Can Word Order in the Left Periphery Emerge from Local Interactions? A Complexity-Inspired Simulation Framework

Elena Callegari[1]
University of Iceland

ABSTRACT

This paper presents a novel framework for modeling the Left Periphery as a complex system, where surface word order emerges from local interactions rather than from fixed global templates. Drawing on tools and methodologies from complexity science, I implement an agent-based simulation in NetLogo to investigate whether word order patterns traditionally captured by cartographic approaches can instead arise through simple, local precedence rules and compatibility constraints. Constituents are modeled as agents whose order is incrementally derived via repeated local updates, without access to a global blueprint, paralleling self-organizing behaviors observed in systems such as bird flocks, neural networks, and cellular automata. Using Rizzi & Bocci (2017) and Frascarelli (2012) as contrasting input models, I show that emergence succeeds only when all constituents fully participate in local reordering. Structures that allow unranked or opaque elements (e.g., Rizzi's broad "Top" category) introduce interactional failures that result in incorrect surface orderings.

Keywords: Word Order; Left Periphery; Cellular Automata, Complexity; Complex Systems; Computational Modeling; cartography

Article Under Review, (2025), pp. 1–45

1 INTRODUCTION

In nature and society, complex patterns often emerge without any centralized design. Bird flocks, ant colonies, traffic flows and neural networks all exhibit organized global behavior that arises from the repeated application of simple local rules. Rather than following a master plan or a fixed underlying blueprint, these systems self-organize: small, local interactions accumulate over time to produce coherent, large-scale structure.

This paper asks whether a similar principle might apply to syntax. Specifically, I examine the Left Periphery of the clause, a domain often modeled as a rigid hierarchy of functional projections, and explore whether its apparent complexity could instead emerge from decentralized, interaction-driven processes. Rather than assigning constituents to fixed positions within a global structural template, I investigate whether well-formed word orders can emerge from the repeated application of strictly local rules.

In syntax, the term *Left Periphery* (LP) designates the structural domain at the *left* edge of the clause. In English, this region typically includes all constituents that precede the canonical subject position. For instance, the examples below illustrate the presence of a topicalized constituent in 1, and a wh-element together with a fronted auxiliary in 2. In each case, the left-peripheral material is highlighted in boldface:

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

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

The LP is crucial for the interpretation of clause type and discourse-related properties such as the encoding of new vs. presupposed information. It is the structural region that hosts elements mediating between sentence-level syntax and discourse-level interpretation, signaling distinctions such as interrogativity vs. declarative force, and the division between old and new information (see Rizzi (1997, 2001, 2004a,b); Cinque and Rizzi (2008); Rizzi (2011); Frascarelli (2012); Benincà and Poletto (2004b); Haegeman (2012); Frascarelli and Hinterhölzl (2007); Rizzi (2013); Rizzi and Bocci (2017); Lambrecht (1994); Benincà (2001); Aboh (2004); Poletto (2000)).

A striking feature of the LP is its ability to host a wide range of different types of linguistic elements, each arising from distinct grammatical processes. For example, consider the following sentence from Italian, a language with particularly rich left-peripheral structures (Rizzi 1997, 2001, 2004a,b; Rizzi and Bocci 2017; Benincà 2001; Benincà and Poletto 2004a; Frascarelli and Hinterhölzl 2007; Haegeman 2012; Samek-Lodovici 2006a; Poletto 2000; Haegeman 2004; Abels 2012):

(3) Ieri, a Luigi, perché improvvisamente il libro Maria
yesterday to Luigi why suddenly the book Maria
glielo ha tirato addosso?
3.DAT = 3.ACC has thrown against
'Why did Maria suddenly throw the book at Luigi yesterday?'

In this sentence, five left-peripheral elements are present: the temporal adjunct *Ieri* ('yesterday'), the topicalized, clitic-resumed indirect object *a Luigi* ('to Luigi'), the interrogative operator *perché* ('why'), the manner adverb *improvvisamente* ('suddenly'), and a second topicalized constituent, the direct object *il libro* ('the book').

Traditional approaches to modeling the LP, particularly within generative linguistics, have very often relied on cartographic frameworks (Rizzi 1997, 2001, 2004a,b; Cinque and Rizzi 2008; Rizzi 2011; Frascarelli 2012; Frascarelli and Hinterhölzl 2007; **?**; Rizzi and Bocci 2017; Benincà and Poletto 2004a; Belletti 2004; Haegeman 2004; Poletto 2000; Munaro *et al.* 2001; Aboh 2004; Benincà 2006; Bocci 2013; Rizzi and Shlonsky 2007), i.a. These models posit that the LP consists of a rigidly ordered hierarchy of functional projections, each assigned a specific grammatical or discourse function. Under this view, every constituent that may appear in the LP is assigned a predetermined structural position within the hierarchy. Such approaches thus rest on the assumption of a fixed, universal "blueprint" for clause structure: the Left Periphery is not assembled dynamically or contextually, but projected wholesale from Universal Grammar, its functional projections predetermined in both nature and position.

This paper explores whether word order in the LP can be captured through an alternative, dynamically driven approach. Specifically, drawing on insights from complexity science and complex systems modeling, I investigate whether localized interactions among LP

constituents are sufficient to derive grammatical word order patterns, without having to assuming an underlying global blueprint.

To explore this hypothesis, I present a simulation built using the NetLogo agent-based modeling environment (Wilensky 1999; Wilensky and Rand 2015; Wilensky 2024), which, to the best of my knowledge, constitutes the first dynamic computational model of word order in the Left Periphery. In this framework, each constituent is modeled as an agent that interacts only with its immediate neighbor, attempting to satisfy a set of local precedence and compatibility rules. Over time, these local operations accumulate into a global ordering. To evaluate the model's performance, I test it against two cartographic proposals that make different assumptions about the structure of the Left Periphery. The first, from Rizzi and Bocci (2017), introduces projections such as QEmb but treats topics as a unified category. The second, from Frascarelli (2012), draws finer distinctions among topic types and assigns them to separate structural positions. These contrasting hierarchies provide predicted surface orders, which I use as targets for comparison. The results of the simulation show that local interactions alone can generate grammatical word order configurations under the right conditions. If certain elements are opaque or fail to participate in local reordering (such as Rizzi's broad 'Top' category), however, the system fails to produce the target configuration.

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 this syntactic domain by applying techniques and insights from complexity science to model its dynamics.
2. To introduce a novel, complexity-science-inspired simulation framework for modeling word order, designed as reusable computational infrastructure. While this paper focuses on the LP, the framework is generalizable: it can be adapted to test other syntactic domains. By requiring theoretical assumptions to be implemented as explicit, executable rules, the model functions as a testbed not just for emergent accounts, but for evaluating the internal consistency of LP theories more broadly.
3. To use this framework as an epistemic probe of word order in

the Left Periphery, testing whether grammatical sequences can be derived through local updates alone, and identifying the specific structural conditions under which this is possible.

The remainder of this paper is organized as follows: Section 2 reviews cartographic models of the Left Periphery, their empirical claims, theoretical variants, and assumptions about universality. Section 3 argues that the Left Periphery exhibits hallmarks of a complex system (numerosity, non-linearity, and emergence), which motivates a complexity-theoretic approach. Section 4 introduces the modeling framework, motivating the use of one-dimensional multi-state cellular automata and describing the specific NetLogo implementation. Section 5 details the simulation's interface, category inventories, and local-interaction architecture. Section 6 presents simulation results for the two reference hierarchies, testing whether their predicted word orders can be generated from local rules alone. Section 7 interprets these results. Finally, Section 8 summarizes the findings, broader contributions, and potential of the NetLogo framework as a reusable, computationally explicit testbed for word-order theories.

2 THE STRUCTURE OF THE LEFT PERIPHERY

The Left Periphery (LP) has been a focal point of research in generative syntax, particularly within the cartographic tradition. Cartographic models posit that the LP comprises a hierarchically ordered sequence of functional projections, each associated with a distinct grammatical or discourse-related function (Rizzi 1997, 2001, 2004a,b; Cinque and Rizzi 2008; Rizzi 2011; Frascarelli 2012; Frascarelli and Hinterhölzl 2007; Rizzi 2013; Rizzi and Bocci 2017) i.a. Within this framework, constituents such as topics, foci, wh-elements, and fronted modifiers are mapped onto structurally dedicated positions, or "slots", within a fixed syntactic hierarchy. For example, Rizzi and Bocci (2017) propose the following sequence of projections:

(4) Force > Top* > Int > Top* > Foc > Top* > Mod > Top* > QEmb > Fin

In 4, the symbol "$>$" indicates linear precedence. Each of these projections is associated with a distinct interpretive or grammatical function:

- **Force** encodes clause type and illocutionary force. It typically hosts finite declarative complementizers (e.g., Italian *che*, 'that'), as well as relative pronouns (Rizzi 1997, 2001).
- **Top** is a recursive (as indicated by the * sign) Topic projection. Topics are generally interpreted as referring to discourse-given material. In languages such as Italian, topicalized constituents are frequently resumed within the clause by clitic pronouns (Rizzi 1997; Benincà and Poletto 2004a).
- **Int** hosts interrogative complementizers, such as *if* in embedded polar questions, as well as base-generated wh-elements like *why* (Rizzi 2001)
- **Foc** hosts fronted focalized constituents, e.g. constituents typically carrying new, contrastive, or corrective information (Rizzi 1997; Bocci 2013; Benincà and Poletto 2004a).
- **QEmb**: introduced by Rizzi and Bocci (2017), this specialized projection hosts wh-elements in embedded interrogatives.
- **Fin** marks the lower boundary of the Left Periphery and connects the CP layer to the IP/TP domain, and may host elements such as auxiliary verbs in V-to-C movement or non-finite complementizers (Rizzi 1997, 2004b).

