

# Réseaux de capteurs avec détecteurs de collision

Mohssen Abboud maboud@liafa.jussieu.fr

Carole Delporte cd@liafa.jussieu.fr

Hugues Fauconnier hf@liafa.jussieu.fr

LIAFA-CNRS et Université Paris 7- 2 place Jussieu, 75251 Paris Cedex 05, France

**Résumé**— On considère ici des réseaux de capteurs communiquant en rondes synchrones par radio-diffusion. Le nombre de capteurs n'est pas connu et les capteurs peuvent être anonymes, de plus certains capteurs peuvent tomber en panne définitive et cesser d'émettre. En présence de collisions de messages, des problèmes très simples comme le consensus ne peuvent pas être résolus. Aussi on suppose que les capteurs sont équipés de détecteurs de collision qui donnent des informations non nécessairement fiables sur les collisions. En considérant d'abord un modèle de communication très rudimentaire sans message, on montre que des détecteurs de collision très simples permettent de résoudre le problème du consensus et de la diffusion fiable. On montre ensuite comment on peut utiliser ces détecteurs de collision pour calculer le maximum des valeurs proposées à la diffusion dans une ronde.

## I. INTRODUCTION

La technologie des réseaux de capteurs sans fil a beaucoup progressé depuis quelques années [2] et de nombreuses applications basées sur des réseaux de capteurs sont actuellement réalisées ou en cours de réalisation. Les capteurs peuvent être utilisés dans un milieu hostile comme, par exemple, sur un champ de bataille, en présence d'incendies, d'inondations, de tremblements de terres. Dans ces environnements les capteurs peuvent tomber en panne, même dans un fonctionnement normal. Par ailleurs, la communication entre les capteurs se fait généralement par diffusion radio et cette diffusion ne peut être supposée fiable. Il est donc intéressant et nécessaire de développer une algorithmique tolérante aux pannes pour ce type de réseaux.

Deux types de défaillances peuvent être considérées : les pannes de capteurs dues par exemple à un défaut d'énergie et les défaillances de communication. En ce qui concerne les premières, nous nous restreindrons ici aux pannes "crash" : un capteur s'arrête définitivement. Pour le deuxième type de défaillances, la communication de ces réseaux de capteurs est généralement par diffusion radio et une des particularités de ce mode de communication est qu'il n'est pas fiable : si plusieurs capteurs émettent de façon simultanée il y a un risque de collision. Dans le cas où le nombre de capteurs est élevé, même si on peut supposer que tous les capteurs ont une horloge globale parfaite, on ne peut pas attribuer des intervalles de temps à chaque capteur qui garantiraient l'absence de collision. Les algorithmes développés doivent donc pouvoir fonctionner même en présence de collision. En général des algorithmes dits de back-off [11], [10], [12], [7] permettent d'assurer de façon probabiliste l'absence de collisions, mais une communication fiable suppose en plus que

les collisions peuvent être détectées afin de pouvoir ré-émettre. Il y a donc une nécessité de détection de ces collisions, mais d'un point de vue pratique il est difficile voire impossible d'assurer que cette détection est toujours fiable [4].

D'un point de vue plus formel, Chockler et al. dans [5], [6] ont introduit la notion de détecteurs de collision. Un détecteur de collision est un mécanisme abstrait qui donne localement aux capteurs des informations sur les collisions. Ils montrent que suivant les propriétés de ces détecteurs de collision, des problèmes classiques comme le consensus [9] peuvent être résolus. Suivant cette démarche, nous allons nous intéresser à un cadre plus restreint dans lequel les messages échangés entre les capteurs n'ont pas de contenu : un capteur peut juste émettre ou ne pas émettre. Dans le modèle que nous considérons, les capteurs émettent en rondes synchronisées : dans une ronde un capteur peut émettre ou non et tous les capteurs sont informés si il y a eu ou non au moins une émission dans la ronde. Mais à cause de la possibilité de collisions un capteur peut ne pas obtenir cette information. Les propriétés des détecteurs de collision que nous considérons ici sont la *complétude* qui assure que toute collision est détectée, l'*exactitude* qui assure qu'aucune fausse collision n'est détectée ainsi que l'*exactitude ultime* qui assure l'exactitude mais uniquement au bout d'un certain temps.

