Calculabilité distribuée

Guillaume Seguin

19 décembre 2008

Résumé

Ce document présente les bases de la calculabilité distribuée en s'appuyant sur le modèle établi dans [Angluin *et al.*, 2004]. Nous introduirons tout d'abord les notions fondamentales de la calculabilité distribuée, illustrées par plusieurs exemples pratiques, puis nous formaliserons cette première approche sous forme d'un modèle précis, celui des *Population Protocols*. Enfin, nous nous intéresserons à un résultat de calculabilité sur les prédicats : pour tout prédicat définissable dans l'arithmétique de Presburger il existe un protocole de population permettant de calculer stablement ce prédicat. Nous évoquerons également trois résultats de complexité sur des protocoles de population randomisés.

Table des matières

1 Introduction à la calculabilité distribuée

Nous donnerons ici une définition aussi concrète que possible de la calculabilité distribuée, puis nous illustrerons cette définition par deux exemples d'applications envisageables.

1.1 Concept de la calculabilité distribuée

La calculabilité distribuée s'intéresse aux propriétés de réseaux d'agents sans identité (sans identifiant unique) disposant d'une certaine quantité de mémoire (finie, ce qui limite le nombre d'états dans lesquels ils peuvent se trouver, et donc le nombre d'états de la population entière) et de puissance de calcul et qui peuvent entrer en communication deux à deux pour échanger des données afin de mettre à jour leur état. Les agents ne contrôlent pas leurs déplacements, ce qui permet de garantir que les interactions se produisent équitablement.

L'objectif d'un tel réseau est de réaliser un calcul donné, déterminé par l'algorithme que suivent les interactions entre agents, sur une entrée qui est inscrite à travers les états initiaux des agents, et dont la sortie est donnée à travers les états des agents. Il est important de noter qu'une telle méthode ne termine pas, on ne peut donc pas parler d'état final ou de sortie finale, mais uniquement de la sortie à chaque étape du calcul.

1.2 Une première application : les renards enragés

Ce cadre, posé de la sorte, peut sembler abrupt au premier abord, mais il est en fait très adapté à de nombreux problèmes pratiques. Il prend en effet en compte les caractéristiques actuelles de certains équipements électroniques (matériels embarqués disposant de peu de mémoire et de puissance de calcul mais productibles à grande échelle, surtout s'ils sont indifférenciés), et s'applique par exemple à la surveillance de populations animales.

Prenons par exemple une population de renards, initialement sains mais pouvant être infectés par la rage. Les autorités sanitaires n'intervenant qu'à partir du moment ou plus d'un certain nombre de renards sont malades (prenons par exemple 5), il faut pouvoir déterminer si cette condition est remplie. Pour cela, chaque renard est muni d'un capteur permettant de déterminer s'il est atteint par la rage, ainsi que d'un compteur, pouvant aller de 0 à 5, et d'un dispositif de communication de courte portée. Quand un renard tombe malade, son compteur est incrémenté de 1. Quand deux renards se rencontrent, leurs dispositifs interagissent, de sorte que l'un des deux compteurs contienne après l'interaction le compteur de

l'un contienne la somme des deux compteurs (tout problème d'overflow étant exclu), et celui de l'autre soit à 0, sauf dans le cas où la somme est supérieure ou égale à 5, auquel cas les deux compteurs sont positionnés à 5. Par propagation, toute la population aura après un certain laps de temps son compteur à 5, et l'observation en un point des agents de passage permettra de déclencher l'alerte.

Cette méthode permet ainsi de résoudre le problème initial, sans avoir à étudier toute la population périodiquement et à moindre coût, au prix d'un délai (mineur) d'obtention du résultat.

