A remark on code-switching under the Equivalence Constraint

Jasper Kyle Catapang

21 November 2026

Bilingual speakers switch languages in the middle of a sentence, and they do not do it just anywhere. Poplack's (1980) Equivalence Constraint is the oldest precise guess about where: a switch is allowed only at a point where the word orders of the two languages agree. I work on code-switched text, mostly Tagalog and English, and I have used that constraint the way most of us in NLP do, as a recipe for generating or filtering data, without asking what set of sentences it actually defines.

This note answers that question for one precise reading of the constraint. If the two languages are modelled by congruent context-free grammars, and a switch is allowed exactly where no word alignment crosses it, then the set of code-switched sentences is itself context-free. There is an explicit grammar for it, about as large as the two grammars put together, and one can read off from the rules alone where a switch may fall. The central statement, which I call the EC Closure Theorem (Theorem 1) and which says that this grammar generates exactly the code-switched sentences, is checked from the definitions up by the Lean proof assistant.[1]

I should be plain about the scale. This is a small theorem. It is not a famous open problem, and part of it was already in Sankoff's hands in 1998 (Sankoff 1998a). I am writing it up because of how it came about, and because of what it says about the data I work with.

Yesterday the news was a claimed proof of the Riemann Hypothesis whose informal write-up runs to a quarter of a million pages. My reaction was a wish for the opposite: results small enough to hold in the head, with proofs a person can follow, and still checked by machine. I had a question of that size from my own field, and I spent this afternoon on it in a tête-à-tête with Claude (Fable 5.2). It was a fruitful one.

It is obvious to me that I should say who did what.

The question is mine, and it is an old one for me. Years of working with code-switched text had taught me that the constraint has competing readings, that our generation pipelines quietly use the alignment one, and that nobody has said what language that reading defines. The asymmetric alternative, in which one language supplies the frame and the other is inserted into it, goes back to Joshi (1982) and is best known as the Matrix Language model of Myers-Scotton (1993). I also came in with the literature in my head. Before any proof was attempted I knew that the rule-level criterion is Sankoff's (1998a), and that grammars for constrained switching exist already, in Sankoff and Poplack (1981) and Sankoff and Mainville (1986). So I knew which part was open, and I chose which reading to formalize on purpose and not by accident of what was easiest to prove.

The idea of the proof is mine too, such as it is. If you are used to synchronous grammars you see quickly that the global equivalence points are fixed by the permutations in the rules, and that they survive only below positions that every rule above keeps in place. I sketched that on a whiteboard;

[1] Files: `ECClosure.lean` (the proof), `ec_grammar_test.py`, `ec_exhaustive.py`, `ec_check.py`, `ec_taglish_check.py` (tests). All five are in `ec_closure_files.zip`, which is posted with a short plain-language version of this note at `https://jaspercatapang.github.io/notes/where-bilingual-speakers-can-switch/`.

it became the block lemma. The Duke de Lorraine sentence that separates the two readings was my example, as was the insistence on looking at real Taglish clauses and not only at a toy subject–predicate swap. And the judgement calls were mine: which of the congruence assumptions are harmless and which a linguist would reject outright, what is worth claiming, against whom, and for which readers.

Claude did the rest, and the rest is most of the pages. It turned the sketch into a complete proof, which means it did the cases a whiteboard skips: a cut that is a boundary in one language and falls inside a child in the other, why empty yields have to be excluded, what happens when a rule repeats a symbol. It wrote the Lean, all 1,659 lines of it: it chose representations on which induction goes through, proved the lemmas about lists and permutations that the core library lacks, and built the layer that gets from trees to grammars. It wrote the exhaustive tree check and the end-to-end grammar comparison and ran them, in minutes. It drafted and typeset the formal sections, the table that maps each statement to its Lean name, and the bookkeeping of what is and is not machine-checked. And it was a second pair of eyes on the literature: it found that Bhat et al. (2016, end of §2.3) remark that their Equivalence Constraint model can also be stated as a context-free grammar, and give neither the grammar nor a proof, and it checked recent work for overlap.

Put shortly, I knew what to ask and roughly why the answer was true. Claude made it true in every case, and then made a machine agree.

What struck me most was not speed. It was that the errors got caught, and the important one was mine. The definition of an "aligned" node that I wrote on the whiteboard was too weak. It is the kind of slip one makes with full confidence, and it was found only when the argument was written out in full. The formal proof then left no room for another. I would like more of my field to work this way.

1 What I should say plainly

- **What is proved.** For every pair of congruent context-free grammars, the language of the constructed grammar G_{EC} equals the set L_{EC} of sentences obtained by switching only at equivalence points (the equality in Theorem 1; context-freeness follows). Lean accepts this with no unproved step, using only its three standard axioms.

- **What is not machine-checked.** That G_{EC} is of linear size, and that the left-to-right scan of Section 5 computes the blocks. The first is a count over the construction; the second is tested but not proved formally (Section 8).

