Comment extraire un discours cohérent de la confusion générale

Silvia Bonomi, Giovanni Farina, Sébastien Tixeuil

Comment extraire un discours cohérent de la confusion générale

Silvia Bonomi[1], Giovanni Farina [1†] et Sébastien Tixeuil[2,3]

[1]*Sapienza Università di Roma, Rome, Italie*
[2]*Sorbonne Université, CNRS, LIP6, F-75005 Paris, France*
[3]*Institut Universitaire de France, France*

Nous étudions le problème du Canal de Diffusion Fiable Byzantin (où une source doit envoyer le même ensemble de messages, ou discours, à tous les participants) dans les systèmes distribués affectés par des pannes Byzantines mobiles (c'est à dire où *tous* les processus sont susceptibles d'avoir un comportement malveillant au cours de l'exécution, contribuant à la confusion générale). Nous montrons que ce problème ne peut pas être résolu même dans le modèle le plus contraint défini jusqu'à présent pour les pannes byzantines mobiles. En dotant les processus d'un nouveau détecteur local de défaillances plus précis, nous apportons une solution au problème.

1 Introduction

Un Canal de Diffusion Fiable Byzantin (CDFB) est une primitive fondamentale de communication dans les systèmes distribués tolérants aux pannes, visant à garantir que tous les processus corrects délivrent ultimement le même ensemble de messages, que l'expéditeur des messages soit correct ou pas. Défini par Bracha [3] comme un élément constitutif d'un protocole de consensus tolérant aux Byzantins, le CDFB a été largement adopté et étudié depuis, en raison de sa capacité à empêcher les processus malveillants d'envoyer différents messages à différents processus corrects. Récemment, Guerraoui et al. [5] et Li et al. [6] ont étendu CDFB aux systèmes distribués avec participants dynamiques : dans une vue donnée (c'est-à-dire en considérant un ensemble de participants, ensemble décidé par les processus eux-mêmes), le sous-ensemble des processus Byzantins reste le même ; cependant, deux vues consécutives admettent des ensembles de processus Byzantins différents (en conservant la contrainte qu'un processus correct ne devient jamais Byzantin).

Un problème ouvert important consiste à considérer un système ou l'ensemble des processus Byzantins peut varier avec le temps, c'est à dire que dans une exécution donnée, *tous* les processus peuvent alterner entre un comportement malveillant, c'est à dire Byzantin (y compris l'expéditeur), et un comportement correct, contribuant à la confusion générale. Nous considérons dans ce travail le modèle des *Pannes Byzantines Mobiles (PBM)* [4, 2], où l'état de défaillance des processus est gouverné par un attaquant externe capable de compromettre et de contrôler un sous-ensemble des processus du système, cet ensemble évoluant dynamiquement. Contrairement aux travaux précédents [5, 6], nous considérons que l'ensemble des participants est fixe, mais que le modèle de défaillance est dynamique. C'est à dire qu'un participant peut changer de comportement (entre correct et Byzantin) plusieurs fois au cours de la même exécution.

Notre contribution. Nous considérons les conditions de solvabilité de CDFB dans le modèle PBM. Plus précisément, nous formalisons et étendons d'abord les spécifications de CDFB pour faire face aux PBM (et obtenons une nouvelle spécification CDFBM). Nous prouvons que CDFBM est impossible à résoudre dans le modèle PBM le plus contraint jusqu'à présent [4]. Enfin, nous définissons un détecteur de défaillances local supplémentaire qui permet de résoudre CDFBM.

[†]Ce travail a été financé par le projet *CBFTFDS - Concrete Byzantine Fault Tolerance and Forecasting in Distributed Systems* (Bando di Ateneo per la Ricerca 2022, Sapienza University of Rome).

