Supplementary material

This supplementary material describes all the necessary steps to repeat the experiments described in the main paper. The first section presents the datasets and their sources. Section 2 is a detailed description of how to use the provided package with our software to repeat the experiments for 1000 Genomes Project dataset or use with any user data. Section 3 specifies all parameters used for other compression methods. Section 4 contains tables with the complete results. Finally, in Section 5 some technical details of the algorithm as well as an example of compressing the byte vectors are given.

1 DATASETS

1.1 The *Homo sapiens* data

This dataset comes from the Phase 1 of the 1000 Genomes Project, which contains data for 1092 human individuals, 525 males and 567 females.

The VCF (Variant Call Format) files can be found at the 1000 Genomes Project's anonymous FTP servers. All used VCF files can be downloaded from the directory containing the final variant calls for the phase1 datasets. It can be either the EBI or NCBI FTP site:
`ftp://ftp.1000genomes.ebi.ac.uk/vol1/ftp/phase1/analysis_results/integrated_call_sets/`
`ftp://ftp.ncbi.nih.gov//1000genomes/ftp/phase1/analysis_results/integrated_call_sets/`
The complete list of compressed VCF files used:

```
ALL.chr1.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr2.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr3.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr4.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr5.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr6.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr7.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr8.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr9.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr10.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr11.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr12.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr13.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr14.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr15.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr16.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr17.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr18.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr19.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr20.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr21.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chr22.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chrX.integrated_phase1_v3.20101123.snps_indels_svs.genotypes.vcf.gz
ALL.chrY.phase1_samtools_si.20101123.snps.low_coverage.genotypes.vcf.gz
```

For all individuals variant calls for chromosomes 1–22 (autosomes) all diploid and genotypes are phased. Therefore, the information about variants found on both chromosomes of chromosome pairs 1–22 is available for all individuals and it is possible to obtain 2,184 sequences (2 for each individual). The data contain also information about pairs of chromosomes X for females (denoted by X-fem in the presented results). The situation is, however, more complex for male individuals due to two pseudoautosomal regions (PAR1 and PAR2) that are common between X and Y chromosomes. These pseudoautosomal regions are treated as autosomes in the description of the results of the 1000GP. Precisely, the diploid genotype calls for variants in two pseudoautosomal regions of X and Y chromosomes are stored in the available VCF file for X chromosome, while haploid genotype calls for non-pseudoautosomal regions (nonPAR, between PAR1 and PAR2) of X and Y chromosomes are stored in the corresponding VCF files (nonPAR of X in the VCF file for X chromosome, nonPAR of Y in the VCF file for Y chromosome).

Therefore, processing of X and Y chromosomes is divided into processing of five separate groups (which are treated as separate chromosomes):

- X-fem—whole chromosome X of females, with diploid genotype calls (2 sequences per individual),
- X-mal—nonPAR region of chromosome X of males, with haploid genotype calls (1 sequence per individual),
- Y-mal—nonPAR region of chromosome Y of males, with haploid genotype calls (1 sequence per individual),

- XY-mal1— PAR1 region of X and Y chromosomes of males, with diploid genotype calls (2 sequences per individual),
- XY-mal2— PAR2 region of X and Y chromosomes of males, with diploid genotype calls (2 sequences per individual).

The reference sequences of assembled chromosomes can be found at GenBank and downloaded from the NCBI's anonymous FTP server: `ftp://ftp.ncbi.nlm.nih.gov/genbank/genomes/Eukaryotes/vertebrates_mammals/Homo_sapiens/GRCh37/Primary_Assembly/assembled_chromosomes/FASTA/`
The complete list of compressed reference sequences: `chr1.fa.gz`, `chr2.fa.gz`, `chr3.fa.gz`, `chr4.fa.gz`, `chr5.fa.gz`, `chr6.fa.gz`, `chr7.fa.gz`, `chr8.fa.gz`, `chr9.fa.gz`, `chr10.fa.gz`, `chr11.fa.gz`, `chr12.fa.gz`, `chr13.fa.gz`, `chr14.fa.gz`, `chr15.fa.gz`, `chr16.fa.gz`, `chr17.fa.gz`, `chr18.fa.gz`, `chr19.fa.gz`, `chr20.fa.gz`, `chr21.fa.gz`, `chr22.fa.gz`, `chrX.fa.gz`, `chrY.fa.gz`

Reference sequences for non-pseudoautosomal and pseudoautosomal regions of male X and Y chromosome have to be formed from appropriate regions of reference sequences of complete chromosomes X and Y, that is (1-based coordinates):

1. X-mal: 2699521–154931043 region of reference sequence of X chromosome (nonPAR),
2. Y-mal: 2649521–59034049 region of reference sequence of Y chromosome (nonPAR),
3. XY-mal1: 60001–2699520 region of reference sequence of X chromosome (PAR1),
4. XY-mal2: 154931044–155260560 region of reference sequence of X chromosome (PAR2).

