# Interaction Pace and User Preferences

Alix Goguey, Carl Gutwin, Zhe Chen, Pang Suwanaposee, Andy Cockburn

# Interaction Pace and User Preferences

Alix Goguey
alix.goguey@univ-grenoble-alpes.fr
Grenoble Informatics Laboratory
38400 Saint-Martin-d'Hères, France

Carl Gutwin
gutwin@cs.usask.ca
University of Saskatchewan
Saskatoon, Saskatchewan

Zhe Chen
zhe.chen@canterbury.ac.nz
University of Canterbury
Christchurch, New Zealand

Pang Suwanaposee
pang.suwanaposee@pg.canterbury.ac.nz
University of Canterbury
Christchurch, New Zealand

Andy Cockburn
andy.cockburn@canterbury.ac.nz
University of Canterbury
Christchurch, New Zealand

## ABSTRACT

The overall pace of interaction combines the user's pace and the system's pace, and a pace mismatch could impair user preferences (e.g., animations or timeouts that are too fast or slow for the user). Motivated by studies of speech rate convergence, we conducted an experiment to examine whether user preferences for system pace are correlated with user pace. Subjects first completed a series of trials to determine their user pace. They then completed a series of hierarchical drag-and-drop trials in which folders automatically expanded when the cursor hovered for longer than a controlled timeout. Results showed that preferences for timeout values correlated with user pace – slow-paced users preferred long timeouts, and fast-paced users preferred short timeouts. Results indicate potential benefits in moving away from fixed or customisable settings for system pace. Instead, systems could improve preferences by automatically adapting their pace to converge towards that of the user.

## CCS CONCEPTS

• **Human-centered computing** → **HCI theory, concepts and models**; **Empirical studies in HCI**.

## KEYWORDS

Interaction pace convergence, timeouts, user preferences.

**ACM Reference Format:**
Alix Goguey, Carl Gutwin, Zhe Chen, Pang Suwanaposee, and Andy Cockburn. 2021. Interaction Pace and User Preferences. In *CHI Conference on Human Factors in Computing Systems (CHI '21), May 8–13, 2021, Yokohama, Japan.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/1122445.1122456

## 1 INTRODUCTION

Results from experiments in communication studies show that *speech rate convergence* – adapting the rate of speech towards that of a communication partner – helps to foster positive experiences during communication. Experiments have indicated that convergence predicts cooperation and acceptance between people [33, 42] and that assessments of competence, social attractiveness, and affinity are increased when speech rate convergence occurs [4, 44]. Conversely, a failure to converge risks negative social outcomes, such as appearing impatient, abrupt, or rude (when speaking quickly to a slow speaker), or dull, soporific, or stolid (when speaking slowly to a fast speaker).

Computer-based user interfaces support a communication dialogue between users and their systems, and like dialogues between humans, both parties in the communication (the user and the computer) contribute to the overall *interaction pace*. On the human side (*user pace*) some users will carefully contemplate each action and express their intention with slow, deliberative input device manipulation. Other users, however, will make their decisions and manipulations as rapidly as possible. Furthermore, any individual user's preferred rate of interaction may periodically change due to factors such as fatigue, workload, illness, and stress.

On the system side of the interaction (*system pace*), many factors can influence the rate at which system states are transitioned, including the use of explicit timeouts, animations, network or processing latency and jitter, and the transfer functions used to map input events into system outcomes (such as scrolling and pointing gain functions). System features contributing to interaction pace are abundant, both at the operating system level and in applications. Examples in computer operating systems include the animated appearance and disappearance of windows, menus, and control panels (e.g., the Windows Start Menu and Mac OS Dock), pop-up tooltips and hotkeys that appear after a hover delay, and accessibility features such as screen readers, which have a speech rate and a hover timeout delay before screen reading initiates. Similarly, mobile operating systems and applications make extensive use of timeouts to discriminate tap, long- and very-long press, built-in delays to determine intentions (e.g., dragging an icon between homescreen pages), and input transfer functions (e.g., iOS keyboard delete key, which accelerates deletion the longer it is held down).

Importantly, while preferences for interaction pace are likely to differ between users (e.g., some sedate, others frenetic) and between time periods for the same user (e.g., when energetic versus fatigued), elements of system pace are frequently set to a fixed value by the designer in a 'one size fits all' manner. Sometimes designers provide configuration facilities that allow elements of system pace to be customised – but customisation features are known to be seldom

https://doi.org/10.1145/1122445.1122456

used [25, 31, 32], leaving most users with the unchanging default set by the designer.

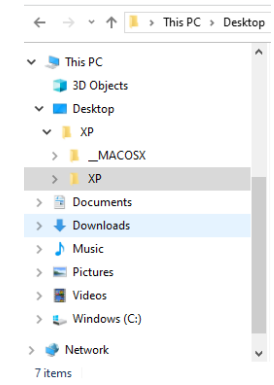
Instead of using fixed or user-customisable elements of system pace, systems could be designed to automatically adapt to better match the user's pace, which we term *interface pace convergence*. Exemplifying this type of behaviour, Giacolone [19] (in [41]) describes a gambling machine interface that adapts to the user's rate of play. When the user is slow to press the machine's "deal" button, an animation depicts hands slowly dealing cards to the user, but the animation is faster when the user is quick on the trigger: *"if one wants to play at a leisurely pace, the game will proceed at its normal play rate, but if the player's excitement level increases and he demonstrates a desire to play faster, the game play rate will be automatically increased"* [19, col. 2:8-11].

The system pace feature that we study later in this paper concerns the hover timeout that is widely used in hierarchical browsers, such as Microsoft's File Explorer, Apple's Finder, and in most email clients and programming IDEs – see Figure 1 for examples. During drag-and-drop activities, these interfaces use a built-in timeout to determine when a hierarchical item will expand to show its children; items automatically expand when the cursor hovers over them for longer than a timeout period. If the timeout is too short then unintended item expansions will occur, which is likely to be frustrating. However, if the timeout is too long then the user will need to excessively pause during their drag-and-drop actions, which is also likely to cause frustration. Although our study focuses on drag-and-drop interactions, users will typically encounter interactive features that influence the system's pace dozens or hundreds of times each day, and every time they do so there may be a small element of frustration associated with the behaviour being slightly too fast or slightly too slow for the user.

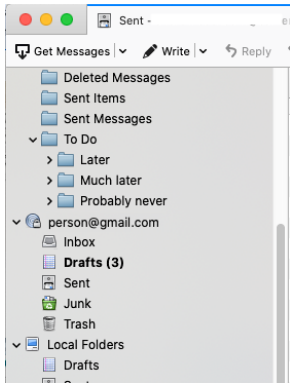
In this paper, we examine whether user preferences for different levels of system pace covary with user pace – do fast users prefer fast interfaces and slow users prefer slow? First, we describe theoretical foundations from communication studies that inspired our work, as well as reviewing prior work relating to interaction pace in HCI. We then describe the experiment that tests our main hypothesis – that preference patterns for slow- and fast-paced interface conditions differ between slow- and fast-paced users. Participants initially completed a set of drag-and-drop trials that did not involve the expansion feature (i.e., no timeout), with the data used to measure and classify *user pace* for each participant. Participants then completed three sets of hierarchical drag-and-drop trials in which folders expanded when a hover timeout was exceeded. The first two sets of trials involved a timeout of 250 ms and 750 ms (counterbalanced), with participants selecting their preferred interface. Before the third set of trials, participants used a slider and sandbox interface to personally configure their preferred timeout setting.

Results supported the hypothesis – experimental participants who were classified as fast predominantly preferred the interface with the shorter timeout, whereas those classified as slow predominantly preferred the longer timeout. Also, participants' configured timeout settings positively correlated with their mean performance on the initial trials that did not involve a timeout. We discuss how these findings relate to the concept of interface pace convergence, as well as possibilities for its implementation.

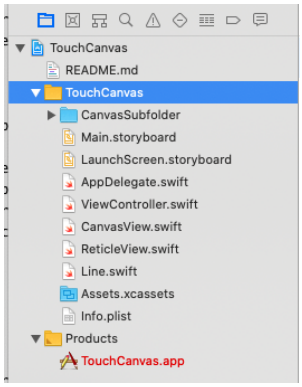
The contributions of this work are as follows:




**(a) Windows File Explorer.**




**(b) Thunderbird folders.**




**(c) Xcode project navigator.**

**Figure 1: Hierarchy interfaces that expand parent items based on a hover timeout during drag-and-drop actions.**

(1) We provide empirical evidence that user preferences for *system pace* (interface conditions that vary only in the duration of interface timeouts) covary with automatically measured *user pace*;
(2) We introduce the concept of *interaction pace convergence* and discuss design implications for improving the subjective appeal of interfaces by automatically measuring user's pace and converging system pace towards that of the user;

(3) We provide a review of factors relating to interaction pace, founded on communication theories from outside HCI.

## 2 BACKGROUND

Two main areas of related work, described below, provide the motivation and background for this research. First, we review findings from communication studies into the occurrence and outcomes of communication convergence, which indicate that convergence is correlated with positive affective outcomes, such as affinity and pro-social behaviour. Second, we review interactive systems and HCI literature that address issues related to interaction pace.

