23 - Formální jazyk (FJ)

(matematické základy teorie FJ, binární relace na množině – reprezentace, vlastnosti, tranzitivní a reflexivní uzávěr, význačné relace. Abeceda, řetězec, jazyk, operace nad jazyky, reprezentace jazyků, gramatika a automat, třídy formálních jazyků podle Chomského klasifikace. Lineární jazyky)

Matematické základy teorie FJ

- Potenční množina - množina všech podmnožin.
- Kartézský součin - množina všech uspořádaných dvojic, kde první prvek patří do první množiny a druhý prvek do druhé množiny.
- Relace - podmnožina kartézského součinu.
- Zobrazení - speciální případ binární relace, kde je zajištěna jednoznačnost obrazu ke každému vzoru.

Binární relace na množině

- Reprezentace - podmnožina kartézského součinu dvou stejných množin.
- Vlastnosti
    - Reflexivita - aRa
    - Tranzitivita - když aRb a bRc, tak aRc
    - Symetrie - když aRb, tak bRa
    - Antisymetrie - když aRb a bRa, tak a = b
    - Asymetrie - když aRb, tak není bRa
    - Úplná - aRb nebo bRa
- Tranzitivní a reflexivní uzávěr - uzávěr je nejmenší doplnění relace, aby splňovala danou vlastnost
    - Tranzitivní uzávěr je R^+ a tranzitivní a reflexivní uzávěr je R^*
- Význačné relace
    - Uspořádání - reflexivní, tranzitivní a antisymetrická relace.
    - Ekvivalence - reflexivní, tranzitivní a symetrická relace.

Abeceda - neprázdná množina symbolů.

Řetězec - konečná posloupnost symbolů dané abecedy (slovo).

Jazyk - podmnožina množiny všech řetězců nad danou abecedou.

Operace nad jazyky - sjednocení, průnik, rozdíl, doplněk (jazyk je množina), konkatenace, iterace (prvky jazyka jsou řetězce).

Reprezentace jazyků - výčtem všech řetězců, množinovým zápisem, formální gramatikou, automatem.

Gramatika a automat

- Gramatika - množina neterminálních symbolů, množina terminálních symbolů, množina přepisovacích pravidel a počáteční neterminál. G = (N, E, P, S).
- Automat (konečný) - virtuální stroj skládající se z řídící jednotky a čtecího zařízení. Množina stavů, abeceda, přechodová funkce, počáteční stav a množina koncových stavů. Dokáže přijmout či zamítnout vstupní slovo, čímž ověří, že slovo patří do daného jazyka.

Třídy formálních jazyků podle Chomského klasifikace

- Typ 0 - na tvar přepisovacích pravidel nejsou kladeny žádné požadavky, jazyky generované touto neomezenou gramatikou přijíma turingův stroj.
- Typ 1 - pravidla ve tvaru αXβ -> αwβ, kde alfa a beta jsou kontext (řetězce terminálů a neterminálů), dle kterého se určí na jaký řetězec terminálů a neterminálů w se přepíše neterminál X. Tuto kontextovou gramatiku přijímá lineárně ohraničený turingův stroj.
- Typ 2 - pravidla ve tvaru X -> w, tedy kontext je prázdný řetězec. Tuto bezkontextovou gramatiku přijímá zásobníkový automat.
- Typ 3 - pravidla ve tvaru A -> xB|x. Tuto regulární gramatiku přijímá konečný automat.

Lineární jazyky - jazyky generované lineární gramatikou, která je ekvivalentní s regulární gramatikou (dá se na ni vždy převést), ale v přepisovacích pravidlech obsahuje místo jednoho terminálu řetězec terminálů.

24 - Konstrukce překladačů

(interpret, kompilátor, lexikální analyzátor, syntaktická analýza metodou rekurzivního sestupu, sémantická analýza, implementace sémantických akcí)

Interpret - program, který provádí (interpretuje) vstupní zdrojový kód.

Kompilátor - program, který přeloží vstupní zdrojový kód do jiného jazyka (většinou strojového kódu).

- Zdrojový kód -> Lexikální analýza -> Řetězec tokenů -> Syntaktická analýza -> Derivační strom -> Sémantická analýza -> Abstraktní syntaktický strom -> Generování vnitřního kódu -> Vnitřní kód -> Optimalizátor -> Optimalizovaný vnitřní kód -> Generování cílového kódu -> Cílový program

Lexikální analyzátor - v podstatě jde o to rozsekat tok znaků na jednotlivé tokeny (slova) a přiřadit jim význam (2 je číslo, *x* je proměnná, + je operátor, …). Často se k tomu používají regulární výrazy nebo konečný automat.

Syntaktická analýza metodou rekurzivního sestupu - cílem je zkontrolovat, že jednotlivé tokeny jdou správně za sebou a dávají tak smysl. Výsledkem je tzv. syntaktický (derivační) strom. Pokud takový strom nalezneme, program je správný, jinak ne. Vytváření derivačního stromu je založeno na bezkontextových gramatikách a zásobníkových automatech.

- Shoda-dolů - Probíhá tak, že se symboly v gramatice postupně rozgenerovávají, takže z počátečního symbolu vznikne posloupnost jiných symbolů podle gramatiky. Používá se častěji.
    - LL syntaktický analyzátor - analyzuje vstup zleva doprava a konstruuje nejlevější derivaci věty.
    - Rekurzivním sestup - sestavuje se derivační strom pomocí rekurzivních volání, kde rekurzivní operace implementuje pravidlo gramatiky.
        - Prediktivní syntaktický analyzátor - rozhodne, které pravidlo použít na základě prohlédnutí dalších symbolů. Lineární čas, ale nesmí obsahovat nejednoznačné gramatiky.
        - Syntaktický analyzátor s backtrackingem - zkouší se každé přepisovací pravidlo. Exponenciální čas.
- Zdola-nahoru - naopak, z posloupnosti symbolů se snažíme udělat počáteční symbol. Příkladem může být precedenční analýza řízená precedenční tabulkou, která udává prioritu a asociativitu všech operátorů, pro zpracování výrazu.

Sémantická analýza - sémantický analyzátor kontroluje sémantické aspekty programu, například kontrola typů, při které může provádět implicitní konverze (např. int-to-real) nebo

kontrola deklarací proměnných. Výstupem je syntaktický strom, který však nereprezentuje každý detail reálné syntaxe, například seskupující závorky či if-podmínka-else může být označena jedním uzlem. Také může být doplněn o další informace, takovému stromu se proto říká abstraktní.

Implementace sémantických akcí - Sémantické akce jsou aktivity implementovány ve vhodném místě syntaktického analyzátoru (např. kontrola deklarací, typová kontrola atd.) Po provedení vhodné sémantické akce se pokračuje v průchodu syntaktické analýzy.