Mongrail2 Flowchart

Sneha Chakraborty and Bruce Rannala

March 30, 2025

Contents

List of Figures

List of Tables

1

Bash script: `run_mongrail.sh`

1.1 Running the program

1.1.1 Getting Mongrail up and running on Mac OS X or Unix

The subfolder example contains a puma dataset (34 scaffolds with 10 markers each). To run the puma dataset type:

```
./run_mongrail.sh -A example/Texas.vcf.gz -B
↪  example/Cypress.vcf.gz -i example/hybrids.vcf.gz -r 1.9
```

1.2 Input Files

The mongrail program requires three input files:

1. *phasedA* file - a Variant Call Format (VCF) file containing the **phased** data for population A.

2. *phasedB* file - a Variant Call Format (VCF) file containing the **phased** data for population B.

3. *hybrid* file - a Variant Call Format (VCF) file containing the multilocus genotype data for hybrids.

Currently the program can handle a maximum of 10 markers per chromosome. If the VCF file contains more than 10 markers per chromosome, the program will choose the first 10 markers. In this example puma dataset, Texas is population A and Cypress is population B.

Note: Here we use the word chromosome and scaffold interchangeably.

1.3 Required options and arguments

The mongrail2 program requires the following three options and arguments:

Option	Value	Description
A	filename	Population A VCF file (**phased**)
B	filename	Population B VCF file (**phased**)
i	filename	Hybrids VCF file

1.4 Command line options

The mongrail2 program needs you to specify the recombination rate (in cM/Mb) to be considered for each chromosomes. If the recombination rate is same for all chromosomes, use option `-r` followed by the value. If you want to use different rates for different chromosomes, use option `-R` followed by the filename that contains the chromosome name and corresponding recombination rates. For example, each line of the input file should have the following format:

```
chromID rec_rate
```

where `chromID` is the name of the chromosome and `rec_rate` is the corresponding recombination rate (in cM/Mb).

2

Functions used in `run_mongrail.sh`

2.1 `vcf_to_GT`

2.1.1 Required arguments

The function `vcf_to_GT` takes four arguments `popA`, `popB`, `hybrid` and `no_marker_scaffold`, where

- `popA` - a Variant Call Format (VCF) file containing the **phased** data for population A.
- `popB` - a Variant Call Format (VCF) file containing the **phased** data for population B.
- `hybrid` - a Variant Call Format (VCF) file containing the multi-locus genotype data for hybrids.
- `no_marker_scaffold` - this variable contains the number of markers corresponding to each chromosome(or, scaffold) names. Each line of the variable stores information in the following format:

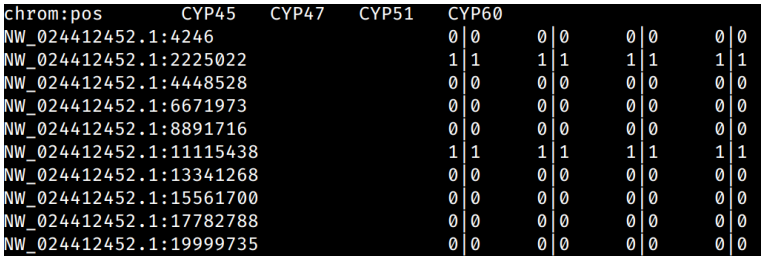
  ```
  n_markers chromID
  ```

  where `chromID` is the name of the chromosome and `n_markers` is the corresponding number of markers. This variable was calculated by using the following bcftools command:

  ```
  no_marker_scaffold=`bcftools query -f '%CHROM\n' ${popA}
  ↪  | uniq -c`
  ```

2.1.2 Output

This function will create intermediate `.GT` files that contains the multilocus genotype data for each chromosome corresponding to each population (population A, population B and hybrids). For example, the puma dataset has 34 scaffolds, thus the function will produce 102 ($= 34 \times 3$) `.GT` files. Below is a screenshot of a `.GT` file for scaffold NW_024412452.1 corresponding to the Cypress population:



chrom:pos	CYP45	CYP47	CYP51	CYP60							
NW_024412452.1:4246				0	0	0	0	0	0	0	0
NW_024412452.1:2225022				1	1	1	1	1	1	1	1
NW_024412452.1:4448528				0	0	0	0	0	0	0	0
NW_024412452.1:6671973				0	0	0	0	0	0	0	0
NW_024412452.1:8891716				0	0	0	0	0	0	0	0
NW_024412452.1:11115438				1	1	1	1	1	1	1	1
NW_024412452.1:13341268				0	0	0	0	0	0	0	0
NW_024412452.1:15561700				0	0	0	0	0	0	0	0
NW_024412452.1:17782788				0	0	0	0	0	0	0	0
NW_024412452.1:19999735				0	0	0	0	0	0	0	0

