GraphWrap: Interactive data extraction from PDF documents using graph matching techniques

Have you ever needed to extract large quantities of data such as product specifications, measurements, contact details or prices from one or more PDF files? Then let us introduce you to *GraphWrap* – a new technique for user-guided data extraction, or "wrapping", which uses graph matching techniques to locate data instances.

There are already a number of existing systems which offer similar wrapping solutions for web pages. These systems make use of the hierarchical structure inherent in the HTML format, which explicitly defines the individual data elements and how they are grouped together. In PDF, there is no such explicit structure, and data extraction from PDF is therefore a far more challenging task.

By developing a graph-based representation of a PDF and accompanying graph matching methods, we have achieved a solution to this problem. This prototype allows you to view and explore the structure of any PDF of your choice and interactively create and run wrapper programs on it.

Installation

- To use the prototype, you need to have recent versions of the Java Runtime Environment (*java.sun.com*) and Ghostscript (*www.ghostscript.com*) installed on your computer. For best results, please install JRE 6 and GNU Ghostscript 8.64 (the latter is available from *http://mirror.cs.wisc.edu/pub/mirrors/ghost/GPL/current/*)

- Extract the zip file *graphwrap-testversion-0.x.zip* to a directory of your choice.

- In the root directory of the distribution, three shell scripts (for Unix) and batch files (for Windows) are supplied. You may need to edit these scripts to point to an installed version of Java. Please also ensure that the *gui* or *gui.bat* script contains a valid path to Ghostscript. This is especially important for Windows users who have installed a version other than 8.64 or in another directory than the default.

- The prototype is now ready for use. To launch the GUI, run the *gui* or *gui.bat* script. The next section describes how you can define a wrapper interactively using the GUI.

