VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Multigenerativní gramatické systémy

Disertační práce

Roman Lukáš

Školitel: prof. RNDr. Alexander Meduna, CSc.

Datum státní doktorské zkoušky: 15.6.2005

Datum odevzdání práce: 15.6.2006

Abstrakt

Disertační práce se zabývá především tzv. generováním multiřetězců pomocí různých druhů multigenerativních gramatických systémů.

Multigenerativní gramatický systém je založen na kooperativní činnosti konečného počtu bezkontextových gramatik. Všechny tyto bezkontextové gramatiky paralelně a synchronně derivují jednotlivé větné formy. V průběhu generování dochází v každém přímém derivačním kroku ke kontrole správnosti jednotlivých vygenerovaných větných forem. Tyto kontroly mohou být provedeny různými způsoby. Výsledkem je potom tzv. multiřetězec (vektor řetězců), pomocí kterého je definován generovaný jazyk.

Klíčová slova

Bezkontextová gramatika, maticová gramatika, gramatický systém, multiforma, multiřetězec.

Poděkování

Na tomto místě bych chtěl poděkovat prof. RNDr. Alexandru Medunovi, CSc., vedoucímu disertační práce, za odbornou pomoc při konzultacích, za ochotu a čas, který mi věnoval při tvorbě této disertační práce.

Obsah

1 Úvod

Hlavním úkolem teoretické informatiky je popsat pomocí různých druhů formálních modelů tzv. formální jazyky. Typickým modelem pro popis formálního jazyka je například gramatika. V teoretické informatice byly zavedeny různé typy gramatik. Tyto typy gramatik se liší především ve složitosti tvaru pravidel, které jsou v nich obsaženy. Ve většině případů platí, že čím složitější tvar pravidel v gramatice povolíme, tím rozsáhlejší třídu jazyků pomocí těchto gramatik můžeme popsat. Problém ale nastává z hlediska implementace. Tam zase naopak platí, že čím jednodušší tvar pravidel povolíme, tím jednodušeji se dá tento gramatický model naimplementovat. Proto má smysl se v rámci teoretické informatiky zamyslet, zda nelze v gramatice povolit pouze jednodušší tvar pravidel a určitým „vylepšením" přesto popsat rozsáhlejší třídu jazyků. Toho lze dosáhnout například přidáním jistého řídicího systému k dané gramatice. Další možností je použít více gramatik s jednodušším tvarem pravidel současně. Na této myšlence jsou založeny různé typy tzv. gramatických systémů. Jedná se obecně o n-tici gramatik, které spolu určitým způsobem komunikují.

V teorii formálních jazyků již byly zavedeny různé druhy gramatických systémů (viz. [2], [3], [4]) obsahující několik kooperujících komponent, které jsou většinou založeny na činnosti tzv. bezkontextových gramatik. Tyto gramatické systémy jsou většinou navrženy tak, že sice každá komponenta generuje jiný řetězec, ale pro generovaný jazyk je důležitý pouze řetězec generovaný právě ***první komponentou***. Ostatní řetězce slouží jen jako kontrolní elementy v průběhu generování, ale jejich výsledný obsah se do generovaného jazyka nijakým způsobem nepromítne.

V této práci jsou různé druhy gramatických systémů navrženy tak, že pro výsledný jazyk jsou podstatné vygenerované řetězce ***ze všech komponent***. S vygenerovanou n-ticí řetězců se potom udělá nějaká operace, pomocí které se vytvoří pouze jeden řetězec, a ten potom bude hrát klíčovou roli pro výsledný jazyk.

Tato práce je rozdělena do následujících logických celků:

- První kapitola obsahuje samotný úvod. Tato část čtenáře neformálně uvede do problematiky gramatických systémů.

- Ve druhé kapitole je čtenář seznámen se základními pojmy a definicemi z teorie formálních jazyků, které jsou dále používány v ostatních kapitolách. Tato kapitola je poslední, která rekapituluje zavedené pojmy. Následující kapitoly obsahují samotný výzkum autora.

- Ve třetí kapitole jsou zavedeny různé druhy tzv. ***kanonických multigenerativních gramatických systémů***. Tyto systémy jsou založeny na bezkontextových gramatikách, přičemž každá z těchto gramatik generuje řetězce pouze pomocí tzv. ***nejlevější derivace***, což znamená, že v každém derivačním kroku je vždy přepsán pouze nejlevější nonterminální symbol. Přesněji bychom mohli specifikovat tyto multigenerativní gramatické systémy následovně: Pro každé přirozené číslo n kanonický n-generativní gramatický systém se skládá z n bezkontextových gramatik, které paralelně generují řetězce pomocí nejlevější derivace. Tyto derivace jsou průběžně kontrolovány n-ticí nonterminálních symbolů nebo n-ticí pravidel. Pomocí těchto kontrol gramatický systém generuje n-tice řetězců, se kterými jsou provedeny nějaké základní operace. Pomocí výsledného řetězce je definován generovaný jazyk. Mezi tyto operace patří především sjednocení, konkatenace a výběr řetězce, který generuje první komponenta. Jádrem této kapitoly je ukázat, jaké platí mezi těmito různými typy multigenerativních gramatických systémů vztahy a jakou mají obecně tyto multigenerativní gramatické systémy generativní sílu. To vše je zkoumáno v závislosti na:

   - druhu provedení kontroly (zda se provádí kontrola n-ticí nonterminálních symbolů nebo n-ticí pravidel),
   - typu zvolené operace, která je provedena s n-ticí řetězců,
   - na počtu komponent (počtu bezkontextových gramatik), ze kterých se gramatický systém skládá.

- Ve čtvrté kapitole jsou zavedeny různé druhy tzv. ***obecných multigenerativních gramatických systémů***. Tyto systémy jsou založeny na bezkontextových gramatikách jako systémy zavedené v předchozí kapitole s tím rozdílem, že každá z těchto gramatik generuje řetězce pomocí tzv. ***obecné derivace***, což znamená, že v každém derivačním kroku je vždy přepsán libovolný nonterminální symbol. Jádrem této kapitoly je opět ukázat, jaké platí vztahy mezi těmito typy multigenerativních gramatických systémů. To vše je zkoumáno v závislosti na:
   - druhu provedení kontroly (zda se provádí kontrola n-ticí nonterminálních symbolů nebo n-ticí pravidel),
   - typu zvolené operace, která je provedena s n-ticí řetězců,
   - na počtu komponent (počtu bezkontextových gramatik), ze kterých se gramatický systém skládá.

- V páté kapitole jsou zavedeny tzv. ***hybridní multigenerativní gramatické systémy***. Tyto systémy vznikly kombinací dvou předchozích typů multigenerativních gramatických systémů. Znamená to tedy, že nějaké gramatiky generují řetězce pomocí ***obecné derivace***, jiné pomocí ***nejlevější derivace***. Tyto systémy byly zavedeny především z důvodu potřeby praxe. (viz. dále)

- V šesté kapitole je gramatický systém z předchozí kapitoly chápán jiným pohledem. Ne jako multigenerátor několika řetězců současně, ale jako speciální překladač, ve kterém z n-tice komponent právě jedna komponenta provádí syntaktickou analýzu vstupního řetězce a tato syntaktická analýza řídí současně generování výstupních řetězců u zbývajících n–1 komponent. Využití takového gramatického systému může být například při překladu jistého assemblerovského programu do více binárních kódů pro různé procesory současně. Pozornost je také věnována na různá omezení těchto systémů, aby překlad mohl probíhat deterministicky. V této kapitole jsou navrženy různé druhy algoritmů, které deterministickým způsobem mohou provádět tento překlad. Pozornost je věnována jak metodě překladu shora dolů, tak metodě zdola nahoru.

- V sedmé kapitole je ukázáno, jakým způsobem může být hybridní multigenerativní gramatický systém využit v praxi jako model překladače obecného assembleru a disassembleru. Pomocí tzv. jazyku ISAC je popsána instrukční sada assembleru a její binární kódování pro daný procesor. Základní myšlenka je následující: Popis této instrukční sady pomocí jazyka ISAC je převeden na ekvivalentní hybridní multigenerativní gramatický systém (zavedený v páté kapitole). Na bázi tohoto modelu mohou být automaticky vygenerovány dva překladače:

- assembler (provádí překlad assemblerovské instrukce do binárního kódu)
- disassembler (provádí překlad z binárního kódu instrukce do assemblerovského formátu)

Činnost obou typů překladačů je potom založena na algoritmech zavedených v šesté kapitole.

2 Základní definice a pojmy

2.1 Definice základních pojmů

2.1.1 Definice abecedy

Abeceda je neprázdná konečná množina prvků, které nazýváme symboly.

2.1.2 Definice řetězce nad danou abecedou

Nechť Σ je abeceda. Potom:

- ε je řetězec nad abecedou Σ.
- Jestliže x je řetězec nad abecedou Σ, $a \in \Sigma$, potom xa je řetězec nad abecedou Σ.

Poznámky:

1) Symbol ε značí *prázdný řetězec*. Prázdný řetězec je takový řetězec, který neobsahuje žádný symbol.
2) Symbolem Σ^* budeme značit množinu všech řetězců nad abecedou Σ.

2.1.3 Definice délky řetězce

Nechť x je řetězec nad abecedou Σ. Délka řetězce x, $|x|$, je definována následovně:

- Pokud $x = \varepsilon$, potom $|x| = 0$.
- Pokud $x = a_1a_2 \ldots a_n$, kde $a_i \in \Sigma$ pro všechna $i = 1, \ldots, n$, potom $|x| = n$.

2.1.4 Definice binární operace konkatenace

Nechť x, y jsou dva řetězce nad abecedou Σ. Konkatenací řetězce x s řetězcem y vznikne řetězec xy (připojením řetězce y za řetězec x). Operace konkatenace je asociativní, ale není komutativní.

2.1.5 Definice reverzace řetězce

Nechť x je řetězec nad abecedou Σ. Reverzace řetězce x, $reverse(x)$, je definována následovně:

- Pokud $x = \varepsilon$, potom $reverse(x) = \varepsilon$.
- Pokud $x = a_1a_2 \ldots a_n$, kde $a_i \in \Sigma$ pro všechna $i = 1, \ldots, n$, potom $reverse(x) = a_n \ldots a_2a_1$.

2.1.6 Definice prefixu řetězce

Nechť x, y jsou dva řetězce nad abecedou Σ. x nazveme *prefixem* řetězce y, pokud existuje řetězec z nad abecedou Σ, pro který platí: $xz = y$.

2.1.7 Definice sufixu řetězce

Nechť x, y jsou dva řetězce nad abecedou Σ. x nazveme *sufixem* řetězce y, pokud existuje řetězec z nad abecedou Σ, pro který platí: $zx = y$.

2.1.8 Definice podřetězce

Nechť x, y jsou dva řetězce nad abecedou Σ. x nazveme *podřetězcem* řetězce y, pokud existují řetězce z, z' nad abecedou Σ, pro které platí: $zxz' = y$.

2.1.9 Definice formálního jazyka

Nechť je dána abeceda Σ. Potom množinu L, pro kterou platí $L \subseteq \Sigma^*$, nazveme *formálním jazykem* nad abecedou Σ.

2.1.10 Definice překladu

Nechť Σ, Ω jsou abecedy. Σ budeme nazývat tzv. *vstupní abecedou*, Ω budeme nazývat tzv. *výstupní abecedou*. *Překladem* jazyka $L_{in} \subseteq \Sigma^*$ do jazyka $L_{out} \subseteq \Omega^*$ nazveme libovolnou relaci τ z L_{in} do L_{out}. Jazyk L_{in} nazveme jazykem vstupním, jazyk L_{out} nazveme jazykem výstupním. Pokud pro řetězce $x \in L_{in}$ a $y \in L_{out}$ platí $y \in \tau(x)$, pak řekneme, že řetězec y je výstupem pro řetězec x.

Poznámka: Překlad je obecně definován jako relace (viz. výše), ale v praxi je většinou touto relací zobrazení, neboť obvykle vyžadujeme, aby každý vstupní řetězec $x \in L_{in}$ byl přeložen na právě jeden výstupní řetězec $y \in L_{out}$.

2.1.11 Definice substituce

Nechť Σ, Ω jsou abecedy a τ je překlad z jazyka $L_{in} = \Sigma^*$ do jazyka $L_{out} = \Omega^*$. Pak τ nazveme *substitucí*, pokud pro každý řetězec $x \in \Sigma^*$ platí:

- Pokud $x = \varepsilon$, potom $\tau(x) = \{\varepsilon\}$.
- Pokud $x = a_1a_2\dots a_n$, kde $a_i \in \Sigma$ pro všechna $i = 1, 2, \dots, n$, potom
  $\tau(x) = \tau(a_1)\tau(a_2)\dots\tau(a_n)$.

Poznámka: Pro úplný popis dané substituce τ z jazyka Σ^* do jazyka Ω^* stačí pouze specifikovat $\tau(a)$ pro každé $a \in \Sigma$. Pro libovolný řetězec $x \in \Sigma^*$ je $\tau(x)$ již jednoznačně určeno podle definice substituce.

Příklad: Uvažujme vstupní abecedu $\Sigma = \{0, 1\}$, výstupní abecedu $\Omega = \{a, b, c, d\}$ a substituci τ z jazyka Σ^* do jazyka Ω^* definovanou jako:

- $\tau(0) = \{a, b\}$,
- $\tau(1) = \{c, d\}$.

Potom například pro řetězec 010 platí: $\tau(010) = \tau(0)\tau(1)\tau(0) = \{aca, bca, ada, bda, acb, bcb, adb, bdb\}$, tedy řetězce *aca*, *bca*, *ada*, *bda*, *acb*, *bcb*, *adb*, *bdb* jsou výstupem pro řetězec 010.

2.1.12 Definice homomorfismu

Nechť Σ, Ω jsou abecedy a τ je substituce z jazyka $L_{in} = \Sigma^*$ do jazyka $L_{out} = \Omega^*$. Pak τ nazveme *homomorfismem*, pokud τ je zobrazení.

Poznámka: Homomorfismus je tedy speciální případ substituce, pro který navíc platí, že ke každému řetězci ze vstupního jazyka existuje právě jeden řetězec, který je jeho výstupem.

Příklad: Uvažujme vstupní abecedu $\Sigma = \{0, 1\}$, výstupní abecedu $\Omega = \{a, b\}$ a homomofismus τ z jazyka Σ^* do jazyka Ω^* definovaný jako:

- $\tau(0) = aa$,
- $\tau(1) = bb$.

Potom například pro řetězec 010 platí: $\tau(010) = \tau(0)\tau(1)\tau(0) = aabbaa$, tedy řetězec *aabbaa* je výstupem řetězce 010.

2.2 Operace nad jazyky

2.2.1 Definice sjednocení dvou jazyků

Nechť L_1, L_2 jsou formální jazyky nad abecedou Σ. *Sjednocení* jazyků L_1 a L_2 (budeme označovat $L_1 \cup L_2$) je jazyk, který je definován následovně:

$$L_1 \cup L_2 = \{x: x \in L_1 \vee x \in L_2\}$$

2.2.2 Definice průniku dvou jazyků

Nechť L_1, L_2 jsou formální jazyky nad abecedou Σ. *Průnik* jazyků L_1 a L_2 (budeme označovat $L_1 \cap L_2$) je jazyk, který je definován následovně:

$$L_1 \cap L_2 = \{x: x \in L_1 \wedge x \in L_2\}$$

2.2.3 Definice konkatenace dvou jazyků

Nechť L_1, L_2 jsou formální jazyky nad abecedou Σ. *Konkatenace* jazyků L_1 a L_2 (budeme označovat $L_1 . L_2$) je jazyk, který je definován následovně:

$$L_1 . L_2 = \{xy: x \in L_1 \wedge y \in L_2\}$$

2.2.4 Definice rozdílu dvou jazyků

Nechť L_1, L_2 jsou formální jazyky nad abecedou Σ. *Rozdíl* jazyků L_1 a L_2 (budeme označovat $L_1 - L_2$) je jazyk, který je definován následovně:

$$L_1 - L_2 = \{x: x \in L_1 \wedge x \notin L_2\}$$

2.2.5 Definice doplňku jazyka

Nechť L je formální jazyk nad abecedou Σ. *Doplněk* jazyka L (budeme označovat $\overline{L}$) je jazyk, který je definován následovně:

$$\overline{L} = \Sigma^* - L$$

2.2.6 Definice mocniny jazyka

Nechť L je formální jazyk nad abecedou Σ. Pak *i-tá mocnina* jazyka L (budeme označovat L^i) je jazyk, který je definován následovně:

- $L^0 = \{\varepsilon\}$.
- $L^i = L.L^{i-1}$ pro všechna $i = 1, ..., n$.

2.2.7 Definice iterace jazyka

Nechť L je formální jazyk nad abecedou Σ. *Iterace* jazyka L (budeme označovat L^*) je jazyk, který je definován následovně:

$$L^* = \bigcup_{n=0}^{\infty} L^n$$

2.2.8 Definice pozitivní iterace jazyka

Nechť L je formální jazyk nad abecedou Σ. *Pozitivní iterace* jazyka L (budeme označovat L^+) je jazyk, který je definován následovně:

$$L^+ = \bigcup_{n=1}^{\infty} L^n$$

2.3 Základní typy gramatik

V této části jsou definovány různé typy gramatik. Jednotlivé typy gramatik se liší především ve tvaru pravidel, který daný typ gramatiky může obsahovat. Na závěr této kapitoly je uveden vztah, který platí z hlediska generativní síly jednotlivých typů gramatik.

2.3.1 Neomezená gramatika

Neomezená gramatika je gramatika, která obsahuje nejobecnější tvar pravidel. Množinu všech jazyků, které jsou generovány nějakou neomezenou gramatikou, nazveme ***třídou rekurzivně vyčíslitelných jazyků*** nebo také ***třídou jazyků typu 0***. Formální definice neomezené gramatiky je následující:

2.3.1.1 Definice neomezené gramatiky

Neomezená gramatika G je čtveřice $G = (N, T, P, S)$, kde:

- N je konečná množina nonterminálních symbolů,
- T je konečná množina terminálních symbolů, přičemž $N \cap T = \emptyset$,
- P je konečná množina pravidel tvaru $x \to y$, kde $x \in (N \cup T)^* N (N \cup T)^*$ a $y \in (N \cup T)^*$,
- S je počáteční nonterminální symbol.

2.3.1.2 Definice přímé derivace u neomezené gramatiky

Nechť $G = (N, T, P, S)$ je neomezená gramatika, nechť $u, v \in (N \cup T)^*$ a $p = x \to y \in P$ je pravidlo. Pak říkáme, že *uxv přímo derivuje uyv* podle pravidla p a zapisujeme $uxv \Rightarrow uyv$ $[p]$ nebo také zkráceně $uxv \Rightarrow uyv$.

2.3.1.3 Definice sekvence derivací u neomezené gramatiky

Nechť $G = (N, T, P, S)$ je neomezená gramatika.

- Nechť $u \in (N \cup T)^*$. Pak říkáme, že *u derivuje v 0-krocích u* a zapisujeme $u \Rightarrow^0 u$ $[\varepsilon]$ nebo také zkráceně $u \Rightarrow^0 u$.
- Nechť $u_0, u_1, \ldots, u_n \in (N \cup T)^*$, nechť pro všechna $i = 1, \ldots, n$ platí: $u_{i-1} \Rightarrow u_i$ $[p_i]$. Pak říkáme, že *u_0 derivuje v n-krocích u_n* a zapisujeme $u_0 \Rightarrow^n u_n$ $[p_1 p_2 \ldots p_n]$ nebo také zkráceně $u_0 \Rightarrow^n u_n$.
- Nechť $u \Rightarrow^n v$ $[\pi]$ pro nějaké $n \geq 1$; $u, v \in (N \cup T)^*$. Pak říkáme, že *u netriviálně derivuje v* a zapisujeme $u \Rightarrow^+ v$ $[\pi]$ nebo také zkráceně $u \Rightarrow^+ v$.
- Nechť $u \Rightarrow^n v$ $[\pi]$ pro nějaké $n \geq 0$; $u, v \in (N \cup T)^*$. Pak říkáme, že *u derivuje v* a zapisujeme $u \Rightarrow^* v$ $[\pi]$ nebo také zkráceně $u \Rightarrow^* v$.

2.3.1.4 Definice větné formy u neomezené gramatiky

Nechť $G = (N, T, P, S)$ je neomezená gramatika. Řekneme, že $u \in (N \cup T)^*$ je větná forma v neomezené gramatice G právě tehdy, když $S \Rightarrow^* u$.

2.3.1.5 Definice jazyka generovaného neomezenou gramatikou

Nechť $G = (N, T, P, S)$ je neomezená gramatika. Jazyk generovaný neomezenou gramatikou G (budeme jej označovat $L(G)$) je definován následovně:

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

2.3.2 Kontextová gramatika

Kontextová gramatika je speciální neomezená gramatika obsahující pravidla tvaru $x \to y$, pro které navíc platí, že $|x| \leq |y|$. Množinu všech jazyků, které jsou generovány nějakou kontextovou gramatikou, nazveme ***třídou kontextových jazyků*** nebo také ***třídou jazyků typu 1***. Formální definice je následující:

2.3.2.1 Definice kontextové gramatiky

Kontextová gramatika G je čtveřice $G = (N, T, P, S)$, kde:

- N je konečná množina nonterminálních symbolů,
- T je konečná množina terminálních symbolů, přičemž $N \cap T = \emptyset$,
- P je konečná množina pravidel tvaru $x \rightarrow y$, kde $x \in (N \cup T)^*N(N \cup T)^*$ a $y \in (N \cup T)^*$, přičemž $|x| \leq |y|$,
- S je počáteční nonterminální symbol.

2.3.2.2 Definice přímé derivace, sekvence derivací a jazyka generovaného kontextovou gramatikou

Definice přímé derivace, sekvence derivací a jazyka generovaného kontextovou gramatikou jsou identické s příslušnými definicemi u neomezené gramatiky.

2.3.3 Bezkontextová gramatika

Bezkontextová gramatika je speciální neomezená gramatika obsahující pravidla tvaru $x \rightarrow y$, pro která navíc platí, že řetězec x je pouze jeden nonterminální symbol. Množinu všech jazyků, které jsou generovány nějakou bezkontextovou gramatikou, nazveme ***třídou bezkontextových jazyků*** nebo také ***třídou jazyků typu 2***. Formální definice je následující:

2.3.3.1 Definice bezkontextové gramatiky

Bezkontextová gramatika G je čtveřice $G = (N, T, P, S)$, kde:

- N je konečná množina nonterminálních symbolů,
- T je konečná množina terminálních symbolů, přičemž $N \cap T = \emptyset$,
- P je konečná množina pravidel tvaru $A \rightarrow x$, kde $A \in N$ a $x \in (N \cup T)^*$,
- S je počáteční nonterminální symbol.

2.3.3.2 Definice přímé derivace u bezkontextové gramatiky

Nechť $G = (N, T, P, S)$ je bezkontextová gramatika, nechť $u, v \in (N \cup T)^*$ a $p = A \rightarrow x \in P$ je pravidlo. Pak říkáme, že *uAv přímo derivuje uxv* podle pravidla p a zapisujeme $uAv \Rightarrow uxv\ [p]$ nebo také zkráceně $uAv \Rightarrow uxv$.

2.3.3.3 Definice nejlevější derivace u bezkontextové gramatiky

Nechť $G = (N, T, P, S)$ je bezkontextová gramatika, $u \in T^*$ a $v \in (N \cup T)^*$ a $p = A \rightarrow x \in P$ je pravidlo. Pak říkáme, že *uAv přímo derivuje v nejlevější derivaci uxv* podle pravidla p, a zapisujeme $uAv \Rightarrow_{lm} uxv[p]$ nebo také zkráceně $uAv \Rightarrow_{lm} uxv$.

2.3.3.4 *Definice nejpravější derivace u bezkontextové gramatiky*

Nechť $G = (N, T, P, S)$ je bezkontextová gramatika, $u \in (N \cup T)^*$ a $v \in T^*$ a $p = A \to x \in P$ je pravidlo. Pak říkáme, že *uAv přímo derivuje v nejpravější derivaci uxv* podle pravidla p, a zapisujeme $uAv \Rightarrow_{rm} uxv[p]$ nebo také zkráceně $uAv \Rightarrow_{rm} uxv$.

2.3.3.5 *Definice sekvence derivací u bezkontextové gramatiky*

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

- Nechť $u \in (N \cup T)^*$. Pak říkáme, že *u derivuje v 0-krocích u* a zapisujeme $u \Rightarrow^0 u$ [ε] nebo také zkráceně $u \Rightarrow^0 u$.
- Nechť $u_0, u_1, \ldots, u_n \in (N \cup T)^*$, nechť pro všechna $i = 1, ..., n$ platí: $u_{i-1} \Rightarrow u_i$ [p_i]. Pak říkáme, že *u_0 derivuje v n-krocích u_n* a zapisujeme $u_0 \Rightarrow^n u_n$ [$p_1p_2\ldots p_n$] nebo také zkráceně $u_0 \Rightarrow^n u_n$.
- Nechť $u \Rightarrow^n v$ [π] pro nějaké $n \geq 1$; $u, v \in (N \cup T)^*$. Pak říkáme, že *u netriviálně derivuje v* a zapisujeme $u \Rightarrow^+ v$ [π] nebo také zkráceně $u \Rightarrow^+ v$.
- Nechť $u \Rightarrow^n v$ [π] pro nějaké $n \geq 0$; $u, v \in (N \cup T)^*$. Pak říkáme, že *u derivuje v* a zapisujeme $u \Rightarrow^* v$ [π] nebo také zkráceně $u \Rightarrow^* v$.

Poznámka: Výše uvedená definice rozšířila přímou derivaci $\Rightarrow$ na sekvence derivací $\Rightarrow^k$ pro libovolné $k \geq 0$, $\Rightarrow^+$ a $\Rightarrow^*$. Analogickým způsobem by se pomocí nejlevější přímé derivace $\Rightarrow_{lm}$ nadefinovala sekvence nejlevějších derivací $\Rightarrow_{lm}^k$ pro libovolné $k \geq 0$, $\Rightarrow_{lm}^+$ a $\Rightarrow_{lm}^*$ a pomocí nejpravější přímé derivace $\Rightarrow_{rm}$ by se nadefinovala sekvence nejpravějších derivací $\Rightarrow_{rm}^k$ pro libovolné $k \geq 0$, $\Rightarrow_{rm}^+$ a $\Rightarrow_{rm}^*$.

2.3.3.6 *Definice jazyka generovaného bezkontextovou gramatikou*

Nechť $G = (N, T, P, S)$ je bezkontextová gramatika. Jazyk generovaný bezkontextovou gramatikou G (budeme jej označovat $L(G)$) je definován následovně:

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

Poznámka:

Pomocí využití sekvence nejlevějších nebo nejpravějších derivací se může jazyk generovaný bezkontextovou gramatikou také definovat jako:

$$L(G) = \{w : w \in T^* \wedge S \Rightarrow_{lm}^* w\}$$

nebo

$$L(G) = \{w : w \in T^* \wedge S \Rightarrow_{rm}^* w\},$$

neboť platí vztah:

$$\{w : w \in T^* \wedge S \Rightarrow^* w\} = \{w : w \in T^* \wedge S \Rightarrow_{lm}^* w\} = \{w : w \in T^* \wedge S \Rightarrow_{rm}^* w\}$$

(viz. tvrzení 5.1.1.1 a tvrzení 5.1.1.2 v [12]).

2.3.4 *Pravá lineární gramatika*

Pravá lineární gramatika je speciální bezkontextová gramatika obsahující pravidla tvaru $A \to x$, pro která navíc platí, že řetězec x obsahuje buď samé terminální symboly a nebo terminální symboly zakončené pouze jedním nonterminálním symbolem. Množinu všech jazyků, které jsou generovány nějakou pravou lineární gramatikou, nazveme ***třídou regulárních jazyků*** nebo také ***třídou jazyků typu 3***. Formální definice je následující:

2.3.4.1 *Definice pravé lineární gramatiky*

Pravá lineární gramatika G je čtveřice $G = (N, T, P, S)$, kde:

- N je konečná množina nonterminálních symbolů,
- T je konečná množina terminálních symbolů, přičemž $N \cap T = \emptyset$,
- P je konečná množina pravidel tvaru $A \to xB$ nebo $A \to y$, kde $A, B \in N$ a $x, y \in T^*$,
- S je počáteční nonterminální symbol.

2.3.4.2 *Definice přímé derivace, sekvence derivací a jazyka generovaného pravou lineární gramatikou*

Definice přímé derivace, sekvence derivací a jazyka generovaného pravou lineární gramatikou jsou identické s příslušnými definicemi u bezkontextové gramatiky.

2.3.5 *Věta o Chomského hierarchii jazyků*

Označme symbolem L_{RE} třídu rekurzivně vyčíslitelných jazyků, symbolem L_{CS} třídu kontextových jazyků, symbolem L_{CF} třídu bezkontextových jazyků a symbolem L_{REG} třídu regulárních jazyků. Mezi těmito třídami jazyků platí vztah:

$$L_{REG} \subset L_{CF} \subset L_{CS} \subset L_{RE}$$

(viz. tvrzení 8.4.1 v [12])

2.4 Speciální typy bezkontextových gramatik

2.4.1 *Bezkontextová gramatika v Chomského normální formě*

2.4.1.1 *Definice bezkontextové gramatiky v Chomského normální formě*

Bezkontextová gramatika $G = (N, T, P, S)$ je v *Chomského normální formě* právě tehdy, když množina pravidel P obsahuje pouze pravidla tvaru $A \to BC$ nebo $A \to a$, kde $a \in T$ a $A, B, C \in N$.

Poznámka: Platí, že pro libovolnou bezkontextovou gramatiku G existuje bezkontextová gramatika v Chomského normální formě G_{CNF} taková, že $L(G_{CNF}) = L(G)$. (viz. algoritmus 5.1.4.1.1 v [12]).

2.4.2 Bezkontextová gramatika v Greibachové normální formě

2.4.2.1 Definice bezkontextové gramatiky v Greibachové normální formě

Bezkontextová gramatika $G = (N, T, P, S)$ je v *Greibachové normální formě* právě tehdy, když množina pravidel P obsahuje pouze pravidla tvaru $A \to ax$, kde $A \in N$, $a \in T$, $x \in N^*$.

Poznámka: Platí, že pro libovolnou bezkontextovou gramatiku G existuje bezkontextová gramatika v Greibachové normální formě G_{GNF} taková, že $L(G_{GNF}) = L(G)$. (viz. algoritmus 5.1.4.2.8 v [12])

2.4.3 Gramatiky LL

2.4.3.1 Definice množiny First

Nechť $G = (N, T, P, S)$ je bezkontextová gramatika. Pro libovolné $x \in (N \cup T)^*$ definujeme množinu $First(x)$ následovně:

$$First(x) = \{a: x \Rightarrow^* ay,\ a \in T,\ y \in (N \cup T)^*\}$$

2.4.3.2 Definice množiny Follow

Nechť $G = (N, T, P, S)$ je bezkontextová gramatika. Pro libovolné $A \in N$ definujeme množinu $Follow(A)$ následovně:

$$Follow(A) = \{a: S \Rightarrow^* xAay,\ a \in T,\ x, y \in (N \cup T)^*\} \cup \{\$: S \Rightarrow^* xA,\ x \in (N \cup T)^*\}$$

2.4.3.3 Množina Predict

Nechť $G = (N, T, P, S)$ je bezkontextová gramatika. Pro libovolné pravidlo $A \to x \in P$ definujeme množinu $Predict(A \to x)$ následovně:

Pokud $x \Rightarrow^* \varepsilon$, potom $Predict(A \to x) = First(x) \cup Follow(A)$,

jinak $Predict(A \to x) = First(x)$

2.4.3.4 Definice LL-gramatiky

Nechť $G = (N, T, P, S)$ je bezkontextová gramatika. Potom řekneme, že G je *LL*- gramatika, pokud pro každou dvojici pravidel tvaru $A \to x, A \to y \in P$, kde $A \in N$; $x, y \in (N \cup T)^*$ platí:

$$Predict(A \to x) \cap Predict(A \to y) = \emptyset$$

2.5 Další typy gramatik založené na bezkontextových gramatikách

V praxi se často používají pro popis jazyků bezkontextové gramatiky, protože mají jednoduchý tvar pravidel, pro které se poměrně snadno naimplementuje tzv. syntaktický analyzátor. Nevýhodou těchto gramatik je, že mají omezenou generativní sílu. (viz. věta o Chomského hierarchii jazyků). Proto vznikla řada modelů pro popis jazyků, které jsou založeny na bezkontextových gramatikách, ale většinou obsahují navíc další komponentu, která jistým způsobem řídí výběr pravidel, které budou v průběhu derivace použity. Tímto způsobem lze ve většině případů zvýšit generativní sílu daného modelu.

2.5.1 Maticová gramatika

Maticová gramatika je rozšířená bezkontextová gramatika G o speciální komponentu M, která obsahuje konečně mnoho sekvencí (tzv. matic) pravidel z bezkontextové gramatiky G. Při derivačním kroku v maticové gramatice je vždy vybrána jedna tato sekvence (uvažujme ji například ve tvaru $p_1p_2\ldots p_n$), přičemž při výběru této sekvence musí být hned po sobě aplikováno právě n přímých derivací postupně podle pravidel $p_1, p_2, \ldots, p_n$ včetně daného pořadí. Formální definice je následující:

2.5.1.1 Definice maticové gramatiky

Maticová gramatika H je dvojice $H = (G, M)$, kde:

- $G = (N, T, P, S)$ je bezkontextová gramatika,
- M je konečný jazyk nad abecedou P, tedy $M \subseteq P^*$.

2.5.1.2 Definice přímé derivace u maticové gramatiky

Nechť $H = (G, M)$ je maticová gramatika, nechť v bezkontextové gramatice G existuje sekvence přímých derivací ve tvaru:

$$u_0 \Rightarrow u_1\ [p_1] \Rightarrow u_2\ [p_2] \Rightarrow \ldots \Rightarrow u_n\ [p_n]$$

a nechť platí, že $m = p_1p_2\ldots p_n \in M$. Pak říkáme, že v maticové gramatice H u_0 *přímo derivuje* u_n podle matice m a zapisujeme $u_0 \Rightarrow u_n\ [m]$ nebo také zkráceně $u_0 \Rightarrow u_n$.

2.5.1.3 Definice sekvence derivací u maticové gramatiky

Nechť $H = (G, M)$ je maticová gramatika.

- Nechť $u \in (N \cup T)^*$. Pak říkáme, že *u derivuje v 0-krocích u* a zapisujeme $u \Rightarrow^0 u\ [\varepsilon]$ nebo také zkráceně $u \Rightarrow^0 u$.
- Nechť $u_0, u_1, \ldots, u_n \in (N \cup T)^*$, nechť pro všechna $i = 1, \ldots, n$ platí: $u_{i-1} \Rightarrow u_i\ [p_i]$. Pak říkáme, že *$u_0$ derivuje v n-krocích u_n* a zapisujeme $u_0 \Rightarrow^n u_n\ [p_1p_2\ldots p_n]$ nebo také zkráceně $u_0 \Rightarrow^n u_n$.
- Nechť $u \Rightarrow^n v\ [\pi]$ pro nějaké $n \geq 1$; $u, v \in (N \cup T)^*$. Pak říkáme, že *u netriviálně derivuje v* a zapisujeme $u \Rightarrow^+ v\ [\pi]$ nebo také zkráceně $u \Rightarrow^+ v$.
- Nechť $u \Rightarrow^n v\ [\pi]$ pro nějaké $n \geq 0$; $u, v \in (N \cup T)^*$. Pak říkáme, že *u derivuje v* a zapisujeme $u \Rightarrow^* v\ [\pi]$ nebo také zkráceně $u \Rightarrow^* v$.