2 Modèle

Nous supposons un ensemble fixe de n processus, chacun étant doté d'un identifiant unique. Nous supposons que tous les processus peuvent s'envoyer des messages, que les messages ne sont pas perdus ou modifiés lors des échanges, et que chaque processus ne peut pas mentir sur son identité lorsqu'il s'adresse à l'un de ses voisins (c'est-à-dire que nous supposons que les canaux de communication sont fiables et authentifiés). Nous mesurons le temps selon une horloge globale fictive $\mathbb{T}$ (non accessible aux processus) couvrant l'ensemble des entiers naturels $\mathbb{N}$. Nous notons t_0 le temps au démarrage du système, t_i le temps au i-ième instant depuis le début de l'exécution, et $T_{b,e}$ une période de temps entre t_b et t_e, soit $T_{b,e} := [t_b, t_e) : t_b, t_e \in \mathbb{T} ; t_b < t_e$. Chaque processus exécute un protocole distribué $\mathcal{P}$ stocké dans une mémoire en lecture seule inviolable. Nous supposons un système *synchrone*. De plus, nous supposons que le calcul évolue en *rondes séquentielles synchrones* $r_1, r_2, \ldots, r_j, \ldots$. Chaque roude r_j est divisé en trois phases : *(i) envoi* où les processus transmettent les messages à leurs destinataires, *(ii) réception* où les processus collectent les messages envoyés pendant la phase d'envoi de la ronde en cours, et *(iii) calcul* où les processus préparent les messages qui doivent être envoyés à la ronde suivante.

Les processus peuvent dévier de ce comportement et nous supposons qu'ils sont affectés par des *Pannes Byzantines Mobiles* (PBM) [4, 2]. Autrement dit, nous supposons l'existence d'un adversaire omniscient qui contrôle simultanément jusqu'à f agents Byzantins mobiles et qui peut «déplacer» ces agents d'un ensemble de processus à un autre. Nous supposons que les processus connaissent la valeur de n et de f. Lorsque l'adversaire place un agent Byzantin sur un processus p_i, l'agent prend le contrôle de p_i. Par exemple, p_i peut omettre d'envoyer/recevoir des messages, modifier le contenu des messages, modifier l'état de son processus quel que soit son algorithme local et exécuter du code arbitraire. Cependant, nous supposons que les agents mobiles Byzantins ne peuvent pas compromettre le code stocké dans la mémoire inviolable. Un processus p_i reprend l'exécution correcte du protocole $\mathcal{P}$ (bien qu'à partir d'un état possiblement corrompu) lorsque l'agent Byzantin quitte p_i suite à un ordre de l'adversaire. Nous supposons que l'adversaire peut déplacer chaque agent mobile indépendamment des autres. Néanmoins, les agents byzantins mobiles ne peuvent se déplacer qu'entre deux rondes consécutives, c'est-à-dire après la phase de calcul d'une ronde r_i, et avant la phase d'envoi de la ronde r_{i+1}. Nous mesurons le temps en termes de rondes. Notons que, dans le modèle PBM, aucun processus n'a la garantie de rester toujours correct, et nous pouvons avoir des processus alternant entre un comportement correct et Byzantin infiniment souvent. Un processus p_i est dit *défectueux* à la ronde r_k s'il est contrôlé par un agent Byzantin mobile à la ronde r_k et ainsi peut exécuter un protocole $\mathcal{P}' \neq \mathcal{P}$, et son état local peut être modifié arbitrairement. Un processus p_i est *correct* lorsqu'il n'est pas défectueux, c'est-à-dire que p_i est correct à la ronde r_k s'il n'est pas contrôlé par un agent Byzantin à la ronde r_k. Nous désignons par $B(T_{b,e})$ l'ensemble des processus défectueux pendant toute la période $T_{b,e}$ et par $C(T_{b,e})$ l'ensemble des processus corrects pendant la même période. On dit qu'un processus p_i est Δ_c-*infiniment souvent correct* si, pour chaque ronde r_j, il existe une période ultérieure $T_{b,e}$ durant laquelle p_i est correct pendant Δ_c. Formellement : $\forall t_j \in \mathbb{T},\ \exists t_b, t_e$ tels que $t_b > t_j,\ t_e - t_b \geq \Delta_c,\ p_i \in C(T_{b,e})$. Étant donné que les processus restent corrects ou défectueux pendant toute la durée d'une ronde, nous considérons des périodes de temps $T_{b,e}$ qui sont des multiples d'une ronde. Ceci implique que, par exemple, $C(r_i)$ représente l'ensemble des processus qui sont corrects pendant la ronde r_i.

