The Distributed Multiple Voting Problem

Nathanaël Courant et Noémie Fong

Table des matières

1 Introduction

1.1 Problème du vote multiple

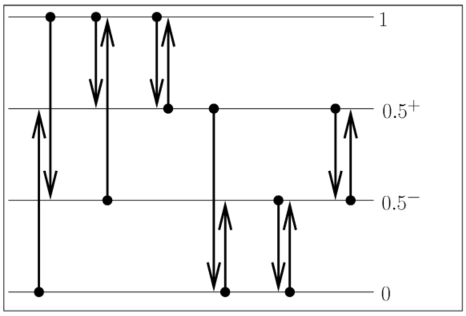
On considère un graphe d'agents pouvant localement interagir, uniquement avec leurs voisins. Chaque agent a une opinion initiale, et on souhaite déterminer l'opinion majoritaire. On désire de plus que les règles suivies par les agents soient constantes au cours du temps, identiques pour tous les agents, et simples. On considérera que les agents sont honnêtes et ne souhaitent pas influencer le résultat final. On appelera *couleurs* l'ensemble des opinions possibles des agents. Finalement, on se restreindra au cas où l'opinion majoritaire l'est strictement.

1.2 Les inconvénients des algorithmes gossip

Une solution classique à ce problème dans le cas de 2 couleurs est un algorithme de type *gossip*, qui converge vers la moyenne : on considère que les couleurs sont respectivement 0 et 1, et on veut savoir si la moyenne est supérieure ou non à $\frac{1}{2}$. Cette solution consiste en des mises à jour locales des moyennes deux-à-deux, où après chaque interaction d'un agent avec un de ses voisins, les deux se souviendront de la moyenne de leurs valeurs précédentes. Cependant, cette valeur moyenne ne peut pas être trouvée en temps fini par tous les nœuds, qui n'en connaissent qu'une valeur approchée. Comme notre objectif n'est pas de trouver la moyenne mais de connaître l'opinion majoritaire, on pourra alors décider du résultat en temps presque sûrement fini, mais au prix de stocker beaucoup d'information dans chaque nœud. Il est donc naturel de se demander si cette quantité peut être réduite, ne serait-ce qu'à un nombre fini de possibilités.

1.3 Le consensus par intervalles

À ces fins, l'idée sera de rechercher dans quel intervalle se trouve la moyenne des opinions, parmi $[0, \frac{1}{2}]$ et $[\frac{1}{2}, 1]$. De plus, ce raisonnement est aisément étendu dans le cas de plus de deux opinions en considérant des barycentres à la place de la moyenne et des ensembles convexes à la place des intervalles. La borne $\frac{1}{2}$ qu'on voit ici apparaître dans deux intervalles aura deux versions, une correspondant à chaque intervalle. L'algorithme met à jour les nœuds paire par paire. Les nouvelles valeurs de la paire dépendent des précédentes, et sont décrites par l'automate ci-dessous :



A chaque itération, on choisit au hasard une paire de nœuds adjacents. Cet algorithme conserve la moyenne. Il termine lorsque toutes les valeurs des noeuds sont dans le même intervalle, avec la borne $\frac{1}{2}$ du bon signe. Cet algorithme s'exécute en temps fini presque sûrement.

2 Des automates de vote

2.1 Construction générale

L'idée va être de paver l'espace des barycentres des différentes couleurs par des simplexes monocolores, pour lesquels chaque sommet correspond à un état. Dans le cas d'un sommet qui correspond à plusieurs couleurs majoritaires, il y aura un état pour ce sommet pour chacune de ces couleurs. Une opération entre deux sommets consistera à les remplacer par deux autres, de manière à ce que le barycentre soit préservé (et quelques autres contraintes concernant les couleurs dans le cas d'égalité). On va de plus demander qu'il existe un potentiel vérifiant que après chaque opération, le potentiel diminue strictement, sauf si les nœuds ont échangé leur place, auquel cas il reste identique. Enfin, on demande à ce que si les états présents ne sont pas concentrés comme les sommets d'un simplexe, il existe deux états vérifiant que le potentiel diminuera après leur rencontre. Toutes ces propriétés ensemble nous garantissent que presque sûrement, le barycentre final, égal au barycentre initial, sera dans un des simplexes que l'on aura délimité, qui sera monocolore : on aura donc trouvé le gagnant du vote.

2.2 Avantages

Un des premiers avantages des automates de vote en général sur un simple algorithme de type gossip est que la quantité d'information que les nœuds ont besoin de s'échanger est bornée. Par ailleurs, plus on augmente le nombre d'états intermédiaires, plus on a de nœuds à la fois actifs pour changer l'opinion des nœuds de l'autre type, pour un état initial fixé, ce qui accélère la convergence, surtout pour des graphes de diamètre élevé.