Cartography thus views the Left Periphery as a highly articulated domain whose structure is fixed.

Within this framework, a central question concerns the ontological status of the cartographic hierarchy. Do all projections exist in every clause, regardless of whether they are overtly filled? Or are only necessary heads being projected?

Rizzi (1997) takes a cautious stance, formulating the economy principle *Avoid Structure*, which favors structural parsimony in the clause spine. In his account, the split between Force and Fin(iteness), for instance, is permitted only when forced by the activation of the Topic–Focus field; otherwise, English will realize Force and Fin on a single head. This "last resort" use of a richer CP structure ensures that additional functional projections appear only when required by selec-

tional or interpretive constraints. Thus, while the full articulated CP is available in principle, its projections are inserted only as needed, with *Avoid Structure* blocking unnecessary layers.

Many later works assume that the entire left-peripheral hierarchy is universally present in the syntactic structure, albeit sometimes with silent heads. For instance, Petrosino (2017) notes that "it is usually assumed that the full cartographic structure is always fully projected", even when no overt material occupies each position. Similarly, Shlonsky and Bocci (2019) characterize the cartographic program as positing a universal, fine-grained hierarchy of functional projections and treat parametric variation largely in terms of whether particular heads or specifiers are pronounced or not, presupposing structural presence even when silent. On this view, the sequence itself is part of Universal Grammar, and its positions are structurally represented in every clause, whether or not they are overtly realized.

Some authors offer a more flexible point of view, allowing for pruning of unmarked projections. Building on Starke (2004)'s feature-based articulation of the Left Periphery, Clercq (2017) for instance assumes that all left-peripheral features are present for interpretation (LF), while unmarked values (e.g., [-WH], [-FOC]) need not be realized in narrow syntax. This preserves the universality of the functional sequence while permitting structural economy.

Even within Rizzi's own work, the fixedness of the hierarchy is emphasized. In a later paper, Rizzi (2013) draws a comparison between the cartographic hierarchy and the DNA sequence. Just as the sequence of nucleotides in DNA is not itself a primitive but nevertheless a real and functionally significant property of biological systems, so too is the relative ordering of syntactic projections understood as a substantive component of the human language faculty. The hierarchy is treated as an observable, biologically grounded structure (see also Rizzi and Bocci (2017) for a restatement of this point).

This view is echoed in related work by Cinque (1999), who argues for a fixed universal hierarchy of adverb classes mapped to a sequence of clausal functional heads; in his analysis, AdvPs occupy the specifiers of distinct functional projections in a rigid order, with the relevant heads sometimes silent. Later work by Cinque (2013) and Cinque and Rizzi (2010) reiterates the expectation of cross-linguistic uniformity at the level of the functional sequence.

Taken together, these works underscore the strength of the cartographic hypothesis. Whether interpreted in its "strong" form (as a fully articulated sequence of functional projections present in every clause, regardless of overt realization) or in its "weaker" form (as a universally available template from which individual languages may prune unused projections), the cartographic hierarchy is assumed to be an invariant structural backbone encoded in Universal Grammar.

The assumptions of the cartographic model can be illustrated with an analogy to a shape-sorting toy, the kind often given to toddlers (see Fig. 1). In such a toy, each opening is cut into a specific shape, and each block corresponds to one of these openings. First, there is a rigid mapping between form and position: a block can only fit into the slot whose shape matches it. Second, the arrangement of slots is fixed: their relative positions never change, regardless of which shapes are actually inserted. Third, the toy's internal structure is always complete: all slots are always present, even if no block occupies them at a given moment.

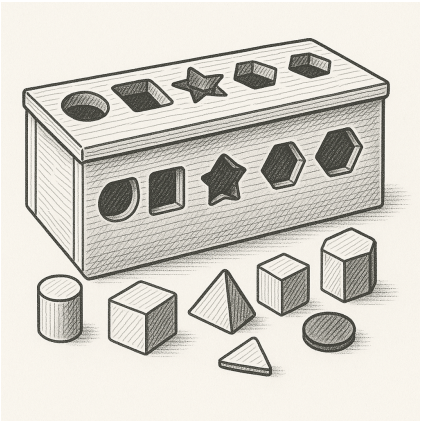


Figure 1: Shape Sorting Box.

The cartographic model treats the Left Periphery as a syntactic "shape-sorting toy": each possible constituent type (the "blocks" in the shape-sorting toy) is associated with a dedicated functional slot. The relative order of the different functional slots is then fixed. Just as the toy enforces a rigid match between form and position, the cartographic model enforces a rigid structure in which each constituent type can

appear only in its designated projection. The projections are taken to be real components of the grammar (Rizzi 2013), and the entire hierarchy is assumed to always be accessible at some level, even if some positions remain empty in actual clauses.

The cartographic approach has proven highly effective in identifying which elements may appear in the Left Periphery, specifying their interpretive roles, and uncovering structural commonalities across languages. However, assuming a universal template that guides sentence production and constrains constituent order is a strong theoretical commitment, particularly given the level of granularity that many of these hierarchies assume (see also Newmeyer (2004) for a critique of the cartographic approach). While this commitment may ultimately be justified (perhaps language users do rely on richly specified, globally ordered templates encoded in Universal Grammar), it is not the only conceivable explanation for the observed patterns.

In this paper, I explore whether comparable structural configurations could arise through a fundamentally different mechanism: one that replaces global blueprints with locally governed reordering operations. If the same surface outcomes can be derived without assuming a fixed underlying hierarchy, this opens up new possibilities for understanding how word order might be computed, and invites reconsideration of how much structure must be specified in advance versus derived through interaction. This alternative perspective is informed by principles from complexity science, where global behavior often emerges from distributed, local interactions.

3 WHAT IS A COMPLEX SYSTEM? AND WHY MODEL THE LEFT PERIPHERY AS ONE

This section develops the first goal of the paper: to highlight that the LP exhibits several of the key properties commonly characterizing complex systems. While this observation does not necessarily entail a specific theoretical architecture, it suggests that we may gain a deeper understanding of this syntactic domain by applying concepts and techniques from complexity science.

Complex systems are all around us. From bird flocks and traffic flows to social networks and the brain, they exemplify cases where structure does not arise from a central blueprint or centralized controller, but from the cumulative effects of local interactions (Holland 1992; Simon 1962; Waldrop 1993; Newman 2011; Sayama 2015). These systems exhibit coherent, adaptive behavior without any centralized control, often defying prediction despite the simplicity of their underlying rules.

In bird flocking, for example, hundreds or even thousands of birds move together in tight, coordinated formations, creating intricate shapes that seem almost choreographed. Watching these fluid, shifting patterns unfold across the sky, one might assume there must be a central plan dictating their movement (or perhaps a leader bird guiding the rest of the flock), but that is not the case. Instead, each bird follows simple local rules, e.g. adjusting its speed and direction based on just a few nearby neighbors (Reynolds 1987; Ballerini *et al.* 2008; Couzin *et al.* 2005). From these small-scale interactions, a globally coherent and dynamic structure emerges (Sumpter 2006). This kind of self-organization (complexity arising without centralized control) is a hallmark of complex systems, and it provides a powerful lens through which to reconsider syntactic domains like the Left Periphery.

A complex system is typically characterized by three core properties: numerosity, non-linearity, and emergence (Bar-Yam 1997; Miller and Page 2007; Mitchell 2009; Ladyman and Wiesner 2020; Holland 1992; Sayama 2015; Simon 1962; Waldrop 1993). The LP exhibits all three of these properties:

Numerosity Complex systems involve many interacting parts. The LP exemplifies this property in a striking way. One of its most distinctive characteristics (and arguably one of the motivations behind the richly articulated hierarchies proposed in cartographic syntax) is the sheer diversity of constituent types that may appear in this domain. Cross-linguistic evidence shows that the LP can accommodate an extensive and heterogeneous inventory of elements, including multiple types of topics (e.g. aboutness, contrastive, familiar topics (Frascarelli 2007, 2012; Frascarelli and Hinterhölzl 2007)), multiple types of focus (e.g., corrective, mirative, contrastive (Bocci 2013; Bianchi and Bocci 2012; Bianchi *et al.* 2015, 2016)), various complementiz-

ers (e.g., polarity complementizers, finite and non-finite declarative complementizers (Rizzi 1997, 2001; Moscati and Rizzi 2021)), interrogative operators and embedded wh-elements (Rizzi and Bocci 2017), fronted modifiers (e.g., temporal and manner adjuncts, (Bayer 2004; Rizzi 2004a)), fronted verbs in V2 languages (Poletto 2002; Bayer 2004), negative polarity adverbs, and relative operators (**?**), arguably making the LP one of the most diverse and compositionally rich regions of the clause.

Non-linearity In complex systems, the relationship between components and overall system behavior is non-linear. This means that the behavior of the whole cannot be predicted by simply summing the behaviors of its parts. Small changes in one part of the system can produce disproportionately large or unexpected effects elsewhere. In the context of syntax, and the Left Periphery in particular, non-linearity is evident in the way multiple constituents interact to produce grammatical or ungrammatical outcomes. For example, two constituents that are independently licit may yield ungrammaticality or interpretive failure when combined, as seen in island effects and relativized minimality violations (Ross 1967; Rizzi 1990, 2004b). Conversely, some configurations yield facilitative effects: certain topicalizations license verb-third (V3) orders that would otherwise be blocked (Angantýsson 2011, 2007), and a range of factors, including the choice of embedding predicate, discourse-linking, and other structural conditions, can improve acceptability in otherwise illicit extraction contexts (Erteschik-Shir 1973; Hooper and Thompson 1973; Sprouse *et al.* 2016). These patterns highlight that the Left Periphery is not a linear stack of autonomous projections, but an interactive domain whose properties arise from the non-linear interplay of its constituents.

