Le coût de la monotonie dans les stratégies d'encerclement réparti

David Ilcinkas, Nicolas Nisse, David Soguet

Le coût de la monotonie dans les stratégies d'encerclement réparti *

David Ilcinkas [1] and Nicolas Nisse [2†] and David Soguet [3]

[1] *CNRS, LaBRI, Université Bordeaux I, France*
[2] *DIM, Université du Chili, Chili*
[3] *LRI, Université Paris-Sud, Orsay, France*

L'encerclement dans les réseaux vise à réaliser le nettoyage, par une équipe d'agents mobiles, d'un réseau contaminé. La stratégie d'encerclement est calculée en temps réel, par les agents eux mêmes, et doit vérifier les trois propriétés suivantes : (1) *connexité* : la zone nettoyée doit toujours être connexe de façon à assurer des communications sécurisées entre les agents, (2) *monotonie* : la zone nettoyée ne doit jamais être recontaminée, ce qui permet un temps de nettoyage polynomial en la taille du réseau, et (3) *optimalité* : le nombre d'agents utilisés doit être le plus petit possible afin de minimiser la taille des ressources utilisées. Etant donné un graphe G, le plus petit nombre d'agents nécessaire pour nettoyer G de façon monotone connexe dans un contexte centralisé est noté $\mathbf{mcs}(G)$.

Plusieurs protocoles répartis ont été proposés pour résoudre le problème de l'encerclement dans les réseaux. Blin *et al.* ont proposé un algorithme distribué permettant à $\mathbf{mcs}(G)$ agents de déterminer et de réaliser une stratégie d'encerclement dans tout graphe inconnu G (inconnu signifie que les agents n'ont aucune connaissance *a priori* concernant le graphe) [AlgoTel'06]. Cependant, la stratégie réalisée n'est pas monotone et peut prendre un temps exponentiel. Nisse et Soguet ont prouvé que, pour résoudre le problème de l'encerclement dans les réseaux, il est nécessaire et suffisant de fournir $\Theta(n \log n)$ bits d'information aux agents par le biais d'un étiquetage des sommets du graphe [AlgoTel'07]. Ainsi, pour nettoyer un graphe inconnu de façon monotone et connexe, il est nécessaire d'utiliser plus d'agents que l'optimal. Dans cet article, nous étudions la proportion d'agents supplémentaires qui sont nécessaires et suffisants pour nettoyer de façon monotone connexe répartie tout graphe inconnu. Nous montrons que la contrainte de monotonie implique une augmentation drastique de ce nombre d'agents.

Nous prouvons que tout protocole distribué ayant pour but de nettoyer tout graphe inconnu de n sommets de façon monotone connexe répartie a un ratio compétitif de $\Theta(\frac{n}{\log n})$. Plus précisément, nous prouvons que pour tout protocole distribué $\mathcal{P}$, il existe une constante c tel que pour tout n suffisamment grand, il existe un graphe G de n sommets tel que $\mathcal{P}$ requiert au moins $c\frac{n}{\log n}\,\mathbf{mcs}(G)$ agents pour nettoyer G. De plus, nous proposons un protocole distribué qui permet à $O(\frac{n}{\log n})\,\mathbf{mcs}(G)$ agents de nettoyer tout graphe inconnu G de n sommets, de façon monotone connexe répartie.

Keywords: Stratégie d'encerclement, monotonie, ratio compétitif

1 Stratégies d'encerclement dans les graphes

L'*encerclement* dans les réseaux (*graph searching*) a été introduit par Parsons [Par76]. Etant donné un graphe dans lequel circule un fugitif invisible, arbitrairement rapide et omniscient, le but est de déterminer une stratégie qui permet à une équipe d'agents mobiles de capturer le fugitif. Dit autrement, l'équipe d'agents doit "nettoyer" un réseau contaminé, la partie contaminée du graphe étant la partie du graphe accessible au fugitif. Dans ce cadre, un graphe représente un réseau "contaminé" qu'une équipe d'*agents* mobiles doit nettoyer. Initialement, toutes les arêtes du graphe sont *contaminées*. Les agents sont situés sur les sommets du graphe et peuvent se déplacer le long des arêtes. Un agent *nettoie* une arête lorsqu'il la traverse. Cependant, une arête *e propre* est *recontaminée* si il existe un chemin entre *e* et une arête contaminée, tel qu'aucun agent n'occupe un sommet ou une arête de ce chemin. Une *stratégie d'encerclement*, ou plus simplement *stratégie*, est une séquence de mouvements des agents le long des arêtes, tel que, initialement,

