Advances in Complex Systems
© World Scientific Publishing Company

MODELING THE LEFT PERIPHERY: A CELLULAR AUTOMATON APPROACH TO LINGUISTIC COMPLEXITY

ELENA CALLEGARI
Department of Language & Technology, University of Iceland,
Sæmundargata 2, 102 Reykjavík, Iceland
ecallegari@hi.is

Received (30 November 2024)

This paper proposes a novel approach to modeling word order in the left periphery (LP) of syntax, challenging traditional cartographic models by framing the LP as a dynamic, self-organizing system. Drawing on principles from complexity science, the model treats LP constituents as agents interacting based on local rules, which collectively give rise to emergent syntactic patterns. Unlike hierarchical frameworks that assume a rigid, predefined structure, this approach allows word order to emerge incrementally through local interactions, offering a more flexible and context-sensitive account of syntactic organization. A simulation, implemented using the NetLogo agent-based modeling environment, demonstrates how such interactions can replicate observed linguistic patterns in the LP. The model's ability to generate syntactic structures without relying on a fixed blueprint offers a computationally simpler and conceptually more adaptable alternative to traditional cartographic approaches. This study bridges linguistic theory and complex systems, highlighting the value of emergent phenomena in understanding the dynamics of language.

Keywords: Word Order; Linguistics; Left Periphery; Cellular Automata, Linguistic Complexity

1. Introduction

The left periphery (LP) in linguistics refers to the portion of a sentence that precedes the main clause. In languages like English, this generally corresponds to everything to the left of the subject. For instance, the left periphery includes a topicalized expression in example 1, and an interrogative constituent and a fronted auxiliary in example 2 (left-peripheral material is highlighted in bold):

(1) **This book,** I have only just skimmed.

(2) **What did** you do?

The left periphery plays an essential role in shaping the meaning and function of a sentence [4, 14, 27]. It is where elements that guide the sentence's interpretation and its connection to the larger conversation are placed. For example, the left periphery helps signal whether a sentence is a question, whether something is being emphasized, or whether certain information is presupposed or assumed to be known.

What makes the LP especially remarkable is its ability to accommodate a wide range of elements, each arising from distinct grammatical processes. For example, consider the following sentence, from Italian. Italian is known to exhibit particularly rich left peripheries:

(3)

Ieri,	a	Luigi,	perché	improvvisamente	il	libro	Maria	glielo
yesterday	to	Luigi	why	suddenly	the	book	Maria	3.DAT=3.ACC

ha	tirato	addosso?
has	thrown	against

'Why did Maria suddenly throw the book at Luigi yesterday?'

In this sentence, five left-peripheral elements co-occur: the temporal modifier *Ieri* ('yesterday'), which situates the action in time; the topical element *a Luigi* ('to Luigi'), marking one discourse topic; the interrogative element *perché* ('why'), marking the clause as being a question; the manner modifier *improvvisamente* ('suddenly'), which describes the modality of the action, and the the topical element *il libro* ('the book'), marking a second discourse topic.

Importantly, none of these elements —except for the wh-word perché ('why') —must appear in the left periphery. However, when these elements are realized in the left periphery, the meaning of the sentence is subtly altered. For instance, the positioning of the temporal modifier, topics, and manner modifier in the left periphery can shift the focus and emphasis of the sentence, affecting how the listener interprets the information in relation to the larger discourse.

Traditional approaches to modeling the LP, particularly within generative linguistics, have predominantly relied on cartographic frameworks [13, 15, 16, 27–33]. These models posit that the LP comprises a rigid hierarchy of functional projections, each assigned a specific grammatical or discourse function. In this framework, every type of constituent that can appear in the LP —such as topics, modifiers, or interrogative phrases— is assigned a predefined slot within the hierarchy.

While cartographic models have been instrumental in capturing cross-linguistic generalizations about the structure of the LP, they rest on the assumption of a universal underlying "blueprint" for sentence structure. This rigidity can lead to issues of overgeneration, i.e. the model predicts the grammaticality of structures that are not, in fact, permissible in any language.

In this paper, I propose an alternative perspective, framing the LP as a *complex system*. Like other complex systems, the LP is characterized by the interaction of multiple components, whose interaction is governed by localized rules, which collectively give rise to emergent global patterns. Rather than assuming a pre-specified hierarchical template, this approach models LP word order as the product of dynamic, self-organizing processes. By emphasizing the interactions between constituents and their local contexts, this perspective aligns with principles from complexity science, which have been successfully applied to other domains of human and natural systems.

To investigate this hypothesis, I present a simulation built using the NetLogo agent-based modeling environment [38], which constitutes the first dynamic simulation of word order in the LP to date. In this framework, LP constituents are represented as agents that interact according to predefined precedence rules and constraints. These local interactions allow global word order patterns to emerge incrementally, obviating the need for a fixed underlying template. The simulation demonstrates that such localized processes can account for observed ordering constraints in the LP, offering a computationally and conceptually simpler alternative to cartographic models.

This paper has three primary objectives:

(1) To highlight that the left periphery shares many features commonly found in complex systems. This suggests that we may gain a deeper understanding of it by applying techniques and insights from complexity science to model its dynamics.
(2) To introduce a novel, complexity-science-inspired framework for modeling word order in the LP.
(3) To present a practical tool —a dynamic simulation implemented in NetLogo— that enables researchers to model LP word order interactively and experiment with different precedence rules, providing a flexible environment for testing and refining theoretical approaches to word order.

The remainder of this paper is organized as follows: Section 2 examines the LP as a complex system, highlighting its emergent properties and flexibility while contrasting traditional cartographic approaches with the proposed dynamic model. Section 3 details the theoretical foundation for modeling LP word order as an emergent phenomenon driven by localized interactions. Section 4 introduces the simulation, explaining its design, key modifications inspired by linguistic theory, and interactive capabilities for exploring word order derivation. Section 5 explores the broader relevance of the model to linguistic theory, complexity science, and computational approaches. Finally, Section 6 summarizes the paper's contributions.

2. The Left Periphery as a Complex System

The left periphery of syntax exhibits the properties *numerosity*, *interconnectedness* and *emergence*, which are key characteristics of complex systems [1, 2, 17, 22, 26, 36, 40]. These properties highlight the LP's capacity to organize multiple linguistic elements into cohesive structures while mediating between syntactic organization and discourse-level functions.

