Lecture 6

Learning Local Dependencies

Strictly local grammars provide a model for local phonological processes that is sufficiently powerful (all local processes can be correctly described), cognitively plausible (low memory load, efficient runtime behavior), and captures some essential properties of natural language phonology (linguistic creativity, generalization from short strings to longer ones, some processes are more complex than others, constraints can apply conjunctively but not disjunctively).

Even in the best case, though, this is insufficient to attain full *explanatory adequacy* in the terminology of Chomsky (1965). Recall the three levels of adequacy:

1. **Observational Adequacy**
   The theory correctly accounts for all the observed data.

2. **Descriptive Adequacy**
   The theory describes a native speaker's knowledge of their language. Hence it gives a full specification of their grammar and thus makes the right predictions even for unobserved data.

3. **Explanatory Adequacy**
   The theory explains how a native speaker acquires their knowledge from the primary linguistic data.

Strictly local grammars are certainly observationally adequate since they can handle every phonological process as long as its application domain is bounded in size. Descriptive adequacy requires the formal model to mirror the speaker's internal grammar; a standard which is rather hard to evaluate. At the very least strictly local grammars can generalize beyond finite data, make typological claims, and offer a rigorous definition for what grammars look like (they are finite sets of n-grams). So they certainly go beyond observational adequacy and might even be descriptively adequate — provided that phonology involves no mappings from underlying forms to surface forms, which is something these grammars do not handle.

Explanatory adequacy is all about the acquisition of the grammar. Given a (possibly infinite) collection of grammars to choose from, how does a speaker arrive at a descriptively adequate grammar from a finite sample of input data? This is an issue we haven't addressed at all yet. As we will see today, strictly local grammars can be learned rather easily given that a crucial piece of information is already specified via our genetic endowment (i.e. Universal Grammar in generative terminology). So strictly local grammars do offer a nativist account of how local phonological processes can be acquired.

Linguists commonly use the term explanatory adequacy in a more general sense nowadays to describe any kind of account that explains a given phenomenon rather than just stipulating a grammar with a bunch of properties that correctly account for the data. Chapter 1 of Chomsky (1965), on the other hand, clearly ties it to matters of learnability.

1 Machine Learning versus Learnability

The question of how grammars can be learned is also of great importance in computational linguistics and NLP. On a purely practical level, it isn't feasible to hand-write grammars for all the languages in the world, not even if we only focus on their phonology. And even if that were possible, hard-coded grammars cannot adapt to local dialects or ongoing changes in the speech community. So it is preferable to have an algorithm that constructs and modifies grammars based on the input it gets. The development, evaluation, and efficient deployment of such algorithms is the purview of *machine learning*.

Unsuprisingly, learning algorithms that work well in real-world applications are very complex beasts that can be hard to understand and analyze. So it is often more insightful to study a more abstract problem: given a possibly infinite space of languages, can we identify properties of this space that guarantee the existence of a learning algorithm provided certain types of input? This question is studied in *learnability*.

Just like the difference between computational linguistics and NLP, the difference between learnability and machine learning is often subtle. The most important distinction, however, is that learnability is not directly about learning/language acquisition but rather about studying the structural properties of language classes and how they facilitate generalization.

In a certain sense, learnability is linguistics at the level of languages rather than the objects these languages describe. In linguistics, we identify structural generalizations for sentences and words — encoded via a grammar — that can be invoked to determine well-formedness and make typological predictions. In learnability, we also identify structural generalizations, but we do not look at the objects within a language but rather at all languages within a given class. The goal is to identify structural properties of the class of languages that are to be learned. More formally, linguistics looks at sets of objects, and learnability looks at sets of such sets. This point of view highlights that learnability is not directly concerned with learning as such, although its result have important ramifications for learning (just like computational linguistics is not about processing language with computers but its results can be applied this way).

Machine learning, on the other hand, is about solving any kind of problem that involves a learning component. This naturally makes it a much messier affair. If you want to teach a computer to filter spam mails, that is a learning problem, but it is not immediately obvious how one could formalize this as a language learning problem or what it even means for the learner to work correctly — although it is very easy to tell for the user when their spam filter screwed up. At any rate the problem certainly isn't approached in terms of, say, a space of spam languages and how its properties can be exploited for learning. That's not to say that such considerations never factor into the initial design of the learning algorithm, but they are not the central object of study.

