tugHall version 2.2: USER-GUIDE-tugHall

Requirements for tugHall simulation:

R version **3.3** or later

libraries: **stringr, actuar, tidyr**

Note that the program has two different procedures in general: the first is the simulation and the second is the analysis of the simulation results. Please, pay attention that the requirements for these procedures are **different**. This User-Guide pertains to the **simulation procedure** alone.

Table of Contents

1. Quick start guide

The simplest way to run tugHall:

- Save the **/tugHall_2_2_CNA/** directory to the working folder;
- Run **tugHall_2.2.R**.

The code has its initial input parameters and input files in the **/Input/** folder. After the simulation the user can see results of the simulation (please, see **User-Guide-Analysis_2** for details) in the dialogue box, which will save to the **/Output/** and **/Figures/** folders. Note that the analysis procedure requires additional libraries and a higher version of R - 3.6.0.

2. Structure of directories

Documentation directory:

User-Guide-tugHall_v_2.2.XXX - user guide for a simulation in the XXX = Rmd, html or pdf formats.

User-Guide-Analysis_v2.2.XXX - user guide for the generation of an analysis and a report in the XXX = Rmd, html or pdf formats.

dir **/tugHall_2_2_CNA/** - the directory that contains the software **tugHall** version 2.2.

/tugHall_2_2_CNA/ directory:

tugHall_2.2.R - R script to run a simulation and to define the parameters.

dir **/Code/** - the folder with a code and a function library.

dir **/Input/** - the folder with the input files.

dir **/Output/** - the folder with the output files.

dir **/Figures/** - the folder with the plot figures.

/Code/ directory:

pic_lic.jpg - the necessary file for the user guide.

tugHall_2.2_functions.R - the file that contains the functions for the simulation / core of program.

Analysis_clones.R - the file to analyze the results of a simulation and to plot figures.

Functions_clones.R - the file with the functions for the analysis of results.

/Input/ directory:

cloneinit.txt - file with a list of initial cells with/without destroyed genes.

gene_cds2.txt - file with hallmark variables and weights.

/Output/ directory:

cloneout.txt - the file with simulation output.

geneout.txt - the file with information about hallmark variables and the weights.

log.txt - the file with information about all parameters.

Weights.txt - the file with information about weights between hallmarks and genes.

Point_mutations.txt - the file contains information about point mutations in genome of clones.

CNA.txt - the file contains information about copy number alterations in genome of clones.

Order_of_malfunction.txt - see **USER-GUIDE-Analysis**.

VAF.txt - see **USER-GUIDE-Analysis**.

/Figures/ directory

In the **/Figures/** directory there are figures in *.jpg format, which appear after the analysis of the simulation results. See **USER-GUIDE-Analysis_2**.

3. Inputs

Input of hallmark variables and gene weights

The file **tugHall/Input/gene_hallmarks.txt** defines the hallmark variables and weights:

Table 1. Input file for genes. Example of input file for hallmarks and weights in the file ***tugHall_2_2_CNA/Input/gene_hallmarks.txt***.

Genes	Suppressor or Oncogene	Hallmark	Weights
APC	s	apoptosis	0.2616483
APC	s	growth	0.3285351
APC	s	invasion	0.3746081
KRAS	o	apoptosis	0.2099736
KRAS	o	growth	0.2881968
KRAS	o	immortalization	0.4735684
KRAS	o	angiogenesis	0.3525394
KRAS	o	invasion	0.0446472
TP53	s	apoptosis	0.2543523
TP53	s	growth	0.3076387
TP53	s	angiogenesis	0.4012288
TP53	s	immortalization	0.5264316
TP53	s	invasion	0.0645107
PIK3CA	o	invasion	0.3588945
PIK3CA	o	growth	0.2879753
PIK3CA	o	angiogenesis	0.3261495

PIK3CA	o	apoptosis	0.2938981

1. **Genes** - name of gene, e.g., TP53, KRAS. The names must be typed carefully. The program detects all the unique gene names.
2. **Suppressor or oncogene.** - Distinction of oncogene/suppressor:

- o: oncogene
- s: suppressor
- ?: unknown (will be randomly assigned)

3. **Hallmark** - hallmark name, e.g., “apoptosis.” Available names:

- apoptosis
- immortalization
- growth
- anti-growth
- angiogenesis
- invasion

Note that “growth” and “anti-growth” are related to the single hallmark “growth/anti-growth.” Note that “invasion” is related to “invasion/metastasis” hallmark.

