Compromis temps-information pour l'élection de leader dans des arbres anonymes

Christian Glacet, Avery Miller, Andrzej Pelc

Compromis temps-information pour l'élection de leader dans des arbres anonymes †

Christian Glacet[1] ‡, Avery Miller[2] et Andrzej Pelc[3]

[1]*CNR - IEIIT, Turin, Italie,* [2]*Université de Tel-Aviv, Israël* [3]*Université du Québec en Outaouais, Gatineau, Canada*

Dans cet article nous étudions l'élection de leader distribuée dans des réseaux anonymes. Dans ce modèle l'ensemble des nœuds produisent en sortie un chemin simple vers l'un d'entre eux, ce dernier sera alors qualifié de leader. Nous étudions en particulier le cas de l'élection dans des arbres anonymes lorsque le temps alloué est inférieur au diamètre de l'arbre considéré. Si aucune information supplémentaire n'est accessible, les nœuds ne pouvant pas avoir une vision complète du réseau, l'élection est impossible et ce quel que soit l'algorithme utilisé. Nous étudions donc le compromis entre la quantité d'information initialement requise par les nœuds et le temps τ leur étant alloué pour réaliser l'élection. Nous utilisons le modèle des algorithmes avec conseil dans lequel un oracle ayant une connaissance complète du réseau, fournit un conseil unique sous forme d'une chaîne binaire, à l'ensemble des nœuds avant que ceux-ci n'initient leurs calculs.

Pour la majorité des valeurs de τ nos bornes inférieures et supérieures sont proches à un facteur multiplicatif près, ou diffèrent seulement d'un facteur logarithmique. Pour un arbre T de diamètre $diam \leq D$ et un temps $\tau = diam - 1$ la borne sur la quantité d'information requise (en nombre de bits) est $\Theta(\log D)$. En réduisant le temps d'une unité, i.e., pour $\tau = diam - 2$ la quantité d'information requise pour élire dépend de la parité du diamètre, elle est de $\Theta(\log D)$ lorsque $diam$ est pair et de $\Theta(\log n)$ sinon. Pour tout temps $\tau \in [\beta \cdot diam, diam - 3]$ avec $\beta > 1/2$, notre borne supérieure est $O(\frac{n \log n}{D})$ et notre borne inférieure de $\Omega(\frac{n}{D})$ à l'exception du cas où $diam$ est pair et τ est exactement $diam - 3$, cas pour lequel le problème reste ouvert. Enfin, pour des temps $\alpha \cdot diam$ avec $\alpha < 1/2$, nous montrons que l'information requise a une taille de $\Theta(n)$.

Mots-clefs : élection de leader, graphe anonyme, complexité, oracle

1 Introduction, Modèle et Résultats

Le problème de l'élection, très largement étudié depuis 1977 [LL77], est une des briques fondamentales de l'algorithmique distribué. Il vise, dans un réseau distribué, à ce que l'ensemble des nœuds s'accordent sur l'identité de l'un d'entre eux. Il n'est cependant pas toujours possible de fonder ces algorithmes sur l'existence d'étiquettes (ou noms) sur les nœuds. En effet, dans certains cas les nœuds ne possèdent pas d'identités, ne souhaitent pas les révéler, en changent régulièrement, ..., c'est pourquoi un modèle de calcul anonyme est intéressant à étudier. Un algorithme d'élection distribué dans un modèle anonyme demande à ce que l'ensemble des nœuds du réseau produisent en sortie un chemin simple vers un unique nœud. Ce problème plus précis a lui aussi été très largement étudié [AS91, YK96, BV99, ...].

Un des modèles classique de calcul distribué est le modèle $\mathcal{LOCAL}$. Dans ce modèle, pour un arbre T, après un temps r, tout nœud u peut obtenir une *vue* du réseau à distance r. Cette vue, $V_T(u,r)$, représente la connaissance maximale de la topologie du réseau qu'un nœud puisse obtenir dans ce laps de temps en échangeant des messages de façon locale uniquement. Cette vue est donc un sous arbre de T, comprenant l'ensemble des nœuds à distance au plus r de u, ainsi que tous les numéros de ports entrant et sortant de ces nœuds. Dans notre étude, nous considérons cette vue comme une donnée d'entrée, que chaque nœud peut utiliser pour calculer localement l'identité du leader. Nous étudions ici le cas de l'élection lorsque le temps imparti est inférieur au diamètre du graphe. Notons que sans information supplémentaire, il est impossible d'élire de façon distribuée et ce même dans un arbre.

