Theorievragen

1. Combinational Logic: Boolean circuits. Je hebt bijvoorbeeld een input A en input B die of 0 of 1 zijn. Als A en B beide op 1 staan, dan is de output True. Dit zijn dus eigenlijk je logic gates (zoals enkele AND, OR gates). Deze soort machine is niet afhankelijk van tijd; dat A vroeger op 0 stond, en nu op 1, heeft geen invloed op de output. Alleen het feit dat A NU op 1 staat heeft uiteindelijk invloed op de output.

   Finite State Machine: Liften. Een lift heeft een eindig aantal staten (of verdiepingen). Stel een lift staat stil op de 1e verdieping. Iemand drukt op een knop voor verdieping 5. De lift moet dus omhoog, want hij staat op de 1e verdieping. De lift blijft omhoog gaan totdat hij op de 5e verdieping is. Daarna stopt hij. Het feit dat de lift op de 1e verdieping stond heeft dus invloed gehad op het gedrag van de lift: de lift moest omhoog. Als de lift op de 6e verdieping stond, dan zou deze omlaag moeten.

   Turing Machine: Is een generiek voorbeeld van een CPU. Het controleert alle data manipulatie dat wordt gedaan door een computer, en kan dit opslaan in een tape (met een eindige lengte). Het kan first-order logic evalueren in een oneindig aantal manieren.

2. Bij een deterministische eindige automaat is er altijd maar 1 begintoestand. Daarnaast weet je dat als je in toestand q1 verkeert, en het symbool y is, dat je dan bijv. nu in toestand q2 zit.

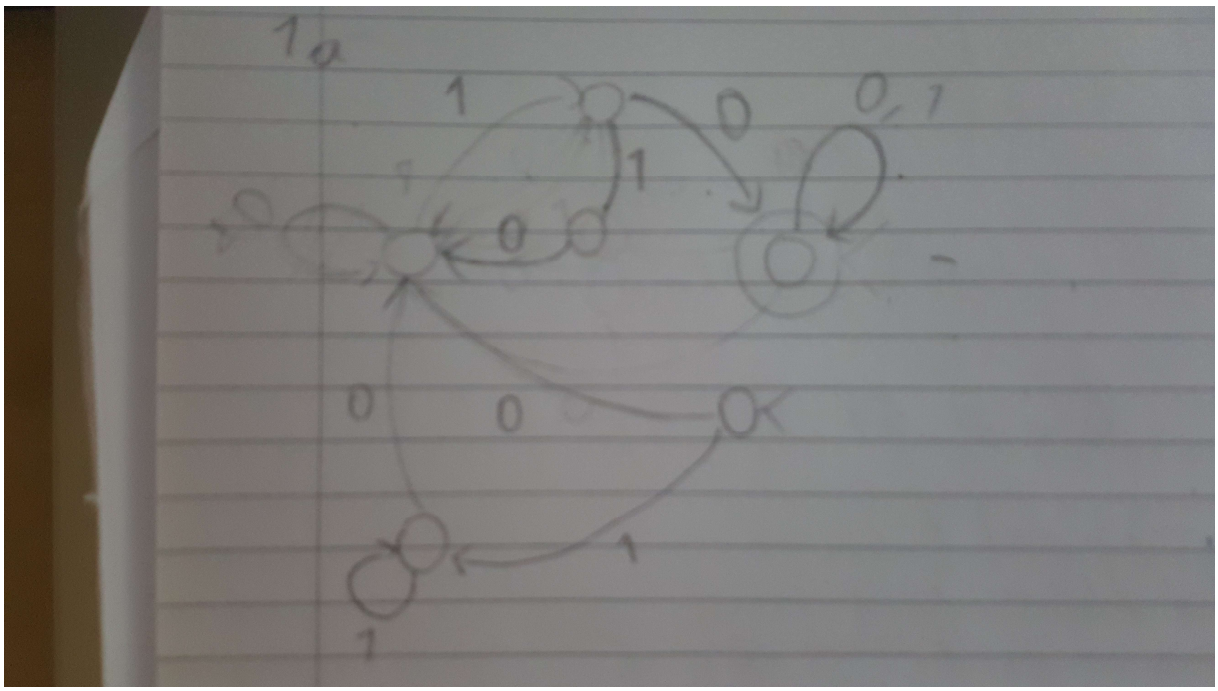
   Dit is bij niet-deterministische eindige automaten dus niet het geval. Ten eerste hebben deze een verzameling begintoestanden (waaruit er eentje wordt gekozen als relevante begintoestand). Ten tweede, als de toestand q1 en is en het symbool is y, dan heb je een verzameling vervolg toestanden, in plaats van dat je nu in een keer in toestand q2 zit. Willekeurig wordt een toestand uit deze verzameling gekozen.

