

# Réseaux de capteurs avec détecteur de collision

Mohssen Abboud maboud@liafa.jussieu.fr

Carole Delporte cd@liafa.jussieu.fr

Hugues Fauconnier hf@liafa.jussieu.fr

## Résumé

On considère 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 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. Enfin nous donnons un algorithme qui implémente la diffusion fiable avec des valeurs en utilisant le calcul successif de maximum.

## 1 Introduction

La technologie des réseaux de capteurs sans fil a beaucoup progressé depuis quelques années [1] et de nombreuses applications basées sur des réseaux de capteurs sont soit réalisées soit en cours de réalisation. Les capteurs peuvent être utilisés dans un milieu hostile comme, par exemple, sur un champ de bataille, en présence d'incendie, d'inondations, de tremblement 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 que concerne les première, 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 [10, 9, 11, 6] 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 [3].

D'un point de vue plus formel, Chockler et al. dans [4, 5] 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 [8] 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 [7] 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. Dans la quatrième section nous prouvons comment utiliser ces détecteurs pour résoudre le problème de diffusion fiable avec des valeurs. Enfin nous concluons en présentant plusieurs extensions à ce travail et un problème ouvert.

## 2 Modèle et définitions

### 2.1 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 si  $p$  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 :

- *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 processus 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.

## 2.2 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  $Rbcast_p$  un booléen indiquant que le capteur  $p$  veut diffuser un message dans la ronde. Le problème de la diffusion fiable [7] 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  $Rbcast_p$  alors pour tous les capteurs  $q$  on a un jour  $del_q \neq False$  et si pour aucun capteur  $p$   $Rbcast_p = True$  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 un capteur correct a voulu émettre et 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.

## 2.3 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* [4, 5]. 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 juste 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 [2], 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  $\diamond\mathcal{P}$ .

### 3 Implémentation de la diffusion fiable

Un détecteur parfait permet facilement de réaliser la diffusion fiable :

**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

---

**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.

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 clair. 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 les 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 désiraient faire  $Rbcast$  : dans ce cas la décision devra être  $False$  sauf 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 soient 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 processus considéré 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 processus 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