Numerosity A key characteristic of complex systems is the large number of components that make up the system, and the LP exemplifies this property. The LP serves as a repository for a vast and diverse array of linguistic elements, each associated with distinct grammatical or discourse functions. Consider the following elements that can appear in the LP across languages:

- **Various types of topics**: elements referring to "old" or presupposed information, e.g., ***As for this book,*** *I already read it.*
- **Different types of foci**: elements emphasizing "new" or highlighted information, e.g., *It was* ***a book*** *that I bought.*
- **Modifiers and frame-setting expressions**: temporal or situational markers, e.g., ***Yesterday,*** *I read a book*; ***At the park,*** *I eat lunch.*
- **Wh-elements**: question words such as *why* or *what*, e.g., ***Why*** *did you buy a book?*
- **Declarative and interrogative complementizers**: words introducing embedded clauses, e.g., *I think* ***that*** *she left*, *I wonder* ***if*** *she is awake.*
- **Relative pronouns**: words introducing relative clauses, e.g., *The woman* ***whom*** *I love.*
- **Negative adverbs**: e.g., ***Never*** *have I witnessed anything more reckless.*

Interconnectedness Complex systems are characterized by the interdependent interactions among their components, and the LP operates in a similar way. The elements within the LP are not independent; their positions and roles dynamically influence one another, creating a network of interrelations that shapes interpretation and syntactic structure.

A particular clear example of this interdependence is the relationship between topics and foci. Both topics and foci are types of constituents that can appear in the left periphery or remain in the main clause. In discourse, topics typically represent what the sentence is "about," often corresponding to presupposed or old information. Foci, on the other hand, represent "new" or emphasized information [7,18,19,24,25]. Since topics and foci convey opposite types of information (*old* vs. *new*, *presupposed* vs. *emphasized*), it follows that it is impossible to define the focus of a sentence without simultaneously defining the topic, and vice versa.

As an illustration of this, consider the following dialogue:

(4) **A:** Who did Maria invite to the party?
B: Maria invited **John**.

In this exchange, *Maria* and *the party* are already established in the discourse. They are thus topical, presupposed, and provide the background for the new information. The focus is *John*, as this represents the new information, i.e., everything that is not topical.

Now consider a variation of that same sentence:

(5) **A:** What did Maria do?
B: Maria **invited John to the party**.

In B's reply, only *Maria* is presupposed by both speaker A and B, making it topical. The entire verb phrase, on the other hand, represents new information

—content known to B but not to A— and thus constitutes the focus of the sentence.

These examples demonstrate how the identification of the topic depends on what is presented as new or focal information, and vice versa. It is impossible to determine what counts as new information without also determining what counts as old information.

Emergence Emergent properties arise when the interactions among components produce outcomes not attributable to any single part in isolation. The LP exemplifies emergence because its meaning cannot be fully determined by examining individual elements independently; instead, interactions among constituents shape the overall interpretation.

This dynamic interplay can be observed in the interaction between wh-elements and foci, as illustrated by the following pair of questions. In example 7, the phrase a book is focalized, receiving particular emphasis.

(6) Why did you buy a book?

(7) Why did you buy *a book*?

As noted by [3, 21], the interpretation of a wh-word like “why” (and, by extension, the meaning of the entire clause) depends on what is in focus. In example 6, the question seeks a general reason for the act of buying. In contrast, in example 7, the focus on a book shifts the interpretation, asking why the speaker chose a book specifically, as opposed to something else. This shift in meaning, driven by the interaction between the focus and the wh-element, exemplifies emergent interpretive properties.

Cross-Linguistic Evidence of Complexity Cross-linguistic data highlight the LP's adaptability, a hallmark of complex systems. While the specific configuration of the LP varies across languages, its core functions —facilitating syntactic organization and mediating discourse priorities— remain consistent. This variability reflects the LP's ability to balance global constraints (e.g., grammatical rules) with local adaptations (e.g., context-specific discourse needs). For instance:

- Languages like Italian, French and Spanish permit multiple LP elements, such as temporal modifiers, topics, and foci, to co-occur. These elements interact dynamically to structure complex discourse.
- Languages like English, Norwegian and Icelandic typically restrict the LP to three elements, e.g. two topics and a fronted verb.

The LP's ability to adapt to different grammatical norms while preserving its core function demonstrates its emergent complexity. The flexibility to accommodate diverse linguistic needs across languages showcases how the LP functions as a self-organizing system that balances constraints and interactions. This cross-linguistic variability reinforces the view that the LP operates as a complex system, driven by universal principles yet adaptable to local linguistic environments.

2.1. *Cartographic Approaches to the Left Periphery, and Their Limitations*

The left periphery has been a focal point of linguistic research, particularly within generative frameworks. Traditional approaches to modeling the LP, particularly *cartographic* models (see Rizzi 1997, et alia) [13, 15, 16, 27–33], propose that it consists of a rigid hierarchy of functional slots. In these models, each type of constituent that may appear in the left periphery is assigned a dedicated slot. In this sense, the left periphery can be conceptualized as a series of shelves, each designed to hold a specific type of element. For example, one "shelf" might be for topics, another for foci, and another for question words like "why" or "how". Crucially, according Rizzi and proponents of cartographic models, these shelves are rigidly ordered with respect to each other, meaning that each type of element must appear in a specific position relative to others.

Over the years, there have been numerous conceptualizations of the left-peripheral hierarchy. In this paper, I present a synthesized hierarchy that combines two of the most recent proposals: the template developed by Rizzi & Bocci (2017) [33] and the hierarchy outlined by Frascarelli (2012) [15]. The decision to synthesize these two models stems from the wide array of potential implementations of the left-peripheral hierarchy available in the literature. By combining the insights of Frascarelli (2012) and Rizzi & Bocci (2017), this approach leverages the strengths of both proposals. Frascarelli's model offers a detailed account of the positioning of different types of topics, providing a nuanced framework for analyzing topic variation. Meanwhile, the hierarchy proposed by Rizzi & Bocci incorporates more recent theoretical developments, such as the inclusion of elements like QEmb, which enrich our understanding of the syntactic and interpretive possibilities of the left periphery. The synthesized hierarchy is shown in 8:

(8) Force >Shift >Contr >Int >Foc >Fam >Mod >QEmb >Fin

In 8, the symbol ">" indicates linear precedence. Each position in the hierarchy in 8 corresponds to a specific type of element. For instance, Force hosts declarative complementizers (e.g., "I believe *that* she is right") and relative pronouns (e.g., "I met the man *who* arrived this morning"). QEmb accommodates embedded interrogative elements (e.g., "I wonder *what* you saw"), and Mod is reserved for modifiers (e.g., "*Suddenly,* he started speaking"). In this framework, speakers construct sentences by mapping elements to their designated positions, much like placing books on specific shelves. If a speaker had to construct a left periphery containing both a modifier and an embedded interrogative element, then, in the resulting sentence, the modifier would appear before the embedded interrogative element because, in the template in 8, the slot "Mod" precedes the slot "QEmb".

