Demi-vagues utilisant des r-opérateurs : applications et terminaison

Bertrand Ducourthial

Laboratoire de Recherche en Informatique (bât. 490) et Institut d'Electronique Fondamentale (bât. 220), Université Paris-Sud, F-91405 Orsay France. *email* : ducourth@lri.fr

---

Résumé
Nous présentons de nouveaux opérateurs assurant la terminaison des demi-vagues, des algorithmes distribués constitués d'une cascade de calculs locaux lancés depuis certains sites dans le réseau. Nous étudions l'intérêt algorithmique des demi-vagues utilisant ces opérateurs, ainsi que leur terminaison.

Mots-clés : algorithme distribué, algorithme en vague, s-opérateur, r-opérateur

---

1. Introduction

1.1. Modèle de calcul

Une *entité calculante* est définie comme étant la plus petite parcelle de matériel ou de logiciel qui effectue un calcul. Elle dispose d'une valeur privée dans le registre b_0, d'autant de registres d'entrée b_i que d'arcs en provenance d'autres entités du réseau, et d'un registre de sortie b_{out}. On note $b_i^t(v)$ la valeur contenue dans le registre b_i de l'entité v à la date t. Ainsi, la valeur $b_0^0(v)$ est la donnée initiale de v.

Le *réseau*, modélisé par le graphe orienté $G(V, E)$, est constitué des liens unidirectionnels (arcs de E) reliant les entités (sommets de V) de la façon suivante : tout arc (u, v) de G part du registre $b_{\text{out}}(u)$ et arrive à un registre entrant $b_i(v)$ de v (communication un vers plusieurs). Les messages envoyés successivement sur un lien peuvent être dupliqués et arriver dans un ordre quelconque. Par contre ils ne peuvent pas être modifiés ni perdus (mais la duplication autorise des protocoles de transmission sûrs assez simples).

Nous appelons A l'ensemble des données utilisées dans le réseau. Si le nombre de liens entrant sur l'entité v est noté $\delta^-(v)$ et si le calcul local sur v nécessite δ_c unités de temps, il est modélisé de la façon suivante : $b_{\text{out}}^{t+\delta_c}(v) \leftarrow b_0^t(v) \diamond \ldots \diamond b_{\delta^-(v)}^t(v)$, où $\diamond$ est un opérateur binaire associatif à gauche ($(x \diamond y) \diamond z = x \diamond y \diamond z$). Nous supposons que chaque entité du réseau utilise le même opérateur $\diamond$ (calcul *uniforme*). Nous n'utilisons pas l'identité des entités (calcul *anonyme*). Lorsqu'une entité v reçoit un signal depuis l'extérieur du système, elle devient *instigatrice*. Elle effectue une initialisation de son registre $b_0(v)$ (dépendant de l'algorithme) et exécute ensuite un calcul local (car l'un de ses registres entrant a changé). Ses descendantes directes dans le réseau vont recevoir la valeur $b_{\text{out}}(v)$ et vont à leur tour exécuter un calcul local (après une initialisation effectuée à la réception du premier message). Cette cascade de calculs locaux définit une demi-vague. Si la demi-vague se réfléchit depuis les dernières entités atteintes jusque vers les instigatrices, on parle de vague.

1.2. Etat de l'art

Les vagues et demi-vagues ont été largement étudiées et utilisées en algorithmique distribuée, notamment en ce qui concerne la détection de leur terminaison, et ce, sous différentes appellations. Dans *[3]*, les auteurs proposent une technique générale de détection de terminaison pour ces *diffusing computations*. Dans *[2]*, l'idée d'un retour vers l'instigatrice (unique) est proposée pour s'assurer de la terminaison de la vague, et la technique est appelée *echo algorithms*. Dans *[6]*, l'auteur étudie les protocoles *PI (Propagation of Information)* (demi-vague) et *PIF (Propagation of Information with Feedback)* (vague), dans les réseaux non orientés, et l'étude est étendue aux réseaux orientés dans *[7]*. Diverses applications sont, en outre, développées. Dans *[5]*, les auteurs étudient les algorithmes en vague sur diverses topologies, et les utilisent pour calculer une fonction d'état global. Dans *[9]*, l'auteur unifie les travaux concernant les vagues (appelées *total algorithms* dans *[8]* et *wave algorithms* dans *[9]*) en les décrivant comme le calcul d'un *infimum*. Ainsi, il montre que si les opérateurs sont associatifs, commutatifs et idempotents, alors les vagues se terminent. Dans la suite, nous appelerons ces opérateurs des s-opérateurs.

Dans cet article, nous présentons une sur-classe des s-opérateurs : les r-opérateurs. Nous montrons que ces opérateurs peuvent être utilisés dans des demi-vagues. Puis nous étudions l'intérêt algorithmique et la terminaison de ces vagues.