There are also huge methodological differences between learnability and machine learning. Learnability is all about mathematical theorems and proofs, whereas experiments and simulations are much more common in machine learning. Just like with computational linguistics versus NLP, that is often a matter of necessity: the learning algorithms used in machine learning are so complex, and the problems so difficult to define in formal terms, that proofs simply aren't feasible. And of course machine learning has a stronger focus on solving real-life problems, so pretty much anything goes as long as it gets the job done, even if nobody can tell why it gets the job done.

So which one of the two perspectives, learnability or machine learning, is more insightful for linguistics? It depends on what kind of linguistics we have in mind. Linguistic performance is a tricky problem with lots of noise, few clear-cut generalizations, and no clear success state for the learner, so a problem like, say, speakers' use of slang words and how it is contingent on the social setting is probably better off with machine learning. Linguistic competence, on the other hand, is sufficiently abstracted that we can have rigorous mathematical definitions of the object of study and what it means to successfully solve a given learning problem. And since mathematical proofs offer the highest standard of truth that we should strive for whenever possible, learnability is the better perspective for our purposes.

2 A Closer Look at Learnability

2.1 General Remarks on Learning

The definition of explanatory adequacy given at the beginning require the theory to explain how a native speaker acquires their grammatical knowledge. But what exactly does that mean? For the sake of simplicity, let's assume that "explain" here just means that we have a learning algorithm, or simply *learner*. Chomsky might have had a stronger condition in mind, e.g. that the learner must somehow be defined via the grammar formalism, but with highly abstract issues like this it is always more instructive to consider the simplest case first before piling on additional complicating factors. Assuming then that any learner will do, what exactly is it the learner has to accomplish? What does it mean to acquire grammatical knowledge, how can we translate that into a mathematical statement?

There are of course many perfectly valid answers to this, so we will have to choose between these alternatives according to what exactly it is we are interested in. First of all, there are two ways of thinking about acquiring grammatical knowledge: learning a language, and learning the grammar for that language. In the first case, a learner of phonology just has to arrive at a state where one can tell for every word whether it is well-formed. In the second case, the learner must also use the right grammar. This is usually how linguists think about acquisition, but it is a rather curious notion because many grammars generate exactly the same language.

For example, if G is a strictly 2-local grammar, and $G' := G \cup \{\ltimes\rtimes\}$, then the two generate exactly the same language because $\ltimes\rtimes$ is a useless bigram, it never occurs in any string. So a learner that acquires G' instead of G is still learning the correct language, only the grammar is slightly larger than it needs to be. Similarly, if strictly local grammars are actually formalized as lists rather than sets, then two grammars can be distinct despite containing exactly the same n-grams just because the order of n-grams differs. So for every grammar G there are $(|G| - 1)!$ many alternative grammars that generate exactly the same language. As a result there is no way to distinguish among these grammars based on well-formedness judgments.

The tacit assumption among linguists is that in such cases smaller grammars are to be preferred, and that differences in grammar can be detected with more sophisticated methods, e.g. psycholinguistic experiments. But the first one is a stipulation at best and a category mistake at one: while it is true that as scientists we should strive to find the simplest solutions and thus the smallest grammar, it is far from obvious that this scientific principle should be a condition on the learner. There are many cases where constructing a grammar for a formal language is possible yet one cannot determine whether there is a smaller grammar. If grammar size is an important factor in distinguishing adequate from inadequate learners, explanatory adequacy is unattainable under such circumstances. So if human language happens to be such a case, explanatory adequacy could never be achieved, which seems rather odd. The natural conclusion, then, is that grammar size is not an essential criterion.

