Phrase Projectivity in Antigone

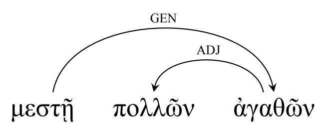
Jonathan Sterling

April 19, 2013

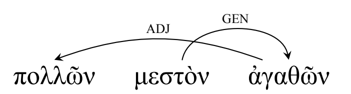
1 Introduction

1.1 Dependency Trees and Projectivity

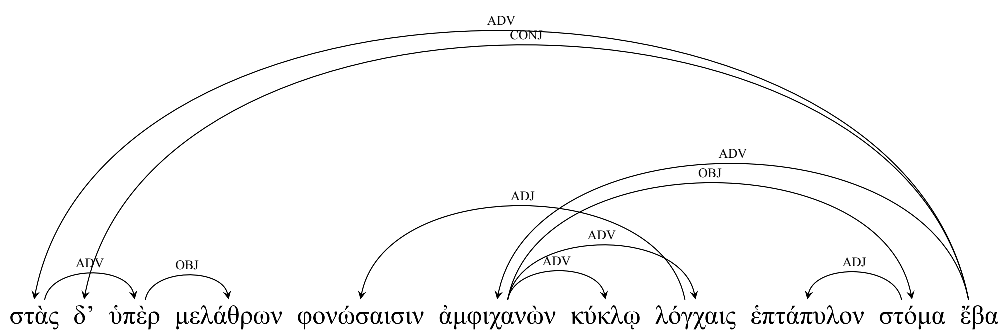
A dependency tree encodes the head-dependent relation for a string of words, where arcs are drawn from heads to their dependents. We consider a phrase *projective* when these arcs do not cross each other, and *discontinuous* inasmuch as any of the arcs do cross. Figure 1 illustrates the various kinds of projectivity violations that may occur.



(a) "Full of plentiful supplies" (Xenophon, *Anabasis* 3.5.1).



(b) "Full of many good things" (Plato, *Laws* 906a).



(c) "And he stood over the rooftops, gaped in a circle with murderous spears around the seven-gated mouth, and left." (Sophocles, *Antigone* 117–120).

Figure 1: A dependency path wrapping around itself is a projectivity violation, as in (b); interlacing adjacent phrases also violate projectivity, as in (c). Examples (a–b) drawn from Devine & Stephens.

$$\textbf{data}\ \mathsf{Tree}\ \alpha = \alpha \curvearrowright [\mathsf{Tree}\ \alpha]$$

$$\begin{aligned}&\mathsf{getLabel} :: \mathsf{Tree}\ \alpha \to \alpha\\&\mathsf{getLabel}\ (l \curvearrowright \_) = l\end{aligned}$$

$$\begin{aligned}&\textbf{type}\ \mathsf{Edge}\ \alpha \quad = (\alpha, \alpha)\\&\textbf{data}\ \mathsf{Range}\ \alpha = \alpha \leftrightarrow \alpha \mid \mathsf{R}_{\emptyset}\\&\textbf{data}\ \mathsf{RangeState}\ \alpha = \mathsf{Integer} \lhd \mathsf{Range}\ \alpha\end{aligned}$$

$$\begin{aligned}&\mathsf{getRange} :: \mathsf{RangeState}\ \alpha \to \mathsf{Range}\ \alpha\\&\mathsf{getRange}\ (\_ \lhd r) = r\end{aligned}$$

$$\begin{aligned}&\mathsf{getViolations} :: \mathsf{RangeState}\ \alpha \to \mathsf{Integer}\\&\mathsf{getViolations}\ (vs \lhd \_) = vs\end{aligned}$$

A Monoid is an algebraic structure which has a zero $\mathcal{E}$ and a binary operation $\cdot \oplus \cdot$, and which satisfies some laws:

$$\begin{aligned}&\textbf{class}\ \mathsf{Monoid}\ \alpha\ \textbf{where}\\&\quad \mathcal{E} :: \alpha\\&\quad \cdot \oplus \cdot :: \alpha \to \alpha \to \alpha\\&\quad \mathsf{associativity} :: l \oplus (c \oplus r) \equiv (l \oplus c) \oplus r\\&\quad \mathsf{leftIdentity} :: l \oplus \mathcal{E} \equiv l\\&\quad \mathsf{rightIdentity} :: \mathcal{E} \oplus l \equiv l\end{aligned}$$

$$\begin{aligned}&\textbf{instance}\ \mathsf{Ord}\ \alpha \Rightarrow \mathsf{Monoid}\ (\mathsf{Range}\ \alpha)\ \textbf{where}\\&\quad \mathcal{E} = \mathsf{R}_{\emptyset}\\&\quad (x \leftrightarrow y) \oplus (u \leftrightarrow v) = \mathsf{rangeFrom}\ [x, y, u, v]\\&\quad \mathsf{R}_{\emptyset} \oplus xy = xy\\&\quad xy \oplus \mathsf{R}_{\emptyset} = xy\end{aligned}$$

$$\begin{aligned}&\textbf{instance}\ (\mathsf{Num}\ \alpha, \mathsf{Ord}\ \alpha) \Rightarrow \mathsf{Monoid}\ (\mathsf{RangeState}\ \alpha)\ \textbf{where}\\&\quad \mathcal{E} = 0 \lhd \mathcal{E}\\&\quad (i \lhd xy) \oplus (j \lhd uv) = count \lhd (xy \oplus uv)\ \textbf{where}\\&\qquad count = \textbf{if}\ \mathsf{rangesIntersect}\ xy\ uv\\&\qquad\quad \textbf{then}\ i + j + 1\\&\qquad\quad \textbf{else}\ i + j\end{aligned}$$