   Het grootste verschil tussen de twee is dus het feit dat deterministische automaten voorspelbaarder zijn in het verloop van de verwerking van input dan niet deterministische automaten. Een DFA is makkelijker te besturen en te plannen dan een NFA, maar een NFA heeft meer soorten transities en kan per symbool en toestand meerdere vervolgtoestanden definiëren.

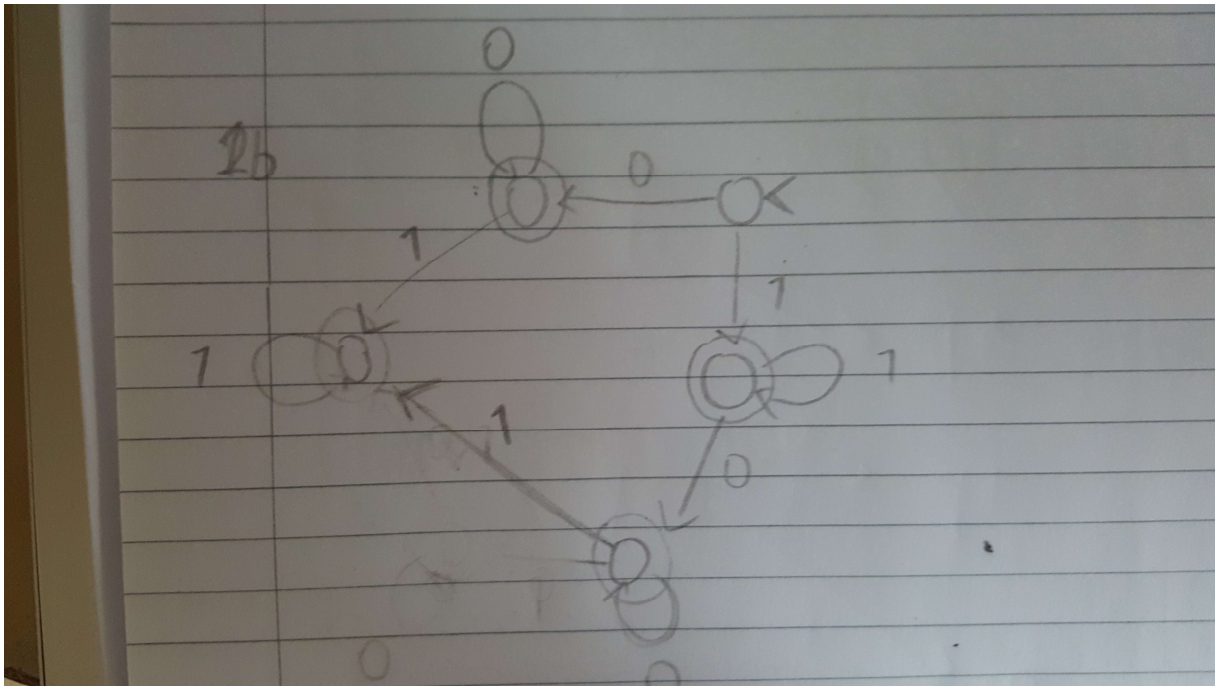
Oefeningen

1.

