



UNIVERSITAT DE GIRONA

PRÀCTICA FINAL

Fonaments de la computació

Francesc Xavier Bullich Parra

Gil Gassó Rovira

Marc Sànchez Pifarré

Tutor de la pràctica
Jaume Rigau

December 18, 2018

Contents

1	Introducció	2
1.1	Definicions	2
2	Autòmata Finit	3
2.1	Definició del context	3
2.2	Dissenyant el nostre autòmata finit	3
2.2.1	Construïm l'autòmata sense RESET	5
2.2.2	Construïm l'Autòmata Complet	5
2.3	Definició formal de $FXBP_{DFA}$	6
2.3.1	δ table representation	7
3	Push-Down Automata	9
4	Turing Machine	10

Chapter 1: Introducció

1.1 Definicions

Es defineixen una sèrie de noms que seran utilitzats al llarg de la pràctica per ajudar a la simplificació i l'enteniment de la mateixa.

- Definim $FXBP_{DFA}$ com l'autòmata finit determinista capaç de reconèixer el llenguatge $L(FXBP_{DFA})$.
- Definim $L(FXBP_{DFA})$ com el conjunt infinit de mots ω que accepta $FXBP_{DFA}$.
- Definim ω com un mot tal que $\omega \in L(FXBP_{DFA})$.
- Definim Σ^*FXBP_{DFA} com el conjunt de símbols amb el que es construeix el llenguatge $L(FXBP_{DFA})$.
- Definim context Φ com l'escenari ideal en el que l'autòmata a dissenyar es comportarà de manera correcta, $\Phi = \langle \Upsilon, \Psi \rangle$.
- Definim Υ com la precondition que s'ha de complir al utilitzar els nostres autòmats.
- Definim Ψ com la postcondició de l'autòmata.

Chapter 2: Autòmata Finit

2.1 Definició del context

Sigui Φ_{DFA} l'espai de funcionament lògic del nostre autòmata com l'espai estipulat a la Secció 3.Patrons[2]. Per tant acabem d'adquirir totes les definicions assumides a l'enunciat del problema, veure [2].

Fem les següents afirmacions sobre Υ :

- $\omega \in P_0 \wedge P_0 \in P$
- γ com a llesca continguda dins d'un P on $\gamma \in \omega \vee \gamma = \epsilon$

Fem les següents afirmacions sobre Ψ :

- retorna True quan $FXBP_{DFA}$ Accepta $\omega \leftrightarrow \exists \gamma \in P$ que compleix PIP.
- retorna False quan $\nexists \gamma \in P$ que compleix PIP.

on : PIP és la propietat dels patrons implícits parcials definida a [2] apartat 3.3.1.


2.2 Dissenyant el nostre autòmata finit


Primerament cal observar possibles propietats del problema que ens puguin servir per a la construcció de l'autòmata.

- Definim 4+ com la seqüència de símbols '++++' d'un mot ω comprés dins del llenguatge $L(FXBP_{DFA})$.

Realitzem les següents observacions :

1. Abans i després de la seqüència 4+ hi pot haver qualsevol símbol 0 o més vegades comprés dins de Σ^*FXBP_{DFA} .

2. Aïllada és la seqüència de símbols ‘-+-’ dins de ω
3. Entre dues seqüències $4+$ hi trobem 3 aïllades.
4. 2 aïllades poden compartir el symbol inicial o el symbol final o ambdós.
5. S’accepta ‘-+-+’ com a dues cèl·lules aïllades on -+ comparteix - amb +-..
6. S’accepta ‘-+-+--+’ coma tres cèl·lules aïllades on -+ comparteix- amb +-+- i -+-+ comparteix - amb +-.
7. Entre 2 aïllades hi pot aparèixer qualsevol combinació de symbols pertanyent a Σ^*FXBP_{DFA} .
8. El symbol  ens fa tornar a començar a cercar el PIP (RESET).

Dissenyarem doncs l’autòmata per construcció tenint en compte les anteriors propietats. En la construcció del nostre autòmata reduim el problema al tractament d’un subconjunt dels símbols de l’alfabet, concretament es construeix a partir del subconjunt +,-, ja que detectem que el símbol  actúa com a RESET. La funció del RESET en $FXBP_{DFA}$ és exemplificada al capítol 1 del llibre [1], concretament als exemples 1.15 i 1.17.

Incorporarem el RESET al nostre disseny a l’últim pas de la construcció.