La plupart des applications de réseaux de capteurs supposent que les capteurs sont capables au moins de récolter et de partager des informations sur l'environnement. Il est important que cette information soit fiable : les informations provenant d'un capteur doivent être reçues par tous les capteurs et si une information est reçue elle doit bien provenir d'un capteur. Aussi nous allons étudier comment on peut assurer une diffusion fiable [8] en présence de collisions. Pour cela nous donnerons une spécification formelle de la diffusion fiable en présence de pannes et nous donnerons des algorithmes pour la résoudre avec des détecteurs de collision. Il est clair que la solution à ce problème est indispensable pour pouvoir développer des algorithmes utiles dans le modèle considéré. Cependant, s'il est assez clair que la classe la plus forte des détecteurs de collision permet de développer une algorithmique distribuée tolérante aux pannes efficace, ce n'est pas aussi clair pour la classe n'assurant que l'exactitude ultime et la complétude. En effet même si la diffusion fiable est réalisable pour cette classe, les capteurs ne terminent pas la diffusion simultanément ce qui pose des problèmes pour composer plusieurs diffusions.

Dans une première section après avoir défini le modèle

nous donnons la spécification de la diffusion fiable et nous définissons les détecteurs de collision que nous considérons, ensuite dans la deuxième section nous montrons comment ces détecteurs de collision permettent de résoudre le problème de la diffusion fiable, dans la troisième section nous montrons que pour la classe la plus forte des détecteurs de collision il est possible de résoudre le problème de trouver le maximum d'un ensemble de valeurs de façon efficace. Enfin nous concluons en présentant plusieurs extensions à ce travail et un problème ouvert. Pour ne pas alourdir le texte et pour des raisons de place nous ne présentons pas ici les preuves formelles des algorithmes. Une version complète avec les preuves est disponible dans [1].

## II. MODÈLE ET DÉFINITIONS

### A. Modèle

$P = \{p_1, p_2, \dots\}$  est l'ensemble des capteurs. Le nombre de capteurs n'est pas connu et les capteurs ne sont pas supposés avoir une identité unique. On suppose que les capteurs exécutent des *rondes synchronisées*. A chaque ronde les capteurs peuvent diffuser un message, recevoir des messages diffusés dans la ronde et changer d'état en conséquence.

Un capteur peut tomber en panne crash, dans ce cas il s'arrête définitivement. S'il tombe en panne crash au cours d'une diffusion le message peut ou non être diffusé. L'ensemble des capteurs vivants à la fin de ronde  $r$  est noté  $Viv(r)$ . Un capteur sera dit *correct* s'il est vivant à toutes les rondes.

Dans le modèle restreint que l'on considère ici, les messages n'ont pas de contenu : à une ronde un capteur  $p$  peut ou non émettre un message vide par un  $send()$ .  $send_p^r$  est un booléen qui est vrai si et seulement si  $p$  émet dans la ronde. Plus généralement,  $send^r$  est un booléen qui est vrai si et seulement au moins un capteur vivant dans la ronde a émis un message. On notera  $M^r$  le nombre de capteurs qui émettent dans la ronde.

Si un message a été émis dans la ronde, le message peut ne pas être reçu par certains capteurs, on dira dans ce cas qu'il y a eu *collision* :  $recv_p^r$  est un booléen qui est vrai si et seulement  $p$  a reçu un message dans la ronde. On supposera toujours qu'un capteur ne peut pas recevoir de message si aucun message n'a été émis. On suppose donc toujours la propriété suivante :

- *Intégrité* : pour tout capteur  $p$  si  $recv_p^r$  alors  $send^r$ .

Cependant en cas de collision il est possible qu'un capteur ne reçoive rien dans la ronde alors qu'un message a été émis.

