Workflow Integration, Benefits and Scope of NOMAD

Aaron Brick, November 5, 2002
UCSF Functional Genomics Core; David Erle, P.I.

introduction

I have been asked to prepare a report on the use of NOMAD; what benefits it provides; and the role it occupies in our labs' workflow. The problem is an interesting one because NOMAD serves researchers in various labs in very different ways. To acquire a broad understanding of NOMAD's usage and placement in these various circumstances I conducted a series of informal interviews with users and developers. These revealed some commonalities and an impressively divergent range of opinions regarding the system. Some users' needs are well-met, and some not at all; it is this discrepancy which currently deserves our attention.

functionality and usage

As you know, NOMAD is a web-based database for the voluminous data generated in microarray experiments. Like any database, its purpose is the ordered storage and rapid retrieval of complex information. It provides for the definition and description of many different entities in the experimental process, as well as recording their relationships. Instantiations of various data types can also exhibit specific, user-assigned parameters.

Researchers use NOMAD in different ways. Some, who work with small numbers of arrays, upload each and query them extensively to find salient data. Others with many arrays, like a core facility, are more interested in determining the quality of the arrays, and thus their production techniques. Still others use NOMAD as a basis on which to build analysis tools. Serving these different needs is a vital challenge; doing so successfully establishes NOMAD as a useful tool, attracting more users and support. At the cost of some administrative and political overhead, we can all take advantage of the work done by our collaborators from other institutions.

development patterns and statistics

The four core development team members are from labs with very different

procedures and goals. Thus, NOMAD is a broader product than reflected by its use in any one place; and development has proceeded in many directions simultaneously. In order to establish more focus we implemented a rotating chairmanship, which has improved coherence and concentration. The chairman at any given time sets the agenda and controls half the development time spent by other members. This system has worked well for us, in that disparate issues are addressed in sequence; for example, we have just moved from fixing some interface bugs on to the rollout of a new navigation system. Next will be some reoganization and modularization of the code, including the notoriously hairy query system.

In-house development allows the system to grow with and adapt to local practices. Our ability to do so is a considerable advantage over inviolable commercial software. Of course, this dynamism depends upon communication between users and developers with regard to new needs and issues. Besides verbal communication, two major channels we use for interacting with users are the users' mailing list and the bug tracker database.

We use online databases to keep track of various kinds of current issues. Problems discovered by users get our attention, and we collaborate to resolve them. Since this system was implemented last year 161 bugs have been filed; only 12 remain open. It is important to note that a bug not reported will probably remain unsolved. Since we employ no dedicated quality assurance engineers, the burden of discovering bugs must be shared between developers and users.

Our first release as a group was in December, 2001; four more have followed, an average of one every other month. Each has included new features and bug fixes, and our users consistently upgrade to newer versions.

problems and approaches

The two main problems users have with NOMAD regard its interface and its integration with analysis tools. Both deserve concerted attention, as they each were named by several users as trouble spots, though some users are fully satisfied with one part and not at all with the other. Challenges also exist for the developers, addressed separately.

1. interface

Since NOMAD's interface has been developed piece by piece, it is less cohesive

than that of more explicitly designed pieces of software. Some parts can be hard to find or unhelpful; of course we address these when they are discovered. Occasionally we are asked to redesign the interface to accomodate alternate, or new, workflows; these are harder problems, but with great rewards.

One bottleneck is the time-consuming nature of uploading arrays, which we are tackling by implementing a batch procedure and reforming the parameterization system; see Appendix B. Other usability concerns with the query builder and results will also be addressed, and are given in detail in Appendix A. Lastly, the new browser, soon to be released, is both easier to use and more powerful than the old one. It will facilitate the creation and navigation of data, consolidating many options to make them more easily available.

2. data preparation and analysis capabilities

Array data goes through many permutations and algorithms before useful analysis. Many users have complained about the tedium of loading files into various separate tools, ad nauseam. Potential tool integration is roundly applauded, so we have designed a plugin system which allows programs, even those written in other languages, to be called from within NOMAD. Any useful, automatable process is a good candidate for integration as a plugin.

There is a practical distinction to be drawn between preliminary analyses and detailed research; the latter requires graphical and interactive software well outside the scope of NOMAD. To that end our group is investigating available commercial software. Array quality control and diagnostics, however, are within our purview, and work is being done to make those facilities seamlessly available.