### 2.1 Inspiration from the Communication Studies Literature

*2.1.1 The occurrence of convergence in communication.* Communication accommodation theory and related work on phonetic convergence and entrainment concerns the adjustments that people make to attune their communication to one another, as well as the outcomes of succeeding or failing to do so. Communication patterns may *converge* towards one another, increasing similarity or synchronisation – for example, a speaker may talk faster and more energetically than they would normally when conversing with an energetic fast speaker. But patterns may also *diverge* away from one another, such as a person using clipped short sentences to respond to a speaker who is perceived to be verbose when time is short. Convergence and entrainment have been examined and demonstrated across many aspects of speech and communication, including the pitch and loudness of speech (e.g., Levitan and Hirschberg [28]), pronunciation (e.g., Pardo et al. [38]), the use of gestures (e.g., Chartrand and Bargh [4]) or lexical constructs (e.g., Bradac et al. [1]). Interested readers are directed to Giles et al. [20], Pardo et al. [38] and Lewandowski and Jilka [29] for more extensive introductions related to these general communication effects.

In this work, our particular interest is on temporal aspects of convergence, including widely studied speech rate effects. Jungers et al. [26], for example, showed that experimental participants adapted their rate of speech to converge towards that of fast or slow speakers in audio recordings, and results from Schultz et al. [42] show similar effects in scripted turn-taking dialogues. Further studies have replicated findings on speech rate convergence using standardised corpora [8, 18].

*2.1.2 Conditions influencing convergence.* Many studies of communication convergence seek to understand the conditions influencing its occurrence and strength, and some of these influences could have implications for human-computer interaction.

Lewandowski and Jilka [29] recently examined the influence of language talent, personality and attention. Results of their study showed that German native speakers who had higher competence in English converged more towards someone speaking English than those with lower English competence, suggesting that some level of fluency is necessary for convergence to occur. This finding might imply that convergence effects in interaction rely on some level of competence or expertise with the interface.

Studies by Dias and Rosenblum [12] suggest that speech convergence is stronger when communication partners are more socially present. In their studies pairs of communicators were seated approximately one and a half meters from one another, but in their two conditions pairs could either see and hear one another, or only hear one another due to a curtain separating their view. Pairs in the 'see and hear' condition converged more than those in the hear-alone condition. This finding could have implications for agent-based interfaces because convergence patterns (and their outcomes) may differ when between interaction with a voice-only agent and an agent that has a voice and visual avatar.

The sex of communicators is also known to influence convergence. Generally, results indicate that female talkers converge more than males, but a proper understanding of the effects of mixed-sex pairings is not yet established (see Pardo et al. [38] for a recent review). In interaction, there are possible implications for the choice of male or female agent voices with or without the visual appearance of an associated avatar.

*2.1.3 The outcomes and motivation for convergence.* While the studies above largely focused on whether communicators converge and the conditions under which they do so, some of the most interesting work for HCI examines the reasons for convergence and the outcomes of it.

In a one-shot prisoners' dilemma study, Manson et al. [33] observed that speech rate convergence indicated pro-social behaviour (i.e., greater cooperation on the prisoners' dilemma task) and that the participants evaluated each other more positively when their speech rates converged. They summarise as follows: *"temporally based behavioral coordination might facilitate pro-social behavior when the joint cooperative effort is itself perceived as a form of coordination"* [33, p. 419]. Manson's findings are echoed in other studies of different communication modes. For example, Chartrand and Bargh [4] observed greater liking between partners who mimic one another's gestures. Other studies have shown related effects of communication mimicry and convergence, including larger tips when a confederate waitress mimicked her customers' orders than when she did not [46], and that adapting language to increase similarity in grammatical structures and word use enhances mutual understanding (e.g., Pickering and Garrod [39]), the success of hostage negotiations [45], and the formation of romantic relationships [23]. In summary, previous work indicates a variety of positive outcomes from convergence, including pro-social behaviour and affinity between people.

### 2.2 Interaction Pace in HCI

In a human-computer dialogue, both parties contribute to the overall pace of interaction, through the *user's pace* and the *system's pace*. Early studies relating to pace effects in HCI largely focused on the implications of delays introduced by the system, with Shneiderman providing an insightful review almost 40 years ago [43]. Interestingly, Shneiderman's review alludes to the potential negative outcomes of a user converging to a system's fast pace: *"As users pick up the pace of a rapid interaction sequence, they may learn less, read with lower comprehension, make ill-considered decisions, and commit more data entry errors"* [43, p. 266][1]. Information about system delays is now routinely included in undergraduate HCI

[1]Jakob Nielsen recently made similar points in favour of slow interaction, although with tongue firmly in cheek. https://www.nngroup.com/articles/slow-ui/.

courses, with approximate threshold delays of less than 0.1 seconds for the user to feel that the system is reacting instantaneously, less than 1.0 seconds for the user to maintain an uninterrupted train of thought, and less than 10 seconds to maintain the user's attention [35].

System delays are often caused by compute-bound processes (such as searching a large data structure) or by limitations in network bandwidth, latency, or jitter. While these limitations still influence interaction, hardware improvements continually reduce the frequency and magnitude of their occurrence. However, unlike the delays that are imposed on the system by processing or transmission requirements, interaction effects with temporal properties such as animations and timeouts are often intentionally engineered into systems, and these intentionally engineered timeout effects are the focus of our work.

In the following paragraphs we first review the state of the art in contemporary user interfaces, demonstrating the widespread use of temporal effects in current interfaces. We then briefly review HCI literature investigating effects related to interface pace convergence.

*2.2.1 Interaction pace features in contemporary interfaces.* We see four main ways in which current interfaces are designed to incorporate elements that influence the system's pace: animation and motion effects, information rate effects, input device transfer functions, and temporal discrimination with overloaded input. Other system factors beyond these four may also influence the overall pace of interaction, such as the availability of interface shortcuts, but in general shortcut facilities are designed to enable the user to obtain the fastest pace possible, rather than imposing some element of system pace on the user.

**Animations** are widely used to accompany the appearance and disappearance of basic interface controls such as windows and menus. The use of animations has several potential advantages for the user, including softening visual effects that might otherwise be perceptually jarring, and providing a spatial cue to the source and destination of objects when they appear or disappear. By default, on Microsoft Windows 10 the Start Menu appears with a fast-in, slow-out animation, windows zoom and fade when they appear/disappear from the taskbar, and menus are animated. Microsoft Windows provides a personalization option to disable all animations, and specific animations can be selectively enabled using settings under 'performance options'. Similar facilities are provided by other operating systems, including Mac OS, iOS, and Android. Although it is possible for expert users to set the duration of menu animations by editing the MenuShowDelay value in the Windows system registry, few users would be expected to do so. Animations such as those described above are typically of short duration at around 300 ms. Interested readers are directed to Chevalier et al. [6], Hudson and Stasko [22] for high quality reviews of animation effects.

**Information rate** effects determine the amount of information presented to the user per unit time, and they can be related to animation effects. These effects are commonly used to manipulate the difficulty of computer games. For example, a game might present waves of items to contend with in each level (e.g., asteroids or enemies), manipulating the number of items and the time available. Visualisation techniques such as Rapid Serial Visual Presentation also manipulate information rate to assist activities such as rapid comprehension or search [11, 47].

**Input device transfer functions** are used to translate low-level signals received from input devices into output control effects displayed on the screen. For example, transfer functions determine the mapping between mouse and cursor movement, and on mobile devices they determine how swiping gestures influence scrolling movement. These functions are often sophisticated, attempting to appropriately amplify the user's input when it appears that the user wishes to move quickly, yet diminish input when it appears that the user wants precision. Transfer functions can therefore influence the system's pace of interaction. For example, a low-acceleration scrollwheel transfer function will require the user to repeatedly rotate the wheel to move through a long document; but conversely a high acceleration transfer function risks a twitchy behaviour that results in extensive over-shooting. Operating systems and input device vendors typically provide configuration interfaces that allow the user to directly control transfer function behaviour for cursor movement and scrollwheel operation. Interested readers are directed to Quinn et al. [40] for a discussion of transfer functions in touch scrolling and to Casiez et al. [2] for their use with mouse input devices.

**Temporal discrimination with overloaded input** is the area examined in our experimental work. Input devices offer a limited vocabulary of possible actions for the user to express their intent – for example, a mouse will typically have only two or three buttons, a scrollwheel, and a displacement sensor, and a touchscreen may report only the coordinates of one or more contacts. To increase the user's ability to express varied intentions designers often exploit the temporal components of user actions to discriminate intent. For example, using a mouse, two successive clicks are only interpreted as a double-click if they occur within a timeout period. If this time period is too short for the user then they will fail to reliably double-click, but if it is too long then the system may misinterpret separate manipulations as single double-click action. Temporal discrimination is also widely used on mobile devices – for example, on the iOS homescreen a single tap on an icon opens the object, a long press (a press longer than a timeout) posts a context menu, and a very long press (longer than another timeout) enters a reconfiguration mode.

Related timeout methods are also commonly employed in interfaces when the input mechanism used for a particular action is temporarily unavailable because it is consumed by an ongoing user action. For example, in hierarchical browsers a mouse left button click is used to expand a parent object and reveal its children. But during drag-and-drop actions the left button must remain pressed because releasing drops the dragged item onto the underlying object. Therefore, designers use a hover timeout to resolve the problem of overloaded input – if the cursor hovers over a parent item for longer than the timeout then the underlying object expands to show its children. This hover-expansion technique (also named 'spring-loaded folders' [9, 17]) is widely used across platforms and applications, including Windows Explorer, Mac Finder, most email clients, and many programming IDEs, and similar timeouts exist in a wide range of interfaces, including dragging items across homescreen pages on mobile devices.