4. **Weights** - Hallmark weights for genes, e.g., 0.333 and 0.5. For each hallmark, the program checks the summation of all the weights. If it is not equal to 1, then the program normalizes it to reach unity. Note that, if the gene belongs to more than one hallmark type, it must be separated into separate lines.

After that, the program defines all the weights. **Unspecified weights** are set to 0. Program performs normalization so that the sum of all weights should be equal to 1 for each column. The **tugHall/Output/Weights.txt** file saves these final input weights for the simulation. Only the first 10 lines are presented here:

Table 2. Weights for hallmarks. Example of weights for hallmarks and genes from ***tugHall/Output/Weights.txt*** file. Unspecified values equal 0.

Genes	Apoptosis, H_a	Angiogenesis, H_b	Growth / Anti-growth, H_d	Immortalization, H_i	Invasion / Metastasis, H_{im}
APC	0.2565501	0.0000000	0.2709912	0.0000000	0.4445540
KRAS	0.2058822	0.3264502	0.2377183	0.4735684	0.0529836
TP53	0.2493962	0.3715365	0.2537549	0.5264316	0.0765560
PIK3CA	0.2881715	0.3020133	0.2375356	0.0000000	0.4259064

1. **Genes** - name of genes.
2. **Apoptosis,** H_a - weights of hallmark “Apoptosis.”
3. **Angiogenesis,** H_b - weights of hallmark “Angiogenesis.”
4. **Growth / Anti-growth,** H_d - weights of hallmark “Growth / Anti-growth.”
5. **Immortalization,** H_i - weights of hallmark “Immortalization.”
6. **Invasion / Metastasis,** H_{im} - weights of hallmark “Invasion / Metastasis.”

Input the probabilities

The input of the probabilities used in the model is possible in the code for parameter value settings, **“tugHall_2_2.R”**:

Probability variable and value	Description
E0 <- 2E-4	Parameter E_0 related to environmental resource limitation
F0 <- 1E0	Parameter F_0 related angiogenesis
m <- 1E-6	Point mutation probability m'
	Gene malfunction probability by point mutation for

uo <- 0.5	oncogene u_o
us <- 0.5	Gene malfunction probability by point mutation for suppressor u_s
s <- 10	Parameter in the sigmoid function s
k <- 0.1	Environmental death probability k'
m_dup <- 0.01	CNA duplication probability m_{dup}
m_del <- 0.01	CNA deletion probability m_{del}
lambda_dup <- 7000	CNA duplication average length λ_{dup} (of the geometrical distribution for the length)
lambda_del <- 5000	CNA deletion average length λ_{del} (of the geometrical distribution for the length)
uo,dup <- 0.8	Gene malfunction probability by CNA duplication for oncogene $u_{o,dup}$
us,dup <- 0	Gene malfunction probability by CNA duplication for suppressor, $u_{s,dup}$. Currently, 0 is assumed.
uo,del <- 0	Gene malfunction probability by CNA deletion for oncogene $u_{o,del}$. Currently, 0 is assumed.
us,del <- 0.8	Gene malfunction probability by CNA deletion for suppressor, $u_{s,del}$.

Filename input

Also in the code **“tugHall_2_2.R”** user can define names of input and output files, and additional parameters of simulation:

Variables and file names	Description
genefile <- ‘gene_hallmarks.txt’	File with information about gene-hallmarks weights
mapfile <- ‘gene_map.txt’	File with information about genes’ map
clonefile <- ‘cloneinit.txt’	Initial Cells
geneoutfile <- ‘geneout.txt’	Gene Out file with hallmarks
cloneoutfile <- ‘cloneout.txt’	Output information of simulation
logoutfile <- ‘log.txt’	Log file to save the input information of simulation
censore_n <- 30000	Max cell number where the program forcibly stops
censore_t <- 200	Max time where the program forcibly stops

Input of the initial clones

The initial states of cells are defined in **“tugHall_2_clones/Input/cloneinit.txt”** file:

Clone ID	List of malfunctioned genes	Number of cells
1	""	1000
2	“APC”	10
3	“APC, KRAS”	100
4	“KRAS”	1

5	“TP53, KRAS”	1
…	…	100
1000	""	10

1. **Clone ID** - ID of clone, e.g., 1, 324.
2. **List of malfunctioned genes** - list of malfunctioned genes for each clone, e.g. ““,”KRAS, APC“. The values are comma separated. The double quotes (”") without gene names indicate a clone without malfunctioned genes.
3. **Number of cells** - number of cells in each clone, e.g., 1, 1000.