The second assumption is that grammars can be distinguished even if they generate exactly the same language. Given a suitable linking theory between competence and performance, it is indeed conceivable that two grammars may make different predictions, e.g. in parsing, word recognition or possibly even the brain patterns on can observe. But to date there is no consensus on what such a linking theory has to look like, and more importantly, we do not know that speakers all have the same grammar in this sense. For example, whether a somebody is right-handed or left-handed has a profound difference on brain function, so we cannot rule out that right-handed people's grammars are different from that of left-handed people. Similarly, a psycholinguistic experiment such as a reading tasks does not measure the grammar in isolation but rather the compound output of working memory load, attention span, and so on. Unless there is a way to control for all these variables to a level of degree where one can detect even subtle differences in grammar, there is no evidence that speaker's learn the same grammar. In fact, behavioral data suggests the opposite: even when speakers agree on all relevant data judgments for a given phenomenon, they can still have different preferences for certain structures, or they may be more likely to use one than the other. Overall, then, there is no strong reason why "acquiring grammatical knowledge" must mean finding the right grammar rather than the right language.

Another important factor missing from the definition of explanatory adequacy are the parameters of learning, i.e. what kind of evidence the learner can draw from and how quickly learning must proceed. If our main interest is modelling human learning, then those issues are first and foremost an empirical matter, and whatever is available to an infant learner should also be available to our formal learner. The problem, though, is that it often unclear what evidence infants can draw from. We know for sure that they get positive evidence by virtue of constantly being exposed to natural language input. But they might also have access to negative evidence, for if you never hear a specific form or construction, that makes for strong probabilistic evidence that it is ill-formed. Setting aside the question of what counts as input, we also do not know how much information is extracted from that input. The actual surface forms — sequences of sounds for phonology, sequences of words for syntax — are definitely accessible, whereas it is much less clear how much syntactic structure can be inferred from prosody and semantics. The only thing we know for sure, then, is that language acquisition involves at least collections of surface forms. The best starting point, then, is to see what can or cannot be learned with such a restricted amount of information.

2.2 The Gold Paradigm

The learnability literature is full of different learning paradigms that make varying assumptions about the input available to the learner and the criteria that determine learning success. One of the oldest and most influential is the *Gold paradigm*, also known as *learning in the limit* or *learning in the limit from positive text*. The Gold paradigm makes very minimal assumptions about the input while enforcing a very strict notion of learning success. The learner is presented with strings from the target language, one after another. After each string presentation the learner formulates a theory (represented by a grammar) as to what language that string was drawn from. The learner is successful iff it guesses the right language after a finite number of steps and, crucially, does not change its guess at a later point.

Let us make these ideas more precise so that we know what exactly we are talking about before moving on.

Definition 6.1 (Text). Given a language L, a *(complete, positive) text* over L is a total mapping T from the natural numbers onto $L \cup \{\#\}$ (that is to say, it is an infinite string in which ever member of L occurs at least once and random noise is represented by $\#$). We denote by $T(i)$ the $i+1$-th element in the text, whereas $T[i]$ represents the length i prefix of T, i.e. $\langle T(0), T(1), \dots, T(i-1)\rangle$. For every sequence s, $content(s)$ is the set of elements of L that occur in s.

Definition 6.2 (Gold learning). A *learner* is a (possibly partial) function ϕ from sequences of expressions and # to languages. The learner *converges* on text T iff there is some i such that $\phi(T[i]) = \phi(T[j])$ for all $j \geq i$. The learner *identifies* language L iff it holds for every text T over L that ϕ converges on T and guesses the correct language (that is to say $L(\phi(T)) = content(T)$). Similarly, ϕ identifies a class $\mathscr{L}$ of languages iff it identifies every language in that class. A class of languages is *identifiable* or *learnable (in the limit)* iff there is a learner that identifies it.

Example 6.1 Every Language is Learnable

Here is a quick example that shows all the terminology and concepts of Gold learning in action. It also illuminates a common source of confusion: learnability is a property of language classes, not individual languages. When considered in isolation, every language is learnable.

Let L be some language. We show that $\{L\}$ is learnable in the limit by exhibiting a learner that identifies $\{L\}$. This learner is the constant function ϕ such that for every expression i $\phi(i) := L$.

Since ϕ is a constant function it makes the same guess for every prefix of any text T over L, so we have $\phi(T[i]) = L$ for every $i \geq 0$. This immediately implies that the learner converges on T, and since T is drawn from L we also have that $L(\phi(T)) = content(T)$. This will be the case for every text T over L, so ϕ identifies L, which means that it also identifies the class $\{L\}$.

