A model of navigation history

Connor G. Brewster
Alan Jeffrey

DRAFT of 2016-08-16

Abstract: Navigation has been a core component of the web since its inception: users and scripts can follow hyperlinks, and can go back or forwards through the navigation history. In this paper, we present a formal model aligned with the WHATWG specification of navigation history, and investigate its properties. The fundamental property of navigation history is that traversing the history by δ then by δ' should be the same as traversing by $\delta + \delta'$. In particular, traversing by $+1$ (forward) then by -1 (back) is the same as traversing by 0 (doing nothing). We show that the specification-aligned model does not satisfy this property, by exhibiting a series of counter-examples, which motivate four patches to the model. We present a series of experiments, showing that browsers are inconsistent in their implementation of navigation history, but that their behaviour is closer to the patched model than to the specification-aligned model. We propose patches to the specification to align it with the patched model.

ACM Classification: D.2.1 Requirements/Specifications.

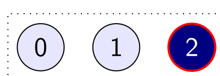
Keywords: Formal model, Navigation, Session history, Specification, Web browsers.

1. Introduction

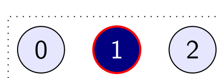
Navigation has been a core component of the web since its inception: users and scripts can follow hyperlinks, and can go back or forwards through the navigation history. Users are exposed to this functionality through following hyperlinks, and by the forward and back buttons. Scripts have many ways of accessing session history, via the navigation API [5, §7.7] and the `element.click()` method.

Prior formalizations of navigation history include [1, 2, 3, 4, 6], which predate and are not aligned with the WHATWG specification [5]. The specification of the navigation API is informal, and has complex dependencies on the rest of the HTML specification. There is little discussion of the goals of the API, and an unclear alignment with browser implementations.

In this paper, we present a formal model of navigation, aligned with the HTML specification, and investigate its properties. The starting point is that there is a total order of *documents*[1], one of which is *active*, for example:

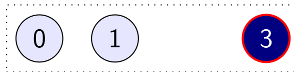


In diagrams, we use left-to-right order to indicate order, and highlight the active document. The user can *traverse* the history which changes the active document, for example pressing the back button:

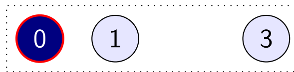


[1] We are eliding some of the technical details of the specification here, in particular we are conflating a *browsing context* with the document it contains, and we are ignoring issues around document loading and unloading, and around the current entry of the joint session history.

The user can also *navigate*, which replaces any document after the currently active document by a fresh active document:

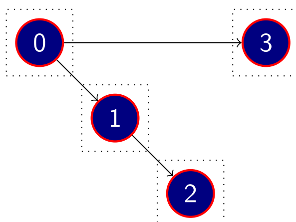


Users can also traverse the history by more than one document at a time, for example by using a pull-down menu from the back or forwards button. This is called *traversing by* δ, for instance we can traverse our running example by -2 to get:

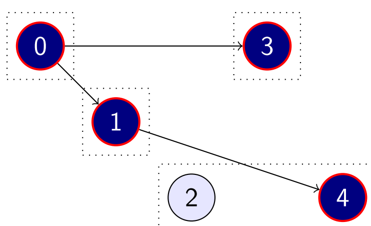


We formalize the notions of traversal and navigation in §2, and show the *fundamental property of traversal*: that traversing by δ then by δ' is the same as traversing by $\delta + \delta'$.

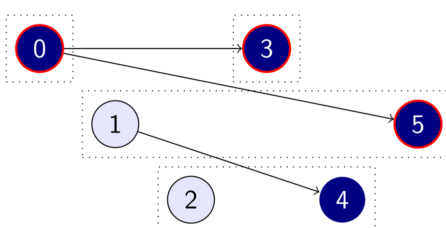
So far, the model is refreshingly simple, and corresponds well to the specification and to browser implementations. Where the problems arise is in the *hierarchical* nature of documents. HTML documents can contain `iframe` elements, which are independent documents in their own right, often used to embed third party content such as advertisements. We can treat each document as a tree, for example:



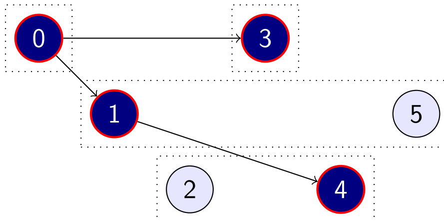
The problem comes from the ability of each document to navigate separately and maintain its own session history, but that traversal is a global operation that operates on the *joint session history*. For example if document 2 in the previous example navigates, the resulting state is:



and then if document 1 navigates, the state is:



Note that node 4 here is in an unusual state: it is active, but has an inactive ancestor. The specification [5, §7.7] distinguishes between *active* documents such as 4, and *fully active* documents such as 0, 3 and 5. Active documents can become fully active by traversals involving their ancestors. For example, after traversing by -1, document 4 is fully active:



As even a simple example like this shows, the combination of features quickly results in a complex mix of session history, ordering, and document hierarchy, which leads to the problems:

- *Formally* there is no simple model, and the model provided by the specification does not satisfy the traverse-then-traverse property.
- *Experimentally* the browsers disagree with each other, and with the HTML specification, about the semantics of navigation.

