Package: fbnet (via r-universe)

October 23, 2026

Title Forensic Bayesian Networks

Version 1.0.3

Description Open-source package for computing likelihood ratios in kinship testing and human identification cases (Chernomoretz et al. (2021) <doi:10.1016/j.fsir.2020.100132>). It has the core function of the software GENis, developed by Fundación Sadosky. It relies on a Bayesian Networks framework and is particularly well suited to efficiently perform large-size queries against databases of missing individuals (Darwiche (2009) <doi:10.1017/CBO9780511811357>).

License GPL (>= 3)

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

URL https://marsicofl.github.io/fbnet/, https://github.com/MarsicoFL/fbnet

BugReports https://github.com/MarsicoFL/fbnet/issues

Imports graphics, grDevices, igraph, stats, Rsolnp

Suggests mispitools (>= 1.4.0), pedtools, forrel, paramlink, testthat (>= 3.0.0)

Config/testthat/edition 3

Config/pak/sysreqs libglpk-dev libxml2-dev

Repository https://marsicofl.r-universe.dev

Date/Publication 2026-08-24 19:29:01 UTC

RemoteUrl https://github.com/marsicofl/fbnet

RemoteRef HEAD

RemoteSha 12cfc8abe76a24330007913b66cd1b3b5c2c6117

Contents

Argentina_STRs . . . 3
bnet . . . 3
buildBN . . . 3
buildCPTs . . . 4
convertPedformat . . . 5
evidencePrunning . . . 6
factorHeteroFounders . . . 6
FamiliasLocus . . . 7
FamiliasPedigree . . . 8
fbnet . . . 9
from_pedtools . . . 10
getConditional . . . 11
getGenotypeTables . . . 12
getLocusCPT . . . 12
getMAP . . . 13
getQSetRMP . . . 14
getValuesOut . . . 14
imposeEvidence . . . 15
initBN . . . 15
initBN.fromPed . . . 16
initBN.fromVars . . . 16
minOrdering . . . 17
pbn . . . 17
preparePed . . . 18
prodFactor . . . 18
pruneNodes . . . 19
removeEvidenceFromPed . . . 19
reportLR . . . 20
reportPQ . . . 20
reverseSplit . . . 21
setOrdering . . . 21
stateRemoval . . . 22
stateRemoval2 . . . 22
stateRemovalSubnucs . . . 23
sumFactor . . . 23
toybase . . . 24
toyped . . . 24
velim.bn . . . 24

Index **26**

Argentina_STRs — *STRs allelic frequencies from Argentina.*

Description

STRs allelic frequencies from Argentina.

Usage

```
Argentina_STRs
```

Format

A data frame with allele frequencies

bnet — *Initialized bayesian network.*

Description

Initialized bayesian network.

Usage

```
bnet
```

Format

Initialized bayesian network, based on toyped.

buildBN — *buildBN: a function for building the bayesian network.*

Description

buildBN: a function for building the bayesian network.

Usage

```
buildBN(pbn, QP)
```

Arguments

pbn	A bayesian network for pedigree object with information of the genotyped members. The ped object must be in Familias format.
QP	Query Persons Ids

Value

A bayesian network based on pedigree evidence and QP definition.

Examples

```
## Not run:
  # Modern entry point via pedtools (recommended):
  library(pedtools); library(forrel)
  ped  <- linearPed(2)
  ped  <- setMarkers(ped, locusAttributes = NorwegianFrequencies[1:2])
  ped  <- profileSim(ped, N = 1, ids = 2, seed = 1)
  pbn  <- from_pedtools(ped, QP = "5")
  bnet <- buildBN(pbn, QP = "5")

## End(Not run)
```

---

buildCPTs — *buildCPTs: a function for building conditional probability tables based on pedigree bayesian network.*

Description

buildCPTs: a function for building conditional probability tables based on pedigree bayesian network.

Usage

```
buildCPTs(
  bn,
  bNodePrunning = TRUE,
  bStateRemoval = TRUE,
  bStateRemoval2 = TRUE,
  lumpingParameter = NULL,
  renorm = "row-wise",
  verbose = FALSE
)
```

Arguments

bn	A bayesian network for pedigree object with information of the genotyped members. The ped object must be in Familias format.
bNodePrunning	Standard pruning.
bStateRemoval	State based pruning.
bStateRemoval2	State based pruning (model 2).
lumpingParameter	Used for stepwise mutational model.
renorm	If "row-wise" is selected, zero probability is assigned for transitions out of range.
verbose	Computations output.