System timeout values such as these influence a system's pace of interaction. In general, long timeouts permit slow-paced interaction,

but they also risk frustrating a user who wants to proceed more quickly. For example, a user who wants to drag an icon across several homescreen pages on their phone must wait for the timeout to expire at the edge of each page. Conversely, short timeouts permit fast interaction but risk frustrating the user by incorrectly identifying their intention – the user might accidentally change to the next homescreen page as a result of briefly dragging an item near the edge of the screen.

To gain insight into the timeout values used in current software systems, we used a screen recorder to inspect the timeouts used for hover expansion in the Mac Finder and in the Thunderbird email client. The Thunderbird value is fixed at 1000 ms. The Finder's value varies depending on which view is enabled. By default, the timeout is 600 ms in the column view, and 1000 ms in other views, although the user can configure the timeout using the 'spring-loaded delay' setting in System Preferences (the range of values available with the setting is 200-1200 ms in the column view, and 500-1200 ms in other views). We suspect that the column view is designed to have a shorter default timeout because users can easily recover from accidental expansions (expanded content is shown in a column to the right of the expanded item, leaving the display of parent items unaltered), whereas in the other views the expanded content replaces the original content, making accidental expansions harder to recover from. The different values used by the Finder suggest that designers have thought carefully about timeout implications. But regardless of this careful thought, in many systems the timeouts are configured in a 'one size fits all' manner, and even if customisation facilities are provided they are known to be seldom used [25, 31, 32]. Consequently, users who would prefer to proceed more quickly or more slowly are likely to be constrained by an invariant system pace.

*2.2.2 HCI research and interaction pace.* As mentioned above, Shneiderman considered interaction pace within his seminal review of system delays [43], and Dix [13] directly examined interaction pace in the context of computer-mediated collaboration between people. A few authors have contemplated how users might benefit from interfaces that promote slower, reflective, laid-back and restful interactions in mobile search [24], in a music player that adapts to the user's pace [15], and at a DIS conference workshop [36]. Others have scrutinised how aspects of performance and satisfaction change as users are prompted to alter their pace of interaction, demonstrating a strong increase in errors in a game [34] and in abstract pointing tasks [48].

The area of HCI research that has paid closest attention to pace and convergence effects is speech interaction. In an early and comprehensive study Oviatt et al. [37] demonstrated that 70-95% of child participants (aged 7-10) quickly converged their rate of speech towards that of an animated agent. More recently Dohsaka et al. [14] examined convergence in the opposite direction – where the duration of the agent's speech pauses converged towards that of the human – with Likert item responses suggesting positive subjective outcomes from system convergence. Related positive findings have been demonstrated for automated speech rate adaptation in synthesised Mandarin speech [5]. Other recent studies have examined speech convergence and entrainment effects to promote positive turn-taking behaviour in human users and to build positive social responses to the system [27, 30].

# 3 EXPERIMENT: USER PREFERENCES AND INTERACTION PACE

The review of prior work suggests that users' affinity and preferences for interfaces may be enhanced if the system's pace is more similar to their own. Furthermore, there may be opportunities to improve users' overall preference for interfaces by having the system converge its pace towards that of the user.

We therefore conducted an experiment to examine the first of these possibilities, scrutinising user preferences for system pace across levels of user pace. The hypothesis is formally expressed as follows:

$H_1$ User preferences for system pace (as exhibited through system delay timeouts) covary with user pace — faster users show stronger preference for shorter timeouts, and slower users show stronger preference for longer timeouts.

If supported, this hypothesis suggests that system designers could improve user preferences by matching the system's interaction pace (e.g., timeout duration) to the user's pace of interaction, in a form of interface pace convergence. However, there are at least three reasons to think that the hypothesis may not be supported. First, the automatic measurement of user pace may be insensitive, providing poor distinction between users. Second, measured user pace may be a poor predictor of user preferences for system pace – for example, all users might prefer faster (or slower) system response, regardless of their user pace. Third, the hypothesised effect may be sufficiently small to make it impractical to demonstrate at reasonable experimental scale.

To briefly summarise the method, participants first completed an initial series of drag-and-drop actions that did not involve the folder-expansion feature, with the data used to characterise their pace of interaction. Each participant then completed a series of hierarchical drag-and-drop tasks using two interfaces – one fast and one slow – that differed only in the hover timeout required before a hierarchical item would automatically expand to reveal its children. They then selected which of the two interfaces they preferred. Finally, they used a slider to configure and test their preferred timeout for a final series of drag-and-drop tasks.

## 3.1 Task Interactions

We based the experiment on drag-and-drop behaviours similar to those widely used in hierarchical file and email interfaces. Subjects had to drag a series of objects onto targets located in a hierarchical structure. At the beginning of each task the structure was maximally contracted, and when the cursor hovered over a hierarchical item for longer than a timeout period the item automatically expanded to reveal its child items. Each time an item expanded any previously expanded item at the same hierarchy level was contracted, so at most a single item at each level of the hierarchy could be expanded at once. The expansion/contraction of items was not animated.

We chose to base the experiment on icon drag-and-drop for four main reasons:

(1) *Familiarity* – the participants would be familiar with these behaviours from everyday computer use and should therefore be able to understand task requirements without extensive training.
(2) *Simple and modelable interface actions* – the participants' actions in completing the tasks include simple manipulations that are amenable to well-validated models such as Fitts's Law [16].
(3) *Ease of experimental control* – related to the previous point, system properties within these behaviours can be precisely configured, including the distance that an icon must be dragged, the target size, and the hover timeout before an item will expand.
(4) *Ecological validity* – several widely disseminated interfaces (including the Mac OS Finder, Windows Explorer, and most email clients) include system-imposed hover timeouts.

Each task was cued by showing a blue icon containing a number series at the top of the display. The number series, such as "3.1.2" indicated the required target. The hierarchical structure was shown immediately beneath the cued item, initially showing the fully contracted view of sixteen items, enumerated 1 – 16 (see Figure 3b, which shows a partially expanded hierarchy).

## 3.2 Timeout setting and pilot studies

The key manipulation in the study was the duration of the hover timeout. If the timeout was too short, then unintended items would expand while the user dragged the item towards the target (false-positive expansions, potentially causing user frustration). And if the timeout was too long, then users would be overly delayed, potentially causing frustration.

Figure 2 shows a characterisation of the effects of different timeout values on fast- and slow-paced users, according to our hypothesis. The figure highlights the importance of choosing timeout values that suffer neither flooring nor ceiling effects (timeout values that are respectively so short or long that nearly all users dislike them).

The timeout values used in our experiment were 250 ms for our *fast* condition and 750 ms for our *slow* condition. These timeout values were selected following a series of pilot studies using an identical method to that described below, except for the tested timeout durations. The first pilot study (n=15) used values of 450 ms and 900 ms, with 100% of participants preferring 450 ms; the second pilot study (n=5) used 400 ms and 800 ms, and again 100% preferred the faster condition; the third pilot (n=12) used 200 ms and 800 ms, with 70% preferring the slow condition. A final pilot study (n=10) used 250 ms and 750 ms, with a 50:50 split in preferences. We therefore used these values of 250 ms and 750 ms for the main experiment.

## 3.3 Procedure

We conducted the experiment on Amazon Mechanical Turk. Each participant proceeded through three experimental stages: 1. user pace determination; 2. system pace experience and preference; 3. setting and experience of user-tailored system pace.

*3.3.1 Stage 1. User pace determination.* After collecting background demographics (age, gender, pointing device, frequency of computer use and game play), participants completed a set of one-dimensional

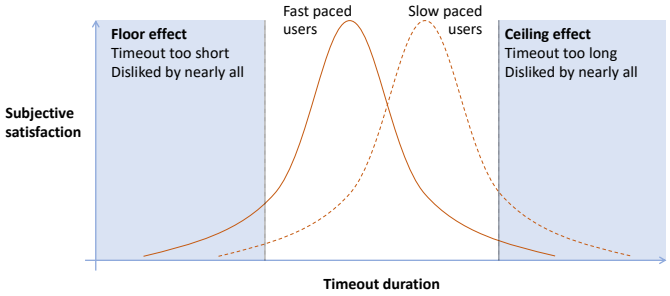
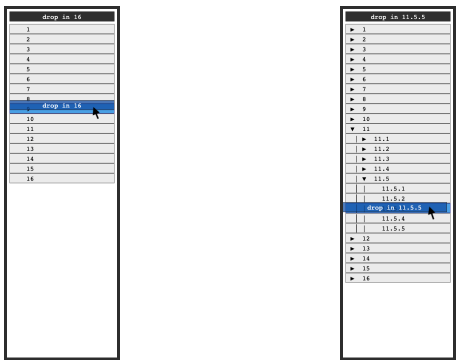



**Figure 2: Characterisation of hypothetical effects of timeout delays on subjective satisfaction for fast- and slow-paced users, including the role of flooring and ceiling effects.**



**(a) Stage 1 task (no hierarchy).** **(b) Stage 2 & 3 task (hierarchy).**

**Figure 3: Drag-and-drop tasks. The item was torn off from the top and dragged to the numerical target.**

drag-and-drop trials. A web page instructed participants that they needed to drag a blue box object containing a number onto the target grey box showing the same number. Participants clicked a button labelled 'Start Tasks' at the bottom of the instruction page when ready to continue.