Creating a wrapper in four simple steps

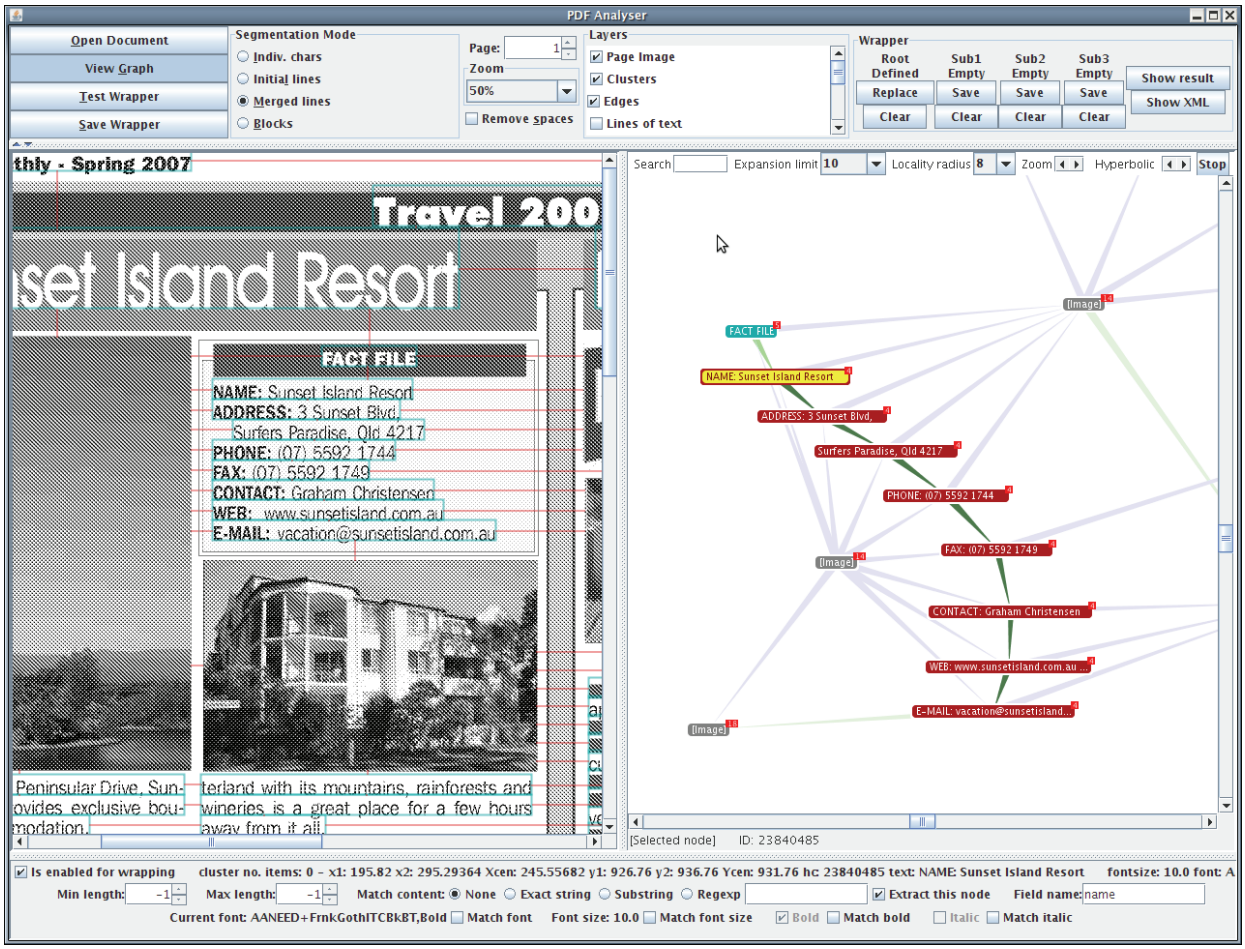
1. Open the document

- Click on "*Open document*" and select the PDF file you wish to open. In this tutorial, the sample document *travel.pdf* in the *example* subdirectory will be used.

- The document will be displayed on the screen. You will notice that the light blue rectangles correspond to each individual line of text. These are the nodes of the graph. For certain extraction tasks, a coarser granularity is more desirable.

- To select a coarser level of granularity, select "*Blocks*" instead of "*Merged lines*" in the "*Segmentation mode*" list. Then repeat Step 1 to re-open the document. Please note that changes to document processing settings become effective only when the document is reloaded.

- To show the edges of the graph, click on the *"Edges"* check-box in the *"Layers"* list. This list allows you to show or hide the other individual layers in our representation, such as images, lines and rectangles. Try it out!



2. Define the wrapper

- Click on the *"View graph"* button. The screen will now be split into two halves: the page view will now occupy the left-hand side and the interactive graph will be displayed on the right-hand side.

- Select a rectangular section of the document by clicking and dragging on the page view. The corresponding graph structure will be displayed on the right-hand side of the screen, as shown in the screenshot above. The edges of the graph represent neighbourhoods (in the four directions) between adjacent nodes. Neighbourhoods from left to right are shown with blue arrows; neighbourhoods from above to below are shown with green arrows.

- This sub-graph is the current wrapper definition. For each node and edge, you can set a wide variety of conditions by clicking on it and setting the appropriate values at the bottom of the screen.

- Other nodes in the document, which are not part of the current wrapper definition, are shown in a pale colour. These can be added to the wrapper definition at any time by right-clicking on the node and toggling the check-box *"Remove from instance"*. Current nodes can be removed from the wrapper definition in the same way.

3. Try out the wrapper

- Click on *"Test wrapper"*. The results will be shaded in purple on the page view. If you did not set any conditions in Step 2, you could receive a large number of overlapping results at this stage!

- Please note that the wrapping algorithm looks for *all possible occurrences* on the page of the sub-graph structure in the wrapper definition. By default, all nodes will be matched to any other nodes, and all edges in a particular direction to all other edges in that same direction. By adding conditions, you can restrict the matching to nodes matching a certain substring or to edges whose length is within a certain range, for example. The following example clarifies how this works.

4. Save the wrapper

- Click on *"Save wrapper"* and give the wrapper a name of your choice. Please ensure that you include the *.xml* extension in the name.

"*Travel Monthly*" example

This example is one of several hundred pages from the *Travel Monthly* newsletter, which can be freely downloaded from *www.travelmonthly.com.au*. The following steps show how database-oriented information, such as the "*fact files*", of which there are four on the page, can be extracted. Once you have created a wrapper for this example file, you can also try it out on other PDFs on the Travel Monthly website. If you have not already opened the *travel.pdf* document with a granularity of "*Merged lines*", please do so now. Then, click "*View graph*" to open the graph view.

1. Simple wrapping

- Select the "*fact file*" on the top-left of the page. To do this, click and drag a box around all the elements belonging to the fact file. Ensure that the heading "*FACT FILE*" also lies within the selection. The respective graph will be shown on the right of the screen.

- As mentioned in the steps above, this sub-graph already defines a wrapper. Click on "*Test wrapper*" to try it out. You will notice that you will receive several overlapping results. This is because the wrapper, at this stage, is looking for all combinations of vertically neighbouring lines of text, regardless of content or edge length.

- In the graph view (right), click on the node with the text "*FACT FILE*". The bottom part of the screen will change to show the available conditions for the node. Click on "*Substring*" and enter *"FACT FILE"* in the nearby text field. Now try the wrapper out again by clicking on "*Test wrapper*". You will notice that only four results are returned, as we have effectively "anchored" the top node to the heading "*FACT FILE*".