In this paper, we address these:

- §2 provides a formal model of navigation history, which is intended to align with the specification. We show, through a series of examples, that it does not satisfy the fundamental property, and give patches to the model for each example. The final model does satisfy the fundamental property.
- §3 investigates how well the patched model aligns with existing browser implementations. We show ways in which the browsers exhibit behaviours which are not aligned with the specification, and discuss how our proposed model matches these behaviours.

Finally, we propose changed wording to the specification, which would bring it in line with our patched model.

2. Model

In this section, we present our formal model of navigation history. §2.1 contains definitions of concepts such as tree and total order, and may be skipped by most readers. The model, together with some examples, is given in §2.2. In §2.3 we define the fundamental property of traversal, show that the model does *not* satisfy it, but can be patched to do so.

2.1. Preliminaries

A *directed graph* $G = (V, \to)$ consists of:

- a set V (the *vertices*), and
- a relation $\to \subseteq (V \times V)$ (the *edges*).

The *transitive closure* of $\to$ is defined as $v \to^+ v'$ whenever there exists $v_0, \ldots, v_n$ such that:

$$v = v_0 \to \cdots \to v_n = v'$$

The *reflexive transitive closure* of $\to$ is defined as $v \to^* v'$ whenever $v \to^+ v'$ or $v = v'$. A *forest* is a directed graph where:

- there is no v such that $v \to^+ v$ (*acyclicity*)

- if $v \to v' \leftarrow v''$ then $v = v''$ (*at most one parent*).

A *root vertex* of a forest is a vertex v such that there is no $w \to v$. A *tree* is a forest with a unique root vertex. A *preorder* is a directed graph $(V, \leq)$ such that:

- every v has $v \leq v$ (*reflexivity*), and
- if $v \leq v' \leq v''$ then $v \leq v''$ (*transitivity*).

A *partial order* is a preorder such that:

- for every v and v', if $v \leq v'$ and $v' \leq v$ then $v = v'$ (*antisymmetry*).

A *total order* is a partial order such that:

- for every v and v', either $v \leq v'$ or $v \geq v'$ (*totality*).

A *equivalence* is a preorder $(V, \sim)$ such that:

- if $v \sim v'$ then $v' \sim v$ (*symmetry*).

2.2. Definitions

We can now formalize our model of navigation history, together with the operations of navigation and traversal. This formalizes the navigation history specification [5], and has a pleasant diagrammatic presentation, but as we shall see in §2.3, it has unexpected properties.

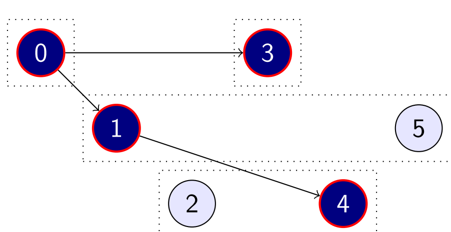
Definition 1 (Navigation history): A *navigation history* $H = (D, A, \to, \leq, \sim)$ consists of:

- a finite set D (the *documents*),
- a subset $A \subseteq D$ (the *active* documents),
- a forest $(D, \to)$ (the *child* relationship),
- a total order $(D, \leq)$ (the *chronological* order), and
- an equivalence relation $(D, \sim)$ (the *same-session* equivalence).

such that:

- for every d there is a unique $d' \in A$ such that $d \sim d'$,
- for every $d \to e \sim e'$ we have $d \to e'$, and
- for every $d \to e$, we have $d \leq e$. □

We present such navigation histories diagrammatically, using left-to-right position for chronological order, and grouping documents in the same session. Since documents in the same session must have the same parent, we only draw the document hierarchy for active children. For example the diagram:



represents:

$$
\begin{array}{l}
D = \{0, 1, 2, 3, 4, 5\} \\
A = \{0, 1, 3, 4\} \\
0 \to 1 \quad 0 \to 3 \quad 0 \to 5 \quad 1 \to 2 \quad 1 \to 4 \\
0 \le 1 \le 2 \le 3 \le 4 \le 5 \\
1 \sim 5 \quad 2 \sim 4
\end{array}
$$

In such a navigation history, we define:

- d_0 is the unique active root document,
- $d \twoheadrightarrow e$ when $d \to e$ and $e \in A$ (the *active child* relationship),
- $FA = \{d \mid d_0 \twoheadrightarrow^* d\}$ (the *fully active* documents),
- $d \lesssim e$ whenever $d \sim e$ and $d < e$,
- the *session future* of d is $\{e \mid d \lesssim e\}$,
- the *session past* of d is $\{e \mid d \gtrsim e\}$,
- the *joint session future* is $\{e \mid \exists d \in FA \,.\, d \lesssim e\}$,
- the *joint session past* is $\{e \mid \exists d \in FA \,.\, d \gtrsim e\}$,

These definitions are intended to align with the specification, for example [5, 7.7.2] has the definition:

> The **joint session history** of a top-level browsing context is the union of all the session histories of all browsing contexts of all the fully active `Document` objects that share that top-level browsing context, with all the entries that are current entries in their respective session histories removed except for the current entry of the joint session history.

which (eliding the concept of "current entry of the joint session history") corresponds to the above definitions of joint session future and past. We now consider how to formalize operations on navigation histories. staring with *navigation*, which is triggered by following hyperlinks, or other actions which trigger document loading.

Definition 2 (Navigation): Define *deleting d from H*, when d is in the joint session future to be $H' = (D', A, \le, \to, \sim)$ where:

- $D' = D \setminus \{e \mid d \to^* e\}$.