2.5.1.4 Definice jazyka generovaného maticovou gramatikou

Nechť $H = (G, M)$ je maticová gramatika, přičemž $G = (N, T, P, S)$. Jazyk generovaný maticovou gramatikou H (budeme jej označovat $L(H)$) je definován následovně:

$$L(H) = \{w : w \in T^* \wedge S \Rightarrow^* w \text{ v maticové gramatice } H\}.$$

2.5.1.5 *Příklad maticové gramatiky*

Dvojice $H = (G, M)$, kde:

- $G = (\{S, A, B\}, \{a, b, c\}, \{1{:}\ S \rightarrow AB,\ 2{:}\ A \rightarrow aA,\ 3{:}\ B \rightarrow bBc,\ 4{:}\ A \rightarrow a,\ 5{:}\ B \rightarrow bc\}, S)$
- $M = \{1, 23, 45\}$

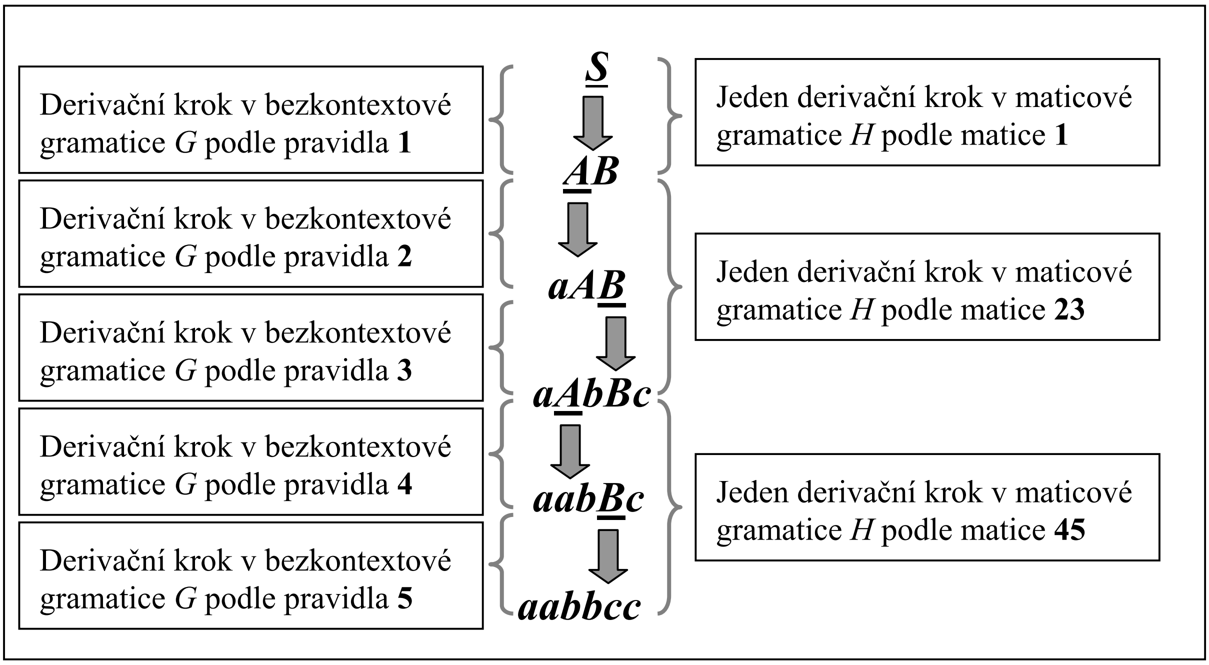
je maticová gramatika.

V této maticové gramatice existují například následující sekvence derivací:

- $S \Rightarrow AB$ [1] $\Rightarrow abc$ [45]
- $S \Rightarrow AB$ [1] $\Rightarrow aAbBc$ [23] $\Rightarrow aabbcc$ [45]
- $S \Rightarrow AB$ [1] $\Rightarrow aAbBc$ [23] $\Rightarrow aaAbbBcc$ [23] $\Rightarrow aaabbbccc$ [45]
- ...

Jazyk generovaný touto maticovou gramatikou je $L(H) = \{a^n b^n c^n : n \geq 1\}$.

Podrobněji provedení sekvence derivací $S \Rightarrow^* aabbcc$ je ilustrováno na obr. 2.1.



(obr. 2.1: ukázka derivace $S \Rightarrow^* aabbcc$ v maticové gramatice)

2.6 Základní typy gramatických systémů

Gramatické systémy jsou založeny na n-tici gramatik, které spolu jednoduchým způsobem komunikují. Rozlišujeme CD (*cooperating distributed*) gramatické systémy a PC (*parallel communicating*) gramatické systémy.

CD gramatické systémy pracují sekvenčně. Tyto systémy se skládají z komponent, které představují gramatiky. Každý derivační krok je proveden pouze jednou gramatikou. Řízení takového systému spočívá ve výběru komponenty, která následně provede derivační krok. Například pomocí jedné gramatiky se provede k-kroků a po té v derivaci pokračuje jiná gramatika, která opět provede k-kroků atd. Rovněž je stanovena podmínka ukončující činnost celého systému. Takové ukončení může být provedeno v okamžiku, kdy žádná gramatika nemůže vykonat další derivační krok.

PC gramatické systémy pracují paralelně. Každá komponenta, kterou je opět gramatika, má svoji vlastní větnou formu, nad kterou provádí derivační kroky. Řízení komponent je realizováno pomocí komunikačních symbolů. Na základě těchto symbolů gramatika určí, kde dojde k vložení větné formy z jiné gramatiky. Jazyk, který generuje první komponenta systému, je výsledný jazyk celého systému.

Formální definice obou druhů gramatických systému lze najít například v [3].

3 Kanonické multigenerativní gramatické systémy

V této kapitole jsou zavedeny zcela nové typy gramatických systémů. Nyní se zaměříme na tzv. kanonické gramatické systémy, které generují řetězce pomocí nejlevější derivace, což znamená, že v každém kroku je v dané větné formě přepsán vždy nonterminální symbol, který je nejvíce vlevo.

Obecně n-generativní gramatický systém se skládá z n bezkontextových gramatik. Jeden derivační krok u n-generativního gramatického systému je proveden tak, že se paralelně u všech těchto n-gramatik aplikuje jisté pravidlo na aktuální větnou formu. Těchto n derivací je kontrolováno n-ticí nonterminálních symbolů nebo n-ticí pravidel. Pod touto kontrolou gramatický systém generuje n-tici řetězců, se kterými jsou na závěr provedeny základní operace. Pomocí výsledku operace je definován generovaný jazyk. Mezi tyto operace patří především sjednocení, konkatenace a výběr řetězce, který generuje první komponenta.

3.1 Kanonický *n*-generativní nonterminálově synchronizovaný gramatický systém

Kanonický n-generativní nonterminálově synchronizovaný gramatický systém se skládá z n bezkontextových gramatik a ze speciální kontrolní komponenty Q. Komponenta Q je množina, jejíž prvky jsou n-tice nonterminálních symbolů. Derivační krok se skládá ze dvou části. V první části je provedena kontrola větných forem vygenerovaných jednotlivými gramatikami následujícím způsobem: Z jednotlivých větných forem odpovídající jednotlivým gramatikám je z nejlevějších nonterminálních symbolů vytvořena n-tice. Pokud tato n-tice je obsažena v kontrolní komponentě Q, pak je vše v pořádku a může se přejít na druhou část derivačního kroku. Pokud ne, je derivace zablokována. V druhé části derivačního kroku je paralelně v každé gramatice aplikováno právě jedno pravidlo v nejlevější derivaci na aktuální větnou formu.

3.1.1 Definice kanonického n-generativního nonterminálově synchronizovaného gramatického systému

Kanonický n-generativní nonterminálově synchronizovaný GS (n-KGN) je n+1-tice

$$\Gamma = (G_1, G_2, \ldots, G_n, Q), \text{ kde:}$$

- $G_i = (N_i, T_i, P_i, S_i)$ je bezkontextová gramatika pro všechna $i = 1, \ldots, n$.
- Q je konečná množina *kontrolních n-tic nonterminálů* tvaru $(A_1, A_2, \ldots, A_n)$, kde $A_i \in N_i$ pro všechna $i = 1, \ldots, n$.

3.1.2 Definice multiformy

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-KGN. Potom *multiforma* je n-tice $\chi = (x_1, x_2, \ldots, x_n)$, kde $x_i \in (T_i \cup N_i)^*$ pro všechna $i = 1, \ldots, n$.

3.1.3 Definice přímého derivačního kroku v n-KGN

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-KGN, nechť $\chi = (u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n)$, $\bar{\chi} = (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$, jsou dvě multiformy, kde $A_i \in N_i$, $u_i \in T_i^*$, v_i, $x_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \ldots, n$. Dále nechť $A_i \rightarrow x_i \in P_i$ pro všechna $i = 1, \ldots, n$ a $(A_1, A_2, \ldots, A_n) \in Q$. Pak říkáme, že χ *přímo derivuje* $\bar{\chi}$ a zapisujeme $\chi \Rightarrow \bar{\chi}$.

3.1.4 Definice sekvence derivačních kroků v n-KGN

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-KGN

- Nechť χ je multiforma. Pak říkáme, že χ *derivuje v 0-krocích* χ a zapisujeme $\chi \Rightarrow^0 \chi$.
- Nechť $\chi_0, \chi_1, \ldots, \chi_k$ jsou multiformy, u kterých pro všechna $i = 1, \ldots, n$ platí: $\chi_{i-1} \Rightarrow \chi_i$. Pak říkáme, že χ_0 *derivuje v k-krocích* χ_n a zapisujeme $\chi_0 \Rightarrow^k \chi_n$.
- Nechť $\chi \Rightarrow^k \bar{\chi}$ pro nějaké $k \geq 1$, kde χ, $\bar{\chi}$ jsou multiformy. Pak říkáme, že χ *netriviálně derivuje* $\bar{\chi}$ a zapisujeme $\chi \Rightarrow^+ \bar{\chi}$.
- Nechť $\chi \Rightarrow^k \bar{\chi}$ pro nějaké $k \geq 0$, kde χ, $\bar{\chi}$ jsou multiformy. Pak říkáme, že χ *derivuje* $\bar{\chi}$ a zapisujeme $\chi \Rightarrow^* \bar{\chi}$.

3.1.5 Definice n-jazyka generovaného n-KGN

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-KGN. Potom *n-jazyk* generovaný Γ (budeme značit n-$L(\Gamma)$) je definován:

$n\text{-}L(\Gamma) = \{(w_1, w_2, \ldots, w_n)\colon (S_1, S_2, \ldots, S_n) \Rightarrow^* (w_1, w_2, \ldots, w_n), w_i \in T_i^* \text{ pro všechna } i = 1, \ldots, n\}$

3.1.6 Definice jazyka v módu sjednocení

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-KGN. Jazyk generovaný Γ v módu *sjednocení* (budeme značit $L_{union}(\Gamma)$) je definován:

$$L_{union}(\Gamma) = \bigcup_{i=1}^{n} \{w_i\colon (w_1, w_2, \ldots, w_n) \in n\text{-}L(\Gamma)\}$$

3.1.7 Definice jazyka v módu konkatenace

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-KGN. Jazyk generovaný Γ v módu *konkatenace* (budeme značit $L_{conc}(\Gamma)$) je definován:

$$L_{conc}(\Gamma) = \{w_1w_2\ldots w_n\colon (w_1, w_2, \ldots, w_n) \in n\text{-}L(\Gamma)\}$$

3.1.8 Definice jazyka v módu první komponenty

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-KGN. Jazyk generovaný Γ v módu *první komponenty* (budeme značit $L_{first}(\Gamma)$) je definován:

$$L_{first}(\Gamma) = \{w_1\colon (w_1, w_2, \ldots, w_n) \in n\text{-}L(\Gamma)\}$$

3.1.9 *Příklad n-KGN*

Trojice $\Gamma = (G_1, G_2, Q)$, kde:

- $G_1 = (\{S_1, A_1\}, \{a, b, c\}, \{\mathbf{1}: S_1 \rightarrow aS_1, \mathbf{2}: S_1 \rightarrow aA_1, \mathbf{3}: A_1 \rightarrow bA_1c, \mathbf{4}: A_1 \rightarrow bc\}, S_1)$
- $G_2 = (\{S_2, A_2\}, \{d\}, \{\mathbf{1}: S_2 \rightarrow S_2A_2, \mathbf{2}: S_2 \rightarrow A_2, \mathbf{3}: A_2 \rightarrow d\}, S_2)$
- $Q = \{(S_1, S_2), (A_1, A_2)\}$

je kanonický 2-generativní nonterminálově synchronizovaný GS (2-KGN).

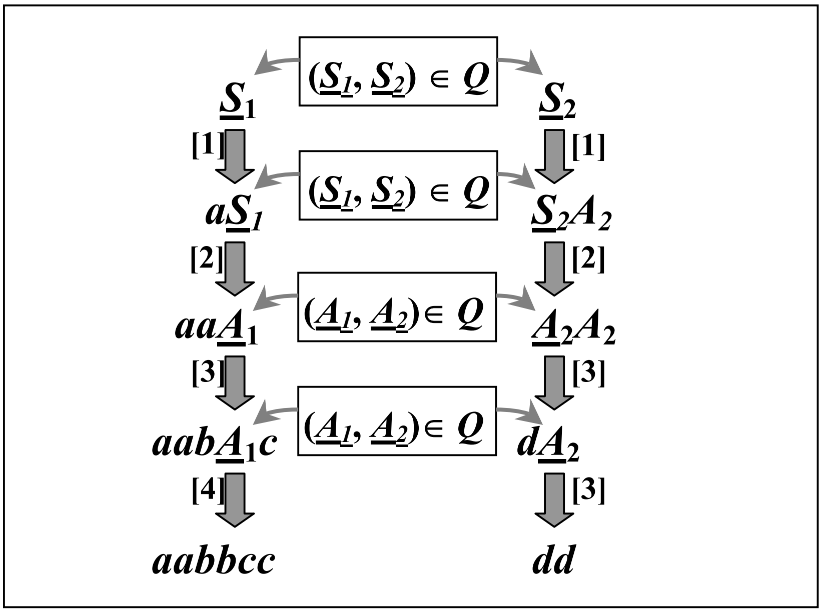
V tomto 2-KGN existují například následující sekvence derivačních kroků:

- $(\underline{S_1}, \underline{S_2}) \Rightarrow (a\underline{A_1}, \underline{A_2}) \Rightarrow (abc, d)$
- $(\underline{S_1}, \underline{S_2}) \Rightarrow (a\underline{S_1}, \underline{S_2}A_2) \Rightarrow (aa\underline{A_1}, \underline{A_2}A_2) \Rightarrow (aab\underline{A_1}c, d\underline{A_2}) \Rightarrow (aabbcc, dd)$
- …

Podrobněji provedení sekvence derivačních kroků $(S_1, S_2) \Rightarrow^* (aabbcc, dd)$ je ilustrováno na obr. 3.1.

Poznamenejme, že pro daný 2-KGN platí:

- $2\text{-}L(\Gamma) = \{(a^nb^nc^n, d^n): n \geq 1\}$,
- $L_{union}(\Gamma) = \{a^nb^nc^n: n \geq 1\} \cup \{d^n: n \geq 1\}$,
- $L_{conc}(\Gamma) = \{a^nb^nc^nd^n: n \geq 1\}$,
- $L_{first}(\Gamma) = \{a^nb^nc^n: n \geq 1\}$.



(obr. 3.1: ukázka derivace $(S_1, S_2) \Rightarrow^* (aabbcc, dd)$ v 2-KGN)

3.2 Kanonický *n*-generativní pravidlově synchronizovaný gramatický systém

Kanonický n-generativní pravidlově synchronizovaný gramatický systém se skládá z n bezkontextových gramatik a ze speciální kontrolní komponenty Q. Komponenta Q je množina, jejíž prvky jsou n-tice pravidel. Derivační krok je proveden následujícím způsobem: Z komponenty Q je vybrána jedna n-tice pravidel, přičemž první pravidlo z této n-tice je aplikováno na větnou formu vygenerovanou první gramatikou, druhé pravidlo z této n-tice je aplikováno na větnou formu vygenerovanou druhou gramatikou atd. Všechna tato pravidla jsou aplikována na nejlevější nonterminální symbol v jednotlivých větných formách.

3.2.1 Definice kanonického n-generativního pravidlově synchronizovaného gramatického systému

Kanonický n-generativní pravidlově synchronizovaný GS (n-KGP) je n+1-tice

$$\Gamma = (G_1, G_2, \ldots, G_n, Q), \text{ kde}$$

- $G_i = (N_i, T_i, P_i, S_i)$ je bezkontextová gramatika pro všechna $i = 1, \ldots, n$.
- Q je konečná množina *kontrolních n-tic pravidel* tvaru $(p_1, p_2, \ldots, p_n)$, kde $p_i \in P_i$ pro všechna $i = 1, \ldots, n$.

3.2.2 Definice multiformy

Multiforma pro n-KGP je definována shodně s multiformou pro n-KGN.

3.2.3 Definice přímého derivační kroku v n-KGP

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-KGP, nechť $\chi = (u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n)$, $\bar{\chi} = (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$, jsou dvě multiformy, kde $A_i \in N_i$, $u_i \in T_i^*$, $v_i, x_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \ldots, n$. Dále nechť $p_i = A_i \rightarrow x_i \in P_i$ pro všechna $i = 1, \ldots, n$ a $(p_1, p_2, \ldots, p_n) \in Q$. Pak říkáme, že χ *přímo derivuje* $\bar{\chi}$ a zapisujeme $\chi \Rightarrow \bar{\chi}$.

3.2.4 Definice sekvence derivačních kroků v n-KGP

Sekvence derivačních kroků pro n-KGP je definována analogicky jako sekvence derivačních kroků pro n-KGN.

3.2.5 Definice n-jazyka a jazyků v různých módech v n-KGP

n-jazyk generovaný n-KGP je definován analogicky jako jazyk generovaný n-KGN. Stejně tak jazyky v různých módech pro n-KGP jsou definovány analogicky jako jazyky generované pomocí n-KGN.

3.2.6 *Příklad n-KGP*

Trojice $\Gamma = (G_1, G_2, Q)$, kde:

- $G_1 = (\{S_1, A_1\}, \{a, b, c\}, \{\mathbf{1:}\ S_1 \rightarrow aS_1, \mathbf{2:}\ S_1 \rightarrow aA_1, \mathbf{3:}\ A_1 \rightarrow bA_1c, \mathbf{4:}\ A_1 \rightarrow bc\}, S_1)$
- $G_2 = (\{S_2\}, \{d\}, \{\mathbf{1:}\ S_2 \rightarrow S_2S_2, \mathbf{2:}\ S_2 \rightarrow S_2, \mathbf{3:}\ S_2 \rightarrow d\}, S_2)$
- $Q = \{(\mathbf{1}, \mathbf{1}), (\mathbf{2}, \mathbf{2}), (\mathbf{3}, \mathbf{3}), (\mathbf{4}, \mathbf{3})\}$

je kanonický 2-generativní pravidlově synchronizovaný GS (2-KGP).

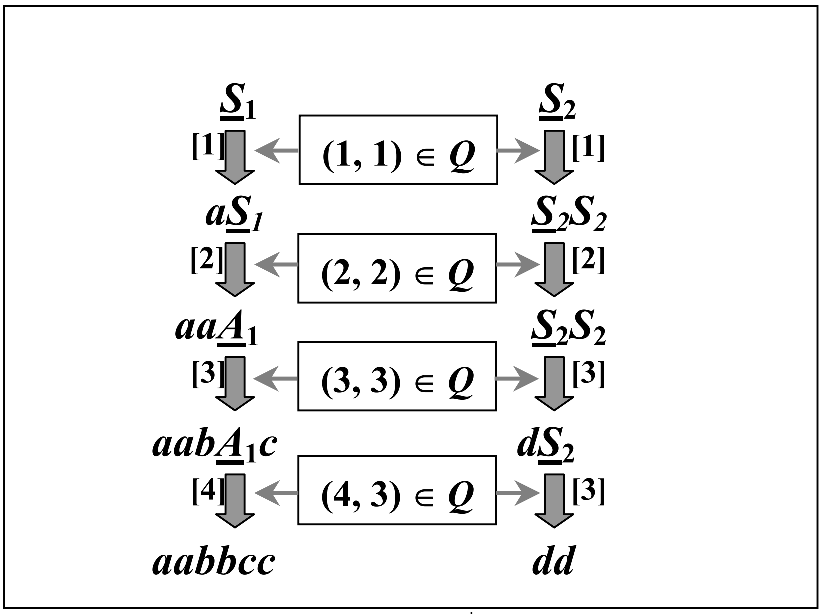
V tomto 2-KGP existují například následující sekvence derivačních kroků:

- $(\underline{S_1}, \underline{S_2}) \Rightarrow (a\underline{A_1}, \underline{S_2}) \Rightarrow (abc, d)$
- $(\underline{S_1}, \underline{S_2}) \Rightarrow (a\underline{S_1}, \underline{S_2}S_2) \Rightarrow (aa\underline{A_1}, \underline{S_2}S_2) \Rightarrow (aab\underline{A_1}c, d\underline{S_2}) \Rightarrow (aabbcc, dd)$
- …

Podrobněji provedení sekvence derivačních kroků $(S_1, S_2) \Rightarrow^* (aabbcc, dd)$ je ilustrováno na obr. 3.2.

Poznamenejme, že pro daný 2-KGP platí:

- $2\text{-}L(\Gamma) = \{(a^nb^nc^n, d^n): n \geq 1\}$,
- $L_{union}(\Gamma) = \{a^nb^nc^n: n \geq 1\} \cup \{d^n: n \geq 1\}$,
- $L_{conc}(\Gamma) = \{a^nb^nc^nd^n: n \geq 1\}$,
- $L_{first}(\Gamma) = \{a^nb^nc^n: n \geq 1\}$.



(obr. 3.2: ukázka derivace $(S_1, S_2) \Rightarrow^* (aabbcc, dd)$ v 2-KGP)

3.3 Vztah mezi *n*-KGN a *n*-KGP a jejich generativní síla

3.3.1 Tvrzení

Nechť Γ je libovolný n-KGN a dále nechť $\overline{\Gamma}$ je libovolný n-KGP pro které platí: $n\text{-}L(\Gamma) = n\text{-}L(\overline{\Gamma})$. Potom $L_X(\Gamma) = L_X(\overline{\Gamma})$ pro každé $X \in \{union, conc, first\}$.

Důkaz:

I. Dokážeme, že $L_{union}(\Gamma) = L_{union}(\overline{\Gamma})$:

$$L_{union}(\Gamma) = \bigcup_{i=1}^{n} \{w_i\text{: } (w_1, w_2, \dots, w_n) \in n\text{-}L(\Gamma)\} = \bigcup_{i=1}^{n} \{w_i\text{: } (w_1, w_2, \dots, w_n) \in n\text{-}L(\overline{\Gamma})\} = L_{union}(\overline{\Gamma}).$$

II. Dokážeme, že $L_{conc}(\Gamma) = L_{conc}(\overline{\Gamma})$:

$$L_{conc}(\Gamma) = \{w_1w_2\dots w_n\text{: } (w_1, w_2, \dots, w_n) \in n\text{-}L(\Gamma)\} = \{w_1w_2\dots w_n\text{: } (w_1, w_2, \dots, w_n) \in n\text{-}L(\overline{\Gamma})\} = L_{conc}(\overline{\Gamma}).$$

III. Dokážeme, že $L_{first}(\Gamma) = L_{first}(\overline{\Gamma})$:

$$L_{first}(\Gamma) = \{w_1\text{: } (w_1, w_2, \dots, w_n) \in n\text{-}L(\Gamma)\} = \{w_1\text{: } (w_1, w_2, \dots, w_n) \in n\text{-}L(\overline{\Gamma})\} = L_{first}(\overline{\Gamma}).$$

3.3.2 Převod n-KGN na ekvivalentní n-KGP

Následující algoritmus popíše, jak lze převést libovolný n-KGN na ekvivalentní n-KGP, tedy na takový, jehož n-jazyk je shodný s n-jazykem původního n-KGN. Neformální popis myšlenky algoritmu je následující: V původním n-KGN byly kontrolovány n-tice daných nonterminálních symbolů, nyní musí být kontrolovány n-tice pravidel. Nový n-KGP tedy bude mít všech n gramatik stejných jako původní n-KGN, pouze komponenta Q bude odlišná. Pokud komponenta Q původního n-KGN obsahovala například prvek $(A_1, A_2, \dots, A_n)$, potom do komponenty $\overline{Q}$ nového n-KGP budou přidány všechny ty n-tice, které mohou být vytvořeny z pravidel mající na levé straně právě tyto nonterminální symboly. Všechny tyto prvky budou tedy ve tvaru $(A_1 \to x_1, A_2 \to x_2, \dots, A_n \to x_n)$. Tím bude zaručeno, že daná n-tice pravidel $(A_1 \to x_1, A_2 \to x_2, \dots, A_n \to x_n) \in \overline{Q}$ může být vybrána právě tehdy, když n-tice $(A_1, A_2, \dots, A_n) \in Q$. Správnost této myšlenky je pak formálně dokázána ve tvrzení 3.3.2.3.

3.3.2.1 Algoritmus: Převod n-KGN na ekvivalentní n-KGP

- ***Vstup:*** n-KGN $\Gamma = (G_1, G_2, \dots, G_n, Q)$
- ***Výstup:*** n-KGP $\overline{\Gamma} = (G_1, G_2, \dots, G_n, \overline{Q})$; $n\text{-}L(\Gamma) = n\text{-}L(\overline{\Gamma})$
- ***Metoda:***
  - Nechť $G_i = (N_i, T_i, P_i, S_i)$ pro všechna $i = 1, \dots, n$, potom:
  - $\overline{Q} := \{(A_1 \to x_1, A_2 \to x_2, \dots, A_n \to x_n)\text{: } A_i \to x_i \in P_i$ pro všechna $i = 1, \dots, n,$
    $(A_1, A_2, \dots, A_n) \in Q\ \}$

3.3.2.2 *Příklad: Převod n-KGN na ekvivalentní n-KGP*

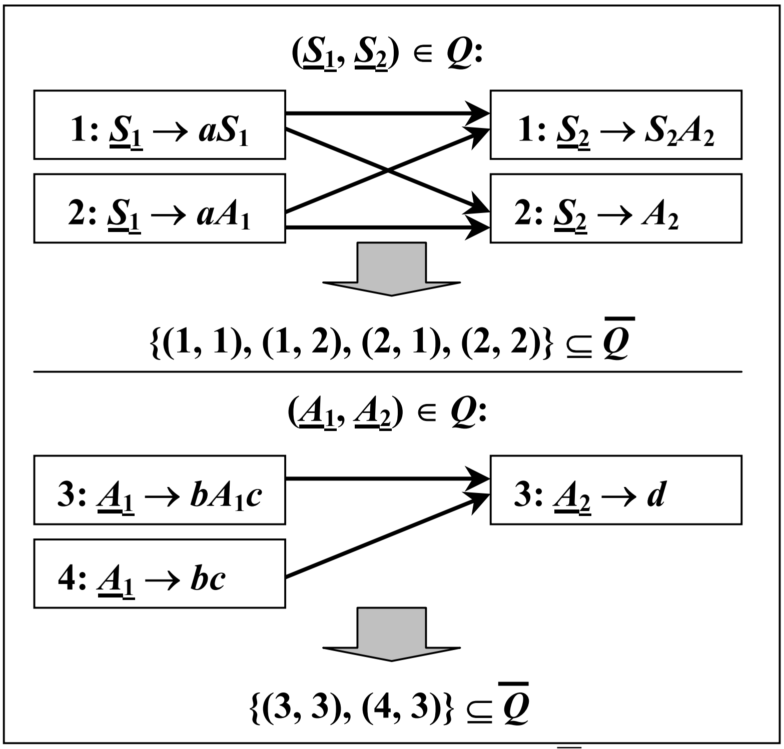
Uvažujme následující 2-KGN $\Gamma = (G_1, G_2, Q)$, kde:

- $G_1 = (\{S_1, A_1\}, \{a, b, c\}, \{\mathbf{1}: S_1 \to aS_1, \mathbf{2}: S_1 \to aA_1, \mathbf{3}: A_1 \to bA_1c, \mathbf{4}: A_1 \to bc\}, S_1)$
- $G_2 = (\{S_2, A_2\}, \{d\}, \{\mathbf{1}: S_2 \to S_2A_2, \mathbf{2}: S_2 \to A_2, \mathbf{3}: A_2 \to d\}, S_2)$
- $Q = \{(S_1, S_2), (A_1, A_2)\}$

Ekvivalentní 2-KGP sestavený podle předchozího algoritmu bude mít strukturu $\overline{\Gamma} = (G_1, G_2, \overline{Q})$, kde:

- $G_1 = (\{S_1, A_1\}, \{a, b, c\}, \{\mathbf{1}: S_1 \to aS_1, \mathbf{2}: S_1 \to aA_1, \mathbf{3}: A_1 \to bA_1c, \mathbf{4}: A_1 \to bc\}, S_1)$
- $G_2 = (\{S_2, A_2\}, \{d\}, \{\mathbf{1}: S_2 \to S_2A_2, \mathbf{2}: S_2 \to A_2, \mathbf{3}: A_2 \to d\}, S_2)$
- $\overline{Q} = \{(\mathbf{1}, \mathbf{1}), (\mathbf{1}, \mathbf{2}), (\mathbf{2}, \mathbf{1}), (\mathbf{2}, \mathbf{2}), (\mathbf{3}, \mathbf{3}), (\mathbf{4}, \mathbf{3})\}$

Vytvoření komponenty $\overline{Q}$ z komponenty Q ilustruje obr. 3.3.



(obr. 3.3: ilustrace vytvoření komponenty $\overline{Q}$ pro příklad 3.3.2.2)

3.3.2.3 *Tvrzení*

Nechť $\Gamma = (G_1, G_2, \dots, G_n, Q)$ je libovolný n-KGN. S n-KGN Γ na vstupu algoritmus 3.3.2.1 zastaví a správně vytvoří n-KGP $\overline{\Gamma} = (G_1, G_2, \dots, G_n, \overline{Q})$, pro který bude platit $n\text{-}L(\Gamma) = n\text{-}L(\overline{\Gamma})$ a $L_X(\Gamma) = L_X(\overline{\Gamma})$ pro každé $X \in \{union, conc, first\}$.

Důkaz:

Tvrzení A: Nechť existuje sekvence derivačních kroků $(S_1, S_2, \dots, S_n) \Rightarrow^m (y_1, y_2, \dots, y_n)$ v n-KGN Γ, kde $m \geq 0$, $y_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \dots, n$. Potom také existuje sekvence derivačních kroků $(S_1, S_2, \dots, S_n) \Rightarrow^m (y_1, y_2, \dots, y_n)$ v n-KGP $\overline{\Gamma}$.

Důkaz: Tvrzení je dokázáno matematickou indukcí

Základní krok: Nechť $m = 0$. Potom $(S_1, S_2, \dots, S_n) \Rightarrow^0 (S_1, S_2, \dots, S_n)$ v n-KGN Γ. Poznamenejme, že také $(S_1, S_2, \dots, S_n) \Rightarrow^0 (S_1, S_2, \dots, S_n)$ v n-KGP $\overline{\Gamma}$ a tvrzení tedy platí.

Indukční hypotéza: Předpokládejme, že tvrzení A platí pro všechny m-kroké derivace, kde $m = 0,\dots, k$ pro nějaké $k \geq 0$.

Indukční krok: Uvažujme sekvenci derivačních kroků $(S_1, S_2, \dots, S_n) \Rightarrow^{k+1} (y_1, y_2, \dots, y_n)$ v n-KGN Γ. Potom existuje multiforma $(u_1A_1v_1, u_2A_2v_2, \dots, u_nA_nv_n)$, kde $u_i \in T_i^*$, $A_i \in N_i$, $v_i \in (N_i \cup T_i)^*$, pro kterou platí $(S_1, S_2, \dots, S_n) \Rightarrow^k (u_1A_1v_1, u_2A_2v_2, \dots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \dots, u_nx_nv_n)$ v n-KGN Γ, kde $u_ix_iv_i = y_i$ pro všechna $i = 1, \dots, n$.

I. Ze sekvence derivačních kroků $(S_1, S_2, \dots, S_n) \Rightarrow^k (u_1A_1v_1, u_2A_2v_2, \dots, u_nA_nv_n)$ v n-KGN Γ plyne, že existuje sekvence derivačních kroků $(S_1, S_2, \dots, S_n) \Rightarrow^k (u_1A_1v_1, u_2A_2v_2, \dots, u_nA_nv_n)$ v n-KGP $\overline{\Gamma}$ podle indukční hypotézy.

II. Nechť existuje derivační krok $(u_1A_1v_1, u_2A_2v_2, \dots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \dots, u_nx_nv_n)$ v n-KGN Γ. Potom zřejmě pro n-KGN Γ platí $(A_1, A_2, \dots, A_n) \in Q$ a dále $A_i \rightarrow x_i \in P_i$ pro všechna $i = 1, \dots, n$. Z algoritmu 3.3.2.1 potom plyne, že $(A_1 \rightarrow x_1, A_2 \rightarrow x_2, \dots, A_n \rightarrow x_n) \in \overline{Q}$ v n-KGP $\overline{\Gamma}$, tedy existuje i derivační krok $(u_1A_1v_1, u_2A_2v_2, \dots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \dots, u_nx_nv_n)$ v n-KGP $\overline{\Gamma}$.

Z částí I. a II. plyne, že $(S_1, S_2, \dots, S_n) \Rightarrow^{k+1} (u_1x_1v_1, u_2x_2v_2, \dots, u_nx_nv_n)$ v n-KGP $\overline{\Gamma}$.

Tvrzení B: Nechť existuje sekvence derivačních kroků $(S_1, S_2, \dots, S_n) \Rightarrow^m (y_1, y_2, \dots, y_n)$ v n-KGP $\overline{\Gamma}$, kde $m \geq 0$, $y_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \dots, n$. Potom existuje sekvence derivačních kroků $(S_1, S_2, \dots, S_n) \Rightarrow^m (y_1, y_2, \dots, y_n)$ v n-KGN Γ.

Důkaz: Tvrzení je dokázáno matematickou indukcí

Základní krok: Nechť $m = 0$. Potom $(S_1, S_2, \dots, S_n) \Rightarrow^0 (S_1, S_2, \dots, S_n)$ v n-KGP $\overline{\Gamma}$. Poznamenejme, že také $(S_1, S_2, \dots, S_n) \Rightarrow^0 (S_1, S_2, \dots, S_n)$ v n-KGN Γ a tvrzení tedy platí.

Indukční hypotéza: Předpokládejme, že tvrzení B platí pro všechny m-kroké derivace, kde $m = 0,\dots, k$ pro nějaké $k \geq 0$.

Indukční krok: Uvažujme sekvenci derivačních kroků $(S_1, S_2, \dots, S_n) \Rightarrow^{k+1} (y_1, y_2, \dots, y_n)$ v n-KGP $\overline{\Gamma}$. Potom zřejmě existuje multiforma $(u_1A_1v_1, u_2A_2v_2, \dots, u_nA_nv_n)$, kde $u_i \in T_i^*$, $A_i \in N_i$, $v_i \in (N_i \cup T_i)^*$, pro kterou platí $(S_1, S_2, \dots, S_n) \Rightarrow^k (u_1A_1v_1, u_2A_2v_2, \dots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \dots, u_nx_nv_n)$ v n-KGP $\overline{\Gamma}$, kde $u_ix_iv_i = y_i$ pro všechna $i = 1, \dots, n$.

I. Ze sekvence derivačních kroků $(S_1, S_2, \dots, S_n) \Rightarrow^k (u_1A_1v_1, u_2A_2v_2, \dots, u_nA_nv_n)$ v n-KGP $\overline{\Gamma}$ plyne, že existuje sekvence derivačních kroků $(S_1, S_2, \dots, S_n) \Rightarrow^k (u_1A_1v_1, u_2A_2v_2, \dots, u_nA_nv_n)$ v n-KGN Γ podle indukční hypotézy.

II. Nechť existuje derivační krok $(u_1A_1v_1, u_2A_2v_2, \dots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \dots, u_nx_nv_n)$ v n-KGP $\overline{\Gamma}$. Potom zřejmě pro n-KGP $\overline{\Gamma}$ platí $(A_1 \rightarrow x_1, A_2 \rightarrow x_2, \dots, A_n \rightarrow x_n) \in Q$.

Z algoritmu 3.3.2.1 potom plyne, že $(A_1, A_2, \ldots, A_n) \in Q$ a dále $A_i \rightarrow x_i \in P_i$ pro všechna $i = 1, \ldots, n$, tedy existuje i derivační krok $(u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$ v n-KGN Γ.

Z částí I. a II. plyne, že $(S_1, S_2, \ldots, S_n) \Rightarrow^{k+1} (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$ v n-KGN Γ.

