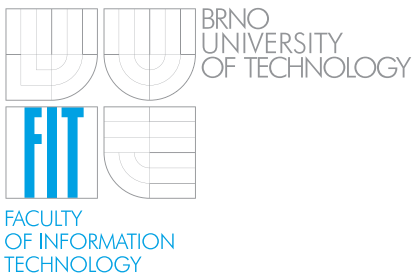
Aplikácia matematiky v informatike

Lukáš Vrábel

Fakulta informačních techonlogií, Vysoké učení technické Brno
Božetěchova 1/2, 612 00 Brno
http://www.fit.vutbr.cz/∼ivrabel



Teoretická informatika

Informatika - mnoho odvetví:

- počítačové siete (teória grafov)
- grafika (geometria)
- datamining (analýza)
- ...

Teoretická informatika

Informatika - mnoho odvetví:

- počítačové siete (teória grafov)
- grafika (geometria)
- datamining (analýza)
- ...

Zameranie prednášky: *teoretická informatika*.

Ciele teoretickej informatiky: formálne a rigorózne študovať

- výpočty
- algoritmy
- programy
- riešenia problémov

Ciele teoretickej informatiky

Teoretická informatika hľadá odpovede na nasledujúce otázky:

- Aké výpočty môžeme spočítať počítačom?
- Aké otázky môžeme rozhodnúť počítačom?
- Akým spôsobom sformalizovať problém tak, aby bol riešitelný počítačom?

Na zodpovedanie týchto otázok používame viacero formálnych modelov - konkrétne sa budeme zaoberať *teóriou formálnych jazykov*.

Formálny jazyk

Prečo jazyky, keď je reč o programoch a počítačoch?

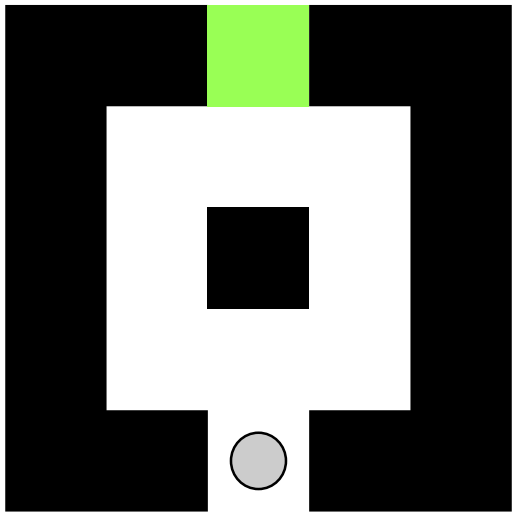
Formálny jazyk

Prečo jazyky, keď je reč o programoch a počítačoch?

- počiatky v lingvistike
- programovacie jazyky
- inštrukcie procesora
- jeden z prístupov k formalizácii problému

Príklad

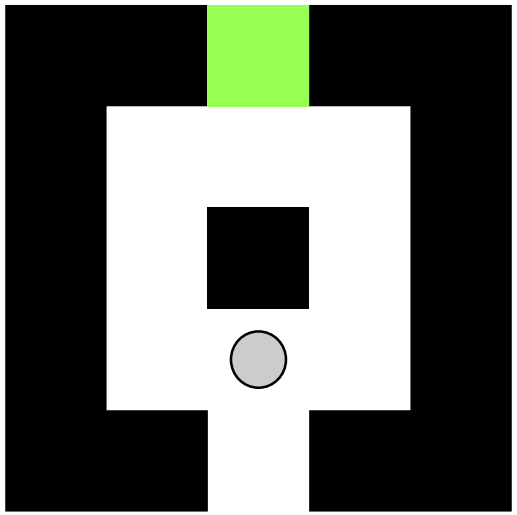
Množina dostupných príkazov: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$



Štartovná pozícia

Príklad

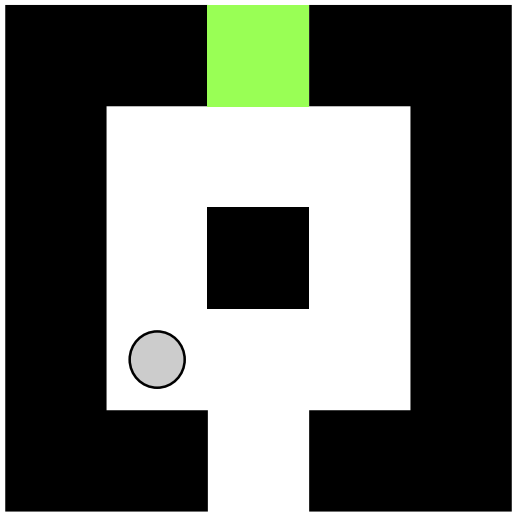
Množina dostupných príkazov: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$



Sekvencia príkazov: $\uparrow$

Príklad

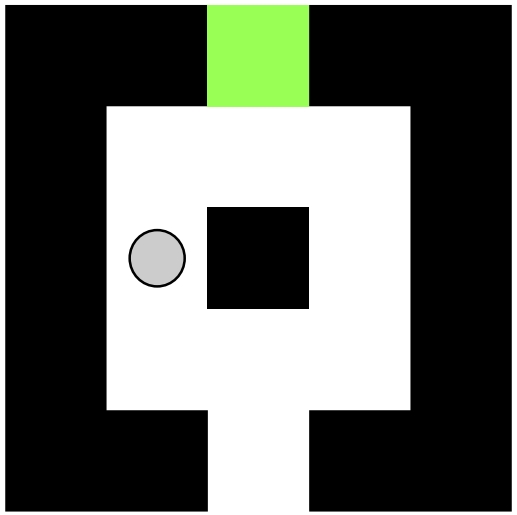
Množina dostupných príkazov: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$



Sekvencia príkazov: $\uparrow\leftarrow$

Príklad

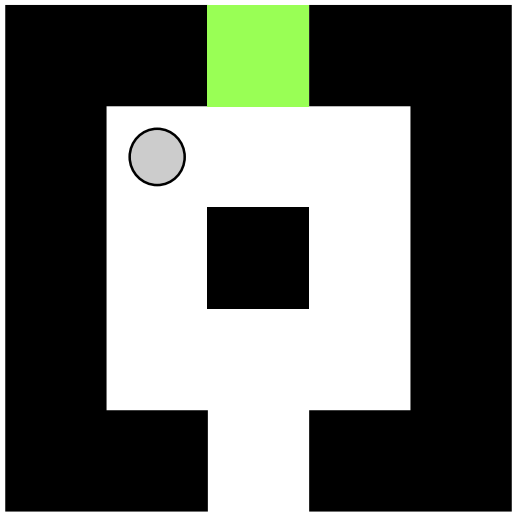
Množina dostupných príkazov: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$



Sekvencia príkazov: $\uparrow\leftarrow\uparrow$

Príklad

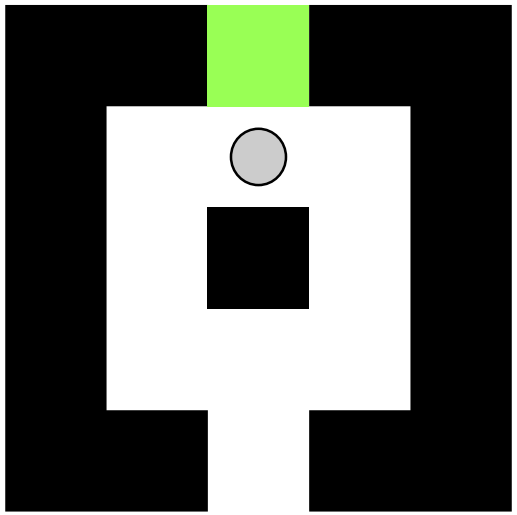
Množina dostupných príkazov: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$



Sekvencia príkazov: $\uparrow\leftarrow\uparrow\uparrow$

Príklad

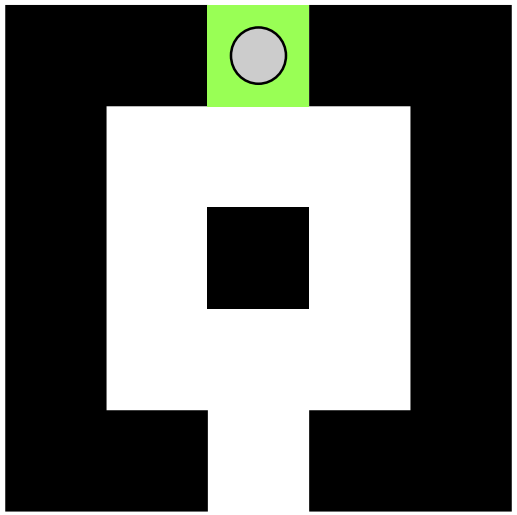
Množina dostupných príkazov: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$



