

Índex

1	Autòmats i gramàtiques	1
1.1	Introduction	1
1.2	Autòmats finits i llenguatges regulars	4
1.2.1	Autòmats deterministes	4
1.2.2	Autòmats indeterministes	5
1.3	Autòmats amb pila i gramàtiques incontextuals	12
1.3.1	Autòmats amb pila	12
1.3.2	Gramàtiques incontextuals	15
1.4	Teoremes d'extracció	18
1.4.1	Per a llenguatges regulars	19
1.4.2	Per a llenguatges incontextuals	20
2	Màquines de Turing	21
2.0.1	Llenguatges decidibles	24
2.0.2	Extensions de Màquines de Turing	25

Capítol 1

Autòmates i gramàtiques

1.1 Introduction

Explicar el perquè de tot això

afegir exemples

Comencem per un **alfabet**, que no és res més que un conjunt finit de símbols. L'exemple potser més evident és el de l'alfabet $\{a, b, \dots, z\}$ que estem acostumats a utilitzar, però també ho són les xifres $\{0, 1, \dots, 9\}$ i qualsevol subconjunt d'aquest, com el de l'alfabet binari $\{0, 1\}$, que utilitzarem sovint en aquest text. De fet, els símbols d'un alfabet podrien ser qualsevol cosa, tot i que normalment utilitzem només caràcters imprimibles.

Una **paraula** sobre un alfabet Σ és una seqüència finita de símbols de Σ . Efectivament la paraula pot ser buida, que normalment escriurem com a λ (i per tant no utilitzarem λ com a símbol de l'alfabet). El conjunt de totes les possibles paraules sobre un alfabet Σ , incloent-hi la paraula buida, es denota per Σ^* . La llargada d'una paraula $x \in \Sigma^*$, representada per $|x|$, ve donada pel nombre de símbols que conté contant repeticions. Dues paraules d'un alfabet es poden combinar per formar-ne una altra en la operació de **concatenació**. Siguin $x, y \in \Sigma^*$, $x = a_1 a_2 \dots a_n$ i $y = b_1 b_2 \dots b_m$, amb $a_i \in \Sigma$ i $b_j \in \Sigma$. Aleshores, diem que $x \circ y = a_1 \dots a_n b_1 \dots b_m \in \Sigma^*$. Per simplificar la notació, quan no hi hagi possible confusió pel context, utilitzarem la notació xy en lloc de $x \circ y$.

Per concatenar una paraula amb ella mateixa, utilitzarem la notació expo-

nencial, escrivint

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

Anomenem **llenguatge** a qualsevol conjunt de paraules d'un alfabet. És a dir, si Σ és un alfabet, aleshores $\forall L \subseteq \Sigma^*$, L és un llenguatge.

Exemple. 1. Per a qualsevol alfabet Σ , \emptyset , Σ , Σ^* són llenguatges. Cal tenir present que $\emptyset \neq \{\lambda\}$, el primer és un conjunt amb 0 elements i el segon és un conjunt amb la paraula buida com a únic element.

2. Prenent l'alfabet català, el conjunt de paraules vàlides en català és un llenguatge.

3. Prenent $\Sigma = \{0, 1\}$, el conjunt de cadenes amb el mateix nombre de 0 que de 1 també és un llenguatge. Notem que mentre que l'alfabet només té dos elements, el llenguatge descrit és infinit numerable

Com que un llenguatge no deixa de ser un conjunt, totes les operacions pròpies d'aquests (unió, intersecció, complementari, diferència...) són vàlides. Siguin L_1 i L_2 llenguatges sobre un alfabet Σ . Definim:

- **Unió:** $L_1 \cup L_2 = \{x \mid x \in L_1 \vee x \in L_2\}$
- **Intersecció:** $L_1 \cap L_2 = \{x \mid x \in L_1 \wedge x \in L_2\}$
- **Complementari:** $\bar{L}_1 = \{x \in \Sigma^* \mid x \notin L_1\}$
- **Diferència:** $L_1 \setminus L_2 = \{x \in L_1 \mid x \notin L_2\}$

A aquestes n'hi afegim les següents operacions:

- **Concatenació:** $L_1 \circ L_2 = \{x \circ y \mid x \in L_1 \wedge y \in L_2\}$, és a dir, el llenguatge de paraules formades a partir de la concatenació d'una paraula d' L_1 i una d' L_2 . De manera anàloga a la concatenació de paraules, utilitzarem la notació exponencial per indicar la concatenació d'un llenguatge amb sí mateix:

$$L^i = \begin{cases} \{\lambda\} & \text{si } i = 0 \\ L^{i-1} \circ L & \text{si } i \geq 1 \end{cases}$$

- **Clausura (de Kleene):** $L^* = \{x_1 \circ x_2 \circ \dots \circ x_n \mid n \geq 0 \wedge x_1, \dots, x_n \in L\}$, és a dir, el conjunt de totes les possibles concatenacions de paraules d' L o, de manera equivalent, el conjunt de totes les paraules finites formades a partir de símbols de Σ .

A les operacions de **unió**, **concatenació** i **clausura** les anomenem **operacions regulars**.

explicar precedència dels operadors

Si un llenguatge és finit, el podem definir totalment donant tots els seus elements. Normalment, però, ens interessaran llenguatges infinits, que els definirem mitjançant generadors que els descriuen. El següent exemple defineix, sobre l'alfabet binari, el llenguatge infinit de paraules formades per n zeros seguits d' n uns: $\{0^n 1^n \mid n \geq 1\}$

Alguns d'aquests llenguatges es poden definir amb el que anomenarem **expressions regulars**, que són expressions que es poden escriure utilitzant només els símbols de l'alfabet i les operacions regulars.

Definició 1.

Podem definir les expressions regulars sobre un alfabet Σ recursivament de la següent manera:

- \emptyset , denotant el conjunt buit, és una expressió regular
- λ , denotant el conjunt format únicament per la paraula, buida és una expressió regular
- $\forall a \in \Sigma$, a és una expressió regular

Aleshores, si α i β són expressions regulars, $(\alpha \cup \beta)$, $(\alpha \circ \beta)$ i (α^) també són expressions regulars.*

Aleshores, si α és una expressió regular, denotem per $L(\alpha)$ el llenguatge generat per α mitjançant les següents regles:

- $L(\emptyset) = \emptyset$ i $L(\lambda) = \{\lambda\}$
- Si $a \in \Sigma$, aleshores $L(a) = \{a\}$
- $L(\alpha \cup \beta) = L(\alpha) \cup L(\beta)$
- $L(\alpha \circ \beta) = L(\alpha) \circ L(\beta)$
- $L(\alpha^*) = L(\alpha)^*$

Direm que un llenguatge L és **regular** si existeix una expressió regular α tal que $L = L(\alpha)$.

afegir exemples!

Més
ex-
em-
ples?

pensar
si
posar
nu-
mero
a les
defini-
cions

1.2 Autòmats finits i llenguatges regulars

connectar els apartats

Però no havíem de parlar de computació? Per què ens interessen els llenguatges? Doncs bé, el primer model de computació que tractarem són els **autòmats finits** (també anomenats deterministes), que tenen molt a veure amb els llenguatges regulars.

1.2.1 Autòmats deterministes

Els autòmats finits són un model computacional on es té una màquina amb un conjunt d'estats i que rep com a entrada una paraula sobre un cert alfabet Σ . A cada pas de càlcul es llegeix, de manera seqüencial, un únic caràcter de la paraula, es calcula l'estat següent en funció del caràcter llegit i l'estat actual i es passa a la següent cel·la, repetint el procés fins haver llegit tota la paraula. Aquest model computacional no dóna cap sortida, només direm que l'autòmat **accepta** o **rebutja** la paraula segons l'estat en el que hagi quedat en llegir i computar l'últim caràcter.

Definició 2. *Un autòmat determinista és una estructura $M = (K, \Sigma, \delta, q_0, F)$ on:*

- K és un conjunt finit i no buit d'estats
- Σ és l'alfabet d'entrada
- $\delta: K \times \Sigma \rightarrow K$ la funció de translació, que donada el caràcter d'entrada i estat actuals determina el següent estat
- $q_0 \in K$ l'estat inicial
- $F \subseteq K$ el conjunt d'estats finals

afegir un exemple

Sovint representarem M mitjançant un graf on:

- Els nodes són els estats
- Es marca l'estat inicial amb una fletxa incident al node
- Es marquen els estats finals amb un doble cercle o una creu

- Si $\delta(q, a) = p$, es fa una aresta dirigida que va des del node representant q fins al node representant p mitjançant una aresta amb etiqueta a .