1.3 Une seconde application : statistiques sous-terraines

La calculabilité distribuée peut également être illustrée à travers les badges d'abonnement aux transports en commun. Imaginons que nos passes NaviGO puissent non seulement communiquer avec les bornes d'entrée mais également entre eux, et disposent d'une mémoire sur quatre bits. On pourrait alors déterminer avec une probabilité assez forte le sexe majoritaire parmi les utilisateurs de transports en commun (en oubliant les utilisateurs de tickets). On utilise pour cela un bit de mémoire qui stocke le sexe de l'utilisateur, un bit pour stocker un état de "meneur" ou "suiveur" (c'est un élément récurrent dans les problèmes de calculabilité distribuée), et deux bits pour stocker un état "résultat" parmi 3 : majorité de femmes, majorité d'hommes, égalité. À l'entrée du métro (au passage de la borne d'entrée), la configuration du passe est initialisée à "meneur" et le sexe majoritaire est celui du porteur de la carte. Le protocole utilisé est alors le suivant (basé sur [Delporte-Gallet *et al.*, 2006]) :

- Quand deux meneurs se rencontrent, l'un d'eux devient suiveur (celui en état d'égalité s'il n'y en a qu'un, un au hasard sinon).
- Quand deux individus de même état résultat se rencontrent, rien ne se passe d'autre.
- Quand deux individus d'état de majorités distinctes se rencontrent, les deux passent dans l'état d'égalité.

En indiçant les meneurs par l'indice "m" et en notant ♂ l'état de majorité d'hommes, ♀ l'état de majorité de femmes et ☼ l'état d'égalité, ces règles se traduisent visuellement par :

- $♂_m♂_m \to ♂_m♂$; $♀_m♀_m \to ♀_m♀$; $☼_m☼_m \to ☼_m☼$
- $♂♀ \to ☼☼$; $♂_m♀ \to ☼_m☼$; $♂♀_m \to ☼☼_m$; $♂_m♀_m \to ☼_m☼$
- $☼_m♂ \to ♂_m☼$; $☼_m♀ \to ♀_m☼$

Ce protocole fait que le meneur final (il sera unique si les gens ne ressortent du métro qu'après un certain temps, d'après la deuxième étape) portera le résultat cherché, qui pourra par exemple être récupéré à son passage au portique de sortie.

Ces méthodes permettent donc par exemple de réaliser des compteurs ou des tests d'égalité ou d'inégalités.

2 Modèle des *Population Protocols*

Il nous faut maintenant munir cette approche d'un modèle précis, d'une part pour formaliser le concept évoqué précédemment, d'autre part pour déterminer si on peut, par ce procédé, répondre à des questions plus complexes, telles que savoir si plus d'un certain pourcentage d'une population remplit une condition donnée. Le modèle présenté ici est celui des *protocoles de population* (*population protocols* en anglais), développé dans [Angluin *et al.*, 2004].

2.1 Populations, protocoles de populations

Définition 1. Un *protocole de population* $\mathscr{A}$ est un 6-uplet (X, Y, Q, I, O, δ), où X et Y sont les alphabets d'entrée et sortie, Q l'ensemble (fini) des états, I un mapping entre l'alphabet d'entrée et les états, O un mapping entre les états et l'alphabet de sortie, et δ une fonction de transition $\delta : Q \times Q \to Q \times Q$ qui détermine la nature des interactions entre les agents. On note que $\delta(x, y)$ n'est pas forcément égal à $\delta(y, x)$ en raison de la présence dans chaque interaction d'un agent *initiateur* et d'un agent *répondant*.

Définition 2. Une *population* $\mathscr{P}$ est un couple (A, E) où A est un ensemble de n agents et E une relation sur $A \times A$ définissant quels couples d'agents peuvent interagir. Pour un élément $(i, r) \in E$, l'agent i est qualifié d'*initiateur* et l'agent r de *répondant*. Cette relation E est irréflexive, c'est à dire qu'aucun agent ne peut interagir avec lui même.