Input of the genes’ maps

This new version of **tugHall** allows to calculate CNAs in the genome. The breakpoints of CNAs may fall on genic regions consisting of exons and introns. That’s why it’s needed to enter information about gene’s map. In the **/Input/** directory you can find **CCDS.current.txt**, which was getting from CCDS database at the National Center for Biotechnology Information and has information about genes. At the beginning of simulation, the program reads this file and extracts genes’ map, which is put into **“tugHall_2_clones/Input/gene_map.txt”**. For example, the map is shown as follow:

Chr	CCDS_ID	Gene	Start	End
5	CCDS4107.1	APC	112754890	112755024
5	CCDS4107.1	APC	112766325	112766409
5	CCDS4107.1	APC	112767188	112767389
5	CCDS4107.1	APC	112775628	112775736
5	CCDS4107.1	APC	112780789	112780902
5	CCDS4107.1	APC	112792445	112792528
5	CCDS4107.1	APC	112801278	112801382
5	CCDS4107.1	APC	112815494	112815592
5	CCDS4107.1	APC	112818965	112819343
5	CCDS4107.1	APC	112821895	112821990

1. **Chr** - Name of the chromosome, e.g., 1, 12, X, Y.
2. **CCDS_ID** - ID of the gene in the CCDS database.
3. **Gene** - the name of the gene.
4. **Start** - the start position of each exon of the gene.
5. **End** - the final position of each exon of the gene.

---

4. Outputs

The output data consists of several files after the simulation.

“log.txt” file

The file **“log.txt”** contains information about probabilities and file names. These variables are explained in the “Inputs”.

Table 3. log.txt file. Example of log.txt file.

Variable	Value
genefile	Input/gene_hallmarks.txt
clonefile	Input/cloneinit.txt

geneoutfile	Output/geneout.txt
mapfile	Output/gene_map.txt
cloneoutfile	Output/cloneout.txt
logoutfile	Output/log.txt
E	1e-04
F	10
m	1e-07
uo	0.5
us	0.5
s	10
k	0.2
m_dup	0.01
m_del	0.01
lambda_dup	7
lambda_del	7
uo,dup	0.8
us,dup	0
uo,del	0
us,del	0.8
censore_n	1e+05
censore_t	100
d0	0.35

“geneout.txt” file

The file **“geneout.txt”** contains input information about the weights that connect the hallmarks and genes, which are defined by the user. These variables also are explained in the “Inputs”.

Table 4. geneout.txt file. Given below is an example of the geneout.txt file.

Gene_name	Hallmark_name	Weight	Suppressor_or_oncogene
APC	apoptosis	0.2565501	s
KRAS	apoptosis	0.2058822	o
TP53	apoptosis	0.2493962	s
PIK3CA	apoptosis	0.2881715	o
KRAS	immortalization	0.4735684	o
TP53	immortalization	0.5264316	s
APC	growth|anti-growth	0.2709912	s
KRAS	growth|anti-growth	0.2377183	o
TP53	growth|anti-growth	0.2537549	s
PIK3CA	growth|anti-growth	0.2375356	o

“cloneout.txt” file

The file **“cloneout.txt”** contains the results of the simulation and includes the evolution data: all the output data for each clone at each time step (only the first 10 lines are presented):

Table 5. Output data. Example of output data for all clones. The names of columns are related to the description in the Tables 1,2 and *USER-GUIDE-Analysis_2*’s figures. Columns are from 1 to 15.

Time	N_cells	AvgOrIndx	ID	Parent_ID	Birth_time	c.	d.	i.	im.	a.	k.	E.	N	Nmax.	M
0	-	avg	-	-		0.0000000	0.1500	1	0	0.0066929	0.2	1e-04	2000	10000	0
0	1000	1	1	0	0	0.0000000	0.1500	1	0	0.0066929	0.2	1e-04	2000	10000	0
0	1000	2	2	0	0	0.0000000	0.1500	1	0	0.0066929	0.2	1e-04	2000	10000	0
1	-	avg	-	-		0.1624990	0.1646	1	0	0.0066929	0.2	1e-04	1854	10000	0
1	909	1	1	0	0	0.1506329	0.1646	1	0	0.0066929	0.2	1e-04	1854	10000	0
1	945	2	2	0	0	0.1739130	0.1646	1	0	0.0066929	0.2	1e-04	1854	10000	0
2	-	avg	-	-		0.3370550	0.1754	1	0	0.0066929	0.2	1e-04	1746	10000	0
2	880	1	1	0	0	0.3541848	0.1754	1	0	0.0066929	0.2	1e-04	1746	10000	0
2	865	2	2	0	0	0.3196084	0.1754	1	0	0.0066929	0.2	1e-04	1746	10000	0
2	1	3	3	1	1	0.3541848	0.1754	1	0	0.0066929	0.2	1e-04	1746	10000	0

