Citrine: Providing Intelligent Copy-and-Paste

Jeffrey Stylos, Brad A. Myers, Andrew Faulring
Computer Science Department and Human-Computer Interaction Institute
Carnegie Mellon University
Pittsburgh, PA 15213, USA
{ jsstylos, bam, faulring }@cs.cmu.edu
http://www.cs.cmu.edu/~citrine/

ABSTRACT
We present Citrine, a system that extends the widespread copy-and-paste interaction technique with intelligent transformations, making it useful in more situations. Citrine uses text parsing to find the structure in copied text to allow users to paste the structured information, which might have many pieces, with a single paste operation. For example, using Citrine, a user can copy the text of a meeting request and add it to the Outlook calendar with a single paste. In applications such as Excel, users can teach Citrine by example how to copy and paste data by showing it which fields go into which columns, and can use this to copy or paste many items at a time in a user-defined manner. Citrine can be used with a wide variety of applications and types of data, and can be easily extended to work with more. It currently includes parsers that recognize contact information, calendar appointments and bibliographic citations. It works with Internet Explorer, Outlook, Excel, Palm Desktop, EndNote and other applications. Citrine is available to download on the internet, and a user study has shown that Citrine is well-liked and allows users to perform common copy-and-paste tasks faster than previously possible.

Author Keywords
Copy-and-paste, web-pasting, intelligent user interfaces.

ACM Classification Keywords
H.5.2. User Interfaces: Interaction Styles.

INTRODUCTION
A common and tedious task encountered while using a computer is that of taking data from one source—an email, webpage, or program—and entering it somewhere else—into a web-based form, contact manager, bibliographic database, or other program. The technique of copying and pasting is useful and easy to use, but is limited, especially when dealing with data that has multiple parts. For example, entering contact information from an email signature into a contact manager might require copying (or manually retyping) a dozen different fields, and require the user to switch between the source and destination window with each copy and paste. Similarly, filling in a web-based form might require a separate copy and paste for each field.

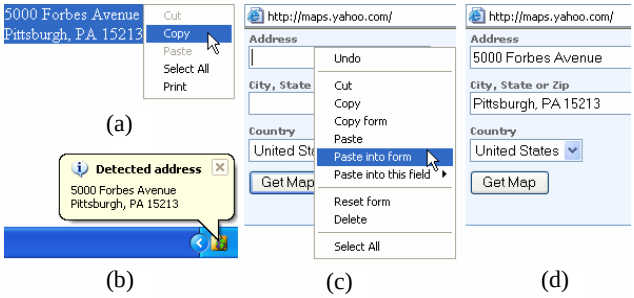

Figure 1: Using Citrine to paste an address

Our approach is to extend the copy-and-paste interaction technique to work better for structured text. This is enabled using a centralized text-parsing and format conversion framework that provides for multiple pluggable text parsers and format converters.

The framework is embodied in Citrine, a new tool that intelligently infers the structure of copied text, adds new representations of the data to the clipboard, and modifies existing applications when necessary to support pasting the new structured data representations. We also modify existing applications to be able to copy structured data in cases where the applications do not support this (for example, to copy multiple fields from a web-form).

Figure 1 shows an example of using Citrine to paste an address. In part (a), the text:

5000 Forbes Avenue
Pittsburgh, PA 15213

is copied from a webpage, causing Citrine to recognize that an address has been copied (b). A user can then paste this into a web-based form such as on Yahoo! Maps that has several fields with a single paste operation (c), and all the fields will be filled in (d).

Citrine works with many different types of data and applications. It recognizes contact information and creates standard contact clipboard formats that can be pasted into Outlook, Palm Desktop, Netscape address book, CardScan address book and many other applications. It recognizes paper citations, and creates a format compatible with the EndNote reference manager. It recognizes calendar appointments, which, with the help of application plug-ins, can be pasted into Outlook and Palm Desktop. All of the above data types can be pasted into Internet Explorer or Excel, where plug-ins let the user define by example how the fields should be mapped to the different fields and columns. After being trained once, Citrine remembers these associations and uses them to paste subsequent items.

Citrine is extensible in a number of ways. First, its parsing can be extended so that it can recognize new types of data. Second, its clipboard-format generation can be extended, so that it can easily provide compatibility without modifications to new applications. Third, it can be extended with new application plug-ins that can easily use Citrine's custom clipboard data format. And last, though no current parser implements it, Citrine is designed to learn from user behavior with text parsers that improve their performance through user corrections [2].

Because the structure of copied data may be ambiguous, Citrine must guess what each section of text represents, and occasionally makes mistakes. Citrine provides various ways for users to verify that the structured information created by Citrine is correct. A user study we conducted showed that (a) users verify and correct the data more quickly than they could enter it without Citrine; (b) users tend to catch mistakes that Citrine makes; (c) users may make fewer errors when using Citrine than when not using it; and (d) users greatly prefer to use our system. A separate comparison showed that Citrine recognized contact information at least as well as AddressGrabber [3], a commercial tool designed for recognizing contact information. In addition, Citrine has proved useful in its daily use by its developers, where it is most commonly use it for are pasting addresses into web-forms and pasting events into the Outlook or Palm Desktop calendar. It is available to download at our web site: http://www.cs.cmu.edu/~citrine/.