- **What is not new.** The setting of congruent grammars and the rule-level notion of equivalence point are Sankoff's. Grammars for equivalence-constrained switching go back to Sankoff and Poplack (1981) and Sankoff and Mainville (1986), so context-freeness of constrained switching in some form is older than this note (Section 9).

- **What seems to be new.** The alignment reading stated precisely, the block lemma that ties it to Sankoff's rule-level criterion, the exact equality with an explicit grammar, and the formal proof.

- **The idealization.** Congruent grammars assume the two languages have the same rules up to the order of the children and a word-for-word lexicon. Real language pairs satisfy neither. The theorem is about the model.

- **Two errors along the way.** My first definition of an "aligned" node was too weak; the definition and its counterexample are recorded in Remark 3. And a bound from an unrelated problem was quoted in passing as the state of the art: that greedy byte-pair encoding achieves at least one third of the optimal compression utility (Kozma and Voderholzer 2024), with the best known upper bound given as 0.625. Both figures were out of date. The upper bound had been lowered to $(1 - e^{-2})/2 \approx 0.432$ in the authors' own revision of 30 October 2026, and the lower bound had been raised to 0.391 by Baliga, Xie and Byju (arXiv version of 9 November 2026), less than two weeks before this note. Both were corrected in the course of the work; neither touches the theorem as stated here.

2 What it means for my own work

Generating code-switched text. Synthetic code-switched data is commonly produced by aligning a sentence with its translation and switching only where no alignment link crosses the switch. That is exactly the reading formalized here. The theorem says that this procedure, taken over all sentences of a grammar pair, yields a context-free language. So membership can be decided by ordinary parsing in cubic time, and valid sentences can be enumerated or sampled directly from a grammar instead of being generated freely and filtered afterwards.

Tagalog and English. Here the news is less comfortable. Tagalog is predicate-initial where English is subject-initial. The block lemma says that a switch point survives only if every rule above it keeps that position fixed, so one reordering high in the clause removes every switch point beneath the constituents it reorders. With a binary subject–predicate clause that is all of them (Example 1); with a flat verb–actor–object clause the switches in and before the object survive (Example 2). Under this reading the constraint licenses very little of the switching inside clauses that Taglish speakers produce all the time. This is the familiar objection that the Equivalence Constraint undergenerates for typologically distant pairs. What changes is that it is now something one can compute: for a given grammar pair, G_{EC} says exactly how little is allowed. Example 2 does this for a small Tagalog–English fragment.

The constraint we implement is not the one linguists stated. The alignment reading is strictly stronger than Sankoff's. His own example, *The original Duke | de Lorraine | had 10 000 men*, is fine in his model and excluded under the alignment reading (Example 3). A system that "implements the Equivalence Constraint" by forbidding crossing alignments therefore implements something narrower than what the linguistic literature formalized. That gap can be measured on real Taglish data, and it is the experiment this note points to.

Evaluating models. The grammar also gives a clean instrument. Minimal pairs that differ only in whether the switch falls at an equivalence point can be generated from G_{EC}, to test whether a language model, or a classifier for code-switched content, is sensitive to where a switch occurs and not merely to the fact that one occurs.

3 How the afternoon went

A rough timeline of 21 November, Tokyo time, reconstructed from the conversation. "I" is the author throughout.

1. 15:36. I open with the question, not with a request for problems. Generation pipelines forbid crossing alignments, Sankoff's constraint is local to a rule, and nobody has said what language either one defines.

2. 15:38. I state the prior work from memory: the rule-level criterion is Sankoff (1998a), and grammars for constrained switching exist in Sankoff and Poplack (1981) and Sankoff and Mainville (1986). I narrow the open part to the alignment reading and its relation to theirs.

3. 15:42. I fix the order of work: the rule-level criterion first, because it is the one linguists stated and the global one is built from it; the alignment reading second, because it is the one pipelines run. Only the criterion of the local reading is used in what follows; the language it defines is left open (Section 10). I also sort the idealization. Unambiguity and distinct right-hand-side symbols, which Sankoff assumes, are dropped as unnecessary; the rule bijection and the word-for-word lexicon are kept and flagged as what a linguist would reject.

4. 15:47. My whiteboard sketch: the global equivalence points are fixed by the rule permutations, and they survive only below positions that every rule keeps fixed. The Duke de Lorraine sentence is the example that separates the two readings.

5. 15:52. Claude writes the sketch out in full. My definition of "aligned" turns out to be too weak and is corrected; the counterexample is the one in Remark 3.

6. 15:58. Claude writes and runs the randomized and exhaustive tests.

7. 16:00. Claude takes a second look at the literature. Bhat et al. (2016, §2.3) state that the Equivalence Constraint model can be written as a context-free grammar, without a construction or a proof. Their check is the rule-level one; Theorem 1 supplies a construction and a proof for the alignment reading. A check of recent work finds no overlap.

8. 16:02–16:50. Claude writes the Lean, trees first and then grammars. In parallel I write a Tagalog–English grammar fragment and mark which Taglish switches each reading licenses.

