Bioinformatics
doi.10.1093/bioinformatics/xxxxxx
Advance Access Publication Date: XXX XXX XXXX
XXXXXXXXXXXX

Subject Section XXXX

HaploForge: A Comprehensive Pedigree Drawing and Haplotype Visualisation Web Application

Mehmet Tekman [1], Alan Medlar [2], Monika Mozere [1], Robert Kleta [1]*, and Horia Stanescu [1]

[1] Division of Medicine, University College London, London, NW3 2PF, UK and
[2] Institute of Biotechnology, University of Helsinki, Helsinki, 00014, Finland.

*To whom correspondence should be addressed.

Associate Editor: XXXXXXX

Received on XXXXX; revised on XXXXX; accepted on XXXXX

Abstract

Motivation: Haplotype reconstruction is an important tool for understanding the aetiology of human disease. Haplotyping infers the most likely phase of observed genotypes conditional on constraints imposed by the genotypes of other pedigree members. The results of haplotype reconstruction, when visualised appropriately, show which alleles are identical by descent despite the presence of untyped individuals. When used in concert with linkage analysis, haplotyping can help delineate a locus of interest and provide a succinct explanation for the transmission of the trait locus. Unfortunately, the design choices made by existing haplotype visualisation programs do not scale to large numbers of markers. Indeed, following haplotypes from generation to generation requires excessive scrolling back and forth. In addition, the most widely-used program for haplotype visualisation produces inconsistent recombination artefacts for the X chromosome.

Results: To resolve these issues, we developed HaploForge, a novel web application for haplotype visualisation and pedigree drawing. HaploForge takes advantage of HTML5 to be fast, portable and avoid the need for local installation. It can accurately visualise autosomal and X-linked haplotypes from both outbred and consanguineous pedigrees. Haplotypes are coloured based on identity by descent using a novel A* search algorithm and we provide a flexible viewing mode to aid visual inspection. HaploForge can currently process haplotype reconstruction output from Allegro, GeneHunter, Merlin and Simwalk.

Availability: HaploForge is licensed under GPLv3 and is hosted and maintained via GitHub.

Web Application and Source Code: `https://github.com/mtekman/haploforge`

Supplementary information: Supplementary data is available from *Bioinformatics* online.

Contact: r.kleta@ucl.ac.uk

1 Introduction

Linkage analysis, together with haplotype reconstruction, is used to identify putative locations of disease traits. Linkage analysis tests whether a given gene region co-segregates with the trait locus, whereas haplotype reconstruction infers the phase information that is lost during genotyping, i.e. the parental origin of each allele. In doing so, regions of interest can be found using linkage analysis and those regions delineated with inferred recombinations from haplotype reconstruction. Once a region has been identified, candidate genes can be selected for sequencing based on information from sequence databases (tissue-specific expression, homology, etc) or, if no candidate presents itself, all genes from the identified region can be screened for mutations using, for example, exome sequencing (Bockenhauer *et al.*, 2012).

Many parametric linkage analysis programs also perform haplotype reconstruction based on maximum likelihood. However, to integrate these analyses together requires advanced visualisation methods to intuitively display haplotypes together with the pedigree structure and to colour haplotypes based on identity by descent (IBD).

There are many programs available for haplotype visualization, such as the highly cited HaploPainter (Thiele and Nürnberg, 2005) as well as the more recent Family Genome Browser (Juan *et al.*, 2015). However, our experience with HaploPainter has shown that viewing haplotypes inline with the pedigree does not scale to large numbers of markers.

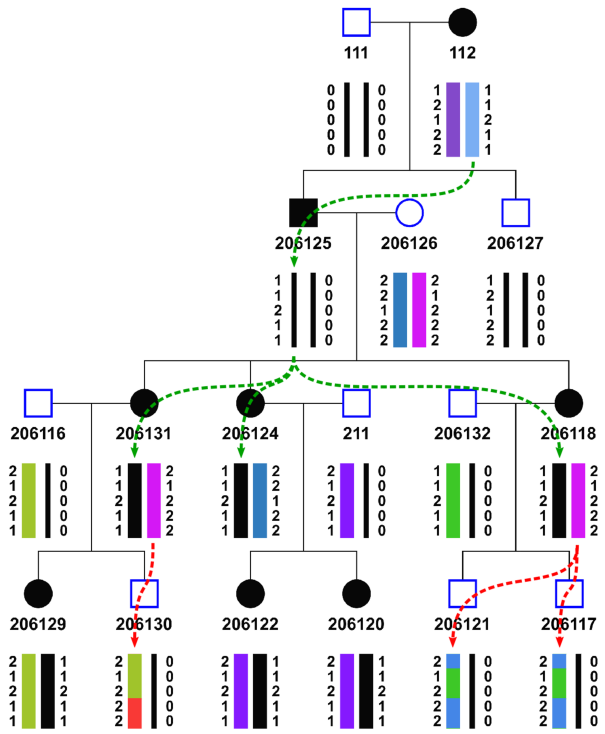

Fig. 1. HaploPainter visualisation of a five marker X-linked analysis. Colours indicate identity by descent. Arrows are overlayed to show the true flow of genetic data based on genotypes, with green showing inconsistent colouring between successive meioses and red showing erroneous inheritance.

