Les pannes Byzantines

Johanne Cohen[1]

[1]LORIA/CNRS, Nancy, France.

Plan

Problème du consensus

- Exemple de problème de consensus : un système de gestion répartie de transactions :
  - tous les processus ayant participé à une transaction doivent finalement décider de sa validation ou de son annulation.
  - Ils doivent tous prendre la même décision.
  - Un des problèmes majeur est d'être robuste vis à vis des pannes (de sites ou de communications).

Ennoncé du problème

Historique : introduit par Lamport, Shostak, Lease (1981)

L'armée byzantine assiège une ville : elle a n campements commandé par un seul général. Cette armée doit attaquer cette ville. Pour réussir il faut que tous les campements attaquent en même temps. Chaque jour, parmi les généraux, un seul donne l'ordre d'attaquer ou d'attendre : c'est le commendant noté ℓ_0. Les autres généraux sont appellés les lieutenants $\ell_1, \ldots, \ell_{n-1}$.

Restriction du modèle

est étudié dans le cadre suivant dans le mode synchrone et dans le monde se restreint au problème suivant :

1. **dans le mode de communication oral** : Les n généraux peuvent communiquer ensemble directement par l'intermédiaire de messagers (chaque destinaire d'un message connait l'expéditeur).
2. **dans une nombre limité d'ordres** : deux possibles $\langle$`ATTAQUER`$\rangle$ ou $\langle$`REPOSER`$\rangle$.
3. **dans le mode de synchrone** : le temps de transmission ne peut pas êtret superieure à δ. Les lieutenants se rendent compte quand il y a une perte de messages. Lors de non-réception de messages, la valeur reçue par défaut sera `DEF`.

Résultat de l'algorithme

Le résultat de l'algorithme doit satifaire les deux conditions suivantes :

IC1 : tous les lieutenants loyaux prennent la même décision processus normaux doivent connaitre/prendre la décision de P_0.

IC2 : Si le commandant ℓ_0 est byzantin, alors chaque lieutenant loyal obeissent à l'ordre du commandant.

Description du problème

- un général et des lieutenants de l'armée byzantine campent autour d'une cité ennemie.
- Ils doivent se mettre d'accord sur le plan de bataille sinon la défaite est assurée.
- Parmi eux, f traitres tentent de faire perdre la bataille à l'armée byzantine.
-
- Les communications se font par l'intermédiaire de messagers.

Modèle

- synchrone (les messages arrivent ou n'arrivent pas)
  Remarque : si le message n'arrive pas, le traitre est identifié
- 2 types de messages : "attaque" ou "repos"

Analogie avec le problème de consensus

Problème byzantin	problème consensus
les n camps	les n processus
les f traites (byzantins)	les f pannes
$v(i)$ info. envoyé par le général i.	$v(i)$ suggestion du processus i
Résultat : "attaque" ou "repos"	Décision prise en consensus

Résultat d'impossibilité

Situation : 3 camps dont un traite.
Question : Comment le lieutenant 1 peut-il différentier les deux situations.
ICI un dessin

Résultat d'impossibilité

Situation : 3 camps dont un traite.
Question : Comment le lieutenant 1 peut-il différentier les deux situations.
ICI un dessin
En fait, il ne le peut pas.

Résultat d'impossibilité

Théorème :
Le problème des généraux byzantins pour f byzantins avec au plus $3f$ camps ne peut pas être résolu.

Démonstration.
L'exemple précédent se généralise en considérant 1 camps comme f camps. □

Plan

Algorithme P(m) avec n généraux et m byzantins ($n > 3m$)

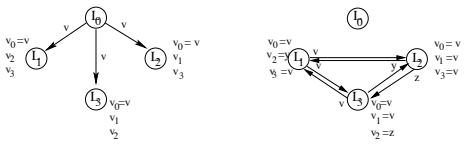
1. Cas P(0) (aucun participant est byzantin : $m = 0$) :
   - 1.1 Le commandant ℓ_0 envoie la valeur $\langle v \rangle$ à tous ses lieutenants.
   - 1.2 Si le lieutenant ℓ_j reçoit la valeur $\langle v \rangle$ alors $v_j = v$ sinon $v_j =$ DEF (détection d'une non réception d'un message).
2. Cas P(m) (m participants byzantins, $m > 0$, $n > 3m > 0$) :
   - 2.1 Le commandant ℓ_0 envoie la valeur $\langle v \rangle$ à tous ses lieutenants.
   - 2.2 Pour chaque lieutenant ℓ_j, si il reçoit la valeur $\langle v \rangle$ alors $v_j = v$ sinon $v_j =$ DEF
   - 2.3 Pour chaque lieutenant ℓ_i et tout $j \neq i$:
     - 2.3.1 Soit v_j = la valeur $\langle v \rangle$ que ℓ_i reçoit de ℓ_j lors de P(m-1) étape 2 ; $v_j =$ DEF si aucune valeur est reçue (avant δ).
   - 2.4 $val = majorite(v_0, \ldots, v_{n-1})$ définie par

$$majorite(v_0, \ldots, v_{n-1}) = \begin{cases} v & \text{si } v \text{ est majoritaire} \\ \text{DEF} & \text{s'il n'existe aucune valeur majoritai} \end{cases}$$

   - 2.5 val est la valeur décidé dans ce tour

Exemple où $m = 1$

4 processeurs $\ell_0, \ldots, \ell_3$ et ℓ_2 est le seul processeur malveillant.



