Using HUPO Proteomics Standards With Software Inputs and Outputs for Galaxy-based Multi-omic Informatics

Thomas McGowan[1]; James Johnson[1]; Pratik Jagtap[2]; Subina Mehta[2]; Praveen Kumar[3]; Timothy Griffin[2]

[1]University of Minnesota Supercomputing Institute,[2]Department of Biochemistry, Molecular Biology and Biophysics, University of Minnesota, [3]BICB (Bioinformatics and Computational Biology), University of Minnesota

Introduction

The Galaxy-P project is developing tools that aggregate, process and visualize multi-omic data. Each tool processes variously formatted and pre-processed files: from proprietary binary files to text files adhering to standards created by time and habit rather than deliberation.
We have created the Multi-omic Visualization Platform (MVP) (a Galaxy visualization plugin) to use mzIdentML and mzML files as standard inputs. The mzIdentML file is required for running the tool.
While XML files excel at unambiguously defining data types, they make querying aggregate data difficult.
In our applications, we are using the power of open standards developed at HUPO along with an easily queried relational database schema.

Methods

The MVP application is a Galaxy visualization plugin allowing researchers to inspect and verify multi-omic data sets. In addition, the application will orient a user between an MS/MS scan, a peptide, a protein, and a genomic exon location all on one screen.
To prepare data for the visualization, we have built a java application to transform mzIdentML, mzML and/or MGF (Mascot generic format) files into a SQLite3 database. Via an established API, Galaxy serves this data to the JavaScript visualization tool. The JS then presents peptide, protein and genome information graphically to the user.
We have also built a Galaxy SQLite query tool allows an advanced user direct access to the SQLite database. When accessing the database, the user is interacting with the combined, transformed mzIdentML / mzML / MGF data.
The java application is a StAX based XML parser. We have ranked running speed as a primary goal for the tool. We run the parser on a multi-core (10 - 20) HPC node. Each parsing thread is assigned a specific XML element for parsing. With 10 - 20 threads running, we can parse very large XML files in very few minutes. Parsing is directly translated to SQLite CREATE, INSERT, and UPDATE commands.
After parsing mzIdentML and mzML files into a SQLite schema, the java application enters a data transformation step. The raw data tables are transformed into mildly denormalized tables aimed at enhancing the performance of the MVP visualization. By creating new tables in addition to the existing tables, we can maintain data integrity on the mzIdentML/mzML data while vastly improving the user's experience of the multi-omics visualizations.

