VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

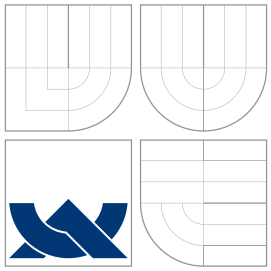
SYSTÉMY FORMÁLNÍCH MODELŮ A JEJICH APLIKACE

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

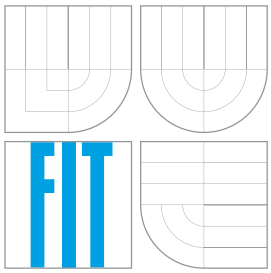
AUTOR PRÁCE
AUTHOR

MARTIN ČERMÁK

BRNO 2008



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

SYSTÉMY FORMÁLNÍCH MODELŮ A JEJICH APLIKACE

SYSTEMS OF FORMAL MODELS WITH APPLICATIONS

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — MARTIN ČERMÁK
AUTHOR

VEDOUCÍ PRÁCE — Prof. RNDr. ALEXANDER MEDUNA, CSc.
SUPERVISOR

BRNO 2008

Abstrakt

Tato práce pojednává o automatových systémech jako o novém způsobu zpracování formálních jazyků. V textu budou zmíněny čtyři modely. První z nich pracuje v sekvenčním módu. V jednom okamžiku počítá jediná komponenta systému. Druhý z nich pracuje v částečně paralelním módu. Zde během jednoho výpočetního kroku aktivně pracují buďto všechny, nebo pouze jedna komponenta systému. V posledních dvou modelech každý automat zpracovává svůj vlastní vstupní řetězec, přičemž jeho výpočet je řízen stavy, resp. přechody ostatních komponent. Stavy, resp. pravidla přechodů komponent mohou zapříčinit i tzv. blokaci, nebo odblokování dílčích automatů.

Klíčová slova

Konečný automat, zásobníkový automat, gramatické systémy, automatové systémy, multipřijímací automatové systémy

Abstract

This paper introduces and discusses automata systems as a new way for formal languages processing. In the text there are four models described. The first model works on sequential mode. At one computation step only one of components works. The second one works on semi-parallel mode. At the one computation step either one or all the components of the automata system work. In the last two models each component of the automata system has its own input string. The computation step of each component is influenced by their states, or used rules. The state, or used rule of the components of automata system can block or unblock some or all automata of the system.

Keywords

Finish automata, push-down automata, grammar systems, automata systems, multirecognizers automata systems

Citace

Martin Čermák: Systémy formálních modelů a jejich aplikace, diplomová práce, Brno, FIT VUT v Brně, 2008

Systémy formálních modelů a jejich aplikace

Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně pod vedením pana Prof. RNDr. Alexandera Meduny, CSc.

.....
Martin Čermák
14. května 2008

Poděkování

Na tomto místě bych rád poděkoval prof. RNDr. Alexandru Medunovi, CSc., vedoucímu diplomové práce, za ochotu, čas a odbornou pomoc při konzultacích diplomové práce. Dále bych zde chtěl poděkovat Ing. Zbyňku Křivkovi, Ph.D., za čas věnovaný recenzi mého článku úzce souvisejícího s touto prací, čímž také nepřímo přispěl ke zkvalitnění této diplomové práce.

Obsah

Kapitola 1

Úvod

Teorie formálních modelů a aplikací se zabývá specifikací tříd formálních jazyků. Tyto třídy jsou dány například typem gramatik, nebo konkrétním typem automatů, pomocí nichž jsme schopni třídu formálních jazyků popsat. Z pravidla platí, že čím je gramatika složitější, tím rozsáhlejší třídu formálních jazyků je schopna vygenerovat. Se složitostí typu gramatických pravidel ale roste i komplexnost typu automatů, kterými jsme schopni shodnou třídu formálních jazyků definovat, což v důsledku souvisí i se složitostí implementace, spotřebou zdrojů, apod. Tento jev je důvodem, proč se věnovat vyvíjení různých vylepšení jednoduchých gramatik za účelem rozšíření tříd jazyků, jež je gramatika schopna generovat.

Jednou z metod zvyšování efektivity popisu jazyků je použití více jednoduchých gramatik pracujících sekvenčně, nebo současně. Tyto gramatiky zde vytvářejí entity, tzv. komponenty. Souhrn takových komponent pak tvoří gramatický systém. Zvýšení generativní síly, při minimálním zvýšení náročnosti popisu jazyků pomocí gramatik však není jediným pozitivem gramatických systémů. Gramatické systémy lze efektivně využít například k přepínané syntaktické analýze [8]. I s gramatickými systémy jsou v souvislosti implementační problémy. Formální modely, které budou sloužit právě k usnadnění implementace gramatických systémů, budou hlavním tématem této práce.

Nejdříve budou zavedeny základní pojmy a definice o něž se bude následující text pevně opírat. V první řadě budou uvedeny jazyky, řetězce a základní operace nad nimy. V souvislosti s tím pak i základní typy gramatik. Následovat bude podsekce zmiňující gramatické systémy a za ní sekce týkající se práce „rozpoznávačů“.

Navazující text bude pojednávat o nově zpracovávanému tématu a to o automatových systémech. Stejně jako u gramatických systémů bude se i zde nejdříve mluvit o sekvenčním módu spolupráce, kdy v jeden okamžik bude aktivní pouze jedna komponenta systému. Takto pracujícímu automatovému systému budeme říkat sekvenční automatový systém. Přepínání mezi komponentami bude prováděno například po provedení předem stanoveného počtu kroků nad daným automatem.

Dále budou popsány paralelní systémy čistě zásobníkových automatů, kdy budou v jeden okamžik výpočet provádět buď všechny komponenty současně (nad společným vstupním řetězcem), nebo pouze jedna komponenta. Tím se docílí stavu, kdy více automatů provádí výpočet „ovlivňující se“ pouze vstupním řetězcem. Příslušnost vstupního řetězce do jazyka je pak určena přijetím řetězce jednotlivými automaty.

Poslední z metod bude uveden multipřijímací automatový systém. Tento systém je nejdůmyslnějším a nejsilnějším z uvedených systémů. To je dáno tím, že jednotlivé kroky každého automatu jsou přímo určující pro ostatní automaty. Dalším posilujícím prvkem je zde fakt, že každý z automatů pracuje nad svým vlastním vstupním řetězcem. Mul-

tipřijímací systém automatů může být řízen n-ticí stavů, nebo n-ticí použitých přechodových pravidel. Jazyky definované tímto způsobem budeme označovat spíše jako multijazyky, jejichž způsob přijetí multipřijímacím systémem je další potřebnou informací k jeho definici.

V posledních sekcích se text věnuje použití multipřijímacích systémů jako analyzátorů C/E systémů, přičemž ukazuje na jejich využitelnost v C/E Petriho sítích, což se nakonec neukázalo jako velice šťastné řešení.

Téma práce je moderní oblastí zpracování formálních jazyků, která si klade za důsledek veliký potenciál využití i v jiných oblastech informačních technologií. Výhodám, nevýhodám, otevřeným problémům a nejpravděpodobnějším budoucím rozšířením bude věnován závěr diplomové práce.

Kapitola 2

Základní definice a pojmy

Tato kapitola obsahuje nejzákladnější definice a pojmy týkající se teorie formálních jazyků, gramatik, automatů a základní formy gramatických systémů, které budou důležité pro pochopení následujícího textu. Pro podrobnější a hlavně úplné informace doporučuji zejména [5], [2], nebo [10]. Další podpůrné literatury budou odkazovány průběžně.

2.1 Využívané množiny

Definice 2.1.0.1 (Množina přirozených čísel)
Množinu přirozených čísel, zapsáno $\mathbb{N}$, definujeme jako $\mathbb{N} = \{0, 1, 2, 3, \ldots\}$.

Definice 2.1.0.2 (Množina kladných celých čísel)
Nechť $\mathbb{N}$ je množinou přirozených čísel. Pak definujeme množinu kladných celých čísel, zapsáno[1] $\mathbb{N}^+$, jako $\mathbb{N}^+ = \mathbb{N}\backslash\{0\}$, nebo-li $\mathbb{N}^+ = \{1, 2, 3, \ldots\}$.

Definice 2.1.0.3 (Intervalová množina I)
Intervalovou množinu I definujeme jako neprázdnou konečnou množinu $I = \{1, \ldots, n\}$ pro nějaké $n \geq 1$, danou posloupností kladných celých čísel.

Definice 2.1.0.4 (Intervalová množina $I(m)$)
Intervalovou množinu $I(m)$, kde $m \geq 1$, definujeme jako neprázdnou konečnou množinu $I(m) = \{1, \ldots, m\}$ danou posloupností kladných celých čísel.

Definice 2.1.0.5 (Kardinalita množiny)
Nechť A je množina. Pak definujeme kardinalitu množiny $card(A)$ jako zobrazeni $card(A) : A \rightarrow \mathbb{N}$ jako:

- $card(A) = 0$ pro $A = \emptyset$,
- $card(A) = n$ pro $A = \{a_1, \ldots a_n\}$, kde $n > 1$ a
- $card(A) = \infty$ pro $A = \{a_1, a_2, a_3, \ldots\}$.

[1] Někdy se můžeme také setkat se značením $\mathbb{Z}^+$.

2.2 Definice základních pojmů

Definice 2.2.0.6 (Abeceda)
Abeceda je neprázdná konečná množina prvků, které se nazývají symboly.

Definice 2.2.0.7 (Řetězec)
Nechť Σ je abeceda, pak

- ε značí *prázdný řetězec*, nad abecedou Σ.[2]
- Jestliže x je řetězec nad abecedou Σ a $a \in \Sigma$, pak i ax je řetězcem nad abecedou Σ.[3]

Definice 2.2.0.8 (Délka řetězce)
Nechť x je řetězec nad abecedou Σ. *Délku řetězce* x pak značíme $|x|$ a definujeme ji následovně:

- Pokud $x = \varepsilon$, pak $|x| = 0$
- Pokud $x = a_1 \dots a_n$, kde $\forall i \in \{1 \dots, n\}$ pro nějaké $n \geq 1$: $a_i \in \Sigma$, potom $|x| = n$.

Definice 2.2.0.9 (Konkatenace dvou řetězců)
Nechť x a y jsou dva řetězce nad abecedou Σ. *Konkatenací řetězců* x a y,zapsáno $x \cdot y$, získáme řetězec xy. Řetězec y čistě připojíme za řetězec y. Binární operace konkatenace je asociativní.

Definice 2.2.0.10 (Opačný, reverzní, řetězec)
Nechť x je řetězcem nad abecedou Σ. Pak *opačný řetězec*, nebo-li *reverzní řetězec*, značíme $reverse(x)$ a definujeme jej následovně:

- Pokud $x = \varepsilon$, pak $reverse(x) = \varepsilon$
- Pokud $x = a_1 a_2 a_3 \dots a_n$, pak $reverse(x) = a_n a_{n-1} a_{n-2} \dots a_2 a_1$.

Definice 2.2.0.11 (Prefix řetězce)
Nechť x a y jsou dva řetězce nad abecedou Σ. Pak nazveme řetězec x *prefixem řetězce* y, pokud existuje takový řetězec z nad abecedou Σ, pro který platí $xz = y$.

