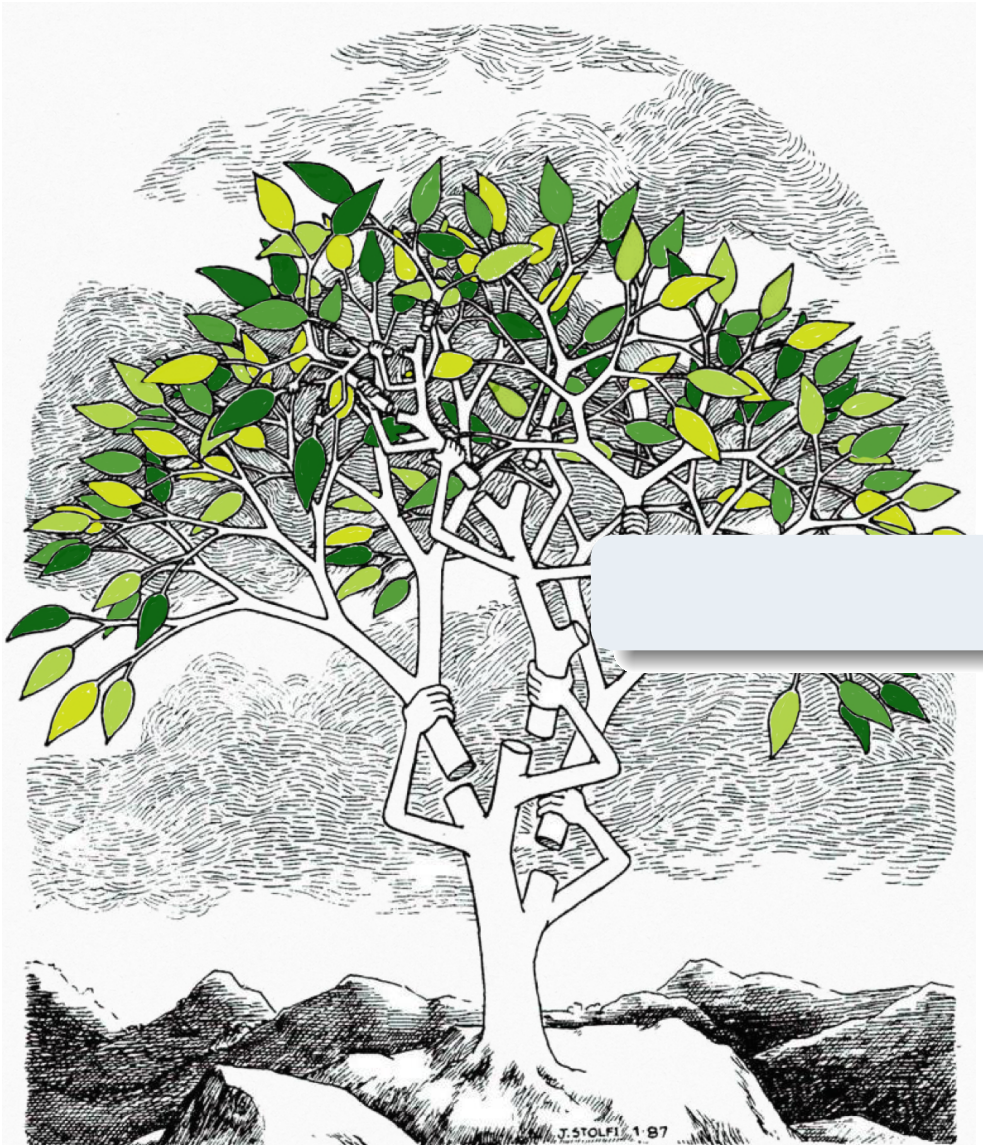
Memory Lower Bounds for Deterministic Self-Stabilization

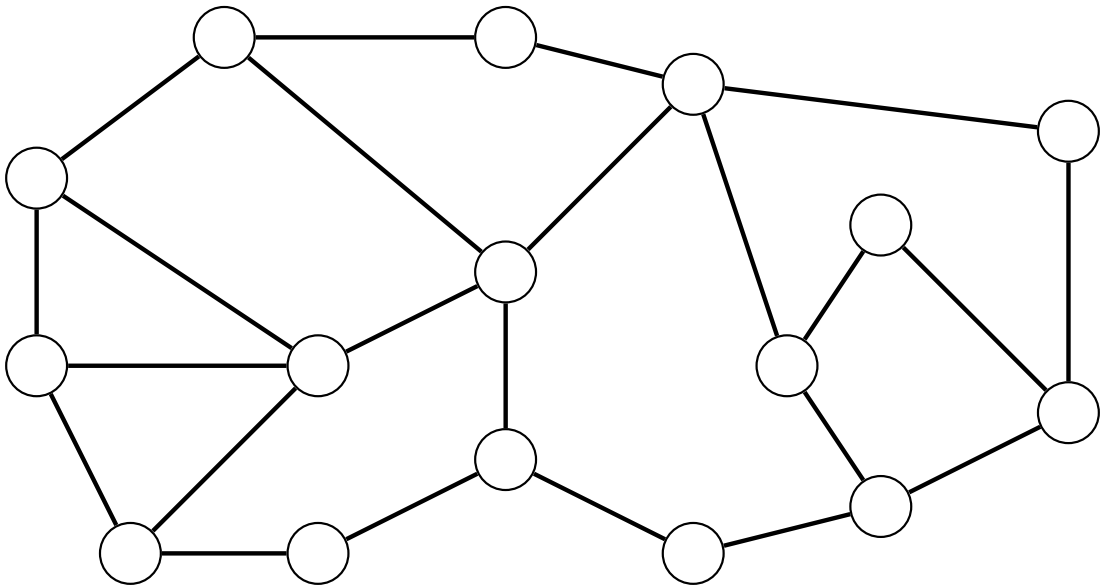
Lélia Blin, Laurent Feuillolet et Gabriel Le Bouder

Sorbonne Université, LIP6.