Define *replacing d by d' in H*, where $d \in FA$ and $d' \notin D$, to be $H' = (D', A', \le', \to', \sim')$ where:

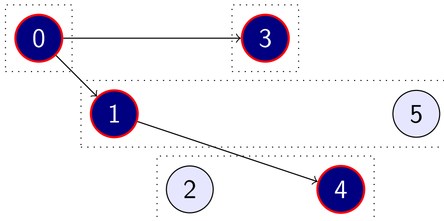
- $D' = D \cup \{d'\}$,
- $e \in A'$ whenever $e \in A$ and $e \ne d$, or $e = d'$,
- $e \le' f$ whenever $e \le f$, or $f = d'$,
- $e \to' f$ whenever $e \to f$, or $e \to d$ and $f = d'$, and
- $e \sim' f$ whenever $e \sim f$, or $e = f$, or $e \sim d$ and $f = d'$, or $d \sim f$ and $e = d'$.

Define *navigating from d to d' in H*, where $d \in FA$ to be the result of:

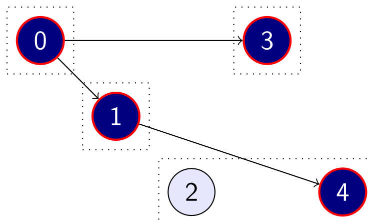
- deleting the session future of d, and then

- replacing d by d'. $\square$

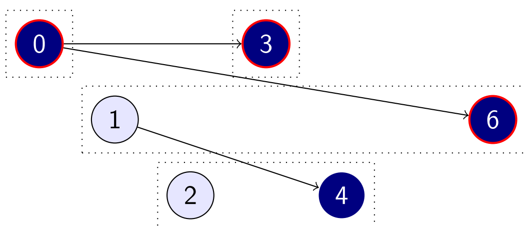
There are two parts to navigation from d to d': deleting the session future of d, followed by replacing d by d'. For example, navigating from 1 to 6 in:



we first delete the session future of 1 (which is 5):



then replace 1 by 6:



We also define *traversing the history*, which changes the active documents.

Definition 3 (Traversal): Define *traversing the history to d in H* to be $H' = (D, A', \leq, \rightarrow, \sim)$ where:

- $e \in A'$ whenever $d \not\sim e \in A$, or $d = e$.

Define H *traverses the history by* $+\delta$ *to* H' when:

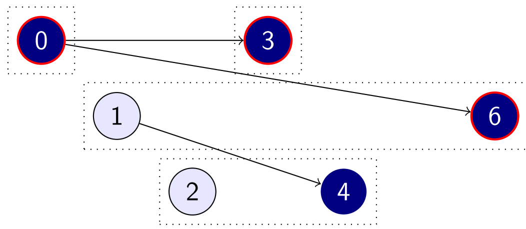
- the joint session future of H is $d_1 < \cdots < d_\delta < \cdots$,
- H traverses the history to d_δ in H'

Define H *traverses the history by* $-\delta$ *to* H' when:

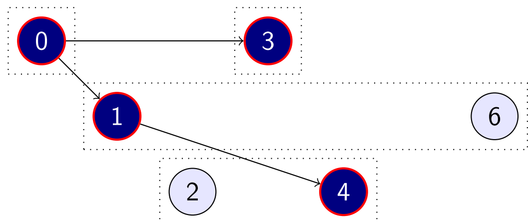
- the joint session past of H is $d_1 > \cdots > d_\delta > \cdots$,
- H traverses the history to d_δ in H'

Define H *traverses the history by* 0 *to* H' when $H = H'$.

For example, to traverse the history by -2 in:



we find the joint session past (which is $2 > 1$) and traverse the history to the second item (which is 1) to arrive at:



These definitions are intended to formally capture the HTML specification, for example [5, §7.7.2] includes:

> To **traverse the history by a delta** δ, the user agent must append a task to this top-level browsing context's session history traversal queue, the task consisting of running the following steps:
>
> 1. If the index of the current entry of the joint session history plus δ is less than zero or greater than or equal to the number of items in the joint session history, then abort these steps.
> 2. Let *specified entry* be the entry in the joint session history whose index is the sum of δ and the index of the current entry of the joint session history.
> 3. Let *specified browsing context* be the browsing context of the specified entry.
> 4. If the specified browsing context's active document's unload a document algorithm is currently running, abort these steps.
> 5. Queue a task that consists of running the following substeps [...]
>
>   3. Traverse the history of the specified browsing context to the specified entry.

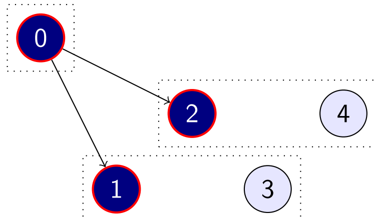
2.3. Properties

We now consider the fundamental property of navigation history:

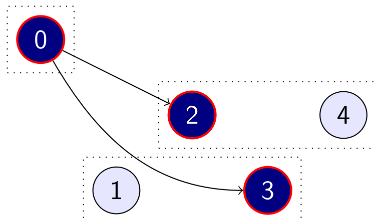
Definition 4 (Fundamental property): H satisfies the *fundamental property of traversal* whenever H traverses the history by δ to H' and H' traverses the history by δ' to H'' implies H traverses the history by $\delta + \delta'$ to H''. □

Unfortunately, navigation histories as specified do not always satisfy the fundamental property, due to ways individual session histories are combined into the joint session history. In this section, we give a series of counterexamples, and propose patches to the model to address each counterexample.