Value

A bayesian network based on pedigree evidence and QP definition.

Examples

```
## Not run:
  library(pedtools); library(forrel)
  ped  <- linearPed(2)
  ped  <- setMarkers(ped, locusAttributes = NorwegianFrequencies[1:2])
  ped  <- profileSim(ped, N = 1, ids = 2, seed = 1)
  pbn  <- from_pedtools(ped, QP = "5")
  bnet <- buildBN(pbn, QP = "5")
  bn1  <- buildCPTs(bnet)

## End(Not run)
```

---

convertPedformat *convertPedformat: a function for converting a pedtools ped onject to a famlink ped object.*

Description

convertPedformat: a function for converting a pedtools ped onject to a famlink ped object.

Usage

```
convertPedformat(x, verbose = FALSE)
```

Arguments

x	A pedtools ped object.
verbose	Function output.

Value

A dataframe with LRs.

`evidencePrunning` *evidencePrunning: a fuction for pruning instantiated variables.*

Description

evidencePrunning: a fuction for pruning instantiated variables.

Usage

```
evidencePrunning(bn)
```

Arguments

`bn`	A bayesian network (output of buildBN function).

Value

A preprocessed bayesian network.

`factorHeteroFounders` *factorHeteroFounders: a function for multiplying probabilities in case of heterocigote founders.*

Description

factorHeteroFounders: a function for multiplying probabilities in case of heterocigote founders.

Usage

```
factorHeteroFounders(rresQ, bn)
```

Arguments

`rresQ`	List of CPTs.
`bn`	A bayesian network for pedigree object with information of the genotyped members. The ped object must be in Familias format.

Value

A dataframe with genotype probabilities.

FamiliasLocus *FamiliasLocus: a function for reading Familias locus data.*

Description

This function has been entirely copied from Familias R package, it has been removed from CRAN. All references and citations about this function should be done to Familias developers. More information could be obtained from: https://www.familias.name/index.html. In fbnet package, this function is only used for dealing with the pedigree object.

Usage

```
FamiliasLocus(
  frequencies,
  allelenames,
  name,
  MutationModel = "Stepwise",
  MutationRate = 0,
  MutationRange = 0.5,
  MutationRate2 = 0,
  MutationMatrix,
  Stabilization = "None",
  MaxStabilizedMutrate = 1,
  femaleMutationModel,
  femaleMutationRate,
  femaleMutationRange,
  femaleMutationRate2,
  femaleMutationMatrix,
  maleMutationModel,
  maleMutationRate,
  maleMutationRange,
  maleMutationRate2,
  maleMutationMatrix
)
```

Arguments

frequencies	allele frequencies
allelenames	names
name	system name
MutationModel	model
MutationRate	rates
MutationRange	range
MutationRate2	rate two, applied for extended stepwise
MutationMatrix	matrix

`Stabilization` stabilization factor
`MaxStabilizedMutrate`
mix factor
`femaleMutationModel`
for females
`femaleMutationRate`
rate
`femaleMutationRange`
range
`femaleMutationRate2`
rate 2
`femaleMutationMatrix`
females matrix
`maleMutationModel`
male matrix
`maleMutationRate`
male rate
`maleMutationRange`
male range
`maleMutationRate2`
rate 2
`maleMutationMatrix`
matrix

Value

Locus analysis.

Examples

```
frequencies <- c(0.1, 0.2, 0.3, 0.4)
allelenames <- c("A", "B", "C", "D")
marker <- FamiliasLocus(frequencies, allelenames)
```

`FamiliasPedigree` *FamiliasPedigree: a function for constructing Familias pedigree format.*

Description

This function has been entirely copied from Familias R package, it has been removed from CRAN. All references and citations about this function should be done to Familias developers. More information could be obtained from: https://www.familias.name/index.html. In fbnet package, this function is only used for dealing with the pedigree object.

Usage

```
FamiliasPedigree(id, dadid, momid, sex)
```

Arguments

id	individual id
dadid	father id
momid	mother id
sex	biological sex

Value

A dataframe with probabilities.