Indeed, to compare haplotypes between generations requires excessive scrolling and the user has to re-identify the same region of interest over and over again in each successive generation. In addition, HaploPainter does not always correctly display which alleles are IBD, creating inconsistent recombination artefacts for the X chromosome. As shown in Figure 1, the last generation of individuals (particularly 206130, 206121 and 206117) appear to have undergone multiple recombinations within a relatively short genetic distance (< 1 cM).

HaploPainter fails to properly account for the fact males have only a single X chromosome and, therefore, can appear to inherit from their father or even from an undisplayed paternal allele (see blue allele in individuals 206121 and 206117 in Figure 1).

To resolve these issues we present HaploForge, a novel web application for haplotype visualization and pedigree drawing. HaploForge is designed specifically for navigating high numbers of markers, providing a more intuitive viewing mode compared to other programs. Secondly, we present a novel A* search-based method that ensures IBD information is correctly displayed for all chromosomes, including the X chromosome.

Finally, HaploForge is web-based and therefore runs in any HTML5-compatible web browser and does not require local installation. Despite being web-based, it is fast and the user experience is similar to a native application, utilizing menus and a drag and drop interface.

2 Approach

HaploForge is a comprehensive web application for haplotype visualisation and pedigree drawing.

Here we will enumerate and expand upon the core features.

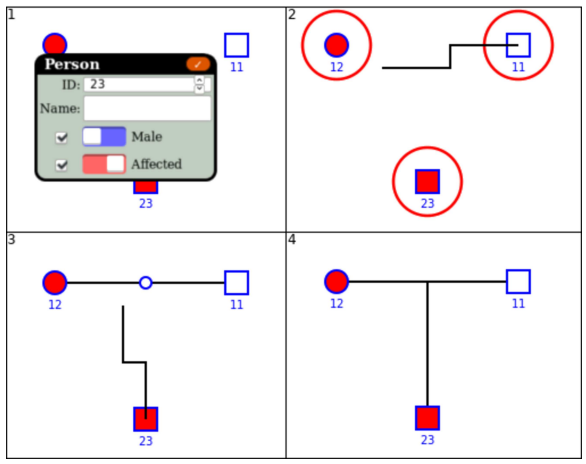

Fig. 2. Pedigree drawing view show the four stages of creating a pedigree: (1) Adding individuals and modifying their properties, (2) Joining mates with a mateline to anchor points made visible with red circles, (3) Joining offspring to their parents through a childline to anchor points made visible with white circles, (4) Completing a trio.

2.1 Pedigree Drawing

Pedigrees are drawn with a simple drag and drop interface, and are compliant with the Pedigree Standardization Work Group (PSWG) specification (Bennett *et al.*, 1995, 2008). The standard is already familiar to clinicians and allows individuals in the pedigree to be annotated with patient metadata.

Individuals are added to the pedigree either through context-dependent sidebars or user-customizable keyboard shortcuts, and the properties of that individual (sex, affection status, etc) can then be edited from a dialog box. Relationships between individuals are added by drawing *matelines* and *childlines*. Matelines indicate marriages and childlines connect children to their parents' mateline. Lines snap to context-dependent anchor points that become visible when adding relationships (Figure 2). Both members of each mateline are vertically aligned with one another and move together as a single unit. Siblings bound to the same mateline are similarly aligned automatically.

Projects can contain multiple families, and complex consanguineous relationships are automatically detected and represented with double-lines. Pedigrees can be loaded from and saved to local browser storage. Pedigrees can be imported and exported in standard LINKAGE (pre-makeped) format.

2.2 Haplotype Visualisation

While HaploPainter visualises haplotypes inline with the pedigree, in HaploForge haplotypes are displayed in a separate viewing mode.

Selected individuals are aligned horizontally and grouped by family. Haplotypes are displayed underneath each individual, allowing for side-by-side comparison, irrespective of generation. The relatedness of individuals can be optionally displayed, with relationship lines stating their degree of separation (see Figure 3).

Haplotypes are displayed within a viewport that defines the locus of interest across the genotypes of all selected individuals, the contents of which are outlined on the chromosome overview. The overview consists of a red vertical bar representing the entire length of the chromosome, and it is overlayed with a region indicator which has a height and position that maps to the size and location of the viewport on the chromosome.