1. **Time** - the time step, e.g., 1, 50.
2. **N_cells** - the number of cells in this clone, e.g. 1000, 2.
3. **AvgOrIndx** - “avg” or “index”: “avg” is for a line with averaged values across different (index) lines at the same time step; “index” shows the cell’s index at the current time step, e.g., avg, 4,7.
4. **ID** - the unique ID of clone, e.g., 1, 50.
5. **Parent_ID** - the parent index, e.g., 0, 45.
6. **Birth_time** - the birthday time step, e.g., 0, 5.
7. **c** - the counter of cell divisions for the clone.
8. **d** - the probability of division for the cell, e.g., 0.1, 0.8.
9. **i** - the probability of immortalization for the cell, e.g., 0.1, 0.8.
10. **im** - the probability of invasion/metastasis for the cell, e.g., 0.1, 0.8.
11. **a** - the probability of apoptosis for the cell, e.g., 0.1, 0.8.
12. **k** - the probability of death due to the environment, e.g., 0.1, 0.8.
13. **E** - the E coefficient for the function of the division probability, e.g., 10^4, 10^5.
14. **N** - the number of primary tumor cells at this time step, e.g., 134, 5432.
15. **Nmax** - the theoretically maximal number of primary tumor cells, e.g., 10000, 5000.
16. **M** - the number of metastasis cells at this time step, e.g., 16, 15439.

Continuation of Table 5. Columns are from 16 to 22.

Time	N_cells	AvgOrIndx	Ha	Him	Hi	Hd	Hb	type	mut_den	PointMut_ID	CNA_ID
0	-	avg	0	0	0	0	0	0	0		
0	1000	1	0	0	0	0	0	0	0		
0	1000	2	0	0	0	0	0	0	0	23,44	
1	-	avg	0	0	0	0	0	0	0		
1	909	1	0	0	0	0	0	0	0		
1	945	2	0	0	0	0	0	0	0	23,44	
2	-	avg	0	0	0	0	0	0	0		
2	880	1	0	0	0	0	0	0	0		
2	865	2	0	0	0	0	0	0	0	23,44	
2	1	3	0	0	0	0	0	0	0	23,44	2

17. **Ha** - the value of the hallmark “Apoptosis” for the cell, e.g., 0.1, 0.4444.
18. **Him** - the value of the hallmark “Invasion / Metastasis” for the cell, e.g., 0.1, 0.4444.
19. **Hi** - the value of the hallmark “Immortalization” for the cell, e.g., 0.1, 0.4444.
20. **Hd** - the value of the hallmark “Growth / Anti-growth” for the cell, e.g., 0.1, 0.4444 .
21. **Hb** - the value of the hallmark “Angiogenesis” for the cell, e.g., 0.1, 0.4444 .

22. **type** - the type of the cell: “0” is primary tumor cell, “1” is the metastatic cell, e.g., 0, 1.
23. **mut_den** - the density of mutations (tumor mutation burden) for the cell, e.g., 0, 0.32.
24. **PointMut_ID** - the index of data row for point mutation data frame saved at the end of simulation in the file **Point_mutations.txt**, e.g., 23, 32.
25. **CNA_ID** - the index of data row for CNA data frame saved at the end of simulation in the file **CNA.txt**, e.g., 44, 21.

There are two columns (24th and 25th) with the indices of point mutations and CNAs in Table 5. Each index corresponds to index in the related data frames for point mutations and for CNAs (Tables 6 and 7 respectively).

Table 6. Point mutation data frame which will be saved to the file **Point_mutations.txt** at the end of simulation.