Figure 2.1: Cypress_NW_024412452.1.GT file

Figure 2.1 there are 10 rows since scaffold NW_024412452.1 has 10 markers. If the VCF file contains more than 10 markers per chromosome, this function will choose the first 10 markers.

Note: A directory named *tmp* is created and all intermediate `.GT` files are stored in it.

2.2 `create_chrom`

2.2.1 Required arguments

The function `create_chrom` takes four arguments `popA`, `rFlag`, `r_rate` and `chrom_name`, where

- `popA` - a Variant Call Format (VCF) file containing the **phased** data for population A.
- `rFlag` - If `rFlag` = 1 it means recombination rate is same for all chromosomes.
- `r_rate` - Recombination rate (cM/Mb) for all chromosomes.
- `chrom_name` - this variable contains the chromosome(or, scaffold) names. Each line of the variable stores information in the following format:

  `chromID`

where `chromID` is the name of the chromosome. This variable was calculated by using the following bcftools command:

```
chrom_name=`bcftools query -f '%CHROM\n' ${popA} | uniq`
```

2.2.2 Output

This function will create intermediate `.chrom` files. For example, the puma dataset has 34 scaffolds, thus the function will produce 34 `.chrom` files. Below is a screenshot of a `.chrom` file for scaffold NW_024412452.1:

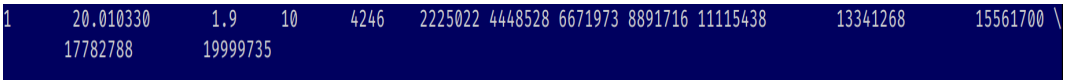


Figure 2.2: NW_024412452.1.chrom file

Figure 2.2 has 14 columns described below:

- The last 10 columns are the position of the markers (in bp).
- The first column is the chromosome index which will be 1 in every scenario as every scaffold has its own `.chrom` file.
- The 2^{nd} column is the length of the chromosome (in Mb). The program doesn't require the user to provide the length. Instead this function obtains the position of the last marker (10^{th}) and adds 1Mb to it, then the sum is considered as the length.
- The 3^{rd} column is the recombination rate (in cM/Mb).
- The 4^{th} column is the number of markers for this chromosome. Highest value is 10 as the program restricts any value greater than 10.

Note: All intermediate `.chrom` files are stored in the directory *tmp*.

2.3 `create_pop_sim`

2.3.1 Required arguments

The function `create_pop_sim` takes one argument `chrom_name` and utilizes the intermediate `.GT` files by accessing them from the *tmp* directory.

- `chrom_name` - this variable contains the chromosome(or, scaffold) names. Each line of the variable stores information in the following format:

  ```
  chromID
  ```

  where `chromID` is the name of the chromosome.

2.3.2 Output

This function will create intermediate `.sim`, `.popA` and `.popB` files.

`.sim` files

Each file contain all compatible diplotypes (phased) for a specific hybrid corresponding to a particular chromosome. In the puma dataset there are 31 putative hybrids and 34 scaffolds, therefore there are 1054 ($= 31 \times 34$) `.sim` files. Below is a screenshot of a `.sim` file of an individual AFP6 (hybrid) for scaffold NW_024412388.1:

chrom:pos	i1	i2	i3	i4
1:37730	0|1	0|1	1|0	1|0
1:11557691	0|0	0|0	0|0	0|0
1:23082228	0|0	0|0	0|0	0|0
1:34599870	0|0	0|0	0|0	0|0
1:46114527	0|0	0|0	0|0	0|0
1:57645184	0|1	1|0	0|1	1|0
1:69160115	0|0	0|0	0|0	0|0
1:80683334	0|0	0|0	0|0	0|0
1:92200788	0|1	0|1	0|1	0|1
1:103724332	0|0	0|0	0|0	0|0

Figure 2.3: AFP6_NW_024412388.1.sim file

chrom:pos	AFP6
NW_024412388.1:37730	0/1
NW_024412388.1:11557691	0/0
NW_024412388.1:23082228	0/0
NW_024412388.1:34599870	0/0
NW_024412388.1:46114527	0/0
NW_024412388.1:57645184	0/1
NW_024412388.1:69160115	0/0
NW_024412388.1:80683334	0/0
NW_024412388.1:92200788	0/1
NW_024412388.1:103724332	0/0

Figure 2.4: Multilocus genotype data (unphased) of individual AFP6 for scaffold NW_024412388.1

Figure 2.3 is the set of compatible diplotypes given the multilocus genotype data of individual AFP6 for scaffold NW_024412388.1 in figure 2.4.