Examples

```
persons <- c("mother", "child", "AF")
sex <- c("female", "female", "male")
ped1 <- FamiliasPedigree(id = persons, dadid = c(NA, "AF", NA),momid = c(NA, "mother", NA), sex=sex)
```

---

fbnet *fbnet: Forensic Bayesian Networks*

Description

'fbnet' is an open source software package written in R statistical languaje. It relies on a Bayesian Networks framework doi:10.1017/CBO9780511811357. It is particularly well suited to efficiently perform large-size queries against databases of missing individuals. It could interact with the main functionallities of other packages for pedigree analysis. In particular, 'fbnet' imports the 'Familias' software doi:10.1016/S03790738(00)00147X. In addition 'pedtools', a software for creating and manipulating pedigrees and markers, is supported. 'fbnet' allows computing LRs and obtaining genotype probability distributions for query individual, based on the pedigree data. 'fbnet' implements the complete GENis functionality, a recently published open-source multi-tier information system developed to run forensic DNA databases to perform kinship analysis based on DNA profiles doi:10.1016/j.fsir.2020.100132.

Author(s)

Maintainer: Franco Marsico <franco.lmarsico@gmail.com>

Authors:

- Ariel Chernomoretz <ariel@df.uba.ar>

See Also

Useful links:

- https://marsicofl.github.io/fbnet/
- https://github.com/MarsicoFL/fbnet
- Report bugs at https://github.com/MarsicoFL/fbnet/issues

from_pedtools *Initialize a Bayesian network from a pedtools pedigree*

Description

Builds the list structure returned by initBN directly from a pedtools::ped object, without routing through the archived paramlink package. This is the recommended entry point for modern workflows that construct pedigrees via pedtools, forrel, or pedsuite helpers such as linearPed, cousinPed, avuncularPed, or nuclearPed.

Usage

```
from_pedtools(x, available = NULL, QP = NULL)
```

Arguments

x	A pedtools::ped object. Must have markers attached (via pedtools::setMarkers) with allele frequencies defined.
available	Character or integer vector of IDs that are genotyped (have non-missing evidence). If NULL (the default), the set of typed individuals is inferred automatically from the pedigree: an individual is considered available if they have at least one non-missing genotype across the attached markers.
QP	Character or integer vector of Query Person IDs (the individuals whose genotype posterior we want to query). Used by buildBN downstream. Can be NULL and supplied later.

Details

The paramlink package was archived from CRAN in 2022, and its replacement pedtools uses a different in-memory representation. The original initBN.fromPed function requires a paramlink linkdat object, which makes it difficult to use fbnet with modern pedigree workflows. from_pedtools constructs the equivalent list structure by hand from a pedtools::ped object, so that the rest of the fbnet pipeline (buildBN, buildCPTs, imposeEvidence, etc.) can be run on pedigrees built with modern tools such as forrel and pedsuite.

Value

A list with the same shape as the return value of initBN, namely

ped An auxiliary pedigree object with fields pedigree, ped, orig.ids, markerdata, available, and nMark, mimicking the subset of the paramlink linkdat interface that downstream fbnet functions consume.

markerEvidence A character matrix with rows indexed by typed individuals and columns <marker>_p / <marker>_m giving the observed allele labels.

markerLinkage A data frame with columns linkedTo and recomb; defaults to unlinked.

alelFreq A named list of allele frequency vectors, one per marker.

mmodel A nested list with the default no-mutation model for each sex.

Note

This function handles the *entry point*. Some downstream fbnet functions still use `@import paramlink` in their roxygen declarations, so a fully `paramlink`-free fbnet pipeline may still require additional refactoring. For basic use cases (single marker, small pedigree), the return value of `from_pedtools` can be passed directly to `buildBN`.

See Also

`initBN`, `initBN.fromPed`, `buildBN`.

Examples

```
## Not run:
  library(pedtools)
  library(forrel)

  # Build a 2nd-degree grandparent-grandchild pedigree with one marker
  ped <- linearPed(2)
  m <- marker(ped, afreq = c(A = 0.3, B = 0.4, C = 0.3), name = "M1")
  ped <- setMarkers(ped, m)
  ped <- setGenotype(ped, id = "1", marker = 1, geno = "A/B")

  pbn <- from_pedtools(ped)
  str(pbn)

## End(Not run)
```

`getConditional` *getConditional: a function for obtaining the coditional probability tables based on a given evidence.*

Description

getConditional: a function for obtaining the coditional probability tables based on a given evidence.

