A phasing algorithm targeted at dairy cattle populations having genotyped ancestors with many genotyped offspring

M. Dahl[1*] and O.F. Christensen[1]

[1] Center for Quantitative Genetics and Genomics, Aarhus University, 8830 Tjele, Denmark, [*]mdahl@qgg.au.dk

Abstract
In dairy cattle, the prevalence of genomic testing is now widespread, and commonly both sires and many offspring in the form of children, grandchildren, and great-grandchildren, are genotyped. Calling phases from genotypes is useful for several reasons, in particular for construction of haplotypes and imputation of missing genotypes. In this paper, we present a new computationally efficient method to call phases of grandfathers and great-grandfathers when enough genotyped offspring are present. The method consists of first identifying shared haplotypes between ancestor and offspring, followed by phasing segments based on homozygote genotypes in offspring, and finally merging haplotype segments to call entire ancestral haplotypes. The implementation is based on using bit vectors and fast binary computations. Results show that the method is effective in finding shared haplotypes to be used for phasing.

Introduction
Phasing using homozygotes in relatives and offspring has explored by others. Kong *et al.* (2008) introduced Long Range Phasing (LRP) that phases based on likely identity by descent (IDB) segments of the chromosome, using homozygote markers of fixed length genotype segments, to call paternal and maternal alleles using surrogate parents. The method is implemented in AlphaPhase (Hickey *et al.* 2011, Money *et al.* 2020). Using offspring for phasing was presented in Ferdosi *et al.* (2014) and Nettelblad (2012). They both show that very accurate phasing can be done on parents when enough genotyped offspring are present. The aim of our paper is to present a computationally efficient method using likely IBD segments to call phases of grandfathers and great-grandfathers when enough genotyped offspring are present.

Materials & Methods
Under the assumption of one recombination per meiosis, the average length of a recombination-free sequence of markers passed from an ancestor is half the size of the shared haplotypes. These recombination-free sequences can be used for phasing. The method presented here, is based on identifying parts of the inherited chromosome in offspring, where a haplotype segment from an ancestor is inherited and no recombination has taken place. After ancestor haplotypes have been computed, the offspring genotypes are used to do internal phasing of the called haplotypes. This is done by the following sequence of operations:

1. Identify likely shared haplotypes between ancestor and offspring, using opposing homozygotes, allowing for small percentage of errors.
2. Split the chromosome into segments of equal length.
3. For each segment extract genotype segment from all offspring where segment is fully contained in likely shared haplotype. Set minimum requirements for number of offspring.
4. Split offspring genotype segments into two recombination free groups.

5. Use these two groups to call phase on the ancestor haplotypes.
6. Do internal phasing of identified haplotype segments using heterozygotes in the called haplotype segments.

Test data was generated using QMSim (Sargolzaei and Schenkel 2009). A chromosome with 3273 markers was created, i.e. as chromosome one on the Illumina BovineSNP50 BeadChip v3. A nucleus population producing bulls and a production population of cows was created. Then 10 generations of dams in the production population were mated to 10 generations of sires from the nucleus population. Offspring in generation 10 was used for phasing, for grandfather phasing generation 9 sires were used, for great-grandfathers, sires in generation 8 were used. Random positions in genotypes were removed or changed to simulate missing genotypes and genotyping errors.

Step 1: Identifying Shared Haplotypes.

