Efficient Computation at the Interfaces

Thomas Graf, UCLA

Introduction

Syntax is constrained by bare output conditions (LF, PF, and the parser). It has been claimed that reference-set computation (transderivationality) comes with a high resource load that fails the computability requirements of the parser, whence it cannot be part of syntax (Collins 1996).

A mathematical perspective, though, shows that certain kinds of reference-set computation are as efficiently computable as normal constraints.

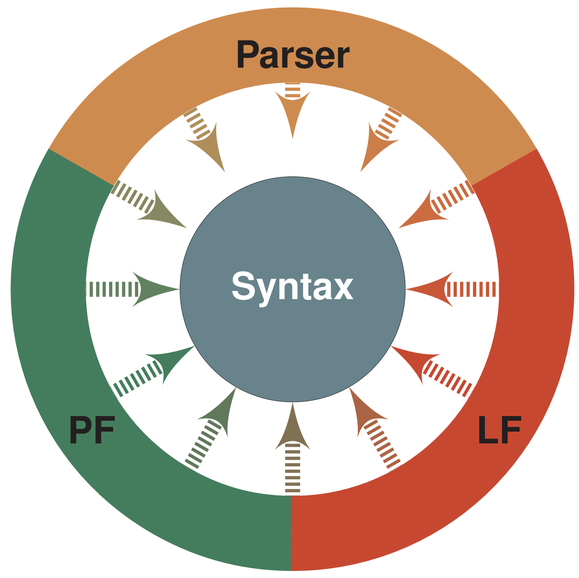

Figure: Syntax is constrained by LF, PF, and the parser — but at least with respect to the latter, syntax may enjoy more leeway than expected.

A Mathematical Model of Reference-Set Computation

Reference-set constraints can be modelled as tree automata with outputs, which may be specified by Optimality Systems (a variant of OT grammars; Frank and Satta 1998).

An Optimality System consists of

- a collection of reference types over the input language,
- a collection of reference sets over the candidate language,
- a function mapping reference types to reference sets,
- a sequence of OT constraints.

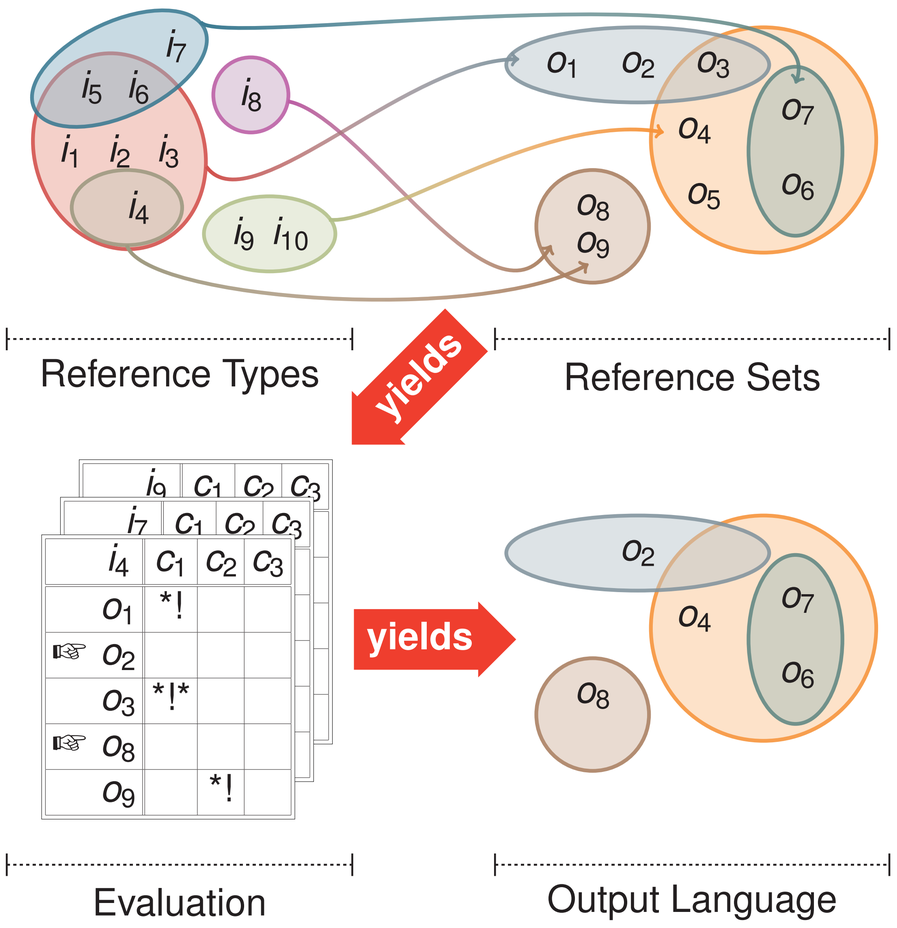

Figure: Reference-set constraints can be modelled by Optimality Systems. The OT generator is defined in a modular fashion using reference types, reference sets, and a map from reference types to reference sets.

