Le nouveau Raccord

version 1.0 du 14.06.2007

1 Raccord

Le logiciel de raccordement des sources pour les inclure dans SIMBAD, c'est-à-dire la cross-identification des sources d'un catalogue avec les sources déja existantes dans SIMBAD, fait un travail remarquable, mais il n'est plus assez flexible pour les besoins de SIMBAD 4. Comme les informations sur un objet dans SIMBAD 4 sont plus nombreuses que celles de SIMBAD 3, il est nécessaire d'adapter raccord à cette évolution.

Le raccord actuel a les options suivantes :

- *O.xxx yyy specifies a mapping between data types, e.g. -O.uvby uvby1*
- *-V min_score asks to verify the objects in Simbad, and to propose possible object merges when several Simbad candidates look compatible. The compatibility may be quantified by a min_score (default 1).*
- *-confl do not generate updating commands if there is any conflict.*
- *-coo asks to look by coordinates in SIMBAD, even when the search by identifiers was successful.*
- *-nocoo asks to NEVER look by coordinates in SIMBAD - this means that if no identifier could be found in Simbad, a new object is created.*
- *-g asks to generate the updating commands, as lines starting by the $ sign. The -g0 option minimizes the output - the input and SIMBAD data are not displayed if the matching is OK, or if a new star was added. The addition of the C option asks to edit the object type in addition to the score.*
- *-ign ignore_banner indicates the data which have to be completely ignored (see Input Structure below) ; the default banner is*
- *-help lists the various options and the default tolerance parameters.*
- *-id ignores the badly written identifiers - i.e. updating commands are generated even when some of the identifiers are not understood by Simbad.*
- *-maj assumes that the input contains everything - original data and SIMBAD results from a previous query. The -U option should therefore not be specified - no SIMBAD query will occur. Only the updating commands are generated. When followed by =filename, the updating commands are also copied (without the $ symbol) to the file specified.*
- *-ne do not accept correspondance (i.e. set a largely negative score) as soon as one element is bad, e.g. when the spectral types differ widely. for instance, if the distance between the input object and the By default, only the position may generate very negative scores. Using -ne=0 limits any score, including the angular distance, to the -1.0 lower bound.*
- *-no/ do not transform invalid identifiers (starting by*
- *-pmax max_printed_objects asks to limit the edition to pmax objects when several are found ; the default value of pmax is 50.*
- *-dif min_score_diff indicates the minimal difference between the rank1 and rank2 of the SIMBAD matching objects to generate updating commands.*

- *-min min_score indicates the minimal score of the matching object to generate updating commands.*
- *-add max_score indicates the maximal score of the best matching object for assuming it's a different object : when score ¡ max_score, a new object is created.*
- *-u astrotypes specifies the data to replace, in case of equal precision only. Use a comma-separated list of astrotypes, e.g. -u M.B,M.V*
- *-v is a verbose option. When set, the score computations are detailed.*
- *-xn cross_file asks to save the cross-identifications in a dedicated file. This file keeps the SIMBAD link to ensure that 2 input objects will not be linked to the same SIMBAD object - in which case an error message ++++This SIMBAD object already connected to xxx is issued. The number n specifies the size of the hash-table used for this, and has a default value of 3000 ; this value should be enlarged for very large catalogues. Note : it is possible with this option to disable this verification that 2 input objects will not be linked to the same SIMBAD object, by setting it to -x - (set the input file to -)*
- *.def value defines a global astrotype value ; for instance, the option .C Star (two words) has the similar effect as the line in the header of the input in the above example.*
- *+def value defines a global astrotype addition for SIMBAD : the parameter and value specified will be inserted into SIMBAD for each object of the input file. A typical example is the addition of a bibliographical reference with +B 1995A&AS.109.267G*
- *astrotypes=astrotypes tells which SIMBAD data should be read in SIMBAD and displayed in the output. The default consists of fundamental data and identifications, i.e. @,C,J,M.B,M.V,S,P,T. Alternatively this option may be present in the input header ; the option has however precedence over the input parameter..*
- *Equinox=equinox tells the equinox in which the input position is expressed ; the equinox may be preceded by the letter B indicating the FK4 system, or J to indicate the FK5 system ; G indicates galactic positions. Alternatively this option may be present in the input header ; the option has however precedence over the input parameter..*
- *Epoch=epoch tells the epoch which the input position refers to ; this value may individually be set by the Ep astrotype. Alternatively this option may be present in the input header ; the option has however precedence over the input parameter..*
- *tolerance_param=value specifies the confidence limits for the validity checkings (see Validity Checks below. Alternatively this option may be present in the input header ; the option has however precedence over the input parameter..*
- *score+B[(score)]=bibcode increases the score (see Score Computation below) by the value of score (1.0 by default) when one of the SIMBAD objects has the specified bibcode in its bibliography Alternatively this option may be present in the input header ; the option has however precedence over the input parameter..*
- *score-B[(score)]=bibcode decreases the score (see Score Computation below) by the value of score (1.0 by default) for the SIMBAD objects having the specified bibcode in its bibliography Alternatively this option may be present in the input header ; the option has however precedence over the input parameter..*