La relation E peut être représentée sous forme d'un graphe orienté, qualifié de *graphe d'interaction*. En pratique, la plupart des approches de la calculabilité distribuée utilisent un *graphe d'interaction complet*, c'est à dire que E contient toutes les paires ordonnées d'agents distincts, i.e. :

$$E = \{(u, v) \in A \times A, u < v\}$$

2.2 Configurations, transitions, calculs

Il faut ensuite formaliser la représentation de l'état des agents à un instant donné :

Définition 3. Une *configuration de population* est un mapping $C : A \to Q$ qui à chaque agent de la population $\mathscr{P}$ associe un état du protocole de population $\mathscr{A}$.

On définit également les notions de *transition* et d'*accessibilité* pour formaliser l'évolution de l'état des agents, ainsi que les évolutions possibles :

Définition 4. S'il existe un élément $e = (u, v) \in E$ tel que $(C'(u), C'(v)) = \delta(C(u), C(v))$ et $C'(w) = C(w)\ \forall w \in A - (u, v)$, on définit la *transition* $C \to C'$, et on note dans ce cas $C \xrightarrow{e} C'$.

Définition 5. Une configuration C' est accessible depuis C s'il existe une suite de transitions $C \to C_1 \to C_2 \cdots \to C_n \to C'$. On note dans ce cas $C \xrightarrow{*} C'$.

Enfin, il faut définir la notion de *calcul* dans ce modèle, et pour cela introduire les notions d'*exécution* et d'*équité* :

Définition 6. Une *exécution* est une suite finie ou infinie de configurations $C_0, C_1, \cdots$ avec $\forall i, C_i \to C_{i+1}$ est une transition.

Définition 7. Un *calcul* est une exécution infinie *équitable*, c'est à dire que pour toute transition $C \to C'$, si C apparaît un nombre infini de fois au cours de l'exécution, C' aussi.

2.3 Entrées/sorties, stabilisation

Il faut, pour compléter ce modèle, définir la manière dont les agents correspondent aux entrées et aux sorties.

Définition 8. Une *affectation d'entrée* (respectivement *de sortie*) est une fonction $x : A \to X$ (resp. $y : A \to Y$), et on note $\mathscr{X} = X^A$ (resp. $\mathscr{Y} = Y^A$) l'ensemble des assignations d'entrée (resp. de sortie) possibles. Pour $x \in \mathscr{X}$ donné, l'exécution du protocole $\mathscr{A}$ commence dans la configuration C_x avec $\forall w \in A, C_x(w) = I(x(w))$, et pour une configuration donnée C, on a la sortie correspondante y_C avec $y_C(w) = O(C(w))$.

Enfin, on introduit des notions de convergence :

Définition 9. Un calcul *converge* s'il contient une configuration dite *à sortie stable* C, c'est à dire que pour toute configuration C' accessible depuis C (i.e. $C \xrightarrow{*} C'$), $y_C = y_{C'}$. On dit alors que ce calcul *se stabilise* sur la sortie y_C.

Définition 10. Étant donnés un protocole de population $\mathscr{A}$ et une population $\mathscr{P}$, une relation R est calculée stablement sur $\mathscr{X} \times \mathscr{Y}$ si $\mathscr{A}$ converge toujours (c'est à dire que tout calcul de $\mathscr{A}$ sur toute entrée $x \in \mathscr{X}$ converge) et si pour tout $(x, y) \in \mathscr{X} \times \mathscr{Y}$, $R(x, y)$ est vrai si tout calcul de $\mathscr{A}$ sur l'entrée x se stabilise sur la sortie y.

3 Résultats de calculabilité et complexité

Maintenant que nous avons muni le concept de calculabilité distribuée d'un modèle robuste, nous allons nous intéresser à plusieurs résultats de calculabilité et de complexité intéressants. Tous les protocoles de population considérés dans cette partie seront *toujours convergents*.

3.1 Prédicats calculables par les protocoles de population

Dans cette section, nous démontrons un théorème de calculabilité sur les *prédicats*[1] définissables dans *l'arithmétique de Presburger*[2]. Les protocoles de population étudiés ici sont tels que $Y = \{0, 1\}$, et prennent en entrée des k-uplets d'entiers naturels.

