Published online 18 January 2017

Nucleic Acids Research, 2017, Vol. 45, No. 9 ***e67***
doi: 10.1093/nar/gkw1358

PBrowse: a web-based platform for real-time collaborative exploration of genomic data

Peter S. Szot[1,2,†], Andrian Yang[1,3,†], Xin Wang[1,3], Chirag Parsania[4], Uwe Röhm[2], Koon Ho Wong[4] and Joshua W. K. Ho[1,3,*]

[1]Victor Chang Cardiac Research Institute, Darlinghurst, NSW 2010, Australia, [2]School of Information Technologies, University of Sydney, NSW 2006, Australia, [3]St. Vincent's Clinical School, University of New South Wales, Darlinghurst, NSW 2010, Australia and [4]Faculty of Health Sciences, University of Macau, Macau SAR, China

Received May 04, 2016; Revised December 23, 2016; Editorial Decision December 23, 2016; Accepted December 29, 2016

ABSTRACT

Genome browsers are widely used for individually exploring various types of genomic data. A handful of genome browsers offer limited tools for collaboration among multiple users. Here, we describe PBrowse, an integrated real-time collaborative genome browser that enables multiple users to simultaneously view and access genomic data, thereby harnessing the wisdom of the crowd. PBrowse is based on the Dalliance genome browser and has a re-designed user and data management system with novel collaborative functionalities, including real-time collaborative view, track comment and an integrated group chat feature. Through the Distributed Annotation Server protocol, PBrowse can easily access a wide range of publicly available genomic data, such as the ENCODE data sets. We argue that PBrowse represents a paradigm shift from using a genome browser as a static data visualization tool to a platform that enables real-time human–human interaction and knowledge exchange in a collaborative setting. PBrowse is available at http://pbrowse.victorchang.edu.au, and its source code is available via an open source BSD 3 license at http://github.com/VCCRI/PBrowse.

INTRODUCTION

Consider the following scenario: a group of scientists from various institutions around the world participate in a project that involves generating new genome-wide ChIP-seq, RNA-seq and whole genome sequencing data. The scientists involved in data generation and data analysis are located in different countries. At regular intervals, meetings are held to discuss the data and share analysis results. To help visualize the data, a bioinformatician may either take screenshots of the genome browser view at many different resolutions and email them to their colleagues, or load their data to a web-based genome browser, such as Ensembl or UCSC, before emailing their colleagues the URL to the view or session. In this common situation, collaborative analysis is carried out through assessing static images or shared screen sessions, typically driven by a single member of the team. The setup time for such events is not trivial and is not conducive for collaborative exploration of the data.

Many current genome browsers were developed to facilitate exploration of genome-scale data by individual researchers. Nonetheless, as illustrated in the above example, an isolated exploration of data may not be fully harnessing the wisdom of the crowd within a research consortium. As these types of collaborative projects are becoming more commonplace, it is increasingly important to facilitate real-time collaborative exploration of genome-scale data. The central goal of PBrowse is to provide an intuitive web-based platform where genome-scale data can be shared and visualized collaboratively in real-time with minimal setup costs. In this paper, we discuss how we implement a real-time collaborative genome browser that fulfils this need.

Evolution of genome browser technology

Genome browsers were developed out of a need to simplify the visualization and analysis of an increasingly vast amount of genomic data. Many public genome browsers also provide data access for the research community. The first generation of genome browsers followed the dynamic-server static-client model of content delivery (Figure 1A), based on the assumption that the client is a slow machine, incapable of performing complex operations. Thus, the server would prepare and pre-render all elements of a page before delivering it to the client, which then displays it as static content. The human genome browser at the University of California at Santa Cruz (UCSC) (1) was one of the first browsers implementing this model. It allows for reliable display of any portion of the genome, at any scale.

*To whom correspondence should be addressed. Tel: +61 2 9295 8645; Fax: +61 2 9295 8601; Email: j.ho@victorchang.edu.au
†These authors have contributed equally to this work as the first authors.