Uvažujme tvrzení A pro nějaké $m \geq 0$ a $y_i \in T_i^*$ pro všechna $i = 1, \ldots, n$. Dostaneme implikaci tvaru: Pokud existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^* (y_1, y_2, \ldots, y_n)$ v n-KGN Γ, potom existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^* (y_1, y_2, \ldots, y_n)$ v n-KGP $\overline{\Gamma}$. Podle definice n-jazyka tedy platí $n\text{-}L(\Gamma) \subseteq n\text{-}L(\overline{\Gamma})$. Dále uvažujme tvrzení B pro nějaké $m \geq 0$ a $y_i \in T_i^*$ pro všechna $i = 1, \ldots, n$. Dostaneme implikaci tvaru: Pokud existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^* (y_1, y_2, \ldots, y_n)$ v n-KGP $\overline{\Gamma}$, potom existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^* (y_1, y_2, \ldots, y_n)$ v n-KGN Γ. Podle definice n-jazyka tedy platí $n\text{-}L(\overline{\Gamma}) \subseteq n\text{-}L(\Gamma)$. Z platnosti vztahů $n\text{-}L(\Gamma) \subseteq n\text{-}L(\overline{\Gamma})$ a $n\text{-}L(\overline{\Gamma}) \subseteq n\text{-}L(\Gamma)$ ihned plyne rovnost $n\text{-}L(\Gamma) = n\text{-}L(\overline{\Gamma})$. Pomocí tvrzení 3.3.1 dále dostáváme $L_X(\Gamma) = L_X(\overline{\Gamma})$ pro každé $X \in \{union, conc, first\}$.

3.3.3 Převod n-KGP na ekvivalentní n-KGN

Následující algoritmus popíše, jak lze převést libovolný n-KGP na ekvivalentní n-KGN, tedy na takový, jehož n-jazyk je shodný s n-jazykem původního n-KGP. Neformální popis myšlenky algoritmu je následující: V původním n-KGN byly kontrolovány n-tice daných pravidel, nyní musí být kontrolovány n-tice nonterminálních symbolů. Gramatiky nového n-KGN budou mít všechny nonterminální symboly (kromě počátečního) ve tvaru $<A, x>$, přičemž na nonterminální symbol tohoto tvaru půjdou aplikovat pouze pravidla tvaru $<A, x> \rightarrow y$, kde x je jistý kód pravé strany pravidla, tedy y. V každém nonterminálním symbolu je již tedy zakódováno, který druh pravidla může být na tento symbol aplikován. Kontrola n-tice nonterminálních symbolů v novém n-KGN tedy nasimuluje kontrolu n-tice pravidel v původním n-KGP. Správnost této myšlenky je pak formálně dokázána ve tvrzení 3.3.3.3.

3.3.3.1 Algoritmus: Převod n-KGP na ekvivalentní n-KGN

- ***Vstup:*** n-KGP $\Gamma = (G_1, G_2, \ldots, G_n, Q)$
- ***Výstup:*** n-KGN $\overline{\Gamma} = (\overline{G}_1, \overline{G}_2, \ldots, \overline{G}_n, \overline{Q})$; $n\text{-}L(\Gamma) = n\text{-}L(\overline{\Gamma})$
- ***Metoda:***
  - Nechť $G_i = (N_i, T_i, P_i, S_i)$ pro všechna $i = 1, \ldots, n$. Potom položme:
  - $\overline{G}_i = (\overline{N}_i, T_i, \overline{P}_i, S_i)$ pro všechna $i = 1, \ldots, n$, kde:
    - $\overline{N}_i := \{<A, x>: A \rightarrow x \in P_i\} \cup \{S_i\}$;
    - $\overline{P}_i := \{<A, x> \rightarrow y: A \rightarrow x \in P_i, y \in \tau_i(x)\} \cup \{S_i \rightarrow y: y \in \tau_i(S_i)\}$, kde τ_i je substituce z $N_i \cup T_i$ do $\overline{N}_i \cup T_i$ definovaná následovně: $\tau_i(a) = \{a\}$ pro všechna $a \in T_i$; $\tau_i(A) = \{<A, x>: A \rightarrow x \in P_i\}$ pro všechna $A \in N_i$;
    - $\overline{Q} := \{(<A_1, x_1>, <A_2, x_2>, \ldots, <A_n, x_n>): (A_1 \rightarrow x_1, A_2 \rightarrow x_2, \ldots, A_n \rightarrow x_n) \in Q\} \cup \{(S_1, S_2, \ldots, S_n)\}$

3.3.3.2 Příklad: Převod n-KGP na ekvivalentní n-KGN

Uvažujme následující 2-KGP $\Gamma = (G_1, G_2, Q)$, kde:

- $G_1 = (\{S_1, A_1\}, \{a, b, c\}, \{\mathbf{1}: S_1 \rightarrow aS_1, \mathbf{2}: S_1 \rightarrow aA_1, \mathbf{3}: A_1 \rightarrow bA_1c, \mathbf{4}: A_1 \rightarrow bc\}, S_1)$,
- $G_2 = (\{S_2, A_2\}, \{d\}, \{\mathbf{1}: S_2 \rightarrow S_2A_2, \mathbf{2}: S_2 \rightarrow A_2, \mathbf{3}: A_2 \rightarrow d\}, S_2)$,
- $Q = \{(\mathbf{1}, \mathbf{1}), (\mathbf{2}, \mathbf{2}), (\mathbf{3}, \mathbf{3}), (\mathbf{4}, \mathbf{3})\}$.

Substituce τ_1 pro všechna $X \in N_1 \cup T_1$ je definována následovně:

- $\tau_1(S_1) = \{<S_1, aS_1>, <S_1, aA_1>\}$,
- $\tau_1(A_1) = \{<A_1, bA_1c>, <A_1, bc>\}$,
- $\tau_1(a) = \{a\}$,
- $\tau_1(b) = \{b\}$,
- $\tau_1(c) = \{c\}$.

Substituce τ_2 pro všechna $X \in N_2 \cup T_2$ je definována následovně:

- $\tau_2(S_2) = \{<S_2, S_2A_2>, <S_2, A_2>\}$,
- $\tau_2(A_2) = \{<A_2, d>\}$,
- $\tau_2(d) = \{d\}$.

Ekvivalentní 2-KGN sestavený podle předchozího algoritmu bude mít strukturu $\overline{\Gamma} = (\overline{G_1}, \overline{G_2}, \overline{Q})$, kde:

- $\overline{G_1} = (\{S_1, <S_1, aS_1>, <S_1, aA_1>, <A_1, bA_1c>, <A_1, bc>\}, \{a, b, c\}$,
  $\{\ \mathbf{0{:}}\ S_1 \rightarrow <S_1, aS_1>, \mathbf{0'{:}}\ S_1 \rightarrow <S_1, aA_1>$,
  $\mathbf{1{:}}\ <S_1, aS_1> \rightarrow a<S_1, aS_1>, \mathbf{1'{:}}\ <S_1, aS_1> \rightarrow a<S_1, aA_1>$,
  $\mathbf{2{:}}\ <S_1, aA_1> \rightarrow a<A_1, bA_1c>, \mathbf{2'{:}}\ <S_1, aA_1> \rightarrow a<A_1, bc>$,
  $\mathbf{3{:}}\ <A_1, bA_1c> \rightarrow b<A_1, bA_1c>c, \mathbf{3'{:}}\ <A_1, bA_1c> \rightarrow b<A_1, bc>c$,
  $\mathbf{4{:}}\ <A_1, bA_1c> \rightarrow bc\ \}, S_1)$
- $\overline{G_2} = (\{S_2, <S_2, S_2A_2>, <S_2, A_2>, <A_2, d>\}, \{d\}$,
  $\{\ \mathbf{0{:}}\ S_2 \rightarrow <S_2, S_2A_2>, \mathbf{0'{:}}\ S_2 \rightarrow <S_2, A_2>$,
  $\mathbf{1{:}}\ <S_2, S_2A_2> \rightarrow <S_2, S_2A_2><A_2, d>, \mathbf{1'{:}}\ <S_2, S_2A_2> \rightarrow <S_2, A_2><A_2, d>$,
  $\mathbf{2{:}}\ <S_2, A_2> \rightarrow <A_2, d>$,
  $\mathbf{3{:}}\ <A_2, d> \rightarrow d\ \}, S_2)$
- $\overline{Q} = \{(<S_1, aS_1>, <S_2, S_2A_2>), (<S_1, aA_1>, <S_2, A_2>)$,
  $(<A_1, bA_1c>, <A_2, d>), (<A_1, bc>, <A_2, d>)\}$

Vytvoření 2-KGN $\overline{\Gamma}$ z 2-KGN Γ ilustruje obr. 3.4.

$\overline{\boldsymbol{G}}_{\mathbf{1}}$: $P_1 = \{S_1 \to aS_1,\ S_1 \to aA_1,\ A_1 \to bA_1c,\ A_1 \to bc\}$

$\overline{\boldsymbol{N}}_{\mathbf{1}} = \{\langle \boldsymbol{S_1, aS_1}\rangle, \langle \boldsymbol{S_1, aA_1}\rangle, \langle \boldsymbol{A_1, bA_1c}\rangle, \langle \boldsymbol{A_1, bc}\rangle\} \cup \{\boldsymbol{S_1}\}$

$\overline{\boldsymbol{P}}_{\mathbf{1}}$: $S_1 \to aS_1 \in P_1$:

$$\langle \boldsymbol{S_1, aS_1}\rangle \to \boldsymbol{a}\langle \boldsymbol{S_1, aS_1}\rangle \in \overline{P}_1$$
$$\langle \boldsymbol{S_1, aS_1}\rangle \to \underbrace{\boldsymbol{a}}_{\tau_1(a)}\underbrace{\langle \boldsymbol{S_1, aA_1}\rangle}_{\tau_1(S_1)} \in \overline{P}_1$$

$S_1 \to aA_1 \in P_1$:

$$\langle \boldsymbol{S_1, aA_1}\rangle \to \boldsymbol{a}\langle \boldsymbol{A_1, bA_1c}\rangle \in \overline{P}_1$$
$$\langle \boldsymbol{S_1, aA_1}\rangle \to \underbrace{\boldsymbol{a}}_{\tau_1(a)}\underbrace{\langle \boldsymbol{A_1, bc}\rangle}_{\tau_1(A_1)} \in \overline{P}_1$$

$A_1 \to bA_1c \in P_1$:

$$\langle \boldsymbol{A_1, bA_1c}\rangle \to \boldsymbol{b}\langle \boldsymbol{A_1, bA_1c}\rangle\boldsymbol{c} \in \overline{P}_1$$
$$\langle \boldsymbol{A_1, bA_1c}\rangle \to \underbrace{\boldsymbol{b}}_{\tau_1(b)}\underbrace{\langle \boldsymbol{A_1, bc}\rangle}_{\tau_1(A_1)}\underbrace{\boldsymbol{c}}_{\tau_1(c)} \in \overline{P}_1$$

$A_1 \to bc \in P_1$:

$$\langle \boldsymbol{A_1, bc}\rangle \to \underbrace{\boldsymbol{b}}_{\tau_1(b)}\underbrace{\boldsymbol{c}}_{\tau_1(c)} \in \overline{P}_1$$

S_1:

$$\boldsymbol{S_1} \to \langle \boldsymbol{S_1, aS_1}\rangle \in \overline{P}_1$$
$$\boldsymbol{S_1} \to \underbrace{\langle \boldsymbol{S_1, aA_1}\rangle}_{\tau_1(S_1)} \in \overline{P}_1$$

$\boldsymbol{G_2}$: $P_2 = \{S_2 \to S_2A_2,\ S_2 \to A_2,\ A_2 \to d\}$

$\boldsymbol{N_2} = \{\langle \boldsymbol{S_2, S_2A_2}\rangle, \langle \boldsymbol{S_2, A_2}\rangle, \langle \boldsymbol{A_2, d}\rangle\} \cup \{\boldsymbol{S_2}\}$

$\overline{\boldsymbol{P}}_{\mathbf{2}}$: $S_2 \to S_2A_2 \in P_2$:

$$\langle \boldsymbol{S_2, S_2A_2}\rangle \to \langle \boldsymbol{S_2, S_2A_2}\rangle\langle \boldsymbol{A_2, d}\rangle \in \overline{P}_2$$
$$\langle \boldsymbol{S_2, S_2A_2}\rangle \to \underbrace{\langle \boldsymbol{S_2, A_2}\rangle}_{\tau_2(S_2)}\underbrace{\langle \boldsymbol{A_2, d}\rangle}_{\tau_2(A_2)} \in \overline{P}_2$$

$S_2 \to A_2 \in P_2$:

$$\langle \boldsymbol{S_2, A_2}\rangle \to \underbrace{\langle \boldsymbol{A_2, d}\rangle}_{\tau_2(A_2)} \in \overline{P}_2$$

$A_2 \to d \in P_2$:

$$\langle \boldsymbol{A_2, d}\rangle \to \underbrace{\boldsymbol{d}}_{\tau_2(d)} \in \overline{P}_2$$

S_2:

$$\boldsymbol{S_2} \to \langle \boldsymbol{S_2, S_2A_2}\rangle \in \overline{P}_2$$
$$\boldsymbol{S_2} \to \underbrace{\langle \boldsymbol{S_2, A_2}\rangle}_{\tau_2(S_2)} \in \overline{P}_2$$

$Q = \{(S_1 \to aS_1, S_2 \to S_2A_2), (S_1 \to aA_1, S_2 \to A_2), (A_1 \to bA_1c, A_2 \to d), (A_1 \to bc, A_2 \to d)\}$

$\overline{Q} = \{(\langle \boldsymbol{S_1, aS_1}\rangle, \langle \boldsymbol{S_2, S_2A_2}\rangle), (\langle \boldsymbol{S_1, aA_1}\rangle, \langle \boldsymbol{S_2, A_2}\rangle), (\langle \boldsymbol{A_1, bA_1c}\rangle, \langle \boldsymbol{A_2, d}\rangle), (\langle \boldsymbol{A_1, bc}\rangle, \langle \boldsymbol{A_2, d}\rangle)\}$

(obr. 3.4: ilustrace vytvoření komponenty $\overline{Q}$ pro příklad 3.3.3.2)

3.3.3.3 Tvrzení

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je libovolný n-KGP. S n-KGP Γ na vstupu algoritmus 3.3.3.1 zastaví a správně vytvoří n-KGN $\overline{\Gamma} = (\overline{G}_1, \overline{G}_2, \ldots, \overline{G}_n, \overline{Q})$, pro který bude platit n-$L(\Gamma) = n$-$L(\overline{\Gamma})$ a $L_X(\Gamma) = L_X(\overline{\Gamma})$ pro každé $X \in \{union, conc, first\}$.

***Důkaz*:**

***Tvrzení A*:** Nechť existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^m (z_1, z_2, \ldots, z_n)$ v n-KGP Γ, kde $m \geq 0$, $z_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \ldots, n$. Potom také existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{m+1} (\overline{z}_1, \overline{z}_2, \ldots, \overline{z}_n)$ v n-KGN $\overline{\Gamma}$ pro libovolné $\overline{z}_i \in \tau_i(z_i)$.

***Důkaz*:** Tvrzení je dokázáno matematickou indukcí

Základní krok: Nechť $m = 0$. Potom $(S_1, S_2, \ldots, S_n) \Rightarrow^0 (S_1, S_2, \ldots, S_n)$ v n-KGP Γ. Poznamenejme, že také $(S_1, S_2, \ldots, S_n) \Rightarrow^1 (\overline{z}_1, \overline{z}_2, \ldots, \overline{z}_n)$ v n-KGN $\overline{\Gamma}$ pro libovolné $\overline{z}_i \in \tau_i(z_i)$, protože podle algoritmu 3.3.3.1 platí $(S_1, S_2, \ldots, S_n) \in \overline{Q}$ a dále $S_i \rightarrow \overline{z}_i \in \overline{P}_i$ pro libovolné $\overline{z}_i \in \tau_i(z_i)$ pro všechna $i = 1, \ldots, n$.

Indukční hypotéza: Předpokládejme, že tvrzení A platí pro všechny m-kroké derivace, kde $m = 0,\ldots, k$ pro nějaké $k \geq 0$.

Indukční krok: Uvažujme sekvenci derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{k+1} (y_1, y_2, \ldots, y_n)$ v n-KGP Γ. Potom zřejmě existuje multiforma $(u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n)$, kde $u_i \in T_i^*$, $A_i \in N_i$, $v_i \in (N_i \cup T_i)^*$, pro kterou platí $(S_1, S_2, \ldots, S_n) \Rightarrow^k (u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$ v n-KGP Γ, kde $u_ix_iv_i = y_i$ pro všechna $i = 1, \ldots, n$.

I. Ze sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^k (u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n)$ v n-KGP Γ plyne, že existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{k+1} (\overline{w}_1, \overline{w}_2, \ldots, \overline{w}_n)$ v n-KGN $\overline{\Gamma}$ pro libovolné $\overline{w}_i \in \tau_i(u_iA_iv_i)$ pro všechna $i = 1, \ldots, n$ podle indukční hypotézy.

II. Nechť existuje derivační krok $(u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$ v n-KGP Γ. Potom zřejmě pro n-KGP Γ platí $(A_1 \rightarrow x_1, A_2 \rightarrow x_2, \ldots, A_n \rightarrow x_n) \in Q$. Z algoritmu 3.3.3.1 potom plyne, že $(<A_1, x_1>, <A_2, x_2>, \ldots, <A_n, x_n>) \in \overline{Q}$ v n-KGN $\overline{\Gamma}$ a dále $<A_i, x_i> \rightarrow \overline{y}_i \in \overline{P}_i$ pro libovolné $\overline{y}_i \in \tau_i(x_i)$ pro všechna $i = 1, \ldots, n$. Nechť $\overline{w}_i$ je větná forma tvaru $\overline{u}_i<A_i, x_i>\overline{v}_i$ pro všechna $i = 1, \ldots, n$, kde $\overline{u}_i \in \tau_i(u_i)$, $\overline{v}_i \in \tau_i(v_i)$. Poznamenejme, že podle definice substituce τ_i platí $<A_i, x_i> \in \tau_i(A_i)$. Tedy v n-KGN $\overline{\Gamma}$ existuje derivační krok $(\overline{u}_1<A_1, x_1>\overline{v}_1, \overline{u}_2<A_2, x_2>\overline{v}_2, \ldots, \overline{u}_n<A_n, x_n>\overline{v}_n) \Rightarrow (\overline{u}_1\overline{y}_1\overline{v}_1, \overline{u}_2\overline{y}_2\overline{v}_2, \ldots, \overline{u}_n\overline{y}_n\overline{v}_n)$, kde $\overline{u}_i\overline{y}_i\overline{v}_i$ je libovolná forma splňující $\overline{u}_i\overline{y}_i\overline{v}_i \in \tau_i(u_ix_iv_i)$ pro všechna $i = 1, \ldots, n$.

Z částí I. a II. plyne, že $(S_1, S_2, \ldots, S_n) \Rightarrow^{k+1} (\overline{u}_1<A_1, x_1>\overline{v}_1, \overline{u}_2<A_2, x_2>\overline{v}_2, \ldots, \overline{u}_n<A_n, x_n>\overline{v}_n) \Rightarrow (\overline{u}_1\overline{y}_1\overline{v}_1, \overline{u}_2\overline{y}_2\overline{v}_2, \ldots, \overline{u}_n\overline{y}_n\overline{v}_n)$ v n-KGN $\overline{\Gamma}$ pro libovolné $\overline{u}_i\overline{y}_i\overline{v}_i \in \tau_i(u_ix_iv_i)$ pro všechna $i = 1, \ldots, n$. Nechť $\overline{z}_i = \overline{u}_i\overline{y}_i\overline{v}_i$ a $z_i = u_ix_iv_i$ pro všechna $i = 1, \ldots, n$. Potom v n-KGN $\overline{\Gamma}$ existuje posloupnost derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{k+2} (\overline{z}_1, \overline{z}_2, \ldots, \overline{z}_n)$ pro libovolné $\overline{z}_i \in \tau_i(z_i)$.

Tvrzení B: Nechť existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^m (\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_n)$ v n-KGN $\bar{\Gamma}$, kde $m \geq 1$, $\bar{z}_i \in (\bar{N}_i \cup T_i)^*$ pro všechna $i = 1, \ldots, n$. Potom také existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{m-1} (z_1, z_2, \ldots, z_n)$ v n-KGP Γ, přičemž $z_i \in \tau_i^{-1}(\bar{z}_i)$ pro všechna $i = 1, \ldots, n$.

Důkaz: Tvrzení je dokázáno matematickou indukcí

Základní krok: Nechť $m = 1$. Potom $(S_1, S_2, \ldots, S_n) \Rightarrow^1 (\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_n)$ v n-KGN $\bar{\Gamma}$, tedy $S_i \rightarrow \bar{z}_i \in \bar{P}_i$ pro všechna $i = 1, \ldots, n$. Podle algoritmu 3.3.3.1 platí, že $\bar{z}_i \in \tau_i(S_i)$ pro všechna $i = 1, \ldots, n$, tedy $(S_1, S_2, \ldots, S_n) \Rightarrow^0 (S_1, S_2, \ldots, S_n)$ v n-KGP Γ a platí, že $S_i \in \tau_i^{-1}(\bar{z}_i)$ pro všechna $i = 1, \ldots, n$.

Indukční hypotéza: Předpokládejme, že tvrzení B platí pro všechny m-kroké derivace, kde $m = 1,\ldots, k$ pro nějaké $k \geq 1$.

Indukční krok: Uvažujme sekvenci derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{k+1} (\bar{y}_1, \bar{y}_2, \ldots, \bar{y}_n)$ v n-KGN $\bar{\Gamma}$. Potom zřejmě existuje multiforma $(\bar{u}_1 < A_1, x_1 > \bar{v}_1, \bar{u}_2 < A_2, x_2 > \bar{v}_2, \ldots, \bar{u}_n < A_n, x_n > \bar{v}_n)$, kde $\bar{u}_i \in T_i^*$, $< A_i, x_i > \in \bar{N}_i$, $\bar{v}_i \in (\bar{N}_i \cup T_i)^*$ pro kterou platí $(S_1, S_2, \ldots, S_n) \Rightarrow^k (\bar{u}_1 < A_1, x_1 > \bar{v}_1, \bar{u}_2 < A_2, x_2 > \bar{v}_2, \ldots, \bar{u}_n < A_n, x_n > \bar{v}_n) \Rightarrow (\bar{u}_1\bar{x}_1\bar{v}_1, \bar{u}_2\bar{x}_2\bar{v}_2, \ldots, \bar{u}_n\bar{x}_n\bar{v}_n)$ v n-KGN $\bar{\Gamma}$, kde $\bar{u}_i\bar{x}_i\bar{v}_i = \bar{y}_i$ pro všechna $i = 1, \ldots, n$.

I. Ze sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^k (\bar{u}_1 < A_1, x_1 > \bar{v}_1, \bar{u}_2 < A_2, x_2 > \bar{v}_2, \ldots, \bar{u}_n < A_n, x_n > \bar{v}_n)$ v n-KGN $\bar{\Gamma}$ plyne, že existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{k-1} (w_1, w_2, \ldots, w_n)$ n-KGP Γ, kde $w_i \in \tau_i^{-1}(\bar{u}_i < A_i, x_i > \bar{v}_i)$ pro všechna $i = 1, \ldots, n$ podle indukční hypotézy.

II. Nechť existuje derivační krok $(\bar{u}_1 < A_1, x_1 > \bar{v}_1, \bar{u}_2 < A_2, x_2 > \bar{v}_2, \ldots, \bar{u}_n < A_n, x_n > \bar{v}_n) \Rightarrow (\bar{u}_1\bar{x}_1\bar{v}_1, \bar{u}_2\bar{x}_2\bar{v}_2, \ldots, \bar{u}_n\bar{x}_n\bar{v}_n)$ v n-KGN$\bar{\Gamma}$. Potom zřejmě pro n-KGN$\bar{\Gamma}$ platí $(< A_1, x_1 >, < A_2, x_2 >, \ldots, < A_n, x_n >) \in \bar{Q}$ a $< A_i, x_i > \rightarrow \bar{x}_i \in \bar{P}_i$ pro všechna $i = 1, \ldots, n$. Z algoritmu 3.3.3.1 potom plyne, že $(A_1 \rightarrow x_1, A_2 \rightarrow x_2, \ldots, A_n \rightarrow x_n) \in Q$ v n-KGP Γ a dále $A_i \rightarrow x_i \in P_i$, přičemž platí $\bar{x}_i \in \tau_i(x_i)$, tedy $x_i \in \tau_i^{-1}(\bar{x}_i)$. Pro všechna $i = 1, \ldots, n$ můžeme w_i zapsat ve tvaru $w_i = u_iA_iv_i$, kde $u_i \in \tau_i^{-1}(\bar{u}_i)$, $v_i \in \tau_i^{-1}(\bar{v}_i)$. Poznamenejme, že podle definice substituce τ_i platí $A_i \in \tau_i^{-1}(< A_i, x_i >)$. Tedy v n-KGP Γ existuje derivační krok $(u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$, kde $u_i \in \tau_i^{-1}(\bar{u}_i)$, $v_i \in \tau_i^{-1}(\bar{v}_i)$ a $x_i \in \tau_i^{-1}(\bar{x}_i)$ pro všechna $i = 1, \ldots, n$, což znamená, že $u_ix_iv_i \in \tau_i^{-1}(\bar{u}_i\bar{x}_i\bar{v}_i)$ pro všechna $i = 1, \ldots, n$.

Z částí I. a II. plyne, že $(S_1, S_2, \ldots, S_n) \Rightarrow^{k+1} (u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$ v n-KGP Γ, kde $u_ix_iv_i \in \tau_i^{-1}(\bar{u}_i\bar{x}_i\bar{v}_i)$ pro všechna $i = 1, \ldots, n$. Nechť $\bar{z}_i = \bar{u}_i\bar{y}_i\bar{v}_i$ a $z_i = u_ix_iv_i$ pro všechna $i = 1, \ldots, n$. Potom v n-KGP Γ existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{k+2} (z_1, z_2, \ldots, z_n)$ pro libovolné $z_i \in \tau_i^{-1}(\bar{z}_i)$.

Uvažujme tvrzení A pro nějaké $m \geq 0$ a $z_i \in T_i^*$ pro všechna $i = 1, \ldots, n$. Dostaneme implikaci tvaru: Pokud existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^* (z_1, z_2, \ldots, z_n)$ v n-KGP Γ, potom existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^* (\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_n)$ v n-KGN $\bar{\Gamma}$, kde $\bar{z}_i \in \tau_i(z_i)$. Protože $\tau_i(a_i) = \{a_i\}$ pro všechna $a_i \in T_i$, musí tedy pro všechna $i = 1, \ldots, n$ platit $\bar{z}_i = z_i$. Tedy existuje i sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^* (z_1, z_2,$

$\ldots, z_n)$ v n-KGN $\overline{\Gamma}$. Podle definice n-jazyka tedy platí $n\text{-}L(\Gamma) \subseteq n\text{-}L(\overline{\Gamma})$.Uvažujme tvrzení B pro nějaké $m \geq 0$, $y_i \in T_i^*$ pro všechna $i = 1, \ldots, n$. Dostaneme implikaci tvaru: Pokud existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^* (\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_n)$ v n-KGN $\overline{\Gamma}$, potom existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^* (z_1, z_2, \ldots, z_n)$ v n-KGP Γ, kde $z_i \in \tau_i^{-1}(\bar{z}_i)$. Protože $\tau_i(a_i) = \{a_i\}$ pro všechna $a_i \in T_i$, musí tedy pro všechna $i = 1, \ldots, n$ platit $z_i = \bar{z}_i$. Tedy existuje i sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^* (\bar{z}_1, \bar{z}_2, \ldots, \bar{z}_n)$ v n-KGP Γ. Podle definice n-jazyka tedy platí $n\text{-}L(\overline{\Gamma}) \subseteq n\text{-}L(\Gamma)$. Z platnosti vztahů $n\text{-}L(\Gamma) \subseteq n\text{-}L(\overline{\Gamma})$ a $n\text{-}L(\overline{\Gamma}) \subseteq n\text{-}L(\Gamma)$ ihned plyne rovnost $n\text{-}L(\Gamma) = n\text{-}L(\overline{\Gamma})$. Pomocí tvrzení 3.3.1 dále dostáváme $L_X(\Gamma) = L_X(\overline{\Gamma})$ pro každé $X \in \{union, conc, first\}$.

3.3.4 Důsledek

Třída jazyků generovaná pomocí n-KGN v módu X, kde $X \in \{union, conc, first\}$, je ekvivalentní s třídou jazyků generovanou pomocí n-KGP v módu X.

***Důkaz*:** Tento důsledek plyne z algoritmu 3.3.2.1 a z algoritmu 3.3.3.1.

3.3.5 Tvrzení

Pro každý rekurzivně vyčíslitelný jazyk L nad abecedou T existuje 2-KGP, $\Gamma = ((\overline{N}_1, T, \overline{P}_1, S_1), (\overline{N}_2, T, \overline{P}_2, S_2), Q)$, přičemž platí následující 2 rovnosti:

1) $\{w: (S_1, S_2) \Rightarrow^* (w, w), w \in T^*\} = L$,
2) $\{(w_1, w_2): (S_1, S_2) \Rightarrow^* (w_1, w_2), w_1, w_2 \in T^*, w_1 \neq w_2\} = \emptyset$.

***Důkaz*:**

Pro každý rekurzivně vyčíslitelný jazyk L nad abecedou T existují dvě bezkontextové gramatiky $G_1 = (N_1, \overline{T}, P_1, S_1)$ a $G_2 = (N_2, \overline{T}, P_2, S_2)$ a homomorfismus h: $\overline{T} \rightarrow T^*$ takový, že $L = \{h(x) : x \in L(G_1) \cap L(G_2)\}$ (viz. tvrzení 10.3.1 v [18]). Dále pro každou bezkontextovou gramatiku existuje ekvivalentní bezkontextová gramatika v Greibachové normální formě (viz. tvrzení 2.1.4.2 v [12]). Proto bez újmy na obecnosti můžeme předpokládat, že gramatiky G_1 a G_2 jsou v Greibachové normální formě. Uvažujme 2-KGP $\Gamma = (G_1, G_2, Q)$, kde:

- $G_1 = (\overline{N}_1, T, \overline{P}_1, S_1)$, kde $\overline{N}_1 = N_1 \cup \{\bar{a} : a \in \overline{T}\}$, $\overline{P}_1 = \{A \rightarrow \bar{a}x : A \rightarrow ax \in P_1, a \in \overline{T}, x \in N_1^*\} \cup \{\bar{a} \rightarrow h(a): a \in \overline{T}\}$,
- $G_2 = (\overline{N}_2, T, \overline{P}_2, S_2)$, kde $\overline{N}_2 = N_2 \cup \{\bar{a} : a \in \overline{T}\}$, $\overline{P}_2 = \{A \rightarrow \bar{a}x : A \rightarrow ax \in P_2, a \in \overline{T}, x \in N_2^*\} \cup \{\bar{a} \rightarrow h(a): a \in \overline{T}\}$,
- $Q = \{(A_1 \rightarrow \bar{a}x_1, A_2 \rightarrow \bar{a}x_2): A_1 \rightarrow \bar{a}x_1 \in \overline{P}_1, A_2 \rightarrow \bar{a}x_2 \in \overline{P}_2, a \in \overline{T}\} \cup \{(\bar{a} \rightarrow h(a), \bar{a} \rightarrow h(a)): a \in \overline{T}\}$.

Nyní, dokážeme, že pro 2-KGP $\Gamma = (G_1, G_2, Q)$ platí:

1) $\{w: (S_1, S_2) \Rightarrow^* (w, w), w \in T^*\} = L$.
2) $\{(w_1, w_2): (S_1, S_2) \Rightarrow^* (w_1, w_2), w_1, w_2 \in T^*, w_1 \neq w_2\} = \emptyset$.

***Tvrzení A*:** $\{w: (S_1, S_2) \Rightarrow^* (w, w), w \in T^*\} = L$:

***Důkaz*:** I. Dokážeme, že $L \subseteq \{w: (S_1, S_2) \Rightarrow^* (w, w), w \in T^*\}$:

Nechť $w \in L$ je libovolný řetězec. Potom, existuje řetězec $a_1a_2...a_n \in \overline{T}^*$, pro který platí:

- $a_1a_2...a_n \in L(G_1)$
- $a_1a_2...a_n \in L(G_2)$
- $h(a_1a_2...a_n) = w$

To znamená, že v gramatikách G_1 a G_2 musí existovat následující sekvence derivačních kroků:

- $S_1 \Rightarrow a_1x_1\ [p_1] \Rightarrow a_1a_2x_2\ [p_2] \Rightarrow \dots \Rightarrow a_1a_2...a_n\ [p_n]$,
- $S_2 \Rightarrow a_1y_1\ [r_1] \Rightarrow a_1a_2y_2\ [r_2] \Rightarrow \dots \Rightarrow a_1a_2...a_n\ [r_n]$,

kde $a_i \in \overline{T}$, $x_i \in N_1^*$, $y_i \in N_2^*$, $p_i \in P_1$, $r_i \in P_2$ pro všechna $i = 1, \dots, n$. Z konstrukce množiny $Q = \{(A_1 \rightarrow \overline{a}x_1, A_2 \rightarrow \overline{a}x_2)\text{: } A_1 \rightarrow \overline{a}x_1 \in \overline{P_1}, A_2 \rightarrow \overline{a}x_2 \in \overline{P_2}, a \in \overline{T}\} \cup \{(\overline{a} \rightarrow h(a), \overline{a} \rightarrow h(a))\text{: } a \in \overline{T}\}$ je zřejmé, že platí následující 2 podmínky:

1) Nechť $p_i = A_i \rightarrow a_iu_i \in \overline{P_1}$, $r_i = B_i \rightarrow a_iv_i \in \overline{P_2}$. Potom zřejmě $(A_i \rightarrow \overline{a}_iu_i, B_i \rightarrow \overline{a}_iv_i) \in Q$ pro všechna $i = 1, \dots, n$,

2) $(\overline{a}_i \rightarrow h(\overline{a}_i), \overline{a}_i \rightarrow h(\overline{a}_i)) \in Q$ pro všechna $i =1, \dots, n$.

V n-KGP Γ tedy zřejmě existuje následující sekvence derivačních kroků:

$$\begin{aligned}(S_1, S_2) &\Rightarrow (\overline{a}_1x_1, \overline{a}_1y_1) \Rightarrow (h(\overline{a}_1)x_1, h(\overline{a}_1)y_1)\\ &\Rightarrow (h(\overline{a}_1)\overline{a}_2x_2, h(\overline{a}_1)\overline{a}_2y_2) \Rightarrow (h(\overline{a}_1)h(\overline{a}_2)x_2, h(\overline{a}_1)h(\overline{a}_2)y_2)\\ &\Rightarrow \dots\\ &\Rightarrow (h(\overline{a}_1)h(\overline{a}_2)\dots h(\overline{a}_n), h(\overline{a}_1)h(\overline{a}_2)\dots h(\overline{a}_n)) =\\ &\quad (h(\overline{a}_1\overline{a}_2\dots\overline{a}_n), h(\overline{a}_1\overline{a}_2\dots\overline{a}_n)) = (w, w)\end{aligned}$$

Tedy $(S_1, S_2) \Rightarrow^* (w, w)$.

II. Dokážeme, že $\{w: (S_1, S_2) \Rightarrow^* (w, w), w \in T^*\} \subseteq L$:

Nechť $(S_1, S_2) \Rightarrow^* (w, w)$. Pak v n-KGP Γ zřejmě existuje následující sekvence derivačních kroků:

$$\begin{aligned}(S_1, S_2) &\Rightarrow (\overline{a}_1x_1, \overline{a}_1y_1) \Rightarrow (h(\overline{a}_1)x_1, h(\overline{a}_1)y_1)\\ &\Rightarrow (h(\overline{a}_1)\overline{a}_2x_2, h(\overline{a}_1)\overline{a}_2y_2) \Rightarrow (h(\overline{a}_1)h(\overline{a}_2)x_2, h(\overline{a}_1)h(\overline{a}_2)y_2)\\ &\Rightarrow \dots\\ &\Rightarrow (h(\overline{a}_1)h(\overline{a}_2)\dots h(\overline{a}_n), h(\overline{a}_1)h(\overline{a}_2)\dots h(\overline{a}_n)) =\\ &\quad (h(\overline{a}_1\overline{a}_2\dots\overline{a}_n), h(\overline{a}_1\overline{a}_2\dots\overline{a}_n)) = (w, w)\end{aligned}$$

Analogicky jako v části I., můžeme dokázat, že v gramatikách G_1 a G_2 musí existovat následující sekvence derivačních kroků:

$S_1 \Rightarrow a_1x_1\ [p_1] \Rightarrow a_1a_2x_2\ [p_2] \Rightarrow \dots \Rightarrow a_1a_2...a_n\ [p_n]$,

$S_2 \Rightarrow a_1y_1\ [r_1] \Rightarrow a_1a_2y_2\ [r_2] \Rightarrow \dots \Rightarrow a_1a_2...a_n\ [r_n]$.