Doing Away with Reference-Set Computation

Given our new perspective on reference-set computation, it is easy to show that some reference-set constraints can be reduced to standard well-formedness conditions (implemented as, say, additional features on lexical items etc.). This allows syntax to use any reference-set constraint for which there is an efficiently computable equivalent for the parser.

A reference-set constraint is reducible if the following holds for its respective Optimality System.

- Output joint preservation: If two reference sets overlap, then the reference types that are mapped to them overlap, too.
- Type Level Optimality: If reference type T is mapped to reference-set R, then an output candidate in R is optimal for some input of type T only if it is optimal for all inputs of type T.
- The OT generator and each OT constraint can be modelled by finite-state tree automata with outputs.

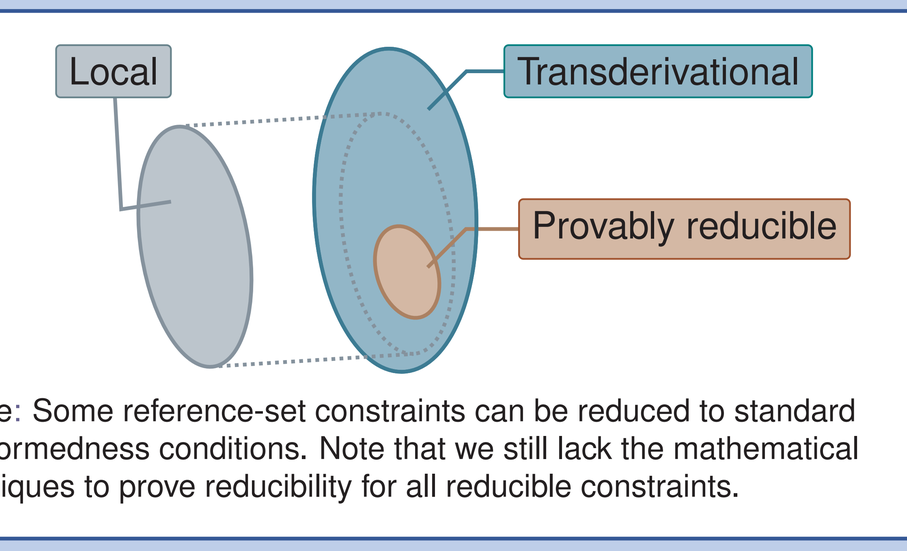

Figure: Some reference-set constraints can be reduced to standard well-formedness conditions. Note that we still lack the mathematical techniques to prove reducibility for all reducible constraints.

Results for Constraints from the Literature

All prominent instances of reference-set computation (Fewest Steps, Rule I, Scope Economy, Focus Economy) obey both output joint preservation and type level optimality.

For Fewest Steps, Merge-over-Move and Focus Economy, the conditions on the generator and the OT constraints are satisfied, too, so both constraints have efficiently computable equivalents that do not involve reference-set computation.

Conclusion and Open Questions

The reducibility of certain reference-set constraints to well-formedness conditions loosens the parser's grip on syntax. The amount of reference-set computation in syntax depends only on the availability of computable equivalents.

This raises two intriguing questions:

- Do similar things happen with LF/PF-conditions?
- Why should syntax prefer reference-set constraints over their efficiently computable doubles?

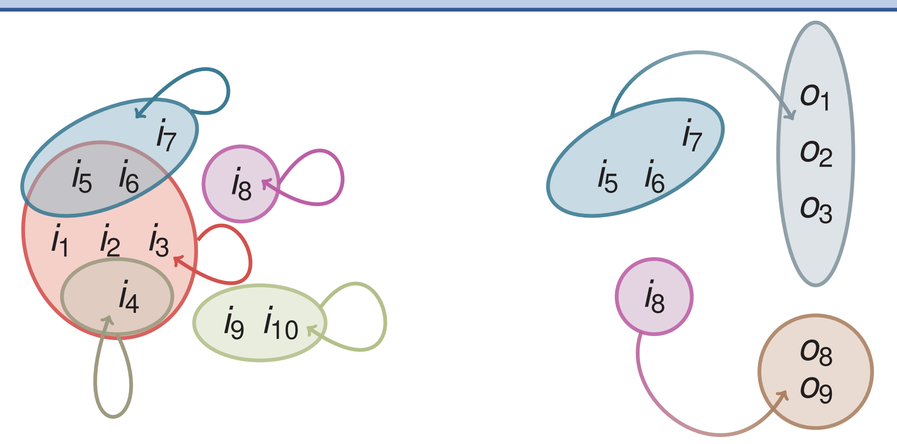

Figure: Almost all instances of reference-set computation in the literature use one of the two configurations above, both of which are output joint preserving.

References & Acknowledgements

Collins, Chris. 1996. *Local economy*. Cambridge, Mass.: MIT Press.

Frank, Robert, and Giorgio Satta. 1998. Optimality theory and the generative complexity of constraint violability. *Computational Linguistics* 24:307–315.

This research was supported by a DOC-fellowship of the Austrian Academy of Sciences. I am grateful to Ed Stabler for helpful discussion.

http://tgraf.bol.ucla.edu tgraf@ucla.edu