1.2 The *Arabidopsis thaliana* data

This dataset comes from 1001 Genomes Project (1001GP), "A Catalog of Arabidopsis thaliana Genetic Variation". It contains data about 775 sequenced strains from 4 different subprojects:

- 80 strains from "MPICao2010—80 Arabidopsis thaliana accessions", release 2012_03_13.
  Data access:
  `http://1001genomes.org/data/MPI/MPICao2010/releases/2012_03_13/strains/`
  Project's publication:
  Cao, J., Schneeberger, K., Ossowski, S., Günther, T., Bender, S., Fitz, J., Koenig, D., Lanz, C., Stegle, O., Lippert, C., Wang, X., Ott, F., Mller, J., Alonso-Blanco, C., Borgwardt, K., Schmid, K. J., and Weigel, D. Whole-genome sequencing of multiple Arabidopsis thaliana populations. Nature Genetics 43, 956–963 (2011).
- 170 from "Salk—Arabidopsis thaliana strains sequenced by the Salk Institute" (data about one strain is corrupted), release 2011_06_28.
  Data access: `http://1001genomes.org/data/Salk/releases/2011_06_28/TAIR10/strains/`
  This data was generated by the Salk Institute.
- 180 strains from "GMINordborg2010—Arabidopsis thaliana strains sequenced by the Gregor Mendel Institute", release 2011_08_04.
  Data access:
  `http://1001genomes.org/data/GMI/GMINordborg2010/releases/2011_08_04/strains/`
  These sequence data were produced by the Nordborg laboratory at the Gregor Mendel Institute of Molecular Plant Biology.
- 345 strains from "MPICWang2013—343 Arabidopsis thaliana accessions" (data about 345 more strains available), release 2013_04_15.
  Data access:
  `http://1001genomes.org/data/MPI/MPICWang2013/releases/2013_04_15/strains/`
  These sequence data were produced by Monsanto Company and the Weigel laboratory at the Max Planck Institute for Developmental Biology.

The VCF files describing all variants that can be found are not provided by the 1001GP. Only information about positions and qualities (not always the case) of variants found in each sequenced strain are given. These were used to produce one joined VCF file for each chromosome, chloroplast and mitochondria. Resultant VCF files describe every variant present in the dataset (in at least one strain) and genotype of all 775 strains. Variants of the same type, found at the same position were joined to report different alleles (e.g. two distinct SNPs at the same position).
In details, the following variants were used and processed (described by subproject):

- MPICao2010:
  – all 1-, 2- and 3-symbol insertions (accessible in `insertion.txt` file for each strain, no quality data available)
  – filtered SNPs and 1-symbol deletions (accessible in `filtered_variant.txt` file for each strain, variants with quality greater or equal to 25 were processed)
- Salk:
  – all SNPs and 1-symbol deletions (accessible in `quality_variant_filtered_[ACCESSION].txt` file for each strain, all were with quality greater or equal to 25)

- GMINordborg2010:
  – all SNPs (accessible in `[ACCESSION].0cf.snp.log` file for each strain, no quality data available)
- MPICWang2013:
  – filtered SNPs and 1-symbol deletions (accessible in `quality_variant_[ACCESSION]_TAIR10.txt` file for each strain, variants with quality greater than or equal to 25 were processed)

In all above cases deletions found on succeeding positions were not joined. Thus all deletions remained 1-symbol. The described variants were found on the 5 Arabidopsis chromosomes (chr1, chr2, chr3, chr4, chr5) and on Chloroplast and Mitochondria chromosome (chrC, chrM).

All sequenced genomes were described in reference to the same reference sequence of Arabidopsis thaliana, TAIR10 assembly. It can be accessed from the anonymous FTP server : `ftp://ftp.arabidopsis.org//Sequences/whole_chromosomes/`.
The complete list of compressed reference sequences:
`TAIR10_chr1.fas`, `TAIR10_chr2.fas`, `TAIR10_chr3.fas`, `TAIR10_chr4.fas`, `TAIR10_chr5.fas`, `TAIR10_chrC.fas`, `TAIR10_chrM.fas`.

As most of the original data is still in waiting period status, because of legal regulations it cannot be redistributed or repackaged. Thus we are not allowed to make the resultant VCF files publicly available.

2 THOUSANDS GENOMES COMPRESSOR DESCRIPTION

Thousands Genomes Compressor (TGC), can be downloaded in a `tgc.tar.gz` package, which is publicly available under a free license at http://sun.aei.polsl.pl/tgc. It includes all programs and scripts presented in this section.

2.1 Basic usage

The `tgc.tar.gz` package allows to perform all the processing for the whole genome and all 1092 individuals from the 1000 Genomes Project (as described in the paper). Due to legal regulations of 1001 Genomes Project, step-by-step processing of the related dataset is not included in the package (see Section 1.2).

To repeat the described experiments for *H. sapiens dataset*, download the `tgc.tar.gz` file, decompress it, build all sources in the `src` directory and run the `run` script with appropriate options to download and process the data. The minimal steps in a unix environment with gcc compiler and basic utilities available, are:

```
tar -xzf tgc.tar.gz
cd tgc
cd src
make
cd ..
./run -abcdefgijkl
```

The long list of `run` switches will be explained in Section 2.3.

The whole processing requires 2.4 GB of RAM and about 16 TB of disk space (most of which is swallowed by 2×6.7 TB of consensus data and 1.2 TB of original VCF data). It should also be noted that the whole computation may take a lot of time (several days), but the dominant part of it are due to the original VCF conversion (details of the representation in the main text).

It is also possible to only examine a single program contained in the package (i.e., `tgc` for compressing the collection of byte vectors), available in the `src` folder. Their command-line specifications can be found in Section 2.2 and some sample scenarios are in Section 2.4.

2.2 Detailed description of all programs

This section is a detailed description of all tools included in the package, together with their command-line specifications. These programs are used to perform experiments on 1000GP data (see Section 2.3), but can also be used with any user data (see Section 2.4).

