Approche asynchrone dans le plan : un algorithme déterministe polynomial

Sébastien Bouchard, Marjorie Bournat, Yoann Dieudonné, Swan Dubois, Franck Petit

Approche asynchrone dans le plan : un algorithme déterministe polynomial †

Sébastien Bouchard[1], Marjorie Bournat[1], Yoann Dieudonné[2], Swan Dubois[1] et Franck Petit[1]

[1] *Sorbonne Université, CNRS, INRIA, Laboratoire d'Informatique de Paris 6, LIP6, DELYS, F-75005 Paris, France*
[2] *Université de Picardie Jules Verne (UPJV), Laboratoire Modélisation Information et Systèmes (MIS), France*

Nous étudions la tâche de l'*approche* entre deux agents mobiles ayant la même portée de vision limitée et se déplaçant de façon asynchrone dans le plan. En partant de positions initiales quelconques, cette tâche consiste à les amener au moyen d'un algorithme déterministe à portée de vision l'un de l'autre en un temps fini. Ils sont autonomes et ne peuvent pas communiquer entre eux. Ils disposent cependant d'identifiants distincts et d'une boussole leur indiquant les points cardinaux. Chaque agent ne connaît que son identifiant. En particulier, il ignore l'identifiant et la position initiale de l'autre agent. Un adversaire choisit les identifiants et les positions initiales des deux agents dans le plan et modifie à volonté la vitesse à laquelle chacun se déplace. Le coût d'une exécution complète d'un algorithme pour cette tâche est la somme des distances parcourues par les deux agents jusqu'à accomplir l'approche.

Dans ces conditions, certains algorithmes déterministes résolvent cette tâche avec un coût exponentiel en fonction de la distance initiale Δ séparant les deux robots et de la longueur des identifiants. D'autres nécessitent que chaque agent connaisse sa position de départ dans un système de coordonnées commun ; d'autres encore fonctionnent avec un adversaire plus faible. Ce papier présente un algorithme déterministe pour la tâche de l'approche dont le coût est polynomial en fonction de Δ et de ℓ, la longueur (de la représentation binaire) du plus petit identifiant.

Mots-clefs : agents mobiles, rendez-vous asynchrone, plan, grille infinie, algorithme déterministe, coût polynomial

1 Introduction

Le système distribué considéré est constitué de deux agents mobiles se déplaçant dans le plan, chacun équipé d'une boussole lui indiquant les points cardinaux. Ils exécutent un même algorithme déterministe et un adversaire leur assigne initialement des positions et des identifiants distincts. Chaque agent connaît son identifiant mais ignore tout de l'autre agent, que ce soit son identifiant ou sa position. De plus, les agents sont incapables de communiquer et asynchrones, ce qui signifie que l'adversaire peut modifier à tout moment et à volonté la vitesse de chaque agent. Intuitivement, le but de l'adversaire est d'empêcher les agents d'accomplir leur tâche ou au moins de rendre la distance qu'ils parcourent avant d'y parvenir aussi grande que possible.

Dans cet article, nous nous intéressons aux algorithmes déterministes dont l'exécution par les agents permet, quelque soit le comportement de l'adversaire, d'accomplir la tâche de l'approche en un temps fini. De tels algorithmes sont appelés algorithmes d'approche. Étant donnée une valeur $\varepsilon > 0$, la tâche de l'approche dans le plan est accomplie au moment où les deux agents se trouvent à une distance au plus ε l'un de l'autre. On peut voir cette distance comme une portée de vision limitée, l'approche consistant donc à amener les agents à portée de vision l'un de l'autre. Cette tâche est une variation de celle du rendez-vous qui est accomplie quand les deux agents se rencontrent en un même point. Le coût d'une exécution complète d'un algorithme d'approche (resp. de rendez-vous) est la distance totale parcourue par les deux agents jusqu'à accomplir l'approche (resp. le rendez-vous).