Sixteen grey target objects were vertically arranged below the blue dragged object, enumerated 1 – 16 (see Figure 3a). The dragged object moved with the cursor and turned green while the cursor was over the target. Participants completed each trial by dropping the dragged object while it was green. Correctly completing one trial caused the next target to be immediately displayed in the blue box at the top of the display. If the object was dropped onto the wrong target, the same blue target item was re-displayed at the top of the list (preventing participants from racing through the experiment without regard to accuracy).

Each participant completed 20 drag-and-drop trials, consisting of four familiarisation tasks (dragging items to randomly selected odd-numbered targets) and 16 controlled tasks (two repetitions for each of the even-numbered targets).

Data from these trials was used to determine each participant's user pace, as determined from their mean time on error-free trials.

*3.3.2 Stage 2. System pace experience and preference.* On completing the final task in stage 1, a web page displaying instructions was

automatically shown for the next experimental stage. In stage 2, participants completed trials with two different settings for system pace, and then chose their preference. The instruction page included a sample hierarchy and instructed participants to drag the blue box to the target identified by its number string, such as '5.2.3', meaning that the item should be dragged to item 5 at the first level, item 2 at the second level, and items 3 at the third level (see Figure 3b). Participants were required to complete four familiarisation trials on the instruction page, each involving dragging the object to a target at the third level of the hierarchy. In these familiarisation trials, each level of the hierarchy opened after a hover timeout of 1000 ms.

Having completed the four familiarisation trials, a 'Next' button appeared, and clicking it advanced to a new page informing participants that they would use two systems – 'System A' then 'System B' – to complete two sets of hierarchical drag-and-drop trials. Participants clicked a button labelled 'Start System A Tasks' when ready to proceed. After completing the set of trials with System A, they then clicked a button labelled 'Start System B Tasks' to begin the second set of trials.

Similar to the previous stage, the blue box to be dragged was shown at the top of the display, immediately above sixteen grey box items, numbered 1-16 and prefixed with a symbol '►'. Consistent with many commercial interfaces, the ► symbol indicted that the object contained children. At the start of each trial all items within the hierarchy were collapsed. When the dragged blue box hovered over a hierarchical parent item for longer than a timeout period, the item would expand to reveal its hierarchical content (see Figure 3b). Only one item at each level of the hierarchy could be expanded at any time, so when a hierarchical item was expanded, any previously expanded item at that level was automatically contracted. Each of the sixteen top-level items contained five child items (enumerated 1-5), and each of the child items contained between one and five grand-child items – item $n.1$ contained four items, $n.2$ three, $n.3$ two, $n.4$ one, and $n.5$ contained five items. This structure was used to reduce target item spatial stability when unintended items were expanded (as normally occurs in real data hierarchies) – if all items had the same number of children then the location of a target would remain constant when one item expands and another contracts, for example, item 10 would remain spatially stable while the user dragged downwards to replace the five children of 2.1 with five children of 2.2. As at most only one item at each level of the hierarchy could be expanded, the maximum number of items displayed at once in the hierarchical view was 26 (16+5+5), plus the dragged item.

The dragged object turned green when it was over the final target. Dropping the object while green completed the trial, causing the next trial to be immediately displayed; dropping the object on the wrong target caused the trial to begin anew, with the hierarchy fully contracted. If the cursor was moved outside of the list while dragging the object, the same blue target item was re-displayed at the top of the list (preventing participants from avoiding the expansion feature).

Each participant completed 12 trials with each of the two systems ('A' and 'B'), comprising one selection at each of the top-levels targets in the range 3-14. Each second and third level target was always item 5. We avoided using the top-most and bottom-most top-level items because they are least likely to induce unintended expansions (e.g., there is no expandable item beneath item 16, so the user can slowly approach item 16 from below without risking unintended expansion, unlike other locations). We always used the last item (item 5) at the second- and third- level because it is the most likely item to suffer unintended expansions – for example, dragging off the bottom of target item 6.5 or 6.5.5 risks the unintended expansion of item 7.

Having completed all of the trials with both Systems A and B, participants were asked to respond to the following forced-choice question:

> If completing these tasks again, I would prefer to use:
> □ System A □ System B

The only difference between System A and System B was the length of the timeout period used to determine that the dragged object had hovered over a hierarchical object for sufficiently long to trigger its expansion. The two settings were *fast* (250 ms) and *slow* (750 ms). For half of the participants, 'System A' was fast and 'System B' was slow, with the inverse for the other half.

*3.3.3 Stage 3. Setting and experience of user-tailored system pace.* Up to this point in the experiment the participants had not been informed about the use of different timeouts in the systems. A web page instructed participants that Systems A and B differed only in the length of the hover timeout used for item expansion. The web page included a slider showing the time settings used for Systems A and B (see Figure 4). A slider handle allowed users to set the timeout for an upcoming set of trials with System C. The web page also included an interactive 'sandbox' version of the System C interface, showing the same hierarchy as that used in Stage 2.

Participants were instructed that they would complete a final set of drag-and-drop trials using 'System C', and that they should first use the slider to set their preferred timeout value in the range 1-1500 ms. The timeout was initially shown to match that of their preferred option (i.e., that used by their preferred of System A or B). Once the participants had manipulated the slider at least once, and completed at least one selection using the sandbox interface, a button at the bottom of the page was enabled stating 'Start System C Tasks'.

Finally, participants were asked to type any final comments they might have about the experiment. The experiment terminated with a web page that thanked them for their time.

## 3.4 Subjects and Apparatus

The experiment involved 208 participants on Amazon Mechanical Turk. Conditions for inclusion in the experiment were that each crowdworker had to be based in the United States, with a HIT approval rate greater than 90%, and with at least 1000 approved HITs. The participants' mean age was 36.6 years (s.d., 10.5, min. 18, max. 70); 61.5% self reported as male, 37.5% as female, and 1% declined to answer. Participants were asked to report the input device used for the study, with 83% reporting that they used a mouse, and 17% using a trackpad. Operating system use was divided between Microsoft Windows (89%), MacOS (8%), and Linux (3%).

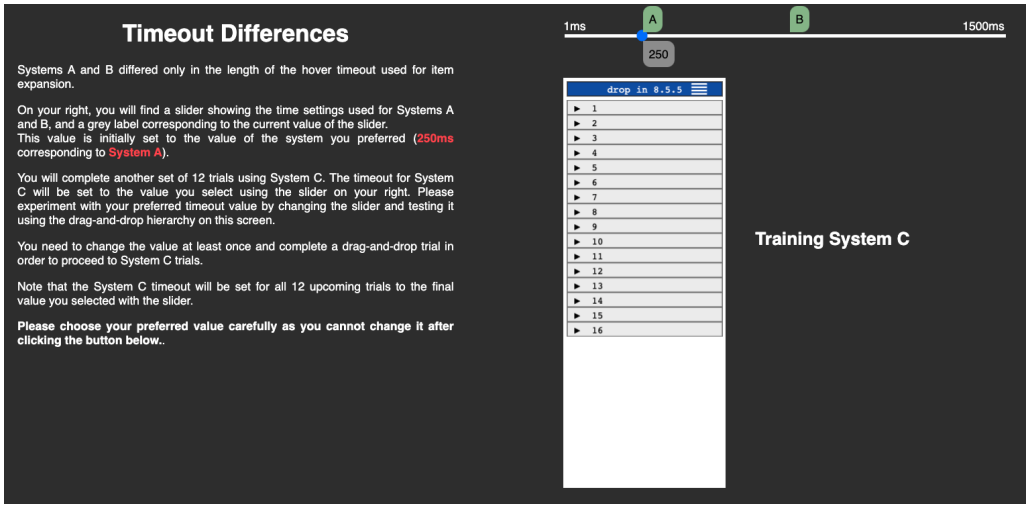



**Figure 4: Timeout configuration page and sandbox hierarchy, used to set timeout in Stage 3 of the experiment.**

The experiment took approximately 10 minutes to complete, with participation rewarded with a payment of USD$2.50 (apportioned from an hourly rate of USD$15 per hour).

Software was written in HTML/CSS/JS, using the Paper.js library and a Firebase database to log all user actions, including the time taken to complete each of the trials and any errors made.[2]

## 3.5 Design

The hypothesis $H_1$ (that fast users have stronger preference for fast timeouts, and slow users have stronger preference for slow timeouts) is tested through two main analyses.

The first analysis compares the proportions of participants who choose the *fast* system in preference to the *slow* system (when choosing between System A and System B) across three quantile bands of user pace classification (slow, medium, and fast) based on their performance in the stage 1 tasks. The dependent measures for this analysis are as follows:

(1) *user pace classification* – the classification of each user's pace from their data in the first stage of the experiment, dividing participants into three quantile bands based on their mean trial completion times in the stage 1 drag-and-drop tasks ($Q_1$, the fastest $33^1/_3\%$ of participants; $Q_2$, the second quantile, representing the middle third of participants; $Q_3$, the third quantile, representing the slowest third of participants).
(2) *system pace preference* – the participant's binary preference choice of either the *fast* or *slow* interface (i.e., the preference of either System A or System B after stage 2 of the experiment).

The hypothesis is tested using a $\chi^2$ test of proportions. The hypothesis requires that a higher proportion of participants who are classified within Q1 (the fastest participants) choose the *fast* interface than do those who are classified within Q3 (the slowest participants).