010 als substring:



010 NIET als substring:



2. automaat 1: deze automaat heeft in zijn alfabet alleen ‘a’ en ‘b’ zitten. Ieder geldig woord in deze taal, moet starten op een a en eindigen op een a. Dus ‘a’ is bijvoorbeeld een geldig woord. Daarnaast mogen er ‘b’s tussen de ‘a’s zitten. Dus ‘aba’ is geldig. Maar ‘ababa’ is ook geldig. Maar ‘abababa’ is ook geldig. Hier kan je dus bijna oneindig in door gaan.

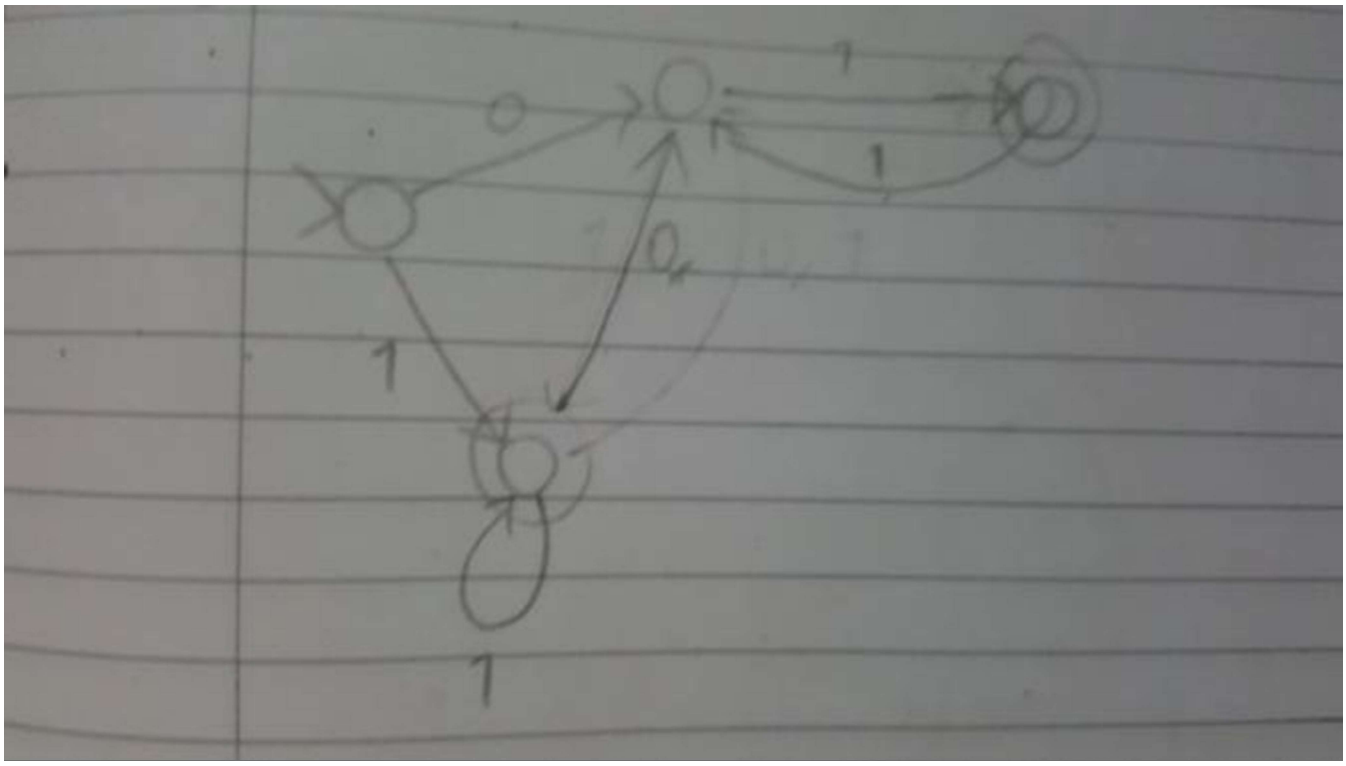
‘b’ is echter ongeldig. ‘bab’ is ook ongeldig. ‘abb’ is ook ongeldig.

