Algoritmuselmélet
1. heti feladatsor megoldás

1. Tekintsük a $3n^2 \log n + 2019n - 27$ függvényt.
Igaz-e, hogy ez a függvény
(a) $O(n^2)$-ben van?
(b) $O(n^3)$-ben van?
(c) $\Omega(n^3)$-ben van?

Amelyik állítás igaz, annál ezt egy megfelelő c konstans és n_0 küszöbérték megadásával igazolja, ahol pedig az állítás nem igaz, ott ezt bizonyítsa be indirekt bizonyítással.

Megoldás

(a) Ez az állítás nem igaz, ezt indirekt bizonyítással látjuk be:

Tegyük fel, hogy van olyan $c > 0$ és $n_0 \geq 1$, hogy

$$3n^2 \log n + 2019n - 27 \leq cn^2$$

igaz, ha $n \geq n_0$.

Ekkor (mivel $2019n - 27 > 0$ minden $n \geq 1$ esetén)

$$3n^2 \log n \leq 3n^2 \log n + 2019n - 27 \leq cn^2$$

is igaz, ha $n \geq n_0$.

Ezt elosztva n^2-tel kapjuk a nyilvánvaló ellentmondást:

$$3 \log n \leq c$$

teljesül, ha $n \geq n_0$ ami lehetetlen, mert ha n tart a végtelenbe, akkor $\log n$ nem maradhat a c korlát alatt.

(b) Ez az állítás igaz, megadunk olyan c és n_0 párt, amivel $3n^2 \log n + 2019n - 27 \leq cn^3$, ha $n \geq n_0$:

$$3n^2 \log n + 2019n - 27 \leq 3n^3 + 2019n \leq 3n^3 + 2019n^3 = 2022n^3$$

ahol mindegyik egyenlőtlenség igaz, ha $n \geq 1$, vagyis a definíció szerinti c és n_0 értékek: $c = 2022, n_0 = 1$.

(c) Ez az állítás nem igaz, ezt indirekt bizonyítással látjuk be:

Tegyük fel, hogy van olyan $c > 0$ és $n_0 \geq 1$, hogy

$$3n^2 \log n + 2019n - 27 \geq cn^3$$

igaz, ha $n \geq n_0$.

Ekkor

$$2022n^2 \log n = 3n^2 \log n + 2019n^2 \log n \geq 3n^2 \log n + 2019n \geq 3n^2 \log n + 2019n - 27 \geq cn^3$$

is igaz, ha $n \geq max(n_0, 2)$ (mert az első egyenlőtlenség $n \geq 2$ esetén áll fenn).

Ezt elosztva n^2-tel kapjuk az ellentmondást:

$$2022 \log n \geq cn$$

teljesül, ha $n \geq max(n_0, 2)$ ami lehetetlen, mert $\dfrac{\log n}{n}$ nullához tart, ha n tart a végtelenbe, nem maradhat mindig nagyobb, mint egy pozitív konstans.

2. Tekintsük az $\{a{,}b\}$ ábécé feletti $L = \{a^n b^{n+2} \mid n \geq 0\}$ nyelvet (azaz a nyelvben benne van a bb szó is).
   (a) Vázoljon szavakkal egy veremautomatát erre a nyelvre.
   (b) Adja meg precízen a veremautomatát (szabályokkal vagy ábrával).
   (c) Mutassa meg, hogy az automata pontosan L szavait fogadja el.

1. megoldás

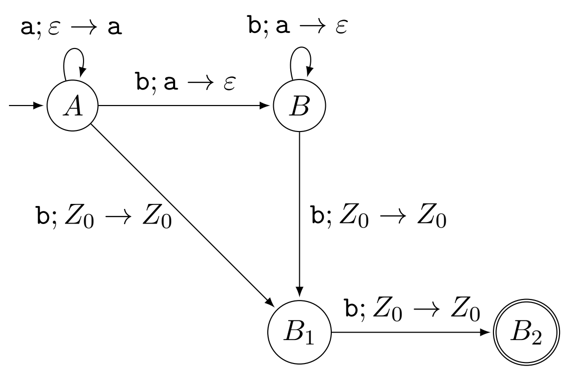
(a)

- A kezdőállapotban amíg a-kat olvasunk, addig beteszünk minden olvasott a-ra egy a-t a verembe (szabályok leírásánál (1) szabály)
- Amikor jönni kezdenek a b-k, akkor átmegyünk B állapotba és minden b-re kiveszünk egy a-t a veremből, ha tudunk, azaz, ha van a a veremben. (szabályok leírásánál (2) és (3) szabály)
- Ha elfogynak a veremből az a-k (azaz a verem alját látjuk), akkor egy első így olvasott b-re B_1, a második így olvasott b-re B_2 állapotba megyünk, ez a B_2 lesz az egyetlen elfogadó állapot.
  (a (4)-es és (5)-ös szabály kezeli azt az esetet, amikor volt legalább egy a a szó elején, a (6)-os szabály pedig azt a helyzetet, amikor nem volt a a szóban).