Representar l'exemple anterior amb un graf

Si en un pas de còmput l'autòmat es troba en l'estat $p \in K$ i la paraula que queda per llegir és $x \in \Sigma^*$, diem que $px \in K\Sigma^*$ és una **configuració de M**. A més, si px i qy , amb $x = a_1 \dots a_n$, són configuracions d' M , direm que px **produeix qy en un pas** si $\delta(p, x) = q$ i $y = a_1 \dots a_{n-1}$, i ho escriurem $px \vdash_M qy$. Si es pot passar d'una configuració px a una configuració qy amb un nombre finit de passos de càlcul, direm que px **produeix qy** i ho escriurem $px \vdash_M^* qy$.

Diem que una paraula $x \in \Sigma^*$ és **reconeguda** per un autòmat M si existeix un estat $q \in K$ tal que $q_0x \vdash_M^* q$. Aleshores, definim el **llenguatge reconegut per M** com

$$L(M) = \{x \in \Sigma^* \mid x \text{ és reconeguda per } M\}$$

Teorema. *Per a tot llenguatge L existeix una expressió regular α tal que $L = L(\alpha)$ si i només si existeix un autòmat determinista M tal que $L(M) = L$.*

Cal fer la demo? Per \Leftarrow hem de veure que la classe dels llenguatges acceptats per autòmats finits és tancada per les operacions regulars, per \Rightarrow

Aquest teorema no el demostrarem directament, sinó que ho farem posteriorment a través de l'equivalència entre autòmats deterministes i no deterministes, que veurem a continuació.

1.2.2 Autòmats indeterministes

Definició 3. *Un autòmat indeterminista és una estructura $M = (K, \Sigma, \Delta, q_0, F)$ on:*

- K és un conjunt finit i no buit d'estats
- Σ és l'alfabet d'entrada
- $\Delta: K \times (\Sigma \cup \{\lambda\}) \rightarrow \mathcal{P}(K)$ la funció de transició, on $\mathcal{P}(K)$ denota el conjunt de les parts de K (i.e. el conjunt de tots els subconjunts de K).
- $q_0 \in K$ l'estat inicial

Notació correcta o fer-ho conjunt?

- $F \subseteq K$ el conjunt d'estats finals

Notem que la definició és igual que pels autòmats deterministes excepte per la funció de transició. En els autòmats deterministes es fa exactament una transició per a cada símbol d'entrada que es llegeix, mentre que els indeterministes permeten fer zero, una o més transicions per cada símbol llegit. A més, es permeten transicions amb la paraula buida, passant d'un estat a un altre sense llegir cap símbol d'entrada. Per a aquest motiu el conjunt d'arribada de la funció de transició Δ és $\mathcal{P}(K)$, ja que per un estat i un símbol d'entrada hi poden haver múltiples estats d'arribada.

Les definicions de **configuració**, **produeix en un pas de còmput** i **produeix, reconeix i llenguatge reconegut** són anàlogues a les dels autòmats deterministes.

Necessitem, però, el concepte de **clausura d'un estat**, que no tenim en el cas d'autòmats deterministes.

Definició 4. *Sigui $M = (K, \Sigma, \Delta, q_0, F)$ un autòmat indeterminista i sigui $p \in K$ un estat d' M . Definim la clausura de p com*

$$\Lambda(p) = \{q \in K \mid p\lambda \vdash_M^* q\}$$

és a dir, el conjunt d'estats als que es pot arribar des de l'estat p mitjançant una o més transicions λ , incloent-hi el propi estat p .

Malgrat que sembli que amb les transicions λ i permetent fer diferents nombres de transicions aquest model és més potent que els autòmats deterministes, veurem que de fet aquests dos models són equivalents.

Teorema. *Per a tot llenguatge L existeix un autòmat determinista M_D tal que $L = L(M_D)$ si i només si existeix un autòmat indeterminista M_I tal que $L(M_I) = L$.*

Demostració.

- $\boxed{\implies}$: Sigui $M_D = (K, \Sigma, \delta, q_0, F)$. Prenent $M_I = (K, \Sigma, \Delta, q_0, F)$ amb

$$\begin{aligned} \Delta: K \times \Sigma &\longrightarrow \mathcal{P}(K) \\ qx &\longmapsto \{\delta(q, x)\} \end{aligned}$$

Fent la identificació $\{\delta(q, x)\} \mapsto \delta(q, x)$ tenim que les funcions de transició Δ i δ són equivalents, tal com la resta de components de M_I i M_D , de manera que els autòmats són idèntics i per tant reconeixen el mateix llenguatge.

- $\boxed{\Leftarrow}$: La idea d'aquesta demostració és pensar que l'autòmat indeterminista M_I , en lloc d'estar en un únic estat després de llegir una certa entrada, està alhora en tot el conjunt d'estats als que es pot arribar havent llegit l'entrada. Amb aquesta idea es pot generar un autòmat determinista els estats del qual siguin tots els possibles conjunts d'estats de l'indeterminista, i modelar δ en funció del conjunt d'estats als que es pugui arribar llegint un símbol.

Sigui $M_I = (K, \Sigma, \Delta, q_0, F)$ un autòmat indeterminista. Construïrem un autòmat determinista $M_D = (K', \Sigma, \delta, q'_0, F')$ on:

- ★ $K' = \mathcal{P}(K)$. Cada estat de M_D és un conjunt d'estats de M_I i el representarem per a aquest conjunt.
- ★ Si $X \in K'$ i $a \in \Sigma$, definim

$$\begin{aligned}\delta(X, a) &= \bigcup \{ \Lambda(q) \mid \exists p \in X \text{ tal que } q \in \Delta(p, a) \} \\ &= \{ q \in K \mid q \in \Lambda(\Delta(r, a)) \text{ per algun } r \in X \}\end{aligned}$$

Observem que això està ben definit ja que el conjunt d'arribada de δ és K' , que és precisament $\mathcal{P}(K)$. Notem, també, que és possible que $\delta(X, a) = \emptyset$. Això, però, no suposa cap problema ni que δ no estigui ben definida, ja que $\emptyset \in K' = \mathcal{P}(K)$. Aquest conjunt es pot obtenir de la següent manera:

1. Prendre tots els estats $q \in K$ pels quals existeix $p \in X$ tal que $q \in \Delta(p, a)$
2. Calcular $\Lambda(q)$ per a tots els estats q obtinguts en el pas (1)
3. Prendre la unió de tots els conjunts $\Lambda(q)$ del pas (2)

- ★ $q'_0 = \Lambda(q_0)$
- ★ $F' = \{X \in K' \mid X \cap F \neq \emptyset\}$

Ara ens cal veure que, efectivament, M_D és un autòmat determinista i que és equivalent a M_I . Que M_D és determinista és evident per construcció i amb les observacions sobre δ ja fetes. Per veure que els autòmats són equivalents, ens cal demostrar que per a qualsevol paraula $w \in \Sigma^*$ i per a qualssevol estats $p, q \in K$ tenim que

$$qw \vdash_{M_I}^* p\lambda \iff \Lambda(q)w \vdash_{M_D}^* P\lambda \quad (1.1)$$

per a algun $P \in \mathcal{P}(K)$ tal que $p \in P$.

Demostrant aquesta proposició, el teorema se segueix: per veure que M_D i M_I són equivalents cal veure que els llenguatges que reconeixen són

Quina de les notacions?

equivalents. Prenent $w \in \Sigma^*$, tenim que $w \in L(M_I) \xLeftrightarrow[(def)] q_0 w \vdash_{M_I}^* f\lambda$ per a algun $f \in F \xLeftrightarrow[(1.1)] \Lambda(q_0)w \vdash_{M_D}^* P\lambda$ per a algun $P \in K'$ que, per definició, és $w \in L(M_I)$.

