Bioinformatics and Computing Curriculum: A New Model for Interdisciplinary Courses

Jacquelyn S. Fetrow
Departments of Physics and Computer Science
Wake Forest University
Winston-Salem NC 27109
fetrowjs@wfu.edu

David J. John
Department of Computer Science
Wake Forest University
Winston-Salem, NC 27109
djj@wfu.edu

ABSTRACT
An interdisciplinary bioinformatics course has been taught at Wake Forest for three semesters. Undergraduate and graduate students from multiple academic specialties are brought together in a single classroom. In addition to focusing on traditional bioinformatics topics, this course concentrates on interdisciplinary collaboration in the in-class exercises and the research-based course project. A team of faculty from complementary disciplines teach the course. Productive communication is one key goal of this course.

Categories and Subject Descriptors: J.3[Life and Medical Sciences]:Bioinformatics; K.3.2[Computer and Information Science Education]:Bioinformatics, collaboration.

General Terms: Design, experimentation

Keywords: Bioinformatics, computational biology, curriculum, interdisciplinary communication

1. INTRODUCTION
Most college and university courses are taught from a specific disciplinary focus: one professor enters the classroom and teaches more or less from his or her specialization for a semester. While such a model is useful for introducing core concepts, there is a growing need to integrate information from several different disciplines. Many of today’s and tomorrow’s careers demand not only depth of knowledge in a specific area, but also a general sense of vocabulary, tools, and concepts across other disciplines as well. The recent Mars expeditions have been large, and usually successful, collaborations between engineers and planetary scientists [4]. Today’s large-scale biomedical research projects increasingly rely on sophisticated algorithms and advanced computer architectures to organize and analyze data and model biological processes. Turning the vast quantities of information generated by the genome and proteome projects into useful science, biomedical tools and cures is requiring collaborations between computer scientists, information specialists, and biomedical researchers [2, 8]. We are failing our students if we do not involve them in collaborative learning and research.

We present here an interdisciplinary model for the teaching of bioinformatics. While our specific example is focused on an interdisciplinary course in bioinformatics, in which computer and biomedical scientists come together in the same classroom, the principles are not limited to the field of bioinformatics. It could just as easily be applied to computer science and physics in the field of scientific visualization, or computer science and art in the field of multimedia. A major focus in our course is communication: introducing the students to both the necessity and pitfalls of communicating across disciplines. The course outline that we present takes a step beyond the linked courses described by LeBlanc and Dyer [3], and focuses on teamwork and communication.

Why is such a focus needed? Our concept for this course draws from the experience of one of us (Fetrow) in the biotechnology industry. While job candidates and employees who had a degree labeled “bioinformatics” had the vocabulary to work across the disciplines of biochemistry, molecular biology, computer science, and sometimes software engineering, they were shallowly trained in all fields. With shallow knowledge, they did not participate fully, to the extent expected of them, in the solution of complex or difficult tasks that required in depth knowledge. In that setting, the ability to communicate and work in a team with those from other disciplines was essential for successful employees who had the depth of knowledge in skills in a single discipline. Progress in this industry depended upon productive communication between computer scientists, software engineers, and biomedical researchers. This need for teaching and learning effective cross-disciplinary teaching skills has recently been noted by others [1].

An obvious goal of our bioinformatics course is to teach some of the foundations and ongoing issues in the field of bioinformatics. Beyond that, we aim to: 1) teach students vocabulary necessary for communication across the two disciplines; 2) indicate some of the issues that arise in communicating with peers in other disciplines; and 3) teach students the skills necessary to communicate with members of an interdisciplinary team collaborating on a project.

2. COURSE DESCRIPTION
Our bioinformatics course crosses the traditional dis-
disciplinary boundaries between biochemistry, biophysics, biology (biomedical sciences) and computer science. We bring together, in a single classroom, advanced undergraduate and graduate students from the departments of biology, biochemistry, chemistry, physics, computer science, and mathematics. Clearly, we do not require the same prerequisites of all students, but we do require that each student bring an appropriate level of their own disciplinary knowledge to the classroom. Thus, the biomedical students are required to have succeeded in a course either in molecular biology or biochemistry; the computer science students must understand algorithm design and analysis, data structures and program design; and, mathematics students must bring knowledge of abstract modeling, and logical and quantitative problem solving. Each discipline has its own course number associated with its own prerequisites, and each student is required to register for that course number for which he or she is qualified.

The course is organized around three components: lecture, in-class exercises, and a course project. So far the course has been team taught with faculty from complementary disciplines (biochemistry, computer science, and physics).

