Comment s'assurer de garder le contact (et nos distances) †

Thibaut Balabonski[1] et Pierre Courtieu[2] et Robin Pelle[1] et Lionel Rieg[3] et Sebastien Tixeuil[4] et Xavier Urbain[5]

[1] *Université Paris-Sud, LRI, CNRS UMR 8623, Université Paris-Saclay*
[2] *Conservatoire national des arts et métiers,* CÉDRIC*, Paris*
[3] *Univ. Grenoble Alpes, CNRS, Grenoble INP*‡*, VERIMAG, 38000 Grenoble, France*
[4] *Sorbonne Université, CNRS, LIP6* [5] *Université de Lyon, Université Claude Bernard Lyon 1, CNRS, LIRIS UMR 5205*

Nous étudions le problème du maintien de connexion dans les réseaux de robots mobiles. On considère un robot incontrôlable (la « cible ») et une flotte de robots volumiques autonomes se déplaçant dans le plan réel et munis de capteurs et transmetteurs à portée limitée. Le problème consiste à maintenir à tout moment une connexion entre un point fixe connu au départ et la cible. Cette situation est par exemple instanciée dans le cas d'une équipe de recherche (la cible) en cours d'exploration et qui doit conserver une liaison avec la base des secours (le point fixe). Dans un tel cas où des vies sont en jeu, le problème devient *critique* : il est impératif d'avoir les plus fortes garanties de correction possibles sur les protocoles candidats.

Nous définissons formellement ce problème et proposons une famille de protocoles que nous prouvons correcte grâce à l'assistant de preuve Coq et la bibliothèque PACTOLE. Nous illustrons en particulier l'utilité de cet outil formel ainsi que de la démarche associée, de la réflexion préliminaire sur un problème à la production d'une solution certifiée.

Mots-clefs : Essaims de robots mobiles, espace euclidien, connexion, assistant de preuve, Pactole, méthodes formelles

1 Introduction

Les réseaux de robots autonomes ont capté l'attention de la communauté de l'algorithmique distribuée, de l'industrie et des grands média par les nouvelles applications et la rupture technologique qu'ils promettent. Exploration d'espaces dévastés par une catastrophe naturelle ou d'origine humaine et recherche de survivants, décontamination et démantèlement de centrales dangereuses sont autant d'exemples qui montrent l'intérêt d'essaims de robots autonomes et mobiles ; ils soulignent en outre l'extrême dynamicité et la complexité de ce modèle émergent.

De nombreuses variantes du modèle introduit en 1999 par Susuki et Yamashita [SY99] sont apparues au fur et à mesure de sa popularité grandissante ; le lecteur intéressé pourra en trouver les descriptions dans de récents ouvrages [FPS19]. Nous considérons le modèle de base où les robots sont équipés du même programme (le protocole) et opèrent selon un cycle Perception-Calcul-Déplacement (Look-Compute-Move). Durant la première phase, le robot reçoit *une perception* d'un instantané de l'environnement, exprimée dans son propre référentiel auto-centré et qui n'a pas forcément les mêmes échelles ou orientations que ceux de ses congénères (il peut même être différent d'un cycle au suivant). À partir de cette perception *uniquement* (et donc *sans mémoire des actions passées*), le robot calcule à l'aide du programme embarqué une destination (toujours exprimée dans son propre référentiel). Il se déplace enfin vers celle-ci au cours de la dernière phase.

En prenant un peu de recul sur les travaux précités [FPS19], on peut remarquer que l'approche basée sur le modèle Suzuki et Yamashita a concentré les efforts sur quelques problèmes de référence qui sont théoriquement intéressants, mais qui sont peu pertinents en pratique, comme l'exploration permanente ou terminale d'un graphe en forme d'anneau, et le rassemblement ou la concentration de tous les robots à un

‡Institute of Engineering Univ. Grenoble Alpes

†Ce travail est financé pour partie par le projet ANR SAPPORO (2019-CE25-0005-1).

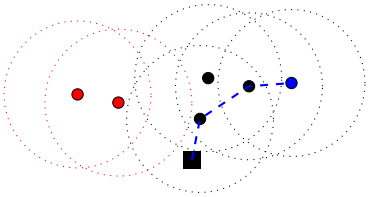

FIGURE 1 – Dans le système ci-contre où ■ est la base, ● la cible, et où les visibilités sont représentées par des cercles pointillés, les robots ● ne peuvent pas détecter les autres. Il y a cependant une chaîne de connexion entre la base et la cible : le chemin pointillé bleu.

endroit particulier. D'un autre côté, de nombreux problèmes pratiques pertinents, tels que la construction d'infrastructures de communication mobile ad hoc pour permettre la communication avec les équipes de secours, restent inétudiés (*a fortiori* avec une approche formelle). Pourtant, leur exactitude est cruciale, et peut-être vitale, elle doit donc être évaluée formellement et mécaniquement vérifiée. Dans l'ensemble, pour ces problèmes pratiques, la question n'est pas vraiment de caractériser quelles hypothèses du système permettent la résolution des problèmes, mais plutôt de savoir comment concevoir une solution prouvée correcte en utilisant des hypothèses qui correspondent à des dispositifs réels. Cet article est le premier pas dans cette direction. Plus précisément, nous partons d'un scénario d'application réel pour concevoir conjointement *(i)* sa spécification formelle, *(ii)* une famille de protocoles capables de résoudre le problème, et *(iii)* leur preuve de correction, le tout exprimé avec le même cadre formel, PACTOLE. Dans cette démarche, nous illustrons comment les méthodes formelles et PACTOLE en particulier pourraient être utilisées pour obtenir des protocoles qui sont corrects par conception avant qu'ils ne soient déployés sur des appareils réels.

2 Le problème

La recherche de survivants dans une zone dévastée est un cas d'application vedette de la robotique, en particulier en essaim. Une équipe d'humains au sol qui part porter assistance ne peut ni être abandonnée à elle-même, ni compter sur une quelque infrastructure fixe de communication ; une flotte de robots embarquant des relais peut alors permettre d'établir une connexion de proche en proche à une base de secours. Le maintien à tout instant d'une telle connexion est la propriété *critique* que nous étudions ici.

On considère un robot particulier (la « cible ») poursuivant l'équipe de recherche, donc à une position incontrôlable par le protocole à l'étude, et une flotte de robots autonomes identiques gérés par le protocole, équipés de capteurs/relais à portées limitées D_{max} et dont les déplacements à chaque cycle sont bornés par une distance D. Une « base » fixe sert de point de départ et stocke un nombre de robots supposé suffisant. Elle émet un robot chaque fois que nécessaire pour maintenir un chemin dans le graphe de visibilité [§] liant la base et la cible. Le protocole doit ainsi garantir au moins deux propriétés pour un essaim : *(i)* les robots n'entrent *jamais* en collision et *(ii)* il existe *à chaque instant* une séquence de robots $r_{i_k}, r_{i_{k-1}} \ldots, r_{i_1}$ telle que r_{i_k} voit la base, r_{i_1} voit la cible et $\forall j$ t.q. $2 \leq j \leq k, r_{i_j}$ est à portée de vue de $r_{i_{j-1}}$ (voir figure 1). Le cas d'application visé mettant des vies en jeu, *nous voulons des garanties formelles* qu'un protocole candidat satisfait ces invariants.

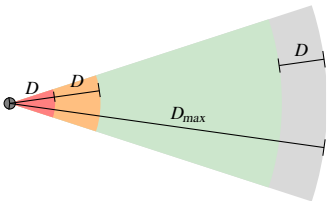

FIGURE 2 – Division de la zone (circulaire) de visibilité d'un robot en couronnes concentriques en fonction de la distance : zone de ■ **Poursuite** d'un robot, fort risque de ■ **Collision**, ■ **Danger** au cycle suivant, enfin une zone de ■ **Relais**. Le comportement d'un robot change en fonction des zones dans lesquelles se trouvent ses voisins.

3 Les différentes étapes vers une solution

L'étude considère que tous les robots sont activés à chaque cycle (modèle FSYNC), ce qui est justifié en pratique par leur uniformité matérielle et logicielle.

Afin d'éviter que la tâche soit trivialement irréalisable, on munit la cible des mêmes capacités de déplacement que les autres robots : elle pourrait sans cela distancer les robots relais.

§. Nous supposons que la distance de perception est la même que la distance de transmission sans perte de généralité.

La vitesse des robots doit être limitée. Plus exactement celle-ci doit être limitée par rapport au rayon de détection D_{max} des robots : si la vitesse maximale d'un robot (de la cible notamment) lui permet toujours de sortir du rayon de détection de tous les autres robots en un seul cycle alors le problème n'admet évidemment pas de solution. La contrainte $D_{max} > 7D$ permet, par exemple et sans prétention d'optimalité, de lancer des robots depuis la base sans risquer ni de collision avec les robots déjà partis ni de rupture de connexion.

L'anonymat total des robots ne semble pas envisageable étant donné la dissymétrie du problème. Comme d'une part l'objectif du protocole est de ne pas perdre de vue la cible et que d'autre part les robots « supplémentaires » partent de la base, un robot participant à la connexion doit suivre en priorité les robots qui sont plus proches de la cible que lui-même, sous peine de rompre la chaîne de connexion. Un robot ne percevant ni la cible ni la base est cependant dans l'incapacité de faire cette distinction en l'absence d'information supplémentaire (robots complètement anonymes).

Sans nécessiter une réelle identification individuelle des robots, un protocole candidat peut supposer qu'un robot sait distinguer le sous-ensemble des robots dont l'identifiant est plus petit que le sien. Nous choisissons d'exploiter cette dissymétrie en maintenant l'invariant suivant : les robots d'identifiants plus petits (et non éliminés, voir plus bas) sont plus proches de la cible.

Savoir différencier les robots « en attente ». Nous avons supposé que la base stockait les robots qui n'étaient pas encore impliqués dans le protocole. Le modèle distingue les robots stockés de ceux qui participent activement et les collisions ne les concernent pas.

Les considérations suivantes sont liées à la nature de la solution envisagée. Il est probable que des solutions existent pour des choix légèrement différents.

Les robots doivent pouvoir se retirer du système pour éviter les collisions. Lorsque la cible se rapproche de la base, le nombre de robots impliqués dans la chaîne de connexion doit diminuer pour éviter les collisions. Cette propriété peut être obtenue en munissant les robots d'une capacité d'auto-élimination, au procédé arbitraire. Notre modèle distingue un robot « vivant » d'un robot « mort ». Les robots morts ne participent plus à la connexion et deviennent indétectables.

Un protocole précautionneux. N'ayant pas la même vision de leur environnement, les robots ne peuvent pas prévoir les mouvements de leurs voisins. Une hypothèse conservatrice est de considérer que chaque robot peut se rendre en une itération à n'importe quel point que sa vitesse maximale lui permet d'atteindre. Afin d'éviter les collisions nous souhaitons maintenir l'invariant suivant : un robot n'arrive jamais à une distance $d \leq D$ d'un autre robot, ce que la capacité d'élimination nous permet d'obtenir.

Prévenir avant de s'éliminer. Les robots, toujours parce qu'ils n'ont pas le même champ de vision, ne sont pas capables de détecter qu'un autre robot va s'éliminer dans l'itération courante. Un robot qui s'élimine peut cependant laisser suffisamment d'espace pour rendre ses voisins susceptibles de rompre la connexion dans la même itération. Nous considérons l'usage d'une « lumière » équipant les robots et signalant que le porteur est susceptible de s'éliminer au prochain tour : un voisin privilégiera un autre robot à suivre pour éviter les ruptures de la chaîne de connexion.

4 Une famille de solutions

Le modèle formel. Le modèle formel tient compte de toute l'analyse de la section précédente. Ainsi, l'état d'un robot comporte sa position, l'état de sa lumière, son état d'élimination (vivant/mort) et d'allumage.

```
Definition info := identifiant*light*alive*launched.     (* Id,bool,bool,bool *)
Definition State := AddInfo info OnlyLocation.
```

Le protocole candidat. Un protocole candidat à la résolution de la tâche sous ces hypothèses a pour type `observation` → `location*light`, en d'autres termes : le robot reçoit une observation et calcule une destination et une nouvelle couleur.

