RAA 2016 Vol. **16** No. **11**, 178 (8pp) doi: 10.1088/1674–4527/16/11/178

http://www.raa-journal.org *http://iopscience.iop.org/raa*

***R**esearch in*
***A**stronomy and*
***A**strophysics*

Astronomical data fusion tool based on PostgreSQL

Bo Han[1], Yan-Xia Zhang[2], Shou-Bo Zhong[1,2] and Yong-Heng Zhao[2]

[1] International School of Software, Wuhan University, Wuhan 430072, China
[2] Key Laboratory of Optical Astronomy, National Astronomical Observatories, Chinese Academy of Sciences, 100012 Beijing, China; *zyx@bao.ac.cn*

Received 2015 December 23; accepted 2016 September 3

Abstract With the application of advanced astronomical technologies, equipments and methods all over the world, astronomical observations cover the range from radio, infrared, visible light, ultraviolet, X-ray and gamma-ray bands, and enter into the era of full wavelength astronomy. How to effectively integrate data from different ground- and space-based observation equipments, different observers, different bands and different observation times, requires data fusion technology. In this paper we introduce a cross-match tool that is developed in the Python language, is based on the PostgreSQL database and uses Q3C as the core index, facilitating the cross-match work of massive astronomical data. It provides four different cross-match functions, namely: (I) cross-match of the custom error range; (II) cross-match of catalog errors; (III) cross-match based on the elliptic error range; (IV) cross-match of the nearest neighbor algorithm. The resulting cross-matched set provides a good foundation for subsequent data mining and statistics based on multiwavelength data. The most advantageous aspect of this tool is a user-oriented tool applied locally by users. By means of this tool, users can easily create their own databases, manage their own data and cross-match databases according to their requirements. In addition, this tool is also able to transfer data from one database into another database. More importantly, it is easy to get started with the tool and it can be used by astronomers without writing any code.

Key words: astronomical databases: miscellaneous — catalogs — surveys

1 INTRODUCTION

In astronomy, data fusion technology is the foundation of multiwavelength astronomical research. Through catalog fusion, we can integrate multiple independent catalogs or images, databases or data sets by positions or object names and other information into a whole, so as to deepen the understanding of celestial bodies and promote the discovery of new objects or new physical phenomena. For example, Metchev et al. (2008) reported new L and T dwarfs in a cross-match of SDSS and 2MASS. Maselli et al. (2015) found new blazars by cross-matching recent multi-frequency catalogs. Multiwavelength data obtained are of great significance for further statistical analysis and data mining (e.g. Zhang & Zhao 2003, 2004; Gao et al. 2009; Zhang et al. 2013). However, for catalog fusion technology, the most important step is the calculation of cross-match, which refers to finding the corresponding entry in a catalog for each source in another catalog by using the source location as the center. In general, an observation has errors due to various factors, which cause difficulty in cross-matching. Any source in different catalogs has one or more counterparts within a certain error radius.

In recent years, there have been many studies focusing on the development of cross-match tools. Thus, cross-match tools are becoming more popular.

VizieR (*http://vizier.u-strasbg.fr/*), operated at CDS, Strasbourg, France, provides access to the most complete library of published astronomical catalogs and data tables available on line, and is organized in a self-documented database. Query tools allow the user to select relevant data tables and to extract and format records matching given criteria. However, cross-matching only supports a small number of records.

SIMBAD (*http://simbad.u-strasbg.fr/simbad/*) is an astronomical database also operated at CDS, Strasbourg, France, which provides basic data, cross-identifications, bibliography and measurements for astronomical objects outside the solar system (Wenger et al. 2000). SIMBAD has many kinds of query modes, such as object name, coordinates and various criteria. SIMBAD also provides links to some other on line services. Users may submit lists of objects and scripts to query. Similar to VizieR, the number of lists cannot be large.

The NASA Extragalactic Database (NED, *http://www.ned.ipac.caltech.edu/*), managed by NASA, contains names, positions and a variety of other data on extragalac-

tic objects, as well as bibliographic references to published papers, and notes from catalogs and other publications. NED may be searched for objects in many ways, including by name, positions, redshifts, types or by object classifications. NED also offers a number of other tools and services. If users want to query a large number of objects, they may submit a NED Batch Job, and retrieve the results at Pick Up Batch Job Results. NED provides another batch query, i.e. one right ascension (RA) and declination (Dec) position or object name per line, with a maximum of 500 positions and/or object names per request.