All described tools performing a conversion from a VCF file (`VCF*`), are able to handle both: VCF and VCFmin files.

- `cut-ref`

  The program creates a new FASTA file by cutting a piece of the input FASTA sequence and adding range information to the header. It is used to produce reference sequences for nonPAR, PAR1 and PAR2 regions of chromosomes X and Y, for male individuals. Usage:
  `cut-ref <input_name> <output_name> <start_pos> <end_pos>`
  `input_name` – name of the input file
  `output_name` – name of the output file
  `start_pos` – start position of the cut sequence
  `end_pos` – end position of the cut sequence

- `processX`

  The program processes the VCF file for X chromosome to create four VCF files: one for female individuals and three for male individuals. Each VCF file for male individuals corresponds to one of the regions of interest of the chromosome X (PAR1, PAR2, nonPAR). Usage:
  `processX <vcf>`
  `vcf` – name of the VCF file for X chromosome

- `VCF2VCFmin`

  The program processes the input VCF file to create its stripped version, VCFmin (name of the input VCF and `*.min` extension). It preserves basic information about variants (possible kinds and positions of changes) together with genotypes of all included individuals. These data are enough to create consensus sequences for all individuals. The reference sequence is used to check the correctness of listed variants and expand information about structural variants having just "<DEL>" in the REF column. By default it is assumed that the first reference character has position 1, but it can be changed with an optional argument (option used for PAR and nonPAR regions in X and Y chromosomes). The created VCFmin file conforms to the VCF version 4.1 format.
  Usage:
  `VCF2VCFmin <vcf> <ref> [start_pos]`
  `vcf` – name of the VCF file
  `ref` – name of the reference sequence

start_pos – start position of the reference sequence (optional, 1 by default)

- VCF2FASTA-h and VCF2FASTA-d
  The programs process the input reference sequence of a chromosome and corresponding VCF file with haploid (VCF2FASTA-h) or diploid (VCF2FASTA-d) genotype calls to create one (VCF2FASTA-h) or two (VCF2FASTA-d) FASTA consensus sequence(s) for each individual described in the VCF file. Names of the output consensus sequences are made by adding to the name of the VCF file: individual's ID, chromosome indicator in case for autosomal chromosomes ('1' or '2', VCF2FASTA-d only) and the ".fa" extension. All variants are taken into account, regardless of the value of the FILTER field (only a warning is output if it's different than "PASS" or "."). Usage:
  VCF2FASTA-h <vcf> <ref> [start_pos]
  VCF2FASTA-d <vcf> <ref> [start_pos]
  vcf – name of the VCF file
  ref – name of the reference sequence
  start_pos – start position of the reference sequence (optional, 1 by default)
- VCF2VDBV-h and VCF2VDBV-d
  The programs process the input VCF file with haploid (VCF2VDBV-h) or diploid (VCF2VDBV-d) genotype calls and creates a new file with the variant database (without the data about individuals), together with one (VCF2VDBV-d) or two (VCF2VDBV-d) byte vector(s) for each individual described in the VCF file. The name of the variant database file is the name of the input VCF file with the ".vd" extension. The names of each output byte vector is created from the individual's ID, chromosome indicator in case of autosomal chromosomes ('1' or '2', VCF2VDBV-d only), chromosome name and the ".bv" extension.
  A variant with single allele is represented in the variant database in a single line (describing its id, location, type and occurring change). In case of variants with multiple alleles, each is represented in the variant database by multiple lines, where number of lines is equal to the number of alternative alleles. These sub-variants (lines) in the variant database have successive ids and describe different changes, but they have identical positions and types. The information if any of the multiple alleles was found on a particular genome is stored in n bytes in a byte vector for that genome, where n is the number of multiple alleles. Usage:
  VCF2VDBV-h <vcf> <chr_name>
  VCF2VDBV-d <vcf> <chr_name>
  vcf – name of the VCF file
  chr_name – chromosome name (optional, taken from name of the 1000GP VCF file by default)

- VDBV2FASTA
  The program processes the input reference sequence of a chromosome and corresponding variant database to create FASTA consensus sequences for all individuals for which the byte vectors are provided. Names of the output consensus sequences are made from names of the input byte vectors with extension ".fa".
  It is checked if there is more than one variant of the same type and at the same position (the successive lines are checked till the current variant's position increments). If so, the algorithm recognise such variants from the database as sub-variants of a variant with multiple alleles and construct consensus accordingly. Usage:
  VDBV2FASTA <ref> <variant_database> <start_pos> [<byte_vector>]+
  ref – name of the reference sequence
  variant_database – name of the variant database file
  start_pos – start position of the reference sequence
  <byte_vector> – name of the byte vector (at least one)
- VDBV2VCFmin-h and VDBV2VCFmin-d
  The programs reconstructs the VCF minimal file (that conforms with the specification of VCF) from the input reference sequence of a chromosome, the corresponding variant database file and individuals' byte vectors. The output VCF minimal file contains genotypes of all inputted individuals. All variants will have value "PASS" in the "FILTER" filed. The genotypes are haploid in case of VDBV2VCFmin-h and diploid (phased) in case of VDBV2VCFmin-d (here each input pair of byte vectors is considered to be from one individuals). The output file has ".min" extension.
  It is checked if there is more than one variant of the same type and at the same position (the successive lines are checked till the current variant's position increments). If so, the algorithm recognise such variants from the database as sub-variants of a variant with multiple alleles and construct VCF minimal accordingly. Usage:
  VDBV2VCFmin-h<ref> <variant_database> <chrom> <start_pos> [<byte_vector>]+
  VDBV2VCFmin-d <ref> <variant_database> <chrom> <start_pos> [<byte_vector> <byte_vector>]+
  ref – name of the reference sequence
  variant_database – name of the variant database file
  chrom – name of the chromosome (added to the first column for each variant in the output VCFmin)

start_pos – start position of the reference sequence
<byte_vector> – name of the byte vector (at least one for VDBV2VCFmin-h, and at least two for VDBV2VCFmin-d)