The second analysis examines the relationship between the participants' user pace (as indicated by their mean performance in the stage 1 trials) and the timeout value that they select using the slider for System C. The dependent measures for this analysis are as follows:

(1) *user pace* – each participant's mean time on stage 1 trials.
(2) *System C timeout setting* – the time that each participant selects as desirable for their set of trials with System C.

In this second analysis, support for the hypothesis requires a positive correlation between user pace and System C timeout setting – fast-pace users with a low mean time on stage 1 trials should set a low timeout value for System C, and slow pace users should set a high timeout value for System C.

Additional secondary analyses are also conducted to further characterise the results, including analyses of the effects of participant gender and gaming experience, as well as participants' comments[3].

# 4 RESULTS

## 4.1 User pace characterisation from stage 1 trials

The analysis of results requires first determining each participant's *user pace* from their performance in stage 1 trials. We only included data for correct selections because including the time for errors could misrepresent a 'rushing' user who is fast but inaccurate as having a slow pace of interaction.

Participants' mean time for stage 1 trials ranged from an extremely fast 656 ms to a sedate 4100 ms, with an overall mean of 1544 ms (s.d., 624, 95% CI [1458, 1631]).

We computed quantile values that split the participants into three equally sized pools according to their mean performance on stage 1 trials, classifying users as 'fast', 'medium', and 'slow'. We used three quantiles (rather than two or four) for three reasons: first, to provide a simple and natural alignment with our hypothesis; second, to

[2]The software can be accessed and run at https://www.cosc.canterbury.ac.nz/andrew.cockburn/AdaptivePace/

[3]Experimental data and analysis scripts are available at https://osf.io/2e5yw/

provide a clear separation between fast and slow categories; third, to ensure a large sample within each of the fast and slow categories. The fastest 33⅓% of participants were categorised as 'Q1, fast' if their mean trial time was less than 1219 ms; the middle 33⅓% were classified as 'Q2, medium' if their mean was in the range 1219-1567 ms; and the slowest third of participant were categorised as 'Q3, slow' if their mean was greater than 1567 ms.

We computed an overall Fitts's Law model [16] of movement time (MT) incorporating all participants' means at each target distance. The model was strong ($MT = 705 + 381 \times ID, R^2 = 0.97$), with the comparatively high intercept (705 ms) and slope (381 ms) values best attributed to the participants' tasks involving dragging rather than traditional Fitts's Law tapping tasks. Participants' individual Fitts's Law models ranged from good ($R^2 = 0.87$) to non-existent ($R^2 < 10^{-5}$). We inspected the logs for participants with particularly poor Fitts's Law conformance (19 participants with Pearson $r < 0.2$) and found no explanation. However, our experimental instructions did not encourage participants to complete tasks at any particular speed or accuracy (unlike typical Fitts's experiments, which prompt participants to complete trials 'as quickly and accurately as possible'). We maintained all participants' data in our analysis.

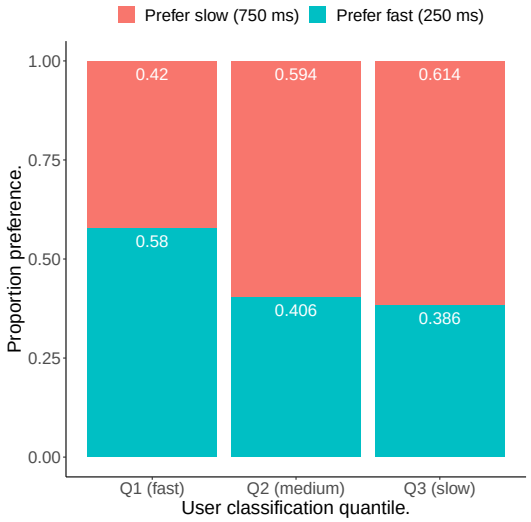
## 4.2 Preference for fast or slow system A/B, across user pace characterisation

Although not part of our hypothesis, there was a general preference for the first system experienced, with 59% of participants choosing System A (used first) over System B ($\chi^2 = 6.58, p = .01$). Similar preference rates for the first condition in forced-choice experiments have been observed in other studies (e.g., [21]). The proportion of participants who experienced the fast condition as System A was similar across quantiles (Q1-fast 49%, Q2-medium 51%, Q3-slow 50%). As intended by our experimental design and pilot testing, when averaged across the three quantiles overall preferences for the fast and slow interfaces were fairly balanced, with 46% preferring fast and 54% preferring slow.

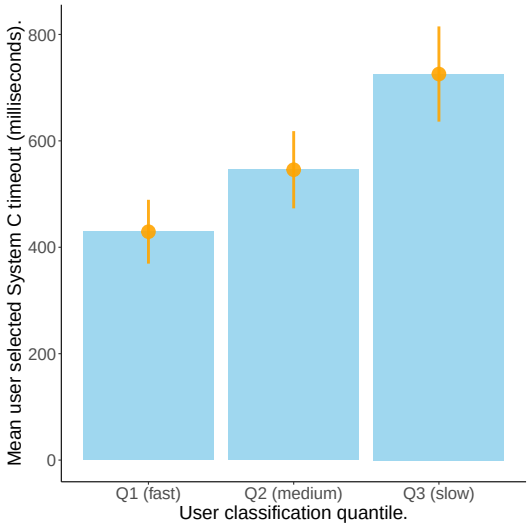
Figure 5a summarises the main result, showing the proportion of users in each classification quantile that selected the fast (250 ms timeout) or slow (750 ms timeout) System A/B as their preferred interface. Importantly, 58.0% of users who were classified in the Q1-fast quantile selected the fast 250 ms system as preferred to the slow 750 ms system. This preference for the fast interface decreased to 40.6% for the Q2-medium paced users, and to 38.6% for the Q3-slow paced users. A three-way $\chi^2$ test of proportions shows that data as extreme as this sample should seldom occur in the absence of an underlying effect: $\chi^2 = 6.35, p = .042$. This supports $H_1$, with further analyses of $H_1$ below.

## 4.3 System C timeout setting across user pace

The second main analysis concerns the relationship between *user pace* and user's preferred slider setting for the System C timeout. The mean value that participants set across quantiles is summarised in Figure 5b, from a low of 429 ms (s.d., 250 ms, 95% CI [369, 489]) for Q1-fast participants, through a mean of 546 ms (s.d., 302 ms, [473, 618]) for Q2-medium, to a maximum mean of 726 ms (s.d., 375 ms, 95% CI [636, 815]) for the Q3-slow participants. Analysis of variance for this data yields $F_{2,205} = 15.77, p < 10^{-5}$.




**(a) Proportion of participants choosing the fast (250 ms) or slow (750 ms) system A/B, across classification quantile.**




**(b) Mean time that participants set for the System C timeout, across quantile. Error bars show 95% confidence interval.**

**Figure 5: Main results from the experiment across 33⅓ quantiles of user pace (Q1 fast to Q3 slow).**

The overall relationship between participant's mean time on stage 1 trials and their selected timeout value for System C is summarised in Figure 6. The Pearson correlation coefficient for the relationship is positive, indicating that faster users tended to select shorter timeouts, and slower users selected longer timeouts ($r = 0.41, p < 10^{-5}$, classifiable as a medium effect [7]). The figure uses color to depict each participant's preference selection of the fast (blue) or slow (red) version of System A/B, indicating a dominance of preference for fast interaction in the bottom-left of

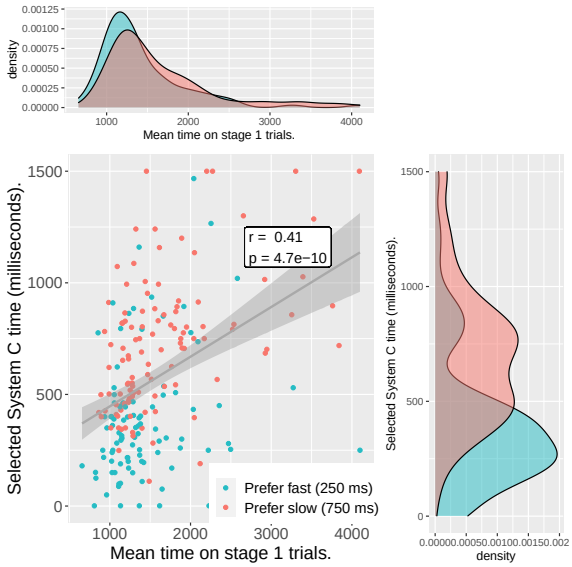



**Figure 6: Relationship between participants' mean time on stage 1 trials and their selected System C time (line of best fit for all data). Blue data encodes participants who preferred the fast system A/B (250 ms), red encodes participants who preferred the slow system (750 ms).**

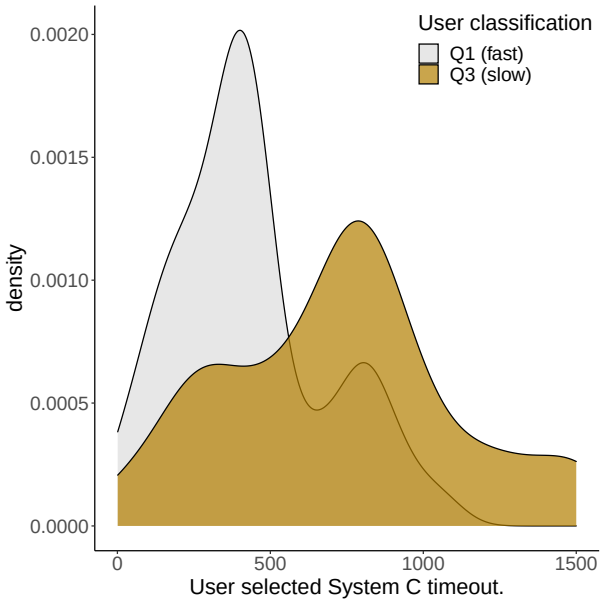



**Figure 7: Distribution of System C timeout selections for users classified in Q1 (fast) and Q3 (slow), corresponding to hypothetical effects characterised in Figure 2.**

the main chart; density plots at the top and right of the figure also highlight distributions for participants who preferred the fast (blue) and slow (red) interfaces.