Nous supposons que chaque processus p_i est *conscient* qu'un agent Byzantin mobile qui l'occupait a été déplacé par l'adversaire. Nous faisons abstraction de cette connaissance en introduisant deux détecteurs de défaillances locaux modélisés par des *oracles* révélant des informations à chaque processus p_i. Plus précisément, nous considérons les oracles *Conscience de Base des Défaillances* (O_{CBD}) [4] et *Conscience Complète des Défaillances* (O_{CCD}). Dans le cas O_{CBD}, un processus p_i sait quand (c'est-à-dire lors de quelle ronde) un agent Byzantin mobile l'a quitté ; dans le cas O_{CCD}, un processus p_i sait en outre quand l'agent est arrivé sur p_i (i.e., p_i connaît toute la dernière période $T_{b,e}$ pendant laquelle il était défectueux). Un processus rejette tous les messages mis en file d'attente pour être envoyés pendant la phase d'envoi si l'agent Byzantin mobile qui l'occupait l'a quitté à la ronde précédente. Toutes les hypothèses que nous avons considérées, à l'exception de l'oracle O_{CCD}, sont celles utilisées par tous les travaux liés au modèle PBM [4, 2].

3 Problème de Canal de Diffusion Fiable Byzantin Mobile

Un Canal de Diffusion Fiable Byzantin [3] est une primitive de communication qui permet à tous les processus corrects d'un système distribué de s'accorder sur l'ensemble des messages délivrés. La primitive fournit deux opérations, *diffuse* et *délivre*. La première vise à diffuser un message, la seconde permet de ne délivrer un message que si certaines propriétés sont satisfaites. La spécification originale de la primitive CDFB suppose que chaque processus est correct ou défectueux de manière permanente. Il faut donc redéfinir la spécification de la primitive afin de tolérer des pannes Byzantines mobiles.

Une primitive CDFBM(Δ_b,Δ_c) doit satisfaire les propriétés suivantes :

— (Δ_b, Δ_c)-*Validité* : s'il existe une période $T_{i,j}$ où un processus p_s est correct durant au moins Δ_b et exécute CDFBM.DIFFUSE(m), alors au moins un processus Δ_c-infiniment souvent correct p_d exécute ultimement CDFBM.DÉLIVRE(s,m) alors qu'il est correct.

— *Pas de duplication* : chaque processus p_d exécute CDFBM.DÉLIVRE(s,m) avec le message m au plus une fois lorsqu'il est correct, c'est-à-dire qu'il CDFBM.délivre un message m de p_s au plus une fois parmi tous les instants t_k telles que $p_d \in C(T_{k,k+1})$.

— Δ_b-*Intégrité* : si un processus p_d est correct à l'instant t_k et exécute CDFBM.DÉLIVRE(s,m), alors soit p_s était correct dans $T_{i,j} = [t_i, t_{i+\Delta_b})$, avec $t_i \leq t_k$, et a exécuté CDFBM.DIFFUSE(m) au temps t_i, ou p_s était défectueux dans $t_i \leq t_k$.

— Δ_c-*Accord* : si un processus est correct à l'instant t_k et exécute CDFBM.DÉLIVRE(s, m), alors chaque processus Δ_c-infiniment souvent correct exécute ultimement CDFBM.DÉLIVRE(s, m).

4 Impossibilité et protocole

Nous prouvons que le problème CDFBM est impossible à résoudre dans le modèle PBM usuel [4, 2]. Ensuite, nous montrons qu'avec l'oracle O_{CCD}, une solution est possible.