2.1 Lectures

The lectures during the semester focus on a small number of topics. For each topic, one of the team provides an overview of the biological side, providing background, technical details, ongoing research and unsolved problems. The other follows with an overview of algorithms and heuristics that have been used for the particular problem and a detailed study of one of the algorithms. This combined approach introduces underlying biology to computer science students and the underlying algorithms to the biology students. an understanding of algorithms in teaching students to be “bioinformatics scientists” rather than “bioinformatics technicians” [5]. In the Fall 2004 semester, the four topics were pairwise sequence alignment, protein secondary structure prediction, gene expression, and genome sequencing/gene prediction. The computational methods covered in detail were dynamic programming algorithms, artificial neural networks, statistics and clustering techniques, and hidden Markov models. We discovered that it is critical (as in any course, but even more so here) to limit the number of topics covered.

Accompanying each lecture is a reading assignment and a reading quiz. The reading preparation, strongly encouraged by the reading quiz, improves the overall class discussion. These quizzes are designed to be simple to ensure that the students read the material prior to the beginning of class. In addition, one question in the reading quiz implements the “Just-in-time” concept pioneered by the physicists Novak, Gavrin and Patterson. They are administered electronically via Blackboard. An example reading quiz is shown in Figure 1.

An in-class final exam is also part of the course. The purpose is to give the student the opportunity to review the entire course and to integrate the different ideas. The expectation is that the quality of the answers should reflect the student’s background and intellectual growth during the semester.

2.2 In-class Exercises

The in-class exercises follow the lectures closely and are designed to accomplish the obvious goal of making the lecture topics “real.” Specifically, the exercises are a set of written questions or problems which are solved by the students working in interdisciplinary teams (at least one student from the computational disciplines and at least one from the biological disciplines) and which give the students hands-on experience with the actual biological databases (including both genomic and proteomic), underlying data, and tools. A sample exercise question is shown in Figure 2.

Figure 1: A sample reading quiz for the course.

![Figure 1](image1)

Figure 2: A question from the in-class exercise that focuses on the topic of pairwise sequence alignment.
The exercises force students to confront issues with real-world data and applications. For instance, the students can compare data from different publicly available genomic databases to see the inconsistencies or compare results from different applications and work to understand why they are different. Working in teams is a key component of teaching communication skills. The interdisciplinary teams force the students to learn to use vocabulary in a way that can be understood by their teammates. For the in-class assignments, we expect answers that reflect the interdisciplinary makeup of the group.

These exercises also provide the opportunity for side-by-side comparisons of the disciplinary meanings and answers to specific questions, and to demonstrate how a question can be interpreted very differently by people from different disciplines. During the Fall 2004 laboratory sessions, we had one particularly memorable discussion about the different disciplinary answers to the question, "what is a gap in a sequence alignment?" Such discussions compel the students to confront the issues of group dynamics and communication across disciplines. In the ideal situation, they learn that they need to actually understand what their peers are saying. In Fall 2004, two lecture hours each week were used for lecture and one hour was for in-class exercises. In Fall 2005, we are trying a studio-style course, in which two-1.5 hour time slots each week will be divided into 25 and 50 minute blocks for lecture and in-class exercises respectively.

2.3 Project

The third aspect of the course is the semester-long research-based project. During the project, students work in interdisciplinary teams to create a significant body of work that could not have been fashioned by any one person individually or only people from one discipline. The project gives the course relevancy by involving the students at the frontier of ongoing proteomic research, far beyond what is in the textbooks. In fact, each project assigned so far has been associated with proteomic scientists within the university. The team work forces a level of communication beyond that done in the in-class exercises and allows each person to bring a unique skill set, without which the project would fail. As an example, in the next section, we describe in detail the course project from the Fall 2004 semester. This project-based approach extends the team projects described by Rice et al [7]. Here students must not only solve a biological problem, but must develop an appropriate algorithm and implement it in an application.

The project follows a software engineering protocol (project scope, application design, implementation, testing and validation, documentation and user training) over the course of the semester. This protocol is based on the Waterfall Method which can be found in most introductory software engineering texts. This protocol requires a meaningful document at each step with appropriate disciplinary input from each member of the group. For example, the testing and validation requires the biology students to use the test cases they created in the application design to validate the implementation.

Just as we cannot expect students from across the disciplines to have similar prerequisite knowledge, we likewise cannot expect them to achieve the same goals. The project teams consist exclusively of either graduate or undergraduate students. The expectation from the undergraduate teams is a convincing and working proof of concept. The expectation of the graduate students teams is a reasonably functioning research tool.