PointMut_ID	Parental_1or2	Chr	Ref_pos	Phys_pos	Delta	Copy_number	Gene_name	MalfunctionedI
B23	2	5	112775687	[112775687,112775697]	[0, -10]	2	APC	TRU
A23	1	5	112775687	[-]	[-]	0	NA	NA

1. **PointMut_ID** - ID of point mutation, ‘B’ indicates the variant allele B, ‘A’ does the original allele A.
2. **Parental_1or2** - indicates either of the two parental chromosomes.
3. **Chr** - name of a chromosome.
4. **Ref_pos** - the reference positions of an allele. The reference position is on the coordinate system of the human reference genome.
5. **Phys_pos** - the physical position of an allele. The physical length of a (parental) chromosome is extended or shrunk by CNA duplications or deletions, respectively. When a duplication happens, the reference position is divided into two or more physical positions, which are represented by multiple elements in a vector. When a deletion happens and the allele is lost, the lost is represented by “-” on the coordinate system of physical positions.
6. **Delta** - difference between the reference and physical positions.
7. **Copy_number** - the copy number of an allele.
8. **Gene_name** - the name of a gene.
9. **MalfunctionedByPointMut** - logical indicator of whether or not the gene is malfunctioned by the point mutation.

Table 7. CNA mutation data frame which will be saved to the file **CNA.txt** at the end of simulation.

CNA_ID	Parental_1or2	DuplicationOrDeletion	Chr	Reference_start	Reference_end	Gene_name(s)	MalfunctionedBy(
2	1	dup	5	112775635	112775637	[APC]	TRUE
3	2	del	5	112775636	112775637	[APC]	FALSE

1. **CNA_ID** - ID of CNA.
2. **Parental_1or2** - indicates either of the two parental chromosomes.
3. **DuplicationOrDeletion** - indicator of duplication or deletion for CNA.
4. **Chr** - name of a chromosome.
5. **Reference_start** - the reference position of the CNA start.
6. **Reference_end** - the reference position of the CNA end.
7. **Gene_name** - the name(s) of a gene(s).
8. **MalfunctionedByCNA** - logical indicator of whether or not the gene(s) is malfunctioned by the CNA.

---

5. How to run

In order to make the simulation, please follow the procedure:

1. Copy **/tugHall_2_2_CNA/** directory into the working directory.
2. CD to the **/tugHall_2_2_CNA/** directory.
3. Run the **tugHall_2.2.R** file, using the command line like

```
R --vanilla < tugHall_2_2.R
```

or using the line by line procedure in **R Studio**. In this case we have:

- **`load library(stringr)`** and **`source(file = "Code/tugHall_2.2_functions.R")`**;
- create the Output and Figures directories, if needed;
- define the simulation parameters;
- make the input file for initial cells, if needed;
- run the *model()* function to simulate;
- run **`source("Code/Analysis_clone.R")`** in order to analyze the results and plot the figures in the dialogue box (see **User-Guide-Analysis_v2.2**).

4. To obtain analysis reports of the simulation, please refer to **User-Guide-Analysis_v2.2.RMD**. In **User-Guide-Analysis_v2.2.RMD**, commands are embedded to include files under **Output/** and **Figure/**. So, after analysis with tugHall, you can generate analysis reports automatically from **User-Guide-Analysis_v2.2.RMD**. For more details, please refer to “Writing reproducible reports in R” on the github (https://nicercode.github.io/guides/reports/).

6. Differences with cell-based code and version 2.0

6.1. Reason to develop clone-based code

- Clone-based code was designed to accelerate calculation and increase number of tumor cell. Advantage of clone-based algorithm is making trial for all cells at 1 clone with one application of **trial()** function. In cell-based algorithm **trial()** applies to each cell. But if number of cells equal number of clones, then speed up is 1. That’s why clone-based code works faster for any cases.
- Another reason is a case, when we need to simulate huge number of cells like 10^7 or 10^9, but mutation rate is very low. Cell-based algorithm takes a huge computational cost, and vice versa clone-based algorithm will work very fast, if mutated cells will appear slowly.

6.2. Usage of *trial()* function

- In **trial()** function program applies several trials like environmental death, apoptosis death, division process, etc. We changed the trials with probability p (for some death process) for each cell in the clone with for 1 trial with procedure:

$$N_{cells} = N_{cells} - Binom(p, N_{cells}),$$

where $Binom(p, N_{cells})$ is random number from the binomial distribution with probability p, N_{cells} is a number of cells in a clone. Probability p is one of probability of death process, for example $p = a'$ or $p = k$ etc.

- For cell division with probability d' the new number of cells will be:

$$N_{cells} = N_{cells} + Binom(d', N_{cells})$$

- Check at the end of **trial()** function: if $N_{cells} = 0$, then the clone has died.