The Tool for OPerations on Catalogues And Tables (TOPCAT, *http://www.star.bris.ac.uk/∼mbt/topcat/*) is an interactive graphical viewer and editor for tabular data (Taylor 2005). It offers a variety of ways to view and analyze tables, including a browser for the cell data themselves, viewers for information about table and column metadata, and facilities for sophisticated interactive 1-, 2-, 3- and higher-dimensional visualization, calculating statistics and joining tables using flexible matching algorithms. It is developed in the Java language and is limited by computer memory when running. When cross-matching very large tables or tables with lots of columns, the computer is inclined to be out of memory. TOPCAT's sister package is the Starlink Tables Infrastructure Library Tool Set (STILTS), which is based on STIL, the Starlink Tables Infrastructure Library. STILTS offers many of the same functions as TOPCAT and forms the command-line counterpart of the Graphical User Interface (GUI) table analysis tool TOPCAT. STILTS is robust, fully documented and designed for efficiency, especially with very large datasets.

The CDS cross-match service (*http://cdsxmatch.u-strasbg.fr/xmatch/*) is a new data fusion and data management tool, which is used to efficiently cross-identify sources between very large catalogs (all VizieR tables, SIMBAD) or between a user-uploaded list of positions and a large catalog (Boch et al. 2012). About the xMatch algorithm, please refer to Pineau et al. (2011). Users interact with the CDS xMatch service through a Web application. Due to narrow network bandwidth, it has some limitations, for example, long jobs are aborted if computation exceeds 100 min while short jobs are aborted if computation exceeds 15 min; the search radius is maximized to $120''$ for a simple cross-match; the cone radius is no more than 15 degrees for a cone search; results are saved for no more than 7 days following their submission. Moreover the total size of uploaded tables is limited to 100 MB for anonymous users and 500 MB for registered users.

The 2MASS catalog server kit, developed by Yamauchi (2011), acts as a high-performance database server for the 2MASS Point Source Catalog and several other all-sky catalogs. This kit uses the open-source PostgreSQL, adopts an orthogonal xyz coordinate system as the database index and applies other techniques (table partitioning, composite expression index and optimization in stored functions) to enable high-speed search and cross-match of huge catalogs.

Pei (2011) and Pei et al. (2011) developed a highly-efficient large-scale catalog oriented fusion toolset based on MySQL database and HTM index. Zhang et al. (2012) developed a toolkit for automated database creation and cross-matching tasks, with which users may create their own databases and easily cross-match catalogs. Although the cross-match speed is quick, it costs a long time to retrieve the cross-matched result. In other words, the second operation of the matched result is necessary before application.

TAPVizieR is a new way to access the VizieR database using the ADQL language and the TAP protocol (Landais et al. 2013). The database is based on PostgreSQL and the sky indexation depends on HEALPix. TAPVizieR provides query and cross-match functions. The resulting access is only limited to an owner as recognized by the associated IP address which is saved for no more than 5 days. The execution time of an ADQL query is limited to 5 hours.

The Large Survey Database (LSD, *http://research.majuric.org/trac/wiki/LargeSurveyDatabase*) is a framework for storing, cross-matching, querying and rapidly iterating through large survey datasets (catalogs of $> 10^9$ rows, > 1 TB) on multi-core machines. It is implemented in Python, and written and maintained by Mario Juric. LSD applies nested butterfly HEALPix pixelization and the catalogs are partitioned into cells in space and time. LSD employs LSD/MapReduce as the high-performance programming model.

To some extent, all cross identification tools strongly promote the study of multiwavelength astronomy and provide a strong cross-match function. Nevertheless these tools are generally only used by managers of large data sets; some are for astronomers; some are difficult to learn; some are accessed by network and limited by network bandwidth, so users cannot upload too large catalogs. Considering all factors, we develop a user-oriented cross-match tool based on the PostgreSQL database with the sky-indexing Quad Tree Cube (Q3C), and continuously improve the system efficiency of identification, making the amount of cross-match records increase to tens of millions of lines and even higher. At the same time, we also develop an auxiliary tool for data exchange between different databases, helping astronomers better manage data.