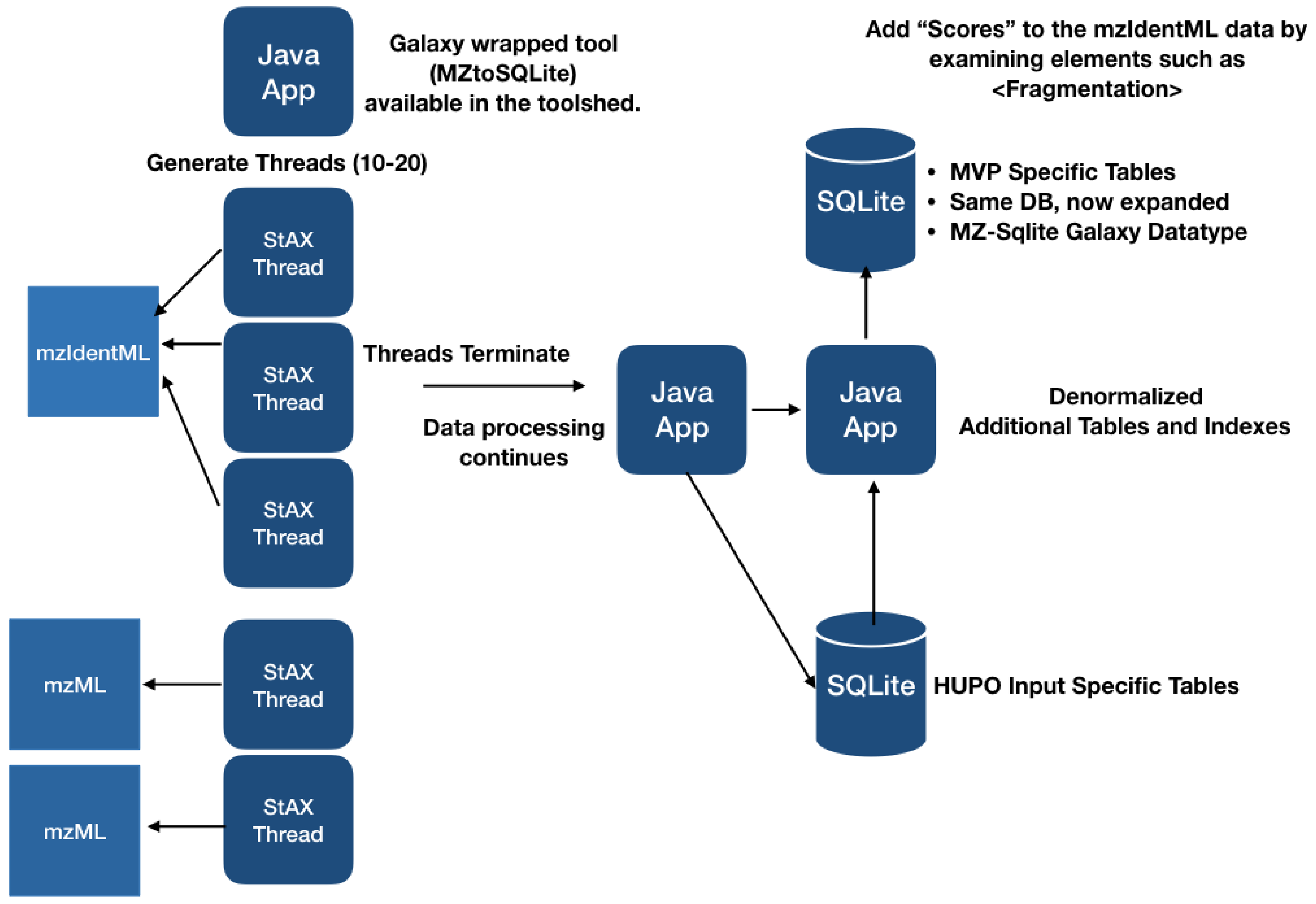
Preliminary Findings and Next Steps

To date, we see excellent performance in parsing multi-gigabit XML inputs into a SQLite database. Runs are completed within 3-5 minutes. Since the database is read-only after creation, we have been prolific in generating table indexes for maximizing run-time reads via the Galaxy API. SQLite3 is performing well for us.
One issue we are seeing is in the lack of consistency in XML output between various protein search applications. Though the applications are honoring the HUPO specification for required and optional fields, often very useful, optional fields are left out of an mzIdentML file. This forces us to present a very lean data profile to our users.
In 2018, we will extend our tool-set to allow a user the ability to generate schema verified mzIdentML and mzML files from a subset of the original data. So, after visual inspection a researcher may generate an XML output based on the peptides of interest for use in further data processing in multi-omics research. In addition, we will be looking at using the SQLite "Write-Ahead Logging" option to allow for multi-threaded database operations.

Acknowledgement

We acknowledge NSF grant 1458524, and NIH grant 1U24CA199347 for funding support to the Galaxy-P team at the University of Minnesota.