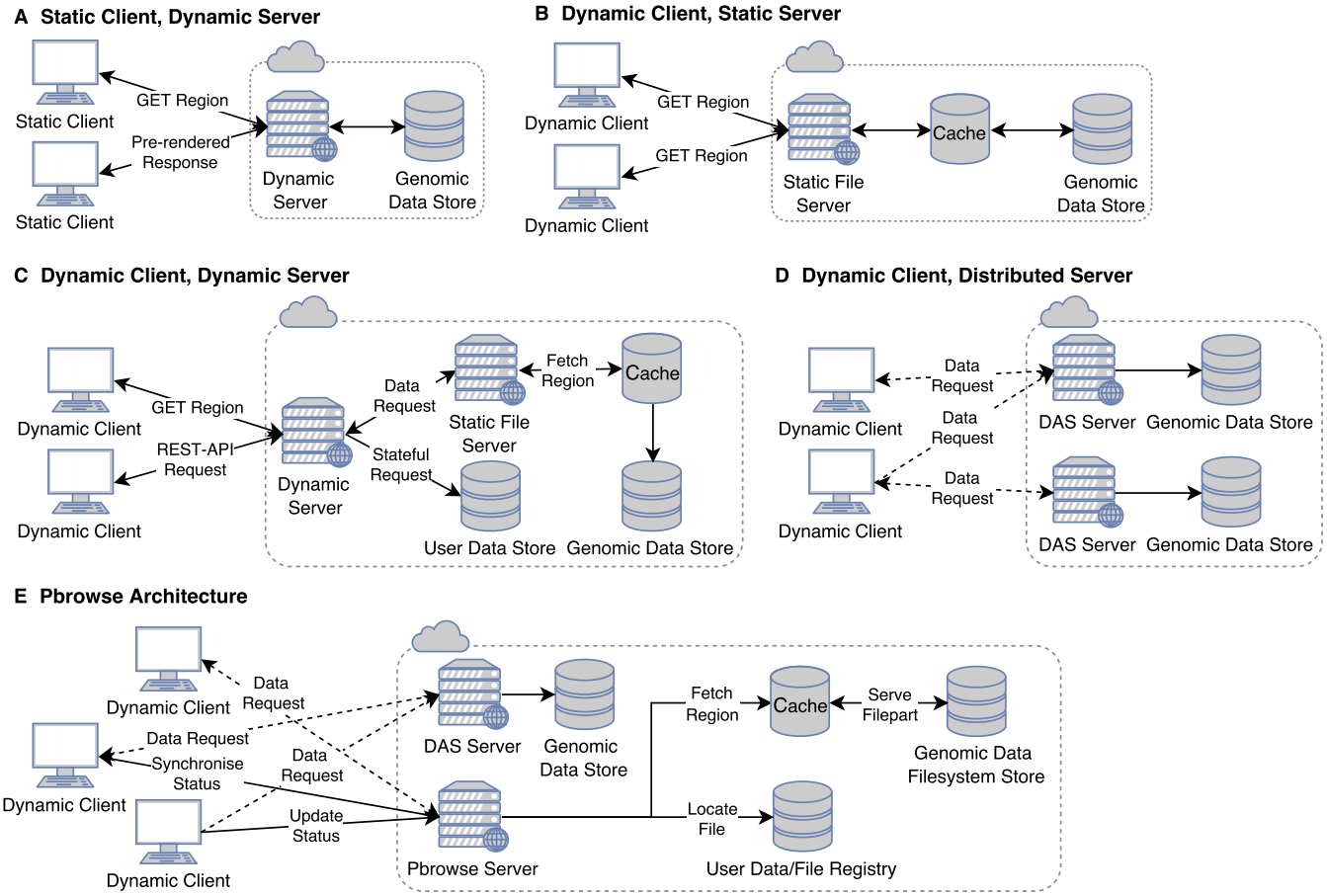

Figure 1. Comparison of different system architecture of web-based genome browsers.

It also enables a simple form of collaboration by allowing the upload of annotation tracks to be viewed by other research groups. While it was originally designed as a 'human' genome browser, it has over the years been upgraded to enable visualization of data from various species. Another major system is the Ensembl project (2), which was designed to offer an integrated, extensible and reusable framework for generating, storing, retrieving and displaying genomic annotation data. Like the UCSC genome browser, it has since received a multitude of updates and has grown dramatically. G-Compass (3) is another browser following the client-server model that attempts to specifically resolve the problem of comparing multiple disparate genomes simultaneously in a meaningful way. The goal of their browser is to provide effective comparisons between human and selected model organisms, promoting the exchange of functional information between different organisms.

With the advent of better client-side technologies for retrieving and processing data through in-browser javascript programming language, the client is no longer completely reliant on the web-server to pre-process data. This effectively frees the server from a significant burden of work thus reducing the server overhead, while utilizing the enhanced power of the client computers to achieve greater responsiveness (Figure 1B). JBrowse (4) is one such genome browser that enables smooth animated panning, zooming, navigation and track selection. This important feature preserves the user's sense of location by avoiding discontinuous transitions.

While a dynamic client can perform a great deal of processing, it is limited in what it can achieve alone. The use of a dynamic server allows for the implementation of significantly more complex functionalities and persistent cross-session interactions between multiple clients (Figure 1C). Developers of ABrowse (5) adopted this model, extending the efficient browser framework established by JBrowse, by enhancing interactivity, opening access to more data sources and providing support for inter-user collaboration in terms of enabling non-real-time commenting and annotation. Genome Maps (6) is another such system, designed specifically to address the browser efficiency issues under increasing data loads brought about by high-throughput sequencing technology. It aims to provide a near seamless browsing experience through real-time navigation along chromosomes, all the way down to individual base pairs. GBrowse (7), like Genome Maps, was designed specifically to handle the large scale of next generation sequencing (NGS) data sets. ChromoZoom (8) is yet another browser that handles high volume NGS data with a focus on using modern web technologies to improve user experience. It facilitates the exploration of experimental data by researchers,

enabling the visualization of custom results alongside a dynamic representation of curated genomic information.