Definice 2.2.0.12 (Sufix řetězce)
Nechť x a y jsou dva řetězce nad abecedou Σ. Pak nazveme řetězec x *sufixem řetězce* y, pokud existuje takový řetězec z nad abecedou Σ, pro který platí $zx = y$.

Definice 2.2.0.13 (Podřetězec)
Nechť x a y jsou dva řetězce nad abecedou Σ. Pak nazveme řetězec x *podřetězcem řetězce* y, pokud existují takové řetězce z a z' nad abecedou Σ, pro které platí $zxz' = y$.

Definice 2.2.0.14 (Formální jazyk)
Nechť je dána abeceda Σ. Pak množinu L pro niž platí $L \subseteq \Sigma^*$ nazveme *formálním jazykem nad abecedou* Σ.

[2] Prázdným řetězcem rozumíme řetězec, který neobsahuje žádný symbol.
[3] Symbolem Σ^* budeme chápat množinu všech řetězců nad abecedou Σ.

Definice 2.2.0.15 (Překlad)
Nechť Σ a Ω jsou dvě abecedy. Symbolem Σ budeme označovat *vstupní abecedu* a symbolem Ω *abecedu výstupní*. Překladem jazyka L_{in} do jazyka L_{out} nazveme libovolnou relaci τ z L_{in} do L_{out}.[4] Jazyk L_{in} nazveme jazykem vstupním a L_{out} jazykem výstupním. Pokud pro řetězce x a y platí, že $y \in \tau(x)$, pak řekneme, že řetězec y je výstupem pro řetězec x.

2.3 Operace nad jazyky

Definice 2.3.0.16 (Sjednocení dvou jazyků)
Nechť L_1 a L_2 jsou formální jazyky nad abecedou Σ. Pak definujeme operaci *sjednocení dvou jazyků*, $L_1 \cup L_2$, následovně:

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

Definice 2.3.0.17 (Průnik dvou jazyků)
Nechť L_1 a L_2 jsou formální jazyky nad abecedou Σ. Pak definujeme binární operaci *průnik dvou jazyků*, $L_1 \cap L_2$, následovně:

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

Definice 2.3.0.18 (Konkatenace dvou jazyků)
Nechť L_1 a L_2 jsou formální jazyky nad abecedou Σ. Pak definujeme binární operaci *konkatenace dvou jazyků*, $L_1 \cdot L_2$, následovně:

$$L_1 \cdot L_2 = \{xy | x \in L_1 \wedge y \in L_2\}$$

Definice 2.3.0.19 (Rozdíl dvou jazyků)
Nechť L_1 a L_2 jsou jazyky nad abecedou Σ. Pak definujeme *rozdíl dvou jazyků*, $L_1 - L_2$, následovně:

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

Definice 2.3.0.20 (Doplněk jazyka)
Nechť L je formální jazyk nad abecedou Σ. Pak definujeme *doplněk jazyka*, $\bar{L}$, následovně:

$$\bar{L} = \{x | x \in \Sigma^* \wedge x \notin L\}$$

Definice 2.3.0.21 (Mocnina jazyka)
Nechť L je formální jazyk nad abecedou Σ. Pak definujeme *i-tou mocninu jazyka*, L^i, následovně:

- $L^0 = \{\varepsilon\}$
- $L^i = L \cdot L^{i-1}$

Definice 2.3.0.22 (Iterace jazyka)
Nechť L je formální jazyk nad abecedou Σ. Pak definujeme *iteraci jazyka*, L^*, následovně:

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

[4] Překlad je definován obecně jako relace. V praxi je často ovšem touto relací zobrazení, neboť je žádoucí, aby pro každý řetězec $x \in L_{in}$ existoval právě jeden řetězec $y \in L_{out}$.

Definice 2.3.0.23 (Pozitivní iterace jazyka)
Nechť L je formální jazyk nad abecedou Σ. Pak definujeme *pozitivní iteraci jazyka*, L^+, následovně:

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

2.4 Základní typy gramatik

V následující sekci budou zmíněny různé typy gramatik. Ty se liší především ve tvaru pravidel, přičemž právě tvar pravidel udává generativní schopnost odpovídajícího typu gramatiky. Ta bude uvedena v závěru sekce.

2.4.1 Neomezená gramatika

Obecná gramatika obsahuje nejobecnější tvar gramatických pravidel. Třídu jazyků, jež lze generovat nějakou neomezenou gramatikou nazveme třídou *rekurzivně vyčíslitelných jazyků*, která je také často označována jako *třída jazyků typu 0*.

Definice 2.4.1.1 (Neomezená gramatika)
Neomezená gramatika je definována jako čtveřice $G = (N, T, P, S)$, kde:

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

Definice 2.4.1.2 (Přímá derivace u neomezených gramatik)
Nechť $G = (N, T, P, S)$ je neomezená gramatika, dále nechť $u, v \in (N \cup T)^*$, $r = x \to y \in P$, kde $x = (N \cup T)^* N (N \cup T)^*$ a $y \in (N \cup T)^*$. Pak řekneme, že uxv *přímo derivuje* uyv podle pravidla r a zapisujeme $uxv \Rightarrow uyv[r]$, nebo zjednodušeně $uxv \Rightarrow uyv$.

Definice 2.4.1.3 (Sekvence derivací u neomezených gramatik)
Nechť $G = (N, T, P, S)$ je neomezená gramatika. Dále:

- Nechť $u \in (N \cup T)^*$. Pak řekneme, že u *derivuje* u *v nula krocích* a zapisujeme $u \Rightarrow^0 u[\varepsilon]$, nebo zjednodušeně $u \Rightarrow^0 u$.
- Nechť $u_0, u_1, \ldots, u_n \in (N \cup T)^*$ a nechť $\forall i = 1, 2, \ldots, n$ platí $u_{i-1} \Rightarrow u_i[r_i]$. Pak řekneme, že u_0 *derivuje* u v n krocích a zapisujeme $u_0 \Rightarrow^n u_n[r_1 r_2 \ldots r_n]$, nebo zkráceně $u_0 \Rightarrow^n u_n$.
- Nechť $u \Rightarrow^n u'[\pi]$, kde $u, u' \in (N \cup T)^*$, $\pi = r_1 r_2 \ldots r_n$ a $n \geq 1$, pak řekneme, že u *netriviálně derivuje* u' a zapisujeme $u \Rightarrow^+ u'[\pi]$, nebo zjednodušeně $u \Rightarrow^+ u'$.
- Nechť $u \Rightarrow^n u'[\pi]$, kde $u, u' \in (N \cup T)^*$, $\pi = r_1 r_2 \ldots r_n$ a $n \geq 0$, pak řekneme, že u *derivuje* u' a zapisujeme $u \Rightarrow^* u'[\pi]$, nebo zjednodušeně $u \Rightarrow^* u'$.

Definice 2.4.1.4 (Větná forma v neomezené gramatice)
Nechť $G = (N, T, P, S)$ je neomezená gramatika. Pokud $S \Rightarrow^* u$, kde $u \in (N \cup T)^*$, řekneme, že u je *větná forma v neomezené gramatice* G.

Definice 2.4.1.5 (Jazyk generovaný neomezenou gramatikou)
Nechť $G = (N, T, P, S)$ je neomezená gramatika, pak definujeme formální jazyk generovaný neomezenou gramatikou G, zapsáno $L(G)$, jako

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

2.4.2 Kontextová gramatika

Kontextová gramatika je speciálním případem neomezené gramatiky, kdy pro pravidla $x \to y$ navíc přibyla vlastnost $|x| \leq |y|$. Množinu všech formálních jazyků generovatelných nějakou kontextovou gramatikou nazýváme *třídou kontextových jazyků*, které jsou často označovány za *jazyky typu 1*.

Definice 2.4.2.1 (Kontextová gramatika)
Neomezená gramatika je definována jako čtveřice $G = (N, T, P, S)$, kde:

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

Definice 2.4.2.2 (Přímá derivace, sekvence derivací a jazyk generovaný kontextovou gramatikou)
Definice derivačního kroku, sekvence derivačních kroků a jazyku generovaného kontextovou gramatikou jsou identické s příslušnými definicemi u neomezených gramatik.

2.4.3 Bezkontextová gramatika

Bezkontextová gramatika je speciálním případem neomezené gramatiky, kdy pro pravidla $x \to y$ platí, že symbol x zastupuje pouze jeden nonterminální symbol. Množinu všech formálních jazyků generovatelných nějakou bezkontextovou gramatikou nazýváme *třídou bezkontextových jazyků*, které jsou často označovány za *jazyky typu 2*.

Definice 2.4.3.1 (Bezkontextová gramatika)
Bezkontextová gramatika je definována jako čtveřice $G = (N, T, P, S)$, kde:

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

Definice 2.4.3.2 (Přímá derivace u bezkontextových gramatik)
Nechť $G = (N, T, P, S)$ je bezkontextová gramatika a nechť $u, v \in (N \cup T)^*$, $r = A \to x \in P$, kde $A \in N$ a $x \in (N \cup T)^*$. Pak řekneme, že uAv *přímo derivuje* uxv podle pravidla r a zapisujeme $uAv \Rightarrow uxv[r]$, nebo zjednodušeně $uAv \Rightarrow uxv$.

Definice 2.4.3.3 (Nejlevější derivace u bezkontextových gramatik)
Nechť $G = (N, T, P, S)$ je bezkontextová gramatika, dále nechť $u \in T^*; v \in (N \cup T)^*$, $r = A \to x \in P$, kde $A \in N$ a $x \in (N \cup T)^*$. Pak řekneme, že uAv *přímo derivuje* v *nejlevější derivaci* uxv podle pravidla r a zapisujeme $uAv \Rightarrow_{lm} uxv[r]$, nebo zjednodušeně $uAv \Rightarrow_{lm} uxv$.

Definice 2.4.3.4 (Nejpravější derivace u bezkontextových gramatik)
Nechť $G = (N, T, P, S)$ je bezkontextová gramatika, dále nechť $v \in T^*; u \in (N \cup T)^*$, $r = A \to x \in P$, kde $A \in N$ a $x \in (N \cup T)^*$. Pak řekneme, že uAv *přímo derivuje* v *nejpravější derivaci* uxv podle pravidla r a zapisujeme $uAv \Rightarrow_{rm} uxv[r]$, nebo zjednodušeně $uAv \Rightarrow_{rm} uxv$.

Definice 2.4.3.5 (Sekvence derivací u bezkontextových gramatik[5])
Nechť $G = (N, T, P, S)$ je bezkontextová gramatika. Dále:

- Nechť $u \in (N \cup T)^*$. Pak řekneme, že u *derivuje* u *v nula krocích* a zapisujeme $u \Rightarrow^0 u[\varepsilon]$, nebo zjednodušeně $u \Rightarrow^0 u$.
- Nechť $u_0, u_1, \ldots, u_n \in (N \cup T)^*$ a nechť $\forall i = 1, 2, \ldots, n$ platí $u_{i-1} \Rightarrow u_i[r_i]$. Pak řekneme, že u_0 *derivuje* u v n krocích a zapisujeme $u_0 \Rightarrow^n u_n[r_1 r_2 \ldots r_n]$, nebo zkráceně $u_0 \Rightarrow^n u_n$.
- Nechť $u \Rightarrow^n u'[\pi]$, kde $u, u' \in (N \cup T)^*$, $\pi = r_1 r_2 \ldots r_n$ a $n \geq 1$, pak řekneme, že u *netriviálně derivuje* u' a zapisujeme $u \Rightarrow^+ u'[\pi]$, nebo zjednodušeně $u \Rightarrow^+ u'$.
- Nechť $u \Rightarrow^n u'[\pi]$, kde $u, u' \in (N \cup T)^*$, $\pi = r_1 r_2 \ldots r_n$ a $n \geq 0$, pak řekneme, že u *derivuje* u' a zapisujeme $u \Rightarrow^* u'[\pi]$, nebo zjednodušeně $u \Rightarrow^* u'$.

