Algoritmus na rozpoznávání formulí výrokové logiky

Následující algoritmus předpokládá, že formule obsahuje všechny závorky, včetně závorek kolem negace a včetně vnějších závorek.

Konečnou posloupnost prvků z množiny $\mathcal{A}$, logických spojek a závorek nazýváme *formule* výrokové logiky, jestliže vznikla podle následujících pravidel:

1. Každá atomická formule $\varphi \in \mathcal{A}$ je výroková formule.
2. Jsou-li φ a ψ výrokové formule, pak $(\neg\varphi)$, $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, $(\varphi \Rightarrow \psi)$ a $(\varphi \Leftrightarrow \psi)$ jsou výrokové formule.
3. Nic jiného než to, co vzniklo konečným použitím bodů 1 a 2, není výroková formule.

Algorithm 1 Test formule — vrátí true, jestliže řetězec je formule, jinak vrátí false

```
 1: procedure FORMULE(str)
 2:     cnt ← 0
 3:     l ← délka str
 4:     if l = 1 and str[0] ∈ A then return true
 5:     else if str[0] = '(' and str[l − 1] = ')' then
 6:         i ← 0
 7: loop:
 8:         if i = l then return false
 9:         if str[i] = '(' then
10:             cnt ← cnt + 1
11:             i ← i + 1
12:         else if str[i] = ')' then
13:             cnt ← cnt − 1
14:             i ← i + 1
15:         else if cnt = 1 then
16:             if i = 1 and str[i] = '¬' then
17:                 substr ← str [2..l − 2]          ▷ do substr se přiřadí znaky str[2] až str[l-2]
18:                 return FORMULE(substr)
19:             else if str[i] ∈ {' ∧ ', ' ∨ ', ' ⇒ ', ' ⇔ '} then
20:                 lstr ← str [1..i − 1]            ▷ do lstr se přiřadí znaky str[1] až str[i-1]
21:                 rstr ← str [i + 1..l − 2]        ▷ do rstr se přiřadí znaky str[i+1] až str[i-2]
22:                                  ▷ Například str='((a∧b)⇒c)', pak lstr='(a∧b)' a rstr='c'
23:                 return FORMULE(lstr) and FORMULE(rstr)
24:             else
25:                 i ← i + 1
26:         else
27:             i ← i + 1
28:         goto loop
29:     else
30:         return false
```

Komentář k algoritmu:

Jedná se o rekurzivní algoritmus, který rozpoznává, zda je řetězec *str* formulí výrokové logiky.

Řetězec délky $l = 1$ rozpozná přímo: je formulí právě, když je to znak z množiny $\mathcal{A}$ (aneb je to atomická formule).

Řetězec délky $l > 1$ může být formulí jen, když začíná levou závorkou a končí pravou závorkou. V tomto případě se postupně čtou znaky řetězce a počítá se počet levých a pravých závorek. Přesněji *cnt* = počet levých závorek minus počet pravých závorek až do právě čteného znaku. V okamžiku, kdy je *cnt* = 1 a čtený znak je logická spojka, jsme nalezli kořen odvozovacího stromu dané formule. Tehdy se umažou vnější závorky a pro levý a pravý podřetězec (u binárních spojek), resp. pro pravý podřetězec (u unární spojky) se rekurzivně rozhoduje, zda se jedná o formule.