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 complexity-inspired framework for modeling word order in the Left Periphery as the outcome of local interactions rather than the assignment of constituents to fixed structural positions. 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 be computed through iterative local interactions between adjacent constituents. Constituents are modeled as agents whose order is incrementally derived via repeated local updates, without reference to a global positional template, paralleling local, interaction-driven organization observed in systems such as bird flocks, neural networks and cellular automata. Using the hierarchies proposed by Rizzi & Bocci (2017) and Frascarelli (2012) as reference models, I show how local interaction rules can reproduce grammatical word-order patterns without assigning constituents to dedicated positions. A proof of concept further illustrates how the same framework can support context-sensitive ordering through more emergent local interactions.

Keywords: Word Order; Left Periphery; Cellular Automata; Complexity; Complex Systems; Computational Modeling; Cartography.

1 INTRODUCTION

In nature and society, complex patterns often arise without centralized control. Bird flocks, ant colonies, traffic flows and neural networks all exhibit large-scale patterns of behavior that result from the repeated application of simple, local interactions. Rather than following a master plan or a fixed underlying blueprint, these systems develop structure through the cumulative effect of local processes, so that large-scale organization is not specified in advance but arises from decentralized interactions.

This paper asks whether a similar perspective might be informative for 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 observed word-order patterns can be computed through decentralized, interaction-driven processes rather than by assigning constituents to dedicated structural positions.

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 both cases, the left-peripheral material is highlighted in boldface:

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

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

The LP is the structural region associated with clause typing, operator scope and information structure, linking the propositional content of the clause to its broader syntactic and discourse context (see Lambrecht 1994; Rizzi 1997; Poletto 2000; Benincà 2001; Rizzi 2001; Aboh 2004; Benincà and Poletto 2004; Rizzi 2004a,b; Frascarelli and Hinterhölzl 2007; Cinque and Rizzi 2008; Rizzi 2011; Frascarelli 2012; Haegeman 2012; Rizzi 2013; Rizzi and Bocci 2017).

A striking feature of the LP is the heterogeneity of the constituents it hosts, bringing together elements associated with distinct grammat-

ical processes. For example, consider the following sentence from Italian, a language with particularly rich left-peripheral structures (Rizzi 1997; Poletto 2000; Benincà 2001; Rizzi 2001; Benincà and Poletto 2004; Haegeman 2004; Rizzi 2004a; Frascarelli and Hinterhölzl 2007; Abels 2012; Rizzi and Bocci 2017):

(3) Ieri, a Luigi, perché improvvisamente il libro Maria
yesterday to Luigi why suddenly the book Maria
glielo ha tirato addosso?
3SG.DAT:3SG.M.ACC have.PRS.3SG throw.PTCP 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 often relied on cartographic frameworks (i.a., Rizzi 1997; Poletto 2000; Munaro et al. 2001; Rizzi 2001; Aboh 2004; Belletti 2004; Benincà and Poletto 2004; Haegeman 2004; Rizzi 2004a,b; Benincà 2006; Frascarelli and Hinterhölzl 2007; Rizzi and Shlonsky 2007; Cinque and Rizzi 2008; Rizzi 2011; Frascarelli 2012; Bocci 2013; Rizzi 2013; Rizzi and Bocci 2017). These approaches 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 this hierarchy. The cartographic approach therefore treats the LP as a highly articulated domain structured by a stable sequence of functional projections, taken to be broadly consistent (and hence "universal") across languages.

This paper explores whether word order patterns in the LP can be captured through an alternative, more dynamically-driven approach. Specifically, drawing on insights from complexity science and complex systems modeling, I investigate whether local interactions among LP constituents are sufficient to compute grammatical word-order patterns without assuming an underlying global hierarchy of dedicated positions.

To explore this hypothesis, I present a simulation built using the NetLogo agent-based modeling environment (Wilensky 1999; Wilensky and Rand 2015; Wilensky 2026), 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 rules. Over time, these local operations accumulate into a global ordering. I assess the model's ability to reproduce the surface orders associated with 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 and 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. We see 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. I then extend the framework with a proof of concept based on a context-sensitive ordering alternation in topic placement, demonstrating how relative order can depend on the evolving local configuration rather than on fixed precedence relations alone.

This paper has three primary objectives:

1. To highlight that the Left Periphery exhibits several properties characteristic of 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 word-order sequences can be derived through local updates alone, and identifying the specific structural conditions under which this is possible.

This model should be understood as targeting the linearization component of the grammar: the component responsible for mapping a syntactically generated object, together with the interface-relevant properties involved in its externalization, onto a surface sequence. In its present form, the model is not a replacement for structure-building syntax. Rather, it asks whether some of the ordering work often encoded in fixed structural positions and global hierarchies, as in cartographic approaches, can instead be handled by local, dynamically applied interaction rules.

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 the hallmarks of a complex system (non-linearity, emergence, multiple interacting components), which legitimizes 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. Section 8 then presents a proof of concept illustrating how the same framework can support a more genuinely emergent, context-sensitive model of word-order alternations. Finally, Section 9 summarizes the main findings, broader contributions and the potential of the framework as a reusable, computationally explicit testbed for investigating theories of word order.

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 universally ordered sequence of functional projections, each associated with a distinct grammatical or discourse-related function (i.a., Rizzi 1997, 2001, 2004a,b; Frascarelli and Hinterhölzl 2007; Cinque and Rizzi 2008; Rizzi 2011; Frascarelli 2012; Rizzi 2013; Rizzi and Bocci 2017). 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, I use the symbol ">" to indicate linear precedence. Each of the projections in 4 is associated with a distinct interpretive or grammatical function:

- **Force** is associated with clause typing, distinguishing, for example, declarative, interrogative and relative clauses. Its head can host finite declarative complementizers (e.g., Italian *che*, 'that'), while relative operators may occupy its specifier (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 Italian, topicalized constituents require clitic resumption if accusative, but not necessarily if dative (Benincà and Poletto 2004; Cruschina 2010).
- **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, typically associated with new, contrastive or corrective information (Rizzi 1997; Bocci 2013; Benincà and Poletto 2004). The extent to which focal elements are realized in this position varies across languages and focus types (Bianchi and Bocci 2012; Cruschina 2012; Bianchi et al. 2015, 2016).
- **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. Its head can host non-finite declarative complementizers, such as Italian *di* (Rizzi 1997; Rizzi and Bocci 2017).

Cartographic approaches thus treat the Left Periphery as a highly articulated domain, characterized by a fixed ordering of functional projections that is assumed to be broadly consistent across languages.

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?

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

Other, later works assume that the entire left-peripheral hierarchy is universally present in the syntactic structure, albeit sometimes with silent heads. For instance, Petrosino (2018, 221) notes that "it is usually assumed that the full cartographic structure is always fully projected", even when no overt material occupies each position. 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.

A related question is whether the hierarchy should be regarded as an innate component of Universal Grammar, or as the outcome of more general constraints. On this issue, the cartographic program has generally adopted an agnostic stance. As Cinque and Rizzi (2008) (page 45) observe:

> "Although some cartographic works occasionally emphasize that certain structural configurations cannot be straightforwardly derived from immediately observable interpretive properties (e.g. Cinque 1990), the question of whether uni-

> versal hierarchies are primitive objects of UG or can be derived from interface or more general external conditions is important but fundamentally orthogonal to the cartographic enterprise, and perhaps not easily determinable at the present state of our knowledge."

While the source of the hierarchy may therefore remain open, cartographic work nevertheless treats the ordering relations it identifies as substantive structural properties of grammar. Rizzi's own discussion emphasizes this point. In particular, Rizzi (2013) (see also Rizzi and Bocci 2017) compares the cartographic hierarchy to the sequence of nucleotides in DNA. Just as the DNA sequence constitutes a real and functionally significant property of biological systems (even if aspects of it may ultimately be explained by deeper biological principles), so too the relative ordering of syntactic projections is taken to reflect a genuine structural property of the human language faculty. As Rizzi and Bocci (2017) note, for instance, even if deeper principles ultimately account for aspects of the functional sequence, this "does not make the observed sequence an epiphenomenon"; rather, the sequence remains a real "object of the world" (p. 22).

Taken together, these works indicate that the functional hierarchy uncovered by cartographic research is treated as a genuine structural property of clause architecture. Whether the hierarchy is assumed to be fully projected in every clause or only partially instantiated, and whether it is taken to be primitive or derived from more general constraints, the sequence itself is regarded as a real component of the grammatical system rather than an epiphenomenal pattern.

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 hierarchy that guides sentence production and constrains constituent order is a strong theoretical commitment[1], particularly given the level of granularity that many of these hierarchies assume, e.g. the hierarchy proposed by Rizzi

[1] See also Newmeyer (2004), who criticizes what he calls "oceans of functional projections" (Newmeyer 2004, 412) in cartographic analyses, questioning the assumption that Universal Grammar specifies a universal inventory and ordering of such projections.

and Bocci (2017) contains ten functional positions. While this commitment may ultimately be justified (perhaps language users do rely on richly specified, globally ordered hierarchies to form grammatical sentences), it is not the only conceivable explanation for the observed patterns [2].

The assumptions of the cartographic model can be illustrated with a deliberately simple analogy[3] to a shape-sorting toy (see Fig. 1).