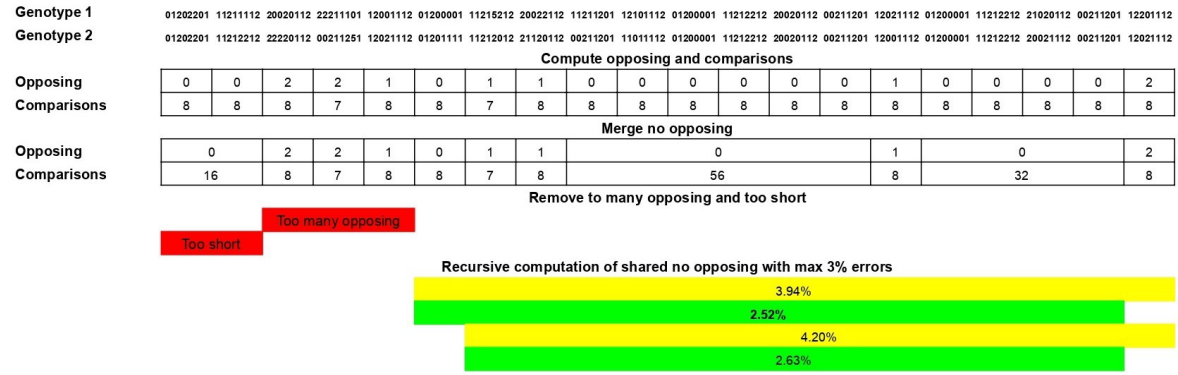

Figure 1. Identification of long shared no opposing SNPs between two genotypes
Genotypes coded as 0,1,2. Parameters: Segment length 8; Minimum Length 24 SNPs; Error window size 16; Max errors in window 3; Min no opposing length 80; Max. error percentage 3.0%. Method: Split genotypes in segments of a length that enables fast computation – in this example 8**;** Compute opposing and comparisons**;** Merge no opposing**;** Remove too many opposing and too short**;** Recursive computation of longest shared no opposing that has less than 3% errors.

Fast identification of shared haplotypes is important for obtaining acceptable runtimes. The method stores genotypes as three bit vectors **aa**, **bb** and **ab**, i.e. a bit set mark whether the genotype has AA, BB or AB SNP in the bit position. This enables use of binary instructions **and**, **or**, **not** or **xor**, when comparing genotypes. These instructions are very fast in most programming languages. Opposing homozygotes between two genotypes can be computed as the number of bits set in the expression: $((aa_1$ **and** $bb_2)$ **or** $(aa_2$ **and** $bb_1))$.
The method is as follows: a) Split the two genotypes into segments with fast binary instructions for fast computing; b) Create arrays of **aa**, **bb,** and **ab** bit segments for all genotypes; c) Do a pairwise comparison between segments by computing number of opposing and number of SNPs present in both genotypes; d) Merge adjacent segments with zero opposing homozygotes; e) Set an error window size and the maximum allowed number of errors in the window; f) Scan all adjacent segments with non-zero opposing homozygotes. If adjacent segments with the length of the window or less, contain more than the maximum allowed number of errors, then remove the adjacent segments from the array; g) Set a minimum length of pairwise shared no opposing haplotypes between the two genotypes and a

maximum error percentage; h) Group segments into groups of adjacent segments. Remove all groups with a total length less than the minimum length; i) For the remaining groups, compute the maximum length of no opposing homozygotes using elements in the group that have fewer errors than the maximum error percentage. This can be done by recursion. Return the longest identified. The method is illustrated in Figure 1.

Steps 2 to 5: Phasing segments based on homozygotes in offspring.
Following step 1, all pairs of offspring and sire have had their shared haplotypes computed. An offspring and a sire may share haplotypes from a shared common ancestor. These can be mistaken as the inherited haplotype. To decrease the possibility of this, we chose segments that are sufficiently long. In the experiments presented here, the simulated genotypes were split into 12 segments. Offspring where an ancestor is grandfather or great-grandfather more than once will not be used for phasing. To get variation, pedigrees of all the offspring must have the grandfather or great grandfather as the only ancestor common in offspring pedigrees. For each of the segments the method does the following: a) Find all elements where the segment is entirely in the shared haplotype and enough offspring exist - store them; b) Split them into the two largest groups that split, around the heterozygotes in the ancestor's genotype. Allow for errors; c) Call phases using homozygotes in the two groups and homozygotes in the ancestor. Allow for errors. In steps b and c fast bit operations are used.