2 METHODS AND IMPLEMENTATION

We develop a cross-match tool based on the PostgreSQL database and Q3C index. This tool has a user friendly interface for convenient use. Moreover we design an auxiliary tool applied for data exchange between different databases (MySQL, PostgreSQL). The detailed development scheme of the cross-match tool is described in the following.

2.1 Database Architecture

The PostgreSQL (*http://www.postgresql.org/*) database has a rich variety of functions and is a very stable version, and its default security configuration has received quite a lot

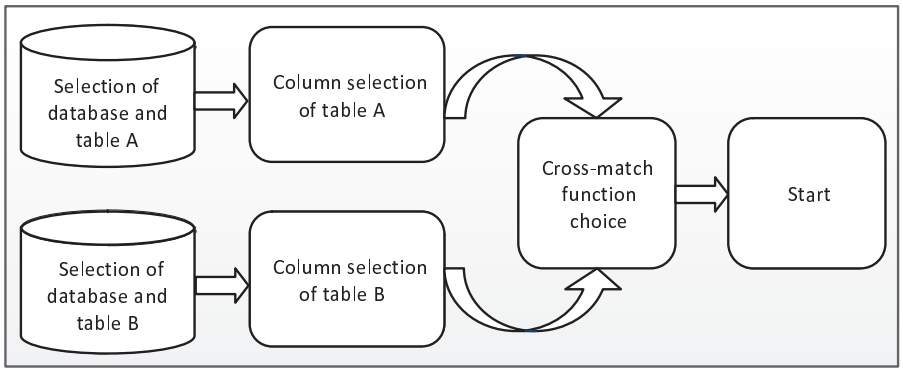

Fig. 1 A schematic diagram illustrating the cross-match tool.

of praise from security specialists. The consistency of the SQL specification and data integrity only allow interacting with the database in a rigorous way, which guarantees high quality of the data. PostgreSQL is an "object/relation" database management system, and it has the characteristics of being open-source, offering multi-platform support, free use, etc. From an academic perspective, it has very advanced and innovative characteristics, and is very rich in third party libraries. It is focused on academic study. It also has very good support in astronomy, for example, the third party libraries PGSphere and Q3C developed for astronomy play a very important role in the study of astronomy. We choose the Q3C index in the PostgreSQL database, and then use the psycopg2 interface program based on the Python language to interact with the PostgreSQL database and operate on the data.

2.2 Q3C

Q3C (*http://code.google.com/p/q3c/*) is a plugin for the PostgreSQL database and is developed for dealing with large astronomical catalogs or any catalog of objects distributed on a sphere. Q3C provides some functions, for example, fast circular, elliptical or polygonal searches on the sphere, fast positional cross-match and nearest neighbor queries (Koposov & Bartunov 2006). The basis of this scheme is a cube inscribed in the sphere, in which each face of the cube is a quad tree structure. The quad tree structure establishes the mapping of 2D coordinates in the square to the bitmask. There are six faces, so the 3 bits pointing to the face number are generated. Therefore, the mapping of the cube to the integer numbers is created. Each point on the sphere has a corresponding integer value, which is called the IPIX value. There are many adjacent IPIX values around the center, which can enable quick searches on the surface of the sphere by creating indexes with the IPIX values. In order to effectively make use of each index, each spatial query will be split into a pixel. Each pixel represents a continuous range of IPIX values. It should be easy and have rapid retrieval from the database for part of the data on the sphere.

2.3 Support of System Environment

These tools are coded in the Python language. It is recommended to use the Linux operating system. Users need to install Python (the general distribution of Linux will include the Python language by itself), and need the following database system (remote or local) and third party libraries as support. (1) PostgreSQL (9.0+): support for the latest SQL grammar and higher functional integrity. (2) MySQL (5.1+): the performance is very efficient. (3) Q3C: the plug-in for the PostgreSQL database. A good virtual terminal is also embedded in these tools, so the user can correct a run error in the program, according to feedback from the virtual terminal, making the program operate perfectly.