Nous définissons en fait une famille de solutions décrite par un protocole générique paramétré par trois fonctions `choose_target`, `choose_new_pos` et `move_to`.

```
Definition protocol obs : location*light :=
 let target := choose_target obs in                          (* robot à suivre *)
 let new_pos := choose_new_pos obs (get_location target) in   (* destination *)
 match move_to obs new_pos with                          (* destination ∈ Danger? *)
 | true ⇒ (new_pos,false)              (* non: déplacement + lumière éteinte. *)
 | false ⇒ (origin, true)                      (* oui: stop + lumière allumée. *)
 end.
```

Preuve de correction. Pour prouver que tout protocole de cette famille maintient la connexion entre la base et la cible (invariant I_4) tout en évitant les collisions (I_0), nous exhibons un ensemble de trois invariants auxiliaires qui permet de conclure : (I_1) un robot qui s'élimine était allumé au cycle précédent ; (I_2) un robot qui en élimine un autre est toujours éteint ; (I_3) si un robot n'a que des voisins allumés, l'un d'eux est hors de la zone de poursuite. Il faut noter que si l'enchaînement des arguments de preuve (figure 3) est complexe, la correction de l'ensemble est validée mécaniquement par Coq via Pactole.

Bien sûr, cette preuve dépend d'hypothèses sur les paramètres du protocole (`move_to`, `choose_new_pos` et `choose_target`). Sous l'hypothèse que chaque robot a au moins un voisin, ces spécifications sont :

- `move_to` renvoie vrai si et seulement s'il n'y a pas de robot dans la zone de **Danger** ;
- `choose_new_pos` renvoie une position à moins de D et dans la zone de **Relais** du robot à suivre ;
- `choose_target` renvoie un robot à suivre qui est vivant, à portée, d'identifiant strictement inférieur, de préférence ¶ éteint et hors de la zone de **Poursuite** (sauf s'il n'y a pas le choix).

Existence d'une solution. Pour montrer que cette famille de solution est non vide, il nous faut donner une instance des trois fonctions paramètres. Cela est facile pour `move_to` qui ne fait que tester la présence de robots proches. Pour `choose_new_pos`, si le robot suivi est dans la zone de poursuite, le robot se déplace de D vers lui, sinon il reste sur sa position actuelle. Enfin, pour `choose_target`, il faut considérer les robots vivants, à portée et d'identifiants strictement inférieurs. Si l'un d'entre eux a sa lumière éteinte, on le choisit, sinon on choisit le plus proche. On vérifie alors que ces définitions vérifient bien les spécifications ci-dessus.

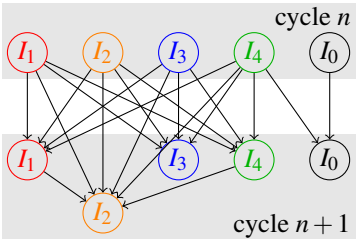

Figure 3 – Schéma de preuve général des invariants.

5 Conclusion

Nous avons montré par l'exemple comment les méthodes formelles, et Pactole en particulier, peuvent aider les concepteurs de protocoles d'essaims robotiques mobiles à spécifier formellement, concevoir leurs solutions et les prouver correctes, en équilibrant l'expressivité et les fondements mathématiques des développements. Le développement complet pour Coq 8.12 est disponible à `https://pactole.liris.cnrs.fr/pub/connection_8.12.tgz`, le détail de certaines des preuves est présenté dans Balabonski et al. [BCP+21]. Bien entendu, prouver que les algorithmes sont corrects pour de nouveaux problèmes n'est que la première étape. Une deuxième étape naturelle est de s'assurer que les *implémentations* des algorithmes maintiennent les invariants pertinents lorsqu'ils sont effectivement déployés sur des appareils réels.

Références

[BCP+21] Thibaut Balabonski, Pierre Courtieu, Robin Pelle, Lionel Rieg, Sébastien Tixeuil, and Xavier Urbain. Computer aided formal design of swarm robotics algorithms. *CoRR*, abs/2101.06966, 2021.

[FPS19] Paola Flocchini, Giuseppe Prencipe, and Nicola Santoro, editors. *Distributed Computing by Mobile Entities*, volume 11340 of *Lecture Notes in Computer Science, Theoretical Computer Science and General Issues*. Springer Nature, 2019.

[SY99] Ichiro Suzuki and Masafumi Yamashita. Distributed Anonymous Mobile Robots : Formation of Geometric Patterns. *SIAM Journal of Computing*, 28(4) :1347–1363, 1999.

¶. Au sens où s'il ne vérifie pas cette propriété, alors aucun autre robot à portée vivant et d'identifiant strictement inférieur non plus.