Two interactive Bioinformatics courses at the Bielefeld University Bioinformatics Server

Alexander Sczyrba, Susanne Konermann and Robert Giegerich

Submitted: 22nd October 2007; Received (in revised form): 17th December 2007

Abstract

Conferences in computational biology continue to provide tutorials on classical and new methods in the field. This can be taken as an indicator that education is still a bottleneck in our field’s process of becoming an established scientific discipline. Bielefeld University has been one of the early providers of bioinformatics education, both locally and via the internet. The Bielefeld Bioinformatics Server (BiBiServ) offers a variety of older and new materials. Here, we report on two online courses made available recently, one introductory and one on the advanced level: (i) SADR: Sequence Analysis with Distributed Resources (http://bibiserv.techfak.uni-bielefeld.de/sadr/) and (ii) ADP: Algebraic Dynamic Programming in Bioinformatics (http://bibiserv.techfak.uni-bielefeld.de/dpcourse/).

Keywords: Bioinformatics education; online courses; sequence analysis; dynamic programming

INTRODUCTION

Conferences in computational biology continue to provide tutorials on classical and new methods in the field. This holds for large international conferences like ISMB (http://www.iscb.org/ismbecb2007/), ECCB (http://www.iscb.org/ismbecb2007/), CSB (http://lifesciencesociety.org/CSB2007/), PSB (http://psb.stanford.edu/psb07/), APBC (http://www.cs.hku.hk/apbc2007/), as well as for some conferences of national or specialized communities such as GCB (http://www.gcb2006.de/) or NETTABI (http://www.nettab.org/2007/). This can be taken as an indicator that education is still a bottleneck in our field’s process of becoming an established scientific discipline. There is a continuous stream of scientists entering the field from neighboring disciplines such as molecular biology and biomedicine. For student education, relatively few places offer fully developed curricula on the BSc, MSc, and PhD level.

For computational biologists, it is a common pattern of their work to use computational tools and data sources via the Internet. Hence, for students and instructors, it should be equally attractive to use WWW-based educational resources, given that they are high quality and well maintained. Such courses can adapt easily to changes in the field, while textbooks are more suitable to the presentation of classical methods.

Bielefeld University has been one of the early providers of bioinformatics education, both locally [1] and via the Internet [2]. In the ‘Education Department’ of the Bielefeld Bioinformatics Server (BiBiServ), a variety of older and new materials...
is offered. Here, we report on two online courses made available recently, one introductory and one on the advanced level:

(1) Sequence Analysis with Distributed Resources (SADR)  http://bibiserv.techfak.uni-bielefeld.de/sadr/

(2) Algebraic Dynamic Programming in Bioinformatics (ADP) http://bibiserv.techfak.uni-bielefeld.de/dpcourse/

Both topics have once been presented as ISMB tutorials, SADR in 1998 [3], ADP in 2000 [4], but the materials now available have been revised completely. Significant advances in the field have been taken into account. Both courses are regularly used in our local bioinformatics curricula at Bielefeld University, which implies that they will remain up-to-date in the foreseeable future.

Many universities make lecture notes available on their web pages. In contrast to these static documents, our courses are designed for interactive usage and the possibility to play with and thereby learn from examples in a straightforward way. While similar courses exist for introduction in sequence analysis [5–8], algebraic dynamic programming is a new technique, for which the course at BiBiServ is the only one available.

THE COURSES
Sequence analysis with distributed resources

Course content
Researchers in molecular biology analyze genetic data resulting from sequencing projects in their daily work. Many of these tasks can easily be solved using Internet-based tools via the World Wide Web, without having to consider technical problems like local installation, maintenance or financial aspects. These tools, offered by different institutions across the world, are indispensable for daily sequence analysis work. Thus, often questions arise like: Where do I find references to published work? What databases are available and what can they be used for? What is the most adequate tool for solving problems like sequence alignment, database searches, primer design or RNA structure prediction and comparison? How do I build a workflow for data analysis?

