Theory of Computation

Thomas Zeugmann

Hokkaido University
Laboratory for Algorithmics

http://www-alg.ist.hokudai.ac.jp/~thomas/ToC/

Lecture 2: Introducing Formal Grammars

Grammars

We have to formalize what is meant by generating a language. If we look at natural languages, then we have the following situation: The set Σ consists of all words in the language. Although large, Σ is finite. What is usually done in speaking or writing natural languages is forming sentences. A typical sentence starts with a noun phrase followed by a verb phrase. Thus, we may describe this generation by

$$< \text{sentence} > \to < \text{noun phrase} >< \text{verb phrase} >$$

Grammars

We have to formalize what is meant by generating a language. If we look at natural languages, then we have the following situation: The set Σ consists of all words in the language. Although large, Σ is finite. What is usually done in speaking or writing natural languages is forming sentences. A typical sentence starts with a noun phrase followed by a verb phrase. Thus, we may describe this generation by

$$< \text{sentence} > \rightarrow < \text{noun phrase} >< \text{verb phrase} >$$

Clearly, more complicated sentences are generated by more complicated rules. If we look in a usual grammar book, e.g., for the English language, then we see that there are, however, only finitely many rules for generating sentences.

Formal Grammars

This suggest the following general definition of a grammar:

Definition 1

$\mathcal{G} = [T, N, \sigma, P]$ is said to be a ***grammar*** if

(1) T and N are alphabets with $T \cap N = \emptyset$;

(2) $\sigma \in N$;

(3) $P \subseteq ((T \cup N)^+ \setminus T^*) \times (T \cup N)^*$ is finite.

Formal Grammars

This suggest the following general definition of a grammar:

Definition 1

$\mathcal{G} = [T, N, \sigma, P]$ is said to be a ***grammar*** if

(1) T and N are alphabets with $T \cap N = \emptyset$;

(2) $\sigma \in N$;

(3) $P \subseteq ((T \cup N)^+ \setminus T^*) \times (T \cup N)^*$ is finite.

We call T the *terminal alphabet*, N the *nonterminal alphabet*, σ the *start symbol* and P the set of *productions* (or *rules*).

Formal Grammars

This suggest the following general definition of a grammar:

Definition 1

$\mathcal{G} = [T, N, \sigma, P]$ is said to be a ***grammar*** if

(1) T and N are alphabets with $T \cap N = \emptyset$;

(2) $\sigma \in N$;

(3) $P \subseteq ((T \cup N)^+ \setminus T^*) \times (T \cup N)^*$ is finite.

We call T the *terminal alphabet*, N the *nonterminal alphabet*, σ the *start symbol* and P the set of *productions* (or *rules*).

Usually, productions are written in the form $\alpha \rightarrow \beta$, where $\alpha \in (T \cup N)^+ \setminus T^*$ and $\beta \in (T \cup N)^*$.

Generating a Language by a Grammar I

Next, we have to explain how to generate a language using a grammar. This is done by the following definition:

Definition 2

Let $\mathcal{G} = [T, N, \sigma, P]$ be a grammar. Let $\alpha', \beta' \in (T \cup N)^*$. α' is said to ***directly generate*** β', written $\alpha' \Rightarrow \beta'$, if there exist $\alpha_1, \alpha_2, \alpha, \beta \in (T \cup N)^*$ such that $\alpha' = \alpha_1 \alpha \alpha_2$, $\beta' = \alpha_1 \beta \alpha_2$ and $\alpha \to \beta$ is in P. We write $\overset{*}{\Rightarrow}$ for the *reflexive transitive closure* of $\Rightarrow$.

Illustration

Example 1

Let $\mathcal{G} = [\{a, b\}, \{\sigma\}, \sigma, P]$, where
$P = \{\sigma \rightarrow \lambda,\ \sigma \rightarrow a,\ \sigma \rightarrow b,\ \sigma \rightarrow a\sigma a,\ \sigma \rightarrow b\sigma b\}$.

Illustration

Example 1

Let $\mathcal{G} = [\{a, b\}, \{\sigma\}, \sigma, P]$, where
$P = \{\sigma \to \lambda,\ \sigma \to a,\ \sigma \to b,\ \sigma \to a\sigma a,\ \sigma \to b\sigma b\}$.

