A Goal-Directed Multi-Level Stylistic Analyzer

Pat Hoyt and **Chrysanne DiMarco**[*]
Department of Computer Science
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1
cdimarco@logos.uwaterloo.ca

Abstract

Sophisticated natural language processing systems should be able to deal with the subtle but significant effects of style on communication, but the difficulties of representing stylistic knowledge in a formal representation had resulted in only simplistic and heuristic approaches to implementation.

In this paper, we take the problem of formally representing stylistic knowledge as our starting point. It is our belief that stylistic knowledge must first be formalized, rendered in a well-defined representation, before a computational analysis of style can be attempted. And it is our further contention that a formal representation will facilitate a very transparent implementation. We show how a formal representation of syntactic style can be used as the basis for a general-purpose stylistic analyzer that can produce descriptions of the stylistic features of an input sentence at multiple levels of abstraction.

1 The importance of stylistic analysis in natural language processing

The importance of dealing with pragmatic aspects of language in computational systems is undeniable. People communicate a great deal of information through pragmatic nuances, and a knowledge of how these subtleties influence meaning is part of a full understanding of language. Systems that could analyze the effects of style on communication would provide information about the implicit meaning that is contained in a text. And generation systems that could control style would produce text that intentionally conveys a specific pragmatic effect. Both stylistic analysis and generation could be used in applications, such as text critiquing, second-language instruction, and machine translation, for which understanding the effects of how something is said is as important as understanding what is said. Ultimately, computational stylistics should be a part of any system that attempts to deal with 'real-world' language.

But very few natural language understanding systems have attempted to deal with issues of style,[1] and those that do have generally taken a simplistic and heuristic approach. Stylistic analysis has not yet developed the systematic and rigorous methods of syntactic analysis and semantic interpretation. Part of the reason is obvious: understanding style is hard. Stylistic effects are difficult to articulate and even more difficult to define.

In this paper, we take the problem of formally representing stylistic knowledge as our starting point. It is our belief that stylistic knowledge must first be formalized, rendered in a well-defined representation, before a computational analysis of style can be attempted. And it is our further contention that a formal representation will facilitate a very transparent implementation. We show how a formal representation of syntactic style can be used as the basis for a general-purpose stylistic analyzer that could be used as a component of a natural language processing system.

2 A theory of syntactic style

A formal representation of stylistic knowledge should ideally be based on an underlying linguistic theory: there should be a vocabulary of concepts, a clear definition of how the concepts are related, and a systematic way for building new concepts out of existing ones. In our earlier work [DiMarco and Hirst 1993; Green 1992], we presented a computational theory of syntactic style that is a multi-level representation of stylistic grammar rules. This section summarizes the details of this work as presented in [DiMarco and Hirst 1993]. Green [1992] develops the linguistic underpinnings of the theory; Hoyt [1993] presents the representation of the complete theory in a syntactic stylistic grammar.

[*]Please direct all correspondence to the second author.

[1]By *style*, we do not mean literary style, but rather the style of texts such as high-quality magazines and newspapers, technical manuals, and business correspondence.

2.1 Fundamental concepts

In designing a computational theory of style, we constructed a vocabulary of stylistic concepts at three levels of abstraction:

- *Primitive elements* are stylistically significant syntactic properties of sentence components.
- *Abstract elements* are general stylistic properties of groups of sentences.
- *Stylistic goals* are the writer's intentions for high-level pragmatic properties of text.

At all levels, the guiding principle of the theory is that style is *goal-directed*, that is, linguistic choices are made to achieve specific stylistic goals, such as clarity or abstraction. Therefore, we tie low-level syntactic choices to high-level stylistic goals. The fundamental concepts that are used to integrate the multiple levels of the theory are stylistic *concord* and *discord*, which we define as follows:

Concord: A stylistic construction that conforms to the norm for a given genre.

Discord: A stylistic construction that deviates from the norm.[2]

2.2 Primitive elements of style

At the lowest level of the theory, there are two views of sentence structure, *connective* and *hierarchic*:[3]

Connective ordering: The result of cohesive bonds drawing together components in a linear ordering.

Hierarchic ordering: The result of bonds of subordination and superordination drawing together components in a nested ordering.

The connective and hierarchic orderings are used in the definition of primitive stylistic elements to provide a precise syntactic basis to the theory, yet also allow a mapping to the abstract elements.