Définition 11. Étant donné un protocole de population $\mathscr{A}$, un k-uplet de $\mathbb{N}^k$ est accepté (respectivement refusé) si, pour tout calcul avec ce k-uplet en entrée, le calcul converge vers une sortie y telle que $\forall w \in A,\ y(w) = 1$ (resp. $\forall w \in A,\ y(w) = 0$).

Définition 12. Un prédicat $\phi : \mathbb{N}^n \to \{0, 1\}$ est calculable par un protocole de population $\mathscr{A}$ si et seulement si ϕ est accepté pour toute entrée $(x_1, x_2, \cdots, x_k)$ telle que $\phi(x_1, x_2, \cdots, x_k) = 1$ et refusé pour toute entrée $(x_1, x_2, \cdots, x_k)$ telle que $\phi(x_1, x_2, \cdots, x_k) = 0$.

Nous présentons ensuite plusieurs lemmes et théorèmes (que nous admettrons) nécessaires à la démonstration du théorème majeur de cette partie.

Définition 13. L'arithmétique de Presburger étendue est l'arithmétique de Presburger à laquelle on a rajouté l'opérateur d'équivalence modulo m, avec $m \geq 2$, noté $\equiv_m$.

Théorème 14 (Presburger)**.** *Tout prédicat définissable dans l'arithmétique de Presburger peut être défini par une formule sans quantificateurs dans l'arithmétique de Presburger étendue.*

Ce théorème, initialement démontré dans [Presburger, 1929] sous une autre forme, et reformulé ainsi dans [Ginsburg et Spanier, 1966].

[1] Un prédicat ϕ est une fonction à valeur dans $\{0, 1\}$

[2] L'arithmétique de Presburger est l'arithmétique définie à partir des axiomes de Péano moins les axiomes sur la multiplication.

Lemme 15. *Soient F et G des prédicats qui sont calculables stablement par des protocoles de populations sur un ensemble d'entrées X, et soit ξ une fonction booléenne à deux variables. Alors le prédicat $\xi(F, G)$ est calculable stablement par un protocole de population avec ensemble d'entrée X.*

Corollaire 16. *Toute formule booléenne sur les prédicats calculables stablement avec un ensemble d'entrées commun X est calculable stablement.*

Lemme 17. *Soient $X = \{\sigma_1, \cdots, \sigma_k\}$ un alphabet d'entrée arbitraire, a_i, c et m des constantes entières avec $m \geq 2$. Alors les prédicats sur les naturels $x_1, \cdots, x_k$ $\sum_i a_i x_i < c$ et $\sum_i a_i x_i \equiv_m c$ sont calculables stablement dans la famille des populations standard.*

On passe enfin au théorème majeur de cette section :

Théorème 18. *Tout prédicat définissable dans l'arithmétique de Presburger est calculable stablement par un protocole de population.*

Démonstration. Soit ϕ un prédicat définissable dans l'arithmétique de Presburger.

On commence par appliquer le théorème de Presburger à ϕ pour obtenir une formule ϕ' sans quantificateur, qui sera représentable par une relation $<$, $\equiv_m$ ou $=$ entre $\sum_i a_i x_i + k_1$ et $\sum_i b_i x_i + k_2$. On montre ensuite que ces prédicats sont calculables stablement :

Dans les cas $<$ et $\equiv_m$, on peut reformuler le prédicat en $\sum_i c_i x_i < k$ (avec $c_i = a_i - b_i$ et $k = k_1 - k_2$), ce qui est calculable stablement d'après le lemme 17. Dans le cas $=$, on définit les prédicats $\sum_i a_i x_i + k_1 < \sum_i b_i x_i + k_2 + 1$ et $\sum_i a_i x_i + k_1 > \sum_i b_i x_i + k_2 - 1$, qui sont calculables stablement d'après le lemme 17, et dont le ET est calculable stablement d'après le lemme 15.