The order of steps in the example is representative of learnability proofs in general.

1. Give a learner.
2. Show that the learner converges, that is to say, it does not alter its guess after a certain point.
3. Show that after it converges, the learner correctly guesses the target language. In other words, it identifies the language.

4. Show that the learner can do this for all languages in the class and all texts over them.

At this point you might already have a hunch that the Gold paradigm is not a fully realistic representation of language acquisition. You would be right, many factors are omitted or simplified. There are no constraints on how quickly a learner has to converge, so a learner that takes 300 years to find the target language is perfectly acceptable. The learner also has perfect recall since its guess at position i in the text is not just determined by the element at this position, which we denote $T(i)$, but by all the elements up to this position, which is represented by the minimally different $T[i]$. Moreover, the text contains a lot of information: noise is explicitly marked as such so that the learner is never mislead by faulty input, and every sentence of the language appears must appear in every text at some point. These four factors simplify the learning problem a lot.

At the same time, some factors make the Gold paradigm harder than the real life task. In particular, the learner has to succeed on every text, no matter how defunct and degenerate that text is. It is highly unlikely that a child could learn from input that consists predominantly of sentences from Wall Street Journal op-ed pieces — child-directed input starts out easy with simple short sentences and increases gradually in complexity. The Gold paradigm requires the learner to succeed even under highly adversarial circumstances. And the restriction to strings that are completely stripped off prosody and semantics also heightens the difficulty of the task.

It is important to keep in mind though that these aren't so much issues of empirical inadequacy as they are consequences of mathematical abstraction. Putting restrictions on the structure of the text makes proofs more involved, and including prosodic or semantic information requires more complex objects than just strings. By keeping the paradigm as mathematically simple as possible, we can quickly gain insights into why certain classes of languages are learnable and others are not, and these insights can be adapted to more adequate paradigms in subsequent work. The process isn't all that different from what we are currently doing in our analysis of phonology: set out the bare essentials that need to be accomplished, see how one might go about this with the least amount of assumptions, and where necessary revise the model in a conservative fashion so that one can easily tell which properties are being preserved, and why.

2.3 Most Classes of Languages are not Learnable

One of the key insights of the Gold paradigm is that a surprisingly large number of language classes is not learnable. More precisely, ever proper superset of the class of finite languages is not identifiable in the limit. This is even more surprising because the class of finite languages is learnable.

Theorem 6.3. The class Fin of finite languages is identifiable in the limit. ⌟

The key idea is that every finite language can be learned within a finite number of steps by simply memorizing all the forms encountered so far. Remember that this was our original idea for the list phonology model, so this theorem tells us that list phonology is a learnable model of phonology.

Proof. Let ϕ be a function that, for every text T and index i, maps $T[i]$ to $content(T[i])$. For every finite language L and text T over L, there must be a smallest i such that

$content(T[i]) = L$. Thus $\phi(T[i]) = \phi(T[j])$ for all $j \geq i$, wherefore the learner converges on T. But we also have $\phi(T[i]) = content(T[i]) = L$, and since T was arbitrary ϕ identifies L. But L, too, was arbitrary, showing that ϕ identifies Fin. □

The learner used in the proof is rather boring because it only memorizes forms without any kind of generalization — one of the biggest qualms we had with list phonology model. But generalization is risky as it involves formulating a hypothesis that goes beyond the mere data. Depending on how the space of languages is structured, it may be more or less safe to generalize in certain ways. In the worst case, safe generalization is impossible, and that is exactly the problem with any language class that properly includes all finite languages.

Theorem 6.4. Every class of languages that properly subsumes Fin is not identifiable in the limit. ⌟

Note that every class of this type must include at least one infinite language. So a learner that tries to learn this class must guess this infinite language after having seen only a finite amount of data — that's the convergence criterion. But there is no guarantee that this finite amount of data isn't actually sampled from a finite language that is included in the infinite one, in which case the learner would have accidentally overgeneralized. Since there is no negative evidence in the Gold paradigm, the learner has no opportunity to revise its decision and makes the wrong guess.