We use the terms *conjunct* and *antijunct* with superscripts to indicate the degree of connectivity or disconnectivity. Syntactic components are classified as either $conjunct^5$ or $conjunct^6$ (excessively connective), $conjunct^3$ or $conjunct^4$ (strongly connective), $conjunct^2$ (moderately connective), $conjunct^1$ (mildly connective), and $conjunct^0$ (neutral). Similarly, the terms $antijunct^0$ through $antijunct^4$ are used to indicate increasingly disconnective effects; $conjunct^0$ and $antijunct^0$ are the same.

There is a complementary vocabulary of primitive elements for the hierarchic view. The stylistic effects of syntactic components are correlated with the degree of subordination or superordination; the classifications are analogous to the connective: $subjunct^4$ through $subjunct^0$ (decreasingly subordinate) and $superjunct^0$ through $superjunct^4$ (increasingly superordinate); $subjunct^0$ and $superjunct^0$ are the same.

We adapted the work of Halliday and Hasan [1976] on cohesive relations to assign classifications to the connective elements. Halliday and Hasan consider substitution, including ellipsis, to be the most strictly cohesive relation, followed by reference, and then conjunction. We adopted this ranking, and so we classify intrasentential substitution and ellipsis as strongly connective ($conjunct^3$), reference as moderately connective ($conjunct^2$), and conjunction as mildly connective ($conjunct^1$). We also classify interpolation, parenthetical constructions, as disconnective ($antijunct^2$).

In assigning a hierarchic classification to a syntactic component, we adapted Halliday's [1985] work on *subordination*, specifically, embedding and hypotaxis, and the definition of the term *superordination* by Quirk *et al.* [1985]. We classify embeddings as strongly subordinate, $subjunct^3$, and hypotactic structures as only mildly subordinate, $subjunct^1$.

2.3 Abstract elements of style

The primitive elements of style are combined into patterns of *abstract elements* that describe general stylistic properties related to syntactic parallelism, structure nesting, and linear ordering. The abstract elements are defined as follows:

Homopoise: A sentence with interclausal coordination of syntactically similar components.

Heteropoise: A sentence in which one or more parenthetical components are syntactically 'detached' and dissimilar from the other components at the same level in the parse tree.[4]

Monoschematic: A sentence with a single main clause with simple phrasal subordination and no accompanying subordinate or coordinate clauses.

Centroschematic: A sentence with a central, dominant clause with one or more of the following optional features: complex phrasal subordination, initial dependent clauses, terminal dependent clauses.

Polyschematic: A sentence with more than one central, dominant clause and at least one dependent clause.

Resolution: A shift in stylistic effect that occurs at the end of a sentence and is a move from a relative discord to a stylistic concord.

Dissolution: A shift in stylistic effect that occurs at the end of a sentence and is a move from a relative concord to a stylistic discord.

The remaining abstract elements describe *concordant* or *discordant* stylistic effects in particular positions. The basic elements are *initial concord*, *medial concord*, and *final concord*, with a similar range of *discord* elements.

[2]Discord, in our view, is not necessarily 'bad'. Indeed, it is the strategic use of discord, deviation from the norm, that can give expressiveness to writing.

[3]These two complementary kinds of analysis are implicit in the work of most stylists and rhetoricians.

[4]A heteropoise can be *initial*, *medial*, or *final*, depending on the position of the parenthesis in the sentence.

2.4 Stylistic goals

As we have noted, the abstract elements are defined in terms of the lower-level primitive elements. The abstract elements are in turn used as the basis for the definition of higher-level *stylistic goals*. Stylistic goals can be organized along orthogonal dimensions. For example, a writer might try to be clear, or obscure, or make no effort either way. *Clarity* and *obscurity* are thus opposite ends of a stylistic dimension. Likewise, the goals of *concreteness* and *abstraction* form a dimension, and so do *staticness* and *dynamism*.

We adapted descriptions of stylistic goals from textbooks of style, such as [Kane 1983], and rewrote these descriptions in terms of our abstract elements. Clarity, for example, is characterized by *simple* monoschematic sentences, *centred* centroschematic sentences, and *parallel* homopoisal sentences. Concreteness is associated with sentences that *highlight* a particular component: these are our heteropoises and discords. And *staticness* is characteristic of *'fixed-form'* sentences in which there is little stylistic variation, that is, monoschematic or homopoisal sentences.