- tgc
  The program compresses / decompresses the collection of byte vectors into / from single file. Usage:
  tgc <mode> <archive_name> [<list_file_name>]
  mode – c (compression) or d (decompression)
  archive_name – name of the archive
  list_file_name – name of the file with the list of files to compress / decompress; required in the c mode, optional in the d mode (if not provided all files are extracted from the archive)
- tgc_db
  The program compresses / decompresses the variant database into / from single file. Usage:
  tgc_db <mode> <input_name> <output_name>
  mode – c (compression) or d (decompression)
  input_name – name of the input file
  output_name – name of the output file

2.3 Main script processing the data

The run script can be used to repeat the experiments with the TGC compressor on the 1000 GP data. It processes the data for each chromosome specified in the configuration file (config.ini), in the $CHROM variable. Possible chromosome names are: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, X-fem, X-mal, X-mal1, X-mal2, Y-mal. By default $CHROM variable is set to 22 (CHROM="22"), so only chromosome 22 will be handled. To process whole genome, all possible chromosome names must be listed by the $CHROM variable (CHROM="1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 X-fem X-mal X-mal1 X-mal2 Y-mal"). It is also possible to process only selected chromosomes, by changing the $CHROM variable accordingly. Two other variables in config.ini are: $FTP specifying the ftp site for download (EBI by default) and $REF specifying folder for reference sequences (REFERENCE/ by default). If not stated otherwise, the output files are placed in the directory named after the processed chromosome. Available options:

- -h
  Presentation of possible switches of the run script. All options are briefly described.
- -a
  Download (wget utility) and decompression (gzip utility) of the reference sequence and VCF file. All the reference sequences are placed in the $REF folder. In case of VCF files for chromosomes 1-22, each is placed in the separate folder dedicated to the specific chromosome (chr1, chr2, chr3, ...). The VCF files for chromosomes X and Y are placed in the main directory, as they require preprocessing (see option -b) before the main treatment. It should be noticed that the download for the whole genome requires about 1.2 TB of disk space.
- -b
  Preprocessing of the reference sequences and VCF files for X and Y chromosomes (which should be available in advance, see option -a). The reference FASTA sequences for PAR1, PAR2 and nonPAR regions are created with cut-ref (chrX-mal.fa, chrXY-mal1.fa, chrXY-mal2.fa, chrY-mal.fa). The processX program is used to create four VCF files, each describing variants corresponding to one of the X chromosome groups (X-fem, X-mal, X-mal1, X-mal2). The cut utility is used to remove data for the NA21313 (additional male individual, which does not occur in other VCF files) from the VCF file for Y chromosome. Each preprocessed VCF file is placed in the separate, dedicated folder (chrX-fem, chrX-mal, chrXY-mal1, chrXY-mal2, chrY-mal).
- -c
  Creation of VCF minimal file (extension *.min) from VCF file (with use of VCF2VCFmin program).
- -d
  Creation of FASTA consensus sequences (extension *.fa) for all individuals from the reference sequence and VCF minimal file (with use of the VCF2FASTA-d and/or VCF2FASTA-h programs). This is a very time-consuming process and consensus sequences for all chromosomes require about 6.7 TB of disk space.
- -e
  Creation of a variant database (extension *.vd) and byte vectors for all individuals (extension *.bv) from the VCF minimal file (with use of the VCF2VDBV-d and/or VCF2VDBV-h programs).
- -f
  Creation of FASTA consensus sequences (extension *.fa) from the reference sequence, variant database and byte vectors of processed individuals (with use of the VDBV2FASTA program). Consensus sequences for all chromosomes require about 6.7 TB of disk space.

- `-g`
  Reconstruction of the VCF minimal file (conforming with the specification of VCF file) from appropriate reference sequence, variant database and byte vectors of all individuals (with use of the `VDBV2VCFmin-d` and `VDBV2VCFmin-h` programs).
- `-i`
  Compression of collection of byte vectors with the `tgc` program. There are two output files. The name of first output file consists of "chr" prefix, chromosome's name and ".tgc_data" extension. Second output file, with "tgc_desc" extension, is a list of all compressed byte vectors.
- `-j`
  Decompression of collection of byte vectors with the `tgc` program.
- `-k`
  Compression of the variant database with the `tgc_db` program. The output file's name consists of "chr" prefix, chromosome's name and ".tgc_db" extension.
- `-l`
  Decompression of the variant database with the `tgc_db` program.

2.4 Sample scenarios

The description of how to use the provided programs (Section 2.2) in some sample scenarios, with user defined data.