Sekvencia príkazov: $\uparrow\leftarrow\uparrow\uparrow\rightarrow$

Príklad

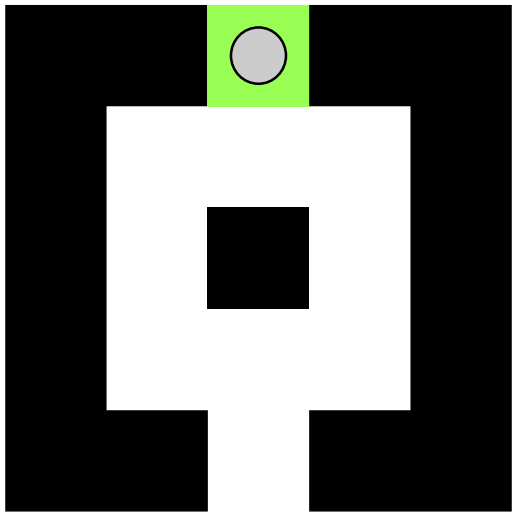
Množina dostupných príkazov: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$



Sekvencia príkazov: $\uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow$

Príklad

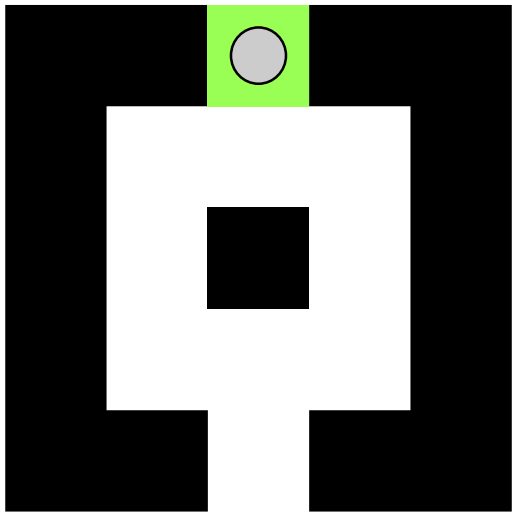
Množina dostupných príkazov: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$



Sekvencia príkazov: $\uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow$
Alternatívna sekvencia: $\uparrow\rightarrow\uparrow\uparrow\leftarrow\uparrow$

Príklad

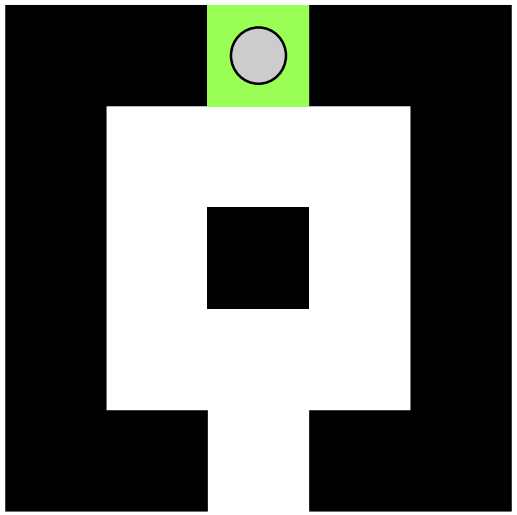
Množina dostupných príkazov: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$



Sekvencia príkazov: $\uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow$
Alternatívna sekvencia: $\uparrow\rightarrow\uparrow\uparrow\leftarrow\uparrow$
Ďalšie možnosti:
$\uparrow\downarrow\uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow$

Príklad

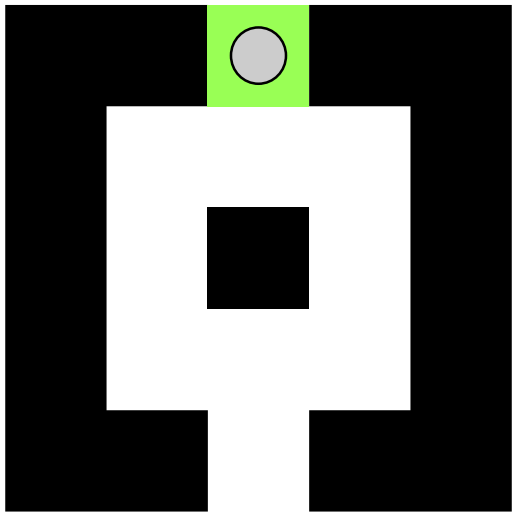
Množina dostupných príkazov: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$



Sekvencia príkazov: $\uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow$
Alternatívna sekvencia: $\uparrow\rightarrow\uparrow\uparrow\leftarrow\uparrow$
Ďalšie možnosti:
$\uparrow\downarrow\uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow$
$\uparrow\downarrow\uparrow\downarrow\uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow$

Príklad

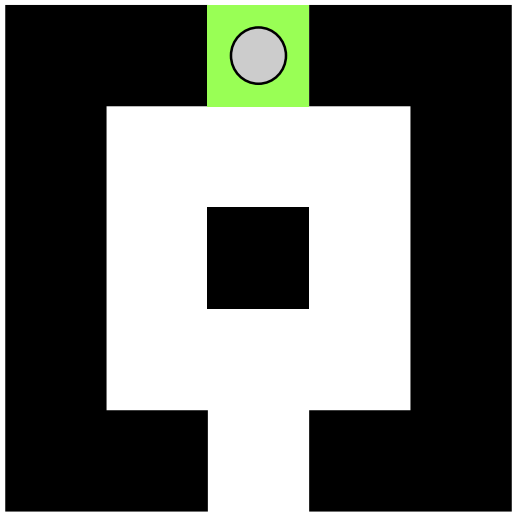
Množina dostupných príkazov: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$



Sekvencia príkazov: $\uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow$
Alternatívna sekvencia: $\uparrow\rightarrow\uparrow\uparrow\leftarrow\uparrow$
Ďalšie možnosti:
$\uparrow\downarrow\uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow$
$\uparrow\downarrow\uparrow\downarrow\uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow$
...

Príklad

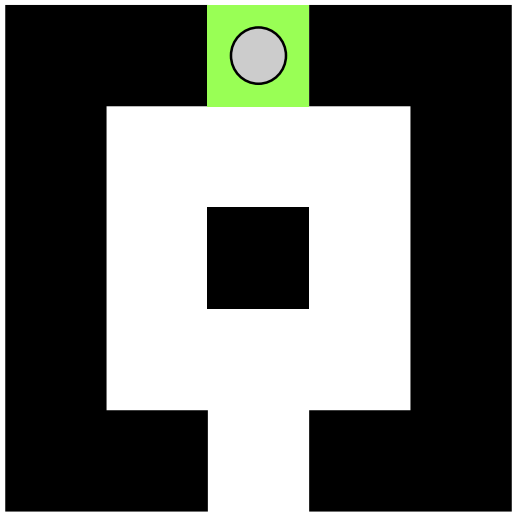
Množina dostupných príkazov: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$



Nesprávne sekvencie: $\uparrow\leftarrow\uparrow\downarrow$, $\uparrow\rightarrow\leftarrow\leftarrow\rightarrow$, ...

Príklad

Množina dostupných príkazov: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$



Nesprávne sekvencie: $\uparrow\leftarrow\uparrow\downarrow$, $\uparrow\rightarrow\leftarrow\leftarrow\rightarrow$, ...
Neplatné sekvencie: $\uparrow\uparrow$, $\downarrow$,...

Príklad

Od *problému* k *jazyku*:

- slová: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$
- validné vety: $\{\uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow, \uparrow\rightarrow\uparrow\uparrow\leftarrow\uparrow, \dots\}$
- nezmyselné vety: $\{\uparrow, \downarrow\leftarrow\rightarrow\downarrow, \rightarrow\uparrow\leftarrow\downarrow, \dots\}$

Príklad

Od *problému* k *jazyku*:

- slová: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$
- validné vety: $\{\uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow, \uparrow\rightarrow\uparrow\uparrow\leftarrow\uparrow, \dots\}$
- nezmyselné vety: $\{\uparrow, \downarrow\leftarrow\rightarrow\downarrow, \rightarrow\uparrow\leftarrow\downarrow, \dots\}$

Analógia s angličtinou:

- slová: { are, hello, how, you, . . . }
- validné vety: { hello how are you, . . . }
- nezmyselné vety: { are hello you, . . . }

Príklad

Od *problému* k *jazyku*:

- slová: $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$
- validné vety: $\{\uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow, \uparrow\rightarrow\uparrow\uparrow\leftarrow\uparrow, \dots\}$
- nezmyselné vety: $\{\uparrow, \downarrow\leftarrow\rightarrow\downarrow, \rightarrow\uparrow\leftarrow\downarrow, \dots\}$

Analógia s angličtinou:

- slová: { are, hello, how, you, . . . }
- validné vety: { hello how are you, . . . }
- nezmyselné vety: { are hello you, . . . }

Hlavná myšlienka

Problém prevedieme na jazyk, a ten potom skúmame.

Formálne jazyky

Abeceda, reťazec, jazyk

Formálny jazyk - formalizácia jazyka pomocou teórie množín.

Abeceda

Neprázdna *konečná* množina sa nazýva *abeceda*. Prvky abecedy nazývame *symboly*.

Abeceda, reťazec, jazyk

Formálny jazyk - formalizácia jazyka pomocou teórie množín.