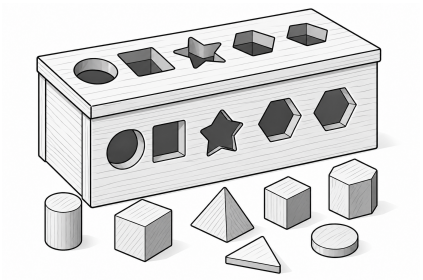


Figure 1: Shape-sorting toy used as an analogy for fixed structural positions.

In a shape-sorting toy, each opening is cut into a specific shape, and each block corresponds to one of these openings; in the present analogy, the blocks represent left-peripheral constituents, and the openings correspond to the structural positions made available by the functional hierarchy. 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 do not change. In the strongest interpretations of cartography, the

[2] Indeed, alternative accounts have sought to derive aspects of left-peripheral ordering from independently motivated grammatical mechanisms. See for example Abels (2012), who derives ordering restrictions from constraints on syntactic movement, and Samek-Lodovici (2006, 2015), who offers an alternative account of apparent left-peripheral focus configurations based on the interaction of focalization, right dislocation and prosody.

[3] As with any analogy, the purpose of this analogy is not to capture every aspect of the theory, but to highlight one particular architectural property as clearly as possible: the existence of a fixed hierarchy of structurally differentiated positions.

analogy extends further, with the full set of structural "slots" assumed to be present even when no element occupies them, while in other approaches projections are instantiated only when required by the derivation. In both cases, however, the ordered sequence of positions (like the sequence of openings in the toy) is treated as a real component of the grammatical system, effectively functioning as a structural "blueprint".

In this paper, I explore whether grammatical word-order configurations could arise through a fundamentally different mechanism: one that replaces global underlying "blueprints" (like in the shape-sorting box) with locally governed reordering operations. Rather than fitting each "block" into a predetermined opening, the model explored here asks whether the blocks themselves, through local interactions, could collectively give rise to the observed ordering. 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 complex systems, where global behavior 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 global patterns of behavior do not arise from an underlying blueprint or centralized controller, but from the cumulative effects of small, local interactions (Simon 1962; Holland 1992; Waldrop 1993; Newman

2011; Sayama 2015). These systems exhibit coherent, complex 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 formations, creating intricate shapes that seem almost choreographed. Watching these fluid, shifting patterns unfold across the sky, one might assume that a single bird is leading the flock and coordinating its movements, 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; Couzin et al. 2005; Ballerini et al. 2008). Through the cumulative effects of these small-scale interactions, coordinated patterns of movement emerge at the level of the flock (Sumpter 2006). This kind of emergent behavior is a hallmark of complex systems and provides a powerful lens through which to reconsider syntactic domains like the Left Periphery.

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

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 or properties 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 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 con-

texts (Erteschik-Shir 1973; Hooper and Thompson 1973; Sprouse et al. 2016). These patterns highlight that the Left Periphery is not simply a collection of autonomous constituents, but a domain whose properties emerge from the non-linear interplay of its components.

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 an inherent lexical property: 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* (Nakanishi 2001; Kuroda 2005) or Korean *nun* (Choi 1986; Kim 2021)), these markers do not reflect a built-in lexical property but instead mark a constituent's role within the discourse. 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 sentence says something (Frascarelli and Hinterhölzl 2007). Identifying a constituent as an aboutness topic is only possible once the rest of the sentence 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.

Multiple interacting components One of the LP's 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 (Bocci 2013; Bianchi and Bocci 2012; Bianchi et al. 2015, 2016)), different types of complementizers (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 (Rizzi 2004a)), fronted verbs in V2 languages (Poletto 2002), fronted negative expressions (Haegeman 2012), and relative operators (Rizzi 1997). This arguably makes the LP one of the most diverse and compositionally rich regions of the clause, bringing together elements associated with clause typing, discourse organization, operator scope and modification. The interaction among these heterogeneous constituent types generates a large combinatorial space of possible configurations: even with a small number of tokens, the interaction of multiple element types and ordering constraints gives rise to a large space of possible linear arrangements.

If the LP exhibits the core properties of a complex system[4], then it is worth exploring whether it can be modeled as one. We might find that such an approach allows us to capture additional generalizations about how left-peripheral word order patterns arise, 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 emergent, relational 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.

[4] The properties discussed here -multiple interacting components, nonlinearity and emergence- are not specific to the Left Periphery, but plausibly characterize the language system more broadly. At that level, the number of interacting components is considerably larger, as different domains of the clause interact. The Left Periphery itself does not operate in isolation, but is systematically related to the structure of the lower clause, from which many of its elements are moved. The focus on the LP in the present study is therefore not meant to suggest that complexity is localized to this domain, but reflects a methodological choice: the LP provides a relatively contained setting in which a rich set of interacting elements can be studied, making it a suitable starting point for modeling how ordering patterns may arise from local interactions.

4 THE FRAMEWORK

This section introduces a framework for modeling word order in the Left Periphery using notions and tools from complex-systems theory. In this framework, the final word order is not obtained by assigning constituents to positions in a predefined global template, but is computed through the repeated application of local interactions among constituents. Rather than assigning dedicated structural positions to elements such as topics, foci and complementizers, the framework determines their relative ordering through local interaction rules applied over an evolving configuration.

I operationalize the framework in NetLogo (Wilensky 1999; Wilensky and Rand 2015; Wilensky 2026), 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 framework is implemented in two related models: a baseline implementation, and a proof of concept focusing on topicalization. Figure 2 provides a first look at the baseline simulation.

The model treats word order in the Left Periphery as the result of an incremental reordering process. The top row displays the initial sequence of syntactic constituents, while each subsequent row shows the result of a single update. The specific screenshot in Figure 2 illustrates a configuration generated with the Frascarelli (2012) reference model; the corresponding labels, including different topic types, are introduced in Section 5. 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. The system updates at each step until either a stable configuration is reached or the simulation halts because a termination condition is triggered.

The framework proposed here targets the computation of linear order, not the full construction of syntactic structure. It therefore assumes a division of labor between structure-building mechanisms and the local ordering procedure implemented in the simula-

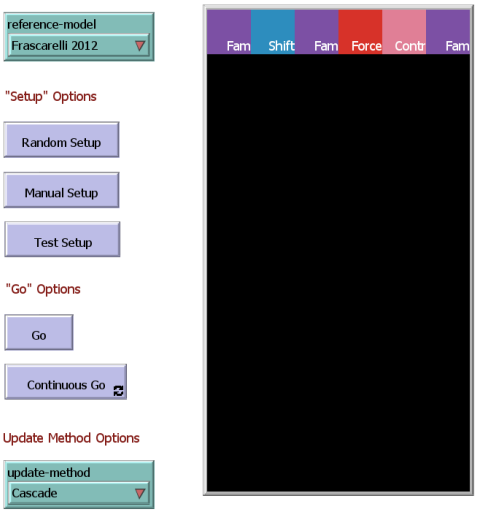


Figure 2: Screenshot of the baseline simulation interface.

tion. Structure-building syntax, broadly construed, is responsible for determining which constituents are present in the relevant derivation, how they are syntactically related, and which interpretive or discourse-structural properties they bear. The framework then models a more restricted but theoretically important question: how these elements are externalized as an ordered sequence. This means that the framework does not attempt to derive constituency, c-command, selection, agreement, case, binding, reconstruction, or the interpretive identification of an XP as a topic or focus. Nor does it model the movement operations that make constituents available in the Left Periphery. These aspects are treated as part of the input to the ordering component. What the model targets is the subsequent ordering problem: given a set of elements with specified syntactic and discourse properties, can their relative surface order be computed through local interactions rather than by assigning each element to an absolute position in a fixed cartographic template?

The NetLogo code for both implementations has been made available in a public repository at: `https://github.com/elecalle/left_periphery`. The repository includes instructions for running the simulations locally or via the NetLogo Web interface after down-

loading the model files.

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 implementations. 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, which is 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 configuration of 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. Over time, repeating these local decision rules across the entire row generates a striking triangular pattern with complex, internal structure, as can be seen in Fig. 3:

What is remarkable about CA like Rule 30 above is that such complex and seemingly patterned structures emerge from an extremely

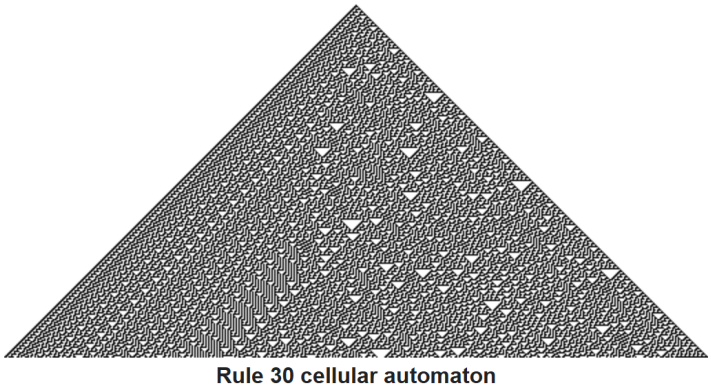


Figure 3: Rule 30

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.

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 simple, 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 1966), CA have since been applied to investigate complex systems in a variety of domains, including physics, ecology and social science.

The framework developed here draws on the architecture of one-dimensional, multi-state cellular automata. 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 illustrates how this sentence is represented as a CA. 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.



