The **Gotoh** package

Takuto ASAKURA (wtsnjp)

v1.1 [2017/07/07]

**Abstract**

This package is an implementation in \TeX of the Gotoh algorithm, which calculates biological sequence alignments. The package provides only two user commands: \texttt{\Gotoh} for executing the algorithm and \texttt{\GotohConfig} for setting various parameters with a key-value list.

1 System Requirements

System requirements of **Gotoh** are shown below.

- \TeX engine: any engine
- \TeX format: \LaTeX2ε
- Document class: any class
- Required package: xkeyval

2 Loading the **Gotoh** Package

To use the **Gotoh** package, load \texttt{gotoh.sty} with \texttt{\usepackage} command in preamble. No package option is available.

\texttt{\usepackage{gotoh}}

3 Calculating the Alignment

\texttt{\Gotoh} The syntax of \texttt{\Gotoh} is shown below.

\texttt{\Gotoh[(key-value list)]{(sequence A)}{(sequence B)}}

The command puts the optimal score of the alignment to specified control sequence (default: \texttt{\GotohScore}) after executing the Gotoh algorithm, and returns the alignment results to other control sequences (default: \texttt{\GotohResultA} and \texttt{\GotohResultB}). Note that these assignments are done globally. Using the optional argument, you can change the configuration temporary with the same keys in \texttt{\GotohConfig} (see the next section).
4 Configuration

You can change various settings and parameters related to this package with \texttt{\GotohConfig} command. The command takes a key-value list of the settings as its argument and changes the values locally.

\GotohConfig{(key-value list)}

4.1 Control Sequences to Store Results

Control sequences which \texttt{\Gotoh} command return results can be specified with following keys:

\begin{align*}
\text{score} &= \langle control \ sequence \rangle \quad \text{\texttt{\GotohScore}} \\
& \text{sets the control sequence to store optimal score of the last alignment which calculated by \texttt{\Gotoh} command.} \\
\text{result A} &= \langle control \ sequence \rangle \quad \text{\texttt{\GotohResultA}} \\
\text{result B} &= \langle control \ sequence \rangle \quad \text{\texttt{\GotohResultB}} \\
& \text{specify the control sequences to store alignment results respectively corresponding to \langle sequence A \rangle and \langle sequence B \rangle of the arguments of \texttt{\Gotoh}.}
\end{align*}

4.2 Algorithm Parameters

The default value of algorithm parameters which define the scoring are

\begin{equation}
\text{match} = 1, \text{mismatch} = -1, \text{d} = 7, \text{e} = 1. \quad (1)
\end{equation}

Parameters \text{match}, \text{mismatch} define the penalties of a \textit{match} and \textit{mismatch} respectively, and \text{d}, \text{e} are used in a gap penalty (see section 5.6). The parameters appeared in the above equations are able to be changed with following keys:

\begin{align*}
\text{match} &= \langle number \rangle \quad 1 \\
\text{mismatch} &= \langle number \rangle \quad -1 \\
\text{d} &= \langle number \rangle \quad 7 \\
\text{e} &= \langle number \rangle \quad 1 \\
& \text{set the parameters appear in the above equations.}
\end{align*}

Though the \texttt{\Gotoh} command calculates sequence alignment with standard dynamic programming as default, you can use memoization functions instead. Note that both of the methods produce exactly the same results.

\begin{align*}
\text{memoization} &= \langle true|false \rangle \quad \text{\texttt{memoization}} \quad \text{false} \\
& \text{If true, use memoization functions to execute the Gotoh algorithm.}
\end{align*}
4.3 Others

gap char = \langle character \rangle . (period)
is inserted to the alignment results where gaps appear. Note that you have to be careful if using \dash (hyphen) as gap characters because successive hyphens automatically converted to dashes by \TeX. In this case, you can specify \texttt{\mbox{-}} instead.

uppercase = \langle true | false \rangle or uppercase \texttt{false}
If true, uppercase both \langle sequence A \rangle and \langle sequence B \rangle before executing the algorithm.