Definice 2.4.3.6 (Jazyk generovaný bezkontextovou gramatikou)
Nechť $G = (N, T, P, S)$ je neomezená gramatika, pak definujeme formální jazyk generovaný bezkontextovou gramatikou G, zapsáno $L(G)$, jako

$$\begin{aligned} L(G) &= \{\omega | \omega \in T^* \wedge S \Rightarrow^* \omega\} \\ &= \{\omega | \omega \in T^* \wedge S \Rightarrow^*_{lm} \omega\} \\ &= \{\omega | \omega \in T^* \wedge S \Rightarrow^*_{rm} \omega\}. \end{aligned}$$

2.4.4 Pravá lineární gramatika

Pravá lineární gramatika je speciálním případem bezkontextové gramatiky, kdy pro pravidla $A \to x$ platí, že symbol x je řetězcem samých terminálních symbolů, nebo terminálních symbolů zakončených pouze jedním nonterminálním symbolem. Množinu všech formálních

[5] Definice rozšiřuje přímou derivaci na sekvenci přímých derivací. Sekvence nejlevějších, resp. nejpravějších derivací by se definovala analogicky pomocí nejlevějších, resp. nejpravějších přímých derivací.

jazyků generovatelných nějakou pravou lineární gramatikou nazýváme *třídou regulárních jazyků*, které jsou často označovány za *jazyky typu 3*.

Definice 2.4.4.1 (Pravá lineární gramatika)
Bezkontextová gramatika je definována jako čtveřice $G = (N, T, P, S)$, kde:

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

Definice 2.4.4.2 (Přímá derivace, sekvence derivací a jazyk generovaný pravou lineární gramatikou)
Přímá derivace, sekvence derivací a jazyk generovaný pravou lineární gramatikou jsou identické s příslušnými definicemi u bezkontextové gramatiky.

2.4.5 Věta o Chomského hierarchii jazyků

Označme symboly po řadě $L_{REG}, L_{CF}, L_{CS}, L_{RE}$ třídu regulárních, bezkontextových, kontextových a rekurzivně vyčíslitelných jazyků. Pak mezi těmito třídami platí vztah

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

2.5 Gramatické systémy

Gramatické systémy (GS) jsou systémy vzájemně kooperujících gramatik generujících řetězce v sekvenčním, paralelním, či semiparalelním módu, přičemž spolu jednotlivé gramatiky jednoduchým způsobem komunikují. Podle zmiňovaného módu rozlišujeme gramatické systémy na *CD(cooperating distributed)* gramatické systémy a *PC(parallel communicating)* gramatické systémy (viz Kooperující formální modely [6]). V následující sekci jsou uvedeny pouze základní definice CD a PC gramatických systému sloužící pouze pro nástin problematiky. Pro bližší informace doporučuji shlédnout příslušnou odkazovanou literaturu uvnitř podkapitoly.

2.5.1 CD gramatické systémy

CD gramatické systémy pracují sekvenčně. Skládají se z konečného počtu komponent, které zde tvoří gramatiky. V jednom derivačním kroku provede derivaci pouze jedna komponenta systému. Přepínání mezi komponentami se děje například na základě absence gramatického pravidla pro následující derivační krok v právě aktivní gramatice, nebo po provedení konstantního předem definovaného maximálního počtu derivačních kroků apod. (viz [3]). Řízení přepínání lze jednoduše provádět i na základě tzv. přepínacích pravidel uvnitř jednotlivých komponent systému (viz [8]). Stanovená musí být opět i podmínka ukončení výpočtu. To se může dít absencí derivačního pravidla u všech komponent systému, nebo u právě aktivní komponenty, pokud jsou komponenty přepínané např. přepínacími gramatickými pravidly, jak je tomu u Univerzálních gramatických systémů, které sem z větší části spadají (viz kapitola 5.2 [9]).

Definice 2.5.1.1 (CD gramatický systém)
CD gramatický systém je $(n+3)$–tice

$$\Gamma = (N, T, S, P_1, \ldots, P_n),$$

kde:

- N, resp. T, je abeceda nonterminálních, resp. terminálních symbolů, přičemž platí $N \cap T = \emptyset$,
- $S \in N$ je startující nonterminál,
- $P_1, P_2, \ldots, P_n$ jsou množiny derivačních pravidel a nazýváme je komponentami gramatického systému.[6]

Definice 2.5.1.2 (Derivace v CD gramatickém systému)
Nechť $\Gamma = (N, T, S, P_1, \ldots, P_n)$ je CD gramatický systém.

- Pro každé $i = 1, 2, \ldots, n$, *zakončovací derivace* i-tou komponentou, zapsáno $\Rightarrow^t_{P_i}$, je definována jako

$$x \Rightarrow^t_{P_i} y \text{ iff } x \Rightarrow^*_{P_i} y \text{ přičemž neexistuje žádné } z \in \Sigma^* \text{ takové, že } y \Rightarrow_{P_i} z.$$

- Pro každé $i = 1, 2, \ldots, n$, *k-kroková derivace* i-tou komponentou, zapsáno $\Rightarrow^{=k}_{P_i}$, je definována jako

$$x \Rightarrow^{=k}_{P_i} y \text{ iff existují } x_1, \ldots, x_{k+1} \text{ a pro každé } j = 1, \ldots, k,\ x_j \Rightarrow_{P_i} x_{j+1}.$$

- Pro každé $i = 1, 2, \ldots, n$, *nejvíce k-kroková derivace* i-tou komponentou, zapsáno $\Rightarrow^{\leq k}_{P_i}$, je definována jako

$$x \Rightarrow^{\leq k}_{P_i} y \text{ iff } x \Rightarrow^{=k'}_{P_i} y \text{ pro nějaké } k' \leq k.$$

- Pro každé $i = 1, 2, \ldots, n$, *nejméně k-kroková derivace* i-tou komponentou, zapsáno $\Rightarrow^{\geq k}_{P_i}$, je definována jako

$$x \Rightarrow^{\geq k}_{P_i} y \text{ iff } x \Rightarrow^{=k'}_{P_i} y \text{ pro nějaké } k' \geq k.$$

Definice 2.5.1.3 (Jazyk generovaný CD gramatickým systémem)
Nechť $\Gamma = (N, T, S, P_1, \ldots, P_n)$ je CD gramatický systém a nechť $f \in D$ je derivační mód gramatického systému, kde $D = \{*, t\} \cup \{\leq k, = k, \geq k \| k \in \mathbb{N}^+\}$. Pak *jazyk generovaný CD automatovým systémem* je definován jako

$$L_f(\Gamma) = \{\omega \in T^* | S \Rightarrow^f_{P_{i_1}} \omega_1 \Rightarrow^f_{P_{i_2}} \ \ldots \Rightarrow^f_{P_{i_m}} \omega_m = \omega, m \geq 1, 1 \leq j \leq m, 1 \leq i_j \leq n\}.$$

[6] Někdy je možné se setkat s definicí $\Gamma = (N, T, S, G_1, \ldots, G_n)$, kde $G_i = (N, T, S, P_i) \forall i = 1, 2, \ldots, n$. To je ale totéž.

2.5.2 PC gramatické systémy

PC gramatické systémy pracují naopak v módu paralelním. Každá z komponent systému, což je opět gramatika, má svou větnou formu nad níž provádí své derivační kroky. Řízení se provádí pomocí speciálních komunikačních (query) symbolů. Ty určují místa, do kterých bude příslušná větná forma vložena. Jazykem celého systému je jazyk první komponenty (viz [6],[3]). Query symboly ale nemusí nutně sloužit jen jako vstupní body větných forem. Mohou sloužit i jako průběžný „kontrolor“, kdy každá z komponent má svou větnou formu nad kterou provádí derivační kroky řízené právě množinou query symbolů. Výsledný jazyk pak není nutně pouze jazyk první komponenty, ale může to být i jazyk vzniklý konkatenací, či sjednocením jazyků jednotlivých gramatik. To už je ale řeč o Multigenerativních gramatických systémech (viz [4, 7]).

Definice 2.5.2.1 (PC gramatický systém)
PC gramatický systém je $(n+3)$–tice

$$\Gamma = (N, Q, T, (S_1, P_1), \ldots, (S_n, P_n)),$$

kde:

- N, resp. T, je abeceda nonterminálních, resp. terminálních symbolů, přičemž platí $N \cap T = \emptyset$,
- $Q = \{Q_1, Q_2, \ldots, Q_n\}$ je množina tzv. *query symbolů*, přičemž $N \cap T \cap Q = \emptyset$ a index i mapuje P_i na i–tou komponentu systému,
- P_i $\forall i = 1, \ldots, n$ je množina přepisovacích pravidel přes $N \cup T \cup Q$ a nazýváme ji komponentami systému,[7]
- $S_i \in N$ $\forall i = 1, \ldots, n$ je startující nonterminální symbol.

Definice 2.5.2.2 (Výpočet PC gramatického systému)
Nechť $\Sigma_\Gamma = N \cup T \cup Q$ a nechť $\Gamma = (N, Q, T, (S_1, P_1), \ldots, (S_n, P_n))$ je PC gramatický systém. Pro dvě n–tice $(x_1, x_2, \ldots, n)$, $(y_1, y_2, \ldots, y_n)$, s $x_i, y_i \in \Sigma_\Gamma$ $\forall i = 1, 2, \ldots, n$, kde $x_i \notin T^*$, píšeme $(x_1, x_2, \ldots, n) \Rightarrow (y_1, y_2, \ldots, y_n)$ pokud platí jedna z možností:

1. $\forall i = 1, 2, \ldots, n$, $|x_i|_Q = 0$[8] a dále $x_i \Rightarrow y_i$ je pravidlo v P_i, nebo $x_i = y_i \in T^*$.
2. Existuje $i = 1, 2, \ldots, n$ takové, že $|x_i|_Q > 0$ a $x_i = z_1 Q_{i_1} z_2 Q_{i_2} \ldots z_t Q_{i_t} z_{t+1}$ pro všechna i a pro $z_j \in (N \cup T)^*$, $1 \leq j \leq t+1$. Pokud $|x_{i_j}|_Q = 0$, pro všechna $j, 1 \leq j \leq t$, pak $y_i = z_1 x_{i_1} z_2 x_{i_2} \ldots z_t x_{i_t} z_{t+1}$ a $y_{i_j} = S_{i_j}, 1 \leq j \leq t$. Jestliže pro nějaké $j = 1, 2, \ldots, t, |x_{i_j}| \neq 0$, pak $y_i = x_i$. Pro všechna $i = 1, 2, \ldots, n$ taková, že y_i neodpovídá žádné z předchozích specifikací, máme $x_i = y_i$.

