Explorer un anneau avec des robots amnésiques et myopes

Ajoy K. Datta, Anissa Lamani, Lawrence Larmore, Franck Petit

Explorer un anneau avec des robots amnésiques et myopes †

Ajoy K. Datta[1], Anissa Lamani[2], Lawrence L. Larmore[1], and Franck Petit[3]

[1] *School of Computer Science, University of Nevada Las Vegas*
[2] *University of Fukuoka, Japan*
[3] *LIP6 UMR 7606, Université Pierre et Marie Curie, Paris 6 Sorbonne Universités, INRIA*

Nous nous intéressons au problème de l'exploration d'un anneau avec arrêt avec k robots mobiles, identiques, amnésiques, dotés de capteurs leur permettant de percevoir leur environnement, mais incapables de communiquer explicitement. Nous considérons que les robots sont également myopes, c'est-à-dire qu'il ne peuvent pas voir au delà d'une certaine distance fixée ϕ. Nous montrons que si $\phi = 1$, l'exploration avec arrêt déterministe n'est possible qu'à la condition que le système soit synchrone. De plus, nous apportons des solutions synchrones déterministes qui sont optimales en nombre de robots.

1 Introduction

Nous considérons des cohortes de k robots dotés d'actionneurs de mouvements et de capteurs. Les robots évoluent par *cycles* qui comprennent chacun une phase de *captation* de l'environnement (également appelée phase de *vision*), suivie d'une phase de *calcul*, puis de *déplacement*. Lors d'un cycle, un robot capte (observe) tout d'abord son environnement (phase de vision). Puis, basé sur cette observation, le robot décide de se déplacer vers une destination ou de rester sur place (phase de calcul). Enfin, quand il décide d'un déplacement, le robot se déplace vers sa destination (phase de déplacement). Nous supposons que les robots se déplacent dans un environnement constitué d'un ensemble fini de lieux reliés entre eux par des passages destinés à la circulation. On représente naturellement ce genre d'environnement par un graphe à n sommets (ou nœuds) représentant des lieux de l'environnement et les arêtes la possibilité pour un robot de se déplacer d'un lieu à un autre.

Nous nous intéressons au problème de l'*exploration avec arrêt* consistant à faire en sorte que les k robots initialement situés dans des lieux différents puissent explorer entièrement un environnement donné, ceci avant de stopper tout déplacement. Ce problème entre dans la résolution de nombreuses applications dévolues à des cohortes de robots. Elles sont surtout associées à la sureté ou la sécurité, par exemple la surveillance, l'exploration, la reconnaissance ou le sauvetage. Cependant, certaines exigences environnementales, économiques ou technologiques peuvent amener à minimiser le nombre de robots mises à disposition dans l'accomplissement de cette tâche. Il s'avère donc important de connaître le nombre minimum de robots requis pour accomplir l'exploration d'un environnement donné. De plus, même en connaissant cette borne inférieure, il se peut que les capacités sensorielles des robots soient insuffisantes, par exemple, être ou ne pas être doté d'une boussole, d'un mécanisme de localisation précis dans un repair global, d'un moyen pour communiquer explicitement, etc. D'ailleurs, dans des cas extrêmes, même bien équipés, les robots peuvent se retrouver dans l'incapacité d'utiliser certains de leurs capteurs parce qu'ils évoluent dans une zone où leurs capacités sont limitées, voire inopérantes. Ceci nous incitent à développer des algorithmes d'exploration qui nécessitent le moins de robots possible, eux-mêmes utilisant des capacités sensorielles les plus faibles possibles.

Nous supposons donc que les robots ont des capacités très faibles : ils n'ont aucun moyen de communication explicite leur permettant de communiquer entre eux ou à une autorité ; ils sont *anonymes* (ils n'ont

† Cet article est un résumé étendu de [**?**].

aucun moyen de se distinguer l'un de l'autre), *uniformes* (ils exécutent tous le même protocole), *amnésiques* (leur mémoire volatile s'efface entre deux cycles), et ne savent pas s'orienter, ni s'accorder sur une direction ou une orientation commune.

L'exploration avec arrêt a fait l'objet de nombreuses publications. On peut notamment citer celles présentées à l'occasion de sessions d'Algotel, notamment [**?**, **?**] sur l'anneau et [**?**] sur l'arbre. A propos de l'anneau, dans [**?**], les auteurs montrent que $\Omega(n)$ robots sont nécessaires pour résoudre le problème de manière déterministe. Dans des travaux complémentaires [**?**], les mêmes auteurs abordent le problème lorsque le graphe est un anneau et prouvent que $\Theta(\log n)$ robots sont nécessaires et suffisants, toujours dans le cas déterministe. Ils donnent également un algorithme qui suppose que le nombre k de robots ne divise pas la taille n de l'anneau. Les auteurs de [**?**] proposent d'adopter une approche *probabiliste* afin de lever les conditions nécessaires et obtenir de meilleures bornes. Ainsi, contrairement à l'approche déterministe, ils suppriment la contrainte de divisibilité entre k et n et montrent que *quatre* robots probabilistes identiques sont nécessaires et suffisants pour résoudre l'exploration avec arrêt d'un anneau anonyme de taille $n \geq 9$.