Figure 5: 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 6 depicts a progression from the basic **Dir. Obj. > Ind. Obj.** structure to the marked **Ind. Obj. > Dir. Obj.** structure.

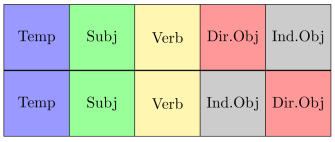


Figure 6: CA progression illustrating the shift from Direct Object > Indirect Object to Indirect Object > Direct Object.

4.2 *Deviations from traditional CA models*

While the framework developed here 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. The application of one syntactic operation may create or eliminate the structural conditions required for another. Depending on their order, these operations can feed, bleed or block one another, properties that are difficult to capture in a system where all elements update simultaneously.

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[5].

5 DESIGN OF THE BASELINE IMPLEMENTATION

The first of the two implementations introduced in the previous section is the *baseline implementation*, described in this Section and tested in Sections 6 and 7. The purpose of this implementation is to make the architecture of the approach explicit: how syntactic elements are represented, how local interactions are defined, how update modes work, and how a global surface order is produced through stepwise local reordering.

5.1 *Reference Hierarchies*

To evaluate whether a local-update, position-free model can reproduce the surface word orders predicted by existing analyses of the Left Periphery, a set of reference predictions is required. Over the years, the cartographic tradition has proposed a number of left-peripheral hierarchies. In this paper, I focus on two: the hierarchy developed by Rizzi and Bocci (2017) and the one 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

[5] The present implementation thus uses asynchronous, right-to-left updating because this most closely approximates the incremental and bottom-up character of syntactic derivations. Other update schedules are technically possible but would produce different outcomes. A synchronous implementation would allow several local configurations to change in parallel, but would obscure dependencies between successive operations and could generate conflicts when overlapping pairs require incompatible updates. Direction-neutral or randomized asynchronous updating would avoid imposing a fixed scan direction, but would make the outcome dependent on the order in which interactions are selected.

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

These two hierarchies 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 different topic types, reflecting a body of research that critiques the underspecification of "Topic" in Rizzi's traditional hierarchy (see Benincà and Poletto 2004; Frascarelli and Hinterhölzl 2007; Bianchi and Frascarelli 2010).

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

Because these hierarchies differ in their level of granularity, in the types and number of constituents they include, and in the relative ordering 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.

5.2 *Model Interface*

In the baseline implementation, the simulation starts with the user selecting from the *Reference Model* dropdown menu which reference model to use. The user can choose between the Rizzi & Bocci (2017) hierarchy and the Frascarelli (2012) hierarchy. Each reference model

specifies a distinct inventory of left-peripheral categories that can appear in the simulation. If the user selects the Rizzi & Bocci (2017) model, the simulation allows for the following constituent types:

- *Force* – elements associated with clause typing, such as finite declarative complementizers.
- *Top* – a topicalized constituent.
- *Int* – an interrogative complementizer.
- *Foc* – a focus constituent.
- *Mod* – a fronted modifier.
- *QEmb* – an embedded interrogative constituent.
- *Fin* – elements associated with finiteness, such as non-finite complementizers.

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

- *Force* – elements associated with clause typing, such as finite declarative complementizers.
- *Shift* – a shifting topic.
- *Contr* – a contrastive topic.
- *Int* – an interrogative complementizer.
- *Foc* – a focus constituent.
- *Fam* – a familiar topic.
- *Fin* – non-finite complementizers.

As discussed in Section 4, the framework draws on the architecture of one-dimensional, multi-state cellular automata. Each left-peripheral category is represented by a distinct state, associated with a specific color. In the NetLogo interface, these states are displayed using *patches*, which correspond to the cells of the CA-inspired representation.

After selecting a reference model, the user can initialize the simulation using one of three *Setup* options. The baseline simulation initializes with a single row of patches; 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.

The different setup options should not all be understood in the same way. *Manual Setup* is closest to a linguistically controlled input: it allows the user to specify a particular set of elements whose ordering behavior can then be inspected. *Random Setup*, by contrast, is not intended as a claim that narrow syntax freely generates arbitrary left-peripheral strings. It is included as an exploratory and diagnostic tool: by exposing the ordering procedure to a wide range of initial configurations, *Random Setup* makes it possible to test whether the local rule system converges on the expected surface order, where it fails, and which assumptions about accessibility or rule coverage are responsible for those failures. The baseline implementation deliberately keeps this input space open in order to evaluate the behavior of the ordering mechanism itself.

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. Clicking *Continuous Go* runs the simulation automatically, advancing through successive update cycles without user intervention. The process continues until the system reaches a stable configuration (i.e., no further reordering operations apply) or until a termination condition is met.

By default, the baseline 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 patches is not fixed: the simulation can be modified to include more or fewer constituents without changing the local interaction rules or update mechanism. Increasing the number of patches increases the number of potential pairwise interactions, and therefore the number of update steps that may be required, but does not alter the logic of the ordering procedure.

Rules Governing Patch Interaction 5.3

This subsection explains how the baseline simulation determines whether two left-peripheral elements (= two patches in the same row in the simulation) should swap positions, and how it prevents certain pairs of elements 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 hierarchy of absolute 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 elements; if the element on the right belongs to a category that is licensed to precede the category of the element on the left, the two are swapped. This decision is based solely on the types of the two patches under comparison. There is no fixed "home" position for any category. For example, the code does not instruct an element of category *Foc* to "end up in patch 3" or *Top* to "stay in patch 5" (which would correspond, in more abstract terms, to approaches in which categories are associated with dedicated structural positions within a fixed hierarchy of projections). The only information available to the program is a list of local transition rules: statements of the form "if constituent X is immediately to the right of constituent Y (and this pair is listed as a permitted transition), X moves leftward over Y."

In the code, precedence rules are implemented by the `should-swap?` procedure. Each left-peripheral category is assigned an integer code, and precedence relations are represented as ordered pairs of these codes. For example, in the Frascarelli (2012) configuration, `2` corresponds to *Shift* and `5` to *Foc*, so the pair `[2 5]` expresses the precedence relation *Shift* > *Foc*. The listing below shows the code implementing this 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.

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 adjacent patches according to the selected update mode. During each comparison, the patch on the right is evaluated against 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).

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?` checks if the reversed pair `[2 5]` appears in `pairs`. Since it does, the procedure returns `true`, triggering

a swap. The resulting configuration is 2 5, which respects the precedence relation. Over successive updates, these local swaps accumulate to produce the final ordering.

Recall that in Rizzi and Bocci's hierarchy, *Top* is an undifferentiated topic category, and topics can appear almost anywhere in the sequence, except above *Force*, below *Fin*, and between *QEmb* and *Fin*. To model this, *Top* (code = 2) is given only three precedence relations: it must follow *Force* (code = 1), and precede *QEmb* (code = 6) and *Fin* (code = 7), encoded as [1 2], [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 categories 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 intended to capture selected restrictions on the co-occurrence of overt left-peripheral elements, such as the incompatibility between the complementizers *che* and *di* within a single clause in Standard Italian[6].

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

- Force and Fin

[6] This restriction is not intended as a universal claim about the distribution of complementizers. While Standard Italian does not permit the overt complementizers *che* and *di* to occur within the same clause, other languages exhibit different patterns. For example, Welsh allows two overt complementizers within a single finite clause, analyzed in cartographic approaches as realizing *Force* and *Fin* (see Rizzi and Bocci 2017, following Roberts 2004). Complementizer doubling is also attested in some Italian dialects (Paoli 2007). Different implementations of the framework could therefore adopt different mutual exclusivity rules to model language-specific patterns (see also the discussion at the end of this subsection).

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

At this point, a natural question concerns the origin and status of the rule systems just introduced: why these particular precedence and incompatibility constraints, and how should they be interpreted theoretically?

One of the aims of this paper is to determine whether ordering patterns predicted by cartographic approaches can be recovered using local interaction rules alone, without reference to a global template of absolute positions. Addressing this question requires an explicit system of local rules. As this paper presents a first implementation of this framework, the rule architecture for the baseline implementation has been kept deliberately simple. Rather than proposing novel or independently motivated rule systems, I adopt a minimal strategy that allows the role of locality to be evaluated as directly as possible. Precedence rules were constructed by translating each reference hierarchy into a set of strictly local transitions. This involves decomposing global ordering statements into pairwise relations of the form "if category X is immediately to the right of category Y, X must move leftward over Y". For example, in Frascarelli's hierarchy, *Shift* precedes *Foc*; this yields the local rule that *Shift* moves leftward over *Foc* when the two are adjacent. Repeating this procedure for all relevant pairs produces a complete set of local precedence rules. To some extent, this reverses a step in the construction of cartographic hierarchies, where observing that one left-peripheral element precedes or follows another helps motivate the postulation of correspondingly ordered structural positions. Translating these hierarchies back into pairwise precedence rules is therefore not entirely removed from the empirical observations that motivated them in the first place. Mutual incompatibility constraints,

by contrast, are not derived through this decomposition procedure but are based on independently motivated assumptions in the literature about which elements cannot co-occur within a single Left Periphery.

These rules should therefore be understood as an operational encoding of theoretical assumptions, not as independently motivated grammatical principles. In particular, the model does not attempt to derive or justify the precedence relations or incompatibility constraints themselves. Rather, it takes them as input and investigates how they behave when implemented as local interaction rules in a dynamical system.