†Cet article est une version courte de l'article *Time vs. Information Tradeoffs for Leader Election in Anonymous Trees* [GMP16]

‡Partiellement soutenu par le programme de bourse ERCIM et le projet ANR DISPLEXITY (ANR-11BS02-014).

Temps	**Taille du conseil**
$diam$	0
$diam-1$	$\Theta(\log D)$
$diam-2$	$\Theta(\log D)$ si $diam$ est pair $\Theta(\log n)$ si $diam$ est impair
$\beta \cdot diam \leq \tau \leq diam-3, \beta > \frac{1}{2}$ constant	$O(\frac{n \log n}{D})$ borne supérieure $\Omega(\frac{n}{D})$ borne inférieure si $diam$ impair ou $\tau < diam-3$
$\alpha \cdot diam, \alpha < \frac{1}{2}$ constant	$\Theta(n)$ pour $diam \in \omega(\log^2 n)$

FIGURE 1: Compromis entre temps et taille du conseil pour un graphe de n noeuds et de diamètre $diam \leq D$.

Notre objectif est donc d'étudier le compromis entre le temps maximum alloué à l'élection et la quantité d'information supplémentaire requise par les nœuds pour la réaliser. Nos résultats sont détaillés en Figure 1. La connaissance initiale des nœuds est donnée sous forme d'un conseil unique et à l'ensemble des nœuds par un *oracle* ayant une connaissance complète du graphe d'entrée T. Pour un temps imparti fixé, τ, tout nœud u peut calculer sa vue dans T à distance τ : $V_T(u,\tau)$ ainsi que le conseil donné par l'oracle. La décision du nœud u est basée uniquement sur ces deux informations.

Les bornes supérieures présentées en Figure 1 sont obtenues en proposant un couple conseil/algorithme permettant d'élire en temps au plus τ dans tout arbre de n nœuds et de diamètre D. Les bornes inférieures sont obtenues en exhibant une famille d'arbres de diamètre $diam \leq D$ pour lesquels l'élection est possible en temps $\leq \tau$ mais requiert une taille minimale pour le conseil fournit et ce quel que soit l'algorithme. Pour le cas particulier d'un diamètre pair et d'un temps exactement $diam-3$ un premier pas vers une solution peut être trouvé dans [GMP16], avec une famille d'arbres permettant d'obtenir une borne de $\Omega(n^{2/D}\log n)$.

Les deux sections suivantes présentent les intuitions et techniques utilisés pour obtenir les bornes supérieures puis inférieures obtenues. Nous montrons en particulier les intuitions des deux preuves pour un temps exactement $diam-2$ lorsque $diam$ est impair. Ces preuves montrent que $\Theta(\log n)$ bits d'information doivent être initialement fournis aux nœuds pour réaliser l'élection.

2 Bornes Supérieures

Les algorithmes que nous utilisons pour chacune des bornes supérieures présentées en Figure 1 sont basés sur l'idée d'élire le nœud central ou un des nœud composant l'arête centrale. Dans les deux cas le premier objectif est de permettre au nœuds de trouver l'élément central puis, pour les arbres de diamètre impair, de déterminer laquelle des deux extrémités de l'arête centrale doit être élue. L'algorithme et le conseil utilisés pour un temps $diam-2$ sont relativement simples mais permettent d'introduire les techniques de base utilisées dans les autres preuves.

Tout arbre T de diamètre impair est centré en une arête $e = \{c_0, c_1\}$, nous pouvons voir T comme l'union deux arbres T_0 et T_1 (ne contenant pas e) respectivement enracinés en c_0 et c_1 et connectés par l'arête e. L'idée générale du conseil que nous utilisons est de trouver une séquence de ports (sortant et entrant) apparaissant uniquement dans T_0 et d'utiliser cette séquence comme marqueur permettant ainsi aux nœuds de T de déterminer de quel coté de l'arête centrale e ils se trouvent. Notons que si un tel chemin n'existe pas alors l'élection est impossible en temps $diam-2$. En plus de ce chemin marqueur, nous ajoutons au conseil : l'entier q, port de sortie de l'arc (c_1, c_0).