Modèle

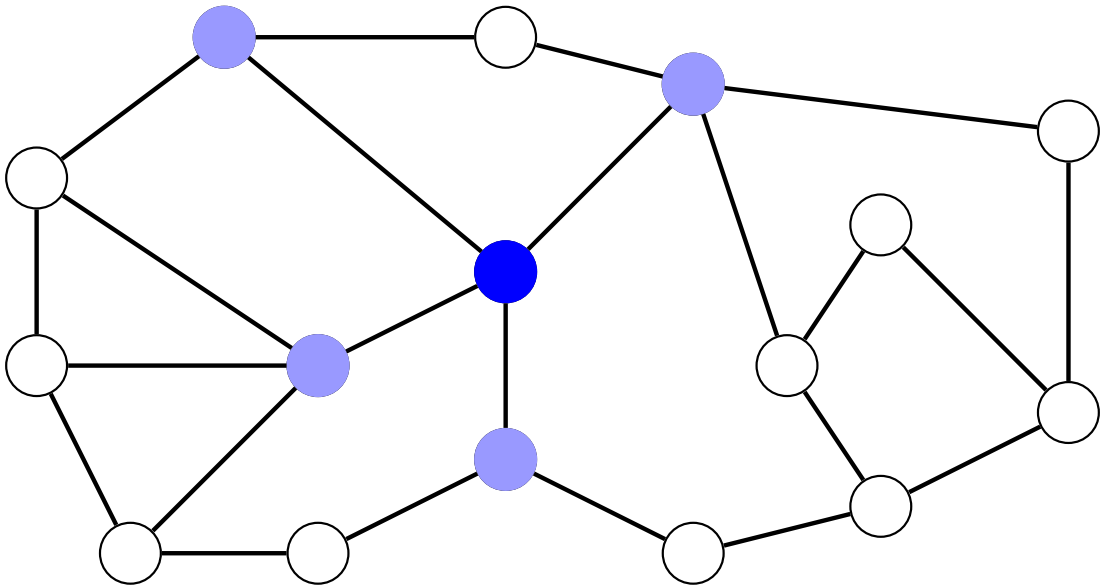
Système réparti



Modèle

- Réseaux asynchrones $G = (V, E)$ avec identifiants.
- Fautes transitoires (corruptions de variables).

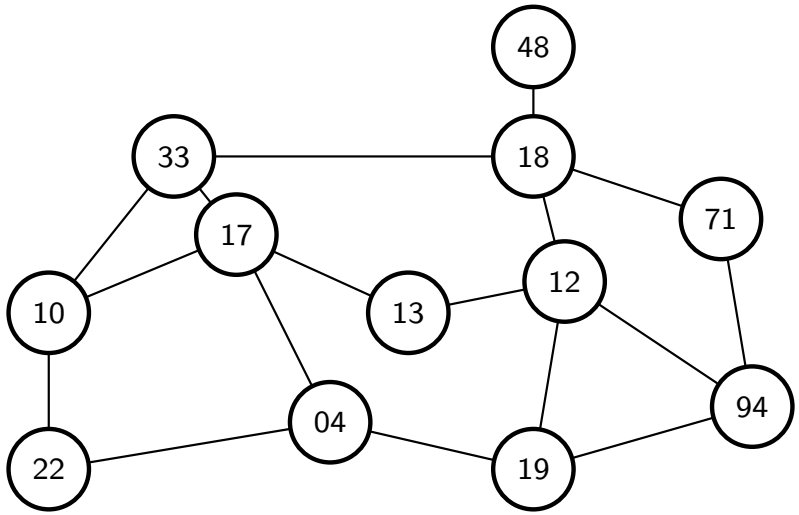
Modèle à Etat



En une étape atomique un noeud v peut

- Lire ses variables et les variables de ses voisins.
- Calculer.
- Mettre à jour ses variables.