Threading

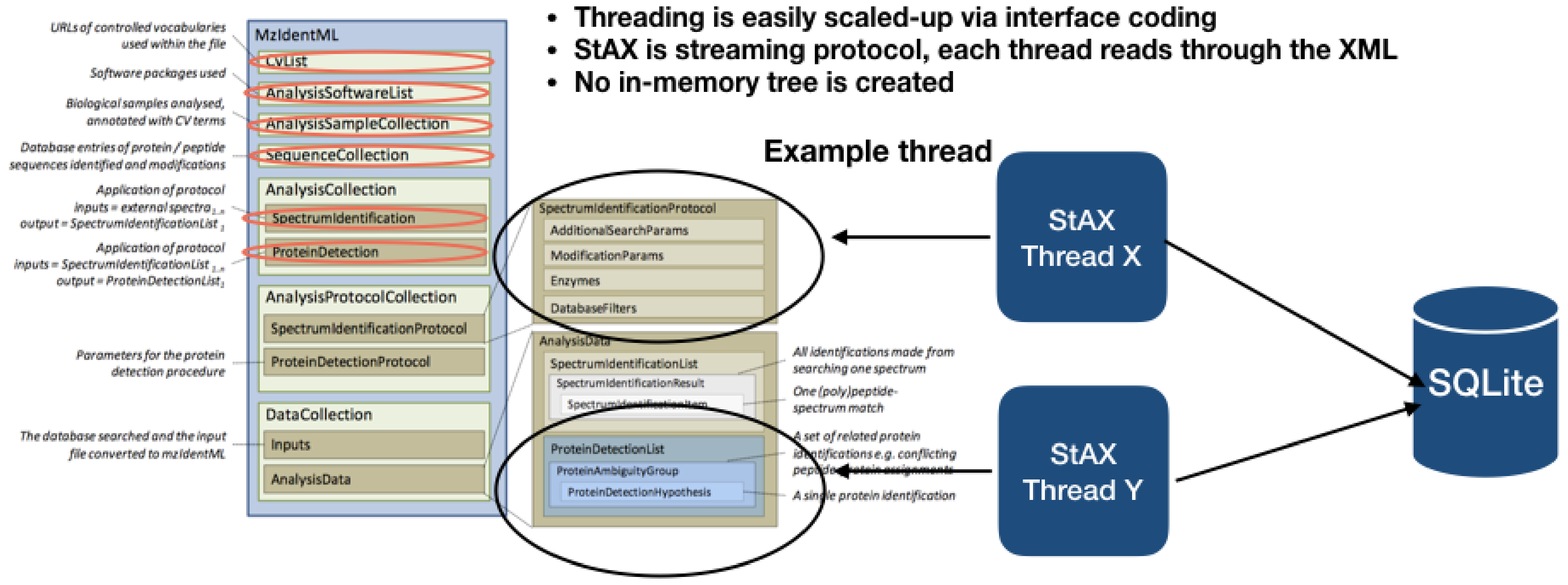

Java App
Galaxy wrapped tool (MZtoSQLite) available in the toolshed.
Generate Threads (10-20)
mzIdentML
StAX Thread
StAX Thread
StAX Thread
mzML
StAX Thread
mzML
StAX Thread
Threads Terminate
Data processing continues
Java App
Java App
Add "Scores" to the mzIdentML data by examining elements such as <Fragmentation>
SQLite

- MVP Specific Tables
- Same DB, now expanded
- MZ-Sqlite Galaxy Datatype

Denormalized
Additional Tables and Indexes
SQLite
HUPO Input Specific Tables

XML Parsing

- Multiple threads generated for parsing
- A thread reads a single XML element, ignores the remainder
- Reading is thread-safe
- StAX is a pull protocol—clean, clear code
- Threading is easily scaled-up via interface coding
- StAX is streaming protocol, each thread reads through the XML
- No in-memory tree is created


Example thread
URLs of controlled vocabularies used within the file
Software packages used
Biological samples analysed, annotated with CV terms
Database entries of protein / peptide sequences identified and modifications
Application of protocol inputs = external spectra$_{1..n}$ output = SpectrumIdentificationList$_1$
Application of protocol inputs = SpectrumIdentificationList$_{1..n}$ output = ProteinDetectionList$_1$
Parameters for the protein detection procedure
The database searched and the input file converted to mzIdentML
MzIdentML
CvList
AnalysisSoftwareList
AnalysisSampleCollection
SequenceCollection
AnalysisCollection
SpectrumIdentification
ProteinDetection
AnalysisProtocolCollection
SpectrumIdentificationProtocol
ProteinDetectionProtocol
DataCollection
Inputs
AnalysisData
SpectrumIdentificationProtocol
AdditionalSearchParams
ModificationParams
Enzymes
DatabaseFilters
AnalysisData
SpectrumIdentificationList
SpectrumIdentificationResult
SpectrumIdentificationItem
ProteinDetectionList
ProteinAmbiguityGroup
ProteinDetectionHypothesis
All identifications made from searching one spectrum
One (poly)peptide-spectrum match
A set of related protein identifications e.g. conflicting peptide-protein assignments
A single protein identification
StAX Thread X
StAX Thread Y
SQLite