5 Algorithm and Implementation

5.1 Required package

Package \texttt{xkeyval} is required to process key-value lists.

\begin{verbatim}
\RequirePackage{xkeyval}
\end{verbatim}

5.2 Messages

Commands for warning and error messages.

\begin{verbatim}
\def\gth@pkgname{gotoh}
\def\gth@warn{\PackageWarningNoLine\gth@pkgname}
\def\gth@error{\PackageError\gth@pkgname}
\end{verbatim}

5.3 Switches, Registers and Constants

All boolean switches, count registers and dimen registers used by this package are declared here.

\begin{verbatim}
\if@gth@first@\if@gth@remain@\if@gth@gap@\if@gth@gapx@\if@gth@gapy@
\gth@max
\fi\fi\fi\fi\fi\fi\fi\fi
\gth@tempcnta
\gth@tempcntb
\gth@tempcntc
\gth@tempcntd
\gth@calc
\end{verbatim}

Scratch four count registers for general use and one for some calculation routines.
This is not a counter register but a macro to store \textit{⟨number⟩} which is used as 'serial number'.

\edef\gth@sn{\number\z@}

This macro stores a large negative \textit{⟨number⟩} which is used instead of $-\infty$. Here \texttt{\gth@M} is a largest constant which can be defined by \texttt{\mathchardef} primitive (in traditional \LaTeX{} engine).

\edef\gth@min{\number\gth@M}

Scratch a dimen register for general use.

\newdimen\gth@tempdima

5.4 Defining and Setting Keys

These are keys for setting the parameters of the algorithm. The default values which are shown in Equations (1) are set here, and stored in the macros \textit{(not count registers)} respectively.

\define@cmdkeys[gth]{config}[gth@]{match, mismatch, d, e}
\setkeys[gth]{config}{match=1, mismatch=-1, d=7, e=1}

The macros to store the result of the algorithms are also stored in internal macros.

\define@key[gth]{config}{score}{\def\gth@score{#1}}
\define@key[gth]{config}{result A}{\def\gth@resulta{#1}}
\define@key[gth]{config}{result B}{\def\gth@resultb{#1}}
\setkeys[gth]{config}{score=GotohScore, result A=GotohResultA, result B=GotohResultB}

Both boolean keys \texttt{\if\gth@memoization} and \texttt{\if\gth@uppercase} are set false as default. The gap char is stored in \texttt{\gth@gap@char}.

\define@boolkeys[gth]{config}[\gth@]{memoization, uppercase}[true]
\define@key[gth]{config}{gap char}{\def\gth@gap@char{#1}}
\setkeys[gth]{config}{memoization=false, uppercase=false, gap char={.}}

5.5 Utility Commands

Some primitives to use, define or copy control sequences are wrapped.

\def\gth@nameuse#1{\csname gth@#1\endcsname}
\def\gth@name@edef#1{\expandafter\edef\csname gth@#1\endcsname}
\def\gth@name@xdef#1{\expandafter\xdef\csname gth@#1\endcsname}
\def\gth@glet{\global\let}

Commands to treate macros which store \textit{⟨number⟩} like count registers.

\def\gth@advance#1#2{%
  \gth@calc#1\advance\gth@calc#2\edef#1{\the\gth@calc}
}\def\gth@increment#1{\gth@advance#1\@ne}
\def\gth@decrement#1{\gth@advance#1\m@ne}
5.6 Calculation Routines

Since these calculation routines (macros) are not full expandable, all commands defined here return the results by storing the value in the counter register \texttt{\gth@calc}. This assignments are done locally.