9. 16:55. I compare G_{EC} with the 1981 and 1986 grammars, and decide what is claimed, against whom, and for which venue.

The rest of this note is the formal part.

4 Setup

Definition 1 (Congruent grammars)**.** Let $G_1 = (N, \Sigma_1, R_1, S)$ and $G_2 = (N, \Sigma_2, R_2, S)$ be context-free grammars with the same nonterminals, $\Sigma_1 \cap \Sigma_2 = \emptyset$, and no ε-rules. Rules are either *lexical*, $A \to w$ with w a terminal, or *structural*, $A \to X_1 \cdots X_m$ with $m \geq 1$ and all $X_i \in N$. The grammars are *congruent* if there is a bijection $h : R_1 \to R_2$ such that

- each lexical rule $A \to w$ is sent to a lexical rule $A \to w'$ (a translation), and
- each structural rule $r = A \to X_1 \cdots X_m$ is sent to $A \to X_{\pi_r(1)} \cdots X_{\pi_r(m)}$ for a permutation π_r of $\{1, \ldots, m\}$.

So $\pi_r(p)$ is the language-1 index of the child that stands in position p in language 2. The permutation π_r is part of the data; when some X_i repeat, the two right-hand sides alone do not determine it. The size $|G|$ of a grammar is the total length of its rules.

A derivation tree t of G_1 therefore also determines a derivation tree of G_2: keep the tree, reorder the children of every node by its π_r, and translate the leaves. A *leaf* is thus a position in the tree, and it carries one word in each language. Write $\mathrm{lin}_1(t)$ and $\mathrm{lin}_2(t)$ for the sequence of leaves of t in language-1 and language-2 order. Both are orderings of the same set of n leaves. For a node v let $Y(v)$ be its set of leaves and $\mathrm{Pre}_a(v)$ the set of leaves that precede all of $Y(v)$ in $\mathrm{lin}_a(t)$. Since $Y(v)$ is contiguous in both orders, $\mathrm{Pre}_a(v)$ is a prefix of $\mathrm{lin}_a(t)$. Because there are no ε-rules, every $Y(v)$ is non-empty.

Definition 2 (Equivalence point)**.** Let $P_a(j)$ be the set of the first j leaves of $\mathrm{lin}_a(t)$. A cut $j \in \{0, \ldots, n\}$ is an *equivalence point* of t if $P_1(j) = P_2(j)$. Write $\mathrm{EP}(t)$ for the set of equivalence points; always $0, n \in \mathrm{EP}(t)$.

Equivalently, no leaf lies on different sides of the cut in the two languages. Since each leaf is aligned with itself (its word with its translation), this says that no word alignment crosses the switch.

Definition 3 (The language L_{EC})**.** Let $0 = j_0 < j_1 < \cdots < j_q = n$ be the elements of $\mathrm{EP}(t)$. Between consecutive equivalence points the two linearizations contain the same leaves, in the same positions $j_{i-1} + 1, \ldots, j_i$. A *realization* of t chooses a language $a_i \in \{1, 2\}$ for each i and outputs, for $i = 1, \ldots, q$ in turn, the leaves in positions $j_{i-1}+1, \ldots, j_i$ of $\mathrm{lin}_{a_i}(t)$, each as its word in language a_i. $L_{\mathrm{EC}}(G_1, G_2, h)$ is the set of all realizations of all derivation trees.

Remark 1. Switching only at a *subset* of the equivalence points gives the same language, because a realization is free to choose the same language on both sides of an equivalence point, and two adjacent pieces realized in the same language concatenate to the corresponding stretch of that language's linearization. Choosing all a_i equal gives the monolingual sentences, so $L(G_1) \cup L(G_2) \subseteq L_{\mathrm{EC}}$.

Remark 2 (Chunk form). By the previous remark, z is a realization of t iff one can write $\mathrm{lin}_1(t) = x_1 x_2 \cdots x_q$ and $\mathrm{lin}_2(t) = y_1 y_2 \cdots y_q$, for some q, so that each chunk y_i is a rearrangement of x_i, and z is obtained by replacing each pair (x_i, y_i) by either x_i in language 1 or y_i in language 2. Indeed, the cut after $x_1 \cdots x_i$ is an equivalence point exactly when $x_1 \cdots x_i$ and $y_1 \cdots y_i$ contain the same leaves, which (leaves being distinct) holds for all i iff every y_i rearranges x_i. The chunks need not be maximal. The formalization states both forms and proves them equivalent (Section 8).

5 The block lemma

Definition 4 (Aligned node; blocks)**.** A node v is *aligned* if $\mathrm{Pre}_1(v) = \mathrm{Pre}_2(v)$. For a permutation π of $\{1, \ldots, m\}$, call $i \in \{0, \ldots, m\}$ a *boundary* if $\{\pi(1), \ldots, \pi(i)\} = \{1, \ldots, i\}$. Consecutive boundaries $b < b'$ delimit a *block* $\{b+1, \ldots, b'\}$. A block of size one is a *singleton* block; its element is necessarily a fixed point of π. (A fixed point need not form a singleton block: in the reversal of $\{1, 2, 3\}$ the point 2 is fixed, but the only block is $\{1, 2, 3\}$.)

