Jeu sur la structure de l'ensemble ouvert des sinogrammes

胡雨軒[*]

mercredi 23 janvier 2014

Résumé

This has been redacted in English, as all abstracts should be ∼ in an effort to keep the general meaning of that document intelligible.

Nous nous interrogeons sur les différentes décompositions possibles[1] des caractères. Nous en isolons deux et en distinguons les cas d'applications. Nous proposons une mise en application de ces idées en cherchant une structure de données et un algorithme convenables. Nous suggérons des applications pratiques pour l'apprentissage d'une langue chinoise. Enfin, nous terminons sur des suggestions d'améliorations et sur de possibles pistes de travail.

Table des matières

[*] <p2b.fac@gmail.com>

1. Aller voir sur wikipedia les solutions proposées avant.

1 Introduction

Les caractères chinois sont composés. L'ensemble de ces caractères est ouvert : on peut en créer de nouveaux ou changer le sens d'anciens. Il y a un problème informatique dans l'encodage des caractères car des différences d'œil existent. En ignorant ces différences d'œil on peut se restreindre dans cette étude aux caractères chinois classiques. Trouvons une base de données en ligne des caractères chinois.

2 Arbre d'un caractère [2]

L'idée source de cet article est un arbre de caractère. Il suffirait donc pour avoir une compréhension des caractères fidèle à celle voulue par les origines de bien placer chaque caractère dans cet arbre.

2.1 Définition de la forêt

Disons-le tout de suite, *arbre* [3] est un abus de langage. Non simple connected digraph. On peut plutôt dire graphe orienté avec des racines (les huit traits éternels) et des feuilles, les caractères. Les éléments de cet arbre qui ne peuvent pas être décomposés en éléments graphiques porteurs de sens plus petits ne sont liés en *in-degree* (par des arcs orientés vers eux) qu'aux huit traits éternels. Des arcs peuvent partir de plusieurs éléments de cet arbre pour aboutir à un caractères composés des éléments de départ. Il n'y a pas stricte bijection car plusieurs mêmes éléments de départ peuvent se combiner de différentes manières (autour, droite⋯). On définit la composition

2. Le titre est correct mais non pas l'utilisation de ce mot dans le corps du texte qui reste à clarifier.
3. Attention, ici le mot arbre est utilisé dans un sens différent que celui défini en théorie des graphes : il est retourné donc a un sommet et des racines et non pas une racine et des feuilles.

des caractères comme une classe de relations entre différents éléments de cet arbre. Les différentes relations de cette classe sont les différentes manières de combiner graphiquement les caractères.

Hmm, arrêtons ici le raisonnement : la non-bijectivité est-elle avérée ? La réponse est oui. De la même manière, la non-bijectivité entre les sons et les caractères est assurée, que la bijectivité que l'on tente d'établir soit faible [4] ou forte.

Donc un élément de cet arbre est le doublet composé d'une liste d'éléments et d'un mode de composition. Un élément est un caractère quand un son, donc un sens, lui est associé. Les modes de composition sont d'arité deux. Il y aurait d'ailleurs à dire à propos de cette simplicité. Les modes de composition sont inductivement définis par le choix de la base de caractères choisie.

2.2 Choix d'une base de caractère

Choisissons cette base de caractères : http://commons.wikimedia.org/wiki/Commons%3aChinese_characters_decomposition

Une composition incertaine est acceptée. L'intérêt de cette base est⋯je ne sais pas je l'ai choisie au pif car les données sont immédiatement accessibles.

Pourquoi les idées exposées ci-dessus ne sont pas communes : parce qu'elles ne respectent pas l'étymologie et l'histoire de la langue chinoise. Donc, si l'on respecte l'étymologie et l'histoire de la langue, on ne peut pas tout faire descendre aux racines des huit traits éternels. Dommage. Est-il souhaitable de créer une nouvelle base qui prendrait cela en compte ?

Justement, l'autre base de caractères parle de ça :

> Only pictorial configurations are used, not semantic ones. Where characters have typeface differences I've used the one provided by the Mainland Chinese contribution to Unicode. When there's more than one possible configuration, I've selected one only.

Donc visiblement les deux options ont été traitées. Voyons voir les deux voix.

2.3 Décomposition étymologique

Si l'on veut avoir la meilleur idée de la langue, il semble d'instinct que ce soit la meilleure car celle qui en respecte l'esprit. La décomposition graphique peut enduire en erreur avec des étymologies dites populaires donc trompeuses.