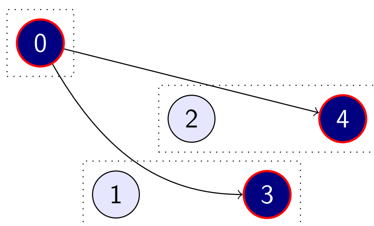
Counterexample 1: Let H be:



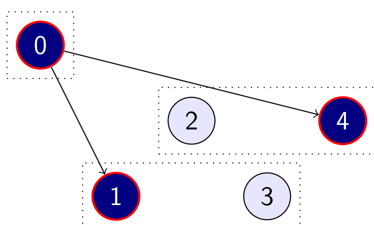
which traverses the history by 1 to:



which traverses the history by 1 to:



but H traverses the history by 2 to:



□

This counterexample is caused by the definition of 'traverses the history by δ' which only traverses one document's session history. Instead, we should traverse the history of all δ documents.

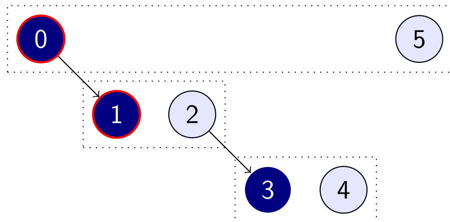
Patch 1 (Traverse intermediaries): Define H *traverses the history by* $+\delta$ *to* H' when:

- the joint session future of H is $d_1 < \cdots < d_\delta < \cdots$,
- there is some $H = H_0, \ldots, H_\delta = H'$, such that
- H_{i-1} traverses the history to d_i in H_i for each $1 \leq i \leq \delta$.

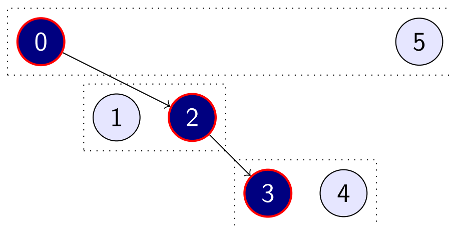
Define H *traverses the history by* $-\delta$ *to* H' when:

- the joint session past of H is $d_1 > \cdots > d_\delta > \cdots$,
- there is some $H = H_0, \ldots, H_\delta = H'$, such that
- H_{i-1} traverses the history to d_i in H_i for each $1 \leq i \leq \delta$. □

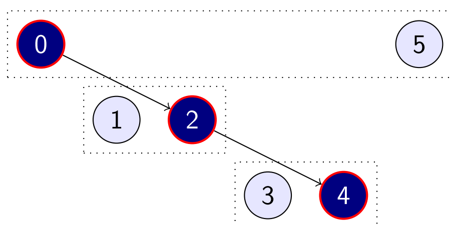
Counterexample 2: Let H be:



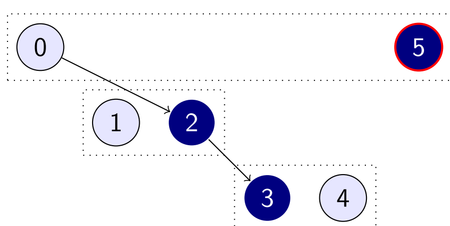
which traverses the history by 1 to:



which in turn traverses the history by 1 to:



but H traverses the history by 2 to:



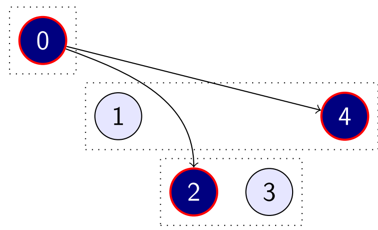
□

The problem this time is that the definition of 'joint session history' only includes the fully active documents, not all active documents.

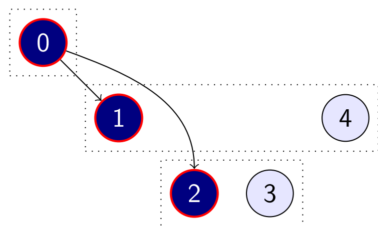
Patch 2 (Active joint session history): Define:

- the *joint session future* is $\{e \mid \exists d \in A \,.\, d \lesssim e\}$, and
- the *joint session past* is $\{e \mid \exists d \in A \,.\, d \gtrsim e\}$. □

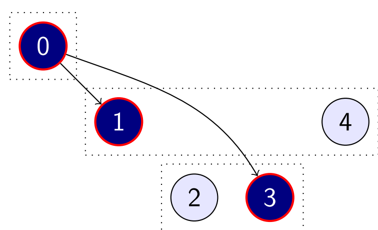
Counterexample 3: Let H be:



which traverses the history by -1 to:



which traverses the history by 1 to:



which is not the same as H. $\square$

This counterexample is caused by an asymmetry in the definition of traversal: it is defined in terms of navigating *to* a document d, and not navigating *from* a document. We fix this by making the definition symmetric:

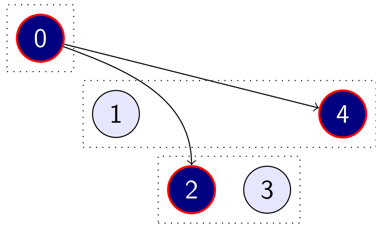
Patch 3 (Symmetric traversal): Define H *traverses the history from* d' when there is some d such that:

- $d \lesssim d'$,
- for any $e \lesssim d'$ we have $e \leq d$, and
- H traverses the history to d.