Abeceda

Neprázdna *konečná* množina sa nazýva *abeceda*. Prvky abecedy nazývame *symboly*.

Reťazec

Nech Σ je abeceda. Ľubovolná *konečná* usporiadaná postupnosť symbolov abecedy sa nazýva *reťazec* nad abecedou Σ.

Prázdny reťazec označujeme znakom ε.

Abeceda, reťazec, jazyk

Formálny jazyk - formalizácia jazyka pomocou teórie množín.

Abeceda

Neprázdna *konečná* množina sa nazýva *abeceda*. Prvky abecedy nazývame *symboly*.

Reťazec

Nech Σ je abeceda. Ľubovolná *konečná* usporiadaná postupnosť symbolov abecedy sa nazýva *reťazec* nad abecedou Σ.

Prázdny reťazec označujeme znakom ε.

Príklad

Abeceda: $\Sigma = \{a, b, c\}$ Reťazce:

- *abc*, *aaaa*, *bcbcb*
- *a*, *b*
- ε

Konkatenácia

Binárna operácia *konkatenácie* označuje spojenie dvoch reťazcov.

Konkatenácia

Binárna operácia *konkatenácie* označuje spojenie dvoch reťazcov.

Príklad

Abeceda: $\Sigma = \{a, b, c\}$
Reťazce:

- $u = aac$, $v = bca$
- $uv = aacbca$
- $vu = bcaaac$

Konkatenácia

Binárna operácia *konkatenácie* označuje spojenie dvoch reťazcov.

Príklad

Abeceda: $\Sigma = \{a, b, c\}$
Reťazce:

- $u = aac$, $v = bca$
- $uv = aacbca$
- $vu = bcaaac$

Pre konkatenáciu platí:

- je *asociatívna* — $u(vw) = (uv)w$
- nieje *komutatívna* — $uv \neq vu$
- ε je neutrálny prvok — $u\varepsilon = \varepsilon u = u$

Reťazce

Množina všetkých reťazcov nad abecedou Σ sa značí Σ^*. Σ^+ označuje množinu všetkých neprázdnych reťazcov nad Σ.

Reťazce

Množina všetkých reťazcov nad abecedou Σ sa značí Σ^*. Σ^+ označuje množinu všetkých neprázdnych reťazcov nad Σ.

- Σ^* je volný monoid generovaný z abecedy Σ pomocou konkatenácie.
- Σ^+ je pologrupa

Reťazce

Množina všetkých reťazcov nad abecedou Σ sa značí Σ^*. Σ^+ označuje množinu všetkých neprázdnych reťazcov nad Σ.

- Σ^* je volný monoid generovaný z abecedy Σ pomocou konkatenácie.
- Σ^+ je pologrupa

Jazyk

$L \subseteq \Sigma^*$ je *jazyk* nad abecedou Σ.

Príklad

Množina všetkých sekvencií príkazov, ktorými prejdeme bludisko, tvorí jazyk nad abecedou $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$

Príklad

Množina všetkých sekvencií príkazov, ktorými prejdeme bludisko, tvorí jazyk nad abecedou $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$

Príklad

Abeceda: prvotné formule + logické spojky vo výrokovej logike $\{p_1, p_2, \ldots, p_n\} \cup \{\vee, \&, \neg, \rightarrow, \equiv, (,)\}$.
Jazyk: množina všetkých výrokových formulí.

- $p_1 \rightarrow (p_3 \& p_8)$ patrí do jazyka
- $\&) p_2 \equiv$ nepatrí do jazyka

Príklad

Množina všetkých sekvencií príkazov, ktorými prejdeme bludisko, tvorí jazyk nad abecedou $\{\leftarrow, \uparrow, \downarrow, \rightarrow\}$

Príklad

Abeceda: prvotné formule + logické spojky vo výrokovej logike $\{p_1, p_2, \ldots, p_n\} \cup \{\vee, \&, \neg, \rightarrow, \equiv, (,)\}$.
Jazyk: množina všetkých výrokových formulí.

- $p_1 \rightarrow (p_3 \& p_8)$ patrí do jazyka
- $\&) p_2 \equiv$ nepatrí do jazyka

Príklad

Medzi jazyky nad abecedou Σ patrí aj $\emptyset$, $\{\varepsilon\}$ a Σ^*

Jazyky

- jednotlivé reťazce musia byť vždy konečné, ale jazyk môže byť *nekonečný*
- problém - počítač je *konečný*
- my však chceme zodpovedať otázky typu "prejde robot bludiskom pomocou danej sekvencie príkazov?"alebo "je daný reťazec výrokovou formulou?"

- jednotlivé reťazce musia byť vždy konečné, ale jazyk môže byť *nekonečný*
- problém - počítač je *konečný*
- my však chceme zodpovedať otázky typu "prejde robot bludiskom pomocou danej sekvencie príkazov?"alebo "je daný reťazec výrokovou formulou?"

Riešenie

Musíme popísať *nekonečné* jazyky pomocou nejakého *konečného* aparátu.

- jednotlivé reťazce musia byť vždy konečné, ale jazyk môže byť *nekonečný*
- problém - počítač je *konečný*
- my však chceme zodpovedať otázky typu "prejde robot bludiskom pomocou danej sekvencie príkazov?"alebo "je daný reťazec výrokovou formulou?"

Riešenie

Musíme popísať *nekonečné* jazyky pomocou nejakého *konečného* aparátu.

- skúmame, ktoré všetky (nekonečné) jazyky môžeme popísať pomocou konečných prostriedkov
- veľa modelov - my sa zameriame na *gramatiky*

Gramatiky

Hlavná myšlienka

Hlavná myšlienka gramatiky spočíva v konečnej množine gramatických pravidiel, ktoré generujú daný jazyk.

Hlavná myšlienka

Hlavná myšlienka gramatiky spočíva v konečnej množine gramatických pravidiel, ktoré generujú daný jazyk.

- Gramatiky zaviedol linguista Noam Chomsky v 60. rokoch.
- Pôvodne mali slúžiť na popis syntaxe prirodzeného jazyka.
- Po čase sa však ukázalo, že gramatiky sú nedostatočujúce.
- Uchytili sa však v informatike, ktorá pracuje s umelými jazykmi (programovacie jazyky), kde sa gramatiky začali intenzívne využívať

Princíp gramatiky vysvetlíme na jazyku výrokovej logiky:

Príklad

Základné stavebné kamene jazyka výrokovej logiky sú *prvotné formule* a *logické spojky*.

Prvotné formule: $\{p, q\}$
Logické spojky: $\{\&, \neg\}$

Princíp gramatiky vysvetlíme na jazyku výrokovej logiky:

Príklad

Základné stavebné kamene jazyka výrokovej logiky sú *prvotné formule* a *logické spojky*.

Prvotné formule: $\{p, q\}$
Logické spojky: $\{\&, \neg\}$

Výrokové formule sú definované nasledovne:

1. Každá prvotná formula je výroková formula
2. Nech A a B sú výrokové formule. Potom $(A\&B)$ a $(\neg A)$ sú tiež výrokové formule.
3. Každá výroková formula vznikne konečným počtom aplikovaní pravidiel (1) a (2).

Výrokové formule

1. Každá prvotná formula je výroková formula
2. Nech A a B sú výrokové formule. Potom $(A\&B)$ a $(\neg A)$ sú tiež výrokové formule.

Abeceda: $\{p, q, \&, \neg, (,)\}$

Výrokové formule

1. Každá prvotná formula je výroková formula
2. Nech A a B sú výrokové formule. Potom $(A\&B)$ a $(\neg A)$ sú tiež výrokové formule.

Abeceda: $\{p, q, \&, \neg, (,)\}$

Špeciálny symbol V, ktorý reprezentuje nedokončenú formulu a nieje súčasťou jazyka.

Výrokové formule

1. Každá prvotná formula je výroková formula
2. Nech A a B sú výrokové formule. Potom $(A\&B)$ a $(\neg A)$ sú tiež výrokové formule.

Abeceda: $\{p, q, \&, \neg, (,)\}$

Špeciálny symbol V, ktorý reprezentuje nedokončenú formulu a nieje súčasťou jazyka.

Gramatické (prepisovacie) pravidlá:

$V \to p$
$V \to q$ podľa (1)

Výrokové formule

1. Každá prvotná formula je výroková formula
2. Nech A a B sú výrokové formule. Potom $(A\&B)$ a $(\neg A)$ sú tiež výrokové formule.

Abeceda: $\{p, q, \&, \neg, (,)\}$

Špeciálny symbol V, ktorý reprezentuje nedokončenú formulu a nieje súčasťou jazyka.

