Natural Language Engineering **2** (4) 303–304. Printed in the United Kingdom

© 1996 Cambridge University Press

Finite state morphology and formal verification

MANUEL VILARES FERRO, JORGE GRAÑA GIL

Computer Science Department, University of Corunna,
Campus de Elviña s/n, 15071 La Coruña, Spain
e-mail: {vilares,grana}@dc.fi.udc.es

PILAR ALVARIÑO ALVARIÑO

Spanish Philology Department, University of Santiago de Compostela,
Burgo de las Naciones s/n, 15705 Santiago de Compostela, Spain
e-mail: fepili@usc.es

(*Received 26 December 1996; Revised 24 January 1997*)

The full paper describes an environment for the generation of non-deterministic taggers, currently used for the development of a Spanish lexicon. In relation to previous approaches, our system includes the use of verification tools in order to assure the robustness of the generated taggers. A wide variety of user defined criteria can be applied for checking the exact properties of the system.

Motivation. The choice of the finite automata model as operational formalism for tagging assures computational efficiency. Safety is guaranteed by the separation between this operational kernel and the high-level descriptive formalism.

However, one of the major services of every lexicon ought to be to provide as much information as possible about errors, because of the complexity of actual implementations, and the natural evolution suffered by these kinds of systems. The goal is to minimize the time dedicated to debugging the system. For this reason, our discussion has a practical sense.

The approach presented allows verification of morphological analysers by computing reductions. We are interested in verification methods combining modularity in the description of the system and flexibility in the verified properties, such as AUTO Madelaine *et al.* 1989. AUTO computes small-scale models of finite transition systems. These reduced systems are quotiens of the system under study. The parameter of the reduction is a user-defined abstraction criterion, which embodies a particular point of view. One is therefore able to build a variety of quotiens of a same system, which are small enough to verify particular properties.

Spanish as a guideline example. In the full paper we exemplify various aspects of the system using the case of Spanish, an inflectional language that displays a great variety of morphological processes going beyond concatenation. At present, we are able to tag the most common 7000 lemmas of Spanish. The corresponding automaton has more than 35,000 states and the average speed is 1400 words tagged per second.

https://doi.org/10.1017/S1351324997001551 Published online by Cambridge University Press

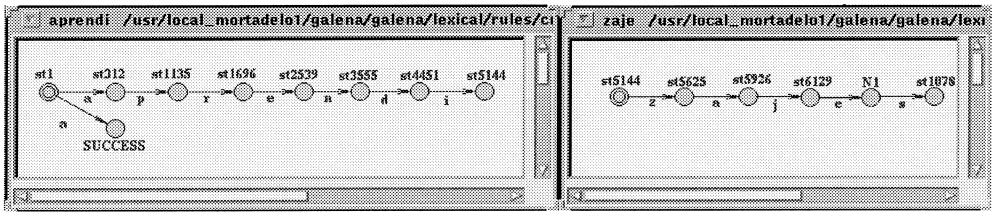


Fig. 1. Error recovery for `aprendizaje` (*learning*), from `aprendicaje` (an incorrect word).

This complexity suggests the need to interface the morphological analysis with a formal proof system which allows us to verify easily the properties demanded.

Verification by reduction. The verification method we want to advocate in this paper is based on reductions of a global FA. These collapse states of the automaton to reach sizes reasonable enough to be displayed and easily understood. We summarize some of the features we have implemented:

- **Tracing facilities.** From the global FA, the verification process lets the user obtain the path, that is, the set of states visited by the tagger, for a given word. This partial view allows us to check that the tagging is correct, and also to validate that the units involved are working correctly.
- **Improving maintenance.** Newer versions should increase the power of previous systems, but updates can introduce erroneous patterns. Our goal is to detect these kinds of bugs at compile time. One way of doing this is to compare patterns. We automatically take patterns from the old tagger, which we assume to be correct, and verify whether they are present in the new model.
- **Automatic error recovery.** When an error causes the automaton to stop the recognition process, the system automatically finds a *bridge transition* and catenates the portion of the erroneous string that we have identified as correct, with the endings corresponding to existing paths. As a result we obtain the set of possible corrections proposed by the verifier. It is clear that this strategy works only for *substitution* errors, but not for *insertion* or *deletion* errors.

Conclusion. The full paper describes work in progress. It represents only a first approach to verification in tagging, but preliminary results are promising, and the operational formalism is well adapted to deal with more complex problems.

References

Madelaine E. and Vergamini, D. 1989. AUTO: A verification tool for distributed systems using reduction of finite automata networks. In *Proc. FORTE'89 Conference*, Vancouver.

https://doi.org/10.1017/S1351324997001551 Published online by Cambridge University Press