Automaat 2: deze automaat heeft op nieuw in zijn alfabet alleen ‘a’ en ‘b’ zitten. Ieder geldig woord in deze taal, moet starten met een ‘a’ en eindigen op een ‘b’. ‘ab’ is dus geldig. ‘aab’ en ‘aaab’ etc. zijn ook geldig. Je mag ook starten met een ‘b’, echter moet het woord dan meteen eindigen (dus niet twee b’s). ‘b’ is dus een geldig woord. ‘bb’ niet.

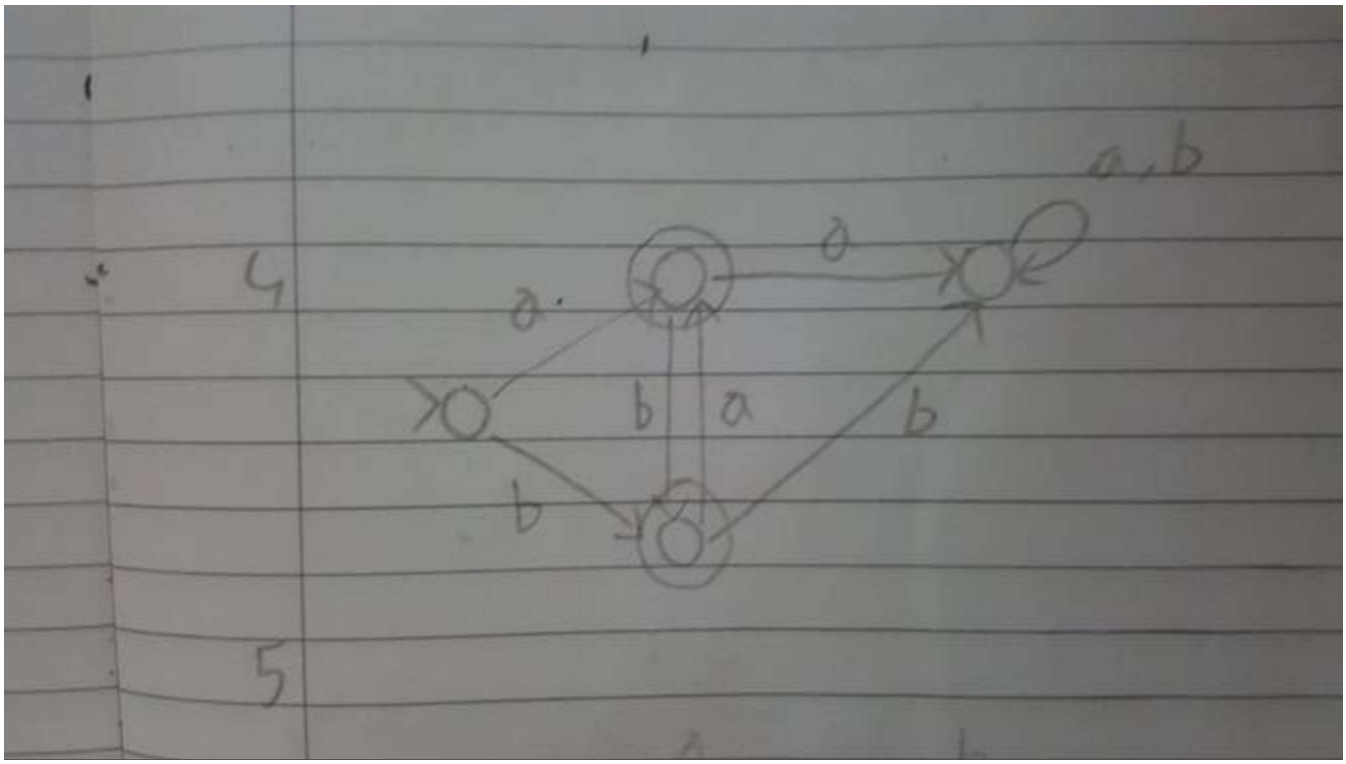
Je mag niet eindigen op een ‘a’. Ook mag je niet eindigen op twee (of meer) b’s.