Gramatické (prepisovacie) pravidlá:

$V \to p$
$V \to q$ podľa (1)

$V \to (V\&V)$
$V \to (\neg V)$ podľa (2)

Pravidlá

$V \to p \quad V \to q \quad V \to (V \& V) \quad V \to (\neg V)$

Príklad

reťazec	pravidlo
V	

Pravidlá

$V \to p \quad V \to q \quad V \to (V \& V) \quad V \to (\neg V)$

Príklad

reťazec	pravidlo
V	$V \to (V \& V)$

Pravidlá

$V \to p \quad V \to q \quad V \to (V \& V) \quad V \to (\neg V)$

Príklad

reťazec	pravidlo
V	$V \to (V \& V)$
$(V \& V)$	

Pravidlá

$V \to p \quad V \to q \quad V \to (V \& V) \quad V \to (\neg V)$

Príklad

reťazec	pravidlo
V	$V \to (V \& V)$
$(V \& V)$	$V \to (\neg V)$

Pravidlá

$V \to p \quad V \to q \quad V \to (V \& V) \quad V \to (\neg V)$

Príklad

reťazec	pravidlo
V	$V \to (V \& V)$
$(V \& V)$	$V \to (\neg V)$
$((\neg V) \& V)$	

Pravidlá

$V \to p \quad V \to q \quad V \to (V \& V) \quad V \to (\neg V)$

Príklad

reťazec	pravidlo
V	$V \to (V \& V)$
$(V \& V)$	$V \to (\neg V)$
$((\neg V) \& V)$	$V \to p$

Pravidlá

$V \to p \quad V \to q \quad V \to (V \& V) \quad V \to (\neg V)$

Príklad

reťazec	pravidlo
V	$V \to (V \& V)$
$(V \& V)$	$V \to (\neg V)$
$((\neg V) \& V)$	$V \to p$
$((\neg V) \& p)$	

Pravidlá

$V \to p \quad V \to q \quad V \to (V\&V) \quad V \to (\neg V)$

Príklad

reťazec	pravidlo
V	$V \to (V\&V)$
$(V\&V)$	$V \to (\neg V)$
$((\neg V)\&V)$	$V \to p$
$((\neg V)\&p)$	$V \to (V\&V)$

Pravidlá

$V \to p \quad V \to q \quad V \to (V\&V) \quad V \to (\neg V)$

Príklad

reťazec	pravidlo
V	$V \to (V\&V)$
$(V\&V)$	$V \to (\neg V)$
$((\neg V)\&V)$	$V \to p$
$((\neg V)\&p)$	$V \to (V\&V)$
$((\neg(V\&V))\&p)$	

Pravidlá

$V \to p \quad V \to q \quad V \to (V\&V) \quad V \to (\neg V)$

Príklad

reťazec	pravidlo
V	$V \to (V\&V)$
$(V\&V)$	$V \to (\neg V)$
$((\neg V)\&V)$	$V \to p$
$((\neg V)\&p)$	$V \to (V\&V)$
$((\neg(V\&V))\&p)$	$V \to q$

Pravidlá

$V \to p \quad V \to q \quad V \to (V\&V) \quad V \to (\neg V)$

Príklad

reťazec	pravidlo
V	$V \to (V\&V)$
$(V\&V)$	$V \to (\neg V)$
$((\neg V)\&V)$	$V \to p$
$((\neg V)\&p)$	$V \to (V\&V)$
$((\neg(V\&V))\&p)$	$V \to q$
$((\neg(V\&q))\&p)$	

Pravidlá

$V \to p \quad V \to q \quad V \to (V\&V) \quad V \to (\neg V)$

Príklad

reťazec	pravidlo
V	$V \to (V\&V)$
$(V\&V)$	$V \to (\neg V)$
$((\neg V)\&V)$	$V \to p$
$((\neg V)\&p)$	$V \to (V\&V)$
$((\neg(V\&V))\&p)$	$V \to q$
$((\neg(V\&q))\&p)$	$V \to p$

Pravidlá

$V \to p \quad V \to q \quad V \to (V\&V) \quad V \to (\neg V)$

Príklad

reťazec	pravidlo
V	$V \to (V\&V)$
$(V\&V)$	$V \to (\neg V)$
$((\neg V)\&V)$	$V \to p$
$((\neg V)\&p)$	$V \to (V\&V)$
$((\neg(V\&V))\&p)$	$V \to q$
$((\neg(V\&q))\&p)$	$V \to p$
$((\neg(p\&q))\&p)$	

Pravidlá

$V \to p \quad V \to q \quad V \to (V\&V) \quad V \to (\neg V)$

Príklad

reťazec	pravidlo
V	$V \to (V\&V)$
$(V\&V)$	$V \to (\neg V)$
$((\neg V)\&V)$	$V \to p$
$((\neg V)\&p)$	$V \to (V\&V)$
$((\neg(V\&V))\&p)$	$V \to q$
$((\neg(V\&q))\&p)$	$V \to p$
$((\neg(p\&q))\&p)$	nemáme V — hotovo

Jazyk výrokovej logiky

Pravidlá

$V \to p \quad V \to q \quad V \to (V\&V) \quad V \to (\neg V)$

Príklad

reťazec	pravidlo
V	$V \to (V\&V)$
$(V\&V)$	$V \to (\neg V)$
$((\neg V)\&V)$	$V \to p$
$((\neg V)\&p)$	$V \to (V\&V)$
$((\neg(V\&V))\&p)$	$V \to q$
$((\neg(V\&q))\&p)$	$V \to p$
$((\neg(p\&q))\&p)$	nemáme V — hotovo

Pracujeme vlastne s dvoma jazykmi:

- cieľový jazyk, ktorý chceme vygenerovať
- pomocný, interný jazyk gramatiky, ktorým popisujeme proces prepisovania

Bezkontextová gramatika

Bezkontextová gramatika je štvorica

$$G = (N, T, P, S)$$

- N je konečná množina neterminálov
- T je konečná množina terminálov
- $P \subseteq N \times (N \cup T)^*$ je konečná množina pravidiel. Namiesto $(A, w) \in P$ píšeme $A \to w$.
- $S \in N$ je počiatočný neterminál

Príklad

$$N = \{V\}$$
$$T = \{p, q, \&, \neg, (,)\}$$
$$P = \{V \to p, V \to q, V \to (V \& V), V \to (\neg V)\}$$
$$S = V$$

Jazyk generovaný gramatikou

Na formálny popis generovania jazyka zavedieme reláciu *priamej derivácie*

Jazyk generovaný gramatikou

Na formálny popis generovania jazyka zavedieme reláciu *priamej derivácie*

Nech $G = (N, T, P, S)$ je bezkontextová gramatika

- *priama derivácia* ($\Rightarrow$) je binárna relácia nad $(N \cup T)^*$
- znázorňuje akúsi *následnosť* reťazcov na základe pravidiel gramatiky

Jazyk generovaný gramatikou

Na formálny popis generovania jazyka zavedieme reláciu *priamej derivácie*

Nech $G = (N, T, P, S)$ je bezkontextová gramatika

- *priama derivácia* ($\Rightarrow$) je binárna relácia nad $(N \cup T)^*$
- znázorňuje akúsi *následnosť* reťazcov na základe pravidiel gramatiky

Definícia:

Priama derivácia

$u \Rightarrow v$ vtedy a len vtedy, keď

- $A \to w \in P$
- $u = xAy$
- $v = xwy$

kde $x, y \in (N \cup T)^*$

Príklad

$$P = \{V \to p, V \to q, V \to (V\&V), V \to (\neg V)\}$$

$(V\&V)$

Príklad

$$P = \{V \to p, V \to q, V \to (V\&V), V \to (\neg V)\}$$

$$(V\&V) \quad \Rightarrow \quad (p\&V)$$

Príklad

$$P = \{V \to p, V \to q, V \to (V\&V), V \to (\neg V)\}$$

$(V\&V) \quad \Rightarrow \quad (p\&V)$ pretože $V \to p \in P$

Príklad

$$P = \{V \to p, V \to q, V \to (V\&V), V \to (\neg V)\}$$

$$\begin{array}{lcll} (V\&V) & \Rightarrow & (p\&V) & \text{pretože } V \to p \in P \\ & \Rightarrow & (q\&V) & \end{array}$$

Príklad

$$P = \{V \to p, V \to q, V \to (V\&V), V \to (\neg V)\}$$

$$\begin{array}{lcll} (V\&V) & \Rightarrow & (p\&V) & \text{pretože } V \to p \in P \\ & \Rightarrow & (q\&V) & \text{pretože } V \to q \in P \end{array}$$

Príklad

$$P = \{V \to p, V \to q, V \to (V\&V), V \to (\neg V)\}$$

$$\begin{array}{lcll} (V\&V) & \Rightarrow & (p\&V) & \text{pretože } V \to p \in P \\ & \Rightarrow & (q\&V) & \text{pretože } V \to q \in P \\ & \Rightarrow & ((V\&V)\&V) & \end{array}$$

Príklad

$$P = \{V \to p, V \to q, V \to (V\&V), V \to (\neg V)\}$$

$$\begin{array}{lll} (V\&V) & \Rightarrow & (p\&V) \qquad \text{pretože } V \to p \in P \\ & \Rightarrow & (q\&V) \qquad \text{pretože } V \to q \in P \\ & \Rightarrow & ((V\&V)\&V) \\ & \Rightarrow & ((\neg V)\&V) \end{array}$$