The cuts 0 and m are always boundaries, and the blocks partition $\{1, \ldots, m\}$ into intervals. They are found by one left-to-right scan: i is a boundary iff $\max\{\pi(1), \ldots, \pi(i)\} = i$.

If v is aligned then the cuts just before and just after $Y(v)$ are both equivalence points, and $Y(v)$ occupies the same positions in both orders. The converse fails (Remark 3).

Remark 3 (The first definition). The definition on the whiteboard was: a node v is aligned if the cut just before $Y(v)$ is an equivalence point, that is, if $P_1(k) = P_2(k)$ for $k = |\mathrm{Pre}_1(v)|$. This compares the first k leaves of the two orders, not what precedes v in each. Take a node with

children $a\,v\,b$ in language 1 and $a\,b\,v$ in language 2, so $\pi = (1, 3, 2)$ with blocks $\{1\}$ and $\{2, 3\}$. The cut after a is an equivalence point, so the weak definition makes v aligned; but $\mathrm{Pre}_1(v) = Y(a)$ and $\mathrm{Pre}_2(v) = Y(a) \cup Y(b)$. Under the weak definition Lemma 1(c) is false: v would be aligned without forming a singleton block, the cuts between its children would count as switch points, and the grammar would emit words of v in one language on the wrong side of b in the other. Requiring $\mathrm{Pre}_1(v) = \mathrm{Pre}_2(v)$ fixes both ends of $Y(v)$ at once.

Lemma 1 (Block lemma)**.** *Let t be a derivation tree.*

(a) The root is aligned. If a node is aligned, so is its parent.

(b) If $j \in \mathrm{EP}(t)$ and j splits $Y(v)$ (some but not all leaves of v are among the first j leaves of $\mathrm{lin}_1(t)$), then v is aligned.

(c) Let v be aligned with children $c_1, \ldots, c_m$ in language-1 order and permutation π, so that its children in language-2 order are $c_{\pi(1)}, \ldots, c_{\pi(m)}$. The cut after $c_1, \ldots, c_i$ is an equivalence point iff i is a boundary of π. The child c_i is aligned iff $\{i\}$ is a singleton block of π.

(d) Consequently $\mathrm{EP}(t)$ consists exactly of the block boundaries of the aligned nodes, and the aligned nodes are exactly those reached from the root through singleton blocks only.

Proof. (a) $\mathrm{Pre}_1(\mathrm{root}) = \mathrm{Pre}_2(\mathrm{root}) = \emptyset$. If u is the parent of v then $\mathrm{Pre}_a(u) = \mathrm{Pre}_a(v) \setminus Y(u)$ for $a = 1, 2$, so $\mathrm{Pre}_1(v) = \mathrm{Pre}_2(v)$ implies $\mathrm{Pre}_1(u) = \mathrm{Pre}_2(u)$.

(b) $P_1(j) = P_2(j)$ contains some but not all leaves of v. As $Y(v)$ is contiguous in $\mathrm{lin}_a(t)$ and $P_a(j)$ is a prefix, $P_a(j)$ contains every leaf before $Y(v)$ and none after it, so $\mathrm{Pre}_a(v) = P_a(j) \setminus Y(v)$ for both a. Hence $\mathrm{Pre}_1(v) = \mathrm{Pre}_2(v)$.

(c) Let $E = \mathrm{Pre}_1(v) = \mathrm{Pre}_2(v)$ and let j be the cut after $c_1, \ldots, c_i$ in $\mathrm{lin}_1(t)$. Then $P_1(j) = E \cup Y(c_1) \cup \cdots \cup Y(c_i)$, while $P_2(j)$ is E together with the first $j - |E|$ leaves of v in language-2 order, which lists the children whole in the order $c_{\pi(1)}, c_{\pi(2)}, \ldots$ Because all yields are non-empty, $P_2(j) = P_1(j)$ iff this prefix consists of whole children and these are exactly $c_1, \ldots, c_i$, i.e. iff $\{\pi(1), \ldots, \pi(i)\} = \{1, \ldots, i\}$. For the second claim, let p be the language-2 position of c_i, so $\pi(p) = i$. Then $\mathrm{Pre}_1(c_i) = E \cup \bigcup_{l<i} Y(c_l)$ and $\mathrm{Pre}_2(c_i) = E \cup \bigcup_{l<p} Y(c_{\pi(l)})$; these agree iff $\{\pi(1), \ldots, \pi(p-1)\} = \{1, \ldots, i-1\}$, which forces $p = i$ and makes both $i-1$ and i boundaries, i.e. $\{i\}$ is a singleton block. The converse is immediate.

