Ce document présente les étapes nécessaires à la réalisation d'[IdRef_2].

Structure d'[IdRef_2]

A l'issue de l'exécution de l'algorithme HHC, on se retrouve avec un ensemble de clusters issus des ambiguous groups [AG]. **En théorie, un cluster correspond à un et un seul auteur** ; il faut ajouter à cela que certains AR ont été dupliqués (accentuation ou noms composés), ce qui est repéré par le champ « duplicId » des AR : deux clusters distincts ayant un duplicId en commun correspondent en théorie à un même auteur.

Une première étape, pour ne plus avoir à traiter le cas particulier des duplicID, est de fusionner les clusters ayant des duplicId en commun. Une étape **[Fusion]** est donc nécessaire, à l'issue de laquelle nous aurons réellement que selon l'algorithme, un cluster correspond à un et un seul auteur.

A partir de ces clusters, notre objectif est désormais de repérer d'éventuelles erreurs dans les bases de données. Nous identifions quatre types de **rectifications** :

- **[New]** Si le cluster ne contient aucun IdRef, il s'agit d'un **auteur non répertorié**. Si on souhaite lui créer un IdRef, le cluster contient toutes les publications à référencer. Notons que la structure de l'algorithme fait que ce cas arrivera très régulièrement, car l'étape [AR] créé un AR pour chaque co-auteur d'une publication, y compris ceux n'étant pas déjà référencés dans les bases de données françaises.
- **[Miss]** Si le cluster contient des AR n'ayant pas d'IdRef d'une part, et des AR ayant tous le même IdRef d'autre part, cela signifie que **l'auteur est référencé, mais certaines de ses publications ne le sont pas.** Dans ce cas, il faut ajouter un IdRef aux publications non référencées.
- **[Split]** Si un cluster contient plusieurs IdRef distincts, cela signifie qu'**un auteur est considéré à tort comme plusieurs personnes différentes** ; il s'est vu attribuer deux (au moins) IdRef distincts dans les bases de données.
- **[Merge]** Si plusieurs clusters contiennent le même IdRef, cela signifie que **plusieurs auteurs ont été considérés à tort comme une même personne** ; ils sont identifiés par un même identifiant dans les bases de données.

Le cas idéal est donc celui où un cluster ne contient que des AR ayant un seul IdRef commun à tous, et que cet IdRef n'apparaisse dans aucun autre cluster. Notons par ailleurs qu'il est possible que les rectifications [Miss] et [Merge] soient relevées pour un même cluster, et de même pour [Split] et [Merge], il faut alors faire attention à l'ordre dans lequel nous relevons ces rectifications, et aux fichiers que nous considérons pour le faire.

Nous effectuerons les étapes dans cet ordre : [Fusion], [New], [Miss], [Split], [Merge].

[Fusion] doit quoi qu'il arrive être exécutée en premier pour avoir d'une part cette correspondance « un auteur ⇔ un cluster », et d'autre part conserver au maximum la structure d'ambiguous group qui permettra de limiter le nombre de comparaisons à l'étape [Merge].

[New] est une rectification incompatible avec les toutes les autres, les clusters considérés pourront donc ne pas être pris en compte pour la suite de l'algorithme, ce qui permettra d'optimiser radicalement la vitesse d'exécution (on rappelle qu'il est possible qu'une majorité de clusters soient concernés par [New]).

[Fusion]

Pour l'exécution de [Fusion], nous commencerons par **relever dans les AR de chaque cluster les champs « duplicId »**. Ce champ est une paire de valeurs *(id, groupes)*, où *id* est un identifiant commun à un AR et ses duplications, et *groupes* liste les groupes ambigus où ces identifiants ont été envoyés.

Pour un cluster donné, on ira alors chercher les clusters ayant les mêmes duplicId dans les autres groupes : cela signifie que tous ces clusters font référence à un seul et même auteur. On construit alors un seul grand cluster, fusion de tous les autres, que l'on place dans chaque groupe ambigu concerné (en remplacement des clusters utilisés pour le former).

Une fois cette étape terminée, au sein d'un même groupe ambigu, on a bien :

- Deux clusters distincts correspondent à deux auteurs différents
- Un cluster contient bien toutes les publications de l'auteur qui lui correspond

On peut alors appliquer les étapes de détection des vérifications à chaque groupe ambigu, indépendamment des autres.

Les rectifications

Comme vu ci-dessus, ces étapes peuvent être appliquées successivement à chaque groupe, indépendamment des autres. On considère alors les données d'un seul groupe ambigu après l'étape [Fusion].

Il ne s'agit désormais plus que de comparer les IdRef des clusters. On commence donc par associer à chaque cluster **un objet D appelé la distribution d'IdRef** : D est un dictionnaire qui a pour clés les IdRef intervenants dans le cluster, et pour valeur le nombre d'AR concernés par chaque IdRef. Nous n'aurons désormais besoin que des distributions d'IdRef.

Pour un cluster donné, l'étude de D seule permet de déterminer si l'on se situe dans les cas de figure [New] (D ne contient que l'IdRef vide), [Miss] (D ne contient que deux IdRef, dont l'IdRef vide) ou [Split] (D contient au moins deux IdRef non vides) : **on peut donc relever ces rectifications en parcourant linéairement le groupe.**

Il reste alors à relever les rectifications de type [Merge]. Pour cela, on parcours les clusters (on peut se permettre d'ignorer ceux concernés par [New], ce qui peut enlever une grande partie des clusters) et

on compare les distributions deux à deux : si deux clusters ont des distributions qui se recoupent en un IdRef différent de l'IdRef vide, on est dans le cas de figure [Merge].

La complexité de ces étapes dépend de la structure du groupe. Notons |G| le nombre de clusters du groupe, et N le nombre de clusters du groupe concernés par [New] (qui ne sont donc pas considérés par l'étape [Merge]). La complexité temporelle est alors en **max(|G| , (|G|-N)^2)**. En effet, l'étape [New] a une complexité de |G|, et l'étape [Merge] une complexité en (|G|-N)^2, mais il est possible que l'on ait N<<|G| tout comme N≈|G|.