Un regard sur les apports de Leslie Lamport à travers le prix Dijkstra

Michel Raynal [1]

Remarque préliminaire. *Quoique dans un style et avec un but différents, cet article peut être vu comme une suite de la présentation des idées et des travaux de Leslie Lamport parue dans le numéro 4 du bulletin 1024 (pp. 135-137), présentation due à Stephan Merz et Anca Muscholl et intitulée « Leslie Lamport : l'importance de la pensée mathématique pour l'informatique ».*

Le « Dijkstra Prize in Distributed Computing »

Ce prix, créé par la communauté internationale du calcul réparti en 2000, est attribué chaque année à un article d'au moins dix ans d'âge. Son but est de reconnaître une contribution qui apparaît comme fondamentale à la communauté de l'algorithmique répartie. Les deux conférences majeures de cette thématique sont l'ACM Symposium on Principles of Distributed Computing [2] (PODC), et le EATCS International Symposium on Distributed Computing [3] (DISC), et le prix Dijkstra est alternativement décerné lors de PODC les années paires et de DISC les années impaires. Jusqu'en 2003, le prix n'était décerné que par l'ACM et s'appelait alors « PODC

1. Institut Universitaire de France — Professeur à l'IRISA, Université de Rennes — Adjunct Professor, Hong Kong Polytechnic University.

2. https://en.wikipedia.org/wiki/Symposium_on_Principles_of_Distributed_Computing

3. https://en.wikipedia.org/wiki/International_Symposium_on_Distributed_Computing

Most Influential Paper Award ». Dijkstra l'a obtenu en 2002 et, après son décès, son nom a été associé à ce prix. La description détaillée de ce prix scientifique, des papiers lauréats, et de leurs auteurs peut être trouvée à l'adresse :

`https://en.wikipedia.org/wiki/Dijkstra_Prize#Winners`.

Prix et distinctions de Leslie Lamport

Avant le prix Turing 2013, qui lui a été décerné récemment, Leslie Lamport a obtenu de nombreux autres prix et distinctions scientifiques. Parmi ceux-ci figurent notamment : le « IEEE Piore Award » (2004), le « ACM SIGOPS Hall of Fame Award » (2007, 2012, 2013), la médaille John von Newmann (2008) et le « Jean-Claude Laprie Award in Dependable Computing » (2013, 2014), chacun pour une contribution particulière. Il a également été le récipiendaire du titre de « Docteur Honoris Causa » décerné par les universités européennes suivantes : Université de Rennes (2003), Christian Albrechts University, Kiel (2003), École Polytechnique Fédérale de Lausanne (2004), Università della Svizzera Italiana, Lugano (2006), et Université Henri Poincaré, Nancy (2007).

Leslie Lamport a également obtenu trois fois le prix Dijkstra : en 2000 (l'année de sa création), puis à nouveau en 2005 et 2014. Il est, à ce jour, la seule personne à l'avoir obtenu trois fois. La suite de cet article détaille les contributions scientifiques de ces trois articles, qui illustrent ainsi — au filtre du prix Dijkstra — de multiples facettes des travaux de Lamport.

Article 1

« Time, clocks, and the ordering of events in a distributed system »,
par Leslie Lamport, *Communications of the ACM*[4], 21(7) :558–565 (1978).

Comme indiqué précédemment, cet article est le premier a avoir été distingué par le prix Dijkstra (2000). Il contient trois idées fondamentales qui ont permis de mieux comprendre le calcul réparti et ont eu une influence sur l'enseignement et la recherche qui ne s'est jamais démentie. Il s'agit des trois concepts suivants.

— Une exécution répartie est un ordre partiel sur les événements produits par un ensemble de processus. Cet ordre est défini par une relation de causalité (appelée « happened before » dans l'article). Si cela peut paraître évident aujourd'hui, ça ne l'était pas en 1978 ! Il n'est pour s'en convaincre que de constater que cet article est l'un des articles les plus cités en informatique, tous domaines confondus.

4. À cette époque, les communications de l'ACM était un journal scientifique ; il est devenu depuis le magazine scientifique de l'ACM.

— Cet article introduit la notion d'horloge scalaire logique (souvent appelée « horloge de Lamport »). Ces horloges peuvent être vues comme une capture opérationnelle de la relation de causalité, à savoir associer une date à tout événement de façon à ce que la datation des événements soit toujours en accord avec la relation de causalité, et cela sans faire référence ni utiliser le temps physique.

— La notion de duplication d'une machine à états. Certes la notion de copie existait déjà pour les données, mais il s'agit là non pas de cas particuliers, mais de la notion beaucoup plus générale de service implémenté par duplication.

Grâce à cet article, l'année 1978 peut être vue comme la date de naissance du calcul réparti. Il y a eu un « avant » au cours duquel les intuitions étaient difficiles à formuler, les algorithmes difficiles à décrire de façon précise (sans parler de la démonstration de leur correction), et il y a surtout eu un « après » à partir duquel la compréhension du réparti a fait de grands pas. En une phrase, cet article marque la fin de la « préhistoire » du calcul réparti et le début d'une période féconde en résultats de la part de la communauté.