Definice 2.5.2.3 (Jazyk generovaný PC gramatickým systémem)
Nechť $\Gamma = (N, Q, T, (S_1, P_1), \ldots, (S_n, P_n))$ je PC gramatický systém. Pak definujeme *jazyk generovaný PC gramatickým systémem* jako

$$L(\Gamma) = \{x \in T^* | (S_1, S_2, \ldots, S_n) \Rightarrow^* (x, \alpha_2, \ldots, \alpha_n),\ \alpha_i \in \Sigma_\Gamma^*, \forall i = 2, 3, \ldots, n\}.$$

[7] Někdy je možné se setkat s definicí $\Gamma = (N, Q.T, G_1, \ldots, G_n)$, kde $G_i = (N \cup Q, T, S_i, P_i) \forall i = 1, 2, \ldots, n$, nebo i dokonce $\Gamma = (Q, G_1, \ldots, G_n)$, kde $G_i = (N_i \cup Q, T_i, S_i, P_i) \forall i = 1, 2, \ldots, n$ bez existující relace mezi T_i a $N_j, i \neq j$.

[8] Zápisem $|x_i|_Q$ je myšlen počet query symbolu v řetězci x_i.

2.5.3 Kanonické multigenerativní gramatické systémy

Kanonické multigenerativní gramatické systémy se skládají z n-tice bezkontextových gramatik. Samotná derivace se provádí tak, že je paralelně aplikováno derivační pravidlo na aktuální větnou formu každé komponenty systému. Těchto n derivací je kontrolováno buď pomocí n-tice nonterminálů, ze kterých byla derivace provedena, nebo pomocí n-tice použitých derivačních pravidel. V textu budeme uvažovat pouze první z případů. Podrobné informace včetně následujících definicí naleznete v [4]. Po ukončení výpočtu je vygenerována n-tice řetězců, nad kterými jsou provedeny operace, čímž je pak definován výsledný jazyk.

Definice 2.5.3.1 (Kanonický n-generativní nonterminálově synchronizovaný GS)
Kanonický n-generativní nonterminálově synchronizovaný GS (n-KGN) je $(n+1)$-tice

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

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

Definice 2.5.3.2 (Multiforma)
Nechť $\Gamma = (G_1, \ldots, Gn, Q)$ je n-KGN. Potom *multiforma* je n-tice $\chi = (x_1, \ldots, xn)$, kde $x_i \in (T_i \cup N_i)^*$ pro všechna $i = 1, \ldots, n$.

Definice 2.5.3.3 (Přímý derivační krok v n-KGN)
Nechť $\Gamma = (G_1, \ldots, Gn, Q)$ je n-KGN, nechť $\chi = (u_1 A_1 v_1, \ldots, u_n A_n v_n)$, $\chi' = (u_1 x_1 v_1, \ldots, u_n x_n v_n)$, 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 $A_i \to x_i \in P_i$ pro všechna $i = 1, \ldots, n$ a $(A_1, \ldots, A_n) \in Q$. Pak říkáme, že χ *přímo derivuje* χ' a zapisujeme $\chi \Rightarrow \chi'$.

Definice 2.5.3.4 (Sekvence derivačních kroků v n-KGN)
Nechť $\Gamma = (G_1, \ldots, Gn, Q)$ je n-KGN.

- Nechť χ je multiforma. Pak říkáme, že χ *derivuje v* 0*-krocích* χ a zapisujeme $\chi \Rightarrow^0 \chi$.
- Nechť $\chi_0, \ldots, \chi_k$ 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 $\chi \Rightarrow^k \chi'$ pro nějaké $k \geq 1$, kde χ, χ' jsou multiformy. Pak říkáme, že χ *netriviálně derivuje* χ' a zapisujeme $\chi \Rightarrow^+ \chi'$.
- Nechť $\chi \Rightarrow^k \chi'$ pro nějaké $k \geq 0$, kde χ, χ' jsou multiformy. Pak říkáme, že χ *derivuje* χ' a zapisujeme $\chi \Rightarrow^* \chi'$.

Definice 2.5.3.5 (n-jazyk generovaný n-KGN)
Nechť $\Gamma = (G_1, \ldots, G_n, Q)$ je n-KGN. Potom *n-jazyk generovaný* Γ (budeme značit n-$L(\Gamma)$) je definován:

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

Definice 2.5.3.6 (Jazyky generované módem sjednocení, konkatenace a první komponenty) Nechť $\Gamma = (G_1, \ldots, G_n, Q)$ je n-KGN. Potom definujeme:

- *jazyk generovaný* Γ *v módu sjednocení* (budeme značit $L_{union}(\Gamma)$) je definován:

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

- *jazyk generovaný* Γ *v módu konkatenace* (budeme značit $L_{conc}(\Gamma)$) je definován:

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

- *jazyk generovaný* Γ *v módu první komponenty* (budeme značit $L_{first}(\Gamma)$) je definován:

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

Zástupné symboly využívané pro definici gramatických systémů jsou často používané při definicích automatů, přičemž přirozeně nesou odlišný význam. Pokud budou GS a automaty v jednom kontextu, budou tyto symboly mezi sebou speciálně rozlišeny.

2.6 Základní typy automatů

Automat je *stroj*, často imaginární, který se z pravidla skládá ze vstupní pásky, čtecí hlavy, stavového řízení a nepovinně i pomocné paměti. Podle mechanismu a funkce čtecí hlavy, typu pomocné paměti a určenosti přechodů mezi stavy automatu, lze rozlišit mezi základními typy automatů, potažmo i třídami jazyků, které je možné těmito stroji efektivně popsat. Primární úloha automatu spočívá v určení příslušnosti vstupního řetězce do jazyka, definovaného právě tímto automatem. V případě, že řetězec, zapsaný na vstupní pásce, patří do jazyka, řekneme, že automat tento řetězec *přijme*, v opačném případě řetězec *zamítne*.

Následující text bude hovořit o některých základních typech takových automatů a odpovídajících tříd jazyků, na jejichž teorii budou navazovat následující kapitoly.

2.6.1 Konečný automat

Konečný automat je nejjednodušší z nástrojů popisovaných v této kapitole. Od toho se ale také odvíjí jeho vyjadřovací schopnost. Třída jazyků, definovatelná konečným automatem, se nazývá třídou *regulárních jazyků*. Jinými slovy, třída jazyků zpracovatelná konečnými automaty je ekvivalentní třídě regulárních jazyků.

Definice 2.6.1.1 (Konečný automat (KA))
Konečný automat je pětice $M = (Q, \Sigma, \delta, s, F)$, kde:

- Q je konečná množina stavů
- Σ je abeceda
- δ je funkcí přechodů tvaru $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to 2^Q$
- $s \in Q$ je startující stav
- $F \subseteq Q$ je konečná množina koncových stavů

Definice 2.6.1.2 (Deterministický konečný automat (DKA))
Nechť $M = (Q, \Sigma, \delta, s, F)$ je konečný automat a δ jeho přechodová funkce. Platí-li navíc $|\delta(q, a)| \leq 1$ pro všechna $q \in Q$ a $a \in \Sigma$, pak tento konečný automat nazveme *deterministickým konečným automatem.*

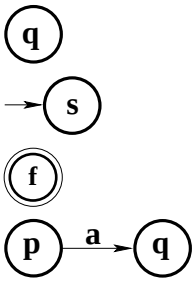
Grafická reprezentace KA

stav $q \in Q$:

startující stav $s \in Q$:

koncový stav $f \in F$:

hrana $q \in \delta(p, a)$



Tabulková reprezentace konečného automatu

- sloupec: prvky z $\Sigma \cup \{\varepsilon\}$
- řádek: stavy z Q
- první řádek: startující stav
- podtržené: koncové stavy

Definice 2.6.1.3 (Konfigurace konečného automatu)
Nechť $M = (Q, \Sigma, \delta, s, F)$ je konečný automat. *Konfigurací C konečného automatu M* nazveme uspořádanou dvojici $C = (q, \omega), kde (q, \omega) \in Q \times \Sigma^*$, a kde q označuje aktuální stav a w je dosud nezpracovaná část vstupního řetězce.

Definice 2.6.1.4 (Přechod konečného automatu)
Nechť $C = (q, a\omega)$ a $C' = (q', \omega)$ jsou dvě konfigurace konečného automatu M, přičemž $q, q' \in Q$, $a \in \Sigma \cup \{\varepsilon\}$[9] a $\omega \in \Sigma^*$. Dále, nechť $r : q' \in \delta(q, a)$. Pak M může provést přechod z konfigurace C do konfigurace C' s využitím hrany r, zapsáno: $C \vdash C'[r]$, nebo zjednodušeně $C \vdash C'$[10].

Definice 2.6.1.5 (Sekvence přechodů konečného automatu)
Nechť $M = (Q, \Sigma, \delta, s, F)$ je konečný automat. Dále:

- Nechť C je konfigurace M. Pak řekneme, že M provede nula výpočetních kroků (přechodů) z konfigurace C do konfigurace C a zapisujeme $C \vdash^0 C[\varepsilon]$, nebo zjednodušeně $C \vdash^0 C$.
- Nechť $C_0, C_1, \ldots, C_k$ jsou konfigurace M, přičemž platí, že $C_{i-1} \vdash C_i[r_i]$ pro všechna $i = 1, 2, \ldots, k$. Pak řekneme, že M provede k přechodů z konfigurace C_0 do konfigurace C_k a zapisujeme $C_0 \vdash^k C_k[r_1 r_2 \ldots r_n]$, nebo zjednodušeně $C_0 \vdash^k C_k$.

[9] Pokud $a = \varepsilon$, automat ze vstupu nepřečte žádný znak.

[10] Striktně matematicky: $\underset{M}{\vdash} \subseteq (Q \times \Sigma^*) \times (Q \times \Sigma^*)$, kde: $(q, a\omega) \underset{M}{\vdash} (q', \omega) \overset{\text{def}}{\iff} q' \in \delta(q, a)$.

- Jestliže $C_0 \vdash^k C_k[\pi]$ pro libovolné $k \geq 1$, $\pi = r_1 r_2 \dots r_n$, pak řekneme, že M netriviálně přejde z konfigurace C_0 do konfigurace C_k a zapisujeme $C_0 \vdash^+ C_k[\pi]$, nebo zjednodušeně $C_0 \vdash^+ C_k$.
- Jestliže $C_0 \vdash^k C_k[\pi]$ pro libovolné $k \geq 0$, $\pi = r_1 r_2 \dots r_n$, pak řekneme, že M přejde z konfigurace C_0 do konfigurace C_k a zapisujeme $C_0 \vdash^* C_k[\pi]$, nebo zjednodušeně $C_0 \vdash^* C_k$.

Definice 2.6.1.6 (Jazyk přijímaný konečným automatem)
Nechť $M = (Q, \Sigma, \delta, s, F)$ je konečný automat. Jazyk přijímaný automatem M, $L(M)$, je definován: $L(M) = \{w | w \in \Sigma^*, (s, w) \vdash^* (f, \varepsilon), f \in F\}$.

