XPARAL
Parametric Sequence Aligment

University Of California Davis
Computer Science

October 13, 1998

Contents

1 Introduction to Parametric Alignment

The optimal alignment between two DNA or amino acid sequences for a given set of weights is computed by classical dynamic programming techniques. However, there is often considerable disagreement about how to weight matches, mismatches, insertions/deletions (indels) and gaps. Parametric Sequence Alignment is the problem of computing the optimal valued alignment between two strings as a function of variable weights for matches, mismatches, spaces and gaps. The goal is to partition the parameter space into regions (which are necessarily convex) such that in each region any alignment that is optimal for some choice of parameters inside the region is optimal throughout that entire region and nowhere else. *XPARAL* is a program that efficiently finds and displays that convex parametric decomposition. It explicitly maps out the full dependence of the optimal alignment on the parameter choices.

An alignment A of two strings S and S' of lengths n and m respectively is obtained by introducing spaces into, or at the ends of, the two strings such that the lengths of the two resulting strings are identical, and then placing these two resultant strings one upon the other subject to the constraint that no column contains two spaces. Any column that contains two identical characters is called a match. Any column that contains two dissimilar characters is called a mismatch and any column that contains a space will be referred to as a space or indel (an abbreviation of insertion/deletion). A series of one or more contiguous spaces in the same string will be referred to as gap. Our use of the term "gap" may be a bit different than typically used in biological literature, where "gap" is sometimes used in place of what we call a "space."

The parametric value of an alignment A may be characterized by the number of matches, mismatches, spaces and gaps it contains. We denote these quantities by m, ms, s, and g respectively. We let w,x,y,z be variables that take on non-negative values and denote variable weights or penalties. Then the value of alignment A as a function of w,x,y, and z is $wm - xms - ys - zg$, which is a linear function of w,x,y and z. (Actually, there are only three independent variables since we could divide the expression by w, say, and still preserve the relative order of the value of all the alignments. We use four variables for convenience.) For fixed values of w, x, y and z, an optimal (maximum value) alignment of two strings can be found by dynamic programming in $O(nm)$ time. Parametric alignment is the problem of solving the optimal alignment problem for all choices of values for w,x, y and z.

XPARAL solves the parametric alignment problem, but for only two choices of parameters at a time. This is because it is easier to view two dimensional figures than three dimensional figures. The user is asked to choose which term (matches, mismatches, indels or gaps) to multiply by x and which by y, and to choose constant multipliers for the two remaining terms. For example, the user might choose the objective function

MAX c1(#matches) - x(#mismatches) -y(#indels) - 0(#gaps)

This gives a constant weight $c1$ to each match, a variable penalty x to each mismatch, a variable penalty y to each indel and a penalty of zero to each gap.

Or the user might choose:

MAX c1(#matches) - c2(#mismatches) - x(#indels) - y(#gaps)

Which varies the gap initiation penalty y against the gap extension penalty x. *XPARAL* allows the user to specify any one of the six possible ways of choosing the two terms to parameterize.

Given two strings, and a choice of the parametric function, *XPARAL* will then solve for the optimal alignment as a function of x and y. The output is a decomposition of the x, y space into convex polygons which have the following property: An alignment that is optimal for at least one x,y point in the interior of a polygon is optimal for all points in that polygon and nowhere else. Hence this polygonal decomposition completely characterizes the dependence of the optimal alignment on the choices of values for x,y. For each polygon in the decomposition, *XPARAL* displays one alignment that is optimal for that polygon. Note that there may be other alignments that are optimal in the polygon, but they too will be optimal for every point in the polygon and nowhere else. Hence the polygonal decomposition is a function of the strings and the objective function, but not of the particular alignments chosen for display.

XPARAL finds each polygon of the decomposition in $O(R + E)$ time, where R is the number of polygons in the decomposition, and E is the time to compute a single alignment when x and y have fixed values. For all the allowed choices in *XPARAL*, R and E are at most nm, hence the time to find each polygon is proportional to just a single alignment computation! This bound, however, is amortized (averaged) over all the computations and the first polygons take much more time to find than that later ones.

2 Program Controls

The main *XPARAL* window (boxed) is used to display the parametric decomposition. It should be maximized to best fit with the vertical text.