With the ever increasing power of client side rendering technologies, the role of the web-server as a central point for accessing data is becoming increasingly redundant. As data sets continue to grow, storage of genomic data on a single server becomes problematic. One solution is to distribute the load. An example of this is depicted in Figure 1D. The Dalliance browser (9) addresses these concerns, opting for a completely self-contained and embeddable module approach, offering high levels of interactivity which is competitive with specialized desktop applications such as the integrative genomics viewer (IGV) (10), all while running entirely within the web browser. Other examples of client side genome browsers include IGV.js (https://github.com/igvteam/igv.js), a web-based implementation of the standalone IGV application and WashU Epigenome Browser (11).

Design of a real-time collaborative genome browser

Many genome browsers mentioned previously have some elements of collaboration integrated, though they are always limited in some regards. In some systems, it may be possible to share a view via URL or upload an annotation track for other users to make use of; none of which can be considered as 'real-time' collaboration. These options also require multiple collaborators to upload or configure their browsers.

The concept of distributed real-time editing has existed for many years and numerous algorithms have since been proposed as solutions to the three primary problems: causality preservation, user intention preservation and convergence. In the case of a collaborative genome browser, we assume the need of users to edit the raw data is limited. Instead, only the viewpoint and superficial changes to tracks are allowed. This reduces the complexity of the operation space which we must consider; in particular, insertion or deletion will have no effect and can be totally ignored.

To address the requirements of high-responsiveness given non-deterministic latency, Sun et al. (12) have proposed a multi-versioning approach. Multiple conflicting operations are made permissible by allowing the creation of branching 'copies'. This preserves the intention of all participating collaborators while providing optional consistency. In a browser context, shifting views while collaborating in different directions is a conflicting operation that will produce two diverged states. Each user will see the position of the others and can resynchronize themselves at will.

An alternative is to utilize the idea of user-roles and access rights (13). In this way, the session owner is considered to have administrator privileges and can grant permissions to other participants, allowing or disallowing them to perform specific operations. This can prevent conflicts from happening so long as users are permitted to only perform certain compatible operations concurrently. In the genome browser context, however, there are too few valid operations to require such restrictions.

The recently developed collaborative web-based Java IDE (CoRED) (14) utilizes many of the aforementioned techniques. CoRED implement differential synchronization whereby the server stores the shared document, and each client has a separate shadow copy of the document both on server and client side, along with the copy they are editing. When edits are made, the changes are processed into a patch that is merged and redistributed by the server. Patches are small as they contain only the modified data, keeping a high responsiveness as well as enabling high concurrency. CoRED as a whole, serves as a useful example of the essential technologies utilized in a modern collaborative editor.

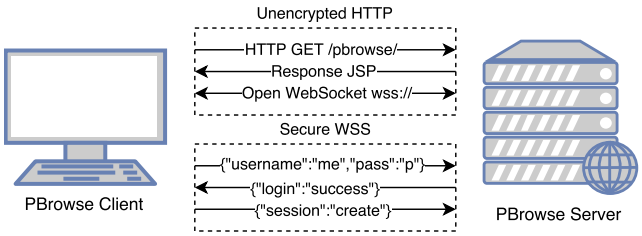

Figure 2. User login workflow. Transition from HTTP transfers to WebSocket SSL communication.

PBrowse is an amalgamation of the best features of many widely used genome browsers and the latest technology for real-time collaboration (Figure 1E). It effectively addresses most of the weaknesses of previous genome browsers, combining highly responsive and flexible collaborative features, with a completely transition free and intuitive browsing experience. Sharing and exploring data collaboratively has been streamlined so that researchers can spend more time finding new insights.

MATERIALS AND METHODS

PBrowse genome browser implementation

The client side web front-end of PBrowse is written in javascript utilizing the Bootstrap and jQuery libraries, on top of the javascript-based Dalliance genome browser. On the server side, PBrowse is implemented using Java, with Jetty Web Server for deploying web archive (war) file. PBrowse makes use of a MySQL database instance to store a range of persistent data, including meta-data for tracks, with the track files themselves being saved to disk in a predetermined location.

PBrowse uses secure WebSocket connections rather than the RESTful web service for server-client communication as it allows for the server to communicate to the client without the need for the client to request updated data. This provides real-time communication between the server and client and is the basis for the real-time collaborative feature of PBrowse. The information flow between client and server is depicted in Figure 2. PBrowse also provides a single REST endpoint for delivering data to the Dalliance genome browser as Dalliance currently has no support for WebSocket connections. Further details of the PBrowse architecture are described in Supplementary Note 1.

RESULTS AND DISCUSSION

The PBrowse genome browser

The front-end of PBrowse is based on the Dalliance genome browser (9). It is embedded and fully contained within a single webpage, with all browsing actions performable without the need for any extra navigation or refreshing of the view. The interface is designed with responsiveness in mind, allowing simultaneous and asynchronous loading of multiple data sources, ensuring transitions to different regions of the genome are as seamless as possible. Data for regions upstream or downstream of the viewed genomic region are loaded pre-emptively in anticipation of a user panning the view. An example of the PBrowse collaborative view can be seen in Figure 3.