Per demostrar la proposició utilitzarem inducció sobre $|w|$:

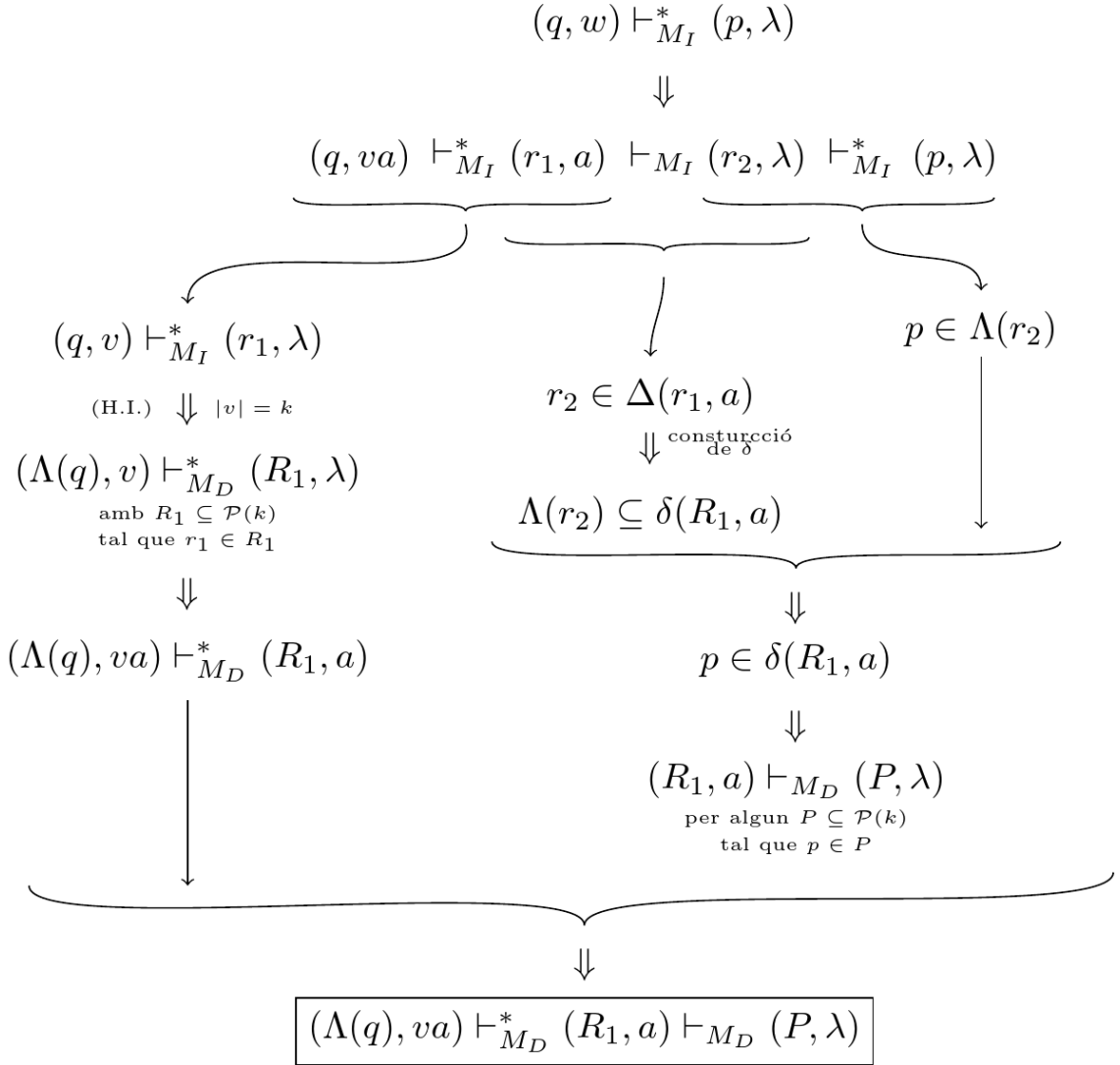
- ★ $|w| = 0$: O, de manera equivalent, $w = \lambda$. Cal veure doncs que $q\lambda \vdash_{M_I}^* p\lambda \iff \Lambda(q)\lambda \vdash_{M_D}^* P\lambda$ per a algun $P \in \mathcal{P}(K)$ tal que $p \in P$. Ara bé, tenim que $q\lambda \vdash_{M_I}^* p\lambda \iff p \in \Lambda(q)$. D'altra banda, com que M_D és un autòmat determinista, $\Lambda(q)\lambda \vdash_{M_D}^* P\lambda \iff P = \Lambda(q)$ per a algun P tal que $q \in P \iff p \in \Lambda(q)$.
- ★ $|w| = k + 1$: Suposem que la proposició és certa per a paraules de llargada com a màxim k per a alguna $k \geq 0$, i vegem que és certa per a paraules de longitud $k + 1$. Sigui $w = va$, amb $v \in \Sigma^*$ tal que $|v| = k$, i sigui $a \in \Sigma$.
 \implies Suposem $qv \vdash_{M_I}^* p\lambda$. Aleshores existeixen estats r_1 i r_2 tals que
 \iff Suposem que $\Lambda(q)va \vdash_{M_D}^* R_1a \vdash_{M_D} P\lambda$ per a algun P tal que $p \in P$ i algun R_1 tal que $\delta(R_1, a) = P$. Per definició, $\delta(R_1, a) = \cup\{\Lambda(r_2) \mid \exists r_1 \in R_1 \text{ tal que } r_2 \in \Delta(r_1, a)\}$. Com que $p \in P = \delta(R_1, a)$, existeix algun r_2 tal que $p \in \Lambda(r_2)$, i per a algun $r_1 \in R_1$ es té que $r_2 = \Delta(r_1, a)$. Aleshores tenim que $r_2\lambda \vdash_{M_I}^* p\lambda$. Per hipòtesi d'inducció, $qv \vdash_{M_I}^* r_1\lambda$ i, per tant $qva \vdash_{M_I}^* r_1a \vdash_{M_I}^* r_2\lambda \vdash_{M_I}^* p\lambda$, que és el que volíem veure.

□

Així doncs, hem demostrat que els autòmats deterministes i indeterministes són equivalents i ho hem fet amb una demostració constructiva, de manera que també tenim un procediment per passar d'un a l'altre. D'ara en endavant, quan ens referim a **autòmats finits** ens estarem referint indistintament a deterministes o indeterministes.

Ara només cal veure que els autòmats finits són equivalents a les expressions regulars. Per fer-ho abans necessitem veure el següent resultat:

Proposició. *La classe dels llenguatges reconeguts per autòmats finits és tancada respecte les operacions d'unió, concatenació i clausura de Kleene. En altres paraules, si tenim dos autòmats finits M_1 i M_2 , aleshores existeixen autòmats M_\cup , M_\circ i M_* tals que $L(M_\cup) = L(M_1) \cup L(M_2)$, $L(M_\circ) = L(M_1) \circ L(M_2)$ i $L(M_*) = L(M_1)^*$.*



Demostració. Siguin $M_1 = (K_1, \Sigma, \Delta_1, q_1, F_1)$ i $M_2 = (K_2, \Sigma, \Delta_2, q_2, F_2)$ dos autòmats indeterministes qualssevol.

- **Unió:** Volem construir un autòmat indeterminista M tal que $L(M) = L(M_1) \cup L(M_2)$. Per fer-ho, mantindrem els dos autòmats originals en paral·lel, afegint un estat inicial amb una transició λ cap a cadascun dels estats inicials de M_1 i M_2 , fent que al primer pas de càlcul es triï si la paraula és de $L(M_1)$ o $L(M_2)$. Els estats acceptadors d' M seran la unió dels són els d' M_1 i M_2 , permetent que es reconeguin les paraules tant d'un llenguatge com de l'altre. Definim $M = (K, \Sigma, \Delta, s, F)$ on:

$$\star K = K_1 \cup K_2 \cup \{s\}$$

\star

$$\Delta(p, a) = \begin{cases} \Delta_1(p, a) & \text{si } p \in K_1 \\ \Delta_2(p, a) & \text{si } p \in K_2 \\ \{q_1, q_2\} & \text{si } p = s \text{ i } a = \lambda \\ \emptyset & \text{si } p = s \text{ i } a \neq \lambda \end{cases}$$

$$\star F = F_1 \cup F_2$$

- **Concatenació:** Volem construir un autòmat indeterminista M tal que $L(M) = L(M_1) \circ L(M_2)$. Per fer-ho posarem M_1 i M_2 seguits, afegint una transició λ des de cadascun dels estats acceptadors d' F_1 cap a l'estat inicial d' M_2 q_2 . Així, en acabar de reconèixer una paraula d' $L(M_1)$ es pot continuar reconeixent una paraula d' $L(M_2)$. Formalment, $M = (K_1 \cup K_2, \Sigma, \Delta, q_1, F_2)$, amb

$$\Delta(p, a) = \begin{cases} \Delta_1(p, a) & \text{si } p \in K_1 \\ \Delta_2(p, a) & \text{si } p \in K_2 \\ \Delta_1(p, a) \cup \{q_2\} & \text{si } p \in F_1 \text{ i } a = \lambda \end{cases}$$