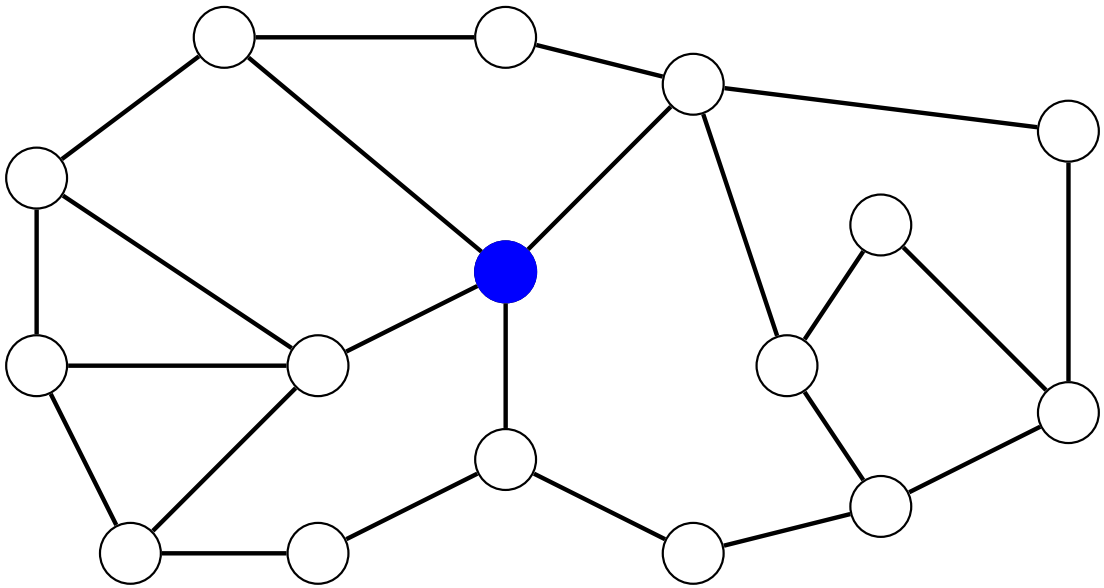
Réseaux non-anonyme



Identifiants

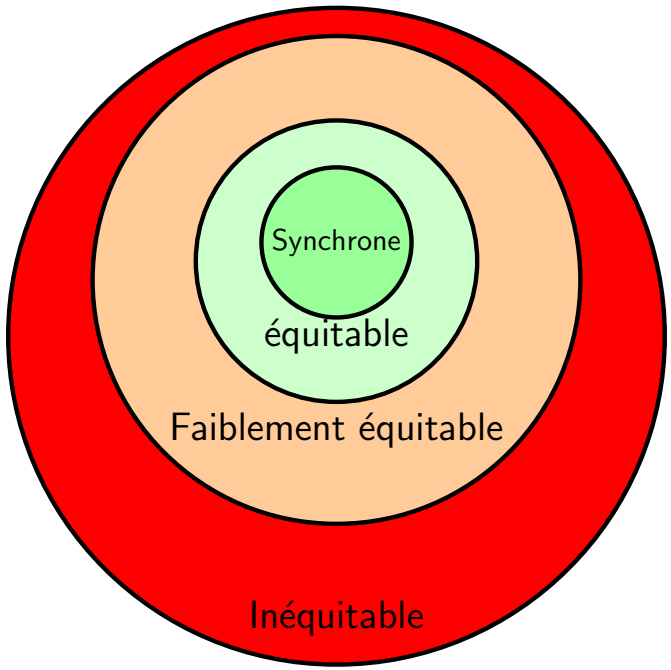
- Identifiants deux à deux distincts.
- $\exists c > 1 : \forall v \in V, id_v \in [1, n^c]$
- Les identifiants ne sont pas stocké dans les variables, ils ne sont pas accessibles aux voisins.

Noeud activable



Un noeud est activable si au moins une des règles de son algorithme est exécutable.

Ordonnanceur



Definition

Ordonnanceur choisit parmi les noeuds activables les noeuds qui s'activent.

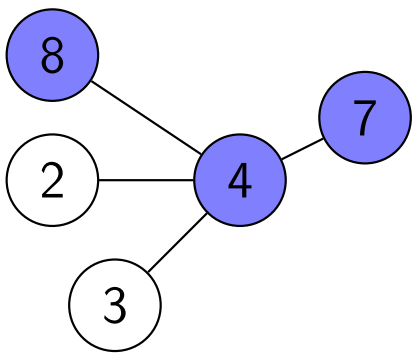
Configurations

Etat

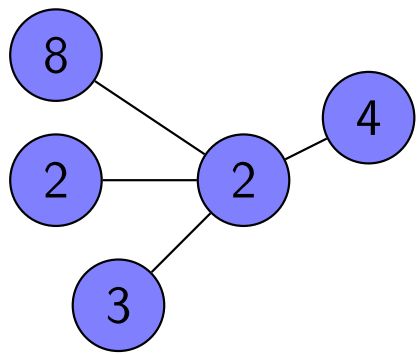
L'état d'un noeud est l'ensemble de ses variables

Configuration

Pour un graphe G, une configuration Γ est l'ensemble des états de ses noeuds à un instant donné.



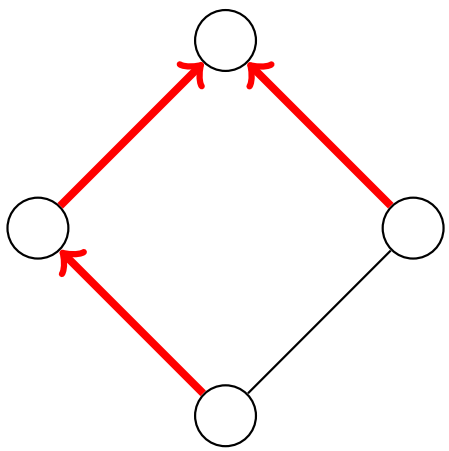
(a)

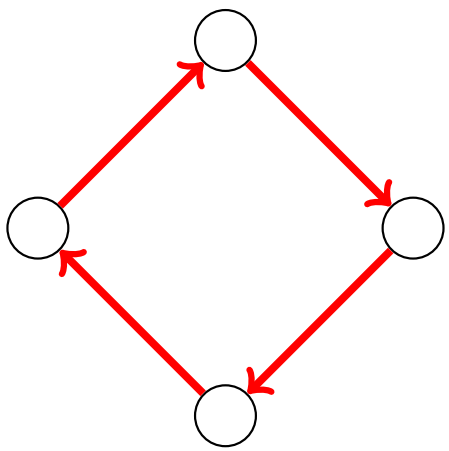


(b)

