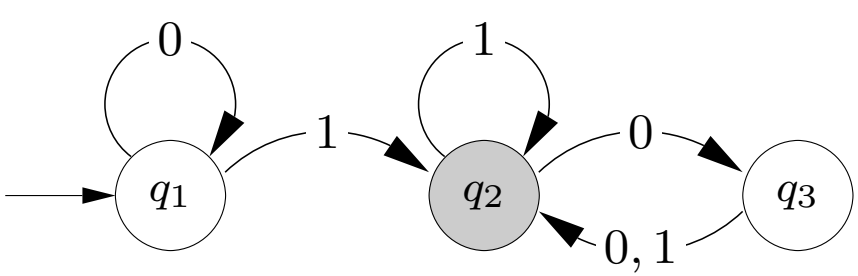
Beräkningsmodeller

- För att bygga upp en matematisk teori runt datorer måste man hantera följande problem.
  - En dator kan se ut på väldigt många sätt, och det finns ingen som är mest "datorig".
  - Konkreta datorer är vanligtvis mycket komplexa.
- En lösning är att använda sig av en idealiserad dator; en *beräkningsmodell*.
- Hur beräkningsmodellen ser ut beror på vad vi vill undersöka.

Deterministiska finita automater

- Vi börjar med en av de enklaste: Den *finita automaten*.
- Ett system som har ett ändligt antal möjliga tillstånd och vars beteende kan beskrivas med enkla transitionsregler , t.ex. om systemet är i tillstånd q och det får input a då går det till tillstånd p.
- Förekommer i hissar, automatiska dörrar, övervakningssystem etc.
- Ligger till grund för många, mer avancerade, beräkningsmodeller.
- Dessutom en utmärkt plaskbasäng där ni kan träna era förmågor i att skriva definitioner, teorem och bevis.

En finit automats anatomi

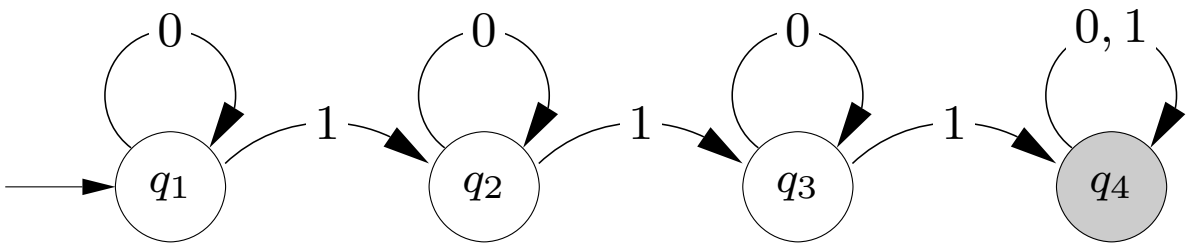


ϵ
0000
00001
0001110
0110100
1001101...1000

- Abstrakta automater består av *tillstånd*, som kan vara *initiala* och/eller *accepterande*, samt *transitioner*.
- En automata konsumerar sin input sträng från vänster till höger, en symbol i taget.
- Beroende på det tillstånd som automaten stannar i när hela strängen är läst, accepteras eller refuseras strängen, så
- varje automat definierar en mängd av strängar - automatens *språk*.

Fråga 1

Vilket språk accepterar automaten nedan?



Den formella definitionen

En *deterministisk finit automat* är en fem-tupel $(Q, \Sigma, \delta, q_0, F)$, där

- Q är en ändlig mängd *tillstånd*,
- Σ är en ändlig mängd symboler – ett *alfabete*,
- $\delta : Q \times \Sigma \mapsto Q$ är en *transitionsfunktion*,
- q_0 är *det initiala tillståndet*, och
- F är en delmängd av Q; *mängden accepterande tillstånd*.

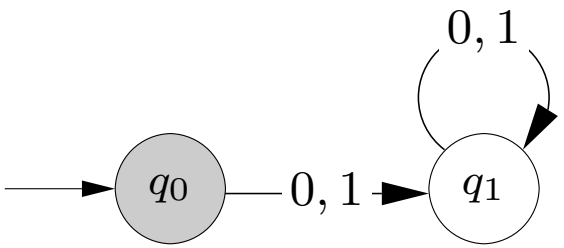
Fråga 2

Hur ser tillståndsdiagrammet ut för automaten $M = (Q, \Sigma, \delta, q_0, F)$, där $Q = \{q_0, q_1\}$, $\Sigma = \{\mathtt{1}, \mathtt{0}\}$, $F = \{q_0\}$, och δ ges av tabellen nedan? Vilket språk accepterar M?

	0	1
q_0	q_0	q_1
q_1	q_1	q_0

Fråga 3

Hur definieras automaten nedan formellt? Vilket språk accepterar den? Finns det en automat som accepterar språket $\emptyset$?