Usage

```
getConditional(lf)
```

Arguments

`lf`	A list of joint probabilities.

Value

A list of conditioned probabilities.

`getGenotypeTables` *getGenotypeTables: a function for obtaining genotypetables after variable elimination and using available genetic evidence.*

Description

getGenotypeTables: a function for obtaining genotypetables after variable elimination and using available genetic evidence.

Usage

```
getGenotypeTables(bn, resQ, geno = NULL, lqp = NULL)
```

Arguments

`bn`	A bayesian network for pedigree object with information of the genotyped members. The ped object must be in Familias format.
`resQ`	List of CPTs.
`geno`	data.frame with genotypes.
`lqp`	list of individuals genotypes.

Value

A dataframe with genotype probabilities.

`getLocusCPT` *getLocusCPT: a function for obtaining the coditional probability table from a specific locus.*

Description

getLocusCPT: a function for obtaining the coditional probability table from a specific locus.

Usage

```
getLocusCPT(bn, locus, lumpingParameter = NULL, renorm = "row-wise")
```

Arguments

`bn`	A bayesian network for pedigree object with information of the genotyped members. The ped object must be in Familias format.
`locus`	Specified locus.
`lumpingParameter`	Used for stepwise mutational model.
`renorm`	If "row-wise" is selected, zero probability is assigned for transitions out of range.

Value

A bayesian network based on pedigree evidence and QP definition.

Examples

```
## Not run:
  library(pedtools); library(forrel)
  ped  <- linearPed(2)
  ped  <- setMarkers(ped, locusAttributes = NorwegianFrequencies[1:2])
  ped  <- profileSim(ped, N = 1, ids = 2, seed = 1)
  pbn  <- from_pedtools(ped, QP = "5")
  bnet <- buildBN(pbn, QP = "5")
  bn1  <- buildCPTs(bnet)
  locCPT <- getLocusCPT(bn1, "D3S1358")

## End(Not run)
```

---

getMAP *factorHeteroFounders: a function for multiplying probabilities in case of heterocigote founders.*

Description

factorHeteroFounders: a function for multiplying probabilities in case of heterocigote founders.

Usage

```
getMAP(resQ, topn = 3)
```

Arguments

resQ	List of CPTs.
topn	Format parameter.

Value

A MAP from the probability table.

`getQSetRMP` *getGenotypeTables: a function for obtaining genotypetables after variable elimination and using available genetic evidence.*

Description

getGenotypeTables: a function for obtaining genotypetables after variable elimination and using available genetic evidence.

Usage

```
getQSetRMP(bn, lqp)
```

Arguments

`bn`	A bayesian network for pedigree object with information of the genotyped members. The ped object must be in Familias format.
`lqp`	list of individuals genotypes.

Value

A dataframe with genotype probabilities.

`getValuesOut` *getValuesOut: a function for getting out variables with zero probability in the bayesian network*

Description

getValuesOut: a function for getting out variables with zero probability in the bayesian network

Usage

```
getValuesOut(cpt, condVar = c())
```

Arguments

`cpt`	conditional probability table from the bayesian network
`condVar`	variables from the conditioning table

Value

A processed conditional probability table

`imposeEvidence` *imposeEvidence: a fuction for imposing evidence in the bayesian network.*

Description

imposeEvidence: a fuction for imposing evidence in the bayesian network.

Usage

```
imposeEvidence(bn)
```

Arguments

`bn`	A bayesian network (output of buildBN function).

Value

A preprocessed bayesian network.

`initBN` *initBN: a function to initialize the bayesian network.*

Description

initBN: a function to initialize the bayesian network.

Usage

```
initBN(ped = NULL, bplotped = FALSE)
```

Arguments

`ped`	A ped object with information of the genotyped members. The ped object must be in Familias format.
`bplotped`	An alternative ped object to be compared.

Value

A bayesian network.

Examples

```
## Not run:
  # Legacy paramlink-based entry point. Use from_pedtools() instead.
  pbn  <- initBN(toyped)

## End(Not run)
```

`initBN.fromPed` *initBN.fromPed: a function to initialize the bayesian network.*

Description

initBN.fromPed: a function to initialize the bayesian network.

