BIOINFORMATICS

Vol. 16 no. 2 2000
Pages 101–103

Darwin v. 2.0: an interpreted computer language for the biosciences

G. H. Gonnet, M. T. Hallett*, C. Korostensky and L. Bernardin

Department of Computer Science, ETH Zürich, Zürich, Switzerland

Received on June 16, 1999; revised on August 31, 1999; accepted on September 14, 1999

Abstract

Motivation: *We announce the availability of the second release of Darwin v. 2.0, an interpreted computer language especially tailored to researchers in the biosciences. The system is a general tool applicable to a wide range of problems.*

Results: *This second release improves Darwin version 1.6 in several ways: it now contains (1) a larger set of libraries touching most of the classical problems from computational biology (pairwise alignment, all versus all alignments, tree construction, multiple sequence alignment), (2) an expanded set of general purpose algorithms (search algorithms for discrete problems, matrix decomposition routines, complex/long integer arithmetic operations), (3) an improved language with a cleaner syntax, (4) better on-line help, and (5) a number of fixes to user-reported bugs.*

Availability: *Darwin is made available for most operating systems free of charge from the Computational Biochemistry Research Group (CBRG), reachable at http://cbrg.inf.ethz.ch.*

Contact: *darwin@inf.ethz.ch*

Motivation

Darwin is an easy to use interpreted computer language especially tailored to research in the biosciences. Its purpose is to serve as a *biochemists' workbench* where researchers can explore molecular sequence data quickly and easily. The Darwin project began in 1991 and reflects much of the research done in the CBRG (Computational Biochemistry Research Group) at the ETH-Zurich. Broadly speaking, it consists of two parts: the *libraries* and the *kernel*.

The *libraries* correspond closely to what one expects from a software package: a pre-defined set of functions offered by the system. The libraries reflect current and past trends in our research efforts but also incorporate many algorithms from the literature, particularly those related to sequence comparison, phylogenetic tree construction, multiple sequence alignment and secondary structure prediction. The libraries themselves are written in the Darwin language and are therefore easy to read even for novice programmers. We briefly describe the current contents of the libraries in the next section.

The *kernel* of Darwin is responsible for the lower-level operations in the system: executing commands and libraries, memory management, input/output, communication with the operating system, load balancing, etc. Darwin itself is written in C although this source code is not made publically available. The kernel also contains *critical* routines, that is, routines which must be performed efficiently due to the number of times they are called or the complexity of the routines themselves. These include (but are not limited to) routines for *pairwise alignment*, *all versus all* alignments, and *tree construction*. Although the kernel is not modifiable, one can execute *native code* (that is, user designed code written in C, Java, etc.) from within Darwin.

Since Darwin is a computer language, it allows one to go beyond the fixed set of routines offered in the kernel and library. The language itself is a high-level interpreted language equipped with `lists`, `sets`, general data structures, and a robust collection of basic mathematical functions allowing the user to quickly prototype new ideas. Darwin programs are relatively fast even when compared with optimized C code. A moderately experienced user of the system will be able to modify existing libraries (written in the Darwin language) when necessary or create new libraries appropriate to whichever problem they are currently exploring. Below is a small example of a Darwin session:

```
unix:  darwin
  Darwin: Sequence Searching Facility
  Version 2.0, August 1998
    (c) E.T.H. Zurich
 DB:=ReadDb('SwissProt37'):
  Peptide file(SwissProt37(54714687),
  77977 entries, 28268293 aminoacids)
 printf( '\nIdentification: %s',
               Entry( 1 )['ID']);
  Identification: 100K_RAT
 printf( 'Accession Number: %s',
               Entry(1)['AC'] );
  Accession Number: Q62671;
```

*To whom correspondence should be addressed.