3 A stratified grammar of style

3.1 The style of the grammar

Our theory of syntactic style is now the basis for a grammar of style, which in turn will provide a specification for our stylistic analyzer. The hierarchical nature of the theory lends itself naturally to a stratified, context-free grammar. It is useful to think of the grammar as a means of recognizing a particular *style tree*, analogous to a syntactic parse tree. Just as a syntax tree is built up from individual words at the leaf nodes to a whole sentence at the root level, a style tree can be thought of as being built up from primitive elements at the leaf level to stylistic goals at the root. The syntax-tree analogy can be extended: we can consider the syntax-tree nodes to be annotated with stylistic terms, starting with the leaves and working up through the intermediate nodes to the root node.

To illustrate the structure of the grammar, we will present selected rules that build from simple syntactic components to full sentences.[5]

3.2 Level of primitive elements

3.2.1 Basic components

In the development of our theory of style, we were especially concerned with the relationship between style and the structure of the nominal group. As a consequence, a large number of the rules in our grammar involve definitions of premodification and postmodification. These definitions are built up from adjectivals, nouns, prepositional phrases, *etc.* One of the connective rules for postmodification defines conjunct^1 postmodification, which deals with the case of a prepositional phrase, a conjunctive element. In the hierarchic view, a prepositional phrase is classified as subjunct^3 postmodification, as it is an embedded element.

```
conjunct¹ postmodification ⟶
    prepositional phrase

subjunct³ postmodification ⟶
    prepositional phrase
```

We introduce the notion of a *transitional level* in the grammar, in order to clearly separate the levels of primitive elements and abstract elements. At this level, primitive elements are combined into transitional elements, which directly indicate the abstract elements of which they can be a part. For example, the rules below define the kinds of postmodification that can be used in building a monoschematic, centroschematic, or concordant sentence.

```
monoschematic postmodification ⟶
    subjunct⁰ postmodification
    subjunct³ postmodification and
    (nominal group or prepositional phrase)

centroschematic postmodification ⟶
    conjunctⁱ postmodification where 0≤ i ≤4
    subjunctⁱ postmodification where 0≤ i ≤3

concordant postmodification ⟶
    conjunctⁱ postmodification where 0≤ i ≤4
    subjunctⁱ postmodification where 0≤ i ≤3
```

3.2.2 Noun phrases

The various types of premodification and postmodification are combined into different kinds of noun phrases. In the examples below, we define the kinds of noun phrases that can be incorporated into monoschematic, centroschematic, and concordant sentences.

```
monoschematic noun phrase ⟶
    noun phrase with
        (monoschematic premodification and
        monoschematic postmodification)

centroschematic noun phrase ⟶
    noun phrase with
        (centroschematic premodification and
        centroschematic postmodification)

concordant noun phrase ⟶
    noun phrase with
        (concordant premodification and
        concordant postmodification)
```

[5]The rules are taken from Hoyt's [1993] full syntactic stylistic grammar of 240 rules, which is a revised and extended version of the preliminary grammar presented in [DiMarco and Hirst 1993]. The notation used in the grammar is explained in the Appendix to this paper.

3.2.3 Main clauses

In an analogous manner, the rules for the other major sentence components (prepositional phrases, complements, verb phrases, and dependent clauses) are built up from primitive elements to form transitional elements. The various types of majors, or main clauses, can then be defined from component transitional elements, as in the following examples.

monoschematic major $\longrightarrow$

  major *with*
    (monoschematic noun phrase *and*
    monoschematic verb phrase)

centroschematic major $\longrightarrow$

  major *with*
    (centroschematic noun phrase *and*
    centroschematic verb phrase)

concordant major $\longrightarrow$

  major *with*
    (concordant noun phrase *and*
    concordant verb phrase)

