Prednáška 2 - Konečné automaty

Ing. Viliam Hromada, PhD.

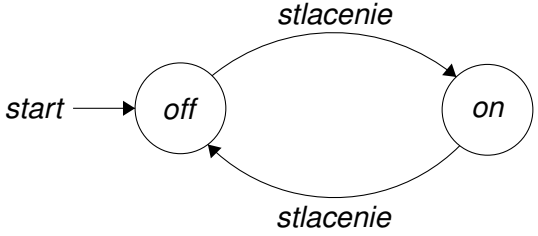
C-510
Ústav informatiky a matematiky
FEI STU

viliam.hromada@stuba.sk

29.9.2020

Neformálna diskusia o konečných automatoch

- Na úvod si povedzme neformálne a motivačne niečo o konečných automatoch...
- Konečný automat je matematický model zariadenia, ktoré sa v každom časovom momente nachádza v tzv. stave, pričom všetkých stavov, v ktorých sa môže nachádzať je konečne veľa.
- To znamená, disponuje množinou týchto stavov.
- Taktiež je riadený logikou, ktorá hovorí, ako sa medzi týmito stavmi prepína, v závislosti od externých "vstupov".

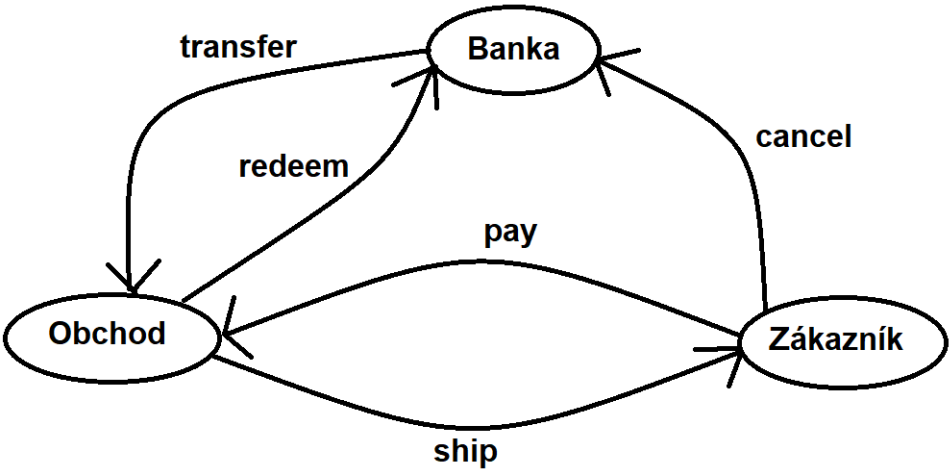


Motivačný príklad

- V motivačnom príklade si ukážeme, ako vieme použiť konečné automaty na modelovanie nejakého požadovaného systému,
- a následne, ako analýzou tohto modelu vieme odhaliť, či má náš systém nejaké bugy, či v ňom dochádza k neželanému správaniu.
- Pokúsime sa modelovať elektronické peniaze.

- V schéme elektronických peňazí budú vystupovať 3 subjekty:
  1. Zákazník
  2. Obchod
  3. Banka
- Pre jednoduchosť predpokladajme, že v obehu je 1 "peniaz".

- Zákazník môže:
  - Platiť (*pay*), t.j. pošle svoj e-peniaz obchodu.
  - Zrušiť (*cancel*) svoj e-peniaz, t.j. pošle ho do banky a tá mu hodnotu pripíše na bežný účet, pričom samotný e-peniaz zneplatní.
- Obchod môže:
  - Zaslať (*ship*) tovar zákazníkovi.
  - Požiadať o výmenu (*redeem*) e-peniazu v banke, t.j. pošle ho do banky s požiadavkou, aby ho banka prepísala na obchod.
- Banka môže:
  - Presunúť (*transfer*) peniaze obchodu tým, že vytvorí nový e-peniaz viazaný na obchod a pošle mu ho.


transfer
Banka
cancel
redeem
pay
Obchod
Zákazník
ship

Samozrejme by sme chceli, aby v našom modeli nemohlo dochádzať k neželaným situáciám, napríklad:

- Zákazník sa môže pokúsiť urobiť kópiu e-peniazu
- Zákazník sa môže pokúsiť minúť ten istý e-peniaz viackrát
- Zákazník sa môže pokúsiť minúť a zároveň zrušiť ten istý e-peniaz
- Banka nesmie prijať na výmenu ten istý e-peniaz od rôznych obchodov
- Banka nesmie dovoliť, aby sa ten istý peniaz aj zrušil zákazníkom, aj zaslal z obchodu na výmenu
- Obchod by nemal odoslať tovar predtým než sa uistí, že zaň dostal riadne zaplatené.
- atď.