2.4 The Scheme Used by the Cross-match Tool

The cross-match tool based on the Q3C partition scheme requires two catalog tables (same or different) saved on the database server (local or remote). But the database must be PostgreSQL which deploys the Q3C index; otherwise the program will not work normally. Users should ensure that the database supports remote access when choosing the remote server. The operating process is shown in Figure 1.

2.5 Graphical User Interface

For convenient application of the cross-match tool, we develop a GUI which is displayed in Figure 2. Users can directly click the program code to run the GUI with a mouse, or can also manually use the command in the command line to run it. The GUI is separated from the main program, which assists users in adjusting the parameters according to a certain standard that is used for the main program. That is to say, the main program does not depend on the GUI, and the user can manually select the related file to run the main program. As shown in Figure 2, this cross-match tool provides four different cross-match functions: (I) cross-match with a custom error range; (II) cross-match with catalog errors; (III) cross-match based on the elliptic

error range; (IV) cross-match by the nearest neighbor algorithm. The matched result can only output the matched records or all results. When the default setting of the parameter list is adopted, the matched result will give all parameters of matched sources from table A and table B as well as distance between the matched sources. In addition, users may choose the parameters they need. If all the required data exist in the database, users can apply the tools to do the cross-match work. After the cross-match work is finished, users can find a matched result folder in the local directory, then the resulting set exists in the folder, which saves all matched results or a conflict does not happen. According to the start time of the cross-match work, users may choose what they want.

In the process of developing the cross-match tool, we also develop an auxiliary tool, which implements the data exchange between different databases MySQL and PostgreSQL, and automatically creates the required index, so that astronomers can directly exchange data between different databases and do not need to do extra work to migrate data. This approach is convenient for astronomers to manage the data. The user interface is shown in Figure 3.

3 EXPERIMENTAL RESULTS

In order to confirm the reliability of the tool, we run a simulation experiment. We randomly select part of the Sloan Digital Sky Survey (SDSS) database to do the experiment, with a total of 10^7 rows as the source catalog, and we set the matching radius to $2.5''$, $3''$ and $5''$ to perform the catalog cross-match with itself. Our test platform uses two computers: one is a local database and program running platform, and the other is the platform for the remote database that is accessed through fast ethernet. The specific configurations are shown in Table 1.

The identification result is shown in Table 2. Table 2 indicates that all data can be matched successfully, and no matter whether the matching radius is $2.5''$, $3''$ or $5''$, the identification rate is 100% and the number of incorrectly matched records is 0. Hence, we can draw the conclusion that the cross-match program is fairly reliable.

In order to demonstrate the effectiveness of our tool, we compare our tool with the tool provided by Pei (2011) and Pei et al. (2011). Under the condition in which the hardware configuration is fixed, the performance of cross-match is mainly related to the size of catalog A and not to that of catalog B. We choose different sample sizes of catalog A to do experiments. Selecting 10^5, 4×10^6 and 10^7 records from the SDSS database as catalog A and 10^7 records from the Wide-field Infrared Survey Explorer (WISE) database as catalog B, the time spent on cross-matching is listed in Table 3. The time that Pei et al.'s tool used for cross-matching consists of two parts: one part is for cross-match and the other is for getting parameters from both catalog A and catalog B. Taking cross-match between 10^7 rows of SDSS and 10^7 rows of WISE for example, the time cost for cross-match is 665 seconds and the time spent on getting parameters is 2980 seconds. The total time is 3645 seconds. With the same data, our tool performs cross-match and getting parameters at the same time and the time spent is 1040 seconds. Obviously, our tool is much faster than that of Pei et al. (2011). The CPU occupation of Pei et al. (2011) and ours is 70% and 92%, respectively.