(d) The second statement follows from (a) and (c). For the first, 0 and n are the outer boundaries of the root. Let $0 < j < n$ be an equivalence point and let v be a deepest node whose yield j splits (the root's yield is split, so such a node exists, and it is not a leaf). By (b), v is aligned; j splits no child of v, so it is a cut between two children of v, and by (c) it is a block boundary. Conversely block boundaries of aligned nodes are equivalence points by (c). □

Corollary 1 (Atomic pieces)**.** *The stretches between consecutive equivalence points of t are exactly (i) the aligned leaves and (ii) the non-singleton blocks of aligned nodes.*

Proof. By Lemma 1(d) it suffices to see that a non-singleton block $\{b+1, \ldots, b'\}$ of an aligned node v contains no equivalence point strictly inside. Such a point would be either a cut between children that is not a boundary, contradicting (c), or would split some child c in the block; then c itself is aligned by (b), so by (c) it forms a singleton block, contradicting the choice of a non-singleton block. □

6 The theorem

Theorem 1 (EC Closure Theorem)**.** *For congruent context-free grammars G_1, G_2, the language $L_{\mathrm{EC}}(G_1, G_2, h)$ is context-free and is generated by a grammar of size $O(|G_1| + |G_2|)$. In particular membership in L_{EC} is decidable in time $O(n^3)$ for fixed grammars.*

Proof. Build G_{EC} with nonterminals A^1, A^2, A^E for each $A \in N$, start symbol S^E, and the following rules.

- *Monolingual copies.* For each structural $r = A \to X_1 \cdots X_m$: $A^1 \to X_1^1 \cdots X_m^1$ and $A^2 \to X^2_{\pi_r(1)} \cdots X^2_{\pi_r(m)}$. For each lexical pair $A \to w$, $A \to w'$: $A^1 \to w$ and $A^2 \to w'$.
- *Aligned nodes.* For each structural r with blocks $B_1, \ldots, B_q$ of π_r: $A^E \to Z_1 \cdots Z_q$, where $Z_k = X_i^E$ if $B_k = \{i\}$, and otherwise Z_k is a fresh nonterminal with the two rules $Z_k \to X^1_{b+1} \cdots X^1_{b'}$ and $Z_k \to X^2_{\pi_r(b+1)} \cdots X^2_{\pi_r(b')}$ for $B_k = \{b+1, \ldots, b'\}$.
- *Aligned leaves.* For each lexical pair: $A^E \to w$ and $A^E \to w'$.

A structural rule with m children contributes two monolingual rules, one aligned rule with at most m symbols on the right, and block rules of total length at most $2m$ plus their heads; a lexical pair contributes four rules. So $|G_{\mathrm{EC}}| = O(|G_1| + |G_2|)$. A derivation from S^E is a derivation tree t of G_1 in which the E-labelled nodes are those reached from the root through singleton blocks — by Lemma 1(d), exactly the aligned nodes — together with an independent choice of language for every aligned leaf and every non-singleton block of an aligned node. By Corollary 1 these are exactly the pieces between consecutive equivalence points, and the two Z_k-rules produce exactly their language-1 and language-2 realizations. Hence $L(G_{\mathrm{EC}}) = L_{\mathrm{EC}}$. Membership is then decided by any general context-free parser. □

7 Consequences

Corollary 2 (Switchability criterion)**.** *Let $0 < j < n$ be a cut of t and let v be the lowest node whose yield j splits, so that j falls between two children of v. A switch is possible at j iff every node on the path from the root down to v is entered through a singleton block of its parent's permutation, and j is a block boundary of v's permutation.*

This is Lemma 1(d) read from the cut upwards. Informally, every rule above the cut must keep the cut's position fixed.

Example 1 (Word-order mismatch)**.** Suppose the clause rule is $S \to NP\ VP$ in language 1 and $S \to VP\ NP$ in language 2. Then π exchanges 1 and 2 and has the single block $\{1, 2\}$: the root is aligned, neither child is, and the only equivalence points are 0 and n. The model admits no switch inside the clause at all, whatever the rules below *NP* and *VP* are; if every sentence is such a clause, then $L_{\mathrm{EC}} = L(G_1) \cup L(G_2)$. This is the formal content of the familiar objection that the Equivalence Constraint undergenerates for language pairs with divergent word order.