Usage

```
initBN.fromPed(ped, bplotped)
```

Arguments

`ped`	A ped object in Familias format.
`bplotped`	An alternative ped object to be compared.

Value

A bayesian network.

`initBN.fromVars` *initBN.fromVars: a function to initialize the bayesian network.*

Description

initBN.fromVars: a function to initialize the bayesian network.

Usage

```
initBN.fromVars(bplotped)
```

Arguments

`bplotped`	An alternative ped object to be compared.

Value

A bayesian network.

`minOrdering` *minOrdering: a function for getting an ordering of bayesian network variables not in Q using min fill criteria on interaction graphs.*

Description

minOrdering: a function for getting an ordering of bayesian network variables not in Q using min fill criteria on interaction graphs.

Usage

```
minOrdering(bn, vars = NULL, method = c("min_degree", "min_fill")[1])
```

Arguments

`bn`	A bayesian network for pedigree object with information of the genotyped members. The ped object must be in Familias format.
`vars`	Subset of tables where the order is calculated
`method`	Elimination method, min_degree or min_fill

Value

A bayesian network after ordering process.

`pbn` *Prepared pedigree for bayesian network trimming.*

Description

Prepared pedigree for bayesian network trimming.

Usage

```
pbn
```

Format

A data.matrix.

preparePed *preparePed: a function for simulating genetic data from untyped individuals conditioned on known genotypes.*

Description

preparePed: a function for simulating genetic data from untyped individuals conditioned on known genotypes.

Usage

```
preparePed(ped, available, lLociFreq, rseed = NULL)
```

Arguments

ped	A ped object with information of the genotyped members. The ped object must be in Familias format.
available	Genotyped individuals IDs.
lLociFreq	Allele frequencies.
rseed	Seed used for simulations.

Value

A ped object.

prodFactor *prodFactor: a function for performing product between probability tables.*

Description

prodFactor: a function for performing product between probability tables.

Usage

```
prodFactor(laux)
```

Arguments

laux	probability distribution aux

Value

A dataframe with probabilities.

pruneNodes *pruneNodes: a fuction for clasical pruning in bayesian networks.*

Description

pruneNodes: a fuction for clasical pruning in bayesian networks.

Usage

```
pruneNodes(bn)
```

Arguments

bn	A bayesian network (output of buildBN function).

Value

A preprocessed bayesian network.

---

removeEvidenceFromPed *removeEvidenceFromPed: a function for removing evidence from specific individuals in a ped object.*

Description

removeEvidenceFromPed: a function for removing evidence from specific individuals in a ped object.

Usage

```
removeEvidenceFromPed(pped, idNotEv)
```

Arguments

pped	A ped object with information of the genotyped members. The ped object must be in Familias format.
idNotEv	A set of individuals whom evidence should be removed.

Value

A ped object.

`reportLR` *reportLR: a function for calculating the LRs of specified genotypes in a pedigree.*

Description

reportLR: a function for calculating the LRs of specified genotypes in a pedigree.

Usage

```
reportLR(bn, resQ, geno = NULL)
```

Arguments

bn	A bayesian network for pedigree object with information of the genotyped members. The ped object must be in Familias format.
resQ	List of CPTs.
geno	data.frame with genotypes.

Value

A dataframe with LRs.

`reportPQ` *reportPQ: a function for calculating the probability of specified genotypes in a pedigree.*

Description

reportPQ: a function for calculating the probability of specified genotypes in a pedigree.

Usage

```
reportPQ(bn, resQ, geno = NULL)
```

Arguments

bn	A bayesian network for pedigree object with information of the genotyped members. The ped object must be in Familias format.
resQ	List of CPTs.
geno	data.frame with genotypes.

Value

A dataframe with genotype probabilities.

`reverseSplit` *reverseSpit: a function for formatting.*

Description

reverseSpit: a function for formatting.

Usage

```
reverseSplit(inList)
```

Arguments

`inList`	input for formatting.

Value

A bayesian network.

`setOrdering` *setOrdering: a function for selecting the ordering method in the elimination process.*

Description

setOrdering: a function for selecting the ordering method in the elimination process.

Usage

```
setOrdering(bn, ordMethod, vars = NULL, orderElim = NULL)
```

Arguments

`bn`	A bayesian network for pedigree object with information of the genotyped members. The ped object must be in Familias format.
`ordMethod`	Ordering method.
`vars`	Vars
`orderElim`	Order elimination criteria.

