Decision-Based Specification and Comparison of Table Recognition Algorithms

Richard Zanibbi[1], Dorothea Blostein[2], James R. Cordy[2]
[1]**Department of Computer Science, Rochester Institute of Technology, 102 Lomb Memorial Drive, Rochester, NY, USA 14623-5603,** `rlaz@cs.rit.edu`
[2]**School of Computing, Queen's University, Kingston, ON, Canada, K7L 3N6** `{blostein,cordy}@cs.queensu.ca`

1 Introduction

The vast majority of algorithms in the table recognition literature are specified informally as a sequence of operations [7]. This has the undesirable side effects that models of table structure are implicit, defined generatively by the sequence of operations, and that the effects of intermediate decisions are often lost as usually a single interpretation is modified in-place.

We wished to compare the Handley [2] and Hu et al. [4]. table structure recognition algorithms and the complete set of table cell hypotheses they each generated, including any rejected in the final result. Rebuilding the systems using procedural code that transformed data structures for interpretations in-place would not have achieved this goal. Initially we translated the strategies to a formal model-based (specifically grammar-based) framework. A well designed model-driven system (such as DMOS by Coüasnon [1]) makes it easier to observe and record decision making, and can be programmed succinctly by a model specification. However, we found mapping the sequence of operations in the strategies to a model based description was difficult, and the formal system required frequent and substantial reconfiguration in order to incorporate unanticipated requirements.

We then considered an intermediate level of formalization. By using a small set of basic graph-based operations we could define recognition algorithms as a series of decisions, where the alternatives for each decision were model operations of a specified type (e.g. classifying table cells as header cells or data cells). This made the model operations considered and applied at each decision point explicit, permitted dependencies between logical types to be automatically recovered, and allowed the complete history of hypothesis creation, rejection, and reinstatement to be automatically captured. The resulting formalization is the Recognition Strategy Language (RSL).

2 RSL: A Simple Table Cell Recognition Example

Figure 1 shows three words in a table being transformed by RSL into a table cell representation using a sequence of decision operations. At run-time, each decision determines which model operations to apply based on the output of a decision function (e.g. selectHorAdjRegions()). The decision syntax defines a space of possible model operations (that classify, segment, or relate regions), and defines which parameters and types of elements in the interpretation may be observed by the decision function. For example, in