XPARAL finds the parametric decomposition of two strings which are displayed just above the main window. The program is initiated with two default strings, both aligned to the left margin without inserted spaces. These input strings can be changed by the user as described below. As *XPARAL* runs, it finds and displays each successive polygon and changes the displayed alignment of the two stings. The alignment then shown is an optimal alignment for the most recently found polygon, which is displayed darker than the other polygons. In the displayed optimal alignment, matches are outlined in color and with a carrot below each match. Once the full decomposition is finished, the user can explore it by clicking the mouse in any polygon. The optimal alignment for that polygon is then displayed.

Below the two strings the word VAR is written. Before the parametric alignment is computed, what follows VAR is a description of the parameter choice currently in effect. Whenever an alignment is computed for a polygon, that description is changed to show the number of matches, mismatches and indels in the current alignment, and the number of gaps if an objective function using gaps has been chosen by the user.

It is possible to use different alignment types, such as global, local, end gaps free, or substring. An OPT window displays the currently selected alignment model.

The chosen boundaries for x and y, and the chosen constant(s) $c1$ and $c2$ (when gaps have been chosen) are also displayed. If weight tables are used for the alignment of two sequences the name of the weighting scheme is also displayed.

The main *XPARAL* menu contains the following selections:

FILE This is the main file menu.

- **Load Aligned Strings** This menu option pulls up a dialog box in which you can input a stored alignment. You must know the names of the two aligned sequences in addition to the name of the file.
- **Load String** Loads a text file as a single string, it can be loaded as either string 1 or 2. The file is loaded in as raw text.
- **Save String** Saves a single string as a text file, either string 1 or 2 can be saved. A dialog box will appear prompting for the name of the file.
- **Reverse String** Reverses either of the strings already inputted so that it appears backwards.
- **Save All Polygons** Saves all of the polygons in the decomposition. You must open a logfile before this operation can be performed.
- **Save Min Distance Polygon** Saves the polygon of minimum distace from an input alignment. You must open a logfile before this operation can be performed.
- **Save Dark Polygon** Saves a single selected polygon (appears dark). You must open a logfile before this operation can be performed.
- **Save Polygon Alignments** Saves the alignments for a polygon but not the geometry. You must open a logfile before this operation can be performed.
- **New Log File** A dialog box will appear prompting for the name of a log file in which alignment information can be saved. This must be done before any polygons can be saved.

- **Quit** Exits the program.

ALIGNMENT This menu allows the user to select the alignment model. The four primary schemes are as follows:

- **Global** (Needleman-Wunsch style) Global alignment finds the best alignment between the entirety of the two input sequences.
- **End Gaps Free** This alignment scheme causes end gaps to have no penalty. This will favor optimal alignments between the beginning of sequence 1 and a end of sequence 2 and vise versa. Free spaces are indicated by `~`, while spaces that are scored are indicated by `-`. When local alignment is chosen, the optimal substrings are enclosed in brackets `[ ]`.
- **Local** (Smith-Waterman style) This alignment sceme will find the maximum optimal alignment amongst all combinations of substrings from sequence 1 and 2. Free spaces are indicated by `~`, while spaces that are scored are indicated by `-`. When local alignment is chosen, the optimal substrings are enclosed in brackets `[ ]`.
- **1st as a Substring of 2nd** This will find the optimal alignment amongst all of the possible substrings of sequence 1 with the entirety of Sequence 2. Free spaces are indicated by `~`, while spaces that are scored are indicated by `-`. When local alignment is chosen, the optimal substrings are enclosed in brackets `[ ]`.

OPT FUNCTION This menu allows the user to select the alignment optimization function. The choices for parameterization include all ways of picking two terms to parameterize from the four terms: match, mismatch, indel, gap. For convenience (and efficiency of the program), the cases when gaps (but not indels) are given a zero penalty are listed separately. The following equations are supported where a parameterized variable appears as the name of an axis, and a constant appears either as $c1$ or $c2$. These constants are by default set to 1.0, but can be changed by the user.

- **(#matches)-X*(#mismatches)-Y*(#indels)**
- **X*(#matches)-(#mismatches)-Y*(#indels)**
- **X*(#matches)-Y*(#mismatches)-(#indels)**

- **(#matches)-(#mismatches)-X*(#indels)-Y*(#gaps)**
- **(#matches)-X*(#mismatches)-(#indels)-Y*(#gaps)**
- **(#matches)-X*(#mismatches)-Y*(#indels)-(#gaps)**
- **X*(#matches)-(#mismatches)-(#indels)-Y*(#gaps)**
- **X*(#matches)-(#mismatches)-Y*(#indels)-(#gaps)**
- **X*(#matches)-Y*(#mismatches)-(#indels)-(#gaps)**

CONSTANTS This menu allows the user to set the values for constants. Normally it should be selected after the optimization function has been chosen.