2.6.2 Zásobníkový automat

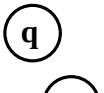
Zásobníkový automat je silnější alternativou konečného automatu. Zde se navíc využívá paměť ve formě zásobníku. Přístupy k zásobníkům se ale mohou lišit podle toho, zda má automat možnost nahlížet pouze na vrchol zásobníku, či může číst i položky pod jeho vrcholem. Zásobníkové automaty jsou schopné rozhodovat o příslušnosti vstupního řetězce do bezkontextových jazyků. Třída jazyků, zpracovatelná zásobníkovými automaty je ekvivalentní třídě bezkontextových jazyků.

Definice 2.6.2.1 (Zásobníkový automat (ZA))
Zásobníkový automat je sedmice $PDA = (Q, \Sigma, \Gamma, \delta, s, Z_0, F)$, kde:

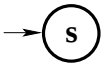
- Q je konečná množina stavů
- Σ je konečná množina všech vstupních symbolů
- Γ je konečná množina všech zásobníkových symbolů
- δ je zobrazení $\delta : Q \times (\Gamma \cup \{\varepsilon\}) \times (\Sigma \cup \{\varepsilon\}) \to 2^{Q \times \Gamma^*}$ popisující funkci přechodů
- $s \in Q$ je počáteční stav řídící jednotky
- $Z_0 \in \Gamma$ je startovací symbol vložený na počátku na zásobník
- $F \subseteq Q$ je množina koncových stavů

Grafická reprezentace ZA

stav $q \in Q$:

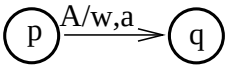


startující stav $s \in Q$:



koncový stav $f \in F$:

hrana $(q, w) \in \delta(p, A, a)$



Definice 2.6.2.2 (Konfigurace zásobníkového automatu)
Nechť $PDA = (Q, \Sigma, \Gamma, \delta, s, Z_0, F)$ je zásobníkovým automatem. Pak *konfigurací C zásobníkového automatu PDA* nazveme trojici $(q, \alpha, w) \in Q \times \Gamma^* \times \Sigma^*$.

Definice 2.6.2.3 (Přechod zásobníkového automatu)
Nechť $PDA = (Q, \Sigma, \Gamma, \delta, s, Z_0, F)$ je zásobníkovým automatem a nechť $C = (q, z, a\omega)$ a $C' = (q', z', \omega)$ jsou jeho dvě konfigurace, kde:

- $q, q' \in Q; z, z' \in \Gamma^*$,
- $\omega \in \Sigma^*; a \in \Sigma$.

Dále, nechť $(q', z') \in \delta(q, z, a)$. Pak můžeme provést přechod z konfigurace C do konfigurace C' s přečtením symbolu a, zapsáno $C \vdash C'$.

Definice 2.6.2.4 (Sekvence přechodů zásobníkového automatu)
Sekvence přechodů zásobníkového automatu je analogická k definici sekvence přechodů konečného automatu.

Definice 2.6.2.5 (Jazyky přijímané zásobníkovými automaty)
Nechť $PDA = (Q, \Sigma, \Gamma, \delta, s, Z_0, F)$ je zásobníkový automat:

- Jazyk přijímaný ZA PDA *přechodem do koncového stavu*, $L(PDA)_f$, je definován:
  $L(PDA)_f = \{w |\ w \in \Sigma^*,\ (s, Z_0, w) \vdash^* (f, z, \varepsilon),\ z \in \Gamma^*, f \in F\}$
- Jazyk přijímaný ZA PDA *vyprázdněním zásobníku*, $L(PDA)_\varepsilon$, je definován:
  $L(PDA)_\varepsilon = \{w |\ w \in \Sigma^*,\ (s, Z_0, w) \vdash^* (f, z, \varepsilon),\ z = \varepsilon,\ f \in Q\}$
- Jazyk přijímaný ZA PDA *přechodem do koncového stavu a vyprázdněním zásobníku*, $L(PDA)_{f\varepsilon}$, je definován:
  $L(PDA)_{f\varepsilon} = \{w |\ w \in \Sigma^*,\ (s, Z_0, w) \vdash^* (f, z, \varepsilon),\ z = \varepsilon, f \in F\}$

Definice 2.6.2.6 (Deterministický zásobníkový automat (DZA))
Nechť $PDA = (Q, \Sigma, \Gamma, \delta, s, Z_0, F)$ je zásobníkový automat. Dále nechť $\forall q \in Q$; $\forall a \in \Sigma$; $\forall z \in \Gamma^*$ platí

- $|\delta(q, z, a)| \leq 1$ a současně $|\delta(q, z, \varepsilon)| = 0$, nebo
- $|\delta(q, z, a)| = 0$ a současně $|\delta(q, z, \varepsilon)| \leq 1$.

Pak řekneme, že PDA je *deterministický zásobníkový automat*.

Tvrzení 2.6.2.7
Třída formálních jazyků zpracovatelná nedeterministickými zásobníkovými automaty je vlastní nadtřídou třídy formálních jazyků popsatelných deterministickými zásobníkovými automaty.

2.6.3 Turingův stroj

Turingův stroj je teoretický model počítače popsaný matematikem Alanem Turingem. Skládá se z procesorové jednotky, tvořené konečným automatem, programu ve tvaru pravidel přechodové funkce a potenciálně nekonečné pásky pro zápis mezivýsledků. Využívá se pro modelování algoritmů v teorii vyčíslitelnosti. Jeden ze způsobu vyjádření Church-Turingovy teze říká, že ke každému algoritmu existuje ekvivalentní Turingův stroj. Citace z [1]. Množina jazyků popsatelných nějakým Turingovým strojem je ekvivalentní třídě rekurzivně vyčíslitelných jazyků.

Definice 2.6.3.1 (Turingův stroj (TS))
Turingův stroj je šestice $TS = (Q, \Sigma, \Gamma, \delta, s, f)$, kde:

- Q je konečná množina stavů,
- Σ je konečná množina všech vstupních symbolů přičemž $\Delta \notin \Sigma$,
- Γ je konečná množina všech páskových symbolů, kde $\Sigma \subset \Gamma$, $\Delta \in \Gamma$,
- δ je zobrazení $\delta : \{Q \backslash f\} \times \Gamma \to Q \times (\Gamma \cup \{L, R\}$ popisující funkci přechodů, kde $L, R \notin \Gamma$,
- $s \in Q$ je počáteční stav řídící jednotky a
- $f \in Q$ je koncový stav řídící jednotky.

Definice 2.6.3.2 (Konfigurace Turingova stroje)
Nechť $TS = (Q, \Sigma, \Gamma, \delta, s, f)$ je Turingův stroj a nechť symbol Δ značí tzv. *blang* (prázdný symbol), který označuje doposud nepoužitá místa pásky(může být na pásku ale zapsán i později). Pak definujeme:

- *konfiguraci pásky* jako dvojici $C_p = (\gamma\Delta^\omega, n)$,kde $\gamma\Delta^\omega$ je nekonečný řetězec reprezentující obsah pásky, přičemž $\gamma \in \Gamma^*$, a $n \in \mathbb{N}$ označuje pozici hlavy nad touto páskou,
- *konfiguraci stroje*, která je dána konfigurací pásky a stavem stroje. Konfigurace stroje je tedy trojice $C_s = (q, \gamma\Delta^\omega, n)$, kde $q \in Q$ reprezentuje řídící stav stroje a zbývající část odpovídá konfiguraci pásky z předchozího bodu definice.

Definice 2.6.3.3 (Přechod Turingova stroje)
Nechť $TS = (Q, \Sigma, \Gamma, \delta, s, f)$ je Turingův stroj. Pro libovolný řetězec $\gamma \in \Gamma^\omega$ a číslo $n \in \mathbb{N}$ označme pomocí γ_n n-tý symbol řetězce a nechť $s_b^n(\gamma)$ označuje řetězec, který vznikne záměnou symbolu γ_n za b v řetězci γ, kde $b \in \Gamma$. Dále nechť $\forall q, q' \in Q, \forall\gamma \in \Gamma^\omega, \forall b \in \Gamma,\ \forall n \in \mathbb{N}$. Pak definujeme přechod Turingova stroje následovně:

- $(q, \gamma, n) \vdash (q', \gamma, n+1)$ pro $\delta(q, \gamma_n) = (q', R)$,
- $(q, \gamma, n) \vdash (q', \gamma, n-1)$ pro $\delta(q, \gamma_n) = (q', L)$ a $n > 0$,
- $(q, \gamma, n) \vdash (q', s_b^n(\gamma), n)$ pro $\delta(q, \gamma_n) = (q', b)$.

Výpočet Turingova stroje

Nechť $TS = (Q, \Sigma, \Gamma, \delta, s, f)$ je Turingův stroj a nechť $K_0, K_1, K_2, \ldots$ je posloupnost konfigurací, přičemž $K_i \vdash K_{i+1}$ pro všechna $i \geq 0$ taková, že K_{i+1} je v dané posloupnosti, která je:

- *nekonečná*, a nebo
- *konečná* s koncovou konfigurací (q, γ, n), přičemž rozlišujeme typy zastavení:
  a) *normální* přechodem do koncového stavu, tj. $q = f$,
  b) *abnormální*, kdy $q \neq f$ a
     i. $|\delta(q, \gamma_n)| = 0$, nebo
     ii. $n = 0$ a zároveň $\delta(q, \gamma_n) = (q', L)$ pro nějaké $q' \in Q$.

Kapitola 3

Systémy založené na spolupráci automatů

V následujícím textu se budeme zabývat spolupráci již zavedených automatů. Jedná se o zcela novou oblast výzkumu, která si bere za cíl využít více automatů, k popisu formálního jazyka, na místo jednoho složitého stroje „hladového“ po zdrojích.

Obecný systém založený na spolupráci automatů, se skládá z konečného počtu automatů, přičemž mezi sebou mohou spolupracovat opět v sekvenčním či paralelním módu. Jazyky přijímané dílčími automaty a způsob přijímání vstupních řetězců pak přímo definují jazyk, který systém popisuje.

Úvodem kapitoly budou letmo zmíněny sekvenční automatové systémy. Ve zbytku textu se zaměříme spíše na paralelní a semiparalelní automatové systémy, které jsou jak z teoretického, tak i praktického hlediska pro nás mnohem zajímavější.

3.1 Sekvenční automatové systémy

Sekvenční automatové systémy jsou postavené proti teorii CD gramatických systémů. Systém opět tvoří n-tice komponent, které jsou zde reprezentované automaty, jenž mohou pracovat nad společnou pamětí (zásobníkem, páskou, apod.). Během jednoho výpočetního kroku provede přechod pouze jedna komponenta. Stejně jako u CD gramatických systémů je přechod mezi automaty realizován např. neexistenci pravidla pro následující přechod, předem definovaný počet přechodů nad automaty apod.