Tedy $a_1a_2...a_n \in L(G_1)$, $a_1a_2...a_n \in L(G_2)$ a $h(a_1a_2...a_n) = w$, což znamená, že $w \in L$.

***Tvrzení B*:** $\{(w_1, w_2): (S_1, S_2) \Rightarrow^* (w_1, w_2), w_1, w_2 \in T^*, w_1 \neq w_2\} = \emptyset$

***Důkaz (sporem)*:**

Nechť $\{(w_1, w_2): (S_1, S_2) \Rightarrow^* (w_1, w_2), w_1, w_2 \in T^*, w_1 \neq w_2\} \neq \emptyset$. Pak tedy zřejmě existují dva různé řetězce $w_1 = h(\overline{a}_1)h(\overline{a}_2)\ldots h(\overline{a}_n)$ a $w_2 = h(\overline{b}_1)h(\overline{b}_2)\ldots h(\overline{b}_n)$, pro které platí: $(S_1, S_2) \Rightarrow^* (w_1, w_2)$.

I. Předpokládejme, že pro všechna $i = 1, \ldots, n$ platí: $\overline{a}_i = \overline{b}_i$. Potom tedy platí $w_1 = h(\overline{a}_1)h(\overline{a}_2)\ldots h(\overline{a}_n) = h(\overline{b}_1)h(\overline{b}_2)\ldots h(\overline{b}_n) = w_2$. Ale w_1 a w_2 jsou různé řetězce, což je spor.

II. Předpokládejme, že existuje $k \leq n$, pro které platí $\overline{a}_k \neq \overline{b}_k$.

Pak v n-KGP Γ zřejmě existuje následující sekvence derivačních kroků:

$$\begin{aligned}(S_1, S_2) &\Rightarrow (\overline{a}_1 x_1, \overline{a}_1 y_1) \Rightarrow (h(\overline{a}_1)x_1, h(\overline{a}_1)y_1)\\ &\Rightarrow (h(\overline{a}_1)\overline{a}_2 x_2, h(\overline{a}_1)\overline{a}_2 y_2) \Rightarrow (h(\overline{a}_1)h(\overline{a}_2)x_2, h(\overline{a}_1)h(\overline{a}_2)y_2)\\ &\Rightarrow \ldots\\ &\Rightarrow (h(\overline{a}_1)h(\overline{a}_2)\ldots h(\overline{a}_{k-1})x_{k-1}, h(\overline{a}_1)h(\overline{a}_2)\ldots h(\overline{a}_{k-1})y_{k-1})\end{aligned}$$

Potom musí zřejmě následovat derivační krok $(x_{k-1}, y_{k-1}) \Rightarrow (\overline{a}_k x_k, \overline{b}_k y_k)$, kde $\overline{a}_k \neq \overline{b}_k$. Protože $Q = \{(A_1 \rightarrow \overline{a}x_1, A_2 \rightarrow \overline{a}x_2): A_1 \rightarrow \overline{a}x_1 \in \overline{P}_1, A_2 \rightarrow \overline{a}x_2 \in \overline{P}_2, a \in \overline{T}\} \cup \{(\overline{a} \rightarrow h(a), \overline{a} \rightarrow h(a)): a \in \overline{T}\}$, musí se nyní aplikovat nějaká dvojici pravidel tvaru $(A_1 \rightarrow \overline{a}_k x_1, A_2 \rightarrow \overline{b}_k x_2)$, přičemž $\overline{a}_k = \overline{b}_k$, což je spor.

3.3.6 Tvrzení

Pro každý rekurzivně vyčíslitelný jazyk L nad abecedou T existuje 2-KGP $\Gamma = (G_1, G_2, Q)$ takový, že $L_{union}(\Gamma) = L$.

Důkaz:

Podle tvrzení 3.3.5 platí: Pro každý rekurzivně vyčíslitelný jazyk L nad abecedou T existuje 2-KGP, $\Gamma = ((\overline{N}_1, T, \overline{P}_1, S_1), (\overline{N}_2, T, \overline{P}_2, S_2), Q)$ takový, že

1) $\{w: (S_1, S_2) \Rightarrow^* (w, w), w \in T^*\} = L$,
2) $\{(w_1, w_2): (S_1, S_2) \Rightarrow^* (w_1, w_2), w_1, w_2 \in T^*, w_1 \neq w_2\} = \emptyset$.

Nechť $\Gamma = \overline{\Gamma}$. Potom $L_{union}(\Gamma) = \bigcup_{i=1}^{2} \{w_i: (S_1, S_2) \Rightarrow^* (w_1, w_2), w_i \in T^*\} = \{w: (S_1, S_2) \Rightarrow^* (w, w), w \in T^*\} \cup \{w_1: (S_1, S_2) \Rightarrow^* (w_1, w_2), w_i \in T^* \text{ pro } i = 1, 2, w_1 \neq w_2\} \cup \{w_2: (S_1, S_2) \Rightarrow^* (w_1, w_2), w_i \in T^* \text{ pro } i = 1, 2, w_1 \neq w_2\} = \{w: (S_1, S_2) \Rightarrow^* (w, w), w \in T^*\} \cup \emptyset \cup \emptyset = \{w: (S_1, S_2) \Rightarrow^* (w, w), w \in T^*\} = L$.

3.3.7 Tvrzení

Pro každý rekurzivně vyčíslitelný jazyk L nad abecedou T existuje 2-KGP $\Gamma = (G_1, G_2, Q)$ takový, že $L_{first}(\Gamma) = L$.

Důkaz:

Podle tvrzení 3.3.5 platí: Pro každý rekurzivně vyčíslitelný jazyk L nad abecedou T existuje 2-KGP $\Gamma = ((\overline{N}_1, T, \overline{P}_1, S_1), (\overline{N}_2, T, \overline{P}_2, S_2), Q)$ takový, že:

1) $\{w: (S_1, S_2) \Rightarrow^* (w, w), w \in T^*\} = L$,
2) $\{(w_1, w_2): (S_1, S_2) \Rightarrow^* (w_1, w_2), w_1, w_2 \in T^*, w_1 \neq w_2\} = \emptyset$.

Nechť $\Gamma = \overline{\Gamma}$. Potom $L_{first}(\Gamma) = \{w_1: (S_1, S_2) \Rightarrow^* (w_1, w_2), w_i \in T^*$ pro $i = 1, 2\} = \{w: (S_1, S_2) \Rightarrow^* (w, w), w \in T^*\} \cup \{w_1: (S_1, S_2) \Rightarrow^* (w_1, w_2), w_i \in T^*$ pro $i = 1, 2, w_1 \neq w_2\} = \{w: (S_1, S_2) \Rightarrow^* (w, w), w \in T^*\} \cup \emptyset = \{w: (S_1, S_2) \Rightarrow^* (w, w), w \in T^*\} = L$

3.3.8 Tvrzení

Pro každý rekurzivně vyčíslitelný jazyk L nad abecedou T existuje 2-KGP $\Gamma = (G_1, G_2, Q)$ takový, že $L_{conc}(\Gamma) = L$.

Důkaz:

Podle tvrzení 3.3.5 platí: Pro každý rekurzivně vyčíslitelný jazyk L nad abecedou T existuje 2-KGP, $\overline{\Gamma} = ((\overline{N}_1, T, \overline{P}_1, S_1), (\overline{N}_2, T, \overline{P}_2, S_2), Q)$ takový, že:

1) $\{w: (S_1, S_2) \Rightarrow^* (w, w), w \in T^*\} = L$,
2) $\{(w_1, w_2): (S_1, S_2) \Rightarrow^* (w_1, w_2), w_1, w_2 \in T^*, w_1 \neq w_2\} = \emptyset$.

Nechť $G_1 = (N_1, T, P_1, S_1)$ a $G_2 = (N_2, \emptyset, \overline{P}_2, S_2)$, kde $\overline{P}_2 = \{A \rightarrow g(x): A \rightarrow x \in P_2\}$, kde g je homomorfismus z $(N_2 \cup T)$ do N_2 definovaný následovně: Pro všechna $X \in N_2$: $g(X) = X$, pro všechna $X \in T$: $g(X) = \varepsilon$. Nyní pro daný 2-KGP Γ dokážeme, že $L_{conc}(\Gamma) = L$.

I. Dokažme: $L \subseteq L_{conc}(\Gamma)$: Nechť $w \in L$. Potom zřejmě v 2-KGP $\overline{\Gamma}$ existuje sekvence derivačních kroků tvaru $(S_1, S_2) \Rightarrow^* (w, w)$ a tedy v 2-KGP Γ existuje sekvence derivačních kroků tvaru $(S_1, S_2) \Rightarrow^* (w, g(w))$, neboť pro druhou gramatiku byly takto předefinovány všechny terminální symboly. Pro všechna $a \in T$: $g(a) = \varepsilon$, tedy také pro všechna $w \in T^*$: $g(w) = \varepsilon$. Můžeme tedy psát, že v 2-KGP Γ existuje sekvence derivačních kroků tvaru $(S_1, S_2) \Rightarrow^* (w, \varepsilon)$. Proto řetězec $w\varepsilon = w \in L_{conc}(\Gamma)$.

II. Dokažme: $L_{conc}(\Gamma) \subseteq L$: Nechť $w \in L_{conc}(\Gamma)$. Potom zřejmě v 2-KGP Γ existuje sekvence derivačních kroků tvaru $(S_1, S_2) \Rightarrow^* (w, \varepsilon)$, protože gramatika G_2 generuje pouze prázdný řetězec. $g(x) = \varepsilon$ pro všechna $x \in T^*$, takže v 2-KGP $\overline{\Gamma}$ existuje sekvence derivačních kroků tvaru $(S_1, S_2) \Rightarrow^* (w, x)$, kde x je nějaký řetězec. Podle tvrzení 3.3.5 platí, že $x = w$, tedy v 2-KGP $\overline{\Gamma}$ existuje sekvence derivačních kroků tvaru $(S_1, S_2) \Rightarrow^* (w, w)$. Tedy $w \in L$.

4 Obecné multigenerativní gramatické systémy

V této kapitole jsou zobecněny kanonické multigenerativní gramatické systémy definované v předcházející kapitole. Toto zobecnění je v tom, že při provedení derivačního kroku v jednotlivých gramatikách není kladen požadavek, aby byla provedena nejlevější derivace. Obecně tedy může být přepsán kterýkoliv nonterminální symbol ve větné formě. Definice obou obecných gramatických systémů jsou shodné s definicemi z minulé kapitoly. Rozdíl je pouze v definici derivačního kroku obou systémů.

4.1 Obecný *n*-generativní nonterminálově synchronizovaný gramatický systém

4.1.1 Definice obecného n-generativního nonterminálově synchronizovaného gramatického systému

Obecný n-generativní nonterminálově synchronizovaný GS (n-OGN) je n+1-tice

$$\Gamma = (G_1, G_2, \ldots, G_n, Q), \text{ kde:}$$

- $G_i = (N_i, T_i, P_i, S_i)$ je bezkontextová gramatika pro všechna $i = 1, \ldots, n$.
- Q je konečná množina *kontrolních n-tic nonterminálů* tvaru $(A_1, A_2, \ldots, A_n)$, kde $A_i \in N_i$ pro všechna $i = 1, \ldots, n$.

4.1.2 Definice multiformy

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-OGN. Potom *multiforma* je n-tice $\chi = (x_1, x_2, \ldots, x_n)$, kde $x_i \in (T_i \cup N_i)^*$ pro všechna $i = 1, \ldots, n$.

4.1.3 Definice přímého derivačního kroku v n-OGN

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-OGN, nechť $\chi = (u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n)$, $\bar{\chi} = (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$, jsou dvě multiformy, kde $A_i \in N_i$, $u_i, v_i, x_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \ldots, n$. Dále nechť $A_i \to x_i \in P_i$ pro všechna $i = 1, \ldots, n$ a $(A_1, A_2, \ldots, A_n) \in Q$. Pak říkáme, že χ *přímo derivuje* $\bar{\chi}$ a zapisujeme $\chi \Rightarrow \bar{\chi}$.

4.1.4 Definice sekvence derivačních kroků v n-OGN

Sekvence derivačních kroků pro n-OGN je definována analogicky jako sekvence derivačních kroků pro n-KGN.

4.1.5 Definice n-jazyka a jazyků v různých módech v n-OGP

n-jazyk generovaný n-OGN je definován analogicky jako n-jazyk generovaný n-KGN. Stejně tak jazyky v různých módech pro n-OGN jsou definovány analogicky jako jazyky generované pomocí n-KGN.

4.2 Obecný *n*-generativní pravidlově synchronizovaný gramatický systém

4.2.1 Definice obecného n-generativního pravidlově synchronizovaného gramatického systému

Obecný n-generativní pravidlově synchronizovaný GS (n-OGP) je n+1-tice

$$\Gamma = (G_1, G_2, \ldots, G_n, Q), \text{ kde}$$

- $G_i = (N_i, T_i, P_i, S_i)$ je bezkontextová gramatika pro všechna $i = 1, \ldots, n$.
- Q je konečná množina *kontrolních n-tic pravidel* tvaru $(p_1, p_2, \ldots, p_n)$, kde $p_i \in P_i$ pro všechna $i = 1, \ldots, n$.

4.2.2 Definice multiformy

Multiforma pro n-OGP je definována shodně s multiformou pro n-OGN.

4.2.3 Definice přímého derivační kroku v n-OGP

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-OGP, nechť $\chi = (u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n)$, $\bar{\chi} = (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$, jsou dvě multiformy, kde $A_i \in N_i$, $u_i, v_i, x_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \ldots, n$. Dále nechť p_i: $A_i \rightarrow x_i \in P_i$ pro všechna $i = 1, \ldots, n$ a $(p_1, p_2, \ldots, p_n) \in Q$. Pak říkáme, že χ *přímo derivuje* $\bar{\chi}$ a zapisujeme $\chi \Rightarrow \bar{\chi}$.

4.2.4 Definice sekvence derivačních kroků v n-OGP

Sekvence derivačních kroků pro n-OGP je definována analogicky jako sekvence derivačních kroků pro n-KGP.

4.2.5 Definice n-jazyka a jazyků v různých módech v n-OGP

n-jazyk generovaný n-OGP je definován analogicky jako n-jazyk generovaný n-KGP. Stejně tak jazyky v různých módech pro n-OGP jsou definovány analogicky jako jazyky generované pomocí n-KGP.

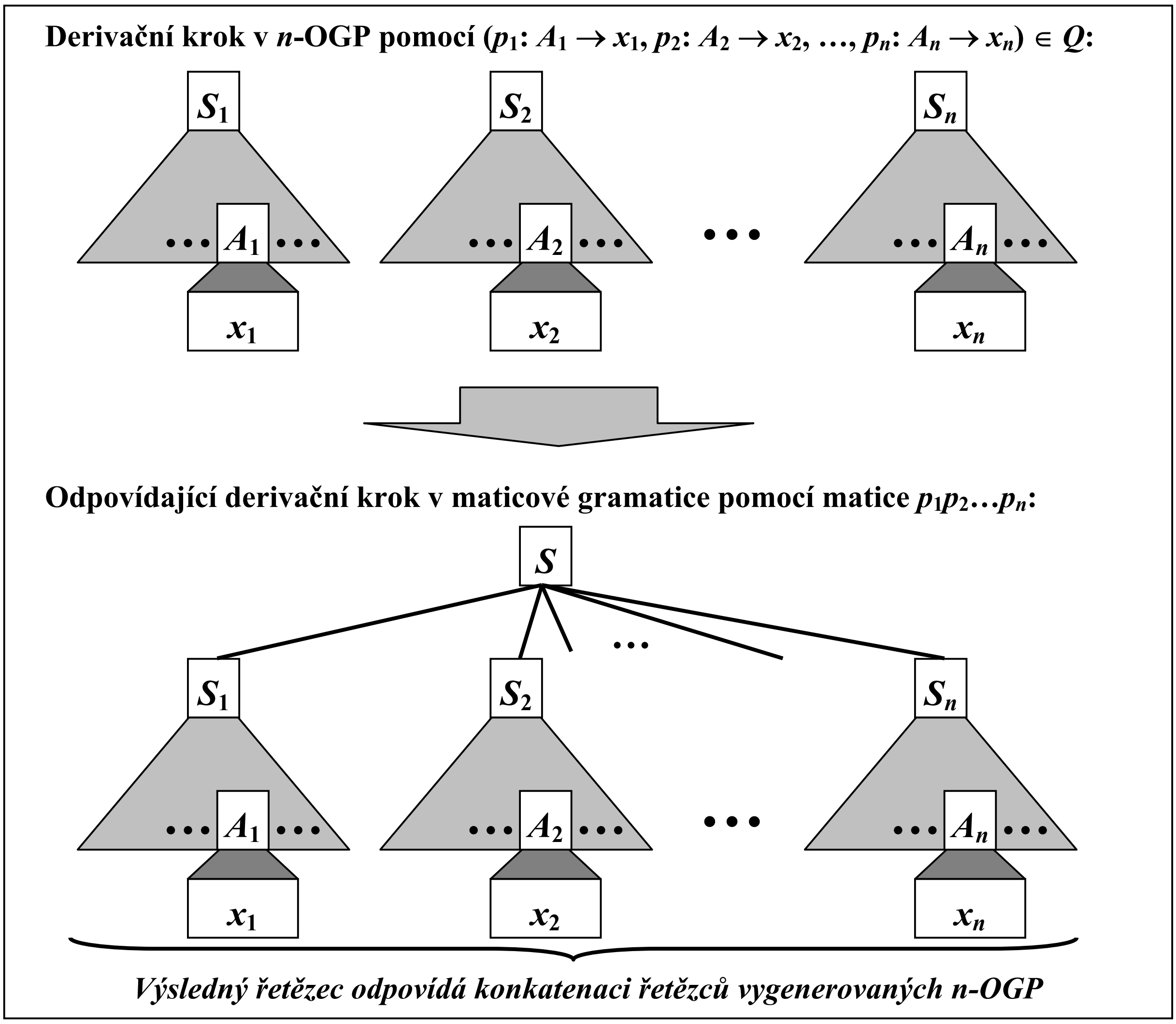
4.3 Vztah mezi *n*-OGN a *n*-OGP a jejich generativní síla

Tvrzení 4.3.1: Třída jazyků generovaná pomocí n-OGN v módu X, kde $X \in \{union, conc, first\}$, je ekvivalentní s třídou jazyků generovanou pomocí n-OGP v módu X.

Důkaz: Algoritmus 3.3.2.1 a algoritmus 3.3.3.1 lze opět použít na vzájemné převody obou typů obecných n-generativních gramatických systémů. Tvrzení 3.3.2.3 a tvrzení 3.3.3.3, které dokazují správnost těchto algoritmů, by se dokázala naprosto identicky s rozdílem, že jakýkoliv výskyt nejlevější derivace v daném důkazu by byl nahrazen obecnou derivací.

4.3.1 *Převod n-OGP definující jazyk v módu konkatenace na ekvivalentní maticovou gramatiku*

Následující algoritmus popíše, jak lze převést libovolný n-OGP definující jazyk v módu konkatenace na ekvivalentní maticovou gramatiku. Neformální popis myšlenky algoritmu je následující: Uvažujme n-OGP $\Gamma = (G_1, G_2, \ldots, G_n, Q)$, přičemž $G_i = (N_i, T_i, P_i, S_i)$ pro všechna $i = 1, \ldots, n$. Dále předpokládejme, že všechny tyto gramatiky mají vzájemně disjunktní množiny nonterminálních symbolů. Není-li tomu tak, můžeme toho dosáhnout například indexací nonterminálních symbolů. Maticová gramatika bude potom obsahovat pravidla všech gramatik, které jsou součástí n-OGP a speciální startující pravidlo $S \rightarrow S_1S_2\ldots S_n$. Matice se vytvoří pouhým zřetězením n-tic pravidel z komponenty Q. Zřejmě tedy bude muset být aplikována tato matice postupně tak, že první její pravidlo bude aplikováno na část vygenerovanou z nonterminálu S_1 druhé pravidlo na část vygenerovanou z S_2 atd. Tím se právě odsimuluje jeden derivační krok v původním n-OGP. Celou situaci ilustruje obr. 4.1. Správnost této myšlenky je pak formálně dokázána ve tvrzení 4.3.1.3.



(obr. 4.1: simulace n-OGP v módu konkatenace pomocí maticové gramatiky)

4.3.1.1 Algoritmus: Převod n-OGP definující jazyk v módu konkatenace na ekvivalentní maticovou gramatiku

- ***Vstup:*** n-OGP $\Gamma = (G_1, G_2, \ldots, G_n, Q)$
- ***Výstup:*** maticová gramatika $H = (G, M)$; $L_{conc}(\Gamma) = L(H)$
- ***Metoda:***
    - Nechť $G_i = (N_i, T_i, P_i, S_i)$ pro všechna $i = 1, \ldots, n$, přičemž předpokládejme, že pro libovolné indexy $j, k = 1,\ldots, n$, kde $j \neq k$ platí: $N_j \cap N_k = \emptyset$; $S \notin N_j$. Není-li tomu tak, můžeme této podmínky dosáhnout přeznačením nonterminálních symbolů.
    - Potom položme:
    - $G = (N, T, P, S)$, kde:
        - $N := \{S\} \cup (\bigcup_{i=1}^{n} N_i)$;
        - $T := \bigcup_{i=1}^{n} T_i$;
        - $P := \{s\colon S \to S_1S_2\ldots S_n\} \cup (\bigcup_{i=1}^{n} P_i)$;
    - $M = \{s\} \cup \{p_1p_2\ldots p_n\colon (p_1, p_2, \ldots, p_n) \in Q\}$;

4.3.1.2 Příklad

Uvažujme následující 2-OGP $\Gamma = (G_1, G_2, Q)$ generující jazyk v módu konkatenace, kde:

- $G_1 = (\{S_1, A_1\}, \{a, b, c\}, \{\mathbf{1}\colon S_1 \to aS_1, \mathbf{2}\colon S_1 \to aA_1, \mathbf{3}\colon A_1 \to bA_1c, \mathbf{4}\colon A_1 \to bc\}, S_1)$
- $G_2 = (\{S_2\}, \{d\}, \{\mathbf{A}\colon S_2 \to S_2S_2, \mathbf{B}\colon S_2 \to S_2, \mathbf{C}\colon S_2 \to d\}, S_2)$
- $Q = \{(\mathbf{1}, \mathbf{A}), (\mathbf{2}, \mathbf{B}), (\mathbf{3}, \mathbf{C}), (\mathbf{4}, \mathbf{C})\}$

Ekvivalentní maticová gramatika sestavená podle předchozího algoritmu bude mít strukturu $H = (G, M)$, kde:

- $G = (\{S, S_1, A_1, S_2\}, \{a, b, c, d\},$
  $\{\ \mathbf{0}\colon S \to S_1S_2,$
  $\mathbf{1}\colon S_1 \to aS_1, \mathbf{2}\colon S_1 \to aA_1, \mathbf{3}\colon A_1 \to bA_1c, \mathbf{4}\colon A_1 \to bc,$
  $\mathbf{A}\colon S_2 \to S_2S_2, \mathbf{B}\colon S_2 \to S_2, \mathbf{C}\colon S_2 \to d\}, S),$
- $M = \{\mathbf{0}, \mathbf{1A}, \mathbf{2B}, \mathbf{3C}, \mathbf{4C}\}$.

4.3.1.3 Tvrzení

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je libovolný n-OGP. S n-OGP Γ na vstupu algoritmus 4.3.1.1 zastaví a vytvoří maticovou gramatiku $H = (G, M)$, pro kterou platí: $L_{conc}(\Gamma) = L(H)$.

***Důkaz*:**

***Tvrzení A*:** Nechť existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^m (y_1, y_2, \ldots, y_n)$ v n-OGP Γ, kde $m \geq 0$, $y_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \ldots, n$. Potom také existuje sekvence derivačních kroků $S \Rightarrow^{m+1} y_1y_2\ldots y_n$ v maticové gramatice H.

***Důkaz*:** Tvrzení je dokázáno matematickou indukcí

Základní krok: Nechť $m = 0$. Potom $(S_1, S_2, \ldots, S_n) \Rightarrow^0 (S_1, S_2, \ldots, S_n)$ v n-OGP Γ. Poznamenejme, že také $S \Rightarrow^1 S_1S_2\ldots S_n$, v maticové gramatice H, neboť s: $S \rightarrow S_1S_2\ldots S_n \in M$.

Indukční hypotéza: Předpokládejme, že tvrzení A platí pro všechny m-kroké derivace, kde $m = 0,\ldots, k$ pro nějaké $k \geq 0$.

Indukční krok: Uvažujme sekvenci derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{k+1} (y_1, y_2, \ldots, y_n)$ v n-OGP Γ. Potom zřejmě existuje multiforma $(u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n)$, kde $A_i \in N_i$, u_i, $v_i \in (N_i \cup T_i)^*$, pro kterou platí $(S_1, S_2, \ldots, S_n) \Rightarrow^k (u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$ v n-OGP Γ, kde $u_ix_iv_i = y_i$ pro všechna $i = 1, \ldots, n$.

I. Ze sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^k (u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n)$ v n-OGP Γ plyne, že existuje sekvence derivačních kroků $S \Rightarrow^{k+1} u_1A_1v_1u_2A_2v_2\ldots u_nA_nv_n$ v maticové gramatice H podle indukční hypotézy.

II. Nechť existuje derivační krok $(u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$ v n-OGP Γ. Potom zřejmě pro n-OGP Γ platí $(p_1\colon A_1 \rightarrow x_1, p_2\colon A_2 \rightarrow x_2, \ldots, p_n\colon A_n \rightarrow x_n) \in Q$. Z algoritmu 4.3.1.1 potom plyne, že $p_1p_2\ldots p_n \in M$. Tedy v maticové gramatice H existuje derivační krok $u_1A_1v_1u_2A_2v_2\ldots u_nA_nv_n \Rightarrow u_1x_1v_1u_2x_2v_2\ldots u_nx_nv_n$ $[p_1p_2\ldots p_n]$

Z částí I. a II. plyne, že $S \Rightarrow^{k+1} u_1A_1v_1u_2A_2v_2\ldots u_nA_nv_n \Rightarrow u_1x_1v_1u_2x_2v_2\ldots u_nx_nv_n$ v maticové gramatice H. Potom tedy v této maticové gramatice i existuje sekvence derivačních kroků $S \Rightarrow^{k+2} u_1x_1v_1u_2x_2v_2\ldots u_nx_nv_n$.

***Tvrzení B*:** Nechť existuje sekvence derivačních kroků $S \Rightarrow^m y$ v maticové gramatice H, kde $m \geq 1$, $y \in (N \cup T)^*$. Potom existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{m-1} (y_1, y_2, \ldots, y_n)$ v n-OGP Γ, kde $y_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \ldots, n$, přičemž platí: $y = y_1y_2\ldots y_n$.

***Důkaz*:** Tvrzení je dokázáno matematickou indukcí

Základní krok: Nechť $m = 1$. V maticové gramatice H zřejmě existuje pouze jednokroká derivace tvaru $S \Rightarrow^1 S_1S_2\ldots S_n$, neboť s: $S \rightarrow S_1S_2\ldots S_n \in M$ je jediné pravidlo s nonterminálním symbolem S na levé straně. $(S_1, S_2, \ldots, S_n) \Rightarrow^0 (S_1, S_2, \ldots, S_n)$ v n-OGP Γ platí triviálně.

Indukční hypotéza: Předpokládejme, že tvrzení B platí pro všechny m-kroké derivace, kde $m = 0,\ldots, k$ pro nějaké $k \geq 1$.

Indukční krok: Uvažujme sekvenci derivačních kroků $S \Rightarrow^{k+1} y$ v maticové gramatice H. Potom zřejmě existuje větná forma w, pro kterou platí: $S \Rightarrow^k w \Rightarrow y$, kde $w, y \in (N \cup T)^*$

I. Ze sekvence derivačních kroků $S \Rightarrow^k w$ v maticové gramatice H plyne, že existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{k-1} (w_1, w_2, \ldots, w_n)$ v n-OGP Γ, přičemž $w = w_1w_2\ldots w_n$ podle indukční hypotézy.

II. Nechť existuje derivační krok $w \Rightarrow y$ v maticové gramatice H, podle matice $p_1p_2\ldots p_n \in M$, přičemž $w = w_1w_2\ldots w_n$, kde $w_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \ldots, n$. Uvažujme pravidlo

p_i ve tvaru $A_i \rightarrow x_i$ pro všechna $i = 1, \ldots, n$. Neboť podle předpokladu pro libovolné indexy j, $k = 1, \ldots, n$, kde $j \neq k$ platí: $N_j \cap N_k = \emptyset$, musí být pravidlo $p_i = A_i \rightarrow x_i$ aplikováno právě v části větné formy označené w_i. Pro všechna $i = 1, \ldots, n$ můžeme tedy část větné formy w_i předpokládat ve tvaru: $w_i = u_iA_iv_i$, kde $A_i \in N_i$, $u_i, v_i \in (N_i \cup T_i)^*$. V maticové gramtice H tedy existuje derivační krok tvaru: $u_1A_1v_1u_2A_2v_2\ldots u_nA_nv_n \Rightarrow u_1x_1v_1u_2x_2v_2\ldots u_nx_nv_n$ podle matice $p_1p_2\ldots p_n \in M$. Z algoritmu 4.3.1.1 plyne, že $(p_1: A_1 \rightarrow x_1, p_2: A_2 \rightarrow x_2, \ldots, p_n: A_n \rightarrow x_n) \in Q$, neboť $p_1p_2\ldots p_n \in M$. V n-OGP Γ tedy existuje derivační krok tvaru: $(u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$.

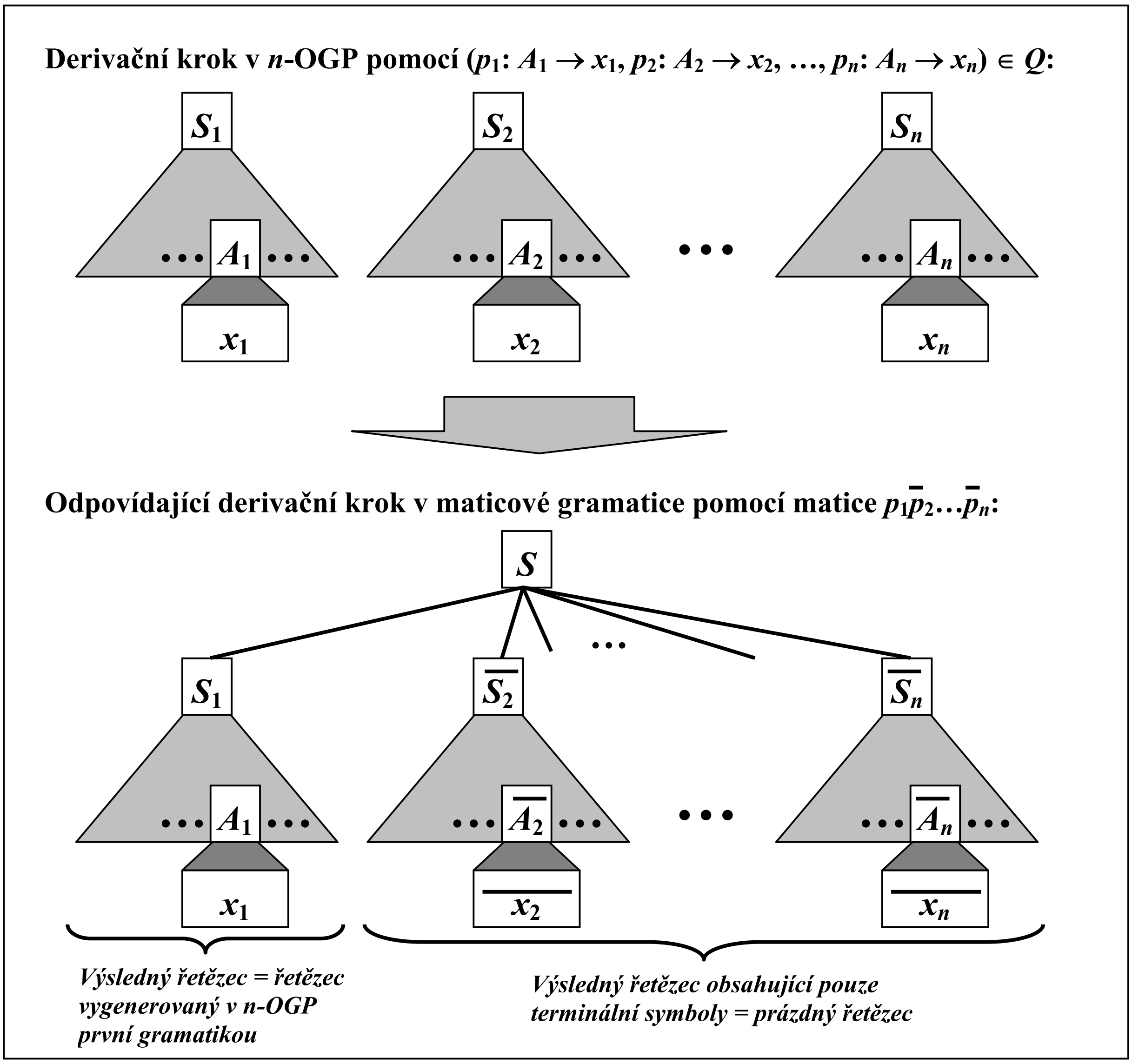
Z částí I. a II. plyne, že $(S_1, S_2, \ldots, S_n) \Rightarrow^{k-1} (u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$. Potom tedy v tomto n-OGP Γ existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^k (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$, přičemž $y = u_1x_1v_1u_2x_2v_2\ldots u_nx_nv_n$.

Uvažujme tvrzení A pro nějaké $m \geq 0$ a $y_i \in T_i^*$ pro všechna $i = 1, \ldots, n$. Dostaneme implikaci tvaru: Pokud existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^m (y_1, y_2, \ldots, y_n)$ v n-OGP Γ, potom také existuje sekvence derivačních kroků $S \Rightarrow^{m+1} y_1y_2\ldots y_n$ v maticové gramatice H. Jelikož $L_{conc}(\Gamma) = \{w_1w_2\ldots w_n: (w_1, w_2, \ldots, w_n) \in n\text{-}L(\Gamma)\}$, zřejmě platí také implikace: Jestliže $y_1y_2\ldots y_n \in L_{conc}(\Gamma)$, potom $y_1y_2\ldots y_n \in L(H)$, což znamená $L_{conc}(\Gamma) \subseteq L(H)$. Dále uvažujme tvrzení B pro nějaké $m \geq 1$ a $y \in T^*$. Dostaneme implikaci tvaru: Pokud existuje sekvence derivačních kroků $S \Rightarrow^m y$ v maticové gramatice H, potom existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{m-1} (y_1, y_2, \ldots, y_n)$ v n-OGP Γ, přičemž platí: $y = y_1y_2\ldots y_n$. Z definice $L_{conc}(\Gamma)$ zřejmě platí implikace: Jestliže $y_1y_2\ldots y_n \in L(H)$, potom $y_1y_2\ldots y_n \in L_{conc}(\Gamma)$, což znamená $L(H) \subseteq L_{conc}(\Gamma)$. Z platnosti vztahů $L_{conc}(\Gamma) \subseteq L(H)$ a $L(H) \subseteq L_{conc}(\Gamma)$ ihned plyne rovnost $L_{conc}(\Gamma) = L(H)$.