Then we can directly generate a from σ, since $\sigma \to a$ is in P. Furthermore, we can generate the string $abba$ from σ as follows by using the rules $\sigma \to a\sigma a$, $\sigma \to b\sigma b$ and $\sigma \to \lambda$; i.e., we obtain

$$\sigma \Rightarrow a\sigma a \Rightarrow ab\sigma ba \Rightarrow abba\,. \tag{1}$$

Illustration

Example 1

Let $\mathcal{G} = [\{a, b\}, \{\sigma\}, \sigma, P]$, where
$P = \{\sigma \to \lambda,\ \sigma \to a,\ \sigma \to b,\ \sigma \to a\sigma a,\ \sigma \to b\sigma b\}$.

Then we can directly generate a from σ, since $\sigma \to a$ is in P. Furthermore, we can generate the string $abba$ from σ as follows by using the rules $\sigma \to a\sigma a$, $\sigma \to b\sigma b$ and $\sigma \to \lambda$; i.e., we obtain

$$\sigma \Rightarrow a\sigma a \Rightarrow ab\sigma ba \Rightarrow abba\,. \tag{1}$$

A sequence like Eq. (1) is called a *generation* or *derivation*. If a string s can be generated from a nonterminal h then we write $h \overset{*}{\Rightarrow} s$.

Generating a Language by a Grammar II

Finally, we can define the language generated by a grammar.

Generating a Language by a Grammar II

Finally, we can define the language generated by a grammar.

Definition 3

Let $\mathcal{G} = [T, N, \sigma, P]$ be a grammar. The ***language*** $\mathbf{L}(\mathcal{G})$ generated by $\mathcal{G}$ is defined as $L(\mathcal{G}) = \{s \mid s \in T^* \text{ and } \sigma \stackrel{*}{\Rightarrow} s\}$.

Generating a Language by a Grammar II

Finally, we can define the language generated by a grammar.

Definition 3

Let $\mathcal{G} = [T, N, \sigma, P]$ be a grammar. The ***language*** $\mathbf{L}(\mathcal{G})$ generated by $\mathcal{G}$ is defined as $L(\mathcal{G}) = \{s \mid s \in T^* \text{ and } \sigma \overset{*}{\Rightarrow} s\}$.

The family of all languages that can be generated by a grammar in the sense of Definition 2 is denoted by $\mathcal{L}_0$. These languages are also called *type-0 languages*, where 0 should remind us to *zero restrictions*.

An Example - Palindromes I

Recall that a *palindrome* is a string that reads the same from left to right and from right to left, e.g.,

AKASAKA

An Example - Palindromes I

Recall that a *palindrome* is a string that reads the same from left to right and from right to left, e.g.,

AKASAKA

トビコミコビト
にわとりとことりとわに
テングノハハノグンテ

An Example - Palindromes I

Recall that a *palindrome* is a string that reads the same from left to right and from right to left, e.g.,

AKASAKA

トビコミコビト
にわとりとことりとわに
テングノハハノグンテ

Let us look again at the language of all palindromes over $\Sigma = \{a, b\}$, i.e., $L_{pal} = \{w \mid w \in \{a, b\}^*,\ w = w^T\}$.

An Example - Palindromes I

Recall that a *palindrome* is a string that reads the same from left to right and from right to left, e.g.,

AKASAKA

トビコミコビト
にわとりとことりとわに
テングノハハノグンテ
Let us look again at the language of all palindromes over $\Sigma = \{a, b\}$, i.e., $L_{pal} = \{w \mid w \in \{a, b\}^*,\ w = w^T\}$.

Consider the grammar from Example 1, i.e.,
$\mathcal{G} = [\{a, b\}, \{\sigma\}, \sigma, P]$, where
$P = \{\sigma \rightarrow \lambda, \sigma \rightarrow a, \sigma \rightarrow b, \sigma \rightarrow a\sigma a, \sigma \rightarrow b\sigma b\}$.

An Example - Palindromes II

We have to show that $L_{pal} = L(\mathcal{G})$.

An Example - Palindromes II

We have to show that $L_{pal} = L(\mathcal{G})$.

Claim 1. $L_{pal} \subseteq L(\mathcal{G})$.

The proof is done inductively. For the induction basis, consider $w = \lambda$, $w = a$ and $w = b$. Since P contains $\sigma \rightarrow \lambda$, $\sigma \rightarrow a$, and $\sigma \rightarrow b$, we get $\sigma \overset{*}{\Rightarrow} w$ in all three cases.