étape 1 : ℓ_0 envoie $\langle v \rangle$ à ℓ_1, ℓ_2 et ℓ_3
étape 2 : ℓ_2, ℓ_3 et ℓ_1 recoivent $\langle v \rangle$.
étape 3 :
- ℓ_2, reçoit $\langle v \rangle$ de ℓ_3 et ℓ_1. il choisit $v = majorite(v, v, v)$
- ℓ_1, reçoit $\langle v \rangle$ de ℓ_3 et $\langle y \rangle$ de ℓ_2. il choisit $v = majorite(v, y, v)$
- ℓ_3, reçoit $\langle v \rangle$ de ℓ_1 et $\langle x \rangle$ de ℓ_2. il choisit $v = majorite(v, x, v)$

Lemme de terminaison

Lemme
L'algorithme se termine

Démonstration.
l'algorithme est récursif
- son argument m est décrémenté à chaque appel
- le cas $P(0)$ est traité.

□

Lemme de dépendance

Lemme
Pour tout m et k, si $n > 2k + m$ avec k byzantins, l'algorithme $P(m)$ satisfait la condition suivante

IC2 "si le commandant ℓ_0 n'est pas byzantin, alors les lieutenants loyaux aboutissent à la même décision que celle du commandant ℓ_0."

Preuve par récurrence sur m

Soit n le nombre de généraux
- $m = 0$: $P(0)$ fonctionne uniquement si
  - si le commandant est fiable
  - et si les messages sont acheminés correctement.
- $m > 0$:
  - Supposons par hypothèse de récurrence que P(m-1) satisfait la condition IC2.
  - Prouvons que P(m) satisfait la condition IC2.

Preuve (2/) : P(m) satisfait la condition IC2.

Regardons le comportement de l'algorithme :

1. A l'étape 1 : le commandant ℓ_0 envoie $\langle v \rangle$ à tous ses lieutenants.
2. A l'étape 2 : chaque lieutenant loyal ℓ_j exécute de nouveau P(m-1). De plus, on peut noter que $n-1 > 2k+(m-1)$ car $n > 2k+m$ et $m > 0$. Donc on peut appliquer l' hypothèse de récurrence.
   Chaque lieutenant loyal ℓ_i obtient $v_j = v$ de chaque lieutenant loyal ℓ_j.
   Sa liste $(v_0, \ldots, v_{n-1})$ contient $1 + n - 1 - k = (n-k)$ éléments ayant la valeur v.
3. Donc pour que le lieutenant loyal ℓ_i choisit v, il faut que la valeur v soit majoritaire dans $(v_0, \ldots, v_{n-1})$.
   il faut $n - k > \frac{n}{2}$
4. Comme $n > 2k+m$, on a $n-1 > 2k+(m-1)$ et $m > 0$ ensuite , $\frac{n-1}{2} > k$
5. Comme $\frac{n-1}{2} > k$, on a $n - k = n - 1 + 1 + k > \frac{n}{2}$
6. Donc v est majoritaire dans $(v_0, \ldots, v_{n-1})$ et chaque

La validation de l'algorithme

Théorème
Si il y a n généraux avec m byzantins ($n > 3m$), l'algorithme P(m) satisfait les deux conditions suivantes

- IC1 : tous les lieutenants loyaux prennent la même décision processus normaux doivent connaitre/prendre la décision de P_0.
- IC2 "si le commandant ℓ_0 n'est pas byzantin, alors les lieutenants loyaux aboutissent à la même décision que celle du commandant ℓ_0."

Preuve par récurrence sur m (le nombre de byzantins)

1. Cas $m = 0$: P(0) fonctionne uniquement si
   1.1 le commandant n'est pas byzantin
   1.2 et qu'il n'y a pas de messages non-délivrés.
   P(0) satisfait IC2 et IC1.
2. Cas $m > 0$: Supposons par hypothèse de récurrence que P(m-1) satisfait IC2 et IC1. Prouvons que P(m) satisfait IC2 et IC1
   Considérons le cas où le commandant n'est pas byzantin.
   D'après le lemme précédent , P(m) satisfait la condition IC2.
   Comme IC2 implique IC1, alors P(m) satisfait IC2 et IC1.
   Et le cas où le commandant est byzantin ?

Preuve de l'hypothèse de récurrence.)

Considérons le cas où le commandant est byzantin.