2. s-opérateurs

Un *s-opérateur* $\star$ (s signifie *sort*, tri) induit un ordre partiel sur l'ensemble des données A par la relation $x \preceq_\star y \equiv x \star y = x$ (*infimum theorem*, voir *[8, 4]*). On définit l'élément neutre de $\star$ comme étant l'unique plus grand élément de A au sens de $\preceq_\star$ (un tel élément peut toujours être fabriqué artificiellement pour un problème donné qui n'en aurait pas). Le résultat suivant a été montré dans *[9]* et *[4]* de deux façons différentes[1].

Théorème 1 *Toute demi-vague basée sur un s-opérateur $\star$ se termine et fournit sur l'entité v la valeur* $\star\left\{b_0^0(u), u \in \widehat{\Gamma_G^{-1}}(v)\right\}$.

Ce théorème indique, par exemple, que les demi-vagues basées sur les opérateurs **min**, et **max** se terminent quel que soit le réseau et les valeurs avec lesquelles il a été initialisé. En outre, chaque entité du réseau obtient comme valeur finale la plus petite (dans le cas du minimum) ou la plus grande (dans le cas d'un maximum) de toutes les valeurs qu'elle a reçues au cours de la vague. Ainsi, toutes les entités d'une même composante fortement connexe du réseau finissent avec la même valeur.

3. r-opérateurs

Un *r-opérateur* (r signifie *réduction [4]*) $\triangleleft$ est défini à partir d'une application bijective $r : A \to A$, appelée *r-application*, comme étant un opérateur binaire vérifiant les propriétés suivantes : $\triangleleft$ doit être *r-associatif* ($(x \triangleleft y) \triangleleft r(z) = x \triangleleft (y \triangleleft z)$), *$r$-commutatif* ($r(x) \triangleleft y = r(y) \triangleleft x$) et *$r$-idempotent* ($r(x) \triangleleft x = r(x)$). En outre, $\triangleleft$ doit avoir un élément neutre à droite e ($x \triangleleft e = x$) et admettre la "simplification à gauche" (pour tout x, l'application $y \mapsto x \triangleleft y$ est injective).

Un r-opérateur est en fait un s-opérateur dissymétrisé, cette dissymétrie étant introduite par la r-application. Ainsi, un s-opérateur est un r-opérateur symétrique, c'est-à-dire pour lequel la r-application est l'identité $\mathbf{Id} : x \mapsto x$. Cette dissymétrie permet de réserver à la valeur privée des entités un traitement différent de celui appliqué aux valeurs reçues lors des calculs locaux.

Cependant, on montre qu'un r-opérateur est deux-commutatif : $\forall x, y, z \in A$, $x \triangleleft y \triangleleft z = x \triangleleft z \triangleleft y$. Notons que la deux-commutativité implique la k-commutativité, pour tout $k \geq 2$, c'est-à-dire la permutation des variables à partir du rang k. Cela signifie que le même résultat est obtenu sur une entité v du réseau si les données reçues par v arrivent par des arcs différents. En d'autres termes, il n'y a pas d'arc entrant distingué.

Le théorème suivant permet de faire le lien entre les r-opérateurs et les s-opérateurs.

Théorème 2 *Si $\triangleleft$ est un r-opérateur sur A défini avec la r-application r, alors l'opérateur $\star$ défini, pour tout x et y de A, par $x \star y = x \triangleleft r^{-1}(y)$ est un s-opérateur sur A.*

Ainsi, tout r-opérateur définit un s-opérateur, et donc un ordre partiel sur A. On montre que l'élément neutre du s-opérateur $\star$ associé au r-opérateur $\triangleleft$ est aussi élément neutre (à droite) de $\triangleleft$. De plus, la r-application est un isomorphisme de $(A, \triangleleft)$ et de $(A, \star)$. En fait, on peut montrer que tout s-opérateur $\star$ associé à un isomorphisme de $(A, \star)$ définit un r-opérateur.

Pour être utilisés dans une demi-vague, les opérateurs doivent être k-idempotents[2], $k \in \mathbb{N}$ *[4]*. On montre que la classe des r-opérateurs idempotents (i.e. 2-idempotents), qui est plus restreinte que celle des r-opérateurs, englobe encore largement la classe des s-opérateurs. En fait, pour que le s-opérateur $\star$ associé à l'isomorphisme r de $(A, \star)$ donne un r-opérateur idempotent, il suffit que r et $\star$ vérifient $x \preceq_\star r(x)$ pour tout x de A. Le théorème suivant autorise l'usage des r-opérateurs idempotents dans les demi-vagues[3].

[1] On note par $\widehat{\Gamma_G^{-1}}(v)$ les antécédants du nœud v dans le graphe orienté G, v lui-même inclus.

[2] Pour tout x de A, $\underbrace{x \diamond \ldots \diamond x}_{k} = x$.