Definice 3.1.0.4 (n-Sekvenční automatový systém (n-SAS))
Nechť $M_1, \ldots, M_n \in \mathcal{M}$, kde $\mathcal{M}$ je množina všech konečných automatů, zásobníkových automatů, nebo Turingových strojů, $I = \{1, \ldots, n\}$ pro nějaké $n \geq 1$ a $\forall i \in I$, Ψ_i je přechodová funkce automatu (ekvivalentního typu s δ_i z definice odpovídajícího typu automatu) prováděná na pozadí. Pak definujeme n-sekvenční automatový systém jako:

$$\vartheta = ((M_1, \Psi_1), \ldots, (M_n, \Psi_n)).$$

Definice 3.1.0.5 (Konfigurace n-SAS)
Nechť $\vartheta = ((M_1, \Psi_1), \ldots, (M_n, \Psi_n))$ je n-SAS nad množinou konečných automatů, zásobníkových automatů, nebo Turingových strojů. Pak definujeme konfiguraci n-sekvenčního

automatového systému jako n-tici:

$$\chi = (c_1, \ldots, c_n, \omega)_{|l}, \text{ kde}$$

- c_i, $\forall i \in \{1, \ldots, n\}$ pro nějaké $n \geq 1$, značí konfiguraci dílčí komponenty (automatu) až na vstupní řetězec,
- ω značí dosud nepřečtenou část vstupního řetězce,
- $l \in \{(i, j)|j \in (\mathbb{N}_0 \cup \{*\})$ určuje počet kroků provedených nad aktivní komponentou systému, přičemž $*$ zastupuje význam slova jakýkoliv a $i \in \{1, \ldots, n\}$ pro nějaké $n \geq 1$, určuje aktivní komponentu systému$\}$.

Neformálně řečeno, konfiguraci n-SAS určují vnitřní konfigurace dílčích komponent spolu s identifikací počítající komponenty a čítačem provedených kroků od její poslední aktivace. U sekvence přechodů, v případech kdy bude stav l zřejmý z kontextu, se bude přípona $|l$ často vynechávat.

Definice 3.1.0.6 (Přechod n-SAS)
Nechť $\vartheta = ((M_1, \Psi_1), \ldots, (M_n, \Psi_n))$ je n-SAS nad množinou konečných automatů, zásobníkových automatů, nebo Turingových strojů. Dále nechť $\chi = (c_1, \ldots, c_n, \omega)_{|l}$ a $\chi' = (c'_1, \ldots, c'_n, \omega')_{|l'}$ jsou dvě konfigurace sekvenčního automatového systému, $I \in \{1, \ldots, n\}$ pro nějaké $n \geq 1$ a nechť $(c_i, \omega) \vdash_{M_i} (c'_i, \omega')[r_i]$ alespoň pro jedno $i \in I$, kde r_i označuje použité pravidlo přechodové funkce δ_i komponenty M_i pro přechod mezi stavy. Pak můžeme pomocí komponenty M_i provést přechod n-SAS, zapsáno $\chi \vdash_{M_i} \chi'$, přičemž pro $j \neq i$, $c'_j = c_j$. Navíc pro $s \in \mathbb{N}$:

- pokud $r_i \notin \Psi$ a $l = (i, s)$, pak $l' = (i, s + 1)$,
- pokud $r_i \in \Psi$ a $l = (i, s)$, pak $l' = l$,
- pokud $r_i \notin \Psi$ a $l = (i', s)$, kde $i' \neq i$, pak $l' = (i, 1)$,
- pokud $l = (i', *)$, pak $l' = (i, *)$.

V n-SAS tedy existují čtyři typy přechodů. První z nich definuje přechod pro případ, kdy ho provádí již aktivovaná komponenta systému. V případě, že použitá přechodová hrana (pravidlo pro přechod) komponenty není zahrnuta v odpovídající funkci Ψ_i, provede komponenta přechod spolu s inkrementací čítače. Pokud použitou hranu ale funkce zahrnuje, provede se přechod komponenty, aniž by byl čítač jakkoliv modifikován. Třetí z případů vypovídá o přechodu, kdy výpočetní krok provádí komponenta odlišná od té, co byla aktivní v minulém výpočetním kroku. Tehdy se čítač nuluje a identifikátor aktivní komponenty ukazuje na nově aktivovanou komponentu. V některých situacích nás bude zajímat pouze aktivní komponenta a čítač kroků nebude nijak důležitý. V tomto případě se použije poslední z definovaných typů přechodů.

Definice 3.1.0.7 (Sekvence přechodů n-SAS)
Sekvence přechodů n-SAS je analogická s definicí sekvence přechodů zásobníkového, resp. konečného automatu.

Definice 3.1.0.8 (Výpočet n-SAS)
Nechť $\vartheta = ((M_1, \Psi_1), \ldots, (M_n, \Psi_n))$ je n-SAS. Pak $\forall i \in \{1, \ldots, n\}$ a pro nějaké $n \geq 1$ definujeme sekvenci přechodů:

- **s absencí hrany**, zapsáno $\vdash^{t}_{M_i}$, jako $\chi_{1|l} \vdash^{t}_{M_i} \chi_{2|l}$ iff $\chi_{1|l} \vdash^{*}_{M_i} \chi_{2|l}$, přičemž neexistuje žádné $\chi_{3|l}$ takové, že $\chi_{2|l} \vdash_{M_i} \chi_{3|l}$, kde $l = (i, *)$, $\forall j \in \{1, 2, 3\}$, $\chi_j = (c_{j_1}, \ldots, c_{j_n}, \omega_j)$ a $\chi_{j|l}$ je konfigurace n-SAS,

- **k-krokové**, zapsáno $\vdash^{=k}_{M_i}$, jako $\chi_{|(i,0)} \vdash^{=k}_{M_i} \chi'_{|(i,k)}$ iff existují $\omega_1, \ldots, \omega_{m+1}$ a $\chi_1, \ldots, \chi_{m+1}$ takové, že $\omega = \omega_1$, $\omega' = \omega_{m+1}$, $\chi = \chi_1$, $\chi' = \chi_{m+1}$ a $\forall j \in \{1, \ldots, m\}$, $\chi_{j|(i,j)} \vdash_{M_i} \chi_{j+1|(i,j')}$, přičemž $j' \geq j$, $m \geq k$, kde $\forall j' \in \{1, \ldots, m+1\}$, $\chi_{j'} = (c_{j'_1}, \ldots, c_{j'_n}, \omega_{j'})$ a $\chi'_{j'|l}$ pro $l \in \{(i, 0), \ldots, (i, k)\}$ je konfigurace n-SAS,

- **s maximálně k kroky**, zapsáno $\vdash^{\leq k}_{M_i}$, jako $\chi_{|l} \vdash^{\leq k}_{M_i} \chi'_{|l'}$ iff $\chi_{|(i,0)} \vdash^{=k'}_{M_i} \chi'_{|(i,k')}$ pro nějaké $k' \leq k$, přičemž $\chi_{|l}$, $\chi'_{|l'}$ jsou konfigurace n-SAS,

- **s minimálně k kroky**, zapsáno $\vdash^{\geq k}_{M_i}$, jako $\chi_{|l} \vdash^{\geq k}_{M_i} \chi'_{|l'}$ iff $\chi_{|(i,0)} \vdash^{=k'}_{M_i} \chi'_{|(i,k')}$ pro nějaké $k' \geq k$, přičemž $\chi_{|l}$, $\chi'_{|l'}$ jsou konfigurace n-SAS.

Každý n-SAS je určen jeho výpočetním módem. V předchozí definici jsou popsány čtyři typy možností. První z nich definuje sekvenci přechodů s absenci hrany. Je to typ výpočtu, kde jednotlivé kroky provádí komponenta, dokud má k dispozici přechody, které umožňují další výpočetní krok. Pokud hrana pro následující přechod komponenty neexistuje, je aktivována další komponenta systému, která ve výpočtu pokračuje. Druhý případ využívá přímo čítače systému. Udává, že komponenta provede přesně k kroků přes hranu, jež není zahrnuta v odpovídající funkci Ψ_i, než se provede přechod. Poslední dva případy vypovídají o výpočetním módu, který pouze shora, resp. zdola omezuje počet výpočetních kroků mimo kroků s absencí hrany v odpovídající funkci Ψ_i.

Definice 3.1.0.9 (Jazyky přijímané n-SAS)
Nechť $\vartheta = ((M_1, \Psi_1), \ldots, (M_n, \Psi_n))$ je n-sekvenční automatový systém nad množinou konečních automatů, zásobníkových automatů, nebo Turingových strojů. Dále nechť $f \in D$, kde $D = \{*, t\} \cup \{=k, \leq k, \geq k | k \in \mathbb{N}^+\}$. Pak definujeme:

- *jazyk přijímaný s-tou komponentou n-SAS* jako

$$L(\vartheta)^f_s = \{\omega \mid (\chi, \omega) \vdash^{f}_{M_{i_1}} (\chi_1, \omega_1) \vdash^{f}_{M_{i_2}} \ldots \vdash^{f}_{M_{i_m}} (\chi_m, \omega_m),\ m \geq 1, 1 \leq i_j \leq n,\ 1 \leq j \leq m,\ \omega, \omega_1, \ldots, \omega_m \in \Sigma^*,\ \omega_m \in \{\varepsilon\} \text{ a } M_s \text{ řetězec přijal}\},$$

- *jazyk přijímaný celým n-SAS* jako

$$L(\vartheta)^f_{\cap} = \{\omega \mid (\chi, \omega) \vdash^{f}_{M_{i_1}} (\chi_1, \omega_1) \vdash^{f}_{M_{i_2}} \ldots \vdash^{f}_{M_{i_m}} (\chi_m, \omega_m),\ m \geq 1, 1 \leq i_j \leq n,\ 1 \leq j \leq m,\ \omega, \omega_1, \ldots, \omega_m \in \Sigma^*,\ \omega_m \in \{\varepsilon\} \text{ a } \forall i \in \{1, \ldots, n\},\ M_i \text{ řetězec přijal}\},$$

přičemž $(\chi, \omega) = ((c_{1_0}, \ldots, c_{n_0}), \omega)$ označuje počáteční a (χ_m, ω_m) koncovou konfiguraci n-SAS ϑ.

Příklad 3.1.0.10
Popis následujícího n-SAS:

- $\vartheta_{=}((M_1, \Psi_1), (M_2, \Psi_2))$,
- $M_1, M_2 \in \mathcal{M}$, kde $\mathcal{M}$ je množina zásobníkových automatů,
- $M_1 = (\{q, q'\}, \Sigma, \{a, b, A, C, S, *, \#\}, \delta_1, q, S, \{q'\})$,
- $M_2 = (\{r, r'\}, \Sigma, \{c, A, B, C, S, *\}, \delta_2, r, S, \{r'\})$,
- $\forall q_1 \in \{q, q'\}, \forall \gamma_1 \in \{a, b, A, C, S, *, \#\}, \forall a \in \Sigma,\ \Psi_1(q_1, \gamma_1, a) =< undefined >$,
- $\forall q_2 \in \{r, r'\}, \forall \gamma_2 \in \{c, A, B, C, S, *\}, \forall a \in \Sigma,\ \Psi_2(q_2, \gamma_2, a) =< undefined >$,
- $\Sigma = \{a, b, c\}$,
- Hrany:

δ_1:

$(q, C*) \in \delta_1(q, S, \varepsilon)$
$(q, aA) \in \delta_1(q, C, \varepsilon)$
$(q, aAb) \in \delta_1(q, A, \varepsilon)$
$(q, \varepsilon) \in \delta_1(q, A, \varepsilon)$
$(q, \varepsilon) \in \delta_1(q, a, a)$
$(q, \varepsilon) \in \delta_1(q, b, b)$
$(q', *) \in \delta_1(q, *, \varepsilon)$
$(q', *) \in \delta_1(q, *, \varepsilon)$

δ_2:

$(r, B*) \in \delta_2(r, S, \varepsilon)$
$(r, A) \in \delta_2(r, b, \varepsilon)$
$(r, Bc) \in \delta_2(r, A, \varepsilon)$
$(r, C) \in \delta_2(r, A, b)$
$(r, C) \in \delta_2(r, c, \varepsilon)$
$(r, \varepsilon) \in \delta_2(r, C, \varepsilon)$
$(r, \varepsilon) \in \delta_2(r, c, c)$
$(r', \#) \in \delta_2(r, *, c)$
$(r', \#) \in \delta_2(r, \#, \varepsilon)$.

