# New Features in PARC Finite State Toolkits

Lauri Karttunen

Palo Alto Research Center

The xfst utility that accompanied the Beesley & Karttunen book on "Finite-State Morphology" in 2003 ([1]) has been updated. In this talk I will describe the new features and show how they can be deployed. The new features include:

- full UTF-8 support
- symbol ranges (e.g. "a-z")
- definitions for lists (sets of symbols)
- list flags (symbols for member of, not a member of)
- insert flags (symbols for an embedded language)
- built-in and user-definable functions

The publicly available xfst tool is the "little sister" of a more powerful utility called fst that is used in commercial applications at companies such as Inxight (recently acquired by Business Objects) and Powerset. One of the new features in the fst tool is a pattern matching algorithm that overcomes the limitations inherent in the approach introduced in the B&K book. As the book explains, in the (x)fst regular expression language it is possible to define pattern networks with expressions such as

*Example 1.* `(Det) Adj* Noun* @-> "<NP>" ... "</NP>"`

where `@->` is the left-to-right longest-match replace operator ([2]). Given appropriate definitions for Det, Adj, and N, the above expression compiles into a transducer inserts XML tags around noun phrases. For example, it maps

*Example 2.* `children are playing on a jungle gym`

into

*Example 3.* `<NP>children</N> are playing on <NP>a jungle gym</NP>.`

While this method works well for many purposes such as the recognition of dates, phone numbers, addresses, it becomes impractical when the number of words in the component expressions of the pattern increases beyond a few thousand words as the case would be in a real noun phrase recognizer.

The principal reason for the blow-up is that the `@->` operator encodes the longest-match constraint in the state and arc space of the resulting transducer. The new pattern matching method that is implemented in fst enforces the longest match constraint in the runtime algorithm and not in the physical network. I will outline the new pattern matching algorithm and illustrate its benefits.

## References

1. Beesley, K.R., Karttunen, L.J.: Finite State Morphology. CSLI Studies in Computational Linguistics. CSLI Publications, Stanford, CA (2003)
2. Karttunen, L.J.: Directed Replacement. In Joshi, A.K., Palmer, M.S., eds.: Proceedings of the 34th annual meeting of the Association for Computational Linguistics. Annual Meeting of the ACL, Morristown, NJ, Association for Computational Linguistics (1996) 108–115