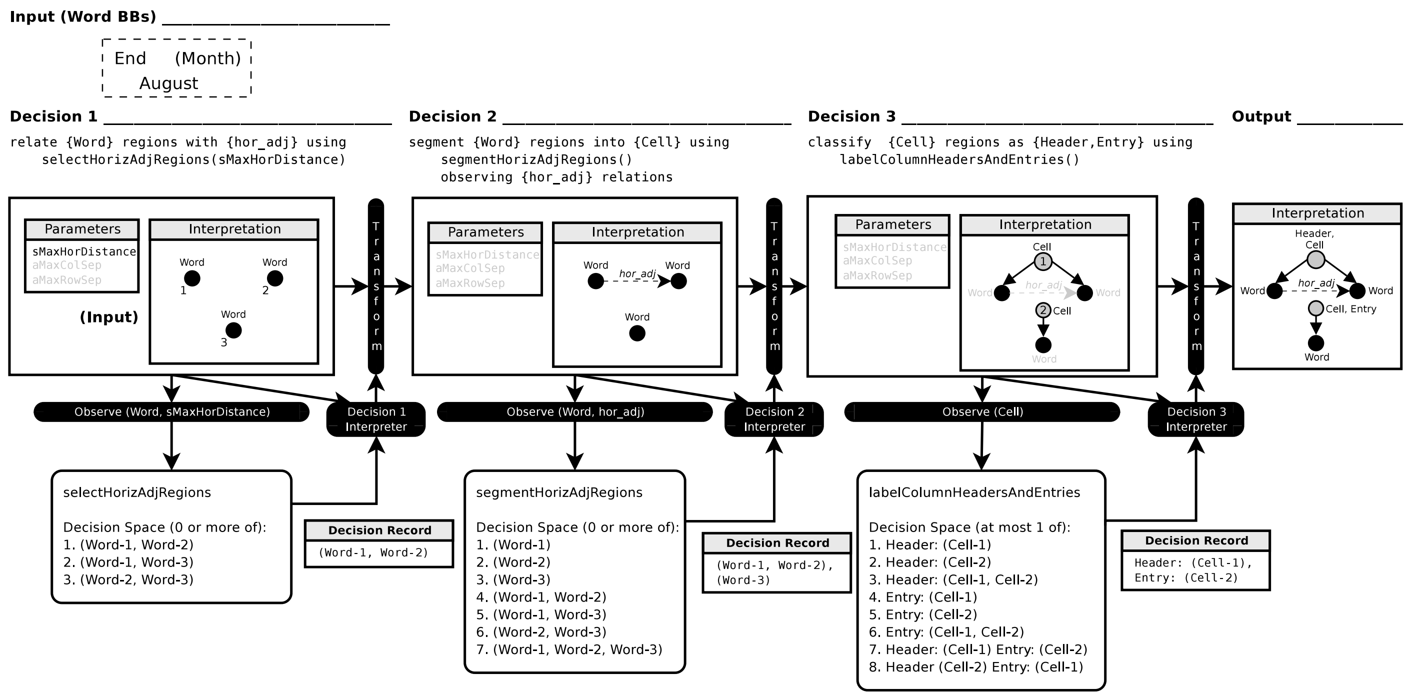


Figure 1: Decision-Based Specification. In this simple example we show a table header cell (‘End (Month)’ and data cell (‘August’) being recognized from the layout of the word bounding boxes. *Note: the decision space (set of possible operations) for the first decision should have six elements, to produce all possible pairs of three elements.*

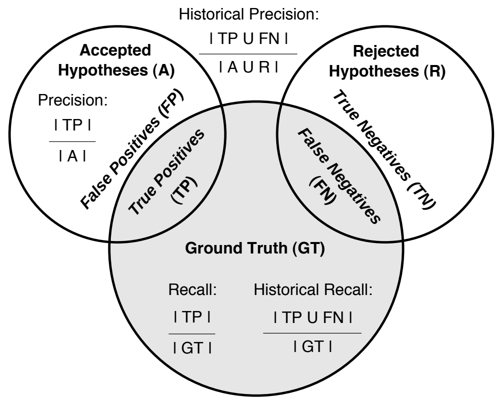


Figure 2: Recall, Precision, Historical Recall, and Historical Precision

Decision 3, only Cell regions and the location of their member words are visible to the decision function labelColumnHeadersAndEntries().

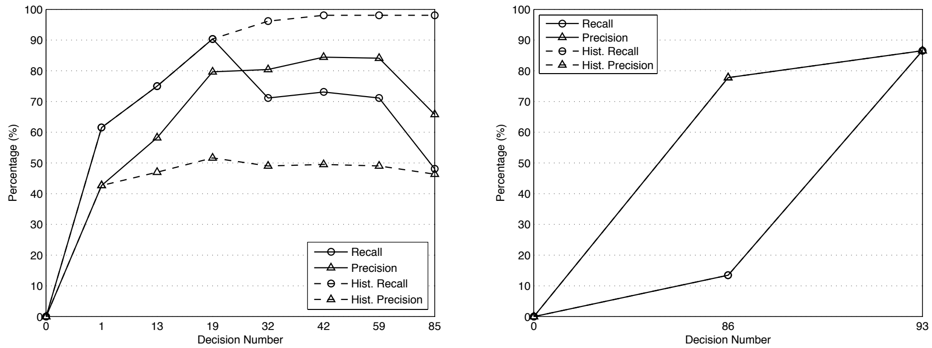
RSL automatically updates interpretations for the programmer; the programmer need only define an RSL specification and the set of decision functions used in the specification. All decision outcomes are passed to a *Decision Interpeter* associated with the decision that insures that the decision returned is valid, i.e. contains only elements from the decision space. Figure 1 shows the three core RSL decision operations: relating, segmenting, and classifying regions in the input. In Figure 1 the input regions are bounding boxes for the three words. These operations create (and accept) hypotheses; other RSL operations reject hypotheses, and accept interpretations for output. A complete definition of the RSL language is available elsewhere [5].