3. het alfabet van deze taal is 0 en 1. Het woord moet starten met 0 of met 1. Het woord, als je start met 0, moet eindigen op een oneven aantal 1, tenzij je met 1 start, in dat geval mag je eindigen met een even aantal 1. Je kan niet eindigen op een 0. 11 en 1111 is geldig. 01, 0111 is ook geldig. 011 is echter ongeldig. Ook is 10 ongeldig. 101 is dan wel weer geldig omdat je eindigt op een 1. 111 is ook geldig, doordat je twee kanten op kan als je start met een input van 1.

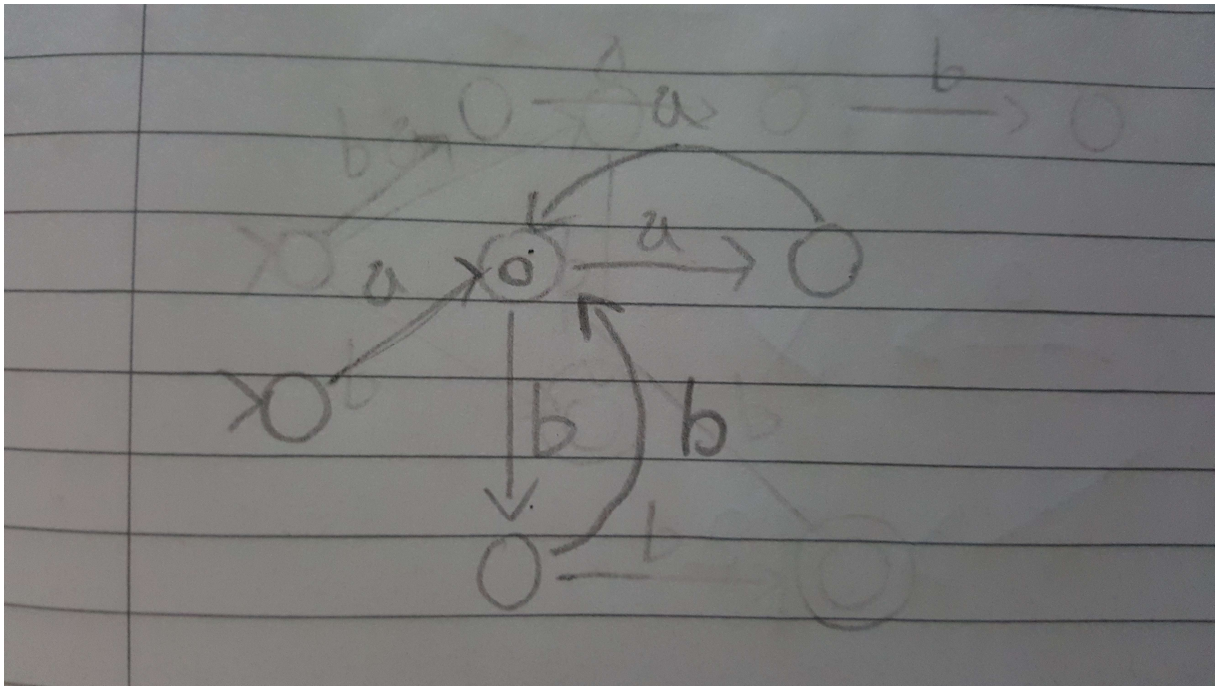
Tot slot kan je geen meerdere nullen naast elkaar hebben. 1001 is niet mogelijk.



4.



5.