We also compare our tool with more popular cross-match tools, such as CDS x-Match service and TOPCAT. By means of the CDS x-Match service, the UKIRT Infrared Deep Sky Survey (UKIDSS) DR9 (82 655 526 rows) database is cross-matched with itself, and the running time takes 24 min. A small table uploaded by users is very quickly cross-matched with large tables provided by the CDS x-Match service in a few minutes and it is very fast to download the result. Although the cross-match speed of the CDS x-Match service is the fastest, the time to download the cross-matched result is very long, especially for a large sky survey database due to internet bandwidth. Users only cross-match their own tables with the tables provided by this service. Moreover, the total size of uploaded tables is limited to 100 MB for anonymous users and 500 MB for registered users. In addition, if there are some other cross-match tasks that will be performed, users should wait in a queue. With TOPCAT in our computer, we select the UKIDSS DR9 to cross-match with itself; the time spent reading data costs about 46 min and the time for cross-match takes about 36 min. The advantages of TOPCAT are easy use and directly getting the result without secondary processing, but the size of tables that can be cross-matched is limited by the memory of the user's computer due to the pure JAVA program used for TOPCAT. Only in terms of cross-match speed is our tool slower than the CDS x-Match service and TOPCAT. As far as ease of use and convenience are concerned, the CDS x-Match service and TOPCAT are worthy of reference and learning. Compared to all these tools, our tool is a user-oriented cross-match tool, by which users may manage and cross-match the catalogs they need, and in addition transfer data from one database (MySQL or PostgreSQL) to another database (PostgreSQL or MySQL). By means of the GUI, users may very easily apply this tool to perform cross-match. The merits of our tool are that users may automatically create their own databases and cross-match the tables according to their requirements, the tables may be stored on their own computers or servers, and the cross-matched result can also be easily input to the database for easy management and later use.

In terms of hardware, the performance of cross-match is affected by many factors including CPU, network and hard disk drive (HDD). Therefore, network bandwidth and a hard drive spin directly influence the speed of cross-match when using the same computational platform, and so upgrading the network and HDD accelerates cross-match. From the above experiments, the CPU occupation is very high. When the data are large enough, especially when the amount of data totals 10^7 rows, the CPU occupation is 92%. If the number of cross-matched data further in-