One user observed to me that querying raw data is relatively useless, and so normalized results must be available within NOMAD. Another complained that she was obliged to refer repeatedly to NOMAD, her lab notebook, and Bioconductor in order to complete an experiment. To both of these ends we are working on integration of the Bioconductor package. The basic functionality of array data export in the R format is in place, as is a mechanism for providing Bioconductor's graphs via a web page; the development group is in ongoing discussions about how to store the resultant normalized data. The easy availability of Bioconductor's output will make NOMAD much more comprehensive for many users.

3. development problems

Development of NOMAD has historically been something of a struggle between

the near-polar goals of stability and newness. There has been tension between developers and the priorities of our respective labs, but our interest in collaboration has always won out over our differences. A more problematic legacy is quickly written and insufficiently tested code, which inconveniences users and developers alike. This problem is addressed by code revision and modularization after its execution; it's an unideal situation, but at least code is provided and refactoring achieved by our collaborators. The project does manage to move ahead.

future features

1. spots display

Although NOMAD traditionally displayed spot images in queries, the feature was removed two releases ago owing to technical impracticalities in their storage. After discussion a new design was accomplished and developers set off to implement the new standard. The setback we encountered was a lack of support from the maintainer of the query system. It is our intention to divorce him from that position, and by modularizing the system, gain the ability to work with it. In the next release cycle, spots will once again be displayed in query results.

2. affymetrix data import

Support for Affymetrix array data would be a great bridge-builder for the NOMAD project. Of course, the problem is non-trivial, but the adaptation is indeed feasible. A framework for different array formats is already in place. Despite differences in the definitions of certain attributes, we hope that simple queries of normalized Affymetrix and GenePix data together will be meaningful. This effort will be given a great boost when, as planned for the next release cycle, the query system is further modularized.

3. quality control

A built-in mechanism for determining and recording the relative quality of arrays will be of great utility for both the core facility and experimenters. This goal is can be pursued in parallel with the Bioconductor connection, since the output of that program is so helpful in making a quality assessment. A simple interface for manually rating each array has already been implemented, and can be expanded and integrated with other relevant parts of the NOMAD interface. Each spot on an array already has such an attribute, the Spot Flag; allowing the user to set this value from within the query results would provide another useful level of control.

conclusion

NOMAD is a robust, if moody, solution. We have done well in making it useful for a variety of labs, and been rewarded by the participation of other institutions. Organizational difficulties are a small price to pay for the output of our collaborators. At any rate, the compromise we have come to - the balance struck between reliability and dynamism - has proved wholly appropriate. The system is largely reliable, and new development is rolled out frequently enough for most needs. As long as users' needs are communicated and respected, I expect NOMAD to become more and more useful.

appendix a: query system upgrade

Query Builder:

In the current implementation, it is not clear that "Filters" et al are actions relevant to specific arrays. Such items should be buttons below the list, not plaintext links above.

Query Results:

Each dimension is indexed with otherwise meaningless integers. Each row has a "record number" and each column a "field index". These values should be clearly separated from the rest of the table. This can be accomplished by listing the column numbers in the table's first row and the row numbers in the first column, neither colorized. Possibly they ought also to be listed in a smaller font size. Then whether or not those indexes are shown can be defined in the "Display Settings" window.

The large gaudy purple arrows can be replaced with the demure grey ones from the browser. Also, sort buttons in a matrix ought to perform a multisort. Currently they do not.

Instead of having the array name on every column, it should be shared between all the fields from itself. One table cell can span as many columns as are from one array and just have the name once, possibly in bold. this will be clearer, less redundant, and a more efficient usage of horizontal space (since the array name is frequently the widest column).

appendix b: parameter system upgrade

Array parameters listed in the selector when setting up an experiment, or filling them in at upload, should have several new characteristics:

Separation of information specific to samples, hybridization, and slide preparation

I am currently considering replacing the ordering of parameters by name with a dual sort, with parameter class preeminent. I believe that this would accomplish the goal, but a little more research is needed.

Pairing of the same attributes for each sample

Lisa has gone ahead and made this change at my request. Instead of sample parameters being named "$sample Sample: $attribute" they will be "$attribute [$sample]", and so will sort together in pairs.

Disambiguation of identically-named parameters by unit type

Lisa had already achieved this when I asked her.

appendix c: references

Users I interviewed were Andrea Barczak, Tanuja Goulet, Christina Lewis, Madeleine Rodriguez, and Dionysos Slaga; I interviewed developers Michael Maibaum, Brian Pulliam, and Lisa Simirenko.