(b) Szabályokkal ez így néz ki:

(1) $\delta(A, a, \varepsilon) = (A, a)$
(2) $\delta(A, b, a) = (B, \varepsilon)$
(3) $\delta(B, b, a) = (B, \varepsilon)$
(4) $\delta(B, b, Z_0) = (B_1, Z_0)$
(5) $\delta(B_1, b, Z_0) = (B_2, Z_0)$
(6) $\delta(A, b, Z_0) = (B_1, Z_0)$

Ugyanez ábrával:



(c) A nyelv szavait elfogadja az automata:
Ha van legalább egy a a szó elején, azaz $a^n b^{n+2}$-ben $n \geq 1$:

A állapotban olvasva az összes a-t a verembe bekerül n darab a a Z_0 verem alja jel fölé (az (1) szabályt használva. Ezután a (2) szabályt használva az első b-re kiveszünk egy a-t a veremből, majd a (3) szabállyal kiszedjük a többi a-kat is és mire elolvassuk a szó elején álló $a^n b^n$ részt, addigra a verem kiürül, csak Z_0 lesz benne. Ekkor az utolsó két b hatására a (4)-es és (5)-ös szabállyal eljutunk B_2-be, ami elfogadó állapot és mivel eddigre az input is elfogyott ezért az $a^n b^{n+2}$ szót elfogadjuk.

Ha nincsen a a szó elején, azaz a bb az input, akkor a (6)-os és az (5)-ös szabályt használva jutunk el B_2-be mire a bb inputot elolvassuk.

Más szót nem fogad el az automata:

Ha a szó nem a-val kezdődik, akkor csak úgy tudunk B_2-be jutni mire elolvassuk a szót, ha a szó maga bb. Tehát egyébként a szónak a-val kell kezdődnie. Ha nem jön b az a-k után, akkor A-ban ragadunk, azaz nem fogadjuk el a szót, mert A nem elfogadó állapot.

Ha az a^*b^* rész után még jönne a, akkor elakadunk B_1 vagy B_2 állapotban, nem olvassuk el a szót, tehát el se fogadjuk.

Az derült ki tehát, hogy (kivéve a bb szót) csak $a^i b^j$ alakú szavaknak van esélye elfogadásra, ahol i,$j \geq 1$.
Ha $j < i + 2$, akkor vagy ki sem tudjuk szedni a veremből az összes a-t (ha $j < i$) vagy ki tudjuk szedni az a-kat, de már nincs két b az inputon, amivel eljutnánk a B_2 elfogadó állapotig.
Ha $j > i + 2$, akor pedig az lesz a baj, hogy miután kiszedem az a-kat a veremből még legalább 3 b van az inputon, de csak kettőt tudok elolvasni, B_2-ben elaladok úgy, hogy a további b-ket nem olvastam el, azaz a szót nem fogadjuk el, mert nem olvastuk végig.

2. megoldás

Csinálunk egy CF nyelvtant erre a nyelvre és az órán tanult eljárással veremautomatává alakítjuk. Ekkor csak azt kell megindokolni, hogy a nyelvtan éppen a nyelv szavait generálja, mert azt már tudjuk, hogy az órai eljárás a veremautomata előállítására helyes.

Egy jó nyelvtan például a következő:

$$S \to aSb \,|\, bb$$

.

Ez minden $a^n b^{n+2}$ alakú szót generálni tud úgy, hogy n-szer használjuk az első szabályt és egyszer a másodikat.

Másrészről csak $a^n b^{n+2}$ alakú szavakat tud generálni, mert az összes levezetési fa olyan, hogy először valahányszor használjuk az első szabályt és ezzel levezetünk egy $a^n S b^n$ szót, majd a második szabállyal befejezzük a levezetést, de az ilyen levezetések éppen az $a^n b^{n+2}$ alakú szavakat generálják.

Ebből a nyelvtanból a tanult konstrukcióval kapjuk az alábbi veremautomatát, ahol q_e az egyetlen elfogadó állapot:

$\delta(q_0, \varepsilon, Z_0) = (q, SZ_0)$
$\delta(q, \varepsilon, S) = \{(q, aSb), (q,bb)\}$
$\delta(q, a, a) = (q, \varepsilon)$
$\delta(q, b, b) = (q,\varepsilon)$
$\delta(q, \varepsilon, Z_0) = (q_e,Z_0)$

Itt az első szabály berakja a verembe a kezdő szimbólumot, a második szabály lépi meg a nyelvtan szabályait, a harmadik és negyedik szabály felismeri, ha éppen olyan karaktert generáltunk a verem tetejére, ami az inputon jön, az utolsó szabály pedig felismeri, ha a verem kiürült.