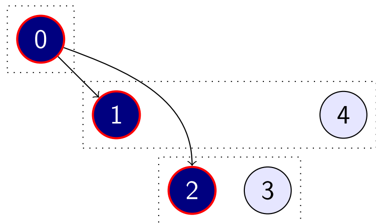
Define H *traverses the history by* $-\delta$ *to* H' when:

- the joint session past and active documents of H are $d_1 > \cdots > d_\delta > \cdots$,
- there is some $H = H_0, \ldots, H_\delta = H'$, such that
- H_{i-1} traverses the history from d_i in H_i for each $1 \leq i \leq \delta$. $\square$

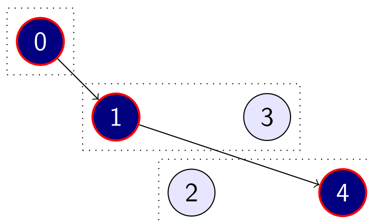
For example, to traverse the history by -1 from:



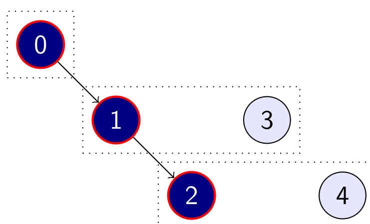
we find the joint session past and active documents (which is $4 > 2 > 1 > 0$) and traverse the history from the first item (which is 4) which is the same as traversing the history to 1:



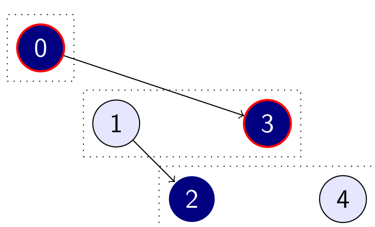
Counterexample 4: Let H be:



which traverses the history by -1 to:

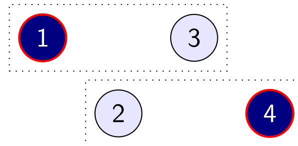


which traverses the history by 1 to:

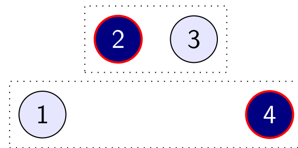


which is not the same as H. $\square$

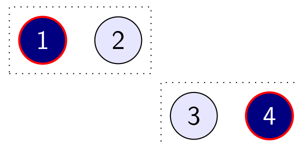
The problem here is not the definition of 'traversing by δ', but the definition of navigation histories themselves. They allow for states such as H from Counterexample 4, which includes the problematic documents:



There are similar problems with documents:



and with documents:

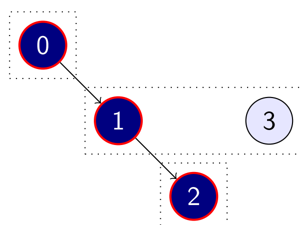


It turns out that these are the only remaining cause of counterexamples, and we will call examples like this not *well-formed*.

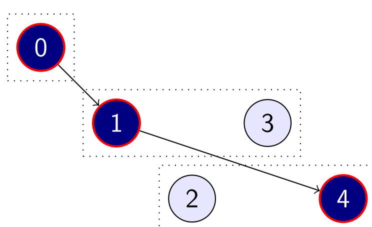
Definition 5 (Well-formed): A navigation history is *well formed* whenever for any $a \lesssim b$ and $c \lesssim d$, if $a \in A$ and $d \in A$ then $d \leq b$.

We have that traversal preserves being well-formed: if H is well-formed, and H traverses by δ to H', then H' is well-formed. Unfortunately, this is not true for navigation, because of the way it clears the session future.

Counterexample 5: Let H be the well-formed history:



which navigates from 2 to:



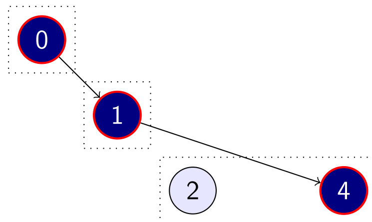
which is not well-formed. □

Fortunately, we can patch navigation to address this, by requiring that we clear the entire joint session future, not just the session future of the document being navigated from.

Patch 4 (Navigation deletes joint session future): Define *navigating from d to d' in H*, where $d \in FA$ to be the result of:

- deleting the joint session future, and then
- replacing d by d'. □

For example in Counterexample 5, navigation from 2 now results in the well-formed history:



With these patches, we can prove the fundamental property of traversal.

Theorem 1: For any well-formed navigation history H, if H traverses the history by δ to H' and H' traverses the history by δ' to H'' then H traverses the history by $\delta + \delta'$ to H''.

Proof: In this paper, we give a proof sketch. The full details have been mechanically verified in Agda [7]. Define:

- a document d *can go back* there is some $c \lesssim d$,
- the *back target* b is the $\leq$-largest active document which can go back, and
- the *forward target* f is the $\leq$-smallest document in the joint session future.