3.2.4 Complete sentences

Finally, we define rules for complete sentences, which consist of at least one main clause, with optional dependent clauses. Selected rules are as follows:

monoschematic complete $\longrightarrow$

  monoschematic major

centroschematic complete $\longrightarrow$

  (concordant clause)* centroschematic major
  (concordant clause)*

concordant complete $\longrightarrow$

  (concordant clause)* concordant major
  (concordant clause)*

initial concordant complete $\longrightarrow$

  concordant major (clause)*

  (concordant clause)* major (clause)*

3.3 Levels of abstract elements and stylistic goals

At the level of abstract elements, the various types of complete sentences are used to define stylistic terms such as monoschematic, centroschematic, and initial concord:

monoschematic $\longrightarrow$

  monoschematic complete

centroschematic $\longrightarrow$

  centroschematic complete

initial concord $\longrightarrow$

  initial concordant complete

Finally, at the top level, the abstract elements are used to define stylistic goals. For example, as we described in section 2.4, clarity would be defined by the following rule:

clarity $\longrightarrow$

  monoschematic

  centroschematic

  homopoise

3.4 An application of the grammar

The following short example illustrates the kind of analysis that the stylistic grammar can be used to produce for the sentence *True, posterity has been kind.*[6]

3.4.1 Primitive-element analysis

The sentence is concordant, for it consists of a concordant main clause, the major, with no subordinate clauses. It begins with a style disjunct, *true*, which is an elliptic adjectival and therefore considered to have a connective, concordant effect, even if used in the initial, parenthetical, position. After the initial disjunct, the sentence continues with the bare noun *posterity*, which, lacking both premodification and postmodification is a minimal, and therefore concordant, noun phrase. The sentence ends with the basic verb phrase *has been kind*, consisting of only the copula *been*, and the concordant, conjunct[1] adjective *kind*; this is an inherently concordant verb phrase.

The sentence is concordant from the hierarchic view as well, for it has the form of a concordant initial heteropoisal complete sentence. This indicates that the sentence begins with a parenthetical construction, which in this case is the disjunct, *true*, a superordinate adjectival. The bare noun *posterity*, lacking both premodification and postmodification, is a monoschematic noun phrase. The verb phrase *has been kind* is basic and therefore monoschematic.

3.4.2 Abstract-element analysis

In the connective view, the significant elements are initial and medial concords—the sentence is both monoschematic and trivially centroschematic. It is also an initial heteropoise.

In the hierarchic view, the sentence is centroschematic and an initial heteropoise. It is the initial disjunct, *true*, that introduces a superordinate effect; this feature makes the sentence slightly too complex to be monoschematic.

[6]The next six paragraphs have been adapted from [DiMarco and Hirst 1993].

3.4.3 Stylistic-goal analysis

The presence of the concords in the connective view, together with the connective and hierarchic centroschematic structures, give the sentence an effect of clarity. In a less obvious manner, the presence of an initial disjunct affects other stylistic goals. Because a superordinate, parenthetical, component is present, the sentence is a heteropoise and therefore considered to be concrete.

To summarize, this is a simple, clear sentence with the slight incongruity of an initial parenthesis to relieve its blandness.

4 A stratified stylistic analyzer

4.1 General design

Our theory of syntactic style is represented by a corresponding set of grammar rules that defines the relationship between syntactic structures and stylistic effects. Now, we will use this grammar of style as the specification for a stylistic analyzer, Asset, that will produce stylistic parses of input sentences. In designing Asset, we were influenced by the following considerations:

Evaluation of the theory: We viewed Asset as an essential tool for testing and evaluating our theory of style.

Parser independence: A syntax-based stylistic analysis of a sentence will obviously include a syntactic parse of the sentence. Thus, an ordinary parser is a necessary part of any stylistic analyzer. Our decision to make Asset totally independent of the parser was in part theoretical—Asset would not have to compromise theory because of limitations and/or methodology of the parser—and pragmatic—developing a parser from scratch was beyond the scope of our work.

This requirement meant that the sentence must be parsed before the stylistic analysis. This allows the substitution of parsers within the system with only the requirement that a module be created to transform the output of a particular parser into the specified format for Asset.

Modularity: Future work on the theory will include refinements to the abstract elements and transition elements, so the prospect of these revisions made modularity, good software engineering practice in any case, a necessity.

Efficiency: Asset must be reasonably efficient.

Independence from potential uses: The potential applications of a stylistic analyzer include intelligent computer-assisted language instruction (ICALI) and machine translation (MT). At the present state of development of ICALI and MT, it is impossible to know exactly which information and what representation would be most useful. This implied that, in addition to letting the user know which stylistic goal(s), if any, have been met, all stylistic information generated during the analysis must be part of the output of Asset.