- Každého aktéra môžeme modelovať pomocou konečného automatu.
- Každý automat bude mať stavy, pomocou ktorých si bude "pamätať", čo daný aktér už vykonal a zároveň indikovať, čo ešte môže vykonať.
- Budeme uvažovať, že môže nastať 5 spomínaných akcií (pay, cancel, ship, redeem, transfer)
- Na ďalšom slajde je ukážka, ako by mohli vyzerať konečné automaty popisujúce možné správanie 3 aktérov **vzhľadom na 1 konkrétny e-peniaz**.
- Pri každom aktérovi uvažujeme len tie akcie, ktoré sa ho dotýkajú.

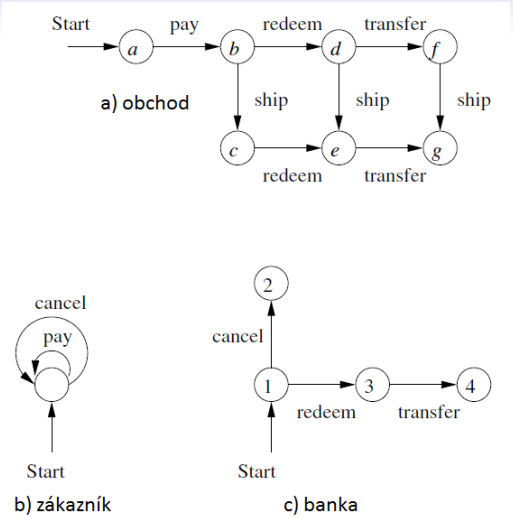


Figure: Tri nezávislé automaty pre aktérov [1]

- Niektoré akcie v automatoch chýbajú - napríklad akcia *cancel* na obchod nemá vplyv, preto v automate pre obchod chýba. Ak by k nej došlo, tak automat obchodu na ňu nevie reagovať a **"zasekol" by sa**.
- Podobne, ak by napríklad zákazník vykonal akciu *pay* dvakrát (čo podľa jeho automatu momentálne môže) a obchod by bol napríklad v stave *e*, tak by automat pre obchod nevedel, čo má robiť a taktiež by sa "zasekol".
- Preto je niekedy žiadúce, aby sme v automate definovali aj akcie, ktoré síce nemajú na automat žiaden vplyv, ale je potrebné, aby ich automat "zaregistroval", t.j. aby mal na ne definované prechody.

Akcie, ktoré budeme ignorovať:

1. Irelevantné akcie vzhľadom na účastníka:
   - *Cancel* pre obchod
   - *Pay, ship* pre banku
   - *Redeem, ship, transfer* pre zákazníka
2. Akcie, ktoré nesmú zaseknúť automat:
   - *Pay* pre obchod v inom stave, než *a*.
   - *Cancel* v banke v stavoch 3 a 4 (aby nebola snaha o zrušenie peniazu po jeho redeem-e)
   - Podobne *redeem* v stavoch 3 a 4.

Ignorovanie akcie bude automat realizovať slučkou v aktuálnom stave, v ktorom sa nachádza.

