Tema 1

Llenguatges formals i computació

Aquest manual té com a objectiu l'estudi dels llenguatges formals, així com dels seus generadors i acceptors juntament amb l'estreta relació que hi ha entre els llenguatges formals i la teoria de la computació.

El problema té una formulació matemàtica molt clara a partir del concepte de símbol, les operacions que s'hi poden definir i de les seues propietats..

1.1 Símbols, cadenes i llenguatges

Considerarem un conjunt *finit* i *no buit* de símbols que anomenarem **alfabet** o **vocabulari**. Per exemple $\{a, b, c\}$, $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ o {`vertader, fals, indefinit`} serien alfabets vàlids. El conjunt de tots els nombres enters[1] no seria un alfabet vàlid.

Podem definir un símbol com una etiqueta o una entitat que no té, en principi, cap significat i que és indivisible. Una **cadena** (de símbols) també anomenada **sentència** o **paraula** és una successió finita de símbols d'un determinat alfabet.

Per exemple, *aaabbba*, 1024 i `vertader`·`vertader`·`indefinit`[2], serien exemples de cadenes sobre cada un dels alfabets anteriors.

Cada cadena de símbols té associada una **longitud** que es defineix com el nombre de símbols que formen una cadena. Les cadenes anteriors tenen longitud 7, 4 i 3, respectivament. Escriurem la longitud d'una cadena x com a $|x|$.

Estendrem la notació per referirnos al nombre de símbols d'un cert tipus que conté una cadena. Així, si x es una cadena sobre Σi A és un subconjunt de Σ, $|x|_A$ representarà el nombre de símbols del conjunt A que conté x. Per al cas particular que A només continga un símbol, a, escriurem $|x|_a$ en lloc de $|x|_{\{}a\}$. Per exemple, si $x = aaabbba$ és una cadena sobre $\{a, b, c\}$, $|x|_a = 4$, $|x|_b = 3$, $|x|_c = 0$,i $|x|_{\{a,b\}} = |x| = 7$.

[1] No els nombres mateixos sinó entesos com a símbols escrits en un paper, per exemple.

[2] Hem utilitzat el símbol · per fer més clara la separació en símbols de la cadena. Una altra pràctica habitual en aquests casos és envoltar els símbols amb < i >. Aleshores escriuríem la cadena com a <`vertader`><`vertader`><`indefinit`>).

Diem que una cadena y és **prefix** d'una altra cadena x si i només si existeix alguna cadena z tal que $x = yz$.

Diem que una cadena y és **sufix** d'una altra cadena x si i només si existeix alguna cadena z tal que $x = zy$.

Diem que una cadena y és **factor** (o **infix** o **subcadena**) d'una altra cadena x si i només si existeixen cadenes z i w tals que $x = zyt$.

Els prefixs i sufixs són casos particulars de factors. Es parla de factors (prefixs o sufixs) propis quan aquests són diferents de la cadena buida i de la cadena a què es refereixen.

Anomenem **factorització** de x a una descomposició de la cadena x en un nombre finit de factors, $x = y_1 \cdot y_2 \cdots y_k$.

Si anomenem $P(\Sigma)$ al conjunt de totes les possibles cadenes sobre un determinat alfabet[3], es poden definir certes operacions tancades sobre aquest conjunt la més important de les quals és la concatenació.

Siguen x i y dues cadenes sobre un determinat alfabet Σ. Anomenem **concatenació** de x amb y a la cadena resultat de juxtaposar les dues cadenes (x a l'esquerra i y a la dreta) i ho representarem com a $x \cdot y$ o simplement com a xy.

Es pot demostrar fàcilment que el conjunt $P(\Sigma)$ amb la operació $\cdot$ té estructura de **monoide** i, per tant, també de **semigrup**. L'element neutre d'aquest conjunt respecte de la concatenació (que ho és per l'esquerra i per la dreta) és una cadena especial que no té cap símbol i que anomenem **cadena buida** o **cadena nul·la** i que escriurem en aquest manual[4] com a ε.

Siga un determinat alfabet Σ, anomenem **llenguatge** a tot conjunt de cadenes sobre Σ.