Essentially, cartographic models suggest that speakers rely on a "blueprint" for sentence construction, with each position corresponding to a specific grammatical or discourse function. Importantly, even when only some positions are filled in a given sentence, the entire structure is assumed to be present in the underlying representation. These models assume that all potential positions are universally present, whether or not they are overtly realized in a given sentence.

While these models have been successful in capturing similarities in how different languages organize word order in the left periphery, they come with notable limitations. A primary limitation of cartographic models is the problem of *overgeneration*. For instance, a cartographic template of the LP as the one in 8, which contains nine slots, theoretically allows for left peripheries with all of these nine positions filled simultaneously. However, this is never the case, as some left-peripheral constituents cannot co-occur with other left-peripheral constituents. For example, in English, an embedded clause cannot grammatically include both a declarative complementizer and an interrogative complementizer, as illustrated by the ungrammaticality of the following example:

(9) *I wonder **that if** she is awake.

While cartographers posit distinct slots for declarative complementizers (Force) and interrogative complementizers (Int) [28], these two positions cannot be filled simultaneously in English. To address such restrictions, additional stipulations must be introduced to account for these incompatibilities. For example, the incompatibility between the complementizers *if* and *that* in English requires supplementary rules beyond the basic hierarchy. This reliance on ad hoc constraints undermines the purported explanatory simplicity of the cartographic framework.

A second limitation of the cartographic approach is its overly static representation of the left periphery, which neglects the intricate interconnectedness of its constituents, as discussed in Section 2. Templates like 8 impose fixed, discrete slots for left-peripheral functions, failing to capture the dynamic interactions that govern the interpretation and positioning of these elements. This rigidity overlooks how the roles of left-peripheral elements are not isolated but mutually influential, with their behavior emerging from interdependent processes rather than predefined structural positions. A more dynamic framework is required to adequately represent these interactions and the emergent properties they produce.

Given these challenges, an alternative framework is worth exploring. Rather than relying on rigid templates, I suggest that the LP can be modeled as a dynamic, self-organizing system in which word order emerges through local interactions between elements. This perspective draws inspiration from complex systems, which emphasize the role of simple, localized changes in generating emergent patterns.

3. The Simulation

This section introduces a framework for modeling the left periphery as a self-organizing system, meaning that its structure emerges dynamically from local interactions among its constituents, without relying on a predetermined blueprint or fixed template to determine the final configuration. Rather than assigning fixed positions to elements such as topics, foci, and complementizers, this approach derives their placement dynamically through relational constraints.

Emergence plays a central role in this framework. In the simulation, local interactions between neighboring elements —governed by rules such as precedence and mutual exclusivity— give rise to coherent, higher-order structures without the need for predefined slots. This mirrors principles observed in complex systems, where global patterns emerge from simple, distributed processes. The simulation provides a flexible and interactive environment for exploring how localized rules and constraints generate complex syntactic structures.

At its core, this simulation explores how elements in a sentence's 'left edge' —known as the left periphery— organize themselves into a specific order. In natural languages, the order of these elements influences meaning and grammaticality, but how this order is determined remains a key question in linguistics. This simulation demonstrates how simple, local rules can lead to complex, well-formed structures, offering insights into how languages may handle this organizational process.

3.1. *Implementation & Code*

The model was implemented using NetLogo [38], a programming environment designed for simulating complex systems.

The NetLogo code for this simulation has been made available in a public repository to facilitate reproducibility and further experimentation. Users can modify the initial configurations and rules to explore variations of the model. The complete code for the simulation is available at: https://github.com/elecalle/LeftPeripheryAgentModel. This repository also provides the option to run the simulation directly using NetLogo Web.

This simulation draws inspiration from one-dimensional cellular automata (CA), where agents are arranged linearly and interact with their immediate neighbors [37, 39]. However, the model incorporates key adaptations, such as right-to-left updates, to better align with linguistic processes.

In this model:

- Different types of left-peripheral constituents are represented as different patches in a single row of the NetLogo world.
- The states of the patches correspond to syntactic roles, and each patch can interact with its immediate neighbors according to a set of predefined local rules.
- Interactions between patches determine word order.
- The model is designed to visualize and log the reordering process, making the simulation behavior transparent to the user.

3.2. *Conceptual Framework*

The simulation draws inspiration from cellular automata (CA), a computational framework commonly used to model complex systems through simple, localized interactions. In this case, the simulation is implemented as a one-dimensional, multi-state cellular automaton. The decision to use a *one-dimensional* CA stems from the inherent nature of word order phenomena. Word order is fundamentally a one-dimensional phenomenon due to the sequential nature of spoken and written language:

- Spoken and written language unfolds sequentially. Words are produced or read one after another in a linear sequence, constrained by the medium of communication (speech or text).
- This linearity imposes a temporal or spatial order, requiring decisions about the arrangement of elements in a sentence —e.g., determining which word or phrase comes first, second, and so on.
- While linguistic structures are often represented hierarchically in syntax (e.g., tree-like configurations [5, 6, 8, 9, 23, 34]), these hierarchies must be linearized into a one-dimensional string of words for actual language use.
- The process of linearization transforms multi-dimensional syntactic relationships into a linear sequence that adheres to precedence rules, such as placing the subject before the verb in English [20, 35].

Since word order is an inherently one-dimensional phenomenon, a single row in a cellular automaton can then be used to provide a representation of word order. For example, imagine we want to represent the word order of a simple declarative sentence using a multi-state one-dimensional CA. Consider the following sentence as an example:

(10) 'Yesterday, John gifted his car to the Icelandic Red Cross.'

