Y-chromosomal analysis tools: logic flow document

Abstract

The aim of this programme is to take literature and consumer Y-DNA data from genetic testing companies, and appropriate meta-data from individuals' genealogical histories, and translate them into a geographically and temporally encoded phylogenic tree.

This document describes the coding logic behind the workflow. It is an evolving document designed to communicate and expedite the workflow, rather than a formal description of how the processes work.

Author

Iain McDonald, 26 Jan 2018

Key

Standard flowchart terminology is used for the most part. In addition:

RED items are things that need attention

PURPLE items are data obtained from other sources

YELLOW items are queries to the file system

GREEN items are database entries

BLUE items are storage lists kept in RAM by the programme

Overall programme structure

There are five main challenges to overcome, corresponding to the six work pages listed:

Work Package A: How can we take this diverse dataset and create a manageable and standardised series of calls from it?

Work Package B: How can we parse these calls to create the best haplogroup tree?

Work Package C: Can we chart the progression of Y-STR mutations and perform haplogroup predictions on testers with or without any non-NGS SNP testing?

Work Package D: Can we map the movement of people across Europe using the paper origins of individuals and ancient DNA results?

Work Package E: Can we assign ages to these haplogroups (and by extension the migrations they trace)? How well can we make these line up with known historical events? How can we interface these outputs with those of others?

Work Package F: What can we provide by way of informative and useful outputs?

Work Package A

The primary difficulties here involve managing the large variety of possible input files, the diversity of their respective information, and the differences in their quality standards. The large disc space of these datasets is also an issue, so clearly unimportant variant candidates need to be carefully rejected first. Typical datasets might include:

FASTQ - raw sequences of reads that need aligned to a reference genome (e.g. GRCh38), sources may include literature data or ancient DNA. These may be large fractions of a GB in size.

BAM - aligned data sequences that need called for variants. Sources may include raw BigY data and are typically ~500 MB.

BigY Build 38 VCF - variant call files from the original BAMs. These should have accompanying coverage files (BED) in which regions we are ok to say a variant is passed. The BED regions also define the count for the age analysis. Typical sizes are $\sim 15 / 80$ MB zipped/unzipped, and we anticipate thousands of these files.

BigY Build 37 - any legacy data from the old Build 37 data (mapped to GRCh37 / hg19). BAM files can be split back to FASTQ files and remapped to Build 38. VCFs can be processed using "liftOver", which is a simpler co-ordinate conversion. However, this does not allow for better mapping of reads in Build 38, so is not ideal. Size is 1-2 MB zipped.

FGC YElite / WGS - BED files need generated for these using callableLoci in GATK. This is something FGC can do. Variants are stored in gtype files. YSeq WGS - the Y chromosomal (chrY) data needs extracted from the results, then processed. Status of VCF/BED extraction is currently unclear.

BigY CSV, SNP pack tests, individual YSeq/FTDNA SNP tests, Chromo2 and literature data - these are essentially a list of variants for which a test is positive. Sometimes there are lists of negatives too. Rarely are the information on negative versus null. These data are unsuitable for some analyses (e.g. age analysis) and have limited input into the haplotree, where they would be expected to be added last.

Variants from these inputs will need to be ingested for later comparison, along with associated meta-data like surnames, ancestral origins and STR results. Other STR results, without SNP results, are desirable for later geographical analysis, where they will provide additional datapoints.

Work Package B

Once ingested and homogenised, this data needs to be parsed to form a haplotree. The difficulties here are minimising the amount of data that needs processed at one time, in order to reduce RAM requirements; and dealing with low quality and missing (uncalled) results.

Parsing will typically require sorting the data into groups of people and clades (haplogroups) of variants to form "blocks". Missing calls will either need to be presumed positive or negative, based foremost on logic, but also on low-quality results in NGS data.

The resulting tree structure is used as input to WPs C, D and E, each of which needs to be capable of running from this input alone, or by including data from the others. Other outputs may compare the output haplotree to those from other sites (e.g. FTDNA's haplotree, ISOGG) to identify where updates need to be made.

Work Package C

This package centres around making use of STR results: both those from the NGS tests and STR-specific tests at the various companies (predominantly FTDNA but also YSeq). Using the NGS test results, we can attempt generation of the ancestral STRs for that clade. This will let us see how STRs have varied across history, and identify the key mutations present in each haplogroup. A further stage of this would be haplogroup estimation for non-NGS-tested individuals.

Not all haplogroups will have a corresponding, stable, identifiable STR mutation. Hence this is only going to be possible for some levels in the tree, and must be an approximate science. Haplogroup estimation can therefore only be done in an approximate sense, but it should provide enough useful data to use in WPs D and E.

Work Package D

This package tries to connect haplogroups to migrations by looking at how the distribution of testers within that haplogroup deviates from the weighted average. This must be done on a clade-by-clade basis to avoid founder effects (e.g. the origin of R-M269 is in Russia, but most R-M269 is now in Europe, due to a founder effect at the R-L11 level).

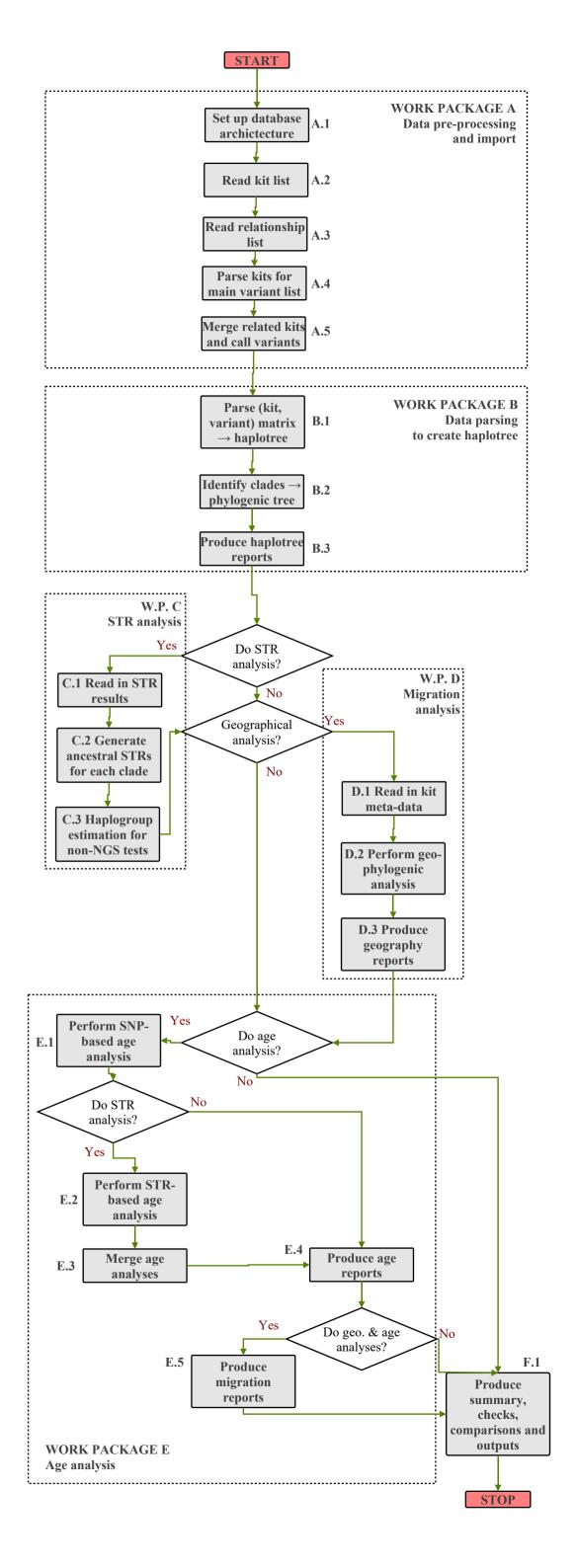
Difficulties here involve getting a sufficient number of diverse members for each clade (hence why the haplogroup estimation in WP C is important). The data are also heavily biased, since some European regions (e.g. the British Isles) have populations and diaspora that are much more widely tested than other countries (e.g. France, Eastern Europe). Weighting entries to debias them is important, as well as correctly weighting sub-clades and ancient DNA results.

Work Package E

This package performs the age analysis, to put each haplogroup in an historical context. The basic principles behind this are effectively worked out, but take a lot of CPU power. STR-based age analysis can be done fairly easily, but the results will need carefully calibrated and cross-checked for consistency. Treatment of uncertainties is also very important. Most of the problems here are likely to be mathematical in nature.

Work Package F

A range of outputs from this data can be considered, from a list of ages and co-ordinates, to bespoke queries on individual tests (haplogroup estimation, STR mutation timeline, migration pathway), to animations of migrations across Europe.



Item A.1: Database architecture

This step sets up the initial database structures for the programme (or imports them), along with any parametric setup files and data pointers.

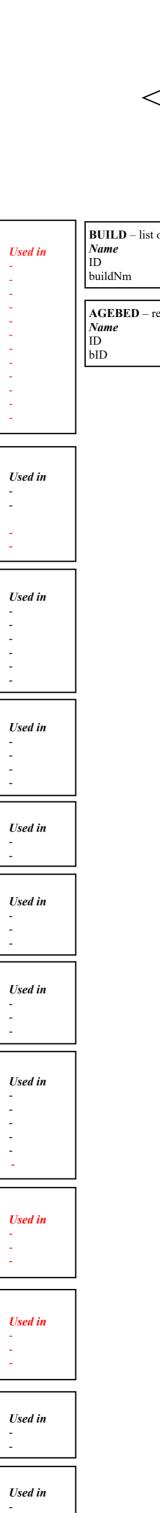
Uses global parameters: Boolean: Drop all tables and remake database ResetDB

External inputs: User parameter file

Database tables

The following tables describe the database structures. Grey fields are currently unused. Red fields are not in the current (24 Jan 2018) schema.