We then show some lemmas:

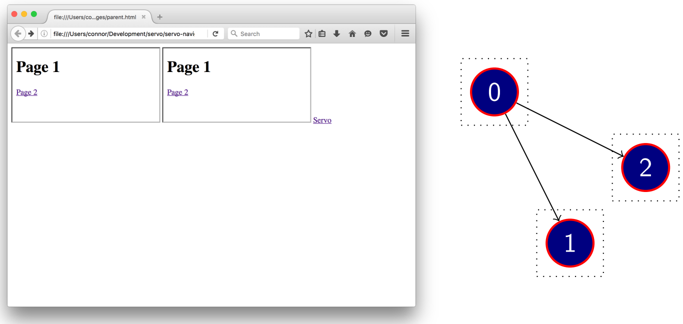
1. H traverses by $+(\delta+1)$ to H' if and only if H traverses to f, then traverses by $+\delta$ to H'.
2. H traverses by $-(\delta+1)$ to H' if and only if H traverses from b, then traverses by $-\delta$ to H'.
3. If H is well-formed and H traverses to f with result H', then f is the back target of H', and H' traverses from f with result H.
4. If H is well-formed and H traverses from b with result H', then b is the forward target of H', and H' traverses to b with result H.
5. If H is well-formed and H traverses to f to H', then H' is well-formed.
6. If H is well-formed and H traverses from b to H', then H' is well-formed.

The result is then an induction on δ. □

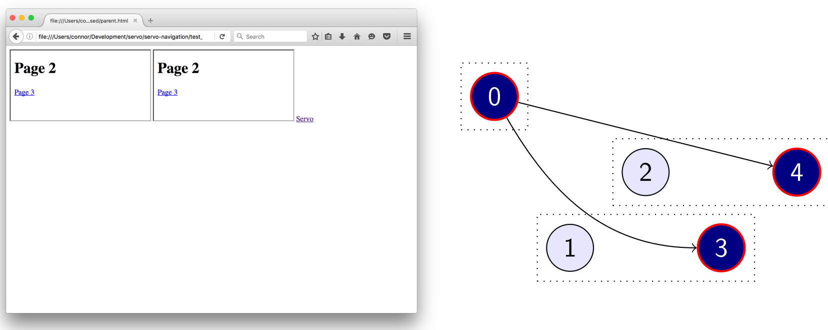
3. Experiments

In this section, we summarize our experiments to validate the conformance of browser implementations with respect to the WHATWG specification, to our proposed changes, and to each other.

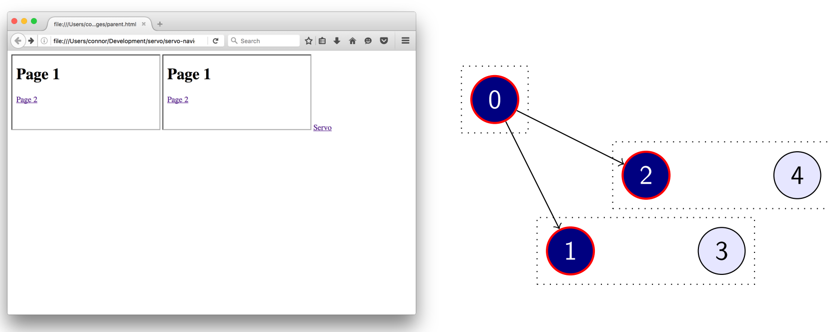
We give details of how to recreate Counterexample in detail, the other counterexamples are similar. We create an HTML page for the parent, containing two `iframe`s, both of which start at `page1.html`, with a hyperlink to `page2.html`:



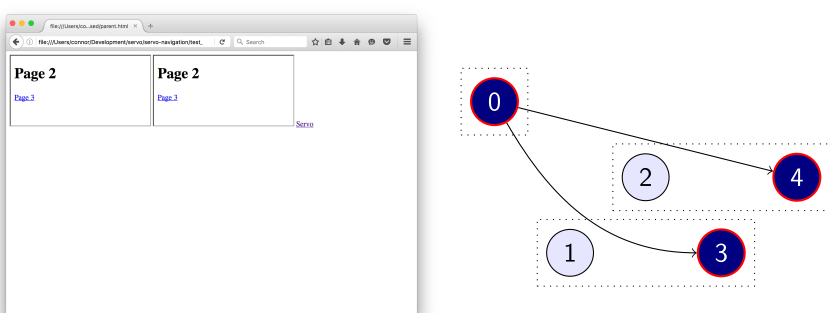
Clicking on both hyperlinks loads both copies of `page2.html`:



Pressing the "back" button twice takes us to the initial state of Counterexample 3:



Now, the user can traverse the history by +2 (by holding down the "forward" button) which results in state:



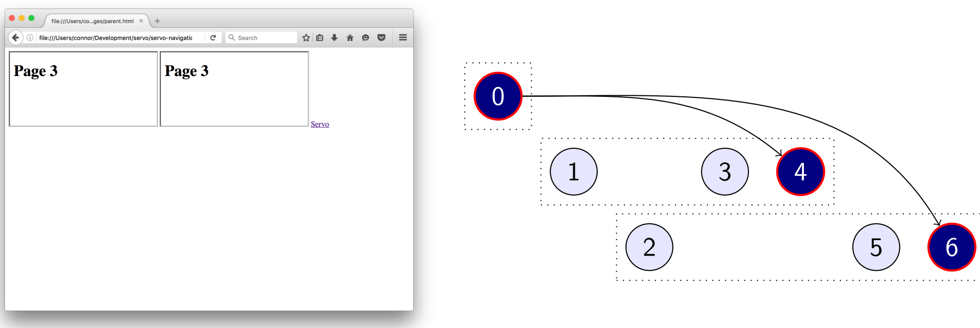
Experimentally, this shows that Firefox is aligned with our patched model, rather than with the unpatched model. We can set up similar experiments for the other counterexamples, and execute them in other browsers, which gives results[2]: [TODO: Complete this table for Edge!] [TODO: Finish experiments for Edge]