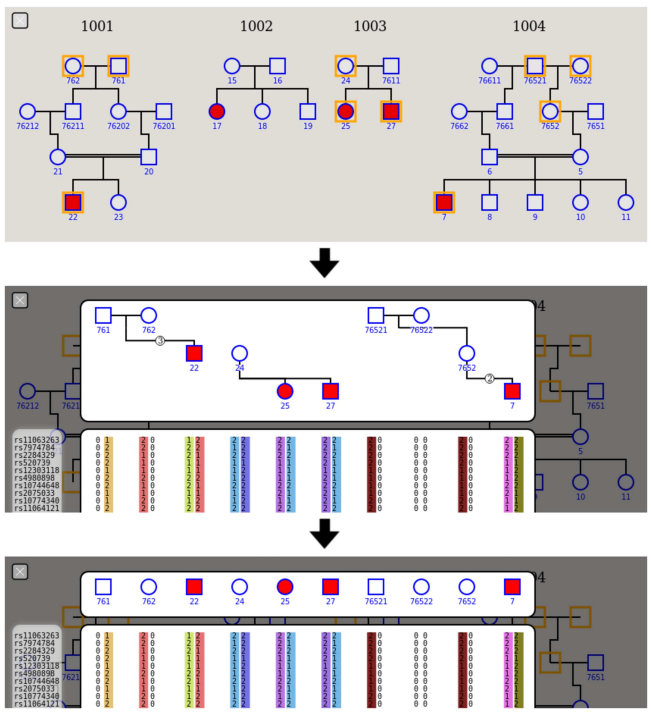

Fig. 3. (Top) Selection view enabling the subselection of individuals across multiple pedigrees. (Middle) Comparison view displaying the haplotypes of selected individuals, connected by lines indicating their degrees of separation from one another. (Bottom) Comparison view representing the same genotypes but with individuals vertically aligned.

The viewport can be manipulated by a variety of different interactions as shown in Figure 4: dragging the region indicator in the overview, scrolling the mouse wheel over the genotypes, or using keyboard shortcuts for fast (PageUp/PageDn) and slow (Up/Down arrows) movement. Precise modes of adjustment are also provided; dragging and releasing haplotypes at a desired vertical position, shifting to the next/previous point of recombination, or specifying a pair of markers from a dropdown list.

The chromosome overview also allows for the size of the viewport to be adjusted through dragging the top and bottom handles on the region indicator as shown in Figure 4 (Bottom). By default, the size of the viewport is locked to the size of the window. However, if the viewport is made bigger than the window, it can be scrolled using the mouse wheel.

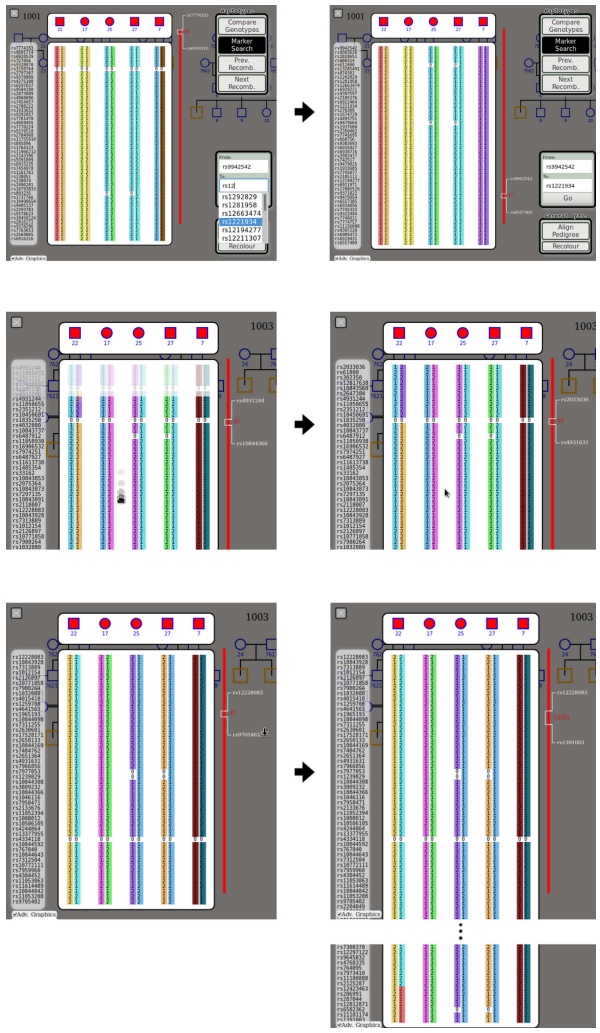

Fig. 4. Modes of interaction: (Top) scrolling the genotypes in the comparison view by dragging and dropping the haplotypes, updating the marker positions in the chromosome overview; (Middle) adjusting the viewport by manually specifying a locus of interest between two markers; (Bottom) resizing the viewport by dragging the upper or lower handle on the chromosome overview, along with a Shift keyboard modifier for permitting slower movement for precision. A viewport larger than the size of the window can be scrolled using the mouse wheel, but the default behaviour can be reverted by holding Ctrl and dragging one of the handles to snap the viewport back to the window size.

2.3 IBD Colouring

In HaploForge, haplotypes are coloured by IBD by converting the task of resolving ambiguous parentage into a path-finding problem and performing A* search to determine the path with least recombinations. A* search is an efficient path-finding algorithm used in real-time mapping applications (Algfoor *et al.*, 2015).

In a connected graph with weighted edges, an optimal path between two nodes is found by minimising the total edge cost.

A* search is a best-first search algorithm that, in the process of finding the optimal path, maintains a "frontier" of nodes from which the node deemed most likely to be the next intermediate node on the path to the target node is selected. The search procedure is admissible on condition that the estimated cost to the target node is not greater than the true cost from the next intermediate node to the target node, under the following heuristic: $f(n) = g(n) + h(n)$, where n is an intermediate node on the path, $g(n)$ is the cumulative cost of the path (from start node to n), and $h(n)$ is the heuristic that estimates the lowest cost from n to the target node (Hart *et al.*, 1968).

