`mtf`, v. 0.4: Move To Front

Bernhard Haubold
Max-Planck-Institute for Evolutionary Biology, Plön, Germany

December 6, 2018

1 Introduction

2 Getting Started

`mtf` was written in C on a computer running Linux and should work on any standard UNIX system. However, please contact me at `haubold@evolbio.mpg.de` if you have any problems with the program.

- Unpack the program

  ```
  tar -xvzf mtf_XXX.tgz
  ```

  where `XXX` indicates the version.
- Change into the newly created directory

  ```
  cd Mtf_XXX
  ```

  and list its contents

  ```
  ls
  ```
- Generate `mtf`

  ```
  make
  ```
- List its options

  ```
  ./mtf -h
  ```

3 Listing

The following listing documents the driver program for `mtf`.

```
/***** mtf.c **************************************
 * Description:
 * Author: Bernhard Haubold, haubold@evolbio.mpg.de
 * Date: Tue Nov 17 08:35:53 2015
 **************************************************/
#include <stdio.h>
#include <stdlib.h>
#include <ctype.h>
#include "interface.h"
#include "eprintf.h"
```

```
#include "list.h"


void scanFile(FILE *fp, Args *args, Node *alphabet){
  int n, i, sum, numNuc;
  double mtfObs, mtfMin, mtfMax, mtf;
  size_t s;
  char **line;

  line = (char **)emalloc(sizeof(char *));

  line[0] = NULL;
  sum = 0;
  numNuc = 0;
  while((n = getline(line,&s,fp)) != -1){
    if(line[0][0] == ’>’){
      printf("%s",line[0]);
      continue;
    }
    if(isprint(line[0][0])){
      if(args->P){
        printf("%s\n",line[0]);
        printList(alphabet);
      }
      alphabet = moveToFront(alphabet,line[0][0]);
      if(!args->s)
        printf("%d",getPos());
      else{
        sum += getPos();
        numNuc++;
      }
      if(args->P && !args->s)
        printf("\n");
    }
    for(i=1;i<n;i++){
      if(isprint(line[0][i])){
        if(args->P)
          printList(alphabet);
        alphabet = moveToFront(alphabet,line[0][i]);
        if(!args->s){
          if(args->d)
            printf("%c%d",args->d,getPos());
          else
            printf("%d",getPos());
        }else{
          sum += getPos();
          numNuc++;
        }
        if(args->P && !args->s)
          printf("\n");
      }
    }
    if(!args->s)
      printf("\n");
```

```c
  }
  if(args->s){
    mtfObs = (double)sum/(double)numNuc;
    mtfMax = 1.5;
    mtfMin = 1.5/numNuc;
    mtf = (mtfObs - mtfMin)/(mtfMax - mtfMin);
    printf("%.3f\n",mtf);
  }
  free(line[0]);
  free(line);
}
/* Sliding window variant of move to front */
void scanFileSw(FILE *fp, Args *args, Node *alphabet){
  int n, i, j, w;
  size_t s;
  char **line;

  line = (char **)emalloc(sizeof(char *));

  line[0] = NULL;
  w = args->w;

  /* scan first window */
  i = 0;
  while(i<w && (n = getline(line,&s,fp)) != -1){
    for(j=0;j<n;j++){
      s+=atoi(line[j]);
      i++;
      if(i==w)
        break;
    }
  }

  /* scan all remaining windows */



  while((n = getline(line,&s,fp)) != -1){
    if(line[0][0] == '>'){
      printf("%s",line[0]);
      continue;
    }


    if(isprint(line[0][0])){
      if(args->P){
        printf("%s\n",line[0]);
        printList(alphabet);
      }
      alphabet = moveToFront(alphabet,line[0][0]);
      printf("%d",getPos());
      if(args->P)
        printf("\n");
    }
    for(i=1;i<n;i++){
      if(isprint(line[0][i])){
        if(args->P)
          printList(alphabet);
        alphabet = moveToFront(alphabet,line[0][i]);
        printf("%c%d",args->d,getPos());
        if(args->P)
          printf("\n");
      }
    }
    printf("\n");
  }
  free(line[0]);
  free(line);
}

int main(int argc, char *argv[]){
  int i;
  char *version;
  Args *args;
  FILE *fp;
  Node *node;

  version = "0.4";
  setprogname2("mtf");
  args = getArgs(argc, argv);
  if(args->v)
    printSplash(version);
  if(args->h || args->e)
    printUsage(version);
  node = getAlphabet(args->a);
  if(args->p){
    printList(node);
    return 0;
  }
  if(args->numInputFiles == 0){
    fp = stdin;
    scanFile(fp, args, node);
  }else{
    for(i=0;i<args->numInputFiles;i++){
      fp = efopen(args->inputFiles[i],"r");
      scanFile(fp, args, node);
      fclose(fp);
    }
  }
  free(args);
  free(progname());
  return 0;
}
```

4 Change Log

- Version 0.3 (Nov. 8, 2016)

  - Do not print null delimiter.
- Version 0.4 (Dec. 5, 2018)
  - Fixed but in `getArgs` in `interface.c`.