- Little Oit Printing	list of people				ALLVARIANTS -	positions and counts for every entry present in	VCF files
Name	Type	4		d in	Name	Type	Descri
ID firstName	INTEGER PRIMARY KEY TEXT DEFAULT NULL		A.4 - A.4 -		pID	INTEGER REFERENCES dataset(ID)	Person
middleName	TEXT DEFAULT NULL		A.4 -		buildID pos	INTEGER REFERENES build(ID) INTEGER	Build Positio
surname	TEXT DEFAULT NULL		A.4 -		refalleleID	INTEGER REFERENCES alleles(ID)	Refere
maidenName	TEXT DEFAULT NULL		A.4 -		deralleleID	INTEGER REFERENCES alleles(ID)	Derive
yHaplogroupId mtHaplogroupId		Most-recent Y-DNA hg Most-recent mt-DNA hg	В -		npos	INTEGER	# posit
sortidx	INTEGER		В -		nneg nmix	INTEGER INTEGER	# negat # mixe
		•			fpos	INTEGER	# failed
DATABASE – prima Name	ary list of kits <i>Type</i>	Description	Filled in	Used in	fneg fmix	INTEGER INTEGER	# failed # failed
ID	INTEGER PRIMARY KEY	-	A.2	-	bneg	INTEGER DEFAULT NULL	# - call
seq	INTEGER DEFAULT 0	Order files arbitrarily	В	-			
kitName DNAID	TEXT INTEGER references person(ID)	A name for the dataset Whose DNA is this?	A.2,A.4 A.2,A.4	- A.4	VCECALLS antri	es present in each VCF file	
contact	TEXT	Holder's contact info	A.2	-	Name	Type	Descri
kitId	TEXT [INDEX fileidx]	Lab ID	A.2	-	pID	INTEGER REFERENCES dataset(ID)	Person
surnameID countryID	INTEGER references surname(ID) INTEGER references country(ID)	MDKA surname MDKA country	A.2,A.4 A.2,A.4	-	vID	INTEGER REFERENCES variants(ID)	VCF e
birthYr	INTEGER DEFAULT NULL	MDKA birth year	A.2,A.4	-	alleleID	[INDEX vcfidx] INTEGER REFERENCES alleles(ID)	Allele
labID	INTEGER references lab(id)	Testing lab	A.2	-	ruleID	INTEGER REFERENCES rules(ID)	Reason
testTypeID buildID	INTEGER REFERENCES testtype(I INTEGER references build(ID)	(D) Type of DNA test Reference build for the te	A.2 st A.2	-			
importDt	TEXT	Date originally imported	A.2 A.2	-	VCFSTATS – repor	ted statistics on VCF files	
fileNm	TEXT	Normalized file/path nam	e A.2	-	Name	Type	Descri
origFileNm	TEXT	Original file name from u		-	pID	INTEGER REFERENCES dataset(ID) INTEGER	Person
otherInfo normalOrigID	TEXT INTEGER references origin(ID)	Freeform text entered by Normalized ancestral orig		-	ny nv	INTEGER	-
lat	TEXT	Ancestral location latitude		-	ns	INTEGER	-
lng	TEXT	Ancestral location longitu	ide A.2,A.4	-	ni	INTEGER	-
policyVer accessToken	TEXT DEFAULT NULL	Data policy agreed to	A.2	-	nr	INTEGER	-
updated	TEXT DEFAULT NULL TEXT	Date the record was update	ted A.2	-	DEDCTATE roman	tad statistics on PED files	
approxHg	TEXT	Haplogroup assigned by I		-	Name	ted statistics on BED files Type	Description
					pID	INTEGER REFERENCES dataset(ID)	Person
SURNAME – list of	surnames				coverage1	INTEGER	Total cover
Name	Type	Description	Filled in	Used in	coverage2 nranges	INTEGER INTEGER	Cov. for ag
ID surname	INTEGER PRIMARY KEY TEXT	- Name	A.2 A.2	-	litaliges	INTEGER	π chiles
surname	ILAI	rame	A.Z		META – meta-data	for the entire data set or run	
TESTTYPE – list of	e tests				Name	Type	Descri
Name	Type	Description	Filled in	Used in	desc val	TEXT TEXT	-
ID	INTEGER PRIMARY KEY		A.2	-	Vai	IEAI	
testNm description	TEXT TEXT	The name of the test Further description	A.2 A.2	-	STRS – table for ST	R definitions	
isNGS	BOOLEAN	Is it a sequencing test?	A.2 A.2	-	Name	Туре	Description
tagNm	TEXT	Field in hg-R schema	-	-	ID	INTEGER PRIMARY KEY	-
priority	INTEGER	Field in hg-R schema	-	-	strname ordering	TEXT INTEGER	Name Put STRs in
CONTIG – defines r	regions of interest				1 — —		
Name	Type	Description	Filled in	Used in	STRCALLS – tester		
ID	INTEGER PRIMARY KEY	-	A.1	-	Name pID	Type INTEGER REFERENCES people(ID)	Description Person
buildID	INTEGER references build(ID)	Reference build, e.g. hg38		-	sID	INTEGER REFERENCES strs(ID)	STR
description length	TEXT INTEGER	e.g. chrY e.g. 57227415	A.1 A.1	-	val	INTEGER	Allele
					VADIANTS list of	f known variants [INDEX varidx on buildID,p	og1
COUNTRY – list of		Dominio di con	EU . J i	T7	Name	Type	Description
<i>Name</i> ID	<i>Type</i> INTEGER PRIMARY KEY	Description -	Filled in A.2	Used in	ID	INTEGER PRIMARY KEY	- 1
country	TEXT	Name	A.2	-	buildID	INTEGER REFERENCES build(ID) INTEGER	Build Position
-] pos anc	INTEGER INTEGER REFERENCES alleles(ID)	Ancestral all
ORIGIN – list of ori	gins				der	INTEGER REFERENCES alleles(ID)	Derived alle
Name	<i>Type</i> INTEGER PRIMARY KEY	Description	Filled in	Used in	sortidx	INTEGER	Haplotree so
		-	A.2	-			
		Name	A.2		I I		
	TEXT	Name	A.2		_	counts of numbers of calls made under each r	
ID origin LAB – list of labs		Name	A.2		Name	Type	ule Description
origin LAB – list of labs Name	TEXT Type	Name Description	Filled in	Used in	Name vID	<i>Type</i> INTEGER REFERENCES variants(ID)	
origin LAB – list of labs Name ID	TEXT Type INTEGER PRIMARY KEY	Description -	Filled in A.2	Used in	Name	Type	
origin LAB – list of labs Name ID	TEXT Type		Filled in	Used in - -	Name vID rID	Type INTEGER REFERENCES variants(ID) INTEGER REFERENCES rule(ID)	
CAB – list of labs Name ID labNm	TEXT Type INTEGER PRIMARY KEY	Description - Name	Filled in A.2	Used in - -	Name vID rID count	Type INTEGER REFERENCES variants(ID) INTEGER REFERENCES rule(ID) INTEGER	
LAB – list of labs Name ID labNm BEDRANGES – min Name	TEXT Type INTEGER PRIMARY KEY TEXT nimum and maximum ranges in BED f	Description - Name file entries Description	Filled in A.2 A.2 Filled in Use	Used in - - d in	Name vID rID count	Type INTEGER REFERENCES variants(ID) INTEGER REFERENCES rule(ID) INTEGER INTEGER INTEGER	Description
origin LAB – list of labs Name ID labNm BEDRANGES – min Name ID	TEXT Type INTEGER PRIMARY KEY TEXT nimum and maximum ranges in BED f Type INTEGER PRIMARY KEY	Description - Name file entries Description	Filled in A.2 A.2 Filled in Use	-	Name vID rID count VARIANTQUALIT Name vID	Type INTEGER REFERENCES variants(ID) INTEGER REFERENCES rule(ID) INTEGER IY – quality flags for variants Type INTEGER REFERENCES variants(ID)	Description Description -
LAB – list of labs Name ID labNm BEDRANGES – min Name ID minaddr	TEXT Type INTEGER PRIMARY KEY TEXT nimum and maximum ranges in BED f Type INTEGER PRIMARY KEY	Description - Name file entries Description - A Position of minimum	Filled in A.2 A.2 Filled in Use	-	Name vID rID count VARIANTQUALIT Name vID qualsum	Type INTEGER REFERENCES variants(ID) INTEGER REFERENCES rule(ID) INTEGER TY – quality flags for variants Type INTEGER REFERENCES variants(ID) FLOAT	Description Call quality
origin LAB – list of labs Name ID labNm BEDRANGES – min Name ID minaddr maxaddr	Type INTEGER PRIMARY KEY TEXT mimum and maximum ranges in BED f Type INTEGER PRIMARY KEY INTEGER [INDEX rangeidx] INTEGER [INDEX rangeidx2]	Description - Name file entries Description - A Position of minimum	Filled in A.2 A.2 Filled in Use A.5 - A.5	-	Name vID rID count VARIANTQUALIT Name vID	Type INTEGER REFERENCES variants(ID) INTEGER REFERENCES rule(ID) INTEGER IY – quality flags for variants Type INTEGER REFERENCES variants(ID)	Description Description -
origin LAB – list of labs Name ID labNm BEDRANGES – min Name ID minaddr maxaddr BED – entries presen	Type INTEGER PRIMARY KEY TEXT mimum and maximum ranges in BED f Type INTEGER PRIMARY KEY INTEGER [INDEX rangeidx] INTEGER [INDEX rangeidx2]	Description - Name file entries Description - Position of minimum Position of maximum	Filled in A.2 A.2 Filled in Use A.5 - A.5	- - d in	Name vID rID count VARIANTQUALIT Name vID qualsum qualpos	Type INTEGER REFERENCES variants(ID) INTEGER REFERENCES rule(ID) INTEGER IY – quality flags for variants Type INTEGER REFERENCES variants(ID) FLOAT BOOLEAN	Description Call quality
origin LAB – list of labs Name ID labNm BEDRANGES – min Name ID minaddr maxaddr BED – entries presen Name	TEXT Type INTEGER PRIMARY KEY TEXT nimum and maximum ranges in BED f Type INTEGER PRIMARY KEY INTEGER [INDEX rangeidx] INTEGER [INDEX rangeidx2] at in each BED file Type	Description - Name file entries Description - Position of minimum Position of maximum Description	Filled in A.2 A.2 Filled in Use A.5 - A.5 - A.5 - C.5 - C.5	-	Name vID rID count VARIANTQUALIT Name vID qualsum qualpos ALLELES – list of a	Type INTEGER REFERENCES variants(ID) INTEGER REFERENCES rule(ID) INTEGER TY – quality flags for variants Type INTEGER REFERENCES variants(ID) FLOAT BOOLEAN alleles in sequencing calls	Description Call quality Tree pos que
LAB – list of labs Name ID labNm BEDRANGES – min Name ID minaddr maxaddr BED – entries presen Name pID	Type INTEGER PRIMARY KEY TEXT mimum and maximum ranges in BED f Type INTEGER PRIMARY KEY INTEGER [INDEX rangeidx] INTEGER [INDEX rangeidx2]	Description - Name file entries Description - Position of minimum Position of maximum Description - Description - A	Filled in A.2 A.2 Filled in Use A.5 - A.5	- - d in	Name vID rID count VARIANTQUALIT Name vID qualsum qualpos	Type INTEGER REFERENCES variants(ID) INTEGER REFERENCES rule(ID) INTEGER IY – quality flags for variants Type INTEGER REFERENCES variants(ID) FLOAT BOOLEAN	Description Call quality
Drigin LAB – list of labs Name ID abNm BEDRANGES – min Name ID minaddr maxaddr BED – entries presen Name DID	Type INTEGER PRIMARY KEY TEXT mimum and maximum ranges in BED f Type INTEGER PRIMARY KEY INTEGER [INDEX rangeidx] INTEGER [INDEX rangeidx2] at in each BED file Type INTEGER REFERENCES dataset(III	Description - Name file entries Description - Position of minimum Position of maximum Description Description A Description Description Description Description Description	Filled in A.2 A.2 Filled in Use A.5 - A.5 - A.5 - Filled in Use A.5 - C.5	- - d in	Name vID rID count VARIANTQUALIT Name vID qualsum qualpos ALLELES – list of a	Type INTEGER REFERENCES variants(ID) INTEGER REFERENCES rule(ID) INTEGER IY – quality flags for variants Type INTEGER REFERENCES variants(ID) FLOAT BOOLEAN alleles in sequencing calls Type	Description Call quality Tree pos que
LAB – list of labs Name ID labNm BEDRANGES – min Name ID minaddr maxaddr BED – entries presen Name pID bID TREE – description	Type INTEGER PRIMARY KEY TEXT nimum and maximum ranges in BED f Type INTEGER PRIMARY KEY INTEGER [INDEX rangeidx] INTEGER [INDEX rangeidx2] at in each BED file Type INTEGER REFERENCES dataset(III INTEGER REFERENCES bedranger of the haplogroup tree	Description - Name file entries Description - Position of minimum Position of maximum Description - Person S(ID) BED entry	Filled in A.2 A.2 Filled in Use A.5 - A.5 - A.5 - A.5 - A.5 A.5	- d in	Name vID count VARIANTQUALIT Name vID qualsum qualpos ALLELES – list of a Name ID allele	Type INTEGER REFERENCES variants(ID) INTEGER REFERENCES rule(ID) INTEGER TY – quality flags for variants Type INTEGER REFERENCES variants(ID) FLOAT BOOLEAN alleles in sequencing calls Type INTEGER PRIMARY KEY TEXT	Description Call quality Tree pos que
origin LAB – list of labs Name ID labNm BEDRANGES – min Name ID minaddr maxaddr BED – entries presen Name pID bID TREE – description on Name	Type INTEGER PRIMARY KEY TEXT nimum and maximum ranges in BED f Type INTEGER PRIMARY KEY INTEGER [INDEX rangeidx] INTEGER [INDEX rangeidx2] at in each BED file Type INTEGER REFERENCES dataset(II INTEGER REFERENCES bedrange: of the haplogroup tree Type	Description - Name file entries Description - Position of minimum Position of maximum Description - Person -	Filled in A.2 A.2 A.2 Filled in Uso A.5 - A.5 - A.5 - A.5 - Filled in Uso A.5 - Filled in Uso A.5 - Comparison Filled in Uso Comparison Filled in	- - d in	Name vID count VARIANTQUALIT Name vID qualsum qualpos ALLELES – list of a Name ID allele SNPNAMES – name	Type INTEGER REFERENCES variants(ID) INTEGER REFERENCES rule(ID) INTEGER TY – quality flags for variants Type INTEGER REFERENCES variants(ID) FLOAT BOOLEAN alleles in sequencing calls Type INTEGER PRIMARY KEY TEXT es and aliases associated with variants	Description Call quality Tree pos que Description - e.g. A,C,G,T
LAB – list of labs Name ID labNm BEDRANGES – min Name ID minaddr maxaddr BED – entries presen Name pID bID TREE – description of Name ID	Type INTEGER PRIMARY KEY TEXT nimum and maximum ranges in BED from the file to the file t	Description Name file entries Description Position of minimum Position of maximum Description	Filled in A.2 A.2 A.2 Filled in Use A.5 - A.5 - A.5 - A.5 - Filled in Use B Use C Filled in C Filled i	- d in d in	Name vID count VARIANTQUALIT Name vID qualsum qualpos ALLELES – list of a Name ID allele	Type INTEGER REFERENCES variants(ID) INTEGER REFERENCES rule(ID) INTEGER TY – quality flags for variants Type INTEGER REFERENCES variants(ID) FLOAT BOOLEAN alleles in sequencing calls Type INTEGER PRIMARY KEY TEXT es and aliases associated with variants Type	Description Call quality Tree pos que
origin LAB – list of labs Name ID labNm BEDRANGES – min Name ID minaddr maxaddr BED – entries presen Name pID bID TREE – description of Name ID	Type INTEGER PRIMARY KEY TEXT nimum and maximum ranges in BED f Type INTEGER PRIMARY KEY INTEGER [INDEX rangeidx] INTEGER [INDEX rangeidx2] at in each BED file Type INTEGER REFERENCES dataset(II INTEGER REFERENCES bedrange: of the haplogroup tree Type	Description Name file entries Description Position of minimum Position of maximum Description	Filled in A.2 A.2 A.2 Filled in Use A.5 - A.5 - A.5 - A.5 - Filled in Use B Use C Filled in C Filled i	- d in d in	Name vID count VARIANTQUALIT Name vID qualsum qualpos ALLELES – list of a Name ID allele SNPNAMES – name Name	Type INTEGER REFERENCES variants(ID) INTEGER REFERENCES rule(ID) INTEGER TY – quality flags for variants Type INTEGER REFERENCES variants(ID) FLOAT BOOLEAN alleles in sequencing calls Type INTEGER PRIMARY KEY TEXT es and aliases associated with variants	Description Call quality Tree pos que Description - e.g. A,C,G,T



Filled in

A.4

A.4

A.4

A.4

A.4

A.4

A.4

Filled in

Filled in

Filled in

Filled in

Filled in

Filled in

A.5

A.5 A.5

A.5

A.5

Filled in

Filled in

Filled in

Filled in

A.5

A.5

A.5

A.5

A.5 A.5

A.5 A.5 A.5

A.5

A.5

A.5 A.5

A.5

A.5

A.5 A.5

Person

Position

Reference allele

Derived allele

positive calls

negative calls

failed - calls

Description

VCF entry

Allele called

Reason called

Description

Description

Total coverage

Cov. for age analysis

Description

Ancestral allele

Haplotree sort index

Call quality sum Tree pos questionable

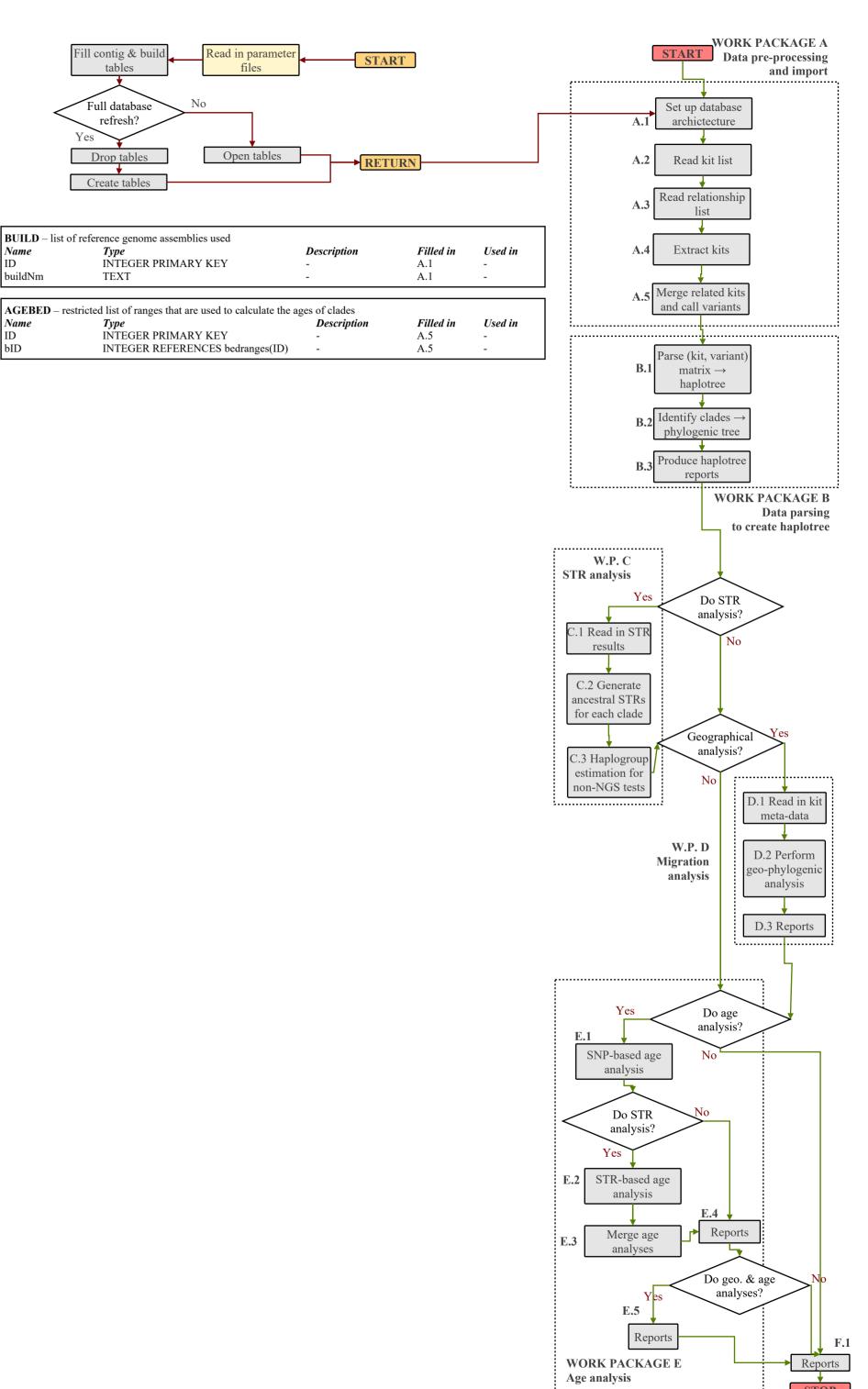
e.g. A,C,G,T,AT,...

Derived allele

Put STRs in order

failed 0/1 calls A.4 # - calls from BED A.5

mixed calls # failed + calls



Item A.2: Read in list of kits

This step identifies whether any new kits are to be read in.

Uses global parameters:

SyncToDB Boolean: synchronise to external database?

External inputs:

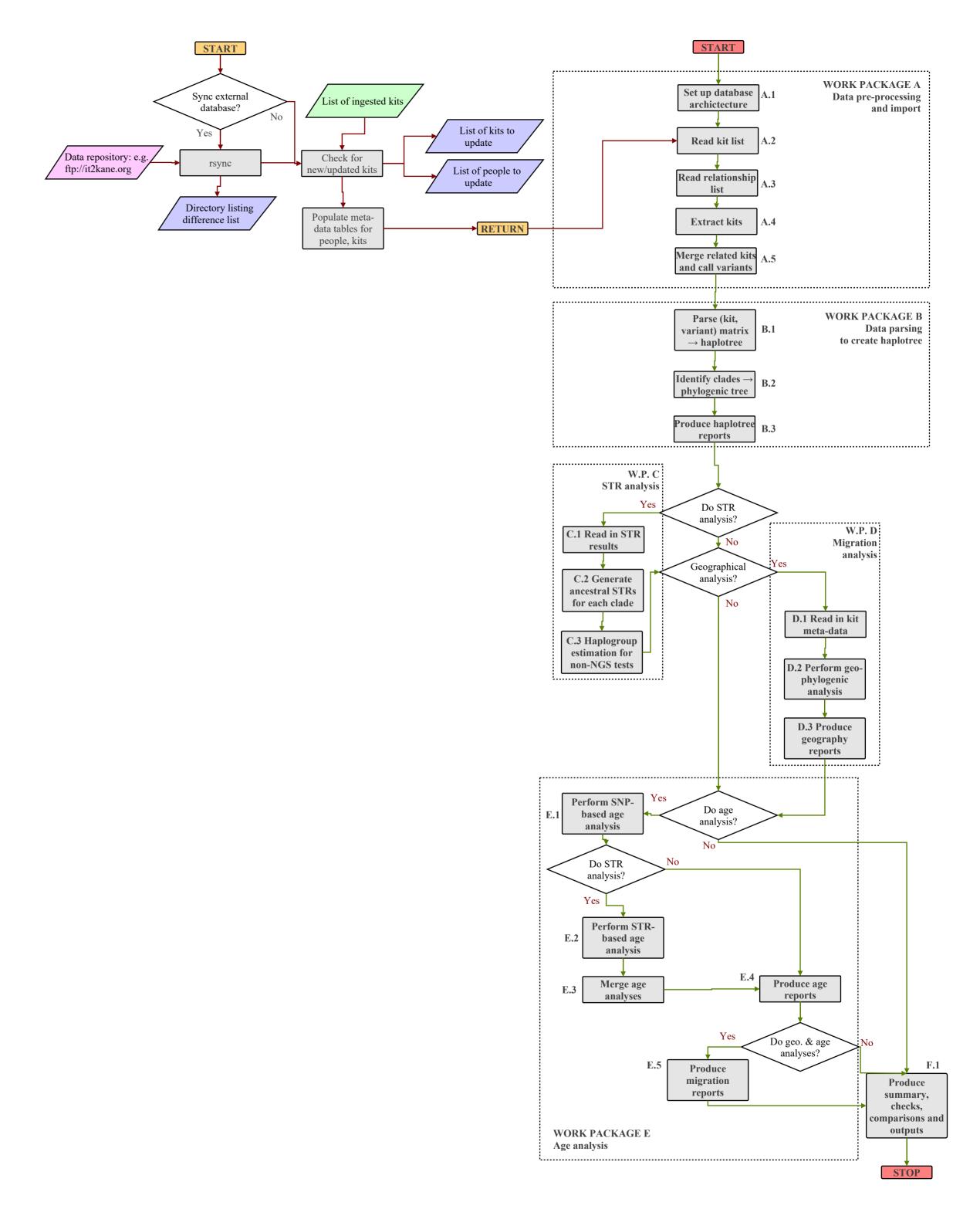
Data repository

Internal inputs:

Internal database kits
Internal database people

Produces:

List of **kits** to update
List of **people** to update



Item A.3: Read in list of relationships

This list identifies and describes the relationships between different kits. Applications are twofold. Firstly, kits belonging to the same person need to be merged for best effect (A.4). Kits

belonging to closely related individuals (e.g. father/son or cousin pairs) may be best merged as well.

Secondly, known relationships become important when performing the age analysis, in

defining limits to relationships. The same list can be used to define the ages of ancient DNA samples.

The input list may be spread over one or more files, and contain three data types:

- MDKA: provides a date (with or without uncertainty) for the kit holder's most-distant known ancestor's birth. Relationships cannot be younger than the latter of two tests' MDKAs.
- DoB: provides a date of birth for a tester, or a probability distribution function (PDF) describing this. This will mainly be useful for carbon-14 dates from ancient DNA results, which may come as arbitrarily complex PDFs.
- Link: a known common ancestor links two tests. This may be two tests from the same person. It may have some uncertainty, and it may be an arbitrarily complex function (e.g. for testers known/suspected to have a common surname origin or emigrant history). It can also be used to set arbitrary boundaries like "clades must be related after the Norman Conquest" or "must be related within the Corded Ware Culture period", etc.

Data formats may be:

```
Type=MDKA, Kit, Date, Unc, LowLimit, UpLimit, Comment
               Must be unique identifier (e.g. sequence number or include company)
Kit
               As common epoch years (CE) [default=zeroage].
Date
               Gaussian uncertainty parameter [default=zeroageunc].
Unc
Low/UpLimit Youngest/oldest possible date [default=currentYear/-inf].
Type=DoB, Kit, Date, Unc, LowLimit, UpLimit, PDFfile, Comment
               [Default=null] Contains arbitrary PDF formatted as:
PDFfile
                         Probability
               Year
Type=Link, Kit1, Kit2, Date, Unc, LowLimit, UpLimit, PDFfile,
Merge?, Comment
               The two kits under consideration
Kit1,2
               Boolean TRUE/FALSE to merge kits into the same person [default=false]
Merge
```

Manual over-rides to table entries can be made at this point too. This allows us to set things such as people's ancestral locations and MDKA information with more accurate information that they have provided but is not available from the primary data sources. Data formats may be:

Table, KitID, PersonID, Field, Data, Comment

```
rable, Ricib, reisonib, Field, baca, commend
```

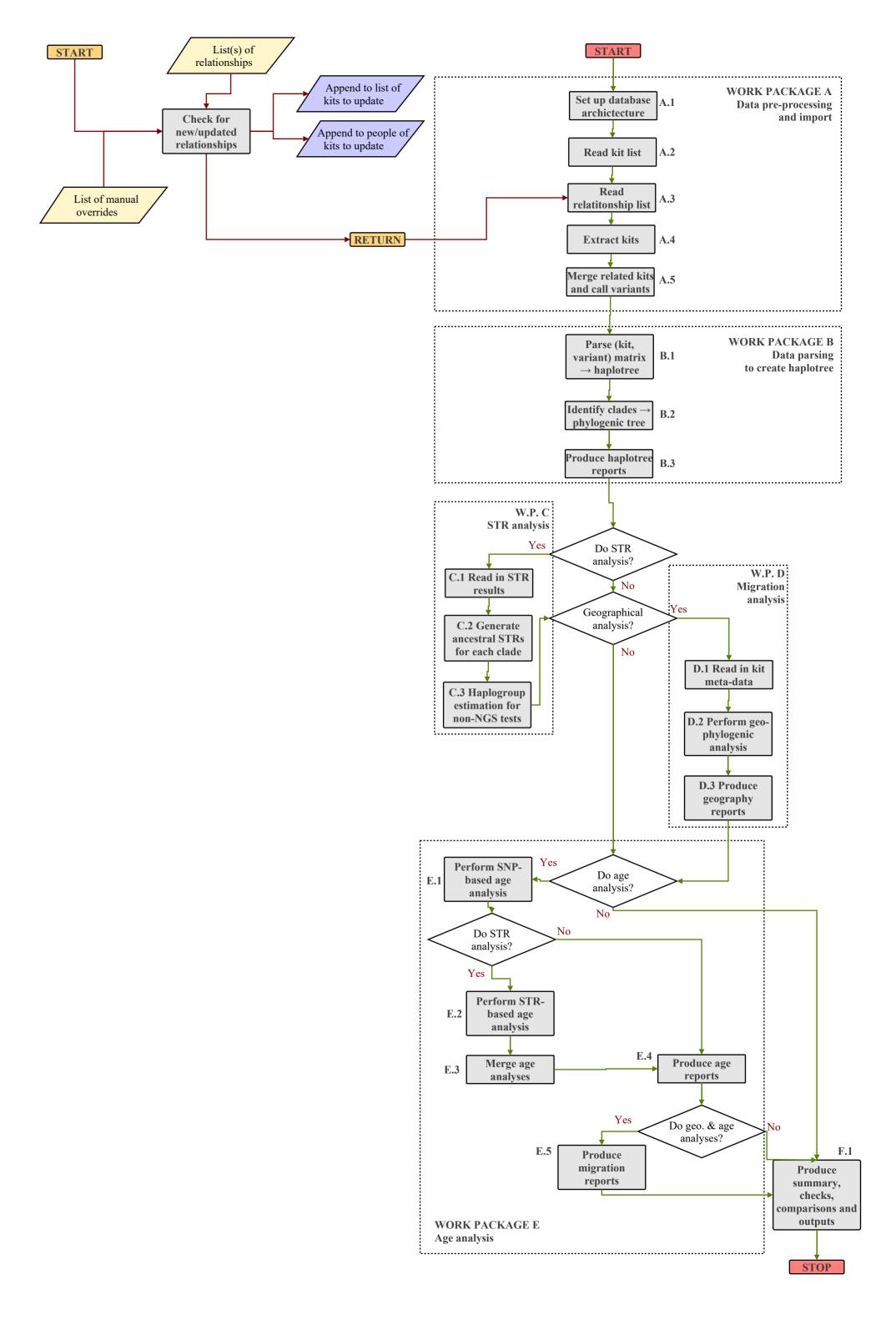
e.g..

Kits, 123456, , mdkaDoB, 1700, IM: updated from E-mail

Uses global parameters:

RelationFile File(s) containing list of relationships zeroage Default DoB of testers (c. 1950).

zeroageunc Associated uncertainty (c. 16 years).



Item A.4: Extract kits

This will be one of the harder parts of this work package to get right, due to the volume of data involved and the diversity of data we have to deal with. This diversity is likely to require specific analyses for each kind of data.

As well as BigY, FGC YElite and WGS, and YSeq WGS tests, we may have to deal with arbitrary lists of calls from CSV calls from BigY, National Geographic, Chromo2, YSeq, literature etc. These may be presented as hg36, hg37 or hg38 positions, or as named SNPs.

BAM and FASTQ files from literature sources may include modern or ancient DNA. Initial quality filters for ancient DNA will likely need to be lowered.

Extraneous data (e.g. negative calls, failed calls, quality data) cannot be completely thrown away at this point, because we don't know which variants we're interested in until we've looked at all the ZIP files. However, the unzipped VCF files *may* ultimately become too large to reasonably store on disc. We can either: (1) keep the unzipped files if they're not too large; or (2) remove the unzipped files each time, then re-unzip them again later. Let the user decide via **keepunzip**. The logic can be simplified if we don't actually edit the files, but can read them from their ZIPped state.

We are only interested in variants where there are a mixture of positive and negative calls. Every other variant is either shared among everyone (above the tree) or never positive (outside the tree). The first VCF pass selects any variant which has the following: \$7=PASS and GT=0/0 (reference call), 1/1 (derived call) or 0/1 (mixed call). These are parsed into an array of: *positions*, *allele(ref)*, *allele(der)*, and a series of seven counts representing the possible states: positive passed, positive failed, mixed passed, mixed failed, negative passed, negative failed and negative passed by BED file (this last one can't be parsed at this time).

At the end of this process, we can select those variants where $(count(positive\ passed)>0\ ||\ count(mixed\ passed)>0)\ \&\&\ (count(negative\ passed)>0\ ||\ sum(all\ counts)<total)$ as those we want to investigate: these are variants found in someone but not everyone (or those where we need to investigate the BED files to make this distinction), so are those needed to make the tree. These make the master variants list, and we assume for now that the reference allele is the ancestral allele (positives in the reference sequence are dealt with in the next item).

Uses global parameters:

useBigYBoolean: extract new BigY results?useFGCBoolean: extract new FGC results?

etc

usehg37 Boolean: extract build 37 results?keepunzip Boolean: keep or delete unzipped files?

External inputs:

Data repository meta-data

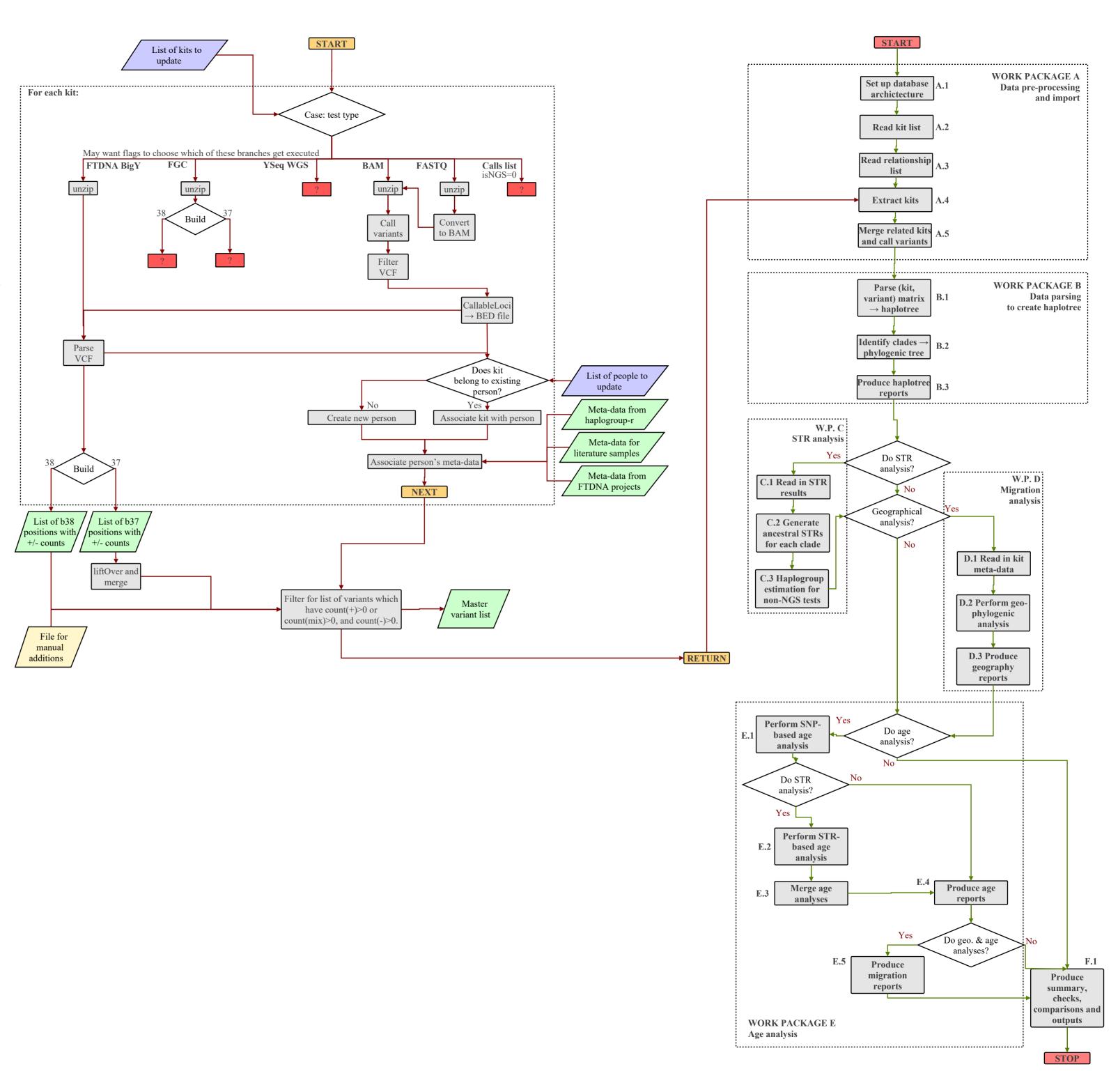
Internal inputs:

Lists of kits and people to update

Produces:

Master list of allvariants

Master list of interesting variants



Item A.5: Merge related kits / call variants

Several kits may be associated with one person. This may include any form of combinations, e.g.: BigY+YElite, BigY+YSeq/WGS, Y-STRs + literature PGP test, Geno 2.0 + BigY, Y-STRs + YSeq SNPs, etc.

One method of dealing with this is to call a list of (people, variants), rather than (kits, variants) to form the haplotree with. Care must be taken to ensure that the most secure determination of a call is taken: e.g., how does a rejected BigY call with BQ=40, MQ=40 and 10/14 reads positive compare with a FGC ** call? This may require careful thought, and the option of guiding the tree structure in part B with a series of manually injected calls may be necessary.

Even if we take a pure (people, variants) list of calls, the merger of two tests must result in a new BED file. This can be a straight merger of data.

chry 120000 125000
chry 127000 128000
and
chry 124000 126000
chry 129000 130000
would merge to:
chry 127000 128000
chry 127000 128000
chry 129000 130000

Where individual SNP calls have been made by packs or individual tests, these entries will need added into the BED file, e.g.:

chrY12060400 12060401

for a Z301 call.

At this point, we also try to fix positives in the reference sequence. These occur because the reference sequence is primarily made from the Y-DNA of a R-U152 male, hence is positive for all SNPs from Y-DNA Adam down through the R-U152 sequence and below. YBrowse's SNP names list maintains the ancestral version, so we can use this to correct these reference-sequence positives when their ancestral values don't match our reference sequence values.

Once this is done, we can make our calls. For calls which pass the quality criteria (field[6] = "PASS" [zero-indexed]), the genotype (field[10,1]) will represent 0/0 for the reference allele, 1/1 for the derived allele, and may be something else (0/1, 1/2, etc.) for a mixture of calls. Note that the derived allele may be comma-separated if there is more than one possible alternative for that call in that test. In the case of mixed (ref/alt or ref/alt1/alt2) calls, we have to choose one to go forwards with.

The quality filter needs recorded: this is -10*log10(Probability call is wrong). So QUAL=10 implies P=0.1, or a 10% chance of a bad call. In a test of 10 million base pairs, QUAL < 70 may be spurious. Variants may also be rejected because the depth (DP) is too high: this could occur if multiple regions of DNA are mapped to the same co-ordinates because they are too similar. It may become more logical to grade the rules more accurately than one simple quality setting.

External inputs:

YBrowse's list of variants

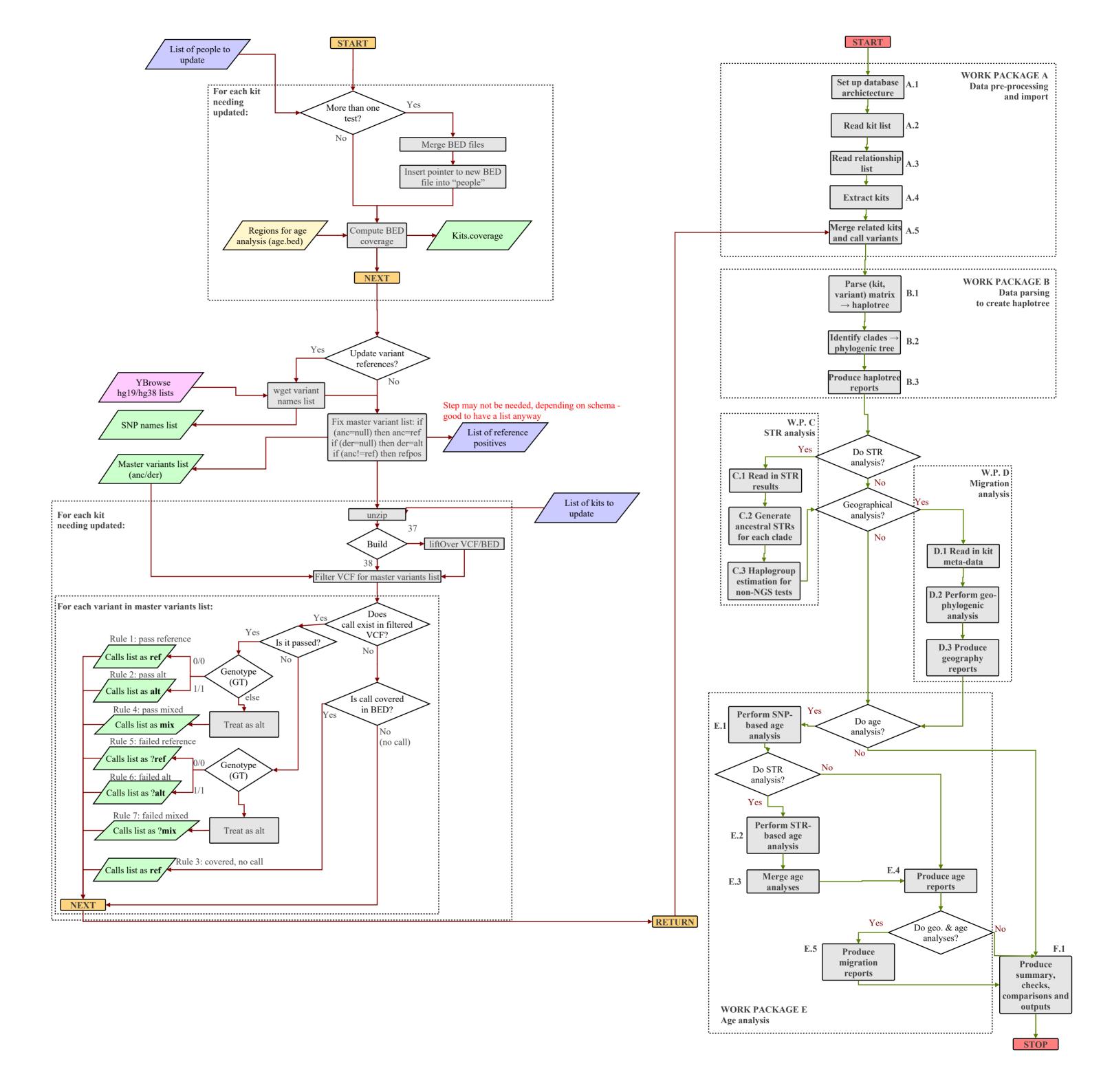
Internal inputs:

Master list of **variants**List of **people** to update

Call quality details on each variant

Produces:

Merged coverage files for people with more than one test Coverage statistics for each kit Reference positive corrections for the variants list A list of snpnames Set of calls for each person/variant



Item B.1: Parse variants to form a haplotree

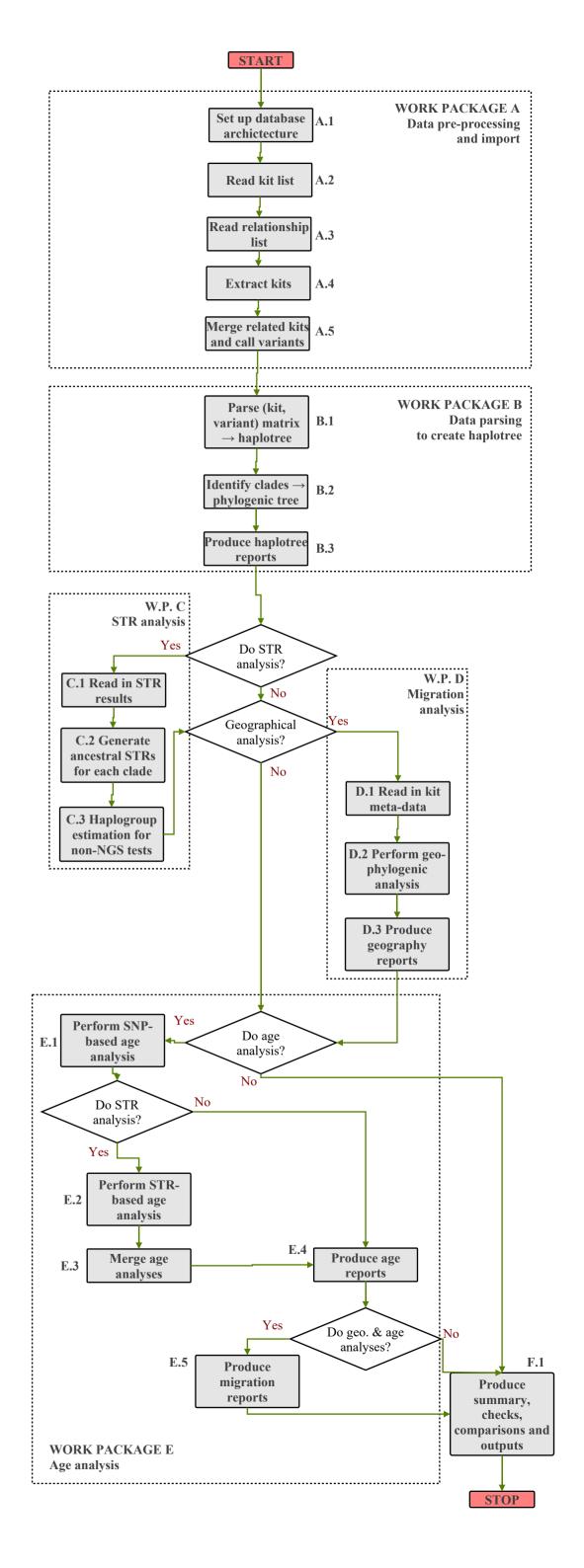
This is one of the more computationally and logically challenging aspects of this process. There are several issues:

- Although rare, a handful of mutations may back-mutate in individual lines, and we need to correct for this.
- Mutations can be recurrent, meaning they cannot be parsed into a phylogenic tree, and need split up.
- Many mutations will be called sporadically among a few tests.
- Indel notation needs standardised among the tests: https://genome.sph.umich.edu/wiki/Variant_Normalization

Many of these processes have known solutions. However, the common solutions, e.g.,

http://www.it2kane.org/2016/07/combining-variant-compare-files/have problems with scalability, reference positives, treating bad data and questionable variants. A more streamlined approach is desired that reduces both CPU time and RAM.

This process will involve several steps, including calculating a quality statistic for each variant, adding each variant into the tree in terms of its quality, and sorting the tree and its variants.



Extended haplotree example

Matrix of {people,variants}

hg38	Name	A	В	C	D	Е	F	G	Н	I	J
3019783	M343	+	+	?	+	+	+	+	+	+	+
6920349	Z 9	-	-	-	+	ı	ı	ı	-	+	-
7378685	Z381	•	+	+	?	ı	ı	+	+	+	ı
8928037	U106	+	+	+	+	ı	+	+	+	+	ı
12060401	Z301	?	?	+	?	?	-	?	+	?	?
12879820	Z18	+	-	-	-	-	+	-	-	-	-
13668461	Z156	-	+	-	-	-	-	+	-	-	-
15732138	L11	+	+	+	+	+	?	+	+	+	+
19538924	Z28	-	-	-	+	-	?	-	-	+	-
19995425	P312	ŀ	ŀ	ŀ		+	ı	ľ	ı	-	+
20029258	Z8	-	-	-	+	1	-	-	-	-	ı
20323911	A297	-	?	-	+	-	-	+	-	?	-
20577481	M269	+	+	+	+	+	+	+	+	+	+
20625892	Z306	-	+	-	ı	-	-	-	-	-	-
21450311	L48	-	-	+	+	-	-	-	+	+	-

Horizontal sort new matrix to create clades

(Swaps AF and BG) Vertical sort swaps Z18 and Z156 (can be needed to merge SNPs in a clade)

hg38	Name	D	Ι	C	Н	В	G	A	F	Е	J
8928037	U106	+	+	+	+	+	+	+	+	-	-
7378685	Z381	+	+	+	+	+	+	-	-	-	-
21450311	L48	+	+	+	+	-	-	-	-	-	ı
6920349	Z9	+	+	-	-	-	-	-	-	-	ı
19538924	Z28	+	+	-	ı	-	-	-	-	-	-
13668461	Z156	ı	1	1	1	+	+	-	-	ı	-
12879820	Z18	-	ı	-	-	-	-	+	+	-	ı
19995425	P312	1	ı	-	-	-	-	-	-	+	+
20029258	Z8	+	-	-	-	-	-	-	-	-	ı
20625892	Z306	ı	-	-	-	+	-	-	-	-	ı
		_	_	_			_				
12060401	Z301	?	?	+	+	?	?	?	-	?	?
20323911	A297	+	?	-	-	?	+	-	-	-	-

Split mutations never called negative Split perfect/imperfect calls Sort (im)perfects both by # positives, then by first positive column

hg38	Name	A	В	С	D	Е	F	G	Н	Ι	J
3019783	M343	+	+	?	+	+	+	+	+	+	+
15732138	L11	+	+	+	+	+	?	+	+	+	+
20577481	M269	+	+	+	+	+	+	+	+	+	+
8928037	U106	+	+	+	+	-	+	+	+	+	-
21450311	L48	ı	ı	+	+	ı	-	ı	+	+	-
6920349	Z9	-	ı	-	+	-	-	ı	-	+	-
12879820	Z18	+	ı	ı	ı	ı	+	ı	·	-	ı
13668461	Z156	ı	+	ı	ı	ı	-	+	-	-	-
19995425	P312	1	-	1	1	+	-	-	-	1	+
20029258	Z8	1	-	-	+	-	-	-	-	1	-
20625892	Z306	-	+	-	-	-	-	-	-	-	-
7378685	Z381	-	+	+	?	-	-	+	+	+	-
12060401	Z301	?	?	+	?	?	-	?	+	?	?
19538924	Z28	-	-	-	+	-	?	-	-	+	-
20323911	A297	-	?	-	+	-	-	+	-	?	-

Horizontal sort perfect calls to create clades

hg38	Name	D	I	C	Н	A	F	В	G	Е	J
8928037	U106	+	+	+	+	+	+	+	+	-	-
21450311	L48	+	+	+	+	-	-	-	-	-	-
6920349	Z9	+	+	-	-	-	-	-	-	1	-
12879820	Z18	1	-	1	1	+	+	1	1	-	-
13668461	Z156	-	-	-	-	-	-	+	+	-	-
19995425	P312	-	-	-	-	-	-	-	1	+	+
20029258	Z8	+	1	-	1	-	-	1	1	-	-
20625892	Z306	-	-	-	-	-	-	+	-	-	-
7378685	Z381	?	+	+	+	<u> </u>	- 	+	+	-	-
12060401	Z301	?	?	+	+	?	-	?	?	?	?
19538924	Z28	+	+	-	-	-	?	-	-	-	-
20323911	A297	+	?	-	-	-	-	?	+	-	-

hg38	Name	D	I	C	Н	A	F	В	G	Е	J
8928037	U106	+	+	+	+	+	+	+	+	-	-
21450311	L48	+	+	+	+	1	1	-	-	ı	1
6920349	Z9	+	+	-	-	-	-	-	-	-	-
12879820	Z18	-	ı	ı	ı	+	+	-	-	-	-
13668461	Z156	-	ı	-	-	-	-	+	+	-	-
19995425	P312	-	ı	ı	ı	-	-	-	-	+	+
20029258	Z8	+	-	-	-	-	-	-	-	1	-
20625892	Z306	-	-	-	-	-	-	+	-	-	-
7378685	Z381	?	+	+	+	-	-	+	+	-	-
12060401	Z301	?	?	+	+	?	-	?	?	?	?
19538924	Z28	+	+	-	-	-	?	-	-	-	-
20323911	A297	+	?	-	-	-	-	?	+	-	-

Select imperfects where one call is missing and fit in from top→bottom:

Names are for illustrative purposes only, and not used in the logic

Z381 has 5-6 +ves \rightarrow between L48 and U106

Test Z381 for U106: A,F are U106+ Z381- so Z381 is a subset of U106

Test Z381 for U106: no U106- Z381+ so phylogenically consistent

Test L48 for U106: B,G are L48- Z381+ so Z381 is a superset of L48&?

Z28 logic:

Test L48 for positives: D,I are L48+

Test Z28 for L48: C,H are L48+ but Z28-, so Z28 is a subset of L48

Test Z28 for L48: no L48- Z28+, so phylogenically consistent

Test Z9 for positives: D,I are Z9+

Test Z28 for Z9: no Z9+ Z28-, so Z28 may be equivalent to Z9

Select imperfects where two calls are missing and fit in from top→bottom: A297 logic:

A297 has 2-4 +ves \rightarrow between P312 and L48

Work from L48 to P312: Test L48 for positives: D and G are positives

Test A297 for L48: C,H are L48+ A297- so A297 is a subset of L48

Test A297 for L48: G is L48- A297+ so A297 is phylogenically inconsistent

→ Recurrent or junk - do not insert yet

Horizontal sort, vertical sort

...Select imperfects where seven calls are missing and fit in from top—bottom: Z301 logic:

Z301 has 2-9 +ves \rightarrow between P312 and U106

Work from U106 to P312:

Test U106 for positives: C,H are U106+ Z301+

Test Z301 for U106: F is U106+ Z301- so Z301 is a subset of U106

Test Z301 for U106: no U106- Z301+ so phylogenically consistent

Test Z381 for positives: C,H are Z381+ Z301+

Test Z301 for Z381: no Z381+ Z301- so Z301 may be equivalent to Z381

Test Z301 for L48: no Z381- Z301+ so Z301 is consistent and equivalent to Z381

Test L48 for positives: C,H are L48+ Z301+

Test Z301 for L48: no L48+ Z301- so Z301 may be equivalent to L48

Test Z301 for L48: no L48- Z301+ so Z301 is consistent and equivalent to L48

Test Z9: no Z9+ Z301+ so Z301 is not a superset of Z9; Z9 is also a subset of U106 Test Z28: no Z28+ Z301+ so Z301 is not a superset of Z28; Z28 is also a subset of U106

Test Z156: no Z156+ Z301+ so Z301 is not a superset of Z156; Z156 is also a subset of U106

Test Z18: no Z18+ Z301+ so Z301 is not a superset of Z18; ; Z18 is also a subset of U106

Test P312: no P312+ Z301+ so Z301 is not a superset of P312 P312 is not a subset of U106, so E,J are Z301-

Z301 is equivalent to U106>Z381 or U106>Z381>L48

Test bottom up: L48, Z381:

D,I are L48+ so are Z301+

B,G are Z381+ so may be Z301+ - presume positive in tree, merge up

Ambiguous calls remaining, insert into tree at highest possible level

Flag as ambiguous position

Return to recurrent/junk mutations:

(Call mutations with two recurrencies first, proceed from best to worst called) A297 logic:

Two recurrencies in ten tests

Does this exceed a maximum tolerance?

Maximum tolerance would normally be ~1 in 1000 so this would be junk, but let's say no... Split recurrencies: A297.1, A297.2, parse as before:

→ A297.1 is either equivalent to Z8 or Z9/Z28, parse as a questionable singleton

→ A297.2 is either equivalent to Z306 or Z156, parse as a questionable singleton

Questionable singletons don't get merged up, so we assume A297.1=Z8, A297.2=Z306

Horizontal sort, vertical sort

20625892 Z306

All mutations assigned \rightarrow Haplotree complete, form clades

		0							_		
hg38	Name	D	I	C	Н	В	G	A	F	Е	J
8928037	U106	+	+	+	+	+	+	+	+	-	-
7378685	Z381	+	+	+	+	+	+	-	-	-	ı
12060401	Z301	+	+	+	+	+	+	-	-	-	- 1
21450311	L48	+	+	+	+	-	-	-	-	-	ı
6920349	Z 9	+	+	-	-	-	-	-	-	-	ı
19538924	Z28	+	+	-	-	-	-	-	-	-	ı
13668461	Z156	-	ı	-	-	+	+	-	-	-	ı
12879820	Z18	-	-	-	-	-	-	+	+	-	-
19995425	P312	-	-	-	-	-	-	-	-	+	+

-	+	-	-	-	-	-	-	-	-	-
-	+	-	-	-	-	-	-	-	-	-
-	-	-	1	-	+	-	-	-	1	-
-	_	-	-	-	-	+	-	-	-	-

DICHBGAFEJ

1450311	L48	+	+	+	+	-	-	-	-	-	-
920349	Z 9	+	+	ı	-	-	-	-	-	-	-
9538924	Z28	+	+	-	-	-	-	-	-	-	-
3668461	Z156	1	ı	ı	-	+	+	1	1	1	-
2879820	Z18	-	ı	ı	-	-	-	+	+	-	-
9995425	P312	1	ı	1	ı	1	1	1	1	+	+
0029258	Z8	+	ı	ı	ı	ı	-	-	-	-	-
0323911	A297.1	+	-	-	-	-	-	-	-	-	-

-	-	-	-	ı	+
-	-	-	-	-	\dashv
-	1	1	-	-	Ŀ
+	-	-	-	ı	Ŀ

Z381 logic:

Work from U106 to L48:

Test U106 for positives: I,C,H,B,G are U106+ Z381+

Test L48 for positives: C,H are L48+ Z381+

D is L48+ so D must be Z381+

All calls fixed, insert into tree

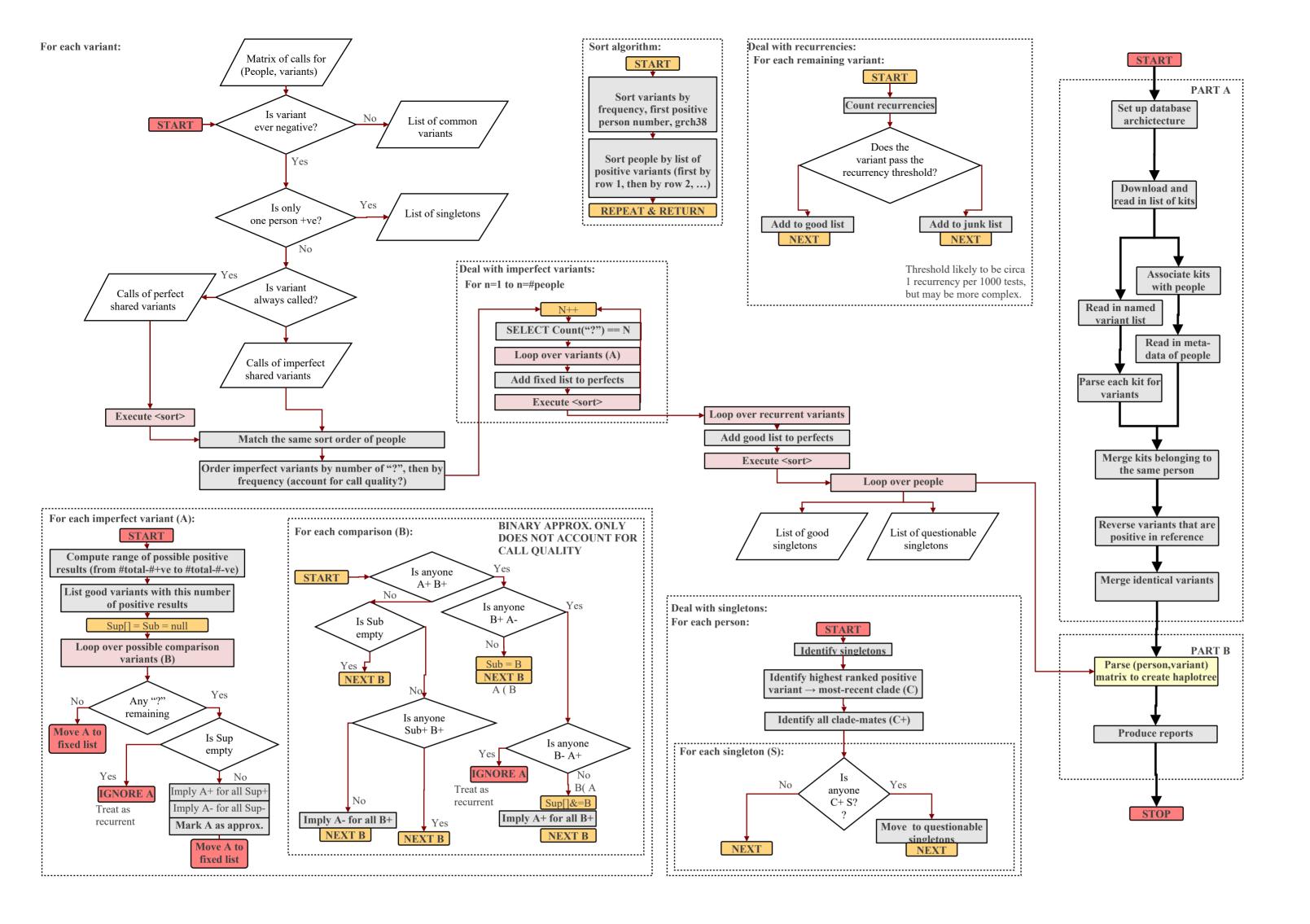
Z28 has 2-3 +ves \rightarrow P312 (lowest 2x) and L48

Work from L48 to P312:

Test Z28 for Z9: no Z9- Z28+, so Z28 is consistent and equivalent to Z9

Test Z18: Z18 is L48-, and A is Z28-, so F must be Z28-

All calls fixed, insert into tree



Basic list		Count	First	274005	75848	289065	584862	B75780	284744	65049	120386	5962	216600	22654	N4826	224096
FGC13445	4484946	3	2	1 2	+	2	5	• E	2				2	+	•	+
FGC13446	7678200	2	2	-	+	-	-	-	-	-	-	-	-	-	-	+
FGC13458	13908115	2	2	-	+	-	-	-	-	-	-	-	-	-	-	+
S18823	14695706	2	3	-	-	+	-	-	-	-	-	-	+	-	-	-
CTS7658	15482393	2	3	_	_	+	-	-	-	_	_	_	+	-	_	_
S1894	16098109	3	4	-	-	-	+	-	+	-	+	-	-	-	-	-
S1900 S1911	17358211 21003759	7	2	-	-	-	+	-	+	+	+	-	-	-	-	- -
51911	21003739	/		-	+	-	+	-	+	+	+	-	-	+	-	+
G 1	\	+		16		16	C)	0			,0					
Sort by: count,first,p	osition	Count	First	274005	75848	289065	584862	B75780	284744	65049	120386	5962	216600	22654	N4826	224096
. 4		Ö	H	27	75	28	58	B7	28	9	12	5	21	2,7	Ż	22
S1911	21003759	7	2	-	+	-	+	-	+	+	+	-	-	+	-	+
FGC13445	4484946	3	2	-	+	-	-	-	-	-	-	-	-	+	-	+
S1894	16098109	3	4	-	-	-	+	-	+	-	+	-	-	-	-	-
S1900	17358211	3	4	-	-	-	+	-	+	-	+	-	-	-	-	-
FGC13446	7678200	2	2	-	+	-	-	-	-	-	-	-	-	-	-	+
FGC13458	13908115	2	2	-	+	-	-	-	-	-	-	-	_	-	-	+
S18823	14695706	2	3	-	-	+	-	-	-	-	-	-	+	-	-	-
CTS7658	15482393	2	3	-	-	+	-	-	-	-	-	-	+	-	-	-
Sort first rov	w, + to -	Count	First	75848	584862	284744	65049	120386	22654	224096	274005	289065	B75780	5962	216600	N4826
S1911	21003759	7	1	+	+	+	+	+	+	+	-	-	-	-	-	-
FGC13445	4484946	3	2	+	-	-	-	-	+	+	-	-	-	-	-	-
S1894	16098109	3	4	-	+	+	-	+	-	-	-	-	-	-	-	-
S1900	17358211	3	4	-	+	+	-	+	-	-	-	-	-	-	-	-
FGC13446	7678200	2	2	+	-	-	-	-	-	+	-	-	-	-	-	-
FGC13458	13908115	2	2	+	-	-	-	-	-	+	-	-	-	-	-	-
S18823	14695706	2	3	-	-	-	-	-	-	-	-	+	-	-	+	-
CTS7658	15482393	2	3	-	-	-	-	-	-	-	-	+	-	-	+	-
Sort subseque from first to result. Check consistent: larow is the saif not.	last positive k results ast positive	Count	First	75848	584862	284744	65049	120386	22654	224096	ı		B75780 sitis			N4826 am
S1911	21003759	7	1	+	+	+	+	+	+	+	-	-	-	-	-	-
FGC13445	4484946	3	1	#	-	-	-		#	#	-	-	_	-	-	-
S1894	16098109	3	4	-	+	+	-	+	-	-	-	-	-	-	-	-
S1900	17358211	3	4	-	+	+	-	+	-	-	-	-	-	-	-	-
FGC13446	7678200	2	2	+	-	-	-	-	-	+	-	-	-	-	-	-
FGC13458	13908115	2	2	+	-	-	-	-	-	+	-	-	-	-	-	-
S18823	14695706	2	3	-	-	-	-	-	-	-	-	+	-	-	+	-
CTS7658	15482393	2	3	-	-	-	-	-	-	-	-	+	-	-	+	-
esult. Check consistent: la	last positive c results ast positive	ınt	st	2	2 25	2 960	1 298	1	49	1 988	Las	st po	row sitiv	es a	ll sa	
ow is the saf not.		Count	First	75848	22654	224096	584862	284744	65049	120386	274005	289065	B75780	5965	216600	N4826
S1911	21003759	7	1	+	+	+	+	+	+	+	_	_	_	_	_	_
FGC13445	4484946	3	1	+	+	+			-	T	-	-	_	-	-	-
	16098109	3	4	-	-	-	F-3		-	-	-	-	-	-	-	-
S1894	10076107						+	+	-	+	-	-	-	-	-	-
S1894 S1900	17358211	3	4		- 1						-	-	-			
		3	2	+	-	+	-	-	-			_			-	-
S1900	17358211			+ +	-	+	-	-	-	-	-	-	-	-	-	-
S1900 FGC13446	17358211 7678200	2	2		- - -		-	-	-	-	-	-+	-	-	-+	- - -
S1900 FGC13446 FGC13458	17358211 7678200 13908115	2	2		- - -	+	-	-	-	-	-	+ +	-	-	- + +	- - -
S1900 FGC13446 FGC13458 S18823	17358211 7678200 13908115 14695706	2 2	2 3		- - -	+	- - -	-	- - -	- - -	-		-	-		- - -
S1900 FGC13446 FGC13458 S18823 CTS7658	17358211 7678200 13908115 14695706 15482393	2 2	2 3		- - - -	+	- - - 4	- - - 4	- - - 4	- - 1	-		- 8	-		- - -
S1900 FGC13446 FGC13458 S18823 CTS7658	17358211 7678200 13908115 14695706 15482393	2 2 2	2 3 3	+ - - 6		+ - 2	4	4	4	- 1 64	-	+	8	1 1 1 082	-	-
S1900 FGC13446 FGC13458 S18823 CTS7658	17358211 7678200 13908115 14695706 15482393	2 2	2 3	-		-	4	4	-	1 1 1 1 1 1 1 1 1 1	-	+	-	B75780 ' ' ' '		N4826 ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' ' '
S1900 FGC13446 FGC13458 S18823 CTS7658 Group every zero positive ogether	17358211 7678200 13908115 14695706 15482393 one with evariants	2 2 2 2 1unoO	First 2	+ · · · 6 8488 b	224096	2 2 6 5 4 2	- 4 284862 - 4 284862 4 284862	- 4 284744	4 4 120386		274005	+	216600	B75780 i i i i	-	-
S1900 FGC13446 FGC13458 S18823 CTS7658 Group every zero positive together S1911	17358211 7678200 13908115 14695706 15482393 one with evariants	2 2 2 2 1 tunoO 7	2 2 3 3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1	+ 75848 9	+ 224096	+ 25654 5	4	4	4	+ 65049 1	-	+	8	B75780 1 1 1 1 1 1 1 1 1	-	-
S1900 FGC13446 FGC13458 S18823 CTS7658 Group every zero positive together S1911 FGC13445	17358211 7678200 13908115 14695706 15482393 one with evariants 21003759 4484946	2 2 2 2 2 7 TunoO 7 3	2 3 3 3 1 Hirst 1 1 1	+ · · · 6 8488 b	224096	+ + 25054 2	4 584862 4	- 4 284744 P	- 4 120386 + 		. 274005	+	216600	B75780	-	N4826 '
S1900 FGC13446 FGC13458 S18823 CTS7658 Group every zero positive ogether S1911 FGC13445 S1894	17358211 7678200 13908115 14695706 15482393 cone with evariants 21003759 4484946 16098109	2 2 2 2 2 7 3 3 3 3	2 2 3 3 3 1 1 1 1 1 4	+ 75848 9	+ 224096	+ 25654 5	+ + 584862 +	+ + + 284744 + + - + + - + + - + + + + + + + + + +	4 120386 + - +		274005	+	216600	B75780	-	N4826 '
S1900 FGC13446 FGC13458 S18823 CTS7658 Group every zero positive ogether S1911 FGC13445 S1894 S1900	17358211 7678200 13908115 14695706 15482393 one with evariants 21003759 4484946 16098109 17358211	2 2 2 2 2 7 3 3 3 3 3	2 2 3 3 3 1 Hzizi	+ + +	1 + + 224096	+ - - 2 2 + + -	4 584862 4	- 4 284744 P	4 120386 + + +		1 1 274005	+	- 1 216600 ∞ -	B75780	-	N4826 .
S1900 FGC13446 FGC13446 S18823 CTS7658 Group every zero positive ogether S1911 FGC13445 S1894 S1900 FGC13446	17358211 7678200 13908115 14695706 15482393 2000 with evariants 21003759 4484946 16098109 17358211 7678200	2 2 2 2 2 7 3 3 3 3 2 2	2 3 3 3 1 1 4 4	+ + + + + + + + + + + + + + + + + + + +	+ + + 224096	+ + 22654	+ + 584862 +	+ + + 284744 + + - + + - + + - + + + + + + + + + +	4 120386 + - +		. 274005	+	216600	B75780	-	N4826 '
S1900 FGC13446 FGC13458 S18823 CTS7658 Group every zero positive ogether S1911 FGC13445 S1894 S1900	17358211 7678200 13908115 14695706 15482393 one with evariants 21003759 4484946 16098109 17358211	2 2 2 2 2 7 3 3 3 3 3	2 2 3 3 3 1 Hzizi	+ + +	1 + + 224096	+ - - 2 2 + + -	+ + 584862 +	+ + + 284744 + + - + + - + + - + + + + + + + + + +	4 120386 + + +		1 1 274005	+	- 1 216600 ∞ -	. I I I I B75780 I I I I I	-	N4826 .

Add some notice (All have a facility but man	full set of	Count	First	75848	224096	22654	584862	284744	120386	65049	289065	216600	274005	B75780	5962	N4826
S1911	21003759	7	1	+	+	+	+	+	+	+	-	-	-	-	-	-
FGC13445	4484946	3	1	+	+	+	-	-	-	-	-	-	-	-	-	_
S1894	16098109	3	4	-	-	-	+	+	+	-	-	-	-	-	-	-
S1900	17358211	3	4	-	-	-	+	+	+	-	-	-	-	-	-	-
FGC13446	7678200	2	1	+	+	-	-	-	-	-	-	•	-	-	-	-
FGC13458	13908115	2	1	+	+	-	-	-	-	-	-	-	-	-	-	-
S18823	14695706	2	9	-	-	-	-	-	-	-	+	+	-	-	-	-
CTS7658	15482393	2	9	-	-	-	-	-	-	-	+	+	-	-	-	-
ZS9483	19123304	2?	-	-	-	-	-	-	-	-	?-	?-	+	-	+	-
-	26662460	2?	-	-	-	-	-	-	-	-	+	-	?-	?-	+	?-
CTS11041	20752203	7?	-	+	+	?+	?+	+	+	+	?-	-	?-	-	-	?-
BY28746	26662435	5?	-	?-	-	+	+	+	+	?+	-	-	?-	?-	_	?-
BY28749	26662459	5?	-	?-	-	+	+	+	+	?+	-	-	?-	?-	-	?-
BY28747	26662448	4?	-	?-	-	•	+	+	?+	?+	?-	•	?-	?-	-	?-