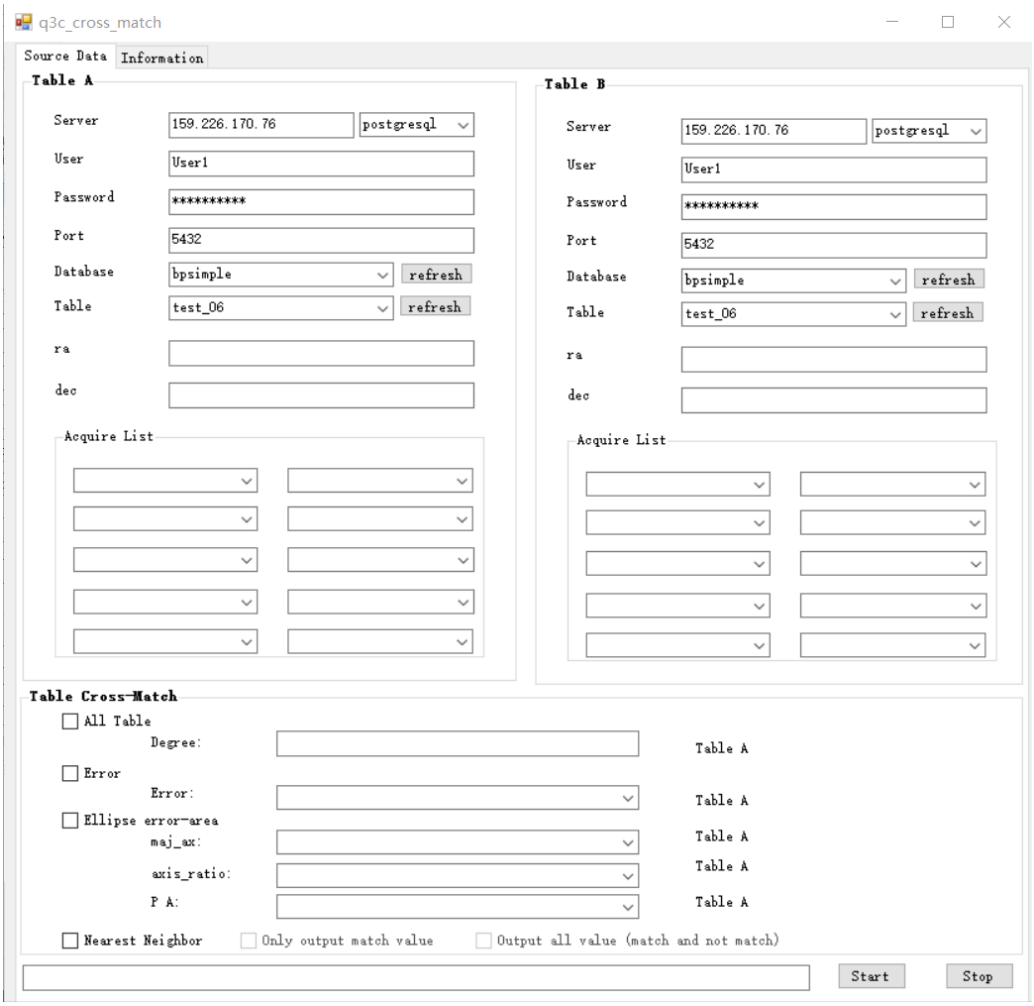

Fig. 2 The operation interface of the cross-match tool.

Table 1 The Configuration of Experiment Platform Software and Hardware

Platform	CPU	Memory	OS	Core	HDD (rpm)	Software
Database	Intel(R) Xeon(E5-4607) six-core at 2.20 GHz	8 GB	Red Hat Enterprise Linux	Linux version 2.6.9–22.ELsmp	10 000	MySQL5.6 PostgreSQL9.3
Program	Intel(R) Xeon(E5630) quad-core at 2.53 GHz	8 GB	Ubuntu 10.04	Linux version 2.6.28-15-generic	7200	Python 2.6.2 MySQL5.6 PostgreSQL8.4

creases, our tool runs unstably based on our present personal computer. As a result, we put forward a solution that a huge table should be split into a number of smaller tables when the number of rows in a table is larger than 10^7. One advantage of PostgreSQL is to permit users seamless access to the split tables. The creation of child tables adopts constraints from PostgreSQL. For much larger tables with more than 10^7 records, our tool automatically splits them into child tables according to source positions, and then users may choose child tables to cross-match one by one.

At present, the bottleneck of any cross-match task is still the I/O limit, not the computation. As the number of survey data increases, the problem of I/O becomes more important. The improvement of I/O speed is the key to cross-match. Pineau et al. (2015) put forward some techniques to strengthen cross-match performance, for instance, large tasks are split into smaller independent pieces for scalability, matching speed to implement multithreading, sequential reads and several tree data-structures. The LSST project adopt a new kind of database, the LSD, which is a framework for storing, cross-matching, querying and rapidly iterating through large survey datasets on multi-core machines and it is implemented in Python. The new techniques developed in computer science can be applied to astronomy. Therefore, the key problem faced in cross-matching will be solved.