Príklad

$$P = \{V \to p, V \to q, V \to (V\&V), V \to (\neg V)\}$$

$$\begin{aligned}
(V\&V) &\Rightarrow (p\&V) && \text{pretože } V \to p \in P \\
&\Rightarrow (q\&V) && \text{pretože } V \to q \in P \\
&\Rightarrow ((V\&V)\&V) \\
&\Rightarrow ((\neg V)\&V) \\
&\Rightarrow (V\&p)
\end{aligned}$$

Príklad

$$P = \{V \to p, V \to q, V \to (V\&V), V \to (\neg V)\}$$

$$\begin{aligned}
(V\&V) &\Rightarrow (p\&V) && \text{pretože } V \to p \in P \\
&\Rightarrow (q\&V) && \text{pretože } V \to q \in P \\
&\Rightarrow ((V\&V)\&V) \\
&\Rightarrow ((\neg V)\&V) \\
&\Rightarrow (V\&p) \\
&\Rightarrow (V\&q)
\end{aligned}$$

Príklad

$$P = \{V \to p, V \to q, V \to (V\&V), V \to (\neg V)\}$$

$$\begin{aligned}
(V\&V) &\Rightarrow (p\&V) && \text{pretože } V \to p \in P \\
&\Rightarrow (q\&V) && \text{pretože } V \to q \in P \\
&\Rightarrow ((V\&V)\&V) \\
&\Rightarrow ((\neg V)\&V) \\
&\Rightarrow (V\&p) \\
&\Rightarrow (V\&q) \\
&\Rightarrow (V\&(V\&V))
\end{aligned}$$

Príklad

$$P = \{V \to p, V \to q, V \to (V\&V), V \to (\neg V)\}$$

$$\begin{aligned}
(V\&V) &\Rightarrow (p\&V) && \text{pretože } V \to p \in P\\
&\Rightarrow (q\&V) && \text{pretože } V \to q \in P\\
&\Rightarrow ((V\&V)\&V)\\
&\Rightarrow ((\neg V)\&V)\\
&\Rightarrow (V\&p)\\
&\Rightarrow (V\&q)\\
&\Rightarrow (V\&(V\&V))\\
&\Rightarrow (V\&(\neg V))
\end{aligned}$$

Príklad

$$P = \{V \to p, V \to q, V \to (V\&V), V \to (\neg V)\}$$

$$\begin{array}{lll}
(V\&V) & \Rightarrow & (p\&V) \quad \text{pretože } V \to p \in P \\
& \Rightarrow & (q\&V) \quad \text{pretože } V \to q \in P \\
& \Rightarrow & ((V\&V)\&V) \\
& \Rightarrow & ((\neg V)\&V) \\
& \Rightarrow & (V\&p) \\
& \Rightarrow & (V\&q) \\
& \Rightarrow & (V\&(V\&V)) \\
& \Rightarrow & (V\&(\neg V))
\end{array}$$

Relácia derivácie spája dokopy dva reťazce, kde prvý reťazec môže byť "prepísaný"nejakým pravidlom tak, aby vznikol druhý reťazec.

Derivácia a Jazyk

- Reláciu priamej derivácie d'alej rozšírime o reflexívny a tranzitívny uzáver, pre ktorý použijeme značku $\Rightarrow^*$.
- Tým pádom dostaneme ku každému reťazcu jeho priamych aj vzdialených následníkov

- Reláciu priamej derivácie d'alej rozšírime o reflexívny a tranzitívny uzáver, pre ktorý použijeme značku $\Rightarrow^*$.
- Tým pádom dostaneme ku každému reťazcu jeho priamych aj vzdialených následníkov
- Potom môžeme jazyk príslušiaci gramatike definovať nasledovným spôsobom:

Jazyk

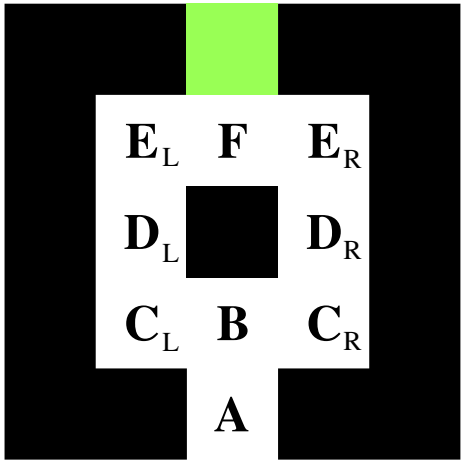
Nech $G = (N, T, P, S)$ je bezkontextová gramatika. Jazyk generovaný gramatikou G:

$$L(G) = \{w \in T^* \mid S \Rightarrow^* w\}$$

Gramatiky

- Gramatiky vychádzajú z linguistiky, a preto sa hodia na popis prirodzených jazykov
- čo s formálnymi jazykmi, ktoré nemajú základ v prirodzenom jazyku?
- napríklad jazyk definujúci prechod bludiskom

Príklad



$$G = (N, T, P, S)$$

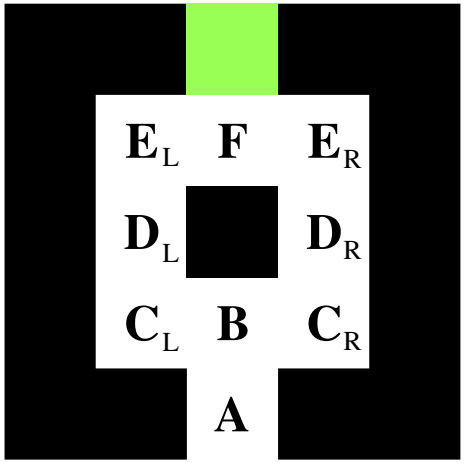
$$T = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$$

$$N = \{A, B, C_L, D_L, E_L, C_R, D_R, E_R, F\}$$

$$S = A$$

Pravidlá

Príklad



$$G = (N, T, P, S)$$

$$T = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$$

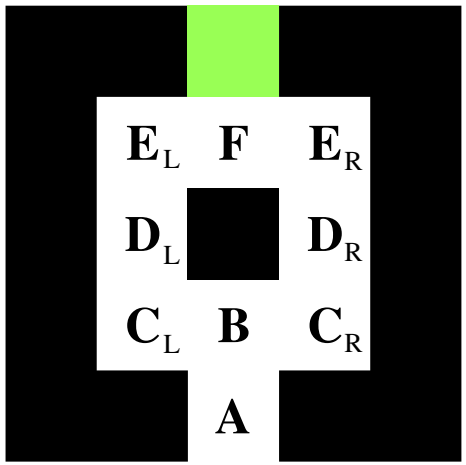
$$N = \{A, B, C_L, D_L, E_L, C_R, D_R, E_R, F\}$$

$$S = A$$

Pravidlá

$$A \rightarrow \uparrow B$$

Príklad



$$G = (N, T, P, S)$$

$$T = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$$

$$N = \{A, B, C_L, D_L, E_L, C_R, D_R, E_R, F\}$$

$$S = A$$

Pravidlá

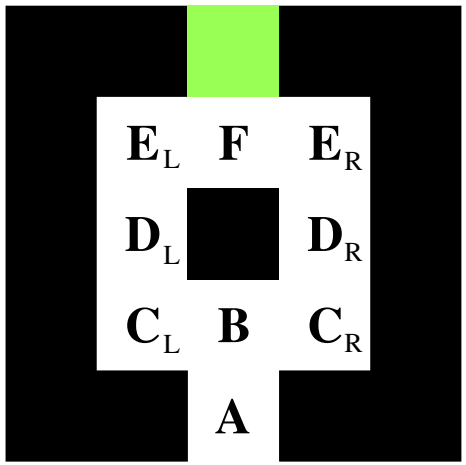
$$A \rightarrow \uparrow B$$

$$B \rightarrow \downarrow A$$

$$B \rightarrow \leftarrow C_L$$

$$B \rightarrow \rightarrow C_R$$

Príklad



$$G = (N, T, P, S)$$

$$T = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$$

$$N = \{A, B, C_L, D_L, E_L, C_R, D_R, E_R, F\}$$

$$S = A$$

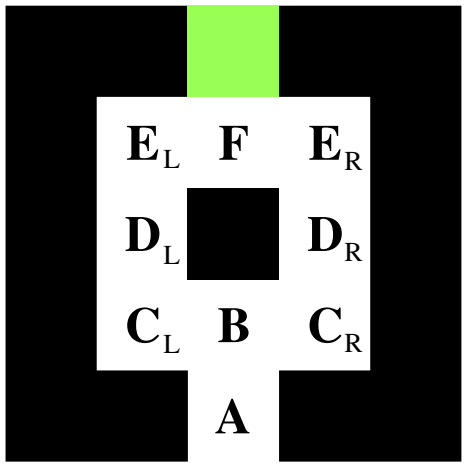
Pravidlá

$$A \rightarrow \uparrow B$$

$$B \rightarrow \downarrow A$$
$$B \rightarrow \leftarrow C_L$$
$$B \rightarrow \rightarrow C_R$$

$$C_L \rightarrow \uparrow D_L$$
$$C_L \rightarrow \rightarrow B$$

Príklad



$$G = (N, T, P, S)$$

$$T = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$$

$$N = \{A, B, C_L, D_L, E_L, C_R, D_R, E_R, F\}$$

$$S = A$$

Pravidlá

$$A \rightarrow \uparrow B$$

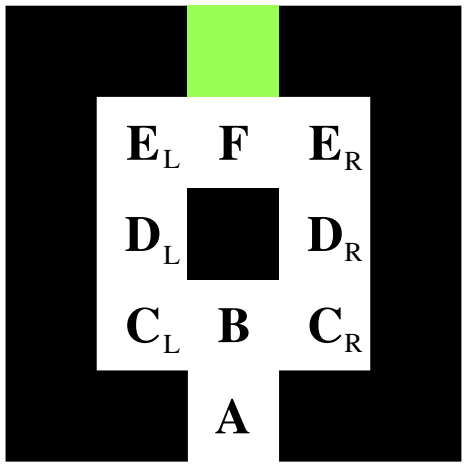
$$B \rightarrow \downarrow A$$
$$B \rightarrow \leftarrow C_L$$
$$B \rightarrow \rightarrow C_R$$

$$C_L \rightarrow \uparrow D_L$$
$$C_L \rightarrow \rightarrow B$$

...

