Manipulation de suites P-récursives avec SageMath

Mathis CARISNTAN & Aurélien LAMOUREUX
sous la responsabilité de Marc MEZZAROBBA

09/03/2017

Résumé

Ce rapport présente le travail que nous avons effectué au cours de ce projet. Nous présentons dans un premier temps ce que sont les suites P-récursives, ainsi que l'outil SageMath. Puis nous expliquons les motivations de ce projet. Enfin, nous détaillons les choix et détails de l'implémentation que nous avons réalisé, avant de discuter des limites de celle-ci et des possibles améliorations.

1 Introduction

...TODO...

1.1 Suites p-récursives & Algèbre d'Ore

Les suites sont beaucoup utilisées en mathématiques et dans différents domaines scientifiques, et on cherche, comme souvent en informatique, à en avoir une représentation exacte. De plus, il est généralement important que cette représentation soit également efficace pour la manipulation mathématique de ces suites.

On s'intéresse ici en particulier aux suites dites p-récursives. Une suite $(u_n)_{n\in\mathbb{N}}$ sur un corps $\mathbb{K}$ est dite p-récursive si elle est solution d'une équation de la forme :

$$\sum_{i=0}^{s} p_i(n) u_{n+i} = 0 \tag{1}$$

où, les p_i sont des polynômes en n. Des exemples communs de suites p-récursives sont par exemple la suite de Fibonacci , ou la fonction factorielle.

$$\text{Fibonacci : } u_{n+2} - u_{n+1} - u_n = 0$$
$$\text{Factorielle : } u_{n+1} - (n+1) * u_n = 0$$

Dans ce cadre, les suites p-récursives sont particulièrement intéressantes. En effet, (Motivations : Repr exactes de suites, utilisée dans différents domaines des maths/sciences) ...TODO...

1.2 Python & Sage

Sage est un outil de calcul formel libre. Il a été créé notamment pour proposer une alternative *opensource* aux logiciels existants comme Mathematica, Matlab, Maple . . . Contrairement à ces logiciels, Sage s'appuie sur des outils et librairies déjà existants comme NumPy, SciPy, matplotlib, FLINT et d'autres... L'utilisation de ces outils et unifiée et uniformisée au travers un langage basé sur Python. Ce langage présente une syntaxe qui diffère légérement de celle de Python. Ainsi, Sage est doté d'un "pré-analyseur", qui transforme les idiomes Sage en pur Python. Ainsi, il est possible d'écrire des librairies pour en Python pur ou en "langage sage". Bien qu'il existe également d'autres méthodes, on ne s'est intéressé qu'à celles-ci au cours du projet.

Comme évoqué plus haut, Sage est basé sur Python, et c'est donc naturellement que nous avons choisi ce langage pour le projet. En particulier, Python 2, puisque Sage n'est pas compatible avec Python 3 (bien que des efforts soient faits en ce sens).

Bien que Sage fournisse de nombreuses librairies mathématiques, il n'inclut pas encore officiellement de librairie pour l'algèbre d'Ore. Nous avons eu donc recours à une bibliothèque en cours de développement par la communauté qui implémente l'algèbre d'Ore.

2 Méthodologie de travail, et progression

La première tâche à laquelle nous nous sommes attelés, a été de chercher à comprendre notre sujet (les suites p-récursives) et nos outils (Python et Sage). Une fois cette étape effectuée, nous avons commencé à discuter de l'implémentation. Nous nous sommes rapidement mis d'accord avec notre encadrant, qu'il était plus pertinent d'un point de vue pédagogique de d'abord créer un module python, utilisant les fonctionnalités de Sage. Puis, une fois ce module eprouvé, le réécrire en utilisant la syntaxe de Sage. Cette manière de procéder nous a permis de nous concentrer initialement sur le fond, et non la forme, puisque nous étions plus familier avec Python.

2.1 Module Python

La base du module a été d'écrire une classe Python (init. n'étend aucun classes). Cette classe devait notamment permettre d'utiliser la représentation basée sur la relation de récurrence, et des conditions initiales. Immédiatement après, nous avons surchargé l'opérateur `__getitem__` pour accéder au n-ième terme de la suite. Initialement, nous calculions tous les termes de la suite, jusqu'à celui voulu, que nous renvoyions, mais cette méthode est très inefficace. Nous avons donc résolu d'utiliser la fonction `forward_matrix` du module ORE_ALGEBRA à la place (exemple et comparaison complex avec Fibo ?)
calculer ts les elts vs calculer que le bon élement)
Par la suite, nous avons également surcharger les opérateurs d'addition, soustraction et multiplication, en accord avec les lois de l'agèbre d'Ore.

Références

[1] A. Bohr and B.R. Mottelson, Nuclear Structure, vol. 2, Benjamin, New York, 1975.

[2] `http://ipnweb.in2p3.fr`

[3] Nick Park, *A Grand Day Out*, 1989, http://en.wikipedia.org/wiki/A_Grand_Day_Out