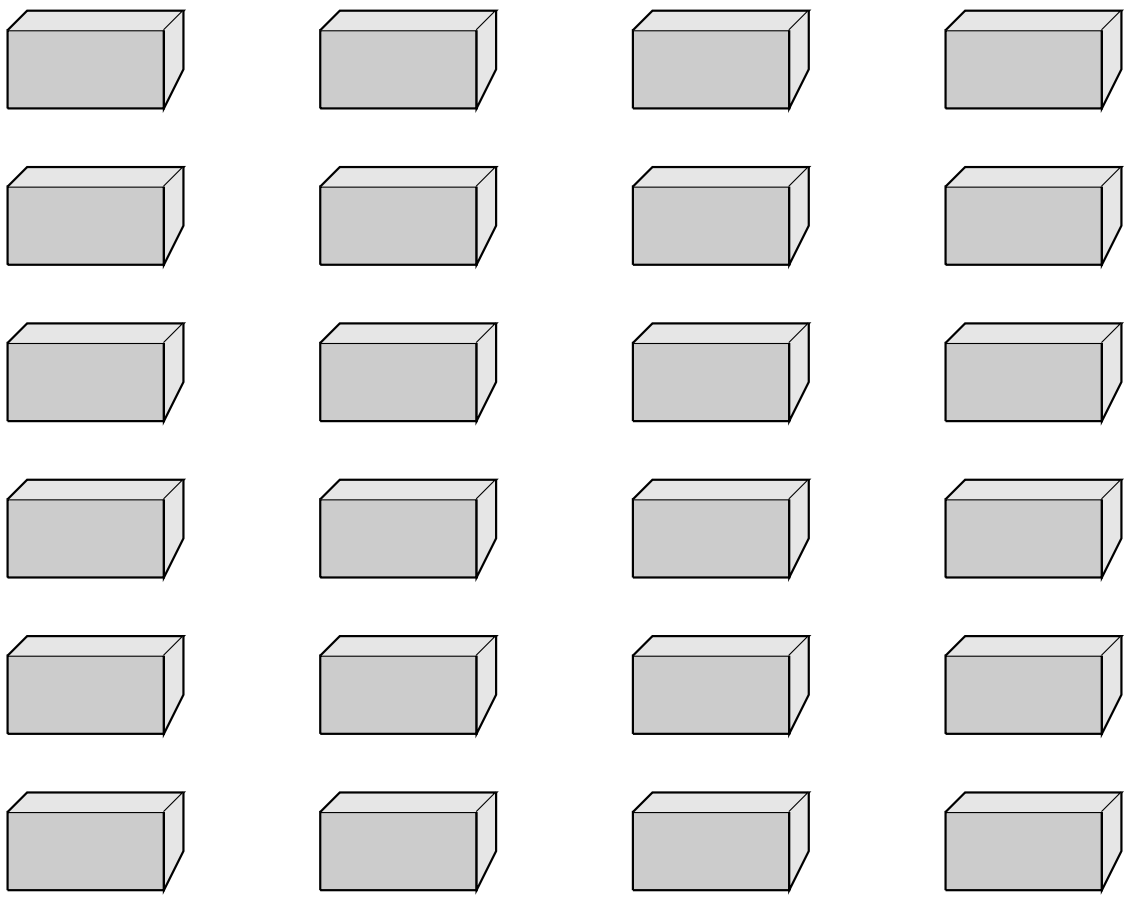
Configurations légitimes

Dépend du prédicat correspondant à la tâche à résoudre.
Exemple : prédicat arbre couvrant.