An Example - Palindromes II

We have to show that $L_{pal} = L(\mathcal{G})$.

Claim 1. $L_{pal} \subseteq L(\mathcal{G})$.

The proof is done inductively. For the induction basis, consider $w = \lambda$, $w = a$ and $w = b$. Since P contains $\sigma \rightarrow \lambda$, $\sigma \rightarrow a$, and $\sigma \rightarrow b$, we get $\sigma \overset{*}{\Rightarrow} w$ in all three cases.

Induction Step: Now let $|w| \geqslant 2$. Since $w = w^{\mathsf{T}}$, w must begin and end with the same symbol, i.e., $w = ava$ or $w = bvb$, where v must be a palindrome, too.

By the induction hypothesis we have $\sigma \overset{*}{\Rightarrow} v$, and thus

$$\sigma \Rightarrow a\sigma a \overset{*}{\Rightarrow} ava \quad \text{proving the } w = ava \text{ case, or}$$
$$\sigma \Rightarrow b\sigma b \overset{*}{\Rightarrow} bvb \quad \text{proving the } w = bvb \text{ case.}$$

This shows Claim 1.

An Example - Palindromes III

Claim 2. $L(\mathcal{G}) \subseteq L_{pal}$.

Induction Basis: If the generation is done in one step, then one of the productions not containing σ on the right hand side must have been used, i.e., $\sigma \to \lambda$, $\sigma \to a$, or $\sigma \to b$. Thus, $\sigma \Rightarrow w$ results in $w = \lambda$, $w = a$ or $w = b$; hence $w \in L_{pal}$.

An Example - Palindromes III

Claim 2. $L(\mathcal{G}) \subseteq L_{pal}$.

Induction Basis: If the generation is done in one step, then one of the productions not containing σ on the right hand side must have been used, i.e., $\sigma \to \lambda$, $\sigma \to a$, or $\sigma \to b$. Thus, $\sigma \Rightarrow w$ results in $w = \lambda$, $w = a$ or $w = b$; hence $w \in L_{pal}$.

Induction Step: Suppose, the generation takes $n + 1$ steps, $n \geqslant 1$. Thus, we have

$$\begin{aligned} \sigma &\Rightarrow a\sigma a \overset{*}{\Rightarrow} ava \quad \text{or} \\ \sigma &\Rightarrow b\sigma b \overset{*}{\Rightarrow} bvb \end{aligned}$$

Since by the induction hypothesis, we know that $v \in L_{pal}$, we get in both cases $w \in L_{pal}$. $\blacksquare$

Regular Grammars

Definition 4

A grammar $\mathcal{G} = [T, N, \sigma, P]$ is said to be ***regular*** provided for all $\alpha \ \rightarrow \ \beta$ in P we have $\alpha \in N$ and $\beta \in T^* \cup T^*N$.

A language L is said to be ***regular*** if there exists a regular grammar $\mathcal{G}$ such that $L = L(\mathcal{G})$. By $\mathcal{REG}$ we denote the set of all regular languages.

Regular Grammars

Definition 4

A grammar $\mathcal{G} = [T, N, \sigma, P]$ is said to be ***regular*** provided for all $\alpha \rightarrow \beta$ in P we have $\alpha \in N$ and $\beta \in T^* \cup T^*N$.

A language L is said to be ***regular*** if there exists a regular grammar $\mathcal{G}$ such that $L = L(\mathcal{G})$. By $\mathcal{REG}$ we denote the set of all regular languages.

Example 2

Let $\mathcal{G} = [\{a, b\}, \{\sigma\}, \sigma, P]$ with $P = \{\sigma \rightarrow ab,\ \sigma \rightarrow a\sigma\}$.
$\mathcal{G}$ is regular and $L(\mathcal{G}) = \{a^n b \mid n \geqslant 1\}$ is a regular language.

Examples for Regular Languages

Example 3

Let $\mathcal{G} = [\{a, b\}, \{\sigma\}, \sigma, P]$ with $P = \{\sigma \rightarrow \lambda,\ \sigma \rightarrow a\sigma,\ \sigma \rightarrow b\sigma\}$.

Again, $\mathcal{G}$ is regular and $L(\mathcal{G}) = \Sigma^*$.