4.3.2 Převod n-OGP definující jazyk v módu první komponenty na ekvivalentní maticovou gramatiku

Následující algoritmus popíše, jak lze převést libovolný n-OGP definující jazyk v módu první komponenty na ekvivalentní maticovou gramatiku. Neformální popis myšlenky algoritmu je následující: Konstrukce maticové gramatiky je stejná jako v předchozím algoritmu s následující modifikací. Nonterminální symboly ze všech gramatik kromě první gramatiky jsou jistým způsobem poznačeny. Uvedený algoritmus je například poznačí pomocí svislého pruhu nad daným symbolem. Toto poznačení se projeví i v rámci pravidel. „Startující pravidlo“ maticové gramatiky je ve tvaru: $S \rightarrow S_1\overline{S}_2...\overline{S}_n$. Poslední modifikace je následující: Pravidla, která byla do maticové gramatiky přidána z G_2 až G_n, jsou modifikována tak, že nonterminální symboly levé i pravé strany pravidla jsou nahrazeny poznačenými nonterminálními symboly, terminální symboly z pravé strany pravidla jsou nahrazeny prázdnými řetězci. Tím se docílí toho, že simulace probíhá přesně jako v případě módu konkatenace s rozdílem, že v části větné formy maticové gramatiky odpovídající v n-OGP větným formám vygenerovaných gramatikami G_2 až G_n jsou terminální symboly nahrazeny prázdným řetězcem. Výsledný vygenerovaný řetězec je tedy roven řetězci, který by v n-OGP vygenerovala pouze gramatika G_1. Celou situaci ilustruje obr. 4.2. Správnost této myšlenky je pak formálně dokázána ve tvrzení 4.3.2.3.

Poznámka: Označování nonterminálních symbolů nemusí být nutně provedeno, v algoritmu však uskutečněno je jednak z didaktických důvodů, protože algoritmus se stává názornější, dále pak z důvodu, že na zobecněném tomto principu bude založen algoritmus 4.3.3.1.



(obr. 4.2: simulace n-OGP v módu první komponenty pomocí maticové gramatiky)

4.3.2.1 Algoritmus: Převod n-OGP definující jazyk v módu první komponenty na ekvivalentní maticovou gramatiku

- ***Vstup:*** n-OGP $\Gamma = (G_1, G_2, \ldots, G_n, Q)$
- ***Výstup:*** maticová gramatika $H = (G, M)$; $L_{first}(\Gamma) = L(H)$
- ***Metoda:***
  - Nechť $G_i = (N_i, T_i, P_i, S_i)$ pro všechna $i = 1, \ldots, n$, přičemž předpokládejme, že pro libovolné indexy j, $k = 1,\ldots, n$, kde $j \neq k$ platí: $N_j \cap N_k = \emptyset$; $S \notin N_j$. Není-li tomu tak, můžeme této podmínky dosáhnout přeznačením nonterminálních symbolů.

- Potom položme:
- $G = (N, T, P, S)$, kde:
  - $N := \{S\} \cup N_1 \cup (\bigcup_{i=2}^{n} \{\overline{A} : A \in N_i\})$;
  - $T := T_1$;
  - $P := \{\, s\colon S \to S_1 h(S_2) \dots h(S_n) \,\} \cup P_1 \cup (\bigcup_{i=2}^{n} \{h(A) \to h(x) : A \to x \in P_i\})$,

    kde h je homomorfismus z $(\bigcup_{i=2}^{n} N_i) \cup (\bigcup_{i=2}^{n} T_i)$ do $\bigcup_{i=2}^{n} \{\overline{A} : A \in N_i\}$ definovaný následovně: $h(a) = \varepsilon$ pro všechna $a \in \bigcup_{i=2}^{n} T_i$; $h(A) = \overline{A}$ pro všechna $A \in \bigcup_{i=2}^{n} N_i$;

    Konvence: Nechť $p = A \to x$ je pravidlo. Potom symbolem $\overline{p}$ je označeno pravidlo $h(A) \to h(x)$
- $M = \{s\} \cup \{\, p_1\overline{p}_2 ... \overline{p}_n : (p_1, p_2, \dots, p_n) \in Q\}$

4.3.2.2 Příklad

Uvažujme následující 2-OGP $\Gamma = (G_1, G_2, Q)$ generující jazyk v módu první komponenty, kde:

- $G_1 = (\{S_1, A_1\}, \{a, b, c\}, \{\mathbf{1}\colon S_1 \to aS_1, \mathbf{2}\colon S_1 \to aA_1, \mathbf{3}\colon A_1 \to bA_1c, \mathbf{4}\colon A_1 \to bc\}, S_1)$
- $G_2 = (\{S_2\}, \{d\}, \{\mathbf{A}\colon S_2 \to S_2S_2, \mathbf{B}\colon S_2 \to S_2, \mathbf{C}\colon S_2 \to d\}, S_2)$
- $Q = \{(\mathbf{1}, \mathbf{A}), (\mathbf{2}, \mathbf{B}), (\mathbf{3}, \mathbf{C}), (\mathbf{4}, \mathbf{C})\}$

Ekvivalentní maticová gramatika sestavená podle předchozího algoritmu bude mít strukturu $H = (G, M)$, kde:

- $G = (\{S, S_1, A_1, \overline{S}_2\}, \{a, b, c\},$

  $\{\, \mathbf{0}\colon S \to S_1\overline{S}_2,$

  $\mathbf{1}\colon S_1 \to aS_1, \mathbf{2}\colon S_1 \to aA_1, \mathbf{3}\colon A_1 \to bA_1c, \mathbf{4}\colon A_1 \to bc,$

  $\overline{\mathbf{A}}\colon \overline{S}_2 \to \overline{S}_2\overline{S}_2,\ \overline{\mathbf{B}}\colon \overline{S}_2 \to \overline{S}_2,\ \overline{\mathbf{C}}\colon \overline{S}_2 \to \varepsilon\,\}, S)$
- $M = \{\mathbf{0}, \mathbf{1}\overline{\mathbf{A}}, \mathbf{2}\overline{\mathbf{B}}, \mathbf{3}\overline{\mathbf{C}}, \mathbf{4}\overline{\mathbf{C}}\}$

4.3.2.3 Tvrzení

Nechť $\Gamma = (G_1, G_2, \dots, G_n, Q)$ je libovolný n-OGP. S n-OGP Γ na vstupu algoritmus 4.3.2.1 zastaví a vytvoří maticovou gramatiku $H = (G, M)$, pro kterou platí: $L_{first}(\Gamma) = L(H)$.

***Důkaz*:**

***Tvrzení A*:** Nechť existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^m (y_1, y_2, \ldots, y_n)$ v n-OGP Γ, kde $m \geq 0$, $y_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \ldots, n$. Potom také existuje sekvence derivačních kroků $S \Rightarrow^{m+1} y_1h(y_2)\ldots h(y_n)$ v maticové gramatice H.

***Důkaz*:** Tvrzení je dokázáno matematickou indukcí

Základní krok: Nechť $m = 0$. Potom $(S_1, S_2, \ldots, S_n) \Rightarrow^0 (S_1, S_2, \ldots, S_n)$ v n-OGP Γ. Poznamenejme, že také $S \Rightarrow^1 S_1h(S_2)\ldots h(S_n)$ v maticové gramtice H, neboť s: $S \rightarrow S_1h(S_2)\ldots h(S_n) \in M$.

Indukční hypotéza: Předpokládejme, že tvrzení A platí pro všechny m-kroké derivace, kde $m = 0,\ldots, k$ pro nějaké $k \geq 0$.

Indukční krok: Uvažujme sekvenci derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{k+1} (y_1, y_2, \ldots, y_n)$ v n-OGP Γ. Potom zřejmě existuje multiforma $(u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n)$, kde $A_i \in N_i$, u_i, $v_i \in (N_i \cup T_i)^*$, pro kterou platí $(S_1, S_2, \ldots, S_n) \Rightarrow^k (u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$ v n-OGP Γ, kde $u_ix_iv_i = y_i$ pro všechna $i = 1, \ldots, n$.

I. Ze sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^k (u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n)$ v n-OGP Γ plyne, že existuje sekvence derivačních kroků $S \Rightarrow^{k+1} u_1A_1v_1h(u_2A_2v_2)\ldots h(u_nA_nv_n)$ v maticové gramatice H podle indukční hypotézy.

II. Nechť existuje derivační krok $(u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$ v n-OGP Γ. Potom zřejmě pro n-OGP Γ platí $(p_1: A_1 \rightarrow x_1, p_2: A_2 \rightarrow x_2, \ldots, p_n: A_n \rightarrow x_n) \in Q$. Z algoritmu 4.3.2.1 potom plyne, že $p_1\overline{p}_2...\overline{p}_n \in M$. Tedy v maticové gramatice H existuje derivační krok $u_1A_1v_1h(u_2A_2v_2)\ldots h(u_nA_nv_n) \Rightarrow u_1x_1v_1h(u_2x_2v_2)\ldots h(u_nx_nv_n)$ podle matice $p_1\overline{p}_2...\overline{p}_n$.

Z částí I. a II. plyne, že $S \Rightarrow^{k+1} u_1A_1v_1h(u_2A_2v_2)\ldots h(u_nA_nv_n) \Rightarrow u_1x_1v_1h(u_2x_2v_2)\ldots h(u_nx_nv_n)$ v maticové gramatice H. Potom tedy v této maticové gramatice H i existuje sekvence derivačních kroků $S \Rightarrow^{k+2} u_1x_1v_1h(u_2x_2v_2)\ldots h(u_nx_nv_n)$.

***Tvrzení B*:** Nechť existuje sekvence derivačních kroků $S \Rightarrow^m y$ v maticové gramatice H, kde $m \geq 1$, $y \in (N \cup T)^*$. Potom existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{m-1} (y_1, y_2, \ldots, y_n)$ v n-OGP Γ, kde $y_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \ldots, n$, přičemž platí: $y = y_1h(y_2)\ldots h(y_n)$.

***Důkaz*:** Tvrzení je dokázáno matematickou indukcí

Základní krok: Nechť $m = 1$. V maticové gramatice H zřejmě existuje jednokroká derivace tvaru: $S \Rightarrow^1 S_1h(S_2)\ldots h(S_n)$ podle matice s: $S \rightarrow S_1h(S_2)\ldots h(S_n)$. $(S_1, S_2, \ldots, S_n) \Rightarrow^0 (S_1, S_2, \ldots, S_n)$ v n-OGP Γ platí triviálně.

Indukční hypotéza: Předpokládejme, že tvrzení B platí pro všechny m-kroké derivace, kde $m = 0,\ldots, k$ pro nějaké $k \geq 1$.

Indukční krok: Uvažujme sekvenci derivačních kroků $S \Rightarrow^{k+1} y$ v maticové gramatice H. Potom zřejmě existuje větná forma w, pro kterou platí: $S \Rightarrow^k w \Rightarrow y$, kde $w, y \in (N \cup T)^*$

I. Ze sekvence derivačních kroků $S \Rightarrow^k w$ v maticové gramatice H plyne, že existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{k-1} (w_1, w_2, \ldots, w_n)$ v n-OGP Γ, přičemž platí: $w = w_1h(w_2)\ldots h(w_n)$ podle indukční hypotézy.

II. Nechť existuje derivační krok $w \Rightarrow y$ v maticové gramatice H, přičemž $w = w_1h(w_2)\dots h(w_n)$, kde $w_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \dots, n$. Uvažujme pravidlo p_1 ve tvaru $A_1 \to x_1$. Dále uvažujme pro všechna $i = 2, \dots, n$ pravidlo $\overline{p}_i$ ve tvaru $h(A_i) \to h(x_i)$. Neboť podle předpokladu pro libovolné indexy k, $l = 1,\dots, n$, kde $k \neq l$ platí $N_k \cap N_l = \emptyset$, musí být pravidlo p_1: $A_1 \to x_1$ aplikováno právě na část větné formy označené jako w_1 a ostatní pravidla $\overline{p}_i : h(A_i) \to h(x_i)$ na části větné formy označené jako $h(w_i)$. Pro všechna $i = 1, \dots, n$ můžeme tedy část větné formy w_i předpokládat ve tvaru: $w_i = u_iA_iv_i$, kde $A_i \in N_i$, u_i, $v_i \in (N_i \cup T_i)^*$. V maticové gramatice H tedy existuje derivační krok tvaru: $u_1A_1v_1h(u_2A_2v_2)\dots(u_nA_nv_n) \Rightarrow u_1x_1v_1h(u_2x_2v_2)\dots h(u_nx_nv_n)$ podle matice $p_1\overline{p}_2...\overline{p}_n \in M$. Z algoritmu 4.3.2.1 plyne, že (p_1: $A_1 \to x_1$, p_2: $A_2 \to x_2$, …, p_n: $A_n \to x_n$) $\in Q$, neboť $p_1\overline{p}_2...\overline{p}_n \in M$. V n-OGP Γ tedy existuje derivační krok tvaru: $(u_1A_1v_1, u_2A_2v_2, \dots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \dots, u_nx_nv_n)$.

Z částí I. a II. plyne, že $(S_1, S_2, \dots, S_n) \Rightarrow^{k-1} (u_1A_1v_1, u_2A_2v_2, \dots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \dots, u_nx_nv_n)$. Potom tedy v tomto n-OGP Γ existuje sekvence derivačních kroků $(S_1, S_2, \dots, S_n) \Rightarrow^{k} (u_1x_1v_1, u_2x_2v_2, \dots, u_nx_nv_n)$, přičemž platí: $y = u_1x_1v_1h(u_2x_2v_2)\dots h(u_nx_nv_n)$.

Uvažujme tvrzení A pro nějaké $m \geq 0$ a $y_i \in T_i^*$ pro všechna $i = 1, \dots, n$. Dostaneme implikaci tvaru: Pokud existuje sekvence derivačních kroků $(S_1, S_2, \dots, S_n) \Rightarrow^{m} (y_1, y_2, \dots, y_n)$ v n-OGP Γ, potom také existuje sekvence derivačních kroků $S \Rightarrow^{m+1} y_1h(y_2)\dots h(y_n)$ v maticové gramatice H. Neboť $h(a) = \varepsilon$ pro všechna $a \in \bigcup_{i=2}^{n} T_i$, můžeme danou sekvenci derivačních kroků psát jako: $S \Rightarrow^{m+1} y_1$. Jelikož $L_{first}(\Gamma) = \{w_1: (w_1, w_2, \dots, w_n) \in n\text{-}L(\Gamma)\}$, zřejmě platí také implikace: Pokud $y \in L_{first}(\Gamma)$, potom $y \in L(H)$, což znamená $L_{first}(\Gamma) \subseteq L(H)$. Dále uvažujme tvrzení B pro nějaké $m \geq 1$ a $y \in T^*$. Dostaneme implikaci tvaru: Pokud existuje sekvence derivačních kroků $S \Rightarrow^{m} y$ v maticové gramatice H, potom existuje sekvence derivačních kroků $(S_1, S_2, \dots, S_n) \Rightarrow^{m-1} (y_1, y_2, \dots, y_n)$ v n-OGP Γ, kde $y_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \dots, n$, přičemž platí: $y = y_1h(y_2)\dots h(y_n)$. Neboť $h(a) = \varepsilon$ pro všechna $a \in \bigcup_{i=2}^{n} T_i$, platí tedy $y = y_1$. Z definice $L_{first}(\Gamma)$ zřejmě platí implikace: Pokud $y \in L(H)$ potom $y \in L_{first}(\Gamma)$, což znamená $L(H) \subseteq L_{first}(\Gamma)$. Z platnosti vztahů $L_{first}(\Gamma) \subseteq L(H)$ a $L(H) \subseteq L_{first}(\Gamma)$ ihned plyne rovnost $L_{first}(\Gamma) = L(H)$.

4.3.3 Převod n-OGP definující jazyk v módu sjednocení na ekvivalentní maticovou gramatiku

Následující algoritmus popíše, jak lze převést libovolný n-OGP definující jazyk v módu sjednocení na ekvivalentní maticovou gramatiku. Neformální popis myšlenky algoritmu je následující: Konstrukce maticové gramatiky je zobecněním předchozího algoritmu takovým způsobem, že maticová gramatika bude obsahovat jak všechny nonterminální symboly použité v n-OGP, tak jejich označené verze. Z hlediska pravidel bude obsahovat všechna pravidla použité v n-OGP a pro každé toto pravidlo i jeho modifikovanou verzi, ve které nonterminální symboly levé i pravé strany pravidla budou nahrazeny poznačenými symboly a terminální symboly z pravé strany pravidla budou nahrazeny prázdnými řetězci. Startující pravidla budou: $S \to S_1\overline{S}_2...\overline{S}_n$, $S \to \overline{S}_1S_2...\overline{S}_n$,…, $S \to \overline{S}_1\overline{S}_2...S_n$. Matice pravidel budou konstruovány tak, že každá matice obsahuje právě jedno původní pravidlo a n–1 modifikovaných pravidel. Pokud je tedy vybráno na počátku pravidlo $S \rightarrow S_1\overline{S}_2...\overline{S}_n$, výsledný vygenerovaný řetězec je ten, který by v n-OGP vygenerovala gramatika G_1. Pokud je vybráno pravidlo $S \rightarrow \overline{S}_1S_2...\overline{S}_n$, výsledný vygenerovaný řetězec je ten, který by v n-OGP vygenerovala gramatika G_2 atd. Správnost této myšlenky je pak formálně dokázána ve tvrzení 4.3.3.3.

4.3.3.1 Algoritmus Převod n-OGP definující jazyk v módu sjednocení na ekvivalentní maticovou gramatiku

- ***Vstup:*** n-OGP $\Gamma = (G_1, G_2, \ldots, G_n, Q)$
- ***Výstup:*** maticová gramatika $H = (G, M)$; $L_{union}(\Gamma) = L(H)$
- ***Metoda:***
  - Nechť $G_i = (N_i, T_i, P_i, S_i)$ pro všechna $i = 1, \ldots, n$, přičemž předpokládejme, že pro libovolné indexy j, $k = 1,\ldots, n$, kde $j \neq k$ platí: $N_j \cap N_k = \emptyset$; $S \notin N_j$. Není-li tomu tak, můžeme této podmínky dosáhnout přeznačením nonterminálních symbolů.
  - Potom položme:
  - $G = (N, T, P, S)$, kde:
    - $N := \{S\} \cup (\bigcup_{i=1}^{n} N_i) \cup (\bigcup_{i=1}^{n} \{\overline{A} : A \in N_i\})$;
    - $T := \bigcup_{i=1}^{n} T_i$;
    - $P := \{\ s_1\colon S \rightarrow S_1h(S_2)\ldots h(S_n),\ s_2\colon S \rightarrow h(S_1)S_2\ldots h(S_n),\ \ldots$
      $s_n\colon S \rightarrow h(S_1)h(S_2)\ldots S_n\ \} \cup$
      $(\bigcup_{i=1}^{n} P_i) \cup (\bigcup_{i=1}^{n} \{h(A) \rightarrow h(x) : A \rightarrow x \in P_i\})$,

      kde h je homomorfismus z $(\bigcup_{i=1}^{n} N_i) \cup (\bigcup_{i=1}^{n} T_i)$ do $\bigcup_{i=1}^{n} \{\overline{A} : A \in N_i\}$ definovaný následovně: $h(a) = \varepsilon$ pro všechna $a \in \bigcup_{i=1}^{n} T_i$; $h(A) = \overline{A}$ pro všechna $A \in \bigcup_{i=1}^{n} N_i$.

      Konvence: Nechť $p = A \rightarrow x$ je pravidlo. Potom symbolem $\overline{p}$ je označeno pravidlo $h(A) \rightarrow h(x)$:
  - $M = \{s_1, s_2, \ldots, s_n\} \cup \{\ p_1\overline{p}_2...\overline{p}_n : (p_1, p_2, \ldots, p_n) \in Q\}$
    $\cup \{\ \overline{p}_1p_2...\overline{p}_n : (p_1, p_2, \ldots, p_n) \in Q\}$
    $\cup \ldots$
    $\cup \{\ \overline{p}_1\overline{p}_2...p_n : (p_1, p_2, \ldots, p_n) \in Q\}$;

4.3.3.2 Příklad

Uvažujme následující 2-OGP $\Gamma = (G_1, G_2, Q)$ generující jazyk v módu sjednocení, kde:

- $G_1 = (\{S_1, A_1\}, \{a, b, c\}, \{\mathbf{1}: S_1 \to aS_1, \mathbf{2}: S_1 \to aA_1, \mathbf{3}: A_1 \to bA_1c, \mathbf{4}: A_1 \to bc\}, S_1)$
- $G_2 = (\{S_2\}, \{d\}, \{\mathbf{A}: S_2 \to S_2S_2, \mathbf{B}: S_2 \to S_2, \mathbf{C}: S_2 \to d\}, S_2)$
- $Q = \{(\mathbf{1}, \mathbf{A}), (\mathbf{2}, \mathbf{B}), (\mathbf{3}, \mathbf{C}), (\mathbf{4}, \mathbf{C})\}$

Ekvivalentní maticová gramatika sestavená podle předchozího algoritmu bude mít strukturu $H = (G, M)$, kde:

- $G = (\{S, S_1, A_1, S_2, \overline{S}_1, \overline{A}_1, \overline{S}_2\}, \{a, b, c, d\},$

  $\{\ \mathbf{0}: S \to S_1\overline{S}_2,\ \mathbf{Z}: S \to \overline{S}_1S_2,$

  $\mathbf{1}: S_1 \to aS_1,\ \overline{\mathbf{1}}: \overline{S}_1 \to \overline{S}_1,\ \mathbf{2}: S_1 \to aA_1,\ \overline{\mathbf{2}}: \overline{S}_1 \to \overline{A}_1,$

  $\mathbf{3}: A_1 \to bA_1c,\ \overline{\mathbf{3}}: \overline{A}_1 \to \overline{A}_1,\ \mathbf{4}: A_1 \to bc,\ \overline{\mathbf{4}}: \overline{A}_1 \to \varepsilon,$

  $\mathbf{A}: S_2 \to S_2S_2,\ \overline{\mathbf{A}}: \overline{S}_2 \to \overline{S}_2\overline{S}_2,\ \mathbf{B}: S_2 \to S_2,\ \overline{\mathbf{B}}: \overline{S}_2 \to \overline{S}_2,$

  $\mathbf{C}: S_2 \to d,\ \overline{\mathbf{C}}: \overline{S}_2 \to \varepsilon\ \},\ S)$
- $M = \{\mathbf{0}, \mathbf{Z}, \mathbf{1}\overline{\mathbf{A}}, \overline{\mathbf{1}}\mathbf{A}, \mathbf{2}\overline{\mathbf{B}}, \overline{\mathbf{2}}\mathbf{B}, \mathbf{3}\overline{\mathbf{C}}, \overline{\mathbf{3}}\mathbf{C}, \mathbf{4}\overline{\mathbf{C}}, \overline{\mathbf{4}}\mathbf{C}\}$

4.3.3.3 Tvrzení

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je libovolný n-OGP. S n-OGP Γ na vstupu algoritmus 4.3.3.1 zastaví a vytvoří maticovou gramatiku $H = (G, M)$, pro kterou platí: $L_{union}(\Gamma) = L(H)$.

***Důkaz*:**

***Tvrzení A*:** Nechť existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^m (y_1, y_2, \ldots, y_n)$ v n-OGP Γ, kde $m \geq 0$, $y_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \ldots, n$. Potom také existuje sekvence derivačních kroků $S \Rightarrow^{m+1} h(y_1)h(y_2)\ldots h(y_{j-1})y_jh(y_{j+1})\ldots h(y_n)$ pro libovolné $j = 1, \ldots, n$ v maticové gramatice H.

***Důkaz*:** Tvrzení je dokázáno matematickou indukcí

Základní krok: Nechť $m = 0$. Potom $(S_1, S_2, \ldots, S_n) \Rightarrow^0 (S_1, S_2, \ldots, S_n)$ v n-OGP Γ. Poznamenejme, že také $S \Rightarrow^1 h(S_1)h(S_2)\ldots h(S_{j-1})S_jh(S_{j+1})\ldots h(S_n)$ v maticové gramtice H pro libovolné $j = 1, \ldots, n$, neboť $s_j: S \to h(S_1)h(S_2)\ldots h(S_{j-1})S_jh(S_{j+1})\ldots h(S_n) \in M$.

Indukční hypotéza: Předpokládejme, že tvrzení A platí pro všechny m-kroké derivace, kde $m = 0,\ldots, k$ pro nějaké $k \geq 0$.

Indukční krok: Uvažujme sekvenci derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{k+1} (y_1, y_2, \ldots, y_n)$ v n-OGP Γ. Potom zřejmě existuje multiforma $(u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n)$, kde $A_i \in N_i$, u_i, $v_i \in (N_i \cup T_i)^*$, pro kterou platí $(S_1, S_2, \ldots, S_n) \Rightarrow^k (u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$ v n-OGP Γ, kde $u_ix_iv_i = y_i$ pro všechna $i = 1, \ldots, n$.

I. Ze sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^k (u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n)$ v n-OGP Γ plyne, že existuje sekvence derivačních kroků:

$S \Rightarrow^{k+1} h(u_1A_1v_1)h(u_2A_2v_2)\ldots h(u_{j-1}A_{j-1}v_{j-1})u_jA_jvh(u_{j+1}A_{j+1}v_{j+1})\ldots h(u_nA_nv_n)$ pro libovolné $j = 1, \ldots, n$ v maticové gramatice H podle indukční hypotézy.

II. Nechť existuje derivační krok $(u_1A_1v_1, u_2A_2v_2, \ldots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \ldots, u_nx_nv_n)$ v n-OGP Γ. Potom zřejmě pro n-OGP Γ platí $(p_1: A_1 \rightarrow x_1, p_2: A_2 \rightarrow x_2, \ldots, p_n: A_n \rightarrow x_n) \in Q$. Z algoritmu 4.3.3.1 potom plyne, že $\overline{p}_1\overline{p}_2...\overline{p}_{j-1}p_j\overline{p}_{j+1}...\overline{p}_n \in M$ pro libovolné $j = 1, \ldots, n$. V maticové gramatice H tedy existuje pro libovolné $j = 1, \ldots, n$ derivační krok $h(u_1A_1v_1)h(u_2A_2v_2)\ldots h(u_{j-1}A_{j-1}v_{j-1})u_jA_jv_jh(u_{j+1}A_{j+1}v_{j+1})\ldots h(u_nA_nv_n) \Rightarrow h(u_1x_1v_1)h(u_2x_2v_2)\ldots h(u_{j-1}x_{j-1}v_{j-1})u_jx_jv_jh(u_{j+1}x_{j+1}v_{j+1})\ldots h(u_nx_nv_n)$ podle matice $\overline{p}_1\overline{p}_2...\overline{p}_{j-1}p_j\overline{p}_{j+1}...\overline{p}_n$.

Z částí I. a II. plyne, že $S \Rightarrow^{k+1} h(u_1A_1v_1)h(u_2A_2v_2)\ldots h(u_{j-1}A_{j-1}v_{j-1})u_jA_jv_jh(u_{j+1}A_{j+1}v_{j+1})\ldots h(u_nA_nv_n) \Rightarrow h(u_1x_1v_1)h(u_2x_2v_2)\ldots h(u_{j-1}x_{j-1}v_{j-1})u_jx_jv_jh(u_{j+1}x_{j+1}v_{j+1})\ldots h(u_nx_nv_n)$ v maticové gramatice H. Potom v maticové gramatice H existuje i sekvence derivačních kroků $S \Rightarrow^{k+2} h(u_1x_1v_1)h(u_2x_2v_2)\ldots h(u_{j-1}x_{j-1}v_{j-1})u_jx_jvh(u_{j+1}x_{j+1}v_{j+1})\ldots h(u_nx_nv_n)$ pro libovolné $j = 1, \ldots, n$.

Tvrzení B: Nechť existuje sekvence derivačních kroků $S \Rightarrow^m y$ v maticové gramatice H, kde $m \geq 1$, $y \in (N \cup T)^*$. Potom existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{m-1} (y_1, y_2, \ldots, y_n)$ v n-OGP Γ, kde $y_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \ldots, n$, přičemž pro nějaký index j platí: $y = h(y_1)h(y_2)\ldots h(y_{j-1})y_jh(y_{j+1})\ldots h(y_n)$.

Důkaz: Tvrzení je dokázáno matematickou indukcí

Základní krok: Nechť $m = 1$. V maticové gramatice H zřejmě existuje jedna z derivací tvaru: $S \Rightarrow^1 S_1h(S_2)\ldots h(S_n)$ podle matice s_1: $S \rightarrow S_1h(S_2)\ldots h(S_n)$ nebo $S \Rightarrow^1 h(S_1)S_2\ldots h(S_n)$ podle matice s_2: $S \rightarrow h(S_1)S_2\ldots h(S_n)$ nebo … $S \Rightarrow^1 h(S_1)h(S_2)\ldots S_n$ podle matice s_n: $S \rightarrow h(S_1)h(S_2)\ldots S_n$, žádná jiná. $(S_1, S_2, \ldots, S_n) \Rightarrow^0 (S_1, S_2, \ldots, S_n)$ v n-OGP Γ platí triviálně.

Indukční hypotéza: Předpokládejme, že tvrzení B platí pro všechny m-kroké derivace, kde $m = 0, \ldots, k$ pro nějaké $k \geq 1$.

Indukční krok: Uvažujme sekvenci derivačních kroků $S \Rightarrow^{k+1} y$ v maticové gramatice H. Potom zřejmě existuje větná forma w, pro kterou platí: $S \Rightarrow^k w \Rightarrow y$, kde $w, y \in (N \cup T)^*$.

I. Ze sekvence derivačních kroků $S \Rightarrow^k w$ v maticové gramatice H plyne, že existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{k-1} (w_1, w_2, \ldots, w_n)$ v n-OGP Γ, přičemž pro nějaký index j platí $w = h(w_1)h(w_2)\ldots h(w_{j-1})w_jh(w_{j+1})\ldots h(w_n)$ podle indukční hypotézy.

II. Nechť existuje derivační krok $w \Rightarrow y$ v maticové gramatice H, přičemž $w = h(w_1)h(w_2)\ldots h(w_{j-1})w_jh(w_{j+1})\ldots h(w_n)$, kde $w_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \ldots, n$. Uvažujme pravidlo p_j ve tvaru $A_j \rightarrow x_j$. Dále uvažujme pro všechna $i = 1, \ldots, n$; $i \neq j$ pravidlo $\overline{p}_i$ ve tvaru $h(A_i) \rightarrow h(x_i)$. Neboť podle předpokladu pro libovolné indexy $k, l = 1, \ldots, n$, kde $k \neq l$ platí: $N_k \cap N_l = \emptyset$, musí být pravidlo p_j: $A_j \rightarrow x_j$ aplikováno právě na část větné formy označené jako w_j a ostatní pravidla $\overline{p}_i$: $h(A_i) \rightarrow h(x_i)$ na části větné formy označené jako $h(w_i)$. Pro všechna $i = 1, \ldots, n$ můžeme tedy část větné formy w_i předpokládat ve tvaru $w_i = u_iA_iv_i$, kde $A_i \in N_i$, $u_i, v_i \in (N_i \cup T_i)^*$. V maticové gramatice H tedy existuje derivační krok tvaru: $h(u_1A_1v_1)h(u_2A_2v_2)\ldots h(u_{j-1}A_{j-1}v_{j-1})u_jA_jv_jh(u_{j+1}A_{j+1}v_{j+1})\ldots h(u_nA_nv_n) \Rightarrow h(u_1x_1v_1)h(u_2x_2v_2)\ldots h(u_{j-1}x_{j-1}v_{j-1})u_jx_jv_jh(u_{j+1}x_{j+1}v_{j+1})\ldots h(u_nx_nv_n)$ podle matice $\overline{p}_1\overline{p}_2...\overline{p}_{j-1}p_j\overline{p}_{j+1}...\overline{p}_n \in M$. Z algoritmu 4.3.3.1 plyne, že $(p_1: A_1 \rightarrow x_1, p_2: A_2 \rightarrow x_2, \ldots, p_n: A_n \rightarrow x_n) \in Q$, neboť $\overline{p}_1\overline{p}_2...\overline{p}_{j-1}p_j\overline{p}_{j+1}...\overline{p}_n \in M$. V n-OGP Γ tedy existuje derivační krok tvaru: $(u_1A_1v_1, u_2A_2v_2, \ldots, u_{j-1}A_{j-1}v_{j-1}, u_jA_jv_j, u_{j+1}A_{j+1}v_{j+1}, \ldots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \ldots, u_{j-1}x_{j-1}v_{j-1}, u_jx_jv_j,$

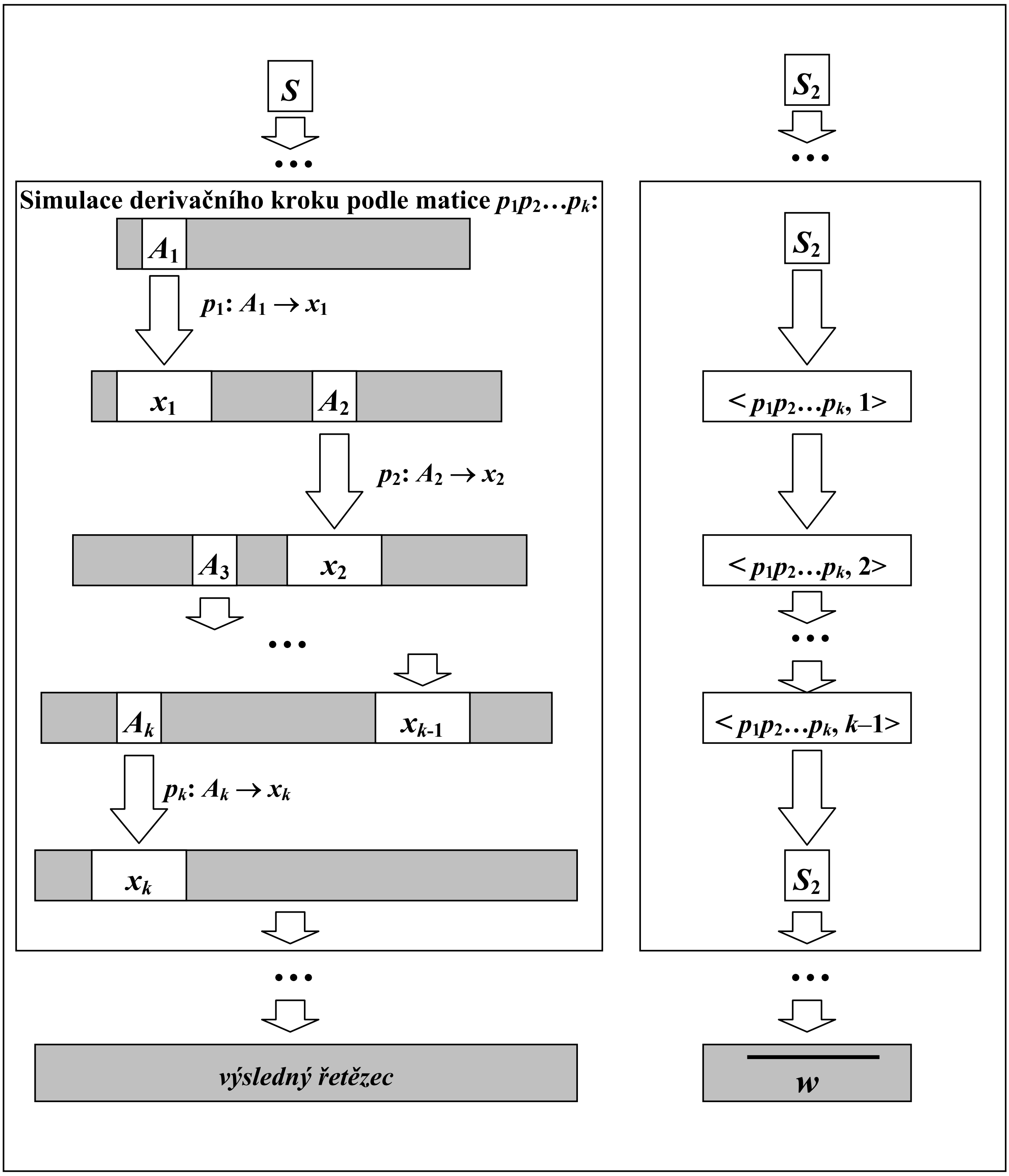
$u_{j+1}x_{j+1}v_{j+1}, \ldots, u_nx_nv_n)$.

