



Weimin Zhu

*MSc. 1993, University of Toronto.
Project Manager, GDB Project, Toronto, until 2000.
Head of Bioinformatics, Synax Pharmar, Toronto, until 2002.
At EMBL-EBI since 2002.*

Database Research and Development

71

INTRODUCTION

In February 2008, the Database Applications team was reorganised to form the Database Research and Development team due to the creation of the PANDA group.

The team's new mandate is to conduct research and development to find new technologies and solutions to meet challenges related to very large databases (VLDB), which includes data distribution problems when network speed is a bottleneck, and the solutions required to manage and query VLDBs efficiently.

The size of bioinformatics databases has been increasing exponentially over the last ten years. Some core resources are approaching, or have already reached, multi-terabytes in size. This trend of growth has accelerated in recent years by the introduction of new data types and high-throughput data producing technologies. Today, we are facing all the challenges a VLDB brings, such as those in data operational management, data access performance, and data mirroring and distribution. Our current infrastructure in these areas thus requires upgrading in order to realise the full potential of data-rich resources, and optimise the usage of our human and hardware resources.

Data distribution or data synchronisation has been a very active research area in the information technology sector for quite a few years. As a result, some useful tools, such as rsync and its variants, have been developed. The core of the technology, also known as delta compression or delta encoding, is to find the differences (deltas) between two sets of files located remotely from each other, and only those deltas are transferred to the target computer to allow it to rebuild a new version of the files based on the older version. If the deltas are significantly smaller than the full data files, a significant network saving can be achieved. The technology is working very well for some applications, such as internal data synchronisation and remote software distributions. However, it has not been successfully adopted by the database community for the distribution of large datasets, although it has been attempted by many people. When applied to large datasets, the latency of the runtime calculation to identify the deltas between source and target files becomes a new major bottleneck. Also, unclustered changes in files, which are common in bioinformatics databases, can result in full data transfer instead of deltas. A few other technologies, such as peer-to-peer solution and data replication, have also been tried by some bioinformaticians. Although some progress has been made, their application in distributing large-scale biological databases is still very limited.

ACTIVITY UPDATE

This year, our main focus was first to develop a new algorithm, sdeltac, to shorten the network latency for distributing large datasets across the network. The algorithm is a delta encoding-based solution but by taking advantage of some of the unique characteristics of biological databases, the algorithm is also a structure-based differential distribution system that overcomes the limitations of existing delta encoding solutions (see figure 1A). Some key features of biological databases include:

- in a typical database file, an entry is a common atomic data unit separated by well-defined delimiters within a specific database, such as '/' for separating entries in the EMBL-Bank archive and UniProt Knowledgebase, '>' for FASTA sequence data files, and '\n' for records in Ensembl MySQL database files. Entries also exist in XML format which has become a common alternative data format across many biological databases. In XML-based databases, entries are separated by unique opening and closing tags.

- most of the entries in a database release are unchanged. Changed entries, either updates or new insertions, represent only a small percentage of the total release. For instance, by comparing the files in non-WGS (whole genome shotgun) data divisions of EMBL-Bank release 93 and 94, only 14.9% entries have changed. By including most of the WGS projects, the result is similar (14.5%). It means that without further optimisation, only 90GB data need to be transferred for a database with total size of 600GB.
- a significant percentage of changed entries are modified entries, not new ones. The modified entries counted for 12% of the total 15% changes in the above EMBL-Bank example. This means that for this 600GB database, only 18GB has to be transferred, while the remaining 72GB data can potentially be delta compressed. Similar results were obtained by analysing other databases.

These unique features allow us to use entry-level comparison between releases as the higher level of encoding (figure 1B). Huge computation power and time can be saved by removing those unchanged entries from the lower level comparison – by word or chunk comparison process. Source entry data are stored in a hash table for quick target entry lookups. This level of comparison also makes the word-level comparison more accurate and manageable when a data file is very big.

Word-level comparison (figure 1C) starts by seeding the source tokens, a part of the source words, along the target file. This overcomes splitting the target file into an arbitrary size of a set number of words, and enhances the possibility for source strings to be mapped to the target. The token mappings are then extended from both sides of the token, to maximise the mappings and minimise the delta that have to be transferred to the client.

Entry-level and word-level comparisons are the core of the sdeltac algorithm. The delta data, string literal and source offset and length to be copied are pre-computed, and used by client to reconstruct the target file set from its local copy of the source file set.

The sdeltac algorithm is implemented using the C language, and consists of server and client components. The delta data are stored in RDBMS, which further reduces network latency. A functional prototype has been implemented including all the components except the interfaces (figure 2).

A workshop funded by the UK Research Councils (RCUK) has been scheduled for 19–20 October in Beijing, China. The participants of the workshop will exchange ideas and solutions to the challenge of distributing large biological databases, from network and computational angles.