L'algorithme utilisant ces conseils se base sur le fait que chaque nœud est capable de trouver le nœud le plus proche de lui appartenant à l'arête centrale, puis en utilisant le chemin "marqueur" il est capable de casser la symétrie et donc d'élire. Si un nœud v voit le chemin marqueur alors il se trouve dans T_0 et élit c_0 en produisant la séquence de ports de sortie du chemin $v \rightsquigarrow c_0$ (contenue dans sa vue). En revanche, si v se trouve dans T_1, il doit concaténer la séquence de ports de sortie du chemin $v \rightsquigarrow c_1$ au numéro de port q fournit par le conseil, produisant ainsi une séquence de ports menant de v à c_0 dans T.

Notons également que le chemin utilisé comme marqueur peut être compressé, en effet, les nœuds n'ont pas besoin de connaitre le chemin entier, mais simplement d'être capable de dire si oui ou non ce dernier appartient à leur vue. Cette information peut être compressé en ordonnant lexicographiquement deux listes, L_0 et L_1 des séquences de ports apparaissant respectivement dans T_0 et T_1. Une fois ces listes ordonnées

nous pouvons prendre la séquence d'indice j, première séquence de ports n'apparaissant pas dans T_1, ainsi que k l'indice du premier numéro de port, p, étant différent dans les séquences $L_0[j]$ et $L_1[j]$. Nous avons donc comme conseil le quadruplet (j,k,p,q) soit $O(\log n)$ bits ce qui conclu la preuve de la borne supérieur.

3 Bornes Inférieures

La technique que nous utilisons pour l'établissement de bornes inférieures est basée sur l'exhibition de familles d'arbres dans lesquelles un conseil différent doit être donné pour chaque arbre de la famille. Autrement dit, nous montrons que pour toute paire d'arbres dans une famille donnée, si le conseil donné par l'oracle est le même alors l'élection échouera dans un des deux arbres (au moins). Rappelons que dans certains graphes, intuitivement trop symétriques, l'élection en un temps donné est impossible quel que soit l'algorithme et la taille du conseil. Dans l'étude de bornes inférieures il est important de veiller à ce que l'élection soit possible dans la famille exhibée. Faute d'espace nous omettrons volontairement ces vérifications qui se résument à prouver qu'en connaissant la carte complète du réseau il est possible d'élire dans le temps imparti.

La preuve que nous utilisons pour un temps exactement $diam-2$ se construit en deux étapes, nous montrons d'abord une borne inférieure sur un jeu de décision que nous appelons *pair breaking*. Nous montrons ensuite une réduction du problème de l'élection dans une famille d'arbres (*balai double*) vers le problème de pair breaking pour obtenir la borne désirée.

Le problème pair breaking. Ce problème de décision peut être vu comme un jeu à deux joueurs anonymes ayant pour unique paramètre un entier Z. Notons X l'ensemble des paires d'entiers distincts dans $\{1,\dots,Z\}$, représentant chacune des instances de ce jeu. L'ensemble de ces instances est coloré en c couleurs par une fonction $\mathcal{C}: X \to \{1,\dots,c\}$. Pour une instance (a,b) donnée, chacun des joueurs connait soit a soit b, ainsi que la couleur attribuée à cette instance. Pour ces deux joueurs, l'objectif de ce jeu est de casser la symétrie en s'accordant sur l'identité de l'un d'eux en disant soit "moi" soit "lui/elle", et ce sans communiquer. Formellement ils doivent trouver une fonction $\mathcal{B}: \{1,\dots,Z\} \times \{1,\dots,c\} \to \{0,1\}$ telle qu'à $\mathcal{C}$ fixé et pour tout a,b l'inéquation $\mathcal{B}(a,\mathcal{C}(a,b)) \neq \mathcal{B}(b,\mathcal{C}(a,b))$ est vérifiée. La question que nous étudions sur ce jeu est : quelle est le nombre de couleurs minimum c pour lequel il existe une fonction de coloration $\mathcal{C}: X \to \{1,\dots,c\}$ permettant le succès des deux joueurs pour toute instance de X (sans restriction sur la fonction de décision $\mathcal{B}$ partagée par les joueurs). Nous obtenons le résultat suivant :

Lemme 1. *Quelle que soit la fonction de coloration $\mathcal{C}: X \to \{1,\dots,c\}$, le problème de pair breaking de paramètre Z nécessite $c \in \Omega(\sqrt{\log Z})$ couleurs.*