3. EXAMPLE PROJECT

In this section, we provide a more detailed description of one course project and indicate issues that might arise and how they might be addressed. In Fall 2004, the project was suggested by colleagues in the Biochemistry Department at Wake Forest University School of Medicine. These researchers had developed a fluorescent probe for identifying all proteins in the cell that were modified to a cysteine sulfenic acid [6]. Proteins are separated on a 2D gel and fluorescent spots identified by mass spectrometry. The output from mass spectrometry is only the mass of the protein and its component peptides. Standard current software matches those masses to proteins in a large database and provides scores for the "matches." Because these researchers were identifying proteins that had been post translationally
modified, the software was not correctly identifying the appropriate proteins in the database. The goal for each team was to create an application which would take as input the data from a mass spectrometry experiment and would produce promising matches from a database, assuming sulfenic acid modification and a specific sequence mass for the fluorescent probe. Figure 3 shows the result from a project that made a query to the new proteomic database.

The project development followed a software engineering protocol, based on the Waterfall Model. According to the protocol, the students were first required to create a Project Scope document in which the students defined the problem, outlined how they would solve it, specified what they would not accomplish, and provided a preliminary time line. The scope document forces the biomedical students to understand the science and technology on which the project is based and to explain it to their computational peers. The group then brainstorm on how best to solve the problem. The Project Scope is developed in two versions, so that we can provide formal feedback early in the process. The second step in the software engineering protocol is the development of the Application Design document. For this document, the biomedical students are responsible for defining specific inputs and outputs, defining and obtaining all testing and validation sets, and walking through at least one example of the process by hand from beginning to end. The computational students are responsible for complete design of the system, including data formats, language for coding, data storage, and algorithm design. After the students have submitted two versions of the Application Design document, they begin the implementation process. The Testing and Validation step is performed by the non-computer science students using the testing data developed earlier. The User Training is part of a presentation made to the entire class.

There are many challenges related to the project, as well as any large scale development assignment. It is important to keep the project focused on a single research goal. It is critical that the software engineering protocol be followed. One of the hardest issues is to prevent computer science students from developing lines of code until the scope and design phases have been completed. The students must keep accurate records of their contributions including content and time, this is essential for determining one student’s grade in relation to the others in the group.

All groups developed a working prototype with a scaled down version of the data. Only the graduate students were required to then produce the full version. The not unexpected problems associated with scaling up from a prototype to a full version were encountered in the Fall 2004 project and required some redesign.

In summary, the project emphasizes the creation of a useful and usable piece of software that could not have been created by any one person or disciplinary group. It forces students to confront issues of interdisciplinary communication beyond what they experience in the regular in-class portion of the course. Communication during the in-class exercises prepares students for collaborative problem solving, but a successful project is only possible if the group truly can work together as researchers with different backgrounds. The project also forces students to deal with the issues of group dynamics, including how they respond if one student does not pull his or her weight. The project is a necessary and meaningful interdisciplinary capstone experience for this course that is not possible in a standard biology, chemistry, computer science or mathematics course.

4. STUDENT CHALLENGES

By and large the students who have enrolled and completed this course have been well prepared in their disciplines and are highly motivated. For most of them, this is their first experience in an interdisciplinary area and their first collaboration with peers in an interdisciplinary arena. For all of them they find the experience initially uncertain—each has a day when they are uncomfortable with the material. Each finds it necessary to talk with others to fill in the gaps of their knowledge and experiences. As the semester progresses, as they learn the vocabulary and become comfortable with the concepts, the students communicate with each other and with the faculty intelligently and effectively. For the faculty one of the challenges is to insure that each student is appropriately challenged individually. Even while presenting material that may be more familiar to one group, that group should be challenged as well.

One concern expressed by most students at the beginning of the semester has been apprehension of collaborating with other students whose focus is in another discipline. Interdisciplinary work is necessary on projects and in-class exercises. This apprehension is quickly replaced by appreciation of the various expertise brought by everyone to the class. By the end of the semester, this initial concern is no longer present. A key indicator of this significantly diminished misgiving is the productive communication among the students.

Each student brings certain academic expertise to the course, e.g. algorithm analysis, biochemical knowledge and techniques, experimental savvy, mathematical modeling, quantitative skills, and software design. As well, each possesses certain voids of knowledge. By the end of the course the voids should be somewhat filled and the expertise should be extensively tapped.