Alguns exemples de llenguatges sobre els alfabets anteriors serien

$\{a, aa, aaa, aaaa, \dots\}$, $\{0, 10, 20\}$ i $\{$<fals><fals>$, \varepsilon\}$. Les talles o grandàries (nombre de cadenes) de cada llenguatge serien ∞, 3 i 2, respectivament. Escriurem la grandària d'un llenguatge L com a $|L|$. Una altra forma més rigorosa d'expressar el primer d'aquests llenguatges és

$$\{a^n \mid n \geq 1\}$$

Sobre qualsevol alfabet, existeixen sempre dos llenguatges especials que reben el nom de **llenguatge buit** i **llenguatge format per la cadena buida** que s'escriuen com a $\emptyset$ i $\{\varepsilon\}$, respectiva-

[3] Fem notar que aquest conjunt és infinit.

[4] En altres texts se l'escriu també com a λ.

ment.

Aquests dos llenguatges són els elements neutres dins del conjunt de tots els possibles llenguatges respecte de les operacions **unió** i **concatenació de llenguatges**[5], respectivament.

Es pot demostrar que, respecte a aquestes dues operacions, el conjunt de tots els possibles llenguatges té estructura de monoide commutatiu o abelià i de monoide, respectivament.

Donat un alfabet Σ qualsevol, s'hi suposa donada una relació d'ordre total que dóna lloc al que s'anomena **ordenació alfabètica**. A partir d'aquesta es pot definir el que s'anomena **ordenació lexicogràfica** o **ordenació canònica** del conjunt $P(\Sigma)$ que consisteix a ordenar les seues cadenes de menor a major longitud i, dins d'una mateixa longitud, ordenar-les segons l'ordre alfabètic[6]

Aquesta ordenació del conjunt $P(\Sigma)$ fa que es puga establir una correspondència o bijecció entre aquest conjunt i els nombres naturals la qual cosa implica que $P(\Sigma)$ és sempre un conjunt **numerable**.

1.1.1 Operacions sobre cadenes

A banda de la concatenació, es poden definir altres operacions sobre cadenes.

Es defineix la **potència** i-èsima d'una cadena x com la concatenació de x amb ella mateixa i vegades. I s'escriu

$$x^i = \underbrace{x \cdots x}_{i \text{ vegades}}$$

Es considera, per definició, que $x^0 = \varepsilon$.

La potència admet també una definició recursiva:

$$x^i = \begin{cases} \varepsilon & \text{si } i = 0 \\ x \cdot x^{i-1} & \text{si } i \geq 1 \end{cases}$$

Es compleixen les següents propietats:

$$x^i x^j = x^{i+j}$$

$$x^{i+1} = x^i \cdot x = x \cdot x^i$$

$$|x^i| = i \cdot |x|$$

[5] La operació concatenació s'estén trivialment al cas de llenguatges en definir

$$L_1 \cdot L_2 = \{z | z = x \cdot y, x \in L_1, y \in L_2\}$$

[6] Com es fa en els diccionaris.

La **inversió** o **reflexió** d'una cadena x, que escriurem com a x^{-1}, és una altra cadena formada pels mateixos símbols però en ordre invers.

$$x^{-1} = a_n \cdots a_2 a_1 \text{ si } x = a_1 \cdots a_n \text{ i } a_i \in \Sigma$$

La inversió també admet la següent definició recursiva:

$$x^{-1} = \begin{cases} \varepsilon & \text{si } |x| = 0 \\ & \wedge\ x = y \cdot a\ \wedge\ |a| = 1 \\ a \cdot y^{-1} & \text{si } |x| \geq 1 \end{cases}$$

1.1.2 Operacions sobre llenguatges

Com que els llenguatges són conjunts (de cadenes) s'hi poden aplicar totes les operacions usuals sobre conjunts com ara la unió, intersecció, complement i resta. Aquestes operacions les escriurem com a

$$L_1 \cup L_2 \qquad L_1 \cap L_2 \qquad \overline{L_1} \qquad L_1 - L_2$$

A més a més, les operacions sobre cadenes es poden estendre al cas de llenguatges de la mateixa manera que s'ha fet amb la concatenació.