En général les collisions proviennent du fait que plusieurs capteurs émettent simultanément. On supposera donc que, au bout d'un certain temps correspondant à une période initiale d'instabilité, si au plus un capteur émet dans la ronde alors il n'y a pas de collision :

- *Absence ultime de collisions* : il existe  $r_0$  tel que pour toute ronde  $r > r_0$  et tout  $p$  de  $Viv(r)$   $M^r = 1 \Rightarrow recv_p^r$

En général les algorithmes classiques de "back-off" assurent qu'un jour au plus un capteur émet. On notera par un booléen

$active_p^r$  le fait que le capteur  $p$  peut, suivant l'algorithme de "back-off", émettre dans la ronde  $r$ . L'algorithme de "back-off" assure la propriété suivante :

- *Propriété du back-off* : Si  $p$  est correct, alors pour une infinité de rondes  $send_p^r$  est vrai alors pour une infinité de ces rondes  $active_p^r$  est vraie et pour tout  $q \neq p$   $active_q^r$  est faux.

On peut assurer cette propriété par un tirage aléatoire : à chaque ronde chaque capteur tire de façon indépendante vrai ou faux avec des probabilités non nulles. Dans le cas où le nombre de capteurs est borné, la propriété précédente est assurée.

Remarquons qu'avec la propriété du back-off et l'absence éventuelle de collision, on peut assurer qu'infiniment souvent il n'y aura pas de collisions et plus précisément que si un capteur émet infiniment souvent ses messages sont reçus infiniment souvent sans collision.

Dans la suite, pour simplifier les notations, on omettra en général l'indice correspondant à la ronde qui est généralement défini par le contexte.

### B. Spécification de la diffusion fiable

Même avec l'intégrité et l'absence ultime de collision, on ne peut arriver à un accord entre les capteurs sur le fait qu'un message a été ou non émis dans une ronde.

Soit  $Rbroadcast_p$  un booléen indiquant que le capteur  $p$  veut diffuser un message dans la ronde. Le problème de la diffusion fiable [8] est d'assurer qu'un jour tous les capteurs sauront si oui ou non au moins un capteur a voulu diffuser dans la ronde. Plus précisément, soit  $del_p$  une variable booléenne locale initialisée à  $\perp$  qui ne peut être modifiée qu'une seule fois (quand un capteur écrit cette variable on dit qu'il décide), un protocole de diffusion fiable doit assurer les propriétés suivantes :

- *Terminaison* : Pour tout capteur correct  $p$  un jour  $del_p$  sera différente de  $\perp$ .
- *Accord* : Pour tout  $p$  et tout  $q$  si  $del_p \neq \perp \wedge del_q \neq \perp$  alors  $del_p = del_q$ .
- *Validité* : S'il existe un capteur correct  $p$  tel que  $Rbroadcast_p$  alors pour tous les capteurs  $q$  on a un jour  $del_q \neq False$  et si pour aucun capteur  $p$   $Rbroadcast_p = True$  alors pour tous les capteurs  $q$  on a  $del_q \neq True$ .

Notons que si uniquement des capteurs non corrects proposent un message dans la ronde il est possible que les capteurs corrects considèrent qu'il y a eu émission mais il est aussi possible qu'ils considèrent qu'il n'y a pas eu d'émission, cependant dans tous les cas ils devront avoir le même avis.

On peut remarquer que le problème de la diffusion fiable peut être formulé comme un problème de consensus avec une condition d'intégrité qui assure que la décision doit être *True* si au moins capteur correct a voulu émettre et ne doit être *False* que si aucun capteur n'a voulu émettre.

On peut montrer que les propriétés d'intégrité, d'absence ultime de collisions et de "back-off" sont insuffisantes pour assurer la diffusion fiable. Intuitivement, quelque soit la séquence des  $send()$  pour chaque ronde, un capteur peut avoir pour un nombre arbitrairement grand de rondes une séquence de réception où tous les  $recv_q$  sont égaux à  $False$ . Ainsi une séquence quelconque d'émissions est indistinguable pendant arbitrairement longtemps d'une séquence où rien n'est jamais émis. La propriété d'absence ultime de collision et de "back-off" assure seulement que s'il y a un nombre infini d'émissions, pour un nombre infini de rondes les capteurs recevront bien les messages.

### C. Détecteurs de collision

Des systèmes dans lesquels la diffusion fiable n'est pas assurée sont particulièrement faibles. Pour assurer cette diffusion fiable on va supposer que les capteurs sont équipés d'un *détecteur de collision* [5], [6]. Un détecteur de collision est un mécanisme local à chaque capteur qui informe chaque capteur du fait qu'il peut y avoir eu une collision dans la ronde. La sortie du détecteur de collision pour le capteur  $p$  sera représentée par un booléen  $col_p$  qui sera vrai si et seulement si  $p$  a été averti d'une suspicion de collision dans la ronde.