Article 2

« Reaching Agreement in the Presence of Faults »,
par Pease M., Shostask R., et Lamport L., *Journal de l'ACM*, 7(2) :228–234 (1980).

Cet article (co-écrit avec M. Pease et R. Shostack) a obtenu le prix Dijkstra en 2005. Il est l'un des articles de référence du calcul réparti tolérant les défaillances. Ceci tient à plusieurs raisons.

— Il y a tout d'abord l'introduction du concept de faute « byzantine » : un processus commet une telle faute lorsque son exécution ne répond plus à sa spécification (c'est-à-dire lorsqu'il se comporte de manière arbitraire). Il s'agit d'un modèle de faute très général.

— Le second point important de cet article (et d'un autre article par les mêmes auteurs paru dans ACM TOPLAS, 1982) est qu'il montre que le vote majoritaire (la technique utilisée jusqu'alors pour tenter de résoudre le problème) ne fonctionne pas. Plus précisément, pour que les n entités d'un calcul réparti (processus) puissent se mettre d'accord, il est nécessaire que moins d'un tiers d'entre elles aient un comportement byzantin (ces articles montrent de plus que cette condition est aussi une condition suffisante si le système sur lequel s'exécutent les processus est synchrone).

— Depuis ces articles (1980, 1982, c'est-à-dire depuis plus 30 ans !) les systèmes avec fautes byzantines ont été et sont toujours un sujet de recherche auquel de nombreux travaux ont été consacrés. Cette problématique (dont la compréhension profonde requiert une théorie sous-jacente solide) se retrouve

aujourd'hui dans de plus en plus d'applications (parfois sous le nom de fautes « malicieuses », intentionnées ou non). Un exemple industriel réside dans la mise en œuvre de serveurs fiables lorsque certains sites participants peuvent être byzantins.

Article 3

« Distributed snapshots : Determining global states of distributed systems », par K.M. Chandy et L. Lamport, *ACM Transactions on Computer Systems*, 3 :63–75 (1985).

Cet article (co-écrit avec K.M. Chandy, à Austin à cette époque) a obtenu le prix Dijkstra en 2014. Il s'agit là du premier article qui a défini précisément la notion d'état global d'une exécution répartie, et présenté un algorithme réparti extrêmement simple pour calculer au vol (c'est-à-dire lors de l'exécution elle-même) un tel état.

— L'obtention d'un état global cohérent d'une application répartie requiert de capturer un état local de chaque processus de façon telle qu'il n'existe pas de paires d'états locaux faisant état de messages reçus par le destinataire et non encore envoyés par l'émetteur. Pour certaines de ses utilisations, un état global cohérent peut de plus demander la collecte des messages envoyés et non encore reçus ; ces messages constituent alors l'état des canaux de communication relativement à l'état global calculé.

— L'algorithme de Chandy et Lamport constitue en fait une brique de base pour résoudre des problèmes fondamentaux du réparti tels que la détection des interblocages, la détection de la terminaison d'une application répartie (et, de façon plus générale, la détection de propriétés stables des exécutions réparties), ou la définition de points de reprise cohérents. Cet algorithme montre de plus que, dans un système asynchrone, on ne peut avoir qu'une vision « relativiste » des états globaux. Un état global rendu en résultat par l'algorithme est toujours cohérent mais il est impossible de savoir si le calcul est passé ou non par cet état, et c'est le mieux que l'on puisse faire !

Cet article et le premier article cité ci-dessus (1978) constituent les bases pour la compréhension de la nature des exécutions réparties.

Pour conclure

Cette courte présentation des idées développées dans trois travaux de Leslie Lamport, appréciée au « filtre des prix Dijkstra », montre que Leslie est le père du calcul réparti (au même sens où — grâce à l'apport des concepts de processus, sémaphore, commandes gardées, plus faibles pré-conditions, etc. — E.W. Dijkstra est le père de la synchronisation). Ses contributions sont des contributions « premières » aux divers sens du mot : premières dans le temps, premières par leur apport, et premières

au sens où elles ont ouvert de nombreux travaux de recherche et de sujets de thèse de par le monde. « Last but not least », il est important de ne pas oublier que les apports de Lamport ne se limitent pas à l'algorithmique répartie. Ses contributions, entres autres, à la logique temporelle et à la vérification des programmes concurrents méritent d'être également citées.

Les résultats précédemment mentionnés ne sont qu'une illustration du travail de Leslie Lamport qui a également obtenu des résultats remarquables (parfois peu connus) sur les nids de boucle (en calcul parallèle), les systèmes à clés publiques (en cryptographie), les critères de cohérence des données réparties, l'exclusion mutuelle (sans présupposer des opérations atomiques sous-jacentes), l'exclusion mutuelle rapide, etc. Le lecteur intéressé pourra se reporter à son document *My Writings* [5] dans lequel il présente chacun de ses articles de façon personnelle avec le style qu'on lui connaît. Il fait partie de ce club très fermé de chercheurs dont les résultats, non seulement font partie des fondements de l'informatique, mais sont enseignés dans tous les cursus d'informatique de par le monde.

5. `http://research.microsoft.com/en-us/um/people/lamport/pubs/pubs.html`