In a genomic context this can be conceptualised as a multi-layer network graph, where edges only exist between consecutive layers. Each layer represents an individual marker locus, and each node a distinct founder allele (Figure 5). The algorithm traverses from one end of a chromosome to the other under the heuristic of minimizing the number of recombinations. A maximum of $2f$ nodes are possible in each layer, where f is the number of founders. This number is often far smaller due to the manner in which the graph is initialised (see Section 3.2.1).

2.4 File Format Support

HaploForge accepts phased genotypes from both binary markers (SNPs) and polymorphic markers (e.g. VNTRs and STRs), in vertical or horizontal pre-MAKEPED or modern LINKAGE-based formats with chromosome pairs delineated on adjacent lines.

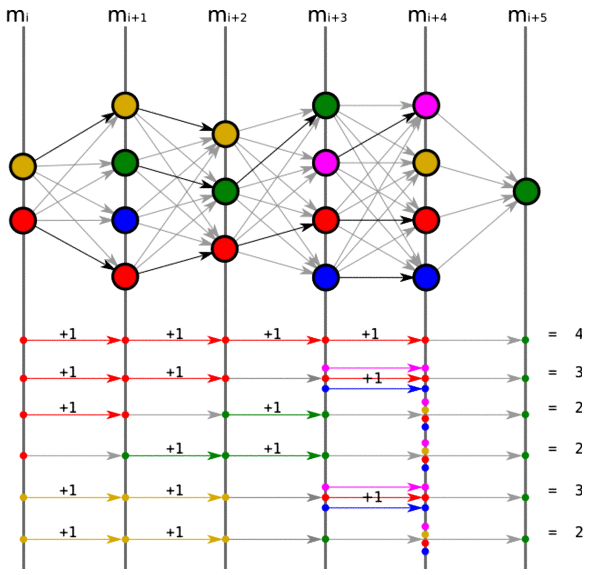

Fig. 5. (Top) A multi-layer network graph depicting five founder alleles as uniquely coloured nodes within a marker locus stretching from m_i to m_{i+5}. Black arrows depict desired contiguous founder-allele stretches, and grey arrows indicate recombinations from one founder-allele to another. (Bottom) Six possible routes explored by the search algorithm, with contiguous stretches being rewarded $+1$ to the total path sum. The first route has the largest path sum of 4 and is the most optimal path in the range considered.

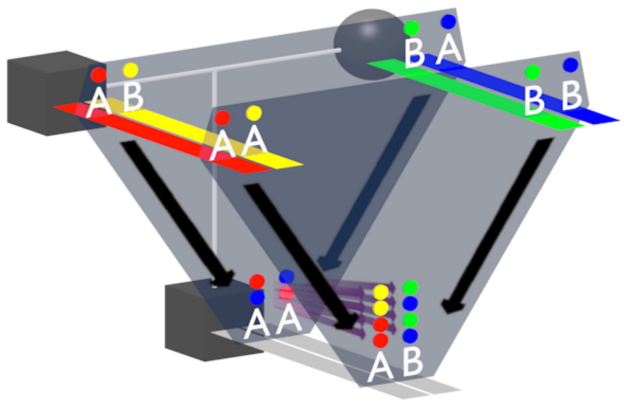

Fig. 6. Parent-Offspring trio for a 2-marker locus. The two vertical layers display the independent founder-allele group initialisation occurring at each marker locus, with the proliferation of founder-alleles (coloured dots) from parent-to-offspring under valid genotype configurations. The A* search process then bisects these layers for each non-founder allele, to maximize the length of a contiguous founder-allele.

HaploForge can additionally utilise supplementary gene flow information output by specific programs. Some applications incorporate this information within the main haplotypes output file, whereas others provide it in a supplementary file. Merlin (Abecasis *et al.*, 2002), for example, outputs founder alleles in a file called `gene.flow`. Allegro (Gudbjartsson *et al.*, 2005) and SimWalk (Sobel *et al.*, 2002) both output the optimal descent graph, stating the gene flow between generations. The haplotypes from GeneHunter (Kruglyak *et al.*, 1996) and Merlin do not include sex data, requiring it be inferred through parentage for all but the last generation whose sex is declared unknown. This information can be provided separately. Further marker data (e.g. SNP ID, genetic distance, etc) is displayed upon discovery and preserved across successive sessions in local storage.

3 Methods

3.1 Web Technology

HaploForge is implemented using HTML5 and JavaScript. Unlike other haplotype visualisation programs that require local installation (including installation of dependencies), HaploForge runs in any compliant web browser. Features like A* search IBD colouring (described in detail in Section 3.2) make use of JavaScript's typed arrays to eliminate the redundancy of the default numeric float type, compacting large numeric sets into small 8 bit decimal arrays. Fast 2D graphics rendering is performed using the HTML5 canvas-based KineticJS library (`http://kineticjs.com`). KineticJS renders graphics to layers which are implemented as separate canvas elements.

HaploForge uses two layers for drawing operations, each acting either as a framebuffer or backbuffer where necessary to limit the number of redraw operations.

Animation is used to transition between the pedigree drawing and haplotype comparison modes. Visual effects can be disabled if, for example, the browser does not support hardware acceleration and performance is degraded. At time of writing, Webkit-based browsers (Chrome, Safari) offer better performance than the Gecko-based Firefox.