To represent example 10 as a single row in a cellular automaton, we can define the states of the cells in the CA as corresponding to different syntactic roles. In the NetLogo framework, this could correspond to a model where the patch states are configured as follows:

- **Temporal Modifier (Temp)**: represents elements like *Yesterday.*
- **Subject (Subj)**: corresponding to the subject of the sentence, e.g. *John.*
- **Verb (Verb)**: denotes the main action or predicate, e.g *gifted.*
- **Direct Object (Dir. Obj)**: the entity directly affected by the action, such as *his car.*
- **Indirect Object (Ind. Obj)**: the recipient or beneficiary of the action, such as *to the Icelandic Red Cross.*

Figure 1 provides a visual representation of this sentence in a CA framework. Each cell represents one of these syntactic constituents, arranged in their canonical order.

By altering the states of the CA cells, we can then model alternative word order configurations. For example, consider the topicalization of the indirect object, as in:

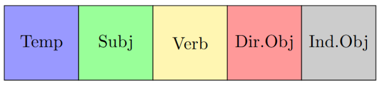


Fig. 1. CA representation of basic word order in a declarative sentence.

(11) To the Icelandic Red Cross, John gifted his car yesterday.

Figure 2 illustrates this derived configuration, where the leftmost patch is now assigned state “Ind. Obj”, to reflect the left-peripheral position of the indirect object.

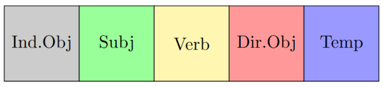


Fig. 2. CA representation of topicalization of the indirect object.

The row progression typical of cellular automata can represent transitions between basic and derived word order configurations. For instance, Figure 3 depicts a progression from the basic **Direct Object** >**Indirect Object** structure to the marked **Indirect Object** >**Direct Object** structure.

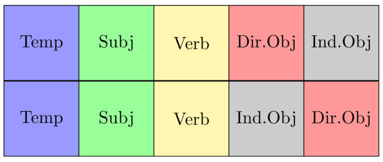


Fig. 3. CA progression illustrating the shift from Direct Object >Indirect Object to Indirect Object >Direct Object.

In this progression, **Row 1** represents the basic word order configuration (‘Yesterday, John gifted his car to the Icelandic Red Cross’), and **Row 2** the derived word order configuration (‘Yesterday, John gifted the Icelandic Red Cross his car’).

In English, the neutral (or basic) word order for ditransitive constructions places the direct object (DO) before the indirect object (IO), as seen in Row 1. This unmarked configuration reflects the default syntactic structure for expressing the relationship between objects. The reversed order, where the IO precedes the DO (as in Row 2), is marked and subject to specific discourse or pragmatic conditions. For example, the marked IO-before-DO order often occurs when the IO is more prominent in the discourse or is considered more ”topical,” while the DO introduces new information.

This progression aligns with linguistic theories of sentence production, which suggest that speakers typically construct sentences starting from an unmarked or basic word order before applying additional syntactic operations to derive marked configurations [**?**].

3.2.1. *Deviations from Traditional Cellular Automata*

While the simulation draws inspiration from multi-state cellular automata, key modifications were made to accommodate the specific requirements of modeling linguistic processes. These adaptations are grounded in linguistic theory developed over the past 30 years.

Two principal deviations are implemented: (i) the lack of simultaneous updates across all cells in a row, and (ii) the introduction of right-to-left updates. Both modifications align with theoretical insights into the incremental and hierarchical nature of syntactic derivations.

Asynchronous updates: In traditional CA models, all cells in a row are typically updated simultaneously. However, in linguistic theory, derivations are understood to proceed incrementally. Since the introduction of phases [11], syntactic theory has posited that sentence building unfolds in discrete, hierarchical chunks, with specific domains being constructed and finalized before others become accessible for further derivation.

This incremental process necessitated adapting the CA framework for word order we have been developing to use asynchronous updates. In my simulation, patches are thus evaluated one pair at a time rather than simultaneously updating all patches in a row. Each row represents a stage in the derivation where a localized interaction between elements is assessed and potentially adjusted. For instance:

- In the first row, the rightmost pair of patches is evaluated.
- In the second row, the pair of patches immediately to the left of the rightmost pair is evaluated.
- Subsequent evaluations proceed sequentially to the left, addressing one pair of neighboring patches at a time.

This asynchronous approach allows the simulation to model the stepwise, context-sensitive nature of syntactic derivations, contrasting with the global synchronization found in traditional CA.

Right-to-left updates: mimicking bottom-up processes in syntax Updates in the simulation proceed from right to left across a row of patches, reflecting the bottom-up nature of syntactic derivations as described in minimalist syntax [11]. In linguistic theory, sentence building starts with the internal components of a structure —such as objects— and progresses outward to higher-order constituents like verbs, subjects, and eventually the left periphery. This hierarchical construction ensures that dependencies within lower-level elements are resolved before higher-level configurations are finalized.

In the simulation, this principle is operationalized through localized right-to-left updates:

- The rightmost pair of patches is evaluated in the first step of a row.
- In subsequent steps, evaluations proceed sequentially to the left until the entire row has been processed.

For example, in a sentence like:

(12) Yesterday, John gifted his car to the Icelandic Red Cross.

The simulation would first evaluate interactions between the indirect object (*to the Icelandic Red Cross*) and the direct object (*his car*). In the next step, it would evaluate interactions between the verb (*gifted*) and the direct object, and so on, progressing toward the subject (*John*) and temporal modifier (*Yesterday*).

This localized, bottom-up approach captures the derivational process described in linguistic theory, where syntactic structures are built incrementally from foundational elements to larger configurations. The right-to-left update direction also emphasizes the interdependence of elements, as earlier updates can dynamically affect subsequent derivational stages.