Accordingly, the specific rule sets used for the baseline implementation are not intended as definitive representations of left-peripheral structure. Alternative encodings (based on different theoretical assumptions, different category inventories, or different formulations of locality) can be implemented within the same framework. Exploring which types of local rule systems give rise to empirically adequate ordering patterns constitutes an important direction for future research within this approach.

Update Modes: Cascade vs. Strict Pairwise 5.4

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

In *Cascade* mode, when a pair of patches is evaluated and swapped, the modified elements remain eligible for further swaps with their new neighbors later in the same simulation run. There are therefore no restrictions on revisiting previously processed elements: earlier material can still be reordered if subsequent interactions place it in a new local configuration.

By contrast, in *Pairwise* mode, once a pair of elements has been evaluated, it is fixed for the remainder of the ordering process. Even if later swaps create a new local configuration that would normally

trigger a further swap, that pair is not revisited. Each comparison is therefore final, and previously processed material remains inaccessible to subsequent local updates.

As will be shown and discussed in detail in Section 6, the choice of update mode affects whether the simulation reaches the target order. Comparing these two update modes allows the simulation to explore how different assumptions about constituent accessibility influence the computation of word order. More specifically, the comparison provides insight into:

- how restricting access to previously processed pairs affects local interactions;
- whether a single right-to-left pass is sufficient for the system to converge on grammatical word-order patterns; and
- the computational consequences of different assumptions about constituent accessibility during the derivation.

From a computational perspective, the two update modes implement distinct update strategies. *Cascade* behaves similarly to an iterative sorting procedure such as bubble sort: elements remain accessible throughout the computation, allowing local swaps to propagate across the entire sequence until all precedence violations have been resolved. *Pairwise*, by contrast, resembles a single-pass repair procedure in which each adjacent pair is evaluated only once.

The distinction between these two update strategies also loosely parallels discussions of accessibility in syntactic theory. Under *Pairwise* updating, previously processed material is no longer available for subsequent reordering, a derivational behavior broadly analogous to the restricted accessibility associated with phase-based approaches (Chomsky 2001). *Cascade*, by contrast, assumes that constituents remain accessible to local updates until the computation reaches a stable state.

To summarize, the simulation proceeds according to the following steps:

1. Initialize the sequence of elements.
2. Select an adjacent pair of elements.
3. Evaluate the relevant precedence rule for the pair.

4. Swap the elements if the rule licenses reordering.
5. Repeat the process until either a stable configuration is reached or the simulation halts due to a mutual exclusivity violation.

A configuration is considered *stable* when the simulation reaches a state in which no further updates are possible under the selected update regime. The precise definition of stability therefore differs across the two update modes. Under *Cascade* updating, a configuration is stable when no adjacent pair of elements satisfies the conditions for a swap; that is, when none of the adjacent pairs of elements needs to be swapped according to the precedence rules. Under *Pairwise* updating, stability is reached once every adjacent pair in the sequence has been evaluated exactly once.

5.5 *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 structural hierarchy of positions. For example, consider the Frascarelli hierarchy from (8), repeated here in (9):

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

The six elements in our example are assigned to their corresponding positions within this hierarchy, as indicated in bold 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 2: the relevant positions (all) exist in "advance", and the correct "block" is placed in each one.

The simulation developed here adopts a fundamentally different strategy. It dispenses entirely with a global, underlying hierarchy of fixed positions, instead treating word order as an emergent outcome of local, pairwise interactions. Each constituent (represented in the model as a patch) 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.

To determine the order for our example, we initialize the simulation with the relevant patch states (Figure 7), which can be done using the *Manual Setup* option. The starting configuration can be any arrangement of six elements [7].



Figure 7: Initial Setup

The simulation then proceeds by evaluating adjacent pairs asynchronously from right to left, applying the precedence rules where appropriate. In this specific example, the update mode is set to *Cascade*. Both update modes in the present implementation use this asynchronous right-to-left evaluation procedure; what distinguishes Cascade from Pairwise is whether previously evaluated elements remain available for subsequent re-evaluation.

Figure 8 illustrates the full sequence of local reordering steps for this example. The discussion below walks through the rows of the figure in order, explaining how each successive configuration is derived from the preceding one.

[7] All arrangements are admissible inputs, although this does not imply that every initial arrangement will converge on the target order under every reference hierarchy and update mode. If the initial configuration already satisfies all precedence constraints, the model will produce the same configuration as output, without further changes.



Figure 8: Final Setup

- **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):**[8] the rightmost pair of patches 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

[8] 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.

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 patches 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[9].

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 BATCH TESTING & ILLUSTRATIVE SIMULATION SCENARIOS

This section evaluates the baseline 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. Once again, our aim is to determine whether the correct word order can arise solely from *local* interaction rules, without relying on a predetermined global template specifying the final positions of the different constituent types.

[9] 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.

In what follows, a simulation run is considered successful only if its final output matches the word order predicted by the relevant reference hierarchy; merely reaching a stable configuration without triggering a failure condition does not count as success.

As a first step, I conducted a batch experiment designed to explore the general behavior of the system across a wide range of input configurations. Rather than focusing immediately on individual derivations, the goal at this stage was to identify broader performance patterns associated with the different reference hierarchies and update regimes. To this end, I implemented an automated experiment using the NetLogo *BehaviorSpace* environment, which allows repeated simulation runs.

The experiment was run on four combinations of reference hierarchy and update mode. For each combination, 50 distinct, randomly generated initial sequences were tested, yielding a total of 200 simulation runs[10].

The aggregate results of these runs are presented in Table 2. Here, *success* is defined as the proportion of runs in which the simulation converges to the word order predicted by the reference hierarchy, expressed as a percentage over the 50 runs in each condition:

Reference hierarchy	Update mode	Runs	Success (%)	Avg rows used	Min rows	Max rows
Frascarelli 2012	Cascade	50	100	8.24	2	14
Frascarelli 2012	Pairwise	50	16	7.00	7	7
Rizzi & Bocci 2017	Cascade	50	66	3.84	2	9
Rizzi & Bocci 2017	Pairwise	50	40	7.00	7	7

Table 2: Success rates and row usage across reference hierarchies and update modes.

[10] The experiment varied two parameters: the reference hierarchy (Frascarelli 2012 vs. Rizzi & Bocci 2017) and the update method (Cascade vs. Pairwise), with 50 repetitions per condition. The reported metrics included success, initial and final configurations and the number of update steps. Input configurations were restricted to sets of elements that are mutually compatible, so that outcomes reflect behavior rather than inconsistencies in the input. Full implementation details and instructions for replication are provided in the accompanying code repository. Note that batch experiments can only be executed in the NetLogo Desktop application.

Several observations are worth highlighting. First, across both hierarchies, Cascade updating consistently outperforms Pairwise updating in terms of success rate. The contrast is particularly striking for the Frascarelli hierarchy, where Cascade updating produces the correct surface order in all runs (100%), whereas Pairwise updating succeeds in only 16% of cases. Second, the two hierarchies differ substantially in how often correct configurations emerge even under the more permissive Cascade regime. While the Frascarelli hierarchy reaches the correct order in every run, the Rizzi & Bocci hierarchy does so in only 66% of cases. Finally, the number of rows required to reach a stable configuration varies considerably across conditions. Under Cascade updating, derivations generated from the Frascarelli hierarchy require substantially more update steps on average (8.24 rows, with a maximum of 14) than those generated from the Rizzi & Bocci hierarchy (3.84 rows on average).

While these aggregate statistics reveal broad behavioral differences across hierarchies and update modes, they do not by themselves explain why particular configurations succeed or fail. To better understand the mechanisms responsible for these outcomes, let us examine four individual simulation runs in detail, corresponding to the two reference hierarchies under the two update modes. These examples illustrate how local rule interactions unfold over time and help clarify why the Frascarelli condition performs so reliably under Cascade updating, whereas the Rizzi & Bocci condition yields correct outputs less consistently.

Two input configurations were used as test cases. Each configuration contains only element types that, according to the relevant reference theory, are compatible with one another and could theoretically co-occur in the same Left Periphery.

- **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.

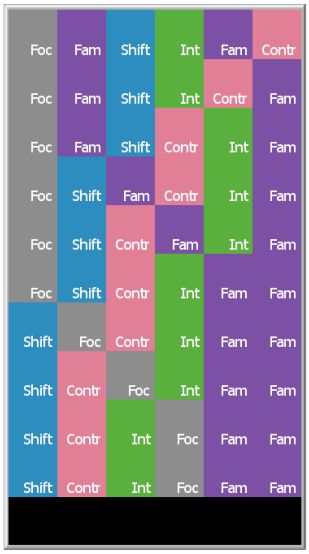
Frascarelli 2012. Under *Cascade* mode (Figure 9a), the model fully matches the order predicted by the reference hierarchy: the shifting topic surfaces first, followed by the contrastive topic, the interrogative, the focus, and finally the two familiar topics. In *Pairwise* mode (Figure 9b), the shifting 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 order. 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 certain constituents from ever becoming adjacent to other constituents they would otherwise swap with.