Figure 2 showed a characterisation of the hypothetical effects of timeout delays on subjective satisfaction, for fast- and slow-paced users. Figure 7 shows the experimental data relating to this characterisation, plotting the distribution of timeout values that participants selected for System C across users classified as Q1-fast and Q3-slow. The two peaks in the user-selected System C timeout values are clearly separated between Q1-fast and Q3-slow participants – a Kolmogorov-Smirnov test indicates that these two samples are unlikely to arise from the same distribution $D = 0.48, p < 10^{-5}$. It is also notable that the distribution is tighter for Q1-fast participants than it is for the Q3-slow participants, which is reflected in the participants' comments, discussed below. In general, fast participants seemed particularly clear about their need and desire for a fast system, whereas the opinions of participants classified in Q3-slow were more variable. These results all support $H_1$.

## 4.4 Effect of gender, gaming frequency, and age

Given the significant gender effects reported in studies of convergence effects (see Section 2.1), we separately analysed the proportions of males and females preferring the fast and slow Systems A/B across their respective quantiles. A numerically higher proportion of males (50%) than females (38.5%) chose the fast system as preferred to slow ($\chi^2 = 2.16, p = .14$). For males, the preference difference was pronounced across classification quantile, with 67.4% of participants classified in Q1 preferring the fast interface compared to only 41.9% for those in Q3 (three sample test of equality of proportions, $\chi^2 = 7.9, p = .02$). In contrast, female preference for the fast interface was much lower at 42.3% for Q1 participants and only 26.9% for those in Q3 ($\chi^2 = 2.3, p = .32$). Despite these suggested preference differences between males and females, there is little evidence that this can be attributed to a performance difference between genders – the mean time on stage 1 trials was 1518 ms for males compared to 1594 ms for females (unpaired T test, $T_{174} = -0.86, p = .39$).

As many computer games expose users to a demanding pace of interaction, we wondered whether the frequency of gaming use might influence preferences for timeout duration, and whether males played games more than females. We therefore conducted a Spearman rank correlation between participants' self-reported frequency of gaming use and their setting for the System C timeout, showing at most a weak relationship ($\rho = 0.13, p = .058$), suggesting that gaming frequency was not a strong influence on our results. Male and female responses for the time spent playing games each day were similar, with 19% of males and 27% of females reporting playing 0-1 hours per day, and 57% of males and 52% of females reporting playing 1-4 hours per day. Finally, analysis of the relationship between participants' age and their setting for System C time also showed no clear relationship (Pearson $r = .08, p = .24$).

## 4.5 Participant comments

Participants' comments amplified the quantitative observations above. Many of the most forceful comments were from the fastest users who expressed strong disliking for the 750 ms timeout in System A/B. For example, the participant with the lowest mean time on stage 1 trials (participant 1389, mean 656 ms) commented *"I hated the length of the delay in system A."* This participant set the

System C delay to 180 ms. The participant with the second lowest mean time on stage 1 trials (participant 1100, mean 808 ms) set the System C timeout to the minimum value available at 1 ms, but subsequently commented that this was too short: *"When I did the initial trials I thought that setting it to the quickest speed would be the most efficient and it was for many of the trials. But, if I got off track at all, it was harder to get back to the right section to make the drop. The overall speed was a positive but it wasn't perfect. Doing a few more trials now makes me think that something around 75ms would work best for me. It's just enough of a delay for me to easily get in the right box, but not so much of a delay as to be annoyingly slow."*

At the other end of the spectrum of user pace, several participants in the Q3-slow quantile made comments referring to the frustration of the timeout being too short. For example, participant 1193 (mean 3254 ms) commented that she had *"a difficult time controlling the mouse to move the lines down to the correct line"* and that her selected timeout value of 857 ms was selected *"because I thought that maybe I could keep up with it better"*, which suggests a desire for an improved match between her pace and that of the system. Similarly, participant 1335 (mean 3760 ms) commented *"It was aggravating when the numbers would drop down before I wanted them to, the precision was very difficult"*, and he set a high timeout value of 897 ms, further commenting after completing the System C trials that even this was too fast *"If I could, I would have gone back and chosen more time before the drop-down occurred."*

# 5 DISCUSSION

The results and participant comments indicate that user preferences are positively influenced when the system's pace better matches an automatic measurement of the user's pace. Participant preferences for slow and fast system pace aligned with our automatic categorisation of participants as having slow, medium, or fast user pace, and the timeout values that users explicitly set for their upcoming trials correlated with their mean time on an earlier set of trials that did not involve a timeout. In short, as hypothesised, fast users preferred a faster interface, and slow users preferred a slower one.

The following subsections discuss the broader meaning of these results, examining the potential causes of the observed effect, the relationship between our results and the concept of interface pace convergence, discussing ways that systems might measure user pace to facilitate interaction pace convergence, and exploring implications for design.

## 5.1 Why do the results matter?

Figure 5a indicates a 19% shift in preferences for the fast version of System A/B across quantiles – from 58% preferring the fast interface in the Q1-fast quantile to 38.6% in the Q3-slow quantile, with a larger shift for males (25.5%) than females (15.4%). And the mean timeout value that participants explicitly set for their System C trials differed by 297 ms between Q1-fast (429 ms) and Q3-slow participants (726 ms), representing a 69% increase across quantiles. While statistical analyses show these and other measures to be unlikely due to chance, there are legitimate questions about the practical significance of the findings.

We see three main reasons that the findings have important practical implications. First, interface designers continually seek reliable means to make small improvements to user experience, and our results suggest a new avenue for doing so – *interaction pace convergence*, discussed further in Section 5.3.

Second, small elements of dissatisfaction or suboptimal system behaviour can have amplified effects on the user when they occur frequently. This was particularly evident in the comments of some of the Q1-fast users regarding the slow behaviour of System A/B, one of whom stated that he 'hated' the delay.

Third, although a difference of 297 ms between the mean times selected by Q1-fast and Q3-slow users may seem small, the features of contemporary interfaces suggest that designers understand that 'small' differences in timeouts have important implications. For example, in the Mac OS Finder the default timeout for spring-loaded folder expansion differs by 400 ms between column view and other views (suggesting an explicit design choice), and the full range of spring-loaded timeouts that users can configure covers only 700 ms from fastest (500 ms) to slowest (1200 ms).

## 5.2 What causes the observed effect?

We were inspired to conduct this research by findings on speech rate convergence from communication studies, as briefly reviewed in Section 2.1. The results reported above are consistent with underlying theories of convergence – our participants' preferences and settings aligned with the notion that users have greater affinity for interfaces that are more similar to them in terms of pace. However, correlation is not causation, and a variety of factors other than affinity for similarity might have contributed to the observed effects. For example, the results could be framed within flow theory [10], with an explanation that participants were best able to experience the desirable state of flow when the difficulty of the interface (due to the timeout setting) was best balanced with their level of skill. Under this theory, participants would become anxious if the timeout was too short for their skill level; and they would become bored if the difficulty of the interface was too low (due to its long timeout). Other factors, such as participants' variable tolerance for errors may have influenced preferences, as might variations in the strategies that users adopted to complete the tasks. Regarding strategies, during our pilot studies we noted that one very fast user assisted his performance with extremely short timeouts by intentionally overshooting beyond the intended target, then moving more slowly upwards to expand the intended target – in this way he reduced the incidence of target movement that occurs when under-shooting in a downward direction.

There are extensive opportunities for further experimental work to examine and tease apart causal factors influencing our results.

## 5.3 Relationship between the results and interface pace convergence

Regardless of the underlying theoretical foundation or cause of the observed effects, the primary finding remains that preferences for interface conditions were improved when the system's pace better matched the user's pace. One important design implication of this is that systems could adapt interface properties such as timeouts to converge the system's pace to better match that of the user in a form of *interface pace convergence*.

Although *some* interfaces provide facilities that allow the user to configure timeouts many others do not, leaving users with a 'one size fits all' system pace. Even when customisation facilities are provided they often go ignored, and they necessarily increase the complexity of the interface presented to the user. Our results suggest that instead of invariant pace and instead of customisation facilities, interfaces could be designed to observe the user's pace of interaction, and automatically adapt interface properties to converge towards the user's pace. In this way, the interface might become more responsive (e.g., a shorter timeout) when used by a fast user or when the user is inferred to be rushing, and the timeout might increase when the user is slow or interacting more leisurely.

However, the conditions tested in our experiment did not directly involve the system converging to the user's pace. Instead, the method measured the users' pace, automatically categorised the participants as fast, medium, or slow, and exposed participants to fast and slow conditions. As predicted, users who were categorised as fast had stronger preference for the fast condition than slow users, and those categorised as slow had stronger preference for the slow condition than fast users.

While this method did not involve interface pace convergence, participants were not aware of the basis for their exposure to the settings of System A and B, and those settings could theoretically have arisen from interface pace convergence. For example, a simple convergence algorithm might be 'if the measured user pace is fast, make the interface more similar to the user's pace by using a short 250 ms timeout; and if the measured user pace is slow, make the interface more similar to the user's pace by using a long 750 ms timeout.' An experiment testing this simple form of interface pace convergence might initially measure the user's pace, then have users choose their preferred version of the interface after completing two sets of trials, one with the algorithm correctly applied, and the other with it incorrectly applied. Fast and slow users should have stronger preferences for the interface with the correctly applied convergence. Importantly, from any participant's perspective, this hypothetical experiment testing interface pace convergence would be identical to the one our participants actually conducted.

As it stands, our results indicate that user preferences for timeout values (which influence system pace) covary with user pace. Further empirical work is needed to test their generalisation to interface pace convergence.

## 5.4 What data could determine the user's pace as a basis for convergence?

The possibility for interface pace convergence raises questions about how the system might measure and determine the user's pace, as well as the frequency of any such measurement and resultant adaptation. To classify participants as slow, medium, or fast, our experiment used data from the stage 1 trials, where every trial had a clear starting and terminating action (initiating the drag at a specific location, and dropping the item on a predetermined target). However real interaction often lacks such clear start and end points. For example, it can be very difficult to determine where and when a particular pointing action begins, and whether the target is successfully hit or missed [3].

There are many opportunities for examining factors that might indicate users' pace. These include the mean time between successive interface manipulations (as suggested by Giacolone Jr [19], Schüll [41]), and the velocity profile of manipulations such as scrolling and pointing. Further work could also examine the use of biometrics such as eye gaze movement, pupil dilation, galvanic skin response, and others as proxies for indicating user pace.

## 5.5 Lessons for hierarchy interfaces

Our experimental tasks involved drag-and-drop behaviours that are a standard part of a wide range of interfaces that support hierarchical browsing. We used these interactions as a convenient and familiar exemplar for studying general concepts relating to user- and system-pace, and not because we wanted to make specific recommendations for hierarchical interfaces. However, the results suggest that a non-configurable predetermined timeout for hierarchy expansion is suboptimal. Furthermore, only 10% of our participants set a timeout value for System C that exceeded one second, yet a one second timeout is used in many systems (including Thunderbird), suggesting that the majority of users would prefer faster operation with a shorter timeout.

However, designers face complex tradeoffs around these timeout values. We suspect that designers are knowingly setting a timeout value that is too long for the majority of users. Setting a value that is too short for a user will cause unintended items to expand, potentially moving the target item from its original location on the display, which is likely to be highly frustrating as the user will effectively need to 'fight a twitchy system' during drag-and-drop. Conversely, setting a value that is too long is likely to be somewhat frustrating to a rushing user, but the frustration is unlikely to be as potent as that for the slow user fighting the system (unless a fast user is continually delayed during a long series of drag-and-drop activities, as was the case in our experiment). Therefore, we suspect that designers choose to avoid the potent frustration of a too short timeout by setting a timeout that is mildly frustrating for the majority. Designers might also contemplate allowing users to configure the timeout value, but doing so would add complexity to the user interface (requiring a configuration dialog and interface mechanisms to activate it), and it is unclear that a large proportion of users would make use of the customisation facilities [31].

## 5.6 Further work on interface pace convergence

There are abundant opportunities for further research on interface pace convergence. As mentioned above, there are interesting questions about how a system might measure and infer the user's pace. There are also interesting questions about the frequency with which a system might beneficially adapt to the user's pace – minute-by-minute adaptations could risk the user perceiving the system as inconsistent, but hour-by-hour might beneficially capture important pace changes stimulated by factors such as post-meal blood sugar levels. Between-subject factors also raise an abundance of interesting questions, especially when the interface is gendered, as is the case with speech interfaces (e.g., Siri and Alexa). Research from communication studies has extensively observed differences

in convergence between mixed and same sex communication partners, and it would be interesting to understand how the effects of interaction pace convergence vary across user genders when the interface itself is gendered.

Finally, our study involved a batch of sequential drag-and-drop activities completed by crowd-workers on Amazon Mechanical Turk for whom there is a clear imperative for completing tasks in the minimum time possible. Like all empirical studies, there is need for replication and method triangulation to examine the boundaries of generalisation and validity.

## 6 CONCLUSION

Extensive results from communication studies indicate that affinity and pro-social behaviour can be enhanced when people adapt their communication patterns, such as the rate of speech, to better match their communication partner. Inspired by these findings, we examined whether users would have stronger preferences for an interface that better matched the user's pace of interaction. Experimental results aligned with our hypothesis – faster users showed stronger preference for a faster interface, and slower users had stronger preference for a slower interface. This finding has three potentially important design implications. First, interface designers often seek efficiency optimisation – trying to make the interface as fast as possible – yet this finding suggests that doing so could impair satisfaction for users who prefer a more leisurely pace[4]. Second, interface designers often build predetermined timeouts into their systems, imposing a 'one size fits all' element of interaction pace into the system; alternatively, designers sometimes provide configuration interfaces that allow users to customise timeouts, but these often go ignored, leaving users with the designer's predetermined default. The experimental results suggest that an invariant default system pace will leave many users with a system pace that is mismatched with their preferred pace. Third, there are abundant design and research opportunities for new interfaces that observe and adapt to the users pace, in a form of interaction pace convergence.

## REFERENCES


[1] James J. Bradac, Anthony Mulac, and Ann House. 1988. Lexical diversity and magnitude of convergent versus divergent style shifting-: Perceptual and evaluative consequences. *Language and Communication* 8, 3 (1988), 213–228.

[2] Géry Casiez, Daniel Vogel, Ravin Balakrishnan, and Andy Cockburn. 2008. The Impact of Control-Display Gain on User Performance in Pointing Tasks. *Human-Computer Interaction* 23, 3 (2008), 215–250.

[3] Olivier Chapuis, Renaud Blanch, and Michel Beaudouin-Lafon. 2007. *Fitts' Law in the Wild: A Field Study of Aimed Movements.* Technical Report. https://hal.archives-ouvertes.fr/hal-00612026 LRI Technical Repport Number 1480, Univ. Paris-Sud, 11 pages.

[4] Tanya L. Chartrand and John A. Bargh. 1999. The Chameleon Effect: The Perception-Behavior Link and Social Interaction. *Journal of Personality and Social Psychology* 76, 6 (1999), 893–910.

[5] Sin-Horng Chen, Chiao-Hua Hsieh, Chen-Yu Chiang, Hsi-Chun Hsiao, Yih-Ru Wang, Yuan-Fu Liao, and Hsiu-Min Yu. 2014. Modeling of Speaking Rate Influences on Mandarin Speech Prosody and Its Application to Speaking Rate-Controlled TTS. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.* 22, 7 (July 2014), 1158–1171. https://doi.org/10.1109/TASLP.2014.2321482

[6] Fanny Chevalier, Nathalie Henry Riche, Catherine Plaisant, Amira Chalbi, and Christophe Hurter. 2016. Animations 25 Years Later: New Roles and Opportunities. In *Proceedings of the International Working Conference on Advanced Visual Interfaces* (Bari, Italy) *(AVI '16)*. Association for Computing Machinery, New York, NY, USA, 280–287. https://doi.org/10.1145/2909132.2909255


[4]One of the authors recalls owning a sports cars that frequently led to unnecessarily, undesirably, and stressfully fast driving, even when he had no intention to speed.


[7] Jacob Cohen. 1988. *Statistical power analysis for the behavioral sciences* (2nd ed.). L. Erlbaum Associates, Hillsdale, N.J.

[8] Uriel Cohen Priva, Lee Edelist, and Emily Gleason. 2017. Converging to the baseline: Corpus evidence for convergence in speech rate to interlocutor's baseline. *Journal of the Acoustical Society of America* 141, 5 (2017), 2989–2996.

[9] Thomas J Conrad and Yin Yin Wong. 2000. Computer system with graphical user interface including spring-loaded enclosures. https://patents.google.com/patent/US6061061 Patent No. US6061061, Filed Jul. 8, 1997, Issued May 9, 2000.

[10] Mihaly Csikszentmihalyi. 2014. *Flow and the foundations of positive psychology: The collected works of Mihaly Csikszentmihalyi.* Vol. 9789401790888.

[11] Oscar de Bruijn and Robert Spence. 2000. Rapid Serial Visual Presentation: A Space-Time Trade-off in Information Presentation. In *Proceedings of the Working Conference on Advanced Visual Interfaces* (Palermo, Italy) *(AVI '00)*. Association for Computing Machinery, New York, NY, USA, 189–192. https://doi.org/10.1145/345513.345309

[12] James W. Dias and Lawrence D. Rosenblum. 2011. Visual Influences on Interactive Speech Alignment. *Perception* 40, 12 (2011), 1457–1466.

[13] AJ Dix. 1992. Pace and interaction. In *Proceedings of HCI '92: People and computers VII.* 171–190.