.popA and .popB files

These files contain the haplotype counts for each population for a particular chromosome. For example, the puma dataset has 34 scaffolds, thus the function will produce 34 .popA files and 34 .popB files. Below is a screenshot of a .popA file for scaffold NW_024412452.1:

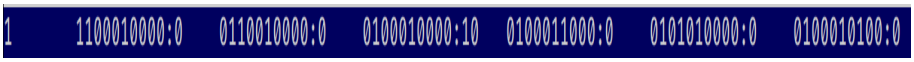


Figure 2.5: Texas_NW_024412452.1.popA file

Below is a screenshot of a .popB file for scaffold NW_024412452.1:

Figure 2.6: Cypress_NW_024412452.1.popB file

1^{st} column in figures 2.5 and 2.6 is the chromosome index which is 1 in every scenario. The rest of the columns are haplotypes with their corresponding count in the population. The format is as follows:
`haplotype:count`
Based on figures 2.5 and 2.6, there are 6 haplotypes 1100010000, 0110010000, 0100010000, 0100011000, 0101010000 and 0100010100 present in the entire population (population A, B and hybrids). Population A and B has 10 and 8 counts of 0100010000 respectively.
Note: All intermediate .sim, .popA and .popB files are stored in the directory *tmp*.

2.4 create_lklhd

This function uses the actual C-program mongrail2 to get the log-likelihood values.

2.4.1 Required arguments

The function create_lklhd takes one argument chrom_name and utilizes the intermediate .popA, .popB, .GT, .sim and .chrom files by accessing them from the *tmp* directory.

- chrom_name - this variable contains the chromosome(or, scaffold) names. Each line of the variable stores information in the following format:
  chromID
  where chromID is the name of the chromosome.

2.4.2 Output

This function will create intermediate `.out` and `.lklhd` files.

`.out` files

An `.out` file of a specific hybrid for a particular chromosome contains the log-likelihood values under each of the six genealogical classes for all it's compatible diplotypes. In the puma dataset there are 31 putative hybrids and 34 scaffolds, therefore there are 1054 ($= 31 \times 34$) `.out` files. Below is a screenshot of a `.out` file of an individual AFP6 (hybrid) for scaffold NW_024412388.1

log_prob_A	log_prob_B	log_prob_C	log_prob_D	log_prob_E	log_prob_F	post_prob_A	post_prob$
-4.642126	-5.447631	-5.021204	-5.706569	-5.007511	-5.497513	0.278139	0.$
-9.084777	-8.097395	-9.180087	-9.467769	-7.896725	-7.220633	0.066639	0.$
-5.717481	-7.430430	-6.472037	-9.467769	-5.991365	-6.621466	0.352183	0.$
-9.084777	-5.509662	-5.596568	-5.205090	-6.336005	-5.997007	0.006436	0.$

Figure 2.7: AFP6_NW_024412388.1.out file

Each row of Figure 2.7 corresponds to the log-likelihood values calculated for every diplotypes in figure 2.3.

`.lklhd` files

Each `.lklhd` file of a specific hybrid contain log-likelihood values for it's multilocus genotype data. For a particular chromosome, the log-likelihood value of the unphased hybrid individual is computed by summing the log-likelihood values for all compatible diplotypes (previously computed in the `.out` file) corresponding to that chromosome. In the puma dataset there are 31 putative hybrids, therefore there are 31 `.lklhd` files. Since there are 34 scaffolds, each `.lklhd` file will have 34 rows. Below is a partial screenshot of a `.lklhd` file of an individual AFP6 (hybrid):