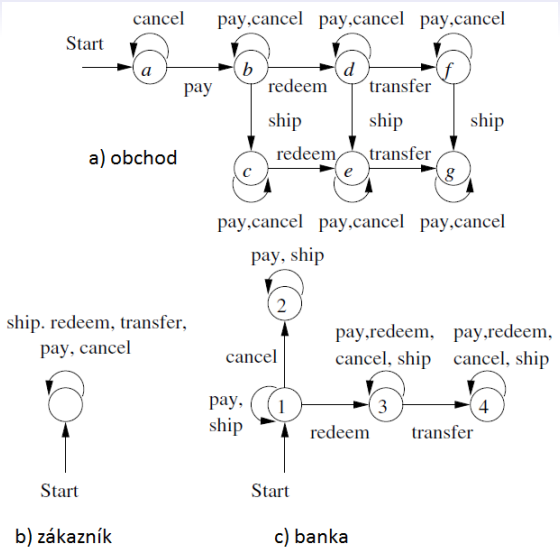


Figure: Tri nezávislé automaty doplnené o irelevantné akcie [1]

- Máme teda 3 nezávislé automaty modelujúce všetkých aktérov vzhľadom na dané akcie.
- Potrebujeme však vytvoriť jeden veľký model (automat), ktorý by v sebe zahŕňal všetkých aktérov.
- Štandardne sa konštruuje tzv. **súčinový automat**, ktorý vlastne vznikne ako kombinácia (kartézsky súčin) všetkých stavov všetkých automatov.
- Automat zákazníka má 1 stav, obchodu 7 stavov, banky 4 stavy, dokopy teda dostaneme automat s $1 * 7 * 4 = 28$ stavmi.

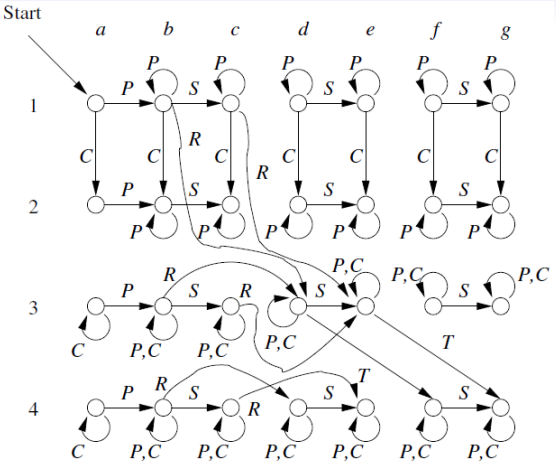


Figure: Výsledný automat pre celý e-peňažný model [1]

- Keď máme teraz model nášho e-peňažného systému pomocou konečného automatu, vieme ho použiť na verifikáciu, či v ňom nedochádza k neželaným situáciám.
- Na prvý pohľad vidíme, že niektoré stavy, napríklad $(2, e)$ alebo $(4, d)$ nie sú dosiahnuteľné zo stavu, v ktorom je automat na začiatku.
- Celkovo je v ňom len 10 dosiahnuteľných stavov.
- Dôležitejšia je však práve analýza neželaných situácií.

Napríklad:

- Je možné, aby obchod odoslal tovar bez toho, že by dostal zaplatené? Teda nastal *ship*, ale nenastal *transfer*?
- Niektoré stavy sú ok, napríklad existuje cesta do stavu $(3, e)$, kde tovar bol už odoslaný (*ship*) a ďalej je možné, že banka vykoná *transfer* do stavu $(4, g)$, čiže obchod môže vidieť svoje peniaze...
- V automate je však možné aj dostať sa do stavu $(2, c)$, napríklad postupnosťou akcií *pay, ship, cancel*. Tovar teda bol zaslaný, avšak obchod ešte nestihol vykonať akciu *redeem* a zákazník naopak, vykonal akciu *cancel*. Preto ak sa obchod pokúsi vykonať *redeem*, tak nebude môcť.

- Preto napríklad analýzou konečných automatov je možné odhaľovať bugy v návrhoch systémov.
- Povedzme si teraz niečo odborného a formálneho o konečných automatoch. Začnime definíciou deterministického konečného automatu.

Deterministický konečný automat

Deterministický konečný automat (DKA) A je pätica $(Q, \Sigma, \delta, q_0, F)$:

1. Konečná množina stavov Q
2. Konečná množina vstupných symbolov Σ
3. Prechodová funkcia $\delta, \delta : Q \times \Sigma \to Q$, ktorá každej dvojici (stav, vstupný symbol) pridelí jeden stav
4. Počiatočný stav q_0, $q_0 \in Q$
5. Množina akceptačných (finálnych) stavov F, $F \subseteq Q$.