- **Clausura de Kleene:** Volem construir un autòmat indeterminista M tal que $L(M) = L(M_1)^*$. Per fer-ho, afegirem una transició λ des dels estats d' F_1 cap a l'estat q_1 , de manera que en acabar de reconèixer una paraula pugui tornar a començar el procés de reconèixer-ne una altra. A més, per permetre acceptar la paraula buida λ , afegirem un estat que serà acceptador i inicial amb una transició λ a q_1 . Així, tindrem $M = (K \cup \{s\}, \Sigma, \Delta, s, F \cup \{s\})$ amb

$$\Delta(p, a) = \begin{cases} \{q_1\} & \text{si } p = s \text{ i } a = \lambda \\ \emptyset & \text{si } p = s \text{ i } a \neq \lambda \\ \Delta_1(p, a) \cup \{q_1\} & \text{si } p \in F \text{ i } a = \lambda \\ \Delta_1(p, a) & \text{en cas contrari} \end{cases}$$

□

Observació. *També es pot demostrar que és tancada per les operacions de complement i intersecció, però aquestes no les farem servir.*

Amb el resultat anterior, ja podem enunciar i demostrar l'equivalència entre autòmats fits i expressions regulars:

Teorema. *Un llenguatge és regular si i només si és acceptat per un autòmat finit.*

Demostració. Sigui α una expressió regular. Volem generar un autòmat indeterminista M tal que $L(M) = L(\alpha)$. Separem la demostració en diferents casos base:

- $\boxed{\alpha = \emptyset}$: Aleshores $L(\alpha) = \emptyset$ i l'autòmat indeterminista $M = (\{q\}, \Sigma, \Delta, q, \emptyset)$ amb $\Delta(p, a) = \emptyset \forall p, a$ reconeix $L(\alpha)$.
- $\boxed{\alpha = \lambda}$: En aquest cas $L(\alpha) = \{\lambda\}$ i l'autòmat indeterminista $M = (\{q\}, \Sigma, \Delta, q, \{q\})$ amb $\Delta(p, a) = \emptyset \forall p, a$ reconeix $L(\alpha)$.
- $\boxed{\alpha = x}$ amb $x \in \Sigma$: En aquest cas $L(\alpha) = \{x\}$ i l'autòmat indeterminista $M = (\{q_1, q_2\}, \Sigma, \Delta, q_1, \{q_2\})$ amb $\Delta(q_1, x) = \{q_2\}$ i $\Delta(p, a) = \emptyset \forall p \neq q_1, \forall a \neq x$ reconeix $L(\alpha)$.

Aleshores, com que hem demostrat que la classe dels llenguatges reconeguts per autòmats és tancada per unió, concatenació i clausura de Kleene i es poden generar qualsevol expressió regular a partir de les tres que hem definit i aquestes tres operacions, podem construir qualsevol autòmat indeterminista d'una expressió regular fent servir els mateixos procediments per construir autòmats que hem fet servir abans.

Sigui ara $M = (K, \Sigma, \Delta, q_0, F)$ un autòmat indeterminista i volem construir una expressió regular α tal que $L(\alpha) = L(M)$. Assignem als estats de K un enter de manera creixent quedant $K = \{q_1, \dots, q_n\}$, amb $q_1 = s$ i per a cada q_k anomenem a $k \in \mathbb{N}$ com l'**índex** de l'estat. Definim per $R(i, j, k)$, amb $i, j = 1 \dots n$ i $k = 0 \dots n$ el conjunt de totes les paraules de Σ^* que poden portar a M des de l'estat q_i fins l'estat q_j passant només pels estats amb índex no superior a k (és a dir, q_1, \dots, q_k). Observem que si $k = n$, aleshores tenim

$$R(i, j, n) = \{w \in \Sigma^* \mid (q_i, w) \vdash_M^* (q_j, \lambda)\}$$

Així, tenim que

$$L(M) = \bigcup_{j=1 \dots n} R(1, j, n)$$

Per tant, si veiem que els $R(i, j, k)$ són expressions regulars, ja haurem demostrat el que volíem. Provem-ho per inducció:

- $\boxed{k = 0}$:

$$R(i, j, 0) = \{a \in \Sigma \cup \{\lambda\} \mid q_j \in \Lambda(q_i, a)\} \text{ si } i \neq j$$

$$R(i, i, 0) = \{\lambda\} \cup \{a \in \Sigma \cup \{\lambda\} \mid q_i \in \Lambda(q_i, a)\}$$

és a dir, l'expressió regular formada per la unió de símbols de les arestes que van des de l'estat q_i a q_j , prenent λ si no hi ha cap aresta.

- $\boxed{k-1 \implies k}$: Suposem ara que $R(i, j, k-1)$ és una expressió regular per a tot i, j , i anem a veure que $R(i, j, k)$ també ho és. Podem escriure

$$R(i, j, k) = R(i, j, k-1) \cup R(i, k, k-1)R(k, k, k-1)^*R(k, j, k-1)$$

La interpretació de la fórmula anterior és que, per passar de l'estat q_i a l'estat q_j sense passar per cap estat amb índex major que k es pot

- ★ O bé anar des de q_i a q_j sense passar per cap estat amb índex major que $k-1$
- ★ O bé anar des de q_i a q_k , seguidament anar de q_k a q_k zero o més vegades i després anar de q_k a q_j , en els tres casos sense passar per cap estat amb índex superior a $k-1$.

Per hipòtesi d'inducció, $R(i, j, k-1)$, $R(i, k, k-1)$, $R(k, k, k-1)$ i $R(k, j, k-1)$ són regulars, de manera que la seva clausura de Kleene, concatenació i unió també ho són. Amb això obtenim que $R(i, j, k)$ és també regular.

□

Amb aquesta demostració hem vist que els autòmats finits (tant deterministes com indeterministes) són equivalents a les expressions regulars.

1.3 Autòmats amb pila i gramàtiques incontextuals

Pensar si introduir primer els autòmats amb pila a partir dels autòmats finits o començar tal com ho hem fet abans amb les gramàtiques incontextuals

1.3.1 Autòmats amb pila

Els models computacionals que hem vist fins ara ens poden semblar, en certa manera, limitats. Són estructures que ens permeten detectar si una paraula

forma part o no d'un llenguatge, però tenen una memòria molt limitada i això els impedeix manipular dades. En aquest apartat prenem les idees essencials dels autòmats finits i les expandim per permetre la manipulació de dades. Per això caldrà definir un nou tipus d'autòmats, que anomenarem autòmats amb pila.

Els autòmats amb pila són una extensió dels autòmats finits que introdueixen una nova dimensió en la capacitat de càlcul. A diferència dels autòmats finits, aquest nou tipus d'autòmats tenen associada una pila, una estructura de dades seqüencial que permet afegir, treure o llegir elements del cim de la pila¹. Aquesta nova capacitat ens permetrà reconèixer llenguatges que no podíem abordar amb els autòmats finits, ja que podem utilitzar la pila per recordar informació i realitzar càlculs més complexos.

Definició 5. *Un autòmat amb pila és una estructura $M = (K, \Sigma, \Gamma, \Delta, q_0, F)$ on:*

- K és un conjunt finit i no buit d'estats
- Σ és un alfabet finit, que anomenarem **alfabet d'entrada**
- Γ és un altre alfabet finit, que anomenarem **alfabet de la pila**
- $q_0 \in K$ és l'estat inicial
- $F \subseteq K$ és el conjunt d'estats acceptadors o finals
- Δ és un subconjunt de $(K \times (\Sigma \cup \{\lambda\})) \times (\Gamma \cup \{\lambda\}) \times (K \times \Gamma^*)$, els elements del qual anomenarem **transicions**

En començar un còmput en l'autòmat M , estarem a l'estat inicial q_0 , tindrem una paraula $w \in \Sigma^*$ i tindrem la pila buida (o, de manera equivalent, hi haurà λ al cim de la pila). En un pas de còmput es pot aplicar la transició $((p, a, b), (q, x))$ si $p \in K$ és l'estat actual, $a \in \Sigma \cup \{\lambda\}$ és el símbol de la cinta que toca llegir i $b \in \Gamma \cup \{\lambda\}$ és al cim de la pila. Aleshores, l'autòmat passa a l'estat q i se substitueix el símbol b per x al cim de la pila.