Emergence A core property of complex systems is emergence: system-level properties or behaviors that arise from the interaction of components, and which are not directly encoded in any individual part. In the Left Periphery, topicality offers a compelling example of this phenomenon. Topic projections account for a substantial portion of the articulated CP domain in cartographic models (Rizzi 1997; Rizzi and Bocci 2017; Frascarelli 2012; Frascarelli and Hinterhölzl 2007), yet topic status is not lexically encoded: a constituent such as *John*

can be interpreted as either topic or focus depending entirely on context. Even in languages with overt topic markers (such as Japanese *wa* (Kuroda 2005; Nakanishi 2001) or Korean *nun* (Kim 2021; Choi 1986)), these markers do not reflect a built-in lexical property but instead mark a constituent's role within a clause-level discourse structure. Topicality is therefore not an intrinsic feature of the item itself, but an emergent property that arises from its relationship to other constituents in the clause. Consider, for instance, aboutness topics, which are interpreted as the entity about which the remainder of the clause says something (Frascarelli and Hinterhölzl 2007). Identifying a constituent as an aboutness topic is only possible once the rest of the clause is evaluated as the associated comment. In this way, the topic–comment structure does not originate in any one node, but emerges from the configuration as a whole.

If the LP exhibits the core properties of a complex system, then it is worth exploring whether it can be modeled as one. We might find that such an approach allows us to capture more, or account for certain patterns more effectively. Consider again the case of topics. Cartographic hierarchies, while descriptively powerful, are inherently static: they treat elements like Topic as fixed projections in a predetermined sequence. But as argued above, topicality is an interactional property, not (just) a positional one. Complex systems are specifically designed to model emergent properties. By adopting them as a framework, we may be able to model the Left Periphery in a way that better reflects the dynamics that give rise to its structure [1].

[1] The properties discussed here -numerosity, non-linearity, and emergence- arguably apply to language more broadly, and I do not claim that they are unique to the Left Periphery. However, the LP offers a natural starting point for exploring these dynamics. It brings together a particularly rich and diverse inventory of elements, many of which are either base-generated or moved, and whose interpretation depends on their interaction with other constituents. This makes it a clear and accessible case of syntactic complexity, and a useful initial domain for testing how complex systems approaches might model the emergence of structure from local interactions.

THE FRAMEWORK 4

This section introduces a framework for modeling word order in the Left Periphery using concepts and tools from complex-systems theory. In this framework, word order emerges dynamically from local interactions among its constituents, without relying on a fixed underlying 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 local reordering rules.

The model was implemented using NetLogo (Wilensky 1999; Wilensky and Rand 2015; Wilensky 2024), a programming environment widely used to model complex systems. NetLogo provides an accessible yet powerful platform for simulating the behavior of interacting components over time, making it well-suited for exploring the kinds of localized interactions and emergent patterns that this paper aims to investigate.

The NetLogo code for this simulation has been made available in a public repository to facilitate reproducibility and further experimentation. This way, users can also modify the initial configurations and rules to explore variations of the model. The complete code for the simulation is available at: https://github.com/linguisticsnet/netlogo-anon-model. This repository includes instructions for running the simulation locally or via the NetLogo Web interface after downloading the model file.

Figure 2 provides a first look at the simulation. The model treats word order in the Left Periphery as the result of an incremental reordering process. Each horizontal row corresponds to a step in the simulation, with the top row displaying the initial input sequence of syntactic constituents (e.g., Topic, Focus, Modifier). Different left-peripheral constituents are represented as blocks of different colors. As the simulation progresses row by row, local interaction rules determine how constituents reorganize, with the system updating at each step until a stable configuration is reached.

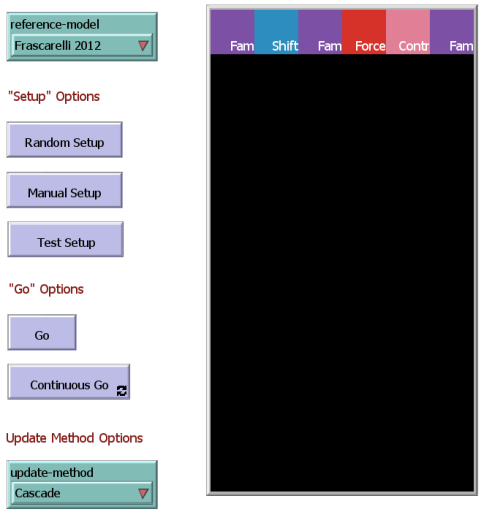


Figure 2: Screenshot of the simulation interface.

4.1 *Using One-Dimensional Cellular Automata to Model Word Order*

Having adopted the perspective that the LP might be fruitfully modeled using tools from complex system modeling, the next step is to identify a modeling framework from complexity science that is compatible with the kinds of structures and processes we aim to capture.

While several modeling approaches from complexity science could in principle serve this purpose, I chose one-dimensional cellular automata (CA) as the basis for the current implementation. Cellular automata are simple yet powerful computational systems in which global complexity emerges entirely from local interactions (Von Neumann 1966; Wolfram 1983). A CA consists of a row (or grid) of "cells", each of which can be in one of a small number of states, most commonly 1 or 0, which can be thought of as "on" and "off" (or, visually, black and white). The system evolves over time in discrete steps: at each step, every cell updates its state based solely on its own current state and the states of its immediate neighbors (typically just the cell to its left and right). These local rules are the only information available to the system; there is no global controller, hierarchy, or template specifying how the final structure should look.

One classic example of CA is Rule 30, a one-dimensional binary CA. It begins with a single black (i.e., "on" = 1) cell in a row of white ("off" = 0) cells. At each time step, the system determines the state of each position for the *next* row based on the pattern formed by that position and its two immediate neighbors in the *current* row. For example, in Rule 30, a position will be black (= 1) in the next row if, in the current row, its left neighbor is white (= 0) and its right neighbor is black (= 1). The complete set of local patterns and resulting states for Rule 30 is shown in Table 1. Repeating these local decision rules across the entire row over time generates a striking triangular pattern with complex, internal structure, as can be seen in 3:

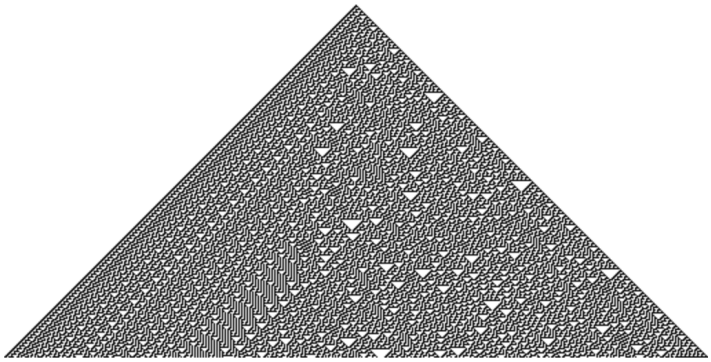


Figure 3: Rule 30

Current pattern	111	110	101	100	011	010	001	000
New state	0	0	0	1	1	1	1	0

Table 1: "Rule 30" update rules. Each pattern maps to the new state of the center cell.

What is remarkable about CA like Rule 30 above is that such complex and seemingly patterned structures emerge from an extremely simple system, i.e. one with no global design, no central controller, and no built-in representation of the final form. The entire structure arises from a single initial condition and a uniformly applied set of

local rules. Despite the absence of hierarchical planning, the system produces intricate patterns that exhibit both local coherence and a global structure. This makes Rule 30 a paradigmatic example of how complexity can be emergent, not imposed, an insight that challenges assumptions about where structure must come from in natural systems, including language.

Although originally developed by von Neumann as a formal model of self-reproducing machines, inspired by biological self-replication (Von Neumann *et al.* 1966; Von Neumann 1966), CA have since been applied to investigate complex systems in a variety of domains, including physics, ecology, and social science.

My own simulation of the LP is modeled as a one-dimensional, multi-state cellular automaton. In a multi-state cellular automaton, each cell can be in more than two possible states, rather than just 'on' or 'off', allowing for a richer range of local configurations and interactions. The decision to use a *one-dimensional* CA stems from the inherent linear nature of word order phenomena, i.e. what we are trying to model with our simulation. 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 generally represented hierarchically in syntax (e.g., tree-like configurations (Chomsky 1957, 1965; Steedman 2000; Jackendoff 1977; Bresnan 2001; Carnie 2007)), these hierarchies must be linearized into a one-dimensional string of words for actual language use.

Since word order is an inherently one-dimensional phenomenon, a single row in a cellular automaton can 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:

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

To represent example 5 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. This could correspond to a model where the cell 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 4 provides a visual representation of this sentence in a CA representation. Each cell represents one of these syntactic constituents, arranged in their unmarked order:

Figure 4: CA representation of unmarked word order in a declarative sentence.

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:

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

Figure 5 illustrates this derived configuration, where the leftmost patch is now assigned state "Ind. Obj", to reflect the left-peripheral position of the indirect object.

The row progression typical of cellular automata can represent transitions between basic and derived word order configurations. For instance, Figure 6 depicts a progression from the basic **Direct Object >Indirect Object** structure to the marked **Indirect Object >Direct Object** structure.

Figure 5: CA representation of topicalization of the indirect object.