DISCUSSION 7

The results of the baseline implementation illustrate the potential of modeling word order in the Left Periphery through local interactions. Crucially, they show that under the right structural conditions, a purely local mechanism can generate fully grammatical global configurations without relying on fixed, pre-specified dedicated positions.

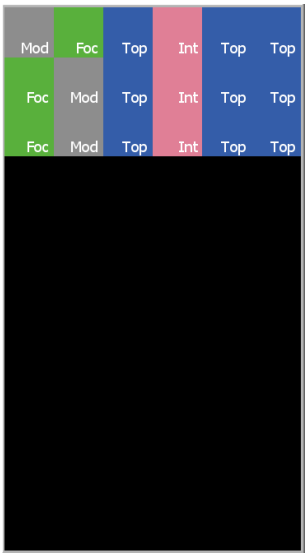
Locality is attractive in this context because it sharply limits the information available at each step: an element responds only to an immediately adjacent element, without access to a representation of



(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.

the complete sequence or to a globally specified target configuration. The model therefore provides a direct way of testing whether repeated interactions with this restricted informational scope can nevertheless produce coherent global ordering patterns. Again, this is not intended as a claim that all syntactic dependencies are local in the same linear sense; relations involving, for example, c-command, binding, scope or probe–goal dependencies would require a richer representational system.

Among the configurations tested, only Frascarelli's hierarchy under *Cascade* mode always produced completely grammatical final word orders. 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 the failure cases in the illustrative simulation runs 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 output partially reflects the expected hierarchy: the `Shift` topic is correctly promoted to the left edge of the sequence, 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 iterative 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[11]:

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

(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 constraints locally 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: its single-pass update mechanism prevents previously evaluated elements from being revisited, even when subsequent interactions create new configurations that would otherwise require further reordering.

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 several positions within the Left Periphery, including both before and after certain other categories. To capture this flexibility, the corresponding local rule system leaves some precedence relations involving *Top* unspecified, including those between `Top` and `Foc`. As a result, subsequences like `Foc > Top > Int` (see Row 10 in example 13) are treated as locally well-formed: `Foc` is permitted to occur to the left of `Top`, and `Top` is permitted to occur 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 *in-*