3.2 A* search IBD Colouring

The procedure to ensure that haplotypes are coloured appropriately to reflect identity by descent occurs at the level of parent-offspring trios, ordered through a top-down pass of the pedigree ensuring that offspring are not processed before their parents. For each trio, processing is split into two distinct phases: initialising the graph, and determining the optimal path using A* search.

3.2.1 Graph Initialisation

Founder alleles are inherited by non-founders transitively via their parents. For each non-founder allele, we define the set of all valid founder allele assignments that could be made based on the genotypes of the parents. This is termed the *founder allele group*. In the trivial case, the founder allele graph only contains a single element and, therefore, the founder allele inherited is unambiguous. However, given that there tend to be multiple valid paths by which a given allele could have been inherited, there will often be multiple possible founder alleles.

Although our implementation of A* search requires phased genotypes, (i.e. resulting from haplotype reconstruction, see next section), the graph initialization procedure does not make use of the phase information, relying solely on the founder allele groups as shown in Figure 6.

3.2.2 Finding the Optimal Path

During graph initialisation, genotypes are processed vertically, parent-to-offspring, to populate the founder allele groups at each marker locus. The A* search algorithm traverses the chromosome, from marker to marker, to find the optimal path defined by the lowest cost combination of founder allele assignments. The search operates under the heuristic of maximising contiguous stretches (or *paths*) of the same founder allele across multiple marker loci, minimizing the number of recombinations. The branching nature of the search requires we consider multiple paths, each of which expands the frontier of possible founder allele groups. Valid paths are restricted to those that meet minimum stretch requirements and fall within accepted founder allele assignments outlined in Algorithm 1.

begin
  $X \leftarrow$ parental exclusion set of illegal founder-alleles
  frontier $\leftarrow$ set of active paths, initialized to first founder-allele
  complete $\leftarrow$ set of completed paths, initialized as empty
  while *frontier* > 0 **do**
    $p \leftarrow$ first path in *frontier*
    $F \leftarrow$ set of founder-alleles at marker locus size$(p)+1$
    for $f \in F$ **do**
      $s \leftarrow$ perform lookahead and count contiguous stretch of f
      if $(s >$ *minStretch*$)$ and $(f \notin X)$ **then**
        $e \leftarrow$ extend path p by length s with founder-allele f
        if size$(e) >$ size$($*markers*$)$ **then**
          push e to *complete*
        else
          push e to *frontier*
        end
      end
    end
    sort *frontier* by desc. length and truncate up to *maxNumPaths*
  end
  sort *complete* by desc. number of recombinations
  return first path in *complete*
end

Algorithm 1: A* search upon a single chromosome pre-initialised with a set of potential founder-alleles at each marker locus.

The founder alleles assigned during graph initialisation do not state whether they are inherited maternally or paternally. To address this, we define a *parental exclusion set* that encapsulates all the founder alleles present in either maternal or paternal alleles. The algorithm then selects from these sets to determine where the allele originated from. In the case of a consanguineous relationship, each maternal and paternal exclusion set subtracts against the union of both sets to permit the inclusion of shared groups.

A maximum working set of eight examined paths are expanded upon with paths added/removed according to the number of recombinations. Paths with the same number of recombinations are included but discounted from the working set in order to encourage path diversity. Once an active path reaches the final locus, it is moved into the set of complete paths from which the path with the lowest number of recombinations will be selected at the end of the procedure.

3.3 Comparative Analysis

The purpose of haplotype visualisation is to help distinguish between loci whose segregation is concordant with the disease trait from those that are not. In such regions there will be a clear distinction in IBD information between affected and unaffected individuals. To give researchers an overview of their data, HaploForge defines a per locus score based on the extent to which haplotypes for affected individuals differ from unaffected individuals

The scoring function is specified by the type of zygosity selected by the user. Under a heterozygous setting, only a single allele from each individual would need to match. For homozygousity and compound heterozygousity, individuals need to have the same set of alleles, with the former requiring those alleles additionally be homozygotes.

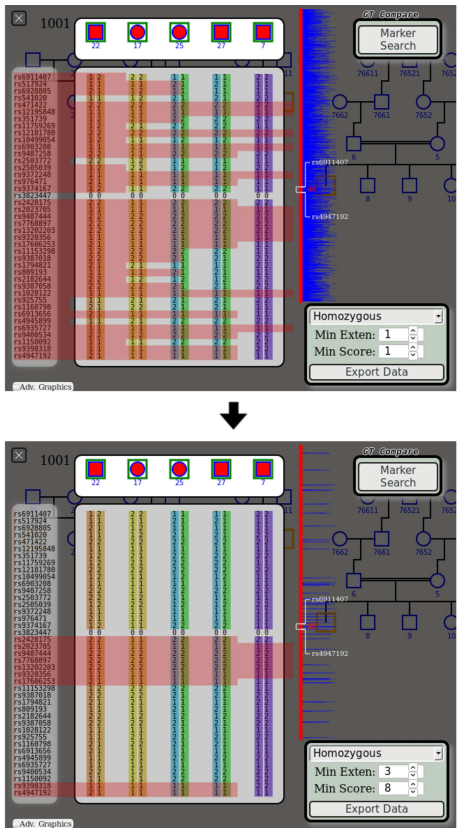

Fig. 7. Comparative view displaying identity scores under a homozygous context, with (Top) no filtering and (Bottom) a minimum score threshold of 8 and a minimum peak size of 3 markers. Scores are overlayed on both the genotypes and the chromosome overview.

