Prednáška 2 - Regulárne jazyky, konečné automaty

Ing. Viliam Hromada, PhD.

C-510
Ústav informatiky a matematiky
FEI STU

viliam.hromada@stuba.sk

Regulárne jazyky

- Pre pripomenutie, jazyk je **regulárny**, ak ho generuje **regulárna gramatika.**
  - $G = (N, T, P, S)$ je gramatika.
  - $A \to xB$,
  - $A \to w$,
  - $A, B \in N, x \in T^+, w \in T^*$.
  - G je regulárna.
- Hoci sa javia obmedzene, postačujú na popis základných symbolov programovacích jazykov - identifikátory, konštanty, kľúčové slová, separátory.
- Každý **konečný** jazyk je **regulárny** (prečo?).

Príklad: $G_1 = (\{S, A, B\}, \{f, i, n, t\}, P, S)$.

$$S \to iA$$
$$A \to f \mid nB$$
$$B \to t$$

$L(G_1) = \{if, int\}$.

Ekvivalentne: $G_1' = (\{S\}, \{f, i, n, t\}, P, S)$.

$$S \rightarrow if$$

$$S \rightarrow int$$

$L(G_1') = \{if, int\} = L(G_1)$.

Príklad: $G_2 = (\{S, A, B\}, \{+, -, 0, 1, 2, ..., 9\}, P, S)$.

$$S \to +A \mid -A \mid 1B \mid 2B \mid ... \mid 0 \mid 1 \mid ... \mid 9$$
$$A \to 1B \mid 2B \mid ... \mid 0 \mid 1 \mid ... \mid 9$$
$$B \to 0B \mid 1B \mid ... \mid 0 \mid 1 \mid ... \mid 9$$

$L(G_2)$ tvoria všetky celočíselné konštanty (prípadne so znamienkom) bez bezvýznamných núl zľava.

Generatívna špecifikácia jazyka

- Gramatiky predstavujú **generatívny** spôsob špecifikácie jazykov.
- Pre ľubovoľný reťazec x nad nejakou abecedou A, $x \in A^*$, vieme uvažovať, či ho gramatika G generuje alebo nie, teda či patrí do jazyka generovaného gramatikou, $x \in L(G)$, alebo nie, $x \notin L(G)$.
- Ak teda vieme pre jazyk $L \subseteq A^*$ nájsť gramatiku G takú, že $L = L(G)$, podarilo sa nám špecifikovať jazyk L generatívne pomocou nejakej gramatiky G.
- Pre každý reťazec $x \in L$ potom bude existovať derivácia v G, $x \in L(G)$ a zároveň každý reťazec $x \notin L$ nebude mať deriváciu v G, $x \notin L(G)$.

Akceptačná špecifikácia jazyka

- Alternatívne môžeme uvažovať nasledovný spôsob špecifikácie jazyka.
- Uvažujme nejaké výpočtové zariadenie, ktorého vstup môže byť ľubovoľný reťazec.
- Toto zariadenie tento reťazec spracuje (napr. prečíta každý jeho symbol) a v konečnom čase jeho výstup predstavuje jedna z dvoch hodnôt: **akceptujem** / **neakceptujem**.
- Na základe tohto výstupu zariadenia potom hovoríme, že **akceptuje** / **neakceptuje** vstupný reťazec.
- Takéto zariadenie teda má nejakú množinu reťazcov, pre ktoré vráti odpoveď **akceptujem** a pre ostatné reťazce vracia odpoveď **neakceptujem**.

Deterministické konečné automaty

- Základným výpočtovým zariadením, ktoré dokáže akceptovať/neakceptovať vstupné reťazce je tzv. **konečný automat**.
- Konečné automaty predstavujú jeden z **akceptačných** spôsobov špecifikácie jazykov.

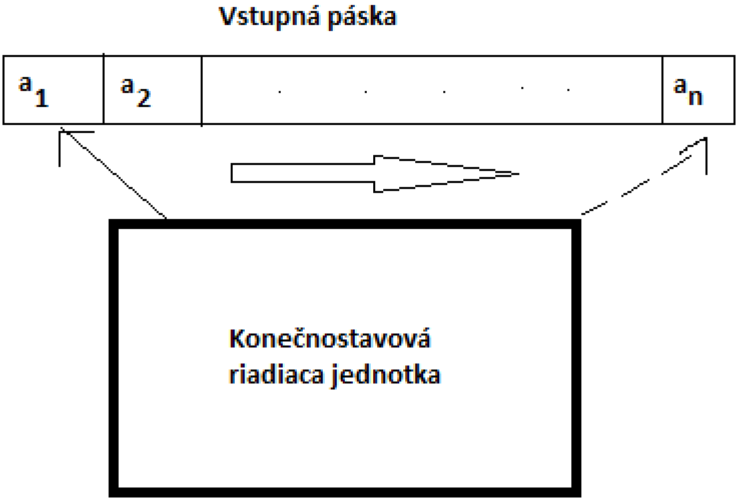
Deterministické konečné automaty

Konečný automat ako výpočtové zariadenie obsahuje viacero obmedzení:

- Vstupný reťazec dokáže spracovať (čítať) od prvého po posledný symbol, pričom po každom prečítanom symbole sa presúva na ďalší a k skôr prečítaným symbolom sa nevie vrátiť.
- Konečný automat nemá k dispozícii žiadnu *pamäť*, do ktorej by vedel ukladať nejaké dáta (na rozdiel od napr. bežného počítača).
- Konečný automat dáta zo vstupu len číta, nevie ich prepisovať.