NW_024412376.1	-0.176472	-0.515894	-0.386852	-0.559464	-0.315223	-0.444265
NW_024412377.1	-0.184175	-1.438266	-1.067943	-1.813458	-0.564651	-0.934975
NW_024412388.1	-4.331187	-4.682998	-4.426497	-4.714179	-4.480204	-4.750665
NW_024412391.1	-5.310740	-3.615140	-4.846434	-2.831531	-5.360380	-4.129086
NW_024412394.1	-1.065570	-1.690904	-1.298994	-2.174231	-1.177231	-1.468020
NW_024412395.1	-0.203564	-0.560441	-0.415511	-0.586556	-0.359771	-0.504701
NW_024412396.1	-0.176472	-0.913229	-0.666436	-1.084172	-0.432974	-0.679767
NW_024412397.1	-0.189696	-1.902140	-1.352846	-2.316284	-0.749461	-1.298754
NW_024412400.1	-0.197080	-1.173144	-0.870468	-1.441276	-0.510655	-0.813331
NW_024412401.1	-0.193846	-0.360620	-0.289156	-0.356776	-0.276022	-0.347486
NW_024412402.1	-3.082744	-1.174724	-1.249611	-1.178418	-1.781387	-1.438279
NW_024412403.1	-1.011755	-1.721964	-1.354265	-2.216346	-1.137876	-1.429228
NW_024412404.1	-1.068290	-0.965789	-1.011408	-0.835253	-1.096896	-1.051277
NW_024412405.1	-6.312188	-1.750062	-4.725740	-1.101399	-5.310591	-2.334913
NW_024412406.1	-0.201794	-1.511930	-1.095612	-1.846578	-0.629286	-1.045604
NW_024412407.1	-0.462320	-0.821553	-0.690355	-0.845312	-0.620882	-0.752080
NW_024412411.1	-0.164973	-0.161053	-0.157629	-0.135156	-0.177446	-0.180870
NW_024412412.1	-6.591674	-2.157971	-5.160928	-1.441276	-5.785941	-2.782985
NW_024412413.1	-5.256673	-2.176098	-2.407544	-2.028747	-3.279232	-2.694795
NW_024412415.1	-0.207465	-6.043946	-5.215430	-7.526314	-1.047485	-1.876001
NW_024412416.1	-0.189696	-2.969967	-2.364447	-3.987130	-0.805686	-1.411206
NW_024412417.1	-0.189696	-1.178986	-0.860370	-1.429903	-0.518783	-0.837400
NW_024412418.1	-0.145954	-0.140950	-0.139481	-0.119679	-0.155390	-0.156859

Figure 2.8: AFP6.lklhd file (partial view)

Note: All intermediate `.out` and `.lklhd` files are stored in the directory *tmp*.

2.5 create_PostProb

2.5.1 Required arguments

The function `create_PostProb` takes one argument `hybrid` and utilizes the intermediate `.lklhd` files by accessing them from the *tmp* directory.

- `hybrid` - a Variant Call Format (VCF) file containing the multi-locus genotype data for hybrids.

2.5.2 Output

This function creates the final output files: `post_prob.txt` and `output.txt`. It calculates the posterior probability for each scaffold from `lklhd` files and calculate the combined posterior probability at the end.

- `output.txt` is a text file that contains the estimated posterior probabilities that each individual (input from the hybrid file) belongs to each of the six different genealogical classes. The classes are ordered as follows: a, b, c, d, e, f.
- `post_prob.txt` is a text file that gives a detailed description of the estimated posterior probabilities for each individual under every chromosome.
- `output.txt` file is nothing but a summary of `post_prob.txt`. It contains the final posterior probabilities calculated over all chromosomes.

For example in the puma dataset, `output.txt` file will have 34 rows corresponding to 34 putative hybrids (excluding the headers). Below is a partial screenshot of `output.txt` file for the puma dataset.