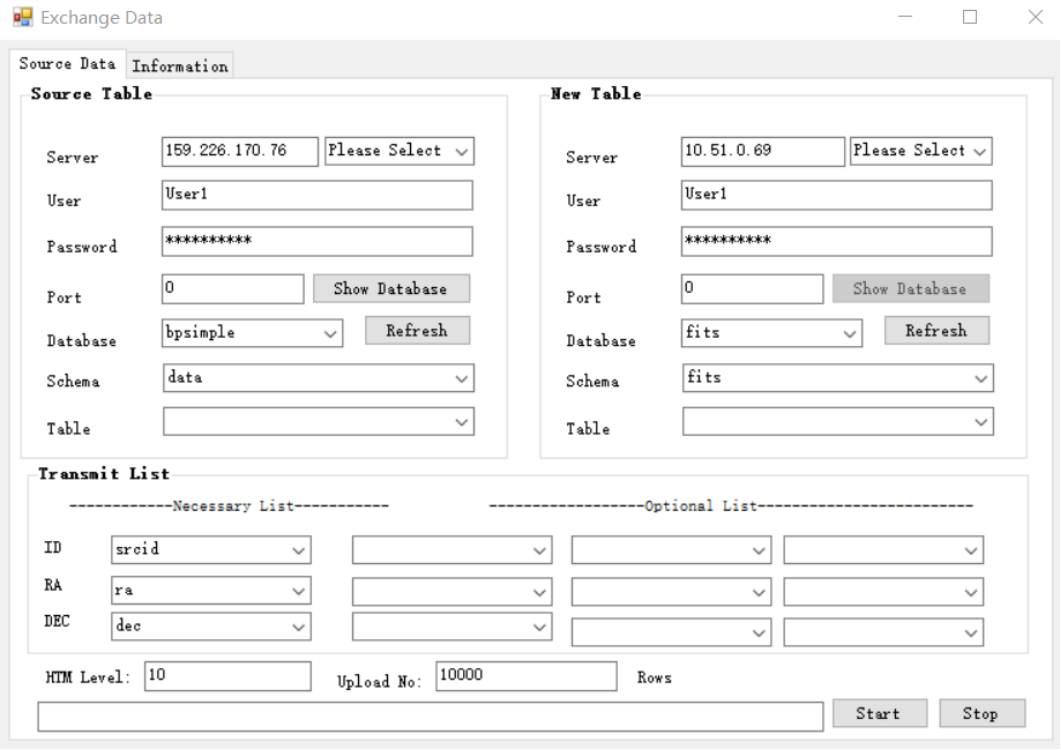

Fig. 3 The operation interface of the data exchange program.

Table 2 The Results of Simulation Experiment

Matching radius (arcsec)	No. of identification results	Matching rate	Wrongly matched no.
2.5	1×10^7	100.0%	0
3.0	1×10^7	100.0%	0
5.0	1×10^7	100.0%	0

Table 3 Time Spent on Cross-match

Method	No. of rows in catalog A	No. of rows in catalog B	Time (s)	CPU
Our tool	1×10^5	1×10^7	4	88.0%
	4×10^6	1×10^7	420	90.0%
	1×10^7	1×10^7	1040	92.0%
Tool of Pei et al.	1×10^5	1×10^7	185	72%
	4×10^6	1×10^7	435	73%
	1×10^7	1×10^7	665	70%

In general, the cross-match refers to the spatial cross-match that only considers position information. In fact, there are many factors to judge which one is the true entry. Budavári & Szalay (2008) presented a Bayesian approach for object matching by means of not only spatial information, but also physical properties, such as color, redshift and luminosity. They provided a practical recipe to implement an efficient recursive algorithm to evaluate the Bayes factor over a set of catalogs with known circular errors in positions. Rots et al. (2009) also applied a Bayesian method to cross identify the Chandra Source Catalog (CSC) with the SDSS DR7 source catalog, which takes into account the positions, position errors, as well as the detailed footprints of the catalogs, and provided a Bayes factor and an associated probability for each match. In terms of cross-match accuracy, there is still much work worth studying. In the present version, our tool has not incorporated parallel computing or distributed storage, and

only implements spatial cross-match. In the future, our tool may feature these aspects, and thus the speed and accuracy of the cross-match task will be improved.

4 CONCLUSIONS

This paper is devoted to the development of an efficient and easy-to-use catalog cross-match tool. Considering astronomical data characteristics of large-scale, multiband and distributiveness, we analyze existing cross-match tools from a number of sources and develop a simple and efficient data fusion tool according to the needs of users. Our tool can be applied to tens of millions of lines of data, or even larger, and we adopt the method of pool processing to fully use the CPU, which ensures the efficiency of cross-match. Users can arbitrarily choose the cross-match functions to obtain the final multiband cross-match result according to their own needs. Given high identification speed and good accuracy, the tool improves large scale catalog cross-match. Moreover, it is easy to learn and use for astronomers. It has been important for subsequent data mining and statistical analysis work. Nevertheless, there are still lots of functions that need to be improved in the future. For example, this tool is limited in terms of system environment, and can only be used in the environment based on the PostgreSQL database system. For the matched result set, users may choose less than 10 columns of data in each table, not counting RA and Dec, or all of the columns are outputs when setting defaults. If users want more than 12 columns, they need adopt the default setting. In the following work, the cross-match services can be fully deployed on Web servers, letting users input data into our database server, using our platform to do the cross-match work, which will greatly reduce the workload in astronomy. Moreover, the user can give feedback through the network service in the process of usage, so our program will be more robust. In the program itself, we can further improve the high efficiency of the program using the multiprocessing module to implement the parallel program. The benefit is that it does not produce potential errors but with its high efficiency, the direct operation of processing usually will be more efficient than the use of a good package library. So, we can consider using the Python subprocess module to initiate multiple processes, but in this case each process will get its own unique result set, then we still need secondary processing to calculate the final result set. Improvements in database performance can directly improve the running efficiency of the program. When the size of tabular data is large enough, the memory reaches a bottleneck, even if using the index, and the query work can still be very slow. Here it is necessary for us to do the table division operation, dividing a large table into multiple small tables to do the cross-match work, so that the efficiency will be greatly improved. Moreover, if the database is arranged in the distributed cluster layout method, the robustness of the program will be much higher. So in the future, we will implement many better ways (e.g. MapReduce, Spark) to improve the efficiency of the program and apply new data mining methods to perform not only spatial cross-match, but also physical matching. Altogether, the development of new computer technologies, new types of databases and database index methods oriented to big data is important for today's multiwavelength observations and the time domain science related to upcoming survey telescopes.