Spracovanie vstupu DKA

- Ako sme už naznačili, (D)KA spracúvajú nejaké vstupy.
- DKA podľa definície navyše dokážu tieto vstupy "akceptovať" alebo "neakceptovať".
- Ide o to, či daný DKA po prečítaní tohto vstupu skončí v nejakom špeciálnom stave alebo nie.

Spracovanie vstupu DKA

- Uvažujme, že $a_1 a_2 ... a_n$ je postupnosť vstupných symbolov, ktorú spracujeme nejakým DKA.
- Každý DKA funguje tak, že číta vstupný reťazec **symbol po symbole**, pričom po prečítaní symbolu sa posúva na ďalší nasledovný symbol a **nevie sa vrátiť späť k už predtým prečítaným symbolom**.
- Na začiatku je DKA v stave q_0. Aby sme zistili, ako zareaguje tento DKA na prvý vstupný symbol a_1, pozrieme sa, aká je hodnota prechodovej funkcie $\delta(q_0, a_1)$. Nech $\delta(q_0, a_1) = q_1$.
- To znamená, že DKA sa po prečítaní symbolu a_1 zo stavu q_0 prepne do stavu q_1.

Spracovanie vstupu DKA

- Teraz sa DKA nachádza v stave q_1 a na vstupe sa na spracovanie nachádza ďalší symbol vstupného reťazca - a_2.
- Znovu sa pozrieme, čo hovorí prechodová funkcia - nech platí, že $\delta(q_1, a_2) = q_2$.
- Automat sa teda nachádza v stave q_1 a po prečítaní symbolu a_2 sa z neho prepne do stavu q_2.
- Automat pokračuje ďalej v čítaní vstupných symbolov a prepínaní sa medzi stavmi podľa prechodovej funkcie, t.j. prepína sa medzi stavmi $q_3, q_4, ..., q_n$ podľa $\delta(q_{i-1}, a_i) = q_i$ pre každé i.
- Ak po prečítaní **celého vstupu** skončí v stave q_n a platí, že $q_n \in F$, t.j. je to tzv. **akceptačný stav**, tak hovoríme, že automat spracovaný reťazec $a_1 a_2 ... a_n$ **akceptuje**.
- Ak stav $q_n \notin F$, t.j. **nie je akceptačný**, tak potom hovoríme, že automat reťazec $a_1 a_2 ... a_n$ **neakceptuje**.

Príklad

Nájdite DKA, ktorý vie rozpoznať (akceptovať) také reťazce zo symbolov $\{0, 1\}$, ktoré v sebe obsahujú ako podreťazec 01.

- Označme množinu všetkých takýchto reťazcov L. Vieme ju popísať napríklad ako:

  $\{w \mid w$ je tvaru $x01y$ pre nejaké reťazce x a y pozostávajúce z núl a jednotiek $\}$

- Prípadne:

  $\{x01y \mid x$ a y sú ľubovoľné reťazce z núl a jednotliek $\}$
- A pre fajnšmekrov: $L = \{x01y \mid x \in \{0, 1\}^*, y \in \{0, 1\}^*\}$
- Príklady takýchto reťazcov sú: $01, 001, 011, 0010, 0011, 1011, 1010, ...$
- Príklady reťazcov, ktoré tento tvar nemajú: $\varepsilon, 0, 1, 00, 10, 11, 1111, ...$

- Takýto DKA teda bude čítať reťazce zložené z núl a jednotiek. Preto vstupná abeceda takéhoto DKA bude $\Sigma = \{0, 1\}$.
- Taktiež sa bude prepínať medzi nejakými stavmi, pričom na začiatku bude v stave q_0.
- Musíme teraz vymyslieť, ako sa bude správať, aby sme s istotou vedeli povedať, že ak v čítanom reťazci narazil na postupnosť 01, tak potom takýto reťazec **musí** akceptovať, t.j. po jeho dočítaní skončiť v jednom z akceptačných stavov.