Consequently, Σ^* is a regular language.

Examples for Regular Languages

Example 3

Let $\mathcal{G} = [\{a, b\}, \{\sigma\}, \sigma, P]$ with $P = \{\sigma \to \lambda, \ \sigma \to a\sigma, \ \sigma \to b\sigma\}$.

Again, $\mathcal{G}$ is regular and $L(\mathcal{G}) = \Sigma^*$.

Consequently, Σ^* is a regular language.

Example 4

Let Σ be any alphabet, and let $X \subseteq \Sigma^*$ be any finite set. Then, for $\mathcal{G} = [\Sigma, \{\sigma\}, \sigma, P]$ with $P = \{\sigma \to s \mid s \in X\}$, we have $L(\mathcal{G}) = X$.

Consequently, every *finite* language is regular.

What else is regular?

Question

Which languages are regular?

What else is regular?

Question

Which languages are regular?

For answering this question, we first deal with *closure* properties.

Closure Properties

Theorem 1

The regular languages are closed under union, product and Kleene closure.

Closure Properties

Theorem 1

The regular languages are closed under union, product and Kleene closure.

Proof. Let L_1 and L_2 be any regular languages. Since L_1 and L_2 are regular, there are regular grammars $\mathcal{G}_1 = [T_1, N_1, \sigma_1, P_1]$ and $\mathcal{G}_2 = [T_2, N_2, \sigma_2, P_2]$ such that $L_i = L(\mathcal{G}_i)$ for $i = 1, 2$. Without loss of generality, we may assume that $N_1 \cap N_2 = \emptyset$ for otherwise we simply rename the nonterminals appropriately.

We start with the union. We have to show that $L = L_1 \cup L_2$ is regular.

Closure Properties

Theorem 1

The regular languages are closed under union, product and Kleene closure.

Proof. Let L_1 and L_2 be any regular languages. Since L_1 and L_2 are regular, there are regular grammars $\mathcal{G}_1 = [T_1, N_1, \sigma_1, P_1]$ and $\mathcal{G}_2 = [T_2, N_2, \sigma_2, P_2]$ such that $L_i = L(\mathcal{G}_i)$ for $i = 1, 2$. Without loss of generality, we may assume that $N_1 \cap N_2 = \emptyset$ for otherwise we simply rename the nonterminals appropriately.

We start with the union. We have to show that $L = L_1 \cup L_2$ is regular. Now, let

$$\mathcal{G}_{union} = [T_1 \cup T_2, N_1 \cup N_2 \cup \{\sigma\}, \sigma, P_1 \cup P_2 \cup \{\sigma \to \sigma_1, \ \sigma \to \sigma_2\}] \,.$$

By construction, $\mathcal{G}_{union}$ is regular.

Closure under Union

Claim 1. $L = L(\mathcal{G}_{union})$.

We have to start every generation of strings with σ. Thus, there are two possibilities, i.e., $\sigma \to \sigma_1$ and $\sigma \to \sigma_2$. In the first case, we can continue with all generations that start with σ_1 yielding all strings in L_1. In the second case, we can continue with σ_2, thus getting all strings in L_2. Consequently, $L_1 \cup L_2 \subseteq L$.

Closure under Union

Claim 1. $L = L(\mathcal{G}_{union})$.

We have to start every generation of strings with σ. Thus, there are two possibilities, i.e., $\sigma \rightarrow \sigma_1$ and $\sigma \rightarrow \sigma_2$. In the first case, we can continue with all generations that start with σ_1 yielding all strings in L_1. In the second case, we can continue with σ_2, thus getting all strings in L_2. Consequently, $L_1 \cup L_2 \subseteq L$.

On the other hand, $L \subseteq L_1 \cup L_2$ by construction. Hence, $L = L_1 \cup L_2$. ∎ (union)

Closure under Product I

We have to show that L_1L_2 is regular. A first idea might be to use a construction analogous to the one above, i.e., to take as a new starting production $\sigma \rightarrow \sigma_1\sigma_2$.

Unfortunately, this production is not regular. We have to be a bit more careful. But the underlying idea is fine, we just have to replace it by a sequential construction.

Closure under Product I

We have to show that L_1L_2 is regular. A first idea might be to use a construction analogous to the one above, i.e., to take as a new starting production $\sigma \rightarrow \sigma_1\sigma_2$.