	Counterexample			
	1	2	3	5
Firefox	P	P	P	P
Chrome	P	P	P	P
Safari	P	P	P	P
Edge	?	?	?	?
Servo	P	P	P	P

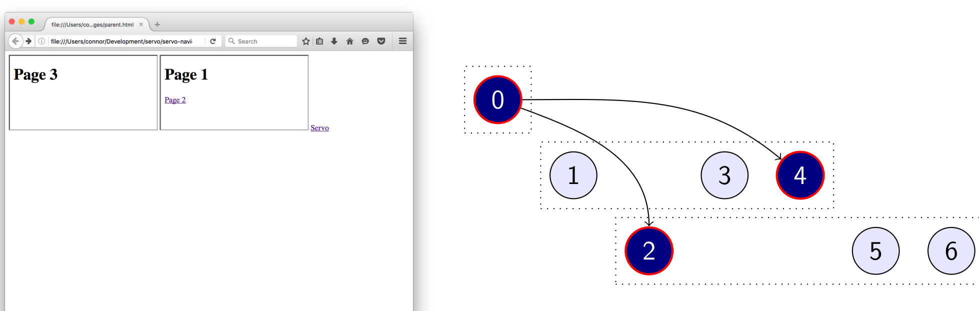
P: aligned with patched model
U: aligned with unpatched model

Most browsers are compatible with the patched model rather than than unpatched model, with the exception of Edge (Internet Explorer has similar behaviour). Servo was designed from the patched model.

Moreover, performing these experiments shows some unexpected behaviours in browser implementations. For example in Firefox, starting in state:



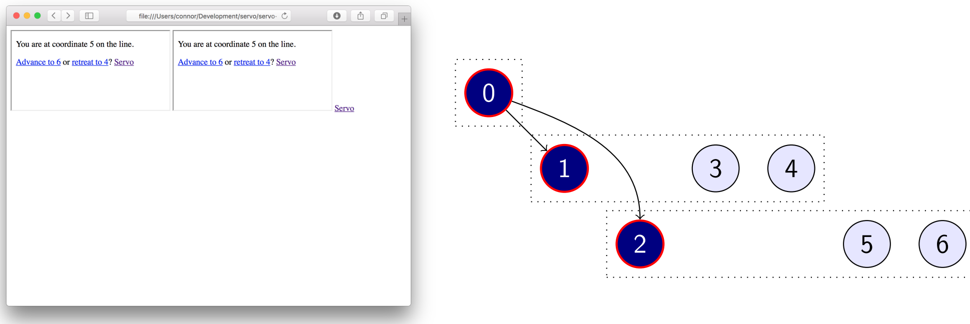
and traversing by -4 results in state:



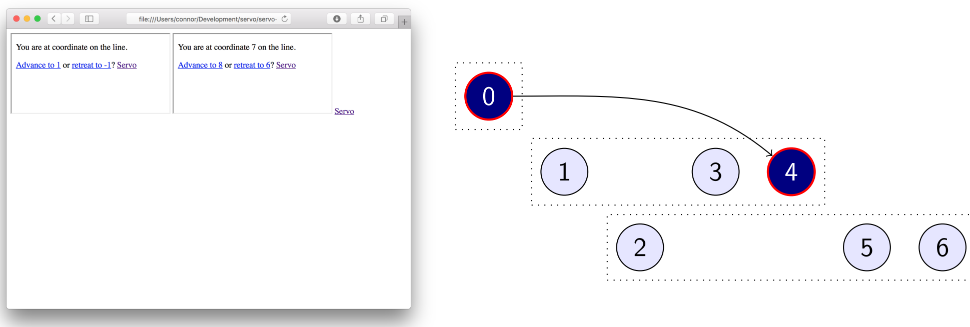
This state is unexpected, as document 4 should have traversed to document 1, and any state showing `page3.html` should be capable of going back.

In Safari, the use of `pushState` and `popState` for navigation has unexpected results. We can use `pushState` and `popState` to construct state:

[2] Recall that Counterexample 4 depends on a non-well-formed navigation history, and that the patch for it is to make such states unreachable, and so experimentally unverifiable.



then traversing by +4 results in:



After this traversal, we are unable to determine the active entry for one of the `iframe`s as its state is `null`.

As these examples show, navigation history is difficult to implement: even major browser implementations give unexpected behaviours when combining separate `iframe` session histories.

4. Specification

In this section, we discuss how the WHATWG specification [5, §7.7.2] can be aligned with the model from §2. This is not a direct translation, due to some of the features we elided in our model. In particular, we did not discuss how documents are *loaded* and *unloaded*, which includes downloading and allocating resources such as HTML or CSS, and activating JavaScript content. Since loading-then-unloading a document is wasteful, the specification should be written to avoid loading intermediate pages when traversing by a delta. This introduces complexity.