Konečný automat teda číta vstupný reťazec symbol za symbolom, pričom sa po každom prečítanom symbole vstupu vždy nachádza v nejakom stave, pričom všetkých možných stavov je konečne veľa. Tieto stavy predstavujú jediný **"typ pamäte"** KA.

High-tech obrázok KA



Definícia

Deterministický konečný automat *(DKA) je usporiadaná pätica* $M = (Q, \Sigma, \delta, q_0, F)$*, kde:*

- *Q je konečná množina stavov automatu,*
- *Σ je konečná množina prípustných vstupných symbolov automatu,*
- *δ je prechodová funkcia automatu, $\delta : Q \times \Sigma \to Q$,*
- *q_0 je počiatočný stav automatu, $q_0 \in Q$,*
- *F je množina akceptačných stavov automatu, $F \subseteq Q$.*

Konečný automat (ďalej len KA)

- Vstupom KA je páska so vstupným slovom, zloženého zo symbolov Σ.
- Automat začína svoju činnosť v stave q_0.
- Automat vždy "vidí" jeden symbol zo vstupnej pásky. Ten prečíta svojou **čítacou hlavou** a na základe aktuálneho stavu a hodnoty symbolu podľa prechodovej funkcie prejde do nasledujúceho stavu.
- Následne sa posúva čítacia hlava na ďalší symbol vstupnej pásky. Posun pri čítaní späť na predchádzajúce symboly vstupného slova **nie je možný**.
- Ak automat po prečítaní celého vstupného slova skončí v akceptačnom stave, vstupné slovo **akceptuje**. Inak slovo **neakceptuje**.

- Na popis činnosti, resp. spracovania nejakého vstupu konečným automatom používame tzv. konfigurácie.
- Konfigurácia KA je spôsob zachytenia informácie o tom, v akom **stave sa aktuálne** konečný automat nachádza a aká je **nespracovaná** (neprečítaná) **časť vstupného reťazca**.

Konfigurácia KA

Definícia

Nech $M = (Q, \Sigma, \delta, q_0, F)$ je konečný automat. Potom:

- *usporiadanú dvojicu $(q, w) \in Q \times \Sigma^*$ nazývame **konfiguráciou konečného automatu** M.*
- *Konfiguráciu (q_0, w), kde w je celý vstupný reťazec, nazývame **začiatočnou** konfiguráciou automatu M,*
- *konfiguráciu (q, ε), kde $q \in F$, nazývame **akceptujúcou** konfiguráciou automatu M.*

Konfiguráciu teda definuje **aktuálny stav** automatu a **neprečítaná** časť vstupného slova.

Prechody KA

Definícia

Nech $M = (Q, \Sigma, \delta, q_0, F)$ *je deterministický konečný automat. Potom nad množinou konfigurácií* $Q \times \Sigma^*$ *definujeme* ***reláciu prechodu*** *(krok výpočtu)* $\vdash$ *takto:*

Nech $q, p \in Q, w \in \Sigma^*, a \in \Sigma$. *Potom* $(q, aw) \vdash (p, w)$ *práve vtedy, keď* $\delta(q, a) = p$.

T.j. ak je automat v stave q, na vstupe je neprečítaná časť slova aw, t.j. číta prvý symbol zľava a a jeho prechodová funkcia $\delta(q, a) = p$, potom po prečítaní tohto symbolu prejde do stavu p a na vstupe ostane neprečítaná časť w.

Prechody KA

Môžeme použiť aj symbol $\vdash_M$, ak chceme zdôrazniť, že relácia prechodu sa týka daného konečného automatu M - ak máme napríklad viacero automatov a potrebujeme ich rozlíšiť.

Výpočet KA

Definícia

Nech $M = (Q, \Sigma, \delta, q_0, F)$ *je konečný automat. Potom postupnosť konfigurácií* $(q_0, w_0), (q_1, w_1), \dots, (q_k, w_k)$ *takých, že*

$$(q_0, w_0) \vdash (q_1, w_1) \vdash \dots \vdash (q_k, w_k),$$

$q_i \in Q, w_i \in \Sigma^*, i = 0, 1, \dots, k$ *sa nazýva* ***výpočet konečného automatu*** M *z* (q_0, w_0) *do* (q_k, w_k) *a vyjadrovať ho budeme v tvare* $(q_0, w_0) \vdash^k (q_k, w_k)$.

Číslo k *predstavuje* ***počet krokov výpočtu***.

Výpočet KA - nešpecifikovaný počet krokov

Definícia

Nech $M = (Q, \Sigma, \delta, q_0, F)$ je konečný automat. Potom:

- ***tranzitívnym uzáverom relácie krok výpočtu*** *nazývame reláciu $\vdash^+$ definovanú na množine konfigurácií nasledovne: $(q, w) \vdash^+ (p, x)$ práve vtedy, keď existuje $n \geq 1$ také, že $(q, w) \vdash^n (p, x)$ a*
- ***reflexívnym a tranzitívnym uzáverom relácie krok výpočtu*** *nazývame reláciu $\vdash^*$ definovanú na množine konfigurácií nasledovne: $(q, w) \vdash^* (p, x)$ práve vtedy, keď existuje $n \geq 0$ také, že $(q, w) \vdash^n (p, x)$,*

pričom $p, q \in Q$ a $w, x \in \Sigma^*$.