Example 2 (A Tagalog–English fragment)**.** Language 1 is English and language 2 is Tagalog. All verbs are past tense and, in Tagalog, actor-focus, so the actor is the *ang*-phrase and the object the *ng*-phrase. Accordingly there are two noun-phrase categories, NP_a for the actor and NP_n for the object, each with its own marker category K_a, K_n, and three verb categories: transitive V_t, intransitive V_i and clause-taking V_c. The lexicon is word for word:

K_a: *the*/*ang*	K_n: *a*/*ng*
Pro: *I*/*ako*, *he*/*siya*	C: *that*/*na*
V_t: *bought*/*bumili*	V_i: *left*/*umalis*
V_c: *said*/*nagsabi*	A: *new*/*bagong*
N: *book*/*libro*, *teacher*/*guro*, *phone*/*telepono*	

Pairing *the* with *ang* and *a* with *ng* is sound here and only here: with an actor-focus verb the *ang*-phrase is definite and the *ng*-object indefinite. The adjective always precedes its noun, so it always carries the linker (*bagong*). The structural rules, with x standing for a or n, are:

	English	**Tagalog**	π_r	**blocks**
r_1	$S \to NP_a\ V_t\ NP_n$	$V_t\ NP_a\ NP_n$	$(2,1,3)$	$\{1,2\},\{3\}$
r'_1	$S \to NP_a\ V_t\ NP_n$	$V_t\ NP_n\ NP_a$	$(2,3,1)$	$\{1,2,3\}$
r_2	$S \to NP_a\ V_i$	$V_i\ NP_a$	$(2,1)$	$\{1,2\}$
r_3	$S \to NP_a\ V_c\ CP$	$V_c\ NP_a\ CP$	$(2,1,3)$	$\{1,2\},\{3\}$
r_4	$CP \to C\ S$	$C\ S$	identity	$\{1\},\{2\}$
r_5	$NP_x \to K_x\ N$	$K_x\ N$	identity	$\{1\},\{2\}$
r_6	$NP_x \to K_x\ A\ N$	$K_x\ A\ N$	identity	$\{1\},\{2\},\{3\}$
r_7	$NP_a \to Pro$	*Pro*	identity	$\{1\}$

So *The teacher bought a new book* pairs with *Bumili ang guro ng bagong libro*, and *He said that the teacher left* with *Nagsabi siya na umalis ang guro*. Tagalog allows the actor and the object in either order after the verb, but h is a bijection and must choose: a grammar pair uses r_1 (actor first) or r'_1 (object first), not both. With r'_1 the pronoun rule r_7 is dropped, because *ako* and *siya* are enclitic and must follow the verb directly. The table uses r_1 unless stated. "Rule-level" means Sankoff's criterion applied at the rule where the cut falls, without his labeling conditions; "alignment" means membership in L_{EC}; "natural" is my judgement as a speaker.

Switch	**natural**	**rule-level**	**alignment**
Bumili ako ng | *new phone* (marker–modifier, object)	yes	yes	yes
Bumili ako ng bagong | *phone* (modifier–noun, object)	yes	yes	yes
Nagsabi siya na | *he left* (complement clause)	yes	yes	yes
Bumili ang | *teacher ng libro* (marker–noun, actor)	yes	yes	no
Nagsabi siya na umalis ang | *teacher* (marker–noun, actor of r_2)	yes	yes	no
Bumili ng libro ang | *teacher* (marker–noun, actor, with r'_1)	yes	yes	no
Nagsabi siya na umalis | *the teacher* (verb–actor, inside r_2)	no	no	no
Bumili ako | *a new phone* (before the object marker)	no	yes	yes

The fourth row is the Taglish counterpart of Example 3: the actor sits in the non-singleton block $\{1,2\}$ of r_1, so it is not aligned, and after two words the Tagalog prefix is $\{$*bumili*, *ang*$\}$ while the English prefix is $\{$*the*, *teacher*$\}$.

Four things follow. First, with a flat verb–actor–object clause the alignment reading keeps every switch from the third constituent on, so it is the binary *NP VP* analysis of Example 1, not predicate-initial order as such, that removes all of them. Second, the two readings come apart exactly on switches inside a constituent that belongs to a non-singleton block: the actor under r_1 and r_2, and everything under r'_1. Rows four to six are of this kind, and they are ordinary Taglish, so here the alignment reading undergenerates where the rule-level one does not. Third, both readings

overgenerate in the last row: the cut before the object is an equivalence point, yet speakers keep the marker in Tagalog and say *Bumili ako ng new phone*. Fourth, a great deal of Taglish is outside the model altogether, because a word-for-word lexicon with a rule bijection has no place for it: English roots with Tagalog affixes (*nag-drive*), enclitic particles (*wait lang*), object-focus verbs, whose actor is a *ng*-form (*sinabi niya*) while English keeps *he*, the linker, which attaches to whichever word comes first (*bagong libro* but *librong bago*) and is a separate word after a consonant (*mabait na guro*), and words with no counterpart on the other side (*ay*, the English copula).

8 Machine-checked proof and tests

Lean 4 formalization. The file `ECClosure.lean` (1,659 lines, Lean 4.36.0, core library only, no Mathlib) is accepted by the Lean kernel with no `sorry`; the theorems depend only on Lean's three standard axioms (`propext`, `Classical.choice`, `Quot.sound`). To re-check: `lean ECClosure.lean`.