This is defined explicitly in the equations below, where for a given marker locus m and family f, scores are generated for each of the three zygosity scenarios as follows:

$$\text{SCORE}\ (m, f) = \sum_i g(A_i, m) - \sum_j g(U_j, m) \tag{3.1}$$

where:

$A =$ Affecteds in f, $\quad U =$ Unaffecteds in f

$A_0 =$ Initially selected affected individual in f

$H_{p,m} =$ Set of haplotypes for individual p at locus m

and g refers to one of:

$$het(p, m) = \begin{cases} 1, & \text{if } \exists\ h \in H_{p,m}\ : h \in A_{0,m} \\ 0, & \text{otherwise} \end{cases}$$

$$chet(p, m) = \begin{cases} 1, & \text{if } H_{p,m} = A_{0,m} \text{ and } |A_{0,m}| > 1 \\ 0, & \text{otherwise} \end{cases}$$

$$hom(p, m) = \begin{cases} 1, & \text{if } H_{p,m} = A_{0,m} \text{ and } |A_{0,m}| = 1 \\ 0, & \text{otherwise} \end{cases}$$

The graphical representation is then the summation of this score across all families, to be overlayed upon both the genotypes and the chromosome overview (Figure 7). Additional refinement can be performed by the user such as specifying a minimum score threshold to filter out less significant peaks, and setting an extension lower-bound to yield broader peaks.

The mode ultimately facilities in the identification of extended haplotypes by calculating IBD between alleles for individuals within the same family and combining it with an IBS score across different families, ultimately outlining a founder effect as a common peak across selected individuals, should such a characteristic exist.

4 Discussion

HaploForge provides a unified environment to create, analyse, and visualize pedigrees together with their associated haplotypes. Pedigree creation allows for large families to be drawn using the mouse and exported or saved to local storage between sessions. Processing for IBD colouring is highly efficient and the results viewable across multiple families simultaneously. We provide several methods to inspect the displayed haplotype data including: displaying haplotypes between user-specified flanking markers, skipping between recombination points and scoring by haplotype consistency with the disease model.

4.1 Linkage Analysis Post High-Throughput Sequencing

It should be noted that the importance of genetic linkage analysis is often understated in the epoch of high-throughput sequence analysis. Though positive logarithm-of-the-odds (LOD) scores are often touted as a necessity to pinpoint loci of interest, negative LOD scores for weak linkage analyses are also useful in this regard under the notion that they indicite where *not* to look.

Combining regions of interest outlined by linkage analysis can greatly assist in filtering out a vast majority of potential causative variants when performing sequence analysis, and linkage analysis paired with haplotype reconstruction is still extremely valuable in this respect, especially in the detection of a founder effect across multiple families.

4.2 Path-finding approach

The A* search processes each sister chromatid independently of one another, however the correct resolution of one chromatid depends upon the correct resolution of the other due to the mutual-exclusivity of the maternal or paternal exclusion set they are processed against. Due to the non parent-specific manner in which HaploForge initialises founder-alleles, this may prompt the A* search to reprocess both chromatids with swapped parental exclusion sets if a path cannot be determined initially. Though seemingly costly, the lack of strict phasing during the founder-allele group initialisation stage and the serial processing of chromatids provides a greater flexibility in resolving a larger scope of genetic disorders. In the future this could be adapted to explore monosomy, trisomy, and tetrasomy cases.

4.3 Visualisation Accuracy

The haplotype visualisation performed by HaploForge was compared with HaploPainter. For all autosomal pedigrees analysed, the same points of recombination were identified from Allegro (ihaplo.out), GeneHunter (haplo.chr), Merlin (merlin.chr), and Simwalk (HEF.ALL) output files. Beyond the simple pedigrees, we have used HaploForge with four non-trivial families: autosomal dominant (27 members, 23-bit); a highly consanguineous autosomal recessive (24 members, 29-bit); and an X-linked dominant (17 members, 15-bit).

Fig 8 provides a side-by-side comparison of the same X-linked pedigree shown previously (Figure 1), but showing a larger number of markers, where only HaploForge shows the correct IBD colouring. HaploPainter normally displays haplotypes inline with the pedigree, but this was modified in Figure 8 for comparison.

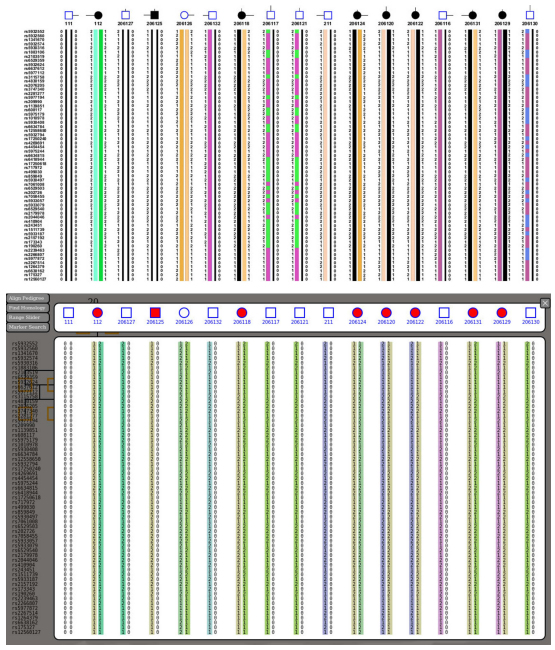

Fig. 8. A comparison of the X-linked dominant pedigree showing a mid-region of chrX spanning 72 markers. (Top) HaploPainter with output modified only for horizontally alignment, and (Bottom) HaploForge showing all members via default comparison view.