T.j. pomocou relácií $\vdash^{+}$ a $\vdash^{*}$ vieme vyjadriť výpočet automatu začínajúci v konfigurácii (q, w) a končiaci v (p, x) s nešpecifikovaným počtom krokov.

T.j. že existuje spôsob, akým sa automat dostane zo stavu q a neprečítaným vstupným slovom w do stavu p a neprečítaným vstupným slovom x.

Je zrejmé, že x je prípona slova w.

Jazyk KA

Definícia

Nech $M = (Q, \Sigma, \delta, q_0, F)$ je konečný automat. Potom jazyk $L(M)$ ***rozpoznávaný(akceptovaný)*** *konečným automatom M je množina:*

$$L(M) = \{w \mid (q_0, w) \vdash^* (q, \varepsilon), w \in \Sigma^*, q \in F\}.$$

T.j. jazyk KA je množina slov, pre ktoré sa automat po ich úplnom spracovaní dostane z počiatočného stavu do jedného z akceptačných stavov.

Príklad: Je daný DKA $M = (\{q_0, q_1, q_2\}, \{0, 1\}, \delta, q_0, \{q_1\})$, pričom prechodová funkcia δ:

δ	0	1
q_0	q_1	q_2
q_1	q_1	q_2
q_2	q_2	q_2

Výpočet pre **neakceptovaný** reťazec 001:

$$(q_0, 001) \vdash (q_1, 01) \vdash (q_1, 1) \vdash (q_2, \varepsilon);\ q_2 \notin F$$

Výpočet pre **akceptovaný** reťazec 000:

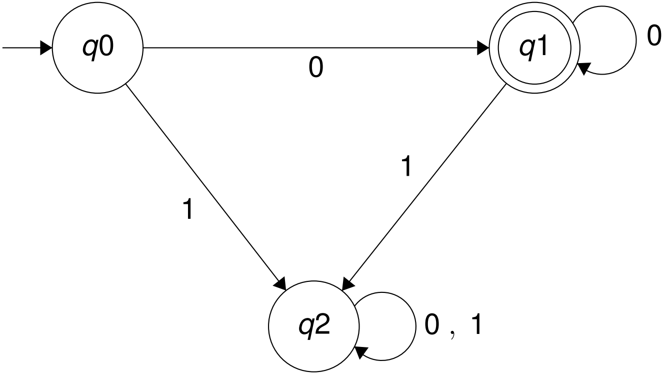
$$(q_0, 000) \vdash (q_1, 00) \vdash (q_1, 0) \vdash (q_1, \varepsilon);\ q_1 \in F$$

$L(M)$ sú reťazce zložené zo samých núl dĺžky aspoň 1.

Grafická reprezentácia KA

KA môžeme reprezentovať **prechodovým diagramom** - orientovaný graf, ktorého vrcholy sú stavy, hrany sú ohodnotené symbolmi vstupnej abecedy a reprezentujú prechodovú funkciu, počiatočný stav je označený vstupujúcou hranou a akceptačné stavy sú označené dvojitým krúžkom.

Grafická reprezentácia KA



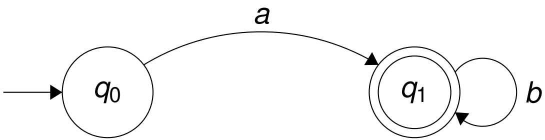
Neúplné DKA

Niekedy je možné, že prechodová funkcia δ nie je úplná, t.j. nie je definovaná pre všetky kombinácie stavov a symbolov vstupnej abecedy.

V grafe by to znamenalo, že z vrcholov by vždy nevychádzalo toľko hrán, koľko je symbolov vstupnej abecedy.

Neúplné DKA

V uvedenom automate nie sú definované prechody: $\delta(q_0, b)$ a $\delta(q_1, a)$:



Reťazce, pre ktoré sa výpočet zasekne v nejakom stave a neboli dočítané do konca, sú **neakceptované**. Napríklad:

- aa neakceptované, pretože: $(q_0, aa) \vdash (q_1, a) \vdash$ ZÁSEK!
- bb neakceptované, pretože $(q_0, bb) \vdash$ ZÁSEK!

Neúplné DKA

Avšak, ak potrebujeme mať úplný DKA, t.j. definovanú prechodovú funkciu pre všetky symboly vstupnej abecedy a pre všetky stavy, nie je problém automat rozšíriť o tzv. **pascu** - nový neakceptujúci stav, ktorý zachytí tie symboly, pre ktoré pôvodne nebola prechodová funkcia definovaná.

