Carl Pollard

**A Parallel Derivational Architecture
for the Syntax-Semantics Interface**

It is commonplace to bifurcate the universe of grammar architectures into (a) the syntactocentric, derivational frameworks, and (b) the parallel, constraint-based frameworks, The former include most avatars of transformational grammar (TG), including EST/GB and the Minimalist Program (MP), as well as most forms of categorial grammar (CG), while the latter include LFG, HPSG, and Simpler Syntax However, except for Lecomte and Retoré (2002), it seems not to be generally recognized that a parallel derivational (PD) architecture is also possible. Here we advocate, and sketch the outlines of, Convergent Grammar (CVG), a new PD framework intended to combine the virtues of HPSG's parallelism and CG's proof-theoretic brand of derivationalism. Perhaps surprisingly, the result preserves much of what was right about GB theory. Moroever we argue that CVG provides a more promising direction for fixing what was wrong with GB than the MP does, indeed it is 'more minimal than Minimalism' since it manages without feature checking (or even features), Spell-out(s), or LF.

The full paper will illustrate the theoretical points with analyses of examples of each of these phenomena: topicalization, tough-movement, violins and sonatas, wh-in situ, QR, pied piping, and multiple wh.

Overall, a CVG provides three independent components that generate, respectively, candidate semantic, syntactic, and phonological derivations (but we ignore the third here), each of which is a natural-deduction proof (with a different logic for each component) employing variable contexts 'to the left of the turnstile' to track undischarged hypotheses (semantic variables—including in-situ elements to be bound by 'delayed' semantic operators—in the semantic derivation; and traces in the syntactic derivation). Additionally, an overarching **sign** component provides a recursive specification of which (phonology-)syntax-semantics tuples of derivations belong to the language in question; here the base clauses of the specification are the lexical entries and the recursion clauses are the grammar rules (examples of each are given below).

The logic for the semantic derivations is just positive intuitionistic propositional logic, so that the derivations themselves can be identified (via the Curry-Howard correspondence) with ordinary lambda terms, which can then be model-theoretically interpreted (if desired) in the usual (Montagovian) fashion. Here the types/formulas are just semantic types.

The logic for the syntactic derivations is a multimodal resource sensitive logic, with the formulas being the syntactic categories. The connectives include include several modes of implications. Among these are some corresponding to grammatical functions like subject and complement, which have an elimination rule but no introduction rule; these play the role of application/Modus Ponens in CG, valence features in HPSG, or Merge in TG. Additionally there is a mode of implication with an introduction rule but no elimination rule, corresponding to HPSG's SLASH, TG's Move, or Hypothetical Proof in CG (e.g. Moortgat's $\uparrow$ connective). This logic has its own system of Curry-Howard proof terms corresponding to derivations, which bear a strong resemblance to GB-style labelled bracketings, with the variables being traces, constants being words, and lambdas being empty operators that bind traces.

But unlike GB (and like MP), the Merges need not be done before the Moves, for the simple reason that in natural deduction proof trees, the Modus Ponens nodes need not be lower than the Hypothetical Proof nodes. Moreover, Strict Cyclicity is observed automatically, since there are no rules of natural deduction that go back and change an earlier part of the proof. Likewise there is no need for a stipulation (cf. Hardt 2006) that 'meaning representations are constructed as early as possible during a bottom-up derivation and the resulting derivations cannot be revised later', since the sign component recursively builds up paired natural-deduction proofs bottom up and in parallel.

**Some CVG Lexical Entries**

$\vdash \mathsf{John}_{\mathsf{nom}}, \mathsf{John'} : \mathrm{Nom}, \mathrm{e} \dashv$

$\vdash \mathsf{liked}, \lambda_y \lambda_x \mathsf{like'}(x, y)$: $\mathrm{Acc} \multimap_{\mathrm{C}} (\mathrm{Nom} \multimap_{\mathrm{SU}} \mathrm{Fin}), \mathrm{e} \rightarrow (\mathrm{e} \rightarrow \mathrm{t}) \dashv$

**Subject Merge**

If $\Gamma \vdash a, c : A, C \dashv$ and $\Gamma' \vdash f, v : A \multimap_{\mathrm{SU}} B, C \rightarrow D \dashv$

then $\Gamma; \Gamma' \vdash (^{\mathrm{SU}}\ a\ f), v(c) : B, D \dashv$

**Complement Merge**

If $\Gamma \vdash f, v : A \multimap_{\mathrm{C}} B, C \rightarrow D \dashv$ and $\Gamma' \vdash a, c : A, C \dashv$

then $\Gamma; \Gamma' \vdash (f\ a\ ^{\mathrm{C}}), v(c) : B, D \dashv$

**Trace**

$t, x : A, B \vdash t, x : A, B \dashv$

**Finite Move**

If $t, x : A, B; \Gamma \vdash s, p : \mathrm{Fin}, \mathrm{t} \dashv$ then $\Gamma \vdash \lambda_t^{\mathrm{SL}} s, \lambda_x p : A \multimap_{\mathrm{SL}} \mathrm{Fin}, B \rightarrow \mathrm{t} \dashv$

**Topicalization**

If $\Gamma \vdash a, b : A, B \dashv$ and $\Gamma' \vdash c, d : A \multimap_{\mathrm{SL}} \mathrm{Fin}, B \rightarrow \mathrm{t} \dashv$

then $\Gamma; \Gamma' \vdash \tau(a, c), d(b) : \mathrm{Top}, \mathrm{t} \dashv$

**A Sign Licensed by this CVG**

$\vdash \tau(\mathsf{Mary}_{\mathsf{acc}}, \lambda_t^{\mathrm{SL}}(^{\mathrm{SU}}\ \mathsf{John}_{\mathsf{nom}}\ (\mathsf{liked}\ t\ ^{\mathrm{C}}))), \lambda_x(\mathsf{like'}(\mathsf{John'}, x))(\mathsf{Mary'}) : \mathrm{Top}, \mathrm{t} \dashv$