- *score+I[(score)]=identifier increases the score (see Score Computation below) by the value of score (1.0 by default) when one of the SIMBAD identifiers matches any identifier specified in the argument (several identifiers are separated by a comma). Alternatively this option may be present in the input header ; the option has however precedence over the input parameter..*
- *score-I[(score)]=identifier decreases the score (see Score Computation below) by the value of score (1.0 by default) when one of the SIMBAD identifiers matches any identifier specified in the argument (several identifiers are separated by a comma). Alternatively this option may be present in the input header ; the option has however precedence over the input parameter..*
- *score+C[(score)]=object_type increases the score (see Score Computation below) by the value of score (1.0 by default) when a possible SIMBAD counterpart shows one of the object types listed.*
- *score-C[(score)]=object_type descreases the score when a possible SIMBAD counterpart shows one of the object types listed.*
- *otypes=list_of_acceptable_otypes gives the list of the object types existing in Simbad that are compatible (with a score of 0.75 instead of 1) with the objects in the list. The different object types are separated by commas or blanks. Alternatively this option may be present in the input header ; the option has however precedence over the input parameter.. Note that this list is only used in the comparison of object types (an object type must therefore exist in the input file)*

Le nouveau raccord incluera ces options avec des options additionnelles et avec plus de flexibilité. Il est aussi prévu de rendre la sortie du nouveau raccord plus lisible.

Du point de vue de la programmation, il y a deux solutions pour le nouveau raccord :

1. utiliser raccord tel qu'il est et développer une couche d'entrée et une couche de sortie. La couche d'entrée traduit les nouvelles options en options raccord existantes. La couche de sortie traduit essentiellement le parfile et génère des scripts awk. Cette solution a l'avantage d'être rapide, mais ne change pas grand chose pour la flexibilité.
2. re-écrire raccord en Java. L'échelle de temps est plus longue (> 6 mois), mais le logiciel sera plus adapté aux besoins actuels. Il y a aussi un souci de rapidité.

2 La procédure pour l'inclusion d'un catalogue dans SIMBAD

La procédure des documentalistes pour inclure un nouveau catalogue dans SIMBAD est la suivante (à compléter) :

1. premier passage de raccord ;
2. étude d'identificateurs mal écrits ;
3. écriture de scripts (awk) pour corriger les identificateurs ;
4. passage de raccord ;

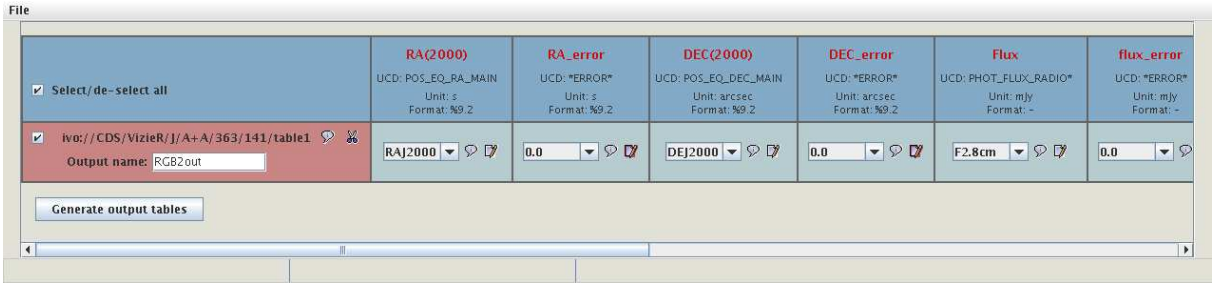

FIG. 1 – Assignation des colonnes de VizieR à des colonnes predéfinies dans le cadre des travaus sur les catalogues radio.