Jazyky, generované n-SAS ϑ_1 pak jsou například:

- $L_{\cap}^{=2}(\vartheta) = L_{\cap}^{\leq 2}(\vartheta) = \{a^n b^n c^n | n \geq 1\}$
- $L_{\cap}^{\geq 2}(\vartheta) = L_{\cap}^{t}(\vartheta) = \emptyset$
- $L_{1}^{=2}(\vartheta) = L_{1}^{\leq 2}(\vartheta) = \{a^n b^n c^m | n \geq 1, 1 \leq m \leq n\}$
- $L_{1}^{\geq 2}(\vartheta) = \{a^n b^n c^m | n, m \geq 1\}$
- $L_{1}^{=2}(\vartheta) = L_{1}^{\leq 2}(\vartheta) = \{a^n b^m c^n | n \geq 1, 1 \leq m \leq n\}$
- $L_{1}^{\geq 2}(\vartheta) = \emptyset$

Nad jazykem $L_{\cap}^{=2}(\vartheta)$ můžeme provést například následující výpočet:

$$\begin{aligned}
(((q,S)(r,S)),aabbcc) &\vdash_{M_1}^{=2} (((q,C*)(r,S)),aabbcc) \vdash_{M_1}^{=2} (((q,aA*)(r,S)),aabbcc)\\
&\vdash_{M_2}^{=2} (((q,aA*)(r,B*)),aabbcc) \vdash_{M_2}^{=2} (((q,aA*)(r,A*)),aabbcc)\\
&\vdash_{M_1}^{=2} (((q,A*)(r,A*)),abbcc) \vdash_{M_1}^{=2} (((q,aAb*)(r,A*)),abbcc)\\
&\vdash_{M_2}^{=2} (((q,aAb*)(r,Bc*)),abbcc) \vdash_{M_2}^{=2} (((q,aAb*)(r,Ac*)),abbcc)\\
&\vdash_{M_1}^{=2} (((q,Ab*)(r,Ac*)),bbcc) \vdash_{M_1}^{=2} (((q,b*)(r,Ac*)),bbcc)\\
&\vdash_{M_2}^{=2} (((q,b*)(r,Cc*)),bcc) \vdash_{M_2}^{=2} (((q,b*)(r,Cc*)),bcc)\\
&\vdash_{M_1}^{=2} (((q,*)(r,Cc*)),cc) \vdash_{M_1}^{=2} (((q',*)(r,Cc*)),cc)\\
&\vdash_{M_2}^{=2} (((q',*)(r,c*)),c) \vdash_{M_2}^{=2} (((q',*)(r',*)),\varepsilon)
\end{aligned}$$

Tvrzení 3.1.0.11
Každý n-CD-gramatický systém tvořený bezkontextovými gramatikami, provádějící výpočet v módu $f \in \{*,t\} \cup \{= k, \leq k, \geq k\}$ a omezený pouze na nejlevější derivace, lze převést na ekvivalentní n-SAS, tvořený zásobníkovými automaty.

Důkaz:
Vytvořme **Algoritmus:**

- **Vstup:** n-CD-gramatický systém $\Gamma_{CD} = (N, T, S_\Gamma, P_1, \ldots, P_n)$ derivující v módu $f_\Gamma \in \{*,t\} \cup \{= k, \leq k, \geq k\}$ omezený pouze na nejlevější derivace.
- **Výstup:** n-SAS $\vartheta = ((M_1, \Psi_1), (M_2, \Psi_2), \ldots, (M_n, \Psi_n))$ složený ze zásobníkových automatů počítajících nad společným zásobníkem a přijímajících s vyprázdněním zásobníku.
- **Metoda:**
  - Pro každé P_i v Γ_{CD} vytvořme M_i následovně:
    1. $M_i = (Q_i, \Sigma, \Gamma, \delta_i, q_{i_0}, S, F_i)$
    2. $Q_i = \{q_i\}$,
    3. $\Sigma = T$, kde T je množina terminálních symbolů gramatického systému Γ_{CD},
    4. $\Gamma = N \cup T$,
    5. δ_i, Ψ_i:
       - pro všechna pravidla $A \to \alpha$ komponenty P_i gramatického systému Γ_{CD}:
         i. $(q, \alpha) \in \delta_i(a, A, \varepsilon)$
         ii. pro každé $a \in substring(\alpha)$, kde $a \in T$ $(q, \varepsilon) \in \delta_i(q, a, a)$ a současně $(q, \varepsilon) \in \Psi_i(q, a, a)$,
    6. $q_{i_0} = q_i$,
    7. $S = S_\Gamma$,
    8. $F_i = \emptyset$,
  - $f_\Gamma = f_\vartheta$.

Mějme tedy n-CD-gramatický systém $\Gamma_{CD} = (N, T, S_\Gamma, P_1, \ldots, P_n)$ derivující v módu $f_\Gamma \in \{*,t\} \cup \{= k, \leq k, \geq k\}$ omezený pouze na nejlevější derivace a k němu dle předchozího

algoritmu vytvořený n-SAS $\vartheta = ((M_1, \Psi_1), \ldots, (M_n, \Psi_n))$ nad zásobníkovými automaty počítajícími nad společným zásobníkem a přijímajícími s vyprázdněním zásobníku.

Ukážeme, že $A \Rightarrow^m \omega$ v módu f, právě když a jen když $((q_1, A), \ldots, (q_n, A), \omega) \vdash^s ((q_1, \varepsilon), \ldots, (q_n, \varepsilon), \varepsilon)$ v módu f pro nějaké $m, s \geq 1$.

Druhou část tvrzení, tj. „jen když“, dokážeme indukcí pro m. Předpokládejme tedy, že $A \Rightarrow^m \omega$:

- Je-li $m = 1$ a $\omega = a_1 \ldots a_k$ pro $k \geq 0$, $l, l' \in \{1 \ldots n\} \times \mathbb{N}$ a $l \neq l'$, pak $A \Rightarrow^m_{P_j} \omega$ a

$$\begin{aligned}((q_1, A), \ldots, (q_n, A), a_1 \ldots a_k)_{|l} &\vdash_{M_j} ((q_1, a_1 \ldots a_k), \ldots, (q_n, a_1 \ldots a_k), a_1 \ldots a_k)_{|l'} \\ &\vdash^k_{M_j} ((q_1, \varepsilon), \ldots, (q_n, \varepsilon), \varepsilon)_{|l'}\end{aligned}$$

- Je-li $m > 1$ první krok derivace musí mít tvar $A \Rightarrow_{P_j} X_1 \ldots X_k$, přičemž $X_i \Rightarrow^{m_i} x_i$ pro $m_i < m$, $1 \leq i \leq k$ a $x_1 \ldots x_k = \omega$. Pak pro $l, l' \in \{1 \ldots n\} \times \mathbb{N}$ a $l \neq l'$

$$((q_1, A), \ldots, (q_n, A), \omega)_{|l} \vdash_{M_j} ((q_1, X_1 \ldots X_k), \ldots, (q_n, X_1 \ldots X_k), \omega)_{|l'}$$

  - Je-li $X_i = x_i : x_i \in \Sigma$, pak

$$((q_1, X_i), \ldots, (q_n, X_i), x_i)_{|(j',s)} \vdash_{M_{j'}} ((q_1, \varepsilon), \ldots, (q_n, \varepsilon), \varepsilon)_{|(j',s)}$$

  - Je-li $X_i \in N$, pak indukcí pro derivační / přechodový mód f
    - $f \in \{*\}$, pak z induktivního předpokladu

$$((q_1, X_i), \ldots, (q_n, X_i), x_i)_{|l} \vdash^*_{M_j} ((q_1, \varepsilon), \ldots, (q_n, \varepsilon), \varepsilon)_{|l'}$$

    - $f \in \{t\} \cup \{\leq k, = k, \geq k\}$, z induktivního předpokladu $X_i \Rightarrow^* x_i$, pak je v uvažovaném módu možné najít v souladu s f pod-derivaci takovou, že $X_i \Rightarrow^*_{P_{j'}} X'_i \Rightarrow^* x_i$. Potom ale také pro ϑ

$$\begin{aligned}((q_1, X_i), \ldots, (q_n, X_i), x_i)_{|l} &\vdash^*_{M_{j'}} ((q_1, X'_i), \ldots, (q_n, X'_i), x_i)_{|l'} \\ &\vdash^* ((q_1, \varepsilon), \ldots, (q_n, \varepsilon), \varepsilon)_{|l''}\end{aligned}$$

Odtud je již vidět, že sekvenci levých derivací

$$\begin{aligned}A \Rightarrow_{P_j} X_1 \ldots X_k \Rightarrow^{m'_1}_{P_{j^1}} X'_1 X_2 \ldots X_k \Rightarrow^{m_1} x_1 X_2 \ldots X_k \Rightarrow^{m'_2}_{P_{j^2}} \ldots \Rightarrow^{m'_k}_{P j^k} \\ \Rightarrow^{m'_k}_{P j^k} x_1 \ldots X'_k \Rightarrow^{m'_k} x_1 \ldots x_k\end{aligned}$$

odpovídá níže uvedená posloupnost

$$\begin{aligned}&((q_1, A), \ldots, (q_n, A), x_1 \ldots x_k)_{|l} \\ &\vdash_{M_j} ((q_1, X_1 \ldots X_k), \ldots, (q_n, X_1 \ldots X_k), x_1 \ldots x_k)_{|l'} \\ &\vdash^{m'_1}_{M_{j^1}} ((q_1, X'_1 X_2 \ldots X_k), \ldots, (q_n, X'_1 X_2 \ldots X_k), x'_1 x_2 \ldots x_k)_{|l'_1} \\ &\vdash^{m_1} ((q_1, X_2 \ldots X_k), \ldots, (q_n, X_2 \ldots X_k), x_2 \ldots x_k)_{|l_1} \\ &\vdash^{m'_2}_{M_{j^2}} \ldots \vdash^{m'_k}_{M_{j^k}} ((q_1, X'_k), \ldots, (q_n, X'_k), x'_k)_{|l'_k} \\ &\vdash^{m_k} ((q_1, \varepsilon), \ldots, (q_n, \varepsilon), \varepsilon)_{|l_k}\end{aligned}$$

Zbývá dokázat, že pokud $((q_1, A), \ldots, (q_n, A), \omega) \vdash^s ((q_1, \varepsilon), \ldots, (q_n, \varepsilon), \varepsilon)$, pak $A \Rightarrow^+ \omega$. Důkaz povedeme opět indukcí pro s.