Les détecteurs de collisions sont définis par leurs propriétés. Dans la suite on supposera toujours la propriété suivante :

- *Complétude* : toute collision est détectée.

Remarquons qu'avec des détecteurs de collision vérifiant la complétude on aura :

*Fait 1* : Pour un détecteur de collision assurant la complétude, à chaque ronde :

- Si aucun capteur vivant dans la ronde n'a fait de  $send()$  alors chaque capteur vivant  $p$  a soit  $col_p$  soit  $\neg recv_p$
- Si au moins un capteur vivant a fait un  $send()$  alors chaque capteur vivant a soit  $recv_p$  soit  $col_p$

La complétude seule n'est pas très intéressante puisqu'elle n'empêche pas que des collisions soient détectées alors qu'aucun message n'a été émis. Comme dans le cas où on suppose uniquement les propriétés d'intégrité, d'absence ultime de collision et la propriété du back-off, on peut montrer qu'ajouter la présence de détecteurs de collision assurant la complétude est insuffisant pour permettre la diffusion fiable. D'un point de vue pratique, il semble raisonnable qu'un détecteur de collision mesure l'existence d'un certain bruit qu'on ne pourrait distinguer de l'émission d'un message, et en l'absence de toute émission on peut supposer que le bruit résiduel est suffisamment faible. On va donc considérer deux autres propriétés : l'exactitude et l'exactitude ultime. Intuitivement, l'exactitude signifie que si aucun message n'est émis alors il ne peut y avoir de détection de collision, l'exactitude ultime assure simplement l'exactitude au bout d'un certain nombre (inconnu) de rondes. Plus précisément,

- *Exactitude* : si  $col_p$  alors  $send$  et  $\neg recv_p$ .
- *Exactitude ultime* : il existe  $r_0$  tel que pour tout  $r > r_0$  si  $col_p$  alors  $send$  et  $\neg recv_p$ .

*Fait 2* : Pour un détecteur de collision assurant l'exactitude ultime :

- Si infiniment souvent rien n'est émis, alors pour une infinité de rondes aucun capteur ne détectera ni collision ni réception de message

Par analogie avec les détecteurs de défaillances [3], un détecteur de collision qui assure à la fois la complétude et l'exactitude est appelé *détecteur parfait* et la classe des détecteurs parfaits sera notée  $\mathcal{P}$ . De même un détecteur de collisions qui assure la complétude et l'exactitude ultime est *ultimement parfait* et la classe correspondante sera notée  $\diamond\mathcal{P}$ .

### III. IMPLÉMENTATION DE LA DIFFUSION FIABLE

Un détecteur parfait permet facilement de réaliser la diffusion fiable :

---

#### Algorithme 1 : Pour chaque capteur $p$

---

Pour chaque ronde :

```

if  $Rbcast_p$  then
    send()
end
if  $recv_p \vee col_p$  then
     $del_p \leftarrow True$ 
else
     $del_p \leftarrow False$ 
end
    
```

---

Fig. 1. Diffusion fiable avec détecteurs parfaits.

*Proposition 1* : L'algorithme de la figure 1 réalise la diffusion fiable en une seule ronde avec un détecteur de collision parfait.

En effet si au moins un capteur vivant émet dans la ronde, si il y a collision pour  $p$ , la complétude assure que  $col_p$  sera vraie. Si personne n'émet dans la ronde la propriété d'exactitude assure que  $col_p$  est faux et la propriété d'intégrité assure que  $recv_p$  est aussi faux. On en déduit la validité et l'accord, la terminaison est claire. Notons que l'on n'a pas utilisé ici la propriété d'absence ultime de collision ni la propriété du "back-off".

Un détecteur ultimement parfait permet aussi d'assurer la diffusion fiable, mais sans assurer de borne sur le nombre de rondes nécessaires pour faire cette diffusion. L'algorithme 2 réalise cette diffusion fiable. Cet algorithme se décompose en groupe de 4 rondes. Dans la première ronde, un capteur émet s'il désire faire un  $rbcast$  et si  $active$  est vrai c'est-à-dire si l'algorithme de back-off l'y autorise. On assure ainsi que si l'algorithme ne s'arrête pas, tous les capteurs corrects pourront émettre infiniment souvent sans collision.