- VCF → VCFmin
  To strip the **data.vcf** file to its minimal form, the corresponding reference file **reference.fa** is needed. Use command:
  `./VCF2VCFmin data.vcf reference.fa`
  Output:
  the VCFmin file **data.vcf.min**.
- VCFmin → VDBV for haploid calls
  To store information from **data.vcf.min** describing haploid calls for chromosome **3** of some organism in the form of variant database together with one byte vector per individual (VDBV), use command:
  `./VCF2VDBV-h data.vcf.min 3`
  Output:
  the variant database **data.vcf.min.vd** and one byte vector **[individual_ID].chr3.bv** for each individual. Individual's IDs are taken from **data.vcf.min**.
- VCFmin → VDBV for diploid calls
  To store information from **data.vcf.min** describing diploid calls (which mudt be phased) for whole chromosome **3** of some organism in the form of variant database together with two byte vector per individual (VDBV), use command:
  `./VCF2VDBV-d data.vcf.min 3`
  Output:
  the variant database **data.vcf.min.vd** and two byte vectors, **[individual_ID].1.chr3.bv** and **[individual_ID].2.chr3.bv**, for each individual. Individual's IDs are taken from **data.vcf.min**.
- VDBV → TGC-archive
  To compress variant database **data.vd** and all related byte vectors into **arch_chr3** TGC-archive, the file with list of all byte vectors to compress is needed. Assuming `data.vd` and all byte vectors to compress are stored in the current folder (with no other byte vectors), the procedure is:
  ```
  ls *bv > bv_list
  ./tgc c arch_chr3 bv_list
  ./tgc_db c data.vd arch_chr3.tgc_db
  ```
  Output: **arch_chr3.tgc** and **arch_chr3.tgc_db**
- TGC-archive → VDBV
  To extract variant database and all byte vectors from **arch_chr3** archive (with **arch_chr3.tgc** and **arch_chr3.tgc_data** files available), the procedure is:
  ```
  ./tgc d arch_chr3
  ./tgc_db d arch_chr3.tgc_db output.vd
  ```
  Output:
  All previously compressed byte vectors (with their original names plus ***.ori** extension) and **output.vd** file with the previously compressed variant database (its name is user's choice).
- TGC-archive → BV for single individual
  To extract the byte vector of single individual (i.e., **[individual_ID].chr3.bv**) from the **arch_chr3** TGC-archive, the file with this single byte vector to decompress is needed, the procedure is: `ls [individual_ID].chr3.bv> decompress_list`

`./tgc d arch_chr3 decompress_list`
Output:
File **[individual_ID].chr3.bv.ori** with required byte vector decompressed.

- VDBV → VCFmin for haploid calls
  To convert variant database **data.vd** and all related byte vectors ***.bv** to VCFmin describing haploid variant calls in whole chromosome **3** of some organism, the reference sequence **reference.fa** is needed. Assuming `data.vd` and all byte vectors are stored in the current folder (with no other byte vectors), the procedure is:
  `VDBV2VCFmin-h reference.fa data.vd 3 1 'ls *bv'`
  Output:
  The VCFmin file **data.vd.min** with haploid calls.
- VDBV → VCFmin for diploid calls
  To convert variant database **data.vd** and all related byte vectors ***.bv** to VCFmin describing diploid calls in whole chromosome **3** of some organism, the reference sequence **reference.fa** is needed. Assuming `data.vd` and all byte vectors, named in the form **[individual_ID].1.chr3.bv** and **[individual_ID].2.chr3.bv** for each individual (diploid), are stored in the current folder (with no other byte vectors), the procedure is:
  `VDBV2VCFmin-h reference.fa data.vd 3 1 'ls *bv'`
  Output:
  The VCFmin file **data.vd.min** with phased diploid call for each successive pair of byte vectors provided.
- VDBV → FASTA for single individual
  To create consensus FASTA sequence for single individual **individual_ID** having its byte vector (i.e., **[individual_ID].chr3.bv**), related variant database **data.vd** and reference sequence **reference.fa** , the procedure is:
  `VDBV2FASTA reference.fa data.vd 1 [individual_ID].chr3.bv`
  Output:
  File **[individual_ID].chr3.bv.fa** with FASTA consensus sequence of **individual_ID**.
- VCFmin → FASTA for haploid calls
  To create consensus sequences for all individuals described in the **data.vcf.min** file (haploid calls), the reference sequence **reference.fa** is required and procedure is:
  `./VCF2FASTA-h data.vcf.min reference.fa` Output: For each individual described in the input VCFmin, **individual_ID.fa** file with its consensus sequence. Individual's IDs are taken from **data.vcf.min**.
- VCFmin → FASTA for diploid calls
  To create consensus sequences for all individuals described in the **data.vcf.min** file with diploid calls, the reference sequence **reference.fa** is required and procedure is:
  `./VCF2FASTA-d data.vcf.min reference.fa`
  Output: For each individual described in the input VCFmin, **individual_ID.1.fa** and **individual_ID.2.fa** files are created with consensus sequences for each of diploid chromosmes. Individuals' IDs are taken from **data.vcf.min**.

3 EVALUATED COMPRESSORS

The parameters used for tools compressing RAW data (collection of consensus sequences in FASTA format):

- 7z a -md1024m — all EOL characters from input data were removed prior to compression
- RLZ files - l 1
- GReEn
- ABRC files ref_file 1 1
- GDC -ma1000000000 -rn1 — denoted as GDC-normal
- GDC -ma1000000000 -rn40 — denoted as GDC-ultra

The parameters used for tools compressing VCF minimal and VDBV data:

- 7z a -md1024m
- gzip -6
- bzip2 -9

As for TGC, the influence of the minimum match length parameter on the compression ratio of byte vectors is shown in Fig. 1. The minimal match length in the final TGC tool is fixed, set to 5.