Proof. We show that $\mathscr{L} := \text{Fin} \cup \{L\}$ is not identifiable, where L is some infinite language. This immediately implies that no proper superset of Fin can be identified in the limit.

Suppose that ϕ identifies $\mathscr{L}$. Then ϕ must identify every member of $\mathscr{L}$, including L. By definition, then, it must hold for every text T over L that there is some i where ϕ converges on T. Clearly $content(T[i])$ is a finite language F. Let T_F be a text over F such that $T_F[i] = T[i]$. Then ϕ must also converge on T_F at index i, but since ϕ identifies L over T it must be the case that $\phi(T_F)$ is L rather than F. So ϕ does not identify F over T_F, wherefore it does not identify F or $\mathscr{L}$ either. □

The two theorems home in on the central problem of learning a given space of languages in the Gold paradigm: on the one hand the learner has to generalize from a finite amount of input to an infinite language, but on the other hand this generalization step needs to be conservative enough that the learner can avoid from overgeneralization. Other learning paradigms may give the learner some means of recovering from overgeneralization to some degree — e.g. via probabilistic reasoning, negative evidence, or an omniscient oracle that can be asked specific questions — but the tension between finite data and infinite target languages remains. This tension can be resolved correctly only if the space is structured in a specific way and the learner is aware of that structure.

There are some clear parallels to Universal Grammar (UG) in linguistics. However, UG does not really talk about the structure of the language space but rather puts restrictions on what the elements of each language in that space look like. By itself that is not enough to guarantee learnability (unless the restrictions are severe enough to render the space of languages finite, see Sec. 3.2). Nonetheless UG considerations and learnability can be linked in intriguing ways, as we will see next in our look at the (non-)learnability of strictly local languages.

3 Learning Strictly Local Languages

3.1 SL_k is Learnable

Remember that the class of strictly local languages properly subsumes the class of all finite languages, so this immediately tells us that the full class is not Gold-learnable.

Corollary 6.5. SL is not identifiable in the limit. ⌟

But the full class of strictly local languages isn't all that interesting for our purposes anyways. By definition there is an upper bound k on the domain of local phonological processes, otherwise they would be non-local processes. It seems reasonable that the value of k is part of UG so that human learners only entertain hypothesis that fall within the realm of strictly k-local languages (or maybe there is some language external factor that limits the class of hypothesis to SL_k; at any rate there is some learning prior in place). And in contrast to SL, SL_k is learnable for every choice of k.

Theorem 6.6. SL_k is identifiable in the limit for every $k \geq 0$. ⌟

By now you probably know already what the learner has to do: memory all k-grams of all strings seen so far.

Proof. Let ϕ_k be such that for every text T of strictly k-local language L, $\phi_k(T[i]) = L(G)$, where $G := \bigcup_{w \in content(T[i])} k\text{-grams}(w)$ is a positive strictly k-local grammar. Since there are only finitely many distinct k-grams, ϕ_k is guaranteed to converge at some index i. At this point, G contains all k-grams that occur in some string in T, and only those, so $L(G) = L$. Since L and T were arbitrary, ϕ_k identifies SL_k. □

While the result is stated in terms of learning, we can also view it as a description of the language space that ϕ_k induces. That is to say, given some finite input sample, ϕ_k will generalize to a (possibly infinite) language, namely the smallest strictly k-local language that includes the input. So if all phonological dependencies are strictly k-local, that might not necessarily be due to a restriction of the grammar. Maybe phonological computations could work just as well with much larger values, or even the whole class of strictly local languages. But the learner ϕ_k only infers SL_k-grammars. No matter what input it is presented with, it will always generalize to a specific type of grammar. This allows us to factor language as an empirical object into at least two distinct components: the class of languages that can be generated by the grammar formalism, and the class of languages that are identified by the learner. Natural languages are the special type of language that belongs to both. So you see, learnablity is not just tied to language acquisition, it can also be invoked to explain typological facts.

3.2 Every Finite Class is Learnable

On a mathematical level, the learnability of SL_k is actually a corollary of a more fundamental fact: every finite class of languages is learnable. Intuitively, that's because the languages in a finite class can be tried one after another to test if they fit the text. As long as one does not pick a needlessly large language, overgeneration can be avoided and the right language will be found eventually. Since there are only a finite number of k-grams for every fixed alphabet, SL_k is finite as well and thus can be explored in this fashion.