Individual tracks can be configured within the browser, depending on their type, allowing for customization of coloration, size, restrictions on the number of displayed features, among numerous other options. The genome browser stores all track configuration information in the web-browser's local storage, allowing past browsing sessions to be seamlessly resumed if the page is closed. Custom tracks are also restored to their last known state, including any style changes made by the user.

Real-time collaboration

The primary purpose of this browser is to enable real-time collaboration among multiple users to improve the efficiency of data exploration tasks. This is achieved by having multiple users connected while viewing a data set, as part of a parallel session. All the users in the session can see the real-time status of all other users, which is updated whenever a user performs an action that modifies their view. This process is depicted in Figure 4. A user can subscribe to the updates of other participants in the same session. Any updates to the current genomic position of the user will cause the other participants who have subscribed to the user to automatically synchronize their view to the new position. Otherwise, the other participants will only see an update to the user's record of the position in their status.

When creating a new parallel session, the creator is assigned the role of leader and as such, has more privileges than any other users, i.e. followers. The leader can perform a number of actions affecting other users in the session, including: modifying the session's public status, inviting new users to join or temporarily removing (kicking out) existing users, blocking other users from joining or re-joining the session, preventing follower interaction, selecting a new replacement leader or even terminating the session entirely. If the current leader leaves the session, the leadership position will be passed onto a random follower. However, if all users leave the session, the session will immediately be terminated and cannot be re-joined.

By default, the session leader is not subscribed to the updates of any user although they may choose to do so. A newly connected user, however, will initially be subscribed to updates from the leader, though they may also modify their subscription set at will. Subscriptions affect only view changes; track modifications affect every user in the session equally and cannot be ignored. This covers all instances whereby the set of visible tracks in the genome browser are modified, including the reordering, removing or adding of new tracks.

The visibility of a session is controlled by its privacy status. If marked public, the session will appear as a listing available to all PBrowse users. They will see its identifier and whether the session requires a passcode to join. Private marked sessions are not listed, requiring a user to know both the identifier and passcode – if required. Users invited to join a session need only to accept the invitation.

In addition to synchronizing views, all the members of a collaborative session are granted temporary access to files uploaded by all other group members. This includes those who were part of the session but since left it. It affects all user uploaded files, except those explicitly marked 'only-me' – which as the name indicates, are visible only to the uploader. This facilitates the quick sharing of new data, requiring the file to be uploaded only once before it is viewable by multiple collaborators. While participating in a collaborative session, the users may also communicate directly via the session chat feature. It allows them to send short text messages to all other participants instantaneously. Users also receive session status updates as special messages, i.e. when users join, leave, are kicked out, etc. This can be useful for communicating intent or discussing collaborative tasks.

A software usability study was conducted where 14 volunteer testers with varying experience in using genome browser were asked to evaluate the usefulness of the real-time collaboration feature (see Supplementary Note 2 for details). The testers reported that the collaborative functionality of PBrowse had helped them to save time in reaching consensus decisions as a group and that the real-time collaborative functionality worked as expected without delay (Supplementary Tables S1 and S2). Furthermore, the testers indicated that they will recommend PBrowse to their collaborators, especially if they plan on working collaboratively (Supplementary Tables S1 and S2).

Track comments

The ability of users to leave comments on data tracks is another core feature of the collaborative framework of PBrowse. In this case, however, it is both real-time and deferred. When a user adds a custom track, they unlock the option to register a new, persistent comment on the currently viewed genomic region. The commenter can specify any message to accompany it, or simply leave it blank – in which case it may serve as a convenient bookmark to a specific region of interest. The commenter can also specify to make the comment public, private, or visible only to the author. If public, when the comment is made, all users currently viewing the track, regardless of whether they share a collaborative session, will be notified of its existence. If private, the comment will only ever be visible to the author, or those within the same collaborative session. Finally, if the comment is marked 'only-me', it will only ever be visible to the author. Consequently, public comments made on private tracks will only be visible to users with the permission to view the track, i.e. those within the same collaborative session.