RSL records all decision outcomes. From the outcomes we may produce a hypothesis history which describes when hypotheses are first proposed (generated), and the subsequent times at which hypotheses are rejected or reinstated. Reinstatement refers to when a rejected hypothesis is itself rejected, resulting in the hypothesis being accepted again. A hypothesis history also records confidence values associated with hypothesis creation, rejection, or reinstatement (e.g. probabilities or fuzzy values).

3 Historical Recall and Precision

From a hypothesis history we may observe new metrics that take rejected hypotheses into account. Figure 2 illustrates the relationship between recall, precision, *historical recall* and *historical precision* [6]. Informally stated, the new historical measures give an algorithm credit for correct hypotheses that it made somewhere along the way, even if the algorithm later rejected these hypotheses.

Conventional and historical recall can be directly compared, as they both describe coverage of the set of ground truth elements. Note that historical recall will always be greater than or equal to recall (refer to Figure 3a). Historical recall never decreases during a recognition algorithm's progress, while recall may increase or decrease at any point. It is harder to relate conventional and historical precision, as precision measures the accuracy of what is accepted as valid, while historical precision measures the accuracy (or *efficiency*) of hypothesis generation, in terms of hypotheses that the algorithm considers within the space of possible interpretations.



a. Handley b Hu et al.

Figure 3: Comparison of Cell detection metrics for Handley and Hu et al. algorithms applied to a Single Table. The decision numbers shown correspond to the sequence of RSL decision operations that altered cell hypotheses

4 Decision-Based Comparison of Algorithms

We can use historical recall and precision along with conventional recall and precision to summarize the decision-making process of algorithms implemented in RSL. Figure 3 presents all four of these metrics for the Handley and Hu et al. algorithms applied to a single table. The Handley algorithm takes an iterative approach to modifying cells (specifically, all words are classified as cells, and then cells are merged in stages), whereas the Hu algorithm only updates cells twice; this is because the Hu algorithm makes many decisions about other hypothesis types before manipulating cells.

Note that the Handley algorithm has higher recall and precision after Decision 19 than the Hu algorithm at any point. However, conventional performance evaluation for table structure recognition considers only the recall and precision of final interpretations, in which case the Hu algorithm appears to perform better. This new information about the decisions made by each algorithm may also be used to better understand and *combine* these strategies. For example, it is possible to combine the RSL decision operations for these two algorithms to produce a new algorithm which performs better than either for the table in question.

In the future, we plan to apply machine learning techniques to optimally combine recognition algorithms specified in RSL, making use of hypothesis histories, historical recall and precision, and static analyses of RSL specifications [3].

References

[1] B. Coüasnon. DMOS: A generic document recognition method, application to an automatic generator of musical scores, mathematical formulae and table recognition systems. In *Proc. Sixth Int'l Conf. Document Analysis and Recognition*, pages 215–220, Seattle, WA, 2001.

[2] J.C. Handley. Table analysis for multi-line cell identification. In *Proc. Document Recognition and Retrieval VIII (IS&T/SPIE Electronic Imaging)*, volume 4307, pages 34–43, San Jose, CA, 2001.

[3] Susan Horwitz and Thomas Reps. The use of program dependence graphs in software engineering. In *ICSE '92: Proceedings of the 14th international conference on Software engineering*, pages 392–411, New York, NY, USA, 1992. ACM Press.

[4] J. Hu, R. Kashi, D. Lopresti, and G. Wilfong. Table structure recognition and its evaluation. In *Proc. Document Recognition and Retrieval VIII (IS&T/SPIE Electronic Imaging)*, volume 4307, pages 44–55, San Jose, CA, 2001.

[5] R. Zanibbi. *A Language for Specifying and Comparing Table Recognition Strategies*. PhD thesis, Queen's University, Kingston (Canada), December 2004. 236 pages.

[6] R. Zanibbi, D. Blostein, and J. R. Cordy. Historical recall and precision: summarizing generated hypotheses. In *Document Analysis and Recognition, 2005. Proceedings. Eighth International Conference on*, pages 202–206 Vol. 1, 2005.

[7] R. Zanibbi, D. Blostein, and J.R. Cordy. A survey of table recognition: Models, observations, transformations, and inferences. *Int'l J. Document Analysis and Recognition*, 7(1):1–16, Sept. 2004.