Enfin, le corollaire 17 permet d'établir que, étant donné que ces prédicats sont calculables stablement, alors ϕ' l'est aussi, ce qui démontre le théorème. □

Il faut noter que ce théorème a été complété dans [Angluin *et al.*, 2006], sous la forme du théorème suivant (admis) :

Théorème 19. *L'ensemble des prédicats calculables par les protocoles de population est exactement l'ensemble des prédicats définissable dans l'arithmétique de Presburger.*

Cette version du théorème affine un précédent théorème de borne supérieure, présenté dans [Angluin *et al.*, 2004] :

Théorème 20. *L'ensemble des prédicats calculables par les protocoles de population, avec l'hypothèse supplémentaire que les interactions peuvent avoir lieu entre tout couple d'agents distincts, est dans* NL.

3.2 Résultats de complexité

Nous donnons ici trois théorèmes (admis), issus de [Angluin *et al.*, 2004], sur des questions de complexité dans le cas où les interactions peuvent avoir lieu entre toutes les paires d'agents distincts et où les interactions sont choisies de manière probabiliste.

Définition 21. Un *automate conjuguant* est un protocole de population dont le graphe d'interaction est complet et dont les interactions sont choisies indépendemment et uniformément dans l'ensemble des couples d'agents distincts.

Théorème 22. *Pour tout prédicat ϕ définissable dans l'arithmétique de Presburger, il existe un automate conjuguant qui calcule ϕ avec une probabilité 1, et où la population des agents converge vers la réponse en, en moyenne, $O(n^2 \log(n))$ interactions.*

Théorème 23. *Soit une fonction f calculable en temps polynomial (en $O(n^d)$) dans le pire cas par une machine de Turing en espace mémoire logarithmique. Alors, pour tout $c > 0$, il existe un automate conjuguant qui calcule $f(x)$ $\forall x \leq n$ avec une probabilité d'erreur en $O(n^{-c} \log(n))$ et en temps moyen polynomial (en $O(n^{d+2} \log(n) + n^{2d+c+1})$).*

Théorème 24. *Chaque prédicat booléen calculable par un automate conjuguant avec une probabilité $1/2 + \epsilon$, $\epsilon \in [1/2; 1]$ est dans P et il existe une machine de Turing randomisée qui le calcule en temps exponentiel et en espace mémoire logarithmique.*

Références

[Angluin *et al.*, 2004] Dana Angluin, James Aspnes, Zoë Diamadi, Michael J. Fischer et René Peralta (2004). Computation in networks of passively mobile finite-state sensors. *In PODC '04 : Proceedings of the Twenty-Third Annual ACM Symposium on Principles of Distributed Computing*, pages 290–299. ACM Press.

[Angluin *et al.*, 2006] Dana Angluin, James Aspnes et David Eisenstat (2006). Stably computable predicates are semilinear. *In PODC '06 : Proceedings of the Twenty-Fifth Annual ACM Symposium on Principles of Distributed Computing*, pages 292–299. ACM Press.

[Delporte-Gallet *et al.*, 2006] Carole Delporte-Gallet, Hugues Fauconnier, Rachid Guerraoui et Eric Ruppert (2006). When birds die : Making population protocols fault-tolerant. *In DCOSS '06 : The 18th Annual Conference on Distributed Computing.*

[Ginsburg et Spanier, 1966] Seymour Ginsburg et Edwin H. Spanier (1966). Semigroups, presburger. formulas and languages. *Pacific Journal of Mathematics*, 16:285–296.

[Presburger, 1929] Mojzesz Presburger (1929). Über die vollständigkeit eines gewissen systems der arithmetik ganzer zahlen, in welchem die addition als einzige operation hervortritt. *In Comptes Rendus du I congrès de Mathématiciens des Pays Slaves*, pages 92–101.