La tâche de l'approche a déjà fait l'objet de plusieurs travaux dont [3, 1, 5] sont les plus proches du nôtre. Les auteurs de [3] présentent tout d'abord un algorithme de rendez-vous dans les graphes dénombrables,

†Cet article est un résumé étendu de [2]. Il est le résultat d'un travail effectué dans le cadre du projet ANR ESTATE (Ref. ANR-16-CE25-0009-03).

c'est-à-dire dont l'ensemble des sommets est dénombrable, fini ou infini. Ensuite, ils montrent qu'il n'existe pas d'algorithme de rendez-vous dans le plan. Cette observation les amène justement à introduire la tâche de l'approche, pour laquelle ils proposent un premier algorithme. Cependant, le coût de chacun des algorithmes de [3] est exponentiel en fonction de la distance séparant initialement les agents et de la longueur de leurs identifiants.

Les auteurs de [1] et [5] proposent des algorithmes d'approche polynomiaux en fonction de la distance initiale Δ entre les agents et de la longueur ℓ (de la représentation binaire) du plus petit des identifiants. Toutefois, ces deux contributions considèrent des conditions plus favorables que celles supposées dans cet article. D'une part, les auteurs de [5] affaiblissent l'asynchronie en supposant que chaque agent a une vitesse constante initialement fixée par l'adversaire. D'autre part, les auteurs de [1] supposent que chaque agent connaît les coordonnées de sa propre position dans un système de coordonnées commun.

Ainsi, nous proposons le premier algorithme déterministe d'approche pour deux agents asynchrones n'ayant pas accès à un système de coordonnées commun dont le coût est polynomial en fonction de Δ et ℓ.

2 Intuition de l'algorithme

2.1 Description informelle en quelques mots...

Dans le but de proposer un algorithme polynomial pour la tâche de l'approche, notre algorithme, comme ceux de [1, 5], s'appuie sur une réduction vers le rendez-vous dans les graphes infinis. Précisément, les auteurs de [5] expliquent comment transformer tout algorithme de rendez-vous dans les grilles infinies dont le coût est polynomial en fonction de la distance initiale (dans la grille) D et ℓ en un algorithme d'approche dans le plan dont le coût est polynomial en fonction de la distance initiale (dans le plan) Δ et ℓ. Ainsi pour obtenir l'algorithme d'approche que nous recherchons, il nous suffit de proposer un algorithme déterministe de rendez-vous dans les grilles infinies dont le coût est polynomial en fonction de D et ℓ.

Avant tout, définissons quelques éléments de vocabulaire. Le chemin emprunté par un agent est la suite des arêtes qu'il traverse. Celui-ci peut être vu comme un couple formé de son point de départ et de sa trajectoire. Une trajectoire peut être représentée comme une suite de points cardinaux, vers lesquels, les uns après les autres, l'agent va se déplacer d'une arête. Remarquons que la boussole dont chaque agent dispose est suffisante pour suivre la trajectoire que lui dicte notre algorithme.

Il est important de comprendre que le chemin emprunté par un agent doit dépendre de son identifiant. En effet, en l'absence d'une façon de distinguer les agents, il n'existe aucun algorithme déterministe de rendez-vous pour les graphes symétriques comme la grille. Des agents identiques suivent toujours les mêmes règles déterministes donc la même trajectoire. Leurs chemins ne diffèrent que par leur sommet initial. Si l'adversaire maintient les agents parfaitement synchrones, la distance entre eux n'évolue pas rendant leur rencontre impossible.

Les identifiants nous permettent de "casser la symétrie". L'idée consiste à faire "lire" à chaque agent la représentation binaire de son identifiant, un bit à la fois, du plus au moins significatif, et à choisir la trajectoire à suivre en fonction du bit lu. Le rendez-vous est assuré par notre algorithme quand les agents lisent des bits différents et donc suivent des trajectoires différentes.