© Oxford University Press 2000

```
 CreateDayMatrices():
 res := AlignOneAll( 1, DB, DM, 120 ):
 length( res );
  19
 hisofar := 0:  index := 0:
 for i from 2 to length( res ) do
     if (res[i,Sim]  hisofar) then
           hisofar := res[i,Sim]:
           index := i:
           fi:
     od:
 printf('\nMost similar is %d with
      score %5.2f', index, hisofar );
Most similar is 11 with score 301.13
```

The above program loads the SwissProt v. 37 dataset (`ReadDb`), then prints out the identification and accession tags for the first entry. After creating the GCB extended Dayhoff matrices, the first entry is compared against all other entries in SwissProt (`AlignOneAll`). In this example, the alignment is performed at a PAM distance of 250 (variable `DM`) and all signficant matches are stored in the variable `res`. A significant match here is defined as any match with a similarity score greater than or equal to 120.[1] We search through the 19 such matches for the alignment which induced the highest similarity score.

Although large, the libraries distributed with Darwin are far from complete (computational biology travels simply too fast to make 'keeping up' viable). Users are invited to submit new libraries for inclusion in future releases of the system.

Results

Combining algorithms both from the literature and research local to the CBRG, our system allows a flexibility that no previous system has offered. This flexibility is an absolute necessity as we enter an age where the analysis of complete genomes will be commonplace. We believe that the power of Darwin remains largely untapped although over 400 research groups have experimented with our software.

Below is a brief description of the contents of the Darwin language, libraries and built-in routines. We note, however, that this description is not complete; there are other libraries for many discrete and continuous mathematical optimization problems plus other smaller tools for manipulating sequence data. The manual contains more information on these topics.

[1] This is a maximum likelihood log-odds score which can be intepreted as meaning that it is $10^{\frac{120}{10}}$ more likely the sequences evolved from a common ancestor than a random alignment.

Basic mathematical operations

The system includes operations for sets, lists, trigometric functions, combinatorial graphs, long integers, real and complex numbers, likelihood/probability calculations, matrices (including LLL decompositions, Gaussian eliminations, Givens eliminations, singular value decompositions, eigenvalue/eigenvector computation, Gram–Schmidt decompositions and linear regressions), control of input/output, and interacting with the operating system.

Pairwise alignment

Darwin comes equipped with routines to align peptide sequences versus peptide sequences, or nucleotide versus peptide sequences (Knecht, 1995). The alignment routines are based on the *full dynamic programming approach* using the GCB matrices. (See Gonnet *et al.*, 1992; Gusfield, 1997; Gonnet and Hallett, 2000.) The system can also perform parametric alignments which seek to find the PAM distance which maximizes similarity score (Gusfield, 1997; Gonnet and Hallett, 2000). Local alignments, global alignments, and cost-free 'end gap' alignments are all possible.

Dataset conversions

The system includes routines for converting raw SwissProt (Bairoch and Apweiler, 1999) or EMBL (Stoesser *et al.*, 1999) flat-files to the Darwin format. The libraries also include a *parser shell* which can be easily modified to parse any flat-file.

All versus all routines

One of the most useful features of Darwin is its ability to perform large-scale comparisons of genomes; that is, the alignment of every sequence in a dataset with every other sequence in the dataset. To this end, Darwin automatically generates a patricia tree[2] when a sequence dataset is loaded and provides various built-in (i.e. located in the kernel) functions for performing fast alignments. Also, Darwin is capable of distributing a large set of jobs over an intranet, can control the computation of these jobs on the foreign machines, and collect the results.

Peptide transition matrices

The following peptide transition matrices are built into Darwin: Dayhoff, GCB (Gonnet *et al.*, 1992), BLOSUM (Henikoff and Henikoff, 1992), amongst others. New matrices can be computed from sample data. There are also routines for converting PAM distance to and from percent identity.

[2] A patricia tree is a close sibling to the *suffix tree*, the more common data-structure in the literature.

Protein identification via peptide mass

Darwin contains routines for protein identification by aligning the masses of small collections of peptides after N- or C-terminal digestion against either a nucleotide and peptide dataset (Korostensky *et al.*, 1998).

Phylogenetic tree and multiple sequence alignment construction

Historically, tree construction in Darwin has been based on distance matrices and the system contains various related routines: tree topology construction algorithms [Neighbour joining (Saitou and Nei, 1987), clustering methods (Hillis *et al.*, 1996), amongst others], least squares fits to tree topologies, and local optimization routines. There are now routines for tree construction based on *circular orders*, a new method developed in Gonnet and Korostensky (2000). Multiple sequence alignments (Gonnet and Benner, 1996) are created relative to a phylogenetic tree and the system includes several methods for scoring the quality of the alignment, including a novel method developed in Gonnet and Korostensky (2000).