Si un capteur ne reçoit rien dans cette première ronde, du fait des propriétés d'intégrité et des propriétés de complétude et d'exactitude des détecteurs de collision cela signifie qu'aucun message n'a été émis dans la ronde et donc qu'aucun des capteurs choisis par l'algorithme de back-off ne désirait faire  $Rbcast$  : dans ce cas la décision devra être  $False$  sauf

**Algorithme 2** : Pour chaque capteur  $p$ 


---

```

Initialisation :  $estimate_p \leftarrow Rbroadcast_p$ 
ronde  $4n$  :
 $Fait_p \leftarrow False$ 
if  $estimate_p \wedge active_p$  then
    send()
end
if  $recv_p$  then /* un message reçu */
     $veto0_p \leftarrow True$ 
     $veto1_p \leftarrow False$ 
     $estimate_p \leftarrow True$ 
else
    if  $col_p$  then /* collision détectée */
         $veto0_p \leftarrow veto1_p \leftarrow True$ 
    else /* pas de collision et rien reçu */
         $veto0_p \leftarrow estimate_p$ 
         $veto1_p \leftarrow True$ 
    end
end
ronde  $4n+1$  : /* ronde de veto sur  $Rbroadcast = False$  */
if  $veto0_p$  then
    send()
end
if  $\neg(recv_p \vee col_p)$  then /* pas de collision et rien reçu */
     $Fait_p \leftarrow True$ 
end
ronde  $4n+2$  : /* ronde de veto sur  $Rbroadcast = True$  */
if  $veto1_p$  then
    send()
end
if  $\neg(recv_p \vee col_p)$  then /* pas de collision et rien reçu */
     $Fait_p \leftarrow True$ 
end
ronde  $4n+3$  : /* ronde de terminaison  $Fait_p$  */
if  $\neg(Fait_p)$  then
    send()
end
if  $\neg(recv_p \vee col_p)$  then /* pas de collision et rien reçu */
     $decider(estimate)$ 
     $halt()$ 
end

```

---

Fig. 2. Diffusion fiable avec des détecteurs de collisions éventuellement parfaits.

si d'autres capteurs voulaient faire une diffusion fiable mais n'avaient pas été choisis par l'algorithme de back-off.

Si un capteur détecte une collision dans cette ronde il s'opposera ensuite à toute décision dans les trois rondes suivantes. Sinon, si un capteur reçoit un message dans cette ronde, la propriété d'intégrité entraîne qu'au moins un capteur voulait faire une diffusion fiable : dans ce cas la décision devra être *True*. Les trois autres rondes servent de confirmation.

La première de ces rondes est une ronde de confirmation sur une décision *False* : n'envoient un message que les capteurs qui refusent une décision *False*. Ce sont ceux qui

soit voulaient faire une diffusion fiable, soit ont reçu un message ou une détection de collision dans la première ronde. Cette ronde de "veto" assure que si au moins un capteur s'oppose à la décision et donc émet dans la ronde alors tous les capteurs le sauront. Si aucun message n'est reçu et s'il n'y a pas eu de détection de collision, la décision *False* est confirmée et le capteur considère est prêt à décider (variable *Fait* mise à *True*). Grâce aux propriétés d'intégrité et de complétude, à la fin de cette première ronde de confirmation, si un capteur s'opposait à une décision *False* alors tous les capteurs refuseront cette décision. D'un autre côté si aucun capteur ne s'oppose à la décision et s'il n'y a pas de collision alors tous les capteurs accepteront cette décision. La propriété d'absence ultime de collision assure qu'ultimement il n'y aura plus de collision si aucun capteur ne s'oppose à cette décision.

La ronde suivante fonctionne de façon symétrique : si un capteur s'oppose à la décision *True* alors il émet et à la fin de la ronde aucun capteur ne considérera que cette décision doit être prise (la variable *Fait* n'est pas modifiée).

La ronde suivante sert à vérifier qu'une décision a bien été prise. Elle fonctionne suivant le même principe, si un capteur n'a pas pris de décision il s'oppose à la décision et il émet. Ainsi à la fin de la ronde si au moins un capteur s'opposait alors aucun capteur ne s'arrêtera.

*Proposition 2*: L'algorithme de la Figure 2 réalise la diffusion fiable avec un détecteur de collision ultimement parfait.

On peut montrer que la terminaison des capteurs peut ne pas être simultanée : dans l'algorithme les capteurs ne décident pas nécessairement dans la même ronde, en effet même si un capteur termine par *halt* dans la ronde  $4n+3$  parce qu'aucun capteur n'a émis dans cette ronde et qu'il n'a pas détecté de collision d'autres capteurs peuvent avoir détecté une collision et ne termineront pas dans cette ronde. L'algorithme assure cependant qu'ils termineront l'algorithme avec la même valeur.

Le fait que la terminaison n'est pas simultanée pose un problème pour enchaîner plusieurs diffusions fiables

#### IV. CALCUL DU MAXIMUM

Dans cette section nous allons supposer que les détecteurs de collision sont parfaits c'est-à-dire qu'ils vérifient toujours la complétude et l'exactitude. Nous allons montrer comment ces détecteurs de collision peuvent être utilisés pour résoudre un problème plus complexe que celui de la diffusion fiable : le problème du maximum. Supposons que les capteurs possèdent des valeurs, il s'agit de trouver le maximum de ces valeurs. Cependant comme certains capteurs peuvent tomber en panne crash et que de ce fait leur valeur peut être inconnue, la valeur qui doit être choisie doit être une des valeurs initiales d'un capteur et doit être supérieure ou égale à toute valeur d'un capteur correct. Plus formellement, chaque capteur  $p$  a une valeur initiale  $v_p$  il s'agit d'un problème de décision vérifiant les propriétés suivantes :

- *Terminaison* : tous les capteurs corrects décident,
- *Accord* : si  $p$  et  $q$  décident, ils décident de la même valeur,



définis ici peuvent servir à réaliser les détecteurs plus généraux de [5], [6].

#### REFERENCES

- [1] M. Abboud, C. Delporte-Gallet, and H. Fauconnier. Sensor networks with collision detectors. Technical report, LIAFA, 2006.
- [2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks : a survey. *Computer Networks*, 38(4) :393–422, 2002.
- [3] T. D. Chandra and S. Toueg. Unreliable failure detectors for reliable distributed systems. *J. ACM*, 43(2) :225–267, Mar. 1996.
- [4] G. Chockler, M. Demirbas, S. Gilbert, N. A. Lynch, C. C. Newport, and T. Nolte. Reconciling the theory and practice of (un)reliable wireless broadcast. In *ICDCS Workshops*, pages 42–48, 2005.
- [5] G. Chockler, M. Demirbas, S. Gilbert, C. Newport, and T. Nolte. Consensus and collision detectors in wireless ad hoc networks. In *PODC '05 : Proceedings of the twenty-fourth annual ACM symposium on Principles of distributed computing*, pages 197–206, New York, NY, USA, 2005. ACM Press.
- [6] G. Chockler, M. Demirbas, S. Gilbert, and C. C. Newport. A middleware framework for robust applications in wireless ad hoc networks. In *Forty-third Annual Allerton Conference on Communication, Control, and Computing*, 2005.
- [7] J. Deng, P. K. Varshney, and Z. J. Haas. A new backoff algorithm for the IEEE 802.11 distributed coordination function. In *Proc. of Communication Networks and Distributed Systems Modeling and Simulation (CNDS '04)*, San Diego, CA, USA, January 18-21 2004.
- [8] V. Hadzilacos and S. Toueg. Fault-tolerant broadcasts and related problems. In S. J. Mullender, editor, *Distributed Systems*, chapter 5, pages 97–145. Addison-Wesley, 1993.
- [9] N. A. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1996.
- [10] K. Nakano and S. Olariu. A survey on leader election protocols for radio networks. In *ISPAN*, pages 71–, 2002.
- [11] G. R. A perspective on multiaccess channels. *IEEE Trans. Inform. Theory*, 31 :124–142, 1985.
- [12] D. E. Willard. Log-logarithmic selection resolution protocols in a multiple access channel. *SIAM J. Comput.*, 15(2) :468–477, 1986.