Teaching students how to solve these problems using web-based sequence analysis resources, the WWW itself is the medium of choice. The web based course SADR assumes only minimal background in bioinformatics; it does not primarily explain the algorithms behind the tools. Instead, it emphasizes practical tool usage in connection with the biological considerations that guide the process of sequence analysis.

Course level
The course is aimed at undergraduate bioinformatics students in their second year as a complement to the lectures introducing the theoretical concepts behind the analysis tools, and also to biology students as an introduction to bioinformatics databases and tools. Users of this course should be familiar with UNIX, the Java programming language and have basic knowledge about molecular biology.

Course structure
The course starts with an overview of the most widely used Databases for bibliographic searches (PubMed [9]), nucleotide and protein sequence retrieval (Genbank, EMBL, DDBJ [10], UniProt [11], InterPro [12], PDB [13]), genomic information for specific organisms (Wormbase [14], FlyBase [15]) and taxonomy data (NCBI Taxonomy [9], NEWT [16]).

To make use of the databases, the information of interest has to be parsed from flat files and converted in different formats that can then be used by sequence analysis programs. As each of these databases has its own Data Format, the second chapter introduces some of these formats, which will be processed by standard Unix tools to answer simple questions about the file contents. For more sophisticated questions, the students have to write XSLT scripts and Java programs to process XML formatted database entries.

The chapter about Sequence Alignments introduces local and global pairwise alignment tools as well as multiple sequence alignments (ClustalW [17], DCA [18] and DIALIGN [19]). Results from different tools are compared with each other and also to the BAliBASE database [20], a database of manually refined multiple sequence alignments specifically designed for the evaluation and comparison of multiple sequence alignment programs.

Database Searches in sequence databases are performed using BLAST [21] and FASTA [22]. Here, students learn which parameters to choose for
different scenarios. The results of different search programs are compared and discussed. As examples for gene discovery and identification of alternative transcripts, ESTs are mapped to genomic sequences using e2g [23].

The chapter about RNA Secondary Structure introduces several tools for secondary structure prediction of RNA (RNAfold [24], RNAspecies [25]), comparative structure prediction (RNAalifold [26], Foldalign [27]) and RNA structure comparison (RNAforester [28]).

In the last chapter, we give an introduction into the currently developing field of Web Services, and how they can be used to build workflows using resources distributed across the world (Figure 1). BioDOM, a Java library to build XML files following the HOBIT format descriptions [29], is used to provide functions to convert native non-XML output of various bioinformatics tools to XML formats that can be validated with the HOBIT XML schemas.

Algebraic dynamic programming in Bioinformatics

Course content
While the SADR course provides interactive material on topics also covered in many bioinformatics textbooks, the ADP course covers advanced methodology [30, 31] on which no textbooks are available yet.

Traditionally, the design of successful DP recurrences has been a matter of experience, talent and luck. Essential ideas and implementation details were closely entangled, and debugging code was tedious. ADP turns dynamic programming into a systematic effort—in a rigorous mathematical framework and on a high level of abstraction. The essential ideas of a DP algorithm are expressed by a two-part specification, consisting of an evaluation algebra and a tree grammar. From these two parts, correct and efficient code is produced automatically, either in C or in Haskell.

This obviates the need to explicitly develop, code and debug dynamic programming recurrences. As such recurrences are commonly seen as the heart of each DP algorithm, the new approach presents a revolution in the way we reason about and solve dynamic programming problems.

Course level
In the Bielefeld curricula, the course is taught to advanced students in second year of their MSc or the first year of their PhD studies. They are expected to bring along some prior acquaintance with dynamic programming in its classical form, and the fallacies associated with it.

Course structure
Overall, the course is structured into a tutorial text and interactive application pages (Figure 2). While the first part of the course is mainly reading, the later part mainly works by inviting the students to practise the technique by modifying existing examples, or writing their own.

The ‘Initial Reading’ covers the small but essential theory behind the ADP approach. It starts with a gently moving introduction, using a stroll through a shopping mall as the first example of a combinatorial optimization problem. It also holds some entertaining elements, which we do not report here. The text then abstracts the notions of problem universe, solution candidates, scoring and so on, and transfers them to applications in biosequence analysis.