Citrine differs from existing techniques in three important ways. First, Citrine operates *between* regular copies and pastes. So, unlike SmartTags, which require modified source applications, or "paste special," which requires modified destination applications, Citrine works transparently in between the source and destination. This allows it to more easily support compatibility with a broad range of unmodified applications, and also does not require the user to learn a new user interface. Second, Citrine works with multiple fields of data at the same time, even when the fields are unlabeled—that is, it can detect a multi-field structure from a single copy, and paste the various pieces with a single paste operation. The sources do not need to be labeled since Citrine uses text parsing. Finally, Citrine provides customized pasting by allowing users to teach it how to paste into web-forms and spreadsheets.

RELATED WORK

Techniques related to those used in Citrine include automatically synthesizing clipboard formats, intelligent pasting, copying multiple items to the clipboard, parsing and recognizing information from plain text, and pasting into web-forms.

Clipboard Format Synthesis

Other programs have monitored the clipboard for changes and automatically modified the clipboard when appropriate. For example, when plain text is copied to the clipboard in Microsoft Windows, the operating system synthesizes a Unicode format for the text and adds it to the clipboard. Copying non-English language text from one program to another often causes seemingly random text to appear in the destination application due to incompatible character sets. The Magic Clipboard addresses this problem by automatically synthesizing text on the clipboard with different character sets [12].

Citrine differs from these by converting not just from one encoding to another, but by synthesizing formats of a higher level of abstraction than the plain text already there.

Intelligent Pasting

Numerous programs support a "Paste Special" command that allows users to choose a format for pasted information. For example, when copying html-formatted text, a user has the option of pasting the information in plain text or html format. Microsoft Office programs also present a small icon that, when clicked, brings up a menu that allows users to change their mind after pasting.

The Silver multimedia editing tool provides an "Intelligent Cut and Paste" for copying audio and video within Silver [9]. Silver attempts to find boundaries between scenes and determine when the audio for one scene overlaps the video for the previous scene. If the user copies a segment where the audio starts ahead of the video, Silver tries to intelligently avoid a gap when pasting by going back to the source video and pasting extra video.

Clipboard Rings

Traditional computer clipboards allow only one piece of data to be on the clipboard at a time. Placing new data on the clipboard destroys the old data. Some programs such as Microsoft Office implement a clipboard ring that holds multiple items, so users can cycle through them to select which item will be pasted.

Text Parsing and Recognition

Several existing systems [7, 10, 11] attempt to recognize portions of structured text from largely unstructured spans of text using hard-coded rules.

Programs such as the Apple Data Detectors [10], the Selection Recognition Agent [11] and, Microsoft Office XP Smart Tags [7] parse text at the source. After identifying particular structures, for instance an email address, a popup menu with relevant commands, like "send an email," becomes available. Unlike with Citrine, this technique typically requires that the actions to be encoded at the source location, though some applications, such as the Selection Recognition Agent, works with the clipboard as an auxiliary method of accessing text.

Several commercial tools parse text to find structured information. AddressGrabber parses text looking for contact information that can be added to a variety of contact list programs [3]. CardScan is a business card scanner that scans business cards, performs optical character recognition on the image, parses the text for fields, and enters the contact information into its address book [1]. Like Citrine, CardScan presents a forms-based interface for correcting errors. One of the authors has scanned over eighty business cards with CardScan, but has never seen a card scanned without error. Even so, correcting the errors is much less tedious than doing the entire process by hand. Some previous systems have provided parsing of bibliographic references from plain text [5, 6, 8].

Citrine takes the generic approach of recognizing different types of structure while still being able to recognize complex structures such as contacts. Citrine provides its text recognition centrally via the clipboard and avoids having to pre-determine the desired action at the source location.

Web Forms

Several tools help users automatically fill in web forms. Web browsers such as Internet Explorer and Mozilla remember the text that a user entered into a form on a web page. When the user starts typing in a field similar to one that has previously been filled, Internet Explorer's autocomplete feature displays suggested completions for the field using previously entered values. Mozilla's Form Manger and the Autofill feature of the Google Toolbar for Internet Explorer also make it easy for users to fill in forms. Users initially supply values for common fields such as their name, address, and credit card number. After they have supplied this data, a single command fills in text for all recognized fields.

Citrine extends this technique by filling in web-forms with dynamic information from the clipboard, rather than static information from a saved database, and uses text parsing to automatically determine which pieces of plain text should go in each field. Citrine can also be trained by example to specify the mappings.

DESCRIPTION OF CITRINE

This section presents Citrine: "**C**lipboard **I**nteraction **T**echniques that **R**ecognize **I**nformation such as **N**ames and **E**vents."

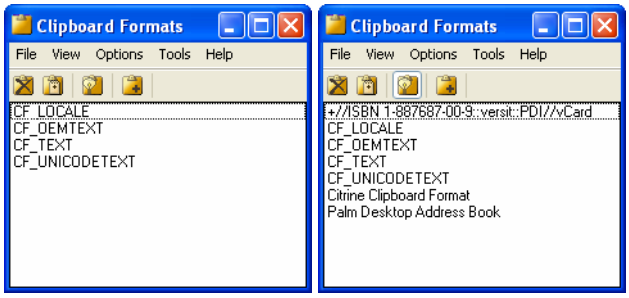

Figure 2: Clipboard formats before and after Citrine recognizes contact information

User Interface

Wherever possible, Citrine uses the applications' own unmodified copy-and-paste user interfaces and is therefore transparent to the user. When applications must be extended with plug-ins, we try to make the added functionality match the existing user interfaces for copying and pasting, as in Figure 1c.