[14] Kohji Dohsaka, Atsushi Kanemoto, Ryuichiro Higashinaka, Yasuhiro Minami, and Eisaku Maeda. 2010. User-Adaptive Coordination of Agent Communicative Behavior in Spoken Dialogue. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue* (Tokyo, Japan) *(SIGDIAL '10)*. Association for Computational Linguistics, USA, 314–321.

[15] Greg T. Elliott and Bill Tomlinson. 2006. PersonalSoundtrack: Context-Aware Playlists That Adapt to User Pace. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems* (Montréal, Québec, Canada) *(CHI EA '06)*. Association for Computing Machinery, New York, NY, USA, 736–741. https://doi.org/10.1145/1125451.1125599

[16] PM Fitts. 1954. The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement. 47 (1954), 381–391.

[17] T. Francis. 1996. *Mac OS 8 Revealed.* Addison-Wesley. https://books.google.co.nz/books?id=xELQOQAACAAJ

[18] Simone Fuscone, Benoit Favre, and Laurent Prevot. 2018. Replicating Speech Rate Convergence Experiments on the Switchboard Corpus. In *Workshop on Replicability and Reproducibility of Research Results in Science and Technology of Language.* Miyazaki, Japan. https://hal.archives-ouvertes.fr/hal-01807796

[19] Louis David Giacolone Jr. 1998. Dynamic rate control method and apparatus for electronically played games and gaming machines. https://patents.google.com/patent/US5758875 Patent No. US5758875A, Filed Jan. 11th, 1996, Issued Jun. 2nd., 1998.

[20] Howard Giles, Anthony Mulac, James J. Bradac, and Patricia Johnson. 1987. Speech Accommodation Theory: The First Decade and Beyond. *Annals of the International Communication Association* 10, 1 (1987), 13–48. https://doi.org/10.1080/23808985.1987.11678638

[21] Chris Harrison, Brian Amento, Stacey Kuznetsov, and Robert Bell. 2007. Rethinking the Progress Bar. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology* (Newport, Rhode Island, USA) *(UIST '07)*. ACM, New York, NY, USA, 115–118. https://doi.org/10.1145/1294211.1294231

[22] Scott E. Hudson and John T. Stasko. 1993. Animation Support in a User Interface Toolkit: Flexible, Robust, and Reusable Abstractions. In *Proceedings of the 6th Annual ACM Symposium on User Interface Software and Technology* (Atlanta, Georgia, USA) *(UIST '93)*. Association for Computing Machinery, New York, NY, USA, 57–67. https://doi.org/10.1145/168642.168648

[23] Molly E. Ireland, Richard B. Slatcher, Paul W. Eastwick, Lauren E. Scissors, Eli J. Finkel, and James W. Pennebaker. 2010;2011;. Language Style Matching Predicts Relationship Initiation and Stability. *Psychological science* 22, 1 (2010;2011;), 39–44.

[24] Matt Jones, Preeti Jain, George Buchanan, and Gary Marsden. 2003. Using a mobile device to vary the pace of search. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 2795 (2003), 390–394.

[25] Wiard Jorritsma, Fokie Cnossen, and Peter M. A. van Ooijen. 2015. Adaptive support for user interface customization: a study in radiology. *International Journal of Human - Computer Studies* 77 (2015), 1–9.

[26] Melissa Jungers, Caroline Palmer, and Shari Speer. 2002. Time after time: The coordinating influence of tempo in music and speech. *Cognitive Processing* 1 (01 2002), 21–35.

[27] Rivk Levitan, Štefan Beňuš, Agustín Gravano, and Juli Hirschberg. 2015. Entrainment and turn-taking in human-human dialogue, Vol. SS-15-07. 44–51.

[28] Rivka Levitan and Julia Hirschberg. 2011. Measuring acoustic-prosodic entrainment with respect to multiple levels and dimensions. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH 2011.* 3081–3084. http://www.cs.columbia.edu/~rlevitan/papers/IS11full_paper.pdf

[29] Natalie Lewandowski and Matthias Jilka. 2019. Phonetic Convergence, Language Talent, Personality and Attention. *Frontiers in Communication* 4 (2019), 18. https://doi.org/10.3389/fcomm.2019.00018

[30] Nichola Lubold. 2017. Building Rapport through Dynamic Models of Acoustic-Prosodic Entrainment. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI EA '17)*. Association for Computing Machinery, New York, NY, USA, 297–300. https://doi.org/10.1145/3027063.3027132

[31] Wendy E. Mackay. 1991. Triggers and Barriers to Customizing Software. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New Orleans, Louisiana, USA) *(CHI '91)*. Association for Computing Machinery, New York, NY, USA, 153–160. https://doi.org/10.1145/108844.108867

[32] Sylvain Malacria, Alix Goguey, Gilles Bailly, and Géry Casiez. 2016. Multi-Touch Trackpads in the Wild. In *Actes de La 28ième Conference Francophone Sur l'Interaction Homme-Machine* (Fribourg, Switzerland) *(IHM '16)*. Association for Computing Machinery, New York, NY, USA, 19–24. https://doi.org/10.1145/3004107.3004113

[33] Joseph H. Manson, Gregory A. Bryant, Matthew M. Gervais, and Michelle A. Kline. 2013. Convergence of speech rate in conversation predicts cooperation. *Evolution and Human Behavior* 34, 6 (2013), 419–426.

[34] Mudit Misra, Elena árquez Segura, and Ahmed Sabbir Arif. 2019. Exploring the Pace of an Endless Runner Game in Stationary and Mobile Settings. In *Extended Abstracts of the Annual Symposium on Computer-Human Interaction in Play Companion Extended Abstracts* (Barcelona, Spain) *(CHI PLAY '19 Extended Abstracts)*. Association for Computing Machinery, New York, NY, USA, 543–550. https://doi.org/10.1145/3341215.3356256

[35] J Nielsen. 1993. *Usability Engineering*. London: Academic Press.

[36] William Odom, Richard Banks, Abigail Durrant, David Kirk, and James Pierce. 2012. Slow Technology: Critical Reflection and Future Directions. In *Proceedings of the Designing Interactive Systems Conference* (Newcastle Upon Tyne, United Kingdom) *(DIS '12)*. Association for Computing Machinery, New York, NY, USA, 816–817. https://doi.org/10.1145/2317956.2318088

[37] Sharon Oviatt, Courtney Darves, and Rachel Coulston. 2004. Toward Adaptive Conversational Interfaces: Modeling Speech Convergence with Animated Personas. *ACM Trans. Comput.-Hum. Interact.* 11, 3 (Sept. 2004), 300–328. https://doi.org/10.1145/1017494.1017498

[38] Jennifer S. Pardo, Jennifer S. Pardo, Adelya Urmanche, Adelya Urmanche, Sherilyn Wilman, Sherilyn Wilman, Jaclyn Wiener, and Jaclyn Wiener. 2017. Phonetic convergence across multiple measures and model talkers. *Attention, Perception, & Psychophysics* 79, 2 (2017), 637–659.

[39] Martin J. Pickering and Simon Garrod. 2004. Toward a mechanistic psychology of dialogue. *The Behavioral and brain sciences* 27, 2 (2004), 169–190.

[40] Philip Quinn, Sylvain Malacria, and Andy Cockburn. 2013. Touch Scrolling Transfer Functions. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology* (St. Andrews, Scotland, United Kingdom) *(UIST '13)*. Association for Computing Machinery, New York, NY, USA, 61–70. https://doi.org/10.1145/2501988.2501995

[41] Natasha D. Schüll. 2012. *Addiction by design: machine gambling in Las Vegas*. Princeton University Press, Princeton.

[42] Benjamin G. Schultz, Irena O'Brien, Natalie Phillips, David H. McFarland, Debra Titone, and Caroline Palmer. 2016. Speech rates converge in scripted turn-taking conversations. *Applied Psycholinguistics* 37, 5 (2016), 1201–1220.

[43] Ben Shneiderman. 1984. Response time and display rate in human performance with computers. *ACM Computing Surveys (CSUR)* 16, 3 (1984), 265–285.

[44] Jr. Street, Richard L. 2006. Speech Convergence and Speech Evaluation in Fact-Finding Interviews. *Human Communication Research* 11, 2 (03 2006), 139–169. https://doi.org/10.1111/j.1468-2958.1984.tb00043.x arXiv:https://academic.oup.com/hcr/article-pdf/11/2/139/22342852/jhumcom0139.pdf

[45] Paul J. Taylor and Sally Thomas. 2008. Linguistic Style Matching and Negotiation Outcome. *Negotiation and conflict management research* 1, 3 (2008), 263–281.

[46] Rick B. van Baaren, Rob W. Holland, Bregje Steenaert, and Ad van Knippenberg. 2003. Mimicry for money: Behavioral consequences of imitation. *Journal of experimental social psychology* 39, 4 (2003), 393–398.

[47] Kent Wittenburg, Clifton Forlines, Tom Lanning, Alan Esenther, Shigeo Harada, and Taizo Miyachi. 2003. Rapid Serial Visual Presentation Techniques for Consumer Digital Video Devices. In *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology* (Vancouver, Canada) *(UIST '03)*. Association for Computing Machinery, New York, NY, USA, 115–124. https://doi.org/10.1145/964696.964709

[48] Jacob O. Wobbrock, Edward Cutrell, Susumu Harada, and I. Scott MacKenzie. 2008. An error model for pointing based on Fitts' law. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems* (Florence, Italy). ACM, New York, NY, USA, 1613–1622. https://doi.org/10.1145/1357054.1357306