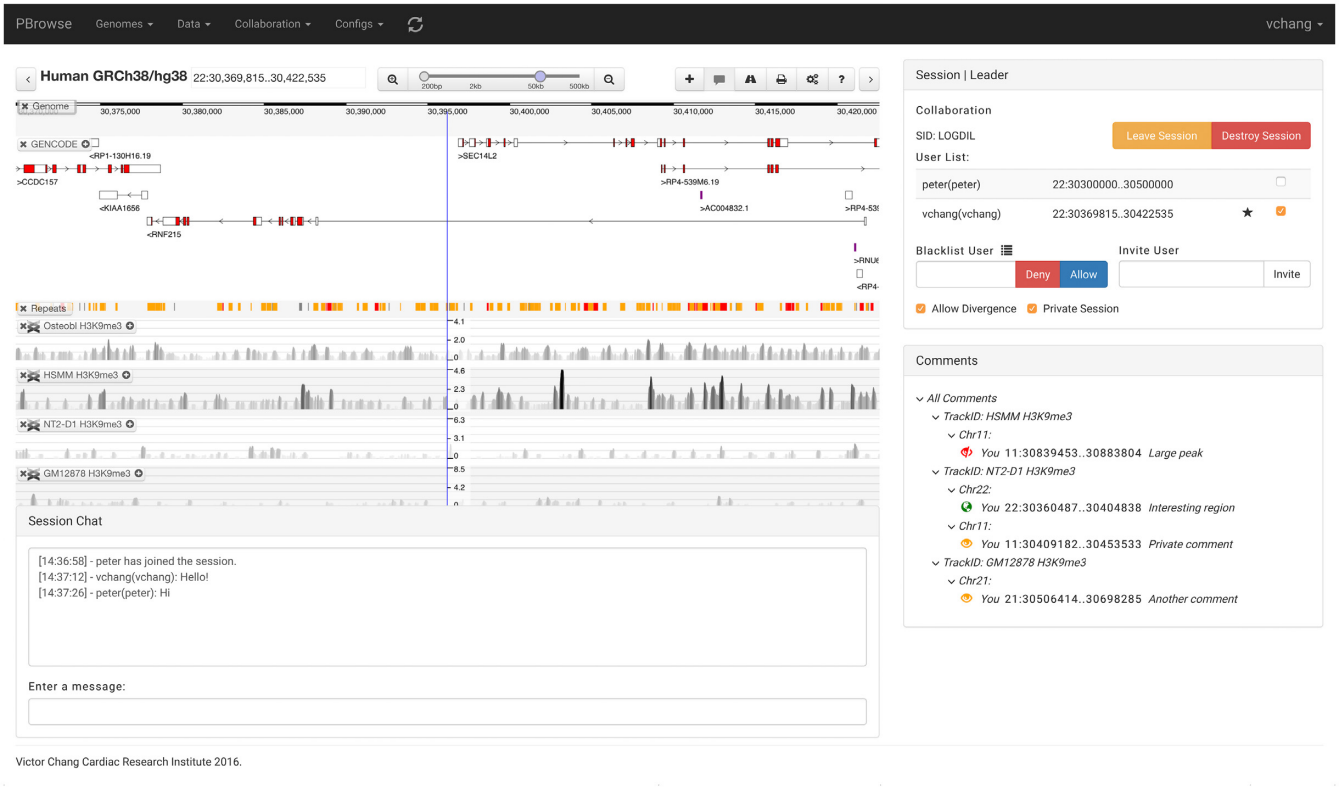

Figure 3. PBrowse collaborative session screenshot. Session management panel (right panel) shows information about the current collaborative session, such as the session ID, users in session and location status. Leader's session management panel has extra functionality such as blacklisting user, inviting user and changing privacy of session. Session chat (bottom panel) allows user in the collaborative session to communicate with each other.

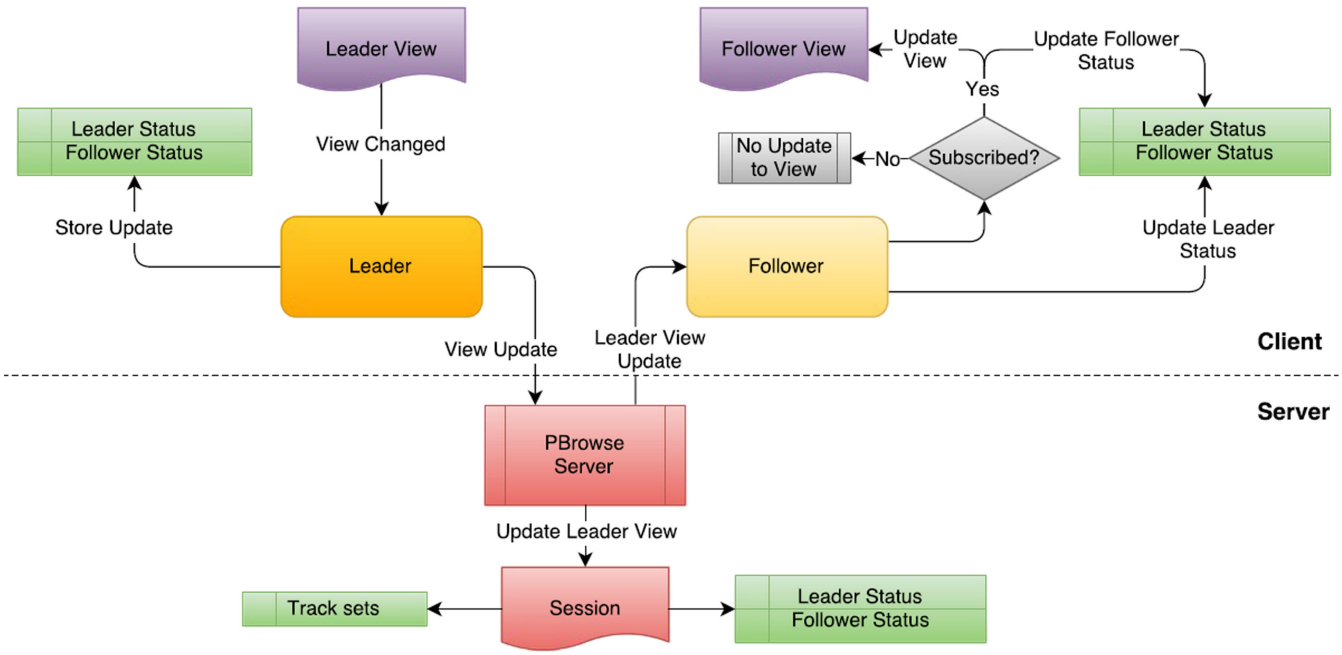

Figure 4. Collaborative synchronization flowchart. Propagation of changes made by leader in their genome browser to follower's genome browser.