Počas čítania vstupného reťazca si teda musí DKA pamätať:

1. Videl už niekde v čítanom vstupe postupnosť 01? Ak áno, to znamená, že čokoľvek, čo ešte číta už nezmení fakt, že v danom vstupe sa táto postupnosť nachádza a preto by od momentu, keď prečíta 01 mal už zostávať len v akceptačnom stave.
2. Nevidel síce ešte postupnosť 01, ale práve prečítal symbol 0 - to znamená, že ak ďalší symbol bude 1, tak prečíta 01 a všetko ostatné už bude akceptovať.
3. Nevidel ešte postupnosť 01 a posledný vstup bol alebo 1, alebo je na začiatku. V takom prípade musí naraziť najprv na 0 a vzápätí na 1, aby mohol vstup akceptovať.

Práve tieto 3 situácie nám indikujú, že DKA bude mať 3 stavy, ktoré budú v podstate realizáciou týchto 3 situácií.

- Počiatočný stav q_0 bude reprezentovať bod číslo 3. Práve sme začali, t.j. ešte sme určite nevideli postupnosť 01. A taktiež, ak sme v stave q_0 a čítame jednotku, tak sme sa nepriblížili k tomu, aby bola vo vstupe niekde 01 a preto musíme zostať v q_0, t.j. $\delta(q_0, 1) = q_0$.
- Ak sme v stave q_0 a prečítame nulu, dostávame sa do situácie z bodu číslo 2, čiže je možné, že z hľadanej časti 01 sme práve prečítali nulu. Preto sa posunieme do nového stavu - povedzme q_1 a teda $\delta(q_0, 0) = q_1$.
- Ak sme v stave q_1, práve sme prečítali jednu nulu. Ak teraz znovu prečítame nulu, tak to znamená, že znovu sme možno prečítali prvú nulu z hľadanej postupnosti 01. Preto $\delta(q_1, 0) = q_1$. Ak teraz prečítame jednotku, tak to znamená, že sme práve prečítali po sebe idúce 01 a presunieme sa do nového stavu q_2, ktorý reprezentuje situáciu z bodu číslo 1 z predchádzajúceho slajdu, t.j. $\delta(q_1, 1) = q_2$.

- Ak je DKA v stave q_2, to znamená, že niekde vo vstupe prečítal časť 01. Čiže nech už bude ďalej na vstupe čokoľvek, bude to len celé súčasť jedného veľkého reťazca, v ktorom sa určite nachádzala postupnosť 01.
- Preto už DKA zotrvá v stave q_2, teda $\delta(q_2, 0) = \delta(q_2, 1) = q_2$.
- Spolu teda bude mať tento automat 3 stavy: $Q = \{q_0, q_1, q_2\}$.
- Akceptačný stav bude stav q_2, t.j. ten, do ktorého sa dostaneme v prípade, že vstup spĺňa požadovanú vlastnosť, t.j. $F = \{q_2\}$.

Výsledný automat, ktorý akceptuje reťazce núl a jednotiek, ktoré obsahujú ako podreťazec 01 by teda bol definovaný ako pätica
$A = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_2\})$, kde:

- $\delta(q_0, 0) = q_1$
- $\delta(q_0, 1) = q_0$
- $\delta(q_1, 0) = q_1$
- $\delta(q_1, 1) = q_2$
- $\delta(q_2, 0) = q_2$
- $\delta(q_2, 1) = q_2$

Iný zápis DKA

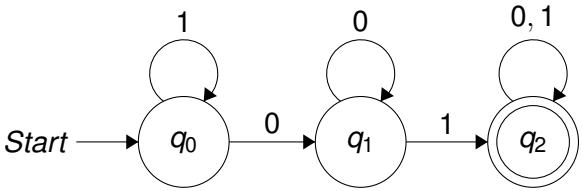
Existuje viacero spôsobov, ako popísať nejaký DKA. Okrem spomínanej pätice a vymenovania prechodovej funkcie existujú minimálne ďalšie dva:

- Prechodový diagram
- Prechodová tabuľka

Prechodový diagram

Pri popise DKA $A = (Q, \Sigma, \delta, q_0, F)$ pomocou prechodového diagramu postupujeme nasledovne