4. Simulation Design

4.0.1. *Model Interface*

The simulation interface consists of the following components:

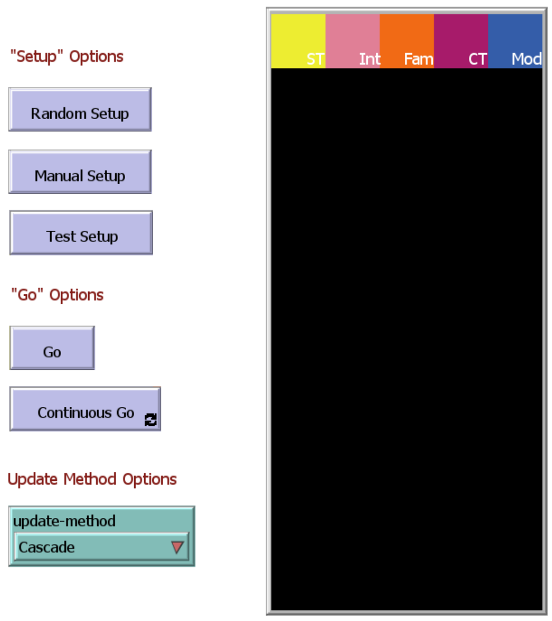
- *Setup Options:* to initialize the patches based on a selected configuration.
- *Go Options:* to run the simulation step by step until stability is reached.
- *Update Modes:* this allows the user to switch between two different update methods, *Cascade* and *Pairwise*.

A screenshot of the simulation interface is provided in Figure 4.

The simulation's grid serves as a visual representation of the left periphery. The patches in the NetLogo world represent different left-peripheral constituent, such as *Fam* (Familiar Topic), *Int* (interrogative complementizer), or *Mod* (modifiers).

The model includes nine possible initial states for patches, corresponding to nine different types of left-peripheral constituents that have dedicated slots in the left-peripheral hierarchy displayed in 8:

- **Int(errogative)** represents interrogative complementizers (e.g., *if, whether*).
- **Rel(ative)** represents relative pronouns (e.g., *whom*) and declarative complementizers (e.g., *that*).



update-method
Cascade

Fig. 4. Simulation Interface

- **Mod(ifier)** represents any left-peripheral modifiers, e.g. the manner modifier “suddenly” or the temporal modifier “yesterday”.
- **Wh** represents main-clause wh-elements (e.g., *who, what, where*, as in “*what* did you do” or “*where* is it?”).
- **CT (= Contrastive Topics)**: topics that contrast with another possible topic, often implying a set of alternatives (e.g., ”*As for John*, he stayed home”).
- **ST (= Shifting Topics)**: topics that indicate a shift in the discourse topic, marking a transition to a new subject (e.g., “Now, about *the party*...”).
- **Fam(iliar Topics)**: topics that refer to information already known or shared in the discourse.
- **Foc(us)**: Elements emphasized as new or important information in the sentence (e.g., “It was *John* who solved the problem”).
- **QEmb (= embedded wh)**: wh-words appearing in embedded interrogatives, e.g. “I wonder *what* she said”.

The current simulation world consists of five patches, representing five slots in the left periphery. This number reflects a moderately complex left-peripheral configuration, suitable for modeling the interactions of multiple constituents. While the default setting is five, the code can be easily edited to add more patches without affecting the simulation's functionality or application.

4.0.2. *Setup and Initialization*

The model initializes with a single row of patches. Users can choose from the following initialization modes:

- **Random Setup:** generates a row of patches with random states.
- **Manual Setup:** allows the user to input a specific sequence of patches. For example, entering `Mod Int CT Fam Rel` creates a row with a Mod, a Int, a CT, a Fam and a Rel patch in this exact order.
- **Test Setup:** generates a predefined configuration for debugging or quick comparisons.

Once an initial row of patches has been setup, clicking on "go" or "continuous go" will make the simulation run.

4.0.3. *Significance of Rows in the Simulation*

Rows represent sequential steps in the reorganization of left-peripheral elements.

- **Initial Row:** the topmost row represents the initial configuration of left-peripheral elements, which is unordered or only partially ordered. This state serves as the starting point for reorganization, analogous to an early stage in syntactic derivation.
- **Intermediate Rows:** each subsequent row captures a step in the reorganization process. During these stages, elements interact locally with their neighbors, swapping positions when precedence rules dictate they should. This stepwise progression models how hierarchical structures might emerge iteratively, through the adjustment and reordering of smaller units.
- **Final Row:** the bottommost row represents the fully derived structure, where elements are arranged in accordance with precedence rules and compatibility constraints. This state corresponds to the final, stable word order configuration of the left periphery.

This row-based progression in the simulation offers a computational interpretation of incremental structure-building in syntax. While cartographic approaches traditionally emphasize static templates, this simulation allows for an exploration of how such templates could arise dynamically through local interactions and constraints. By simulating the derivation as a series of discrete steps, the model highlights the potential for left-peripheral configurations to emerge through a process of iterative reorganization.

4.0.4. *Rules Governing Patch Interactions*

The core innovation of this model lies in its use of local rules to govern interactions between left-peripheral constituents. The simulation evaluates pairs of neighboring patches sequentially, applying specific rules to decide their relative order.

For instance, consider a pair of patches, A and B. If the initial configuration is A >B, but a local rule dictates that B should precede A, the simulation will swap their positions in the next iteration (represented as the subsequent row in the cellular automaton). This process repeats iteratively: each neighboring pair of patches is evaluated and reordered as needed. Through these incremental adjustments, the overall word order emerges as the cumulative result of these localized interactions, eliminating the need for a predefined global structure. This dynamic approach mirrors how complex systems produce coherent patterns from simple, distributed processes.

The simulation allows users to explore different ways of defining the local rules that govern the behavior of left-peripheral elements. In the current implementation, the local rules are divided into two types:

- Precedence Rules
- Mutual Exclusivity Rules

In the current implementation, precedence rules are modeled as a localized version of the cartographic hierarchy (8). Each patch state is assigned a specific precedence rank, represented as a numerical value. Lower values indicate higher precedence, meaning elements with lower ranks are given priority in ordering. The arrangement of patches is determined by comparing the precedence ranks of neighboring patches and swapping them if necessary.

For example:

- Relative pronouns (**REL**) have the highest precedence (rank 1).
- Contrastive Topics (**CT**) have lower precedence (rank 2).
- Modifiers (**Mod**) have an even lower precedence (rank 7).