[†] Nicolas Nisse a reçu le support du projet Anillo en Redes, ACT 07
* Une version étendue de cet article est parue dans les actes de OPODIS 2007, Springer LNCS 4878, pages 415-428.

tous les agents sont placés sur un sommet particulier du graphe, appelé la *base*. De plus, cette séquence de mouvements doit permettre de nettoyer le graphe, avec la contrainte qu'un agent peut bouger si et seulement si aucune arête n'est recontaminée à la suite de ce mouvement. Autrement dit, une arête propre reste toujours propre. Dans un contexte réparti, que nous considérons dans cet article, le *problème d'encerclement* consiste, étant donnés un graphe G connexe et une base $v_0 \in V(G)$, à déterminer une stratégie de nettoyage pour G qui utilise le moins d'agents possible, de telle sorte que la stratégie est calculée en temps réel par les agents eux mêmes. Dans la suite, $\mathbf{mcs}(G, v_0)$ dénote le nombre minimum d'agents nécessaire pour nettoyer G de façon monotone connexe en partant de $v_0 \in V(G)$, dans un contexte centralisé. Notons que dans un contexte centralisé, le problème de déterminer $\mathbf{mcs}(G, v_0)$ est NP-complet [MHG+88].

Les contraintes imposées sur les mouvements des agents impliquent deux propriétés importantes des stratégies : la monotonie et la connexité. Une stratégie est *monotone* si la partie propre du graphe ne peut que s'étendre, et une stratégie est dite *connexe* si la partie propre du graphe est toujours connexe. Les stratégies monotones connexes ont été fréquemment utilisées pour concevoir des protocoles distribués permettant de nettoyer un réseau [BFFS02, BFNV06, FLS06, FHL05, NS07]. En effet, les propriétés de monotonie et de connexité assurent respectivement un nettoyage du graphe en un nombre d'étapes polynomial en la taille du graphe, et des communications sûres entre les agents. La principale différence entre ces protocoles répartis est la quantité d'information sur le réseau dont les agents disposent *a priori*. Le protocole proposé par Blin *et al.* [BFNV06] permet de nettoyer tout réseau inconnu (i.e., les agents ne disposent pas *a priori* d'information sur le réseau). Ce protocole permet à $\mathbf{mcs}(G, v_0)$ agents de nettoyer tout graphe G, à partir de toute base $v_0 \in V(G)$, de manière connexe et répartie (un agent supplémentaire est nécessaire lorsque le réseau est asynchrone [FHL05]). Cependant la stratégie effectuée n'est pas monotone. Ceci est un inconvénient majeur puisque la stratégie calculée peut avoir un nombre d'étapes exponentiel en la taille du réseau. Pour permettre à $\mathbf{mcs}(G, v_0)$ agents de nettoyer tout graphe G, à partir de toute base $v_0 \in V(G)$, de façon monotone connexe et répartie, une méthode possible est de supposer que les agents disposent *a priori* de certaines informations à propos du graphe. Certains protocoles supposent que les agents disposent d'une information sur la topologie du réseau. Ainsi, les agents savent à l'avance quelle est la structure du réseau dans lequel ils se trouvent (arbres [BFFS02], tores [FLS06], hypercubes [FHL05]). Nisse et Soguet ont étudié la quantité d'information qu'il est nécessaire et suffisant de fournir aux agents pour résoudre le problème de l'encerclement. En utilisant la notion de *conseil* [FIP06] pour modéliser la quantité d'information fournie aux agents, Nisse et Soguet ont prouvé que pour permettre à $\mathbf{mcs}(G, v_0)$ agent de nettoyer tout graphe G, à partir de toute base $v_0 \in V(G)$, de façon monotone connexe et répartie, $\Theta(n \log n)$ bits d'information sont nécessaires et suffisants. Notons, que tous les protocoles précédemment énoncés impliquent des agents avec une mémoire de taille logarithmique en la taille du réseau. De plus, Blin *et al.* [BFNV06] supposent que les sommets disposent d'une zone locale de mémoire de taille polynomiale en la taille du réseau, accessible par les agents.

Pour résumé, deux méthodes ont été proposées pour résoudre le problème de l'encerclement : soit relâcher la contrainte de monotonie, soit fournir des informations aux agents. Dans cet article nous considérons une nouvelle méthode qui consiste à autoriser l'utilisation de plus d'agents que l'optimal. Plus précisément, nous considérons la proportion d'agents supplémentaires qui sont nécessaires et suffisants pour nettoyer de façon monotone connexe répartie tout graphe inconnu. Nous montrons que la contrainte de monotonie implique une augmentation drastique de ce nombre d'agents.

2 Modèle

Les agents sont modélisés par des entités mobiles autonomes, avec des identités et des mémoires distinctes. Le réseau est synchrone, et est modélisé par un graphe connexe non orienté. Le réseau est anonyme, i.e. les sommets ne sont pas étiquetés. Cependant, pour permettre à un agent de distinguer les différentes arêtes incidentes à un sommet, pour tout sommet u, les $\deg(u)$ arêtes incidentes à u sont étiquetées de 1 à $\deg(u)$. Ces étiquettes sont appelées *numéro de port*. De plus, un agent présent à un sommet peut laisser de l'information sur ce sommet. Pour cela, chaque sommet a une zone de mémoire locale, appelée *tableau blanc*, accessibles par les agents grâce à un processus d'exclusion mutuelle équitable, sur lequel un agent peut lire, effacer, et écrire des symboles.

Le nombre d'agents utilisés par un protocole $\mathcal{P}$ pour nettoyer un graphe G en partant d'une base $v_0 \in V(G)$, est le maximum, sur toutes les étapes d'exécution de $\mathcal{P}$, du nombre de d'agents placés sur les sommets de G. La qualité d'un protocole $\mathcal{P}$ pour nettoyer G en partant de v_0, est le nombre de d'agents utilisés par $\mathcal{P}$ pour nettoyer G, divisé par $mcs(G,v_0)$. Ce ratio, maximisé sur tous les graphes et toutes les bases, est appelé le *ratio compétitif* du protocole $\mathcal{P}$.

3 Nos résultats

Nous prouvons que tout protocole distribué visant à nettoyer tout graphe inconnu de n sommets de façon monotone connexe répartie a un ratio compétitif de $\Theta(\frac{n}{\log n})$. Plus précisément, nous prouvons que pour tout protocole distribué $\mathcal{P}$, il existe une constante c tel que pour tout n suffisamment grand, il existe un graphe G de n sommets et une base $v_0 \in V(G)$, tel que $\mathcal{P}$ requiert au moins $c\frac{n}{\log n}\,\mathbf{mcs}(G,v_0)$ agents pour nettoyer G de manière monotone connexe. De plus, nous proposons un protocole distribué qui permet à $O(\frac{n}{\log n})\,\mathbf{mcs}(G,v_0)$ agents de nettoyer tout graphe inconnu G de n sommets, de façon monotone connexe, en partant de toute base $v_0 \in V(G)$. Notons que notre protocole réalise le nettoyage des graphes de n sommets avec des agents ayant au plus $O(\log n)$ bits de mémoire, et des tableaux blancs de taille $O(n)$ bits. Nous prouvons tout d'abord le théorème suivant :

Théorème 1 *Tout protocole permettant de nettoyer tout graphe inconnu de n sommets de façon monotone connexe répartie a un ratio compétitif de* $\Omega(\frac{n}{\log n})$.