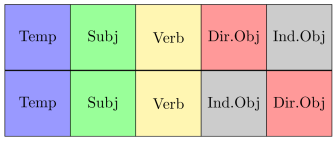


Figure 6: CA progression illustrating the shift from Direct Object > Indirect Object to Indirect Object > Direct Object.

4.1.1 Deviations from traditional CA models

While my LP simulation is inspired by multi-state cellular automata, it departs from standard CA implementations in two key ways: (i) asynchronous updates and (ii) right-to-left updates. These changes are motivated by specific properties of linguistic structure, especially as conceived in minimalist syntax.

Traditional CA are computational models where a grid of cells evolves over discrete time steps. Each cell's state is updated *synchronously* at every step, meaning that *all* cells in a row are updated *at the same time*.

Language, however, does not operate like a standard cellular automaton. In syntactic theory, operations like Merge and Move apply locally and in a specific order, often with later operations dependent on earlier ones. The timing of these operations can feed, bleed, or block one another, properties that are difficult to capture in a system where all elements update simultaneously. Synchronous updating, in other words, fails to reflect the assumed bottom-up, derivational nature of syntactic computation.

To reflect this property, my simulation replaces synchronous updates with an *asynchronous* mechanism: in my model, only one interaction is evaluated at a time. Specifically, at each update step, a single pair of adjacent cells (representing two syntactic elements) is examined, and if the local rule applies, their configuration is altered. Each horizontal row in the output corresponds to the result of exactly one such local update. This setup is meant to mirror the incremental

nature of syntactic derivations, in which each operation applies to a small part of the structure before the next operation begins.

A second deviation concerns update direction. As already mentioned, in traditional CA, all cells are updated simultaneously, based on the states of their immediate neighbors. Because every update happens at the same time, there is no need to specify a direction: the update rule is applied uniformly across the grid, without reference to left-to-right or right-to-left order. In such systems, directionality is simply irrelevant. In linguistic theory, however, derivations are assumed to proceed bottom-up, beginning with internal constituents (e.g., VP-internal arguments) and proceeding toward higher structural domains (e.g., CP and left-peripheral material) (Chomsky 1995). The simulation captures this by evaluating adjacent pairs from *right to left*, mirroring how lower-level dependencies are typically resolved before higher projections are assembled.

4.2 *Reference Theories*

To evaluate whether a local-update, position-free model can approximate attested left-peripheral word order configuration, we need a point of empirical comparison. Over the years, the cartographic tradition has come up with a number of proposed Left-Peripheral hierarchies. In this paper, I focus on two: the hierarchy developed by Rizzi and Bocci (2017), and the topic-rich hierarchy proposed by Frascarelli (2012). The Rizzi & Bocci hierarchy is repeated below in (7); Frascarelli's model is presented in (8):

(7) Force > Top* > Int > Top* > Foc > Top* > Mod > Top* > QEmb > Fin

(8) Force > Shift > Contr > Int > Foc > Fam* > Fin

These two models were chosen for their contrastive theoretical commitments. Rizzi & Bocci's hierarchy represents one of the most recent updates of the original cartographic template proposed by Rizzi (1997), incorporating newer projections such as QEmb. Rizzi & Bocci's hierarchy offers a relatively coarse-grained treatment of topicality; Frascarelli's model, by contrast, provides a more articulated representation of topic types (e.g., aboutness, contrastive, familiar), reflecting

a body of research that critiques the underspecification of "Topic" in Rizzi's traditional hierarchy (see Belletti (2004); Frascarelli and Hinterhölzl (2007); Bianchi and Frascarelli (2010); Samek-Lodovici (2006b)).

In Frascarelli's system, *Shift* marks aboutness topics, which introduce a new discourse referent or shift the current discourse topic; *Contr* marks contrastive topics, which set up an explicit opposition with alternatives; and *Fam* marks familiar topics, which refer back to already given discourse material (see also Frascarelli and Hinterhölzl (2007)).

For the purposes of the simulation, each reference hierarchy was converted into a set of strictly local precedence rules. This process took the global order proposed in the hierarchy and decomposed it into minimal pairwise permissions: statements of the form "if category X is immediately to the right of category Y, X will move leftward over Y." For example, in Frascarelli's hierarchy, *Shift* precedes *Foc* in the global order. This yields a local rule: "Shift will move leftward over Focus if Shift is to the immediate right of Foc." Repeating this step for all possible pairs in the hierarchy produces a complete list of local transitions. These rules contain no reference to absolute positions and do not encode the hierarchy as a whole. Instead, they represent the smallest set of local constraints that, when applied repeatedly, could generate the same global orderings predicted by the original hierarchy.

Because these hierarchies differ in their level of granularity, in the types and number of constituents they include, and in how they constrain the order of those constituents, they offer distinct test cases for evaluating the behavior of a local, interaction-driven model. Importantly, they are used here not as theoretical commitments, but as reference theories, i.e., as frameworks that generate specific predictions about surface word order, against which my simulation's outputs can be compared.

The overarching question is whether a position-free, locally governed model such as the one developed here can replicate the surface word orders predicted by these hierarchies, and if so, what structural assumptions are required for those patterns to emerge from local interactions alone.

SIMULATION DESIGN 5

Model Interface 5.1

The simulation interface consists of the following components:

- *Reference Model (dropdown menu):* this allows the user to select between two different inventories of constituent types, corresponding to the hierarchies proposed by Rizzi & Bocci (2017) and Frascarelli (2012).
- *Setup Options (three buttons):* these are used to initialize the simulation with different starting configurations. The options include (i) a randomly shuffled sequence of constituents, (ii) a manually editable sequence, and (iii) a test setup that always initializes the simulation with the same sequence of elements.
- *Go Options (two buttons):* these control the progression of the simulation. The *Step* button advances the simulation by one update cycle, while the *Go* button runs the simulation continuously until a stable configuration is reached.
- *Update Modes (dropdown menu):* this allows the user to choose between different update methods, currently *Cascade* and *Pairwise*, which define how local interactions are applied across the sequence.

The simulation starts with the user selecting which reference model to use. The user can choose between the Rizzi & Bocci (2017) hierarchy and the Frascarelli (2012) model. Each reference model specifies a distinct inventory of left-peripheral constituents that can appear in the simulation. If the user selects the Rizzi & Bocci (2017) model, the simulation allows for the following constituent types:

- *Force* – the highest functional head, responsible for clause typing.
- *Top* – a topicalized constituent.
- *Int* – an interrogative complementizer.
- *Foc* – a focus constituent.
- *Mod* – a fronted modifier.
- *QEmb* – an embedded interrogative constituent.

- *Fin* – the lowest complementizer head in the Left Periphery, generally said to host non-finite complementizers.

If the user instead selects the Frascarelli (2012) model, the simulation loads a different set of categories:

- *Force* – the highest functional head, responsible for clause typing.
- *Shift* – a shifting topic.
- *Contr* – a contrastive topic.
- *Int* – an interrogative complementizer.
- *Foc* – a focus constituent.
- *Fam* – a familiar topic.
- *Fin* – the lowest complementizer head in the Left Periphery.

In the simulation interface, each constituent type is assigned a distinct cell color. As noted in Section 4, the model is implemented as a multi-state cellular automaton, where each state corresponds to a specific left-peripheral constituent type. In the current setup, every state is linked to a unique color, allowing constituent types to be visually distinguished during the simulation.

After selecting a reference model, the user can initialize the simulation using one of three setup options. The simulation initializes with a single row of cells; the setup options determine how the initial sequence of constituents is populated:

- *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 the string `"Foc Int Mod Top QEmb Force"` into the popup window will initialize the simulation with those elements arranged in the specified order.
- *Test Setup:* generates a predefined configuration for debugging or quick comparisons.

Once an initial sequence has been set, the user can run the simulation using one of two controls: **Go** or **Continuous Go**. Clicking **Go** advances the simulation by a single time step, allowing for fine-grained observation of individual updates. Clicking **Continuous Go** runs the simulation continuously until the system reaches a stable configuration, that is, when no further reordering occurs.

By default, the simulation initializes with a row of six patches, representing a Left Periphery with six constituents. This provides a sufficiently rich space for modeling interactions among multiple elements. However, the number of positions is not fixed: it can be easily modified in the code to include more or fewer constituents without affecting the simulation's functionality or overall behavior.

Rules Governing Patch Interaction 5.2

This section explains how the simulation determines whether two left-peripheral constituents (= two cells in the same row in the simulation) should swap positions, and how it prevents certain pairs from appearing next to each other. These two operations are the sole mechanisms by which ordering is established in the model. Both are strictly local: the simulation never compares more than two neighboring elements at a time and never refers to a fixed global template of positions.

Precedence Rules The primary decision-making mechanism is the *precedence rule* system. At each step in the simulation, the program inspects a pair of adjacent constituents (represented as "patches" in NetLogo); if the right-hand constituent type is licensed to precede the left-hand constituent type according to the local precedence list, the two are swapped. This decision is based solely on the types of the two constituents under comparison. There is no fixed "home" position for any category. For example, the code does not instruct *Foc* to "end up in patch 3" or *Top* to "stay in patch 5". The only information available to the program is a list of **permitted local transitions**: statements of the form "if constituent X is immediately to the right of constituent Y, it may move leftward over it".

In the code, precedence rules are implemented by the `should-swap?` procedure. The listing below shows the actual code that spells out the procedure. Readers unfamiliar with NetLogo can view this as a lookup procedure: the function `should-swap?` checks whether the *reversed* order of the current pair of constituent codes appears in the list of precedence constraints. If it does, the elements are in the wrong order and the function returns "true", triggering a swap. Note that each constituent type is assigned a unique integer code, which is used to

represent it in the model's internal state. For example, in the *Frascarelli* (2012) configuration, 2 corresponds to *Shift* and 5 corresponds to *Focus*.