Thereafter, the mathematical framework of ADP is introduced, starting with gentle motivation and leading to rigorous mathematical definitions of the central notions—evaluation algebras, tree grammars, yield parsers and algebra products. The introduction ends with pointers to the literature, both on the general method and on bioinformatics tools that have already been created with it.

The experimental part begins with inviting the student to choose from a rich supply of (mostly classical) algorithms that have been reformulated within the ADP framework. Examples include the Needleman–Wunsch [32], Gotoh [33], Smith–Waterman [34], Nussinov [35], Zuker [36] and Wuchty [37] algorithms. We also provide some other examples more popular in the computer science literature, such as matrix chain multiplication. A student’s favourite is El Mamun’s Caravan, supposedly the oldest dynamic programming problem in the world.

The constituents of each example can be combined in various ways, tested and modified. Towards the end, the student is more and more pursuing his or her own investigation, while the tutorial only gives suggestions on what to try first, or where to go back when something does not work.

Presentation techniques
Both courses provide theoretical background information as well as hands-on experience. The theory is
**Figure 1:** This figure shows an advanced chapter of the SADR course. General concepts are explained in the main window, here asynchronity in WebServices. After an introduction to the topic, example code is shown to give students an idea of the structure of the program they have to write. The exercises (accessible through the red 'Exercise' button) provide code of working examples and instructions on how to use them. In the exercise on the left, the student has to combine two web services, which retrieve a sequence from the database via SOAPDB and transform the FASTA output to SequenceML XML output using the BioDOM web service.
Figure 2: This figure shows one of the content pages (large panel) of the application examples. The page shows the general course navigation bar on the left, and the ADP source code for sequence alignment. Specific parts of the code can be popped-up and pinned to the screen via the blue and red buttons. The pop-up ‘How To’ gives some suggestions for experimenting with this example. The pop-up ‘Yield Grammar’ shows the grammar which describes the Needleman–Wunsch algorithm—this ASCII-notation of a tree grammar is ADP’s substitute of the classical dynamic programming recurrences. (Note the absence of subscripts, and hence, of subscript errors.) Two evaluation algebras are shown as pop-ups. One implements the familiar unit distance. The other is unusual as its ‘scores’ are copies of input strings, padded with gaps—this is ADP’s way to present solutions without the need to program backtracing routines. In the panel labelled ‘Online’, the grammar can be called with a variety of different algebras. As shown, the student has selected the two algebras called ‘unitDistance’ and ‘prettyprint’ as the arguments of an algebra product (denoted by ‘***’), and has chosen the poly-peptides ‘darling’ and ‘airline’ for alignment. After hitting ‘submit’, the result shows up in the ‘Output’ field—two co-optimal alignments of distance score 3.
presented in small screen-readable portions in the main window. After an introduction to the topic, example code is shown to give students an idea of the structure of the program they have to write.

Exercises or demos open in smaller windows. That way all the necessary information can be arranged on the screen side-by-side to the introductory text, allowing the transfer of theoretical background knowledge to hands-on experience. Online- (input forms), Code- (code fragments) or Howto-windows (hints for experimenting) shall motivate students to modify the examples and thereby understand the topics in more detail. Tooltips give either further background information or motivating and diverting comments.

**Key Point**
- Education is still a bottleneck in bioinformatics' process of becoming an established scientific discipline. The Bielefeld Bioinformatics Server makes two online courses available.
- The SADR course is a renovation of a course that once already had a large user ship.
- The ADP course, now including the advances made in the field since 2000, is announced here for the first time.
- We hope that both courses find widespread use, and colleagues and students will provide us with a continuous flow of comments and suggestions leading to further improvements.

**Acknowledgements**
We acknowledge contributions to the courses by Sven Hartmeier, Jan Krüger, Kai Löwenthal, Henning Mersch, Jens Reeder, and Peter Steffen. This work was supported by grant BIZ 7 from the DFG-Initiative Bioinformatics.

**References**