[3] $\mathrm{d}_G(u, v)$ désigne la distance de u à v dans le réseau G.

Théorème 3 *Toute demi-vague basée sur un r-opérateur $\lhd$, construit avec le s-opérateur $\star$ et la r-application r, termine et fournit sur l'entité v la valeur* $\star\left\{r^{\mathrm{d}_G(u,v)}\left(b_0^0(u)\right), u \in \widehat{\Gamma_G^{-1}}(v)\right\}$.

Ce théorème indique que les demi-vagues basées sur des r-opérateurs incluent la distance dans le résultat obtenu par chaque entité. Notons que s'il s'agit d'un s-opérateur ($r = \mathbf{Id}$), on retrouve le résultat du théorème 1. Nous donnons ici le principe de la preuve et nous renvoyons le lecteur à *[4]* pour une preuve formelle. On exprime les résultats sur chaque entité par une expression régulière formée avec l'opérateur $\star$, la r-application r et les valeurs initiales $b_0^0(u)$ des entités u du réseau, que l'on étiquette pour s'affranchir des doubles (pour toutes entités u et v, $b_0^0(u) \neq b_0^0(v)$). On montre par récurrence qu'une entité v contiendra $r^k\left(b_0^0(u)\right)$ si u est un antécédant de v et si $\mathrm{d}_G(u,v) = k$, et qu'il est impossible qu'elle reçoive $r^{k'}\left(b_0^0(u)\right)$ avec $k' < \mathrm{d}_G(u,v)$. On en déduit le théorème lorsque le réseau est une arborescence. Supposons maintenant qu'il y ait deux chemins différents entre u et v (avec éventuellement un circuit). Dans ce cas, certaines entités vont recevoir plusieurs fois les mêmes valeurs initiales, sous forme de puissances de r. Mais puisque $x \preceq_\star r(x)$, elles ne retiendront que la plus petite de ces puissances, c'est-à-dire celle qui correspond à la distance.

4. Intérêt algorithmique

Les demi-vagues utilisant des s-opérateurs fournissent un seul résultat par composante fortement connexe du graphe. L'information codée par le réseau lui-même n'est donc que peu utilisée. Seule la notion de forte connexité influence le résultat.

Au contraire, les demi-vagues basées sur les r-opérateurs idempotents donnent des résultats différents sur les entités d'une même composante fortement connexe. Le résultat inclut non seulement la notion de forte connexité, mais aussi celle de distance. L'information additionnelle codée par le réseau est donc dans ce cas mieux exploitée pour construire le résultat.

Les r-opérateurs idempotents ont d'intéressantes propriétés algorithmiques. L'application la plus naturelle réside dans le calcul de distances depuis des sources dans le réseau. On utilise pour cela l'opérateur **minc** (pour minimum et incrément) défini par $\mathbf{minc}(x,y) = \mathbf{min}(x, y+1)$. On montre que l'opérateur **minc** est un r-opérateur sur $\overline{\mathbb{Z}} = \mathbb{Z} \cup \{-\infty, +\infty\}$ *[4]*. Pour calculer la distance à la source s la plus proche, on lance une demi-vague en donnant la valeur 0 aux sources, qui sont les instigatrices. Les autres entités sont initialisées à $+\infty$, l'élément neutre de **minc**. A la fin de la demi-vague, toutes les entités connaîtront leur distance depuis la source la plus proche. L'arbre en largeur peut alors être facilement obtenu en choisissant un seul registre entrant parmi ceux ayant la plus petite valeur.

Si maintenant les données de A sont des listes de sommets, et si l'on considère un r-opérateur opérant sur de telles listes, on peut alors obtenir un arbre en profondeur à l'aide d'une seule demi-vague.

Enfin, notons que ces opérateurs binaires peuvent se généraliser aux opérateurs n-aires. Pour ces derniers, c'est l'information relative aux chemins depuis les antécédants qui est intégrée dans le résultat de chaque entité (ce qui inclut l'information relative à la distance). Les algorithmes de recherche de chemins de poids minimaux *[1]* peuvent alors être décrit en une seule demi-vague utilisant un r-opérateur n-aire.

5. Terminaison

Une demi-vague basée sur un r-opérateur ne peut pas s'appliquer sur une topologie particulière telle qu'un anneau ou une arborescence couvrante quelconque du réseau. En effet, puisque le résultat utilise la distance et non simplement la forte connectivité, la topologie est primordiale. Sauf cas particulier, il n'est donc pas possible de détecter la terminaison d'une demi-vague basée sur un r-opérateur en utilisant les propriétés de la topologie utilisée par l'algorithme.