Démonstration. Notons X' l'ensemble des instances (a,b) d'un ensemble X donné telles que $a<b$. Remarquons que pour tout $a,b,c \in \{1,\dots,Z\}$, la fonction $\mathcal{C}$ utilise au moins deux couleurs. En effet, si $\mathcal{C}$ n'utilise qu'une seule couleur, γ, alors les deux joueurs échouent à la vérification d'une des inégalités $\mathcal{B}(a,\gamma) \neq \mathcal{B}(b,\gamma)$, $\mathcal{B}(b,\gamma) \neq \mathcal{B}(c,\gamma)$ ou $\mathcal{B}(a,\gamma) \neq \mathcal{B}(c,\gamma)$. Plus généralement, pour toute fonction de coloration $\mathcal{C}: X' \to \{1,\dots,c\}$ et tout ensemble $S = \{s_1,\dots,s_m\}$ il existe un ensemble $S_\alpha = \{x \mid (x,s_m) \in X' \text{ et } \mathcal{C}(x,s_m) = \alpha\}$ tel que $|S_\alpha| \geq \frac{|S|-1}{c}$ (principe des tiroirs). En utilisant la remarque sur les triplets de paires, nous pouvons conclure que l'existence de cet ensemble S_α va forcer une partie des instances $X_{\setminus\alpha}$ de X' a ne pas utiliser la couleur α, i.e., $X_{\setminus\alpha} = \{(x,y) \mid x<y \text{ et } x,y \in S_\alpha\}$. Ces nouvelles instances sont définies sur un ensemble $S_{\setminus\alpha}$ de taille $|S_\alpha|-1$. En utilisant cet argument de manière récursive sur les sous-ensembles d'instances successifs étant privés d'une couleur au moins, nous pouvons conclure que le nombre de couleurs nécessaires pour colorer correctement $X \supseteq X' = \{(a,b) \mid a,b \in S \text{ et } a<b\}$ est de $\Omega(\sqrt{\log |S|})$. □

Balai double de paramètre δ. Pour tout δ, la famille d'arbres que nous décrivons pour cette réduction contient un arbre de $m = D-4+2\delta(\delta+1)+1$ nœuds par instance $(a,b) \in X$ du problème de pair breaking de paramètre $Z = \delta^\delta$. Tout arbre T de la famille est constitué d'une chaîne de longueur $D-4$ dont les numéros de ports lus d'une extrémité à l'autre forment un palindrome $(0,0,1,1,0,0,\dots,1,1,0,0)$. Sont ensuite attachés aux extrémités de celle-ci, deux arbres de profondeur deux, T_a et T_b. L'arbre T_a est constitué d'un fils de degré a_0, ainsi que de δ fils, de degrés respectifs $a_1,\dots,a_\delta$ déterminés par une fonction bijective quelconque $f: \{1,\dots,Z\} \to \{1,\dots,\delta\}^\delta$. La valeur a_0 quant à elle est fixée de manière à obtenir un arbre

T de taille exactement m. Autrement dit $a_0 = (m-(D-4)-1)/2 - \delta - \sum_{i=1}^{\delta} a_i$. L'arbre T_b est définie de façon analogue. Notons DB_δ la fonction bijective traduisant toute instance de pair breaking appartenant à $X = \{(a,b) \mid a,b \in \{1,\ldots,\delta^\delta\}\})$ en l'arbre "balai double" de paramètre δ correspondant.

Réduction. Supposons désormais qu'il existe un algorithme ELECT, utilisant un oracle O et permettant d'élire dans la famille des arbres balai double de paramètre δ en temps $D-2$ et avec $o(\log\log Z), Z = \delta^\delta$, bits de conseil. Construisons maintenant deux fonctions : $\mathcal{C}$ et $\mathcal{B}$ qui permettent de résoudre problème de pair breaking en utilisant respectivement l'oracle O et l'algorithme ELECT. Pour une instance donnée $(x,y) \in X$ du problème, la fonction de coloration $\mathcal{C}(x,y)$ construit le balai double $DB_\delta(x,y)$, donne cet arbre à l'oracle O et récupère le conseil γ, qu'elle produit alors comme couleur de sortie. La fonction $\mathcal{B}(x,\gamma)$ effectue quant à elle les opérations suivantes (le second joueur effectue les opérations analogues pour $\mathcal{B}(y,\gamma)$) :

1. Inventer la partie de l'arbre inconnue. Par exemple en construisant l'arbre $T = DB_\delta((x,z))$ avec z le plus petit entier de $\{1,\ldots,Z\} \setminus \{x\}$ (T appartient à la famille des balais double de paramètre δ).
2. Utiliser $\texttt{ELECT}(T)$ avec le conseil γ pour trouver la position du nœud élu dans T.
3. Si le nœud élu est plus proche des feuilles de l'arbre T_x produire 1 en sortie, sinon produire 0.