Par ailleurs, la façon dont nous concevons les trajectoires de chaque agent nécessiterait de connaître une borne supérieure sur D et ℓ. Comme nous supposons que les agents ne disposent pas de telles informations, chacun d'entre eux procède par hypothèses successives dans le but de les deviner, ce que nous appelons phases. Les agents commencent à la phase 0, et si la phase i ne mène pas au rendez-vous, ils passent à la phase $i+1$, et ainsi de suite. Plus précisément, la trajectoire suivie par chaque agent durant la phase i est construite de telle sorte que le rendez-vous est accompli si 2^i est une borne supérieure sur D et ℓ. Ainsi, notre approche nécessite que les deux conditions suivantes soient réunies : à peu près au même moment, les agents doivent (1) traiter des bits différents et (2) considérer chacun une phase i telle que 2^i est supérieur à D et ℓ.

Toutefois, pour réunir ces conditions, nous nous heurtons à deux obstacles majeurs. Tout d'abord, puisque l'adversaire peut modifier à volonté la vitesse de chaque agent, rien dans ce que nous décrivons ci-dessus n'empêche l'adversaire de faire en sorte qu'à chaque instant, les agents traitent des bits de même valeur. Ensuite, le coût d'une trajectoire dépend du numéro de la phase à laquelle elle appartient. Si un agent venait

à considérer une phase i telle que la valeur de i soit exponentielle en fonction de D et ℓ, notre algorithme ne serait pas polynomial.

Pour surmonter ces obstacles, l'idée est d'empêcher l'adversaire de faire exécuter l'algorithme arbitrairement plus vite par un agent que par l'autre, sous peine de permettre le rendez-vous. Pour chacun des deux obstacles, nous avons un "mécanisme de synchronisation" dédié : le premier permet aux agents de lire et traiter les bits des représentations binaires de leurs identifiants à peu près à la même vitesse et le second assure qu'ils commencent la phase α presque au même moment. C'est là que réside le tour de force : orchestrer ces synchronisations dans un contexte complètement asynchrone tout en garantissant un coût polynomial. Dans le prochain paragraphe, nous détaillons davantage notre algorithme.

2.2 Quelques éléments à la loupe

L'approche décrite ci-dessus nous permet d'accomplir le rendez-vous quand il existe un indice auquel les représentations binaires des identifiants diffèrent. Cependant, ce n'est pas toujours le cas, et ce en particulier quand une représentation binaire est préfixe de l'autre (comme 10 et 1010). Ainsi, chaque agent ne lit pas son identifiant directement, mais une version transformée de celui-ci. Cette transformation est empruntée à [4] et garantit justement l'existence de la différence souhaitée.

Comme nous l'expliquons dans la sous-section précédente, notre solution fonctionne par phases numérotées 0, 1, 2, 3, 4, ... Durant la phase i, chaque agent suppose que D et ℓ sont tous deux inférieurs ou égaux à 2^i et traite un à un chacun des 2^i premiers bits de son identifiant transformé. Si 2^i est strictement supérieur à la longueur de son identifiant transformé, chaque agent considère que les bits manquants sont des 0. Le traitement d'un bit par un agent consiste à suivre une trajectoire dépendante de la valeur du bit et du numéro de la phase. Les trajectoires associées respectivement au bit 0 et au bit 1 sont différentes et construites de sorte que le rendez-vous a lieu lorsque les conditions suivantes sont remplies. (1) Chaque agent considère une phase correspondant à une borne supérieure sur D. (2) L'un d'eux suit la trajectoire correspondant à 0 tandis que l'autre suit celle correspondant à 1. (3) Ils commencent à suivre ces trajectoires presque simultanément. À la lumière de cela, si α est le plus petit entier tel que 2^α et supérieur ou égal à D et ℓ, il serait idéal que les agents démarrent simultanément la phase α et que durant celle-ci, ils traitent leurs bits à peu près à la même vitesse. Les agents accompliraient le rendez-vous durant cette phase, avant de finir de traiter le j-ème bit de leur identifiant transformé, où j désigne le plus petit indice auquel les représentations binaires de leurs identifiants transformés diffèrent. Cependant, atteindre une situation aussi idéale en présence d'un adversaire complètement asynchrone semble être un vrai défi. C'est là que les deux mécanismes de synchronisation mentionnés plus haut entrent en jeu.

Si les agents commencent la phase α presque au même moment, le premier mécanisme de synchronisation oblige l'adversaire à laisser les agents traiter leurs bits respectifs à des vitesses similaires, sous peine de voir le rendez-vous se produire prématurément, avant même le traitement du j-ème bit. Plus précisément, chaque traitement de bits est subdivisé en un certain nombre d'étapes et le premier mécanisme oblige l'adversaire à laisser les agents effectuer celles-ci à des vitesses similaires. Cette contrainte est imposée à l'adversaire par une trajectoire spécifique que chaque agent suit à la fin de chaque étape de chaque traitement de bits. Suivre cette trajectoire à la fin d'une étape donnée permet de "pousser" l'autre agent, c'est-à-dire, de le forcer à terminer cette même étape, sous peine de voir le rendez-vous se produire prématurément. Le principe sur lequel repose cette trajectoire est le suivant. Soit a_2 un agent qui suit un chemin donné C inclus dans une région donnée R de la grille. Appelons couverture de la zone R un chemin permettant d'en traverser au moins une fois chaque arête. Chaque couverture de R par l'agent a_1 permet de forcer a_2 à traverser au moins une arête de C. Ainsi, a_1 peut "pousser a_2 sur C" en couvrant R au moins autant de fois que ce chemin compte de traversées d'arêtes.

Notre principale contribution technique est le second mécanisme de synchronisation garantissant que les agents commencent la phase α à peu près simultanément. À première vue, il pourrait être tentant d'utiliser un principe analogue à celui que nous utilisons pour le premier. En effet, si l'agent a_1 suit un chemin couvrant r fois la région R de la grille où les $\alpha - 1$ premières phases de l'agent a_2 ont lieu avec r le nombre maximum de traversées d'arêtes que ce dernier effectue durant ces phases, alors a_1 pousse a_2 à compléter ces $\alpha - 1$ premières phases et donc à commencer la phase α. Néanmoins, si nous appliquions ce principe de façon stricte au cas du second mécanisme de synchronisation, le coût de notre algorithme serait super-

polynomial en fonction de D et ℓ à cause d'un effet cumulatif qui n'apparaît pas dans le cas du premier mécanisme.

Par conséquent, pour forcer un agent à démarrer sa phase α, le second mécanisme n'utilise pas le principe décrit ci-dessus mais une trajectoire bien plus complexe. Celle-ci fonctionne en ne se limitant pas au nombre d'arêtes traversées et en utilisant pleinement les informations disponibles à propos du "motif" dessiné par le second agent sur la grille. Il s'agit de la partie la plus délicate de notre algorithme. L'une de ses principales idées repose en particulier sur le fait que certains chemins arbitrairement longs peuvent être poussés à relativement bas coût par d'autres, judicieusement choisis.

Illustrons ce point avec l'exemple suivant. Soit A et B n'importe quelles trajectoires. Notons $\overline{A}$ et $\overline{B}$ leurs inverses respectifs à savoir les trajectoires formées par les points cardinaux "opposés" mais dans l'ordre inverse. Soit a_2 un agent qui suit depuis un sommet v_2 une suite arbitrairement longue de C_i où chaque C_i correspond soit à $A\overline{A}$ soit à $B\overline{B}$. L'agent a_1 se trouvant sur le sommet v_1 à distance au plus d de v_2 peut forcer a_2 à terminer sa suite de C_i (ou autrement le rendez-vous est accompli) simplement en suivant $A\overline{A}B\overline{B}$ depuis chacun des sommets à distance au plus d de v_1, et ce quelque soit le nombre de C_i.