Unfortunately, this production is not regular. We have to be a bit more careful. But the underlying idea is fine, we just have to replace it by a sequential construction.

The idea for doing that is easily described. Let $s_1 \in L_1$ and $s_2 \in L_2$. We want to generate s_1s_2. Then, starting with σ_1 there is a generation $\sigma_1 \Rightarrow w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow s_1$. But instead of finishing the generation at that point, we want to have the possibility to continue to generate s_2. Thus, all we need is a production having a right hand side resulting in $s_1\sigma_2$.
This idea can be formalized as follows:

Let $\mathcal{G}_{prod} = [T_1 \cup T_2, N_1 \cup N_2, \sigma_1, P]$, where

$$P = P_1 \setminus \{h \rightarrow s \mid s \in T_1^*,\ h \in N_1\} \cup \{h \rightarrow s\sigma_2 \mid h \rightarrow s \in P_1,\ s \in T_1^*\} \cup P_2\ .$$

By construction, $\mathcal{G}_{prod}$ is regular.

Let $\mathcal{G}_{prod} = [T_1 \cup T_2, N_1 \cup N_2, \sigma_1, P]$, where

$$P = P_1 \setminus \{h \rightarrow s \mid s \in T_1^*,\ h \in N_1\} \cup \{h \rightarrow s\sigma_2 \mid h \rightarrow s \in P_1,\ s \in T_1^*\} \cup P_2\ .$$

By construction, $\mathcal{G}_{prod}$ is regular.

Claim 2. $L(\mathcal{G}_{prod}) = L_1L_2$.

Clearly, $L(\mathcal{G}_{prod}) \subseteq L_1L_2$. We show $L_1L_2 \subseteq L(\mathcal{G}_{prod})$. Let $s \in L_1L_2$. Then, there are $s_1 \in L_1$ and $s_2 \in L_2$ such that $s = s_1s_2$. Since $s_1 \in L_1$, there is a generation $\sigma_1 \Rightarrow w_1 \Rightarrow \cdots \Rightarrow w_n \Rightarrow s_1$ in $\mathcal{G}_1$. So, w_n must contain precisely one nonterminal, say h, and thus $w_n = wh$. Since $w_n \Rightarrow s_1$ and $s_1 \in T_1^*$, we must have applied a production $h \rightarrow s$, $s \in T_1^*$ such that $wh \Rightarrow ws = s_1$.

Let $\mathcal{G}_{prod} = [T_1 \cup T_2, N_1 \cup N_2, \sigma_1, P]$, where

$$\begin{aligned} P &= P_1 \setminus \{h \rightarrow s \mid s \in T_1^*,\ h \in N_1\} \\ &\quad \cup \{h \rightarrow s\sigma_2 \mid h \rightarrow s \in P_1,\ s \in T_1^*\} \cup P_2 \,. \end{aligned}$$

By construction, $\mathcal{G}_{prod}$ is regular.

Claim 2. $L(\mathcal{G}_{prod}) = L_1L_2$.

Clearly, $L(\mathcal{G}_{prod}) \subseteq L_1L_2$. We show $L_1L_2 \subseteq L(\mathcal{G}_{prod})$. Let $s \in L_1L_2$. Then, there are $s_1 \in L_1$ and $s_2 \in L_2$ such that $s = s_1s_2$. Since $s_1 \in L_1$, there is a generation $\sigma_1 \Rightarrow w_1 \Rightarrow \cdots \Rightarrow w_n \Rightarrow s_1$ in $\mathcal{G}_1$. So, w_n must contain precisely one nonterminal, say h, and thus $w_n = wh$. Since $w_n \Rightarrow s_1$ and $s_1 \in T_1^*$, we must have applied a production $h \rightarrow s$, $s \in T_1^*$ such that $wh \Rightarrow ws = s_1$. But in $\mathcal{G}_{prod}$ all these productions have been replaced by $h \rightarrow s\sigma_2$. Hence, the last generation $w_n \Rightarrow s_1$ is now replaced by $wh \Rightarrow ws\sigma_2$. Now, we apply the productions from P_2 to generate s_2 which is possible, since $s_2 \in L_2$. ∎ (product)

Closure under Kleene Closure

Let L be a regular language, and let $\mathcal{G} = [T, N, \sigma, P]$ be a regular grammar such that $L = L(\mathcal{G})$. We have to show that L^* is regular.

