Introduction to Scientific Computing
Report Practical Example
Anonymous s123456
Nov 13, 2014

Assignment 1 Hamming distance of two DNA sequences

a.

b. The Hamming distance HD for two strings `s` and `t` of the same length, say `len_s`, is defined as the number of character positions in which they differ. So we can compute HD by a loop in which we compare the character at each position *i* of the two strings, and increment HD by 1 for each position where the characters are not the same. HD must be initialized with the value 0. The corresponding Matlab code is:

```
HD=0;
for i=1:len_s
    if (s(i)~=t(i)) HD=HD+1;
    end
end
```

After reading the input file first a check is done whether the input strings are of the same length. The complete code of the file `hamming1.m` is given in Appendix A on page 3.

c. Running the program `hamming1.m` in Matlab gives a value HD=22. This is indeed the correct value, as can be seen by inspecting the two strings in the input file `input.txt`.

d. To compute the positions where `'|'` symbols or spaces have to be inserted, we can extend the loop above. We introduce a new string, say `v`: whenever there is a match on position i, we put `v`(i) equal to `'|'`, otherwise `v`(i) is put equal to a space. In Matlab code:

```
for i=1:len_s
    if (s(i)~=t(i)) HD=HD+1; v(i)=' ';
    else v(i)='|';
    end
end
```

To display the strings `s`, `v`, `t` below one another, we can use Matlab's `print` function for each string. An alternative way is the following. We define a matrix `A` with 3 rows, each of length `len_s`, where the first row of `A` equals the string `s`, the second row equals the string `v`, and the third row equals the string `t`. Then we can use the `disp()` function of Matlab to display the matrix `A`, which will show the desired alignment.

The complete code of the file `hamming2.m` is given in Appendix B on page 4.

e. Running the program `hamming2.m` gives the following output (copied from the command window of Matlab):

```
GGTCCAATGGGATTATGGCCTCTCTATATTATCCA
|   | |     ||     ||  |||  |||
GTCACCAACTTCTTTATATCTGGCTAGCTTAGATT
```

which indeed is correct.

f. To print the alignment, which is defined by the 3×`len_s` matrix `A`, to a file `output`, we use the fprintf command with the output file as the first argument. The `%s` parameter indicates we are printing characters. In pseudocode:

```
for i=1:3
  for j=1:len_s
      fprintf(output,'%s',A(i,j));
  end
  fprintf(output,'\n');
end
```

The complete code of the file `hamming3.m` is given in Appendix C on page 5.

Running the extended program `hamming3.m` gives the output file `hamming3-output.txt` which is included in Appendix D on page 7.

It indeed contains the correct information.

The requested files `hamming1.m`, `hamming2.m`, `hamming3.m`, and `hamming3-output.txt` are contained in the subdirectory `results` of this directory.

A Appendix: Matlab code of hamming1.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Introduction to Scientific Computing - WBCS14003             %
%                                                              %
%  Compute the Hamming distance for 2 sequences in Matlab      %
%                                                              %
%                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all;                           % remove items from the workspace

in=fopen(’input.txt’);               % open file
s=fgetl(in);                         % read line 1 of the input
t=fgetl(in);                         % read line 2 of the input
fclose(in);                          % close file
len_s=length(s);                     % length string s
len_t=length(t);                     % length string t

% Here comes your code
% First test whether the input strings are of the same length. If not, stop.
if (len_s ˜= len_t)
  disp(’lengths of input strings are not equal’);
  return;
end

HD=0;  % initialize the Hamming distance to zero

% Increase HD for each mismatch

for i=1:len_s
    if (s(i)˜=t(i)) HD=HD+1;
    end
end

% Print the Hamming distance on the screen:

fprintf(’Hamming distance=%d\n’,HD);
```

B Appendix: Matlab code of hamming2.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Introduction to Scientific Computing - WBCS14003             %
%                                                              %
%  Compute the Hamming distance for 2 sequences in Matlab      %
%                                                              %
%                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all;                           % remove items from the workspace

in=fopen(’input.txt’);               % open file
s=fgetl(in);                         % read line 1 of the input
t=fgetl(in);                         % read line 2 of the input
fclose(in);                          % close file
len_s=length(s);                     % length string s
len_t=length(t);                     % length string t

% Here comes your code
if (len_s ˜= len_t)
  disp(’lengths of input strings are not equal’);
  return;
end

% The result should be a real number HD equal to the Hamming distance of s and t

HD=0;  % initialize the Hamming distance to zero

% Put string s into first row and t into third row of matrix A
A(1,:)=s;
A(3,:)=t;
% Put ’|’ symbol at all positions of the second row of matrix A
% where there is a match between the letters in string s and string t
% Also, increase HD for each mismatch

for i=1:len_s
    if (s(i)˜=t(i)) HD=HD+1; A(2,i)=’ ’;
    else  A(2,i)=’|’;
    end
end

% Print the Hamming distance on the screen:

fprintf(’Hamming distance=%d\n’,HD);

% Print the alignment on the screen:

fprintf(’\nAlignment:\n\n’);
disp(A);
```

C Appendix: Matlab code of `hamming3.m`

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Introduction to Scientific Computing - WBCS14003             %
%                                                              %
%  Compute the Hamming distance for 2 sequences in Matlab      %
%                                                              %
%                                                              %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all;                           % remove items from the workspace

in=fopen(’input.txt’);               % open file
s=fgetl(in);                         % read line 1 of the input
t=fgetl(in);                         % read line 2 of the input
fclose(in);                          % close file
len_s=length(s);                     % length string s
len_t=length(t);                     % length string t

% Here comes your code
if (len_s ~= len_t)
  disp(’lengths of input strings are not equal’);
  return;
end

% The result should be a real number HD equal to the Hamming distance of s and t

HD=0;  % initialize the Hamming distance to zero

% Put string s into first row and t into third row of matrix A
A(1,:)=s;
A(3,:)=t;
% Put ’|’ symbol at all positions of the second row of matrix A
% where there is a match between the letters in string s and string t
% Also, increase HD for each mismatch

for i=1:len_s
    if (s(i)~=t(i)) HD=HD+1; A(2,i)=’ ’;
    else  A(2,i)=’|’;
    end
end

% Print the Hamming distance on the screen:

fprintf(’Hamming distance=%d\n’,HD);

% Print the alignment on the screen:

fprintf(’\nAlignment:\n\n’);
disp(A);

% Print the alignment to a file:
output=fopen(’hamming3-output.txt’, ’w’);      % open file
```

```
fprintf(output,'Name: Anonymous\n');           % enter your name(s)
fprintf(output,'IBC, Practical Example\n');

fprintf(output,'Hamming distance=%d\n',HD);
fprintf(output,'\nAlignment:\n\n');
% Here comes the code for printing the alignment
% That is, the rows of matrix A below one another
for i=1:3
  for j=1:len_s
  fprintf(output,'%s',A(i,j));
  end
  fprintf(output,'\n');
end
fclose(output);                            % close file
```

D Appendix: output file `hamming3-output.txt`

```
Name: Anonymous
ISC, Practical Example
Hamming distance=22

Alignment:

GGTCCAATGGGATTATGGCCTCTCTATATTATCCA
|   | |     ||     ||  |||  |||
GTCACCAACTTCTTTATATCTGGCTAGCTTAGATT
```