5. étude du parfile et statistique sur création en NAME, not correctly written, unknown, Invalid, Bad Identifier, not present, UPDATE from name, UPDATE from position, NEW, %I.0 à cause du mauvais score, %I.0 à cause des objets multiple, conflicting, already connected,
6. passage de raccord avec commandes de mise à jour en sortie ;
7. vérification et révision des commandes de mise à jour ;
8. inclusion des objets qui passent ;
9. étude des objets multiples et %I.0 ;

3 La structure du nouveau raccord

Les étapes d'utilisation du nouveau raccord seront les suivantes :

1. extraction du catalogue de VizieR ;
2. choix des préférences et des options ;
3. détections des identificateurs incorrects ;
4. création d'un script (si possible) ;
5. étude des résultats.

Dans la suite ces points seront détaillés.

3.1 Extraction du catalogue de VizieR

On suppose que le catalogue en question existe déjà dans VizieR. Le nouveau raccord demande de rentrer le numéro du catalogue et donne le choix de toutes les colonnes comme c'est déjà le cas dans VizieR. L'utilisateur choisit en cliquant les colonnes qui sont pertinentes pour SIMBAD. Une aide sera donnée par un bouton qui permet de visualiser le README de VizieR dans un web browser. Puis, le logiciel propose des paramètres de SIMBAD qui correspondent aux colonnes de VizieR. La Fig. 1 montre ce qu'on fait pour l'homogénéisation des catalogues radio , où nous avons des problèmes similaires. L'utilisateur peut ensuite choisir le paramètre de SIMBAD qu'il veut assigner à une colonne VizieR. On pourrait aussi imaginer d'afficher d'abord les paramètres de SIMBAD pour assigner les colonnes de VizieR

après. La fin de l'opération sera une liste de correspondance entre les colonnes VizieR et les paramètres de SIMBAD. Dans une deuxième ligne on pourra spécifier quelles données de VizieR vont dans les données fondamentales ou dans les mesures de SIMBAD.

3.2 Préférences et options

La différence entre les préférences et les options est que les préférences sont spécifiées une seule fois et sont valables pour la plupart des opérations, tandis que les options sont changées d'opération en opération. Cela implique qu'il faut un workspace pour se rappeler les préférences. Toutes les options auront un default.

Les préférences sont :

- format de sortie,
- un ou des types d'objet qu'on veut ignorer ou permettre,
- l'époque et l'équinoxe,
- prendre en compte la qualité des données ou non,
- conversion des unités.

Les options sont :

- cross-identification par coordonnées (spécification : min, max),
- cross-identification par nom,
- conditions sur la qualité des données,
- conditions pour l'augmentation ou la dimunition du score de raccord (bibcode, identificateur, type(s) d'objets,
- tolérances sur les différences entres les entrées SIMBAD et les données VizieR,
- conditions d'acceptation de cross-identification qui peut inclure plusieurs colonnes VizieR et plusieurs données SIMBAD,
- conditions pour ignorer des objets SIMBAD.

Des conditions logiques et arithmétiques seront nécessaires.

Il y a la question s'il faudrait traduire les préferences et options dans un script qu'on pourra ensuite modifier pour rendre la procédure plus flexible.

Il faudrait réflechir de nouveau sur la compatibilité des types d'objets pour le score. Demander aux gens qui s'occupe de l'ontologie.

3.3 Identificateurs incorrects

Après le premier lancement du nouveau raccord les identificateurs mal écrits sont sortis. La nomenclature et une recherche par position sont utilisés pour trouver la bonne écriture de l'identificateur. Une table de correspondances est établie. Correction des colonnes VizieR en retour.

3.4 Résultats

Une fois que tous les identificateurs sont reconnus, raccord est lancé de nouveau. Les résultats sont :

1. une statistique sur les differents problèmes : création en NAME, not correctly written, unknown, Invalid, Bad Identifier, not present, UPDATE from name,

UPDATE from position, NEW, %I.0 à cause du mauvais score, %I.0 à cause des objets multiples, conflicting, already connected,

2. sortie de 3 fichiers : (i) cross-identifications, (ii) objets multiples, (iii) %I.0 (mauvais score). Format : ASCII lisible pour tout le monde.

Pour les objets multiples l'utilisateur choisira en cliquant sur le bon objet avec la possibilité d'afficher les sources dans Aladin.

Possibilité de vérifier tous les objet avec Aladin. Assignation de la cross-identification directement par Aladin. Fusion facile de plusieurs objets dans SIMBAD. La scission des objets se fera dans le logiciel de mise a jour. Possibilité de recharger les fichiers de sortie de raccord pour les vérifications (avec Aladin). Géneration des commandes de mise à jour.