Definició 6. *Si $M = (K, \Sigma, \Gamma, \Delta, q_0, F)$ és un autòmat amb pila, definim una **configuració** de M com una paraula $\alpha p x \in \Gamma^* K \Sigma^*$. Si en un pas de còmput la configuració d' M és $\alpha p x$, l'autòmat es troba a l'estat p , α és el contingut de la pila i x és la part de la paraula d'entrada que encara no s'ha llegit.*

¹Es pot pensar literalment com una torre de dades: es pot accedir o treure únicament l'element de dalt de tot de la pila i només es pot afegir un element a dalt de tot. Així, l'últim element que es posa a la pila ha de ser el primer a sortir-ne

De manera molt similar als autòmats finits, definim també els següents termes:

- Si $\alpha p x$ i $\beta q y$ són configuracions d'un autòmat M , diem que $\alpha p x$ **produeix** $\beta q y$ **en un pas de còmput** si es pot passar de la primera configuració a la segona aplicant una transició de Δ . Seguint la nomenclatura d'autòmats finits, ho representarem amb $\alpha p x \vdash_M \beta q y$.
- De manera més general, direm que $\alpha p x$ **produeix** $\beta q y$ si es pot passar de la primera configuració a la segona mitjançant un nombre finit de transicions de Δ . Ho representarem per $\alpha p x \vdash_M^* \beta q y$.
- Una paraula $x \in \Sigma^*$ **és reconeguda o acceptada** per l'autòmat M si $\exists q \in F$ tal que $\lambda p x \vdash_M^* \lambda q \lambda$. És a dir, una paraula es reconeix només si en llegir tota la paraula, s'arriba a un estat acceptador amb la pila buida.
- Definim el **llenguatge associat a M** com

$$L(M) = \{x \in \Sigma^* \mid x \text{ és reconeguda per } M\}$$

Afegir exemple d'autòmat amb pila

Tal com en els autòmats finits, amb els autòmats amb pila també podem fer la distinció entre autòmats amb pila deterministes i indeterministes. Per definir-los correctament, però, necessitem el concepte de transicions compatibles:

Definició 7. Sigui $M = (K, \Sigma, \Gamma, \Delta, q_0, F)$ un autòmat amb pila. Diem que dues transicions $((p_1, a_1, b_1), (q_1, \alpha_1))$ i $((p_2, a_2, b_2), (q_2, \alpha_2))$ de Δ són **compatibles** si es compleixen les següents condicions:

1. $p_1 = p_2$
2. $a_1 = a_2$ o bé $a_1 = \lambda$ o bé $a_2 = \lambda$
3. $b_1 = b_2$ o bé $b_1 = \lambda$ o bé $b_2 = \lambda$

En altres paraules, diem que dues transicions són compatibles si es poden aplicar en el mateix pas de còmput.

Definició 8. Direm que un autòmat amb pila és **determinista** si no té dues transicions diferents que siguin compatibles.

1.3.2 Gramàtiques incontextuals

Pensar com enllaçar els dos apartats (el paràgraf de sota és només provisional)

Les expressions regulars ens permetien veure si una paraula pertanyia al llenguatge que codificaven, però no ens deia res sobre la sintaxi i estructura del llenguatge. D'altra banda, una **gramàtica incontextual** actua com a generador de llenguatge, definint les regles que fan que un llenguatge sigui vàlid.

Definició 9. Una **gramàtica incontextual** és una estructura $G = (V, \Sigma, P, S)$ on:

- V és un alfabet, els símbols del qual anomenarem **variables**
- Σ és un alfabet de símbols disjunt de V , els elements del qual anomenarem **terminals**
- $P \subseteq V \times (V \cup \Sigma)^*$ els elements del qual anomenarem **produccions**
- $S \in V$ és la variable inicial

Per a $A \in V$ i $x \in (V \cup \Sigma)^*$, si $(A, x) \in P$, escriurem $A \longrightarrow x$. En una paraula $w \in (V \cup \Sigma)^*$ que contingui A , aplicar una producció $A \longrightarrow x$ consisteix en canviar una de les aparicions d' A en w per x . Per simplificar la notació, si a tenim les produccions $(A, x_1), \dots, (A, x_n) \in P$, escriurem $A \longrightarrow x_1 | \dots | x_n$.

Afegir exemple de producció

Definició 10. Siguí $G = (V, \Sigma, P, S)$ una gramàtica incontextual. Si $u, v \in (V \cup \Sigma)^*$, direm que v **es deriva (o és una derivació) de u en un pas** i escriurem $v \Rightarrow_G u$ si podem obtenir v aplicant una producció de P sobre u . Direm que v **es deriva (o és una derivació) d' u** i ho escriurem $v \Rightarrow_G^* u$ si es pot obtenir v aplicant un nombre finit de produccions de P sobre u . Direm que v **es deriva d' u per l'esquerra en un pas** si v es pot derivar d' u en un pas substituint la primera variable d' u (i.e. la variable situada més a l'esquerra de la paraula u). Respectivament, direm que v **es deriva d' u per la dreta en un pas** si v es pot derivar d' u en un pas substituint la última variable d' u (i.e. la variable situada més a la dreta d' u). Escriurem v es deriva d' u per l'esquerra en un pas (respectivament dreta) com $u \xRightarrow{L} v$ (respectivament $u \xRightarrow{R} v$). Direm també que v es deriva d' u per l'esquerra (resp. dreta) si

existeixen paraules $u_1, \dots, u_n \in V^*$ tals que $u \xRightarrow{L} u_1 \xRightarrow{L} \dots \xRightarrow{L} u_n \xRightarrow{L} v$ (resp. amb \xRightarrow{R}), i ho escriurem $u \xRightarrow{L}^* v$ (resp. $u \xRightarrow{R}^* v$).

Definició 11. Sigui $G = (V, \Sigma, P, S)$ una gramàtica incontextual. Definim el llenguatge generat per G com

$$L(G) = \{x \in \Sigma^* \mid S \Rightarrow_G^* x\}$$

Direm que un llenguatge L és **incontextual** si existeix una gramàtica incontextual G tal que $L = L(G)$.

Teorema. Per a qualsevol llenguatge L , existeix un autòmat amb pila M tal que $L(M) = L$ si i només si existeix una gramàtica incontextual G tal que $L(G) = L$

Demostració. Demostrarem primer que tot llenguatge incontextual és reconegut per un autòmat amb pila. Per fer-ho, prenem una gramàtica incontextual $G = (V, \Sigma, P, S)$ i construirem un autòmat amb pila M tal que $L(M) = L(G)$.

Sigui $M = (\{p, q\}, \Sigma, V, \Delta, p, \{q\})$, on Δ conté les següents transicions:

1. $((p, \lambda, \lambda), (q, S))$
2. $((q, \lambda, A), (q, x))$ per a cada producció $A \rightarrow x$ de P
3. $((q, a, a), (q, \lambda))$ per a cada $a \in \Sigma$.

Així, M utilitza l'alfabet de variables V de la gramàtica incontextual G com a alfabet de la pila i l'alfabet de terminals de G com a alfabet d'entrada d' M . Observem també que a l'autòmat M només hi ha dos estats: p , que és l'estat inicial, i q l'únic estat acceptador, al que forçosament haurem de passar en el primer pas de còmput (ja que és la única transició possible amb la pila buida). A cada pas posterior hi ha dues opcions: o bé hi ha una variable al cim de la pila a la que s'aplica una producció de P , o bé treu del cim de la pila un terminal, suposant que coincideixi amb el símbol que toca llegir de la cinta. Ara cal provar, doncs, que $L(M) = L(G)$, que ho farem a partir de la següent proposició:

Arreglar la notació dels estats, que no és consistent amb la de les definicions!

Sigui $w \in \Sigma^*$ i $\alpha \in (V \setminus \Sigma)V^* \cup \{\lambda\}$. Aleshores $S \xRightarrow{L}^* w\alpha$ si i només si $(q, w, S) \vdash_M^* (q, \lambda, \alpha)$.

