Des consensus tout sauf mous

Anne-Marie Kermarrec [1]

Michel Raynal, Professeur à l'université de Rennes 1 et membre senior de l'Institut universitaire de France est un passionné. Il aime les voyages, adore la littérature, ne se lasse pas des côtes bretonnes (malgré ses origines du Sud-Ouest), encore moins du rugby (probablement en raison de ces mêmes origines). Michel aurait pu devenir un grand scientifique dans bien des domaines. C'est dans l'informatique qu'il est tombé, et plus particulièrement dans cette même région de l'Ouest où d'autres se sont retrouvés dans une marmite de potion magique. Michel Raynal est le lauréat du SIROCCO Innovation Award 2015 pour ses travaux en informatique distribuée. C'est l'occasion de découvrir ce domaine passionnant.

Un *système distribué* est un système composé de plusieurs entités (téléphone, capteur, ordinateur par exemple), connectées par un réseau de communication, c'est-à-dire qui peuvent communiquer (en filaire, par wifi, etc.) et qui, ensemble, s'attaquent à un problème comme de réaliser un calcul ou chercher de l'information. Nous vivons aujourd'hui dans un monde où la majorité des systèmes est distribuée. Par exemple, les données relatives aux comptes Facebook d'un ensemble d'amis sont dispersées aux quatre coins du monde ; pourtant lorsqu'un utilisateur met à jour son statut, tous ses amis doivent être notifiés, qu'ils soient connectés depuis le Maroc ou la Chine et que leurs données soient hébergées en Inde ou dans la Silicon Valley. Nos données, comme celles des entreprises, sont distribuées sur des serveurs ; on parle du *cloud*. La même donnée peut être répliquée sur plusieurs serveurs pour des raisons de fiabilité notamment.

1. Inria Rennes, Anne-Marie.Kermarrec@inria.fr.

Turing s'est un jour posé la question de ce que sa machine universelle était capable de calculer. Malheureusement, son formalisme ne s'applique pas aux systèmes distribués. Évidemment, les experts se sont posé la question de savoir ce qu'il est possible de calculer dans un système distribué, notamment le problème du consensus qui a été particulièrement étudié. Le problème est le suivant : chaque entité propose une valeur et, à la fin du calcul, toutes doivent s'être mises d'accord sur l'une de ces valeurs. C'est le type de fonctionnalité dont on a besoin, par exemple, pour assurer la cohérence de données répliquées dans le *cloud*. Pour illustrer le besoin de consensus, prenons l'exemple de données bancaires répliquées sur les machines A en France et B aux États-Unis. Ces données sont répliquées pour des raisons de tolérance aux pannes, de sorte que si une machine contenant ces données tombe en panne, l'autre continue de fonctionner. Ceci permet en outre aussi aux utilisateurs des États-Unis d'avoir un accès rapide depuis la machine B et aux utilisateurs européens d'avoir un accès rapide depuis la machine A. Considérons maintenant la situation suivante : Alice effectue un débit de 100 $ depuis la France (sur la machine A) de son compte, dont le solde est de 1000 $. L'ordre de débit est envoyé à la machine A mais aussi à la machine B pour assurer que les comptes soient cohérents. Le banquier lui, depuis la machine B, décide de verser des intérêts de 10% sur ce compte. Cet ordre de calcul d'intérêt est aussi envoyé aux deux machines. La communication prend du temps pour traverser l'Atlantique, si bien que la machine A reçoit d'abord l'ordre de débit puis celui des intérêts, appliquant donc 10% à un solde de 900 $, le solde résultant est alors de 990 $, alors que la machine B, applique les intérêts, puis le débit, le solde étant alors de 1000 $. Pour assurer un bon fonctionnement il est essentiel que les deux machines exécutent les instructions dans le même ordre et, pour ce faire, elles doivent atteindre un consensus.

Fisher, Lynch et Patterson [1] ont montré en 1985 que le consensus était impossible dans un système réparti asynchrone [2] dans lequel les machines peuvent tomber en panne. La preuve est longue et intriquée et il faudrait demander à Michel Raynal de nous l'expliquer. Pour faire court, dans un système asynchrone, si une entité ne répond pas, on ne sait pas différencier si c'est une question de lenteur, ou si c'est une défaillance de l'entité. Et on peut soit attendre éternellement le message alors que l'entité est défaillante ou alors prendre une décision et voir le message arriver trop tard et contredire notre décision. Michel Raynal a reçu le *Sirocco Innovation Award* en particulier pour ses travaux relatifs à ce problème de consensus, développant de nouveaux résultats à partir du théorème d'impossibilité de Fisher, Lynch et Patterson.

2. Dans un système synchrone, les opérations sont coordonnées sous un contrôle centralisé basé sur les signaux d'une horloge. Par opposition, un système asynchrone, en revanche, n'a pas d'horloge globale. Les différentes entités doivent utiliser des communications pour coordonner leurs tâches.

Que peut faire un chercheur en algorithmique distribuée quand un problème est impossible à résoudre ? Il étudie les hypothèses à relâcher pour que le problème devienne résoluble. Par exemple, pour le problème du consensus, on peut décider de prendre l'hypothèse que les messages entre les entités prennent un temps borné. Le consensus devient alors possible. Que peut-il faire quand on sait résoudre le problème ? Il peut chercher un algorithme plus efficace, par exemple plus rapide, demandant moins de messages, moins de calcul. Afin de contourner l'impossibilité du consensus en asynchrone, Michel Raynal, avec ses collègues Achour Mostéfaoui et Sergio Rajsbaum, a par exemple proposé une nouvelle méthode à base de conditions [2]. L'idée est de dénombrer les « conditions », c'est-à-dire les configurations pour lesquelles on sait comment obtenir un consensus. Michel a alors conçu des algorithmes qui permettaient de s'adapter à ces conditions, rendant ainsi l'obtention d'un consensus possible [3].

Le prix de Michel Raynal reconnaît ses contributions scientifiques. Pour conclure, on pourrait souligner que Michel assure la relève en formant des jeunes chercheurs, et en leur transmettant son enthousiasme et sa passion.

Références

[1] Michael J. Fisher, Nancy A. Lynch, and Michael S. Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the Association for Computing Machinery* 32(2):374–382 (1985).

[2] Achour Mostéfaoui, Sergio Rajsbaum and Michel Raynal. Conditions on input vectors for consensus solvability in asynchronous distributed systems. *Journal of the Association for Computing Machinery* 50(6):922–954 (2003).

[3] Achour Mostéfaoui, Sergio Rajsbaum and Michel Raynal. Efficient Condition-Based Consensus. SIROCCO 2001.

Cet article est également paru sur le blog BINAIRE [3].

3. http://binaire.blog.lemonde.fr/