Closure under Kleene Closure

Let L be a regular language, and let $\mathcal{G} = [T, N, \sigma, P]$ be a regular grammar such that $L = L(\mathcal{G})$. We have to show that L^* is regular.

By definition $L^* = \bigcup_{i \in \mathbb{N}} L^i$. Since $L^0 = \{\lambda\}$, we have to make sure that λ can be generated. This is obvious if $\lambda \in L$. Otherwise, we simply add the production $\sigma \to \lambda$. The rest is done analogously as in the product case, i.e., we set

$$\mathcal{G}^* = [T, N \cup \{\sigma^*\}, \sigma^*, P^*], \text{ where}$$

$$P^* = P \cup \{h \to s\sigma \mid h \to s \in P,\ s \in T^*\} \cup \{\sigma^* \to \sigma,\ \sigma^* \to \lambda\}\,.$$

We leave it as an exercise to prove that $L(\mathcal{G}^*) = L^*$. ∎

Equivalence of Grammars

We finish this lecture by defining the equivalence of grammars.

Definition 5

Let $\mathcal{G}$ and $\hat{\mathcal{G}}$ be any grammars. $\mathcal{G}$ and $\hat{\mathcal{G}}$ are said to be ***equivalent*** if $L(\mathcal{G}) = L(\hat{\mathcal{G}})$.

Equivalence of Grammars

We finish this lecture by defining the equivalence of grammars.

Definition 5

Let $\mathcal{G}$ and $\hat{\mathcal{G}}$ be any grammars. $\mathcal{G}$ and $\hat{\mathcal{G}}$ are said to be ***equivalent*** if $L(\mathcal{G}) = L(\hat{\mathcal{G}})$.

For having an example for equivalent grammars, we consider
$\mathcal{G} = [\{a\}, \{\sigma\}, \sigma, \{\sigma \to a\sigma a,\ \sigma \to aa,\ \sigma \to a\}]$,
and the following grammar:
$\hat{\mathcal{G}} = [\{a\}, \{\sigma\}, \sigma, \{\sigma \to a,\ \sigma \to a\sigma\}]$.

Equivalence of Grammars

We finish this lecture by defining the equivalence of grammars.

Definition 5

Let $\mathcal{G}$ and $\hat{\mathcal{G}}$ be any grammars. $\mathcal{G}$ and $\hat{\mathcal{G}}$ are said to be ***equivalent*** if $L(\mathcal{G}) = L(\hat{\mathcal{G}})$.

For having an example for equivalent grammars, we consider
$\mathcal{G} = [\{a\}, \{\sigma\}, \sigma, \{\sigma \rightarrow a\sigma a,\ \sigma \rightarrow aa,\ \sigma \rightarrow a\}]$,
and the following grammar:
$\hat{\mathcal{G}} = [\{a\}, \{\sigma\}, \sigma, \{\sigma \rightarrow a,\ \sigma \rightarrow a\sigma\}]$.

Now, it is easy to see that $L(\mathcal{G}) = \{a\}^+ = L(\hat{\mathcal{G}})$, and hence $\mathcal{G}$ and $\hat{\mathcal{G}}$ are equivalent.

Equivalence of Grammars

We finish this lecture by defining the equivalence of grammars.

Definition 5

Let $\mathcal{G}$ and $\hat{\mathcal{G}}$ be any grammars. $\mathcal{G}$ and $\hat{\mathcal{G}}$ are said to be ***equivalent*** if $L(\mathcal{G}) = L(\hat{\mathcal{G}})$.

For having an example for equivalent grammars, we consider
$\mathcal{G} = [\{a\}, \{\sigma\}, \sigma, \{\sigma \to a\sigma a,\ \sigma \to aa,\ \sigma \to a\}]$,
and the following grammar:
$\hat{\mathcal{G}} = [\{a\}, \{\sigma\}, \sigma, \{\sigma \to a,\ \sigma \to a\sigma\}]$.

Now, it is easy to see that $L(\mathcal{G}) = \{a\}^+ = L(\hat{\mathcal{G}})$, and hence $\mathcal{G}$ and $\hat{\mathcal{G}}$ are equivalent.

Note, however, that $\hat{\mathcal{G}}$ is regular while $\mathcal{G}$ is *not*.

Thank you!