1. Pre každý stav z Q je v diagrame samostatný vrchol označený menom stavu.
2. Nech pre stav $q \in Q$ a vstupný symbol $a \in \Sigma$ je podľa prechodovej funkcie $\delta(q, a) = p$. Potom v prechodovom diagrame je vedená orientovaná hrana z vrcholu q do vrcholu p označená symbolom a. Ak je v automate viacero prechodov z q do p na rôzne vstupné symboly, je možné v diagrame použiť len jednu hranu z q do p označenú zoznamom týchto symbolov.
3. Do vrcholu počiatočného stavu q_0 vstupuje hrana označená ako *Start*. Táto hrana nevychádza zo žiadneho vrchola.
4. Vrcholy prislúchajúce akceptačným stavom sú označené dvojitou kružnicou. Vrcholy neakceptujúcich stavov len obyčajnou kružnicou.


1
0
0, 1
Start
q_0
0
q_1
1
q_2

Prechodová tabuľka

Pri popise DKA $A = (Q, \Sigma, \delta, q_0, F)$ pomocou prechodovej tabuľky postupujeme nasledovne

1. Záhlavia riadkov v tabuľke predstavujú jednotlivé stavy.
2. Záhlavia stĺpcov v tabuľke predstavujú jednotlivé vstupné symboly.
3. Riadok prislúchajúci počiatočnému stavu q_0 je označený šípkou.
4. Riadky prislúchajúce akceptačným stavom sú označené hviezdičkou.
5. Ak je v automate prechod zo stavu q do stavu p pre symbol a, t.j. $\delta(q, a) = p$, tak v prechodovej tabuľke je na pozícii: riadok q stĺpec a vpísaný stav p.

	0	1
$\rightarrow q_0$	q_1	q_0
q_1	q_1	q_2
$*q_2$	q_2	q_2

Rozšírená prechodová funkcia

- Z toho, čo sme si povedali, by už malo byť zrejmé, že pre každý DKA môžeme uvažovať, aké reťazce akceptuje / neakceptuje.
- Pre každý reťazec totižto vieme uvažovať, či sa automat po jeho spracovaní ocitne v nejakom akceptačnom stave alebo nie.
- Aby sme mali formálne zadefinované správanie DKA pre spracovanie reťazca, zadefinujme si tzv. *rozšírenú prechodovú funkciu.*
- Prechodová funkcia δ totižto popisuje správanie sa DKA len pre 1 vstupný symbol.
- *Rozšírená prechodová funkcia* $\hat{\delta}$ bude popisovať správanie sa DKA pre reťazec vstupných symbolov.

Rozšírená prechodová funkcia

- Rozšírená prechodová funkcia $\hat{\delta}$ je funkcia, ktorá pre stav q a reťazec w vráti stav p, do ktorého sa dostane DKA, ak začne v stave q a spracuje vstupný reťazec w. Definujeme ju indukciou na dĺžke vstupného reťazca w:

1. **Základ:** $\hat{\delta}(q, \varepsilon) = q$. T.j. ak je DKA v stave q a na vstupe nič nie je, ostane v stave q.
2. **Indukcia:** Nech w je reťazec v tvare $w = xa$, a je posledný symbol w a x je reťazec, ktorý vznikne z w odstránením posledného symbolu a. Potom

$$\hat{\delta}(q, w) = \delta(\hat{\delta}(q, x), a)$$

Rozšírená prechodová funkcia

- Indukčná časť definície nehovorí nič iné, než len to, že na to, aby sme zistili, kam sa dostane DKA zo stavu q po prečítaní slova w:
  1. Najprv zistíme, do akého stavu sa dostane zo stavu q po prečítaní prefixu x slova w, t.j. $\hat{\delta}(q, x)$ - nech napríklad $\hat{\delta}(q, x) = p$
  2. A následne zistíme, kam sa z tohto stavu p dostaneme po spracovaní posledného symbolu a slova w, t.j. $\delta(p, a)$.
  3. Dokopy nám to logicky povie, kam sa dostane DKA pre celý reťazec w zo stavu q, t.j. $\hat{\delta}(q, w)$.