RANGES This menu allows the user to set the ranges for the parameters.

GAP TYPE This menu allows the user to select the gap model from the three that are currently available. *XPARAL* will use different algorithms and datastructured based on the chosen gap type. Convex gap models will run slower than affine and currently they can only be used with global alignments.

- **g(x)=x** Affine gap model where the cost to extend the gap is always 1.
- **g(x)=log(x)+1.0** Convex gap model, log version, where the total gap cost is $gapinit + g(x)$. Note the added 1.0, this insures that a gap of size 2 is always more expensive than two gaps of size 1.
- **g(x)=log(x + 1.0)** Convex gap model, log version, where the total gap cost is $gapinit + g(x)$. Note the added 1.0, again this insures that a gap of size 2 is always more expensive than two gaps of size 1.
- **g(x)= root(x)** Convex gap model, root version, where the total gap cost is $gapinit + g(x)$.
- **g(x)= root(x)+1.0** Convex gap model, root version, where the total gap cost is $gapinit + g(x)$. Note the added 1.0.
- **g(x)= root(x+1.0)** Convex gap model, root version, where the total gap cost is $gapinit + g(x)$. Note the added 1.0.

SCORING SYSTEM Choose the method of scoring matches and mismatches. The defaule is not to use a scoring matrix. However many scoring matricies are included with the program, you also have the option of loading your own.

- **No Scoring Matrix** If you are using a scoring matrix you can turn it off with this.
- **PAM 250**
- **PAM250 + 8**
- **McClure**
- **Gonnet**
- **Gribskow**
- **Taylor**
- **Other Scoring Matrix** Load your own scoring matric file.

GET POLYGONS Generates the optimal alignments.

- **Find All Polygons** Decomposes the entire parameter space into convex polygons.
- **Find Closest to Input Alignment** Finds the closest alignment polygon to an Input Alignment.
- **Find Polygon(s) for Point** Finds the polygon(s) that contain a single point chosen by the user.

3 Tutorial

Generating an alignment of two sequences is both simple and fun. The following is a list of basic steps used to generate the alignment shown on the *XPARAL* web page.

- Load the file containing two previously aligned strings. Select *Load Aligned Strings* from the **File** menu. The filename is `bartongo` and the names of string 1 and 2 are `fabvl` and `fabvh`, respectively. If you don't have this file or wish to try out different sequences you can type in your own in the sequence dialog boxes under the main menu bar.

- Now select a weight matrix. From the **Scoring System** menu select *PAM250 +8*.For this decomposition we will use the PAM250 matrix that has been normalized by adding 8 to each entry.

- Choose the parameters, from the **Opt Function** menu select *(#matches)-(#mismatches)-X*(#indels)-Y*(#gaps)*.

- For this alignment we will use the default ranges for the X and Y axies and the default values for the constants $c1$ and $c2$. However, they can be changed with the *ranges* or *constants* menu options.

- Choose the gap model from the **Gap Type** menu. For this alignment we will use the affine model so select *g(x)=x*.

- Now it is possible to decompose the entire parameter space by selecting *Find All Polygons* from the **Get Polygons** menu. It should finish in just a few seconds.

- Once the parameter space has been decomposed you can click on any polygon to view the optimal alignments generated for the range of parameters it encompasses. You can even scroll through the co-optimals using the *Prev* and *Next* buttons.

- Now try saving the alignments generated in one polygon. From the **File** menu choose *New Log File*. Enter the name of the file you wish your alignments to be saved in, e.g. `alignments.log`, and click the

OK button. You can click on the polygon containing the alignments you wish to save and select *Save Dark Polygon* from the **File** menu. A dialog box should appear containing the name of the log file, all you need to do is click OK.

- Now see what the decomposition looks like with convex gaps. Choose *g(x)=log(x)* from the **Gap Type** menu. You can now select *Find All Polygons* from the **Get Polygons** menu to see the new decomposition. Note: this will take a bit longer.

4 Appendix

XPARAL is distributed without warranty of any kind.
Questions and comments can be sent to:

gusfield@cs.ucdavis.edu

```
Dan Gusfield
Department of Computer Science
Engineering II
University of California, Davis
Davis California, 95616
```


XPARAL was developed with support from the Department of Energy Grant DE-FG03-90ER60999.