$$F \rightarrow \uparrow$$
$$F \rightarrow \leftarrow E_L$$
$$F \rightarrow \rightarrow E_R$$

Príklad



$$G = (N, T, P, S)$$

$$T = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$$

$$N = \{A, B, C_L, D_L, E_L, C_R, D_R, E_R, F\}$$

$$S = A$$

Pravidlá

$$A \rightarrow \uparrow B$$

$$B \rightarrow \downarrow A$$
$$B \rightarrow \leftarrow C_L$$
$$B \rightarrow \rightarrow C_R$$

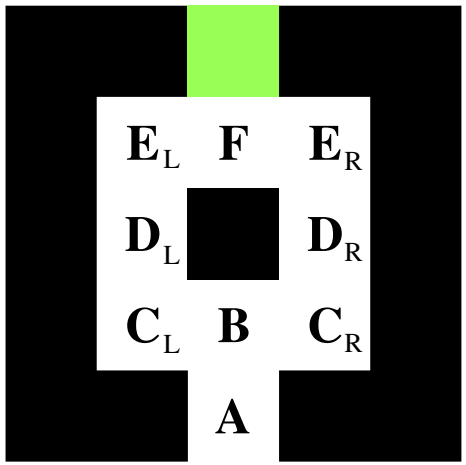
$$C_L \rightarrow \uparrow D_L$$
$$C_L \rightarrow \rightarrow B$$

...

$$F \rightarrow \uparrow$$
$$F \rightarrow \leftarrow E_L$$
$$F \rightarrow \rightarrow E_R$$

A

Príklad



$$G = (N, T, P, S)$$

$$T = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$$

$$N = \{A, B, C_L, D_L, E_L, C_R, D_R, E_R, F\}$$

$$S = A$$

Pravidlá

$$A \rightarrow \uparrow B$$

$$B \rightarrow \downarrow A$$
$$B \rightarrow \leftarrow C_L$$
$$B \rightarrow \rightarrow C_R$$

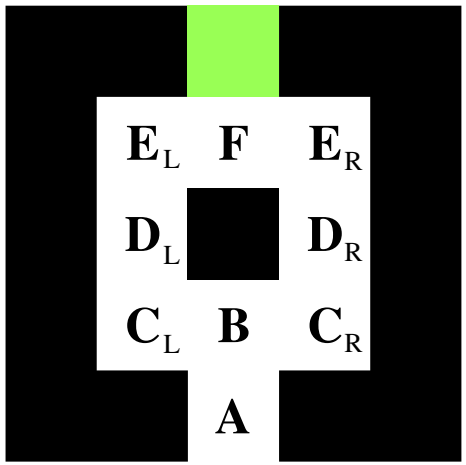
$$C_L \rightarrow \uparrow D_L$$
$$C_L \rightarrow \rightarrow B$$

...

$$F \rightarrow \uparrow$$
$$F \rightarrow \leftarrow E_L$$
$$F \rightarrow \rightarrow E_R$$

$$A \Rightarrow \uparrow B$$

Príklad



$$G = (N, T, P, S)$$

$$T = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$$

$$N = \{A, B, C_L, D_L, E_L, C_R, D_R, E_R, F\}$$

$$S = A$$

Pravidlá

$$A \rightarrow \uparrow B$$

$$B \rightarrow \downarrow A$$
$$B \rightarrow \leftarrow C_L$$
$$B \rightarrow \rightarrow C_R$$

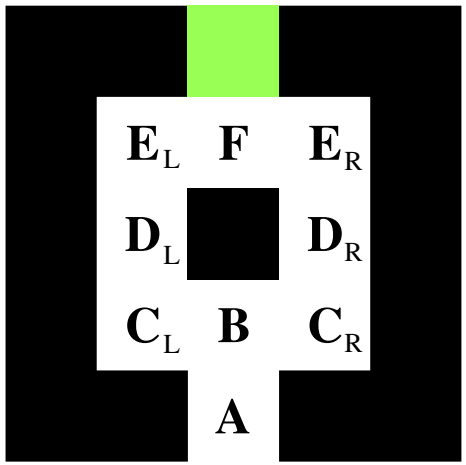
$$C_L \rightarrow \uparrow D_L$$
$$C_L \rightarrow \rightarrow B$$

...

$$F \rightarrow \uparrow$$
$$F \rightarrow \leftarrow E_L$$
$$F \rightarrow \rightarrow E_R$$

$$A \Rightarrow \uparrow B$$
$$\uparrow B \Rightarrow \uparrow\leftarrow C_L$$

Príklad



$$G = (N, T, P, S)$$

$$T = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$$

$$N = \{A, B, C_L, D_L, E_L, C_R, D_R, E_R, F\}$$

$$S = A$$

Pravidlá

$$A \rightarrow \uparrow B$$

$$B \rightarrow \downarrow A$$
$$B \rightarrow \leftarrow C_L$$
$$B \rightarrow \rightarrow C_R$$

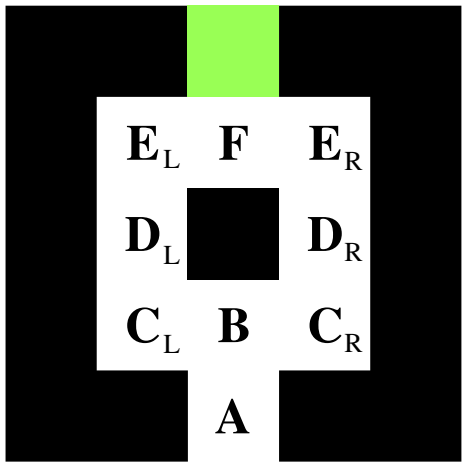
$$C_L \rightarrow \uparrow D_L$$
$$C_L \rightarrow \rightarrow B$$

...

$$F \rightarrow \uparrow$$
$$F \rightarrow \leftarrow E_L$$
$$F \rightarrow \rightarrow E_R$$

$$A \Rightarrow \uparrow B$$
$$\uparrow B \Rightarrow \uparrow\leftarrow C_L$$
$$\uparrow\leftarrow C_L \Rightarrow \uparrow\leftarrow\uparrow D_L$$

Príklad



$$G = (N, T, P, S)$$

$$T = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$$

$$N = \{A, B, C_L, D_L, E_L, C_R, D_R, E_R, F\}$$

$$S = A$$

Pravidlá

$$A \rightarrow \uparrow B$$

$$B \rightarrow \downarrow A$$
$$B \rightarrow \leftarrow C_L$$
$$B \rightarrow \rightarrow C_R$$

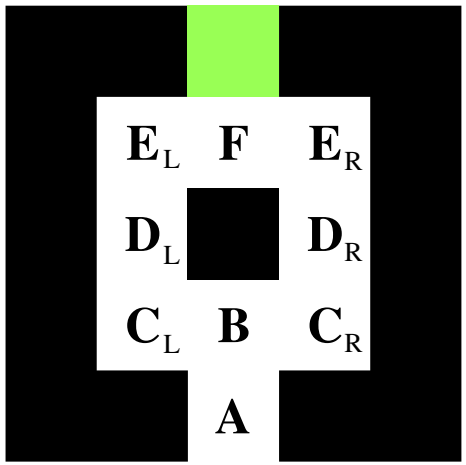
$$C_L \rightarrow \uparrow D_L$$
$$C_L \rightarrow \rightarrow B$$

...

$$F \rightarrow \uparrow$$
$$F \rightarrow \leftarrow E_L$$
$$F \rightarrow \rightarrow E_R$$

$$A \Rightarrow \uparrow B$$
$$\uparrow B \Rightarrow \uparrow\leftarrow C_L$$
$$\uparrow\leftarrow C_L \Rightarrow \uparrow\leftarrow\uparrow D_L$$
$$\uparrow\leftarrow\uparrow D_L \Rightarrow \uparrow\leftarrow\uparrow\uparrow E_L$$