\texttt{\gth@max} This command take comma-separated list of (number) and returns the max value of them.

\begin{verbatim}
\def\gth@max#1{
  \@gth@first@true
  \@for\gth@member:=#1\do{
    \if@gth@first@
      \gth@calc\gth@member
      \@gth@first@false
    \else
      \ifnum\gth@member>\gth@calc
        \gth@calc\gth@member
      \fi
    \fi}
}\gth@gap@penalty
\end{verbatim}

\texttt{\gth@gap@penalty} This command takes \( l \), the length of a gap, as its argument and calculates its penalty which defined by following function\(^1\):

\[
g(l) = -d - (l - 1)e.
\] (2)

In this equation, \( d \) and \( e \) are parameters of the algorithm, which have default values shown in Equations (1).

\begin{verbatim}
\def\gth@gap@penalty#1{
  \gth@calc#1\relax
  \advance\gth@calc1\relax
  \multiply\gth@calc\gth@d
  \advance\gth@calc\gth@e}
\end{verbatim}

5.7 Printing Matrices

\texttt{\gth@print@matrix} This command shows the matrices \( M \), \( I^x \) or \( I^y \) (See section 5.8). The argument of this macro specifies which matrix to show, so you can detect \texttt{m}, \texttt{ix}, or \texttt{iy}. However, this macro is currently only for the author of this package to debug.

\begin{verbatim}
\def\gth@print@matrix#1{
  \bgroup\ttfamily
  \sbox\z@ 0\%
  \expandafter\gth@tempdima\ht\z@
  \multiply\gth@tempdima 8\%
  \mbox{\hbox to\gth@tempdima{\hss #1}}%
  \egroup}
\end{verbatim}

\(^1\)This form of gap penalty is called `Affine gaps'.

5
5.8 Executing the Gotoh Algorithm

This is the user interface to execute the Gotoh algorithm. First, the ‘serial number’ is incremented. The whole other process executed by this macro contained in a group because a number of temporary macros are defined during processing.

These inner macros are for preparing: \texttt{\gth@gotoh\setkeys} process temporary configuration specified with the optional argument of \texttt{\gth@gotoh}. In \texttt{\gth@gotoh\pre}, if the boolean switch \texttt{\if\gth@uppercase} is true, the arguments are processed by \texttt{\gth@gotoh} preamble before passing them to \texttt{\gth@gotoh} macro.

This macro actually executes the Gotoh algorithm \cite{2}. The input of the algorithm is two biological sequences

\[ A \equiv a_1a_2a_3\ldots a_m, \; B \equiv b_1b_2b_3\ldots b_n \]

where \(a_i\) and \(b_j\) are chosen from a finite alphabet, e.g. \{A, T, G, C\}. The command takes these sequences as its arguments, and calculate following recurrence formula\footnote{This calculation is done in \(O(mn)\) time.} to get optimal score of the algorithm:

\[ M_{i+1,j+1} = \max \left\{ M_{ij}, I^x_{ij}, I^y_{ij} \right\} + c_{ij} \]
where

\[ I_{i+1,j} = \max \{ M_{ij} - d, I^x_{ij} - e, I^y_{ij} - d \}, \quad (4) \]
\[ I^y_{i+1,j} = \max \{ M_{ij} - d, I^y_{ij} - e \}, \quad (5) \]

and \( c_{ij} \) is a score for a pair \((a_i, b_j)\), namely

\[ c_{ij} = \begin{cases} 1 & \text{if } a_i = b_j \quad \text{(match)} \\ -1 & \text{otherwise} \quad \text{(mismatch)}. \end{cases} \]

Note that the Equations (4) and (5) have asymmetric form because the order of inserting gaps when alternating \( A \) and \( B \) does not affect the optimal score.

5.8.1 Getting sequences

Here each character \( a_i, b_j \) in the input sequences are stored in \( \text{seqa}_a \) and \( \text{seqa}_b \), and the lengths of the sequences \( m, n \) are stored in \( \text{m} \) and \( \text{n} \) respectively.

