Types and Clauses: Two styles of probabilistic processing in CUF

Chris Brew
Human Communication Research Centre
University of Edinburgh
2 Buccleuch Place, Edinburgh EH8 9LW, SCOTLAND
Chris.Brew@edinburgh.ac.uk

May 19, 1998

Introduction

In this comment I provide further explication of the general approach I previously advocated [Bre93] for integration of probabilities into CUF[1]. This is in a response to the somewhat different proposal which has been made by Eisele in this volume [Eis94]. My comments bear on two matters, both of which arise from a single difference between Eisele's proposals and my own.

The major difference between my proposals and those made by Eisele is in the choice of encoding. He chooses to associate probabilities with definite clauses, with the effect that probabilistic calculations proceed in parallel with the (attempted) construction of proof trees. By contrast, I choose to focus more closely on types. There are two reasons for this, the first is that I believe it to be appropriate to parameterise stochastic grammars with schemes for the combination of information, which is easier to do in a coherent way with types than with clauses. The second is simply that I find types easier to think about in probabilistic terms than I do definite clauses. In what follows I expand on these points.

Fixed or variable combination schemas?

As Eisele points out, numerically annotated proof trees have a close resemblance to stochastic context free grammars, which leads naturally to an assumption about the independence of probabilities for different branches of the proof tree. This assumption is, I think, a direct consequence of the design choice which Eisele describes in the following way:

> With a probabilistic interpretation in mind, there is usually not much choice concerning the method how the quality scores of partial results should be combined to get an overall score. Therefore this calculation scheme will be fixed in our implementation. While at times this may feel like a restriction, it has some important advantages. It simplifies work in so much as the grammar writer has

[1]Release 2.30 of CUF differs from its predecessors in providing explicit support for foreign functions and types – including numbers and arithmetic. This makes it feasible to build and test probabilistic mechanisms.

> no need and no possibility to fiddle with combination schemes. Instead, a general mechanism provides support for ... efficient search, compile time optimizations, training procedures and the like, which could not be done in general if arbitrary schemes were allowed ... [Eis94]

While highly sympathetic to the computational motivations for this decision I wish to question the wisdom of this assumption. It is interesting to observe that empirical studies of stochastic CFGs by Magerman and Marcus regard this assumption as a primary culprit for the previously poor performance of models based on stochastic grammars. They note that:

> According to probability theory the likelihood of two independent events occurring at the same time is the product of their individual probabilities. Previous statistical techniques apply this technique to the co-occurrence of two theories in a parse, and claim that the likelihood of the two theories being correct is the product of the probabilities of the two theories.
>
> This application of probability theory ignores two vital observations about the domain of statistical parsing:
>
> - Two constructs occurring in the same sentence are not necessarily independent (and frequently are not). If the independence assumption is violated, then the product of the individual probabilities has no meaning with respect to the joint probability of two events.
> - Since statistical parsing suffers from sparse data, probability estimates of low frequency events will usually be inaccurate estimates. Extreme underestimates of the likelihood of low frequency events will produce misleading joint probability estimates.
>
> ¿From these observations, we have determined that estimating joint probabilities of theories using individual probabilities is too difficult with available data. We have found that geometric mean of these probability estimates provides an accurate assessment of a theory's viability [MM94]

Since the empirical basis for the use of geometric mean is relative to a particular grammar and corpus, it would be premature to endorse it for all grammars and corpora, still less for all constellations of probabilistic definite clauses. However, for present purposes what matters is that situations may arise under which the standard mode of probabilistic combinations of theories has undesirable consequences in practice. I therefore believe that it is more appropriate to continue to allow grammar writers greater control over the combination methods than Eisele currently wishes to provide.

Unfortunately, if one goes down the route which I advocate, one is left with two unsolved problems:

- One needs to provide a specification language which allows the grammar writer to describe combination schemes.
- One needs to enable the implementation to carry out the specified operations with acceptable efficiency.