Comments are displayed in a tree-like view, with a single root node having a branch for each currently loaded track. Under each track node, comments are further grouped by the chromosome under which they appear. At the lowest level, the full comment details are visible, including its author, its exact location and any textual annotation. Clicking on any node will expand or contract it, while clicking on a comment node will open a menu allowing for further operations. All users can use this to immediately navigate to the region of interest, while only the comment author can delete it. When a user loads a track for the first time, all the public comments on that track are retrieved from the database and rendered. Removing the track from view will subsequently hide its associated comment branch.

Custom tracks

The Dalliance browser alone offers some means of managing data tracks, including adding tracks from external sources, primarily via the Distributed Annotation Server (DAS) protocol. These data are served as XML to the client, which processes and renders it as a track within the browser. Dalliance can additionally render tracks from indexed binary data files served directly from HTTP, supporting the 2bit, bigWig, bam and bigBed formats (15) with the possibility for random access via range queries. The BED, WIG and VCF formats are also accessible in this way, but they must be loaded in their entirety into memory before particular regions can be retrieved. This is undesirable as the client must waste bandwidth transferring all non-essential parts of the file. A solution is to convert the data into its associated binary form, (i.e. WIG to bigwig, BED to bigBED) or to provide a tabix index along with the compressed data file. The binary conversion approach is made easily possible by a number of publicly available utilities provided by UCSC and is the recommended approach when visualising such data in PBrowse.

A number of different genomes are made available for browsing within PBrowse, the default being the GRCh37 / hg19 human genome, which has been preloaded with a large data set (2,485 tracks) from the ENCODE project (16), ready for immediate viewing. In addition, PBrowse currently also provides the hg18, and GRCh38 / hg38 human genomes, the Zv9 zebrafish genome, the GRCm37 / mm9 and GRCm38 / mm10 mouse genomes and the WS220 worm genome.

New genomes may be added by users through a simple, two-step procedure. First, a 2bit file containing sequence information for the particular genome must be uploaded. Then the user changes the active genome to 'generic' via the drop-down menu, and follows the prompts, providing a name, as well as the sequence file ID and an optional gene file ID. The new genome exists temporarily but can be preserved by saving the current track configuration. The saved configurations are stored by PBrowse and recalled for each user, who can have multiple save states. They can also be shared with other users who are in the same collaborative session.

Data upload and sharing

In addition to collaboration, PBrowse provides an efficient means for users to upload and share custom track data files. When initiating an upload, users can specify: a track-name – the primary means of identifying the data file; a description – which can contain any extra relevant information relating to the annotation; a study ID – which allows for easy grouping of related data files; the genome tag – which states the data's associated genome assembly (e.g. hg38), and finally the track's public status – which determines its visibility to other users of PBrowse. The file metadata is tabulated and presented to the user who can sort entries based on the contents of particular columns, or perform keyword searches to locate specific tracks, as depicted in Figure 5.

The visibility of an uploaded file can have one of three distinct values: 'only-me', private or public. If a file is set to 'only-me', only its owner can see it. The same applies to the private setting, except during a collaborative session where all users share their privately accessible files automatically with other members of the session. Publicly accessible files, on the other hand, are always visible to every user of PBrowse and can be visualized without needing to log in. Any changes affecting a file's visibility are propagated in real-time, with all affected users being notified as well.

As an alternative to uploading files to the PBrowse server, it is possible to register the location of a remotely-hosted file. The user can still provide meta-information for the source and a corresponding file listing will be produced and displayed within the file manager. All standard file management options can be applied to a remotely registered file, including privacy settings and search filtering, with the obvious exception that deleting the entry will not destroy the remote source.

Once uploaded or registered, clicking on a listing within the file manager will present the user with the available options for the file. Multiple files can also be selected at once to perform batch operations. However, this requires the user to have the correct privilege to perform the action for all selected files, otherwise, the operation will fail. The batch management feature is particularly useful for adding multiple related tracks to the browser at once.

Case study: collaborative exploration of human cardiac enhancers with ENCODE data

To demonstrate how PBrowse enables collaborative exploration of genome-wide data, we constructed a case study involving the sharing and exploration of EP300 ChIP-seq data set from foetal and adult human heart tissues (16). We have created a short video of this demonstration (Supplementary Material S1).