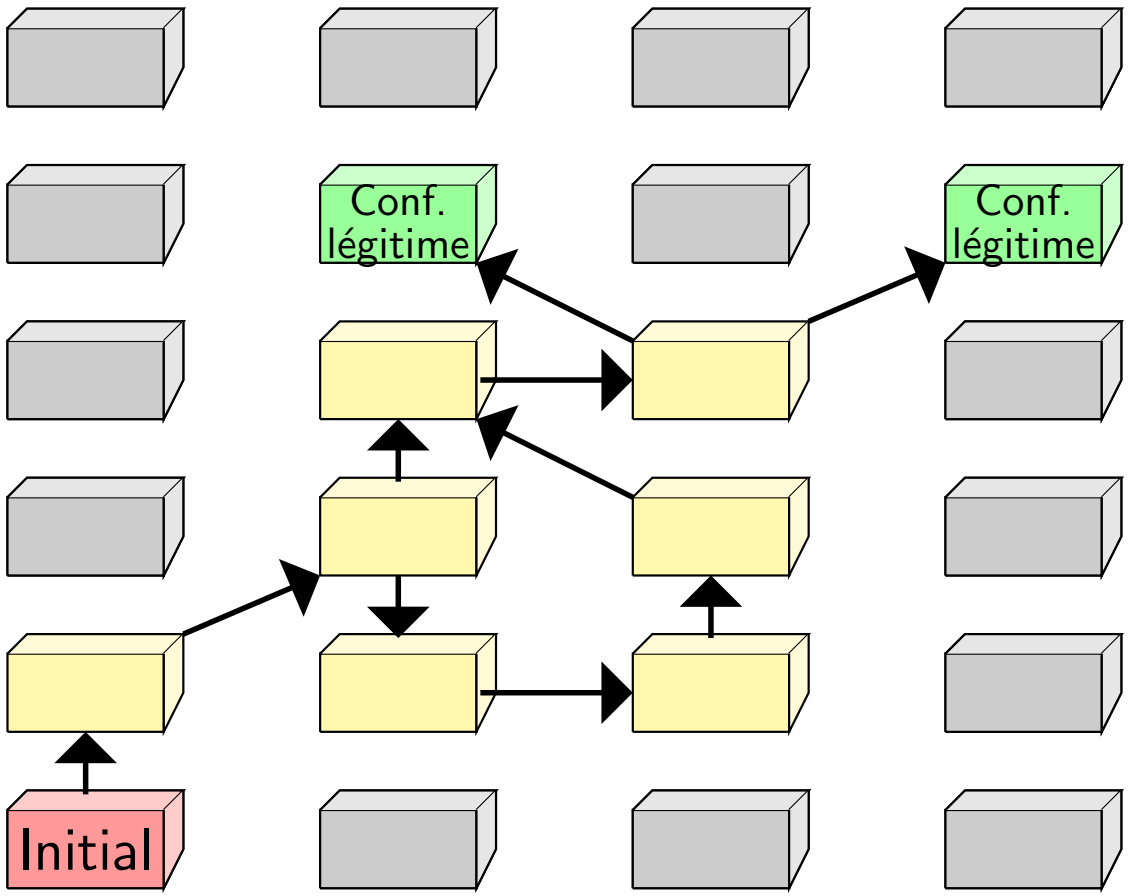




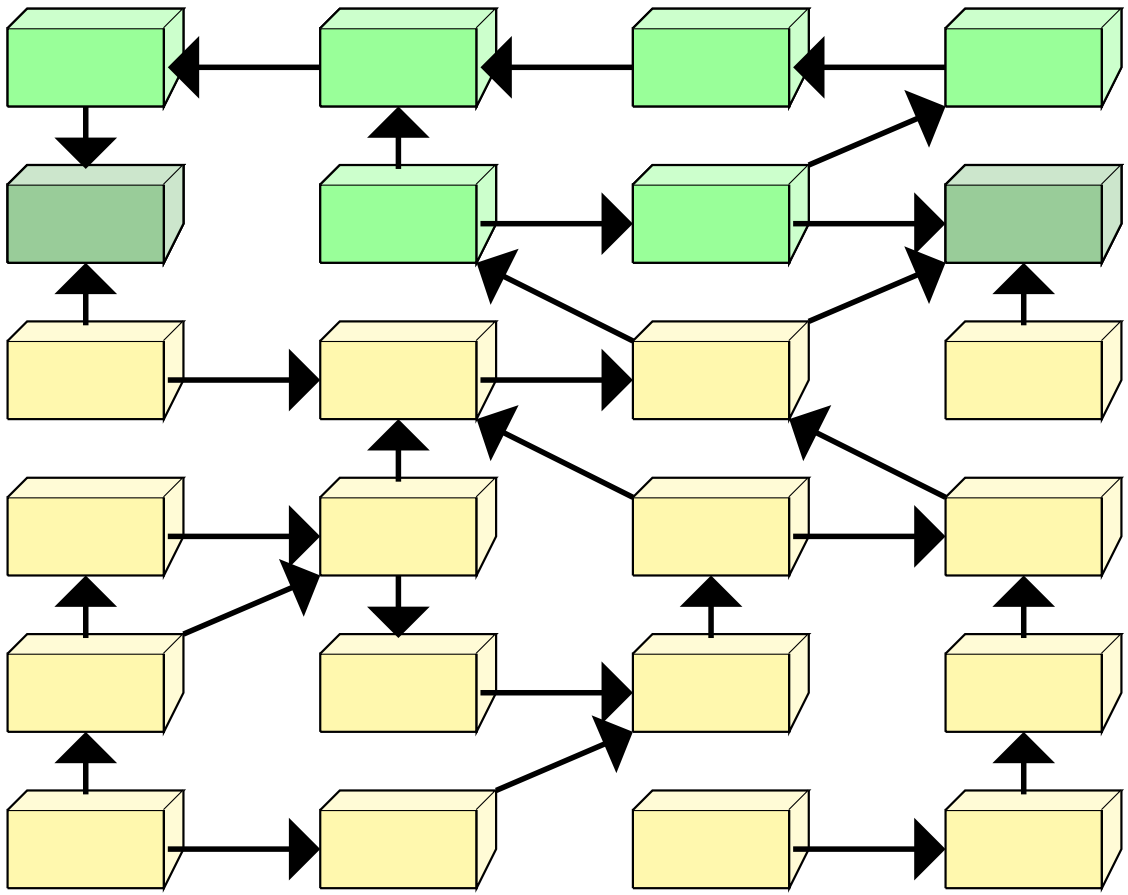
Ensemble de toutes les configurations



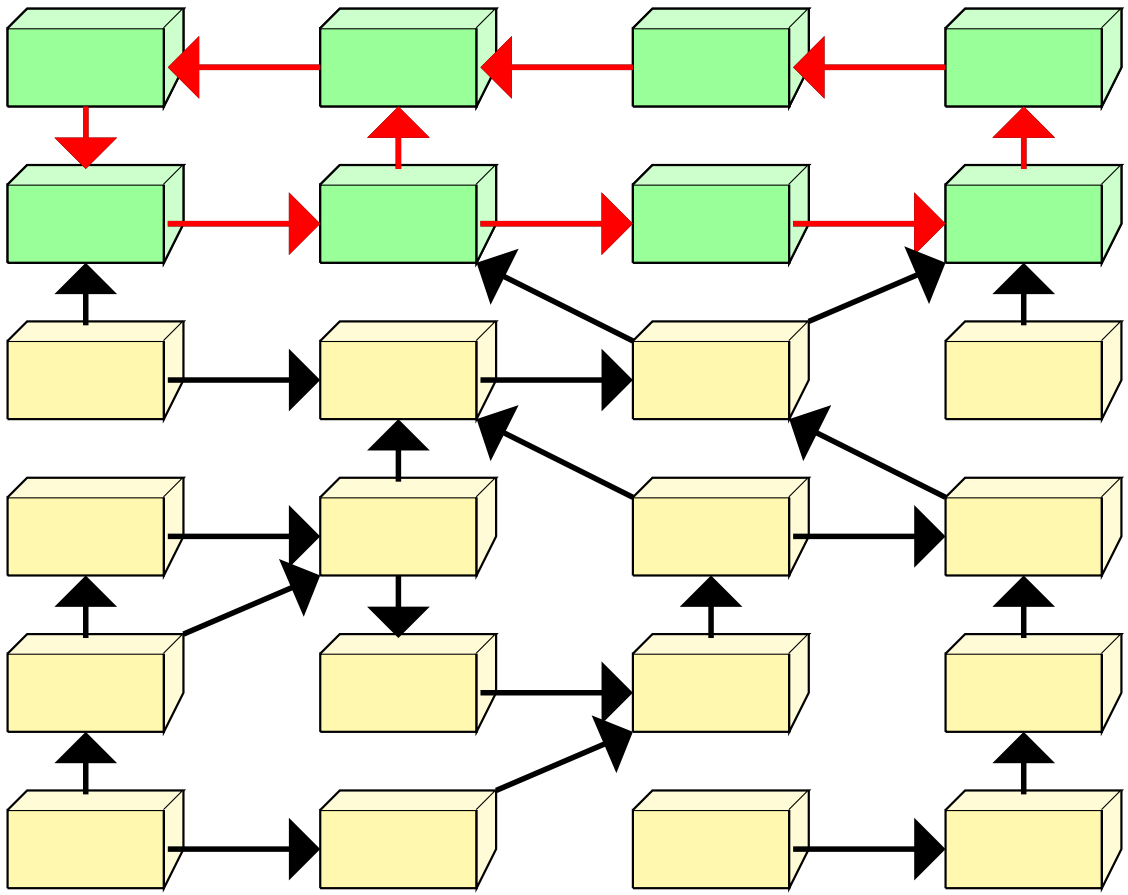
Algorithmes distribuées "classique"



Auto-stabilisation propriété de Silence



Auto-stabilisation



Algorithme auto-stabilisant

Dijkstra, 1974

Un algorithme auto-stabilisant résolvant une tâche $\mathcal{T}$ est un algorithme distribué $\mathcal{A}$ satisfaisant :

1. Convergence : Démarrant d'une configuration arbitraire, $\mathcal{A}$ finit par rejoindre une configuration légale.
2. Cloture : Démarrant d'une configuration légale, le système reste dans une configuration légale.