Príklad



$$G = (N, T, P, S)$$

$$T = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$$

$$N = \{A, B, C_L, D_L, E_L, C_R, D_R, E_R, F\}$$

$$S = A$$

Pravidlá

$$A \rightarrow \uparrow B$$

$$B \rightarrow \downarrow A$$
$$B \rightarrow \leftarrow C_L$$
$$B \rightarrow \rightarrow C_R$$

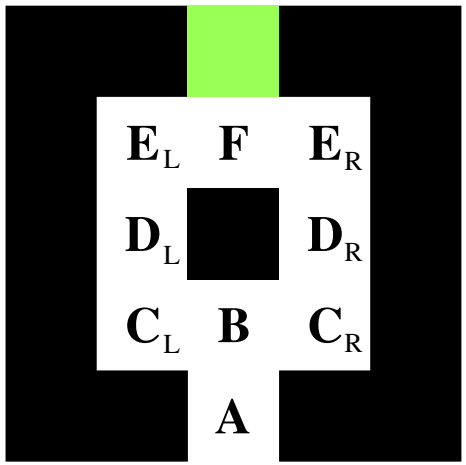
$$C_L \rightarrow \uparrow D_L$$
$$C_L \rightarrow \rightarrow B$$

...

$$F \rightarrow \uparrow$$
$$F \rightarrow \leftarrow E_L$$
$$F \rightarrow \rightarrow E_R$$

$$A \Rightarrow \uparrow B$$
$$\uparrow B \Rightarrow \uparrow\leftarrow C_L$$
$$\uparrow\leftarrow C_L \Rightarrow \uparrow\leftarrow\uparrow D_L$$
$$\uparrow\leftarrow\uparrow D_L \Rightarrow \uparrow\leftarrow\uparrow\uparrow E_L$$
$$\uparrow\leftarrow\uparrow\uparrow E_L \Rightarrow \uparrow\leftarrow\uparrow\uparrow\rightarrow F$$

Príklad



$$G = (N, T, P, S)$$

$$T = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$$

$$N = \{A, B, C_L, D_L, E_L, C_R, D_R, E_R, F\}$$

$$S = A$$

Pravidlá

$$A \rightarrow \uparrow B$$

$$B \rightarrow \downarrow A$$
$$B \rightarrow \leftarrow C_L$$
$$B \rightarrow \rightarrow C_R$$

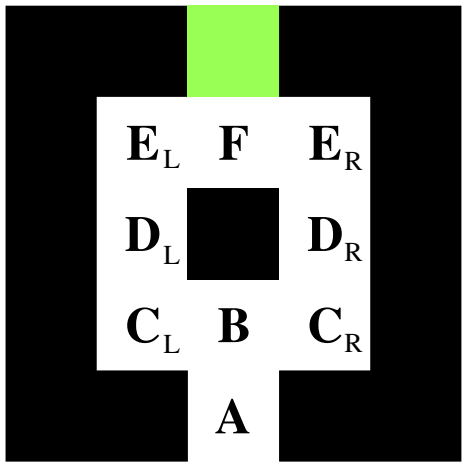
$$C_L \rightarrow \uparrow D_L$$
$$C_L \rightarrow \rightarrow B$$

$$\ldots$$

$$F \rightarrow \uparrow$$
$$F \rightarrow \leftarrow E_L$$
$$F \rightarrow \rightarrow E_R$$

$$A \Rightarrow \uparrow B$$
$$\uparrow B \Rightarrow \uparrow\leftarrow C_L$$
$$\uparrow\leftarrow C_L \Rightarrow \uparrow\leftarrow\uparrow D_L$$
$$\uparrow\leftarrow\uparrow D_L \Rightarrow \uparrow\leftarrow\uparrow\uparrow E_L$$
$$\uparrow\leftarrow\uparrow\uparrow E_L \Rightarrow \uparrow\leftarrow\uparrow\uparrow\rightarrow F$$
$$\uparrow\leftarrow\uparrow\uparrow\rightarrow F \Rightarrow \uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow$$

Príklad



$$G = (N, T, P, S)$$

$$T = \{\leftarrow, \rightarrow, \uparrow, \downarrow\}$$

$$N = \{A, B, C_L, D_L, E_L, C_R, D_R, E_R, F\}$$

$$S = A$$

Pravidlá

$$A \rightarrow \uparrow B$$

$$B \rightarrow \downarrow A$$
$$B \rightarrow \leftarrow C_L$$
$$B \rightarrow \rightarrow C_R$$

$$C_L \rightarrow \uparrow D_L$$
$$C_L \rightarrow \rightarrow B$$

$$\ldots$$

$$F \rightarrow \uparrow$$
$$F \rightarrow \leftarrow E_L$$
$$F \rightarrow \rightarrow E_R$$

$$A \Rightarrow \uparrow B$$
$$\uparrow B \Rightarrow \uparrow\leftarrow C_L$$
$$\uparrow\leftarrow C_L \Rightarrow \uparrow\leftarrow\uparrow D_L$$
$$\uparrow\leftarrow\uparrow D_L \Rightarrow \uparrow\leftarrow\uparrow\uparrow E_L$$
$$\uparrow\leftarrow\uparrow\uparrow E_L \Rightarrow \uparrow\leftarrow\uparrow\uparrow\rightarrow F$$
$$\uparrow\leftarrow\uparrow\uparrow\rightarrow F \Rightarrow \uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow$$

$$A \Rightarrow^* \uparrow\leftarrow\uparrow\uparrow\rightarrow\uparrow$$

Generatívna sila

Bezkontextová gramatika:

- výhody: jednoduchá štruktúra
- vieme efektívne rozhodnúť, či daný reťazec patrí do jazyka (tj. či je daná sekvencia riešením problému)
- môžeme pomocou nej popísať veľa užitočných jazykov (napr. syntax niektorých programovacích jazykov)
- nevýhoda: malá generatívna sila

Generatívna sila

Bezkontextová gramatika:

- výhody: jednoduchá štruktúra
- vieme efektívne rozhodnúť, či daný reťazec patrí do jazyka (tj. či je daná sekvencia riešením problému)
- môžeme pomocou nej popísať veľa užitočných jazykov (napr. syntax niektorých programovacích jazykov)
- nevýhoda: malá generatívna sila

Príklad

Jazyky, nad abecedou $\Sigma = \{a, b, c\}$, pre ktoré neexistuje bezkontextová gramatika:

- $\{a^n b^n c^n \mid n \geq 1\}$
- $\{ww \mid w \in \Sigma^*\}$

Generatívna sila

- používame mnoho formálnych modelov na popis jazykov
- najsilnejší - *turingov stroj*

Generatívna sila

- používame mnoho formálnych modelov na popis jazykov
- najsilnejší - *turingov stroj*

Dogma teoretickej informatiky

Turingov stroj reprezentuje formálny model algoritmu/počítača (tj. pre každý algoritmus existuje nejaký turingov stroj).

Generatívna sila

- používame mnoho formálnych modelov na popis jazykov
- najsilnejší - *turingov stroj*

Dogma teoretickej informatiky

Turingov stroj reprezentuje formálny model algoritmu/počítača (tj. pre každý algoritmus existuje nejaký turingov stroj).

Praktické využitie:

- problém prevedieme na jazyk
- dokážeme, že pre daný jazyk (ne)existuje bezkontextová gramatika/iný konečný model
- tým pádom problém (ne)môžeme riešiť algoritmicky

Problém zastavenia

- turingov stroj (= algoritmus) - konečný aparát
- vezmime si jazyk všetkých algoritmov, ktoré vždy skončia pre ľubovolný vstup
- je tento jazyk algoritmicky popísateľný?

Problém zastavenia

- turingov stroj (= algoritmus) - konečný aparát
- vezmime si jazyk všetkých algoritmov, ktoré vždy skončia pre ľubovolný vstup
- je tento jazyk algoritmicky popísateľný?
- odpoveď: NIE

Problém zastavenia

Nemôžeme obecne rozhodnúť, či daný algoritmus vždy skončí.

Uzáverové vlastnosti

Trieda bezkontextových jazykov (**BJ**) - všetky také jazyky, pre ktoré existuje bezkontextová gramatika.

Veta

Trieda bezkontextových jazykov nieje uzavretá voči prieniku.

Trieda bezkontextových jazykov (**BJ**) - všetky také jazyky, pre ktoré existuje bezkontextová gramatika.

Veta

Trieda bezkontextových jazykov nieje uzavretá voči prieniku.

Veta

Trieda bezkontextových jazykov je uzavretá voči zjednoteniu.

Veta

Trieda bezkontextových jazykov je uzavretá voči zjednoteniu.

Veta

Trieda bezkontextových jazykov je uzavretá voči zjednoteniu.

Dôkaz

Nech $L_1, L_2 \in \mathbf{BJ}$. Potom $L_1 \cup L_2 \in \mathbf{BJ}$.

Veta

Trieda bezkontextových jazykov je uzavretá voči zjednoteniu.