6.3. Usage of mutation function

- In mutation function we have changed the mutation to birth of a new clone (one mutation is a birth of one clone):

$$N_{new\_clones} = Binom(m, N_{new\_cells}),$$
$$N_{new\_cells} = Binom(d', N_{cells}).$$

- Passenger or Driver mutations do not matter for new clone’s generation. Only during analysis, we will distinguish Passengers or Drivers clones.

6.4. Average function

- The average values $\bar{x}$ of probabilities or hallmarks are found by summation on the x_i with multiplication by cells number $N_{cells,i}$ of this clone:

$$\bar{x} = \sum_i x_i \times w_i,$$

where $w_i = \frac{N_{cells,i}}{N_{cells,tot}}$ is ith clone’s occupancy in whole cell population $N_{cells,tot} = \sum_i N_{cells,i}$, x_i is the value for ith clone, summation applies for all clones $i = 1..N_{clones}$.

- For this purpose, we added the calculation of cells number (primary and metastasis) before average and hallmarks update.

6.5. Difference with version 2.0

In the current version we use library *actuar* to make non-zero-binom calculation faster, and we use the approximation for big numbers of cells in **trial()** function, because **rbinom()** function in R has restriction for big numbers like $n \times p > 10^{12}$.

7. Differences with clone-based code and version 2.1

7.1. Reason to develop CNA-based code

New version of tugHall with copy number alteration (CNA) was designed for correct calculation of VAF influenced by CNA and tumor purity. It's expected that this design should improve comparison between observation VAF $\in [0; 1]$ and calculated VAF. The previous versions of tugHall have VAF in the range $[0; 0.5]$ because of the neglect of CNA and tumor purity.

7.2. Calculation of point and CNA mutations

Probabilities of CNA mutations are calculated in the same way as point mutations:

- $m_{point} = m_0 \times l_{CDS}$ - for point mutation, where l_{CDS} is the length of all exons of a gene and m_0 is a constant per base pairs defined by users;
- $m_{0,dup}$ and $m_{0,del}$, or we collectively call $m_{0,CNA}$, indicate the first breakpoint event of a CNA and is a constant per base pairs defined by users. $m_{CNA} = m_{0,CNA} \times l_{total}$, where l_{total} is the total region size of all genes of interest which consists of exons as well as introns.
- a length of CNA is calculated using geometrical distribution: $l_{CNA} = rgeom(1, 1/\lambda_{CNA})$+1, where λ_{CNA} is average base-pair length defined by users.
- probability of malfunctioning a gene $u = u_{s,CNA}$ for suppressor and $u = u_{o,CNA}$ for oncogene.

So, the algorithm of CNA is as follow:

```
if ( runif(1) < m_dup + m_del ) then 'Generate CNA':
  - define which event should occur - duplication or deletion using ratio m_dup/m_del like:
      event  <-  sample(c('dup', 'del'), 1, prob = c( m_dup, m_del )/sum(m_dup, m_del) )

  - find randomly first position within the regions of genes of interest;
  - find the length of CNA from geometrical distribution
  - define with probability 0.5 is it the parental chromosome 1 or 2;
  - define the list of genes in CNA;
  - define with probabilty u = {u_o or u_s} is it malfunction for each gene;
  - check overlap of position for other mutations (and if it's necessesary change their
positions).
```

The calculation of probabilities and hallmarks variables is not changed.

At the end of a simulation the VAF frequencies are calculated in accordance with formulation:

$$VAF^i = \frac{(1-\rho)n^i_{B,N} + \sum_{s=1}^{\#sp} \tau_s n^i_{B,S}}{(1-\rho)(n^i_{A,N} + n^i_{B,N}) + \sum_{s=1}^{\#sp} \tau_s (n^i_{A,S} + n^i_{B,S})},$$

where:

i is position (site) index,

s is subpopulation index,

τ is subpopulation fraction,

ρ is tumor purity,

n is copy number,

A denotes an original allele A, B - variant B, and N - normal.

In usual application and in program we used for normal cells $n^i_{A,N} = 2$ and $n^i_{B,N} = 0$, so VAF is calculated as follow:

$$VAF^i = \frac{\sum_{s=1}^{\#sp} \tau_s n^i_{B,S}}{2(1-\rho) + \sum_{s=1}^{\#sp} \tau_s (n^i_{A,S} + n^i_{B,S})}$$