Acknowledgements We are very grateful to the referee for his insightful comments. This paper is funded by the National Key Basic Research Program of China (2014CB845700), the National Natural Science Foundation of China (NSFC, Grant Nos. 61272272, 11178021 and 11033001) and NSFC-Texas A&M University Joint Research Program (No. 11411120219). We acknowledgment the SDSS, WISE and UKIDSS databases.

References

Boch, T., Pineau, F., & Derriere, S. 2012, in Astronomical Society of the Pacific Conference Series, 461, Astronomical Data Analysis Software and Systems XXI, eds. P. Ballester, D. Egret, & N. P. F. Lorente, 291

Budavári, T., & Szalay, A. S. 2008, ApJ, 679, 301

Gao, D., Zhang, Y.-X., & Zhao, Y.-H. 2009, RAA (Research in Astronomy and Astrophysics), 9, 220

Koposov, S., & Bartunov, O. 2006, in Astronomical Society of the Pacific Conference Series, 351, Astronomical Data Analysis Software and Systems XV, eds. C. Gabriel, C. Arviset, D. Ponz, & S. Enrique, 735

Landais, G., Ochsenbein, F., & Simon, A. 2013, in Astronomical Society of the Pacific Conference Series, 475, Astronomical Data Analysis Software and Systems XXII, ed. D. N. Friedel, 227

Maselli, A., Massaro, F., D'Abrusco, R., et al. 2015, Ap&SS, 357, 141

Metchev, S. A., Kirkpatrick, J. D., Berriman, G. B., & Looper, D. 2008, ApJ, 676, 1281

Pei, T. 2011, , Master's thesis, National Astronomical Observatories, Chinese Academy of Sciences

Pei, T., Zhang, Y., Peng, N., Zhao, Y. 2011, SCPMA, 41, 102

Pineau, F.-X., Boch, T., & Derriere, S. 2011, in Astronomical Society of the Pacific Conference Series, 442, Astronomical Data Analysis Software and Systems XX, eds. I. N. Evans, A. Accomazzi, D. J. Mink, & A. H. Rots, 85

Pineau, F., Boch, T., Derriere, S., & Arches Consortium. 2015, in Astronomical Society of the Pacific Conference Series, 495, Astronomical Data Analysis Software an Systems XXIV (ADASS XXIV), eds. A. R. Taylor & E. Rosolowsky, 61

Rots, A., Budavari, T., Szalay, A., & Hagler, J. 2009, in Chandra's First Decade of Discovery, eds. S. Wolk, A. Fruscione, & D. Swartz, abstract #188

Taylor, M. B. 2005, in Astronomical Society of the Pacific Conference Series, 347, Astronomical Data Analysis Software and Systems XIV, eds. P. Shopbell, M. Britton, & R. Ebert, 29
Wenger, M., Ochsenbein, F., Egret, D., et al. 2000, A&AS, 143, 9
Yamauchi, C. 2011, PASP, 123, 1324
Zhang, Y., & Zhao, Y. 2003, PASP, 115, 1006
Zhang, Y., & Zhao, Y. 2004, A&A, 422, 1113
Zhang, Y., Zheng, H., Pei, T., & Zhao, Y. 2012, in Proc. SPIE, 8451, Software and Cyberinfrastructure for Astronomy II, 84511Z
Zhang, Y.-X., Zhou, X.-L., Zhao, Y.-H., & Wu, X.-B. 2013, AJ, 145, 42