Clipboard Formats

The clipboard in Microsoft Windows (and other operating systems) supports multiple representations (formats) of the same data. A key feature of Citrine is that it adds formats to the clipboard. Citrine watches the clipboard for changes. When new text is copied, Citrine tries to recognize the content, and if so, it adds new formats to the clipboard. It does this without disturbing the formats that were already on the clipboard. Figure 2 shows our debugging application displaying the list of formats on the clipboard before and after Citrine recognizes contact information.

The new formats on the clipboard cause applications that recognize these formats to enable their paste functions, which otherwise would be disabled when the clipboard only contains plain text. For example, copying text containing a bibliographic citation while Citrine is running enables the EndNote bibliographic manager's paste command. When copying plain text without Citrine running, the paste command is disabled. After Citrine adds formats to the clipboard, applications that previously allowed the pasting of the plain text still work as before. Applications that previously allowed pasting of plain text but *also* allow pasting of contacts (such as Outlook's contact manager, which pastes plain text by creating an empty new contact with the clipboard contents in the "notes" field) will now paste the preferred contact format instead.

Citrine can also convert between existing structured data formats. Unlike conversions of plain text, these conversions can be done unambiguously. This is useful for providing clipboard compatibility between applications. For example, by automatically generating additional contact formats, Citrine enables copying and pasting between the Netscape and Outlook address books, which otherwise is not supported. The same type of conversions also enables copying and pasting appointments between the Palm Desktop and Outlook calendars.

Application Plug-Ins

Many applications natively support the copying and pasting of some clipboard format and work with Citrine without modification. These applications include the EndNote bibliographic manager, the Outlook, Palm Desktop, Netscape and CardScan address books, and many others.

However, some applications do not support any copying and pasting of particular data at all. For these applications, we needed to create plug-ins that extend the applications to allow users to copy and paste. For example, the Palm Desktop calendar does not support the copying or pasting of appointments at all without the Citrine plug-in. The plug-in adds new "Copy appointment" and "Paste appointment" buttons to Palm Desktop.

We created a special "Paste appointment" function in Outlook's calendar as well. Outlook natively supports pasting appointments only as plain text into the currently selected date and time. Our added paste function creates a new appointment window with the subject, location, and date and time fields filled in using the information from the copied text. Showing the new appointment window allows the user to verify that the detected fields are correct and to modify any fields before saving an appointment.

Web-Pasting

We also extended Internet Explorer to support pasting into multiple fields of a web-form at once. In this case, there is an existing paste function available, but it only pastes into one field at a time, and pastes only the first line of text when pasting into regular edit fields.

We created an additional function, "Paste into this form," that fills in multiple form fields at once. For example, Figure 1 showed how a copied address can be pasted into Yahoo! Maps with one action. The plug-in fills in the fields by matching the names of the form fields with the fields of the parsed data in the clipboard. For example, it matches the field from Yahoo! Maps' form that has the internal name of "addr" with the street-address field of a parsed address. Some fields need to be filled in with multiple items. Citrine maps the "City, State or Zip" field in the Yahoo! Maps page to the city, state and zip-code fields of a recognized address and combines the fields appropriately with spaces and a comma. Furthermore, Citrine can fill in fields that use pop-up menus, such as the state field on many forms. The plug-in is not specific to pasting addresses, and can paste any structures that Citrine's parsers recognize.

Citrine has a default set of mappings of form fields that are designed to work with many common websites. Users can teach Citrine about new websites by demonstrating which piece of the source data goes in which form field. This uses an added function, "Paste into this field," which lets users explicitly paste a single piece of the source data into one field (see Figure 3). After doing this once, the "Paste into this form" function will continue to use the demonstrated mappings to fill in the fields when pasting new data into the form.

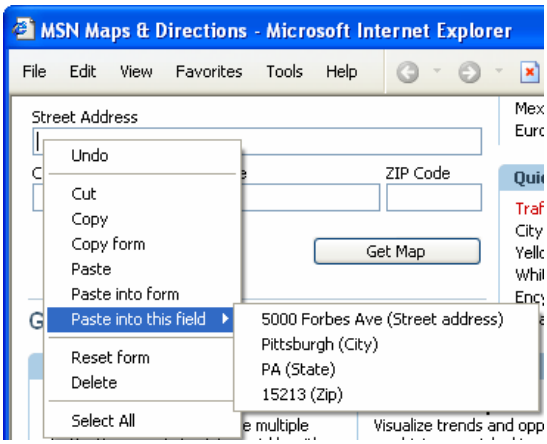

Figure 3: Menu items added to Internet Explorer. The "Paste into form" command automatically fills in fields of the form it recognizes. The items in the "Paste into this field" submenu show the currently recognized fields on the clipboard.

We also added a "Copy form" function for copying from multiple fields in a web-form. This can be used for copying between web-forms or in combination with the structured format generation to paste into a contact manager or other destination. For example, a user can copy the data from a filled-in web-form and paste it into an Excel spreadsheet.

Excel

Citrine includes a plug-in for Excel that lets users copy and paste objects with different fields from and into spreadsheet cells. The plug-in lets users define mappings between object fields and spreadsheet columns by demonstration. Users can then use these mappings to easily paste new items (or multiple new items at the same time) or to copy rows as contacts, for example, to paste into other applications.

Users can teach Citrine the Excel associations in much the same way as with the Internet Explorer plug-in, by pasting individual fields into cells one at a time with the "Paste into cell" submenu, and then reusing these associations to paste whole items into a row with one "Paste into row" menu action.