Contribution. Dans ce papier, nous restreignons encore les capacités des robots en supposant qu'ils sont *myopes*, c'est-à-dire que leur capacité visuelle est réduite à une certaine distance ϕ. Nous nous intéressons aux cas de myopies les plus sévères, à savoir qu'un robot ne peut pas voir au delà des nœuds adjacents au nœud sur lequel il se trouve, c'est-à-dire $\phi = 1$. Par comparaison, tous les travaux traitant de l'exploration jusqu'alors supposaient que les robots disposent d'une capacité de vision au moins égale à l'ensemble des nœuds du réseaux, en d'autres termes, que $\phi = +\infty$.

Notre contribution est triple. Nous montrons tout d'abord qu'avec de telles hypothèses, l'exploration avec arrêt dans un anneau est possible de manière déterministe dans un environnement asynchrone si et seulement si l'anneau est réduit à un seul nœud. Nous nous intéressons ensuite au cas d'un environnement synchrone. Nous prouvons que lorsque $n > 6$, il n'existe pas non plus un tel algorithme fonctionnant avec moins de 5 robots. Enfin, nous donnons deux algorithmes déterministes optimaux en nombre de robots qui résolvent le problème dans les deux cas suivants : $3 \leq n \leq 6$ et $n > 6$.

2 Modèle

Nous considérons une cohorte de k robots évoluant sur un anneau de n nœuds (avec $k \leq n$), $G = u_0, \ldots, u_{n-1}$, c'est-à-dire que u_i est connecté à u_{i-1} et à u_{i+1} (les calculs sur les indices s'effectuant modulo n). Les indices sont utilisés uniquement à des fins de notation : les nœuds sont *anonymes* et l'anneau n'est *pas orienté*, *i.e.*, étant donnés deux nœuds voisins u et v, il n'y a aucun étiquetage explicite ou implicite qui permette de déterminer que u est à la droite ou à la gauche de v. Les robots sont anonymes, semi-synchrone, amnésiques et uniformes.

Zéro, un ou plusieurs robots peuvent être situés sur un nœud. Etant donné un nœud u_i et un instant t, ce nombre à l'instant t est appelé la *multiplicité* de u_i à l'instant t, noté $M_i(t)$ ou simplement M_i lorsqu'il n'y a pas d'ambigüité à propos de t. Le nœud u_i est dit *libre* à l'instant t si $M_i(t) = 0$ et *occupé* sinon. Lorsque $M_i(t) > 1$, on dit qu'il y a une $M_i(t)$.*tour* (ou simplement une *tour*) sur u_i à l'instant t. Une *configuration* à un instant t est le mot $\gamma_t = M_0(t), \ldots, M_i(t), \ldots, M_{n-1}(t)$. On appelle configuration *initiale*, notée γ_0, la configuration à l'instant $t = 0$. Etant donné deux instants distincts t et t' ($t \neq t'$), γ_t et $\gamma_{t'}$ sont dites *indiscernables* ssi il existe un automophisme f sur G tel que $\forall i \in \{1, \ldots, n\}$, $M_i = M_j$ avec $v_j = f(v_i)$. Un *block B* est une séquence maximale élémentaire de nœuds $u_i \ldots u_{i+l}$ ($l \geq 0$) telle que chaque nœud contient au moins un robot. La *taille* et la *longueur* d'un block B sont respectivement le nombre de robots et le nombre de nœuds que B inclut. Lorsque la taille et la longueur sont toutes deux égales à 1, l'unique robot du block est dit *isolé*.

Chaque robot est équipé d'un capteur abstrait lui permettant de mesurer la multiplicité. Puisque nous supposons que l'anneau n'est pas orienté et que $\phi = 1$, la *vue* d'un robot r situé sur le nœud u_i est une séquence s de 3 entiers x_{-1}, x_0, x_1 telle que $x_0 = M_i$ et soit $x_{-1} = M_{i-1}$ et $x_1 = M_{i+1}$, soit $x_{-1} = M_{i+1}$ et $x_1 = M_{i-1}$. Un algorithme se présente sous la forme d'un ensemble de règles de la forme : <*Etiquette*> <*Garde*> *: :* <*déplacement*>. La garde est la vue $s = x_{-1}, x_0, x_1$ captée et localement orientée par r. Le déplacement indique la direction du mouvement de r vis-à-vis de son orientation locale, à savoir : (i) $\rightarrow$, r se déplacement vers u_{i+1}, (ii) $\leftarrow$, r se déplacement vers u_{i-1}. Lorsque la vue est symétrique et que l'algorithme

impose à r de se déplacer, nous notons le déplacement de la sorte : $\leftarrow \vee \rightarrow$. Nous considérons alors que le choix de l'arête traversée est décidé par un adversaire.

Nous supposons un ordonnanceur *distribué* (à chaque instant, tout sous-ensemble non vide de robots peut être activé) et *équitable* (tout robot est activé infiniment souvent dans une exécution). Pendant la phase de vision, il se peut que les deux arêtes adjacentes à un nœud v occupé par un robot semblent identiques, c'est-à-dire que la vue de v est symétrique. Dans ce cas, si le robot décide de bouger, il peut traverser l'une ou l'autre des arêtes : nous considérons le pire cas où le choix de l'arête traversée est arbitrairement décidé par l'ordonnanceur.

3 Impossibilité

Système Semi-Synchrone. L'exploration est clairement triviale lorsque $n = k$. Dans la suite de cette section, nous admettons que $0 < k < n$ et $\phi = 1$. De plus, nous supposons que chaque robot connaît k et n.

Soit Π, la classe des algorithmes déterministes semi-synchrones qui résolvent le problème de l'exploration avec un ordonnanceur équitable, $\phi = 1$ et $2 \leq k < n$. Puisqu'aucun robot ne peut voir plus loin qu'à distance 1, il n'y a que quatre possibilités de règles pour Π dans la configuration initiale (γ_0) :

$\mathcal{R}_{\texttt{sgl}}$: $0(1)0 :: \rightarrow \vee \leftarrow$ $\quad$ $\mathcal{R}_{\texttt{out}}$: $0(1)1 :: \leftarrow$ $\quad$ $\mathcal{R}_{\texttt{in}}$: $0(1)1 :: \rightarrow$ $\quad$ $\mathcal{R}_{\texttt{swp}}$: $1(1)1 :: \rightarrow \vee \leftarrow$

Lemma 1 *Pour tout k tel que $2 \leq k < n$, il n'existe pas d'algorithme $\mathcal{P} \in \Pi$ permettant que tout robot soit initialement isolé.*

Puisque tous les robots sont isolés, chacun d'entre eux à la même vue et ne peut exécuter que $\mathcal{R}_{\texttt{sgl}}$. En les faisant tous exécuter $\mathcal{R}_{\texttt{sgl}}$, l'adversaire amène le système dans une configuration γ_1 indiscernable de γ_0. Le lemma suivant se montre par élimination de règles mutuellement exclusives :

Lemma 2 *Pour tout $\mathcal{P} \in \Pi$, $\mathcal{P}$ contient la règle $\mathcal{R}_{\texttt{in}}$ et ne contient ni $\mathcal{R}_{\texttt{swp}}$ ni $\mathcal{R}_{\texttt{out}}$.*

Etant donnée une configuration γ_t, S^x.séquence est un sous-mot $\gamma_t = M_i(t), \ldots, M_{i+j}(t)$ de γ_t tel que (i) $j \geq 0$, (ii) $M_{i+j}(t) > 1$ (aucun nœud de la séquence est inoccupé) et (iii) $M_i(t) = M_{i+j}(t) = x$, c'est-à-dire que les deux extrémités de la séquence sont constituées d'une x.tour. Le lemme suivant découle directement des lemmes **??** et **??** :

Lemma 3 *Pour tout $\mathcal{P} \in \Pi$, pour tout k ($5 \leq k < n$), il existe des exécutions de $\mathcal{P}$ contenant des S^2-sequences, des nœuds isolés et des blocks de taille strictement inférieure à 5.*

Considérons les robots situés sur le bord d'un block. Soit $x \geq 1$, le nombre de robots situés sur l'un de ces bords et la liste de règles génériques suivantes :

$\mathcal{T}\alpha(x)$: $0(x)1 :: \leftarrow$ $\quad$ $\mathcal{T}\beta(x)$: $0(x)1 :: \rightarrow$

$\mathcal{T}\gamma(x)$: $x(1)1 :: \leftarrow$ $\quad$ $\mathcal{T}\delta(x)$: $x(1)1 :: \rightarrow$