Fem les següents definicions en l’escenari sense RESET :

- Definim pa com la seqüència ‘-+’
- Definim fa com la seqüència ‘+-’
- Definim l’operació CONCATENACIÓ amb el symbol |
- Definim 3aïllades com :

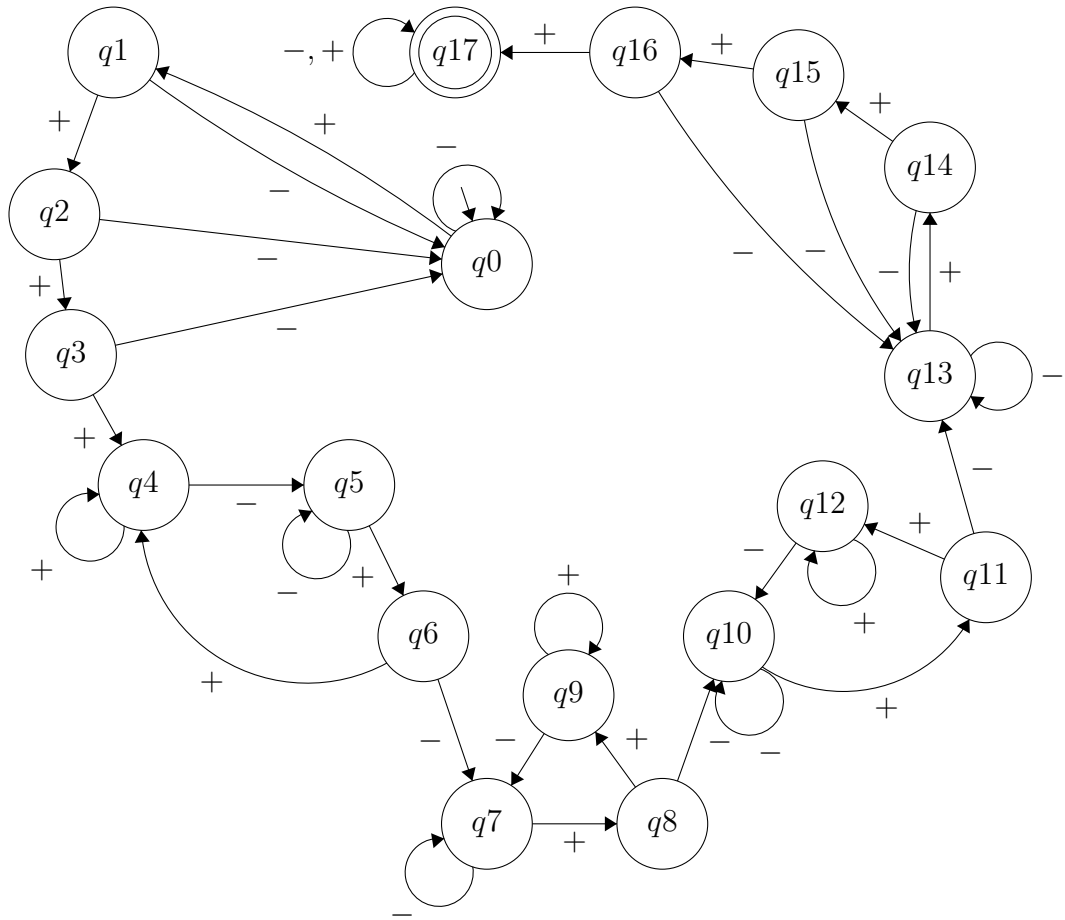
$$- (+-* | pa | (-+*)^* | pa | (-+*)^* | pa | - | +-*) \vee (+-* | - | fa | (-+*)^* | fa | (-+*)^* | fa | +-*)$$

Per tant podem concebre que en un entorn on no hi ha RESET es cerca si el mot conté la següent seqüència de símbols :