Maximum memory consumption compressing VCFmin and VDBV data:

- 7z: 10 GB
- gzip: 0.55 MB
- bzip2: 7.8 MB
- TGC: 2.4 GB

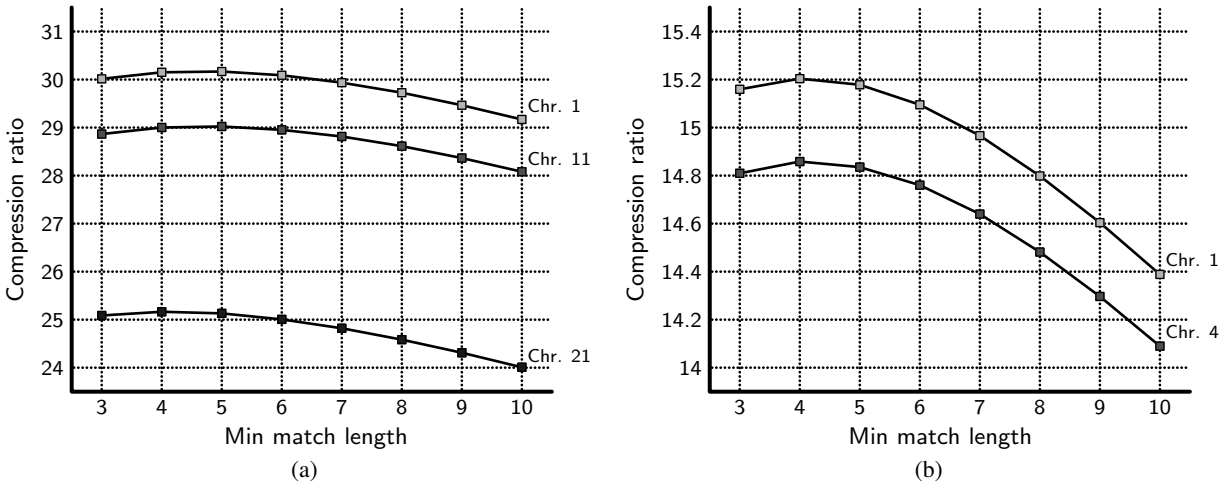

Fig. 1. Influence of the minimum match length parameter on the compression ratio of byte vectors for: (a) *H. sapiens* data and (b) *A. thaliana* data

SpeedGene, tested only on the1000GP data in the LINKAGE/PLINK format, was executed with default parameters.

4 COMPLETE RESULTS

Tables 1 and 2 present complete results of compressing VCFmin and VDBV for all chromosomes with different tools.

Table 1. Evaluation of universal compressors for variant data of 1,092 individuals (both chromosomes of each pair). The input data are in VCF format after removing all non-essential data (VCFmin). For comparison, also the size of the original VCF files from the 1000GP are given. All sizes are in MB. Columns 'c-time' contain compression time in seconds. The number of sequences that can be produced for each chromosome is 2184 (both chromosomes of each pair of 1,092 individuals), except for: †— 1,134 sequences, ‡— 525 sequences, and ⋆ — 1,050 sequences.

Data	VCF	VCFmin	VCFmin + gzip		VCFmin + bzip2		VCFmin + 7z	
	size	size	size	c-time	size	c-time	size	c-time
				H. sapiens				
Chr. 1	93,087	13,249	306.6	321	138.9	6,259	144.2	2,003
Chr. 2	102,434	14,569	330.5	345	149.4	6,892	156.3	2,088
Chr. 3	85,579	12,173	282.3	289	126.3	5,834	130.9	1,767
Chr. 4	84,768	12,056	287.0	290	126.6	5,665	129.7	1,817
Chr. 5	78,364	11,145	256.2	255	114.6	5,281	119.3	1,637
Chr. 6	75,066	10,680	259.9	252	115.3	4,992	117.4	1,619
Chr. 7	68,602	9,761	233.0	227	106.0	4,583	110.1	1,471
Chr. 8	67,654	9,619	219.8	216	98.6	4,465	103.0	1,420
Chr. 9	51,167	7,280	170.6	166	79.2	3,414	83.4	1,108
Chr. 10	58,300	8,295	197.2	203	89.2	3,968	92.3	1,326
Chr. 11	58,671	8,351	196.5	205	89.0	3,960	91.5	1,400
Chr. 12	56,593	8,054	189.0	196	86.0	3,886	88.9	1,349
Chr. 13	42,523	6,049	144.7	149	65.1	2,914	66.7	1,027
Chr. 14	38,957	5,545	130.3	134	59.6	2,670	61.9	920
Chr. 15	34,999	4,982	116.3	118	54.7	2,371	57.6	834
Chr. 16	37,491	5,333	122.5	128	57.9	2,526	62.4	902
Chr. 17	32,386	4,613	107.7	107	50.9	2,187	54.3	802
Chr. 18	33,723	4,797	113.2	112	52.0	2,280	54.2	824
Chr. 19	25,237	3,597	90.8	93	42.9	1,752	45.2	658
Chr. 20	26,476	3,767	87.7	92	40.5	1,807	43.0	624
Chr. 21	16,065	2,286	55.9	58	25.7	1,090	26.9	467
Chr. 22	15,304	2,179	53.9	55	25.9	1,053	27.5	447
Chr. X-fem†	23,798	3,431	74.0	70	35.3	1,550	43.6	556
Chr. X-mal‡	15,317	1,564	36.5	32	17.3	668	23.9	267
Chr. Y-mal‡	163	20	0.4	0.3	0.2	8.8	0.3	2.4
Chr. X-mal1⋆	684	99	3.2	2.8	1.9	51	2.2	18
Chr. X-mal2⋆	61	9	0.3	0.3	0.2	4.4	0.2	1.4
Complete	1,223,470	173,505	4,066.0	4,114	1,849.2	82,128	1,936.8	27,352
				A. thaliana				
Chr. 1	—	4,945	111.3	99	63.3	2,596	86.4	1,002
Chr. 2	—	3,629	80.8	71	45.8	1,710	63.2	699
Chr. 3	—	4,295	96.0	85	55.0	2,041	75.7	822
Chr. 4	—	3,386	76.3	67	43.7	1,608	60.0	700
Chr. 5	—	4,323	100.3	89	56.6	2,062	76.6	852
Chr. C	—	0.09	0.004	0.01	0.003	0.1	0.003	0.04
Chr. M	—	176	2.2	2.8	0.6	90	1.0	27
Complete	—	20,755	466.8	414	265.0	10,107	362.8	4,100