5.8.2 Initialization

We have to be careful with initialization because many implementations of the Gotoh algorithm have problems here [1]. Specifically, the package initializes the matrices with following equations:

\[ M_{i0} = -\infty, M_{0j} = -\infty, M_{00} = 0, \]
\[ I^x_{i0} = -g(i), I^x_{0j} = -\infty, \]
\[ I^y_{i0} = -\infty, I^y_{0j} = -g(j). \]

Note that function \( g(l) \) is a gap penalty (shown in Equation (2)) and can be calculated with the macro \( \text{\texttt{\textbackslash gap\textbackslash penalty}} \).
5.8.3 Memoization

If the switch \texttt{if@gth@memoization} is true, memoization functions (See section 5.9) are called recursively.

5.8.4 Dynamic Programming

To fill matrices $M$, $I^x$, and $I^y$ the package use the recurrence formula

First, $I^x$ is calculated with Equation (4).
Secondly, $I^y$ is calculated with Equation (5). 

\begin{verbatim}
\the\gth@calc
\end{verbatim}

Finally, $M$ is calculated with Equation (3) and a loop is completed.

\begin{verbatim}
\the\gth@calc
\end{verbatim}

5.8.5 Printing Matrices

This is piece of code for debugging the package (so usually commented out).

\begin{verbatim}
\the\gth@calc
\end{verbatim}
5.8.6 Returning the Optimal Score

The calculated optimal score of the alignment stored here to the control sequences which stored in \texttt{gth@score}.

\begin{verbatim}
\begingroup
\gth@decrement\gth@m \gth@decrement\gth@n
\expandafter\xdef\gth@score{%
\gth@nameuse{\gth@sn \m@\gth@m \n@\gth@n}}%
\egroup
\end{verbatim}

5.8.7 Trace Back

After processing dynamic programming (or memoization functions), matrices $M$, $I^x$, and $I^y$ are all filled. Using these matrices, we can determine the ‘trace’ from the optimal score $M_{m,n}$ to the start $M_{0,0}$ and get the result of alignment $(x, y)$.

Considering the form of Equation (3)–(5), the calculation formulae of the former values depend on which matrix the ‘current position’ exists. In order to take into account these differences, trace back process is calculated while switching three modes: default, gap $x$, and gap $y$.

First, make sure to empty \texttt{gth@rseq@x} and \texttt{gth@rseq@y} which is going to store result alignment sequences $x$ and $y$ respectively. Note that this is the trace back process, so new characters are prepended to the existing sequences in each loop. In the next line, the switches \texttt{if@gth@gapx@} and \texttt{if@gth@gapy@} are both turned off (which means processing starts from the default mode).

\begin{verbatim}
\let\gth@rseq@x\empty\let\gth@rseq@y\empty
\@gth@remain@true\@gth@gapx@false\@gth@gapy@false
\end{verbatim}

In the main loop of trace back, process the condition either $x$ or $y$ is already completed first. In this situation, the other sequence should be filled all the remaining with gaps.

\begin{verbatim}
\@whilesw\if@gth@remain@\fi{%
\ifnum\gth@m=\z@
\gth@decrement\gth@n
\expandafter\expandafter\expandafter\expandafter\expandafter\expandafter\expandafter
\if\gth@nameuse{seqb@\gth@n} \relax\else
\edef\gth@rseq@x{\gth@gap@char\gth@rseq@x}%
\edef\gth@rseq@y{\gth@nameuse{seqb@\gth@n} \gth@rseq@y}%
\fi
\else\ifnum\gth@n=\z@
\gth@decrement\gth@m
\fi\fi
\else\ifnum\gth@m=\z@
\gth@decrement\gth@n
\expandafter\expandafter\expandafter\expandafter\expandafter\expandafter\expandafter
\if\gth@nameuse{seqa@\gth@m} \relax\else
\edef\gth@rseq@x{\gth@nameuse{seqa@\gth@m} \gth@rseq@x}%
\edef\gth@rseq@y{\gth@gap@char\gth@rseq@y}%
\fi
\else
\edef\gth@rseq@x{\gth@nameuse{seqa@\gth@m} \gth@rseq@x}%
\edef\gth@rseq@y{\gth@gap@char\gth@rseq@y}%
\fi
\fi
\fi
\fi
\fi
\end{verbatim}
mode: gap x Prepend former base to x and a gap to y. If $M_{m-1,n} - d > I_{m-1,n}^x - e$ back to default mode. Else if $I_{m-1,n}^x - e < I_{m-1,n}^y - d$ switch to mode gap y and otherwise stay in mode gap x.

```latex
\texttt{\textbf{\textcolor{red}{mode: gap x}} \texttt{Prepend former base to \texttt{x} and a \texttt{gap} to \texttt{y}. If $M_{m-1,n} - d > I_{m-1,n}^x - e$ back to default mode. Else if $I_{m-1,n}^x - e < I_{m-1,n}^y - d$ switch to mode gap y and otherwise stay in mode gap x.}}
```