Listing 1: Code for Swap Check

```
to-report should-swap? [left-state right-state]
  if left-state = right-state [report false]
  if reference-model = "Frascarelli 2012" [
    let pairs [
      [1 2] [1 3] [1 4] [1 5] [1 6] [1 7]
      [2 3] [2 4] [2 5] [2 6] [2 7]
      [3 4] [3 5] [3 6] [3 7]
      [4 5] [4 6] [4 7]
      [5 6] [5 7]
      [6 7]
    ]
    report member? (list right-state left-state) pairs
  ]
  if reference-model = "Rizzi & Bocci 2017" [
    let pairs [
      [1 2] [1 3] [1 4] [1 5] [1 6] [1 7]
      [2 6] [2 7]
      [3 4] [3 5] [3 6] [3 7]
      [4 5] [4 6] [4 7]
      [5 6] [5 7]
      [6 7]
    ]
    report member? (list right-state left-state) pairs
  ]
  report false
end
```

The model proceeds by examining one adjacent pair of patches at a time, specifically the *current* patch (the right member of the pair) and its immediate left neighbor. The procedure `should-swap? [left-state right-state]` determines whether the right element is permitted to move to the left of its neighbor (i.e., whether the two elements should be swapped).

For each reference model, the code features a list `pairs`, in which each entry is an *ordered* pair `[right-state left-state]` representing a configuration in which the right element is allowed to cross over the left element. A swap is permitted if and only if the exact ordered pair `[right-state left-state]` appears in `pairs`; if the pair is absent, the procedure returns `false` and no swap is performed.

As a concrete example, again in the *Frascarelli* configuration, suppose we are evaluating a pair of adjacent patches, and the patch on the right has state `2` (*Shift*) while the patch on the left has state `5` (*Focus*). The procedure `should-swap? [left-state right-state]` checks if the reversed pair `[2 5]` appears in `pairs` . Because `[2 5]` does appears in the list `pairs`, the model determines that the elements are in the wrong order and swaps them. After the swap, the local configuration becomes `2 5`, which now respects the precedence relation. Over successive cycles, these purely local swaps accumulate to produce the final ordering.

Recall that in Rizzi and Bocci's model, *Top* is an undifferentiated topic category, and a Topic position may occur in between almost any other element, with the exceptions of before *Force*, after *Int*, and in between *QEmb* and *Fin*.To model this, *Top* (code = 2) is given only two precedence relations: it must precede *QEmb* (code = 6) and *Fin* (code = 7), encoded as [2 6] and [2 7]. There are no precedence pairs involving *Top* with *Int* (3), *Foc* (4), or *Mod* (5) in either direction (no [2 3], [3 2], [2 4], [4 2], etc). Consequently, for any *Top–Int*, *Foc*, *Mod* adjacency, no swaps are ever licensed, so their relative order remains exactly the starting one throughout the derivation.

Mutual Exclusivity Rules The simulation also enforces a set of *mutual exclusivity* constraints. These prevent certain constituent types from ever appearing next to one another. If such a configuration is detected, the simulation stops and reports a failure. Like the precedence rules, these constraints are purely local: they are triggered only when two disallowed categories are immediately adjacent.This mechanism is designed to block configurations that are structurally impossible, for example, a Left Periphery containing both a *Force* element and a *Fin* element, which cannot co-occur in a single clause (at least under standard cartographic assumptions).

For example, in the Rizzi & Bocci's configuration, the simulation bans the following adjacencies:

- Force and Fin
- Force and Interrogative
- Fin and Interrogative
- QEmb and Interrogative
- Force and QEmb

These bans are not intended as universal claims. They represent one possible encoding of constituent incompatibilities in the Left Periphery. The list of prohibited pairs can always be modified to reflect different language-specific grammars or to explore alternative theoretical assumptions.

5.3 *Update Modes: Cascade vs. Strict Pairwise*

In addition to the content of the local rules themselves, the simulation also allows variation in *how* these rules are applied during the ordering process. This procedural setting is referred to as *update mode*. An update mode determines whether elements that have already been compared and possibly swapped can be later revisited and moved again before the ordering process is complete. Two update modes are implemented: *Cascade* and *Pairwise*.

Cascade Mode In Cascade mode, when a pair of patches is evaluated and swapped, the modified elements remain eligible for further swaps with their new neighbors at later points in the same simulation run. In other words, there are no restrictions on re-accessing elements that have already been evaluated and swapped; earlier material can still be altered if later interactions bring it into a new local configuration. This behavior is loosely inspired by sorting algorithms such as bubble sort, which continue to iterate through a sequence until no further swaps are possible. In linguistic terms, Cascade mode corresponds to a system in which left-peripheral constituents remain accessible for reordering until the derivation reaches a point of stability.

By contrast, in *Pairwise* mode, once a pair of elements has been compared, it is fixed for the remainder of the reordering process. Even

if later swaps create a new local configuration that would normally trigger a swap, that earlier pair is not revisited. Each evaluation is therefore final, and earlier material is effectively "frozen" once processed. This models a more restrictive derivational system, where elements become inaccessible to further reordering once a certain structural step is complete.

The distinction between these two modes loosely parallels discussions about cyclic accessibility in syntactic theory. In particular, *Pairwise* models a highly localist interpretation of phase impenetrability (Chomsky 2001), where previously derived material becomes inaccessible for further reordering once completed. *Cascade*, by contrast, reflects a more globally dynamic system in which left-peripheral constituents remain re-orderable until full stabilization is reached.

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

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

5.4 *The Simulation in Practice*

To illustrate the type of ordering scenarios that this model is designed to capture, consider a case where a speaker must construct a Left Periphery containing six elements: two Familiar Topics, a Focus, a Force element, a Contrastive Topic, and a Shifting Topic. What the speaker needs to figure out is the relative order in which these elements will surface.

In a traditional cartographic analysis, the order is determined by mapping each constituent to a pre-defined functional position within a rigid sructural template. For example, using the Frascarelli hierarchy from (8), repeated here in (9), each constituent is assigned to its designated slot.

(9) Force > Shift > Contr > Int > Foc > Fam* > Fin

Following this hierarchy, the six elements in our example are placed as in (10):

(10) **Force** > **Shift** > **Contr** > Int > **Foc** > **Fam*** > Fin

This produces the following linear order for the set we began with:

(11) Force > Shift > Contr > Foc > Fam > Fam

In this approach, the final outcome is thus determined by "plugging" each constituent into its designated structural slot. Recall our shape-sorting-box analogy from Section 1: potential positions all exist in advance, and the correct "shape" is placed in each one.

The simulation developed here adopts a fundamentally different strategy. It dispenses entirely with a global template of fixed positions, instead treating constituent order as an emergent outcome of local, pairwise interactions. Each element (represented in the model as a "cell") interacts only with its immediate neighbor according to a fixed set of local rules. At no stage is there any reference to a specific constituent type occupying a predetermined position. Crucially, the system only operates over the elements that are actually present in the input. There are no "empty" positions waiting to be filled: the final configuration depends entirely on the specific inventory of constituents provided at the outset.

To determine the order for our example, we initialize the simulation with the relevant cell states (Figure 7). The starting configuration can be any random arrangement of the six elements -though some initial orders will require more swaps than others, but all are valid inputs.[2]

Figure 7: Initial Setup

[2] If the initial configuration already satisfies all precedence constraints, the model will produce the same configuration as output, without further changes.

The simulation then proceeds by evaluating adjacent pairs from right to left, applying the precedence rules where appropriate. In this specific example, the update mode is set to *Cascade*.

- **Initial row:** the first row records the initial configuration, as provided by the user or generated randomly. This row is preserved for reference and remains unchanged throughout the simulation.
- **Second row (First evaluation):** [3] the rightmost pair of cells in the initial row, **Fam** and **Foc**, is evaluated. According to the local rules, **Foc** should precede **Fam**; therefore, the two are swapped. The second row reflects this change, showing the pair as **Foc** > **Fam**.
- **Third row:** evaluation again begins with the rightmost pair, as recorded in the second row (**Foc** and **Fam**), which is already in the correct order. Moving one position left, the next pair (**Shift** and **Foc**) is also correctly ordered, so no swap occurs. The following pair (**Fam** and **Shift**) is then evaluated. Since **Shift** should precede **Fam**, they are swapped. This change is reflected in the third row, which now reads: **Contr** > **Force** > **Shift** > **Fam** > **Foc** > **Fam**.
- **Fourth row:** evaluation begins with the rightmost pair in the third row (**Foc** and **Fam**), which is already in the correct order. Moving one position left, the next pair (**Fam** and **Foc**) is evaluated. Since **Foc** should precede **Fam**, the two are swapped. The fourth row therefore reads: **Contr** > **Force** > **Shift** > **Foc** > **Fam** > **Fam**.

This process continues row by row, with at most one swap per row. Each adjacent pair of constituents is evaluated in right-to-left order, and swapped if necessary, until no further changes occur (or until the stopping condition defined by the update mode is met). Figure 8 presents the complete configuration produced by the simulation. The final row displays the resulting surface order. [4]

[3] For ease of exposition, rows here are numbered from the top of the table downward (Row 1 = initial configuration). In the NetLogo Command Center output, however, rows are technically numbered in reverse, with the initial configuration appearing as the highest-numbered row.

[4] The last row of the cellular automaton duplicates the final row in which a change occurred, making it explicit that the reordering process has concluded.