The plug-in also enables the pasting of *multiple* clipboard items at the same time, into consecutive rows. For example, a user can select many different contacts in the Palm Desktop address book (by holding down CTRL while selecting new items, for example) and then paste all of them into an Excel spreadsheet with one "Paste all items into rows" menu action (Figure 4).

Figure 4 shows how, after pasting each field of one contact, one or more rows can be pasted with one action.

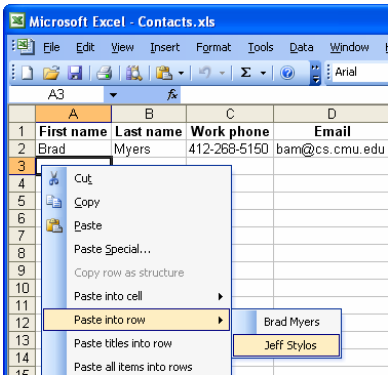

Figure 4: The Excel Citrine plug-in can be used to paste structures with multiple fields into a spreadsheet row with a single action. The column mappings are taught by demonstration. "Paste into row" pastes the selected item, and "Paste all items into rows" pastes both contacts into rows.

Miscellaneous Tools

While investigating the copy-and-paste interaction technique and creating Citrine, the authors also created some related tools.

The copy-append feature (built into the main Citrine application), lets users copy text that is not contiguous or cannot be selected at once, by appending copied text to the clipboard, rather than replacing it. This is performed by holding down the Windows-key and pressing C. The ability to copy text that is not contiguous is especially useful for combining related data that Citrine will recognize as a contact, appointment or other structure.

We also created a plug-in for Internet Explorer that replaces the existing paste function to allow users to paste multiple-line URLs into the IE address bar. This helps alleviate the inconvenience of dealing with email programs and websites that add newlines to long URLs.

The Citrine Dialer, available on the Citrine website, makes it easy to copy and dial phone numbers from the computer. The dialer, whose novelty comes from automatically learning how to dial different types of numbers from example, also takes advantage of Citrine's contact parsing and clipboard format to extract the phone numbers from copied contacts. Figure 5 shows the Citrine Dialer's option menu for dialing any of the phone numbers, identified by label, in a contact recognized by Citrine. The phone number has automatically been transformed to only use its last five digits based on previous dialing examples.

Figure 5: The Citrine Dialer application can dial the phone numbers extracted by Citrine

System-Tray Icon and Popup Notifications

Because Citrine is often transparent—using existing applications' copy and paste functions—we optionally provide some non-intrusive feedback to notify the user that something has taken place and that now they can paste into a wider range of applications.

Citrine does this in two ways. First, Citrine displays an icon in the system tray that changes to represent the format of the data detected on the clipboard. Figure 6 shows icons used to represent different formats. This icon also provides access to a menu that can enable or disable Citrine and can clear the contents of the clipboard.

Figure 6: Icons used by Citrine to represent the format it has detected on the clipboard. From left to right: an empty clipboard, text, an image, a citation, an address, a contact, an appointment, and other data.

Second, Citrine optionally creates a balloon popup notification near the system tray when information is recognized and formats are generated from plain text (see Figure 7). This popup lets the user know when information is parsed and provides a summary of what is recognized. We initially intended the popup mainly for new users, but found it unobtrusive and useful enough for experienced users as well.

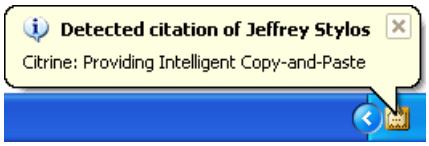

Figure 7: A popup notification showing that Citrine has detected a bibliographic citation.

Clipboard Viewer and Editor

Citrine includes a tool for viewing and editing the clipboard text and recognized fields (see Figure 8). This is a way for users to enter new information or correct parser mistakes. We plan to use this interface in the future as a way for user's to provide feedback that could improve our parsers when machine learning techniques are added (see future work section).

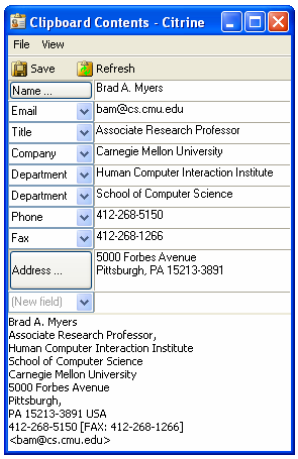

Figure 8: The clipboard editor

IMPLEMENTATION

Citrine consists of a main application that runs in the background and of a collection of application plug-ins.

Main Application

The main Citrine application watches the clipboard to detect copy-events, parses the available clipboard formats, and uses this information to generate new clipboard formats. There are two main types of components in this architecture, parsers and format generators. The Citrine-specific clipboard format is the sole method of communication between these two types of components. The components are separate from each other and new ones of each can be easily added.

Parsing

There are two types of parsers, structured-format based parsers, and heuristic plain-text parsers. Each of these parsers performs two tasks: deciding if the clipboard contains relevant information, and if so, extracting and labeling it and adding it to the Citrine-specific clipboard format.