Les techniques classiques de détection de terminaison peuvent bien sûr s'appliquer à cet algorithme distribué particulier. Nous proposons cependant une transformation de la demi-vague en vague pour qu'elle inclut une détection de sa terminaison, dans le cas où il n'y a qu'une seule instigatrice et où l'opérateur est un r-opérateur idempotent vérifiant $x \neq r(x)$, i.e. $x \prec_\star r(x)$ (ce qui exclut les s-opérateurs).

On suppose que des messages de contrôle peuvent circuler en sens inverse sur les arcs de G, et qu'une entité peut lire les valeurs des registres de sortie de ses voisines (antécédantes et descendantes). Une entité v du réseau est dite *cohérente à la date t* si elle vérifie : pour toute antécédante directe u, $b^t_{\mathrm{out}}(v) \preceq_\star$

$r\,(b^t_{\text{out}}(u)) \neq e$ et pour toute descendante directe w, $b^t_{\text{out}}(w) \preceq_\star r\,(b^t_{\text{out}}(v)) \neq e$. On appelle D^t le graphe défini à la date t par les arcs (u, v) de G tels que $b^t_{\text{out}}(u) = r\,(b^t_{\text{out}}(v)) \neq e$. D^t est, pour toute date t, un graphe orienté acyclique, ou *dag*, qui évolue en fonction des mises à jour de la vague.

La détection de la terminaison est faite ainsi : toute entité du dag envoie le message de contrôle "je suis cohérente" à ses parents si et seulement si elle est cohérente et elle a reçu un tel message de toutes ses filles dans le dag.

Proposition 4 *La demi-vague est achevée si et seulement si l'instigatrice est cohérente.*

Nous donnons ici une idée de la preuve. Si la vague est terminée, alors toutes les entités du dag (qui couvre dans ce cas tout le réseau) sont cohérentes. Pour la réciproque, constatons d'abord qu'une feuille du dag ne peut être cohérente que lorsque le dag ne peut plus s'étendre dans le réseau à partir d'elle, et que si une feuille n'est pas cohérente, alors l'instigatrice (la racine du dag) ne peut pas l'être. Considérons donc un dag couvrant et supposons la demi-vague non terminée. Montrons que dans ce cas l'instigatrice n'est pas cohérente. Si la vague n'est pas terminée, il existe un arc (u, v) dans le réseau tel que v n'est pas cohérente, et la demi-vague doit mettre à jour v et toutes ses descendantes. Si v se met à jour, elle ne devient cohérente immédiatement que si elle n'a pas de descendante. Sinon, elle devra attendre que ses descendantes directes soient elles-mêmes cohérentes (elles ne l'étaient plus suite à la mise à jour de v), qui elles-mêmes attendront après leurs descendantes directes, et ainsi de suite. Autrement dit, v ne sera cohérente que lorsque toutes ses descendantes dans le dag le seront.

6. Conclusion

Nous avons défini dans cet article la classe des r-opérateurs. Ces opérateurs binaires incluent les s-opérateurs (associatifs, commutatifs et idempotents), qui étaient les seuls, à notre connaissance, à être utilisés dans des vagues. Nous avons montré que les demi-vagues basées sur des r-opérateurs idempotents terminaient en incluant dans leur résultat la distance, ce qui ouvre des possibilités algorithmiques intéressantes. Enfin, nous avons proposé une technique de détection de terminaison pour ce type de vague.

References

[1] K. M. Chandy and J. Misra. Distributed computation on graphs: shortest path algorihm. *Communications of the ACM*, 25(11):833–837, November 1982.

[2] E. J. H. Chang. Echo algorithms: Depth parallel operations on general graphs. *IEEE Transactions on Software Engineering*, SE 8(4):391–401, July 1982.

[3] E. W. Dijkstra and C. S. Scholten. Termination detection for diffusing computations. *Information Processing Letters*, 11(1):1–4, August 1980.

[4] B. Ducourthial. Properties of the r-operators to insure termination of distributed computations. Technical Report 1140, Laboratoire de Recherche en Informatique, Bât. 490. Université Paris-Sud, 91405 Orsay, France, December 1997. *(also in technical report IEF-97-01).*

[5] M. Raynal and J.-M. Hélary. *Synchronisation et contrôle des systèmes et des programmes répartis.* Eyrolles, Paris, France, 1988.

[6] A. Segall. Distributed network protocols. *IEEE Transactions on Information Theory*, IT 29(1):23–35, January 1983.

[7] G. Tel. Directed network protocols. In J. van Leeuwen, editor, *Distributed Algorithm. Second International Workshop*, volume 312 of *Lecture Notes in Computer Science*, pages 13–29, Amsterdam, The Netherlands, July 1987. Springer Verlag.

[8] G. Tel. *Topics in Distributed Algorithms*, volume 1 of *Cambridge International Series on Parallel Computation.* Cambridge University Press, Cambridge, UK, 1991.

[9] G. Tel. *Introduction to Distributed Algorithms*, volume 1. Cambridge University Press, Cambridge, UK, 1994.