teractional 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:  —
```

Within the baseline implementation, successful convergence therefore depends on two conditions:

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 restriction on re-evaluating previously processed elements may leave precedence violations unresolved. 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.

Overall, the results of the baseline implementation suggest that it is not necessary to encode left-peripheral ordering patterns in terms of fixed, pre-specified structural positions in order to reproduce them. The simulations show that well-formed global word orders can arise from the interaction of purely local, iterative processes, without assigning elements to dedicated positions in advance, provided that the set of precedence rules covers all relevant element pairs and that constituents remain accessible for re-evaluation throughout the derivation. When these conditions are met, as in the Frascarelli–Cascade case, the system converges on the target order through distributed, stepwise reorganization. More broadly, these findings indicate that it

is possible, at least in principle, to model certain aspects of syntactic ordering as the outcome of dynamically interacting elements, rather than as the realization of a fixed hierarchy of underlying absolute positions.

What is particularly notable is how much the baseline implementation already achieves despite its deliberately restricted architecture. Its limitations should not be interpreted as evidence that local-interaction models are intrinsically unable to capture the remaining ordering patterns, nor as evidence that one of the reference theories is thereby disconfirmed. Rather, the results show that the success of a local implementation depends both on how theoretical categories are translated into interaction rules and on how those rules are applied. The Rizzi and Bocci results, for example, reflect the difficulty of operationalizing an underspecified, recursive *Top* category as a fully connected system of pairwise relations, while the *Pairwise* results expose the limitations of a restrictive update procedure. More articulated rule systems, including rules that encode additional information about constituent relations or allow selected non-local interactions, may therefore extend the empirical coverage of the framework.

8 LIMITATIONS AND A PROOF OF CONCEPT

8.1 *Limitations*

The baseline implementation presented in the preceding sections has a number of limitations that should be made explicit. Some of these reflect the scope of the present framework, while others arise from specific choices made for the baseline implementation.

First, the model abstracts away from the hierarchical component of syntactic computation and operates exclusively on linearized representations. Constituents are treated as atomic symbols arranged in a one-dimensional array, whose ordering is determined through local swap operations. Within this system, locality is defined purely in linear terms: elements may interact only with their immediately adjacent neighbors in the string. This differs from standard syntactic notions of

locality, where relations are defined over hierarchical structure (e.g., in terms of c-command).

This should not be interpreted as a claim that the organization of the Left Periphery can be reduced to linear precedence alone, nor that hierarchical information could not, in principle, be incorporated into this or other complexity-inspired models. The narrower question pursued in the baseline implementation is how much of the observed ordering can be obtained from strictly local precedence relations alone. Accordingly, the goal of the baseline implementation is not to reproduce the full syntactic derivation of the Left Periphery[12], but to investigate whether word-order patterns of the kind described in the cartographic literature can be reproduced through systems governed entirely by local precedence interactions. The broader framework proposed here is intended as a computational model of linear ordering rather than as a replacement for hierarchical syntactic theories. It is worth noting, however, that cartographic hierarchies themselves do not, in isolation, provide a complete account of syntactic derivation either. Their primary function is to provide a structural map of positions within the clause. In cartographic analyses, explanatory force typically arises from the interaction between this structural map and additional mechanisms (such as movement, criterial positions, interface conditions and interpretive constraints) rather than from the hierarchy alone.

A second limitation concerns the rule system adopted in the baseline implementation. As discussed in Section 5.3, the local precedence rules are based directly on the two reference hierarchies used as empirical benchmarks (Rizzi and Bocci 2017; Frascarelli 2012). These hierarchies consolidate relative-order generalizations noticed in the cartographic literature. In the baseline implementation, the consolidated hierarchies were converted into binary precedence relations and implemented as local swap rules. In this sense, the procedure returns the global sequence to a pairwise format resembling the relative-order observations from which cartographic hierarchies are constructed. The resulting rules should nevertheless be understood as operational en-

[12] Doing so would anyway also require modeling the structure of the clause as a whole, together with the movement operations that feed into the Left Periphery.

codings of those ordering generalizations rather than as independently derived linguistic principles.

This choice was deliberate. One of the primary goals of the baseline simulation is to introduce the framework and show in explicit terms how left-peripheral ordering can be modeled as an iterative process of local interactions. To make the architecture and its behavior as transparent as possible, the baseline implementation retains familiar precedence generalizations rather than combining the new modeling framework with a wholly independent theory of the rules themselves. This allows the paper to address a focused initial question: whether, once expressed as strictly local relations, such precedence generalizations can produce the intended global order without assigning constituents to fixed absolute positions.

This choice, however, limits the extent to which the baseline implementation can be characterized as emergent. The baseline system is emergent in a limited sense. It does not encode a fixed mapping between categories and absolute structural positions: the final configuration is not assigned in advance, but arises through the interaction of the elements present in a given derivation. At the same time, category assignments are fixed, so that a given constituent is always subject to the same category-based precedence relations, and the rules governing local swaps are deterministic. The space of possible outcomes is therefore still strongly constrained by these predefined precedence relations. In this respect, the implementation developed in the previous sections does not yet fully exploit the potential of a complexity-inspired framework.

However, one of the broader motivations for adopting a complexity-inspired framework is its potential to model complex interactions and emergent properties. As discussed in Section 3, the Left Periphery exhibits properties that may be better understood in relational terms. Topicality provides a particularly clear example: whether an XP functions as a topic depends on its discourse relation to an associated comment, rather than being an intrinsic lexical property of the XP. A complexity-inspired framework may therefore offer a way of capturing not only the ordering of constituents through local interactions, but also the effects of the broader configuration in which those constituents occur. A natural next step is therefore to investigate whether aspects of ordering can emerge from context-sensitive local interac-

tions rather than being encoded directly in the rule system.

The proof of concept presented in the following subsection serves this purpose. It is not intended as a second comprehensive model of the Left Periphery. Rather, it demonstrates that the same computational framework can accommodate a qualitatively different type of local interaction rule. Specifically, it models the relative ordering of Topic and the interrogative complementizer (*Int*) without stipulating either Topic > Int or Int > Topic as a fixed precedence relation. Instead, the observed ordering emerges from the interaction between a local update rule and the size of the focus domain from which the topicalized constituent must be excluded. The aim is not to provide a complete new implementation, but to illustrate that the framework introduced in this paper is flexible enough to support more genuinely context-sensitive and emergent implementations.

8.2 *TOP-INT proof of concept*

The proof of concept is based on an Italian word-order alternation discussed by Callegari (2018). As discussed in Section 2, cartographic approaches associate specific grammatical or discourse-related functions with dedicated projections in the left-peripheral hierarchy, such that constituents bearing a particular function are expected to occupy the corresponding projection. For example, in Frascarelli (2012), contrastive and shifting topics are assigned to separate projections above Int, which hosts interrogative complementizers and base-generated wh-elements. Because these projections are arranged in a fixed hierarchy, this analysis predicts that both topic types should always precede interrogative material associated with Int. Callegari (2018), however, presents evidence that these topic types can also occur below interrogative material, suggesting that their relative ordering with respect to interrogative elements is more flexible than the hierarchy predicts.

The issue can be illustrated with the Catalan examples in (14), from Callegari (2018, 121). In (14a), the interrogative element precedes the contrastive topic. In (14b), the contrastive topic precedes the interrogative element.

(14) a. Per què el pa l'has venut,
why the bread 3SG.M.ACC:have.PRS.2SG sell.PTCP
i el peix l'has regalat?
and the fish 3SG.M.ACC:have.PRS.2SG give.away.PTCP
'Why did you sell the bread, and give the fish away?'

b. Entenc per què vols estudiar
understand.PRS.1SG why want.PRS.2SG study.INF
francès, però anglès, per què en vols
French but English why it.CL want.PRS.2SG
estudiar?
study.INF
'I understand why you want to study French, but English, why do you want to study it?'

Callegari (2018) shows that the same general pattern is found in French, Italian and Spanish. To account for the flexibility in the relative ordering of topics and interrogative elements, Callegari (2018) develops a focus-driven analysis of topicalization, building on an insight originally proposed by Platzack (1996) and further developed by van Craenenbroeck (2006). The central idea is that a topic cannot remain inside the portion of the clause marked as [+focus]. If a topical constituent originates within that domain, it must move leftward until it reaches a position outside it.

On this analysis, topicalization is not triggered by attraction to a dedicated projection associated with a specific topic type. Instead, it is triggered by an incompatibility between the topical constituent and the focus domain containing it. The eventual position of the topic therefore depends on the left boundary of the relevant focus domain. If the focus domain is large, the topic must move farther left. If the focus domain is smaller, the topic can stop lower in the clause. This means that different placements of the same topicalized XP need not correspond to different topic types; what differs is the focus structure of the sentence (see Callegari 2018 for details).

The word-order alternation modeled in the proof of concept concerns the position of the topicalized XP *il canestro* 'the hoop' with respect to the interrogative element *perché* 'why' in the Italian examples in (15) and (16) (Callegari 2018, 145). In (15), the topic precedes *perché*; in (16), *perché* precedes the topic.

(15) Il canestro, perché non lo regali al
the hoop why not 3SG.M.ACC give.PRS.2SG to.the
vicino?
neighbor
'The hoop, why don't you give it to the neighbor?'

(16) Perché il canestro non lo regali al
why the hoop not 3SG.M.ACC give.PRS.2SG to.the
vicino?
neighbor
'Why, the hoop, don't you give it to the neighbor?'

Callegari (2018) notes that both orders can be preceded by the expression *cambiando completamente discorso* 'changing topic completely', which legitimates a shifting-topic interpretation of the topicalized XP. This is important because in (16) the topic appears after the interrogative element. This is unexpected under an analysis in which shifting topics must occupy positions above Int (Frascarelli 2012)[13].

The analysis in Callegari (2018) treats this alternation as depending on the size of the focus domain from which the topic must be excluded. In (15), the focus domain comprises the whole question apart from the topicalized XP. This domain therefore includes *perché*:

(17) Il canestro, [perché non lo regali al
the hoop why not 3SG.M.ACC give.PRS.2SG to.the
vicino]?
neighbor
'The hoop, why don't you give it to the neighbor?'
Topic: *il canestro* ($= x$)
Focus domain: [*perché non lo regali al vicino*]
Focus value: {why don't you sell x, where did Tom buy x, Mary bought x, …}

Because *perché* belongs to the focus domain, the topic must move past it in order to leave that domain. It consequently surfaces to the left of *perché*.

[13] See again the hierarchy in example 8.

In (16), by contrast, the relevant focus domain is narrower. It contains only the material below *perché*:

(18) Perché il canestro [non lo regali al vicino]?
why the hoop not 3SG.M.ACC give.PRS.2SG to.the neighbor
'Why don't you give the hoop to the neighbor?'