Step 6: Internal phasing of the computed haplotype segments.
The haplotype segments phased in step 2 to 5 has no internal phase. In this section a method to call internal phase is described.
The method consist of: a) Ensure that all haplotype segment pairs have 3 heterozygotes, by doing the following: In all ancestor haplotype pairs with less than 3 heterozygotes, delete heterozygote calls; Merge all homozygote haplotype pairs to an adjacent segment pair. b) For each pair of adjacent haplotype segments, called left segment and right segment: Find the three rightmost heterozygote positions in left haplotype pair; Create two short haplotypes with three heterozygote positions; Similar for the right haplotype pair; c) Find the internal phase of left and right haplotype pair making a vote between all offspring segments that contains all the positions in the two short haplotypes from the left and right haplotype pair.
The proposed method was implemented in a Python3.10 program where bit-vectors are stored in a bytearray, that support fast binary computations.
A test was run on a HP notebook (Intel i7-8565U CPU – 16GB RAM, cache size 8192 KB). On that hardware on average 1468 genotype comparisons per second were executed, during the test run. Parameter's settings in program test were as follows. Missing marker percent in genotypes: 3%; Genotype SNP count: 3273; Error window size: 272: Max opposing count in window: 6; Min no opposing length: 648; Max errors percent: 1% for 0.1 SNP error pct, 2% for 1% error pct.

Results

Table 1. Phasing of grandfathers

Simulated Offspring	Random SNP call percentage	Phased great grandfathers	Haplotype calls	Haplotype call percent	Haplotype error percent	Haplotype segment calls	Haplotype segment call percent	Haplotype segment error percent	Missing haplotype segment phases
50	0.1	1000	1988	99.569	0.077	24000	99.572	0.051	6
200	0.1	100	200	99.552	0.211	2400	99.557	0.073	0

50	1.0	999	1954	99.241	0.471	23976	99.241	0.342	22
200	1.0	100	196	99.197	0.603	2400	99.203	0.319	2

Table 2. Phasing of great-grandfathers

Simulated Offspring	Random SNP call percentage	Phased great grandfathers	Haplotype calls	Haplotype call percent	Haplotype error percent	Haplotype segment calls	Haplotype segment call percent	Haplotype segment error percent	Missing haplotype segment phases
100	0.1	1000	1982	99.562	0.137	24000	99.564	0.051	9
400	0.1	100	196	99.549	0.078	2400	99.553	0.088	2
100	1.0	998	1960	99.207	0.633	23952	99.202	0.330	18
400	1.0	100	198	99.209	0.309	2400	99.217	0.313	1

It is clearly seen that the identified shared haplotypes can be used to call haplotype segments in grand- and great-grandfathers while correcting genotyping errors. In most cases the shared haplotypes can be used for calling the entire haplotype.

Discussion

A new and computationally fast method for identifying shared haplotypes between genotype individuals is presented, and we have shown that the method is effective in finding shared haplotypes that can be used for phasing. Future work involves an extensive comparison with other software, in particular AlphaPhase (Hickey *et al.* 2011, Money *et al.* 2020) and Fimpute (Sargolzaei, Chesnais and Schenkel (2014). Preliminary investigations show that our algorithm is superior in terms of both speed and accuracy compared to AlphaPhase, but memory usage should also be a criterion for comparison.

References

Ferdosi, M. H., Kinghorn, B. P., van der Werf, J. H. J and Gondro, C. (2014) Genet. Sel. Evol 46:11 https://doi.org/10.1186/1297-9686-46-11

Hickey J. M., Kinghorn, B. P., Tier, B., Wilson, J. F., Dunstan, N. *et al.* (2011) Genet. Sel. Evol 43:12. https://doi.org/10.1186/1297-9686-43-12

Kong A., Masson G., Frigge M. L., *et al.* (2008) Nat Genet. 40(9):1068-75 https://doi.org/10.1038/ng.216

Money, D., Wilson, D., Jenko, J. Whalen, A. Thorn, S. *et al.* (2020) Genet. Sel. Evol 52:38. https://doi.org/10.1186/s12711-020-00558-2

Nettelblad, C. (2012) BMC Genetics 13:85 https://doi.org/10.1186/1471-2156-13-85

Sargolzaei, M., Chesnais J. P. and Schenkel F. S. (2014) BMC Genomics 15:478. https://doi.org/10.1186/1471-2164-15-478

Sargolzaei, M. and F. S. Schenkel (2009) Bioinformatics 25: 680-681. https://doi.org/10.1093/bioinformatics/btp045