---

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

---

```
Initialisation :  $estimate_p \leftarrow Rbcast_p$ 
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  $Rbcast = False$  */
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  $Rbcast = True$  */
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.

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 processus 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.*

**Définition 1** *Soit  $r_0$ , la ronde à partir de laquelle l'exactitude ultime est satisfaite.*

**Lemme 1** *S'il existe une ronde  $4n$ , telle que le capteur  $x$  au début de la ronde  $4n$  à  $Estimate_x = True$  alors  $Estimate_x = True$  dans toutes les rondes  $r \geq 4n$  où  $x$  est vivant.*

PREUVE. Dans aucune des rondes de l'algorithme  $x$  ne peut recevoir la valeur *False*.  $\square$

**Lemme 2** *S'il existe une ronde  $4n$ , telle que si tous les capteurs  $x$  vivants au début de la ronde  $4n$  ont  $Estimate_x = False$  alors  $Estimate_x = False$  dans toutes les rondes  $r \geq 4n$ .*

PREUVE. On montre cette propriété par récurrence. Soit  $P(r)$  la propriété : tous les capteurs  $x$  vivants au début de la ronde  $4r$  ont  $Estimate_x = False$ . Par hypothèse  $P(n)$  est vrai. On suppose que  $P(r)$  est vrai.  $Estimate_x$  n'est modifié que dans la ronde  $4r$ . Aucun message n'est émis à la ronde  $4r$ , par intégrité aucun message n'est reçu, donc  $Estimate_x$  reste à *False*. Ce qui implique  $P(r+1)$  et la lemme.  $\square$

Nous allons maintenant montrer que notre algorithme satisfait les propriétés de la diffusion fiable.

## Terminaison

Dans cette partie on montre que l'algorithme de la Figure 2 satisfait la propriété de Terminaison de la diffusion fiable.

**Lemme 3** *Pour tout  $n$ , si il existe un capteur  $q$ , vivant à la fin de la ronde  $4n+1$  tel que  $veto0_q = True$  alors tous les capteurs qui terminent la ronde ont  $Fait_p = False$ .*

PREUVE. Si  $veto0_q = True$  et  $q$  est vivant à la fin de la ronde alors  $q$  émet un message. Par la propriété de complétude ce message est reçu ou une détection de collision est faite et donc tous les capteurs  $x$  vivants à la fin de cette ronde ne modifient pas leurs valeurs de  $Fait_x$  et ont donc  $Fait_x = False$ .  $\square$

**Lemme 4** *Pour tout  $n$  tel que  $4n+1 > r_0$ , si tous les capteurs  $q$ , commençant la ronde  $4n+1$  ont  $veto0_q = False$  alors tous les capteurs  $x$  qui terminent la ronde ont  $Fait_x = True$ .*

PREUVE. Si pour tous les capteurs  $q$  qui commencent la ronde  $4n + 1$   $veto0_q = False$ , alors aucun processus n'émet dans cette ronde. Si  $x$  vivant à la fin de la ronde alors par la propriété d'intégrité  $del_x = False$ . Par la propriété d'exactitude ultime  $col_x = False$ . Par conséquent  $Fait_x = True$ .  $\square$

**Lemme 5** *Pour tout  $n$ , si il existe un capteur  $q$ , vivant à la fin de la ronde  $4n + 2$  tel que  $veto1_q = True$  alors tous les capteurs  $x$  qui terminent la ronde ne modifient pas leurs valeurs de  $Fait_x$ .*

PREUVE. (Similaire à la preuve du Lemme 3) Si  $veto1_q = True$  et  $q$  est vivant à la fin de la ronde alors  $q$  émet un message. Par la propriété de complétude ce message est reçu ou une détection de collision est faite et donc tous les capteurs  $x$  vivants à la fin de cette ronde ne modifient pas leurs valeurs de  $Fait_x$ .  $\square$

**Lemme 6** *Pour tout  $n$  tel que  $4n + 2 > r_0$ , si tous les capteurs  $q$ , commençant la ronde  $4n + 2$  ont  $veto1_q = False$  alors tous les capteurs  $x$  qui terminent la ronde ont  $Fait_x = True$ .*

PREUVE. Identique à la preuve du Lemme 4.  $\square$

**Lemme 7** *Pour tout  $n$ , si il existe un capteur  $q$ , vivant à la fin de la ronde  $4n + 3$  tel que  $Fait_q = False$  alors aucun capteur  $x$  exécutant la ronde  $4n + 3$  ne décide.*

PREUVE. (Similaire à la preuve du Lemme 3) Si  $Fait_q = False$  et  $q$  est vivant à la fin de la ronde alors  $q$  émet un message. Par la propriété de complétude ce message est reçu ou une détection de collision est faite et donc aucun capteur  $x$  exécutant cette ronde ne décident.  $\square$

**Lemme 8** *Pour tout  $n$  tel que  $4n + 3 > r_0$ , si tous les capteurs  $q$ , commençant la ronde  $4n + 3$  ont  $Fait_q = True$  alors tous les capteurs  $x$  vivant à la fin de la ronde ont décidé et terminé l'algorithme.*

PREUVE. (Similaire à la preuve du Lemme 4) Si pour tous les capteurs  $q$  qui commencent la ronde  $4n + 1$  ont  $Fait_q = True$ , alors aucun capteur n'émet dans cette ronde. Si  $x$  vivant à la fin de la ronde alors par la propriété d'intégrité  $del_x = False$ . Par la propriété d'exactitude ultime  $col_x = False$ . Par conséquent le capteur décide et termine l'algorithme.  $\square$