[[[[[none], complement], [[[runs], lexical_verb], verb], verb_phrase], [[[[[[[none], postmodifier], [[[park], lexical_noun], noun], [[[[the], definite_article], adjectival], premodifier], nominal_group], [[in], preposition], prepositional_phrase], postmodifier], [[[man], lexical_noun], noun], [[[[the], definite_article], adjectival], premodifier], nominal_group], noun_phrase], major], complete]

Figure 1: Asset's input in its list-structure form.

The need to have all stylistic information available further implied that the analysis of one part of the sentence, *e.g.*, the noun phrase, cannot constrain that of another, *e.g.*, the verb phrase. To obtain some degree of efficiency, in spite of the lack of constraints on the analysis, a bottom-up, or leaf-to-root, approach is used. A syntax tree that parallels the syntactic organization of our grammar is the basic structure of Asset. This tree is represented as a list structure that describes a breadth-first, right-to-left traversal. Figure 1 shows the list structure that is the input to Asset for the simple sentence *The man in the park runs.*

The parser used in the development of Asset is Pundit[7] (Prolog UNderstands Integrated Text), chosen because of its fairly large syntactic coverage and its comprehensive treatment of conjunctions. These are necessary features for the analysis of stylistically interesting sentences. Pundit uses a *restrictive grammar*, written as a set of BNF (Backus-Naur Form) rules. Pundit's output consists of a syntactic tree in this BNF form; it is this output that is transformed into the parse tree input into Asset.

4.2 The representation of the grammar in ASSET

Asset's processing mechanism is data-driven, so that the grammar rules are represented declaratively in a database. Asset is essentially a 'tree-walker' that traverses the parse tree, annotating the nodes with stylistic information. The grammar rules have the form shown in Figure 2.[8]

As Asset walks through the parse tree, it uses the grammar representation to match the pattern of annotations currently recorded at a node. At each stage, Asset 'knows' what it needs to look for because of the consistency in the way the grammar is constructed: The grammar is *stratificational*,[9] so that elements at each level are composed from elements at the level below. As a consequence, Asset need only look at a small set of

[7]Pundit is a natural language understanding system developed by the Unisys Corporation.

[8]In Figures 2 and 6, the abstract-element terms *connective, hierarchic, monoschematic, centroschematic*, and *concordant* have been abbreviated to *conn, hier, mono, centro*, and *concord* respectively. The *te* affix indicates a transition-element analysis that is dependent on the corresponding transition-element analysis of the syntactic component's (*i.e.*, the current node's) children.

[9]The stratificational nature of our grammar was influenced by Sydney M. Lamb's work, in particular, *Outline of stratificational grammar*, Georgetown University Press, 1966.

```
postmodification(conn, prepositional_phrase,
                 conjunct1).
postmodification(hier, prepositional_phrase,
                 pp_subjunct3).

postmodification(conn_te, conjunct1,
                 [centro, concord]).
postmodification(hier_te, pp_subjunct3,
                 [mono, concord]).

nominal_group(postmodification(concord),
              [premodification(concord)], concord).
noun_phrase(nominal_group(centro), [], centro).

major(noun_phrase(concord), [verb_phrase(concord)],
      concord).
complete(major(centro), [], centro).

abstract_elements(complete(centro), [], centro).
stylistic_goals(abstract_elements(centro), [],
                clarity).
```

Figure 2: Sample Asset grammar rules

1: Transform the parser output into format specified for Asset (Transformation Module).
2: Annotate the input tree with primitive element classifications (Annotation Module).
3: Assign abstract elements to the input sentence (Abstract Element Module).
4: Assign stylistic goal(s) to the input sentence (Stylistic Goal Module).
5: Output the annotated tree structure.

Figure 3: The general algorithm for Asset

possible rules at each stage to decide on the next increment in the annotation of the stylistic parse tree.

4.3 The processing modules

The general algorithm for the Asset system is shown in Figure 3, along with an accompanying illustration of its architecture in Figure 4. The Transformation Module is responsible for changing Pundit's output into the form, as shown in Figure 1, specified for Asset.