There is nothing very difficult about the first requirement, although it might well be worth providing a mechanism by which standard library modules (e.g. SEQUENCE_OF_INDEPENDENT_EVENTS) can be bound to particular patterns of types occurring in a grammar. The second requirement is more onerous. It would obviously help if the implementations of standard combination mechanisms were as efficient as possible. One might also be able to get some useful mileage from user supplied annotations. The system might for example be able to memoize if it knows a given combination operation is monotonic decreasing (as multiplication is and geometric mean isn't). Or it might be able to exploit the knowledge that an operation is associative in order to memoize in some appropriate way.

Types or clauses?

The second issue is that of the locus at which probabilistic information sits. The consequences of the differences in approach are, I think, not so severe as those which arose with respect to fixed or variable combination schemes, but still merit explanation beyond what is implicit in my own work or that of Eisele.

Where Eisele prefers to associate probabilities with definite clauses, I choose to place greater emphasis on the role of types. This is because CUF's type system is essentially propositional, and I wish to exploit the close links between propositional calculus and probability theory. Probability theory is a conservative extension of propositional logic in which conjunction of propositional variables corresponds to multiplication of probabilities, disjunction of propositional variables corresponds to addition of probabilities, negation is complement with respect to unity, and so on[2]. Although the extension might be interesting, it is not obvious that this parallel extends in any straightforward way to the fuller description language which CUF uses for sorts. In particular a theoretically and practically satisfactory general treatment for reentrancies and relational dependencies seems like a hard task.

It is not clear to me that this is more than a matter of viewpoint. In both cases we have a means for evading commonly made criticisms of stochastic context-free grammars. As Eisele writes

> The decision to base the granularity of the probabilistic description on the granularity of the predicate definition is not a severe restriction [and hence the criticisms of stochastic context-free grammars do not apply to the extension]. Should it turn out that we need more fine-grained distinctions or more context dependencies than is possible within a given description, we can always add auxiliary definitions for a new predicate involving all relevant information, whose only purpose is to provide attachment points for additional probabilistic information. [Eis94]

An entirely analogous approach within my preferred framework is to create new types, associated with probabilistic information, along with new predicates whose sole purpose is to pass around this probabilistic information. The only advantage which I would want to claim for the type based approach is that the existence of a very simple probabilistic interpretation predisposes the grammar writer to make use of types which do indeed correspond to the events of a suitably formulated probabilistic theory. Of course, once the grammar writer has identified places in which probabilistic information is representationally appropriate, corpora and well-designed training algorithms should be pressed into place to estimate values as needed.

[2]Clearly, this parallel is going to be far less useful if we are using some non-standard method for combining numbers, as advocated in the previous section.

Conclusion

I have focussed on a couple of problematic aspects of Eisele's decision to associate probabilities only with the definite clauses of a CUF theory. This means that I have had nothing to say about the detailed and explicit accounts of search and parameter estimation which form the largest part of his contribution. It is entirely appropriate that a framework as general as CUF should be decorated with similarly general accounts of statistical processing. I await with great interest the development of new hybrid theories which successfully make use of the powerful techniques which he advocates.

It is clear that the style of operation which I advocate has more limited goals, in particular the faithful encoding of currently useful techniques, and that it will one day be superseded by something much closer to Eisele's approach. Whichever approach is adopted the important task for the near future is that of ensuring that hybrid systems based partially on feature logic are able to participate fully in the current move towards effective technologies relying on statistical regularities.

Bibliography

References

[Bre93] Chris Brew. Adding preferences to CUF. In Jochen Dörre, editor, *Computational Aspects of Constraint-Based Linguistics Description I*. July 1993. DYANA-2 deliverable R1.2.A.

[Eis94] Andreas Eisele. Towards probabilistic extensions of constraint-based grammars. In Jochen Dörre, editor, *Computational Aspects of Constraint-Based Linguistics Description II*. August 1994. DYANA-2 deliverable R1.2.B.

[MM94] David M. Magerman and Mitchell P. Marcus. Pearl: A probabilistic chart parser. In *Proceedings, 2nd International Workshop on Parsing Technologies*, May 1994.