Idée de la preuve. Pour prouver ce théorème, nous considérons un jeu tour par tour entre un protocole arbitraire $\mathcal{P}$ et un adversaire $\mathcal{A}$. Le but de $\mathcal{P}$ est de nettoyer le graphe de façon monotone connexe en utilisant le moins d'agents possible. Le rôle de $\mathcal{A}$ est d'obliger $\mathcal{P}$ à utiliser le plus grand nombre possible d'agents pour nettoyer ce graphe. Pour cela, informellement, $\mathcal{A}$ construit graduellement un graphe de n sommets. Le graphe construit par $\mathcal{A}$ est en fait un arbre enraciné de degré maximum trois. Le fait que $\mathcal{A}$ puisse construire le graphe au cours de l'exécution de $\mathcal{P}$ est possible car les agents n'ont *a priori* aucune information concernant le graphe qu'ils nettoient. Pour expliquer cette construction, nous utilisons la définition suivante. Un *graphe partiel* est un graphe simple connexe qui peut avoir deux types d'arêtes : des *demi-arêtes* qui n'ont qu'une seule extrémité, et des *arêtes entières* qui ont deux extrémités. Informellement, dans la suite, les demi-arêtes dénotent des arêtes du graphe non encore explorées par $\mathcal{P}$. De plus, si G est un graphe partiel, alors G^+ dénote le graphe obtenu en rajoutant une extrémité de degré un à toute demi-arête de G.

Soit $n \geq 5$. Nous considérons un arbre enraciné inconnu de n sommets et de degré maximum trois T, que $\mathcal{P}$ doit nettoyer en partant de $v_0 \in V(T)$. A chaque tour p, T_p correspond à la partie de T qui est actuellement connue par $\mathcal{P}$. Décrivons le jeu entre $\mathcal{P}$ et $\mathcal{A}$. Initialement, le graphe partiel T_p consiste en un sommet unique, la base v_0, incident à trois demi-arêtes, tel que tous les agents sont placés sur v_0. Alors, $\mathcal{P}$ et $\mathcal{A}$ jouent alternativement, en commençant par $\mathcal{P}$. $\mathcal{P}$ choisit un agent et il déplace cet agent le long d'une arête e de T_p. Un tel déplacement est toujours possible puisque $\mathcal{P}$ est un protocole de nettoyage connexe monotone réparti, et donc, il finit toujours par nettoyer T. Notons que e peut être une demi-arête ou une arête entière. Si e est une arête entière, alors $\mathcal{A}$ passe son tour. Sinon, deux cas sont possibles. Soit $|V(T_p^+)| < n-1$, ou $|V(T_p^+)| = n-1$. Dans le premier cas, $\mathcal{A}$ ajoute une nouvelle extrémité v à e tel que v est un sommet incident à deux nouvelles demi-arêtes. Dans le deuxième cas, $\mathcal{A}$ ajoute une nouvelle extrémité v à e tel que v est un sommet incident à seulement une nouvelle demi-arête. De nouveau, ceci est possible puisque $\mathcal{P}$ ne connaît pas à l'avance le graphe que les agents doivent nettoyer. Le jeu termine quand $|V(T_p^+)| = n$. A un tel tour, $\mathcal{A}$ décide que le graphe T est en fait T_p^+.

En utilisant cette construction, la preuve du théorème consiste tout d'abord à prouver par récurrence que T a au moins $(n+2)/2$ feuilles, et que, pour éviter la recontamination, il existe un tour où chaque parent d'une feuille de T est occupé par un agent. Or, pour $n \geq 5$, chaque sommet est parent de au plus deux feuilles. Nous en déduisons que, $\mathcal{P}$ utilise au moins $k \geq n/4$ agents pour nettoyer T. De plus, d'après [MHG+88, BFST03], $\mathbf{mcs}(T,v_0) \leq 2(1+\log_3(n-1))$. On en déduit que $k \geq \mathbf{mcs}(T,v_0) \times \frac{n}{8(1+\log_3(n-1))}$. □

Nous prouvons ensuite que la borne inférieure du théorème précédent est optimale :

Théorème 2 *Il existe un protocole qui nettoie tout graphe inconnu de n sommets, de façon monotone connexe répartie, avec un ratio compétitif de* $O(\frac{n}{\log n})$.