The Annotation Module is responsible for the task of analyzing the style of the input sentence at the primitive-element and transition-element levels. The algorithm is shown in Figure 5. There are two submodules that annotate the nodes of the input tree with stylistic information:

Primitive Element Module (PEM): This module is responsible for analyzing the appropriate nodes by using the primitive-element layer of our computational theory. Each node is analyzed from both the connective and hierarchical viewpoints. The result of the analysis is a node annotated with the primitive stylistic descriptions: either a conjunct or an antijunct element and either a

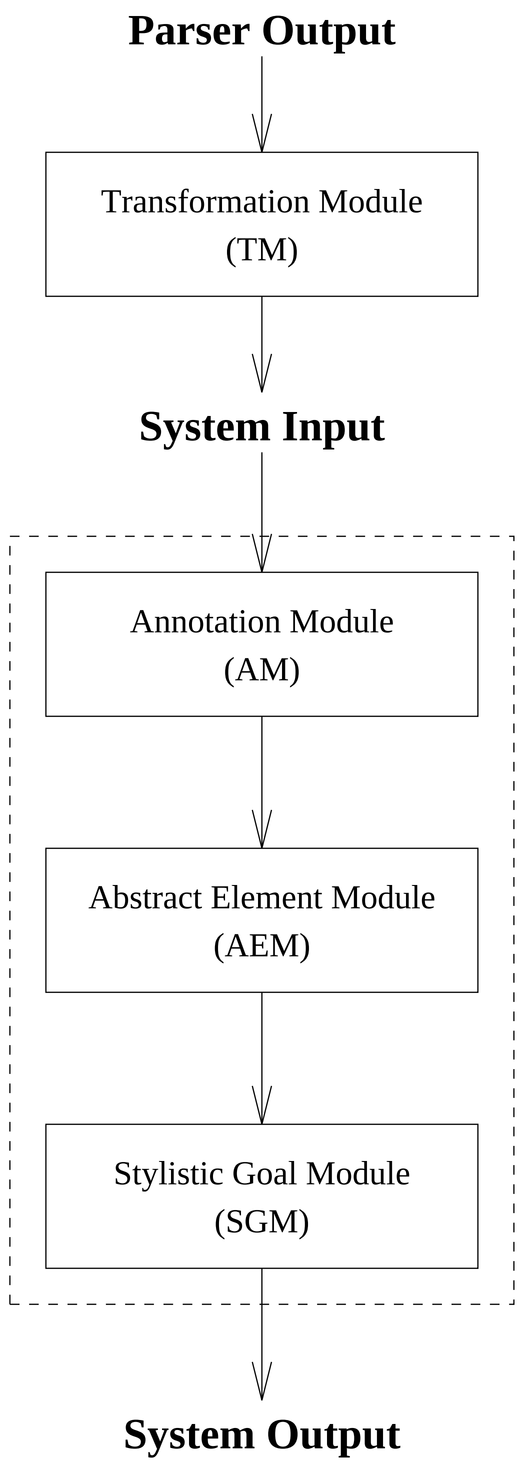

Figure 4: The overall architecture of Asset

```
1: Annotate 'near' leaf nodes with primitive-element
   classifications (PRIMITIVE ELEMENT MODULE).
     1a. Analyze from the connective viewpoint
         (CONNECTIVE MODULE).
     1b. Analyze from the hierarchical viewpoint
         (HIERARCHICAL MODULE).
2: Annotate the rest of the nodes with transition-
   element classifications (TRANSITION ELEMENT MODULE).
```

Figure 5: The algorithm for the Annotation Module.

subjunct or a superjunct element.

Transition Element Module (TEM): This module takes the parse tree, previously annotated by the PEM, and annotates the rest of the nodes with abstract-element terms. The TEM uses information provided by the primitive-element classification of nodes lower in the parse tree.

It should be noted that the PEM and the TEM do not work sequentially: because of the bottom-up processing, calls to the TEM occur whenever the PEM has annotated a sufficient number of nodes lower in the parse tree. Thus, calls to the PEM and the TEM are interleaved with each other.

After the primitive-element and transition-element analyses have been completed, the input sentence is then classified in terms of the abstract elements and the stylistic goals. A fully annotated parse tree is input to the Abstract Element Module, which then adds abstract element information to the structure and passes it on to the Stylistic Goal Module. Once the stylistic goals have been determined, the output structure is complete. Figure 6 shows all the stylistic information contained in the output structure for the sentence *The man in the park runs.*

5 Conclusion

Computational stylistic analysis and generation should ideally be components of any sophisticated natural language processing system. However, the difficulties of representing stylistic knowledge in a form amenable to computational implementation meant that only *ad hoc* approaches had previously been attempted in dealing with matters of style in NLP systems.