Pour étayer cette affirmation, supposons par contradiction qu'elle est fausse. L'agent a_1 va finir par suivre $A\overline{A}B\overline{B}$ depuis v_2. Remarquons que si l'un des agents commence à suivre $A\overline{A}$ (resp. $B\overline{B}$) depuis v_2 tandis que l'autre est déjà en train de la suivre, alors les agents se rencontrent. En effet, cela implique que l'un des agents finit par suivre $\overline{A}$ (resp. $\overline{B}$) depuis un sommet v_3 vers v_2 alors que l'autre suit A (resp. B) depuis v_2 vers v_3, ce qui conduit au rendez-vous. Remarquons qu'après avoir suivi $A\overline{A}$ ou $B\overline{B}$ depuis v_2, chaque agent est de retour sur v_2. Intéressons-nous au moment t où a_1 se trouve sur v_2 alors qu'à la fois, il finit de suivre $A\overline{A}$ et commence à suivre $B\overline{B}$. À ce moment, a_2 ne peut pas être sur v_2. Il est donc en train de suivre soit $A\overline{A}$ soit $B\overline{B}$. Dans le premier cas, d'après les arguments énoncés ci-dessus, le rendez-vous a eu lieu pendant que les deux agents suivaient $A\overline{A}$, avant t. Dans le second, pour les mêmes raisons, il va se produire avant que le premier d'entre eux finisse de suivre $B\overline{B}$. Par conséquent, dans cet exemple, une seule trajectoire, relativement simple, suffit à l'agent a_1 pour forcer a_2 à finir de suivre une trajectoire arbitrairement longue. En fait, notre second mécanisme de synchronisation utilise précisément cet exemple.

Chaque partie de chaque trajectoire de chaque phase doit être orchestrée très précautionneusement pour permettre, à terme, cette synchronisation à bas coût tout en garantissant le rendez-vous. Cependant, c'est grâce à cette nouvelle façon de se déplacer que nous obtenons le coût polynomial.

3 Conclusion

Nous avons proposé le premier algorithme déterministe garantissant l'approche de deux agents asynchrones n'ayant pas accès à un système de coordonnées commun dont le coût est polynomial en fonction de la distance initiale entre les agents et de la longueur du plus petit de leurs identifiants. Cependant, lors de la conception de l'algorithme, nous n'avons pas cherché à en optimiser le coût. Ainsi, il est naturel de rechercher le coût optimal pour accomplir l'approche dans le modèle que nous considérons. Celui-ci serait d'autant plus intéressant à découvrir qu'il pourrait être comparé au coût de la même tâche dans le modèle où les agents disposent de leurs propres coordonnées dans un système de coordonnées commun, ce qui est analogue au fait de disposer d'un GPS. Cette comparaison permettrait de déterminer si, oui ou non, cette hypothèse est naturellement pertinente pour minimiser la distance parcourue.

Références

[1] Evangelos Bampas, Jurek Czyzowicz, Leszek Gasieniec, David Ilcinkas, and Arnaud Labourel. Almost optimal asynchronous rendezvous in infinite multidimensional grids. In *Proceedings of DISC 2010*, pages 297–311, 2010.

[2] Sébastien Bouchard, Marjorie Bournat, Yoann Dieudonné, Swan Dubois, and Franck Petit. Asynchronous approach in the plane : A deterministic polynomial algorithm. In *Proceedings of DISC 2017*, pages 8 :1–8 :16, 2017.

[3] Jurek Czyzowicz, Andrzej Pelc, and Arnaud Labourel. How to meet asynchronously (almost) everywhere. *ACM Transactions on Algorithms*, 8(4) :37, 2012.

[4] Anders Dessmark, Pierre Fraigniaud, Dariusz R. Kowalski, and Andrzej Pelc. Deterministic rendezvous in graphs. *Algorithmica*, 46(1) :69–96, 2006.

[5] Yoann Dieudonné and Andrzej Pelc. Deterministic polynomial approach in the plane. *Distributed Computing*, 28(2) :111–129, 2015.