Project Name: Automated Processing for e-HTPX

Name of pipeline: Processor.py

Author information: Graeme Winter

Current Status: Working but not released to general public

Purpose

To automatically process X-Ray diffraction data from raw images to properly scaled & truncated reflection files, while also determining as much useful information on the was as possible.

High Level Description

The pipeline consists of a number of existing programs from the CCP4 suite and elsewhere, a few specially written programs, a couple of extension modules and a great deal of Python code. The existing programs which have been incorporated into the pipeline are:

mosflm	sortmtz	mtz2sca
scala	mtzutils	cad
truncate	rebatch	gnuplot
labelit	reindex	
best	f2mtz	

The new programs are:

scalem (in development) – a program to calculate an appropriate error model for diffraction data
mattprob – a tool for determining the most likely number of molecules in the ASU based on a published method (Kantartdjieff and Rupp)
mtzq – a "jiffy" utility to get information from mtz files

The modules are:

DiffractionImage – a library for accessing and analyzing X-Ray diffraction images
StatisticalTools – a couple of statistical tools from gsl boosted to Python

The Python code represents the actual pipeline!

Jiffies

The pipeline includes "jiffies" to convert between reflection file formats, query mtz files for information and lots of things to manipulate reflection files. In addition a framework for "running" programs has been developed, which includes resource management and control of parallel tasks.

Decision Making

A fair amount of the code in the pipeline is involved in decision making. Examples include:

- comparison of solutions from indexing
- analysis of integration statistics to decide which batches to use for indexing
- decisions about which point groups to test for scaling & merging
- analysis of the scaling statistics for radiation damage and rescaling
- combination of data collection strategies from different sources

Data Standards and Management

The data standards adopted for the DNA project have been used in these developments. These data standards start from a description of the data model in XML-Schema. This is then transformed into Python and Java implementations as a class structure, including marshaling & un-marshaling methods.

Within the DNA framework all communications take place by sending these XML documents via HTTP through sockets. In this automated pipeline the XML representation is not used, but the native Python objects are used.

Languages

This pipeline is developed in Python, Fortran and C/C++.

External dependencies

The system includes some packages written using mtzlib from CCP4, and gsl the Gnu Scientific Library.

Context/Audience/Environment

The system is designed to work at the beamline but will also work elsewhere (I tend to use it on my laptop and workstation). Due to the large data requirements, the system is not naturally suited to working remotely unless the data can be provided there somehow – for instance operating as a web service is probably not efficient, although it should be possible.

Links to Supporting Documents

http://www.dna.ac.uk

References

Winter, Briggs & Ballard, ACA 2004