Z částí I. a II. plyne: $(S_1, S_2, \ldots, S_{j-1}, S_j, S_{j+1}, \ldots, S_n) \Rightarrow^{k-1} (u_1A_1v_1, u_2A_2v_2, \ldots, u_{j-1}A_{j-1}v_{j-1}, u_jA_jv_j, u_{j+1}A_{j+1}v_{j+1}, \ldots, u_nA_nv_n) \Rightarrow (u_1x_1v_1, u_2x_2v_2, \ldots, u_{j-1}x_{j-1}v_{j-1}, u_jx_jv_j, u_{j+1}x_{j+1}v_{j+1}, \ldots, u_nx_nv_n)$. Potom tedy v tomto n-OGP Γ existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_{j-1}, S_j, S_{j+1}, \ldots, S_n) \Rightarrow^k (u_1x_1v_1, u_2x_2v_2, \ldots, u_{j-1}x_{j-1}v_{j-1}, u_jx_jv_j, u_{j+1}x_{j+1}v_{j+1}, \ldots, u_nx_nv_n)$, přičemž platí: $y = h(u_1x_1v_1)h(u_2x_2v_2)\ldots h(u_{j-1}x_{j-1}v_{j-1})u_jx_jv_jh(u_{j+1}x_{j+1}v_{j+1})\ldots h(u_nx_nv_n)$.

Uvažujme tvrzení A pro nějaké $m \geq 0$ a $y_i \in T_i^*$ pro všechna $i = 1, \ldots, n$. Dostaneme implikaci tvaru: Pokud existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^m (y_1, y_2, \ldots, y_n)$ v n-OGP Γ, potom také existuje sekvence derivačních kroků $S \Rightarrow^{m+1} h(y_1)h(y_2)\ldots h(y_{j-1})y_j h(y_{j+1})\ldots h(y_n)$ pro libovolné $j = 1, \ldots, n$ v maticové gramatice H. Neboť $h(a) = \varepsilon$ pro všechna $a \in \bigcup_{i=1}^{n} T_i$, můžeme danou sekvenci derivačních kroků psát jako: $S \Rightarrow^{m+1} y_j$. Jelikož $L_{union}(\Gamma) = \bigcup_{i=1}^{n} \{w_i: (w_1, w_2, \ldots, w_n) \in n\text{-}L(\Gamma)\}$, zřejmě platí také implikace: Pokud $y \in L_{union}(\Gamma)$, potom $y \in L(H)$, což znamená $L_{union}(\Gamma) \subseteq L(H)$. Dále uvažujme tvrzení B pro nějaké $m \geq 1$ a $y \in T^*$. Dostaneme implikaci tvaru: Pokud existuje sekvence derivačních kroků $S \Rightarrow^m y$ v maticové gramatice H, potom existuje sekvence derivačních kroků $(S_1, S_2, \ldots, S_n) \Rightarrow^{m-1} (y_1, y_2, \ldots, y_n)$ v n-OGP Γ, kde $y_i \in (N_i \cup T_i)^*$ pro všechna $i = 1, \ldots, n$, přičemž pro nějaký index j platí: $y = h(y_1)h(y_2)\ldots h(y_{j-1})y_jh(y_{j+1})\ldots h(y_n)$. Neboť $h(a) = \varepsilon$ pro všechna $a \in \bigcup_{i=1}^{n} T_i$, platí tedy $y = y_j$. Z definice $L_{union}(\Gamma)$ zřejmě platí implikace: Pokud $y \in L(H)$, potom $y \in L_{union}(\Gamma)$, což znamená $L(H) \subseteq L_{union}(\Gamma)$. Z platnosti vztahů $L_{union}(\Gamma) \subseteq L(H)$ a $L(H) \subseteq L_{union}(\Gamma)$ ihned plyne rovnost $L_{union}(\Gamma) = L(H)$.

4.3.4 Převod maticové gramatiky na 2-OGP

Následující algoritmus popíše, jak lze převést libovolnou maticovou gramatiku na 2-OGP, který generuje stejný jazyk v módě sjednocení, konkatenace nebo první komponenty. Algoritmus k libovolné maticové gramatice a k libovolnému řetězci $\overline{w}$ sestrojí 2-OGP, kde první komponenta generuje právě ty řetězce, které generuje zadaná maticová gramatika, přičemž právě pro každý tento případ druhá komponenta generuje řetězec $\overline{w}$. Neformální popis myšlenky algoritmu je následující: První komponenta bude tzv. hlavní komponenta, která bude simulovat derivace v maticové gramatice; druhá komponenta bude pouze jistým způsobem kontrolovat, zda simulace derivací odpovídá skutečně nějaké matici. První komponenta bude tedy obsahovat stejnou bezkontextovou gramatiku, která je součástí maticové gramatiky, druhá komponenta bude pouze generovat větné formy ve tvaru S_2, pokud má být právě odsimulována derivace podle nějaké matice nebo ve tvaru <***m***, ***i***>, kde ***m*** je jedinečný identifikátor dané matice a ***i*** je pozice aktuálního pravidla, které se v dané matici nachází a které se právě simuluje. Pomocí tohoto mechanizmu se zkontroluje korektnost derivace podle jisté matice. Tato simulace je ukončena opět větnou formou tvaru S_2 nebo případně řetězcem $\overline{w}$, pokud je již pomocí první komponenty vygenerován výsledný řetězec. Vše je ilustrováno na obr. 4.3. V dalších podkapitolách je dokázáno, jak se dá vhodnou volbou řetězce $\overline{w}$ docílit skutečnosti, že nově vytvořený 2-OGP potom generuje stejný jazyk v módu první komponenty, v módu konkatenace nebo módu sjednocení jako původní maticová gramatika.



(obr. 4.3: simulace maticové gramatiky pomocí 2-OGP)

4.3.4.1 Algoritmus: Převod maticové gramatiky na 2-OGP

- ***Vstup:*** Maticová gramatika $H = (G, M)$; řetězec $\overline{w} \in \overline{T}^*$, kde $\overline{T}$ je libovolná abeceda
- ***Výstup:*** 2-OGP $\Gamma = (G_1, G_2, Q)$; $\{w_1\colon (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\} = L(H)$
- ***Metoda:***
    - Nechť $G = (N, T, P, S)$
    - Potom položme:
    - $G_1 := G$;
    - $G_2 = (N_2, T_2, P_2, S_2)$, kde:
        - $N_2 := \{S_2\} \cup \{<p_1p_2\ldots p_k, j>\colon p_1p_2\ldots p_k \in M, 1 \le j \le k-1\}$;
        - $T_2 := \overline{T}$;
        - $P_2 :=$

$$
\begin{aligned}
&\{S_2 \to <p_1p_2\ldots p_k, 1>\colon p_1p_2\ldots p_k \in M, k \ge 2\} \cup \\
&\{<p_1p_2\ldots p_k, j> \to <p_1p_2\ldots p_k, j+1>\colon p_1p_2\ldots p_k \in M, k \ge 2, \\
&\qquad\qquad\qquad\qquad\qquad 1 \le j \le k-2\}\} \cup \\
&\{<p_1p_2\ldots p_k, k-1> \to S_2\colon p_1p_2\ldots p_k \in M, k \ge 2\} \cup \\
&\{S_2 \to S_2\colon p_1 \in M, |p_1| = 1\} \cup \\
&\{<p_1p_2\ldots p_k, k-1> \to \overline{w}\colon p_1p_2\ldots p_k \in M, k \ge 2\} \cup \\
&\{S_2 \to \overline{w}\colon p_1 \in M, |p_1| = 1\};
\end{aligned}
$$

- $Q = $

$$
\begin{aligned}
&\{(p_1, S_2 \to <p_1p_2\ldots p_k, 1>)\colon p_1p_2\ldots p_k \in M, k \ge 2\} \cup \\
&\{(p_{j+1}, <p_1p_2\ldots p_k, j> \to <p_1p_2\ldots p_k, j+1>)\colon p_1p_2\ldots p_k \in M, k \ge 2, \\
&\qquad\qquad\qquad\qquad\qquad 1 \le j \le k-2\} \cup \\
&\{(p_k, <p_1p_2\ldots p_k, k-1> \to S_2)\colon p_1p_2\ldots p_k \in M, k \ge 2\} \cup \\
&\{(p_1, S_2 \to S_2)\colon p_1 \in M, |p_1| = 1\} \cup \\
&\{(p_k, <p_1p_2\ldots p_k, k-1> \to \overline{w})\colon p_1p_2\ldots p_k \in M, k \ge 2\} \cup \\
&\{(p_1, S_2 \to \overline{w})\colon p_1 \in M, |p_1| = 1\};
\end{aligned}
$$

4.3.4.2 Příklad

Uvažujme následující maticovou gramatiku $H = (G, M)$, kde:

- $G = (\{S, A, B\}, \{a, b, c\}, \{\mathbf{1}\colon S \to AB, \mathbf{2}\colon A \to aA, \mathbf{3}\colon B \to bBc, \mathbf{4}\colon A \to a, \mathbf{5}\colon B \to bc\}, S)$,
- $M = \{\mathbf{1}, \mathbf{23}, \mathbf{45}\}$.

Pro maticovou gramatiku H a řetězec $\overline{w} = abc$ předchozí algoritmus sestrojí 2-OGP Γ, který bude mít strukturu $\Gamma = (G_1, G_2, Q)$, kde:

- $G_1 = (\{S, A, B\}, \{a, b, c\}, \{\mathbf{1}: S \rightarrow AB, \mathbf{2}: A \rightarrow aA, \mathbf{3}: B \rightarrow bBc, \mathbf{4}: A \rightarrow a, \mathbf{5}: B \rightarrow bc\}, S)$
- $G_2 = (\{S_2, <\mathbf{23}, \mathbf{1}>, <\mathbf{45}, \mathbf{1}>\}, \{a, b, c\}, \{\mathbf{1}: S_2 \rightarrow S_2, \mathbf{2}: S_2 \rightarrow <\mathbf{23}, \mathbf{1}>, \mathbf{3}: <\mathbf{23}, \mathbf{1}> \rightarrow S_2,$

  $\mathbf{4}: <\mathbf{23}, \mathbf{1}> \rightarrow abc, \ \mathbf{5}: S_2 \rightarrow <\mathbf{45}, \mathbf{1}>, \mathbf{6}: <\mathbf{45}, \mathbf{1}> \rightarrow S_2, \mathbf{7}: <\mathbf{45}, \mathbf{1}> \rightarrow abc \ \}$
- $Q = \{(\mathbf{1}, \mathbf{1}), (\mathbf{2}, \mathbf{2}), (\mathbf{3}, \mathbf{3}), (\mathbf{3}, \mathbf{4}), (\mathbf{4}, \mathbf{5}), (\mathbf{5}, \mathbf{6}), (\mathbf{5}, \mathbf{7})\}$

4.3.4.3 Tvrzení

Nechť $H = (G, M)$ je libovolná maticová gramatika. S maticovou gramatikou H a libovolným řetězcem $\overline{w} \in \overline{T}^*$ na vstupu algoritmus 4.3.4.1 zastaví a vytvoří 2-OGP $\Gamma = (G_1, G_2, Q)$, pro který bude platit:

1) $\{w_1: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\} = L(H)$.

2) $\{(w_1, w_2): (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 \neq \overline{w}\} = \emptyset$.

Tvrzení A: $\{w_1: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\} = L(H)$

***Důkaz*:**

***Tvrzení A1*:** Nechť v maticové gramatice H existuje derivační krok $x \Rightarrow y$, kde $x, y \in (N \cup T)^*$. Potom také existují sekvence derivačních kroků $(x, S_2) \Rightarrow^* (y, S_2)$ a $(x, S_2) \Rightarrow^* (y, \overline{w})$ v 2-OGP Γ, sestrojeným pomocí algoritmu 4.3.4.1.

***Důkaz*:**

I. Uvažujme případ, kdy v maticové gramatice H je proveden derivační krok podle matice obsahující pouze jedno pravidlo p_1: $A_1 \rightarrow x_1$, tedy $uA_1v \Rightarrow ux_1v\ [p_1]$, přičemž $uA_1v = x$, $ux_1v = y$. Z algoritmu 4.3.4.1 potom plyne, že $(A_1 \rightarrow x_1,\ S_2 \rightarrow S_2) \in Q$ a $(A_1 \rightarrow x_1,\ S_2 \rightarrow \overline{w}) \in Q$, tedy zřejmě v 2-OGP Γ existují derivace $(uA_1v, S_2) \Rightarrow^1 (ux_1v, S_2)$ a $(uA_1v, S_2) \Rightarrow^1 (ux_1v, \overline{w})$.

II. Uvažujme případ, kdy v maticové gramatice $H = (G, M)$ je proveden derivační krok $x \Rightarrow y$ podle matice tvaru $p_1p_2 ... \ p_k$, kde $k \geq 2$. Podle definice derivačního kroku v maticové gramatice musí tedy i v gramatice G existovat sekvence derivačních kroků:

$$x \Rightarrow y_1\ [p_1] \Rightarrow y_2\ [p_2] \Rightarrow \ldots \Rightarrow y_{k-1}\ [p_{k-1}] \Rightarrow y_k\ [p_k], \text{ přičemž } y_k = y.$$

Z algoritmu 4.3.4.1 potom plyne, že:

1) $(p_1, S_2 \rightarrow <p_1p_2... \ p_k, 1>) \in Q$,

2) $(p_{j+1}, <p_1p_2... \ p_k, j> \rightarrow <p_1p_2... \ p_k, j+1>) \in Q$, kde $j = 1, \ldots, k–2$,

3) $(p_k, <p_1p_2... \ p_k, k–1> \rightarrow S_2) \in Q$,

4) $(p_k, <p_1p_2... \ p_k, k–1> \rightarrow \overline{w}) \in Q$.

Zřejmě tedy v 2-OGP Γ existuje sekvence derivačních kroků:

$(x, S_2) \Rightarrow (y_1, <p_1p_2... \ p_k, 1>) \Rightarrow (y_2, <p_1p_2... \ p_k, 2>) \Rightarrow \ldots \Rightarrow (y_{k-1}, <p_1p_2... \ p_k, k–1>) \Rightarrow (y_k, S_2)$, přičemž $y_k = y$,

$(x, S_2) \Rightarrow (y_1, <p_1p_2... \ p_k, 1>) \Rightarrow (y_2, <p_1p_2... \ p_k, 2>) \Rightarrow \ldots \Rightarrow (y_{k-1}, <p_1p_2... \ p_k, k–1>) \Rightarrow (y_k, \overline{w})$, přičemž $y_k = y$.

Tedy $(x, S_2) \Rightarrow^* (y, S_2)$ a $(x, S_2) \Rightarrow^* (y, \overline{w})$ v 2-OGP Γ.

Tvrzení A2: Nechť v maticové gramatice H existuje sekvence derivačních kroků $x \Rightarrow^m y$ pro nějaké $m \geq 1$, kde $y \in (N \cup T)^*$. Potom také existuje sekvence derivačních kroků $(x, S_2) \Rightarrow^* (y, \overline{w})$ v 2-OGP Γ, sestrojeným pomocí algoritmu 4.3.4.1.

Důkaz: Tvrzení je dokázáno matematickou indukcí

Základní krok: Nechť $m = 1$. Pokud $x \Rightarrow^1 y$ v maticové gramatice H, potom podle tvrzení A1 také existují sekvence derivačních kroků $(x, S_2) \Rightarrow^* (y, \overline{w})$ a $(x, S_2) \Rightarrow^* (y, S_2)$ v 2-OGP Γ, přičemž postačuje platnost sekvence derivačních kroků $(x, S_2) \Rightarrow^* (y, \overline{w})$.

Indukční hypotéza: Předpokládejme, že tvrzení A2 platí pro všechny m-kroké derivace, kde $m = 1,\ldots, k$ pro nějaké $k \geq 1$.

Indukční krok: Uvažujme sekvenci derivačních kroků $x \Rightarrow^{k+1} y$ v maticové gramatice H. Potom zřejmě existuje větná forma $w \in (N \cup T)^*$, pro kterou platí: $x \Rightarrow w \Rightarrow^k y$.

I. Ze sekvence derivačních kroků $w \Rightarrow^k y$ v maticové gramatice H plyne, že existuje sekvence derivačních kroků $(w, S_2) \Rightarrow^* (y, \overline{w})$ v 2-OGP Γ podle indukční hypotézy.

II. Nechť existuje derivační krok $x \Rightarrow w$ v maticové gramatice H. Podle tvrzení A1 také existuje sekvence derivačních kroků $(x, S_2) \Rightarrow^* (w, \overline{w})$ a $(x, S_2) \Rightarrow^* (w, S_2)$ v 2-OGP Γ.

Z částí I. a II. plyne, že $(x, S_2) \Rightarrow^* (w, S_2) \Rightarrow^* (y, \overline{w})$, tedy $(x, S_2) \Rightarrow^* (y, \overline{w})$.

Tvrzení A3: Nechť v 2-OGP Γ existuje sekvence derivačních kroků $(y_0, S_2) \Rightarrow (y_1, z_1) \Rightarrow (y_2, z_2) \Rightarrow \ldots \Rightarrow (y_{k-1}, z_{k-1}) \Rightarrow (y_k, S_2)$ nebo $(y_0, S_2) \Rightarrow (y_1, z_1) \Rightarrow (y_2, z_2) \Rightarrow \ldots \Rightarrow (y_{k-1}, z_{k-1}) \Rightarrow (y_k, \overline{w})$, přičemž pro všechna $i = 1,\ldots, k-1$ platí: $z_i \neq S_2$. Potom v maticové gramatice H existuje přímý derivační krok $y_0 \Rightarrow y_k$.

Důkaz:

I. Uvažujme případ, kdy v 2-OGP Γ existuje tato sekvence derivačních kroků jako pouhý jeden derivační krok ve tvaru $(uA_1v, S_2) \Rightarrow (ux_1v, S_2)$ nebo $(uA_1v, S_2) \Rightarrow (ux_1v, \overline{w})$, přičemž $uA_1v = y_0$, $ux_1v = y_1$. Zřejmě tedy v 2-OGP musí platit: $(A_1 \rightarrow x_1, S_2 \rightarrow S_2) \in Q$ nebo $(A_1 \rightarrow x_1, S_2 \rightarrow \overline{w}) \in Q$. Z algoritmu 4.3.4.1 potom plyne, že maticová gramatika obsahuje matici, která obsahuje právě jedno pravidlo tvaru $A_1 \rightarrow x_1$, tedy zřejmě v maticové gramatice H existuje přímý derivační krok $uA_1v \Rightarrow ux_1v$.

II. Uvažujme případ, kdy v 2-OGP Γ existuje sekvence k derivačních kroků, kde $k \geq 2$, tvaru: $(y_0, S_2) \Rightarrow (y_1, z_1) \Rightarrow (y_2, z_2) \Rightarrow \ldots \Rightarrow (y_{k-1}, z_{k-1}) \Rightarrow (y_k, S_2)$ nebo $(y_0, S_2) \Rightarrow (y_1, z_1) \Rightarrow (y_2, z_2) \Rightarrow \ldots \Rightarrow (y_{k-1}, z_{k-1}) \Rightarrow (y_k, \overline{w})$, přičemž pro všechna $i = 1,\ldots, k-1$ platí: $z_i \neq S_2$. Z algoritmu 4.3.4.1 potom plyne, že musí existovat v maticové gramatice H matice tvaru $p_1p_2\ldots p_k$, kde $k \geq 2$, pro kterou platí: $z_i = \langle p_1p_2\ldots p_k, i\rangle$ pro všechna $i = 1,\ldots, k-1$. Dále z algoritmu 4.3.4.1 plyne, že:

1) $(p_1, S_2 \rightarrow \langle p_1p_2\ldots p_k, 1\rangle) \in Q$,

2) $(p_{j+1}, \langle p_1p_2\ldots p_k, j\rangle \rightarrow \langle p_1p_2\ldots p_k, j+1\rangle) \in Q$, kde $j = 1, \ldots, k-2$,

3) $(p_k, \langle p_1p_2\ldots p_k, k-1\rangle \rightarrow S_2) \in Q$ nebo $(p_k, \langle p_1p_2\ldots p_k, k-1\rangle \rightarrow \overline{w}) \in Q$.

V gramatice G musí tedy existovat sekvence derivačních kroků:

$$y_0 \Rightarrow y_1\ [p_1] \Rightarrow y_2\ [p_2] \Rightarrow \ldots \Rightarrow y_{k-1}\ [p_{k-1}] \Rightarrow y_k\ [p_k]$$

Této sekvenci odpovídá v $H = (G, M)$ jeden derivační krok $y_0 \Rightarrow y_k$ podle matice $p_1p_2\ldots p_k$.

Tvrzení A4: Nechť v 2-OGP Γ existuje sekvence derivačních kroků $(y_0, S_2) \Rightarrow (y_1, z_1) \Rightarrow (y_2, z_2) \Rightarrow \ldots \Rightarrow (y_{r-1}, z_{r-1}) \Rightarrow (y_r, \overline{w})$. Položme $m = Card(\{i: 1 \leq i \leq r-1, z_i = S_2\})$. (Neformálně řečeno, m je počet větných forem z_i, které jsou tvaru S_2). Potom v maticové gramatice H existuje sekvence derivačních kroků $y_0 \Rightarrow^{m+1} y_r$.

Důkaz: Tvrzení je dokázáno matematickou indukcí

Základní krok: Nechť $m = 0$. Potom zřejmě pro všechna $i = 1,\ldots, k-1$ platí: $z_i \neq S_2$. Tvrzení A3 ihned implikuje, že v maticové gramatice H existuje přímý derivační krok $y_0 \Rightarrow y_r$, tedy $y_0 \Rightarrow^1 y_r$.

Indukční hypotéza: Předpokládejme, že tvrzení A4 platí pro všechna $m = 0,\ldots, k$, kde $k \geq 0$.

Indukční krok: Uvažujme v 2-OGP Γ sekvenci derivačních kroků $(y_0, S_2) \Rightarrow (y_1, z_1) \Rightarrow (y_2, z_2) \Rightarrow \ldots \Rightarrow (y_{r-1}, z_{r-1}) \Rightarrow (y_r, \overline{w})$, kde $Card(\{i: 1 \leq i \leq r-1, z_i = S_2\}) = k+1$. Potom zřejmě existuje $p \in \{1,\ldots, r-1\}$, pro které platí: $z_p = S_2$, přičemž $Card(\{i: 1 \leq i \leq p-1, z_i = S_2\}) = 0$, $Card(\{i: p+1 \leq i \leq r-1, z_i = S_2\}) = k$. Sekvence derivačních kroků tedy můžeme rozepsat na $(y_0, z_0) \Rightarrow \ldots \Rightarrow (y_p, z_p) \Rightarrow \ldots \Rightarrow (y_{r-1}, z_{r-1}) \Rightarrow (y_r, \overline{w})$.

I. Ze sekvence derivačních kroků $(y_p, z_p) \Rightarrow \ldots \Rightarrow (y_{r-1}, z_{r-1}) \Rightarrow (y_r, \overline{w})$, kde $z_p = S_2$ a z platnosti vztahu $Card(\{i: p+1 \leq i \leq r-1, z_i = S_2\}) = k$ plyne, že existuje sekvence derivačních kroků $y_p \Rightarrow^{k+1} y_r$ v maticové gramatice H podle indukční hypotézy.

II. Zřejmě ze vztahu $Card(\{i: 1 \leq i \leq p-1, z_i = S_2\}) = 0$ plyne, že pro všechna $i = 1, \ldots, p$ platí: $z_i \neq S_2$. Z platnosti tohoto vztahu a z existence sekvence derivačních kroků $(y_0, z_0) \Rightarrow \ldots \Rightarrow (y_p, z_p)$, kde $z_p = S_2$, tvrzení A3 ihned implikuje, že v maticové gramatice H existuje přímý derivační krok $y_0 \Rightarrow y_p$.

Z částí I. a II. plyne, že $y_0 \Rightarrow y_p \Rightarrow^{k+1} y_r$, tedy $y_0 \Rightarrow^{k+2} y_r$.

Uvažujme tvrzení A2 pro nějaké $m \geq 1$, $x = S$, $y \in T^*$. Dostaneme implikaci tvaru: Pokud existuje sekvence derivačních kroků $S \Rightarrow^* y$ v maticové gramatice H, potom také existuje sekvence derivačních kroků $(S, S_2) \Rightarrow^* (y, \overline{w})$ v 2-OGP Γ, což znamená: Pokud $y \in L(H)$, potom $(y, \overline{w}) \in 2\text{-}L(\Gamma)$, tedy platí: $L(H) \subseteq \{w_1: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\}$. Uvažujme tvrzení A4 pro nějaké $m \geq 1$, $y_0 = S$, $y_r \in T^*$. Dostaneme implikaci tvaru: Pokud existuje sekvence derivačních kroků $(S, S_2) \Rightarrow (y_1, z_1) \Rightarrow (y_2, z_2) \Rightarrow \ldots \Rightarrow (y_{r-1}, z_{r-1}) \Rightarrow (y_r, \overline{w})$ v 2-OGP Γ, tedy pokud $(S, S_2) \Rightarrow^* (y_r, \overline{w})$, potom také existuje sekvence derivačních kroků $S \Rightarrow^{m+1} y_r$ v maticové gramatice H, kde $m = Card(\{i: 1 \leq i \leq r-1, z_i = S_2\})$, tedy $S \Rightarrow^* y_r$, což znamená: Pokud $(y, \overline{w}) \in 2\text{-}L(\Gamma)$, potom $y \in L(H)$, tedy platí: $\{w_1: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\} \subseteq L(H)$. Z platnosti vztahů $L(H) \subseteq \{w_1: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\}$ a $\{w_1: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\} \subseteq L(H)$ ihned plyne rovnost: $\{w_1: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\} = L(H)$.

Tvrzení B: $\{(w_1, w_2): (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 \neq \overline{w}\} = \emptyset$

Důkaz:

Podle algoritmu 4.3.4.1 gramatika $G_2 = (N_2, T_2, P_2, S_2)$ obsahuje pouze pravidla tvaru: $A \rightarrow B$ nebo $A \rightarrow \overline{w}$, kde $A, B \in N$. Gramatika G_2 tedy musí generovat jeden z jazyků: $\emptyset$, $\{\overline{w}\}$. Pokud tuto gramatiku obsahuje druhá komponenta 2-OGP Γ, je podle obecné definice 2-OGP zřejmé, že na druhé pozici multiformy lze vygenerovat maximálně řetězec $\overline{w}$.

4.3.4.4 Tvrzení

Nechť $H = (G, M)$ je libovolná maticová gramatika. S maticovou gramatikou H a s libovolným řetězcem $\overline{w} \in \overline{T}^*$ na vstupu algoritmus 4.3.4.1 vytvoří 2-OGP $\Gamma = (G_1, G_2, Q)$, pro který bude platit: $L_{first}(\Gamma) = L(H)$.

Důkaz: Podle tvrzení 4.3.4.3 platí:

1) $\{w_1: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\} = L(H)$,

2) $\{(w_1, w_2): (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 \neq \overline{w}\} = \emptyset$.

Zřejmě tedy platí: $L_{first}(\Gamma) = \{w_1: (w_1, w_2) \in 2\text{-}L(\Gamma)\} = \{w_1: (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 = \overline{w}\} \cup \{w_1: (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 \neq \overline{w}\} = \{w_1: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\} \cup \emptyset = \{w_1: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\} = L(H)$.

4.3.4.5 Tvrzení

Nechť $H = (G, M)$ je libovolná maticová gramatika. Položme $\overline{w} = \varepsilon$. S maticovou gramatikou H a s řetězcem $\overline{w}$ na vstupu algoritmus 4.3.4.1 vytvoří 2-OGP $\Gamma = (G_1, G_2, Q)$, pro který bude platit: $L_{conc}(\Gamma) = L(H)$.

Důkaz: Podle tvrzení 4.3.4.3 platí:

1) $\{w_1: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\} = L(H)$,

2) $\{(w_1, w_2): (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 \neq \overline{w}\} = \emptyset$.

Zřejmě tedy platí: $L_{conc}(\Gamma) = \{w_1w_2: (w_1, w_2) \in 2\text{-}L(\Gamma)\} = \{w_1w_2: (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 = \overline{w}\} \cup \{w_1w_2: (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 \neq \overline{w}\} = \{w_1\overline{w}: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\} \cup \emptyset = \{w_1\overline{w}: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\} = L(H)$, neboť $\overline{w} = \varepsilon$.

4.3.4.6 Tvrzení

Nechť $H = (G, M)$ je libovolná maticová gramatika. Pokud $L(H) = \emptyset$, zvolme $\overline{w}$ libovolně, pokud $L(H) \neq \emptyset$, tedy existuje $x \in L(H)$, položme $\overline{w} = x$. S maticovou gramatikou H a s řetězcem $\overline{w}$ na vstupu algoritmus 4.3.4.1 vytvoří 2-OGP $\Gamma = (G_1, G_2, Q)$, pro který bude platit: $L_{union}(\Gamma) = L(H)$.

Důkaz:

1) Uvažujme situaci, kdy $L(H) = \emptyset$. Dokažme, že $L_{union}(\Gamma) = \emptyset$:

Důkaz sporem:

Předpokládejme, že $L_{union}(\Gamma) \neq \emptyset$. Potom zřejmě existuje prvek $(w_1, w_2) \in 2\text{-}L(\Gamma)$.

- Nechť $w_2 = \overline{w}$: Podle tvrzení 4.3.4.3 platí: $\{w_1: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\} = L(H)$. Existence prvku $(w_1, \overline{w})$ implikuje, že $w_1 \in L(H)$, přičemž $L(H) = \emptyset$, což je spor.

- Nechť $w_2 \neq \overline{w}$: Existence prvku (w_1, w_2), kde $w_2 \neq \overline{w}$, implikuje $\{(w_1, w_2): w_1 \in T^*, w_2 \in \overline{T}^*, w_2 \neq \overline{w}\} \neq \emptyset$. Avšak podle tvrzení 4.3.4.3 platí: $\{(w_1, w_2): w_1 \in T, w_2 \in \overline{T}, w_2 \neq \overline{w}\} = \emptyset$, což je spor.

2) Uvažujme situaci, kdy $L(H) \neq \emptyset$. Dokažme, že $L_{union}(\Gamma) = L(H)$:

Důkaz: Podle tvrzení 4.3.4.3 platí:

1) $\{w_1: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\} = L(H)$,

2) $\{(w_1, w_2): (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 \neq \overline{w}\} = \emptyset$.

Dále položme: $M_1 = \{w_1: (w_1, w_2) \in 2\text{-}L(\Gamma)\}$, $M_2 = \{w_2: (w_1, w_2) \in 2\text{-}L(\Gamma)\}$.

Protože bylo zvoleno $\overline{w} = x$, kde $x \in L(H)$, zřejmě z 1) plyne: $(\overline{w}, \overline{w}) \in 2\text{-}L(\Gamma)$, tedy $\overline{w} \in M_1$ a $\overline{w} \in M_2$.

Potom zřejmě platí:

- $M_1 = \{w_1: (w_1, w_2) \in 2\text{-}L(\Gamma)\} = \{w_1: (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 = \overline{w}\} \cup \{w_1: (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 \neq \overline{w}\} = \{w_1: (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 = \overline{w}\} \cup \emptyset = \{w_1: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\}$
- $M_2 = \{w_2: (w_1, w_2) \in 2\text{-}L(\Gamma)\} = \{w_2: (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 = \overline{w}\} \cup \{w_2: (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 \neq \overline{w}\} = \{w_2: (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 = \overline{w}\} \cup \emptyset = \{w_2: (w_1, w_2) \in 2\text{-}L(\Gamma), w_2 = \overline{w}\} = \{\overline{w}\}$.

Jelikož $\overline{w} \in M_1$ a $\overline{w} \in M_2$ a dále $M_2 = \{\overline{w}\}$, musí tedy platit $M_2 \subseteq M_1$, tedy $M_1 \cup M_2 = M_1$.

Zřejmě tedy platí: $L_{union}(\Gamma) = \bigcup_{i=1}^{2} \{w_i: (w_1, w_2) \in 2\text{-}L(\Gamma)\} = \{w_1: (w_1, w_2) \in 2\text{-}L(\Gamma)\} \cup \{w_2: (w_1, w_2) \in 2\text{-}L(\Gamma)\} = M_1 \cup M_2 = M_1 = \{w_1: (w_1, \overline{w}) \in 2\text{-}L(\Gamma)\} = L(H)$.

4.3.4.7 Tvrzení

Pro libovolný n-OGP Γ existuje maticová gramatika H, pro kterou platí: $L(H) = L_{union}(\Gamma)$.

***Důkaz*:**

Plyne z algoritmu 4.3.3.1.

4.3.4.8 Tvrzení

Pro libovolnou maticovou gramatiku H existuje 2-OGP, pro který platí: $L_{union}(\Gamma) = L(H)$.

***Důkaz*:**

Plyne z tvrzení 4.3.4.6.

4.3.4.9 Tvrzení

Pro libovolný n-OGP Γ existuje 2-OGP $\overline{\Gamma}$, pro který platí $L_{union}(\overline{\Gamma}) = L_{union}(\Gamma)$.

***Důkaz*:**

Plyne z tvrzení 4.3.4.7 a tvrzení 4.3.4.8.

4.3.4.10 Tvrzení

Pro libovolný n-OGP Γ existuje maticová gramatika H, pro kterou platí: $L(H) = L_{conc}(\Gamma)$.

***Důkaz*:**

Plyne z algoritmu 4.3.1.1.

4.3.4.11 Tvrzení

Pro libovolnou maticovou gramatiku H existuje 2-OGP, pro který platí: $L_{conc}(\Gamma) = L(H)$.

***Důkaz*:**

Plyne z tvrzení 4.3.4.5.

4.3.4.12 Tvrzení

Pro libovolný n-OGP Γ existuje 2-OGP $\overline{\Gamma}$, pro který platí $L_{conc}(\overline{\Gamma}) = L_{conc}(\Gamma)$.

***Důkaz*:**

Plyne z tvrzení 4.3.4.10 a tvrzení 4.3.4.11.

4.3.4.13 Tvrzení

Pro libovolný n-OGP Γ existuje maticová gramatika H, pro kterou platí: $L(H) = L_{first}(\Gamma)$.

***Důkaz*:**

Plyne z algoritmu 4.3.2.1.

4.3.4.14 Tvrzení

Pro libovolnou maticovou gramatiku H existuje 2-OGP, pro který platí: $L_{first}(\Gamma) = L(H)$.

***Důkaz*:**

Plyne z tvrzení 4.3.4.4.

4.3.4.15 Tvrzení

Pro libovolný n-OGP Γ existuje 2-OGP $\overline{\Gamma}$, pro který platí $L_{first}(\overline{\Gamma}) = L_{first}(\Gamma)$.

***Důkaz*:**

Plyne z tvrzení 4.3.4.13 a tvrzení 4.3.4.14.

5 Hybridní multigenerativní gramatické systémy

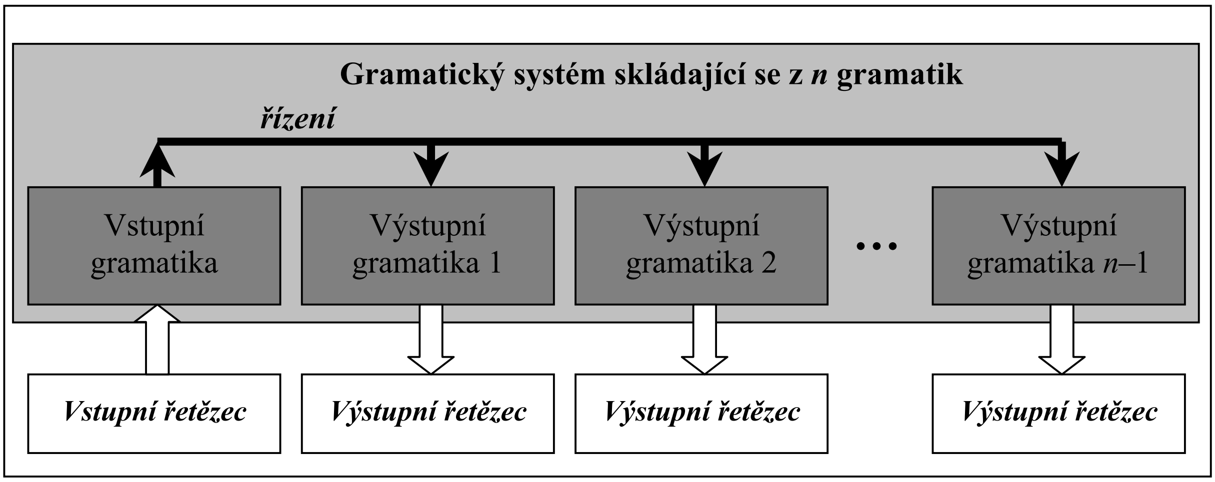
V předcházejících dvou kapitolách byly zavedeny kanonické a obecné multigenerativní gramatické systémy. Kanonické multigenerativní gramatické systémy obsahovaly bezkontextové gramatiky, ve kterých se jednotlivé větné formy generovaly pouze pomocí nejlevější derivace, což znamená, že v každé větné formě byl přepsán vždy nejlevější nonterminální symbol. Zcela analogicky by mohly být nadefinovány duální kanonické multigenerativní gramatické systémy, ve kterých by se naopak na jednotlivé větné formy aplikovaly pouze nejpravější derivace.