$$\begin{aligned}&\mathsf{rangesIntersect} :: \mathsf{Ord}\ \alpha \Rightarrow \mathsf{Range}\ \alpha \to \mathsf{Range}\ \alpha \to \mathsf{Bool}\\&\mathsf{rangesIntersect}\ (x \leftrightarrow y)\ (u \leftrightarrow v) =\end{aligned}$$

$$
\begin{array}{l}
\quad \neg\,((x < u \wedge y < u) \vee (u < v \wedge v < x)) \\
\mathsf{rangesIntersect}\ \_\ \_ = \mathsf{False}
\end{array}
$$

$$
\begin{array}{l}
\mathsf{rangeFrom} :: (\mathsf{Foldable}\ \phi, \mathsf{Ord}\ \alpha) \Rightarrow \phi\ \alpha \to \mathsf{Range}\ \alpha \\
\mathsf{rangeFrom}\ xs = \mathsf{minimum}\ xs \leftrightarrow \mathsf{maximum}\ xs
\end{array}
$$

$$
\begin{array}{l}
\mathsf{analyzePath} :: (\mathsf{Num}\ \alpha, \mathsf{Ord}\ \alpha) \Rightarrow [\alpha] \to \mathsf{RangeState}\ \alpha \\
\mathsf{analyzePath}\ path = \mathsf{foldl}\ op\ \mathcal{E}\ (reverse\ path)\ \textbf{where} \\
\quad op\ (c \lhd r)\ i = \textbf{if}\ \mathsf{inRange}\ r\ i \\
\qquad \textbf{then}\ (c+1) \lhd r \\
\qquad \textbf{else}\ c \lhd (\mathsf{extend}\ r\ i)
\end{array}
$$

$$
\begin{array}{l}
\mathsf{analyzeTree} :: (\mathsf{Num}\ \alpha, \mathsf{Ord}\ \alpha) \Rightarrow \mathsf{Tree}\ \alpha \to \mathsf{Tree}\ (\mathsf{RangeState}\ \alpha) \\
\mathsf{analyzeTree}\ tree = \\
\quad \textbf{case}\ \mathsf{treeOrPath}\ tree\ \textbf{of} \\
\qquad \mathsf{Left}\ (i \curvearrowright ts) \to c' \lhd \mathsf{extend}\ r\ i \curvearrowright children\ \textbf{where} \\
\qquad\quad children = \mathsf{analyzeTree}\ \langle\$\rangle\ ts \\
\qquad\quad c \lhd r = \mathsf{fold}\ (\mathsf{getLabel}\ \langle\$\rangle\ children) \\
\qquad\quad c' = c + (genericLength\ (filter\ (\lambda r' \to \mathsf{inRange}\ r'\ i)\ (\mathsf{getRange} \circ \mathsf{getLabel}\ \langle\$\rangle\ children))) \\
\qquad \mathsf{Right}\ path \to \mathsf{analyzePath}\ path \curvearrowright [\,]
\end{array}
$$

$$
\begin{array}{l}
\mathsf{treeOrPath} :: \mathsf{Tree}\ \alpha \to \mathsf{Either}\ (\mathsf{Tree}\ \alpha)\ [\alpha] \\
\mathsf{treeOrPath}\ (i \curvearrowright [\,]) = \mathsf{Right}\ [i] \\
\mathsf{treeOrPath}\ (i \curvearrowright [x]) = (i{:})\ \langle\$\rangle\ \mathsf{treeOrPath}\ x \\
\mathsf{treeOrPath}\ t \qquad\quad = \mathsf{Left}\ t
\end{array}
$$

$$
\begin{array}{l}
\mathsf{extend} :: \mathsf{Ord}\ \alpha \Rightarrow \mathsf{Range}\ \alpha \to \alpha \to \mathsf{Range}\ \alpha \\
\mathsf{extend}\ (x \leftrightarrow y)\ z = \mathsf{rangeFrom}\ [x, y, z] \\
\mathsf{extend}\ \mathsf{R}_{\emptyset}\ z = z \leftrightarrow z
\end{array}
$$

$$
\begin{array}{l}
\mathsf{inRange} :: \mathsf{Ord}\ \alpha \Rightarrow \mathsf{Range}\ \alpha \to \alpha \to \mathsf{Bool} \\
\mathsf{inRange}\ (x \leftrightarrow y)\ z = z > x \wedge z < y \\
\mathsf{inRange}\ \mathsf{R}_{\emptyset}\ \_ = \mathsf{False}
\end{array}
$$

$$
\begin{array}{l}
\mathsf{maximalPoint} :: \mathsf{Eq}\ \alpha \Rightarrow [\mathsf{Edge}\ \alpha] \to \mathsf{Maybe}\ \alpha \\
\mathsf{maximalPoint}\ es = \\
\quad \mathsf{find}\ (\lambda x \to x \notin \mathsf{snd}\ \langle\$\rangle\ es)\ (\mathsf{fst}\ \langle\$\rangle\ es)
\end{array}
$$