Value

A bayesian network after ordering process.

`stateRemoval` *stateRemoval: a function for processing the bayesian network.*

Description

stateRemoval: a function for processing the bayesian network.

Usage

```
stateRemoval(bn)
```

Arguments

`bn`	A bayesian network (output of buildBN function).

Value

A preprocessed bayesian network.

`stateRemoval2` *stateRemoval2: a function for processing the bayesian network. It implements another approach from the described in stateRemoval function.*

Description

stateRemoval2: a function for processing the bayesian network. It implements another approach from the described in stateRemoval function.

Usage

```
stateRemoval2(bn, verbose = FALSE)
```

Arguments

`bn`	A bayesian network (output of buildBN function).
`verbose`	Computation output.

Value

A preprocessed bayesian network.

stateRemovalSubnucs *stateRemovalSubnucs: a fuctiong for variable state pruning.*

Description

stateRemovalSubnucs: a fuctiong for variable state pruning.

Usage

```
stateRemovalSubnucs(bn, verbose = FALSE)
```

Arguments

bn	A bayesian network (output of buildBN function).
verbose	Computation output.

Value

A preprocessed bayesian network.

sumFactor *prodFactor: a function for performing sum between probability tables.*

Description

prodFactor: a function for performing sum between probability tables.

Usage

```
sumFactor(cpt, Z)
```

Arguments

cpt	Conditional probability table
Z	factor

Value

A dataframe with probabilities.

toybase *Toy allele frequency database.*

Description

Toy allele frequency database.

Usage

```
toybase
```

Format

A data frame two markers allele frequencies

toyped *STRs allelic frequencies from specified country.*

Description

STRs allelic frequencies from specified country.

Usage

```
toyped
```

Format

A toy pedigree. Nuclear family.

velim.bn *velim.bn: a function for variable elimination in a bayesian network.*

Description

velim.bn: a function for variable elimination in a bayesian network.

Usage

```
velim.bn(
  bn,
  ordMethod = c("id", "min_degree", "min_fill", "fixed")[2],
  orderElim = NULL,
  verbose = FALSE
)
```

Arguments

bn	A bayesian network for pedigree object with information of the genotyped members. The ped object must be in Familias format.
ordMethod	Selected ordering method between id, min_degree, min_fill and fixed.
orderElim	Elimination order.
verbose	Computation output.

Value

Variable elimination result.

Examples

```
## Not run:
  library(pedtools); library(forrel)
  ped  <- linearPed(2)
  ped  <- setMarkers(ped, locusAttributes = NorwegianFrequencies[1:2])
  ped  <- profileSim(ped, N = 1, ids = 2, seed = 1)
  pbn  <- from_pedtools(ped, QP = "5")
  bnet <- buildBN(pbn, QP = "5")
  bn1  <- buildCPTs(bnet)
  resQ <- velim.bn(bn1, ordMethod = "min_fill", verbose = FALSE)

## End(Not run)
```

Index

∗ **datasets**
  Argentina_STRs, 3
  bnet, 3
  pbn, 17
  toybase, 24
  toyped, 24

Argentina_STRs, 3

bnet, 3
buildBN, 3, *10, 11*
buildCPTs, 4, *10*

convertPedformat, 5

evidencePrunning, 6

factorHeteroFounders, 6
FamiliasLocus, 7
FamiliasPedigree, 8
fbnet, 9
fbnet-package *(*fbnet*)*, 9
from_pedtools, 10

getConditional, 11
getGenotypeTables, 12
getLocusCPT, 12
getMAP, 13
getQSetRMP, 14
getValuesOut, 14

imposeEvidence, *10*, 15
initBN, *10, 11*, 15
initBN.fromPed, *10, 11*, 16
initBN.fromVars, 16

minOrdering, 17

pbn, 17
preparePed, 18
prodFactor, 18
pruneNodes, 19

removeEvidenceFromPed, 19
reportLR, 20
reportPQ, 20
reverseSplit, 21

setOrdering, 21
stateRemoval, 22
stateRemoval2, 22
stateRemovalSubnucs, 23
sumFactor, 23

toybase, 24
toyped, 24

velim.bn, 24