Príklad

Navrhnite DKA, ktorý bude akceptovať nasledovnú množinu reťazcov:

$$L = \{w \mid w \text{ má párny počet núl a párny počet jednotiek } \}$$

- Je jasné, že DKA si bude musieť vedieť nejakým spôsobom počítať / pamätať, či bol doteraz prečítaný počet núl párny / nepárny, a taktiež, či bol doteraz prečítaný počet jednotiek párny / nepárny.
- Pomôže nám fakt, že ak je nejaké číslo párne, tak číslo o 1 väčšie je nepárne a naopak, ak je nejaké číslo nepárne, tak číslo o 1 väčšie je párne.

Príklad

Keďže potrebujeme rozlišovať, či sme doteraz videli párny/nepárny počet jednotiek a párny/nepárny počet núl, potrebujeme dokopy minimálne 4 stavy:

1. q_0: V tomto stave bude DKA, ak doteraz prečítal párny počet núl a párny počet jednotiek. Tento stav je zároveň aj počiatočným stavom, keďže na začiatku ešte nebolo prečítané nič a nula je párne číslo. A taktiež bude akceptačným stavom, lebo v ňom skončíme práve vtedy, keď doteraz prečítaný reťazec spĺňa požadované podmienky.
2. q_1: V tomto stave bude DKA, ak doteraz prečítal párny počet núl a nepárny počet jednotiek.
3. q_2: V tomto stave bude DKA, ak doteraz prečítal nepárny počet núl a párny počet jednotiek.
4. q_3: V tomto stave bude DKA, ak doteraz prečítal nepárny počet núl a nepárny počet jednotiek.

Príklad

Prechodová funkcia automatu by vyzerala nasledovne:

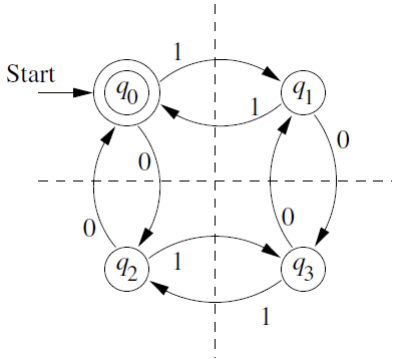


Figure: Výsledné riešenie [1]

Príklad

Ukážme si ešte na príklade (detailný) popis výpočtu automatu pre vstup 110101 v zmysle definície rozšírenej prechodovej funkcie:

1. $\hat{\delta}(q_0, \varepsilon) = q_0$.
2. $\hat{\delta}(q_0, 1) = \delta(\hat{\delta}(q_0, \varepsilon), 1) = \delta(q_0, 1) = q_1$.
3. $\hat{\delta}(q_0, 11) = \delta(\hat{\delta}(q_0, 1), 1) = \delta(q_1, 1) = q_0$.
4. $\hat{\delta}(q_0, 110) = \delta(\hat{\delta}(q_0, 11), 0) = \delta(q_0, 0) = q_2$.
5. $\hat{\delta}(q_0, 1101) = \delta(\hat{\delta}(q_0, 110), 1) = \delta(q_2, 1) = q_3$.
6. $\hat{\delta}(q_0, 11010) = \delta(\hat{\delta}(q_0, 1101), 0) = \delta(q_3, 0) = q_1$.
7. $\hat{\delta}(q_0, 110101) = \delta(\hat{\delta}(q_0, 11010), 1) = \delta(q_1, 1) = q_0$.

Jazyk DKA

Každý DKA rozpoznáva (akceptuje) nejakú množinu reťazcov nad svojou vstupnou abecedou.

Definícia

*Nech DKA $A = (Q, \Sigma, \delta, q_0, F)$. **Jazyk akceptovaný** deterministickým konečným automatom je množina označená $L(A)$ a definovaná:*

$$L(A) = \{w \mid \hat{\delta}(q_0, w) \in F\}$$

To znamená, jazyk DKA tvoria také reťazce, pre ktoré sa DKA po ich **celom prečítaní** dostane z počiatočného stavu q_0 do niektorého zo svojich akceptačných stavov.

Každý jazyk L akceptovaný nejakým deterministickým konečným automatom, t.j. ak $L = L(A)$ pre nejaký DKA A, nazývame **regulárnym jazykom**.

Použitá literatúra

1 Hopcroft, Motwani, Ullman - Introduction to Automata Theory, Languages and Computations, 3rd Ed.