- However, on closer inspection, you will notice that this wrapper fails to select two of the results completely. This is because they contain more lines of information – and therefore, more nodes – than the example instance. In order to create wrappers which allow such variations, *multiple match edges* need to be used.

2. Multiple match edges

- Click on the edge joining the nodes *NAME:* and *ADDRESS:*. You will notice that this edge has a length of 17.0pt. Set "*Multiple match*" to "*Last*" and "*Max length*" to this length, 17.0pt. Repeat this step for each edge below until you get to the final node, *EMAIL:*. Now try out the wrapper again. You should find that it now correctly selects all the lines of text in each record.

- Multiple match edges which neighbour each other are actually treated as one edge by the algorithm. A quicker way to define the above wrapper would be to remove all nodes (and edges) below *ADDRESS:*. This way, there would be only one normal and one multiple match edge in the wrapper, which would then match all edges until the bottom of the fact file (no more nearby nodes) is reached. Try it out!

- So far, we have defined the wrapper by extracting all nodes below *NAME:* until there was no further vertical neighbour within a given distance. An alternative way to define a suitable wrapper is to say: "Let's extract all nodes between *NAME:* and *EMAIL:*. To do this, select the relevant edge and change the multiple match setting to "*First*". Now, click on the *NAME:* node and set the condition "*Match content: Substring*" to "*NAME*". For the node *ADDRESS:* set the condition "*Match content: Substring*" to "*E-MAIL*". Note that the original text in the node plays no role here.

- There are important differences between "*Match content: last"* and "*Match content: first"*. With *match content* set to *last*, the additional intermediate nodes all need to match at least one of the nodes either side of the edge. The edge will then carry on matching nodes in the same direction until it comes across a node which matches neither node in the original instance. With *match content* set to *first*, no node needs to match either node in the original instance; the algorithm will stop looking as soon as it finds a node that matches.

3. Blocks

- If you select the segmentation mode "*Blocks*" and re-open the document, you will see that the fact files are segmented as one complete block. The inbuilt algorithm recognizes logically grouped elements such as paragraphs and represents them as one node. In this case, you could avoid using multiple mach edges altogether and match just the first neighbour below the node "*FACT FILE*". *Please note that it is currently not possible to mix different granularities in different layers in a wrapper.*

4. Hierarchical wrapping

- Wrappers can be nested within another. For each root wrapper result, the sub-wrappers will be run on the extracted nodes within that particular result. In this way, you can extract a complete record in the root wrapper and the individual data items in the sub-wrappers. In order to define nested wrappers, it is necessary to generate the wrappers for each level separately using the GUI. The wrappers can then be combined using a text editor.

- Look in the directory *example* of the distribution. Here, the wrapper *example.xml* is a combination of *factfile.xml*, the root, and *name.xml* and *phone.xml*, the sub-wrappers.

- The wrappers *name.xml* and *phone.xml* are defined by single nodes. The purpose is to match the single data item (using a *substring* condition) and output the text contents. In order to do this, the field "*Extract this node*" is enabled and an appropriate XML tag name is set.

- The attribute "*area-based"* for a wrapper determines how the extracted data is passed to its child wrappers. In the default setting, *true*, all nodes whose centres lie within the rectangular bounding box of the matched nodes are passed down to the child wrappers. If this setting is *false*, only those nodes which have been explicitly matched are passed down. This setting is of particular importance if multiple match edges are used, as nodes which lie in between will not be passed down unless this setting is *true*.

5. Command-line execution

- Once saved as an XML file, the wrapper can be run from the command line using the shell script *graphwrap* or batch file *graphwrap.bat*. Take a look at *graphwrap-example* or *graphwrap-example.bat*. This script runs the wrapper *example.xml* on the *travel.pdf* document and returns the output to the console. In order to save the output to an XML file, remove the switch *-toconsole* and add the desired filename as a second command-line parameter.

6. Demo buttons

- The buttons in the top-right of the screen were added for the CeBIT presentation. Here, a simple hierarchical wrapper can be defined, and the output shown either on the page or as an XML file. Please note that it is not possible to save wrappers generated in this way.

- If you have generated a wrapper using the GUI and wish to visually try it out on another document without saving it to a file, you can save it here as a *root wrapper*, open the new document and execute the wrapper by clicking on *Show result*. Please note that, on opening a new document, the existing wrapper will otherwise be deleted.

Contact details

Tamir Hassan
Database and Artificial Intelligence Group 184/2
Information Systems Institute
Vienna University of Technology
Favoritenstraße 9-11
A-1040 Wien
Austria

Email: *graphwrap@tamirhassan.com* Web: *http://www.tamirhassan.com*

This work is funded by the Austrian Federal Ministry for Transport, Innovation and Technology (Project no: 815128/9306)