4.1 Bioscience Students

The bioscience students are a mixture of biology, biochemistry, chemistry, and physics students. Each is strong in scientific methodology. All have a good understanding of the central dogma of biochemistry. All can recognize protein and nucleic acid sequences and structures. Most have no experience in traditional algorithmic problem solving, though most have developed laboratory processes which has made them successful in their respective areas.

4.2 Computer Science Students

Computer science students are expected to bring into the class a strong background in algorithmic problem solving. Specifically they should have completed courses in data structures and algorithms. Ideally they have also completed a data base course. The computer science graduate students have excellent preparations in computer science and strong backgrounds in mathematics. Very few of them have had courses in Biology, Chemistry or Biochemistry beyond the high school level. Most have been exposed to some software engineering concepts, but not in a large team-based development environment.

On a daily basis, these students contribute with their computationally based experiences and also with their naive (but often very good) questions related to the biological as-
pects of the course. Within a few weeks they learn the basics of biochemistry needed for this course and can communicate effectively with their biological counterparts.

The software engineering process used to manage the project is new to most of the computer science students. They want to rush in and start programming but soon realize that everyone is needed to carefully define the requirements and overall design. They concretely learn the differences between validation and verification during the constant testing phases.

4.3 Mathematics Students

Generally these students are very proficient problem solvers and bring a vast collection of mathematical techniques to the course. Almost all have had some exposure to algorithmic problem solving. As with the computer science students, very few have any in-depth knowledge in biology.

5. FACULTY CHALLENGES

One of the faculty (Fetrow) is an expert in structural and computational biology, but the remainder of the faculty associated with this course are newcomers with various backgrounds. As is often the case with new courses in new areas it is necessary to entice existing faculty to participate and collaborate as is appropriate. Recruiting faculty is easiest when there is an associated research payoff, e.g. the problems in bioinformatics can become some part of their research interests. Given the material of the course, for all faculty there have been times when they are outside their comfort zones.

Similar to the experience of Rice et al [7], we found that the team teaching approach has been most effective in allowing faculty to complement each other which allows for a higher level of expertise than any one could bring alone to the course. The team approach is expensive for the university but is consistent with the goal of collaborative research and learning found currently in most universities, including Wake Forest.

6. ADMINISTRATIVE CHALLENGES

As often happens with relatively new courses, it has been a challenge to find a text that is appropriate to our mix of students. Often the texts focus on students with extensive backgrounds in one area, and these generally will not serve this interdisciplinary audience very well. An administrative headache is the numerous course numbers that must be attached to this one course, e.g. CSC363, PHY381 and BIO222, because of the differing prerequisites and course requirements.

7. CONCLUSIONS

The success of this course can be measured in several different anecdotal ways. First, each time the course has been offered, the graduate students have presented their project in the annual Research Day sponsored by the graduate school. Of course, they were strongly encouraged to do so. Second, in Spring 2005 two computer science graduate students graduated following completion of a bioinformatics MS thesis. Both had participated in the first offering of this course. Third, in the Computer Science Department this year two graduate students are working on research problems in bioinformatics that should lead to master’s theses. Fourth, one of the computer science graduate students is applying to PhD programs in computational biology. Fifth, from the course this year one of the Mathematics Graduate students is planning his thesis related to bioinformatics. Sixth, despite the course’s reputation as demanding a great deal of work, the registrations for the next offering are strong, especially with biology undergraduates. Seventh, two computer science faculty and one mathematics faculty have become extensively involved in computational biology research, none of which had previous backgrounds in computational biology.

The easiest course of action would be for the faculty of the university to offer a separate course in bioinformatics in each department. Students and faculty in each department would be more comfortable with the idea that the course is one of "theirs". However, we have discovered that the best learning is most uncomfortable. This interdisciplinary approach has been most successful in getting students and faculty out of their comfortable disciplinary boxes and into the intellectually challenging area of interdisciplinary learning and research.

8. ACKNOWLEDGMENTS

Jacque Fetrow thanks Susan Baxter and Steph Reis-inger for long discussions which helped formulate the ideas for this course. Both authors thank the students who have worked hard and successfully in the courses, and who have allowed us to share their experiences and work. Both authors thank Professors Allen, Burg, Fahrbach, Miller, Nelson and Poole who have been directly or indirectly involved with the course. The author thank the reviewers for their excellent suggestions for improvement of this manuscript.

9. REFERENCES

[4] PBS. Mars: Dead or alive. DVD.