Formell definition av acceptans

Låt $M = (Q, \Sigma, \delta, q_0, F)$ vara en deterministisk finit automata, och låt $w = w_1 w_2 \cdots w_n$ vara en sträng där varje w_i tillhör alfabetet Σ. Vi säger att M *accepterar* w om det finns en sekvens av tillstånd $r_0, r_1, \ldots, r_n$ i Q så att följande tre villkor uppfylls:

1. $r_0 = q_0$,
2. $\delta(r_i, w_{i+1}) = r_{i+1}$, och
3. r_n tillhör F.

Vi säger att M *accepterar* eller *känner igen* språket L om $L = \{w \mid M \text{ accepterar } w\}$.

Fråga 4

Låt $M = (Q, \Sigma, \delta, q_0, F)$, där $Q = \{q_0, q_1\}$, $\Sigma = \{\mathtt{1}, \mathtt{0}\}$, $F = \{q_0\}$, och δ är som i tabellen nedan vara en DFA. Visa att M accepterar strängen $w = \mathtt{1011010}$ enligt den formella definition.

$$\delta(q_0, 0) = q_0$$
$$\delta(q_0, 1) = q_1$$
$$\delta(q_1, 0) = q_1$$
$$\delta(q_1, 1) = q_0$$

Reguljära språk

Ett språk sägs vara *reguljärt* om det finns en finit automata som accepterar det.

- Är alla språk reguljära?
- Är språket $\{ww \mid w \in \Sigma^*\}$ reguljärt?
- Hur designar man en automat som känner igen ett givet språk?

Design av finita automater

- En bra metod är att tänka sig hur man skulle gjort om man själv var tvungen att lösa problemet, och bara kunde minnas det som stod uppskrivet på några få papperslappar.
- Om man förstår vilka lappar man behöver, och hur man använder dom, då har man löst uppgiften.
- Hur ser en automat ut som känner igen språket "alla strängar över `0` och `1` som innehåller ett jämt antal ettor och inte slutar på en nolla"? $\{\epsilon, 011, 101, 1100, 1001, 0101, 11011, \ldots\}$

Reguljära operationer

Låt A och B vara (reguljära) språk. Vi definierar de reguljära operationerna *union*, *konkatenation*, and *stjärna* som följer:

- **Union:** $A \cup B = \{x \mid x \in A \text{ or } x \in B\}$.
- **Konkatenation:** $A \circ B = \{xy \mid x \in A \text{ and } y \in B\}$.
- **Stjärna:** $A^* = \{x_1 x_2 \cdots x_k \mid k \geq 0 \text{ and each } x_i \in A\}$.

Funderingar...

Låt A vara {banana, fanna, foo}, låt B vara {phazer, tazer, lazer}, och låt C vara {gun}. Vad är då $(A^* \circ C) \cup (B^* \circ C)$? Är det samma som $(A \cup B)^* \circ C$?

Om A är $\{a\}$ och B är $\{b, \epsilon\}$, i vilka av mänderna A, B, $A \cup B$, $A \circ B$, A^*, och B^* ingår då ϵ?

Union

Teorem 1.25 Klassen av reguljära språk är sluten under union.

Bevis (skiss) Låt A_1 och A_2 vara reguljära språk, vilket innebär att det finns automater M_1 och M_2 som accepterar A_1 respektive A_2. Vi ska visa att även $A_1 \cup A_2$ är reguljärt, och enklast är att göra det genom att konstruera en automat M som accepterar $A_1 \cup A_2$. Då en finit automat bara får läsa sin input sträng en gång kan vi inte lösa problemet genom att först låta M simulera A_1 och därefter A_2. Istället måste M simulera både M_1 och M_2 samtidigt. Det kan göras om M's tillstånd är den cartesiska produkten av M_1's och M_2's tillstånd.

Union

Bevis Låt $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$ och $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ vara dfa:er sådana att $L(M_1) = A_1$ och $L(M_2) = A_2$.

Vi konstruerar $M = (Q, \Sigma, \delta, q_0, F)$ som accepterar M, där

- $Q = \{(r_1, r_2) \mid r_1 \in Q_1 \text{ och } r_2 \in Q_2\}$.
- Σ är som i M_1 och M_2.
- δ är definierad som följer. För varje (r_1, r_2) i Q och a i Σ låter vi

$$\delta((r_1, r_2), a) = (\delta_1(r_1, a), \delta_2(r_2, a)) \ .$$

- q_0 är (q_1, q_2).
- F är $\{(r_1, r_2) \mid r_1 \in F_1 \text{ eller } r_2 \in F_2\}$.

Union

Vilken automat accepterar enligt beviset unionen av dessa automaters språk? Finns det en enklare automat som accepterar samma språk?