En general, donada qualsevol operació interna m-ària sobre cadenes, OP la seua extensió a llenguatges ve donada per

$$OP(L_1, \ldots, L_m) = \{x \in \Sigma \mid x = OP(x_1, \ldots, x_m), x_i \in L_i, i = 1, 2, \ldots m\}$$

A banda, hi ha altres operacions específiques sobre llenguatges

El **quocient** de dos llenguatges, L_1 i L_2, ve donat per aquells prefixs de cadenes de L_1 que es poden factoritzar com al mateix prefix seguit d'un sufix en L_2.

$$L_1/L_2 = \{x \mid \exists y \in L_2\ :\ xy \in L_1\}$$

El **tancament** o **clausura** (també anomenat tancament estrella o tancament de Kleene) d'un llenguatge L es defineix com la unió (infinita) de les successives potències del llenguatge:

$$L^\star = \bigcup_{i \geq 0} L^i$$

Si de l'anterior definició s'exclou el terme L^0 s'obté l'anomenat **tancament positiu**:

$$L^+ = \bigcup_{i \geq 1} L^i$$

Si, abusant de la notació, considerem l'alfabet Σcom un llenguatge finit format per cadenes de longitud 1 podem escriure

$$\Sigma^\star = P(\Sigma) \qquad \Sigma^+ = P(\Sigma) - \{\varepsilon\}$$

Aquests dos conjunts reben el nom de **monoide lliure** i **semigrup lliure** engendrats per Σ. A partir d'ara utilitzarem aquests noms i la notació $\Sigma^\star$ i Σ^+ per referir-nos a aquests conjunts.

1.2 Generació de llenguatges

En el context de la teoria de llenguatges formals només tenen importància els llenguatges que són infinits. Fins i tot dins d'aquests, els més importants són els que defineixen les cadenes que hi pertanyen en funció d'una certa estructura dins de les cadenes.

Un exemple d'aquest tipus de llenguatge és el format per aquelles cadenes sobre l'alfabet $\Sigma_{\text{ex}} = \{x, +, *, (,)\}$ que són expressions aritmètiques vàlides (ben parentitzades). És a dir,

$$L_{\text{ex}} = \{x, x + x, x + x * x, (x), (x + x) * x, \dots\}$$

Encara que està ben clar quin és aquest llenguatge, no se'l pot definir d'una forma compacta (amb una descripció finita) de la mateixa manera que s'ha fet en els exemples anteriors.

Una forma de definir aquest tipus de llenguatges es mijantçant l'ús d'una definició recursiva. Una forma convenient de fer açò en el present context ve donada pels sistemes de reescriptura un cas particular dels quals en són les gramàtiques.

Una **gramàtica** és una estructura de la forma $G(V_T, V_N, P, S)$ on

- $V_T = \Sigma$ és l'alfabet o vocabulari terminal.
- V_N és l'alfabet no terminal (o conjunt de símbols auxiliars).
- P és un conjunt de **produccions** de la forma $\alpha \to \beta$ on es compleix que $\beta \in V^*$ i $\alpha \in V^* V_N^* V^*$ $(V = V_T \cup V_N)$.
- $S \in V_N$ és un símbol no terminal especial anomenat **axioma**.

Una producció $\alpha \to \beta$ significa que la cadena α es pot reescriure com a β (es pot canviar una per l'altra).

Diem que una cadena y es deriva directament a partir d'una cadena x segons una gramàtica G, i ho escrivim com a $x \underset{G}{\Rightarrow} y$ si i solament si $\exists u \to v \in P \ \wedge \ x = \alpha u \beta, \ y = \alpha v \beta$.

Diem que una cadena y es deriva a partir d'una cadena x segons una gramàtica G, i ho escrivim com a $x \overset{\star}{\underset{G}{\Rightarrow}} y$ si i solament si $x = y$ o si existeix una seqüència finita de paraules $\{a_i\}_{i=1}^k$ tal que

$x \underset{G}{\Rightarrow} a_1 \underset{G}{\Rightarrow} a_2 \underset{G}{\Rightarrow} \cdots \underset{G}{\Rightarrow} y$.

Les cadenes de V que es poden derivar a partir de l'axioma reben el nom de **formes sentencials** de G. Si, a més a mes, són cadenes exclusivament de V_T es diu que són sentències de G.

Anomenem **llenguatge generat per una gramàtica**, G, i l'escriurem com a $L(G)$ el conjunt de cadenes de símbols terminals que es poden obtenir mijantçant derivacions a partir de l'axioma. És a dir,

$$L(G) = \{w \in V_T^* \mid S \overset{+}{\underset{G}{\Rightarrow}} w\}$$

Per exemple, el llenguatge L_{ex} és generat per la gramàtica $G_{\text{ex}} = (\Sigma_{\text{ex}}, \{S\}, P, S)$ on P és

$$S \to S + S | S * S | (S) | x$$

Un exemple de derivació amb aquesta gramàtica seria

$$\underline{S} \Rightarrow \underline{S} * S \Rightarrow (\underline{S}) * S \Rightarrow (\underline{S} + \underline{S}) * \underline{S} \quad \overset{+}{\Rightarrow} \quad (x + x) * x$$

En aquest exemple hem subratllat en cada forma sentencial els no terminals sobre els quals s'aplica la següent producció (la producció que s'aplica és òbvia per inspecció de la cadena derivada).

Es diu que dues gramàtiques són equivalents si generen el mateix llenguatge.

$$L(G_1) = L(G_2)$$

Es pot demostrar mijantçant un argument purament numèric i relativament senzill que existeixen llenguatges que no són generats per cap gramàtica. Açò ve donat perque el conjunt de totes les gramàtiques sobre un determinat alfabet és numerable (per ser una gramàtica una descripció finita) mentre que el conjunt de tots els possibles llenguatges sobre el mateix alfabet (a l'igual que el conjunt potència de qualsevol conjunt infinit) és no numerable.

1.3 La jerarquia de Chomsky

Es pot distingir entre quatre tipus de gramàtiques en funció de la forma de les produccions.

tipus 0 o també *gramàtiques estructurades per frases* o *gramàtiques sense restriccions.* Les produccions no tenen cap tipus de restricció.

tipus 1 o també *gramàtiques sensibles al context* o *gramàtiques contextuals.* Les produccions són necessàriament de la forma

$$xAy \to x\beta y \qquad o \qquad S \to \varepsilon$$

on x i y són cadenes de $V^\star$, β és una cadena de V^+ i A és qualsevol símbol no terminal. El prefix x i el sufix y de la part esquerra de cada producció rep el nom de **context**.

tipus 2 o també *gramàtiques de context lliure*, *gramàtiques independents del context* o *gramàtiques incontextuals*. Les produccions han de ser de la forma

$$A \to \alpha$$

on A es qualsevol no terminal i α és qualsevol cadena de $V^\star$.

tipus 3 o també *gramàtiques regulars*. Les produccions han de ser de la forma

$$A \to aB \qquad o \qquad A \to a$$

on A i B són símbols no terminals i a pot ser qualsevol cadena de terminals. Aquestes gramàtiques s'haurien d'anomenar pròpiament *gramàtiques regulars per la dreta*. Es poden definir de la mateixa manera les *gramàtiques regulars per l'esquerra* intercanviant l'ordre de a i A en la definició anterior. Es pot demostrar que ambdós tipus de gramàtiques són equivalents en el sentit que per a tota gramàtica d'un tipus n'existeix una de l'altre que és equivalent. En aquest manual parlarem únicament de gramàtiques regulars i ens referirem sempre a les regulars per la dreta a no ser que s'especifique el contrari.

Els quatre tipus de gramàtiques que s'ha introduït indueix quatre famílies de llenguatges segons el tipus de gramàtiques amb que es puguen generar.

> Es diu que un llenguatge es de tipus N $(N = 0, 1, 2, 3)$ si existeix alguna gramàtica de tipus N que el genera.

Com a conseqüencia d'aquesta definició s'obtenen quatre classes de llenguatges incloses unes dins d'altres com es mostra a la figura 1.1. Escriurem la classe formada pels llenguatges de tipus N com a $\mathcal{L}_N$. Les classes de llenguatges de tipus 1, 2 i 3 les escriurem també com a $\mathcal{L}_R$, $\mathcal{L}_I$ i $\mathcal{L}_C$, respectivament.

1.4 Operacions amb gramàtiques

Les produccions de les gramàtiques es poden manipular de diverses maneres de forma que el llenguatge generat no canvia.

1.4.1 Substitució o desplegament

Qualsevol subcadena w present en alguna part dreta de producció es pot sustituir per allò en què la cadena w es pot reescriure.

És a dir, si es tenen les següents produccions associades a w

$$w \to u_1 | u_2 | \ldots | u_k$$

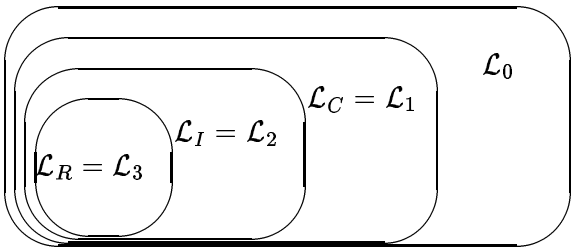

Figura 1.1. Relació entre les diferents classes de llenguatges de la jerarquia de Chomsky.

aleshores la producció $x \to ywz$ es pot substituir per

$$x \to yu_1z|yu_2z|\ldots|yu_kz$$

Si la gramàtica és incontextual i es sustitueix w en *totes* les seues aparicions en parts dretes de produccions, es poden eliminar de la gramàtica les produccions associades a w sense que es modifique el llenguatge generat.

Una conseqüència d'açò és que sempre es poden eliminar les produccions nul·les d'una gramàtica incontextual (llevat de la producció $S \to \varepsilon$).

1.4.2 Modificació de bucles

Un bucle simple en una gramàtica incontextual consisteix en dos conjunts de produccions associades a un mateix símbol. En el primer conjunt apareix aquest símbol i en el segon conjunt no. Si aquest símbol apareix en la part dreta en la posició més a l'esquerra es parla de bucle a esquerres

$$A \to Ax_1|\ldots|Ax_k|y_1|\ldots|y_p$$

i si apareix en la possició més a la dreta es parla de bucle a dretes.

$$A \to x_1A|\ldots|x_kA|y_1|\ldots|y_p$$

Sempre es pot sustituir un tipus de recursivitat per l'altre sense que el llenguatge es modifique. Per exemple, la recursivitat a dretes anterior es pot substituir per les següents produccions:

$$A \to By_1|\ldots|By_p$$

$$B \to Bx_1|\ldots|Bx_k|\varepsilon$$

La conversió en sentit contrari es faria de la mateixa manera.

1.5 Acceptació de llenguatges i computabilitat

De la mateixa manera que s'ha estudiat l'aspecte generatiu dels llenguatges es pot plantejar el problema contrari. És a dir, donada una cadena x qualsevol, i una descripció (finita) d'un llenguatge pertany la cadena x a aquest llenguatge?

Aquesta pregunta es pot plantejar per a descripcions de llenguatges mijantçant gramàtiques, però el problema no és gens trivial inclús per a llenguatges relativament senzills.

Per això es convenient definir el que s'anomenen **acceptors** o **autòmats** que es poden veure com dispositius lògics que processen cadenes i decideixen sobre aquestes (les accepten o no). Açò no és més que una altra forma de descriure llenguatges. Donat qualsevol autòmat no trivial es pot definir el llenguatge associat a l'autòmat com el format per les cadenes que aquest accepta.

Definirem al llarg d'aquest manual una família d'autòmats el més general dels quals es el que es coneix com a **Màquina de Turing**. Veurem que hi ha estretes relacions entre els diferents tipus d'autòmats i les gramàtiques de la jerarquia de Chomsky. A més a més, s'estudiarà la relació entre el tipus de processament que pot fer l'autòmat més general i els límits del que es pot o no es pot computar des del punt de vista d'obtenció de solucions algorísmiques a problemes.