The simulation determines whether a swap is required between neighboring patches using the `should-swap?` procedure. A swap occurs if the right-hand patch has higher precedence (a lower rank number) than the left-hand patch.

```
to setup-precedence
  set precedence-hierarchy table:make
  table:put precedence-hierarchy 9 1 ; REL
  table:put precedence-hierarchy 7 2 ; ST
  table:put precedence-hierarchy 6 3 ; CT
  table:put precedence-hierarchy 4 4 ; Int
  table:put precedence-hierarchy 8 5 ; Foc
  table:put precedence-hierarchy 5 6 ; Fam
  table:put precedence-hierarchy 1 7 ; Mod
  table:put precedence-hierarchy 2 8 ; QEmb
```

```
end

to-report precedence-rank [patch-state]
  if table:has-key? precedence-hierarchy patch-state [
    report table:get precedence-hierarchy patch-state
  ]
  report 999 ;; Default for unknown states (lowest priority)
end

to-report should-swap? [left-patch-state right-patch-state]
  let left-rank precedence-rank left-patch-state
  let right-rank precedence-rank right-patch-state

  ;; Swap if the right state has higher precedence (lower rank number)
  if right-rank < left-rank [report true]

  ;; Otherwise, no swap
  report false
end
```

Listing 1. Code for Precedence Setup and Swap Check

In addition to precedence rules, the simulation includes mutual exclusivity constraints. These constraints ensure that certain combinations of patch states cannot appear adjacent to one another. If two mutually exclusive states occur next to each other, the simulation terminates when that pair of patches is evaluated. The mutual exclusivity rules implemented in this simulation are as follows:

- **WH** (*wh*-elements) and **INT** (interrogative complementizers) cannot appear adjacent to each other.
- **INT** and **REL** (relative pronouns or declarative complementizers) are mutually exclusive.
- **WH** and **REL** cannot co-occur in adjacent positions.

Crucially, the final word order configuration in this simulation always emerges from local updates rather than relying on a predefined global template. No fixed blueprint is required to define all positions in advance; instead, the word order arises dynamically through iterative applications of local precedence and mutual exclusivity rules.

This implementation represents one possible way of structuring local rules. Users can experiment with alternative configurations to explore different syntactic theories, model particular languages or test different hypotheses. For example:

- Some patch states could be assigned the same precedence rank, introducing variability in their final positioning.
- Left-peripheral constituents could be conceptualized using a different inventory of patch states.
- Precedence could be determined through alternative mechanisms, rather

than by defining a strict precedence hierarchy. For example, one possible approach is to group patch states into priority zones, where states within the same zone are treated as equivalent at a coarse level of precedence. Patches would first be sorted by zone, and then their order within each zone could be refined using finer precedence rules.

This flexibility allows researchers to investigate a wide range of syntactic configurations without being constrained by a fixed global template, making the simulation a versatile tool for exploring dynamic syntactic structures.

4.0.5. *Update Modes: Cascade vs. Strict Pairwise*

The model allows users to select between two update modes, which govern how the system processes interactions: *Cascade*, and *Strict Pairwise*.

In *Cascade* mode, the evaluation and potential swap of a pair of patches can ripple backward, affecting earlier patches in the row. This creates a dynamic chain reaction where a single change may propagate through the system. Cascade mode simulates a system where interactions are interdependent, leading to emergent patterns that reflect the influence of localized feedback loops.

In *Strict Pairwise* mode, only the currently evaluated pair of patches can be rearranged. Older patches remain fixed, even if the new swap would affect their positioning. This results in a more controlled progression where each interaction is isolated from the rest of the row.

The comparison between Cascade and Strict Pairwise modes was inspired by linguistic theories such as Chomsky's concept of *phases* [11]. In syntax, phases are discrete chunks of a sentence that are worked on and then "frozen", meaning they are no longer accessible for further modifications. This concept raises interesting questions about the dynamics of left-peripheral organization.

- **Cascade Mode:** cascade mode represents a dynamic system where changes propagate backward. Each interaction can influence earlier patches, creating a fluid and interconnected process of reorganization. This mode reflects a scenario where the left periphery is treated as a single, flexible structure, with no restrictions on revisiting previously modified elements.
- **Strict Pairwise Mode:** strict Pairwise mode reflects the idea of "freezing" configurations once they are processed. When a pair of patches is evaluated and potentially swapped, earlier patches remain fixed, regardless of how the new change might affect them. This mode mirrors the phase-based view that syntactic structures may become inaccessible after they have been worked on.

By comparing these two modes, the model explores how different assumptions about the accessibility of previously modified elements impact the emergence of syntactic word order. This comparison provides insights into:

- How localized interactions are influenced by constraints on revisiting earlier configurations.
- Whether "frozen" configurations lead to grammatical word order patterns.
- The implications of phase-like constraints for the dynamic organization of the left periphery.

4.1. *The Simulation in Practice*

To illustrate the problem this simulation addresses, consider a scenario in which a speaker must construct a left periphery containing the following five elements: a Contrastive Topic, a Modifier, a Focus, a Familiar Topic, and a Shifting Topic. What the speaker needs to figure out is the relative order in which these elements will surface.

In a cartographic approach, the order of these elements would be determined by mapping each constituent to a predefined functional slot in a rigid hierarchical template. Using the hierarchy outlined in 8 (repeated below), each element is assigned to its corresponding slot:

(13) Force >Shift >Contr >Int >Foc >Fam >Mod >QEmb >Fin

Following this template, the elements would be placed as follows:

(14) Force >**Shift** >**Contr** >Int >**Foc** >**Fam** >**Mod** >QEmb >Fin

Although not all slots in the hierarchy are filled, the entire template is assumed to exist in the underlying representation, dictating the final word order.

In contrast, the dynamic model implemented in this simulation eliminates the need for an underlying template of dedicated slots where to "plug gin" the different constituents. Instead, the order of elements emerges dynamically through local interactions governed by pre-defined rules. Each element interacts only with its immediate neighbor.

To determine the final order of the constituents using the simulation, we first initialize the simulation with the appropriate patch states:

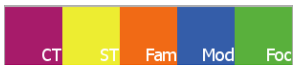


Fig. 5. Initial Setup

The initial setup consists of a Contrastive Topic (patch state = CT), a Modifier (patch state = Mod), a Familiar Topic (patch state = Fam), a Focus (patch state = Foc), and a Shifting Topic (patch state = ST).

The simulation proceeds iteratively, evaluating pairs of neighboring patches from right to left.

