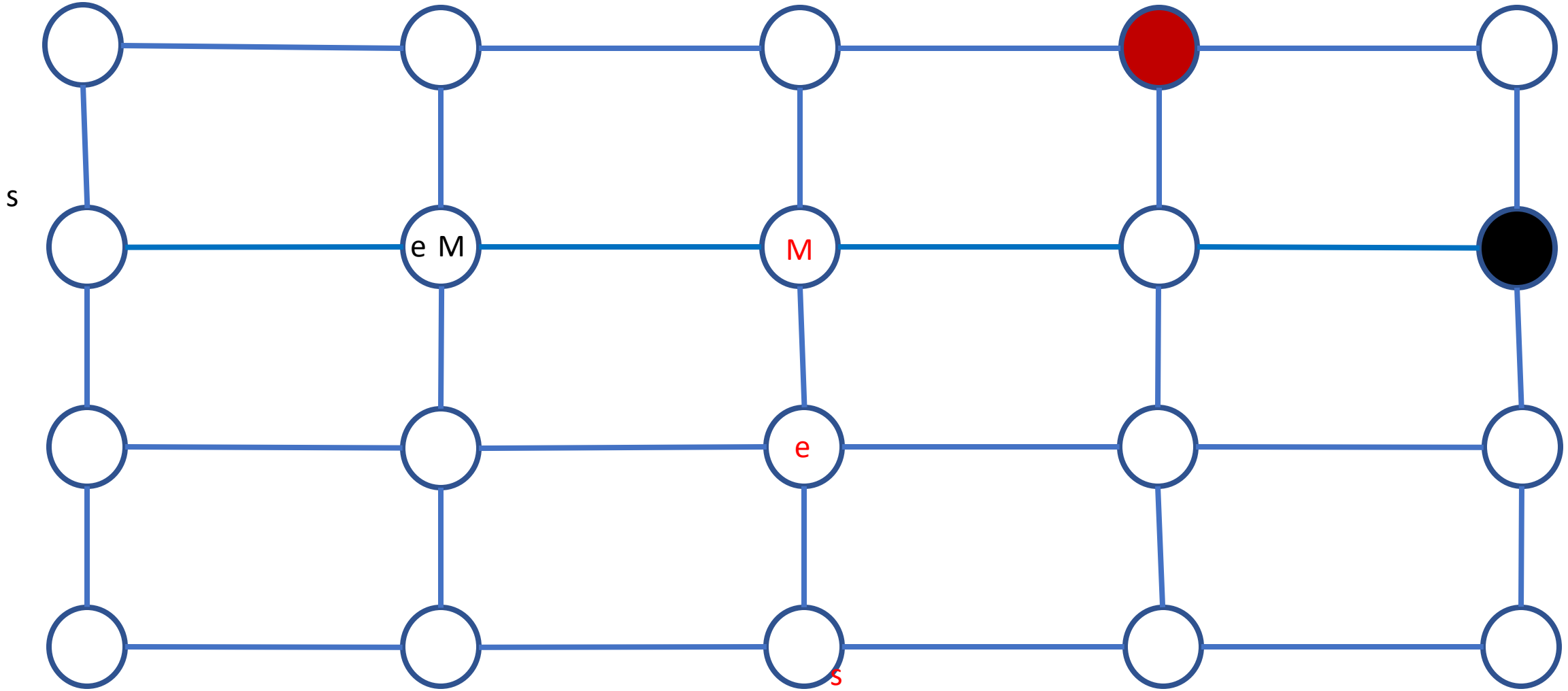
Comment ça marche



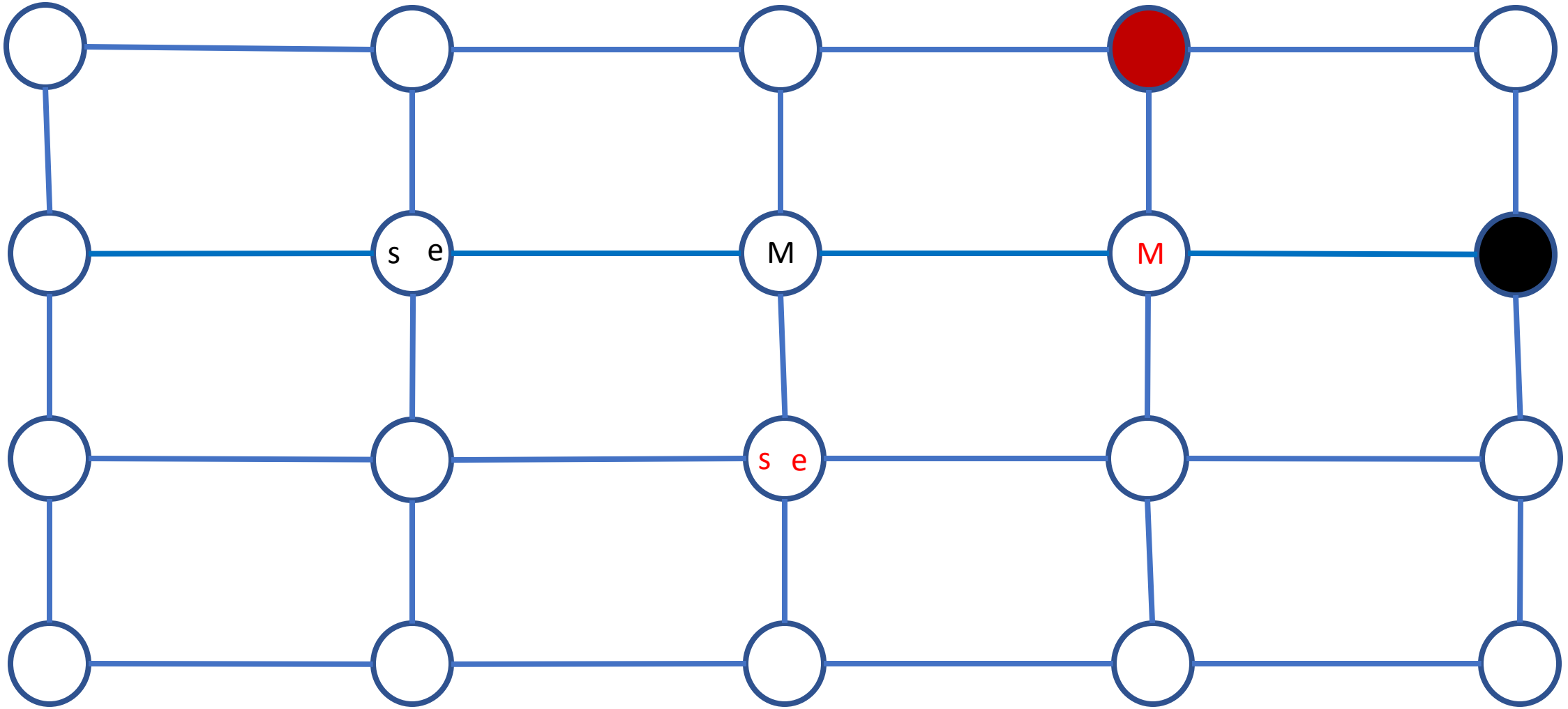
Comment ça marche



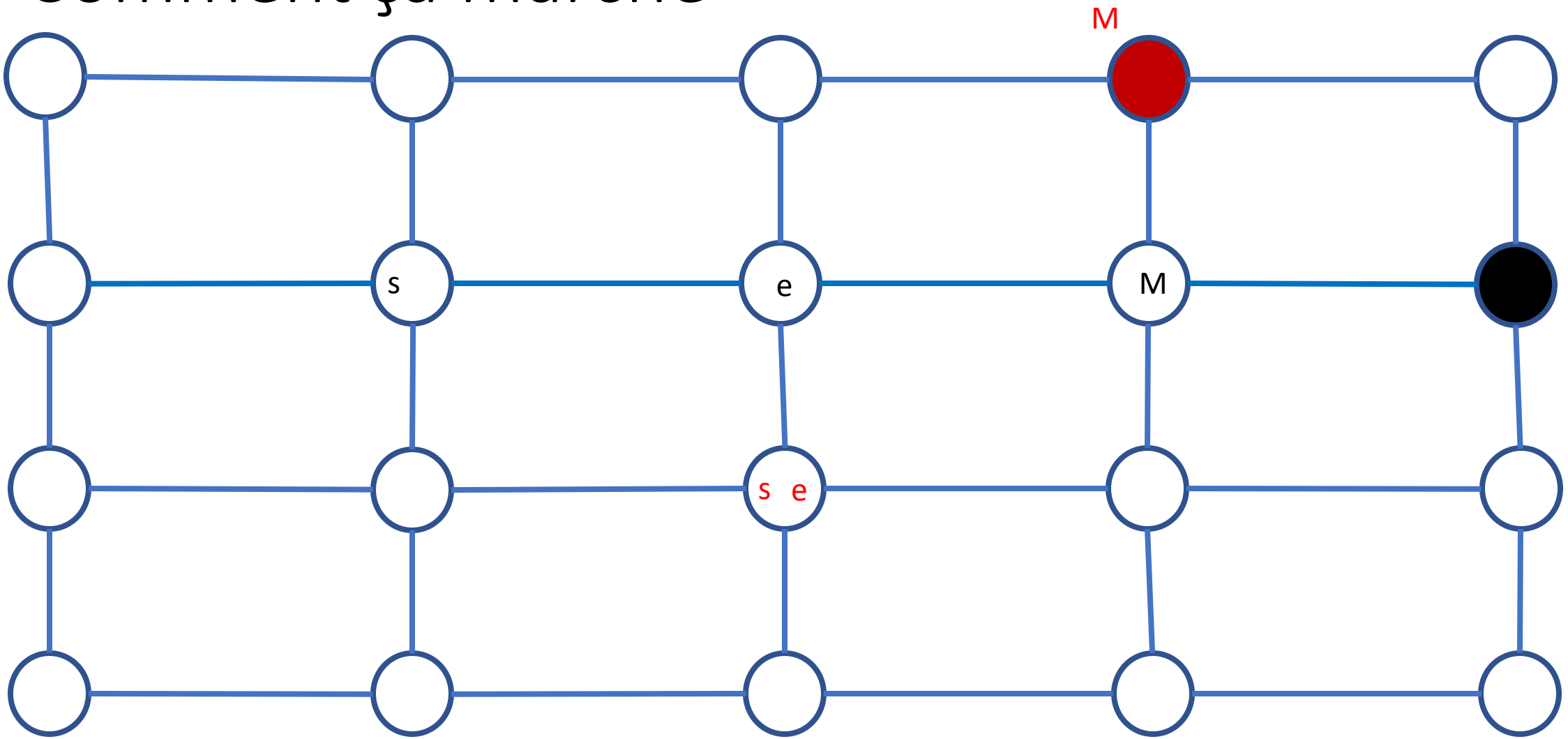
Comment ça marche



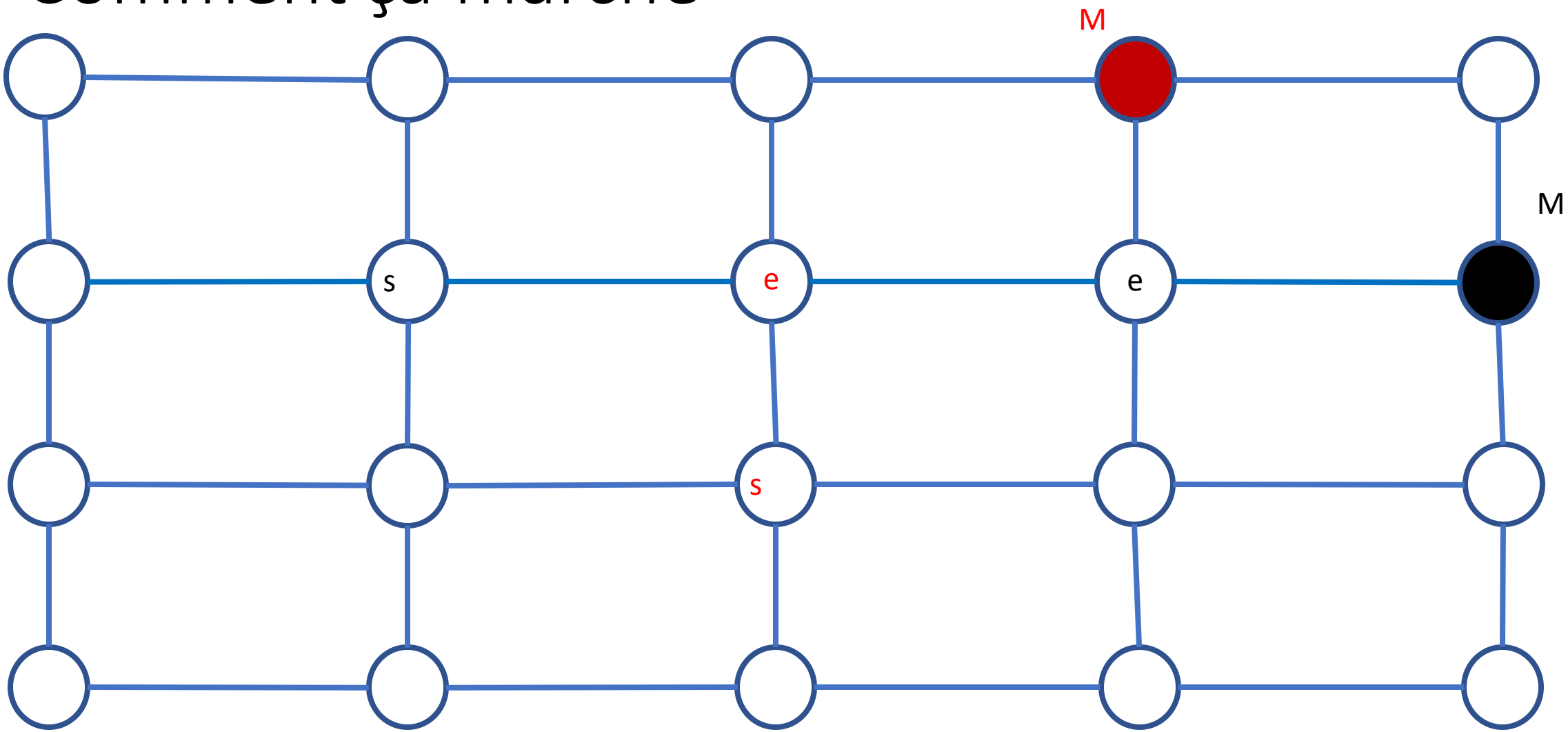
Comment ça marche



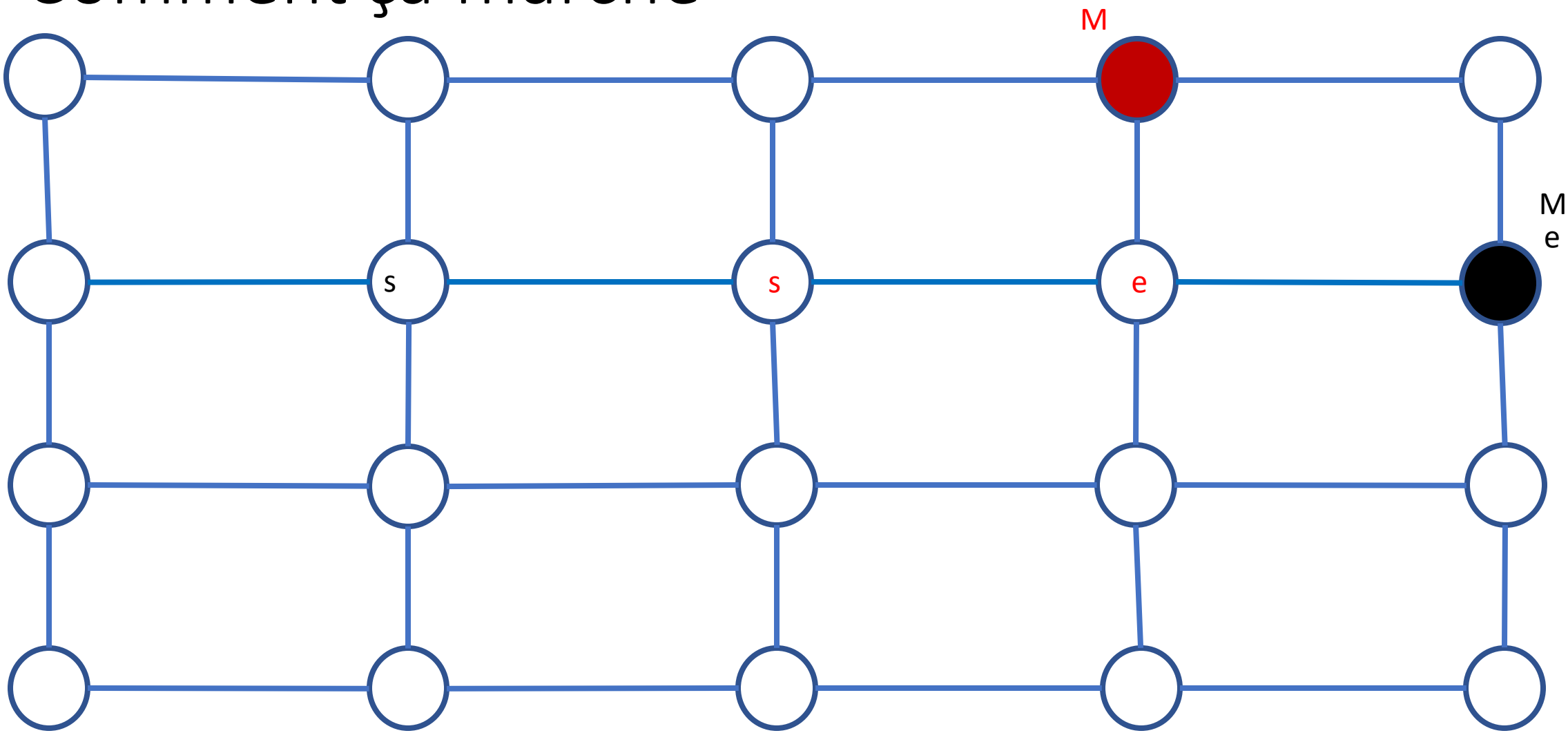
Comment ça marche