Théorème 1. *Si $\Delta_b \in \mathbb{N}^+$, $\Delta_b \geq 2$ rondes, et que les processus sont équipés de l'oracle O_{CBD}, alors il n'existe aucun protocole $\mathcal{P}$ implémentant un canal de diffusion fiable avec des Byzantins mobiles.*

Démonstration. Par contradiction, supposons qu'un tel protocole $\mathcal{P}$ existe. Supposons un processus toujours correct p_s (c'est-à-dire $\forall r_j,\ p_s \in C(r_j)$) qui exécute CDFBM.DIFFUSE(m) lors de la ronde r_1. Supposons un processus p_1 défectueux dans la période $[r_a, r_b)$. Dans la ronde r_b, le processus p_1 sait qu'il a été défectueux depuis une durée indéterminée grâce à l'oracle O_{CBD}. Un agent lors de la période $[r_a, r_b)$ peut avoir modifié arbitrairement la mémoire locale du processus p_1. En particulier, le processus p_1 ne peut pas savoir avec certitude si, lors d'une ronde $r_j < r_2$, il a été correct et a CDFBM-délivré le message m. Si le protocole $\mathcal{P}$ fait que p_1 délivre le message m de s lors d'une ronde $r_k \geq r_2$, mais que le message a déjà été délivré par le processus lors d'une ronde $r_j < r_2$ où il était correct, la propriété *Pas de duplication* est violée. Si le protocole $\mathcal{P}$ fait que p_1 ne délivre pas le message m de s lors d'une ronde $r_k \geq r_2$, et que le processus n'était pas correct auparavant et n'avait pas délivré le message lors d'une ronde $r_j < r_2$, alors l'une des propriétés de (Δ_b, Δ_c)-*Validité* ou Δ_c-*Accord* est violée. Cela conduit à une contradiction, quelles que soient les valeurs de Δ_b et Δ_c. □

Nous présentons maintenant, de manière informelle, le protocole $\mathcal{P}_{CDFBM-RB}$, une extension de la solution de Bracha [3] au problème du canal de diffusion fiable Byzantin mobile. Un contenu m est diffusé par $\mathcal{P}_{CDFBM-RB}$ en utilisant plusieurs types messages, respectivement SEND, ECHO, READY, et ABORT (ce dernier type n'étant pas présent dans la solution de Bracha). Chaque message de protocole (SEND, ECHO, READY, and ABORT) contient comme charge utile le contenu m, l'identifiant de la source et la ronde où le contenu a été généré. Chaque instance CDFBM.DIFFUSE(m) commence à la ronde r_i avec l'envoi du message SEND, qui est échangé au début de la ronde r_{i+1} ; le contenu est ensuite inclus dans les messages ECHO et READY par tous les autres processus respectivement corrects dans les rondes r_{i+1} et r_{i+2}, et échangés respectivement dans les rondes r_{i+2} et r_{i+3}. Si la source est correcte dans les rondes r_i et r_{i+1} et exécute CDFBM.DIFFUSE(m) dans r_i, alors le contenu sera CDFBM-délivré par tous les processus corrects dans la ronde r_{i+3}. Les processus Δ_c-*infiniment souvent corrects* et défectueux à la ronde r_{i+3} CDFBM-délivreront à la première ronde où ils redeviennent corrects. Le message SEND est généré

par la source à la ronde r_i, le message ECHO est généré à la ronde r_{i+1} par tous les processus qui ont reçu le message SEND dans la ronde courante ; le message READY est calculé à la rounde r_{i+2} par tous les processus ayant reçu le message ECHO dans la ronde courante de $(n + f)/2$ processus distincts ; un contenu est CDFBM-délivré par un processus à la ronde r_{i+3} si plus de $2f$ messages READY provenant de pairs distincts sont reçus dans la ronde en cours. Si plus de $2f$ messages READY sont reçus de pairs distincts dans la ronde en cours par un processus, alors il diffuse le message READY dans la ronde suivante (à savoir, le message READY est diffusé à chaque rounde après r_{i+3}). Le message ABORT garantit la propriété d'accord dans le cas d'une source Byzantine qui ne diffuse un contenu qu'à une partie des autres processus. Plus en détail, si à la fin de la ronde r_{i+2} le quorum de messages ECHO n'est pas atteint mais que plus de f messages ECHO sont échangés dans la ronde en cours, alors le message ABORT est diffusé à la ronde r_{i+3} ; si plus de f messages ABORT sont reçus par un processus correct dans une ronde, tous les messages READY avec le même contenu sont rejetés à la même ronde. $\mathcal{P}$ inclut également le compteur de rondes distribué défini par Bonnet et al. [2]. Un tel compteur de rondes garantit que tous les processus corrects partagent le même index de ronde pendant la phase de calcul de chaque ronde. L'oracle O_{CDD}, conjointement avec le compteur de ronde, permet aux processus de CDFBM-délivrer un contenu en au plus une ronde si ils sont corrects : soit à la ronde r_{i+3}, soit à la première ronde où il redeviennent correct.