4. Faible, donc pas tout à fait correcte, peut-être améliorée : c'est une espèce d'approximation de bijectivité. L'ensemble faible est plus vaste que l'ensemble fort. Donc on ne met pas les tons. C'est censé être une explication mais si elle ne convient pas alors c'est une définition.

L'étudiant en chinois devra donc apprendre constater que les inophones depuis plus de quatre mille ans apprennent leur langue de la manière la plus naturelle possible d'abord en retenant un certain nombre de clefs par cœur, comme nous retenons nos lettres ou les pianistes les difficultés techniques (ou les classes de difficultés techniques au sens de Cortot) pour ensuite en inférer le sens, puis le son dans la cas d'un caractère phonosémantique (90 % des caractères actuels).

2.4 Décomposition graphique

Néanmoins, la décomposition graphique a l'avantage de l'œil et du pinceau, c'est-à-dire que l'on peut remonter aux traits primitifs. Toutefois, je suppose que les calligraphes les plus renommés n'ont jamais incapables de comprendre ce qu'ils écrivaient et il semble raisonnable que ces ancêtres aspiraient à être plus que des porte-plumes.

Il ne reste donc à cette décomposition que l'avantage du pinceau. Sans doute donc celle-ci pourra être utilisée en informatique où elle pourrait résoudre les problèmes de régionalismes dans l'ordre des traits et la manière d'agencer les caractères, à condition que l'ordre des traits, par exemple dans le tracé japonais, ne se joue pas des composants. Deux composants tracés simultanément ne sont pas acceptables, ni pour une méthode, ni pour l'autre.

3 Elaboration pratique

Concentrons-nous d'abord sur l'apprentissage de la langue qui est notre but principal.

3.1 Structure de stockage

Eternelle question : comment stocker de manière efficace un graphe orienté ? Liste d'adjacence (avec des pointeurs dedans). Il y a de la vraie merde à débrouiller là-dedans pour faire quelque chose de propre et d'optimisé. La linéarité algorithmique est-elle accessible ?

En implémentant ça avec des objets, on peut···hmm, on peut faire quelque chose de gigantesque pour y adosser une recherche prononciation si on stocke intelligemment et subtilement les sons (nativement en bpmf, adapté au pinyin). La même contruction finale pourrait permettre de rechercher tous les homophones de shi (pinyin malheuresement) puis d'apprendre les caractères.

Bref, en implémentant ça avec des objets, on peut dire que tout est élément et qu'un caractère est un élément dont le son n'est pas nul. Mais alors, comment rechercher rapidement les carctères dont les sons répondent à des critères ? Je comprends que l'objet n'est pas la panacée. Si l'on utilise des pointeurs (ou en détriment de la mémoire, stocker les caractères dans chacune des listes correspondante aux types d'éléments de sons (une pour un)), on peut utiliser parcourir ce lien dans les deux sens : du caractère on obtient le son (évidemment puisqu'on l'a écrit !) et d'un élément de son on obtient les caractères qui dont le son le contient. En recoupant les trois types d'éléments sonores, on arrive aux homonymes.

What a pity ! Je marque sur mon résumé que je peux programmer en C et il faut encore que j'aille lire sur wikipedia les articles sur les pointeurs pour les comprendre. Snif. Penser aussi à l'allocation dynamique, aux pointeurs intelligents, aux pointeurs de pointeurs et à tout les raffinements possibles [5].

3.2 Algorithme de construction

3.2.1 Naïf

Récursif. En fait, le graphe connecté orienté improprement appelé arbee est plutôt une forêt : lui-même n'est pas un arbre mais chaque caractère est le sommet d'un arbre. Pour chaque caractère, l'algorithme descend jusqu'à ses racines.

3.2.2 Améliorations

On va commencer par implémenter tout ça, faire une version fonctionnelle, voir la complexité temporelle de l'algo puis on en reparlera.

3.3 Implémentation

3.3.1 Technologies utilisées

3.3.2 Détail du code

5. La structure de mon penser ne s'accomode décidément pas aux méthodes d'enseignement de langues traditionnelles : je cherche pour comprendre la règle les exceptions à icelle ; je cherche pour comprendre une idée générale tous les cas particuliers. Est-ce une bonne chose ? Est-ce une pensée inductive ou déductive ? Inductive. Me sera-ce bénéfique ? Hmm···Je ne sais. Je suis donc un produit de la civilisation informatique. Comme c'est dommage, la pensée déductive, qui se concenctre sur l'idée générale pour en déduire les cas particuliers est si belle ! C'est la pensée des mathématiques.