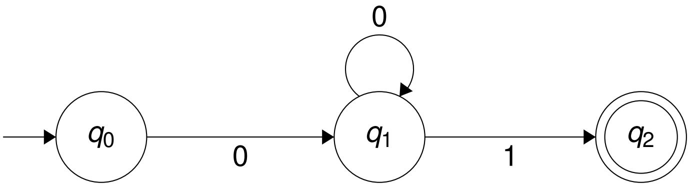
Kompletizácia DKA

Veta

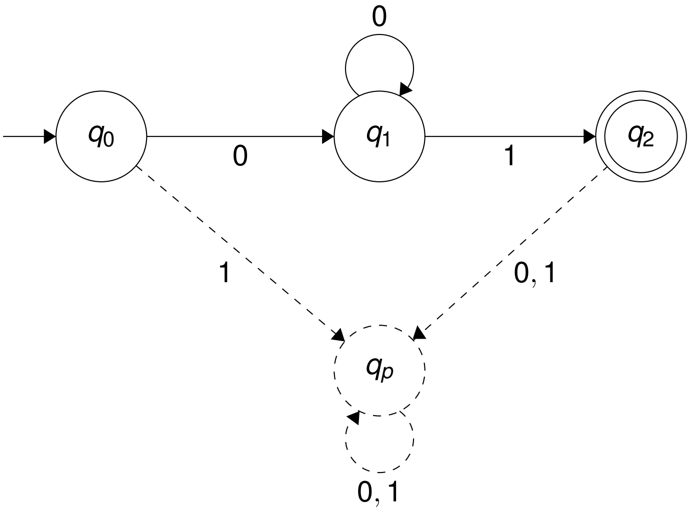
Nech $M = (Q, \Sigma, \delta, q_0, F)$ *je DKA. Potom k nemu možno zostrojiť DKA* $M_K = (Q_K, \Sigma, \delta_K, q_0, F)$ *taký, že* $(q_0, w) \vdash^*_{M_K} (q, \varepsilon)$ *pre každé* $w \in \Sigma^*$ *a navyše* $L(M) = L(M_K)$.

Inými slovami, vždy sa dá doplniť prechodová funkcia na úplnú bez toho, aby sa zmenil jazyk, ktorý automat akceptuje.

Príklad neúplného DKA:



Doplnenie na úplný DKA:



Nedeterministické konečné automaty

- Sú **zovšeobecnením** deterministických konečných automatov.
- Aktuálny stav a symbol na vstupe nemusia jednoznačne určovať nasledujúci stav, ale určujú **množinu** možných nasledujúcich stavov.
- Taktiež môžu vykonávať prechody bez toho, aby prečítali symbol zo vstupu, tzv. ε-prechody.
- Z formálneho hľadiska sa mení len definícia prechodovej funkcie.

Nedeterministické konečné automaty

Definícia

Nedeterministický konečný automat *(NKA) je usporiadaná pätica $M = (Q, \Sigma, \delta, q_0, F)$, kde:*

- *Q je konečná množina stavov automatu,*
- *Σ je konečná množina prípustných vstupných symbolov automatu,*
- *δ je prechodová funkcia automatu, $\delta : Q \times \Sigma_\varepsilon \to 2^Q$, pričom $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$ a 2^Q označuje množinu všetkých podmnožín (t.j. potenčnú množinu) množiny stavov Q,*
- *q_0 je počiatočný stav automatu, $q_0 \in Q$,*
- *F je množina akceptujúcich (koncových) stavov automatu, $F \subseteq Q$.*

UIM

Relácia prechodu NKA

Definícia

Nech $M = (Q, \Sigma, \delta, q_0, F)$ je nedeterministický konečný automat.Potom nad množinou konfigurácií $Q \times \Sigma^*$ definujeme ***reláciu prechodu*** *(krok výpočtu) NKA $\vdash$ takto:*

Nech $q, p \in Q, w \in \Sigma^*, a \in \Sigma_\varepsilon$. Potom $(q, aw) \vdash (p, w)$ práve vtedy, keď $p \in \delta(q, a)$.

Pojmy ako konfigurácia, začiatočná konfigurácia a akceptujúca konfigurácia majú rovnaký zmysel, ako pri DKA.

Akceptácia vs. neakceptácia slova NKA

- NKA akceptuje slovo vtedy, ak **existuje** výpočet NKA taký, že slovo bude akceptované.
- NKA neakceptuje slovo vtedy, ak **neexistuje** akceptujúci výpočet, t.j. pre všetky možné výpočty vzhľadom na vstupné slovo platí:
  - alebo sa automat **zasekne** (t.j. ostane mu nespracovaný vstup, ale nevie sa dostať do nasledujúceho stavu),
  - alebo po spracovaní celého vstupného slova automat neskončí v **akceptačnom stave**.

Príklad: Uvažujme NKA $M = (\{q_0, q_1, q_2, q_f\}, \{+, -, 0, 1, 2, ..., 9\}, \delta, q_0, \{q_f\})$, kde prechodová funkcia δ:

δ	ε	+ | -	0	1 | ... | 9
q_0	$\{q_1\}$	$\{q_1\}$	$\emptyset$	$\emptyset$
q_1	$\emptyset$	$\emptyset$	$\{q_f\}$	$\{q_2, q_f\}$
q_2	$\emptyset$	$\emptyset$	$\{q_2, q_f\}$	$\{q_2, q_f\}$
q_f	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

Symbolom | sú oddelené alternatívne symboly pre jednoduchší zápis.