- **First Evaluation:** in the second row, the simulation evaluates the right-most pair of patches: **Mod** and **Foc**. Since **Foc** has a higher precedence rank than **Mod**, the two elements are swapped. The relative order of these two patches is now **Foc** >**Mod**.
- **Second Evaluation:** in the third row, the second rightmost pair of patches is evaluated: **Fam** and **Foc**. Again, **Foc** has a higher precedence rank than **Fam**, so the two elements are swapped. The relative order is now **Foc** >**Fam**.

This process continues row by row, with each pair of neighboring patches evaluated and swapped if necessary, until no further swaps are required. At this point, the derivation is complete, and the configuration becomes stable. The last row of the CA displays the word order configuration that the speaker will actually produce.

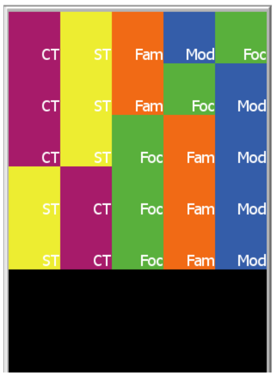


Fig. 6. Final Setup

This approach allows the final order to emerge naturally through a stepwise process of negotiation, without assuming an overarching blueprint. This means that while the rules governing the interactions between patches (representing the different left-peripheral elements) remain consistent throughout the simulation, the final configuration varies depending on the specific elements present at the start of the derivation, ensuring that the outcome is tailored to the initial input.

5. Discussion

The simulation successfully captures all the patterns modeled by the hierarchy in 8, but only when the Cascade mode is selected. This result confirms the feasibility of using local updates and dynamic emergent structures to model word order phenomena.

In contrast, the Strict Pairwise mode frequently fails to produce grammatical configurations, especially in cases requiring significant reordering. The inclusion of two update modes—Cascade and Strict Pairwise—allowed for a comparison of interaction mechanisms and their impact on word order derivations.

5.1. *Cascade Mode vs. Strict Pairwise Mode*

Cascade mode incorporates a feedback mechanism, where changes to a pair of patches propagate dynamically, revisiting and adjusting earlier pairs when necessary. This approach ensures:

- Grammatical configurations consistently emerge, adhering to observed syntactic constraints.
- Resulting structures align with predictions from cartographic models, demonstrating that localized rules are sufficient for deriving complex word orders.

Strict Pairwise mode evaluates and updates one pair of patches at a time, without revisiting earlier pairs. While this stepwise process stabilizes configurations more quickly, it frequently fails to produce grammatical configurations due to the inability to correct misalignments introduced by later interactions.

The comparison highlights the critical role of feedback mechanisms in dynamic systems. Cascade mode's ability to consistently generate grammatical configurations demonstrates the importance of cyclic processes, where earlier states influence subsequent steps, much like cyclic derivations in syntactic theory.

5.2. *Implications for Syntactic Theory*

The success of this dynamic simulation under Cascade mode challenges the necessity of pre-defined cartographic hierarchies for explaining left-peripheral syntax. Traditional frameworks, such as Rizzi's cartographic model, assume a rigid hierarchy universally present in the underlying structure. In contrast, this dynamic model demonstrates that grammatical outputs can emerge from localized interactions without relying on static templates.

Viewing the left periphery as a complex system shifts the focus from static representations to dynamic processes. This approach bridges linguistic theory with complexity science, offering a computationally grounded alternative for exploring syntax. By emphasizing emergent properties, the model provides a parsimonious explanation for syntactic variation and flexibility.

The Strict Pairwise mode's inability to consistently produce grammatical configurations highlights the limitations of rigid, stepwise processes. While computationally efficient, this mode fails to capture the feedback and flexibility required for dynamic syntactic phenomena. These findings suggest that linguistic derivations inherently depend on feedback mechanisms, akin to those implemented in Cascade mode, to replicate observed patterns effectively.

5.3. *Future Directions*

This study's exploratory nature points to several promising avenues for further development:

- **Empirical Exploration:** extending the model to a wider range of linguistic data and syntactic phenomena could further demonstrate its flexibility and refine its predictive capacity.
- **Integration with Broader Syntax:** extending the model to interact with other syntactic domains, such as verb phrases or adverbial hierarchies, could provide a more comprehensive account of sentence structure.
- **Application to Other Domains:** the cellular automaton framework could be applied to syntactic phenomena beyond the left periphery, such as Cinque's adverbial hierarchy [12], to explore its broader applicability.
- **Alternative Precedence Rules:** exploring alternative ways to define precedence rules could yield valuable insights. For example, precedence relations could be simplified or modeled differently, potentially reducing the complexity of rules needed to account for left-peripheral word order.

These extensions would enhance the model's relevance and capacity to contribute to the study of dynamic syntactic processes, offering new insights into the interplay between localized rules and emergent structures.

6. Conclusion

This paper has introduced a novel approach to modeling left-peripheral word order by leveraging principles from complex systems and implementing them through a cellular automaton framework. By replacing the rigid, pre-defined templates of traditional cartographic models with a system where word order emerges through localized interactions, this model offers a new perspective on modeling word order phenomena. The results demonstrate that such a dynamic, localized approach can reliably produce grammatical configurations, particularly under the Cascade update mode, and align with linguistic observations.

The study highlights the potential of treating the left periphery as a complex system, emphasizing the role of feedback mechanisms and emergent patterns in capturing syntactic phenomena. By doing so, it not only challenges the necessity of fixed hierarchical templates but also aligns with minimalist principles that prioritize simplicity and flexibility in explaining linguistic structures [10, 11].

While the findings are promising, this work represents an initial step toward rethinking the dynamics of left-peripheral syntax. Future work could extend the model to incorporate interactions with other syntactic domains or adapt it to test a broader range of cross-linguistic data. By bridging insights from linguistics and computational modeling, this study highlights the value of dynamic, context-sensitive approaches in advancing our understanding of syntax.

References

[1] Anderson, P. W., More is different, *Science* **177** (1972) 393–396, doi:10.1126/science.177.4047.393.

[2] Artime, O. and De Domenico, M., From the origin of life to pandemics: Emergent phenomena in complex systems, *arXiv preprint arXiv:2205.11595* (2022), `https://arxiv.org/abs/2205.11595`.