Figure 8: Final Setup

Because this process relies exclusively on local comparisons, the final configuration is not determined by a pre-existing blueprint. Instead, it emerges step by step from the cumulative effect of local interactions between the specific elements present at the outset: a dynamic alternative to the "fixed slots" architecture of the cartographic approach.

6 EMPIRICAL DEMONSTRATION

This section evaluates the simulation's ability to reproduce the surface orders predicted by our two reference theories (Frascarelli 2012; Rizzi and Bocci 2017) under the two update modes described in Section 4. The aim is to determine whether the correct word order can arise solely from *local* precedence rules and *local* mutual exclusivity constraints, without relying on a predetermined global template of final positions where the different constituent types should end up.

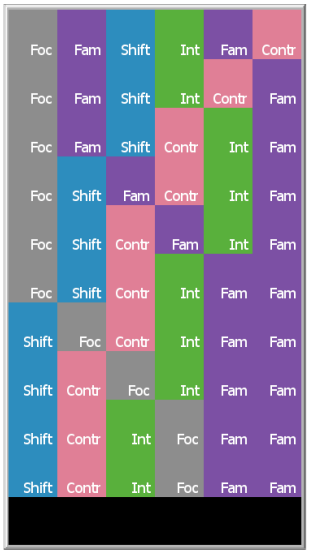
Two input configurations were used as test cases. Each configuration contains only constituent types that, according to the relevant reference theory, are compatible with one another and could theoretically co-occur in the same Left Periphery. While such richly populated sequences are unlikely to occur frequently in spontaneous speech, they are valuable as stress tests: their complexity makes them more likely to reveal differences between the two update modes, and they require the model to resolve multiple precedence relations.

- **Frascarelli 2012:** `[Foc Fam Shift Int Fam Contr]`, where *Foc* = fronted focus, *Fam* = familiar topic, *Shift* = shifting topic, *Int* = polarity complementizer, and *Contr* = contrastive topic.
- **Rizzi & Bocci 2017:** `[Mod Foc Top Int Top Top]`, where *Mod* = fronted modifier, *Foc* = focus, *Int* = interrogative complementizer, and *Top* = a topic constituent without specification for discourse subtype.

Figure 9 presents the results for all four combinations of reference theory and update mode. The upper row shows outputs for the Frascarelli configuration; the lower row shows the Rizzi & Bocci configuration. Within each row, the left panel displays the result of the *Cascade* mode, and the right panel displays the result of the *Pairwise* mode.

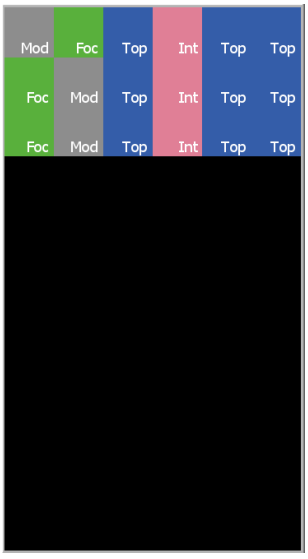
Frascarelli 2012. Under *Cascade* mode (Figure 9a), the model fully matches the order predicted by the reference hierarchy: the shifting topic surfaces first, followed by the contrastive topic, the interrogative, the focus, and finally the two familiar topics. In *Pairwise* mode (Figure 9b), the shifting topic is correctly placed first, but the focus constituent is prematurely fixed before constituents that should appear to the left of the focus according to Frascarelli (2007), i.e. Contr and Int, leading to a globally incorrect order. The two Fam topics also occur in two distinct positions.

Rizzi & Bocci 2017. In both *Cascade* (Figure 9c) and *Pairwise* (Figure 9d) modes, the model fails to produce the expected hierarchy. The interrogative remains blocked behind one of the undifferentiated "Top" blocks, which are licit in multiple positions. These *Top* elements end up functioning as "interactional barriers", preventing cer-



(a) Frascarelli, Cascade

(b) Frascarelli, Pairwise



(c) Rizzi and Bocci, Cascade

(d) Rizzi and Bocci, Pairwise

Figure 9: Simulation outputs for the two reference theories under the two update modes.

tain constituents from ever becoming adjacent to other constituents they would otherwise swap with.

7 DISCUSSION

The simulation results demonstrate both the viability and the boundaries of modeling the LP as the dynamic outcome of an iterative process driven entirely by locally specified rules. Crucially, they show that under the right structural conditions, a purely local mechanism can generate fully grammatical global configurations without relying on a fixed, pre-specified structural template.

Among the configurations tested, only Frascarelli's hierarchy under *Cascade* mode produced a completely grammatical final word order. The other three configurations (Frascarelli under *Pairwise* mode, and both *Cascade* and *Pairwise* runs of the Rizzi and Bocci model) yielded outputs that departed from the target hierarchy. Examining these failure cases more closely reveals why they arise.

Frascarelli–Cascade vs. Frascarelli–Pairwise While the Frascarelli–Cascade configuration produces the correct output, the Frascarelli–Pairwise run does not. Starting from the input `Foc Fam Shift Int Fam Contr`, the Pairwise mode arrives at `Shift Foc Fam Contr Int Fam`. This sequence partially reflects the expected hierarchy: the `Shift` topic is correctly promoted to the left edge, but `Foc` appears too high, surfacing before `Contr` and `Int`, which in Frascarelli's hierarchy should precede fronted foci. In addition, the two `Fam` elements appear in different positions rather than being both at the right edge, where they should occur as the lowest-occurring elements in this particular sequence.

These discrepancies arise from unresolved precedence constraints that the Pairwise mode fails to enforce due to its *no-revisit* update mechanism. Unlike Cascade mode, which allows recursive restructuring across multiple elements in the sequence, Pairwise operates via single-pass adjacent swaps and cannot revise earlier placements once

they have been evaluated. Consider the full simulation output, presented in 12 as a table[5]:

(12)

Row 10:	Foc Fam Shift Int Fam Contr
Row 9:	Foc Fam Shift Int Contr Fam
Row 8:	Foc Fam Shift Contr Int Fam
Row 7:	Foc Fam Shift Contr Int Fam
Row 6:	Foc Shift Fam Contr Int Fam
Row 5:	Shift Foc Fam Contr Int Fam
Row 4:	Shift Foc Fam Contr Int Fam
Row 3:	—

In row 6 of the simulation, we obtain the subsequence `Shift > Fam > Contr`, which obtains after `Shift` has swapped with `Fam` in row 7. At this point, `Fam` is adjacent to `Contr`, and `Contr` should move past `Fam`, but because `Contr` has already been evaluated, it is frozen in place. The result is a sequence that is locally ordered but globally unordered. This illustrates a central limitation of Pairwise updates: without the ability to re-evaluate earlier pairs, the system can become trapped in configurations that satisfy local constraints but fail to achieve the intended global ordering.

Rizzi & Bocci–Pairwise and Cascade The Rizzi & Bocci–Pairwise configuration fails for the same reason as Frascarelli–Pairwise: the lack of feedback (in the complex-systems sense of information from later evaluations feeding back to alter earlier configurations) prevents correction of early misplacements.

The more revealing contrast lies between Rizzi & Bocci–Cascade and Frascarelli–Cascade, since only the latter produces the target order. The difference here stems from the treatment of the *Top* category in the Rizzi and Bocci hierarchy. In this model, *Top* is an undifferentiated category that can appear in multiple positions, and to capture this flexibility, the precedence list omits swap rules for pairs such as `Top` and `Foc`. As a result, subsequences like `Foc > Top > Int` (see Row 10 in example 13) in are treated as locally well-formed: `Foc` is permitted to occur to the left of `Top`, and `Top` is permitted to occur

[5] In the simulation, this output is displayed in the "Command Center" panel at the bottom of the interface.

to the left of `Int`. However, because `Foc` and `Int` never become adjacent, the swap rule that would place `Int` before `Foc` is never triggered. In effect, unranked, undifferentiated *Top* functions as an *interactional barrier*, preventing certain precedence relations from ever being evaluated. This effect can be seen in the simulation output in (13), where we see that `Int` remains "stuck" under `Foc` throughout the simulation:

(13)
```
Row 10:  Mod Foc Top Int Top Top
 Row 9:  Foc Mod Top Int Top Top
 Row 8:  Foc Mod Top Int Top Top
 Row 7:  —
```

The simulation results make clear that a dynamic, locally governed approach to modeling word order can reproduce target hierarchies, but only when the following conditions are met:

1. Swap rules must exist for all relevant element pairs; no elements may block movement or act as inert barriers.
2. Constituents that have been evaluated in earlier steps must remain accessible for subsequent re–evaluation, allowing the system to revise prior configurations when new interactions require it.

When these conditions hold, as in the Frascarelli–Cascade case, local precedence rules suffice to produce a fully grammatical outcome. Pairwise will only produce correct results when elements that require reordering are already adjacent or need very few swaps; in more complex configurations, its lack of feedback prevents convergence on the target order. The Rizzi & Bocci model under Cascade fails for a different reason: the underspecified *Top* category disrupts the connectivity of the interaction network, preventing certain reorderings from taking place.

Finally, the contrast between Cascade and Strict Pairwise modes highlights the importance of feedback and cyclic reorganization in dynamic systems. In the Cascade model, constituents remain accessible and re-orderable throughout the derivation. This flexibility allows the system to recover from locally correct yet globally incorrect configurations and to resolve complex misorderings. In linguistic terms, this aligns with the idea that syntactic derivations may require recursive accessibility.