Indiv	PostPr(a)	PostPr(b)	PostPr(c)	PostPr(d)	PostPr(e)	PostPr(f)
AFP1	0.000000	0.000114	0.817834	0.000000	0.158117	0.023935
AFP10	0.000000	0.000274	0.000068	0.000000	0.004877	0.994780
AFP11	0.721332	0.000000	0.000000	0.000000	0.266486	0.012182
AFP12	0.000001	0.015412	0.043534	0.000001	0.113942	0.827110
AFP13	0.475736	0.000036	0.000045	0.000000	0.162207	0.361975
AFP14	0.000018	0.000057	0.000031	0.000000	0.047202	0.952692
AFP15	0.000000	0.262739	0.303391	0.000003	0.001249	0.432618
AFP16	0.000000	0.000004	0.001661	0.000000	0.852107	0.146228
AFP17	0.107633	0.000000	0.000000	0.000000	0.872022	0.020345
AFP18	0.856569	0.000000	0.000000	0.000000	0.132519	0.010912
AFP19	0.000000	0.000046	0.001468	0.000000	0.545678	0.452807
AFP2	0.000000	0.011172	0.412574	0.000000	0.113834	0.462420
AFP20	0.000559	0.000010	0.000514	0.000000	0.739041	0.259876
AFP21	0.000122	0.000031	0.006028	0.000000	0.866953	0.126866
AFP22	0.001203	0.000006	0.000023	0.000000	0.263946	0.734821
AFP23	0.000227	0.000001	0.000007	0.000000	0.370086	0.629679
AFP24	0.000000	0.000852	0.000888	0.000000	0.033804	0.964456
AFP25	0.000091	0.000000	0.000476	0.000000	0.991837	0.007596
AFP26	0.000000	0.000002	0.001141	0.000000	0.885534	0.113323
AFP27	0.000671	0.000007	0.000052	0.000000	0.435848	0.563423
AFP28	0.029431	0.000000	0.000000	0.000000	0.045534	0.925034
AFP29	0.999075	0.000000	0.000000	0.000000	0.000922	0.000002

Figure 2.9: `output.txt` file (partial view)

3

Miscellaneous

3.1 Helper Programs

3.1.1 `gendiplo`

This program generates all compatible diplotypes. This was used in function `create_pop_sim`.

- `gendiplo 0 0 530`
  generates compatible diplotypes in binary format for multilocus genotype specified by 0 530. The output is as follows:
  ```
  0000000000 0100100001
  0000100000 0100000001
  0000000001 0100100000
  0000100001 0100000000
  ```

- `gendiplo 1 233 235`
  generates compatible diplotypes in decimal format for multilocus genotype specified by 0 530. The output is as follows:
  ```
  0 530
  16 514
  512 18
  528 2
  ```

3.2 Utility scripts

3.2.1 `transpose.sh`

Transposes a matrix. In our script we transpose a multilocus genotype data file for each hybrid to obtain its diplotypes in subsequent processing steps. Thus we use `transpose.sh` on `.GT` file containing the hybrids (each hybrid at a time). For example, we apply `transpose.sh` to the multilocus

genotype data of hybrid individual AFP6 (Figure 2.4) for scaffold NW_024412388.1 and we get the following figure:

```
chrom:pos       NW_024412388.1:37730    NW_024412388.1:11557691 NW_024412388.1:23082228 NW_024412388.1:34599870 NW_024412>
AFP6    0/1     0/0     0/0     0/0     0/0     0/1     0/0     0/0     0/1     0/0
```

Figure 3.1: `transpose.sh` applied to multilocus genotype data of hybrid AFP6 (truncated view)

3.2.2 Awk scripts

The `.awk` scripts: `get_hyb_diplo.awk`, `hap_count.awk`, `uniq_hap.awk` and `final_update_hap_count.awk` are used in function `create_pop_sim` to get the the intermediate `.sim`, `.popA` and `.popB` files.

1. `get_hyb_diplo.awk`
   - INPUT: The output from `transpose.sh` (refer Figure 3.1)
   - OUTPUT: We obtain the diplotype data from the multilocus genotype data of hybrid AFP6 in binary format which will be `0000000000 1000010010`
2. `hap_count.awk`
   Find unique haplotypes for a single population and their corresponding counts from a transposed `.GT` file.
3. `uniq_hap.awk`
   Find unique set of haplotypes from a larger set that consists of haplotypes from population A, B and all compatible diplotypes pooled from every hybrids.
4. `final_update_hap_count.awk`
   Using the unique haplotypes to get the modified haplotype counts for each population A and B, thus accounting missing haplotypes.
5. `sum_all_dip.awk`
   Sum over all possible diplotypes under a fixed model to obtain the log-probability of multilocus genotype of a hybrid individual under the six different models.
   - INPUT: `.out` file
   - OUTPUT: `.lklhd` file
6. `post_prob.awk`
   Find the posterior probability for each scaffold from `.lklhd` files and calculate the combined posterior probability at the end.

- INPUT: `.lklhd` file
- OUTPUT: `post_prob.txt` file