Naopak obecné multigenerativní gramatické systémy obsahovaly bezkontextové gramatiky, ve kterých se jednotlivé větné formy generovaly pomocí obecné derivace, tedy přepsán mohl být libovolný nonterminální symbol.

Pro potřeby praxe je někdy vhodné mít takový multigenerativní gramatický systém, ve kterém nějaké gramatiky budou generovat větné formy pomocí kanonické derivace (tj. pomocí nejlevější nebo nejpravější), jiné zase pomocí obecné derivace. Tyto gramatické systémy budou označovány jako hybridní multigenerativní gramatické systémy. Analogicky jako u kanonických a obecných multigenerativních gramatických systémů může být definován hybridní n-generativní gramatický systém nonterminálově synchronizovaný (označený jako n-HGN) a hybridní n-generativní gramatický systém pravidlově synchronizovaný (označený jako n-HGP). Využití takovýchto gramatických systémů bude ukázáno v následující kapitole. Formální definice hybridních multigenerativních gramatických systémů jsou obdobné jako definice kanonických a obecných multigenerativních gramatických systémů, přičemž musí být explicitně vyjádřeno, u kterých gramatik je prováděna nejlevější respektive nejpravější derivace a u kterých obecná derivace.

6 Aplikace multigenerativních gramatických systémů

V předcházejících kapitolách bylo pohlíženo na gramatické systémy jako na multigenerátory, které paralelně generovaly *n*-tice řetězců. Z těchto vygenerovaných *n*-tic řetězců se určitou operací vytvořily jednoduché řetězce, pomocí nichž byl definován jazyk. Pokud ovšem zůstaneme u daných *n*-tic řetězců, můžeme pomocí nich definovat *n*-nární relaci, tedy jakýsi zobecněný překlad mezi více jazyky. Tento multipřeklad budeme provádět právě pomocí gramatického systému následujícím způsobem: V dané *n*-tici gramatik je jedna gramatika vstupní a ostatních *n*–1 gramatik je výstupních. Pomocí vstupní gramatiky se provede syntaktická analýza vstupního řetězce, která v tomto gramatickém systému řídí generování *n*–1 výstupních řetězců pomocí výstupních gramatik. Celá situace je znázorněna na obr. 6.1.



(obr. 6.1: struktura gramatického systému jako multipřekladače)

Příklady využití takového modelu multipřekladače:

- Překladač může být použit opět jako pouhý deterministický akceptor složitějších jazyků
- Věta z jistého přirozeného jazyka má být přeložena do několika jiných přirozených jazyků
- Assemblerovský kód má být přeložen do více binárních kódů pro různé procesory

V této kapitole bude popsáno omezení dříve definovaných gramatických systémů tak, aby tento multipřeklad mohl být proveden deterministickým způsobem. Překlad může být proveden sériovým i paralelním způsobem.

Při sériovém přístupu se nejprve provede syntaktická analýza pro vstupní řetězec. Až je tato syntaktická analýza zcela dokončená, je předána jistá informace (např. levý nebo pravý rozbor) výstupním gramatikám, které na základě této informace vygenerují výstupní řetězce. Pro tento přístup nejsou přímým modelem dříve zmíněné gramatické systémy, ale jejich modifikace.

Při paralelním přístupu syntaktická analýza pro vstupní řetězec současně řídí generování řetězců u výstupních gramatik. Již ve chvíli, kdy se úspěšně dokončí syntaktická analýza pro vstupní řetězec, mají výstupní gramatiky vygenerovány výstupní řetězce. Zbývající kapitoly se budou zabývat paralelním přístupem. Předpokládá se, že je čtenář seznámen se základními pojmy a algoritmy pro syntaktickou analýzu, které jsou například uvedeny v [1].

6.1 Deterministická paralelní syntaktická analýza shora dolů pro *n*-KGP

U vstupní gramatiky je provedena syntaktická analýza shora dolů pro daný řetězec, přičemž použití daného pravidla při syntaktické analýze přímým způsobem řídí generování řetězce u ostatních výstupních gramatik. Jedná se tedy o ***přímou simulaci n-KGP***, neboť v každém kroku syntaktické analýzy se právě simuluje jeden derivační krok *n*-KGP.

6.1.1 Slabý deterministický LL n-KGP

V této kapitole je nadefinován slabý deterministický *LL n*-KGP. Bude se jednat o jistým způsobem omezený *n*-KGP, aby pro něj mohla být provedena syntaktická analýza shora dolů deterministickým způsobem. Myšlenka je následující: Z hlediska omezení pro *n*-KGP bude vyžadováno, aby vstupní gramatika byla *LL*-gramatika, což zaručí, že bude možno provést deterministickým způsobem syntaktickou analýzu shora dolů pro vstupní gramatiku. Dále bude vyžadováno, aby komponenta Q obsahovala pouze takové *n*-tice pravidel, ve kterých je každé pravidlo ze vstupní gramatiky obsaženo maximálně v jednom prvku. Tím bude zaručeno, že použití jistého pravidla během syntaktické analýzy jednoznačně určí *n*-tici pravidel, která bude aplikována i na ostatní gramatiky.

Poznámka: Podrobný popis, jak lze pro danou *LL*-gramatiku sestavil *LL*-tabulku, je uveden například v [1].

6.1.1.1 Definice slabého deterministického LL n-KGP

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je *n*-KGP. Potom řekneme, že Γ je *slabý deterministický LL n-KGP* pro vstupní gramatiku s indexem *IN*, kde $1 \leq IN \leq n$, pokud jsou splněny následující podmínky:

1) Gramatika G_{IN} je *LL*-gramatika.
2) Pro libovolné dva prvky $(p_1, p_2, \ldots, p_n) \in Q$ a $(r_1, r_2, \ldots, r_n) \in Q$ platí:
   Pokud $p_{IN} = r_{IN}$, potom také $p_i = r_i$ pro všechna $i = 1, \ldots, n$.

6.1.1.2 Algoritmus: Syntaktická analýza shora dolů pro slabý deterministický LL n-KGP

- ***Vstup:*** slabý deterministický *LL n*-KGP $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ pro vstupní gramatiku s indexem *IN*; *LL*-tabulka pro vstupní gramatiku; řetězec $w \in T_{IN}^*$, kde T_{IN} je množina terminálních symbolů vstupní gramatiky
- ***Výstup:*** Pokud existuje $(w_1, w_2, \ldots, w_{IN-1}, w_{IN}, w_{IN+1}, \ldots w_n) \in n\text{-}L(\Gamma)$, přičemž $w_{IN} = w$, pak výstupem je $(w_1, w_2, \ldots, w_{IN-1}, w_{IN+1}, \ldots, w_n)$, jinak chyba
- ***Metoda:***
  - Nechť $G_i = (N_i, T_i, P_i, S_i)$ pro všechna $i = 1, \ldots, n$;
    $M = \{1, 2, \ldots, n\}$; $M_{IN} = M - \{IN\}$;
  - Pro všechna $i \in M$: StackInit(St_i); Push(St_i, \$); Push(St_i, S_i);
  - Pro všechna $i \in M_{IN}$: $w_i := \varepsilon$;

- $w := w\$$; $a :=$ první symbol z w;
- **repeat**

  $A := \text{Top}(St_{IN})$;

  case A **of**

  $A = \$$: **if** $a = \$$ **and** pro všechna $i \in M_{IN}$ $\text{Top}(St_i) = \$$ **then** Úspěch

  else Chyba;

  $A \in T_{IN}$: **if** $A = a$ **then** $\text{Pop}(St_{IN})$; $a :=$ další symbol z w **else** Chyba;

  $A \in N_{IN}$: **if** existuje p: $A \rightarrow x \in LL$-tabulka[a, A] **and**

  existuje $(p_1: A_1 \rightarrow x_1, p_2: A_2 \rightarrow x_2, \ldots, p_n: A_n \rightarrow x_n) \in Q$,

  kde $p_{IN} = p$ **then**

  Pro všechna $i \in M$:

  if $\text{Top}(St_i) = A_i$ **then** $\text{Pop}(St_i)$; $\text{Push}(St_i, \mathit{reversal}(x_i))$

  else Chyba;

  Pro všechna $i \in M_{IN}$:

  while $\text{Top}(St_i) \in T_i$ **do** $a := \text{TopPop}(St_i)$; $w_i := w_i a$;

  else Chyba;

  until Chyba **or** Úspěch;

Poznámky k algoritmu:

- Pokud je v průběhu syntaktické analýzy zavolána funkce Top na prázdný zásobník, je algoritmus ukončen chybou (plyne to ze standardní definice sémantiky funkce Top).
- Pokud je na zásobník vkládán řetězec znaků, je tím míněno postupné vkládání symbolů na zásobník z daného řetězce v tomto pořadí.

6.1.1.3 Příklad

Uvažujme následující slabý deterministický LL n-KGP $\Gamma = (G_1, G_2, Q)$ pro vstupní gramatiku s indexem 1, kde:

- $G_1 = (\{S_1, A_1\}, \{a, b, c\}, \{\mathbf{1}: S_1 \rightarrow aS_1, \mathbf{2}: S_1 \rightarrow A_1, \mathbf{3}: A_1 \rightarrow bA_1c, \mathbf{4}: A_1 \rightarrow \varepsilon\}, S_1)$,
- $G_2 = (\{S_2\}, \{d\}, \{\mathbf{1}: S_2 \rightarrow S_2S_2, \mathbf{2}: S_2 \rightarrow S_2, \mathbf{3}: S_2 \rightarrow d, \mathbf{4}: S_2 \rightarrow \varepsilon\}, S_2)$,
- $Q = \{(\mathbf{1}, \mathbf{1}), (\mathbf{2}, \mathbf{2}), (\mathbf{3}, \mathbf{3}), (\mathbf{4}, \mathbf{4})\}$.

LL-tabulka pro gramatiku G_1 je ve tvaru:

	a	***b***	***c***	**$**
S_1	**1**	**2**		**2**
A_1		**3**	**4**	**4**

Ukažme, že tento gramatický systém je skutečně slabý deterministický *LL* n-KGP:

1) Vstupní gramatika je *LL*-gramatikou (*LL*-tabulka neobsahuje žádné kolize).
2) Pro libovolné dva prvky $(p_1, p_2) \in Q$ a $(r_1, r_2) \in Q$ platí:
   Pokud $p_1 = r_1$, potom také $p_2 = r_2$.

Proveďme syntaktickou analýzu řetězce ***aabbcc***:

Zásobník St_1	**Vstupní řetězec w**	**Vybrané pravidlo p_1**	**Zásobník St_2**	**Výstupní řetězec w_2**	**Přidružené pravidlo p_2**
$\$\boldsymbol{S_1}$	$\boldsymbol{a}abbcc\$$	**1**: $S_1 \to aS_1$	$\$\boldsymbol{S_2}$	ε	**1**: $S_2 \to S_2S_2$
$\$S_1\boldsymbol{a}$	$\boldsymbol{a}abbcc\$$		$\$S_2\boldsymbol{S_2}$	ε	
$\$\boldsymbol{S_1}$	$\boldsymbol{a}bbcc\$$	**1**: $S_1 \to aS_1$	$\$S_2\boldsymbol{S_2}$	ε	**1**: $S_2 \to S_2S_2$
$\$S_1\boldsymbol{a}$	$\boldsymbol{a}bbcc\$$		$\$S_2S_2\boldsymbol{S_2}$	ε	
$\$\boldsymbol{S_1}$	$\boldsymbol{b}bcc\$$	**2**: $S_1 \to A_1$	$\$S_2S_2\boldsymbol{S_2}$	ε	**2**: $S_2 \to S_2$
$\$\boldsymbol{A_1}$	$\boldsymbol{b}bcc\$$	**3**: $A_1 \to bA_1c$	$\$S_2S_2\boldsymbol{S_2}$	ε	**3**: $S_2 \to d$
$\$cA_1\boldsymbol{b}$	$\boldsymbol{b}bcc\$$		$\$S_2\boldsymbol{S_2}$	d	
$\$c\boldsymbol{A_1}$	$\boldsymbol{b}cc\$$	**3**: $A_1 \to bA_1c$	$\$S_2\boldsymbol{S_2}$	d	**3**: $S_2 \to d$
$\$ccA_1\boldsymbol{b}$	$\boldsymbol{b}cc\$$		$\$\boldsymbol{S_2}$	dd	
$\$c\boldsymbol{c}$	$\boldsymbol{c}c\$$	**4**: $A_1 \to \varepsilon$	$\$\boldsymbol{S_2}$	dd	**4**: $S_2 \to \varepsilon$
$\$\boldsymbol{c}$	$\boldsymbol{c}\$$		$\boldsymbol{\$}$	dd	
$\boldsymbol{\$}$	$\boldsymbol{\$}$		$\boldsymbol{\$}$	dd	

Překlad řetězce *aabbcc* na řetězec *dd* proběhl úspěšně, tedy $(aabbcc, dd) \in 2\text{-}L(\Gamma)$.

Poznamenejme, že $2\text{-}L(\Gamma) = \{(a^nb^nc^n, d^n): n \geq 0\}$. V tomto případě je překladač použit jako pouhý deterministický akceptor jazyka, který není bezkontextový.

6.1.1.4 Definice slabého n-cestného deterministického LL n-KGP

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-KGP. Potom řekneme, že Γ je *slabý n-cestný deterministický LL n-KGP*, pokud Γ je slabý deterministický *LL* n-KGP pro libovolnou vstupní gramatiku s indexem *IN*, kde $1 \leq IN \leq n$.

6.1.2 Silný deterministický LL n-KGP

Přestože předchozí příklad ilustroval, že pomocí slabého deterministického systému může být provedena syntaktická analýza i některých jazyků, které nejsou bezkontextové, mohou být tyto podmínky ještě slabší, a přesto syntaktická analýza může proběhnout deterministickým způsobem. Myšlenka je následující: U slabého deterministického *LL* n-KGP nebylo využito podstatné informace, že známe hodnoty symbolů na vrcholech jednotlivých zásobníků. S využitím znalosti těchto hodnot lze podstatně oslabit jak podmínku na očekávání *LL*-gramatiky jako vstupní gramatiky, tak podmínku kladenou na komponentu Q ve slabém deterministickém *LL* n-KGP. Tato podmínka musí zaručit, že jestliže *LL*-tabulka obsahuje na daném políčku více pravidel, potom obsahy jednotlivých zásobníků rozhodnou, které z těchto pravidel bude použito.

6.1.3 Definice silného deterministického LL n-KGP

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-KGP. Potom řekneme, že Γ je *silný deterministický LL n-KGP* pro vstupní gramatiku s indexem IN, kde $1 \leq IN \leq n$, pokud je splněna následující podmínka:

Uvažujme vstupní gramatiku G_{IN}. Potom pro libovolné dva prvky $(p_1: A_1 \rightarrow x_1, p_2: A_2 \rightarrow x_2, \ldots, p_n: A_n \rightarrow x_n) \in Q$, $(r_1: B_1 \rightarrow y_1, r_2: B_2 \rightarrow y_2, \ldots, r_n: B_n \rightarrow y_n) \in Q$ platí: Pokud $A_i = B_i$ pro všechna $i = 1,\ldots, n$, potom $Predict(p_{IN}) \cap Predict(r_{IN}) = \emptyset$.

6.1.3.1 Algoritmus: Syntaktická analýza shora dolů pro silný deterministický LL n-KGP

- ***Vstup:*** silný deterministický LL n-KGP $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ pro vstupní gramatiku s indexem IN; LL-tabulka pro vstupní gramatiku; řetězec $w \in T_{IN}^*$, kde T_{IN} je množina terminálních symbolů vstupní gramatiky
- ***Výstup:*** Pokud existuje $(w_1, w_2, \ldots, w_{IN-1}, w_{IN}, w_{IN+1}, \ldots w_n) \in n\text{-}L(\Gamma)$, přičemž $w_{IN} = w$, pak výstupem je $(w_1, w_2, \ldots, w_{IN-1}, w_{IN+1}, \ldots, w_n)$, jinak chyba
- ***Metoda:***
  - o Nechť $G_i = (N_i, T_i, P_i, S_i)$ pro všechna $i = 1, \ldots, n$;

    $M = \{1, 2, \ldots, n\}$; $M_{IN} = M - \{IN\}$;
  - o Pro všechna $i \in M$: StackInit(St_i); Push(St_i, \$); Push(St_i, S_i);
  - o Pro všechna $i \in M_{IN}$: $w_i := \varepsilon$;
  - o $w := w\$$; $a :=$ první symbol z w;
  - o **repeat**

    $A :=$ Top(St_{IN});

    case A **of**

    $A = \$$: **if** $a = \$$ **and** pro všechna $i \in M_{IN}$ Top(St_i) = \$ **then** Úspěch

    else Chyba;

    $A \in T_{IN}$: **if** $A = a$ **then** Pop(St_{IN}); $a :=$ další symbol z w **else** Chyba;

    $A \in N_{IN}$: Pro všechna $i \in M$: $A_i :=$ Top(St_i);

    if existuje p: $A \rightarrow x \in LL$-tabulka$[a, A]$ **and** existuje

    $(p_1: A_1 \rightarrow x_1, p_2: A_2 \rightarrow x_2, \ldots, p_n: A_n \rightarrow x_n) \in Q$, kde $p_{IN} = p$ **then**

    Pro všechna $i \in M$:

    if Top(St_i) = A_i **then** Pop(St_i); Push(St_i, $reversal(x_i)$)

    else Chyba;

    Pro všechna $i \in M_{IN}$:

    while Top(St_i) $\in T_i$ **do** $a :=$ TopPop(St_i); $w_i := w_i a$;

    else Chyba;

    until Chyba **or** Úspěch;

6.1.3.2 Příklad

Uvažujme následující silný deterministický *LL n*-KGP $\Gamma = (G_1, G_2, Q)$ pro vstupní gramatiku s indexem 1, kde:

- $G_1 = (\{S_1, A_1, X_1, Y_1, F_1\}, \{a\}, \{\mathbf{1}: S_1 \rightarrow aA_1X_1, \mathbf{2}: A_1 \rightarrow a, \mathbf{3}: X_1 \rightarrow X_1, \mathbf{4}: X_1 \rightarrow Y_1A_1X_1,$
  $\mathbf{5}: X_1 \rightarrow F_1, \mathbf{6}: Y_1 \rightarrow Y_1A_1, \mathbf{7}: Y_1 \rightarrow \varepsilon, \mathbf{8}: F_1 \rightarrow F_1, \mathbf{9}: F_1 \rightarrow \varepsilon\}, S_1)$,
- $G_2 = (\{S_2, A_2, X_2, Y_2\}, \{a\}, \{\mathbf{1}: S_2 \rightarrow aaX_2, \mathbf{2}: A_2 \rightarrow a, \mathbf{3}: X_2 \rightarrow X_2, \mathbf{4}: X_2 \rightarrow Y_2A_2A_2X_2,$
  $\mathbf{5}: X_2 \rightarrow \varepsilon, \mathbf{6}: Y_2 \rightarrow Y_2A_2A_2, \mathbf{7}: Y_2 \rightarrow \varepsilon\}, S_2)$,
- $Q = \{(\mathbf{1}, \mathbf{1}), (\mathbf{2}, \mathbf{4}), (\mathbf{2}, \mathbf{6}), (\mathbf{3}, \mathbf{7}), (\mathbf{4}, \mathbf{2}), (\mathbf{5}, \mathbf{2}), (\mathbf{6}, \mathbf{2}), (\mathbf{7}, \mathbf{3}), (\mathbf{8}, \mathbf{2}), (\mathbf{9}, \mathbf{5})\}$.

LL-tabulka pro gramatiku G_1 je ve tvaru:

	a	**$**
$\boldsymbol{S_1}$	**1**	
$\boldsymbol{A_1}$	**2**	
$\boldsymbol{X_1}$	**3, 4**	**3, 5**
$\boldsymbol{Y_1}$	**6, 7**	
$\boldsymbol{F_1}$		**8, 9**

Ukažme, že tento gramatický systém je skutečně silný deterministický *LL n*-KGP:
Uvažujme všechny prvky množiny Q:
$(\mathbf{1}: S_1 \rightarrow aA_1X_1, \mathbf{1}: S_2 \rightarrow aaX_2), (\mathbf{2}: A_1 \rightarrow a, \mathbf{4}: X_2 \rightarrow Y_2A_2A_2X_2), (\mathbf{2}: A_1 \rightarrow a, \mathbf{6}: Y_2 \rightarrow Y_2A_2A_2),$ $(\mathbf{3}: X_1 \rightarrow X_1, \mathbf{7}: Y_2 \rightarrow \varepsilon), (\mathbf{4}: X_1 \rightarrow Y_1A_1X_1, \mathbf{2}: A_2 \rightarrow a), (\mathbf{5}: X_1 \rightarrow F_1, \mathbf{2}: A_2 \rightarrow a), (\mathbf{6}: Y_1 \rightarrow Y_1A_1,$ $\mathbf{2}: A_2 \rightarrow a), (\mathbf{7}: Y_1 \rightarrow \varepsilon, \mathbf{3}: X_2 \rightarrow X_2), (\mathbf{8}: F_1 \rightarrow F_1, \mathbf{2}: A_2 \rightarrow a), (\mathbf{9}: F_1 \rightarrow \varepsilon, \mathbf{5}: X_2 \rightarrow \varepsilon)$
Nyní vyberme jen ty dvojice $(p_1: A_1 \rightarrow x_1, p_2: A_2 \rightarrow x_2) \in Q$, $(r_1: B_1 \rightarrow y_1, r_2: B_2 \rightarrow y_2) \in Q$, pro které platí $A_1 = B_1$ a $A_2 = B_2$.
Je to jen dvojice $(\mathbf{4}: X_1 \rightarrow Y_1A_1X_1, \mathbf{2}: A_2 \rightarrow a)$, $(\mathbf{5}: X_1 \rightarrow F_1, \mathbf{2}: A_2 \rightarrow a)$
Poznamenejme, že $Predict(X_1 \rightarrow Y_1A_1X_1) = \{a\}$ a $Predict(X_1 \rightarrow F_1) = \{\$\}$, tedy:
$Predict(X_1 \rightarrow Y_1A_1X_1) \cap Predict(X_1 \rightarrow F_1) = \emptyset$.

Proveďme syntaktickou analýzu řetězce ***aa***:

Zásobník $\boldsymbol{St_1}$	**Vstupní řetězec $\boldsymbol{w}$**	**Vybrané pravidlo $\boldsymbol{p_1}$**	**Zásobník $\boldsymbol{St_2}$**	**Výstupní řetězec $\boldsymbol{w_2}$**	**Přidružené pravidlo $\boldsymbol{p_2}$**
$\$\boldsymbol{S_1}$	$\boldsymbol{aa}\$$	$\mathbf{1}: S_1 \rightarrow aA_1X_1$	$\$\boldsymbol{S_2}$	ε	$\mathbf{1}: S_2 \rightarrow aaX_2$
$\$X_1A_1\boldsymbol{a}$	$\boldsymbol{aa}\$$		$\$\boldsymbol{X_2}$	aa	
$\$X_1\boldsymbol{A_1}$	$\boldsymbol{a}\$$	$\mathbf{2}: A_1 \rightarrow a$	$\$\boldsymbol{X_2}$	aa	$\mathbf{4}: X_2 \rightarrow Y_2A_2A_2X_2$
$\$X_1\boldsymbol{a}$	$\boldsymbol{a}\$$		$\$X_2A_2A_2\boldsymbol{Y_2}$	aa	
$\$\boldsymbol{X_1}$	**$**	$\mathbf{3}: X_1 \rightarrow X_1$	$\$X_2A_2A_2\boldsymbol{Y_2}$	aa	$\mathbf{7}: Y_2 \rightarrow \varepsilon$
$\$\boldsymbol{X_1}$	**$**	$\mathbf{5}: X_1 \rightarrow F_1$	$\$X_2A_2\boldsymbol{A_2}$	aa	$\mathbf{2}: A_2 \rightarrow a$
$\$\boldsymbol{F_1}$	**$**	$\mathbf{8}: F_1 \rightarrow F_1$	$\$X_2\boldsymbol{A_2}$	aaa	$\mathbf{2}: A_2 \rightarrow a$
$\$\boldsymbol{F_1}$	**$**	$\mathbf{9}: F_1 \rightarrow \varepsilon$	$\$\boldsymbol{X_2}$	$aaaa$	$\mathbf{5}: X_2 \rightarrow \varepsilon$
$	**$**		**$**	$aaaa$	

Překlad řetězce *aa* na řetězec *aaaa* proběhl úspěšně, tedy $(aa, aaaa) \in 2\text{-}L(\Gamma)$. Poznamenejme, že $2\text{-}L(\Gamma) = \{(a^{2^n}, a^{2^{n+1}}): n \geq 1\}$. V tomto případě je překladač použit jako pouhý deterministický akceptor jazyka, který není bezkontextový.

6.1.3.3 Definice silného n-cestného deterministického LL n-KGP

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-KGP. Potom řekneme, že Γ je *silný n-cestný deterministický LL n-KGP*, pokud Γ je silný deterministický *LL* n-KGP pro libovolnou vstupní gramatiku s indexem *IN*, kde $1 \leq IN \leq n$.

6.2 Deterministická paralelní syntaktická analýza zdola nahoru pro *n*-KGP

U vstupní gramatiky je provedena syntaktická analýza zdola nahoru pro daný řetězec, přičemž použití daného pravidla při syntaktické analýze řídí generování řetězce u ostatních výstupních gramatik. Jedná se o ***nepřímou simulaci n-KGP***.

6.2.1 Slabý deterministický LR n-KGP

V této kapitole je nadefinován slabý deterministický *LR* n-KGP. Bude se jednat o jistým způsobem omezený n-KGP takový, aby pro něj mohla být provedena syntaktická analýza deterministickým způsobem. Myšlenka je následující: Z hlediska omezení n-KGP bude vyžadováno, aby vstupní gramatika byla *LR*, což zaručí, že bude možno provést deterministickým způsobem syntaktickou analýzu zdola nahoru pro vstupní gramatiku. Dále bude vyžadováno, aby komponenta Q obsahovala pouze takové n-tice pravidel, ve kterých je každé pravidlo ze vstupní gramatiky obsaženo maximálně v jedné n-tici. Tím bude zaručeno, že použití jistého pravidla během syntaktické analýzy jednoznačně určí n-tici pravidel, která bude aplikována i na ostatní gramatiky. Tato kapitola předpokládá, že je čtenář seznámen s problematikou *LR*-gramatik, konstrukcí *LR*-tabulky a s algoritmem, pomocí kterého lze provést *LR*-syntaktickou analýzu. Všechny tyto potřebné pojmy jsou uvedeny například v [1].

6.2.1.1 Definice slabého deterministického LR n-KGP

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-KGP. Potom řekneme, že Γ je *slabý deterministický LR n-KGP* pro vstupní gramatiku s indexem *IN*, kde $1 \leq IN \leq n$, pokud jsou splněny následující podmínky:

1) Gramatika G_{IN} je *LR*-gramatika.
2) Pro libovolné dva prvky $(p_1, p_2, \ldots, p_n) \in Q$ a $(r_1, r_2, \ldots, r_n) \in Q$ platí:
   Pokud $p_{IN} = r_{IN}$, potom také $p_i = r_i$ pro všechna $i = 1,\ldots, n$.

6.2.1.2 *Algoritmus: Syntaktická analýza zdola nahoru pro slabý deterministický LR n-KGP*

- ***Vstup:*** slabý deterministický *LR* n-KGP $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ pro vstupní gramatiku s indexem *IN*; *LR*-tabulka pro vstupní gramatiku; řetězec $w \in T_{IN}^*$, kde T_{IN} je množina terminálních symbolů vstupní gramatiky
- ***Výstup:*** Pokud existuje $(w_1, w_2, \ldots, w_{IN\text{-}1}, w_{IN}, w_{IN+1}, \ldots w_n) \in n\text{-}L(\Gamma)$, přičemž $w_{IN} = w$, pak výstupem je $(w_1, w_2, \ldots, w_{IN\text{-}1}, w_{IN+1}, \ldots, w_n)$, jinak chyba
- ***Metoda:***
  - Nechť $G_i = (N_i, T_i, P_i, S_i)$ pro všechna $i = 1, \ldots, n$;

    $M = \{1, 2, \ldots, n\}$; $M_{IN} = M - \{IN\}$;
  - Pro všechna $i \in M_{IN}$: StackInit(St_i);
  - **repeat**

    Pomocí *LR*-syntaktické analýzy prováděj syntaktickou analýzu pro vstupní řetězec w, dokud není úspěch nebo chyba. Pokud je provedena aktuálně redukce podle pravidla p, potom paralelně proveď následující krok:

    Vyber jednoznačně určenou n-tici $(p_1, p_2, \ldots, p_n) \in Q$, kde $p = p_{IN}$;

    Pro všechna $i \in M_{IN}$:

    $w_{new} := \varepsilon$; Nechť $p_i = A \rightarrow X_1X_2\ldots X_k$:

    for $j := k$ **downto** 1 **do**

    case X_j **of**

    $X_j \in T_j$: $w_{new} := X_jw_{new}$;

    $X_j \in N_j$: $<B, w_{act}> :=$ TopPop(St_i);

    if $X_j = B$ **then** $w_{new} := w_{act}w_{new}$ **else** Chyba;

    Push(St_i, $<A, w_{new}>$);

    until Úspěch **or** Chyba;
  - Pro všechna $i \in M_{IN}$:

    $<B, w_{act}> :=$ TopPop(St_i); **if** $B = S_i$ **and** SEmpty(St_i) **then** $w_i := w_{act}$;

    else Chyba;

6.2.1.3 *Příklad*

Uvažujme následující slabý deterministický *LR* n-KGP $\Gamma = (G_1, G_2, G_3, Q)$ pro vstupní gramatiku s indexem 1, kde:

- $G_1 = (\{E_1, T_1, F_1\}, \{i, +, *, (,)\}, \{$**1**: $E_1 \rightarrow E_1+T_1$, **2**: $E_1 \rightarrow T_1$, **3**: $T_1 \rightarrow T_1*F_1$,
  4: $T_1 \rightarrow F_1$ **5**: $F_1 \rightarrow (E_1)$, **6**: $F_1 \rightarrow i\}, E_1)$,

- $G_2 = (\{E_2, T_2, F_2\}, \{i, +, *\}, \{$**1**: $E_2 \rightarrow +E_2T_2$, **2**: $E_2 \rightarrow T_2$, **3**: $T_2 \rightarrow *T_2F_2$,
  4: $T_2 \rightarrow F_2$ **5**: $F_2 \rightarrow E_2$, **6**: $F_2 \rightarrow i\}, E_2)$,

- $G_3 = (\{E_3, T_3, F_3\}, \{i, +, *\}, \{\mathbf{1}: E_3 \rightarrow E_3T_3+, \mathbf{2}: E_3 \rightarrow T_3, \mathbf{3}: T_3 \rightarrow T_3F_3*,$
  $\mathbf{4}: T_3 \rightarrow F_3\ \mathbf{5}: F_3 \rightarrow E_3, \mathbf{6}: F_3 \rightarrow i\}, E_3),$
- $Q = \{(\boldsymbol{i}, \boldsymbol{i}, \boldsymbol{i}): \mathbf{1} \leq \boldsymbol{i} \leq \mathbf{6}\}$

LR tabulka pro gramatiku G_1:

Stav	$\boldsymbol{i}$	+	*	(	)	$\$$	$\boldsymbol{E_1}$	$\boldsymbol{T_1}$	$\boldsymbol{F_1}$
0	s5			s4			1	2	3
1		s6				acc			
2		r2	s7		r2	r2			
3		r4	r4		r4	r4			
4	s5			s4			8	2	3
5		r6	r6		r6	r6		9	3
6	s5			s4					10
7	s5			s4					
8		s6			s11				
9		r1	s7		r1	r1			
10		r3	r3		r3	r3			
11		r5	r5		r5	r5			

Ukažme, že tento gramatický systém je skutečně slabý deterministický *LR n*-KGP:

1) Vstupní gramatika je *LR*-gramatikou (*LR*-tabulka neobsahuje žádné kolize).

2) Pro libovolné dva prvky $(p_1, p_2, p_3) \in Q$ a $(r_1, r_2, r_3) \in Q$ platí: Pokud $p_1 = r_1$, potom také $p_2 = r_2$ a $p_3 = r_3$

Proveďme syntaktickou analýzu pro řetězec $i*i$:

Zásobník St_1	w	**Vybrané pravidlo p_1**	**Zásobník St_2**	**Přidružené pravidlo p_2**	**Zásobník St_3**	**Přidružené pravidlo p_3**
$<\$,0>$	$i*i\$$	$\alpha[0, i]$ = s5				
$<\$,0><i,5>$	$*i\$$	$\alpha[5, *]$ = r6: $F_1 \rightarrow i$		$F_2 \rightarrow i$		$F_3 \rightarrow i$
$<\$,0><F_1,3>$	$*i\$$	$\alpha[3, *]$ = r4: $T_1 \rightarrow F_1$	$<F_2,i>$	$T_2 \rightarrow F_2$	$<F_3,i>$	$T_3 \rightarrow F_3$
$<\$,0><T_1,2>$	$*i\$$	$\alpha[2, *]$ = s7	$<T_2,i>$		$<T_3,i>$	
$<\$,0><T_1,2><*,7>$	$i\$$	$\alpha[2, i]$ = s5	$<T_2,i>$		$<T_3,i>$	
$<\$,0><T_1,2><*,7><i,5>$	$\$$	$\alpha[5, \$]$ = r6: $F_1 \rightarrow i$	$<T_2,i>$	$F_2 \rightarrow i$	$<T_3,i>$	$F_3 \rightarrow i$
$<\$,0><T_1,2><*,7><F,10>$	$\$$	$\alpha[10, \$]$ = r3: $T_1 \rightarrow T_1*F_1$	$<T_2,i><F_2,i>$	$T_2 \rightarrow *T_2F_2$	$<T_3,i><F_3,i>$	$T_3 \rightarrow T_3F_3*$
$<\$,0><T_1,2>$	$\$$	$\alpha[2, \$]$ = r2: $E_1 \rightarrow T_1$	$<T_2,*ii>$	$E_2 \rightarrow T_2$	$<T_3,ii*>$	$E_3 \rightarrow T_3$
$<\$,0><E_1,1>$	$\$$	$\alpha[1, \$]$ = acc	$<E_2, *ii>$		$<E_3,ii*>$	

Překlad řetězce $i{*}i$ na řetězce $w_2 = {*}ii$, $w_3 = ii{*}$ proběhl úspěšně, tedy $(i{*}i, {*}ii, ii{*}) \in 3\text{-}L(\Gamma)$.
V tomto případě je překladač použit jako multipřekladač, který paralelně překládá zjednodušené aritmetické výrazy do prefixové a postfixové notace.

6.2.1.4 Definice slabého n-cestného deterministického LR n-KGP

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-KGP. Potom řekneme, že Γ je *slabý n-cestný deterministický LR n-KGP*, pokud Γ je slabý deterministický *LR* n-KGP pro libovolnou vstupní gramatiku s indexem *IN*, kde $1 \leq IN \leq n$.

6.3 Deterministická paralelní syntaktická analýza shora dolů pro *n*-HGP

U vstupní gramatiky je provedena syntaktická analýza shora dolů pro daný řetězec, přičemž použití daného pravidla při syntaktické přímým způsobem řídí generování řetězce u ostatních výstupních gramatik. Jedná se tedy o ***přímou simulaci n-HGP***, neboť v každém kroku syntaktické analýzy se právě simuluje jeden derivační krok *n*-HGP.