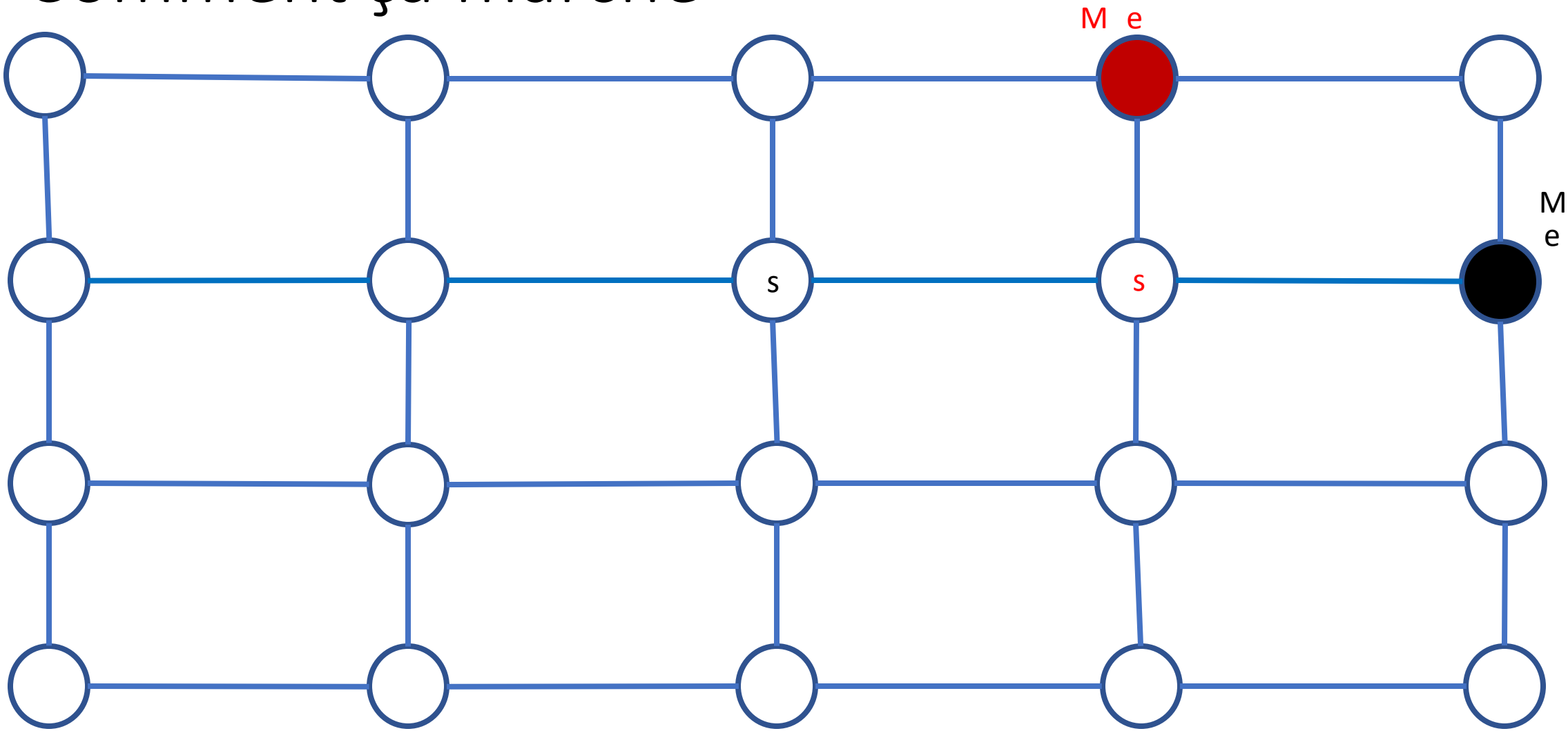
Comment ça marche



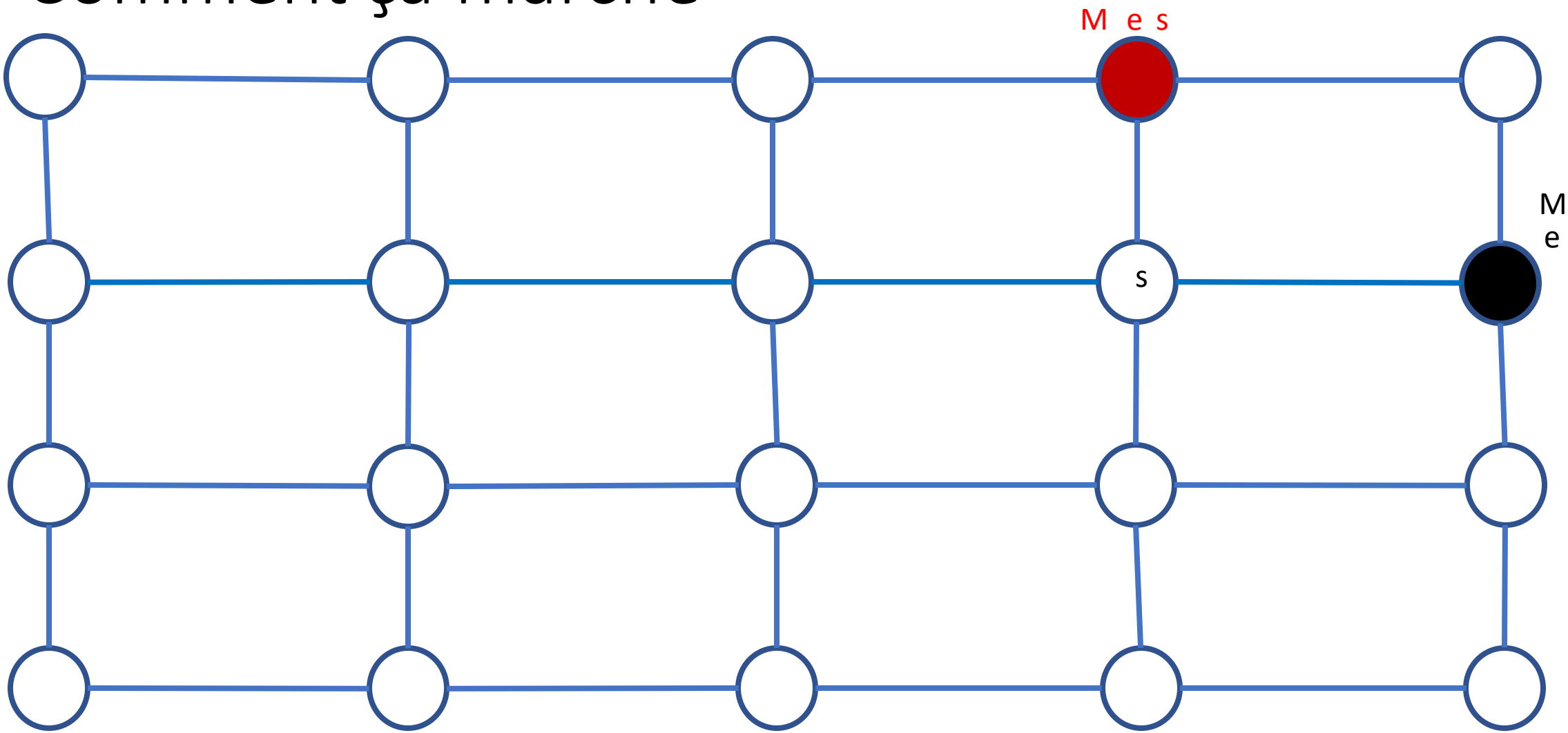
Comment ça marche



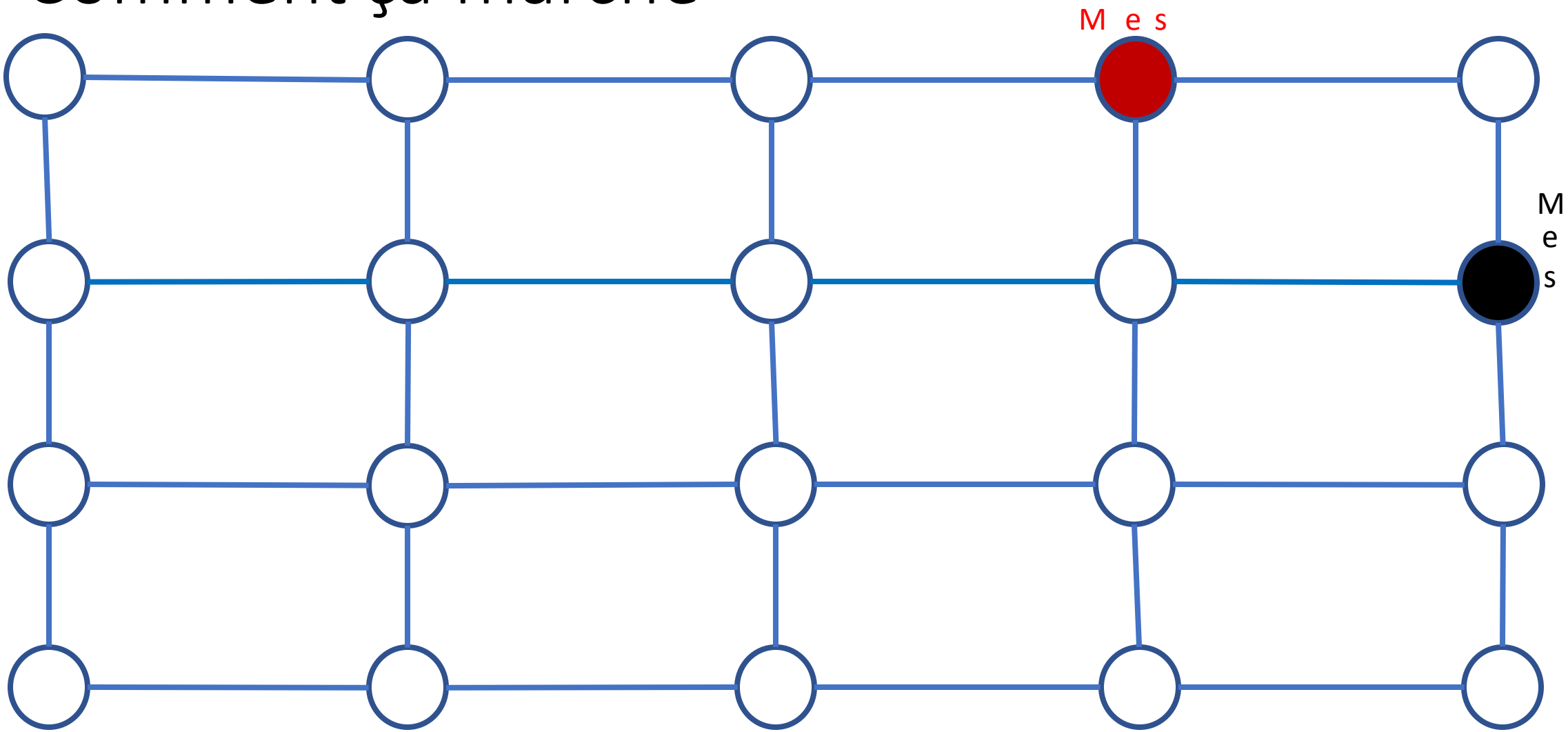
Comment ça marche



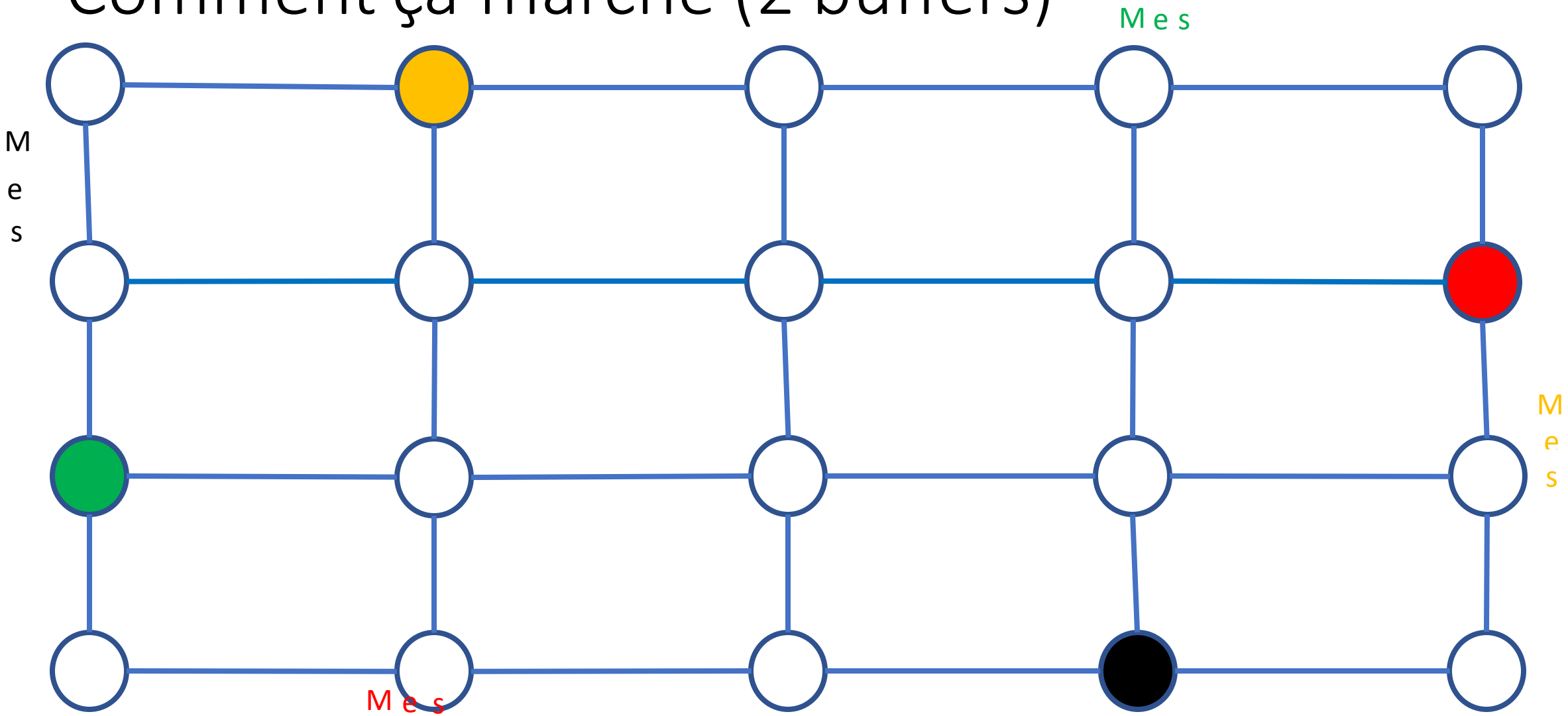
Comment ça marche



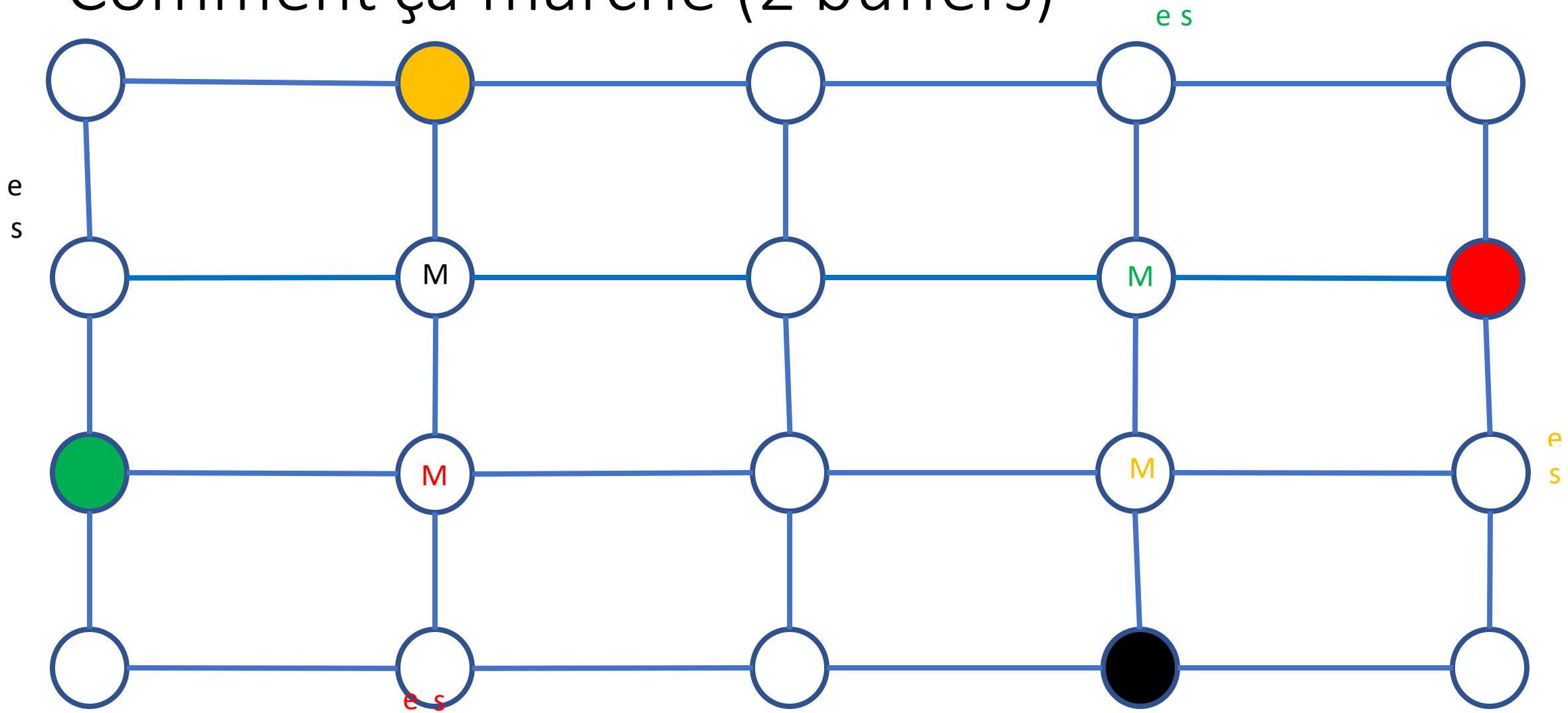
Comment ça marche



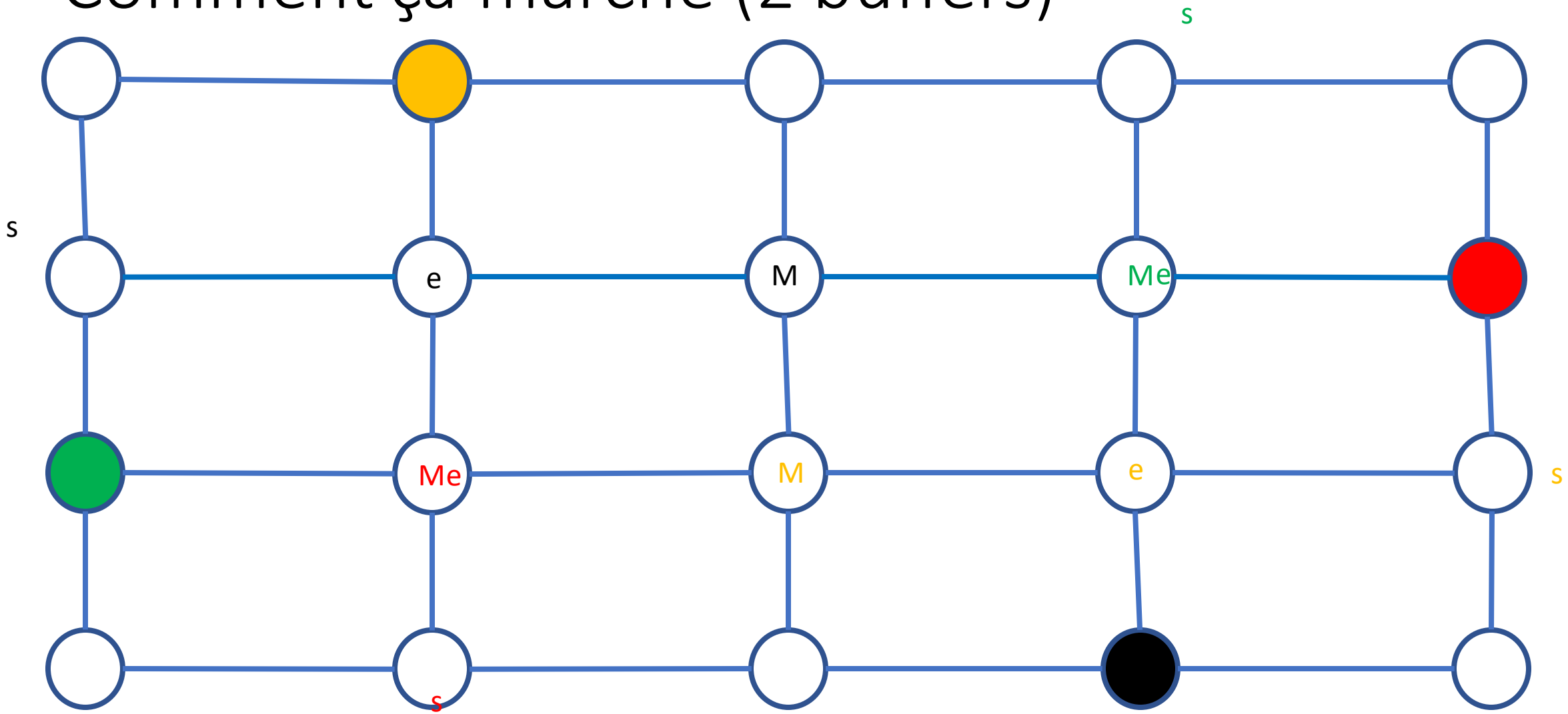
Comment ça marche (2 buffers)



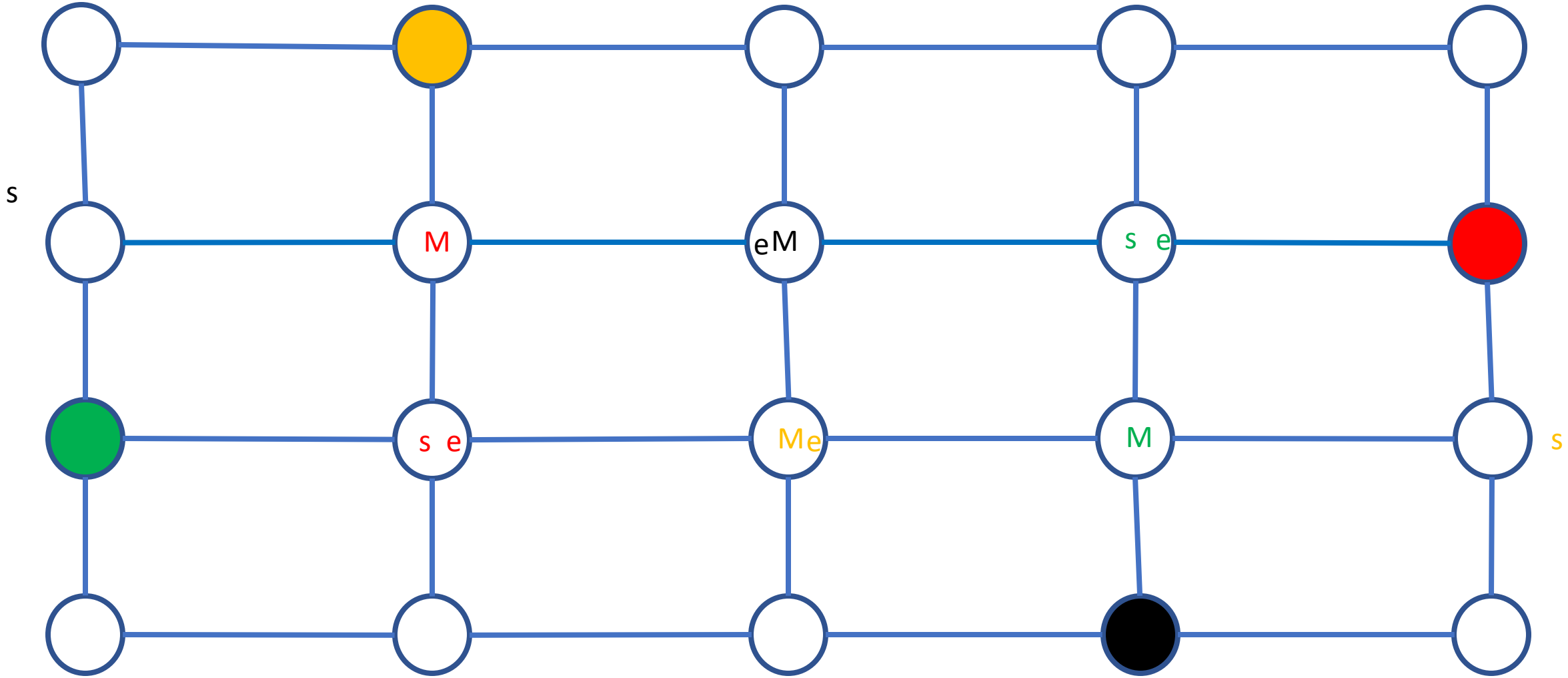
Comment ça marche (2 buffers)



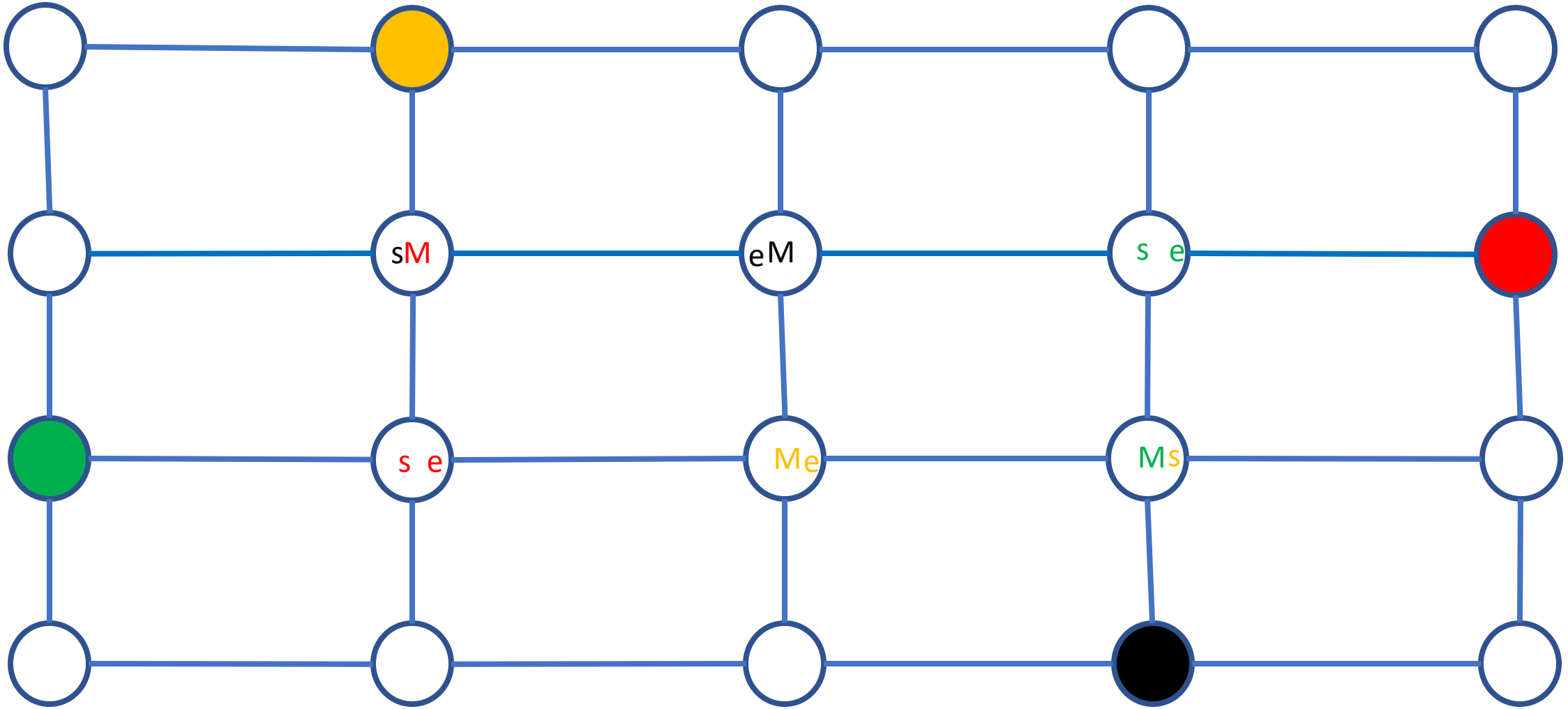
Comment ça marche (2 buffers)



Comment ça marche (2 buffers)



Comment ça marche Pas (2 buffers)

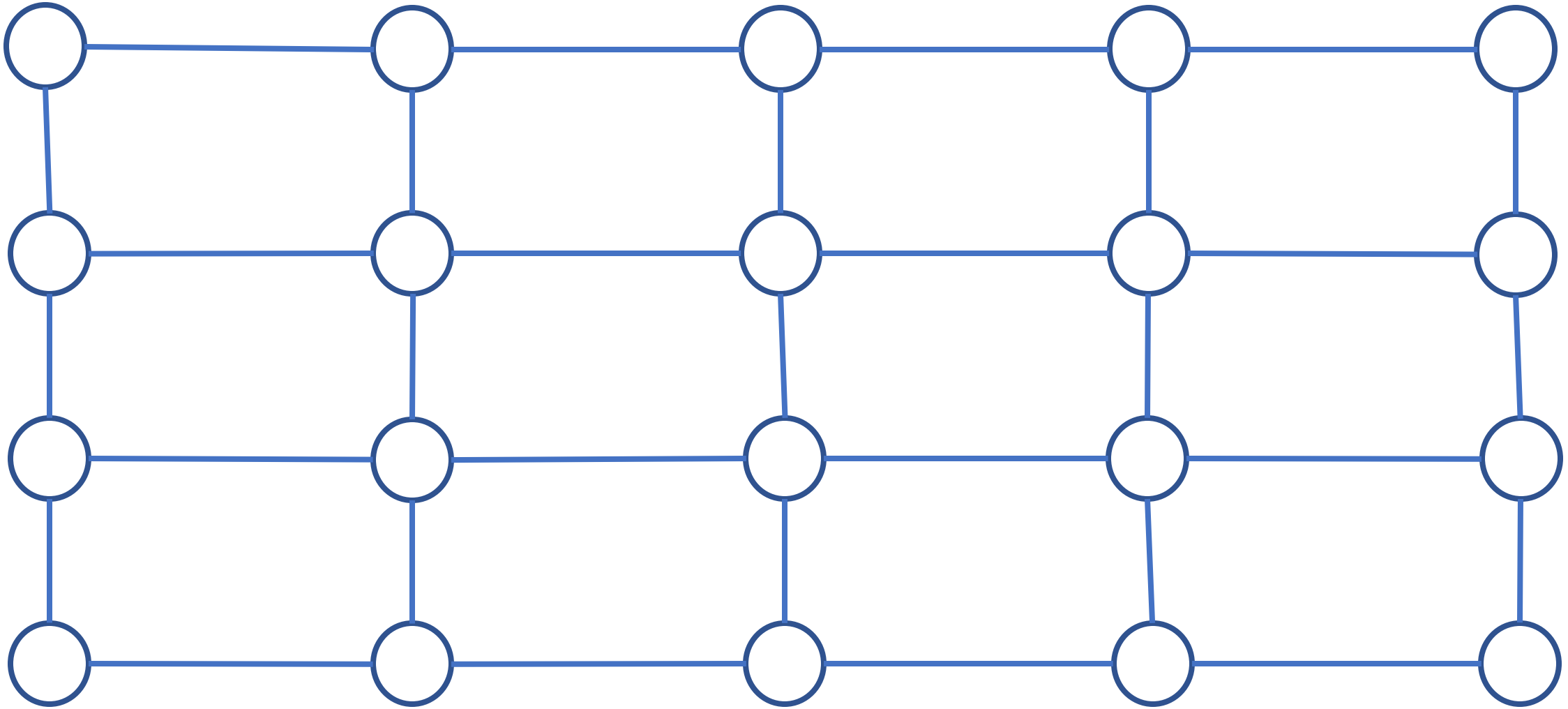


Pourquoi ?

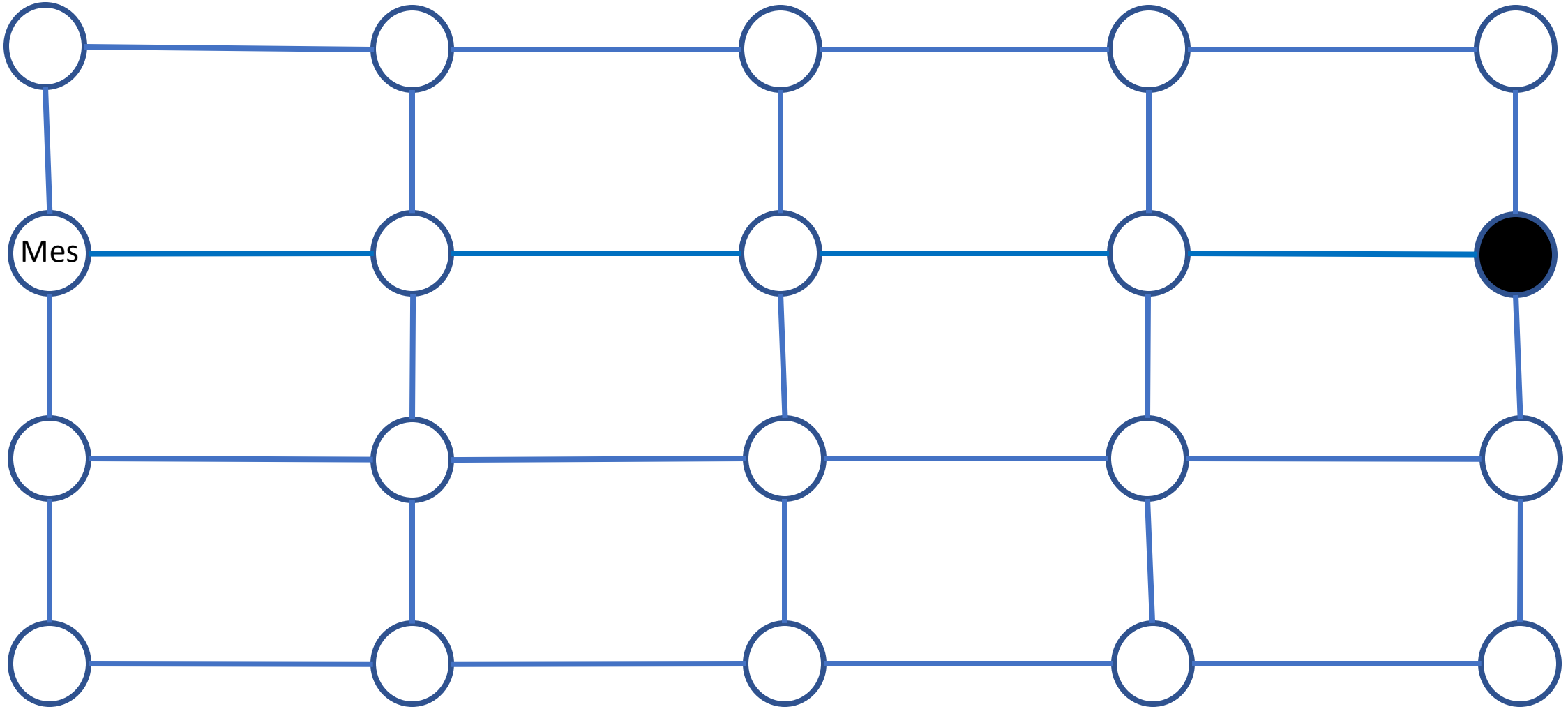
- M monopolise 2 buffers sur (2,2), 1 sur (1,2), en attend 1 sur (3,2)
- M monopolise 2 buffers sur (1,1), 1 sur (1,2), en attend 1 sur (2,2) ou 1 sur (1,2)
- M monopolise 2 buffers sur (3,2), 1 sur (3,1), en attend 1 sur (2,1) ou 1 sur (3,1)
- M monopolise 2 buffers sur (2,1), 1 sur (3,1), en attend 1 sur (1,1)

- C'est un bête problème de gestion de ressources comme vu en système d'exploitation.
- Est-ce la faute de la méthode ?

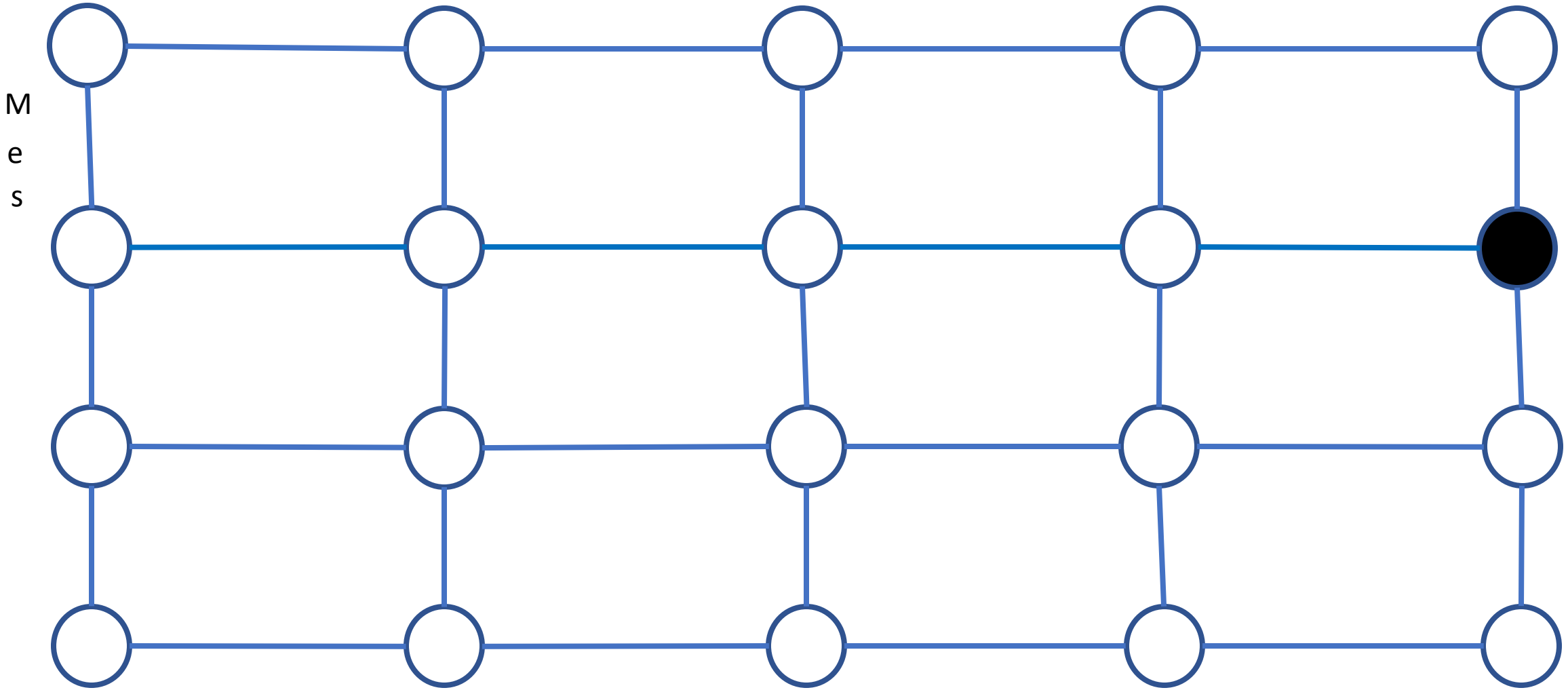
Commutation Wormhole



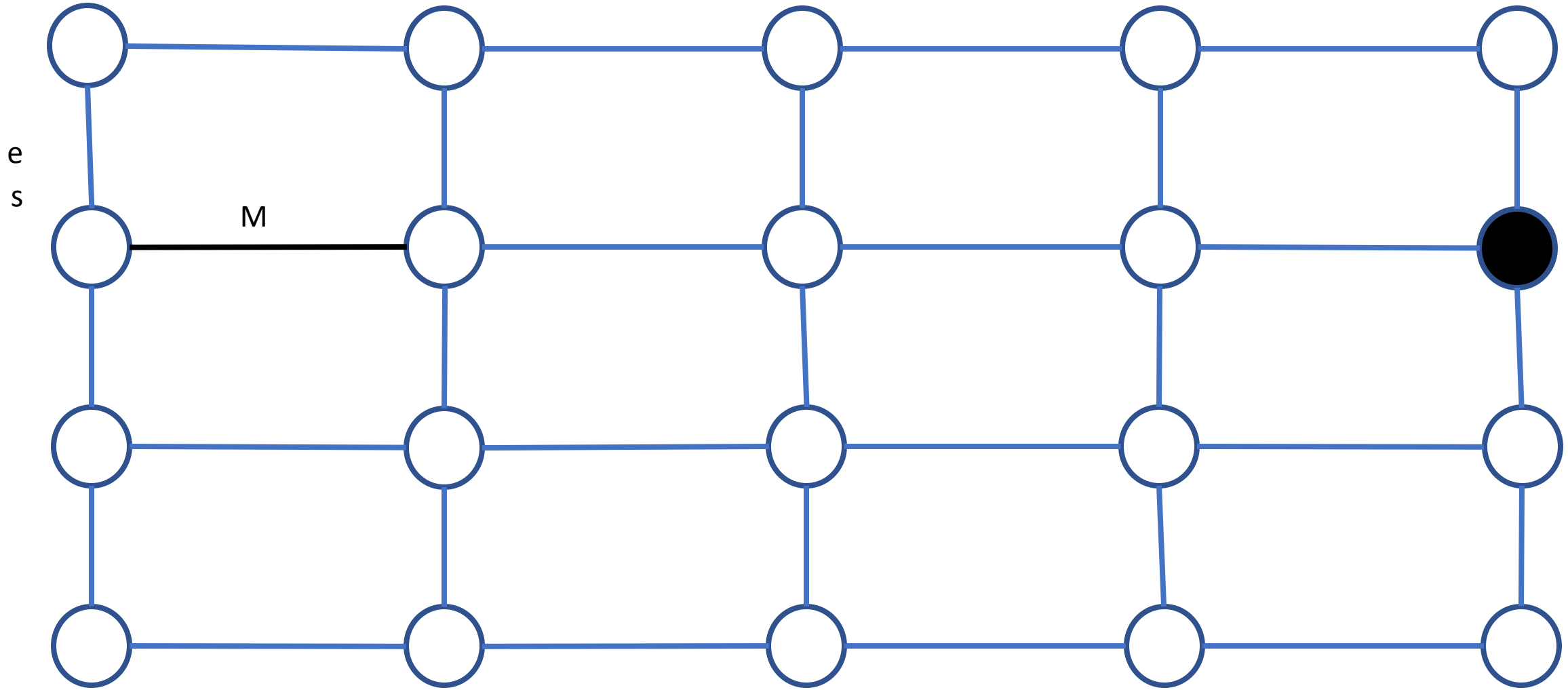
Comment ça marche



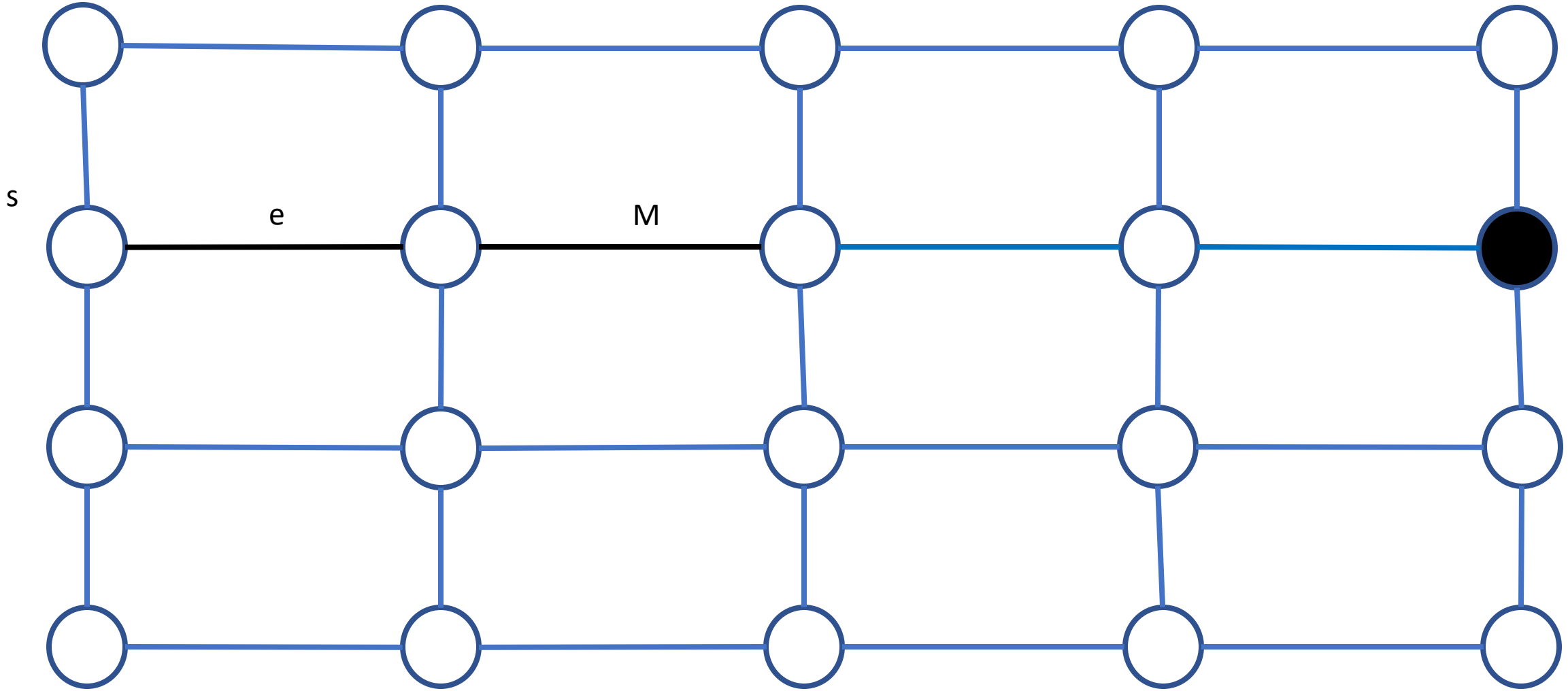
Comment ça marche



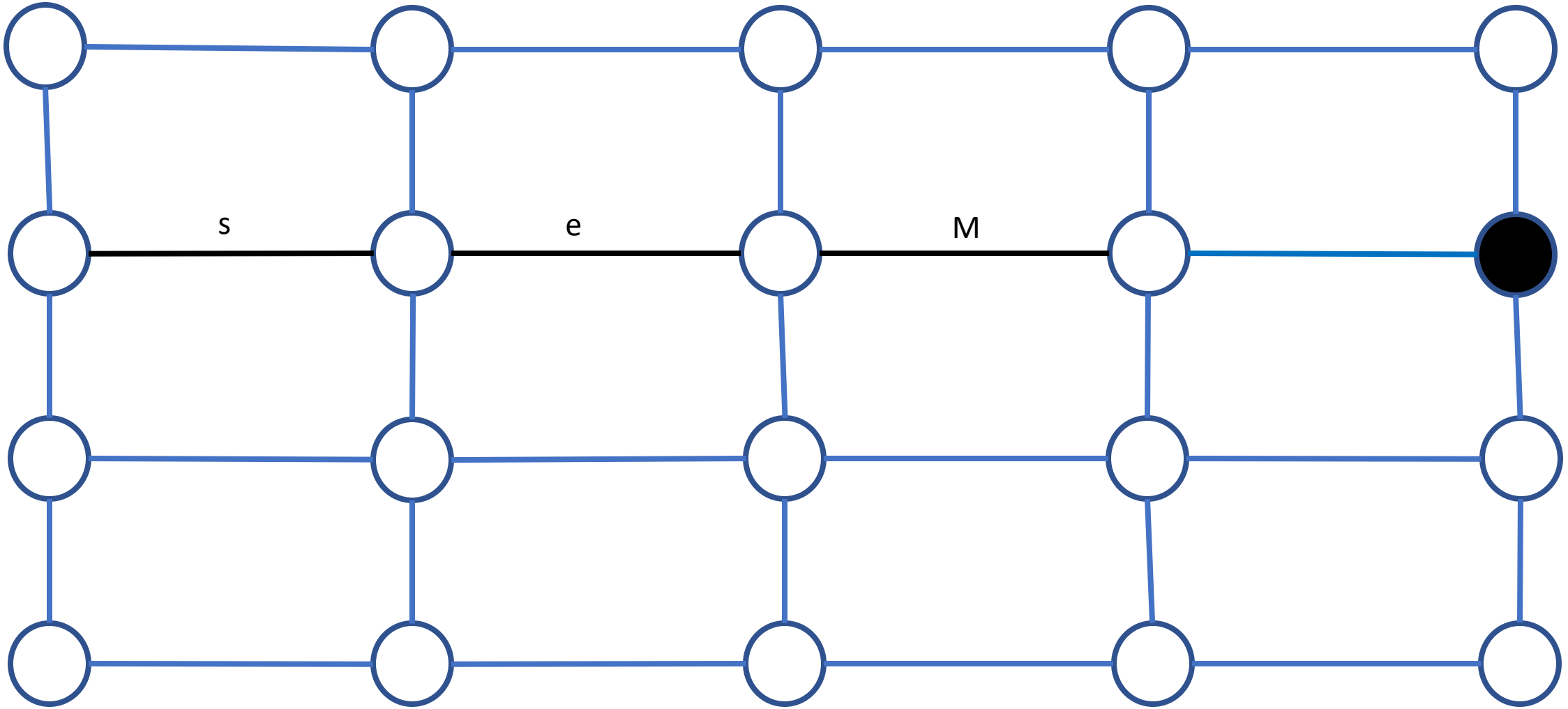
Comment ça marche



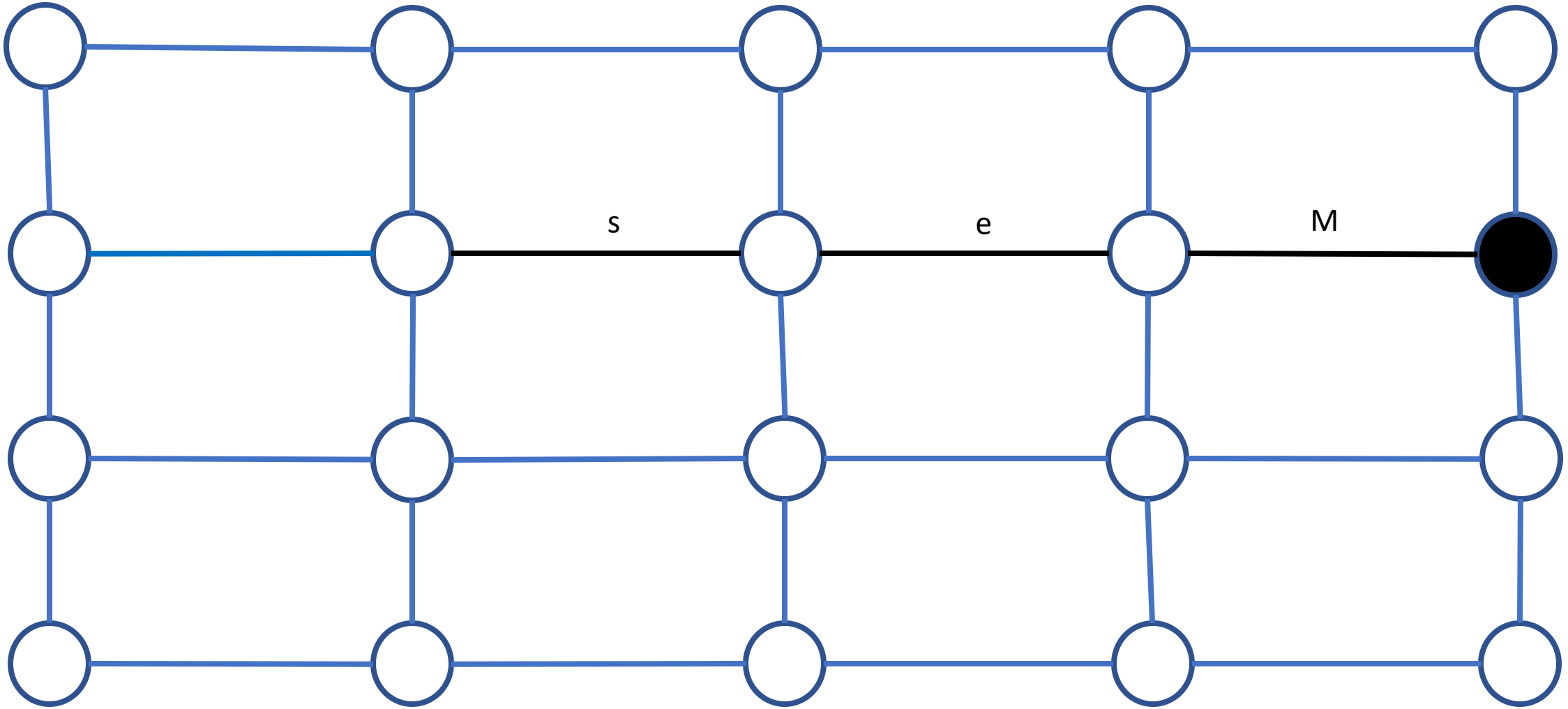
Comment ça marche



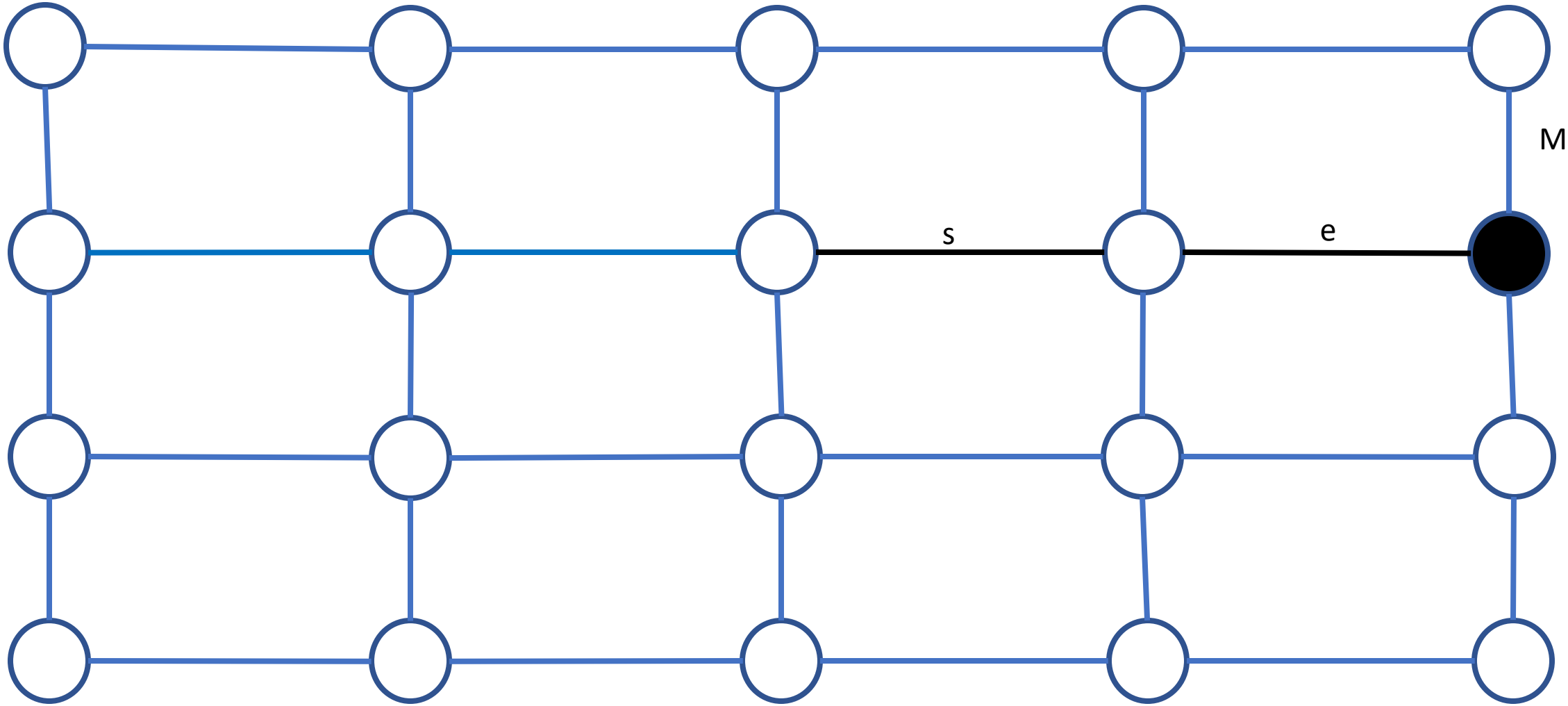
Comment ça marche



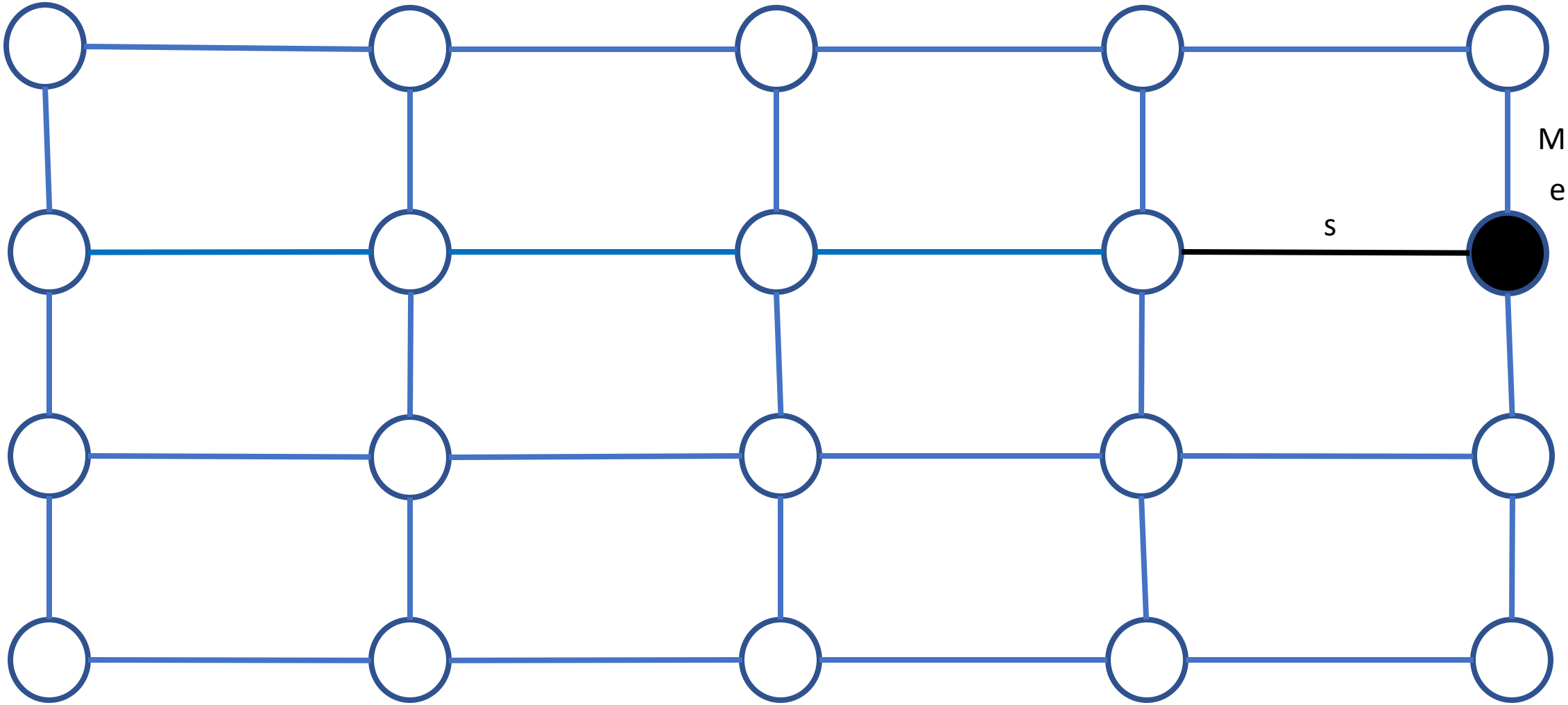
Comment ça marche



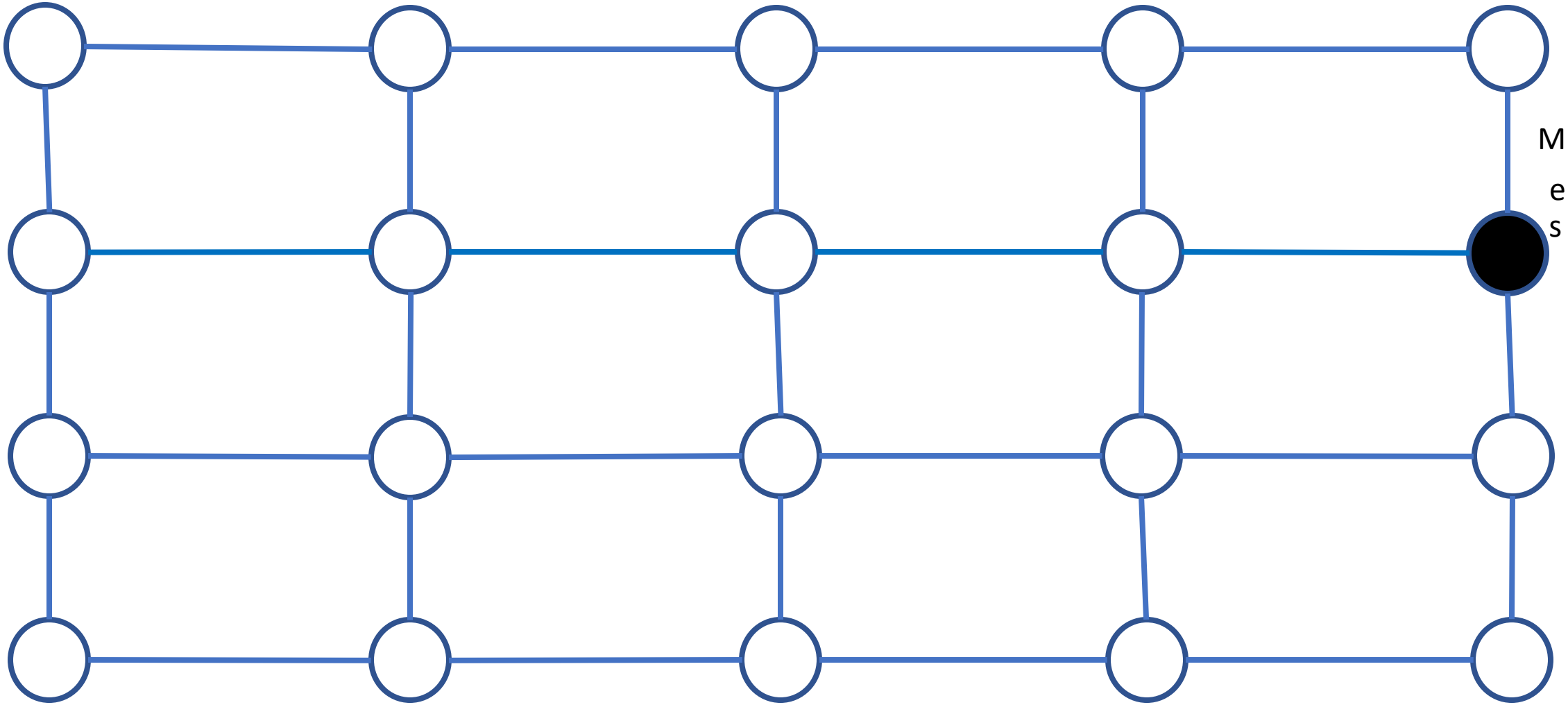
Comment ça marche



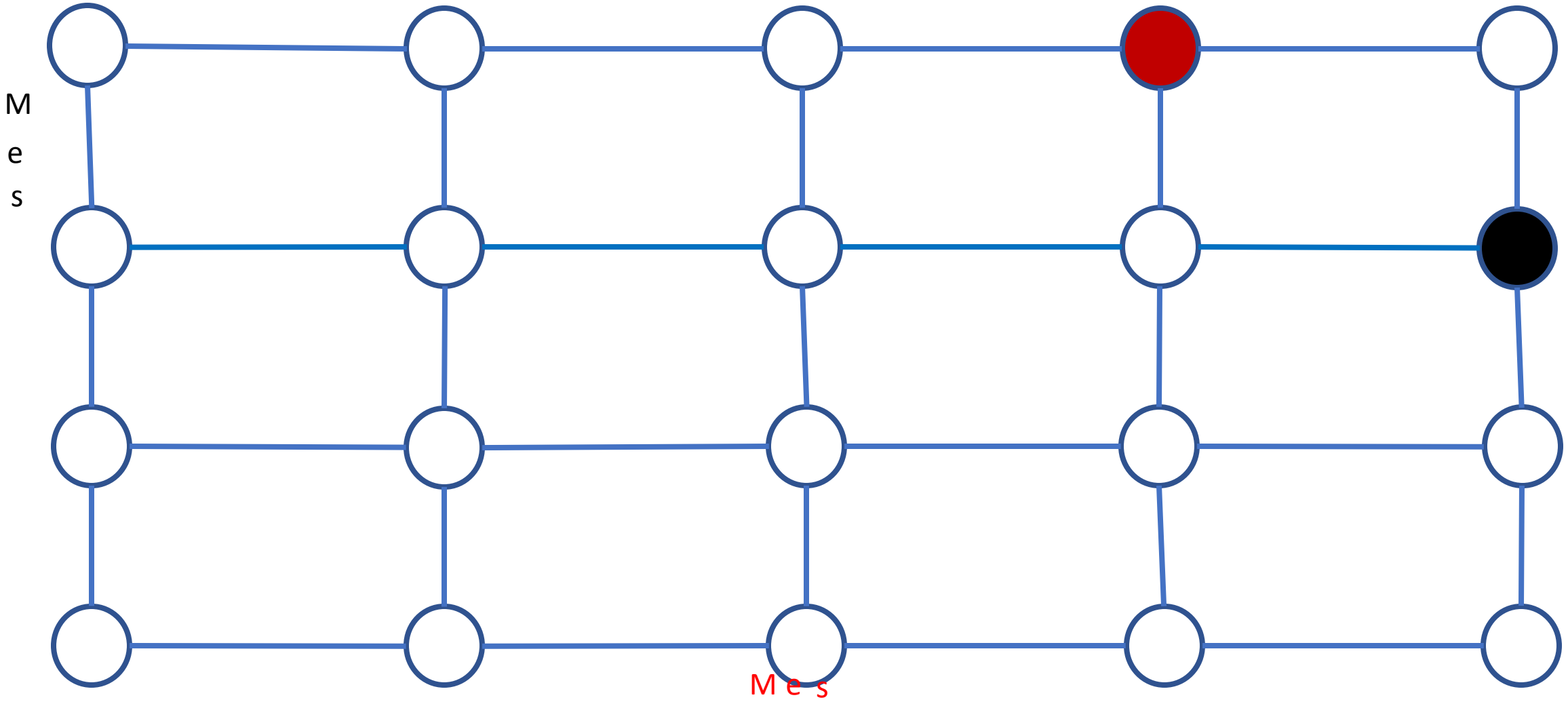
Comment ça marche



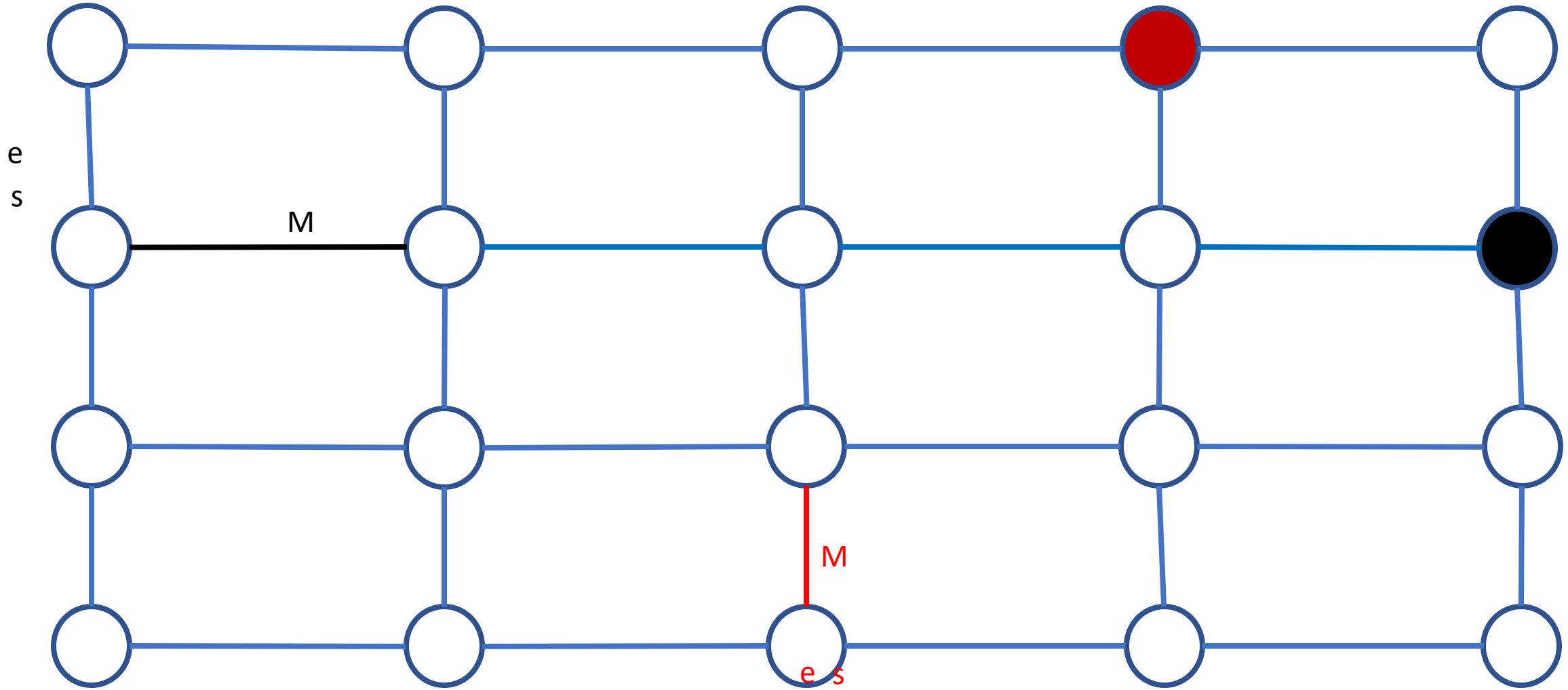
Comment ça marche



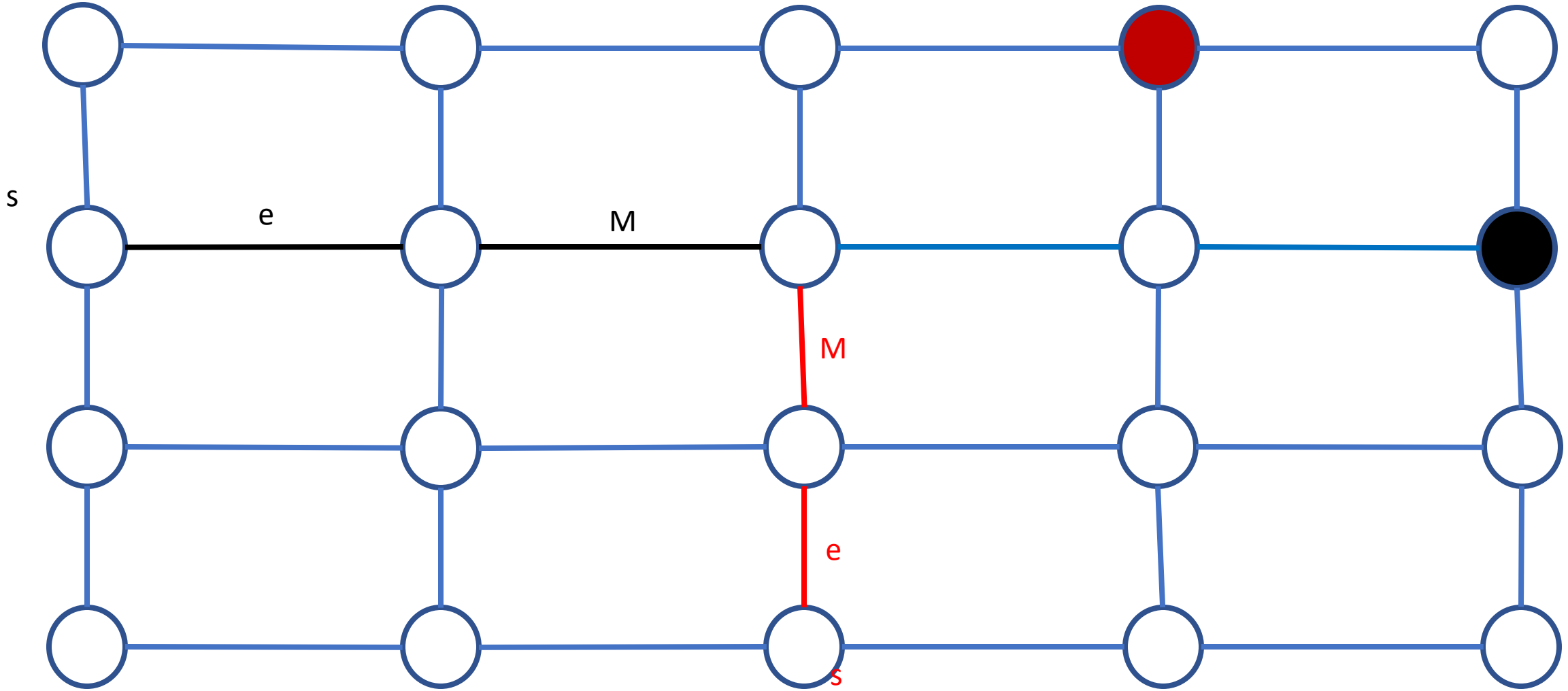
Comment ça marche



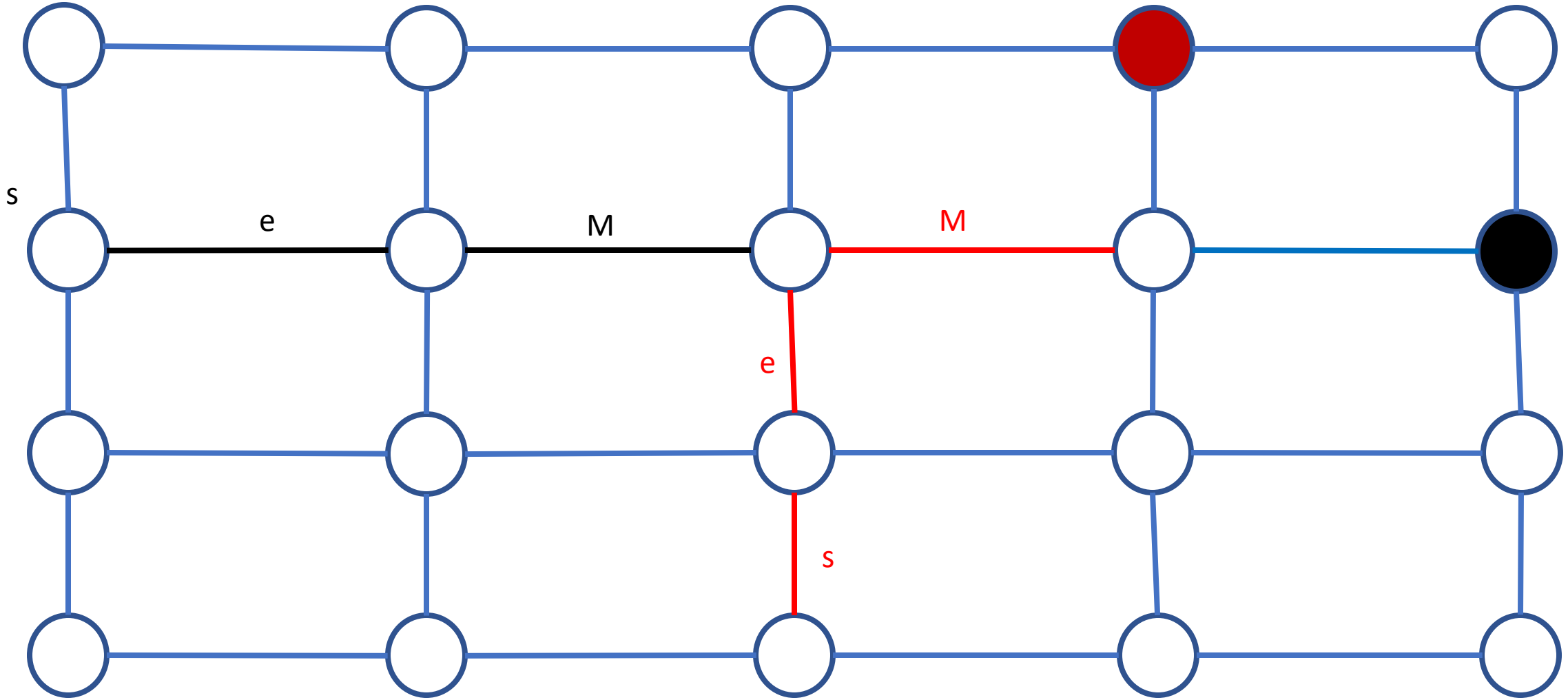
Comment ça marche



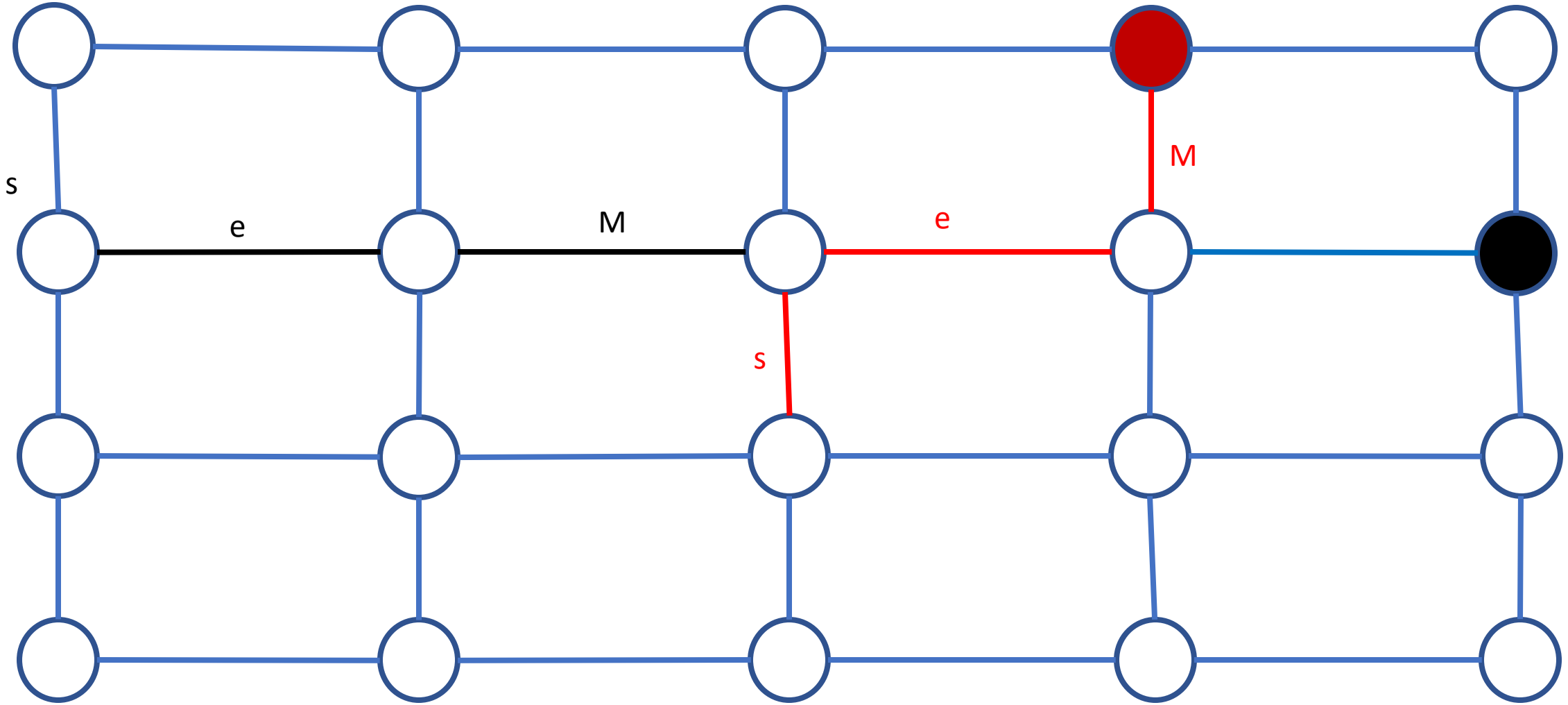
Comment ça marche



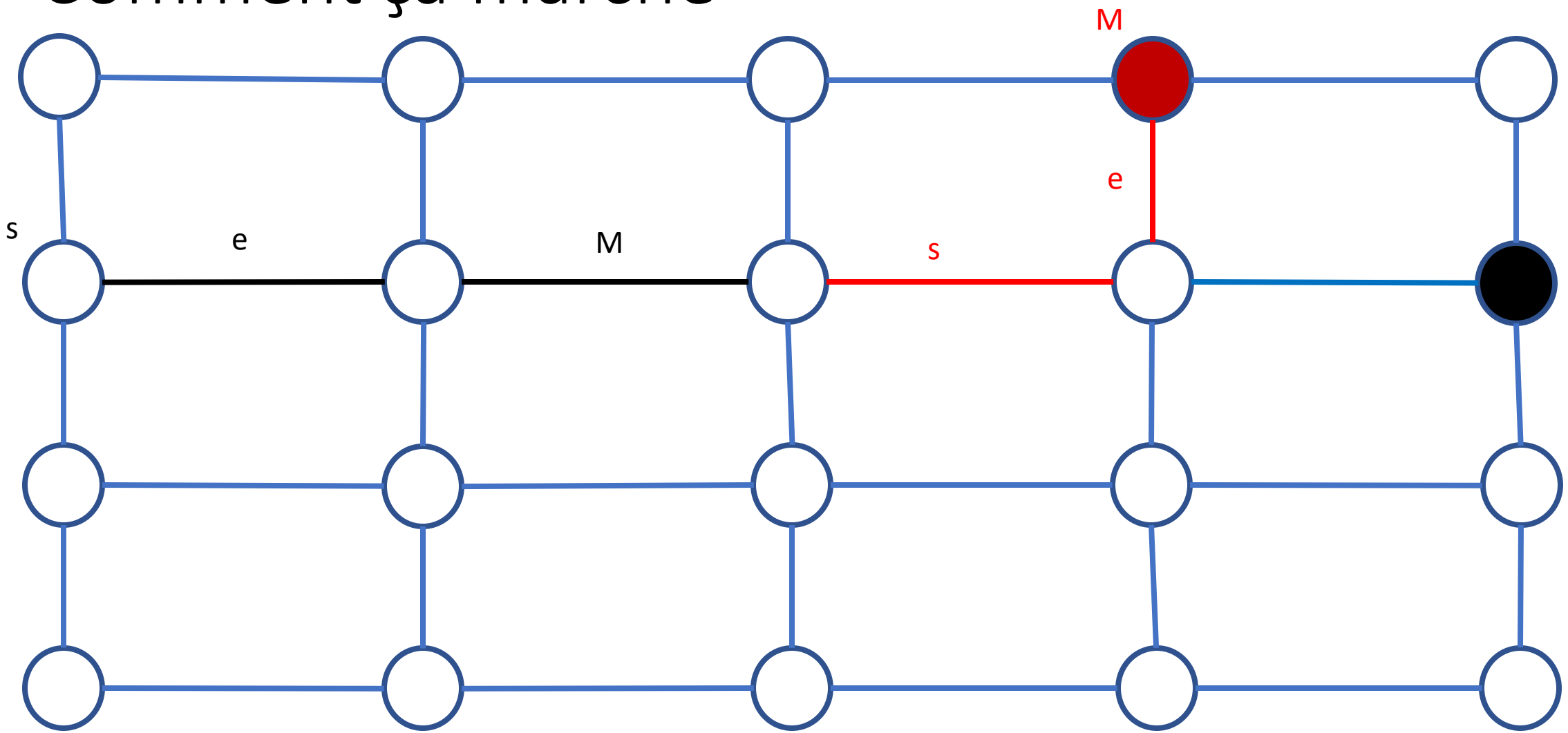
Comment ça marche



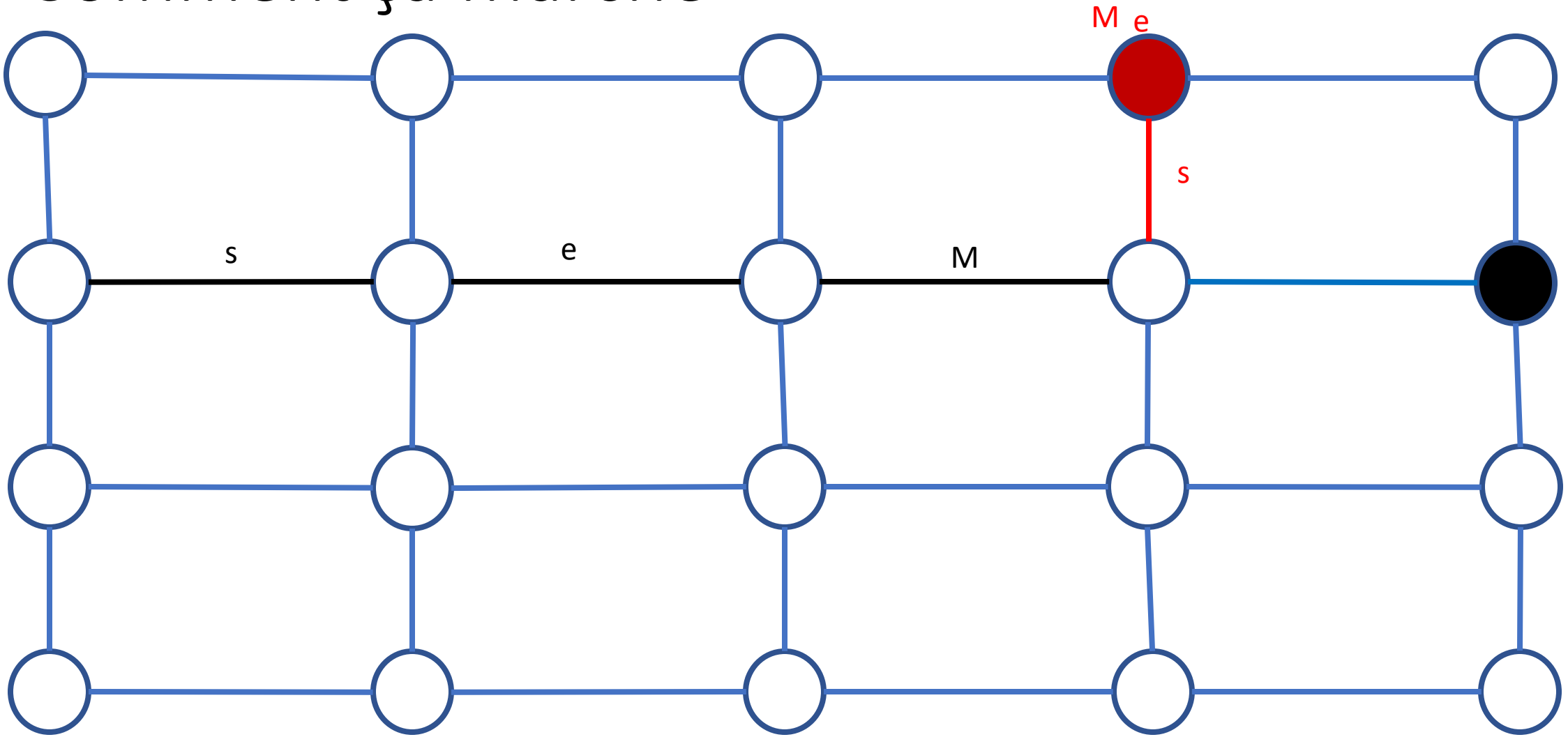
Comment ça marche



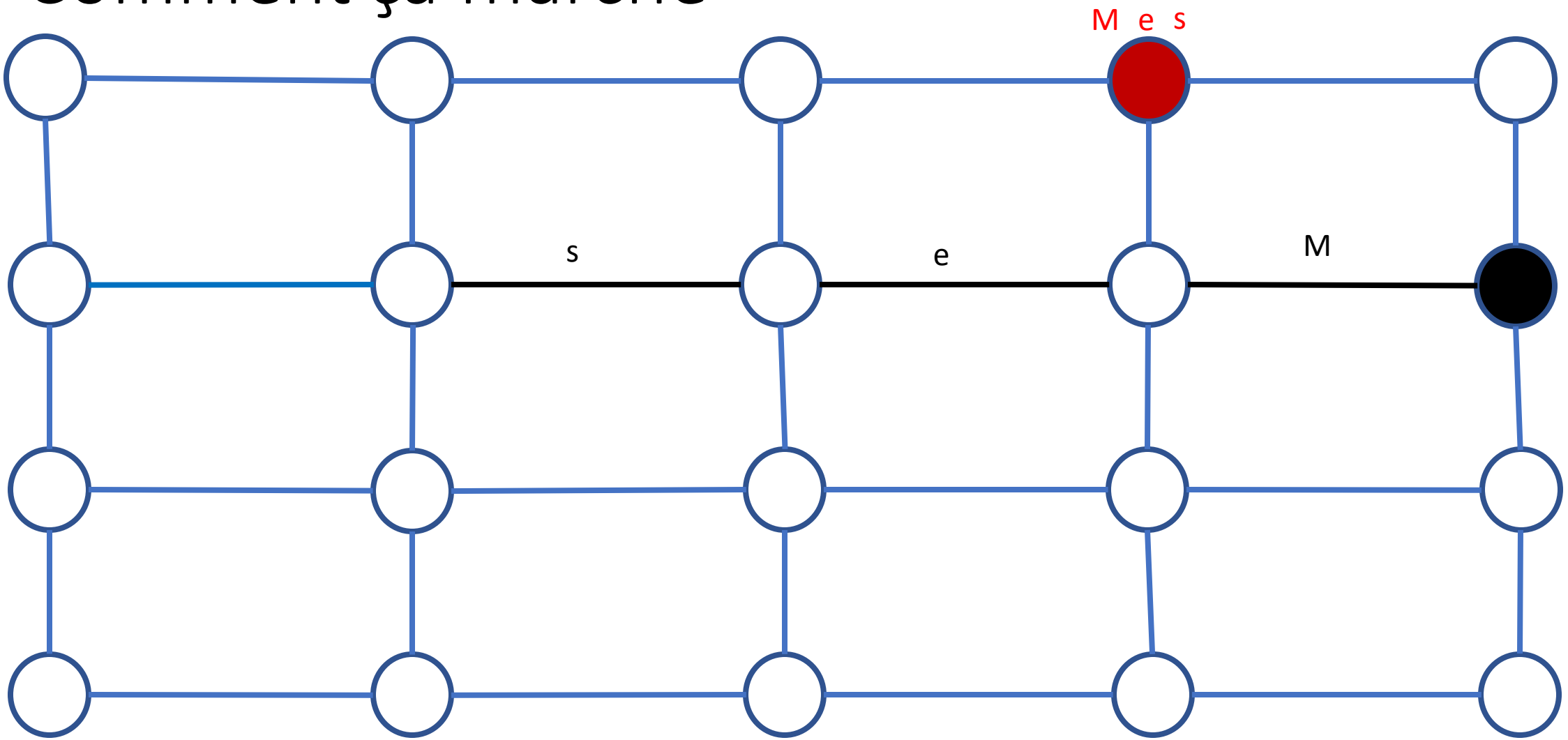