Theorem 6.7. Every finite class $\mathscr{C}$ of languages is learnable. ⌟

Proof. Assume w.l.o.g. that $\mathscr{C}$ contains only n languages. Put these languages into order $L_1 \cdots L_n$ such that for all L_i, L_j, $i < j$ implies that L_i is not a proper superset of L_j. We now define a learner ϕ that always picks the "left-most" language that subsumes the content of the text: for all i, $\phi(T[i]) = L_k$, where $L_k \supseteq \mathit{content}(T[i])$ and there is no L_j such that $L_j \supseteq \mathit{content}(T[i])$ and $j < k$. Obviously ϕ converges on every text.

To see that ϕ also identifies $\mathscr{C}$, suppose that T is some text over $L_k \in \mathscr{C}$. Once ϕ converges at $T(i)$, it conjectures some $L_j \supseteq \mathit{content}(T[i])$. It cannot be the case that $j > k$ because ϕ picks the language with the lowest index. If $j < k$, then $L_j \not\supseteq L_k$, so there is some string $w \in L_k$ such that $w \notin L_j$. But w must occur in T at some point p, so $w \in \mathit{content}(T[p]) \not\subseteq L_j$, contradicting the earlier assumption that ϕ converges at $T(i)$ with $\phi(T[i]) = L_j \neq L_k$. The only possible case, then, is that $j = k$, wherefore $L_j = L_k = \phi(T[i])$. □

This theorem has several important implications. First of all, finite language classes are trivial to learn because we can always induce a suitable structure over this class that guarantees its learnability. Interesting learnability results are about infinite language classes, or home in on properties of finite language classes that make learning more efficient or would also guarantee learnability if the class were in fact infinite. Fortunately that is the case for the SL$_k$-learner.

3.3 Generalizing the Learner

The proof that SL$_k$ is identifiable in the limit is deceptively simple. In just five lines it establishes the learnability result, but it does not quite home in on why this works, what aspects of strictly local languages are essential for the proof, and what exactly the explored space looks like. In particular, is the learnability of SL$_k$ simply due to the fact that every finite class of languages is learnable, or is there more to SL$_k$ that would guarantee learnability even for These are important points; if we want to revise our formalism at some later point without losing the learnability property, we need to know what we can safely change.

Let's first think about the overall shape of SL$_k$. One useful trick is to identify each language with a grammar that generates it and think about that space of grammars instead. After all, one is just a different representation of the other. Suppose we are interested in the strictly 1-local languages over a and b. There are 4 different 1-grams over this alphabet: $\rtimes$, $\ltimes$, a, and b. Since every strictly 1-local grammar is a set of these 1-grams, the class of SL$_1$ grammars over a and b is the powerset of the set of these 4 bigrams. This means that there are $2^4 = 16$ different SL$_1$-grammars, and these grammars are partially ordered by the subset relation. We can actually represent this space graphically as a Hasse diagram, see Fig. 6.1.

Background Partially ordered sets

A (weak) partial order is a binary relation R over a set S that satisfies three conditions:

reflexive for all $x \in S$, $x \mathrel{R} x$

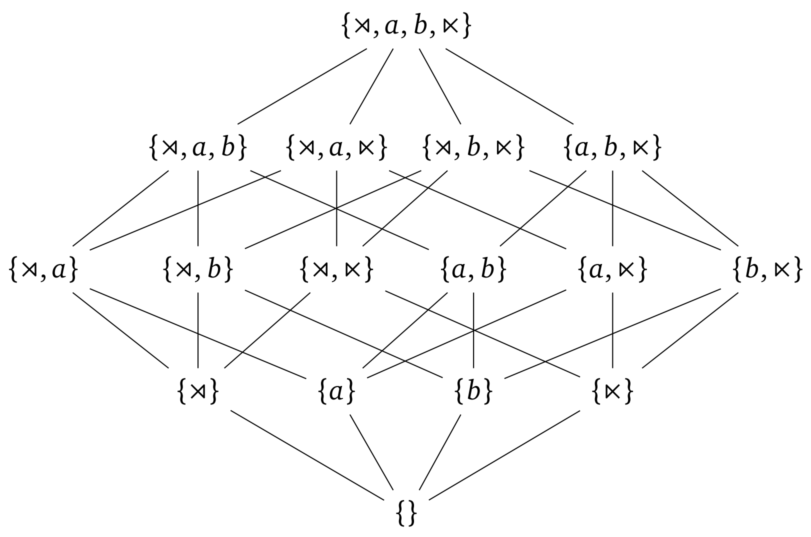

Figure 6.1: Lattice representation of the space of SL_1-grammars over a and b

antisymmetry for all $x, y \in S$, $x\ R\ y$ and $y\ R\ x$ jointly imply $x = y$

transitivity for all $x, y, z \in S$, $x\ R\ y$ and $y\ R\ z$ jointly imply $x\ R\ z$

A partially ordered set, or *poset*, is a set with a partial order defined over it.

As you can see this space has a lot of structure. First of all, there are unique bottom and top elements, which are $\{\}$ and $\{\rtimes, a, b, \ltimes\}$ in this case. It also holds that any two nodes have a unique greatest lower bound (*infimum*) and a unique least upper bound (*supremum*). For example, $\{\rtimes, a\}$ and $\{\rtimes, b\}$ have the greatest lower bound $\{\rtimes\}$ and the least upper bound $\{\rtimes, a, b, \}$. A partially ordered set where all nodes x and y have both an infimum and a supremum is called a *lattice*, and if it also has unique bottom and top elements it is a *bounded lattice*.

It is fairly easy to see that the class of SL_k grammars always forms a bounded lattice, the size of which depends on k the number of symbols in the alphabet. For example, there are 9 distinct bigrams over a, b (strictly speaking there are more, like $\ltimes a$, but those can never occur in a string), so the corresponding lattice has $2^9 = 512$ distinct nodes, which is way too much to draw. Still, they are all lattices.

The strategy used by ϕ_k can be viewed as a particular way of moving through such a lattice. The learner starts at the bottom of the lattice and the moves upwards whenever the current input cannot be accounted for by the grammar represented by the current node in the lattice. In the specific case of SL_k learning, this means that the input string contains a k-gram that is not part of the current grammar. Crucially, though, the learner can determine the shortest upward move that will take it to a grammar that can deal with all the input so far. Suppose that ϕ_1 is currently at the node $\{\rtimes, a, \ltimes\}$ and is now presented with the input b. That string is not generated by the grammar $\{\rtimes, a, \ltimes\}$, as its set of 1-grams is $\{\rtimes, b, \ltimes\}$. The learner infers from this that the grammar is $\{\rtimes, a, b, \ltimes\}$, but notice what is happening in the lattice: the learner isn't just moving upwards, it's moving to the supremum of the nodes that correspond to these two grammars!

By moving to the supremum rather than just any random higher node, the learner generalizes in the most careful and conservative way possible that is consistent with the data, which avoids overgeneralizing. If the supremum isn't the right grammar, then there will be more evidence further down the line and we can move to the next higher supremum, and so on. If, on the other hand, the learner simply moved to some arbitrary higher node, there might be no positive evidence that it has the wrong grammar and thus no way for it to correct its mistake.

The lattice-based learning perspective is so appealing because it is both intuitive and general. It doesn't really matter that the nodes in the lattice correspond to strictly local grammars, what matters is that the learner has a way to explore this space in a principled fashion without overgeneralizing. This can be done as long as the learner can derive a second node from the input such that the supremum of that second node and the learner's current position corresponds to the most conservative, empirically adequate revision of the learner's current hypothesis. So if a given class of languages can be shown to have a lattice structure — which is possible for both finite and infinite classes — and if we can define a method for picking the right suprema, then we have a learner for that class of languages.

This is also highly appealing from a psycholinguistic perspective since one would assume that learning strategies do not differ significantly for, say, local and non-local phonological processes. Once we move from local to non-local processes, we will see that this is indeed the case: both processes involve exactly the same lattice-based learning algorithms, with the only difference being what each lattice represents.