6.3.1 Slabý deterministický LL n-HGP

V této kapitole je nadefinován slabý deterministický *LL* *n*-HGP. Bude se jednat o jistým způsobem omezený *n*-HGP, jehož vstupní gramatika provádí nejlevější derivace a všechny výstupní gramatiky provádějí obecné derivace. Z hlediska omezení *n*-HGP bude vyžadováno, aby vstupní gramatika byla *LL*, což zaručí, že bude možno provést deterministickým způsobem syntaktickou analýzu shora dolů pro vstupní gramatiku. Dále bude vyžadováno, aby komponenta *Q* obsahovala pouze takové *n*-tice pravidel, ve kterých je pravidlo ze vstupní gramatiky obsaženo maximálně v jednom prvku. Tímto bude zaručeno, že použití jistého pravidla během syntaktické analýzy jednoznačně určí *n*-tici pravidel, která bude aplikována i na ostatní gramatiky. Poslední omezení bude pro vygenerované multiformy: Každá vygenerovaná multiforma musí splňovat podmínku, že v každé její části se musí konkrétní nonterminální symbol vyskytovat maximálně jedenkrát. Tím bude zaručeno, že daná *n*-tice pravidel půjde na danou multiformu aplikovat maximálně jedním způsobem.

6.3.1.1 Definice slabého deterministického LL n-HGP

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-HGP, $G_i = (N_i, T_i, P_i, S_i)$ je bezkontextová gramatika pro všechna $i = 1, \ldots, n$. Potom řekneme, že Γ je *slabý deterministický LL n-HGP* pro vstupní gramatiku s indexem *IN*, kde $1 \leq IN \leq n$, ve kterém gramatika G_{IN} generuje větné formy v nejlevější derivace a ostatní gramatiky v obecné derivaci, pokud jsou splněny následující podmínky:

1) Gramatika G_{IN} je *LL* gramatika.
2) Pro libovolné dva prvky $(p_1, p_2, \ldots, p_n) \in Q$ a $(r_1, r_2, \ldots, r_n) \in Q$ platí:
   Pokud $p_{IN} = r_{IN}$, potom také $p_i = r_i$ pro všechna $i = 1,\ldots, n$.
3) Pro libovolnou multiformu $(x_1, x_2, \ldots, x_n)$, kde $x_i \in (T_i \cup N_i)^*$ splňující podmínku $(S_1, S_2, \ldots, S_n) \Rightarrow^* (x_1, x_2, \ldots, x_n)$ platí: Pro všechna $i = 1,\ldots, n$: pro všechna $A \in N_i$: A se vyskytuje v x_i nejvýše jedenkrát.

6.3.1.2 Algoritmus: Syntaktická analýza shora dolů pro slabý deterministický LL n-HGP

- ***Vstup:*** slabý deterministický LL n-HGP $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ pro vstupní gramatiku s indexem IN; LL-tabulka pro vstupní gramatiku; řetězec $w \in T_{IN}^*$, kde T_{IN} je množina terminálních symbolů vstupní gramatiky
- ***Výstup:*** Pokud existuje $(w_1, w_2, \ldots, w_{IN-1}, w_{IN}, w_{IN+1}, \ldots w_n) \in n\text{-}L(\Gamma)$, přičemž $w_{IN} = w$, pak výstupem je $(w_1, w_2, \ldots, w_{IN-1}, w_{IN+1}, \ldots, w_n)$, jinak chyba
- ***Metoda:***
    - Nechť $G_i = (N_i, T_i, P_i, S_i)$ pro všechna $i = 1, \ldots, n$;
      $M = \{1, 2, \ldots, n\}$; $M_{IN} = M - \{IN\}$;
    - Pro všechna $i \in M_{IN}$: $w_i := S_i$;
    - **repeat**

      Pomocí prediktivní syntaktické analýzy prováděj syntaktickou analýzu pro vstupní řetězec w, dokud není úspěch nebo chyba. Pokud je provedena aktuálně expanze podle pravidla p, potom paralelně proveď následující krok:

      Vyber jednoznačně určenou n-tici $(p_1, p_2, \ldots, p_n) \in Q$, kde $p = p_{IN}$;

      Pro všechna $i \in M_{IN}$:

      Nechť p_i je ve tvaru $A_i \rightarrow x_i$:

      if $w_i = u_iA_iv_i$ **then** $w_i = u_ix_iv_i$ **else** Chyba;

      until Úspěch **or** Chyba;
    - **if** existuje $i \in M_{IN}$, pro které platí $w_i \notin T_i^*$ **then** Chyba;

6.3.1.3 Příklad

Uvažujme následující slabý deterministický LL n-HGP $\Gamma = (G_1, G_2, Q)$ pro vstupní gramatiku s indexem 1, kde:

- $G_1 = (\{<\mathrm{start}_1>, <\mathrm{regs}_1>, <\mathrm{reg}_1>, <\mathrm{src}_1>, <\mathrm{dest}_1>\}, \{\mathrm{ADD, SUB, AX, BX, CX, DX}\},$

  $\{$**1:** $<\mathrm{start}_1> \rightarrow \mathrm{ADD}<\mathrm{regs}_1>$, **2:** $<\mathrm{start}_1> \rightarrow \mathrm{SUB}<\mathrm{regs}_1>$,

  3: $<\mathrm{regs}_1> \rightarrow <\mathrm{src}_1>\ <\mathrm{dest}_1>$, **4:** $<\mathrm{src}_1> \rightarrow <\mathrm{reg}_1>$, **5:** $<\mathrm{dest}_1> \rightarrow <\mathrm{reg}_1>$,

  6: $<\mathrm{reg}_1> \rightarrow \mathrm{AX}$, **7:** $<\mathrm{reg}_1> \rightarrow \mathrm{BX}$, **8:** $<\mathrm{reg}_1> \rightarrow \mathrm{CX}$, **9:** $<\mathrm{reg}_1> \rightarrow \mathrm{DX}\}, <\mathrm{start}_1>)$
- $G_2 = (\{<\mathrm{start}_2>, <\mathrm{regs}_2>, <\mathrm{reg}_2>, <\mathrm{src}_2>, <\mathrm{dest}_2>\}, \{0, 1\},$

  $\{$**1:** $<\mathrm{start}_2> \rightarrow 000<\mathrm{regs}_2>$, **2:** $<\mathrm{start}_2> \rightarrow 001<\mathrm{regs}_2>$,

  3: $<\mathrm{regs}_2> \rightarrow <\mathrm{dest}_2>\ <\mathrm{src}_2>$, **4:** $<\mathrm{src}_2> \rightarrow <\mathrm{reg}_2>$, **5:** $<\mathrm{dest}_2> \rightarrow <\mathrm{reg}_2>$,

  6: $<\mathrm{reg}_2> \rightarrow 00$, **7:** $<\mathrm{reg}_2> \rightarrow 01$, **8:** $<\mathrm{reg}_2> \rightarrow 10$, **9:** $<\mathrm{reg}_2> \rightarrow 11\}, <\mathrm{start}_2>)$
- $Q = \{(\boldsymbol{i}, \boldsymbol{i}, \boldsymbol{i}): \mathbf{1} \leq \boldsymbol{i} \leq \mathbf{9}\}$

Dále uvažujme vstupní řetězec ADD AX BX. Následující tabulka ilustruje překlad tohoto řetězce pomocí předchozího algoritmu. Pro zjednodušení není v tabulce prováděna celá prediktivní syntaktická analýza pro vstupní řetězec, ale pouze je uvedena sekvence pravidel, kterou tato syntaktická analýza vyprodukuje.

Vybrané pravidlo p_1	**Přidružené pravidlo p_2**	**Hodnota řetězce w_2**
		<start$_2$>
1: <start$_1$> $\rightarrow$ ADD<regs$_1$>	**1:** <start$_2$> $\rightarrow$ 000<regs$_2$>	000<regs$_2$>
3: <regs$_1$> $\rightarrow$ <src$_1$> <dest$_1$>	**3:** <regs$_2$> $\rightarrow$ <dest$_2$> <src$_2$>	000<dest$_2$> <src$_2$>
4: <src$_1$> $\rightarrow$ <reg$_1$>	**4:** <src$_2$> $\rightarrow$ <reg$_2$>	000<dest$_2$> <reg$_2$>
6: <reg$_1$> $\rightarrow$ AX	**6:** <reg$_2$> $\rightarrow$ 00	000<dest$_2$>00
5: <dest$_1$> $\rightarrow$ <reg$_1$>	**5:** <dest$_2$> $\rightarrow$ <reg$_2$>	000<reg$_2$>00
7: <reg$_1$> $\rightarrow$ BX	**7:** <reg$_2$> $\rightarrow$ 01	0000100

Překlad řetězce ADD AX BX na řetězec w_2 = 0000100 proběhl úspěšně, tedy
(ADD AX BX, 0000100) $\in$ 2-$L(\Gamma)$.

6.3.1.4 Definice slabého n-cestného deterministického LR n-KGP

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-HGP. Potom řekneme, že Γ je *slabý n-cestný deterministický LL n-HGP*, pokud Γ je slabý deterministický *LL n*-KGP pro libovolnou vstupní gramatiku s indexem *IN*, kde $1 \leq IN \leq n$.

6.4 Deterministická paralelní syntaktická analýza zdola nahoru pro *n*-HGP

U vstupní gramatiky je provedena syntaktická analýza zdola nahoru pro daný řetězec, přičemž použití daného pravidla při syntaktické analýze řídí generování řetězce u ostatních výstupních gramatik. Jedná se o ***nepřímou simulaci n-HGP***.

6.4.1.1 Definice slabého deterministického LR n-HGP

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-HGP, $G_i = (N_i, T_i, P_i, S_i)$ je bezkontextová gramatika pro všechna $i = 1, \ldots, n$. Potom řekneme, že Γ je *slabý deterministický LR n-HGP* pro vstupní gramatiku s indexem *IN*, kde $1 \leq IN \leq n$, ve kterém gramatika G_{IN} generuje větné formy v nejpravější derivace a ostatní gramatiky v obecné derivaci, pokud jsou splněny následující podmínky:

1) Gramatika G_{IN} je *LR*-gramatika.
2) Pro libovolné dva prvky $(p_1, p_2, \ldots, p_n) \in Q$ a $(r_1, r_2, \ldots, r_n) \in Q$ platí: Pokud $p_{IN} = r_{IN}$, potom také $p_i = r_i$ pro všechna $i = 1,\ldots, n$.
3) Pro libovolnou multiformu $(x_1, x_2, \ldots, x_n)$, kde $x_i \in (T_i \cup N_i)^*$ splňující podmínku $(S_1, S_2, \ldots, S_n) \Rightarrow^* (x_1, x_2, \ldots, x_n)$ platí: Pro všechna $i = 1,\ldots, n$: pro všechna $A \in N_i$: A se vyskytuje v x_i nejvýše jedenkrát.

6.4.1.2 Algoritmus Syntaktická analýza zdola nahoru pro slabý deterministický LR n-HGP

- ***Vstup:*** slabý deterministický LR n-HGP $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ pro vstupní gramatiku s indexem IN; LR-tabulka pro vstupní gramatiku; řetězec $w \in T_{IN}^*$, kde T_{IN} je množina terminálních symbolů vstupní gramatiky
- ***Výstup:*** Pokud existuje $(w_1, w_2, \ldots, w_{IN-1}, w_{IN}, w_{IN+1}, \ldots w_n) \in n\text{-}L(\Gamma)$, přičemž $w_{IN} = w$, pak výstupem je $(w_1, w_2, \ldots, w_{IN-1}, w_{IN+1}, \ldots, w_n)$, jinak chyba
- ***Metoda:***
  - Nechť $G_i = (N_i, T_i, P_i, S_i)$ pro všechna $i = 1, \ldots, n$;

    $M = \{1, 2, \ldots, n\}$; $M_{IN} = M - \{IN\}$;
  - Pro všechna $i \in M_{IN}$: Pro všechna $A \in N_i$: $X_{<i, A>} :=$ *nedefinováno*;
  - **<u>repeat</u>**

    Pomocí LR-syntaktické analýzy prováděj syntaktickou analýzu pro vstupní řetězec w, dokud není úspěch nebo chyba. Pokud je provedena aktuálně redukce podle pravidla p, potom paralelně proveď následující krok:

    Vyber jednoznačně určenou n-tici $(p_1, p_2, \ldots, p_n) \in Q$, kde $p = p_{IN}$;

    Pro všechna $i \in M_{IN}$:

      Nechť p_i je ve tvaru $A_i \rightarrow x_i$:

      Potom polož: $X_{<i, Ai>} := h_i(x_i)$, kde h_i je homomorfismus z $N_i \cup T_i$ do T_i definovaný následovně:

      $h(a) = a$ pro všechna $a \in T_i$; $h(B) = X_{<i, B>}$ pro všechna $B \in N_i$;

      Pro všechna $B \in N_i - \{A_i\}$, kde B je obsaženo v x_i, polož:

      $X_{<i, B>} :=$ *nedefinováno*;

    <u>until</u> Úspěch **<u>or</u>** Chyba;
  - Pro všechna $i \in M_{IN}$: $w_i := X_{<i, Si>}$
  - **<u>if</u>** existuje $i \in M_{IN}$, pro které platí $w_i \notin T_i^*$ **<u>then</u>** Chyba;

6.4.1.3 Příklad

Uvažujme slabý deterministický LR n-HGP $\Gamma = (G_1, G_2, Q)$ pro vstupní gramatiku s indexem 1, který je identický s n-HGP Γ z příkladu 6.3.1.3.

Dále uvažujme vstupní řetězec ADD AX BX. Následující tabulka ilustruje překlad tohoto řetězce pomocí předchozího algoritmu. Pro zjednodušení není v tabulce prováděna celá LR-syntaktická analýza pro vstupní řetězec, ale pouze je uvedena sekvence pravidel, kterou tato syntaktická analýza vyprodukuje. Neboť tento systém obsahuje pouze jednu výstupní gramatiku, nehraje index této gramatiky žádnou roli, a proto je tento index u pomocných proměnných vynechán.

Vybrané pravidlo p_1	**Přidružené pravidlo p_2**	**Hodnoty jednotlivých proměnných**
		$X_{<\mathrm{start}>}$ = *nedefinováno* $X_{<\mathrm{src}>}$ = *nedefinováno* $X_{<\mathrm{regs}>}$ = *nedefinováno* $X_{<\mathrm{dest}>}$ = *nedefinováno* $X_{<\mathrm{reg}>}$ = *nedefinováno*
6: <reg> → AX	**6:** <reg> → 00	$X_{<\mathrm{start}>}$ = *nedefinováno* $X_{<\mathrm{src}>}$ = *nedefinováno* $X_{<\mathrm{regs}>}$ = *nedefinováno* $X_{<\mathrm{dest}>}$ = *nedefinováno* $X_{<\mathrm{reg}>}$ = 00
4: <src> → <reg>	**4:** <src> → <reg>	$X_{<\mathrm{start}>}$ = *nedefinováno* $X_{<\mathrm{src}>}$ = 00 $X_{<\mathrm{regs}>}$ = *nedefinováno* $X_{<\mathrm{dest}>}$ = *nedefinováno* $X_{<\mathrm{reg}>}$ = *nedefinováno*
7: <reg> → BX	**7:** <reg> → 01	$X_{<\mathrm{start}>}$ = *nedefinováno* $X_{<\mathrm{src}>}$ = 00 $X_{<\mathrm{regs}>}$ = *nedefinováno* $X_{<\mathrm{dest}>}$ = *nedefinováno* $X_{<\mathrm{reg}>}$ = 01
5: <dest> → <reg>	**5:** <dest> → <reg>	$X_{<\mathrm{start}>}$ = *nedefinováno* $X_{<\mathrm{src}>}$ = 00 $X_{<\mathrm{regs}>}$ = *nedefinováno* $X_{<\mathrm{dest}>}$ = 01 $X_{<\mathrm{reg}>}$ = *nedefinováno*
3: <regs> → <src> <dest>	**3:** <regs> → <dest> <src>	$X_{<\mathrm{start}>}$ = *nedefinováno* $X_{<\mathrm{src}>}$ = *nedefinováno* $X_{<\mathrm{regs}>}$ = 0100 $X_{<\mathrm{dest}>}$ = *nedefinováno* $X_{<\mathrm{reg}>}$ = *nedefinováno*
1: <start> → ADD <regs>	**1:** <start> → 000 <regs>	$X_{<\mathrm{start}>}$ = 0000100 $X_{<\mathrm{src}>}$ = *nedefinováno* $X_{<\mathrm{regs}>}$ = *nedefinováno* $X_{<\mathrm{dest}>}$ = *nedefinováno* $X_{<\mathrm{reg}>}$ = *nedefinováno*

Překlad řetězce ADD AX BX na řetězec w_2 = 0000100 proběhl úspěšně, tedy (ADD AX BX, 0000100) $\in$ 2-$L(\Gamma)$.

6.4.1.4 Definice slabého n-cestného deterministického LR n-HGP

Nechť $\Gamma = (G_1, G_2, \ldots, G_n, Q)$ je n-HGP. Potom řekneme, že Γ je *slabý n-cestný deterministický LR n-HGP*, pokud Γ je slabý deterministický *LR n*-HGP pro libovolnou vstupní gramatiku s indexem *IN*, kde $1 \leq IN \leq n$.

7 Využití *n*-HGP pro projekt LISSOM

Projekt Lissom se zabývá vývojem jazyka pro popis instrukční sady a architektury, generováním softwarových částí procesoru a jejich simulací. Základem je tedy navržený jazyk ISAC, který vychází z jazyka LISA. Jazyk ISAC je nadstavbou nad libovolným vyšším programovacím jazykem, např. C/C++. Tento jazyk má dvě oddělené části, které mohou existovat samostatně:

- definici zdrojů a hardwarové části procesoru,
- definici operační části a chování.

V další kapitole se budeme zabývat pouze druhou částí. V této části je popsána instrukční sada assembleru a její binární kódování pro daný procesor. Popis této instrukční sady pomocí jazyka ISAC je převeden na ekvivalentní hybridní multigenerativní gramatický systém, který byl zavedený v páté kapitole. Na bázi tohoto modelu mohou být automaticky vygenerovány dva překladače:

- assembler (provádí překlad assemblerovské instrukce do binárního kódu),
- disassembler (provádí překlad z binárního kódu instrukce do assemblerovského formátu).

Jádrem obou překladačů je syntaktický analyzátor pracující na bázi algoritmu 6.4.1.2.

7.1 Popis operační části jazyka ISAC

Na nejvyšší úrovni popisu operační části jsou dvě základní deklarace, a to deklarace skupiny a deklarace operace.

- Deklarace skupiny sdružuje operace s podobným významem. Tvar zápisu je následující:

  "**<u>GROUP</u>**" *group_id* "=" *operation_id* { "," *operation_id* }

  - *group_id* je identifikátor skupiny
  - *operation_id* je identifikátor operace nebo skupiny

- Deklarace operace popisuje její formu v textové a binární podobě a její chování. Tvar zápisu je následující:

  "**<u>OPERATION</u>**" *operation_id* "{" [**`instances`** ";"]
`assembler` ";"
`coding` ";"
[**`behavior`** ";"] "}"

  - *operation_id* je identifikátor operace.

Nyní si rozeberme jednotlivé sekce, které obsahuje tělo operace, tedy sekce *instances*, *assembler*, *coding* a *behavior*.

- Sekce **`instances`** má tvar:

  "<u>**INSTANCE**</u>" **`inst`** { "," **`inst`** }

  sekce **`inst`** má tvar:

  instance_id ["**ALIAS**" "**{**" *alias_id* { "**,**" *alias_id* } "**}**"]

    - *instance_id* je identifikátor jiné operace nebo skupiny
    - *alias_id* je identifikátor aliasu *instance_id*

- Sekce **`assembler`** má tvar:

  "<u>**ASSEMBLER**</u>" "**{**" {*instance_id* | *text* | **`attribute`**} ["**{**" *semantic_action* "**}**"] "**}**"

  - *instance_id* je identifikátor operace, skupiny nebo aliasu
  - *text* je textová konstanta
  - *semantic_action* je sémantická akce spojená se sekcí **`atribute`**

  - Sekce **`attribute`** má tvar:

    attr_left_id "=" *attr_type* ["=" *attr_right_id*]

    - *attr_left_id* je levý identifikátor atributu
    - *attr_type* určuje typ atributu a může nabývat hodnot "**#U**" pro číslo bez znaménka a "**#S**" pro číslo se znaménkem
    - *attr_rigth_id* je pravý identifikátor atributu

- Sekce **`coding`** má tvar:

  "<u>**CODING**</u>" "**{**" {*instances_id* | *binary* | **`attribute`**} ["**{**"se*mantic_action* "**}**"] "**}**"

  - *instances_id* je identifikátor operace, skupiny nebo aliasu
  - *binary* je binární konstanta
  - *semantic_action* je sémantická akce spojená se sekcí **`attribute`**

  - Sekce **`attribute`** má tvar:

    attr_left_id "=" *attr_type* ["=" *attr_right_id*]

    - *attr_left_id* je levý identifikátor atributu
    - *attr_type* určuje typ atributu a může nabývat hodnot "**0bx[*k*]**" pro číslo bez znaménka a "**0bsx[*k*]**" pro číslo se znaménkem, kde *k* označuje bitové pole o *k* bitech.
    - *attr_rigth_id* je pravý identifikátor atributu

Sekce **`behavior`** je sekvence akcí popsaná v jazyku C, které vyjadřují chování operace. Tato sekce je využívána pro simulaci chování procesoru.

7.1.1 *Algoritmus: Převod popisu překladu v jazyce ISAC na 2-HGP*

- ***Vstup:*** program v jazyce ISAC
- ***Výstup:*** 2-HGP $\Gamma = (G_1, G_2, Q)$
- ***Metoda:***
  - Nechť $G_i = (N_i, T_i, P_i, S_i)$ pro $i = 1, 2$.
  - $N_1 := \emptyset$, $N_2 := \emptyset$; $T_1 := \{num\}$, $T_2 := \{0, 1\}$, $P_1 := \emptyset$, $P_2 := \emptyset$;

    S_1 := identifikátor hlavní skupiny, S_2 := identifikátor hlavní skupiny, $Q := \emptyset$;
  - Pro každou deklaraci skupiny ***B*** tvaru "**GROUP *B* = *A*₁, *A*₂, …, *A*ₙ**":

    přidej $B, A_1, A_2, \ldots, A_n$ do N_1 i do N_2;

    přidej $B \to A_1, B \to A_2, \ldots, B \to A_n$ do P_1 i do P_2;

    přidej $(B \to A_1, B \to A_1), (B \to A_2, B \to A_2),\ldots,(B \to A_n, B \to A_n)$ do Q;
  - Pro každou deklaraci operace ***X*** tvaru

    "**OPERATION *X* { INSTANCE *I*; ASSEMBLER *A*; CODING *C*; … }**":

    Nechť ***I*** je ve tvaru "$\boldsymbol{I_1, I_2, \ldots, I_n}$". Potom pro každou instanci $\boldsymbol{I_i}$, kde $i = 1,\ldots, n$:

    <u>if</u> $\boldsymbol{I_i}$ je ve tvaru "***B***" **<u>then</u>**

    přidej ${}_XB$ do N_1 i do N_2;

    přidej ${}_XB \to B$ do P_1 i do P_2;

    přidej $({}_XB \to B, {}_XB \to B)$ do Q;

    <u>if</u> $\boldsymbol{I_i}$ je ve tvaru "**$\boldsymbol{B}$ ALIAS $\boldsymbol{\{A_1, A_2, \ldots, A_m\}}$**" **<u>then</u>**

    přidej ${}_XA_1, {}_XA_2, \ldots, {}_XA_m$ do N_1 i do N_2;

    přidej ${}_XA_1 \to B, {}_XA_2 \to B, \ldots, {}_XA_m \to B$ do P_1 i do P_2;

    přidej $({}_XA_1 \to B, {}_XA_1 \to B), ({}_XA_2 \to B, {}_XA_2 \to B), \ldots,$
    $({}_XA_m \to B, {}_XA_m \to B)$ do Q;

    Nechť je ***A*** ve tvaru "$\{A_1A_2\ldots A_r\}$", kde $r \geq 1$.

    položme ${}_XA = \{A_1, A_2, \ldots, A_r\}$;

    přidej všechna $a \in {}_XA$, která jsou konstantním textem, do množiny T_1.

    Nechť h_A je homomorfismus z ${}_XA$ do $N_1 \cup T_1$ definovaný následovně:

    $h_A(a) = a$ pro všechna $a \in {}_XA$, které jsou konstantním textem,

    $h_A(a) = {}_Xa$ pro všechna $a \in {}_XA$, které jsou identifikátory operace nebo skupiny,

    $h_A(a) =$ "num" pro všechna $a \in {}_XA$, které jsou atributy.

    Nechť je ***C*** ve tvaru "$\{C_1C_2\ldots C_s\}$", kde $s \geq 1$.

    položme ${}_XC = \{C_1, C_2, \ldots, C_s\}$;

přidej všechna $c \in {}_XC$, které jsou binárními konstantami, do množiny T_2.

Všechna $c \in {}_XC$, které jsou bitovým polem, reprezentuj terminálním symbolem "b_k" a tuto reprezentaci přidej do množiny T_2.

Nechť h_C je homomorfismus z ${}_XC$ do $N_2 \cup T_2$ definovaný následovně:

$h_C(c) = c$ pro všechna $c \in {}_XC$, které jsou binárními konstantami,

$h_C(c) = {}_Xc$ pro všechna $c \in {}_XC$, které jsou identifikátory operace nebo skupiny,

$h_C(c) =$ "b_k" pro všechna $c \in {}_XC$, které jsou bitovým polem o délce k, kde $k \geq 1$.

Potom:

přidej $X \rightarrow h_A(A_1A_2 \ldots A_r)$ do P_1;

přidej $X \rightarrow h_C(C_1C_2 \ldots C_s)$ do P_2;

přidej $(X \rightarrow h_A(A_1A_2 \ldots A_r), X \rightarrow h_C(C_1C_2 \ldots C_s))$ do Q;

7.1.2 Příklad

Uvažujme následující kód programu popsaný v jazyce ISAC:

```
GROUP alu = add, move
GROUP register = ax, bx

OPERATION add {
 INSTANCE register ALIAS { dest, src };
 ASSEMBLER { "ADD" dest "," src ","  attr=#U };
 CODING { 0b0001 dest src attr=0bx[4] };
}

OPERATION move {
 INSTANCE register ALIAS { dest, src };
 ASSEMBLER { "MOVE" dest "," src };
 CODING { 0b0011 dest src };
}

OPERATION ax {
 ASSEMBLER { "AX" };
 CODING { 0b01 };
}

OPERATION bx {
 ASSEMBLER { "BX" };
 CODING { 0b10 };
}
```

2-cestný 2-HGP Γ vygenerovaný předchozím algoritmem je ve tvaru

$\Gamma = ((N_1, T_1, P_1, S_1), (N_2, T_2, P_2, S_2), Q)$, kde:

- $N_1 = N_2$ = {<alu>, <add>, <move>, <register>, <ax>, <bx>, <$_{\text{add}}$dest>, <$_{\text{add}}$src>, <$_{\text{move}}$dest>, <$_{\text{move}}$src>},
- T_1 = {“ADD”, “MOVE”, “,”, “AX”, “BX”, “num”}
- T_2 = {“0”, “1”, “b_4”},
- P_1 = { **1:** <alu> → <add>, **2:** <register> → <ax>, **3:** <register> → <bx>,
  4: <$_{\text{add}}$dest> → <register>, **5:** <$_{\text{add}}$src> → <register>,
  6: <add> → “ADD” <$_{\text{add}}$dest> “,”<$_{\text{add}}$src> “,” “num”
  7: <$_{\text{move}}$dest> → <register>, **8:** <$_{\text{move}}$src> → <register>,
  9: <move> → “MOVE” <$_{\text{move}}$dest> “,” <$_{\text{move}}$src>,
  10: <ax> → “AX”, **11:** <bx> → “BX” }

- P_2 = { **1:** <alu> → <add>, **2:** <register> → <ax>, **3:** <register> → <bx>,
  4: <$_{\text{add}}$dest> → <register>, **5:** <$_{\text{add}}$src> → <register>,
  6: <add> → “0” “0” “0” “1” <$_{\text{add}}$dest> <$_{\text{add}}$src> “b_4”
  7: <$_{\text{move}}$dest> → <register>, **8:** <$_{\text{move}}$src> → <register>,
  9: <move> → “0” “0” “1” “1” <$_{\text{move}}$dest> <$_{\text{move}}$src>
  10: <ax> → “0” “1”, **11:** <bx> → “1” “0” }
- $S_1 = S_2$ = <alu>

- $Q = \{(\boldsymbol{i}, \boldsymbol{i}, \boldsymbol{i}): \mathbf{1} \leq \boldsymbol{i} \leq \mathbf{11}\}$

Poznámka:

Aby mohly být z 2-HGP vytvořeny překladače assembleru a disassembleru, jejichž jádro by bylo založeno na algoritmu 6.4.1.2, je třeba vyžadovat, aby výsledný 2-HGP byl 2-cestný. Z konstrukce 2-HGP podle předchozího algoritmu je možné formálně dokázat, že body 2) a 3) z definice 6.4.1.1 jsou splněny automaticky pro libovolně zvolenou vstupní gramatiku, pouze není zaručeno splnění podmínky první, která vyžaduje, aby libovolně zvolená vstupní gramatika byla *LR*-gramatika. Opět je ale možné dokázat, že vždy existuje transformace, která obě gramatiky převede na ekvivalentní *LR*-gramatiky.

8 Závěr

V této disertační práci byly zavedeny různé typy multigenerativních gramatických systémů. U těchto multigenerativních gramatických systémů byla zkoumána generativní síla především v závislosti na:

- typu derivace, která je v systému povolena (nejlevější derivace, nejpravější derivace, obecná derivace),
- typu gramatického systému (zda se provádí kontrola n-ticí nonterminálů nebo n-ticí pravidel),
- typu zvolené operace, která je provedena s n-ticí řetězců,
- počtu komponent (počtu bezkontextových gramatik), ze kterých se gramatický systém skládá.

Podle těchto kritérií byly multigenerativní gramatické systémy rozděleny do následujících skupin:

Kanonické multigenerativní gramatické systémy

Typickým rysem této skupiny bylo, že v každé gramatice byla prováděna pouze nejlevější derivace. V závislosti na typu provedení kontroly se tyto systémy dále dělily na:

1) Kanonické n-generativní nonterminálově synchronizované gramatické systémy (n-KGN)
2) Kanonické n-generativní pravidlově synchronizované gramatické systémy (n-KGP)

Pro tyto typy systémů byla dokázána následující tvrzení:

1) Oba druhy těchto systémů jsou vzájemně převoditelné, tj. pro každý n-KGN existuje ekvivalentní n-KGP a naopak.
2) Počet komponent obou druhů systému lze redukovat na dvě, aniž by se snížila generativní síla těchto MGS, a to v libovolném módu z módů sjednocení, konkatenace a výběr řetězce z první komponenty.
3) Generativní síla obou druhů systému je ekvivalentní síle Turingova stroje, a to v libovolném módu z módů sjednocení, konkatenace a výběr řetězce z první komponenty.

Obecné multigenerativní gramatické systémy

Typickým rysem této skupiny bylo, že v každé gramatice byla prováděna obecná derivace. V závislosti na typu provedení kontroly se tyto systémy dále dělily na:

1) Obecné n-generativní nonterminálově synchronizované gramatické systémy (n-OGN)
2) Obecné n-generativní pravidlově synchronizované gramatické systémy (n-OGP)

Pro tyto typy systémů byla dokázána následující tvrzení:

1) Oba druhy těchto systémů jsou vzájemně převoditelné, tj. pro každý *n*-OGN existuje ekvivalentní *n*-OGP a naopak.
2) Počet komponent obou druhů systému lze redukovat na dvě, aniž by se snížila generativní síla těchto MGS, a to v libovolném módu z módů sjednocení, konkatenace a výběr řetězce z první komponenty.
3) Generativní síla obou druhů systému je ekvivalentní generativní síle maticových gramatik, a to v libovolném módu z módů sjednocení, konkatenace a výběr řetězce z první komponenty.

Hybridní multigenerativní gramatické systémy

Základní vlastností těchto gramatických systémů bylo, že v některých gramatikách byla prováděna obecná derivace, v jiných nejlevější derivace. Tyto systémy se ukázaly vhodné pro praxi.

Dále se disertační práce zabývala praktickým využitím těchto multigenerativních gramatických systémů. Pomocí *n*-tic řetězců byla definována *n*-nární relace, tedy jakýsi zobecněný překlad mezi více jazyky. Tento multipřeklad byl prováděn pomocí gramatického systému následujícím způsobem: V dané *n*-tici gramatik byla jedna gramatika zvolena jako vstupní a ostatních $n-1$ gramatik jako výstupní. Pomocí vstupní gramatiky se provedla syntaktická analýza vstupního řetězce, která v tomto gramatickém systému řídila generování $n-1$ výstupních řetězců pomocí výstupních gramatik. V této části byly navrženy algoritmy, které pro speciální typy kanonických a hybridních multigenerativních gramatických systémů provedly multipřeklad, a to jak metodou shora dolů, tak metodou zdola nahoru.

V příkladech bylo ukázáno, že zavedení těchto speciálních překladačů má přínos jak teoretický, tak praktický:

Teoretický přínos: Pouhou kooperující činností dvou bezkontextových gramatik může být deterministickým způsobem provedena syntaktická analýza složitých kontextových jazyků, například $L_1 = \{a^n b^n c^n : n \geq 0\}$ nebo $L_2 = \{ a^{2^n} : n \geq 1\}$.

Praktický přínos: Tyto systémy mohou sloužit jako formální model pro popis překladu, na jehož bázi lze deterministickým a paralelním přístupem provést syntaktickou analýzu. Tohoto modelu bylo využito například při implementaci projektu LISSOM.

9 Literatura

[1] Beneš, M., Hruška, T, Češka, M.: Překladače, Brno: VUT, 1994.

[2] Csuhaj-Varju, E., Dassow, J., Kelemen, J., Paun, Gh.: Grammar Systems: A Grammatical Approach to Distribution and Cooperation, Gordon and Breach, London, 1994.

[3] Dassow, J., Paun, Gh., and Rozenberg, G.: Grammar Systems, Handbook of Formal Languages, Rozenberg, G. and Salomaa, A. (eds.), Volumes 2, Springer, Berlin, 1997.

[4] Harrison, Michael, A.: Introduction to Formal Language Theory. Addison-Wesley, London, 1978.

[5] Lukáš, R.: General Parsing: A New Approach. Proceedings of the International Conference and Competition, Brno, CZ, FEKT VUT, 2004.

[6] Lukáš, R.: Nový přístup k obecné syntaktické analýze. Proceedings of 9th Conference and Competition, Brno, CZ, FEKT VUT, 2004.

[7] Lukáš, R.: Multigenerative Grammar Systems. Proceedings of the 11th Conference, Brno, CZ, 2005.

[8] Lukáš, R.: Power of Multigenerative Grammar Systems. Proceedings of the 12th Conference, Brno, CZ, 2006.

[9] Lukáš, R., Hruška, T., Kolář, D., Masařík, K.: Two-Way Deterministic Translation and Its Usage in Practice. Proceedings of 8th Spring International Conference - ISIM'05, Ostrava, CZ, MARQ, 2005.

[10] Lukáš, R., Meduna, A.: Multigenerative Grammar Systems. Pre-proceedings 1st Doctoral Workshop on Mathematical and Engineering Methods in Comupter Science (MEMICS), 2005.

[11] Masařík, K., Hruška, T., Kolář, D., Lukáš R.: System for design and simulation of microprocessors. Proceedings of 8th Spring International Conference - ISIM'05, Ostrava, CZ, MARQ, 2005.

[12] Meduna, A.: Automata and Languages: Theory and Applications. Springer, London, 2000.

[13] Meduna, A.: Two-Way Metalinear PC Grammar Systems and Their Descriptional Complexity. Acta Cybernetica, 2003.

[14] Meduna, A., Lukáš, R.: A Note on Iteratively Extendable Strings. Rostocker mathematisches kolloquium, Rostock, DE, 2005.

[15] Meduna, A., Lukáš, R.: Multigenerative Grammar Systems. Schedae Informaticae, Krakov, PL, 2006. (v tisku)

[16] Meduna, A., Lukáš, R.: Multigenerative Grammar Systems. Proceedings of 1st International Workshop - WFM, Přerov, CZ, MARQ, 2006.

[17] Paun, Gh., Salomaa, A. and S. Vicolov, S.: On the generative capacity of parallel communicating grammar systems. International Journal of Computer Mathematics 45, 45-59, 1992.

[18] Salomaa, A.: Formal Languages. Academic Press, New York, 1973.

[19] Vaszil, G.: On simulating Non-returning PC grammar systems with returning systems. Theoretical Computer Science (209) 1-2, 319-329, 1998.