[3] Beaver, D. R. and Clark, B. Z., *Sense and Sensitivity: How Focus Determines Meaning* (Wiley-Blackwell, Malden, MA, 2008).

[4] Benincà, P., The position of topic and focus in the left periphery, in *Current studies in Italian syntax: Essays offered to Lorenzo Renzi*, eds. Cinque, G. and Salvi, G. (Elsevier, 2001), pp. 39–64.

[5] Bresnan, J., *Lexical-Functional Syntax* (Blackwell, Oxford, 2001).

[6] Carnie, A., *Syntax: A Generative Introduction*, 2nd edn. (Wiley-Blackwell, Malden, MA, 2007).

[7] Chafe, W., Givenness, contrastiveness, definiteness, subjects, topics, and point of view, in *Subject and Topic*, ed. Li, C. N. (Academic Press, New York, 1976), pp. 25–55.

[8] Chomsky, N., *Syntactic Structures* (Mouton, The Hague, 1957).

[9] Chomsky, N., *Aspects of the Theory of Syntax* (MIT Press, Cambridge, MA, 1965).

[10] Chomsky, N., The minimalist program (1995).

[11] Chomsky, N., Derivation by phase, in *Ken Hale: A Life in Language*, ed. Kenstowicz, M. (MIT Press, Cambridge, MA, 2001), pp. 1–52.

[12] Cinque, G., *Adverbs and Functional Heads: A Cross-Linguistic Perspective* (Oxford University Press, Oxford, 1999).

[13] Cinque, G. and Rizzi, L., The cartography of syntactic structures, *Studies in Linguistics* **2** (2008) 42–58.

[14] Frascarelli, M., The syntax-discourse interface and the left periphery: The role of topics in interpreting sentence structure, in *The Handbook of Syntax* (Wiley, 2007), pp. 159–176.

[15] Frascarelli, M., The interpretation of discourse categories: Cartography for a crash-proof syntax, in *Enjoy Linguistics! Papers offered to Luigi Rizzi on the occasion of his 60th birthday*, eds. Bianchi, V. and Chesi, C. (Siena, CISL Press, 2012), pp. 180–191.

[16] Frascarelli, M. and Hinterhölzl, R., Types of topics in german and italian, in *On information structure, meaning and form: Generalizations across languages*, eds. Schwabe, K. and Winkler, S. (John Benjamins, 2007), pp. 87–116.

[17] Gell-Mann, M., *The Quark and the Jaguar: Adventures in the Simple and the Complex* (W.H. Freeman, New York, 1994).

[18] Halliday, M. A. K., Notes on transitivity and theme in english: Part 2, *Journal of Linguistics* **3** (1967) 199–244, doi:10.1017/S0022226700016613.

[19] Halliday, M. A. K. and Hasan, R., *Cohesion in English* (Longman, London, 1976).

[20] Harley, H., *English Words: A Linguistic Introduction*, 2nd edn. (Wiley-Blackwell, Malden, MA, 2017).

[21] Higginbotham, J., Interrogatives, in *The View from Building 20: Essays in Linguistics in Honor of Sylvain Bromberger*, eds. Hale, K. and Keyser, S. J. (MIT Press, Cambridge, MA, 1993), pp. 195–227.

[22] Holland, J. H., *Hidden Order: How Adaptation Builds Complexity* (Perseus Books, Reading, Massachusetts, 1995).

[23] Jackendoff, R., *X-Bar Syntax: A Study of Phrase Structure* (MIT Press, Cambridge, MA, 1977).

[24] Krifka, M., Basic notions of information structure, in *Interdisciplinary Studies on*

Information Structure, eds. Féry, C., Fanselow, G., and Krifka, M., Vol. 6 (Universitätsverlag Potsdam, Potsdam, 2007), pp. 13–56.

[25] Lambrecht, K., *Information Structure and Sentence Form: Topic, Focus, and the Mental Representations of Discourse Referents* (Cambridge University Press, Cambridge, UK, 1994).

[26] Mitchell, M., *Complexity: A Guided Tour* (Oxford University Press, New York, 2009).

[27] Rizzi, L., The fine structure of the left periphery, in *Elements of grammar: Handbook in generative syntax*, ed. Haegeman, L. (Kluwer, 1997), pp. 281–337.

[28] Rizzi, L., On the position "int(errogative)" in the left periphery of the clause, in *Current studies in Italian syntax: Essays offered to Lorenzo Renzi*, eds. Cinque, G. and Salvi, G. (Elsevier, 2001), pp. 287–296.

[29] Rizzi, L., Locality and left periphery: Structures and beyond, in *The cartography of syntactic structures 3* (2004), pp. 223–251.

[30] Rizzi, L., The structure of cp and ip: The cartography of syntactic structures, vol. 2, in *The cartography of syntactic structures* (Oxford University Press, 2004), pp. 52–75.

[31] Rizzi, L., Some issues in the cartography of the left periphery, Gent Workshop, May 13-15 2011 (2011).

[32] Rizzi, L., Notes on cartography and further explanation, *International Journal of Latin and Romance Linguistics* **25** (2013) 197–226.

[33] Rizzi, L. and Bocci, G., The left periphery of the clause, in *The Companion to Syntax, 2nd edition*, eds. Everaert, M. and van Riemsdijk, H. (Blackwell, 2017).

[34] Steedman, M., *The Syntactic Process* (MIT Press, Cambridge, MA, 2000).

[35] Uriagereka, J., An f position in western romance, in *Discourse configurational languages* (1995), pp. 153–175.

[36] Varley, T. F., Causal emergence in discrete and continuous dynamical systems, *arXiv preprint arXiv:2003.13075* (2020), `https://arxiv.org/abs/2003.13075`.

[37] Von Neumann, J., *Theory of Self-Reproducing Automata* (University of Illinois Press, Urbana, IL, 1966).

[38] Wilensky, U., *NetLogo*, Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL (1999), `http://ccl.northwestern.edu/netlogo/`.

[39] Wolfram, S., Statistical mechanics of cellular automata, *Reviews of Modern Physics* **55** (1983) 601–644, doi:10.1103/RevModPhys.55.601.

[40] Yuan, B. *et al.*, Emergence and causality in complex systems: A survey on causal emergence and related quantitative studies, *arXiv preprint arXiv:2312.16815* (2023), `https://arxiv.org/abs/2312.16815`.