We have shown how a formal theory of style can be represented by a multi-level grammar that describes the relationship between low-level syntactic structures and high-level stylistic goals. In turn, this grammar has been used as the specification for an implementation, ASSET, which produces stylistic analyses at multiple levels of abstraction. We can foresee such a general-purpose stylistic analyzer used as a component of NLP systems, such as for second-language instruction or machine translation, to produce additional information that contributes to the full understanding of the implicit meaning of a text.

Acknowledgements

We would like to thank Graeme Hirst for helpfully reading earlier drafts of this paper and providing very useful advice. We acknowledge the financial support of the Natural Sciences and Engineering Research Council of Canada and the Information Technology Research Centre.

Appendix: Notes on terminology

At all levels of the grammar, the left-hand side of each rule identifies what is being defined, and the right-hand side lists one or more alternative realizations, one per line.

In the grammar, we use various shorthand notations to simplify the presentation of the rules. However, these abbreviated forms can be expanded into standard context-free grammar rules. The shorthand notations are as follows; they are illustrated by particular examples, but are intended for general use:

1. adjectival ⟶ intensifier adjective

   The juxtaposition of terms on the right-hand side of a rule indicates a concatenation of instances of these terms. For example, the rule above allows the intensifier *very* to be followed by the adjective *happy* to form an adjectival, *very happy*.

2. adjectival ⟶ (intensifier)* adjective

   The Kleene star indicates zero or more occurrences of the form within parentheses.

3. noun phrase *with* centroschematic postmodification

   Where a rule has several alternatives, this shorthand notation using *with* abbreviates a long sequence of alternatives (here, the different types of centroschematic postmodification).

4. noun phrase *with*
   (centroschematic premodification *and*
   centroschematic postmodification)

   And indicates that all conditions on the right-hand side of a rule must simultaneously be satisfied by a single constituent.

5. noun phrase *with*
   postmodification *and*
   (nominal group *or* prepositional phrase)

   Or indicates that any one of the conditions on the right-hand side of a rule must be satisfied.

References

DiMarco, Chrysanne and Hirst, Graeme. "A computational theory of goal-directed style in syntax." *Computational Linguistics*, **19**(3), 448–497, September 1993.

```
stylistic_goals(clarity,staticness)

abstract_elements(mono,concord,initial_concord,medial_concord, final_concord)

complete([mono,concord,initial_concord,medial_concord, final_concord])
    major([mono,concord])

    noun_phrase(([centro,mono,concord])
      nominal_group(([centro,mono,concord])

          premodification([conjunct1],[subjunct2], [centro,mono,concord])
            adjectival([conjunct1],[subjunct2])
              definite_article(the)

          noun([conjunct0])
              lexical_noun(man)

          postmodification([conjunct1],[subjunct3], [centro,mono,concord])
            prepositional_phrase([centro,mono,concord])

              preposition(in)
              nominal_group([centro,mono,concord])

                premodification([conjunct1],[subjunct2], [centro,mono,concord])
                  adjectival([conjunct1],[subjunct2])
                    definite_article(the)

                noun([conjunct0])
                     lexical_noun(park)

                postmodification([conjunct0],[subjunct0], [centro,mono,concord])
                  postmodification(none)

    verb_phrase([mono,concord])
      verb(runs)

      complement([mono,concord])
            complement(none)
```

Figure 6: An example of ASSET's output.

Green, Stephen J. *A functional theory of style for natural language generation.* Master's thesis, Department of Computer Science, University of Waterloo, 1992. [University of Waterloo Faculty of Mathematics Technical Report CS-92-48].

Halliday, M.A.K. *Introduction to functional grammar.* Edward Arnold, London, 1985.

Halliday, M.A.K. and Hasan, Ruqaiya. *Cohesion in English.* Longman Group Limited, London, 1976.

Hoyt, Patricia A. *A goal-directed functionally-based stylistic analyzer.* Master's thesis, Department of Computer Science, University of Waterloo, 1993. [University of Waterloo Faculty of Mathematics Technical Report CS-93-48].

Quirk, Randolph, Greenbaum, Sidney, Leech, Geoffrey, and Svartvik, Jan. *A comprehensive grammar of the English language.* Longman Group Limited, 1985.