Topic: *il canestro* (= x)
Focus domain: [*non lo regali al vicino*]
Focus value: {why don't you give x to your cousin, why don't you give x to your brother, why doesn't Tom sell x, …}

Here, *perché* lies outside the focus domain. The topic must still leave the focused portion of the clause, but it can do so without crossing the interrogative element. It therefore surfaces below *perché*.

The difference in focus-domain size is also associated with a difference in interpretation. Callegari (2018) characterizes (15), in which the whole question minus the topic is in broad focus, as having a neutral-question reading. Since the focus value is computed over the whole CP, the relevant alternatives are not restricted to clausal complements of *perché*: they may include different types of questions, and even declarative alternatives. In (16), by contrast, the narrower focus domain yields a more restricted focus value. Since *perché* lies outside the focus domain, the alternatives are computed only over the material below it, and must therefore be possible clausal complements of *perché*. The ordering alternation therefore reflects a difference in focus structure rather than necessarily a difference in topic type.

The proof-of-concept simulation implements this contrast. A screenshot of the simulation is provided in Fig. 10. Unlike the baseline simulation, which represents only the left-peripheral field, this proof of concept represents the full clause as a short linear sequence.

For display purposes, the simulation uses the shorter expressions *il pane* 'the bread' and *a Leo* 'to Leo' in place of the longer constituents in Callegari's original examples[14]. The initial state is held constant

[14] Longer labels are difficult to display clearly within the NetLogo patches. The lexical substitution is immaterial to the ordering mechanism.



Figure 10: Proof of Concept Simulation Interface

and corresponds to an assumed base order for the examples, with the direct object preceding the indirect object in Italian, and with *perché* merged directly in the Left Periphery (Rizzi 2001):

perché – non – regali – il pane – a Leo

Each patch corresponds to one word or phrasal constituent. The initial sequence remains constant across runs; what varies is the information-structural configuration assigned to patches in that sequence. The model randomly assigns values to two parameters: which of the two XPs, *il pane* or *a Leo*, is topicalized, and whether *perché* is included in the relevant focus domain.

The interface displays the focus-domain status of each element using different symbols. A plus sign indicates that an element belongs to the active focus domain. A minus sign marks the topicalized XP, which is excluded from that domain and must therefore move to its left. Finally, zero indicates that an element lies outside the active focus-domain calculation.

The update rule is constant across all four scenarios:

> *A topicalized XP moves leftward across an immediately adjacent element if and only if that element belongs to the focus domain that the XP must evacuate.*

The visible symbols provide a simplified representation of this rule. In practical terms, the topicalized XP, marked with `-`, crosses adjacent elements marked `+`. It does not cross an element marked `0`, because that element lies outside the relevant focus domain.

In the simulation, each derivational row records the result of an update step, during which at most one swap is performed. The model scans the current sequence from right to left, checking whether the topicalized XP can swap with the element immediately to its left. A swap takes place if that element belongs to the focus domain from which the topic must be excluded. Once a swap has occurred, the resulting sequence is displayed in the next row, and the scan terminates for that step. The process then repeats, allowing the topicalized XP to move leftward one element at a time, as long as its immediate left neighbor belongs to the relevant focus domain. The derivation stops when a complete scan produces no change. At that point, the final configuration is displayed, and the topicalized XP is labeled TOPIC.

The rule can therefore be understood as a local implementation of focus evacuation. Crucially, the model does not specify a target position for the topicalized XP. The topicalized XP does not move toward a predefined topic position: it moves across elements belonging to the relevant focus domain and stops when it encounters material outside that domain. Since the extent of the focus domain may vary with the information structure of the sentence, the same XP may terminate in different surface positions.

Since the model randomly assigns values to two binary parameters (which of the two XPs, *il pane* or *a Leo*, is topicalized, and whether *perché* is included in the relevant focus domain), and since each parameter has two possible values, their combination yields four possible input configurations. In each configuration, the simulation always converges on a grammatical word order pattern.

If the direct object is topicalized and *perché* is outside the active focus domain, the direct object crosses the TP-internal material but stops below *perché*:

perché – il pane – non – regali – a Leo

If the same XP is topicalized but *perché* is included in the broader focus domain, the direct object also crosses *perché*:

il pane – perché – non – regali – a Leo

The same logic applies when the PP *a Leo* is topicalized. If *perché* is outside the active focus domain, the PP crosses the clause-internal material but stops below the interrogative element:

perché – a Leo – non – regali – il pane

If *perché* is included in the broader focus domain, the PP crosses it as well:

a Leo – perché – non – regali – il pane

Crucially, the implementation contains no fixed precedence rule relating Topic and Int. It does not state either that a topic must precede *perché* or that *perché* must precede a topic. The relative order follows from whether *perché* belongs to the focus domain that the topicalized XP must evacuate.

The proof of concept is intentionally narrow. It does not attempt to provide a complete syntactic or semantic analysis of the Italian examples. For example, it abstracts away from clitic resumption. In Italian left dislocation, resumption is sensitive to the grammatical role of the dislocated constituent: accusative direct-object topics require clitic resumption, whereas dative or prepositional topics may be resumed but do not require resumption in the same way (Cruschina 2012). This asymmetry is not represented in the derivations above, which model only the relative order of the topicalized XP and *perché* [15]. Despite this restricted scope and deliberately simple design, the model captures a generalization that is not straightforwardly expressed by a fixed hierarchy such as that proposed by Frascarelli (2012). In that hierarchy, shifting topics are assigned a position above Int. Accommodating the

[15] A separate realization rule could easily be added to inspect the final configuration and insert the appropriate resumptive clitic when the topicalized XP has the relevant argumental status. Such a rule would, however, concern the morphosyntactic realization of left dislocation rather than the ordering mechanism targeted by the proof of concept, and has therefore not been included in the present implementation.

post-*perché* shifting topic would therefore require weakening the hierarchy, for example by introducing an additional Shift projection below Int. Such an extension would permit both linear orders, but the duplication of Shift positions would not by itself explain why the two orders differ in interpretation. The focus-driven analysis for the proof of concept on the other hand derives both the positional alternation and its interpretive correlate from the same factor: whether *perché* is included in the relevant focus domain[16].

More broadly, the proof of concept clarifies what a more genuinely emergent use of the framework can look like. In the baseline simulations, the final order is computed through local interactions, but the ordering behavior of individual elements remains tied to fixed category assignments and predefined precedence relations derived from the reference hierarchies. In the present implementation, by contrast, the Topic–Int order is not specified in advance in either direction. It arises from the interaction between a constant local rule and context-sensitive information about the extent of the focus domain. The model therefore does more than reconstruct a target sequence through decentralized updates: it derives alternative surface orders from the same mechanism as the information-structural conditions vary. This is significant because it shows how the framework can be used to model emergence not merely as the absence of fixed positions, but as the generation of context-dependent ordering patterns that are not themselves encoded as precedence statements.

The code for running this simulation is available in the same GitHub repository as the code for the baseline implementation[17].

8.3 *Future Directions*

As this paper presents a first implementation of a dynamic, complexity-inspired framework for modeling word order in the Left Periphery, it opens up several directions for further development.

[16] The same implementation strategy could be extended to configurations involving multiple topics and nested focus domains, which are also discussed in Callegari (2018). Different XPs could be specified as excluded from differently sized domains, and their final positions would follow from the boundaries of the specific focus domains they must evacuate.

[17] `https://github.com/elecalle/left_periphery`

A first line of extension concerns the rule system itself. The baseline implementation uses precedence rules defined over constituent types, whereas the proof of concept derives ordering from the focus-domain status of the elements involved. These two implementations illustrate only a small subset of the rule systems that could be explored within the framework. Other local rules could be formulated on the basis of different syntactic, information-structural or morphosyntactic properties. Different rule types could also be combined within the same model. For example, some local interactions might be governed by category-based precedence, while others might depend on whether an element belongs to a particular focus domain. Comparing such rule regimes would make it possible to investigate which kinds of information are sufficient to derive particular ordering patterns and whether different sources of ordering information interact, reinforce one another, or produce conflicting outcomes.

A second direction concerns language-specific variation. The local interaction rules can be modified to represent ordering relations found in different languages. For example, the interaction between *Foc* and *Int* could be reversed to model languages such as Bulgarian, in which a focused constituent may precede an interrogative complementizer (Callegari 2022). Other cases involve variation in the position of topics with respect to declarative complementizers. In Bulgarian and Latin, for instance, topicalized material may precede the declarative complementizer (Bianchi 1999; Rudin 1986). Such patterns could be represented by altering the relevant local relations and examining whether the resulting system converges on the attested orders. More generally, language-specific rule sets could be implemented and compared in order to determine which local configurations reproduce the ordering possibilities of particular languages and which differences require changes elsewhere in the model.

A third direction concerns the update dynamics of the system. The baseline implementation compares two update modes, but these do not exhaust the possible ways in which local rules might be applied. Alternative update procedures could vary the order in which interactions are evaluated, the number of updates permitted within a derivational cycle, or the conditions under which previously evaluated elements remain accessible. Different constituent types could also be assigned different spans of interaction, allowing some relations to be

strictly adjacent while others operate over a wider but still limited domain. This would make it possible to investigate more flexible notions of locality within the same general architecture.

Future versions could also introduce probabilistic rule application. Instead of applying every licensed operation categorically, the model could assign different probabilities to different interactions. This would allow the framework to represent optionality, variable ordering and competition between derivational paths. It would also shift the analysis from the question of whether a particular output can be generated in principle to the question of how frequently or robustly it arises under a given rule system. The distribution of outcomes across large numbers of runs could then be compared with corpus frequencies or experimentally observed preferences.

A fourth line of investigation concerns the category inventory. The contrast between the Rizzi & Bocci and Frascarelli implementations shows that the success of a local-interaction model depends, in part, on how its constituent categories are defined. There are several theoretically motivated ways of classifying left-peripheral constituents, reflecting different assumptions about which grammatical and discourse-related distinctions are relevant. Future research could therefore explore alternative category inventories and investigate how these choices affect the model's ability to derive the attested ordering patterns.

Finally, the framework could be extended beyond the Left Periphery. Other syntactic domains characterized by rich and highly constrained ordering relations provide natural test cases. One such domain is the hierarchy of adverbials proposed by Cinque (1999).