Complexités

Espace mémoire

Espace mémoire maximum utilisé par l'ensemble des variables (en binaire).

Temps : nombre d'étapes

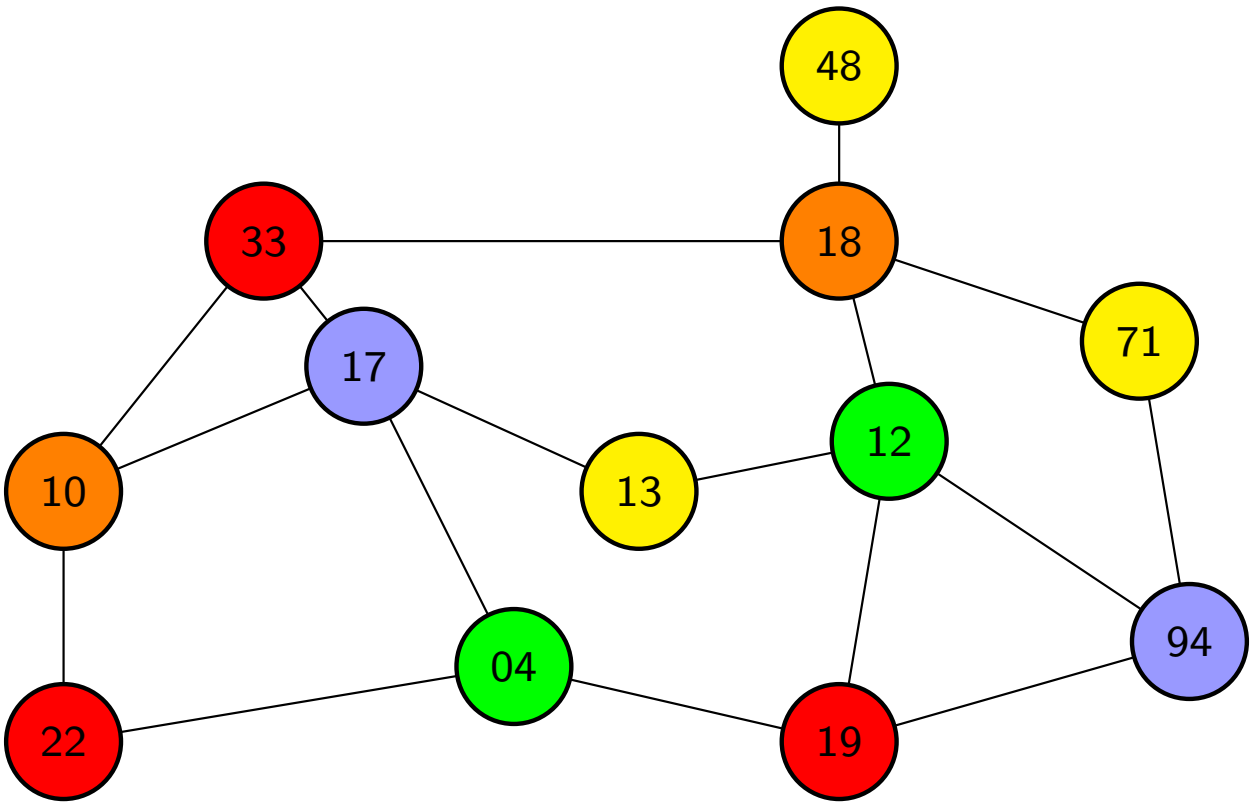
Une étape est une transition d'une configuration vers une autre.

Temps : Le nombre de rondes

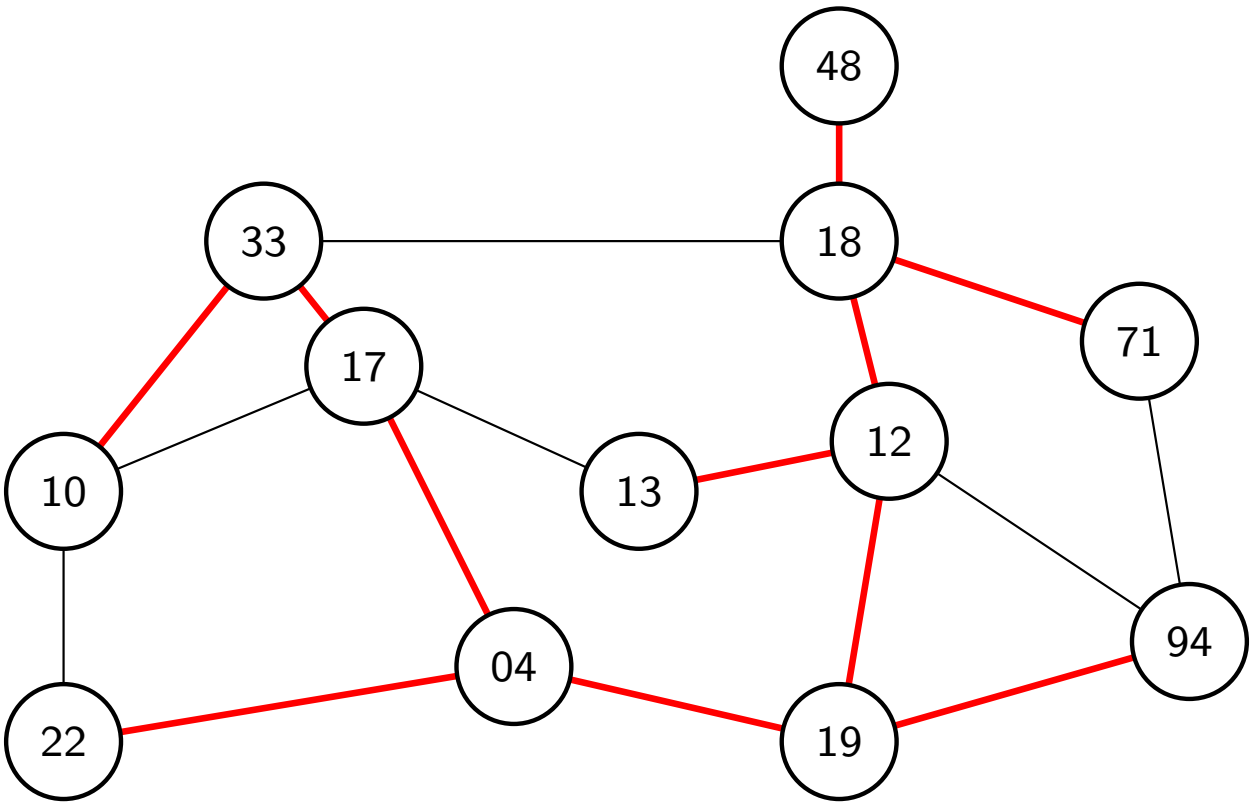
Une ronde est la plus petite portion d'exécution où tout noeud activable est activé ou devient non activable.