Statistics and visualization

This system includes routines for drawing histograms, dot plots and bar graphs. The system can also draw unrooted trees, rooted trees, split trees, and combinatorial graphs. There are a large number of routines for producing random permutations, combinations, distributions and specific biological objects such as sequences, trees and multiple sequence alignments.

A manual is now available (Gonnet and Hallett, 2000) which describes the Darwin language and all of the basic functionality of the language, including the basic commands, constructors, data types, built-in data structures and descriptors for all library functions. Darwin v. 1 suffered from a somewhat scattered and nonintuitive naming scheme for its predefined functions. In order to make Darwin more usable, we have adopted a standardized naming convention (included in the manual). Furthermore, a substantial subset of the manual is available via on-line help from within Darwin and the remainder is available via the WWW (CBRG, 1999). Lastly, a large number of bugs reported by our user base have been fixed.

Availability and contact

Darwin is available free of charge from our WWW server http://cbrg.inf.ethz.ch or via email at darwin@inf.ethz.ch. Interested users are asked to fill out a short form indicating which platform(s)[3] are desired. The system will be e-mailed shortly after we receive your signed document.

References

Bairoch,A. and Apweiler,R. (1999) The SWISS-PROT protein sequence data bank and its supplement TrEMBL in 1999. *Nucl. Acids Res.*, **27**, 49–54.

Computational Biochemistry Research Group (CBRG) (1999) http://cbrg.inf.ethz.ch.

Gonnet,G.H. and Benner,S.A. (1996) Probabilistic ancestral sequences and Multiple alignments. In *Proceedings of Fifth Scandinavian Workshop on Algorithm Theory (1996).* Reykjavik, pp. 380–391. LNCS 1097, Springer, Berlin

Gonnet,G.H. and Hallett,M.T. (1999) *Darwin: A User Manual.* Book, available at http://cbrg.inf.ethz.ch.

Gonnet,G.H. and Korostensky,C. (2000) Evaluation measures of multiple sequence alignments. *J. Comput. Biochem.*, accepted.

Gonnet,G.H., Cohen,M. and Benner,S. (1992) Exhaustive matching of the entire protein sequence database. *Science*, **256**, 1443–1445.

Gusfield,D. (1997) *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, Cambridge.

Henikoff,S. and Henikoff,J.G. (1992) Amino acid substitution matrices from protein blocks. *Proc. Natl Acad. Sci. USA*, **89**, 915–919.

Hillis,D.M. *et al.* (1996) *Molecular Systematics*. Sinauer Assoc. Inc, Sunderland, Massachusetts, USA.

Knecht,L. (1995) Pairwise alignment with scoring on tuples. In Galil,Z. and Ukkonen,E. (eds), *Proceedings of Combinatorial Pattern Matching (CPM '95)* Vol. 937, Springer.

Korostensky,C. *et al.* (1998) Identification of proteins in sequence databases using peptides with ragged n- or c-termini generated by sequential endo- and exopeptidase digestions. *Electrophoresis*, **19**, 1933–1940.

Korostensky,C. and Gonnet,G.H. (1999) Near optimal multiple sequence alignments using a travelling salesman problem approach. *Proc. 6th Annual String Processing and Information Retrieval (SPIRE'99),* IEEE Computer Society, September, Cancun, Mexico, pp. 105–114.

Saitou,N. and Nei,M. (1987) The neighbour-joining method: a new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.*, **4**, 406–425.

Stoesser,G. *et al.* (1999) The EMBL nucleotide sequence database. *Nucl. Acids Res.*, **27**, 18–24.

[3]Darwin is available on the following platforms: DEC Alpha/Digital Unix 4.0, SGI Irix 6.x, Sun Sparc Solaris 2.5 and up, HP-UX 10.x, Linux 2.x, Windows '95, Windows NT.