9 CONCLUSION

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 hierarchy of dedicated positions. Rather than assigning each constituent type to a fixed slot within an underlying hierarchy, the simulation attempts to derive ordering incrementally through the cumulative application of strictly local rules.

At its core, this work wanted to challenge the assumption that complex syntactic ordering patterns necessarily require an underlying, highly articulated hierarchy of dedicated structural positions. 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, serves as an ideal test case for this hypothesis.

The baseline simulation shows that the ordering patterns described in cartographic analyses can be derived through local interactions alone, without requiring the model to consult a global hierarchy or assign constituents to fixed absolute positions. This result is conditional, however, on how local interactions are defined and applied within the model. Under the encoding adopted here, the Frascarelli (2012) hierarchy converges on the target order in *Cascade* mode, whereas *Pairwise* updating cannot always repair earlier misplacements, and the present implementation of the Rizzi and Bocci (2017) hierarchy leaves some local relations unresolved because of the underspecified *Top* category. 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. The proof of concept goes a step further by showing that the relative order of Topic and Int need not be encoded directly: both Topic > Int and Int > Topic can be derived from the same local rule as a function of whether *perché* is included in the relevant focus domain. Taken together, the results show that the framework can serve both as an executable test of existing ordering assumptions and as a basis for modeling genuinely context-sensitive ordering alternations.

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 infrastructure 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 assumptions 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, and moving on to more recent work such as Stanisz et al. (2024), 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

Abels, K. (2012). The Italian left periphery: A view from locality. *Linguistic Inquiry 43*(2), 229–254.

Aboh, E. O. (2004). *The Morphosyntax of Complement-Head Sequences: Clause Structure and Word Order Patterns in Kwa*. Oxford: Oxford University Press.

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

Angantýsson, Á. (2011). *The Syntax of Embedded Clauses in Icelandic and Related Languages*. Phd thesis, University of Iceland, Reykjavík. Published by Hugvísindastofnun Háskóla Íslands.

Ballerini, M., 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.

Bar-Yam, Y. (1997). *Dynamics of Complex Systems*. Reading, MA: Addison-Wesley.

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

Benincà, P. (2006). A detailed map of the left periphery of medieval romance. In R. Zanuttini, H. Campos, E. Herburger, and P. H. Portner (Eds.), *Crosslinguistic Research in Syntax and Semantics: Negation, Tense and Clausal Architecture*, pp. 53–86. Washington, DC: Georgetown University Press.

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

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

Bianchi, V. (1999). *Consequences of antisymmetry: Headed relative clauses*, Volume 46. Walter de Gruyter.

Bianchi, V. and G. Bocci (2012). Should i stay or should i go? optional focus movement in italian. In C. Piñón (Ed.), *Empirical Issues in Syntax and Semantics*, Volume 9, pp. 1–18.

Bianchi, V., G. Bocci, and S. Cruschina (2015). Focus fronting and its implicatures. In E. O. Aboh, J. Schaeffer, and P. Sleeman (Eds.), *Romance Languages and Linguistic Theory 2013: Selected Papers from Going Romance Amsterdam 2013*, Volume 8 of *Romance Languages and Linguistic Theory*, pp. 1–20. Amsterdam: John Benjamins.

Bianchi, V., G. Bocci, and S. Cruschina (2016). Focus fronting, unexpectedness, and evaluative implicatures. *Semantics and Pragmatics 9*(3).

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

Bocci, G. (2013). *The Syntax–Prosody Interface: A Cartographic Perspective with Evidence from Italian*, Volume 204 of *Linguistik Aktuell/Linguistics Today*. Amsterdam: John Benjamins.

Bresnan, J. (2001). *Lexical-Functional Syntax*. Blackwell Textbooks in Linguistics. Oxford: Blackwell.

Callegari, E. (2018). *Understanding Word Order in the Left Periphery*. Ph. D. thesis, University of Oslo.

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

Carnie, A. (2007). *Syntax: A Generative Introduction* (2nd ed.). Malden, MA: Blackwell Publishing.

Choi, S.-Y. (1986). Solving the problem of the korean topic/subject particles nun and ka: A paradigm and a text analysis. *Linguistics 24*(2), 351–369.

Chomsky, N. (1957). *Syntactic Structures*. The Hague: Mouton.

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

Chomsky, N. (1995). *The Minimalist Program*. Cambridge, MA: MIT Press.

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

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

Cinque, G. and L. Rizzi (2008). The cartography of syntactic structures. In V. Moscati (Ed.), *STiL – Studies in Linguistics*, Volume 2 of *CISCL Working Papers on Language and Cognition*, pp. 43–59. Siena: CISCL, University of Siena.

Couzin, I. D., J. Krause, N. R. Franks, and S. A. Levin (2005). Effective leadership and decision-making in animal groups on the move. *Nature 433*(7025), 513–516.

Cruschina, S. (2010). Syntactic extraposition and clitic resumption in italian. *Lingua 120*(1), 50–73.

Cruschina, S. (2012). *Discourse-Related Features and Functional Projections*. Oxford Studies in Comparative Syntax. Oxford University Press.

Erteschik-Shir, N. (1973). *On the Nature of Island Constraints*. Ph. D. thesis, Massachusetts Institute of Technology, Cambridge, MA.

Frascarelli, M. (2007). Subjects, topics and the interpretation of referential pro: An interface approach to the linking of (null) pronouns. *Natural Language & Linguistic Theory 25*(4), 691–734.

Frascarelli, M. (2012). The interpretation of discourse categories: Cartography for a crash-proof syntax. In V. Bianchi and C. Chesi (Eds.), *Enjoy Linguistics! Papers Offered to Luigi Rizzi on the Occasion of His 60th Birthday*, pp. 180–191. Siena: CISCL Press.

Frascarelli, M. and R. Hinterhölzl (2007). Types of topics in german and italian. In K. Schwabe and S. Winkler (Eds.), *On Information Structure, Meaning and Form: Generalizations across Languages*, Volume 100 of *Linguistik Aktuell/Linguistics Today*, pp. 87–116. Amsterdam: John Benjamins.

Haegeman, L. (2004). Topicalization, CLLD and the left periphery. *ZAS Papers in Linguistics 35*(1), 157–192.

Haegeman, L. (2012). *Adverbial Clauses, Main Clause Phenomena, and Composition of the Left Periphery*. Oxford: Oxford University Press.

Holland, J. H. (1992). Complex adaptive systems. *Daedalus 121*(1), 17–30.

Hooper, J. B. and S. A. Thompson (1973). On the applicability of root transformations. *Linguistic Inquiry 4*(4), 465–497.

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

Kim, K.-h. (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.

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

Ladyman, J. and K. Wiesner (2020). *What Is a Complex System?* New Haven, CT: Yale University Press.

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

Larsen-Freeman, D. (1997). Chaos/complexity science and second language acquisition. *Applied Linguistics 18*(2), 141–165.

Miller, J. H. and S. E. Page (2007). *Complex Adaptive Systems: An Introduction to Computational Models of Social Life*. Princeton, NJ: Princeton University Press.

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

Moscati, V. and L. 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.

Munaro, N., C. Poletto, and J.-Y. Pollock (2001). Eppur si muove! on comparing French and Bellunese wh-movement. *Linguistic Variation Yearbook 1*(1), 147–180.

Nakanishi, K. (2001). Prosody and information structure in japanese: A case study of topic marker wa. In N. Akatsuka and S. Strauss (Eds.), *Japanese/Korean Linguistics*, Volume 10, pp. 434–447. Stanford, CA: CSLI Publications.

Newman, M. E. J. (2011). Resource letter CS–1: Complex systems. *American Journal of Physics 79*(8), 800–810.

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

Paoli, S. (2007). The fine structure of the left periphery: Comps and subjects: Evidence from romance. *Lingua 117*(6), 1057–1079.

Petrosino, R. (2018). The left periphery fragmented: Evidence from italian. In S. Hucklebridge and M. Nelson (Eds.), *Proceedings of the 48th Annual Meeting of the North East Linguistic Society (NELS 48)*, Volume 2, Amherst, MA, pp. 221–230. GLSA Publications.

Platzack, C. (1996). Germanic verb second languages: Attract vs. repel: On optionality, a-bar movement and the symmetrical/asymmetrical verb second hypothesis. In E. Lang and G. Zifonun (Eds.), *Deutsch – typologisch*, pp. 92–120. Berlin and New York: Walter de Gruyter.

Poletto, C. (2000). *The Higher Functional Field: Evidence from Northern Italian Dialects*. Oxford: Oxford University Press.

Poletto, C. (2002). The left periphery of v2-rhaetoromance dialects: A new view on v2 and v3. In S. Barbiers, L. Cornips, and S. van der Kleij (Eds.), *Syntactic Microvariation*, pp. 214–242. Amsterdam: Meertens Institute.

Reynolds, C. W. (1987). Flocks, herds, and schools: A distributed behavioral model. *Computer Graphics 21*(4), 25–34.

Rizzi, L. (1990). *Relativized Minimality*. Cambridge, MA: MIT Press.

Rizzi, L. (1997). The fine structure of the left periphery. In L. Haegeman (Ed.), *Elements of Grammar: Handbook in Generative Syntax*, pp. 281–337. Dordrecht: Kluwer Academic Publishers.

Rizzi, L. (2001). On the position "Int(errogative)" in the left periphery of the clause. In G. Cinque and G. Salvi (Eds.), *Current Studies in Italian Syntax: Essays Offered to Lorenzo Renzi*, Volume 59 of *North-Holland Linguistic Series: Linguistic Variations*, pp. 287–296. Brill.

Rizzi, L. (2004a). Locality and left periphery. In A. Belletti (Ed.), *Structures and Beyond*, Volume 3 of *The Cartography of Syntactic Structures*, pp. 223–251. Oxford and New York: Oxford University Press.

Rizzi, L. (Ed.) (2004b). *The Structure of CP and IP*, Volume 2 of *The Cartography of Syntactic Structures*. Oxford and New York: Oxford University Press.

Rizzi, L. (2011, May). Some issues in the cartography of the left periphery. Paper presented at GIST 3: Cartographic Structures and Beyond, Ghent University, Ghent, Belgium. 13 May 2011.

Rizzi, L. (2013). Notes on cartography and further explanation. *Probus 25*(1), 197–226.

Rizzi, L. and G. Bocci (2017). Left periphery of the clause: Primarily illustrated for italian. In M. Everaert and H. C. van Riemsdijk (Eds.), *The Wiley Blackwell Companion to Syntax* (2nd ed.). John Wiley & Sons.

Rizzi, L. and U. Shlonsky (2007). Strategies of subject extraction. In U. Sauerland and H.-M. Gärtner (Eds.), *Interfaces + Recursion = Language? Chomsky's Minimalism and the View from Syntax-Semantics*, Volume 89 of *Studies in Generative Grammar*, pp. 115–160. Berlin: Mouton de Gruyter.

Roberts, I. (2004). The c-system in brythonic celtic languages, v2, and the epp. In L. Rizzi (Ed.), *The Structure of CP and IP: The Cartography of Syntactic Structures, Volume 2*, pp. 297–328. Oxford: Oxford University Press.

Ross, J. R. (1967). *Constraints on Variables in Syntax*. Ph. D. thesis, Massachusetts Institute of Technology, Cambridge, MA. Reproduced by the Indiana University Linguistics Club in 1968.

Rudin, C. (1986). *Aspects of Bulgarian Syntax: Complementizers and WH Constructions*. Columbus, OH: Slavica Publishers.

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

Samek-Lodovici, V. (2015). *The Interaction of Focus, Givenness, and Prosody: A Study of Italian Clause Structure*. Oxford University Press.

Sayama, H. (2015). *Introduction to the Modeling and Analysis of Complex Systems*. Open SUNY Textbooks.

Shlonsky, U. and G. Bocci (2019). Syntactic cartography. In M. Aronoff (Ed.), *Oxford Research Encyclopedia of Linguistics*. Oxford University Press.

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

Sprouse, J., I. Caponigro, C. Greco, and C. Cecchetto (2016). Experimental syntax and the variation of island effects in english and italian. *Natural Language & Linguistic Theory 34*(1), 307–344.

Stanisz, T., S. Drożdż, and J. Kwapień (2024). Complex systems approach to natural language. *Physics Reports 1053*, 1–84.

Steedman, M. (2000). *The Syntactic Process*. Cambridge, MA: MIT Press.

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

van Craenenbroeck, J. (2006). Transitivity failures in the left periphery and foot-driven movement operations. *Linguistics in the Netherlands 23*(1), 52–64.

von Neumann, J. (1966). *Theory of Self-Reproducing Automata*. Urbana, IL: University of Illinois Press. Edited and completed by Arthur W. Burks.

Waldrop, M. M. (1993). *Complexity: The Emerging Science at the Edge of Order and Chaos*. New York: Simon and Schuster.

Wilensky, U. (1999). *NetLogo*. Evanston, IL: Center for Connected Learning and Computer-Based Modeling, Northwestern University.

Wilensky, U. (2026). NetLogo (version 7.0.4). Center for Connected Learning and Computer-Based Modeling, Northwestern University.

Wilensky, U. and W. Rand (2015). *An Introduction to Agent-Based Modeling: Modeling Natural, Social, and Engineered Complex Systems with NetLogo*. Cambridge, MA: MIT Press.

Wolfram, S. (1983). Statistical mechanics of cellular automata. *Reviews of Modern Physics 55*(3), 601–644.

Elena Callegari
ⓘ 0009-0000-6839-4470
ecallegari@hi.is

Centre for Digital Humanities and
Arts (CDHA)
University of Iceland
Reykjavík, Iceland

Elena Callegari (2026) Can Word Order in the Left Periphery Emerge from Local Interactions? A Complexity-Inspired Simulation Framework,

This work is licensed under the *Creative Commons Attribution 4.0 Public License*.
ⓒⓘ http://creativecommons.org/licenses/by/4.0/