4.4 Benchmarks

Haplotype reconstruction runs most often in conjunction with genetic linkage analysis, which tend to be Lander-Green derived for calculating LOD scores, such that the complexity scales linearly with the number of markers and exponentially with the number of individuals (Lander and Green, 1987).

As a result, haplotype datasets are typically quite small due to the computational cost, and independent chromosomal segregation splits these sets further such that a single haplotype output file may contain no more than 2000 markers for large chromosomes in a typical analysis. Resolving and rendering data from real genotyping sets therefore is quite trivial.

In order to properly test the capabilities of HaploForge, an exhaustive benchmarking suite was developed within the application that generates large complex pedigrees and simulates meioses and recombination events over successive generations. A total of 1282 random pedigree sets were generated in the benchmark process, seeded by four input variables; number of root founder couples, allele size, maximum number of generations, and the probability of a consanguineous relationship. The stochastic nature of the generation process meant that an analysis seeded by the same variables would often yield a different number of total individuals in a family for those settings, inciting the need for performing multiple runs on the same input parameters.

Figure S1 within the Supplementary Material plots this data as the number of total individuals against rendering times, showing how the performance diverges significantly between allele sizes for even moderately-sized pedigrees. Small allele-sized pedigrees yield linear performances and load almost immediately ($\leq$ 1 second) in standard use-cases ($\leq$ 400 individuals, $\leq$ 1000 markers), though undergo a more logarithmic performance profile past a threshold of 4000 individuals. Larger allele-sized pedigrees presented a low linear performance throughout but succeeded in handling extremely large allele sets (1 million markers) for more nuclear families, making it potentially useful in the examination of large whole-genome sets with trios such as those provided by the 1000 Genomes Project (Consortium, 2015).

The side-by-side comparative analysis of haplotypes for large allele sets did not pose any problems with the viewing of haplotypes, but was impractical in the examination of large pedigree sets, and the end-user is henceforth strongly encouraged to initially choose a more manageable subselection of individuals to analyze from the selection view.

4.5 Comparison with Additional Haplotyping Software

HaploPainter dominated much of prior discussion due to the better feature-overlap with HaploForge (pedigree creation, in-house haplotype recombination detection and rendering), but various different programs were considered for comparison. A dissection of the two main contenders are outlined below, but a review of many others can be viewed in Table 1 within the Supplementary Material.

4.5.1 Family Genome Browser

The recent Family Genome Browser (FGB) (Juan *et al.*, 2015) is a hybrid Java/Javascript web and desktop application which visualizes variant data under a pedigree context and hosts many useful features such as potential recombination events, IBD visualization, parent-of-origin highlighting, and linkage disequilibrium (LD) annotations. Java is not required to run the application within the browser, suggesting that the Java component operates on the server and the client interacts with the Javascript front-end, which in itself relies on extensive frameworks such as *jQuery* and *Raphael*. The desktop deployment of their utility is a Java servlet wrapper of their web-based version.

In contrast, HaploForge is written entirely in Javascript and relies only on the KineticJS framework for rendering to the HTML5 canvas. Drawing functions are abstracted from the underlying graphics library, allowing it to be easily exchanged in favour of another without having to compromise the rest of the code. By having no actual dependents, HaploForge can be effortlessly deployed to any platform (Web, Desktop, Mobile) that supports a web-engine under a myriad of different coding strategies (Java, C++, Python, Perl, etc.), making it highly portable across operating systems.

The FGB is primarily sequence-variant oriented in the formats that it accepts (VCF / CGF / BED), but does not process any common haplotype formats (see Section 2.4), suggesting that recombination events are limited to examining the pre-set phasing within the variant data and evaluating haplotypes on an individual marker-by-marker basis as opposed to resolving beyond a more global scope. The parent-of-origin highlighting feature indicates the founder of a given marker-variant, but this is implicitly represented in HaploForge via distinct colouring where the origin of a given non-founder allele block is inferred by the colouring of the founder allele it originated from (e.g. a small RED allele block in a last-generation individual can be traced back to the entire RED allele in a first-generation founder).

The inclusion of LD data is desirable in a variant-centric analysis but less so in a more haplotype/IBD-driven context, and HaploForge was originally developed to explore how alleles segregated within a pedigree setting only.

4.5.2 Cryllic

Cryllic is an application similar to HaploPainter for drawing pedigrees and rendering haplotypes, but it is ultimately closed-source and provides (paid) Windows-only binaries making portability and access difficult. The program as of version 2 can render pedigrees as large as 10,000 individuals up to 250 markers on each chromosome (Strom, 1996), a feat that HaploForge can easily accomplish as shown in Section 4.4.