Fondamental Problems

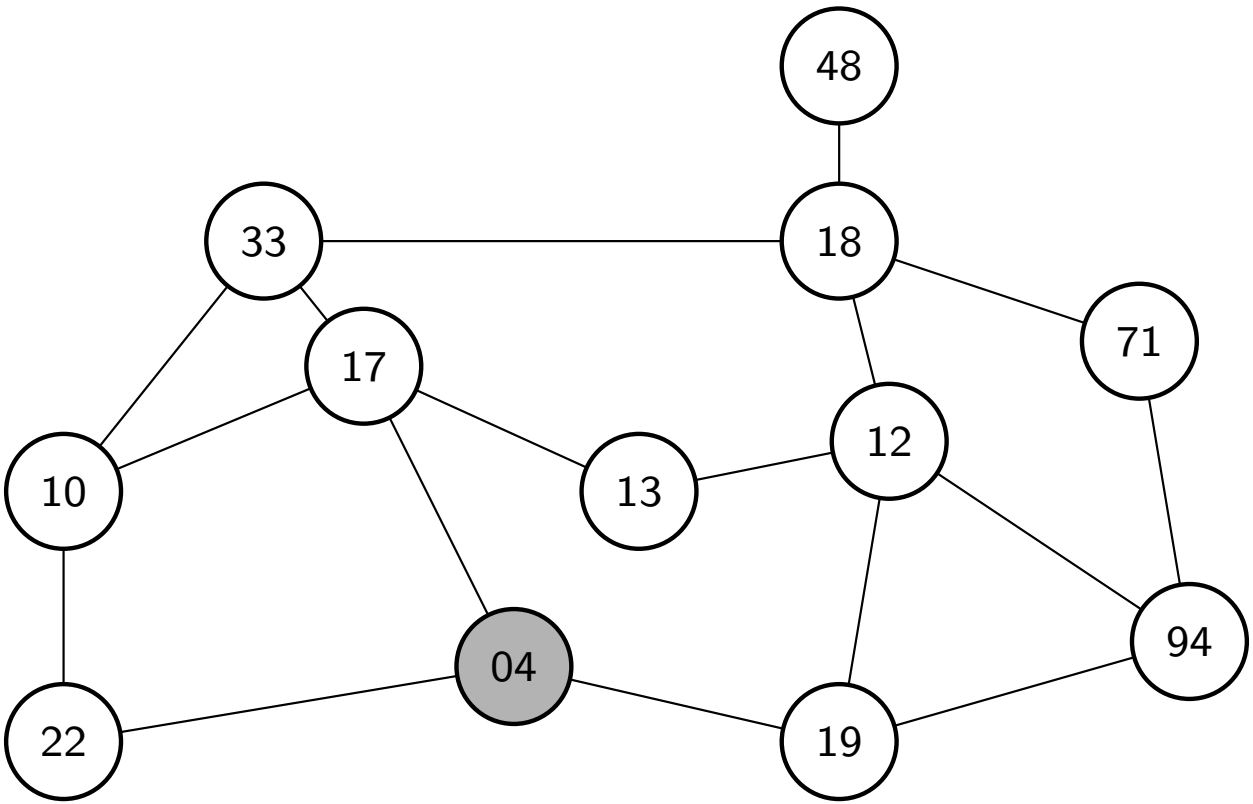
$(\Delta + 1)$-coloration



Spanning-tree construction



Leader Election



Performances

Definition : Space complexity

Number of bits per node

Parameters

- n **number of nodes**
- Δ **degree of the graph**

State of the art, Silent algorithms (1/2)

Memory requirements for silent stabilization : lower bound

[DolevGS99] prove that the leader election, the spanning tree construction, and the identification of the centers of a graph, require $\Omega(\log n)$ bits per edge.

Memory requirements for silent stabilization : upper bound

There exist algorithms for the leader election, the spanning tree construction, the identification of the centers of a graph, and the coloration, that use only $O(\log n)$ bits per node.

State of the art : Silent algorithms (2/2)

Definition : Proof Labelling Scheme (PLS) : [KormanKP07]

An **oracle**, assigning to every node v a label $l(v)$ based on its local state, and a **verifier**, a distributed predicate that, on node v, reads both the states and the labels of v and the label of its neighboors, such that

- for every legal state, the verifier returns true at each node
- for every illegal state, the verifier returns false at at least one node

Lower Bound : [BlinFP14]

If there exist a silent self-stabilizing algorithm using k bits, then there exist a PLS using at most k bits.

State of the art : General case

Best complexity achieved

[BlinT18] provide algorithm that require $O(\log\log n + \log\Delta)$ bits per node for the problems of $(\Delta+1)$-coloration, spanning tree, and leader election.

Lower Bound

[BeauquierGJ99] prove that the leader election cannot be solved with a constant number of bit per node.

Our result

Space complexity of the $(\Delta + 1)$-coloration

The space complexity of the $(\Delta + 1)$-coloring problem is $\Theta(\log \log n + \log \Delta)$ bits per node.