Théorème 2. *Le problème du CDFBM peut être résolu avec O_{CDD} si et seulement si $n > 5f$.*

Démonstration. Condition nécessaire : Le résultat s'obtient en observant que l'attaquant utilisé par Backes et Cachin [1] et par Raynal [7] peut être simulé dans notre modèle.

Condition suffisante : $\underline{(\Delta_b = 2, \Delta_c = 1)\text{-}Validité}$: L'hypothèse $\Delta_b = 2$ implique que chaque processus correct à la ronde r_{i+1} recevra le message SEND et générera le message ECHO. A la ronde r_{i+2}, $n - 2f$ processus diffusent les messages ECHO, cela implique que tous les processus corrects atteignent le quorum pour générer le message READY. De même, à la ronde suivante, $n - 2f$ processus diffusent les messages READY, puis tous les processus corrects CDFBM-délivrent le contenu, et le message READY continue d'être diffusé dans toutes les rondes suivantes. $\underline{Pas\ de\ duplication}$: la propriété est garantie par le compteur de ronde et l'oracle O_{CCD} ; chaque contenu est CDFBM-delivré à la ronde r_{i+3} si le processus est correct ou à la première ronde où le processus redevient correct. $\underline{\Delta_b = 1\ ronde\ -\ Intégrité}$: les agents Byzantins mobiles ne peuvent pas CFMB-délivrer un contenu d'une source spécifique (en raison des quorums de protocole) sans la compromettre. Si la source n'a jamais été compromise, elle doit nécessairement effectuer l'opération CFMB-diffuse puis diffuser le message SEND. $\underline{\Delta_c\text{-}Accord}$: il reste à considérer le cas d'une source contrôlée par un agent Byzantin. Si une source défectueuse fait que plus de $2f$ processus corrects atteignent le quorum pour générer le message READY alors tous les processus corrects fourniront un seul contenu. Si une source défectueuse fait qu'au plus $2f$ processus corrects atteignent le quorum pour générer le message READY, aucun processus correct ne fournira le contenu en raison du mécanisme ABORT. □

Références

[1] Michael Backes and Christian Cachin. Reliable broadcast in a computational hybrid model with byzantine faults, crashes, and recoveries. In *2003 International Conference on Dependable Systems and Networks (DSN 2003)*, 2003.

[2] François Bonnet, Xavier Défago, Thanh Dang Nguyen, and Maria Potop-Butucaru. Tight bound on mobile byzantine agreement. *Theor. Comput. Sci.*, 609 :361–373, 2016.

[3] Gabriel Bracha. Asynchronous byzantine agreement protocols. *Inf. Comput.*, 75(2) :130–143, 1987.

[4] Juan A. Garay. Reaching (and maintaining) agreement in the presence of mobile faults (extended abstract). In *Distributed Algorithms, 8th International Workshop, WDAG '94*, 1994.

[5] Rachid Guerraoui, Jovan Komatovic, Petr Kuznetsov, Yvonne-Anne Pignolet, Dragos-Adrian Seredinschi, and Andrei Tonkikh. Dynamic byzantine reliable broadcast. In *24th International Conference on Principles of Distributed Systems, OPODIS 2020.*

[6] Jing Li, Tianming Yu, Ye Wang, and Roger Wattenhofer. Dynamic byzantine broadcast in asynchronous message-passing systems. *IEEE Access*, 10 :91372–91384, 2022.

[7] Michel Raynal. On the versatility of bracha's byzantine reliable broadcast algorithm. *Parallel Process. Lett.*, 31(3) :2150006 :1–2150006 :9, 2021.