$$+ -* | 4+ | 3aïllades | 4+ | + -*$$

2.2.1 Construïm l'autòmata sense RESET

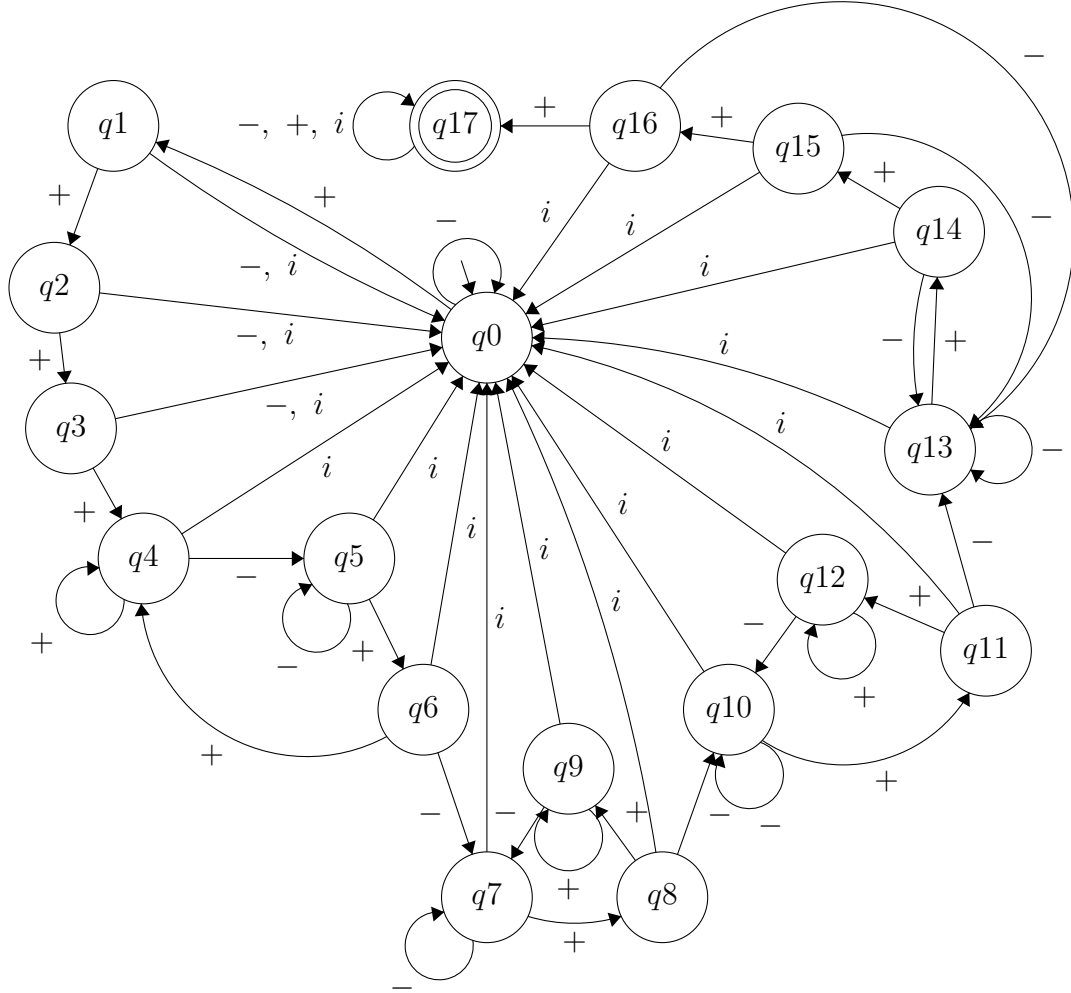
Veiem que els estats de l'autòmata van avançant a mesura que es van trobant el patró. Hi distingim cicles que ens interessen per controlar els casos que on per exemple hi pot haver qualsevol cosa entre cel·les actives o entre cel·les actives i principi o final del patró.



2.2.2 Construïm l'Autòmata Complet

Afegim RESET a l'autòmata, en aquest cas tots els estats menys l'estat final tindran una transició a l'estat inicial amb el símbol . L'estat final com que el mot ja està acceptat tindrà una transició a ell mateix amb el símbol .

En aquest disseny, per simplificar la codificació del caràcter el codifiquem com una *i*.




2.3 Definició formal de $FXBP_{DFA}$

Definim formalment el nostre autòmata.

$$FXBP_{DFA} = \{Q, \Sigma^*, \delta, q_0, q_{17}\}$$

- $Q = \{q_0, q_1, q_2, q_3, q_4, q_5, q_6, q_7, q_8, q_9, q_{10}, q_{11}, q_{12}, q_{13}, q_{14}, q_{15}, q_{16}, q_{17}\}$
- $\Sigma^* = \{+, -, \boxed{\leftarrow}, \epsilon\}$
- $\delta = Q \times \Sigma^* \rightarrow Q$ Represented by :
- $q_0 \in Q$

Table 2.1: δ table			
State	+	-	
$\rightarrow q0$	q1	q0	q0
q1	q2	q0	q0
q2	q3	q0	q0
q3	q4	q0	q0
q4	q4	q5	q0
q5	q6	q5	q0
q6	q4	q7	q0
q7	q9	q7	q0
q8	q8	q7	q0
q9	q8	q10	q0
q10	q12	q10	q0
q11	q11	q10	q0
q12	q11	q13	q0
q13	q14	q13	q0
q14	q15	q13	q0
q15	q16	q13	q0
q16	q17	q13	q0
q17	q17	q17	q17

- $q17 \subseteq Q$

2.3.1 δ table representation

Un exemple de llesca horitzontal que compleix el patró :

1 +++++-+-+--+++++

Un exemple de llesca horitzontal que no compleix el patró :

1 +++++-+----+-+++++

Exemple de codi incrustat en latex

```
1 #include <iostream>
2 #include <vector>
3 using namespace std;
4 #include "Matrix.h"
5
6 Matrix::Matrix()
7 {
8
9 }
10
11 void Matrix::generateFromString(string matrixL2, char delimiter){
12
13 }
```

Listing 2.1: Matrix example

Chapter 3: Push-Down Automata

Chapter 4: Turing Machine

Bibliography

- [1] Fundamental of Computation - Michael Sipser
- [2] Problema Patterns (i.e., Cerca de Patrons) - Jaume Rigau