Provant aquesta proposició haurem demostrat que qualsevol llenguatge in-contextual és acceptat per un autòmat amb pila ja que, prenent $\alpha = \lambda$, tindrem que $S \xRightarrow{*} w \iff (q, w, S) \vdash_M^* (q, \lambda, \lambda)$. Utilitzant la primera transició de Δ descrita a dalt, tindrem que $(p, w, \lambda) \vdash_M (q, w, S) \vdash_M^* (q, \lambda, \lambda)$, és a dir que $w \in L(G) \iff w \in L(M)$.

Així doncs, procedim a demostrar les dues implicacions per inducció sobre el nombre de derivacions:

- $\boxed{\Leftarrow}$: Suposem que $S \xRightarrow{*} w\alpha$ per un cert nombre k de derivacions i volem veure que $(q, w, S) \vdash_M^* (q, \lambda, \alpha)$
 - ★ $\boxed{k = 0}$: Si la derivació té longitud $k = 0$ necessàriament $w = \lambda$ i $\alpha = S$, i aleshores $(q, w, S) \vdash_M^* (q, \lambda, \alpha)$
 - ★ $\boxed{k \implies k + 1}$: Suposem que si $S \xRightarrow{*} w\alpha$ amb una derivació de longitud $n \leq k$ aleshores $(q, w, S) \vdash_M^* (q, \lambda, \alpha)$. Sigui ara una derivació per l'esquerra amb longitud $k + 1$ de la següent manera:

$$S = u_0 \xRightarrow{L} u_1 \xRightarrow{L} \dots \xRightarrow{L} u_k \xRightarrow{L} u_{k+1} = w\alpha$$

Suposem que A és la primera variable de u_k . Aleshores, com que $u_k \xRightarrow{L} u_{k+1}$, tenim que $u_k = xAy$ i $u_{k+1} = xzy$, per a alguns $x \in \Sigma^*$, $y, z \in V^*$ i que $A \longrightarrow z$ és una producció de P . Aplicant la hipòtesi d'inducció sobre $u_k = xAy$ i prenent $\alpha = Ay$ obtenim

$$(q, x, S) \vdash_M^* (q, \lambda, Ay)$$

Com que $A \longrightarrow z$, aplicant una transició del tipus (2) tenim que $(q, \lambda, Ay) \vdash_M (q, \lambda, zy)$. Observem ara que $u_{k+1} = w\alpha$ però també $u_{k+1} = xzy$, de manera que existeix una paraula $\beta \in \Sigma^*$ tal que $w = x\beta$ i per tant $\beta\alpha = zy$. Aleshores

$$(q, \beta, zy) = (q, \beta, \beta\alpha) \vdash_M^* (q, \lambda, \alpha)$$

realitzant una seqüència de $|\beta|$ transicions de tipus (3). Amb tot això obtenim

$$(q, w, S) = (q, x\beta, S) \vdash_M^* (q, \beta, Ay) \vdash_M (q, \beta, zy) = (q, \beta, \beta\alpha) \vdash_M^* (q, \lambda, \alpha)$$

com volíem veure.

- $\boxed{\implies}$: Suposem ara que $(q, w, S) \vdash_M^* (q, \lambda, \alpha)$ i volem veure que $S \xRightarrow{*} w\alpha$. Tornarem a fer la demostració per inducció, ara sobre el nombre de transicions de tipus (2) en el còmput:

- ★ $\boxed{k = 0}$: Observem, abans de res que immediatament després de fer la transició (1) inicial, com que la pila és buida, només es pot fer una transició de tipus (2). Si no hi ha cap transició de tipus (2), aleshores $(q, w, S) \vdash_M^* (q, \lambda, \alpha) \implies w = \lambda \wedge \alpha = S$, i per tant $S = w\alpha$.
- ★ $\boxed{k \implies k + 1}$: Suposem que si $(q, w, S) \vdash_M^* (q, \lambda, \alpha)$ amb $n \leq k$ transicions de tipus (2) (possiblement amb transicions de tipus (3) abans i després) aleshores $S \xRightarrow{*} w\alpha$. Suposem ara que $(q, w, S) \vdash_M^* (q, \lambda, \alpha)$ amb $k + 1$ transicions de tipus (2) i separem les transicions a la penúltima transició de tipus (2):

$$(q, w, S) \vdash_M^* (q, \beta, Ay) \vdash_M (q, \beta, zy) \vdash_M^* (q, \lambda, \alpha)$$

amb $w = x\beta$ per a algun $x, \beta \in \Sigma^*$ i $A \longrightarrow z$ és una producció de P . Com que $(q, x\beta, S) \vdash_M^* (q, \beta, Ay)$ deduïm que $(q, x, S) \vdash_M^* (q, \lambda, Ay)$. Ara, per la hipòtesi d'inducció, tenim que $S \xRightarrow{*} xAy$ i, aplicant la producció $A \longrightarrow z$, $S \xRightarrow{*} xzy$. Com que també $(q, \beta, zy) \vdash_M^* (q, \lambda, \alpha)$ utilitzant només transicions de tipus (3), obtenim que $\beta\alpha = zy$ i per tant $S \xRightarrow{*} xzy = x\beta\alpha = w\alpha$.

Ara cal demostrar que qualsevol llenguatge associat a un autòmat amb pila és un llenguatge incontextual. Tal com hem fet a anteriors demostracions, definirem un autòmat equivalent però més simple, de manera que sigui més senzill procedir amb la demostració.

Definició 12. Direm que un autòmat amb pila és **simple** si es compleix que, per a qualsevol transició $((p, a, b), (q, x))$ on q no és l'estat inicial, b és un element de l'alfabet de la pila ($b \in \Gamma$) i $|x| \leq 2$. És a dir, és un autòmat que només llegeix el caràcter del cim de la pila i o bé l'elimina o bé el substitueix per un o dos símbols.

□

1.4 Teoremes d'extracció

Els models computacionals que hem vist fins ara no tenen una semblança aparent al que considerem ordinadors. És evident que poden fer un cert tractament de dades i poden reconèixer llenguatges, però aquestes capacitats no les associem a computació arbitrària. De fet, ni tan sols són universals en quant al

tractament de llenguatges, sinó que estan limitats a llenguatges regulars (en el cas d'autòmats finits) i llenguatges incontextuals (en el cas d'autòmats amb pila). A continuació es presenten uns resultats, anomenats teoremes d'extracció, que donen una manera de distingir els llenguatges regulars (resp. incontextuals) d'aquells que no ho són.

Fer les introduccions corresponents

1.4.1 Per a llenguatges regulars

Teorema. *Sigui L llenguatge regular infinit. Aleshores existeix $n_L \in \mathbb{N}$ (dependent d' L) tal que per a tota paraula $w \in L$ amb $|w| \leq n$ existeix una representació de la paraula $w = xyz$ tal que:*

1. $|xy| \leq n$
2. $y \neq \lambda$
3. $xy^iz \in L \forall i \geq 0$

Demostració. Com que L és un llenguatge regular, existeix un autòmat determinista $M = (K, \Sigma, \delta, q, F)$ que reconeix L . Sigui n el nombre d'estats en K i considerem una paraula $w \in L(M)$ tal que $|w| \geq n$. Suposem que $w = a_1 \dots a_k$, amb $a_i \in \Sigma \forall i = 1 \dots k$. Tenim que $k \geq n$, de manera que per reconèixer la paraula w necessàriament M ha de passar dues vegades per un mateix estat de K . Per a tot $m \leq k$, sigui q_m l'estat en el que es troba M després d'haver llegit els m primers símbols de w . Aleshores, existeixen i, j amb $i < j \leq n$ tals que $q_i = q_j$. Definim, doncs

- $x = a_1 \dots a_i$
- $y = a_{i+1} \dots a_j$
- $z = a_{j+1} \dots a_k$

Com que $j \leq n$, tenim que $|xy| \leq n$; com que $i < j$, $y \neq \lambda$. Finalment, com que $q_i = q_j$ es pot repetir el fragment y tantes vegades com es vulgui, obtenint que $xy^iz \in L \forall i \geq 0$. □

1.4.2 Per a llenguatges incontextuals

Teorema. *Sigui L un llenguatge incontextual infinit. Aleshores existeix $n_L \in \mathbb{N}$ tal que, per a tota paraula $w \in L$ amb $|w| \geq n_L$, existeix una representació $w = uvxyz$ tal que:*

- $|vxy| \leq n$
- $vy \neq \lambda$
- $uv^i xy^i z \in L \forall i \geq 0$

Fer la demo i afegir exemples

Capítol 2

Màquines de Turing

Hem vist, doncs, que els models computacionals que ens donen els autòmats no són del tot universals, ja que hi ha llenguatges que no poden ser reconeguts per aquests. Així, si es pretén obtenir un model de computació universal, es necessita algun model més potent, més universal, que trobarem en les màquines de Turing.