In this note	**In Lean**
Part I: trees	
equivalence point; realization of t (Definitions 2–3)	`EqPt`, `RealFrom`, `RealEP`
realization of t, chunk form (Remark 2)	`Real`
the two forms agree	`realEP_iff_real`
block of a permutation	`Irr` on the two child orders
no equivalence point inside a non-singleton block	`no_cut_inside`, `irr_flat`
a realization can be split at any equivalence point	`Real.split`
Lemma 1 and Corollary 1 at one node	`node_theorem`
tree-level theorem	`main_theorem`, `main_theorem_EP`
Example 1 (no switch when the orders are opposite)	`swapClause_gram`
Part II: grammars	
context-free grammar and its language	`CFG`, `CFG.Gen`, `CFG.Language`
congruent pair of grammars (Definition 1)	`PRule`, `WFG`
derivation tree of the pair	`DTree`, `DTree.Valid`
the language L_{EC}	`LEC`
the grammar G_{EC}	`gec`, `rulesOf`, `NT`
$L_{\mathrm{EC}} \subseteq L(G_{\mathrm{EC}})$	`gec_complete`
$L(G_{\mathrm{EC}}) \subseteq L_{\mathrm{EC}}$	`gec_sound`
Theorem 1, the equality $L(G_{\mathrm{EC}}) = L_{\mathrm{EC}}$	`gec_correct`
L_{EC} is context-free	`LEC_contextFree`
$L(G_1) \cup L(G_2) \subseteq L_{\mathrm{EC}}$	`lang1`, `lang2`, `mono_subset`

The final statements, for a grammar pair `P` with start symbol `S` and a sentence `s`, are

$$\texttt{WFG P} \to \big(\texttt{(gec P S).Language s} \leftrightarrow \texttt{LEC P S s}\big),$$
$$\texttt{WFG P} \to \exists\, \texttt{G},\ \forall\, \texttt{s},\ \big(\texttt{G.Language s} \leftrightarrow \texttt{LEC P S s}\big).$$

Here `LEC` is defined with `RealEP`, the literal equivalence-point definition, on derivation trees whose leaves are identified by their address in the tree; `WFG` says only that every structural rule has at least one child and that its π is a permutation. The Lean statement is slightly more general than Definition 1, in that the two vocabularies need not be disjoint. Like Definition 1, and unlike Sankoff's setting, it allows ambiguous grammars and repeated symbols on a right-hand side. The Lean proof is organised differently from Section 5: instead of aligned nodes it shows that a realization splits at every block boundary (`Real.split`) and is monolingual on every non-singleton block (`irr_flat`); the content is the same.

What is not machine-checked. (i) The size bound: that $|G_{\mathrm{EC}}| = O(|G_1| + |G_2|)$ is the count given in the proof of Theorem 1; it is not stated in Lean. The cubic membership bound is the standard one for context-free parsing and is not formalized either. (ii) Computability of the blocks: Lean obtains the block decomposition of each π_r from an existence proof (it exists and the grammar built from it is correct); that the left-to-right scan of Section 5 computes it is not formalized. The first test below runs exactly that scan. (iii) The worked examples other than Example 1. Neither the Tagalog–English fragment (Example 2) nor the Duke de Lorraine sentence (Example 3) is in the Lean file. For the fragment, the "alignment" column is checked against the prefix-set definition by `ec_taglish_check.py`; its "rule-level" column and Example 3 are derived by hand.

Tests. These are independent of the Lean development.

- `ec_grammar_test.py` builds G_{EC} from random congruent grammar pairs exactly as in the proof of Theorem 1 (blocks computed from the permutations), enumerates its language with a generic context-free enumerator, and compares it with L_{EC} computed from the prefix-set definition by enumerating derivation trees. The two sets agree on all 150 random grammar pairs tested (78,237 sentences of length at most 5, of which 68,998 mix both languages).
- `ec_exhaustive.py` checks Lemma 1, Corollary 1 and the block construction on *every* tree with at most 7 leaves, every arity ≥ 2 and every permutation at every node (169,193 trees).
- `ec_check.py` checks Lemma 1 and Corollaries 1 and 2 on 12,600 random larger trees (8 to 25 leaves, arities 1 to 6, so unary nodes are included), and the block construction on the 11,626 of them with at most 2^{14} realizations.

9 Relation to Sankoff (1998a)

The comparison is with Sankoff's 1998 article in *Bilingualism: Language and Cognition* (1998a); page numbers refer to it. The conference version (1998b) states the same rule-level criterion more briefly and extends the model to insertional switching and borrowing. In summary:

- **Same setting.** He models the two languages by congruent context-free grammars with a one-to-one correspondence between rules whose right-hand sides contain the same symbols in possibly different orders (pp. 42–43). He further assumes unambiguous grammars and no repeated symbol on a right-hand side; neither is needed here.
- **His equivalence point is our block boundary.** His equivalence constraint (pp. 46–47) prohibits a switch between the i-th and $(i+1)$-st subconstituents when the sets of the first i right-hand-side symbols differ in the two grammars. This is exactly the notion of *boundary* of π_r in Section 5, and the credit for it is his.
- **Different question.** His theorems derive the constraint from a production model (left-to-right copying, most-nested-first, constituent labeling, no forced switches). The paper does not characterize the resulting set of sentences as a formal language and gives no grammar for it; it explicitly aims to proceed without a code-switching grammar. Earlier grammar-based treatments exist (Sankoff and Poplack 1981; Sankoff and Mainville 1986), so context-freeness of equivalence-constrained switching in some form is older than this note. Sankoff and Poplack (1981) first reject the plain union of the two monolingual rule sets, which lets a switch fall anywhere. They then admit the rules of both grammars and attach language superscripts,

inherited downward, to the symbols of any rule whose order is not shared, so that the words next to a non-equivalent boundary are drawn from the language of that rule. Sankoff and Mainville (1986) recast this with language labels on constituents and add a probabilistic grammar to predict how often each switch point is used across word-order types. Both encode the rule-level reading: a superscript or label constrains the material at one rule's boundaries, and a reordering higher in the tree does not remove switch points below it. Both say that the result is context-free: the 1981 paper designs the superscripts so that the grammar stays a phrase-structure grammar of that kind, and the 1986 paper remarks that its mechanism amounts to one without writing it out. Neither states the switched language independently of the grammar and proves the two equal. G_{EC} differs on both counts: it carries a single bit, aligned or not, down the tree, which is what the global reading needs, and it is proved equal to L_{EC} as defined from prefix sets.

- **Local versus global.** His constraint is local to one rule; the alignment reading used here is global and strictly stronger: every global equivalence point is a rule-level one (Lemma 1(d)), and Example 3 shows that the converse fails.

Example 3 (The readings differ)**.** Sankoff's example (8), *The original Duke | de Lorraine | had 10 000 men*, is acceptable in his model: inside the lowest noun phrase, noun and prepositional phrase come in the same order in both languages. Under the alignment reading the first switch is not allowed. The noun phrase is *The original Duke de Lorraine* in English and *le duc de Lorraine original* in French; after three words the English prefix contains {the, original, duke} and the French prefix, written with the same English glosses, {the, duke, de}, so the cut is not an equivalence point. In the terms of Section 5, the lowest noun phrase sits inside the non-singleton block formed by adjective and noun phrase, so it is not aligned.

What this note adds is therefore: the alignment reading stated precisely; the block lemma, which says that the global equivalence points are exactly Sankoff's rule-level equivalence points at nodes reached through singleton blocks only; the exact equality $L(G_{\mathrm{EC}}) = L_{\mathrm{EC}}$ with an explicit grammar; and a machine-checked proof.

10 Open questions

1. **The local reading.** Sankoff's constraint is local to a rule (Section 9) and admits more sentences than the alignment reading. Characterize the language of his full production model (with the labeling rules and no forced switches) and give a grammar for it with a checked proof.

2. **Congruence.** Real grammar pairs have rules without counterparts and non-isomorphic trees. What is the right relaxation (e.g. synchronous grammars with unpaired rules), and does closure survive?

3. **Separation from the Matrix Language model.** Characterize the triples (G_1, G_2, h) on which the Equivalence Constraint and the Matrix Language model (Joshi 1982; Myers-Scotton 1993) generate the same language.

4. **The gap on real data.** Measure, on attested Taglish, how many switches the local reading licenses and the alignment reading excludes (Section 2).

References

- K. Baliga, A. Xie, A. Byju (2026). Greedy byte-pair encoding achieves 0.391 of the optimum, past the one-third barrier for packing-based analyses. arXiv:2611.08671.
- G. Bhat, M. Choudhury, K. Bali (2016). Grammatical constraints on intra-sentential code-switching: from theories to working models. arXiv:1612.04538.
- A. K. Joshi (1982). Processing of sentences with intra-sentential code-switching. *COLING 1982*, 145–150. ACL Anthology C82-1023.
- L. Kozma and J. Voderholzer (2024). Theoretical analysis of byte-pair encoding. arXiv:2411.08671 (version 2, 30 October 2026).
- C. Myers-Scotton (1993). *Duelling Languages: Grammatical Structure in Codeswitching.* Oxford: Clarendon Press.
- S. Poplack (1980). Sometimes I'll start a sentence in Spanish y termino en español: toward a typology of code-switching. *Linguistics* 18(7–8), 581–618. doi:10.1515/ling.1980.18.7-8.581.
- D. Sankoff (1998a). A formal production-based explanation of the facts of code-switching. *Bilingualism: Language and Cognition* 1(1), 39–50. doi:10.1017/S136672899800011X.
- D. Sankoff (1998b). The production of code-mixed discourse. *COLING-ACL 1998*, vol. 1, 8–21.
- D. Sankoff and S. Mainville (1986). Un modèle pour l'alternance de langue sous la contrainte d'équivalence. *Revue québécoise de linguistique* 15(2), 233–245. doi:10.7202/602568ar.
- D. Sankoff and S. Poplack (1981). A formal grammar for code-switching. *Paper in Linguistics* 14(1), 3–45. doi:10.1080/08351818109370523.