Our first proposed change is that the current specification is defined in terms of the "joint session history" and makes use of the "current entry of the joint session history", neither of which are used by our model. We propose to remove the definition of "joint session history" and "current entry of the joint session history", and add the following:

> The **session past** of a browsing context is the entries of the session history added before the current entry (and does not include the current entry).
>
> The **session future** of a browsing context is the entries of the session history added after the current entry (and does not include the current entry).
>
> If an entry has a next entry in the chronologically ordered session history, it is its **successor**.

> If an entry has a previous entry in the chronologically ordered session history, it is its **predecessor**.
>
> The **joint session past** of a top-level browsing context is the union of all the session pasts of all browsing contexts that share that top-level browsing context.
>
> Entries in the joint session past are in decreasing chronological order of the time they were added to their respective session histories.
>
> The **joint session future** of a top-level browsing context is the union of all the session futures of all browsing contexts that share that top-level browsing context.
>
> Entries in the joint session future are in increasing chronological order of the time their predecessor were added to their respective session histories.

The second proposed change is to replace the definition of how a user agent should "traverse the history by a delta" by the following:

> To **traverse the history by a delta** *delta*, the user agent must append a task to this top-level browsing context's session history traversal queue, the task consisting of running the following steps:
>
> 1. Define the *entry sequence* as follows:
>   1. If *delta* is a positive integer $+n$, and if the length of the joint session future is less than or equal to n then let the *entry sequence* be the first n entries of the joint session future.
>   2. If *delta* is a negative integer $-n$, and if the length of the joint session past is less than or equal to n then let the *entry sequence* be the first n entries of the joint session past.
>   3. Otherwise, abort traversing the history by a delta.
> 2. A session entry is said to **become active** when it is a member of the *entry sequence*, and no session entry after it in the *entry sequence* has the same browsing context.
> 3. A session entry is said to **stay active** when it it the current entry of its browsing context, and there are no members of the *entry sequence* with the same browsing context.
> 4. A session entry is said to be **activating** when either it will become active or stay active.
>
>   **Note:** the activating documents will be active after traversal has finished.
> 5. A session entry is said to be **fully activating** if is activating, and either its browsing context is a top-level browsing context, or it has a parent browsing context and the session entry through which it is nested is itself fully activating.
>
>   **Note:** the fully activating documents will be fully active after traversal has finished.
> 6. Queue a task that consists of running the following substeps. The relevant event loop is that of the specified browsing context's active document. The task source for the queued task is the history traversal task source.
>   1. For each *specified entry* in the *entry sequence*, run the following substeps.
>   1. Let *specified browsing context* be the browsing context of the *specified entry*.

2. If there is an ongoing attempt to navigate *specified browsing context* that has not yet matured (i.e. it has not passed the point of making its `Document` the active document), then cancel that attempt to navigate the browsing context.
3. If the *specified browsing context*'s active document is not the same `Document` as the `Document` of the specified entry, then run these substeps:
   1. Prompt to unload the active document of the *specified browsing context*. If the user refused to allow the document to be unloaded, then abort these steps.
   2. Unload the active document of the *specified browsing context* with the recycle parameter set to false.
4. If the *specified entry* is activating but not fully activating, then set the current entry of the session history of *specified browsing context* to be the *specified entry*.
   Note: in this case, the current entry of the session history should be updated, but the document will not be fully active, so should not be loaded.
5. If the *specified entry* is fully activating, then traverse the history of the *specified browsing context* to the *specified entry*.
   Note: in this case, the document will be fully active, so should be loaded.

We believe that these changes bring the specification in line with our model, and so satisfies the fundamental property of navigation.

5. Conclusion

We have proposed a model of web navigation compatible with the WHATWG specification, and investigated its "fundamental property": that traversing by δ then by δ' is the same as traversing by $\delta + \delta'$. Unfortunately, the specified model does not satisfy this property, but we have shown that a patched model does. Experimentally, it appears that the patched model is closer to the behaviour of existing browser implementations.

References

[1] Minmin Han and Christine Hofmeister. Modeling and verification of adaptive navigation in web applications. In *Proc. Int. Conf. Web Engineering*, pages 329–336, 2006.

[2] May Haydar. Formal framework for automated analysis and verification of web-based applications. In *Proc. IEEE Int. Conf. Automated Software Engineering*, pages 410–413, 2004.

[3] May Haydar, Alexandre Petrenko, and Houari Sahraoui. Formal verification of web applications modeled by communicating automata. In *Proc. Int. Conf. Formal Techniques for Networked and Distributed Systems*, pages 115–132, 2004.

[4] Karl R. P. H. Leung, Lucas Chi Kwong Hui, S. M. Yiu, and Ricky W. M. Tang. Modeling web navigation by statechart. In *Proc. Int. Computer Software and Applications Conf.*, pages 41–47, 2000.

[5] WHATWG. HTML living standard. https://html.spec.whatwg.org/, 2016.

[6] Marco Winckler and Philippe Palanque. StateWebCharts: A formal description technique dedicated to navigation modelling of web applications. In *Proc. Int. Workshop Interactive Systems. Design, Specification, and Verification*, pages 61–76, 2003.

[7] Mechanical verification of navitation history. Agda formalization, 2016. https://github.com/ConnorGBrewster/ServoNavigation/tree/master/notes.