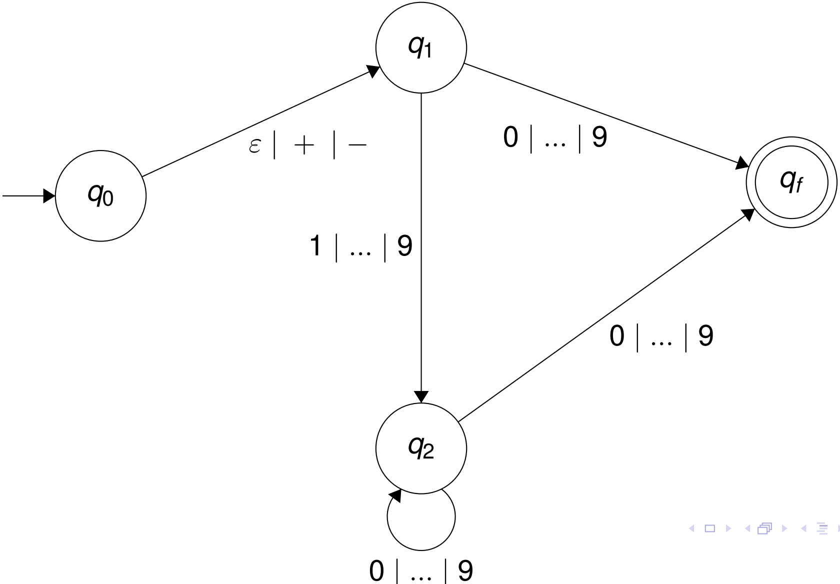
Príklad, pokr. Vstup: $+1$:

$(q_0, +1) \vdash (q_1, +1)$ (použilo sa $\delta(q_0, \varepsilon) = q_1$)

$(q_0, +1) \vdash (q_1, 1) \vdash (q_2, \varepsilon)$

$(q_0, +1) \vdash (q_1, 1) \vdash (q_f, \varepsilon)$

Príklad, pokr.



Vzťah medzi DKA a NKA

- Je jasné, že keďže NKA sú zovšeobecnením DKA, tak každý jazyk, ktorý je akceptovateľný deterministickým konečným automatom, je aj akceptovateľný nedeterministickým konečným automatom.
- Čo je prekvapivejšie je fakt, že to platí aj naopak, t.j. ku každému jazyku, ku ktorému sa dá zostrojiť NKA, ktorý ho akceptuje, sa dá zostrojiť aj DKA, ktorý ho akceptuje.
- NKA a DKA majú teda **rovnakú silu/akceptačnú schopnosť**.

Ekvivalencia NKA a DKA

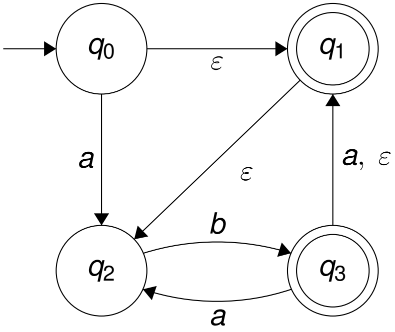
Veta

Nech L je jazyk akceptovaný nejakým NKA. Potom ***existuje*** *DKA, ktorý akceptuje ten istý jazyk L.*

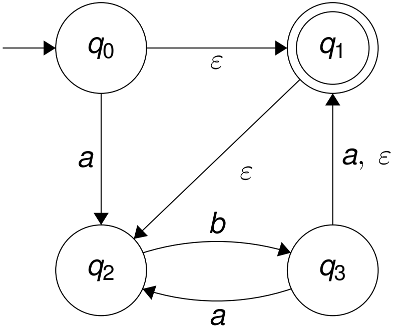
Idea zostrojenia ekvivalentného DKA je v tom, že nový DKA „sleduje “všetky výpočtové cesty, ktorými mohol ísť pôvodný NKA.

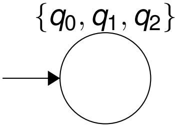
Ekvivalencia NKA a DKA - idea

Nech je daný NKA:

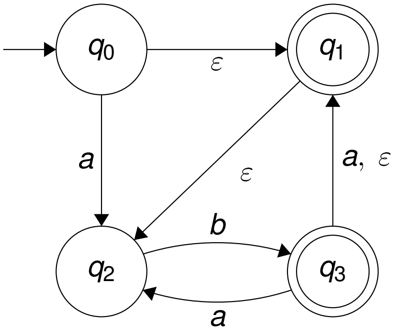


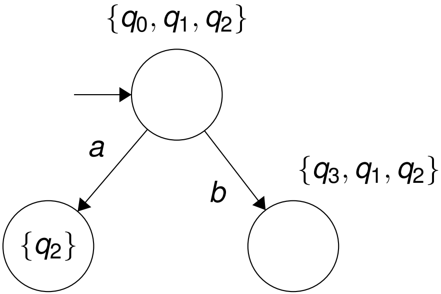
DKA počiatočný stav



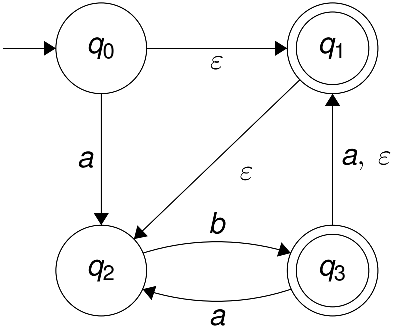


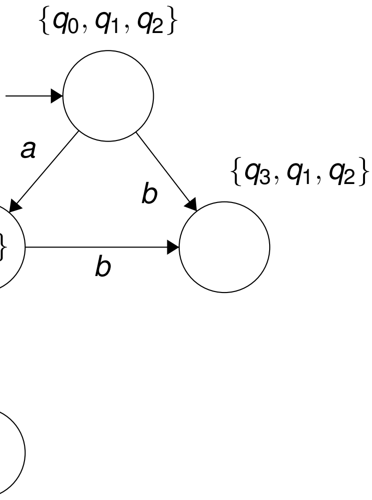
Z počiatočného stavu na a, b



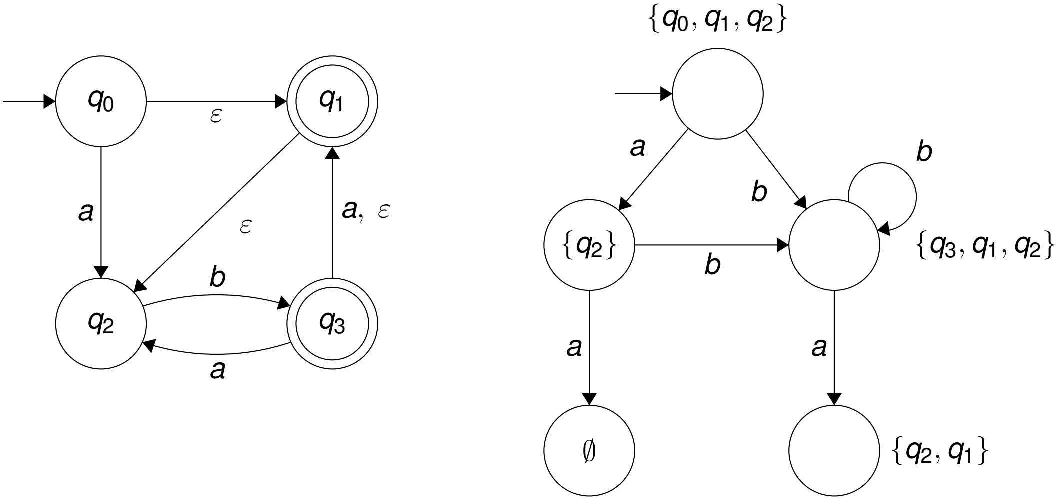


Zo stavu $\{q_2\}$

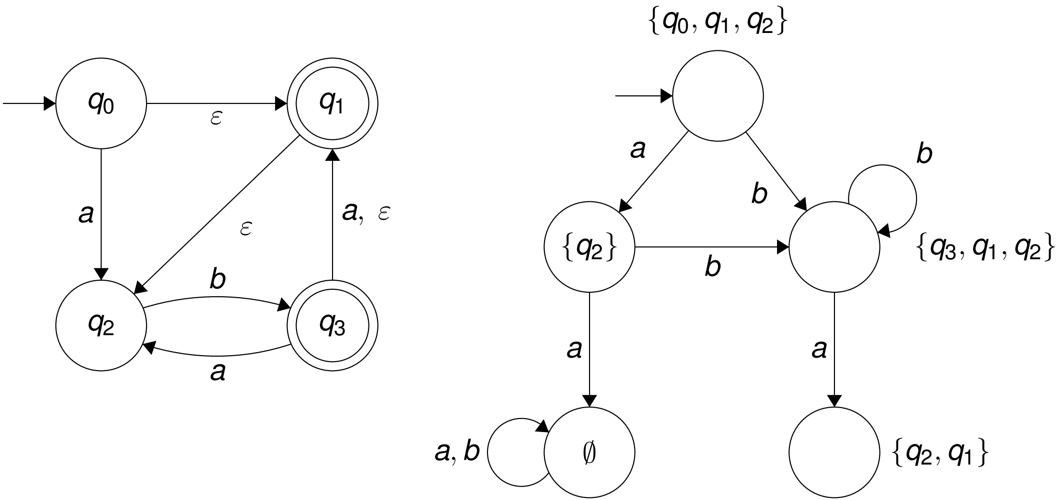

q_0
q_1
q_2
q_3
ε
a
ε
a, ε
b
a


$\{q_0, q_1, q_2\}$
a
b
$\{q_3, q_1, q_2\}$
$\{q_2\}$
b
a
$\emptyset$

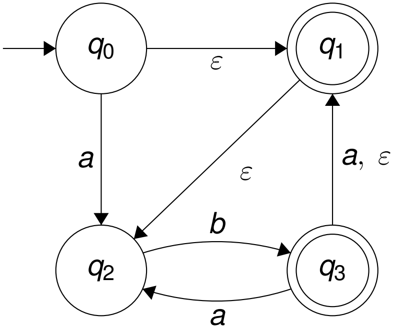
Zo stavu $\{q_3, q_1, q_2\}$

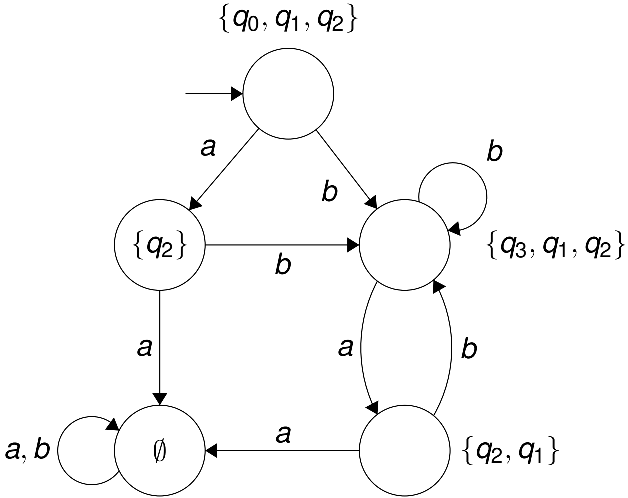


Zo stavu $\emptyset$

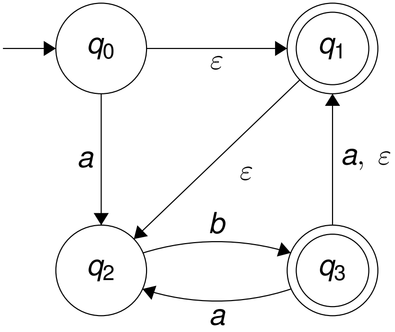


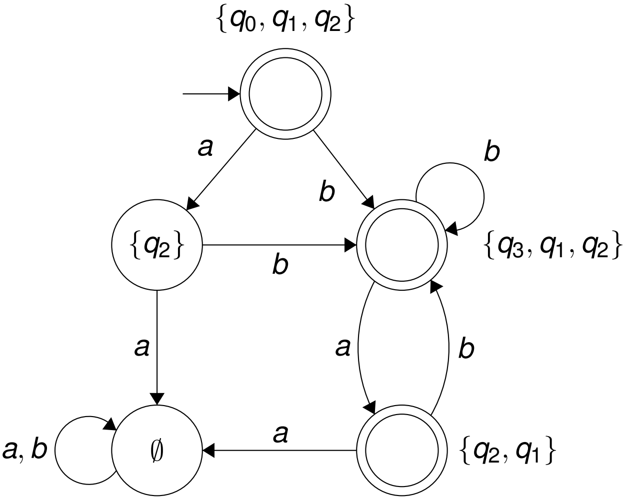
Zo stavu $\{q_2, q_1\}$





Akceptačné stavy





Ekvivalencia NKA a DKA - idea

1. Stavy DKA, ekvivalentného k NKA, tvoria množiny stavov NKA.
2. Počiatočný stav DKA predstavuje množina stavov NKA, do ktorých sa NKA vie dostať bez spracovania vstupných symbolov (t.j. pôvodné q_0 + teoreticky iné stavy, ak sa do nich dá dostať $\vdash^*$ z q_0 pomocou ε-prechodov).
3. Ďalšie stavy DKA predstavujú množiny stavov, do ktorých sa NKA vedel dostať spracovaním jednotlivých vstupných symbolov (a znovu je potrebné zohľadniť ε-prechody).