The structured-format based parsers unambiguously translate from structured clipboard formats to the Citrine-specific format. For example, copying a contact from Palm Desktop causes Citrine to extract each of the fields and saves the data in the Citrine-specific clipboard format. This is useful for pasting with Citrine plug-ins and also to enable the conversion from one structured format to another. The structured-format based parsers were easy to create for documented clipboard formats but more difficult to create for private, undocumented formats (as was often the case). For undocumented formats, we performed much experimentation to reverse engineer the formats used. The structured-format based parsers decide if the clipboard contains relevant information by checking to see if their targeted format is on the clipboard.

Much of Citrine's power comes from the heuristic text based parsers. When text is copied, each heuristic parser decides if the copied text contains relevant information. If so, it extracts and labels all of the fields of the structure it can and adds them to the Citrine-specific clipboard format. When text contains multiple types of data—such as an event notice that includes contact information—both parsers decide that the text contains relevant information, both add their structures to Citrine's clipboard format, and both will have format generators create additional clipboard formats. Because the clipboard can hold many different formats at the same time, this is not a conflict.

Citrine currently includes plain text parsers that use ad-hoc, but well tuned, parsing techniques, rather than the more advanced techniques from the fields of information extraction or text classification [2]. The current parsers look for individual fields of structured information by checking for patterns, such as the number pattern of a phone number, and keywords, such as "Fax," "Researcher," or "University." The parsers also look for some fields based on the location of text. For example, the contact parser looks in the beginning of copied text for a name. To decide whether text contains relevant information, the parsers first attempt to parse it and then check to see whether it contains a sufficient set of fields.

While the techniques in the contact and event parsers are relatively simple, they have been tuned to achieve good results and are useful for day-to-day use. The citation parser has not yet been tuned as much; we plan to use a more sophisticated algorithm as in [6].

The Citrine architecture allows new parsers to be added with little additional work. The programming interface also can provide feedback to the parsers so that parsers that support learning can be told about user corrections. These corrections are made by the dialog shown in Figure 8. After saving changes, the dialog sends the new structure and original text back to the parsers. Though our current hand-coded parsers cannot use this additional information, we plan to implement more sophisticated parsers that can [2].

The structured-format parsers run before the heuristic parsers, so that when both structured and plain text formats are on the clipboard, the unambiguous, structured parsers are used instead.

Citrine-Specific Clipboard Format

The Citrine-specific clipboard format, "Citrine Clipboard Format" shown in Figure 2, serves as the means of communication between the parsers and format generators and is also created and used by the application plug-ins.

The format is generic and easily parsed by plug-ins. It consists of a list of structures labeled by type (such as contact or citation). Each structure is a list of name-value pairs of null-terminated strings. The format can accommodate multiple types of structures and multiple instances of structures of any given type at the same time.

Format Generators

The format generators take the labeled information from the Citrine-specific clipboard format and, if it contains a relevant structure, use it to generate other structured clipboard formats. This is much the same as the task of the structured-format parsers, but in reverse.

The following example shows how Citrine's architecture works. When plain text containing a contact is copied, Windows places the plain text on the clipboard, along with a few other formats (Figure 1a). Each of the different parsers then decides if the clipboard contains an applicable structure. The structured format parsers check first, but since only plain text is available, they do not doing anything. The heuristic parsers check next. The citation and event parsers decide that the text does not contain a relevant structure. The contact parser decides finds relevant structure and then extracts all of the fields of the contact

information, uses them to create a new Citrine-specific clipboard format item and adds it to the clipboard. The format generators then each look at the Citrine-specific format and decide if they can convert it. The contact related ones will decide they can, and will separately create new VCard, Palm Desktop Address Book, and CF_HDROP (for Outlook) clipboard formats and add them to the clipboard (Figure 1b). Detecting compatible formats on the clipboard, various destination applications will then enable their paste actions.

Application Plug-Ins

We created plug-ins for the Palm Desktop, Outlook, Internet Explorer and Excel. The plug-ins communicate with each other and the main Citrine application only through the Citrine-specific clipboard format on the clipboard.

Event Plug-ins

The Outlook and Palm Desktop event plug-ins are written in C# and use their applications' plug-in APIs to automate the copying and pasting of events. To copy an event, the plug-ins use the APIs to query the selected event and get its different fields, and then use this data to create the Citrine-specific format and add it to the clipboard. To paste an event, the plug-ins read the Citrine-specific format from the clipboard and then use the APIs to create a new event, sets the event's fields to those of the clipboard item and display a window for user-confirmation of the event.

Internet Explorer Plug-in

The Internet Explorer plug-in is written in C++ and uses Windows' subclassing to add additional items to the right click context menu (Figure 3). The Browser Helper Object interface provided by Internet Explorer also allows plug-ins to add right-click menu items, but it does not allow their modification during runtime. Because we needed to enable and disable our menu items based on the clipboard contents and to dynamically add new menu items in the "Paste into this field" submenu, we had to use the lower-level technique of subclassing in Windows to replace the function that creates Internet Explorer's menus with our own.

The associations between form fields and object fields that the plug-in uses is made by the internal HTML name of the form field and by the "name" field of the name-value pair element in the Citrine-specific clipboard format. We considered using additional data such as the name of the form, the URL of the page, and nearby text labels to associate form fields with clipboard elements. But our experience with using HTML field names has shown that different forms rarely use the same field name for conflicting purposes. However, the other types of information could potentially be useful in automatically learning field associations. Currently, the associations are learned by demonstration. (Citrine is installed with associations for many common websites and so that many forms will work without explicit end-user demonstration.)