In the case study, we have a leader (Xin) who has loaded a few EP300 tracks from human and foetal heart in his genome browser. The user then starts a new collaborative session and invites his collaborator (Andrian) to join the session. Upon joining the collaborative session, the collaborator's genome browser view is synchronized to the view of the leader's genome browser, allowing the collaborator to view the same EP300 tracks at the same view. Changes made by the leader on their genome browser, such as scrolling to a different region and an addition

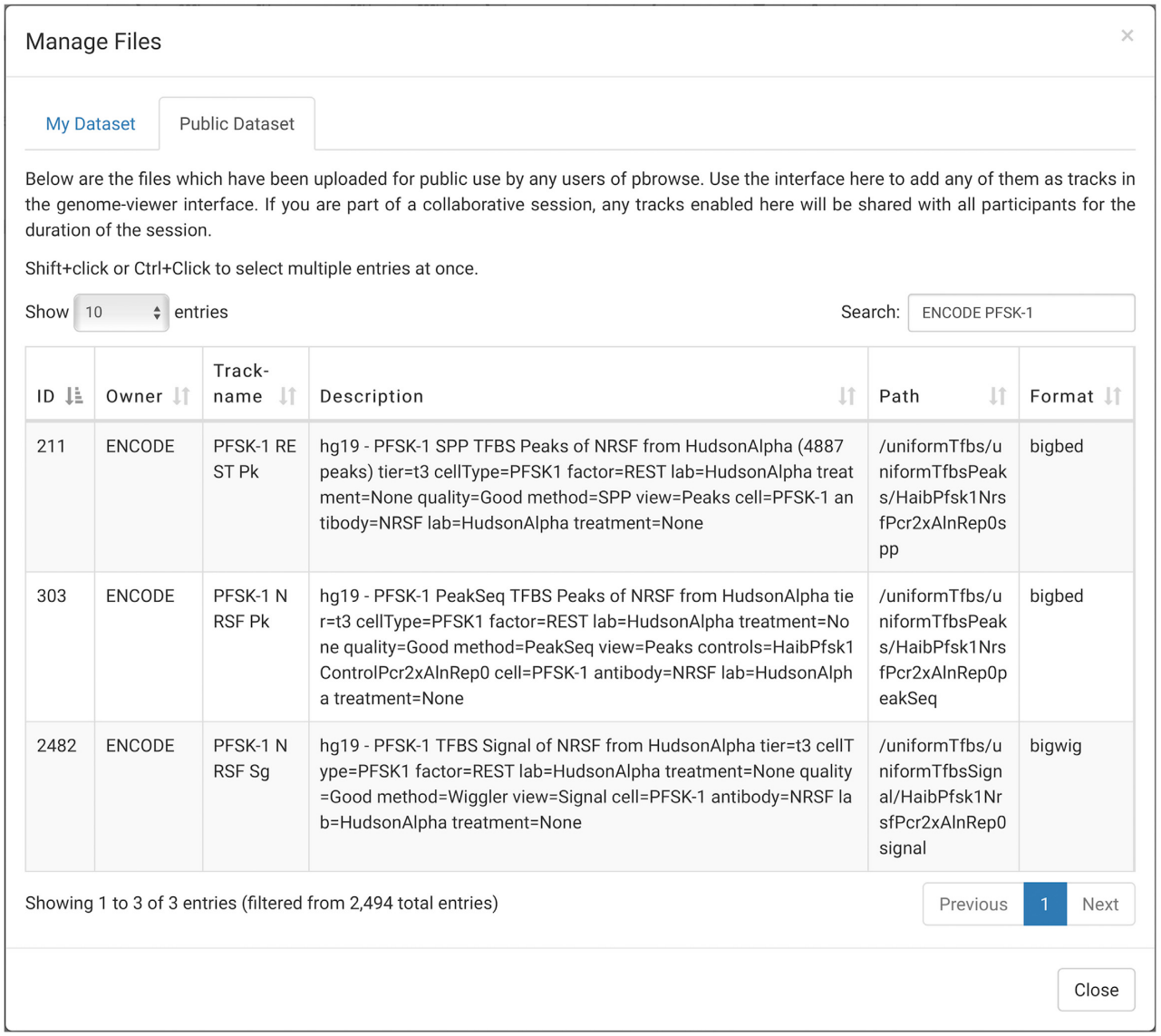

Figure 5. File management interface. User is able to access their uploaded files (My Data set) and files made public by other user, such as the ENCODE data set, (Public Data set) and add them as tracks to PBrowse.

of the H3K27ac tracks from a local computer and the ENCODE DAS server, are reflected on the collaborator's genome browser instantly, demonstrating the real-time nature of the collaborative session. The case study also demonstrates the other collaborative features of PBrowse, such as instant messaging—allowing the users to communicate in real-time—and comments – which allows users to share regions of interest between themselves.

CONCLUSION

The technology of web-based genome browsers has evolved significantly over the last decade. Nonetheless, the focus of these browsers has been largely around personal exploration of data. To fully harness the collective wisdom of a collaborative group, PBrowse places a strong emphasis on real-time human–human interactions through the medium of the genome browser. As illustrated in our case study, PBrowse enables a level of real-time knowledge exchange between multiple users that is not currently achievable by other browsers. Thus, we argue that PBrowse represents a paradigm shift in the way we see and use genome browsers.