En una màquina de Turing continuem tenint una cinta amb la paraula d'entrada i una unitat de control amb un conjunt d'estats. En aquest model, però, la cinta és infinita (encara que la paraula no ho sigui) i les cel·les de la cinta no és només permeten la lectura, sinó que també s'hi poden escriure (o sobreescriure) caràcters. A més, el punter que defineix quina cel·la de la cinta es llegeix ara es podrà moure tant cap a l'esquerra com cap a la dreta. Així doncs, en certa manera, es pot utilitzar la cinta com a memòria sense les limitacions de les piles.

Decidir si anomenem al capçal punter o capçal i vigilar que sigui consistent

Definició 13. Una màquina de Turing determinista és una estructura $M = (K, \Sigma, \Gamma, \delta, q_0, q_F)$ on:

- K és un conjunt finit i no buit d'estats
- Σ és l'alfabet d'entrada, que no conté els símbols $\$$ i $*$
- Γ és l'alfabet de la cinta, complint que $\Sigma \cup \{\$, *\} \subseteq \Gamma$ (de fet, en la majoria de casos usals tindrem que $\Gamma = \Sigma \cup \{\$, *\}$)
- $q_0 \in K$ és l'estat inicial
- $q_F \in K$ és l'únic estat acceptador

perquè
no
un
con-
junt?

- $\delta: (K \setminus \{q_F\}) \times \Gamma \rightarrow K \times \Gamma \times \{L, R, S\}$ una funció parcial tal que,
 $\forall q \in K \setminus \{q_F\}$

posar
les
aclara-
cions
a
sota
quan
s'ha-
gi
definit
tot?

- ★ Si $\delta(q, \$) = (p, b, i) \implies b = \$ \wedge i \neq L$ (i.e. si la màquina és a l'inici de la cinta, no es pot sobre escriure el símbol d'inici de cinta i no podem moure el capçal més cap a l'esquerra)
- ★ Si $a \neq \$$, $\delta(q, a) = (p, b, i) \implies b \neq \$$ (i.e. no es pot escriure el símbol d'inici de la cinta en cap posició que no sigui l'inici de la cinta)

Prendrem la cinta com a acotada per l'esquerra i amb $*$ el símbol de l'alfabet de la cinta utilitzat per denotar la cel·la buida. A la primera casella de la cinta hi haurà sempre el símbol inicial $\$$, i a continuació d'aquest símbol hi haurà escrita la paraula d'entrada. La resta de la cinta contindrà el caràcter de cel·la buida. En un pas de còmput, el punter podrà escriure un caràcter a la cel·la a la que apunta i podrà moure's a la cel·la anterior o següent. Denotarem per

- L = moviment del punter cap a l'esquerra
- R = moviment del punter cap a la dreta
- S = sense moviment de punter

Inicialment la màquina de Turing M es troba a l'estat inicial q_0 i el punter està situat a la primera cel·la de la paraula (i per tant a la segona cel·la de la cinta).

Els còmputos de la màquina es fan a partir de la funció δ que, de manera equivalent als autòmats, indica com passar d'un pas de còmput al següent en funció de l'estat actual i el símbol de la cinta al que apunta el punter. Així, si $a \in \Gamma$ és el símbol al que apunta el punter i $q \in K$ és l'estat actual, si $\delta(q, a) = (p, b, i)$, la màquina passarà a l'estat p , escriurà el símbol b a la cel·la que indica el punter (la que apuntava el punter i contenia a) i finalment,

$$\begin{cases} \text{es mou el punter una cel·la cap a la dreta} & \text{si } i=R \\ \text{es mou el punter una cel·la cap a l'esquerra} & \text{si } i=L \\ \text{no es mou el punter} & \text{si } i=S \end{cases}$$

Definició 14. Sigui $M = (K, \Sigma, \Gamma, \delta, q_0, q_F)$ una màquina de Turing determinista. Definim una **configuració** de M com (α, q, β) on $q \in K$ i $\alpha, \beta \in \Gamma^*$ tal que β no acaba en $*$.

Direm que si, en un pas de còmput la configuració de M és (α, q, β) , aleshores M està en l'estat q , el contingut de la cinta és $\alpha\beta$ i el punter senyala al primer caràcter de β si $\beta \neq \lambda$ i a una cel·la buida si $\beta = \lambda$.

Si (α, q, β) i (α', q', β') són configuracions de M , direm que (α, q, β) **produeix** (α', q', β') **en un pas de còmput** (i ho escriurem $(\alpha, q, \beta) \vdash_M (\alpha', q', \beta')$) si podem passar de (α, q, β) a (α', q', β') aplicant una vegada la funció δ . Farem servir la notació exponencial sobre el símbol \vdash per expressar que un estat en produeix un altre amb n passos de còmput de la següent manera: \vdash_M^n . Finalment, direm que (α, q, β) **produeix** (α', q', β') i ho escriurem $(\alpha, q, \beta) \vdash_M^* (\alpha', q', \beta')$ si existeix $n \geq 0$ tal que $(\alpha, q, \beta) \vdash_M^n (\alpha', q', \beta')$.

Observem que, de moment, no hem parlat en cap moment de la parada de la màquina. Observem, també, que en la definició de màquina de Turing, la funció de transició δ és parcial i que el domini exclou l'estat final q_F . Aquesta exclusió és ben intencional i, de fet, serà la que ens permetrà definir el concepte de parada:

Definició 15. Sigui $M = (K, \Sigma, \Gamma, \delta, q_0, q_F)$ una màquina de Turing determinista, (α, q, β) una configuració de M i sigui b el primer caràcter de β (el que apunta el punter). Direm que M **para** sobre la configuració (α, q, β) si $\delta(q, b)$ no està definida. En particular, M para si $q = q_F$.

Cal remarcar que, segons aquesta definició de parada, M sempre para en arribar a l'estat q_F , però que no sempre que para ho fa en una configuració amb l'estat q_F .

Definició 16. Sigui $M = (K, \Sigma, \Gamma, \delta, q_0, q_F)$ una màquina de Turing determinista i sigui $x \in \Sigma^*$.

Direm que M **para sobre la paraula** x si el còmput de M sobre la entrada x para, i en aquest cas escriurem $M(x) \downarrow$. En cas que el còmput sobre x no pari, escriurem $M(x) \uparrow$. En cas que M pari sobre una entrada x , denotem per $M(x)$ la paraula de Σ^* que apareix en la última configuració del còmput de M sobre x .

Direm, també, que M és una màquina de **parada segura** si $\forall x \in \Sigma^*$ tenim que $M(x) \downarrow$.

Definició 17. Si M és una màquina de Turing amb un alfabet d'entrada Σ , definim la **funció associada** a M com $f_M: \Sigma^* \rightarrow \Sigma^*$ tal que, per a tot $x \in \Sigma^*$

$$f_M(x) = \begin{cases} M(x) & \text{si } M(x) \downarrow \\ \text{indefinit} & \text{si } M(x) \uparrow \end{cases}$$

Aleshores, direm que una funció $f: \Sigma^* \longrightarrow \Sigma^*$ és **computable** si existeix una màquina de Turing M tal que $f = f_M$.

També es pot definir el concepte de funció computable de diverses variables de la següent manera:

Definició 18. *Sigui $n \geq 2$, sigui $f: (\Sigma^*)^n \longrightarrow \Sigma^*$ i sigui $M = (K, \Sigma, \Gamma, \delta, q_0, q_F)$ una màquina de Turing determinista tal que $\Sigma \cup \{\$, *, c\} \subseteq \Gamma$, on c és un caràcter de control tal que $c \notin \Sigma$. Diem que M **computa** f si, per a qualssevol $u_1, \dots, u_n \in \Sigma^*$*

1. *Si $f(u_1, \dots, u_n) = u$ aleshores el còmput de M sobre l'entrada $u_1cu_2c \dots cu_n$ para i dóna com a resultat u .*
2. *Si $f(u_1, \dots, u_n)$ no està definida, aleshores el còmput sobre l'entrada $u_1cu_2c \dots cu_n$ no para.*

Es pot demostrar que totes les funcions aritmètiques són computables per màquines de Turing.