**Définition 2** *Soit  $r_1$ , la ronde telle que tous les capteurs incorrects sont tombés en panne avant la ronde  $r_1$ . Soit  $r_2$ , la ronde satisfaisant la propriété d'absence ultime de collisions, après  $r_2$  s'il n'y a qu'une émission dans la ronde le message est reçu sans collision.*

**Lemme 9** (Terminaison) *Pour tout capteur correct  $p$  un jour  $del_p$  sera différent de  $\perp$ .*

PREUVE. Supposons qu'il existe un capteur correct  $p$  tel que  $del_p$  est toujours différent de  $\perp$ . C'est à dire que  $p$  ne décide jamais. Comme  $p$  est correct,  $p$  exécute une infinité de rondes. Soit  $r_3$  tel que plus aucun capteur ne décide après  $r_3$ . Soit  $r$ , tel que  $4r > \max(r_0, r_1, r_2, r_3)$ .

**cas 1** : Aucun des capteurs  $q$  exécutant la ronde  $4r$  ne veut émettre i.e.  $estimate_q = False$ .

La propriété d'intégrité assure que pour tout capteur  $x$  exécutant la ronde  $4r$  :  $recv_x = False$ . Par définition de  $r$ , on a à cette même ronde  $col_x = False$ . Donc  $veto0_x = False$  et  $veto1_x = True$ . Par le Lemme 4, tous les capteurs  $x$  qui terminent la ronde ont  $Fait_x = True$ . Par le Lemme 5 aucun capteur ne modifie sa valeur de  $Fait$  qui reste à  $True$ . Par le Lemme 8, tous les capteurs vivants décident et terminent. Donc en particulier  $p$ , ce qui est en contradiction avec l'hypothèse que  $p$  ne décide pas.

**cas 2** : Un des capteurs exécutant la ronde  $4r$  veut émettre i.e il existe un capteur  $q$  t.q  $Estimate_q = True$ .

D'après le Lemme 1,  $Estimate_q$  est à  $True$  dans toutes les rondes que  $q$  exécute. Par définition de  $r$ ,  $q$  exécute une infinité de rondes. Donc pour une infinité de rondes i.e  $\{4v | v \geq r\}$ ,  $q$  veut émettre. Donc par la propriété du back-off, pour une infinité de ces rondes  $active_q$  sera vrai et pour tous les autres processus  $active$  sera  $False$ . Soit  $4v$  une telle ronde.

La propriété d'absence ultime de collision assure que pour tous les capteurs  $x$   $recv_x^{4v} = True$ . Donc :  $veto0_x^{4v} = True$  et  $veto1_x^{4v} = False$ .

Par le Lemme 6, tous les capteurs, lors de l'exécution de  $x$  qui terminent la ronde  $4v + 2$  ont  $Fait_x = True$ . Par le Lemme 8, tous les capteurs décident et terminent. Donc en particulier  $p$ , ce qui est en contradiction avec l'hypothèse que  $p$  ne décide pas.

□

## Accord

Dans cette partie on montre que l'algorithme de la Figure 2 satisfait la propriété d'accord de la diffusion fiable.

**Lemme 10** (Accord) *Pour tout  $p$  et pour tout  $q$  si  $del_p \neq \perp$  et  $del_q \neq \perp$  alors  $del_p = del_q$*

PREUVE. Soit  $r_p$  et  $r_q$  deux rondes telles que  $p$  décide à la ronde  $4r_p + 3$  et  $q$  à la ronde  $4r_q + 3$ . Supposons  $r_p \leq r_q$ . Quand  $p$  décide  $recv_p = False$  et  $col_p = False$  donc par complétude aucun message n'a été envoyé dans la ronde  $4r_p + 3$ . Donc tous les capteurs  $x$  vivants lors de l'émission des messages de cette ronde ont  $Fait_x = True$ . Par conséquent  $Fait_x$  a été affecté à la valeur  $True$  dans la ronde  $4r_p + 1$  ou dans la ronde  $4r_p + 2$ .

**cas 1**  $Fait_p$  a été affecté à la valeur  $True$  dans la ronde  $4r_p + 1$ .

A cette ronde  $recv_p = False$  et  $col_p = False$  donc par complétude aucun message n'a été envoyé dans la ronde  $4r_p + 1$ . Donc pour tous les capteurs vivants  $x$  dans cette ronde  $veto0_x = False$ .  $veto0$  est modifié dans la ronde  $4r_p$ . Il ne peut recevoir la valeur  $False$  que de  $Estimate$  donc tous les capteurs ont  $Estimate$  à  $False$ . Lorsque  $p$  décide dans la ronde  $4r_p + 3$ ,  $p$  décide  $False$ . Si  $q$  décide dans cette ronde il décide aussi  $False$ . Si  $q$  décide dans une autre ronde, il décide sur la

valeur de  $Estimate_q$ . Or par le lemme 2,  $Estimate_q$  ne changera plus de valeur et restera à  $False$  donc  $q$  décidera  $False$  dans la ronde  $r_q$ .

**cas 2**  $Fait_p$  a été affecté à la valeur  $True$  dans la ronde  $4r_p + 2$ .

A cette ronde  $recv_p = False$  et  $col_p = False$  donc par complétude aucun message n'a été envoyé dans la ronde  $4r_p + 2$ . Donc pour tous les capteurs vivants  $x$  dans cette ronde  $veto1_x = False$ .  $veto1$  est modifié dans la ronde  $4r_p$ . Il reçoit la valeur  $False$  en même temps que  $Estimate$  reçoit la valeur  $True$ . Lorsque  $p$  décide dans la ronde  $4p_r + 3$ ,  $p$  décide  $True$ . Si  $q$  décide dans cette ronde il décide aussi  $True$ . Si  $q$  décide dans une autre ronde, il décide sur la valeur de  $Estimate_q$ . Or par le lemme 1,  $Estimate_q$  ne changera plus de valeur et restera à  $True$  donc  $q$  décidera  $True$  dans la ronde  $r_q$ .  $\square$

## Validité

Dans cette partie on montre que l'algorithme de la Figure 2 satisfait la propriété de Validité de la diffusion fiable.

**Lemme 11** *S'il existe un capteur correct  $p$  tel que  $Rbroadcast_p$  alors pour tous les capteurs  $q$  on a un jour  $del_q \neq False$ .*

PREUVE. Si  $Rbroadcast_p$  est vrai alors  $Estimate_p^0 = True$ . Par le lemme 1, pour toute les rondes que  $p$  exécute,  $Estimate_p$  est  $True$ . Par le lemme 9 (Terminaison),  $p$  décide. Or  $p$  décide sa valeur de  $Estimate_p$  donc  $p$  décide  $True$ . Si  $q$  décide alors par le lemme 10 (Accord),  $q$  décide  $True$ . Donc la valeur de  $del_q$  est soit  $\perp$  soit  $True$ .  $\square$

**Lemme 12** *Si pour aucun capteur  $p$   $Rbroadcast_p = True$  alors pour tous les capteurs  $q$  on a un jour  $del_q \neq True$ .*

PREUVE. Tous les capteurs  $x$  ont leur valeur de  $Rbroadcast_q$  à  $False$  et donc  $Estimate_x^0 = False$ . Par le lemme 2 tous les capteurs  $x$  gardent  $False$  comme valeur de  $Estimate_x$ . Par le lemme 9 (Terminaison),  $p$  décide. Or  $p$  décide sa valeur de  $Estimate_p$  donc  $p$  décide  $False$ . Si  $q$  décide alors par le lemme 10 (Accord),  $q$  décide  $False$ . Donc la valeur de  $del_q$  est soit  $\perp$  soit  $False$ .  $\square$

Des Lemmes 11 et 12, on a :

**Lemme 13** (Validité) *S'il existe un capteur correct  $p$  tel que  $Rbroadcast_p$  alors pour tous les capteurs  $q$  on a un jour  $del_q \neq False$ . Si pour aucun capteur  $p$   $Rbroadcast_p = True$  alors pour tous les capteurs  $q$  on a un jour  $del_q \neq True$ .*

On peut montrer que la terminaison des capteurs peut ne pas être simultanée :

**Lemme 14** *Dans l'algorithme de la Figure 2 les capteurs ne décident pas nécessairement dans la même ronde.*

PREUVE. 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. Par contre d'après les Lemmes 9, 10 et 13, l'algorithme assure cependant qu'ils termineront l'algorithme avec la même valeur.  $\square$

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

## 4 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 entières, 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. De plus si aucun capteur correct ne propose de valeur la décision peut être  $\perp$ . Plus formellement, chaque capteur  $p$  propose une valeur  $v_p$  ( $\emptyset$  si  $p$  ne propose pas de valeur). 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,
- *Validité* : Si  $p$  décide la valeur  $v$  alors  $v = v_q$  pour un certain capteur  $q$  et pour tout capteur correct  $r$   $v \geq v_r$ . Si  $p$  décide  $\perp$  alors pour tous les capteurs corrects  $p$   $v_p = \emptyset$ .

L'algorithme de la Figure 3 réalise ce calcul du maximum. Cet algorithme se décompose en deux phases. Dans la première phase, les capteurs déterminent le nombre de bits du maximum. La deuxième phase réalise un accord bit à bit sur le maximum. Pour chaque bit, un capteur va émettre si sa valeur peut être le maximum (c'est-à-dire si le nombre de bits du maximum qui a été déterminé dans la première phase est égal au nombre de bits de sa valeur) et si le bit considéré est à un. Si un capteur ne reçoit rien et ne détecte pas de collision à cette ronde cela signifie que le bit considéré est à zéro ou que le capteur qui avait le maximum est crashé. On fait donc dans ce cas une autre ronde de confirmation pour vérifier si le bit est à zéro.

Comme les capteurs peuvent tomber en panne pendant l'algorithme les deux phases sont répétées jusqu'à ce que l'algorithme termine.

Si les deux phases ont lieu sans nouvelle panne de capteur, l'algorithme se termine en un nombre de rondes de l'ordre de la taille du maximum. Chaque nouvelle panne provoque au pire une nouvelle phase. On a :

**Proposition 3** *L'algorithme de la Figure 3 résout le problème du calcul du maximum avec un détecteur de collision parfait en un nombre de rondes de l'ordre de la taille des entrées multipliée par le nombre de pannes.*

Nous devons prouver que l'algorithme satisfait les trois propriétés de la spécification du maximum dans le cas où les capteurs peuvent tomber en panne.

**Lemme 15** *Si tous les capteurs commencent la  $j$ -ème phase 1 à la ronde  $x$ , alors il existe une ronde  $x'$ ,  $x' \geq x$ , telle que tous les capteurs qui sont vivants à la fin de*

---

**Algorithme 3** : Code pour chaque capteur  $p$  :  $\text{MAX}(v_p)$ 

---

```
/*  $v_p$  est la valeur proposée par  $p$ , par convention  $v_p=0$  si  $p$  propose  $\emptyset$  */
/*  $v[i]$  est le  $i$ -ème bit de  $v$ ,  $\&$  est l'opérateur et bit-à-bit */
estimate  $\leftarrow v_p$ ; if ( $v_p = 0$ ) then
    nbsize = 1
end
nbsize  $\leftarrow \text{Log}_2(v_p) + 1$  /* nbsize est le nombre de bits de  $v_p$  */
decide  $\leftarrow \text{False}$ ; phase  $\leftarrow \text{phase1}$ 
while ( $\neg \text{decide}$ ) do
    if ( $\text{phase} = \text{phase1}$ ) then
        continue  $\leftarrow \text{True}$ ;  $r \leftarrow 1$ 
        while continue do
            if ( $r < \text{nbsize}$ ) then
                send()
            end
            if ( $\neg \text{recv}_p \wedge \neg \text{col}_p$ ) then
                continue  $\leftarrow \text{False}$ 
            end
             $r \leftarrow r + 1$ 
        end
        phase  $\leftarrow \text{phase2}$ ; size  $\leftarrow r - 1$ 
    end
    if ( $\text{phase} = \text{phase2}$ ) then
        max  $\leftarrow 0$ 
        while ( $\text{size} > 0$ )  $\wedge$  ( $\text{phase} = \text{phase2}$ ) do
            continue  $\leftarrow \text{True}$ 
            if ( $\text{estimate}[\text{size}] = 1$ )  $\wedge$  ( $(\text{estimate} \& \text{max}) = \text{max}$ ) then
                send()
            end
            if ( $\text{recv}_p \vee \text{col}_p$ ) then
                max[size]  $\leftarrow 1$ ; continue  $\leftarrow \text{False}$ 
            end
            if continue then
                if ( $\text{estimate}[\text{size}] = 0$ )  $\wedge$  ( $(\text{estimate} \& \text{max}) = \text{max}$ ) then
                    send()
                end
                if ( $\text{recv}_p \vee \text{col}_p$ ) then
                    max[size]  $\leftarrow 0$ 
                else
                    phase  $\leftarrow \text{phase1}$ 
                end
            end
            size  $\leftarrow \text{size} - 1$ 
        end
        if ( $\text{size} = 0$ ) then
            decide  $\leftarrow \text{True}$ 
        end
    end
end
if ( $\text{max} = 0$ ) then
    return  $\perp$ 
end
return max
```

---

la ronde  $x'$ , terminent la  $j$ -ème phase 1 à la ronde  $x'$  avec la même valeur de  $size$ . De plus aucun capteur ne termine cette phase entre les rondes  $x$  et  $x'$ .

PREUVE. Si un capteur émet dans une ronde, par la propriété de complétude, toute collision est détectée. Donc si pour un capteur  $p : r_p < nbsize_p$ , alors tous les capteurs  $q$  qui exécutent cette ronde trouvent  $recv_q \vee col_q$  et continuent la phase 1.

Si aucun capteur n'émet dans une ronde, par la propriété d'intégrité et d'exactitude aucun message n'est reçu et aucune collision n'est détectée. Donc si pour tous les capteurs  $p : (r_p \geq nbsize_p)$ , alors tous les capteurs  $q$  qui exécutent cette ronde trouvent  $(\neg recv_q \wedge \neg col_q)$  et arrêtent la phase 1.

Comme  $r$  est incrémenté dans chaque ronde de la phase 1, il existe une ronde  $x'$  telle que pour la première fois à la ronde  $x'$  tous les capteurs  $p$  vivants ont  $r_p \geq nbsize_p$ , et tous les capteurs qui exécutent cette ronde terminent la  $j$ -ème phase 1.

La valeur de  $size$  à la fin de la phase est  $x' - x + 1$ , donc tous les capteurs ont la même valeur pour  $size$ .  $\square$

**Lemme 16** *Si tous les capteurs commencent la  $j$ -ème phase 2 à la ronde  $x$  avec la même valeur pour  $size$ , alors il existe une ronde  $x'$ ,  $x' > x$ , telle que tous les capteurs qui sont vivants à la fin de la ronde  $x'$ , terminent la  $j$ -ème phase 2 avec la même valeur de  $size$  et de  $max$ . De plus aucun capteur ne termine cette phase entre les rondes  $x$  et  $x'$ .*

PREUVE. Si un capteur émet dans une ronde, par la propriété de complétude, toute collision est détectée. Donc si pour un capteur  $p : (estimate_p[size_p] = 1) \wedge ((estimate_p \& max_p) = max_p)$ , alors tous les capteurs  $q$  qui exécutent cette ronde trouvent  $recv_q \vee col_q$  et affectent  $continue_q$  à faux et  $max_q[size_q]$  à 1 puis décrémentent la valeur de  $size_q$ . Si  $size_q = 0$  alors ils décident et terminent ensemble la phase.

Si pour tous les capteurs  $p : \neg (estimate_p[size_p] = 1) \vee \neg ((estimate_p \& max_p) = max_p)$ , alors pour tous les capteurs  $p$  qui exécutent cette ronde  $(\neg recv_p \wedge \neg col_p)$  et ils gardent  $continue_p$  à vrai.

Dans la ronde suivante, si pour un capteur  $p : (estimate_p[size_p] = 0) \wedge ((estimate_p \& max_p) = max_p)$ , alors tous les capteurs  $q$  qui exécutent cette ronde trouvent  $(recv_q \vee col_q)$ , affectent  $max_q[size_q]$  à 0 et continuent la phase 2.

Si par contre pour tous les capteurs  $p : (estimate_p[size_p] \neq 0) \vee ((estimate_p \& max_p) \neq max_p)$ , alors pour tous les capteurs  $p$  qui exécutent cette ronde  $\neg (recv_p \vee col_p)$  et ils terminent cette phase 2 pour recommencer une phase 1.

On a montré que tous les capteurs décident ou terminent cette phase 2 pour exécuter la phase 1 suivante à la même ronde. Il reste à montrer que les capteurs ne restent pas dans cette phase indéfiniment. A chaque itération de boucle *while*  $size$  décroît et si  $size$  est nulle, les capteurs décident. Donc au plus tard à la ronde  $x + 2 * size - 1$  les capteurs ont terminé cette phase 2. Si  $s$  est la valeur de  $size$  quand les capteurs commencent la phase 2, la valeur de  $size$  à la fin de la phase est donc  $s - (x' - x + 1)/2$ , donc tous les capteurs ont la même valeur pour  $size$  et à chaque ronde  $max$  a été affectée de la même manière.  $\square$

**Lemme 17** *Si un capteur commence la  $j$ -ème phase 1 à la ronde  $x$ , alors tous les capteurs qui sont vivants à la ronde  $x$  commencent la même phase. De plus il existe*

une ronde  $x'$ ,  $x' \geq x$ , telle que tous les capteurs qui sont vivants à la fin de la ronde  $x'$ , terminent la  $j$ -ème phase 1 à la ronde  $x'$  avec la même valeur de  $size$ . De plus aucun capteur ne termine cette phase entre les rondes  $x$  et  $x'$ .

Si un capteur commence la  $j$ -ème phase 2 à la ronde  $x$ , alors tous les capteurs qui sont vivants à la ronde  $x$  commencent la même phase. De plus il existe une ronde  $x'$ ,  $x' > x$ , telle que tous les capteurs qui sont vivants à la fin de la ronde  $x'$ , terminent la  $j$ -ème phase 2 à la ronde  $x'$  avec la même valeur de  $size$  et de  $max$ .

PREUVE. Tous les capteurs commencent la première phase 1 en même temps, par les lemmes 15 et 16 on obtient le résultat.  $\square$

## Terminaison

Dans cette partie on montre que l'algorithme de la Figure 3 satisfait la propriété de Terminaison du calcul du  $max$ .

**Définition 3** Soit  $r_0$ , la ronde telle que tous les capteurs incorrects sont tombés en panne avant la ronde  $r_0$ .

**Lemme 18** (Terminaison) Tous les capteurs corrects décident.

PREUVE. Supposons qu'un capteur correct ne décide pas, par le lemme 17, aucun capteur correct n'a décidé et tous les capteurs corrects exécutent une infinité de phase 1 et de phase 2 ensemble. Soit  $r_1 > r_0$ , la première ronde après  $r_0$ , où les capteurs corrects commencent une phase 1. Soit  $m$  le maximum des valeurs proposées par les capteurs corrects et  $nbs$  le nombre de bits utilisé pour représenter  $m$ . On remarque que  $nbs \geq 1$ .

Soit  $p$  le capteur ayant  $m$  comme valeur proposée. A chaque ronde entre  $r_1$  et  $r_1 + nbs - 2$ ,  $p$  émet et par exactitude et intégrité tous les capteurs continuent. A la ronde  $r_1 + nbs - 1$ , tous les capteurs ont  $r \geq nbsize$  et personne n'émet terminant ainsi la phase 1 avec  $nbsize = nbs$ .

On montre que, tant que  $size_p$  est positif à chaque itération de la boucle *while* (1)  $p$  émet, (2) si pour tout capteur correct  $q$ , à la  $x + 1$ -ème itération, les  $x$  bits de poids forts de  $max_q$  sont égaux à ceux de  $m$  au début de l'itération alors les  $x + 1$  bits de poids forts de  $max_q$  sont égaux à ceux de  $m$  à la fin de l'itération. Comme  $nbs \geq 1$ , et  $size = nbs$  on a toujours au moins une itération. A la première itération de la phase 2, à la ronde  $r_1 + nbs$ , il y a 2 cas :

(a) Si  $m=0$  alors personne n'émet dans cette ronde et  $p$  émet dans la suivante car  $(estimate[size] = 0) \wedge ((estimate \& max) = max)$ . Puis tous les capteurs ont  $size=0$  et terminent l'algorithme.

(b) Si  $m \neq 0$  alors  $p$  émet car  $(estimate_p[size_p] = 1) \wedge ((estimate_p \& max_p) = max_p)$ . Par intégrité et exactitude tous les capteurs  $q$  ont  $max_q[nbs] = 1$  y compris  $p$ .

On suppose qu'on a montré les propriétés (1) et (2) jusqu'à la  $x$ -ème itération. On les montre pour l'itération suivante. Si  $(estimate_p[size_p] = 1)$ , par hypothèse  $((estimate_p \& max_p) = max_p)$  alors  $p$  émet, par intégrité et exactitude tous les capteurs adoptent  $max_s[size_q] = 1$  et aucun capteur n'émettra dans la ronde suivante. Si  $(estimate_p[size_p] = 0)$  alors aucun capteur  $q$  ne peut avoir  $(estimate_q[size_q] = 1) \wedge ((estimate_q \& max_q) = max_q)$  car  $p$  à la valeur maximale et les  $x$  bits de poids forts de  $max_q$  sont égaux à ceux de  $m$ .  $p$  émettra à cette ronde

et tous les capteurs adopteront sa valeur. Donc dans tous les cas les  $x + 1$  bits de poids forts de  $max_q$  sont égaux à ceux de  $m$  à la fin de l'itération.

$size$  décroît de 1 à chaque itération, quand  $size = 0$ , tous les capteurs décident et terminent, contredisant l'hypothèse. □

## Accord

Dans cette partie on montre que l'algorithme de la Figure 3 satisfait la propriété d'accord du max.

**Lemme 19** (*Accord*) *Si  $p$  et  $q$  décident, ils décident de la même valeur.*

PREUVE. Soient  $p$  et  $q$  deux capteurs tels que  $p$  décide en premier. On considère la phase 2 dans laquelle  $p$  décide. D'après le Lemme 17 les deux capteurs terminent la deuxième phase avec la même valeur de  $size$  et de  $max$  donc si  $p$  décide,  $q$  décide aussi et sur la même valeur. □

## Validité

Dans cette partie on montre que l'algorithme de la Figure 3 satisfait la propriété de validité du max.

**Lemme 20** *Si  $max_p = v$  quand  $p$  décide alors il existe un capteur  $q$  tel que  $estimate_q = v$*

PREUVE. On considère la dernière itération de la boucle *while* (pour  $size = 1$ ) avant la décision de  $p$ , il y a 3 cas :

(1) Soit pour tous les capteurs qui exécutent cette ronde  $((estimate \& max) = max)$  n'est pas vérifié, dans ce cas aucun capteur n'émet et  $p$  ne décidera pas.

(2) Soit certains capteurs ont  $((estimate \& max) = max)$  et  $estimate[1] = 1$ , dans ce cas ils émettent et c'est leur valeur qui est décidée, donc  $v$  est la valeur proposée par au moins un capteur.

(3) Soit certains capteurs ont  $((estimate \& max) = max)$  et parmi eux aucun n'a  $estimate[1] = 1$ , la seule possibilité est que  $estimate[1] = 0$ . Donc aucun capteur n'a émis dans la première ronde mais ils émettent dans la deuxième et c'est leur valeur qui est décidée, donc  $v$  est la valeur d' $estimate$  d'au moins un capteur. □

**Lemme 21** *Si  $max_p = v$  quand  $p$  décide alors  $v \geq estimate_q$  pour tout capteur correct  $q$ .*

PREUVE. Supposons qu'il existe un capteur correct  $q$  tel que  $estimate_q > v$ . Soit  $nbs$  le nombre de bits pour représenter  $v$ . On a 2 cas :

**cas 1** :  $nbsize_q > nbs$  : lors de la phase 1  $q$  émet à chaque ronde, donc par exactitude et complétude aucun capteur ne peut arrêter la phase 1 avant  $nbsize_q$  rondes. Ce cas est donc impossible.

**cas 2 :**  $nbsize_q = nbs$ . Soit  $x$  le premier bit différent dans  $estimate_q$  et  $v$  à partir des bits de poids forts, comme par hypothèse  $v < estimate_q$  alors  $estimate_q[x] = 1$  et  $v[x] = 0$ . Si on considère la phase 2 dans laquelle  $p$  décide,  $q$  va émettre dans la première ronde de la  $x$ -ème iteration. Par exactitude et intégrité, tous les capteurs vont adopter 1 pour le  $x$ -ème bit et aucun n'émettra dans la deuxième ronde. La valeur  $v$  ne pourra être décidée. Ce cas est donc aussi impossible. □

**Lemme 22** *Si  $p$  décide  $\perp$  alors pour tous les capteurs corrects  $p$   $v_p = \emptyset$ .*

PREUVE. Si  $p$  décide  $\perp$  alors  $max_p$  à pour valeur 0 au moment de la décision, d'après le lemme 21 tous les capteurs corrects ont pour  $estimate$  une valeur inférieure donc ont  $estimate$  à 0. C'est à dire, par convention, ont proposé  $\emptyset$ . □

Des lemmes 20, 21 et 22, on a :

**Lemme 23 (Validité)** *Si  $p$  décide la valeur  $v$  alors  $v = v_q$  pour un certain capteur  $q$  et pour tout capteur correct  $q$   $v \geq v_q$ . Si  $p$  décide  $\perp$  alors pour tous les capteurs corrects  $q$   $v_q = \emptyset$ .*

**Proposition 4** *L'algorithme de la Figure 3 qui calcule le maximum, en utilisant des détecteurs de collision parfaits nous garantit que tous les capteurs décideront sur la valeur maximale dans la même ronde.*

PREUVE. Par la propriété de terminaison (lemme 18) tous les capteurs corrects décident. Cette décision se produit quand  $size$  est nulle. La décision est obtenue à la fin d'une phase 2. D'après le lemme 17, tous les capteurs obtiennent la condition de décision à la même ronde. □

## 5 Diffusion fiable avec des valeurs

Dans cette section nous supposons que les détecteurs de collision sont parfaits. Nous allons montrer comment en utilisant ces détecteurs de collision on peut résoudre le problème de la diffusion fiable avec des valeurs dans les réseaux de capteurs. Les capteurs qui ont une valeur à diffuser proposent cette valeur à la diffusion fiable, les capteurs qui n'ont pas de valeur à diffuser ne propose aucune valeur. Les capteurs décident sur un ensemble  $Dif$ , l'ensemble des messages délivrés, qui contient au moins toutes les valeurs diffusées par les corrects. De plus si deux capteurs décident, ils décident sur le même ensemble  $Dif$ . Comme les capteurs n'ont pas d'identité si un message  $m$  est diffusé plusieurs capteurs, la multiplicité ne sera pas conservée dans l'ensemble des messages délivrés.

Plus précisément, supposons que les capteurs possèdent des valeurs, un protocole de diffusion fiable avec des valeurs doit assurer les propriétés suivantes :

- *Terminaison* : Pour tout capteur correct  $p$ ,  $p$  décide.

- *Accord* : Pour tout capteur  $p$  et tout capteur  $q$ , si  $p$  et  $q$  décident alors  $Dif_p = Dif_q$
- *Validité* : S’il existe un capteur correct  $p$  qui a une valeur  $v_p$  à diffuser alors pour tous les capteurs  $q$  qui décident,  $v_p$  appartient à  $Dif_q$ . Si pour un capteur  $q$ ,  $v$  appartient à  $Dif_q$  alors il existe un capteur  $p$  tel que  $v$  a été proposée par  $p$ .

Nous allons présenter un algorithme qui implémente ce protocole par calcul itéré du maximum. Dans cette approche on utilise l’algorithme qui calcule le maximum pour construire un algorithme qui fait la diffusion fiable avec des valeurs. A partir de l’algorithme du calcul du maximum qui termine simultanément pour tous les capteurs (tel l’algorithme proposé dans la section précédente), il est en effet facile de réaliser une diffusion fiable avec décision : il suffit de calculer les max successifs en éliminant à chaque itération le max trouvé dans l’itération jusqu’à ce que plus aucun capteur ne propose de valeur. Dans chaque itération on assure que tous les capteurs sont d’accord sur la valeur du maximum existante parmi eux, puis on ajoute cette valeur à  $Dif$  (l’ensemble des valeurs délivrées par le capteur). Enfin tous les capteurs qui avait cette valeur à diffuser proposeront  $\emptyset$  dans les appels suivants au max. L’algorithme termine et les capteurs décident quand plus aucun capteur ne propose au max une valeur différente de  $\emptyset$ . Ainsi on peut garantir que par le calcul itéré du maximum on aura pour un capteur  $i$  toutes les valeurs diffusées dans  $Dif_i$ .

---

**Algorithme 4** : Diffusion fiable

---

```

 $val_p$  la valeur à diffuser par le capteur  $p$ 
 $val_p = \emptyset$  si  $p$  n’a pas de valeur à diffuser
 $Dif_p \leftarrow \emptyset$ 
 $Fait_p \leftarrow false$ 
 $V_p \leftarrow val_p$ 
while  $\neg Fait_p$  do
   $v \leftarrow MAX(V_p)$ 
  if  $v = V_p$  then
     $V_p \leftarrow \emptyset$ 
  end
  if  $v = \perp$  then
     $Fait_p \leftarrow true$ 
  else
     $Dif_p \leftarrow Dif_p \cup \{v\}$ 
  end
end
return  $Dif_p$ 

```

---

FIG. 4 – Diffusion fiable avec des valeurs à partir de MAX.

**Proposition 5** *En utilisant l’algorithme de la figure 3, l’algorithme de la figure 4 réalise la diffusion fiable avec des valeurs.*

**Lemme 24** *Pour tout  $i$ , si un capteur correct commencent le  $i$ -ème appel à max à la ronde  $k$ , alors tout capteur vivant à la ronde  $k$  fait son  $i$ -ème appel au max et aucun capteur ne fait son  $i$ -ème appel au max à une ronde différente de  $k$*

PREUVE. Tous les capteurs commencent l'algorithme en même temps. Par la proposition 4 tous les capteurs finissent les appels à max en même temps.  $\square$

**Terminaison :**

Dans cette partie on montre que l'algorithme de la Figure 4 satisfait la propriété de Terminaison de la diffusion fiable avec valeurs.

**Lemme 25** *(Terminaison) Pour tout capteur correct  $p$ ,  $p$  décide.*

PREUVE. Soit  $W(k)$  l'ensemble des valeurs données en argument à max lors de la  $k$ -ème itération de la boucle *while*. Après l'appel à max, par la propriété de terminaison de max, les capteurs terminent l'appel. Par la propriété d'accord du max, tous les capteurs vivants obtiennent la même valeur au retour de l'appel. Par la propriété de validité du max, cette valeur est une des valeurs de  $W(k)$ . Soit cette valeur est égale à  $\perp$ , et dans ce cas l'algorithme termine. Soit cette valeur est différente de  $\perp$ , dans ce cas,  $W(k+1) \subset W(k)$ . Les  $W(k)$  décroissent, après  $|W(0)|$  itérations de la boucle *while* au plus, tous les capteurs proposent  $\emptyset$ . Par la propriété de validité du max, la valeur retournée par max est  $\perp$ , et, l'algorithme termine.  $\square$

**Accord :**

Dans cette partie on montre que l'algorithme de la Figure 4 satisfait la propriété d'Accord de la diffusion fiable avec valeurs.

**Lemme 26** *(Accord) Pour tout capteur  $p$  et tout capteur  $q$ , si  $p$  et  $q$  décident alors on a  $Dif_p = Dif_q$ .*

PREUVE. D'après le lemme 24,  $p$  et  $q$  exécutent le même nombre d'itérations avant de décider. D'après la propriété d'accord du max, les capteurs obtiennent la même valeur à chaque itération. Donc on a  $Dif_p = Dif_q$ .  $\square$

**Validité :**

Dans cette partie on montre que l'algorithme de la Figure 4 satisfait la propriété de Validité de la diffusion fiable avec valeurs.

**Lemme 27** *(Validité) S'il existe un capteur correct  $p$  qui a une valeur  $v_p$  à diffuser alors pour tous les capteurs  $q$  qui décident,  $v_p$  appartient à  $Dif_q$ . Si pour un capteur  $q$ ,  $v$  appartient à  $Dif_q$  alors il existe un capteur  $p$  tel que  $v$  a été proposée par  $p$ .*

PREUVE. Soit  $p$  un capteur correct qui a une valeur  $v_p$  à diffuser et  $q$  un capteur qui décide, supposons que  $v_p$  n'appartiennent pas  $Dif_q$ . Par la propriété d'accord (lemme 26) et de terminaison (lemme 25) de la diffusion fiable avec des valeurs  $p$  décide  $Dif_p$  avec  $Dif_p = Dif_q$ . On considère la dernière itération de la boucle *while* dans laquelle  $q$  à  $\perp$  comme valeur de retour du max. Comme  $v_p$  n'appartiennent pas à  $Dif_p$ ,  $p$  appelle max avec comme paramètre la valeur  $v_p$ . Par la propriété de validité du max, la valeur retournée par max ne peut être  $\perp$ . Donc l'hypothèse que  $v_p$  n'appartiennent pas  $Dif_q$  est absurde.

Si pour un capteur  $q$ ,  $v$  appartient à  $Dif_q$  alors  $v$  est obtenue par le retour d'un appel à  $\max$ , par la propriété de validité du  $\max$  il existe un capteur  $p$  tel que  $v$  a été proposée par  $p$ .  $\square$

## 6 Conclusion et perspectives

Nous avons présenté un modèle pour des réseaux de capteurs anonymes communiquant par diffusion radio et utilisant des détecteurs de collision. On a considéré ici deux classes de détecteurs de collision : la classe des détecteurs parfaits et la classe des détecteurs ultimement parfaits. Les détecteurs de collisions parfaits permettent d'assurer une communication fiable et permettent de résoudre de façon efficace des problèmes comme la recherche du maximum. Dans la mesure où ils peuvent être réalisés pratiquement, ils peuvent servir de mécanisme de base pour définir une algorithmique distribuée sur des réseaux de capteurs.

Il ne semble pas que ce soit le cas avec les détecteurs éventuellement parfaits. Malgré le fait que l'on peut réaliser une diffusion fiable avec des messages sans contenu, une question ouverte reste de savoir s'il existe un moyen efficace de résoudre le problème du maximum avec de tels détecteurs.

[4] est à notre connaissance le premier article à avoir introduit la notion de détecteurs de collision. Les détecteurs de collision y sont définis dans un cadre plus général de communication où les capteurs peuvent émettre des messages avec contenu. Pour l'essentiel les définitions présentées ici sont des adaptations de ces définitions générales au cas de la communication avec des messages sans contenu.

Une extension de ce travail consiste à élargir à une communication de messages avec contenu les résultats présentés ici et d'étudier dans quelle mesure les détecteurs de collision définis ici peuvent servir à réaliser les détecteurs plus généraux de [4, 5].

## Références

- [1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks : a survey. *Computer Networks*, 38(4) :393–422, 2002.
- [2] T. D. Chandra and S. Toueg. Unreliable failure detectors for reliable distributed systems. *J. ACM*, 43(2) :225–267, Mar. 1996.
- [3] 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.
- [4] 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.
- [5] 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.
- [6] 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.
- [7] 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.
- [8] N. A. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1996.
- [9] K. Nakano and S. Olariu. A survey on leader election protocols for radio networks. In *ISPAN*, pages 71–, 2002.
- [10] G. R. A perspective on multiaccess channels. *IEEE Trans. Inform. Theory*, 31 :124–142, 1985.
- [11] D. E. Willard. Log-logarithmic selection resolution protocols in a multiple access channel. *SIAM J. Comput.*, 15(2) :468–477, 1986.