Comment ça marche



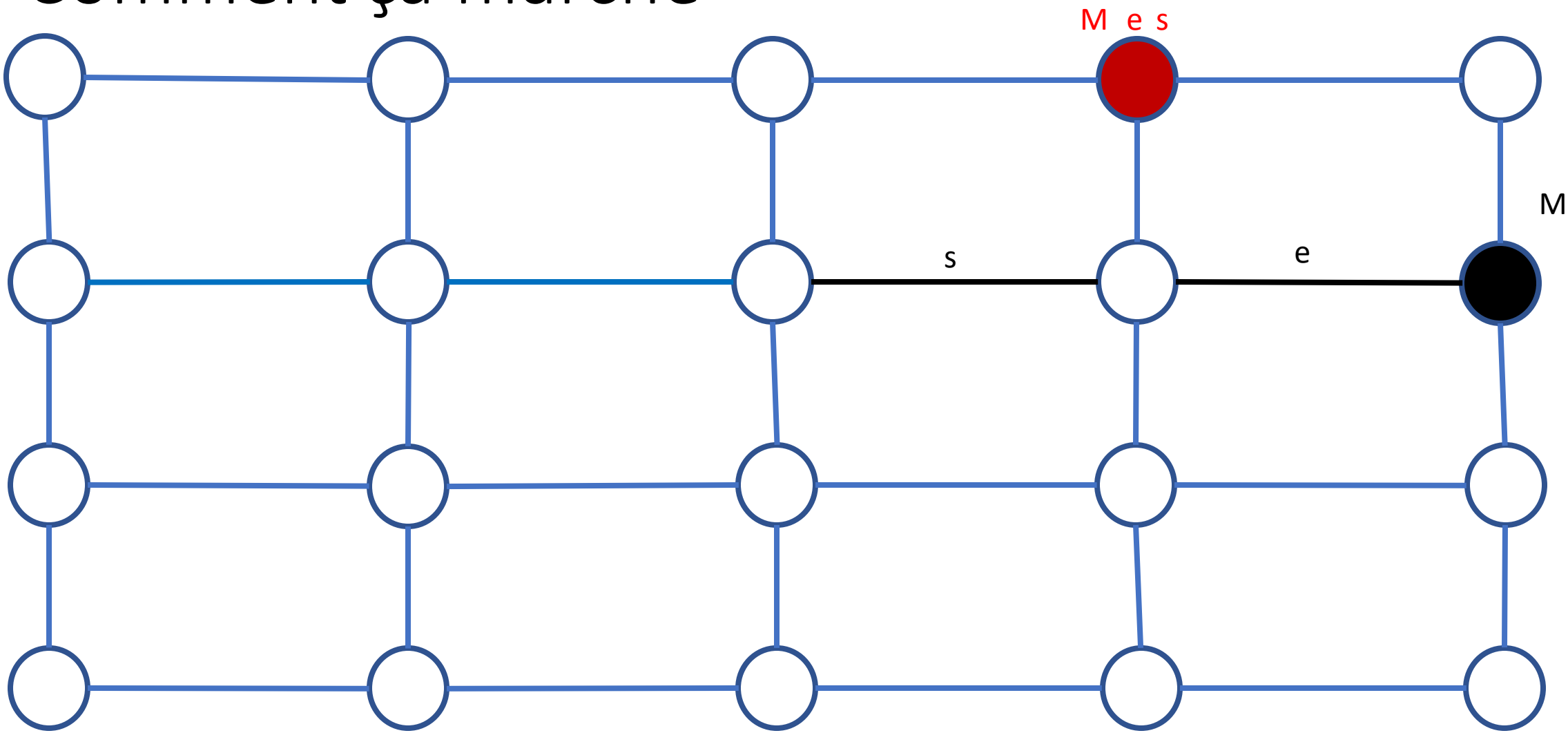
Comment ça marche



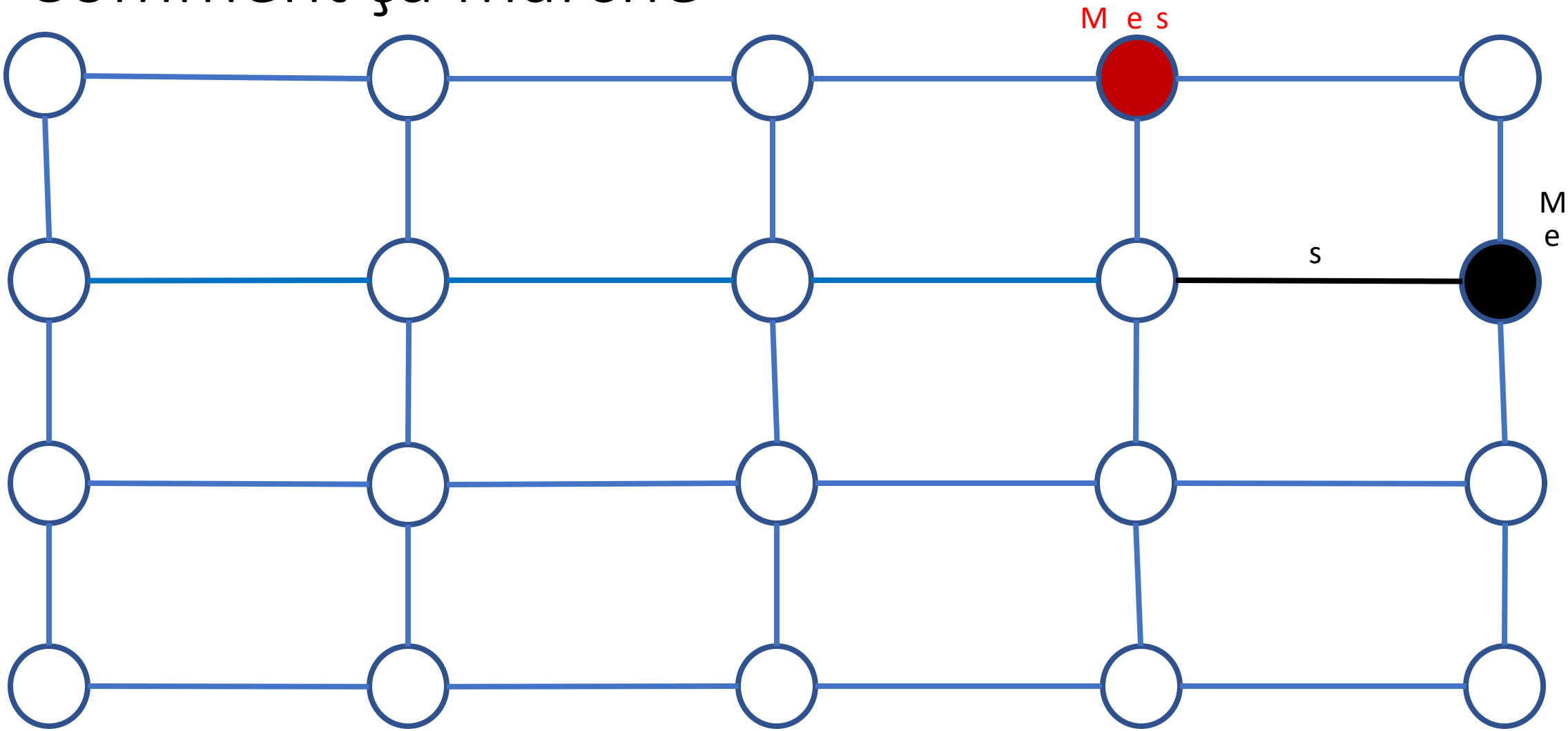
Comment ça marche



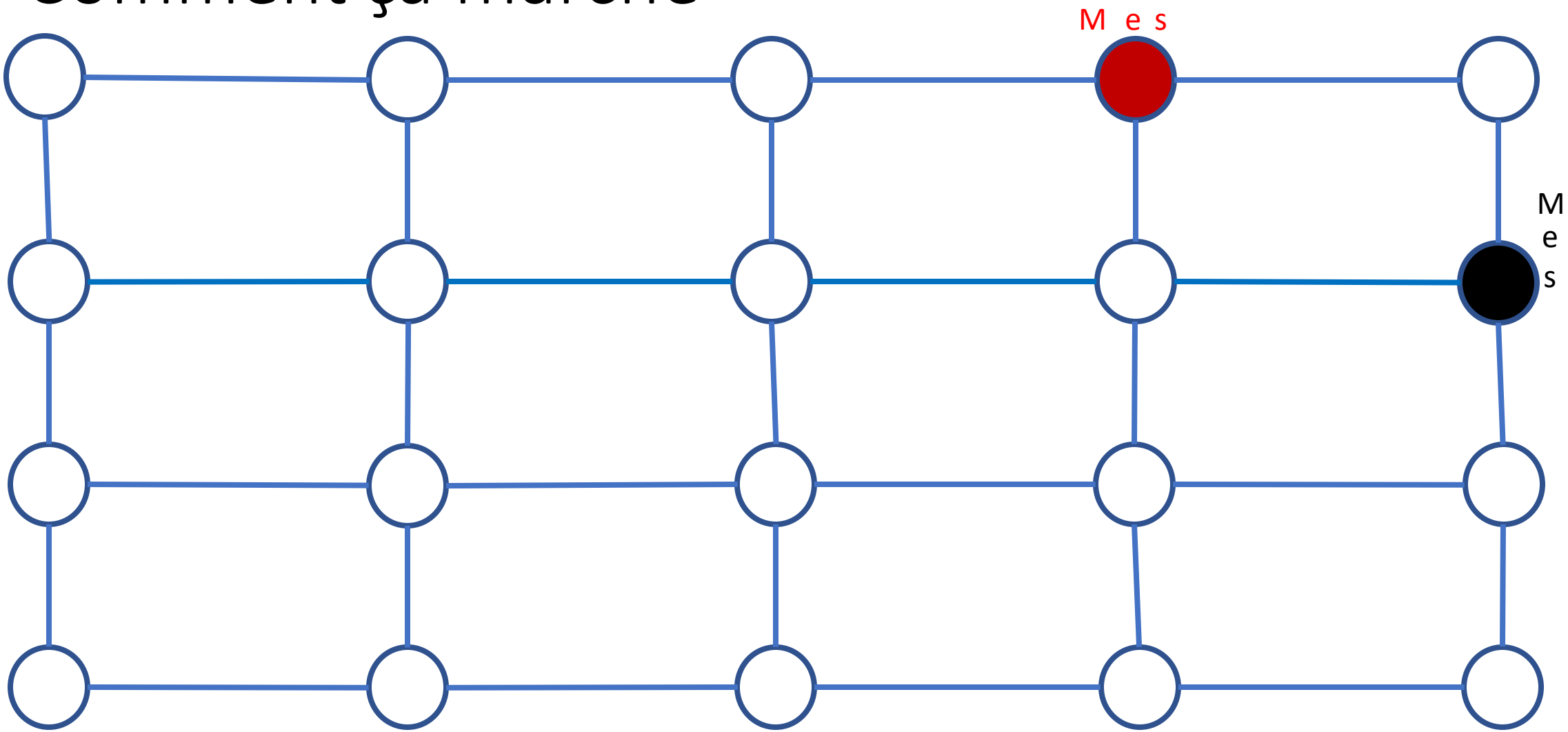
Comment ça marche



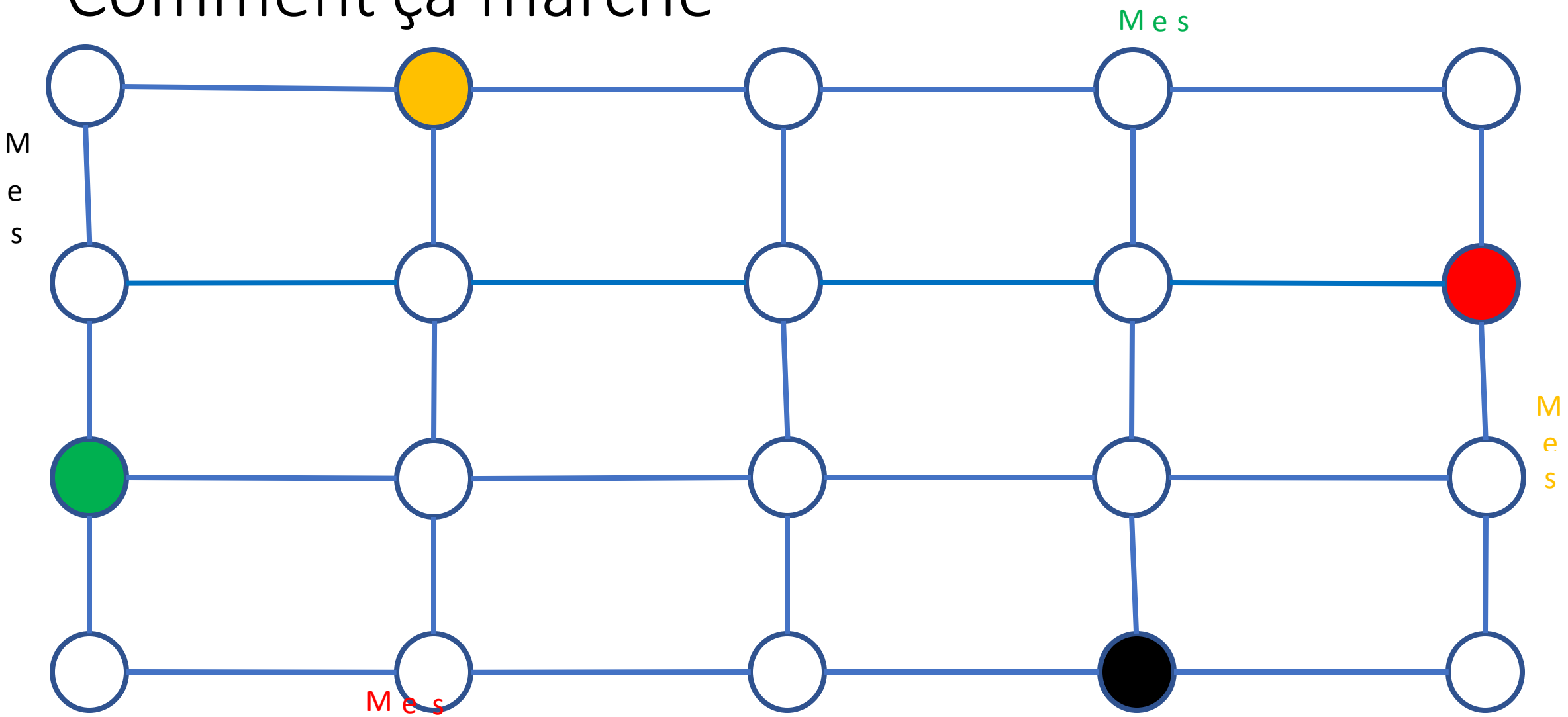
Comment ça marche



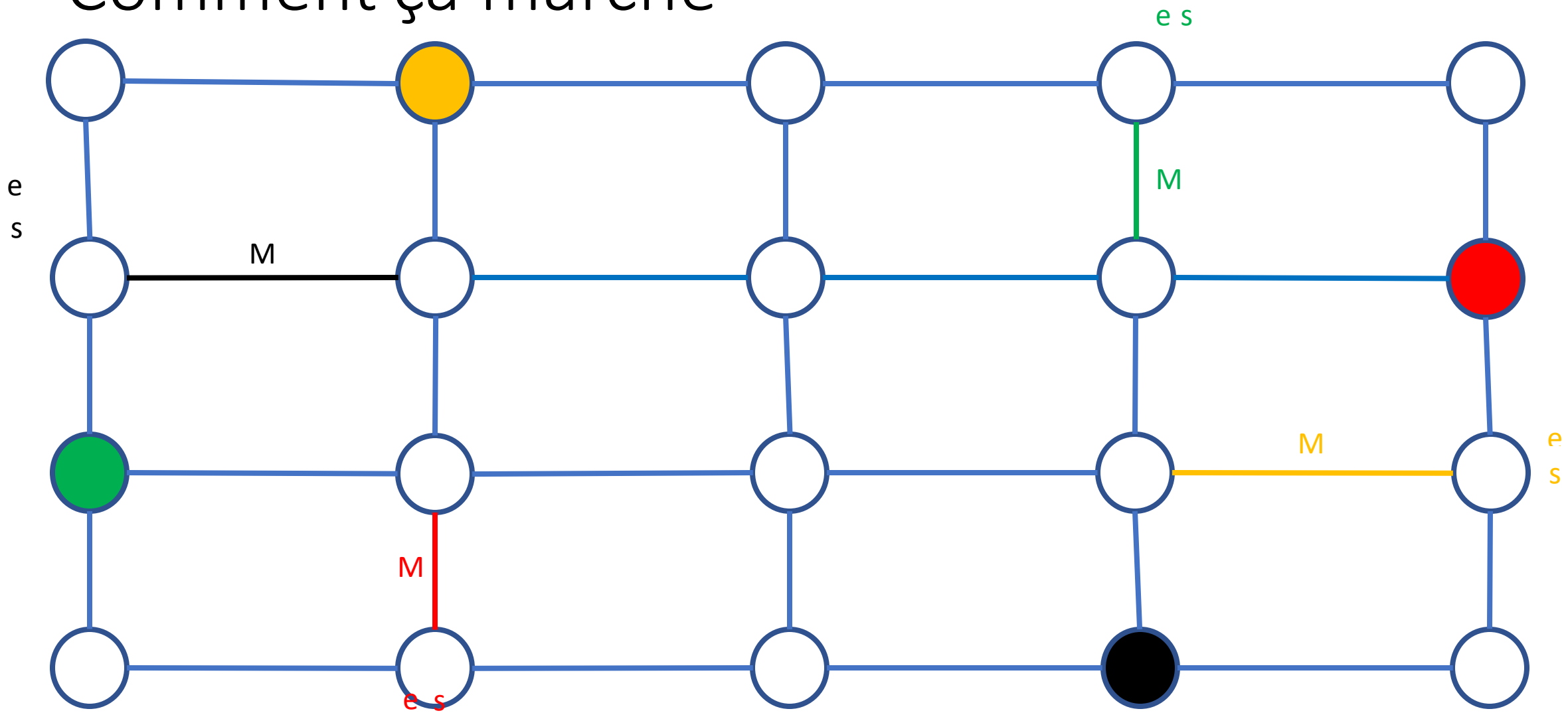
Comment ça marche



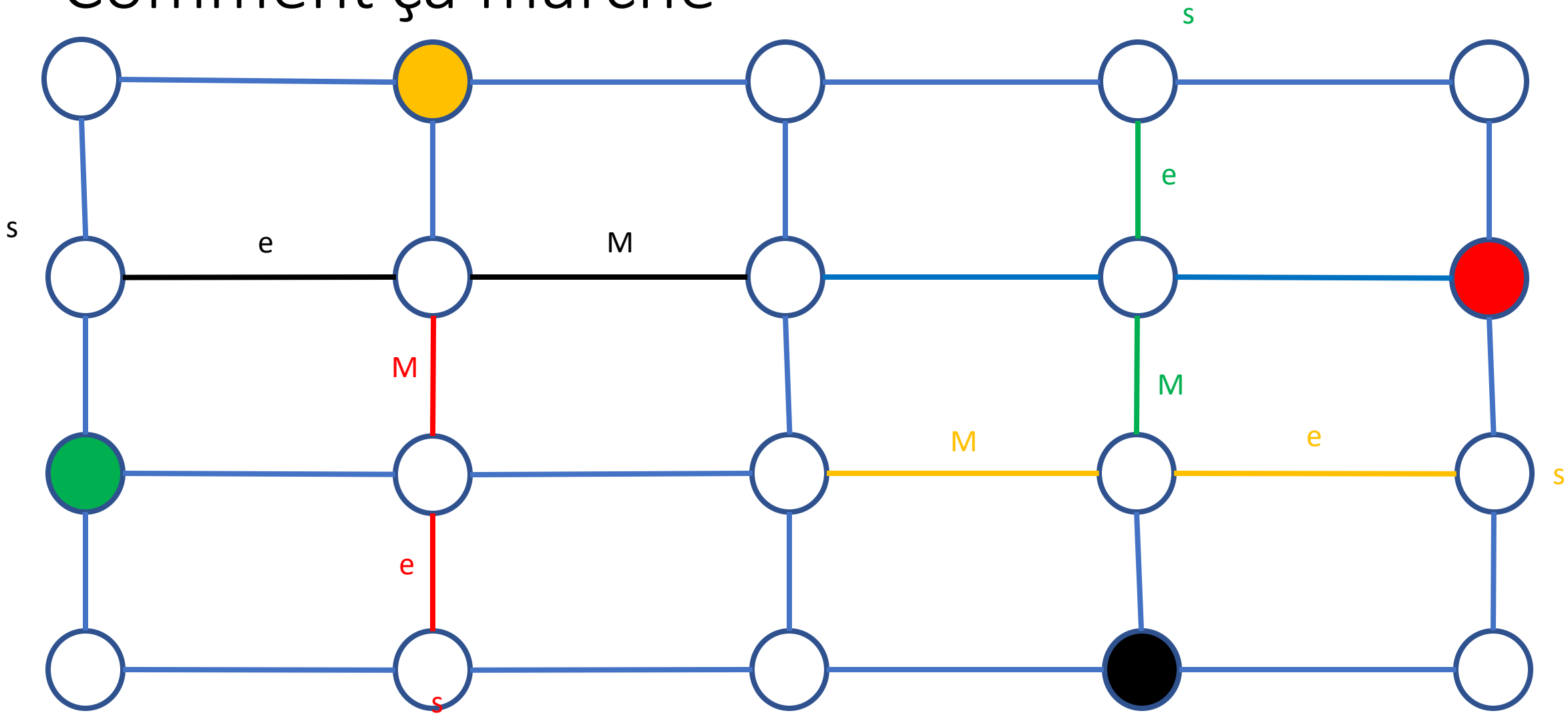
Comment ça marche



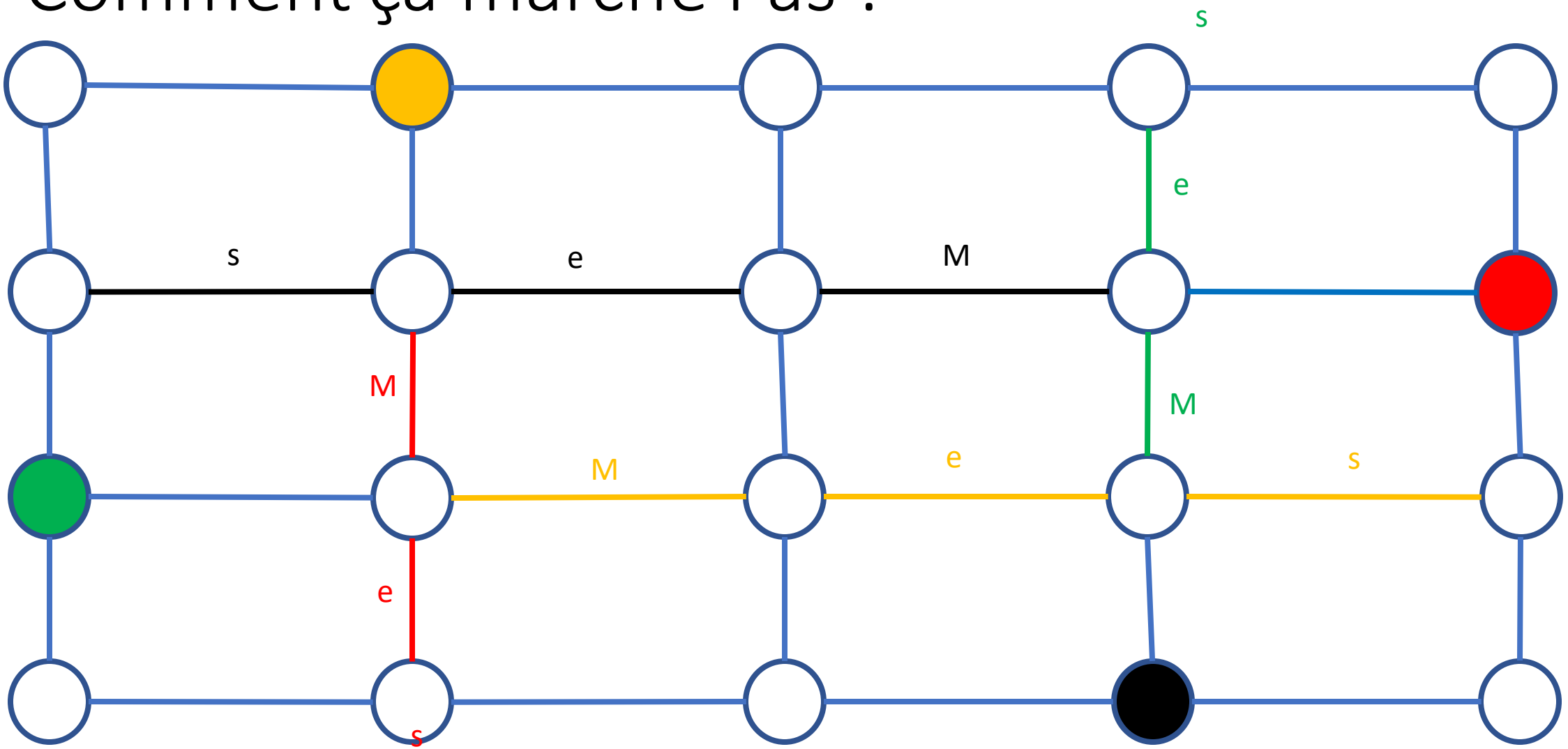
Comment ça marche



Comment ça marche



Comment ça marche Pas !



Pourquoi ?

- M monopolise les liens (0,2,E), (1,2,E) et (1,3,E) et attend le lien (2,3,S)
- M monopolise les liens (1,0,N) et (1,1,N) et attend le lien (1,2,E)
- M monopolise les liens (4,1,W), (3,1,W) et (2,1,W) et attend le lien (1,1,N)
- M monopolise les liens (3,3,S) et (3,2,S) et attend le lien (3,1,W)

- C'est un bête problème de gestion de ressources comme vu en système d'exploitation.
- Est-ce la faute de la méthode ?

Comment résoudre ce problème ?