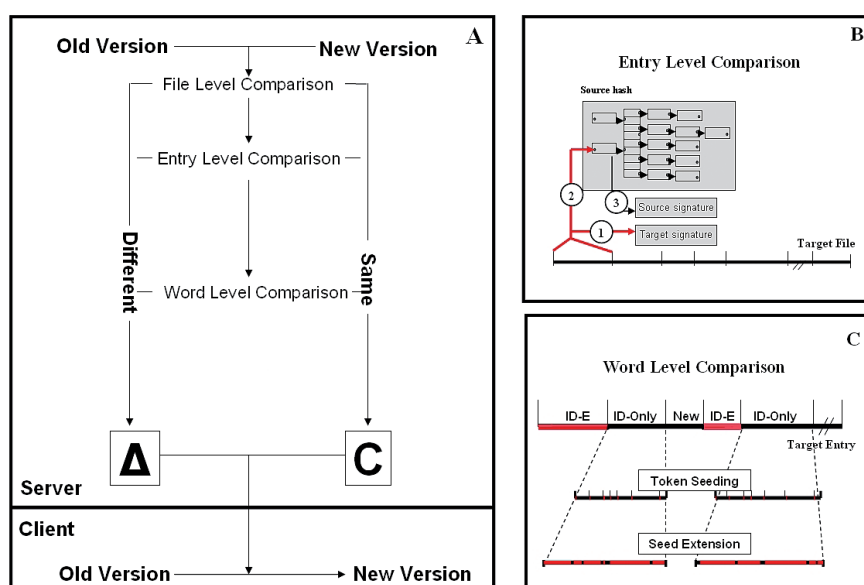


Figure 1. Overview of the sdeltac algorithm. A. The differences and similarities, shown as Δ and C respectively, between old and new sets of files are computed on file, entry and word levels on the server side. The client reconstructs the new file set from its local copy of old files, and the delta data (Δ and C) that are transferred over the network. B. Entry-level comparison. An entry read from a target/new file is calculated for its signature (1), and identifier's hash value. The hash value and entry identifier are used for looking into source/old file hash tables (2). The returned source entry signature (3) is compared with target signatures to determine whether the target entry is changed, unchanged or new. C. Word-level comparison. The changed target entries, with same entry identifiers but different signatures (ID-only) from their source counterparts, are subjected to two tiers of word matching processes. Token seeding is the process of seeding the source tokens (1/10 of predefined word size) to a target entry. The token matching is further extended by token extension process, to maximise the matching between source and target entries and minimise the Δ . Matched areas are in red, and unmapped areas are in black.

FUTURE PROJECTS AND GOALS

The continuing development and optimisation on sdeltac, on both algorithm and implementation, will be our focus for next year. New developments will include:

- algorithm adaptation and implementation work on compressed file formats;
- further development of RDBMS database schema, to allow selection of a subset of a database and version skipping;
- collaborations with database projects and data centres to test the utilisation of the program as a potential data distribution and data mirroring solution;
- commencing work on interfaces for the client to retrieve delta data from the RDBMS database.

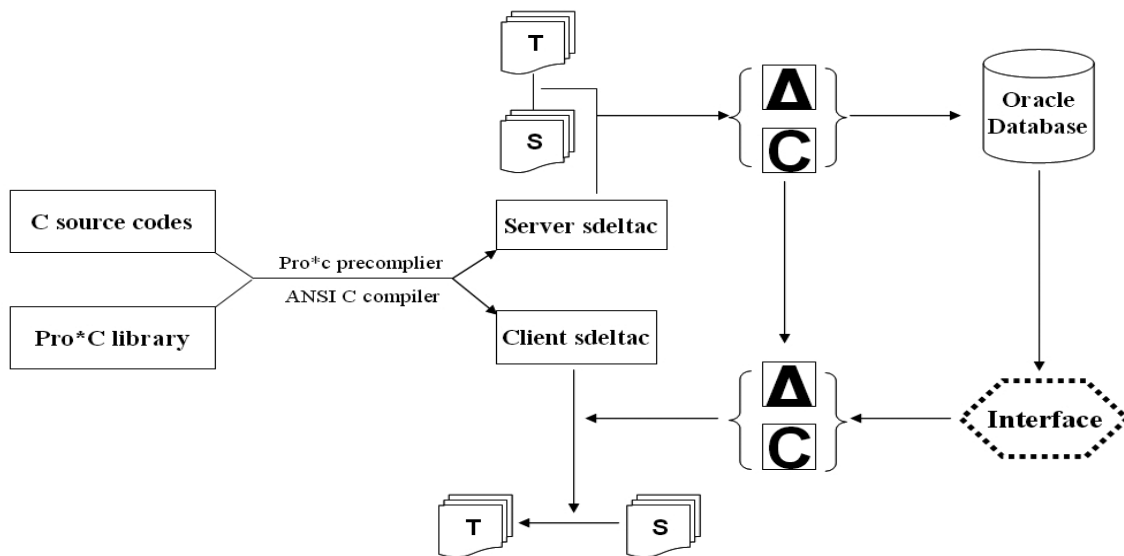


Figure 2. Implementation of sdeltac algorithm. The C program suite has server and client components. The server sdeltac processes source and target file sets. Resulting delta data (Δ and C) are stored in RDBMS, which can be transferred directly to the client, or through client interfaces, such as web services, browser or API. Client reconstructs the target file set from these delta data and the local copy source file set. The future development of the database schema and interfaces will offer the client the flexibility to select subsets of a database, and choose the versions of the source and target file sets.