Overall, these results support the view that rigid cartographic hierarchies are not strictly necessary for modeling complex functional ordering patterns. The simulations demonstrate that well-formed global word order can emerge from the interaction of purely local, iterative processes, without recourse to a fixed, pre-specified structural template, provided that the interaction network is fully connected, swap rules cover all relevant element pairs, and constituents remain accessible for re-evaluation throughout the derivation. When these structural conditions are met, as in the Frascarelli–Cascade case, the system converges on the target order through distributed, stepwise reorganization. From a broader perspective, these findings indicate that it is at least possible, in principle, to model certain aspects of syntactic ordering as emergent properties of a dynamically connected system, without assuming a rigid, globally defined hierarchy.

7.1 *Future Directions*

As this paper represents the very first instantiation of a complexity-inspired, dynamic framework to model word order, the possibilities for extension are considerable. Several immediate avenues stand out.

First, the precedence rules could be adjusted to test language-specific ordering patterns. For example, the interaction between *Foc* and *Int* could be reversed, e.g. to model Bulgarian, where a focused constituent may precede an interrogative complementizer (Callegari 2022). Alternative update modes could also be explored. In addition, different definitions of locality could be tested; for instance, evaluating the next two cells instead of only the immediately adjacent one, or relativizing locality so that different constituent types examine different spans of the sequence.

Second, the results highlight the importance of the category inventory in determining whether the system converges on a grammatical configuration. In particular, in the Rizzi & Bocci model, the undifferentiated *Top* category prevents certain global orderings from emerging, whereas in Frascarelli's model, a fully connected topic inventory allows successful convergence. This invites testing the framework against a broader range of theoretical proposals for the Left Periphery,

including non-cartographic accounts with fewer projections and alternative cartographic groupings that merge or reorganize specific LP categories.

Third, the framework could be extended beyond the Left Periphery to encompass additional regions of the clause. Expanding the model from the VP domain upward would make it possible to capture left-peripheral effects arising from interactions that originate lower in the structure. More targeted applications could also address other richly ordered syntactic domains, such as Cinque's (1999) hierarchy of adverbials.

Fourth, while the present simulation yield the correct result under a specific constituent inventory and update mode, an open question concerns whether repeated local interactions are in fact more cognitively economical than using a fixed, highly articulated template. Further work is needed to test how the number and type of constituents, as well as the update mode and definition of locality, affect processing economy. Presumably, a system with few elements is more economical to model through local relations, as the number of local rules is small; with many elements, the proliferation of local rules might make a fixed template comparatively more efficient.

CONCLUSION 8

This paper has introduced a complexity–inspired framework for modeling left–peripheral word order as the outcome of locally constrained, interaction–driven processes. Implemented as a one–dimensional, multi–state cellular automaton, the model was designed as a computational testbed to determine whether well–formed surface orders can in principle emerge dynamically from the application of local rules, without recourse to a pre–specified global template of dedicated positions. Rather than assigning each constituent type to a fixed slot within an underlying cartographic hierarchy, the simulation attempts to derive ordering incrementally through the cumulative application of local precedence rules and compatibility constraints.

At its core, this work wanted to challenge the assumption that complex syntactic sequences must be hardwired into a universal

blueprint. Instead, it suggested the possibility that language, like complex systems, may exhibit emergent order: patterns that arise not from top-down planning but from the repeated, local interactions of constituent parts. The Left Periphery, with its high constituent density and varied constituent types, served as an ideal test case for this hypothesis.

The results show that such an approach can, under the right structural conditions, replicate the surface configurations predicted by established cartographic hierarchies. In the case of Frascarelli's (2012) finely articulated topic hierarchy, the model generated the target order when run in the *Cascade* update mode, which allows earlier ordering subsequences to be revisited and reordered. By contrast, the *Pairwise* mode, where once–evaluated pairs are frozen, could not repair early misplacements, leading to globally incorrect configurations. With Rizzi and Bocci's (2017) hierarchy, the model failed under both modes. Here, the broad *Top* category, which can appear in multiple positions, acted as an "interactional barrier", preventing crucial constituents from becoming adjacent and triggering the swaps required to reach the target order.

Overall, the findings demonstrate that complex word–order sequences *can* emerge from local interactions within a dynamic system. Successful derivation, however, depends on three key conditions: all elements must actively participate in reordering, interaction pathways must remain open, and the system must be able to revise earlier configurations when new interactions require it. Rather than pointing to a single "correct" theory of the LP (i.e. Frascarelli (2012) vs. Rizzi & Bocci (2017)), these results highlight that both the choice of constituent types and their local interaction patterns are crucial design parameters for making an emergent, interaction–driven framework succeed.

Beyond these specific empirical results, this paper makes several broader contributions:

1. **It introduces a modular, reusable simulation environment to model word order.** The NetLogo implementation is fully extensible: researchers can alter the constituent inventory, change precedence relations, and experiment with alternative update modes. While this paper has focused on the Left Periphery, the same in-

frastructure could be applied to other domains, such as Cinque's (1999) adverbial hierarchy.

2. **It makes word order computationally testable.** Implementing a theory in this framework requires explicitly specifying every precedence relation and mutual–exclusivity constraint it assumes. This forces the underlying assumptions to be made transparent, revealing exactly how many and which ones are needed to produce the intended orders. Because even a single missing constraint can yield incorrect outputs, the framework serves both as a diagnostic tool for identifying theoretical gaps and as a falsifiable, executable test of the theory itself.
3. **It concretely connects word order and complexity science.** This framework links two distinct research fields (syntax and complexity science) in a concrete, empirically testable system. It implements key concepts from complexity science, such as emergence, dynamic processes, and simple local rules, to model word order. While the connection between language and complex systems has a long tradition, beginning with work such as Larsen-Freeman (1997) on language acquisition and chaos theory, the present framework is completely novel in its focus and type, and constitutes the first complexity-inspired model specifically targeting word order in the Left Periphery.

REFERENCES

K. Abels (2012), The italian left periphery: A view from locality, *Linguistic Inquiry*, 43(1):229–254.

Enoch O. Aboh (2004), *The morphosyntax of complement-head sequences: Clause structure and word order patterns in kwa*, Oxford University Press, Oxford.

Ásgrímur Angantýsson (2007), Verb-third in embedded clauses in icelandic, *Studia Linguistica*, 61(3):237–260.

Ásgrímur Angantýsson (2011), *The syntax of embedded clauses in icelandic and related languages*, Hugvísindastofnun Háskóla Íslands.

M. Ballerini, N. Cabibbo, R. Candelier, A. Cavagna, E. Cisbani, I. Giardina, V. Lecomte, A. Orlandi, G. Parisi, A. Procaccini, M. Viale, and V. Zdravkovic (2008), Interaction ruling animal collective behavior depends on topological rather than metric distance: Evidence from a field study, *Proceedings of the National Academy of Sciences*, 105(4):1232–1237, doi:10.1073/pnas.0711437105.

Yaneer Bar-Yam (1997), *Dynamics of complex systems*, Addison-Wesley, Reading, MA, ISBN 0-201-55748-7.

Josef Bayer (2004), Decomposing the left periphery: Dialectal and cross-linguistic evidence, *The syntax and semantics of the left periphery*, 59:95.

Adriana Belletti (2004), Aspects of the low ip area, in Luigi Rizzi, editor, *The Structure of CP and IP: The Cartography of Syntactic Structures, Vol. 2*, pp. 16–51, Oxford University Press, Oxford.

Paola Benincà (2006), A detailed map of the left periphery of medieval romance, in Montserrat Batllori, Maria Lluïsa Hernanz, M. Carme Picallo, and Francesc Roca, editors, *Crosslinguistic Research in Syntax and Semantics: Negation, Tense and Clausal Architecture*, pp. 53–86, John Benjamins, Amsterdam.

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

P. Benincà and C. Poletto (2004a), Topic, focus, and v2: defining the cp sublayers, in L. Rizzi, editor, *The Structure of CP and IP. The Cartography of Syntactic Structures, vol. 2*, pp. 52–75, Oxford University Press.

Paola Benincà and Cecilia Poletto (2004b), Topic, focus and v2: Defining the cp sublayers, in Luigi Rizzi, editor, *The Structure of CP and IP: The Cartography of Syntactic Structures, Vol. 2*, pp. 52–75, Oxford University Press, Oxford.

V. Bianchi and G. Bocci (2012), Should i stay or should i go? optional focus movement in italian, *Empirical issues in syntax and semantics*, 9:1–18.

V. Bianchi, G. Bocci, and S. Cruschina (2015), Focus fronting and its implicatures, pp. 1–19.

V. Bianchi, G. Bocci, and S. Cruschina (2016), Focus fronting, unexpectedness, and evaluative implicatures, *Semantics and Pragmatics*, 9(3), doi:10.3765/sp.9.3.

Valentina Bianchi and Mara Frascarelli (2010), Is topic a root phenomenon?, *Iberia: an international journal of theoretical linguistics*.

G. Bocci (2013), *The syntax prosody interface: A cartographic perspective with evidence from italian*, volume 204, John Benjamins.

Joan BRESNAN (2001), *Lexical-functional syntax*, Blackwell, Oxford.

Elena CALLEGARI (2022), The relative order of foci and polarity complementizers: A slavic perspective, *Linguistic Variation*, 22(1):78–122.

Andrew CARNIE (2007), *Syntax: A generative introduction*, Wiley-Blackwell, Malden, MA, 2nd edition.

Soo-Young CHOI (1986), Solving the problem of the korean topic/subject particles nun and ka: A paradigm and a text analysis.

Noam CHOMSKY (1957), *Syntactic structures*, Mouton, The Hague.

Noam CHOMSKY (1965), *Aspects of the theory of syntax*, MIT Press, Cambridge, MA.