Table 2. Evaluation of universal compressors and the proposed algorithm (TGC) for variant data stored in the intermediate VDBV (Variant database + byte vectors representation) format. All sizes are in MB and times are in seconds. The 'VDBV c-time' column contains the conversion times from VCFmin format to VDBV format. In the remaining columns titled 'c-time', total compression time is given (i.e., 'VDBV+7z c-time' means sum of time of conversion from VCFmin to VDBV and compression of the results with 7z compressor). Note that the variant database (part of the VDBV representation) is of size 933 MB for *H. sapiens* and 320 MB for *A. thaliana*. After compressing by TGC their sizes (included in 'TGC size' column) are about 51.0 MB and 12.5 MB, respectively. The number of byte vectors for each chromosome is 2184 (both chromosomes of each pair of 1,092 individuals), except for: †— 1,134 byte vectors, ‡— 525 byte vectors, and ⋆ — 1,050 byte vectors.

Data	VCFmin	VDBV		VDBV + gzip		VDBV + bzip2		VDBV + 7z		TGC	
	size	size	c-time	size	c-time	size	c-time	size	c-time	size	c-time
					H. sapiens						
Chr. 1	13,249	890.6	136	368.9	311	326.3	251	55.9	1,070	**32.3**	690
Chr. 2	14,569	979.7	145	397.4	334	354.0	268	60.6	1,067	34.2	701
Chr. 3	12,173	818.1	127	334.2	282	289.9	233	48.3	746	28.3	541
Chr. 4	12,056	810.1	125	343.8	283	298.8	245	47.8	957	28.6	609
Chr. 5	11,145	748.9	116	297.4	259	253.0	211	45.5	653	26.0	514
Chr. 6	10,680	717.4	110	296.9	256	254.1	216	44.4	794	25.9	483
Chr. 7	9,761	655.2	104	269.4	229	226.6	190	41.1	560	24.6	626
Chr. 8	9,619	645.7	103	258.5	222	213.5	200	38.3	632	22.3	427
Chr. 9	7,280	488.4	75	200.0	173	158.6	141	31.7	396	18.4	326
Chr. 10	8,295	556.5	88	233.0	196	190.2	160	35.5	540	20.4	399
Chr. 11	8,351	560.1	86	238.7	200	194.8	160	33.1	598	**20.2**	392
Chr. 12	8,054	540.3	84	225.6	189	182.9	162	34.4	533	19.9	505
Chr. 13	6,049	405.5	66	178.6	150	136.0	123	25.3	336	14.8	282
Chr. 14	5,545	371.5	59	154.7	131	115.2	108	23.2	316	13.7	258
Chr. 15	4,982	333.6	53	139.7	120	102.0	101	21.1	266	13.0	231
Chr. 16	5,333	357.0	56	144.9	123	106.9	101	22.6	311	14.0	254
Chr. 17	4,613	308.6	50	126.0	108	90.0	89	21.4	248	12.8	201
Chr. 18	4,797	321.0	52	137.2	117	98.9	95	20.7	274	12.1	232
Chr. 19	3,597	240.5	39	99.6	81	68.3	70	18.2	185	11.0	161
Chr. 20	3,767	252.0	42	101.5	88	68.3	76	15.8	192	9.7	162
Chr. 21	2,286	153.0	25	66.8	54	39.3	44	10.4	107	**6.3**	96
Chr. 22	2,179	145.7	23	61.0	50	36.6	41	10.6	101	6.5	100
Chr. X-fem†	3,431	244.4	38	91.4	78	68.0	71	17.3	191	10.3	126
Chr. X-mal‡	1,564	126.7	17	45.1	40	33.0	37	9.1	95	5.3	42
Chr. Y-mal‡	20	1.6	0.6	0.4	0.9	0.3	0.9	0.2	1.4	0.17	1.5
Chr. X-mal1⋆	99	7.0	1.1	2.5	2.1	1.5	2.1	1.0	4.7	0.73	4.8
Chr. X-mal2⋆	9	0.6	0.3	0.1	0.4	0.1	0.5	0.1	1.0	0.06	0.9
Complete	173,505	11,679.9	1,819	4,813.6	4,075	3,907.0	3,396	733.8	11,176	**431.6**	8,364
					A. thaliana						
Chr. 1	4,945	405.8	64	117.3	121	102.9	107	34.7	395	**26.1**	231
Chr. 2	3,629	303.9	47	85.7	91	73.6	78	24.9	275	19.2	128
Chr. 3	4,295	360.7	57	99.5	108	86.7	95	29.9	361	22.9	150
Chr. 4	3,386	281.4	42	79.7	92	67.9	72	23.7	271	**18.2**	120
Chr. 5	4,324	357.8	57	106.0	110	92.5	97	30.1	363	23.1	175
Chr. C	0.09	0.01	0.6	0.010	0.7	0.005	0.7	0.004	3.0	0.01	2.8
Chr. M	176	12.9	2.7	0.8	3.2	0.6	3.7	0.3	5.1	0.18	7.9
Complete	20,756	1,722.5	270	489.0	526	424.1	453	143.5	1,673	**109.7**	815