Idée de la preuve. Pour prouver ce théorème, nous proposons un protocole qui nettoie tout graphe inconnu de n sommets de façon monotone connexe répartie, avec un ratio compétitif de $O(\frac{n}{\log n})$. Son fonctionnement, ainsi que la preuve du théorème, utilisent la notion de *mineur* de graphe. Nous rappelons qu'un graphe H est un mineur d'un graphe G si H est obtenu par une succession de contraction ou de suppression d'arêtes de G. Les caractéristiques principales de notre protocole est de maintenir dynamiquement un arbre enraciné S, tel que S est un arbre de degré au plus trois, et S est un mineur de la partie propre de G. De plus, à chaque étape le protocole nettoie une arête de G tel que S soit le plus proche possible d'un arbre enraciné de degré trois tel que toutes les feuilles soient à égale distance de la racine. Soit T_k un tel arbre de hauteur k. Nous prouvons que ce protocole est tel que, (1) à chaque étape $V(S)$ est l'ensemble des sommets de G occupés par un agent à cette étape, et (2) S est de profondeur $k \geq 1$ si et seulement si il existe une étape précédente où S était un arbre T_{k-1}. Ainsi, si k est la hauteur maximum de S, alors d'après (1) nous déduisons que le protocole utilise au plus $N = |V(T_k)| = \frac{|V(T_k)|}{\log |V(T_k)|} \times \log |V(T_k)|$ agents. Nous omettons les détails par manque de place mais en utilisant (2) et des résultats des articles [MHG+88, BFST03] nous prouvons ensuite que $\log |V(T_k)| \leq O(mcs(G, v_0))$. Finalement, puisque la fonction $\frac{x}{\log x}$ est strictement croissante et $|V(T_k)| = 3|V(T_{k-1})| + 1 \leq 3|V(G)| + 1 = 3n + 1$, nous obtenons que $N = O(\frac{n}{\log n} \times \mathbf{mcs}(G, v_0))$, ce qui conclut la preuve du théorème. □

Références

[BFFS02] L. Barrière, P. Flocchini, P. Fraigniaud, and N. Santoro. Capture of an intruder by mobile agents. In 14th ACM Symp. on Parallel Algorithms and Architectures (SPAA), pages 200-209, 2002.

[BFST03] L. Barrière, P. Fraigniaud, N. Santoro, and D. Thilikos. Connected and Internal Graph Searching. In 29th Workshop on Graph Theoretic Concepts in Computer Science (WG), Springer-Verlag, LNCS 2880, pages 34–45, 2003.

[BFNV06] L. Blin, P. Fraigniaud, N. Nisse and S. Vial. Distributing Chasing of Network Intruders. In 13th Colloquium on Structural Information and Communication Complexity (SIROCCO 2006), Springer-Verlag, LNCS 4056, pages 70-84.2006.†

[FLS06] P. Flocchini, F.L. Luccio, and L. Song. Decontamination of chordal rings and tori. Proc. of 8th Workshop on Advances in Parallel and Distributed Computational Models (APDCM), 2006.

[FHL05] P. Flocchini, M. J. Huang, F.L. Luccio. Contiguous search in the hypercube for capturing an intruder. Proc. of 18th IEEE Int. Parallel and Distributed Processing Symp. (IPDPS), 2005.

[FIP06] P. Fraigniaud, D. Ilcinkas and A. Pelc. Oracle Size : a New Measure of Difficulty for Communication Tasks. In 25th Annual ACM Symp. on Principles of Distributed Computing (PODC), pages 179-187, 2006.†

[MHG+88] N. Megiddo, S. Hakimi, M. Garey, D. Johnson and C. Papadimitriou. The complexity of searching a graph. Journal of the ACM 35(1), pages 18-44, 1988.

[NS07] N. Nisse and D. Soguet. Graph searching with advice. In 14th Colloquium on Structural Information and Communication Complexity (SIROCCO), Springer-Verlag, LNCS 4474, pages 51-67, 2007.†

[Par76] T. Parsons. Pursuit-evasion in a graph. Theory and Applications of Graphs, Lecture Notes in Mathematics, Springer-Verlag, pages 426-441, 1976.

† Ces articles ont donné lieu à des versions françaises publiées dans les actes de Algotel 2006 et 2007.