mode: gap y Prepend a gap to x and former base to y. If $M_{m,n-1} - d > I_{m,n-1}^y - e$ back to default mode and otherwise stay in mode gap y.

```latex
\texttt{\textbf{\textcolor{red}{mode: gap y}} \texttt{Prepend a \texttt{gap} to \texttt{x} and former base to \texttt{y}. If $M_{m,n-1} - d > I_{m,n-1}^y - e$ back to default mode and otherwise stay in mode gap y.}}
```

mode: default Prepend former base to x and y respectively. Only if $M_{m-1,n-1} > I_{m-1,n-1}^x$ and $M_{m-1,n-1} > I_{n-1,m-1}^y$ stay on default mode, and else if $I_{m-1,n-1}^x > I_{n-1,m-1}^y$ go to gap x otherwise go to gap y.

```latex
\texttt{\textbf{\textcolor{red}{mode: default}} \texttt{Prepend former base to \texttt{x} and \texttt{y} respectively. Only if $M_{m-1,n-1} > I_{m-1,n-1}^x$ and $M_{m-1,n-1} > I_{n-1,m-1}^y$ stay on default mode, and else if $I_{m-1,n-1}^x > I_{n-1,m-1}^y$ go to gap x otherwise go to gap y.}}
```
Finally, if we achieve to $M_{0,0}$, exit from the main loop.

5.8.8 Returning Results

Finally, the results of alignments which stored in $\texttt{gth@resq@x}$ and $\texttt{gth@resq@y}$ are copied globally to the control sequences which stored in $\texttt{gth@resulta}$ and $\texttt{gth@resultb}$.

5.9 Memoization Functions

These are memoization functions for calculating $I^x, I^y, M$ respectively. Since these functions change many ‘temporary’ values, all of the processes are wrapped in a group and only the return values and components of matrices are assigned globally.
5.10 Configuration Command

\GotohConfig  This is just a wrap command of \setkeys.
\newcommand{\GotohConfig}[1]{\setkeys[\gth]{config}{#1}}

References


Index

Numbers written in italic refer to the page where the corresponding entry is described; numbers underlined refer to the code line of the definition; numbers in roman refer to the code lines where the entry is used.

Symbols

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Page Numbers</th>
</tr>
</thead>
<tbody>
<tr>
<td>@for</td>
<td>39, 42, 38, 252</td>
</tr>
<tr>
<td>\gth@first@true</td>
<td>253, 254, 184, 234</td>
</tr>
<tr>
<td>\gth@first@false</td>
<td>220, 259, 184, 265</td>
</tr>
<tr>
<td>\gth@gap@true</td>
<td>200, 220, 257</td>
</tr>
<tr>
<td>\gth@gap@false</td>
<td>218, 220, 265</td>
</tr>
<tr>
<td>\gth@gapx@true</td>
<td>184, 234, 257</td>
</tr>
<tr>
<td>\gth@gapx@false</td>
<td>220, 259</td>
</tr>
<tr>
<td>@ifnextchar</td>
<td>76</td>
</tr>
</tbody>
</table>
Change History

v1.0
General: The first version ........ 1

v1.1
\Gotoh: Add a new optional argument for changing config temporary ........ 6