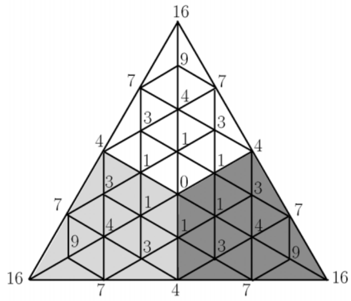
3 Quelques constructions

3.1 Automates de vote à 2 choix

L'automate pour 2 choix, même généralisé, se calcule aisément en forme close ; nous avons donc écrit un simple fragment de code générant la table de transition de l'automate en question. Pour un k donné, il a des états correspondant à $0, \frac{1}{2k}, \ldots, \frac{1}{2}^-, \frac{1}{2}^+, \ldots, \frac{2k-1}{2k}, 1$, les $k+1$ premiers de ceux-ci correspondant à un résultat en faveur de 0 et les $k+1$ autres en faveur de 1.

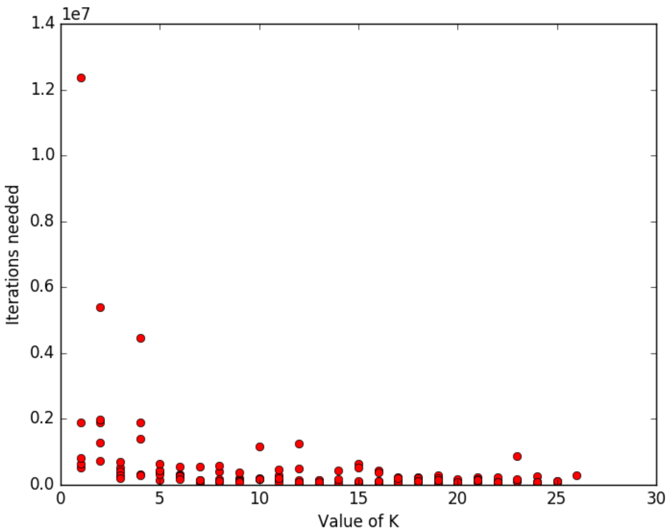
3.2 Automates de vote à 3 choix

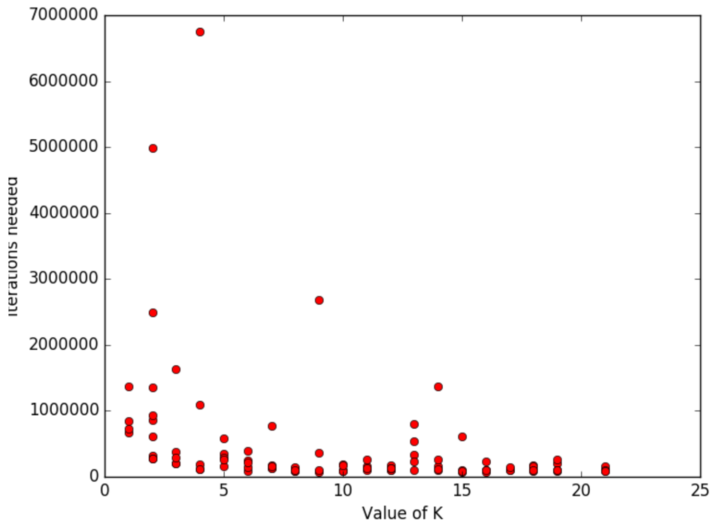
Pour ce qui est du cas de 3 couleurs, obtenir l'automate en forme fermée est plus difficile (même si ce n'est probablement pas impossible, mais contiendrait alors une disjonction de beaucoup de cas). Nous construisons donc l'automate en appliquant la méthode de l'article, la triangulation et le potentiel étant donnés (mais généralisés à une taille quelconque) par la figure suivante :



4 Queques statistiques

Nous avons fait tourner l'algorithme de vote sur plusieurs instances, tout d'abord dans le cas d'un vote à 2 choix, puis d'un vote à 3 choix. Dans les deux cas, nous avons utilisé des automates construits comme dans la section précédente, itérativement. De plus, on ajoute aléatoirement des arêtes entre les différents agents, afin de simuler un effet petit monde. La vitesse de convergence varie grandement entre l'ajout ou non de ces arêtes, ce qui est compréhensible : l'information n'est plus contrainte à se déplacer lentement d'une zone vers une autre, elle prend parfois des raccourcis. Les résultats des tests que nous avons effectués se trouvent dans les figures suivantes :





Références

[1] Florence Bénézit, Patrick Thiran, Martin Vetterli, *The Distributed Multiple Voting Problem*