Pour toute instance (a,b) du problème de pair breaking, d'après la définition de $\mathcal{B}$ et comme les balais double ont un diamètre impair, nous pouvons montrer que $\mathcal{B}(a,\gamma) \neq \mathcal{B}(b,\gamma)$ (le nœud élu est nécessairement non-équidistant des feuilles de T_a et T_b). Pour prouver ce point il est également important de remarquer que la vue à distance $D-2$ d'une feuille dans T_a est, pour tout $z \in \{1,\ldots,Z\} \setminus \{a\}$, la même dans $DB_\delta((a,z))$ que dans $DB_\delta((a,b))$ impliquant ainsi qu'à conseil fixé $\texttt{ELECT}(DB_\delta((a,z))) = \texttt{ELECT}(DB_\delta((a,b)))$ est toujours vérifiée. Autrement dit, la manière d'inventer la partie inconnue de l'arbre lors de la première étape de la fonction $\mathcal{B}$ n'a aucune influence sur l'issue de l'élection.

La fonction $\mathcal{B}$ permet donc bien de résoudre le problème de pair breaking en utilisant uniquement les $o(\log\log Z)$ bits de conseil fournis par l'oracle O. Une fonction de coloration $\mathcal{C}$ utilisant $o(\sqrt{\log Z})$ couleurs différentes permet donc pour résoudre le problème de pair breaking sur un ensemble de taille Z, ce qui est en contradiction avec le lemme 1, prouvant ainsi qu'un tel algorithme ELECT n'existe pas. Autrement dit $o(\log\log\delta^\delta)$ bits de conseil ne sont pas suffisant pour élire dans un arbre balai double satisfaisant $\delta(\delta+1) = (m-(D-4)-1)/2$ ($\delta \in \Theta(\sqrt{n})$). Nous pouvons donc conclure que l'élection dans un tel arbre requiert $\Omega(\log\log\delta^\delta)$ bits de conseil, soit $\Omega(\log\delta)$ ou encore $\Omega(\log n)$ ce qui conclu la preuve de la borne inférieure.

4 Problèmes Ouverts

Quelques problèmes restent encore ouverts sur le sujet étudié, en particulier, sont encore à trouver : (i) une borne inférieure pour $\tau = diam - 3$ et $diam$ pair [§] ; (ii) des bornes supérieure et inférieures pour un temps proche de la hauteur de l'arbre ($\tau = diam/2 \pm o(diam)$), de même pour des arbres de petit diamètre, i.e., lorsque $D \in O(\log^2 n)$; (iii) une borne inférieure pour un temps $\tau \in o(diam)$. Il pourrait d'abord sembler évident que la borne inférieure $\Omega(n)$ s'applique dans ce dernier cas, mais il s'avère qu'il est en fait difficile de construire une famille d'arbres assez grande et pour lesquels l'élection est possible en un temps si faible.

Références

[AS91] Hagit Attiya and Marc Snir. Better computing on the anonymous ring. *J. Algorithms*, 12(2) :204–238, 1991.

[BV99] Paolo Boldi and Sebastiano Vigna. Computing anonymously with arbitrary knowledge. In *Proceedings of the Eighteenth Annual ACM Symposium on Principles of Distributed Computing, PODC, '99Atlanta, Georgia, USA, May 3-6, 1999*, pages 181–188, 1999.

[GMP16] Christian Glacet, Avery Miller, and Andrzej Pelc. Time vs. information tradeoffs for leader election in anonymous trees. In *Proceedings of the Twenty-Seventh Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2016, Arlington, VA, USA, January 10-12, 2016*, pages 600–609, 2016.

[LL77] Gérard Le Lann. Distributed systems-towards a formal approach. In *IFIP Congress*, volume 7, pages 155–160. Toronto, 1977.

[Pel00] David Peleg. Distributed computing. *SIAM Monographs on discrete mathematics and applications*, 5, 2000.

[YK96] Masafumi Yamashita and Tsunehiko Kameda. Computing on anonymous networks : Part i-characterizing the solvable cases. *IEEE Trans. Parallel Distrib. Syst.*, 7(1) :69–89, 1996.

§. Nous présentons dans [GMP16] une borne de $\Omega(n^{2/D}\log n)$ pour ce problème