Turing Machine

1.

aqbb _ bb _ _ _ aba q0

abqb _ bb _ _ _ aba q0

abbq_ bb _ _ _ aba q0

abb_qbb _ _ _ aba q0

abb_bqb _ _ _ aba q0

abb_bbq _ _ _ aba q0

abb_bb _q _ _ aba q0

abb_bb _ _ q_ aba q0

abb_bb _ _ _ qaba q0

abb_bb _ _ q_ aba q1

abb_bb _ q_ _ aba q1

abb_bbq _ _ _ aba q1

abb_bqb _ _ _ aba q1

abb_bbq _ _ _ aba q2

abb_bbq _ _ _ aba h

2.

M = (K, Sigma, delta S, s, F) waarbij:

K = (q0, q1, h)

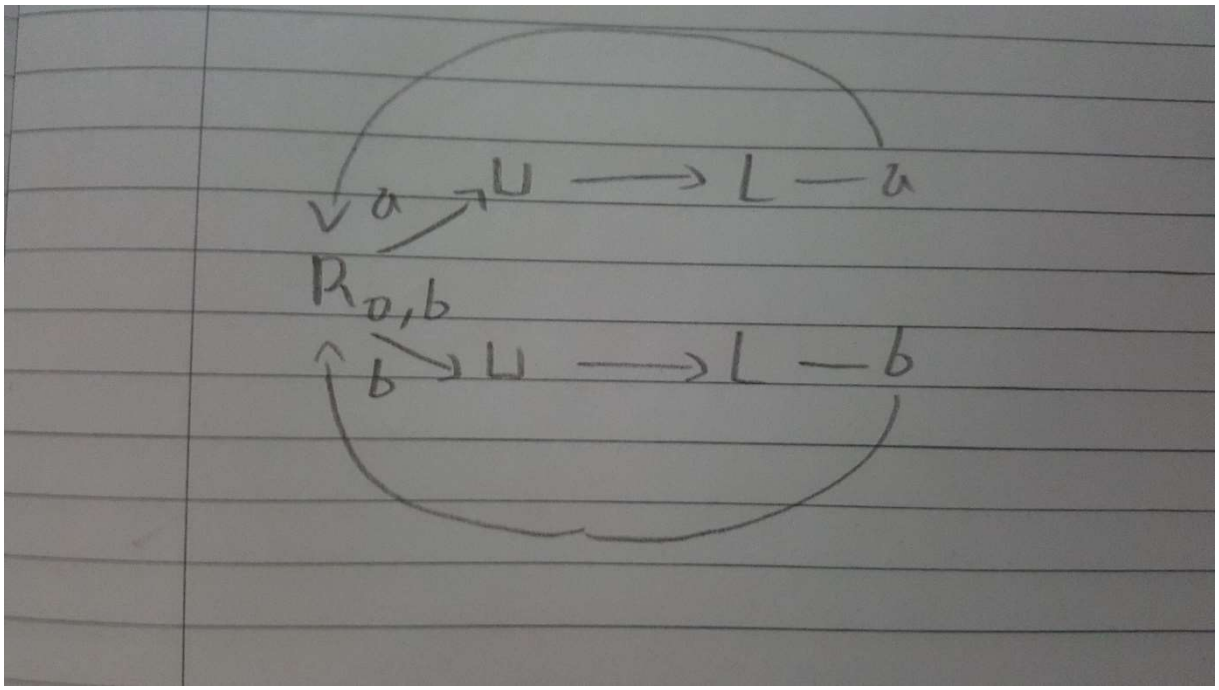
Sigma = (a, b, _)

s = q0

H = {h}

q	sigma	Delta S (q, sigma)
q0	a	(q1, ->)
q0	b	(q0, ->)
q0	_	(q0, ->)
q1	a	(h, _)
q1	b	(q0, ->)
q1	_	(q0, ->)

3.



4. De machine gaat rechts totdat hij aan het eind van de input is, daarna gaat de machine naar links. Iedere 1 wordt omgezet naar 0. Een 0 wordt omgezet naar een 1, en dan stopt de machine. Een blank wordt ook omgezet naar een 1, en opnieuw stopt de machine. De machine lijkt binaire code te veranderen.

Het tegenovergestelde van de machine: