Obtention de témoins en arithmétique réelle non-linéaire

Sujet proposé par David Monniaux, chargé de recherche au CNRS, laboratoire VERIMAG, Grenoble.

Sujet

Des outils informatiques tels que Coq, Isabelle etc. permettent de prouver des théorèmes de mathématiques, ou la conformité d'un logiciel à une spécification. Plus exactement, ces *assistants de preuve* vérifient les étapes d'une preuve fournie par un humain : ce dernier fournit les grandes étapes, et l'assistant fait de lui-même certaines étapes bien circonscrites. Bien entendu, plus l'assistant est puissant, plus il peut effectuer d'étapes de preuve sans aide humaine, et moins la tâche de l'humain est difficile et fastidieuse.

L'outil Coq a ceci de particulier parmi les assistants de preuve qu'il construit explicitement une preuve dans un petit langage élémentaire, et qu'ensuite il vérifie cette preuve à l'aide d'un algorithme de typage relativement simple. Ainsi, il n'y a pas besoin de faire confiance aux procédures assez complexes qui permettent d'automatiser les étapes de preuves, car celles-ci fournissent non seulement un résultat « prouvé, OK », mais également un *témoin* de la correction de leur réponse.

Dans ce stage, nous nous intéresserons à l'arithmétique réelle non linéaire, autrement dit à des conditions de la forme $P(x_1, \ldots, x_n) \geq 0$ où $P \in \mathbb{Q}[X_1, \ldots, X_n]$, et on voudra produire des témoins pour des théorèmes tels que :

- Le polynôme P est partout positif ou nul. (On produira alors un témoin exprimant explicitement P sous la forme d'une somme de carrés de polynômes, ou plus généralement comme un quotient de deux sommes de carrés de polynômes ; l'existence d'un tel témoin est garantit par un résultat d'Artin donnant la réponse au dixième problème de Hilbert.)
- Un système d'équations de la forme $P_1(x_1, \ldots, x_n) \geq 0 \wedge \ldots \wedge P_m(x_1, \ldots, x_n) \geq 0$ n'a pas de solution. (On produira alors un témoin par exemple de la forme $\sum Q_i P_i = -1$ avec Q_i écrit comme sommes de carrés.)

L'obtiention de pareils témoins est assez malaisée. On a proposé de le faire par réduction à des problèmes d'optimisation convexe (plus précisément, de *programmation semi-définie*), mais cette approche fonctionne difficilement dans certains cas de géométrie dégénérée : on voudrait obtenir un résultat rationnel exact, mais on utilise des algorithmes numériques, et si l'ensemble des solutions est d'intérieur vide (infiniment fin), on ne peut pas trouver exactement un point dedans !

Nous avons proposé une méthode contournant cette difficulté, s'appuyant notamment sur l'usage de la réduction de réseau par l'algorithme LLL. L'objet du stage est de l'étudier, de travailler certaines preuves, et/ou de l'améliorer. Il faudra implémenter les algorithmes conçus.

Compétences requises

Par souci de commodité, nous implémentons en Sage, un logiciel de calcul formel basé sur le langage Python. Ce logiciel est d'abord facile et il n'est pas demandé de connaissances sophistiquées en Python ; le stagiaire pourra apprendre la programmation en Sage en quelques jours.