2.0.1 Llenguatges decidibles

Moure aquesta definició abans de la definició de funcions computables?

Definició 19. *Sigui $M = (K, \Sigma, \Gamma, \delta, q_0, q_F)$ una màquina de Turing determinista. Diem que una paraula $x \in \Sigma^*$ és **reconeguda o acceptada** per M si M para sobre l'entrada x en l'estat acceptador q_F . Direm que és **rebutjada o no és reconeguda** si M para en qualsevol altre estat de K o bé no para.*

No parar ho he afegit jo, si no no té sentit la distinció entre acceptable i recursiu

Definició 20. *Definim el llenguatge associat a una màquina de Turing M per $L(M) = \{x \in \Sigma^* \mid x \text{ és reconeguda per } M\}$. Direm que un llenguatge L és **acceptable** si existeix una màquina de Turing determinista M tal que $L(M) = L$.*

Definició 21. *Sigui L un llenguatge. Diem que L és **recursiu o decidible** si existeix una màquina de Turing M determinista i de parada segura tal que $L = L(M)$. En aquest cas, direm que M **decideix** el llenguatge L .*

Aleshores tot llenguatge recursiu és acceptable, però l'invers no és cert, ja que les màquines de Turing que accepten un llenguatge poden no parar en les paraules rebutjades.

2.0.2 Extensions de Màquines de Turing

A part de la definició que ja hem donat de la màquina de Turing, existeixen diverses definicions alternatives amb algunes variacions sobre la estructura o comportament. A continuació es presenten dues variacions: una màquina de Turing amb múltiples cintes i una màquina de Turing indeterminista. A priori pot semblar que aquestes modificacions augmenten, d'alguna manera, la capacitat de càlcul de les màquines, però veurem també que les “màquines ampliades” són equivalents a la definició original, reforçant la idea de màquina de Turing com a model computacional universal.

Màquines de Turing amb múltiples cintes

Una màquina de Turing amb múltiples cintes és una màquina de Turing estàndard a la que se li afegeix un nombre finit de cintes i un punter associat a cada una d'aquestes cintes. Considerem la primera cinta com a cinta d'entrada i la resta com a cintes auxiliars. A l'inici de cada cinta hi haurà escrit el caràcter \$ d'inici de cinta seguit de la paraula d'entrada i espais en blanc a la primera cinta i d'espais en blanc a la resta de cintes. Inicialment els punters a cada cinta apunten a la segona posició (i.e. l'inici de la paraula a la primera cinta i el primer espai en blanc a la resta de cintes).

El funcionament d'aquesta màquina és idèntic al d'una màquina de Turing amb una sola cinta, amb la diferència que en un sol pas de còmput es pot escriure qualsevol de les posicions de les cintes indicades pels punters i es pot moure els punters de manera independent per a cada cinta. A més, en un pas de còmput es permet fer el canvi d'estat no només a partir de l'estat actual i el símbol de la cinta d'entrada, sinó també en funció dels símbols de les cintes auxiliars. Això, però, fa que la funció de transició sigui significativament més complicada. Vegem-ne la definició formal:

Definició 22. Prenent $k \geq 2$, diem que una màquina de Turing determinista de k cintes és una estructura $M = (K, \Sigma, \Gamma, \delta, q_0, q_F)$ on:

- K, Σ, Γ, q_0 i q_F es defineixen de la mateixa manera que en la màquina de Turing tradicional
- $\delta: (K \setminus \{q_F\}) \times \Gamma^k \rightarrow K \times (\Gamma \times \{L, S, R\})^k$

De manera que en un pas de còmput es determina el canvi d'estat, l'escriptura sobre les cintes i el moviment de cada punter en funció de l'estat actual i els símbols sota el punter de cada cinta.

La resta de conceptes associats a les màquines de Turing es defineixen de la mateixa manera que en les màquines tradicionals tenint en compte la nova definició de δ .

Com a tota extensió de model computacional que s'ha presentat durant el treball, sembla natural preguntar-nos si aquesta modificació aporta més capacitat de càlcul al model o bé és equivalent al model original.

Teorema. *Tota màquina de Turing determinista M amb múltiples cintes es pot simular per una màquina de Turing determinista M' d'una sola cinta.*

Idea de la demostració. *Suposem que M és una màquina de Turing determinista de $k \geq 2$ cintes amb un alfabet d'entrada Σ i alfabet de cinta Γ . Creem una màquina de Turing M' amb*

- *Un alfabet d'entrada Σ i alfabet de cinta $\Gamma' = \Sigma \cup \{\bar{x} \mid x \in \Sigma\} \cup \{\$, *, \#\}$*
- *El contingut inicial de la cinta és el símbol $\$$ seguit de la concatenació de les k cintes d' M , separades pel símbol $\#$. Això genera una partició de la cinta en k fragments, cadascun limitat per l'esquerra per $\$$.*
- *Per simular el fet de tenir k capçals, en cada fragment se substituirà el símbol x de la posició on apuntaria el capçal pel símbol \bar{x} , com si es tractés d'un capçal virtual.*

Per simular un pas de còmput la màquina de k cintes, es faria el següent procediment:

1. *Es mou el punter cap a l'esquerra fins a trobar el símbol $\$$ d'inici de cinta.*
2. *Es recorre la cinta d'esquerra a dreta des de $\$$ fins al k -èssim símbol $\#$ per tal de determinar els símbols sota els capçals virtuals, codificant d'alguna manera aquestes posicions amb en l'estat de la màquina M' .*
3. *Aleshores es fa una altra passada a la cinta de dreta a esquerra, fent les modificacions de continguts de les cintes i punters virtuals que dictaria la funció de transició d' M . Si en algun moment cal moure un punter virtual a una posició de la cinta amb un símbol $\#$ on comença la següent cinta, la màquina haurà d'escriure el símbol $*$ en aquella casella i moure tots els símbols posteriors una casella cap a la dreta.*

Evidentment a aquesta idea hi falten molts detalls per esdevenir una demostració formal: caldria veure com definir els estats i l'enorme funció de transició d' M , que en qualsevol moment podem moure els símbols a partir d'una posició cap a la dreta i després continuar amb l'operació, que si M para M' també i que aleshores $L(M) = L(M')$... Tots aquestes comprovacions, però, queden fora del treball i es poden trobar més desenvolupades a <https://courses.engr.illinois.edu/cs373/sp2013/Lectures/lec20.pdf> o a <http://staff.um.edu.mt/afra1/teaching/coco4.pdf>

citar bé les pàgines

Màquines de Turing indeterministes

Una altra possible modificació prové d'agafar el concepte de indeterminisme desenvolupat en l'apartat d'autòmats indeterministes i adaptar-lo a màquines de Turing.

Definició 23. *Una màquina de Turing indeterminista és una estructura $M = (K, \Sigma, \Gamma, \delta, q_0, q_F)$ on:*

- K, Σ, Γ, q_0 i q_F es defineixen de la mateixa manera que en la màquina de Turing tradicional
- $\delta: ((K \setminus \{q_F\}) \times \Gamma) \rightarrow \mathcal{P}((K \times \Gamma \times \{L, S, R\}))$. És a dir, que en lloc de ser un element de $(K \times \Gamma \times \{L, S, R\})$ tal com ho era abans, $\delta(p, a)$ n'és un subconjunt finit.

També ho he posat com a funció, tal com es va fer als autòmats indeterministes. Ho deixem així o ho poso com a les transparències?

Podem definir tota la resta de conceptes associats a una màquina de Turing indeterminista de la mateixa manera que ho hem fet a les deterministes.

Referències

- [1] Harry Lewis i Christos H. Papadimitriou. *Elements of the Theory of Computation*. Second Edition. Upper Saddle River, NJ: Prentice-Hall, 1998. ISBN: 0132624788 9780132624787 0132727412 9780132727419.
- [2] Michael Sipser. *Introduction to the Theory of Computation*. Third Edition. Boston, MA: Course Technology, 2013. ISBN: 113318779X.