Space complexity of the spanning tree construction

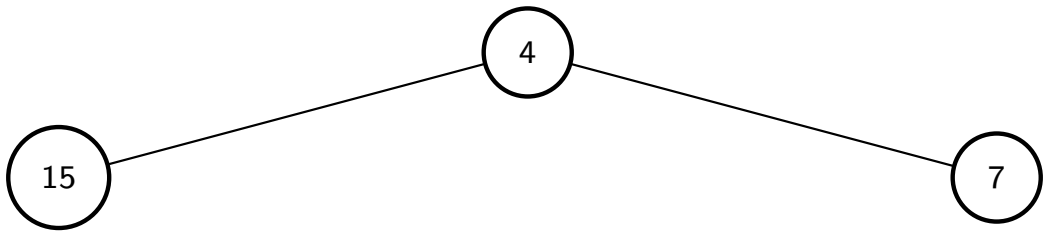
The space complexity of the spanning tree construction problem is $\Theta(\log \log n + \log \Delta)$ bits per node.

Space complexity of the leader election

The leader election problem requires $\Omega(\log \log n)$ bits per node.

Sketch of proof 1/2

We consider the n-nodes ring.



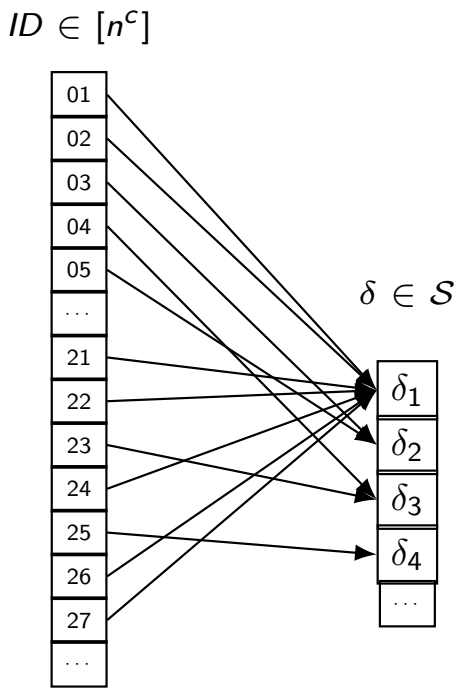
Idea

- Main algorithm : $\mathcal{A} : [n^c] \times \{0,1\}^{3f(n)} \rightarrow \{0,1\}^{f(n)}$
- $\forall v \in V$, with ID id_v, $\exists \delta_{id_v} : \{0,1\}^{3f(n)} \rightarrow \{0,1\}^{f(n)}$
- $\mathcal{S}$ denotes the set of all functions δ_i
- If $f(n) = o(\log \log n)$, then we can find n different IDs that match the same function

Sketch of proof 2/2

$$|\mathcal{S}| = 2^{f(n)\times 2^{3f(n)}}$$

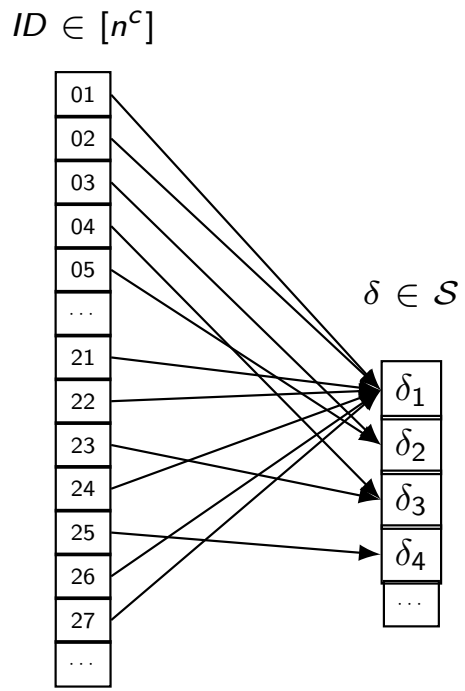
$$f(n) = o(\log\log n) \Rightarrow |\mathcal{S}| = o(n^{c-1})$$

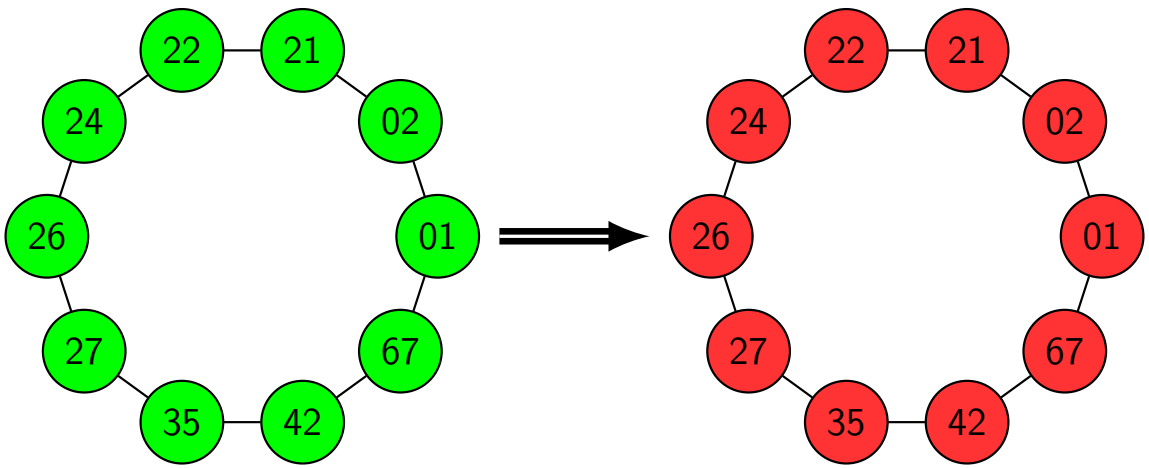


Sketch of proof 2/2

$$|\mathcal{S}| = 2^{f(n)\times 2^{3f(n)}}$$

$$f(n) = o(\log\log n) \Rightarrow |\mathcal{S}| = o(n^{c-1})$$





Open Problems

Message passing

Does the generic bound $\Omega(\log \log n)$ still holds in the message passing model ?

Thight bound for leader election ?

Does there exist an algorithm for the leader election that does not require the extra $O(\log \Delta)$ bits per node quantity ?