Dôkaz

Nech $L_1, L_2 \in \mathbf{BJ}$. Potom $L_1 \cup L_2 \in \mathbf{BJ}$.

- keďže $L_1, L_2 \in \mathbf{BJ}$, tak k nim existujú príslušné gramatiky $G_1 = (N_1, T_1, P_1, S_1)$ a $G_2 = (N_2, T_2, P_2, S_2)$

Veta

Trieda bezkontextových jazykov je uzavretá voči zjednoteniu.

Dôkaz

Nech $L_1, L_2 \in \mathbf{BJ}$. Potom $L_1 \cup L_2 \in \mathbf{BJ}$.

- keďže $L_1, L_2 \in \mathbf{BJ}$, tak k nim existujú príslušné gramatiky $G_1 = (N_1, T_1, P_1, S_1)$ a $G_2 = (N_2, T_2, P_2, S_2)$
- bez ujmy na obecnosti predpokladajme, že neterminály a pravidlá su disjunktné

Veta

Trieda bezkontextových jazykov je uzavretá voči zjednoteniu.

Dôkaz

Nech $L_1, L_2 \in \mathbf{BJ}$. Potom $L_1 \cup L_2 \in \mathbf{BJ}$.

- keďže $L_1, L_2 \in \mathbf{BJ}$, tak k nim existujú príslušné gramatiky $G_1 = (N_1, T_1, P_1, S_1)$ a $G_2 = (N_2, T_2, P_2, S_2)$
- bez ujmy na obecnosti predpokladajme, že neterminály a pravidlá su disjunktné
- potom môžeme zostrojiť gramatiku $G = (N, T, P, S)$ nasledovným spôsobom:

Veta

Trieda bezkontextových jazykov je uzavretá voči zjednoteniu.

Dôkaz

Nech $L_1, L_2 \in \mathbf{BJ}$. Potom $L_1 \cup L_2 \in \mathbf{BJ}$.

- keďže $L_1, L_2 \in \mathbf{BJ}$, tak k nim existujú príslušné gramatiky $G_1 = (N_1, T_1, P_1, S_1)$ a $G_2 = (N_2, T_2, P_2, S_2)$
- bez ujmy na obecnosti predpokladajme, že neterminály a pravidlá su disjunktné
- potom môžeme zostrojiť gramatiku $G = (N, T, P, S)$ nasledovným spôsobom:

$$\begin{aligned} N &= N_1 \cup N_2 \cup \{S\} \\ T &= T_1 \cup T_2 \\ P &= P_1 \cup P_2 \cup \{S \to S_1, S \to S_2\} \end{aligned}$$

Veta

Trieda bezkontextových jazykov je uzavretá voči zjednoteniu.

Dôkaz

- $G = (N, T, P, S)$, kde

$$\begin{aligned} N &= N_1 \cup N_2 \cup \{S\} \\ T &= T_1 \cup T_2 \\ P &= P_1 \cup P_2 \cup \{S \to S_1, S \to S_2\} \end{aligned}$$

Veta

Trieda bezkontextových jazykov je uzavretá voči zjednoteniu.

Dôkaz

- $G = (N, T, P, S)$, kde

$$\begin{aligned} N &= N_1 \cup N_2 \cup \{S\} \\ T &= T_1 \cup T_2 \\ P &= P_1 \cup P_2 \cup \{S \to S_1, S \to S_2\} \end{aligned}$$

- prvé pravidlo musí byť z množiny $\{S \to S_1, S \to S_2\}$

Veta

Trieda bezkontextových jazykov je uzavretá voči zjednoteniu.

Dôkaz

- $G = (N, T, P, S)$, kde

$$\begin{aligned} N &= N_1 \cup N_2 \cup \{S\} \\ T &= T_1 \cup T_2 \\ P &= P_1 \cup P_2 \cup \{S \to S_1, S \to S_2\} \end{aligned}$$

- prvé pravidlo musí byť z množiny $\{S \to S_1, S \to S_2\}$
- ďalej derivujeme buď podľa P_1, alebo P_2 (neterminály a pravidlá sú disjunktné)

Veta

Trieda bezkontextových jazykov je uzavretá voči zjednoteniu.

Dôkaz

- $G = (N, T, P, S)$, kde

$$\begin{aligned} N &= N_1 \cup N_2 \cup \{S\} \\ T &= T_1 \cup T_2 \\ P &= P_1 \cup P_2 \cup \{S \to S_1, S \to S_2\} \end{aligned}$$

- prvé pravidlo musí byť z množiny $\{S \to S_1, S \to S_2\}$
- ďalej derivujeme buď podľa P_1, alebo P_2 (neterminály a pravidlá sú disjunktné)

Výsledný jazyk $L(G) = L_1 \cup L_2$, a keďže G je zjavne bezkontextová gramatika, tak $L(G) \in \mathbf{BJ}$

Príklad

$$G_1: \begin{array}{l} S_1 \to aS_1b \\ S_1 \to ab \end{array}$$

$$L_1 = \{a^n b^n \mid n \geq 1\}$$

$$G_2: \begin{array}{l} S_2 \to acS_2 \\ S_2 \to ac \end{array}$$

$$L_2 = \{(ac)^n \mid n \geq 1\}$$

$$G: \begin{array}{ll} S \to S_1 & S \to S_2 \\ S_1 \to aS_1b & S_2 \to acS_2 \\ S_1 \to ab & S_2 \to ac \end{array}$$

Veta

Trieda bezkontextových jazykov nieje uzavretá voči prieniku.

Veta

Trieda bezkontextových jazykov nieje uzavretá voči prieniku.

Dôkaz

Ukážeme dva jazyky — $L_1, L_2 \in \mathbf{BJ}$ — ich prienik nepatrí do **BJ**.

Veta

Trieda bezkontextových jazykov nieje uzavretá voči prieniku.

Dôkaz

Ukážeme dva jazyky — $L_1, L_2 \in$ **BJ** — ich prienik nepatrí do **BJ**.

$$L_1 = \{a^n b^n c^m \mid n, m \geq 1\}$$

Veta

Trieda bezkontextových jazykov nieje uzavretá voči prieniku.

Dôkaz

Ukážeme dva jazyky — $L_1, L_2 \in$ **BJ** — ich prienik nepatrí do **BJ**.

$$L_1 = \{a^n b^n c^m \mid n, m \geq 1\}$$

- Pravidlá príslušnej gramatiky:
  $S \to AC$
  $A \to aAb \quad A \to ab$
  $C \to Cc \quad C \to c$

Veta

Trieda bezkontextových jazykov nieje uzavretá voči prieniku.

Dôkaz

Ukážeme dva jazyky — $L_1, L_2 \in$ **BJ** — ich prienik nepatrí do **BJ**.

$$L_1 = \{a^n b^n c^m \mid n, m \geq 1\}$$

- Pravidlá príslušnej gramatiky:
  $S \to AC$
  $A \to aAb \quad A \to ab$
  $C \to Cc \quad C \to c$
- tým pádom $L_1 \in$ **BJ**.
- podobne pre $L_2 = \{a^m b^n c^n \mid n, m \geq 1\}$

Veta

Trieda bezkontextových jazykov nieje uzavretá voči prieniku.

Dôkaz

- $L_1 = \{a^n b^n c^m \mid n, m \geq 1\}$
- $L_2 = \{a^m b^n c^n \mid n, m \geq 1\}$

Veta

Trieda bezkontextových jazykov nieje uzavretá voči prieniku.

Dôkaz

- $L_1 = \{a^n b^n c^m \mid n, m \geq 1\}$
- $L_2 = \{a^m b^n c^n \mid n, m \geq 1\}$
- $L_1 \cap L_2 = \{a^n b^n c^n \mid n \geq 1\}$

Veta

Trieda bezkontextových jazykov nieje uzavretá voči prieniku.

Dôkaz

- $L_1 = \{a^n b^n c^m \mid n, m \geq 1\}$
- $L_2 = \{a^m b^n c^n \mid n, m \geq 1\}$
- $L_1 \cap L_2 = \{a^n b^n c^n \mid n \geq 1\}$
- avšak pre jazyk $\{a^n b^n c^n \mid n \geq 1\}$ neexistuje bezkontextová gramatika

Veta

Trieda bezkontextových jazykov nieje uzavretá voči prieniku.

Dôkaz

- $L_1 = \{a^n b^n c^m \mid n, m \geq 1\}$
- $L_2 = \{a^m b^n c^n \mid n, m \geq 1\}$
- $L_1 \cap L_2 = \{a^n b^n c^n \mid n \geq 1\}$
- avšak pre jazyk $\{a^n b^n c^n \mid n \geq 1\}$ neexistuje bezkontextová gramatika
- takže $L_1 \cap L_2 \notin \mathbf{BJ}$

Záver

Hlavné myšlienky:

- jazyk - formalizácia problému
- počítač je konečný, preto používame konečné prostriedky pre popis jazykov
- pomocou nich skúmame, ktoré problémy môžeme riešiť algoritmicky

Ďakujem za pozornosť