4.5.3 Program Summary

Applications that can both draw pedigrees and resolve haplotypes are visibly uncommon, and HaploForge replenishes this software void by providing a host of beneficial features. In addition to its client-only in-browser deployment and its novel A* recombination detection algorithm, HaploForge provides side-by-side evaluation of haplotypes (as opposed to the more cumbersome generational view), automatic scrolling to points of recombination, an adjustable window of scope operated through a manner of different gestures and search tools, and a very useful identity-mapping tool to assist in the search for a founder effect across multiple families by IBS metrics.

4.6 Privacy

HaploForge operates entirely *in-situ* within the browser, with analyses restricted to a single user. In the interests of scientific collaboration, it is likely that the end-user would want to share their analysis with other researchers working on the same project. Due to the sensitivities of patient data, however, as well as the possibility of identifying individuals based on pedigree structure alone, HaploForge was designed with the intention of not requiring any client-server communication after the web application has loaded. Server-side processing required by other web applications (such as FGB) are not a concern in HaploForge, removing any need for the end-user to transfer sensitive patient data onto remote servers via potentially insecure protocols. The discretion of patient data is ultimately left to the user, and we provide the option to strip patient names and other annotations on export.

4.7 Future Work

HaploForge was built on top of KineticJS because of its stability; active development being frozen since 2014. However, in order for HaploForge to benefit from performance improvements it will need to migrate to one of the primary alternatives, either ConcreteJS (`http://concretejs.com/`), by the author of KineticJS or KonvaJS (`https://github.com/konvajs/`) that both other distinct features and advantages that will need to be evaluated.

Future versions of HaploForge will aim to integrate the visualization and creation modes to provide more flexibility, for example, to allow for modifying an existing pedigree after haplotype data is loaded. Additional features could include SVG export and selective visualisation of multiple regions to help produce publication quality figures.

The inclusion of LD annotations could also be considered, as it would provide a welcome broader population insights to the haplotype analysis from the default family-only setting.

Acknowledgements

R.K. is supported by St. Peter's Trust for Kidney, Bladder and Prostate Research, the David and Elaine Potter Charitable Foundation, Kids Kidney Research, Garfield Weston Foundation, Kidney Research UK, the Lowe Syndrome Trust, the Mitchell Charitable Trust, and the European Union, FP7 (grant agreement 2012-305608 "European Consortium for High-Throughput Research in Rare Kidney Diseases (EURenOmics)".

Conflict of interest: None declared.

References

Abecasis, G. R., Cherny, S. S., Cookson, W. O., and Cardon, L. R. (2002). Merlin-rapid analysis of dense genetic maps using sparse gene flow trees. *Nature Genetics*, **30**(1), 97–101.

Algfoor, Z. A., Sunar, M. S., and Kolivand, H. (2015). A comprehensive study on pathfinding techniques for robotics and video games. *International Journal of Computer Games Technology*, **2015**, 7.

Bennett, R. L., Steinhaus, K. A., Uhrich, S. B., O'Sullivan, C. K., Resta, R. G., Lochner-Doyle, D., Markel, D. S., Vincent, V., and Hamanishi, J. (1995). Recommendations for standardized human pedigree nomenclature. *Journal of Genetic Counseling*, **4**(4), 267–279.

Bennett, R. L., French, K. S., Resta, R. G., and Doyle, D. L. (2008). Standardized human pedigree nomenclature: update and assessment of the recommendations of the National Society of Genetic Counselors. *Journal of genetic counseling*, **17**(5), 424–433.

Bockenhauer, D., Medlar, A. J., Ashton, E., Kleta, R., and Lench, N. (2012). Genetic testing in renal disease. *Pediatric Nephrology*, **27**(6), 873–883.

Consortium, T. . G. P. (2015). A global reference for human genetic variation. *Nature*, **526**(7571), 68–74. Article.

Gudbjartsson, D. F., Thorvaldsson, T., Kong, A., Gunnarsson, G., and Ingolfsdottir, A. (2005). Allegro version 2. *Nature Genetics*, **37**(10), 1015–1016.

Hart, P. E., Nilsson, N. J., and Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, **4**(2), 100–107.

Juan, L., Liu, Y., Wang, Y., Teng, M., Zang, T., and Wang, Y. (2015). Family genome browser: visualizing genomes with pedigree information. *Bioinformatics*, **31**(14), 2262.

Kruglyak, L., Daly, M. J., Reeve-Daly, M. P., and Lander, E. S. (1996). Parametric and nonparametric linkage analysis: a unified multipoint approach. *American Journal of Human Genetics*, **58**(6), 1347–1363.

Lander, E. S. and Green, P. (1987). Construction of multilocus genetic linkage maps in humans. *Proceedings of the National Academy of Sciences*, **84**(8), 2363–2367.

Sobel, E., Papp, J. C., and Lange, K. (2002). Detection and Integration of Genotyping Errors in Statistical Genetics. *American Journal of Human Genetics*, **70**(2), 496–508.

Strom, C. M. (1996). Cyrillic 2 program for pedigree drawing. *Journal of Assisted Reproduction and Genetics*, **13**(7), 611–612.

Thiele, H. and Nürnberg, P. (2005). HaploPainter: a tool for drawing pedigrees with complex haplotypes. *Bioinformatics*, **21**(8), 1730–1732.