4. Keďže stavy DKA predstavujú množiny stavov, v ktorých sa mohol nachádzať NKA po spracovaní príslušného vstupného reťazca, ak stav DKA obsahuje niektorý z akceptačných stavov NKA, tak aj tento stav DKA musí byť akceptačný.

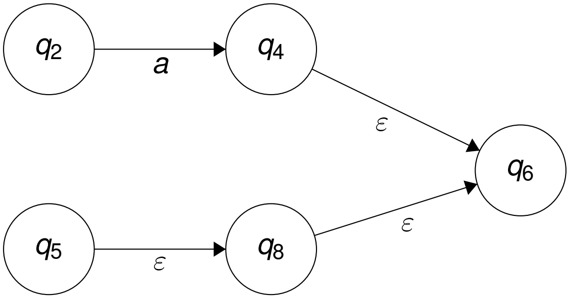
Operácia $CLOSURE_\varepsilon$

Pri konštrukcii DKA si zadefinujeme operáciu $CLOSURE_\varepsilon$, ktorá pre množinu stavov NKA vráti množinu stavov uzavretú na kroky na ε.

Vstup: Prechodová funkcia δ NKA M, množina stavov $S \subseteq Q$.
Výstup: Množina stavov S uzavretá na kroky na ε.

1: **opakuj**
2: $\quad \acute{S} \leftarrow S;$
3: $\quad S \leftarrow \acute{S} \cup \bigcup_{q \in \acute{S}} \delta(q, \varepsilon);$
4: **pokiaľ** $S \neq \acute{S}$
5: **vráť** S;

Operácia $CLOSURE_\varepsilon$



$$CLOSURE_\varepsilon(\{q_2, q_5\}) = \{q_2, q_5, q_6, q_8\}.$$

Späť k ekvivalencii NKA, DKA

Časť dôkazu: K danému NKA $M = (Q, \Sigma, \delta, q_0, F)$ zostrojíme DKA $\acute{M} = (\acute{Q}, \Sigma, \acute{\delta}, \acute{q}_0, \acute{F})$, ktorý bude akceptovať ten istý jazyk nasledovne:

- **Množina stavov** $\acute{Q}$ automatu $\acute{M}$: $\acute{Q} \subseteq 2^Q$, t.j. stavy $\acute{q} \in \acute{Q}$ automatu $\acute{M}$ budú v tvare $\{q_1, q_2, ..., q_i\}$, kde $q_k \in Q$ pre $k = 1, 2, ..., i$.
- **Počiatočný stav** $\acute{q}_0 = CLOSURE_\varepsilon(\{q_0\})$.
- **Množina akceptačných stavov** $\acute{F}$ je tvorená tými stavmi z $\acute{Q}$, ktoré obsahujú aspoň 1 akceptujúci stav z M.
- **Prechodová funkcia**:

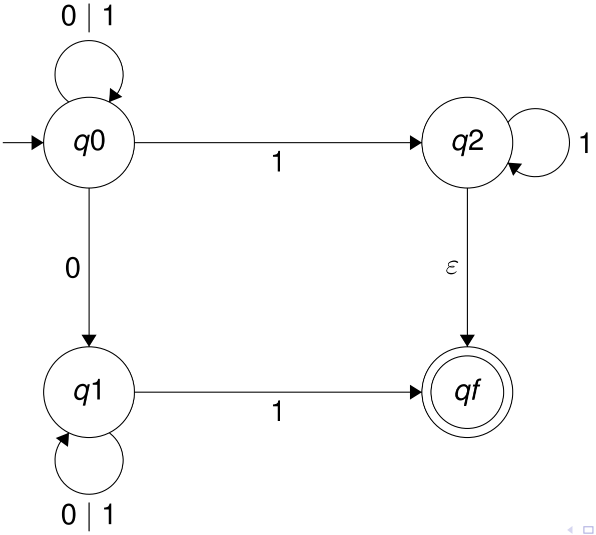
$$\acute{\delta}(\{q_1, ..., q_i\}, a) = CLOSURE_\varepsilon\left(\left\{\bigcup_{k=1}^{i} \delta(q_k, a)\right\}\right), \tag{1}$$

t.j. sleduje všetky možné cesty, ktorými mohol ísť NKA na vstupný symbol a pre stavy $q_1, ..., q_i$.

Vstup: NKA $M = (Q, \Sigma, \delta, q_0, F)$
Výstup: Množina stavov $\acute{Q}$ a prechodová funkcia $\acute{\delta}$ ekvivalentného DKA.

```
zaraď stav CLOSURE_ε({q_0}) do Q́ a označ ho ako nespracovaný
pokiaľ v Q́ existuje nespracovaný stav rob
    vyber z Q́ ľubovoľný nespracovaný stav q a označ ho ako spracovaný
    pre všetky a ∈ Σ rob
        urči stav p = δ́(q, a) podľa vzťahu (1)
        ak p ∉ Q́ potom
            zaraď p do Q́ a označ ho ako nespracovaný
        koniec ak
        zaznamenaj prechod zo stavu q do stavu p na symbol a
    koniec pre
koniec pokiaľ
```

Príklad: Nájdite ekvivalentný DKA pre NKA:



Prechodová funkcia δ:

δ	0	1	ε
q_0	$\{q_0, q_1\}$	$\{q_0, q_2\}$	$\emptyset$
q_1	$\{q_1\}$	$\{q_1, q_f\}$	$\emptyset$
q_2	$\emptyset$	$\{q_2\}$	$\{q_f\}$
q_f	$\emptyset$	$\emptyset$	$\emptyset$

Prechodová funkcia $\acute{\delta}$:

δ	0	1
$CLOSURE_{\varepsilon}(\{q_0\}) = \{q_0\}$	$\{q_0, q_1\}$	$\{q_0, q_2, g_f\}$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_1, q_2, q_f\}$
$\{q_0, q_2, q_f\}$	$\{q_0, q_1\}$	$\{q_0, q_2, q_f\}$
$\{q_0, q_1, q_2, q_f\}$	$\{q_0, q_1\}$	$\{q_0, q_1, q_2, q_f\}$

Prechodová funkcia $\acute{\delta}$, počiatočný stav a akceptujúce stavy:

δ	0	1
$CLOSURE_{\varepsilon}(\{q_0\}) = \{q_0\}$	$\{q_0, q_1\}$	$\{q_0, q_2, g_f\}$
$\{q_0, q_1\}$	$\{q_0, q_1\}$	$\{q_0, q_1, q_2, q_f\}$
$\{q_0, q_2, q_f\}$	$\{q_0, q_1\}$	$\{q_0, q_2, q_f\}$
$\{q_0, q_1, q_2, q_f\}$	$\{q_0, q_1\}$	$\{q_0, q_1, q_2, q_f\}$

Ekvivalencia NKA a DKA

- Počet možných stavov DKA rastie exponenciálne k počtu stavov NKA.
- Výhodou NKA je ich jednoduchšia štruktúra.
- Výhodou DKA je ich jednoduchšia implementácia a následná simulácia ich činnosti.
- Simulácia NKA pre dané vstupné slovo má exponenciálnu zložitosť.
- Simulácia DKA pre dané vstupné slovo má lineárnu zložitosť.

Použitá literatúra

Dedera, Ľ: Počítačové jazyky a ich spracovanie.

Linz, P.: An Introduction to Formal Languages and Automata.

Molnár, Ľ.: Gramatiky a jazyky.