1. Il y a $m-1$ lieutenants byzantins.
2. Regardons le comportement de l'algorithme :
   2.1 le commandant ℓ_0 exécute l'étape 1 à sa facon (n'importe comment)
   2.2 les lieutenants vont rentrer dans l'étape 2 : chaque lieutenant exécute de nouveau P(m-1) en considérant qu'il est le général et qu'il a $n-2$ lieutenants. Remarquons, que $n > 3m$ implique $n - 2 > 3m - 2 > 3(m-1)$
   2.3 Donc par l' hypo. de récurrence, les exécutions de P(m-1) satisfont IC2 et IC1. Considérons la valeur de v_j après l'exécution de P(m-1)
   2.3.1 Si j est un byzantin, alors pour toutes les valeurs v_j des lieutenants loyaux sont identiques (par IC1).
   2.3.2 Si j n'est pas un byzantin, alors pour toutes les valeurs v_j des lieutenant loyaux sont identiques (par IC2).
   2.4 Comme ils appliquent la même fonction majorité, ils obtiennent le même résultat. Ce qui vérifie la condition IC2

Preuve de l'hypothèse de récurrence.)

Considérons le cas où le commandant est byzantin.

1. Il y a $m-1$ lieutenants byzantins.
2. Regardons le comportement de l'algorithme :
   2.1 le commandant ℓ_0 exécute l'étape 1 à sa facon (n'importe comment)
   2.2 les lieutenants vont rentrer dans l'étape 2 : chaque lieutenant exécute de nouveau P(m-1) en considérant qu'il est le général et qu'il a $n-2$ lieutenants. Remarquons, que $n-2 > 3(m-1)$
   2.3 Donc par l' hypo. de récurrence, les exécutions de P(m-1) satisfont IC2 et IC1. Considérons la valeur de v_j après l'exécution de P(m-1)
   2.3.1 Si j est un byzantin, alors pour toutes les valeurs v_j des lieutenants loyaux sont identiques (par IC1).
   2.3.2 Si j n'est pas un byzantin, alors pour toutes les valeurs v_j des lieutenant loyaux sont identiques (par IC2).
   2.4 Comme ils appliquent la même fonction majorité, ils obtiennent le même résultat. Ce qui vérifie la condition IC2

21/20 21

Preuve de l'hypothèse de récurrence.)

Considérons le cas où le commandant est byzantin.

1. Il y a $m-1$ lieutenants byzantins.
2. Regardons le comportement de l'algorithme :
   - 2.1 le commandant ℓ_0 exécute l'étape 1 à sa facon (n'importe comment)
   - 2.2 les lieutenants vont rentrer dans l'étape 2 : chaque lieutenant exécute de nouveau `P(m-1)` en considérant qu'il est le général et qu'il a $n-2$ lieutenants. Remarquons, que $n-2 > 3(m-1)$
   - 2.3 Donc par l' hypo. de récurrence, les exécutions de `P(m-1)` satisfont `IC2` et `IC1`. Considérons la valeur de v_j après l'exécution de `P(m-1)`
     - 2.3.1 Si j est un byzantin, alors pour toutes les valeurs v_j des lieutenants loyaux sont identiques (par `IC1`).
     - 2.3.2 Si j n'est pas un byzantin, alors pour toutes les valeurs v_j des lieutenant loyaux sont identiques (par `IC2`).

     Donc tous les lieutenants loyaux ont la même liste $(v_0, \ldots, v_{n-1})$
   - 2.4 Comme ils appliquent la même fonction majorité, ils obtiennent le même résultat. Ce qui vérifie la condition `IC2`

Preuve de l'hypothèse de récurrence.)

Considérons le cas où le commandant est byzantin.

1. Il y a $m-1$ lieutenants byzantins.
2. Regardons le comportement de l'algorithme :
   - 2.1 le commandant ℓ_0 exécute l'étape 1 à sa facon (n'importe comment)
   - 2.2 les lieutenants vont rentrer dans l'étape 2 : chaque lieutenant exécute de nouveau `P(m-1)` en considérant qu'il est le général et qu'il a $n-2$ lieutenants. Remarquons, que $n-2 > 3(m-1)$
   - 2.3 Donc par l' hypo. de récurrence, les exécutions de `P(m-1)` satisfont `IC2` et `IC1`. Considérons la valeur de v_j après l'exécution de `P(m-1)`
     - 2.3.1 Si j est un byzantin, alors pour toutes les valeurs v_j des lieutenants loyaux sont identiques (par `IC1`).
     - 2.3.2 Si j n'est pas un byzantin, alors pour toutes les valeurs v_j des lieutenant loyaux sont identiques (par `IC2`).

     Donc tous les lieutenants loyaux ont la même liste $(v_0, \ldots, v_{n-1})$
   - 2.4 Comme ils appliquent la même fonction majorité, ils obtiennent le même résultat. Ce qui vérifie la condition `IC2`

Nombre de messages échangés.

Soit n le nombre de généraux

algorithme `P(m)`	messages échangés
P(0)	$n-1$
P(1)	$(n-1)(n-2)$
P(2)	$(n-1)(n-2)(n-3)$
...	...
P(m)	$(n-1)(n-2)(n-3)\ldots(n-m-1)$
P(m)	$O(n^m)$