Noam CHOMSKY (1995), *The minimalist program*, MIT Press, Cambridge, MA, ISBN 9780262531283.

Noam CHOMSKY (2001), Derivation by phase, in Michael KENSTOWICZ, editor, *Ken Hale: A Life in Language*, pp. 1–52, MIT Press, Cambridge, MA.

G. CINQUE and L. RIZZI (2008), The cartography of syntactic structures, *Studies in Linguistics*, 2:42–58.

Guglielmo CINQUE (1999), *Adverbs and functional heads: A cross-linguistic perspective*, Oxford University Press, Oxford.

Guglielmo CINQUE (2013), Cognition, universal grammar, and typological generalizations, *Lingua*, 130:50–65, doi:10.1016/j.lingua.2012.10.007.

Guglielmo CINQUE and Luigi RIZZI (2010), The cartography of syntactic structures, in Bernd HEINE and Heiko NARROG, editors, *The Oxford Handbook of Linguistic Analysis*, pp. 51–65, Oxford University Press, doi:10.1093/oxfordhb/9780199544004.013.0003.

Karen De CLERCQ (2017), Prosody as an argument for a layered left periphery, *Nederlandse Taalkunde*, 22(1):31–39.

Iain D. COUZIN, Jens KRAUSE, Nigel R. FRANKS, and Simon A. LEVIN (2005), Effective leadership and decision-making in animal groups on the move, *Nature*, 433(7025):513–516, doi:10.1038/nature03236.

Nomi ERTESCHIK-SHIR (1973), *On the nature of island constraints*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA.

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

Mara FRASCARELLI and Roland HINTERHÖLZL (2007), Types of topics in german and italian, *Natural Language & Linguistic Theory*, 25(4):697–740.

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

Liliane HAEGEMAN (2004), Topicalization, clld and the left periphery, in Adriana BELLETTI, editor, *Structures and Beyond: The Cartography of Syntactic Structures, Vol. 3*, pp. 163–192, Oxford University Press, Oxford.

Liliane HAEGEMAN (2012), *Adverbial clauses, main clause phenomena, and composition of the left periphery*, Oxford University Press, Oxford.

John H. HOLLAND (1992), Complex adaptive systems, *Daedalus*, 121(1):17–30.

Joan B. HOOPER and Sandra A. THOMPSON (1973), On the applicability of root transformations, *Linguistic Inquiry*, 4(4):465–497.

Ray JACKENDOFF (1977), *X-bar syntax: A study of phrase structure*, MIT Press, Cambridge, MA.

Kyu-hyun KIM (2021), Korean 'topic'particle nun as a categorization resource for organizing retro-sequence: Redressing the situated action 'on the periphery', *Journal of Pragmatics*, 183:225–241.

S-Y KURODA (2005), Focusing on the matter of topic: A study of wa and ga in japanese, *Journal of east asian linguistics*, 14(1):1–58.

James LADYMAN and Karoline WIESNER (2020), *What is a complex system?*, Yale University Press, New Haven, CT, ISBN 978-0-300-25110-4.

Knud LAMBRECHT (1994), *Information structure and sentence form: Topic, focus, and the mental representations of discourse referents*, Cambridge University Press, Cambridge.

Diane LARSEN-FREEMAN (1997), Chaos/complexity science and second language acquisition, *Applied Linguistics*, 18(2):141–165, doi:10.1093/applin/18.2.141.

John H. MILLER and Scott E. PAGE (2007), *Complex adaptive systems: An introduction to computational models of social life*, Princeton University Press, Princeton, NJ, ISBN 978-0691127023.

Melanie MITCHELL (2009), *Complexity: A guided tour*, Oxford University Press, New York, ISBN 0195124413.

Vincenzo MOSCATI and Luigi RIZZI (2021), The layered syntactic structure of the complementizer system: functional heads and multiple movements in the early left-periphery. a corpus study on italian, *Frontiers in Psychology*, 12:627841.

Nicola MUNARO, Cecilia POLETTO, and Jean-Yves POLLOCK (2001), Eppur si muove! on comparing french and bellunese wh-movement, *Generative Grammar in Geneva*, 2:153–193.

Kimiko NAKANISHI (2001), Prosody and information structure in japanese: A case study of topic marker wa, *Japanese/Korean Linguistics*, 10:434–447.

Mark E. J. NEWMAN (2011), Complex systems: A survey, *arXiv preprint arXiv:1112.1440*.

Frederick J. NEWMEYER (2004), On split-CPs, uninterpretable features, and the 'perfectness' of language, *ZAS Papers in Linguistics*, 35(2):399–421.

Roberto PETROSINO (2017), The left periphery fragmented: evidence from italian, in *Proceedings of the 48th Annual Meeting of the North East Linguistic Society (NELS 48)*, university of Connecticut.

Cecilia POLETTO (2000), *The higher functional field: Evidence from northern italian dialects*, Oxford University Press, Oxford.

Cecilia POLETTO (2002), The left periphery of v2-rhaetoromance dialects: A new view on v2 and v3, in Sjef BARBIERS, Leonie CORNIPS, and Susanne VAN DER KLEIJ, editors, *Syntactic Microvariation*, pp. 214–242, Meertens Institute, Amsterdam.

Craig W. REYNOLDS (1987), Flocks, herds and schools: A distributed behavioral model, *ACM SIGGRAPH Computer Graphics*, 21(4):25–34, doi:10.1145/37402.37406.

L. RIZZI (1997), The fine structure of the left periphery, in L. HAEGEMAN, editor, *Elements of grammar: Handbook in generative syntax*, pp. 281–337, Kluwer.

L. RIZZI (2001), On the position "int(errogative)" in the left periphery of the clause, in G. CINQUE and G. SALVI, editors, *Current studies in Italian syntax: Essays offered to Lorenzo Renzi*, pp. 287–296, Elsevier.

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

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

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

L. RIZZI and G. BOCCI (2017), The left periphery of the clause, in M. EVERAERT and H. VAN RIEMSDIJK, editors, *The Companion to Syntax, 2nd edition*, Blackwell.

Luigi RIZZI (1990), *Relativized minimality*, MIT Press, Cambridge, MA.

Luigi RIZZI (2013), Notes on cartography and further explanation, *Probus*, 25(1):197–226, doi:10.1515/probus-2013-0010.

Luigi RIZZI and Ur SHLONSKY (2007), Strategies of subject extraction, in Norbert CORVER and Henk VAN RIEMSDIJK, editors, *Interfaces + Recursion = Language? Chomsky's Minimalism and the View from Syntax-Semantics*, pp. 115–160, Mouton de Gruyter, Berlin.

John Robert Ross (1967), *Constraints on variables in syntax*, Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA, published as part of the Indiana University Linguistics Club, 1968.

Vieri Samek-Lodovici (2006a), When right dislocation meets the left-periphery: A unified analysis of italian non-final focus, *Lingua*, 116(6):836–873.

Vieri Samek-Lodovici (2006b), When right dislocation meets the left-periphery.: A unified analysis of italian non-final focus, *Lingua*, 116(6):836–873.

Hiroki Sayama (2015), *Introduction to the modeling and analysis of complex systems*, Open SUNY Textbooks.

Ur Shlonsky and Giuliano Bocci (2019), Syntactic cartography, in Mark Aronoff, editor, *Oxford Research Encyclopedia of Linguistics*, Oxford University Press, doi:10.1093/acrefore/9780199384655.013.310, `https://oxfordre.com/linguistics/display/10.1093/acrefore/9780199384655.013.310`, oxford Research Encyclopedia entry.

Herbert A. Simon (1962), The architecture of complexity, *Proceedings of the American Philosophical Society*, 106(6):467–482.

Jon Sprouse, Ivano Caponigro, Ciro Greco, and Carlo Cecchetto (2016), Experimental syntax: Island effects and extraction licensing, *Natural Language & Linguistic Theory*, 34(1):307–344, doi:10.1007/s11049-015-9286-8.

Michal Starke (2004), On the inexistence of specifiers and the nature of heads, in Adriana Belletti, editor, *Structures and Beyond: The Cartography of Syntactic Structures, vol. 3*, pp. 252–268, Oxford University Press, Oxford.

Mark Steedman (2000), *The syntactic process*, MIT Press, Cambridge, MA.

David J. T. Sumpter (2006), The principles of collective animal behaviour, *Philosophical Transactions of the Royal Society B: Biological Sciences*, 361(1465):5–22, doi:10.1098/rstb.2005.1733.

John Von Neumann (1966), *Theory of self-reproducing automata*, University of Illinois Press, Urbana, IL.

John Von Neumann, Arthur W Burks, *et al.* (1966), Theory of self-reproducing automata, *IEEE Transactions on Neural Networks*, 5(1):3–14.

Mitchell M. Waldrop (1993), *Complexity: The emerging science at the edge of order and chaos*, Simon and Schuster, New York, ISBN 9780671872342.

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

Uri Wilensky (2024), Netlogo (version 6.4), Center for Connected Learning and Computer-Based Modeling, Northwestern University, available at `https://ccl.northwestern.edu/netlogo/`.

Uri WILENSKY and William RAND (2015), *An introduction to agent-based modeling: Modeling natural, social, and engineered complex systems with netlogo*, MIT Press, Cambridge, MA.

Stephen WOLFRAM (1983), Statistical mechanics of cellular automata, *Reviews of Modern Physics*, 55(3):601–644, doi:10.1103/RevModPhys.55.601.

Elena Callegari (2025) Can Word Order in the Left Periphery Emerge from Local Interactions? A Complexity-Inspired Simulation Framework,