AVAILABILITY

A public demonstration version is available at http://pbrowse.victorchang.edu.au, while the source code is available at GitHub, http://github.com/VCCRI/PBrowse, under BSD 3 license.

SUPPLEMENTARY DATA

Supplementary Data are available at NAR Online.

ACKNOWLEDGEMENTS

The authors thank the IT department of the Victor Chang Cardiac Research Institute for providing technical support. The authors also thank the Information and Communication Technology Office (ICTO) at the University of Macau for providing access to and support on a High-Performance Computer, and Jacky Chan and William Pang for their expert technical support. The authors thank Djordje Djordjevic, David Humphreys and Tomasz Szczesnik for their critical evaluation of the manuscript. The authors also thank all the volunteer testers who participated in the software usability study.

FUNDING

New South Wales Ministry of Health; National Health and Medical Research Council/National Heart Foundation Career Development Fellowship [1105271]; Ramaciotti Establishment [ES2014/010]; Amazon Web Services (AWS) Credits for Research; Australian Postgraduate Award; Science and Technology Development Fund of Macau S.A.R [(FDCT) (085/2014/A2)]. Funding for open access charge: Victor Chang Cardiac Research Institute.
Conflict of interest statement. None declared.

REFERENCES

1. Kent,W.J., Sugnet,C.W., Furey,T.S., Roskin,K.M., Pringle,T.H., Zahler,A.M. and Haussler,D. (2002) The human genome browser at UCSC. *Genome Res.*, **12**, 996–1006.
2. Stalker,J. (2004) The Ensembl Website: mechanics of a genome browser. *Genome Res.*, **14**, 951–955.
3. Kawahara,Y., Sakate,R., Matsuya,A., Murakami,K., Sato,Y., Zhang,H., Gojobori,T., Itoh,T. and Imanishi,T. (2009) G-compass: a web-based comparative genome browser between human and other vertebrate genomes. *Bioinformatics*, **25**, 3321–3322.
4. Skinner,M.E., Uzilov,A.V., Stein,L.D., Mungall,C.J. and Holmes,I.H. (2009) JBrowse: a next-generation genome browser. *Genome Res.*, **19**, 1630–1638.
5. Kong,L., Wang,J., Zhao,S., Gu,X., Luo,J. and Gao,G. (2012) ABrowse–a customizable next-generation genome browser framework. *BMC Bioinformatics*, **13**, 2.
6. Medina,I., Salavert,F., Sanchez,R., de Maria,A., Alonso,R., Escobar,P., Bleda,M. and Dopazo,J. (2013) Genome Maps, a new generation genome browser. *Nucleic Acids Res.*, **41**, W41–W46.
7. Stein,L.D. (2013) Using GBrowse 2.0 to visualize and share next-generation sequence data. *Brief. Bioinform.*, **14**, 162–171.
8. Pak,T.R. and Roth,F.P. (2013) ChromoZoom: a flexible, fluid, web-based genome browser. *Bioinformatics*, **29**, 384–386.
9. Down,T.A., Piipari,M. and Hubbard,T.J.P. (2011) Dalliance: interactive genome viewing on the web. *Bioinformatics*, **27**, 889–890.
10. Robinson,J.T., Thorvaldsdóttir,H., Winckler,W., Guttman,M., Lander,E.S., Getz,G. and Mesirov,J.P. (2011) Integrative genomics viewer. *Nat. Biotechnol.*, **29**, 24–26.
11. Zhou,X., Maricque,B., Xie,M., Li,D., Sundaram,V., Martin,E. a, Koebbe,B.C., Nielsen,C., Hirst,M., Farnham,P. *et al.* (2011) The human epigenome browser at Washington University. *Nat. Methods*, **8**, 989–990.
12. Sun,C. and Chen,D. (2002) Consistency maintenance in real-time collaborative graphics editing systems. *ACM Trans. Comp. Hum. Interact.*, **9**, 1–41.
13. Imine,A., Cherif,A. and Rusinowitch,M. (2009) A flexible access control model for distributed collaborative editors. In: Jonker,W and Petković,M (eds). *Secure Data Management*. Lecture Notes in Computer Science, Springer, Berlin Heidelberg, pp. 89–106.
14. Lautamäki,J., Nieminen,A., Koskinen,J., Aho,T., Mikkonen,T. and Englund,M. (2012) CoRED: browser-based collaborative real-time editor for Java web applications. *Proc. ACM 2012 Conf. Comp. Supported Cooperative Work*, 1307–1316.
15. Kent,W.J., Zweig,A.S., Barber,G., Hinrichs,A.S. and Karolchik,D. (2010) BigWig and BigBed: enabling browsing of large distributed datasets. *Bioinformatics*, **26**, 2204–2207.
16. The ENCODE Project Consortium (2012) An integrated encyclopedia of DNA elements in the human genome. *Nature*, **489**, 57–74.