Nous définissons également la règle $\mathcal{T}_{\texttt{sgl}}(y)$ ($y \geq 2$) ainsi : $0(y)0 ::\leftarrow \vee \rightarrow$. En considérant toutes suffix d'exécutions démarrant d'une configuration contenant des S^2-séquences :

Lemma 4 *Pour tout $\mathcal{P} \in \Pi$, pour tout $x \geq 1$, $\mathcal{P}$ contient uniquement la règle $\mathcal{T}\beta(x)$.*

Il s'ensuit que $\mathcal{P}$ ne peut pas assurer la progression et la terminaison simultanément. D'où :

Theorem 1 ($\Pi = \emptyset$) *Il n'existe pas d'algorithme d'exploration déterministe dans le modèle semi-synchrone lorsque $\phi = 1$, $n > 1$, $1 \leq k < n$.*

La conséquence directe de ce théorème est qu'il n'existe pas non plus d'algorithme d'exploration déterministe dans le modèle asynchrone.

Système synchrone. Clairement, le lemme **??** est encore vrai dans le modèle synchrone. Donc, dans la configuration initiale γ_0, il existe au moins un block dans la taille est plus grande ou égale à 2. Le théorème suivant se montre facilement par contradiction :

Theorem 2 *Soit $\mathcal{P}$, un algorithme synchrone d'exploration avec $\phi = 1$ et $2 \leq k < n$. Si $n \geq 7$, alors $k \geq 5$.*

4 Algorithmes synchrones

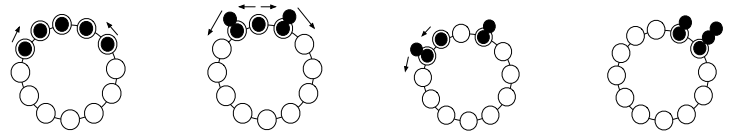
Nous présentons deux algorithmes synchrones. Le premier est un algorithme général qui utilise 5 robots et fonctionne sur tout anneau de taille $n \geq 7$. Il est facile de montrer que pour fonctionner avec seulement 5 robots, il faut que la configuration initiale contienne un seul block. Le second est fait pour les petits anneaux de taille $3 \leq n \leq 6$ et qui utilise $(n-1)$ robots, excepté le cas où $n = 6$ qui ne requière que 4 robots seulement.

Algorithme général pour $k = 5$ et $n \geq 7$. L'algorithme est présenté figure 1. En premier lieu, seuls les robots situés sur un bord de block sont autorisés à bouger dans γ_0. Puisque nous sommes dans un environnement synchrone, tous les robots bougent sur le nœud occupé adjacent en exécutant la règle 1*A*1—voir figure **??**, cas (*a*). La configuration suivante contient donc un robot seul entouré de deux tours.

1A1 :	0(1)1	::	$\rightarrow$	*// Vers un voisin occupé*
1A2 :	2(1)2	::	$\rightarrow \vee \leftarrow$	*// Vers l'un de mes voisins*
1A3 :	0(2)1	::	$\leftarrow$	*// Vers un voisin libre*
2A4 :	2(1)0	::	$\leftarrow$	*// Vers la tour*

FIGURE 1: Exploration synchrone pour $n \geq 7$

A partir de là, chaque tour bouge dans la direction opposée au robot seul (règle 1*A*3), alors que le robot seul bouge vers l'une des deux tours—règle 1*A*2, figure **??**, (*b*). Notons que la configuration qui résulte de ces mouvements permet d'orienter l'anneau. Ensuite, la tour seule ne bouge plus. Elle va permettre de détecter la terminaison lorsque les trois autres robots l'atteindront après $n-4$ étapes grace aux règles 1*A*3 et 2*A*4. Finalement, en exécutant la règle 2*A*4 une dernière fois, le robot seul crée une 3.tour—figure **??**, (*d*), marquant ainsi la fin de l'exploration.



(a) *(b)* *(c)* *(d)*

FIGURE 2: Exploration synchrone pour $3 \leq n \leq 6$

Algorithme spécifique pour les cas $3 \leq n \leq 6$. La figure 3 présente le programme des robots. Le même algorithme fonctionne pour le cas $n = 6$ et $k = 4$. Les exécutions respectives pour les différents cas de $n \in [3,6]$ sont donnés dans la figure **??**.

1A'1 :	0(1)1	::	$\leftarrow$	*// Vers le voisin libre*
1A'2 :	0(1)0	::	$\leftarrow \vee \rightarrow$	*// Vers un voisin*

FIGURE 3: Exploration synchrone pour $3 \leq n \leq 6$

$n = 3,\ k = 2.$ $n = 4,\ k = 3.$ $n = 5,\ k = 4.$ $n = 6,\ k = 4.$

FIGURE 4: Exploration synchrone pour $3 \leq n \leq 6$.