5 SOME ALGORITHMIC DETAILS

Elements of pairs and triples of matches in byte vectors are encoded by an arithmetic coding based on their contextual statistics, as follows:

- binary flags (distinguishing between matches and literals) — the context is 9 most recently encoded flags,
- literals — the context is a concatenation of 3 most recent flags and the number of set bits in two recently encoded literals,
- match lengths, the more significant part, i.e., $\lceil log_2(len - mml + 1) \rceil$) — the context is calculated as a weighted average: the most recently encoded value (weight $1/4$) and the previous context occurrence (weight $3/4$),
- match lengths, the less significant part — the context is the value of the most significant part,
- vid, 8 most significant bits — no context,
- vid, 8 least significant bits — the context are the most significant bits of vid.

The contexts used for encoding variant database are as follows:

- differences between position — no context,
- type of variant — the context is the type of recently encoded variant,
- literals — no context,
- lengths — no context.

```
pos:   0  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
S0 :  96  0   8   0   1 200   0   0   0   0 142  12  69   0   0  32

S0 encoded as a run of literals:
(0, 96), (0,  0), (0,  8), (0,  0), (0,  1), (0,200), (0,  0), (0,  0), (0,  0), (0,  0), (0,142),
(0, 12), (0, 69)  (0,  0), (0,  0), (0, 32)

Short-key sampled positions: 0, 3, 6, 9, 12[, 15]
Long-key sampled positions:  0, 4, 8[, 12]

Short keys added to HT1, in order:
(96, 0; 0), (0, 1; 3), (0, 0; 6), (0, 142; 9), (69, 0; 12)

Long keys added to HT2, in order:
(96, 0, 8, 0, 1; 0), (1, 200, 0, 0, 0; 4), (0, 0, 142, 12, 69; 8)


pos:  0  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
S1:  96  0   8   6   1 204   0   0   0   0 142  12  77   0   0  40

S1 encoded using the current HT1 and HT2 contents:
(0, 96), (0,  0), (0,  8), (0,  6), (0,  1), (0,204), (1, 0, 6), (0, 77), (0,  0), (0,  0), (0, 40)

Short keys added to HT1, in order:
(96, 0; 0), (6, 1; 3), (0, 0; 6), (0, 142; 9), (77, 0; 12)

Long keys added to HT2, in order:
(96, 0, 8, 6, 1; 0), (1, 204, 0, 0, 0; 4), (0, 0, 142, 12, 77; 8)


pos:  0  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
S2:  96  0   8   6   1 204   0  16   0   0 142  12  69   0   0  41

S2 encoded using the current HT1 and HT2 contents:
(1, 1, 7), (0, 16), (1, 0, 7), (0, 41)

Short keys added to HT1, in order:
(96, 0; 0), (6, 1; 3), (0, 16; 6), (0, 142; 9), (69, 0; 12)

Long keys added to HT2, in order:
(96, 0, 8, 6, 1; 0), (1, 204, 0, 16, 0; 4), (0, 0, 142, 12, 69; 8)


pos:  0  1   2   3   4   5   6   7   8   9  10  11  12  13  14  15
S3:  82 96  24   6   1 204   0  18   0   0 142  12  69  33   0  41

S3 encoded using the current HT1 and HT2 contents:
(0, 82), (0, 96), (0, 24), (1, 2, 4), (0, 18), (0, 0), (1, 2, 4), (0, 33), (0,  0), (0, 41)

Short keys added to HT1, in order:
(82, 96; 0), (6, 1; 3), (0, 18; 6), (0, 142; 9), (69, 33; 12)

Long keys added to HT2, in order:
(82, 96, 24, 6, 1; 0), (1, 204, 0, 18, 0; 4), (0, 0, 142, 12, 69; 8)
```

Fig. 2. Compressing four sequences ($S_0, \ldots, S_3$) of length 16 bytes each. Two hash tables, H_1 and H_2, are used. H_1 adds short keys (length 2 in the example and in TGC implementation) while H_2 adds long keys (length 5 in the example and length 11 in TGC implementation). The extracted keys are in sampled positions (to reduce memory requirements); in steps 3 and 4 for short and long keys, respectively, in the example, and in steps 4 and 4 in the real implementation. The [] brackets in the listed positions denote skipped positions (the key would exceed the sequence boundary). The position of a substring extracted from a sequence is part of the key added to the hash tables, presented after a semicolon. The encoding of each sequence is presented with tuples; if the first value (flag) is 0, then we have a literal and what follows is its byte value. If the flag is 1, then we have a match and what follows is the referenced sequence ID followed by the match length. Match lengths are not shorter than the minimal match length, set to 4 in the example and to 5 in the real implementation.