Share

Technology Challenges for Lasair

Created by Roy Williams
Last updated Mar 09, 2020 • Analytics

Roy Williams

Our mission is to provide rich access to the stream of transient alerts from the LSST survey, giving users immediate notification of exciting science, and also providing an interface to the archive of past alerts.

We already have an ingestion and delivery system for the prototype (Lasair-ZTF), that has been running to nearly 2 years. In another 2 years, we will be prototyping Lasair-LSST, that will need to cope reliably with about 50 times as much data. The computing hardware will be provided by IRIS, the UK science grid. The current system is based on Kafka, MySQL, and Apache. We expect the future system to achieve higher performance using a combination of: parallel processing, software technology, data caches, and high-performance hardware. In the following, we consider some of the key questions.

Queries

In the Lasair-ZTF, users can type in SQL to query the database tables that represent past alerts. This interface encourages innovation through flexibility and wide scope. But these queries are being built by non-experts, and can be inefficient, or run for a very long time, or even fill up the temporary disk space. Once a user has built a static query, they can make it a streaming query, running many times as alerts are ingested, so that exciting science is immediately available to that user. It has been a challenge in Lasair-ZTF to unite the different semantics of static and streaming queries. We are investigating different dialects of SQL: MySQL, KSQL (Kafka), CQL (Cassandra), and ElasticSearch. It would be a challenge to have different dialects for static and for streaming queries. It may be necessary to control more closely the syntax of the user-generated queries, so that (a) selected attributes can only be from a pre-selected set, for example "`mean_mag_g`", "`jd - 2400000.5 AS mjd`", and (b) the constraint clauses can only come from a limited subset, for example "`mean_mag_g < 19`", "`classification IN (SN, NT)`". This approach follows the way Vizier tables are queried.

Beyond the Query

Users will not always be content with database queries to discriminate the truly exciting event from the dross; they will want to pull images and light-curves from the Lasair-LSST data stores, as well as context from the global astronomical archives (virtual observatory).

The existing prototype Lasair-ZTF already provides a rudimentary Jupyter notebook service for deeper mining, but we could expand this in Lasair-LSST, with a solid API, long-running jupyter notebooks, and other ways to mine more deeply. In particular, the LSST project is developing the "LSST Science Platform", and it should be integrated with Lasair-LSST.

Relational Database

The LSST data will be parsed into relational and binary categories. The former is a set of database tables with relational integrity, and the latter a filesystem or blob store of some kind. Tradtional relational systems provide a wealth of access methods, optimisation mechanisms, and familiarity, while newer systmes score better on scalability and speed than over convenience and utility. It is crucial to size the database correctly to decide which path to take. The database will have a set of "diaObject" records that corresponds to multiple observations of a single star or galaxy; along with this "summary" is a set of "diaSource" records, one for each brightness measurement in the light curve. If these diaSource records are all represented in the relational database, it becomes much larger. An alternative is to define each light curve by a few "features" that are more useful than the individual diaSource records; the feature vector will also be easier and more efficient to query (no JOIN), and the database can be smaller and faster. The attached data estimates databases sizes by extrapolating from ZTF.

Blob store

Images and perhaps light curves will be stored in a "blob store" -- each binary large object (blob) is identified by a identifier. This has been a file system in Lasair-ZTF. We need to be able to push in blobs fast when the ingest happens, and for a researcher, push out a list of blobs for a list of identifiers. Should be manageble and scalable.

Hardware

Instead of the traditional spinning disk, a more expensive option is the solid-state disk (SSA). Sometimes SSA is much faster, sometimes not. We will present evidence on where the SSA is worth its extra cost. Of course, if we are using cloud services, whether academic or commercial, it may not be possible to ask for VMs that have SSA resources attached, especially if we want a large number of VMs in a parallel cluster.

Caches

Here we mean two things: re-use and flow-control. A re-use cache is used by the Sherlock classification engine -- if we have already classified a specific point in the sky, it is quicker to look up that classification than to re-compute it. Note that loss of a re-use cache is not critical, it just slows things down. A flow-control cache can be used when many processes contribute to a single process, for example ingestion nodes pushing to a database. Each node can keep content locally (disk, memory) until the downstream database is ready to receive. Loading of the database is spread over a longer time, reducing the required peak flow rate. But the downside is that the local cache may fill up, leading to loss of data.

User content

To what extent will users be able to store data with the Lasair-LSST system? Already in the prototype, they can upload a "watchlist" of several 1000 sources they are interested in, as well as store queries and make comments on interesting objects, Perhaps the watchlist capability can be expanded to millions? Perhaps users will store results of queries and convert those to watchlists? Once user data is stored, the questions are how much data, and for how long is it kept.

Reliability

We do not want users to be disappointed with Lasair-LSST: data lost, an exciting event and the system is down, a first time user never comes back because of failure when they try. We must have a match between capability and expectation. Both Lasair-LSST and the IRIS cloud platform have minimal staff resources and no 24-hour response. We need to be careful with backups, disaster recovery, and risk analysis.