The associations are learned when the user uses the plug-in to paste a field from the clipboard into a form field using the "Paste into this field" menu item (Figure 3). After this, the name of the field and the name of the clipboard element are saved. This action optionally will save the association to the Windows registry so it will be available for "Paste into this form" for all future times the forms with that field name are used.

Excel Plug-in

The Excel plug-in is written in C# and uses Excel's plug-in API to add new items to the right-click menu (Figure 4). Pasting new clipboard items is implemented by associating elements of the clipboard items with column numbers of the spreadsheet. As in the Internet Explorer plug-in, the associations are learned when the user pastes individual fields by using the added "Paste into this cell" submenu.

In addition to column numbers, we considered using titles from title row to make the associations. This way, associations for existing spreadsheets could be learned without any demonstration (for example, a column title of "Name" could be easily matched to the "name" field of the clipboard item). However, difficulty in reliably obtaining the title row (which is often not the first row) caused us to use column numbers, though we plan to add title-finding and title-based associations in the future.

The same column-associations that are used to paste items are also used to copy rows as structured items when our new "Copy row as structure" menu item is used. (The plug-in does not use the regular "copy" action to do this because we were not able to add code to modify the regular copy action in Excel.) The plug-in creates a new Citrine-specific clipboard format item with a name-value pair for each column with a valid association. The name in this pair is the clipboard-field-name in the Excel plug-in's association list and the value is the contents of that column in the currently selected row. When this is added to the clipboard, the main Citrine application recognizes that the clipboard's contents have changed and its format generators add additional formats to the clipboard based on the Citrine-specific format added by the Excel plug-in.

USER STUDY

We ran an experimental study of users' performance with Citrine compared to using the applications' standard copy-and-paste functionality. The study focused on creating contact entries and calendar appointments in Microsoft Outlook, using Outlook's integrated email, contact list, and appointment calendar components.

Our study tested two hypotheses:

Hypothesis 1: Users will perform copy-and-paste tasks faster when using Citrine as compared to standard copy-and-paste.

Hypothesis 2: Despite the inference errors that Citrine makes, users will notice and correct enough errors such that they make no more errors with Citrine than they would with standard copy-and-paste.

Participants

Twelve college students (six female, six male), aged 18 to 23 years old, participated in our study. Participants rated themselves as average to expert computer users with 10 of the 12 participants working on a computer at least 25 hours per week. 10 of the 12 participants used Microsoft Windows as their primary operating system; one used the Apple MacOS, and the other Linux. The study took about an hour, and we paid participants for their time.

Tasks

The study consisted of two tasks in which participants created contact entries and appointments from information contained in email messages.

Contacts

In the Contacts task, participants processed email messages containing only "signatures," which listed contact information of the sender such as an address and phone numbers. We asked participants to create entries in the Outlook contact list from the information contained in the signatures.

In Outlook, contacts are edited using a forms-based interface displayed in a separate window. The window contains several tabs, and each tab contains many text fields, such as "Full name" and "E-mail." Our study only required the participant to edit fields in the first tab labeled "General." The relevant fields for this task include: Full Name, Job title, Company, Address (Street, City, State, and Postal code), Business (phone), Business Fax, Home (phone), Mobile (phone), E-mail, and Web page address. The number of fields that could be filled in from a given signature varied from 9 to 11.

Ten template signatures were created by anonymizing real emails sent to the authors. Two of the templates were used for training and eight for testing. For the training templates, Citrine correctly extracted all the fields for one and failed to recognize the company field for the other. For the testing templates, Citrine correctly extracted all the fields for 6 of the 8 templates. Of the remaining 2 templates Citrine failed to recognize the company name for both and swapped the fax and business phone numbers for one. From the 8 testing templates, we created two sets of eight signatures. By using templates, we ensured that the two signature sets matched in terms of the formatting and fields present, and we counterbalanced the matching of sets with the tested conditions.

We arranged the screen as follows: the inbox filled the left two-thirds of the screen, and displayed the message headers listed at the top and the contents of the selected message below. The contact list window filled the right third.

Each condition of the task started with eight unread emails in the inbox. After selecting a message, the signature text appeared, and the participant could copy the message text to the clipboard. Using the paste command, the participant created a new contact from the text that had been copied. In the standard copy-and-paste condition, the paste command created a new contact window with the copied text in the notes field. Participants had to copy the information from the notes to the appropriate form fields. In the Citrine condition, the paste command displayed a new contact window with the copied text in the notes field and most of the form fields already filled in. Participants had to verify the accuracy of answers and make any necessary corrections.

Appointments

The Appointments task was very similar to the Contacts task. In the Appointment task participants received email messages about events such meetings or seminars. They created appointments in the calendar for those events. As with contacts, Outlook provides a forms-based interface to edit the details for each appointment.

We asked participants to create an appointment in their calendar from each email message. The relevant fields for this task included: Subject, Location, Start time, and End time. The number of fields that could be filled in from a given email message varied from two to four. Each appointment had a subject and start time. Some appointments, however, did not have a location or an end time.

The subject field for appointments appears in the calendar view. Unlike other fields, there is no "correct" subject; people's choices are often idiosyncratic. Automatic text summarization is a known to be a difficult problem. Our experience with Citrine showed that it did not do a very good job creating appointment subjects. For the study we chose to not score the subject field when calculating field- and item-errors. To keep the times of the task realistic, we still asked participants to fill in the subject field. Note that Citrine does still make an effort to provide a subject field, so participants had something to start with when entering text.

Similar to the Contacts task, ten templates (two for training and eight for testing) were created from announcement e-mails sent to the authors and from event descriptions on web pages. The templates contained both plain text and HTML. For the training templates, Citrine correctly extracted all the fields for one and generated an incorrect end time for the other. For the testing templates, Citrine correctly extracted all of the fields for 6 of the 8 template appointments. Of the remaining 2 templates, Citrine missed the end time for both (the duration was specified as "one hour") and missed the location for one. We created two testing sets from the eight testing templates.

The display was arranged in a manner similar to that of the contacts task, except that the calendar appeared on the right side of the screen in place of the contact list.

Each condition started with 8 unread emails in the inbox. The 8 emails contained the appointment information from one of the two sets. After selecting a message header, participants could copy the message text to the clipboard. In the standard copy-and-paste condition, participants navigated through the calendar and created an appointment at the specified time and date. After selecting the appointment's time range on the calendar, the paste command brought up a new appointment window with the text from the clipboard in the notes field and the start and end time set based upon the participant's selection. Participants had to fill in each text box with appropriate information from the appointment text. In the Citrine phase, participants clicked on the special "Paste Appointment" button, which automatically navigated the calendar to the appointment's date and brought up a new appointment window. The new appointment window contained the clipboard text in the notes field and had most of the text box fields already filled in. Participants had to verify the accuracy of answers and make any corrections.

Procedure and Design

Each task followed a single factor (copy-and-paste-type: standard vs. Citrine) within-subjects design. We counterbalanced the order of the standard and Citrine conditions and randomized the order of email messages. All participants completed both conditions of the contact task followed by both conditions of the appointment task. After finishing the tasks, participants completed a questionnaire.

For each task × condition pair we collected the following dependent measures: (a) *duration*, time in minutes to create all eight contacts/appointments; (b) *item-errors*, count of contacts/appointments with at least one incorrect field; (c) *field-errors*, count of incorrect fields across all contacts/appointments.

Results

We used a single factor within-subjects analysis of variance (ANOVA) to analyze all three dependent measures. Figure 9 shows the means and standard error for all three measures in both tasks.

Contacts

For the duration of Contacts we found a significant main effect of copy-and-paste-type ($F(1,11)=19.09$, $p<0.001$), with mean times of 4 minutes and 36 seconds for Citrine and 6 minutes and 11 seconds for standard copy-and-paste (25% less time). For item-errors, we did not find a significant effect ($F(1,11)=2.10$, n.s.), although the trend is towards fewer errors with Citrine. Participants averaged 0.33 item-errors using Citrine and 0.75 item-errors using standard copy-and-paste. For field-errors, we also did not find a significant effect ($F(1,11)=0.85$, n.s.). Participants averaged 0.75 field-errors using Citrine and 1.25 field-errors using standard copy-and-paste.

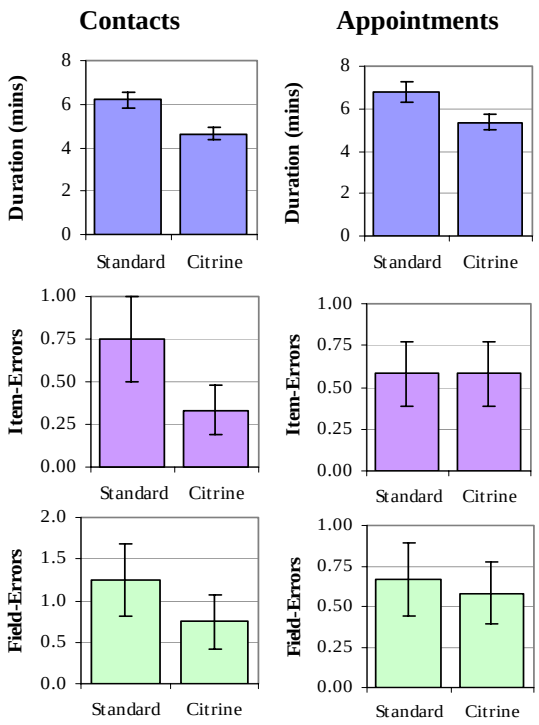

Figure 9: When using Citrine participants performed faster than when using standard copy-and-paste techniques. We found no evidence of a speed vs. accuracy tradeoff.

Appointments

For the duration of Appointments we found a significant main effect of Clipboard ($F(1,11)=7.53$, $p<0.02$), with mean times of 5 minutes and 21 seconds for Citrine and 6 minutes and 47 seconds for standard copy-and-paste (21% less time). For item-errors, we did not find a significant effect ($F(1,11)=0.00$, n.s.). Participants averaged 0.58 item-errors in both conditions. For field-errors, we also did not find a significant effect ($F(1,11)=0.07$, n.s.), although again the trend is in favor of Citrine. Participants averaged 0.58 field-errors using Citrine and 0.67 field-errors using the standard copy-and-paste.

Questionnaire

Participants completed a questionnaire about their experience with Citrine. They responded on a 5-point Likert scale.

When asked to rate Citrine's usefulness (1="Not useful", 5="Very useful"), participant responses ranged from 3 to 5 with a median of 5. We asked participants how they found the process of finding and correcting any errors that Citrine made (1="Not really a problem," 3="Slightly annoying," 5="Frustrating"). Responses ranged from 1 to 3 with a median of 1. We also asked participants if they would prefer for smart clipboard to guess less (1="fewer fields filled in, but no errors") or guess more (5="lots of fields filled in, but more errors"). Responses ranged from 1 to 5 with a median of 3. We also asked how users felt about the popup notification that occurred when Citrine recognized a con-

tact or an appointment. We asked participants to rate how "useful," "annoying" and "redundant" the notifications were. Ratings ranged from 1 to 5 for all three questions. Useful and annoying had a median of 4; redundant had a median of 3.

Discussion

Our questionnaire showed that Citrine was well-liked, that its level of inferencing seemed appropriate, and that the popup notification should be user customizable. Many participants commented that Citrine saved them much time.

Our study demonstrated that participants performed faster using Citrine, supporting Hypothesis 1. Observing the users during the study, we noticed that they spent a fair amount of time checking Citrine's answers. To obtain an upper bound on how much faster Citrine could be one of the authors completed the study without checking for errors. The author completed the tasks in one third of the time (3 times faster).

Despite Citrine's inference errors, we did not find a significant difference in errors between the two conditions. Hence, we did not find clear evidence to support Hypothesis 2. However, the data trends are in the right direction. Without correcting Citrine's incorrect inferences, participants would have made 2 item-errors in both tasks as compared to the 0.33 and 0.58 item-errors that they actually made using Citrine for the contact and appointment tasks. Uncorrected, Citrine made more errors compared to a person. However, the time that it saves can be spent checking for and correcting any errors. Overall, compared to standard copy-and-paste, Citrine allowed participants to complete the task in less time and we found no evidence of a speed vs. accuracy tradeoff.

FUTURE WORK

Citrine is being created as part of the RADAR project (http://www.radar.cs.cmu.edu/). Through the work of this project we hope to have access to many intelligent parsers (e.g., [2]) that use state of the art machine learning techniques and can be integrated with Citrine. These techniques could learn the structure of new types of data given examples by the user [4] and improve its parsing accuracy over time by observing corrections made by the user to recognized data. Such techniques could also be used to learn a user's preferences in interpreting parsed data—for example, how concise or verbose a user likes the subject field of an appointment to be.

A challenge in creating parsers of this type will be for them to learn from only a few examples. Most existing learning techniques used in parsing are statistically-based and require many examples to improve parsing performance, but we are optimistic about the new techniques being developed by our colleagues [2].

CONCLUSIONS

We have described Citrine, a new software tool that combines text parsing, clipboard-format generation and application plug-ins to extend the copy-and-paste interaction technique. It demonstrates that copy-and-paste can be extended to work better and in more situations. Citrine achieves these results mostly using real applications' existing user interfaces so that there is very little for users to learn. A user study showed that Citrine made common copy-and-paste tasks quicker and less tedious, and we have found it very useful ourselves. The software is available if you want to try it: http://www.cs.cmu.edu/~citrine/.

ACKNOWLEDGMENTS

We thank Jacob Wobbrock, Andrew Ko, Desney Tan and members of the Radar project for their comments on this work.

This material is based upon work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. NBCHC030029. Andrew Faulring is supported by a National Science Foundation (NSF) Graduate Research Fellowship.

REFERENCES

1. CardScan, http://www.cardscan.com.
2. Cohen, W.W. and Sarawagi, S., "Exploiting Dictionaries in Named Entity Extraction: Combining Semi-Markov Extraction Processes and Data Integration Methods." 2004. Submitted for Publication.
3. eGrabber, "AddressGrabber Business 3.2.1," http://www.egrabber.com/addressgrabberbusiness/index.html.
4. Lieberman, H., Nardi, B.A., and Wright, D. "Training Agents to Recognize Text by Example," in *ACM Conference on Autonomous Agents.* 1999. Seattle, WA: ACM Press. pp. 116–122.
5. Maulsby, D., Greenberg, S., and Mander, R. "Prototyping an Intelligent Agent through Wizard of Oz," in *Proceedings INTERCHI'93: Human Factors in Computing Systems.* 1993. Amsterdam, The Netherlands: pp. 277-284.
6. McCallum, A.K., Nigam, K., Rennie, J., and Seymore, K., "Automating the construction of internet portals with machine learning." *Information Retrieval Journal*, 2000. **3**: pp. 127--163.
7. Microsoft, "Microsoft Office XP Smart Tags," http://www.microsoft.com/technet/treeview/default.asp?url=/technet/prodtechnol/office/officexp/maintain/xptags.asp.
8. Myers, B.A. "Text Formatting by Demonstration," in *Proceedings SIGCHI'91: Human Factors in Computing Systems.* 1991. N.O., LA: pp. 251-256.
9. Myers, B.A., Casares, J.P., Stevens, S., Dabbish, L., Yocum, D., and Corbett, A. "A Multi-View Intelligent Editor for Digital Video Libraries," in *The First ACM+IEEE Joint Conference on Digital Libraries, JCDL'01.* 2001. Roanoke, VA: pp. 106–115. http://www.cs.cmu.edu/~silver/dl2001paper.pdf.
10. Nardi, B.A., Miller, J.R., and Wright, D.J., "Collaborative, programmable intelligent agents." *Communications of the ACM*, 1998. **41**(3): pp. 96–104. Apple Data Detectors.
11. Pandit, M.S. and Kalbag, S. "The selection recognition agent: instant access to relevant information and operations," in *International Conference on Intelligent User Interfaces.* 1997. Orlando, FL: ACM Press. pp. 47–52.
12. Smart Link Corporation, "Magic Clipboard 1.0," http://www.smartlinkcorp.com/new/Magic-Clipboard-language-foreign-SL99801-info.html.