- Pro $s = 1, \omega = \varepsilon$ a $A \to \varepsilon$ je pravidlo v P_j pro nějaké $j = 1 \ldots n$ s ohledem na f
- Předpokládejme, že dokazovaná relace platí pro všechna $s' < s$. Pak první přechod ϑ musí vypadat následovně:

  $$((q_1, A), \ldots, (q_n, A), \omega) \vdash_{M_j} ((q_1, X_1 \ldots X_k), \ldots, (q_1, X_1 \ldots X_k), \omega)$$

  a $((q_1, X_i), \ldots, (q_1, X_i), x_i) \vdash^{m'_i + m_i} ((q_1, \varepsilon), \ldots, (q_n, \varepsilon), \varepsilon)$ pro $i = 1 \ldots k$, kde $\omega = x_1 \ldots x_k$, pak nutně $A \to X_1 \ldots X_k$ je pravidlem z P_j a derivace $X_i \Rightarrow^+ x_i$ vyplývá z induktivního předpokladu. Je-li $X_i \in \Sigma$, pak $X_i \Rightarrow^0 x_i$.

  Tím pádem

  $$A \Rightarrow_{P_j} X_1 \ldots X_k \Rightarrow^{m'_1}_{P_{j^1}} X'_1 X_2 \ldots X_k \Rightarrow^{m_1} x_1 X_2 \ldots X_k \Rightarrow^{m'_2}_{P_{j^2}} \ldots \Rightarrow^{m'_k}_{P_{j^k}}$$
  $$\Rightarrow^{m'_k}_{P_{j^k}} x_1 \ldots X'_k \Rightarrow^{m'_k} x_1 \ldots x_k = \omega$$

  je levou derivací ω z A.

  Položíme-li $S = A$, dostaneme $S \Rightarrow^+ \omega$, právě když

  $$((q_1, S_\vartheta), \ldots, (q_n, S_\vartheta), \omega) \vdash^+ ((q_1, \varepsilon), \ldots, (q_n, \varepsilon), \varepsilon).$$

  Tím pádem tedy $L(\Gamma_{CD}) = L_\varepsilon(\vartheta)$.

□

Klasické automaty mohou simulovat pouze levé, či pravé derivace, což způsobuje větší omezenost oproti CD-gramatickým systémům. Abychom dokázali, že sekvenční automatový systém má stejnou popisovací schopnost, museli bychom dokázat, že každý CD-gramatický systém lze převést na ekvivalentní CD-gramatický systém využívající pouze levých, resp. pravých, derivací. Problém zůstává otevřený. Pravděpodobně ale pro mód $f \in \{t\} \cup \{=k, \leq k, \geq k\}$ platí vztah CFL $\subset$ n-SASL $\subseteq$ ET0L pro $n > 1$, kde CFL je třída bezkontextových jazyků, n-SASL třída jazyků popsatelných n-SAS systémem a ET0L je třída jazyků generovatelných nějakou ET0L gramatikou.

3.1.1 Využití n-SA systému

Moderním trendem překladačů je snaha o přepínání mezi jednotlivými metodami syntaktické analýzy. Důvod proč kombinovat takové metody spočívá hlavně ve zjednodušení gramatických pravidel, ve dřívějším odhalení syntaktické chyby a v neposlední řadě i v šetření se zdroji. Kombinovaná syntaktická analýza otevírá ale dveře i do nových oblastí, jako je například automatizované odhalování plagiátů mezi prácemi studentů, separování důležitých částí kódů, atd. Chytré metody přepínání také zajišťují větší vyjadřovací schopnost přijímaných jazyků.

n-SAS lze realizovat například pomocí zásobníkových a rozšířených zásobníkových automatů, přičemž automaty mohou pracovat i nad stejným zásobníkem, čímž vlastně spolu vytváří automat s více řídícími hlavami. Takových automatů může být v n-SAS přirozeně i více. Samotné přepínaní mezi metodami se děje podle výpočetního módu systému, nad kterým se překlad spustí. Typicky se bude jednat o přepínaní s absencí hrany.

Nástroj tedy velice elegantním a nenásilným způsobem umožňuje sestavit překladač, který využívá pro každou větnou formu jiného způsobu zpracování části vstupního řetězce.

3.2 Paralelní systémy založené na automatech

Přestože pomocí n-SAS lze simulovat jakýsi paralelní výpočet, lze brát paralelní automatové systémy za logické následovatele sekvenčních automatových systémů. Odpadá zde však výpočetní mód typu $f \in \{*, t\} \cup \{= k, \leq k, \geq k\}$. Toto místo nahradí množina takzvaných aktivačních symbolů, díky nimž bude celý běh systému řízen. Zde nemá příliš velký smysl uvažovat komponenty pracující nad společnou pamětí (zásobníkem, páskou, ap.), přesto i zde není tato možnost nijak zakázána. Je ale nutné si uvědomit možné konflikty vycházející ze sdílení „prostředků".

Definice 3.2.0.1 (Paralelní automatový systém (n-PAS))
Nechť $M_1, \ldots, M_n \in \mathcal{M}$, kde $\mathcal{M}$ je množina všech zásobníkových automatů a Act je konečná množina aktivačních symbolů. Pak definujeme paralelní automatový systém (n-PAS) jako:

$$\vartheta = (M_1, \ldots, M_n, Act).$$

Definice 3.2.0.2 (Konfigurace n-paralelního systému automatů)
Nechť $\vartheta = (M_1, \ldots, M_n, Act)$ je n-PAS, kde $\forall i \in \{1, \ldots, n\}$ pro nějaké $n \geq 1$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, F_i)$ je zásobníkový automat. Pak definujeme jeho konfiguraci jako:

$$\chi_\vartheta = (c_1, \ldots, c_n, \omega)_{|\text{ď}}, \text{ kde:}$$

- $\forall i \in \{1, \ldots, n\}$, $c_i \in Q_i \times (\Gamma_i \cup Act)^*$ značí konfiguraci dílčí komponenty (automatu) až na vstupní řetězec,
- $\text{ď} \in \{1, \ldots, n\} \cup \{All\}$ značí aktivní komponentu systému,
- $\omega \in \Sigma^*$ značí dosud nepřečtenou část vstupního řetězce.

Definice 3.2.0.3 (Přechod n-paralelního systému automatů)
Nechť $\vartheta = (M_1, \ldots, M_n, Act)$ je n-PAS, kde $\forall i \in \{1, \ldots, n\}$ pro nějaké $n \geq 1$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, F_i)$ je zásobníkový automat přijímající v koncovém stavu s vyprázdněním zásobníku, a dále nechť $\chi_\vartheta = (c_1, \ldots, c_n, a\omega)_{|\text{ď}}$ a $\chi'_\vartheta = (c'_1, \ldots, c'_n, \omega)_{|\text{ď}'}$ jsou dvě konfigurace, přičemž $a \in \Sigma \cup \{\varepsilon\}$. Pak definujeme přechod n-paralelního automatového systému ϑ, zapsáno $\chi_\vartheta \vdash \chi'_\vartheta$, následovně:

- Pokud $\text{ď} \in \{1, \ldots, n\}$, pak $\forall j \in \{1, \ldots, n\}$ taková, že $j \neq \text{ď}$, $c'_j = c_j$. Dále pokud:
  a) $c_\text{ď} = (q_\text{ď}, \varepsilon)$ a $q_\text{ď} \in F_\text{ď}$, pak $\text{ď}' \in \{All\}$ a $c'_\text{ď} = (q_0, z_{\text{ď}_0})$, kde $z_{\text{ď}_0}$ je počáteční zásobníkový symbol a $q_0 \in Q_\text{ď}$ počáteční stav komponenty $M_\text{ď}$,
  b) $c_\text{ď} = (q_\text{ď}, z_\text{ď} y_\text{ď})$, kde $q_\text{ď} \in Q_\text{ď}$, $y_\text{ď} = \{Act, \Gamma_\text{ď}\}^*$ a $z_\text{ď} = Act_l$, pak $c'_\text{ď} = (q_\text{ď}, y_\text{ď})$ a $\text{ď}' = l$ pro $l \in \{1, \ldots, n\}$,
  c) jinak, za předpokladu, že platí $c_\text{ď} \vdash c'_\text{ď}$, komponenta $M_\text{ď}$ přejde z $c_\text{ď}$ do $c'_\text{ď}$, přičemž $\text{ď}' = \text{ď}$,
- Pokud $\text{ď} \in \{All\}$ a jestliže:
  a) $\forall i \in \{1, \ldots, n\}$, $c_i = (q_i, z_i y_i)$, kde $q_i \in Q_i$, $y_i = \{Act, \Gamma_i\}^*$, $z_i = \Gamma_i \cup \{\varepsilon\}$, a jestliže $c_i \vdash c'_i$, pak komponenta M_i přejde z c_i do c'_i, $\text{ď}' = \text{ď}$,
  b) pro nějaké $i \in \{1, \ldots, n\}$, $c_i = (q_i, z_i y_i)$, kde $q_i \in Q_i$, $y_i = \{Act, \Gamma_i\}^*$ a $z_i = Act_j$, pak $c'_i = (q_i, y_i)$. Navíc $\text{ď}' = j$ a $\forall k \neq i$ platí, že $c'_k = c_k$.

Konfiguraci n-PAS tedy tvoří samotné dílčí konfigurace komponent (mimo vstup), spolu s definicí jejich aktivit a společným vstupním řetězcem. V jednu chvíli jsou aktivní buď všechny, nebo pouze jeden z automatů n-PAS. Pokud výpočet provádí všechny komponenty, pak všechny dělají výpočetní krok současně, dle definice jejich přechodových pravidel a to do chvíle, kdy se na vrcholu zásobníku alespoň jednoho zásobníkového automatu objeví aktivační symbol. V tu chvíli se všechny komponenty zablokují a zůstane aktivní pouze ta, na niž aktivační symbol ukazuje. Ten se následně z vrcholu zásobníku odstraní. Aktivní komponenta pak provádí výpočet, než celý svůj podřetězec v pořádku zpracuje. Tehdy se u této komponenty nastaví opět počáteční konfigurace a znovu se aktivují všechny automaty.

Definice 3.2.0.4 (Sekvence přechodů n-paralelního systému automatů)
Nechť $\vartheta = (M_1, \ldots, M_n, Act)$ je n-PAS. Dále:

- Nechť χ je n-PAS konfigurace. Pak řekneme, že ϑ provede nula výpočetních kroků z konfigurace χ do konfigurace χ a zapisujeme $\chi \vdash^0 \chi$.
- Nechť $\chi_0, \chi_1, \ldots, \chi_k$ jsou n-PAS konfigurace, přičemž platí, že $\chi_{i-1} \vdash \chi_i$ pro všechna $i = 1, \ldots, k$. Pak řekneme, že ϑ provede k přechodů z konfigurace χ_0 do konfigurace χ_k a zapisujeme $\chi_0 \vdash^k \chi_k$.
- Jestliže $\chi_0 \vdash^k \chi_k$ pro libovolné $k \geq 1$, pak řekneme, že ϑ netriviálně přejde z konfigurace χ_0 do konfigurace χ_k a zapisujeme $\chi_0 \vdash^+ \chi_k$.
- Jestliže $\chi_0 \vdash^k \chi_k$ pro libovolné $k \geq 0$, pak řekneme, že ϑ přejde z konfigurace χ_0 do konfigurace χ_k a zapisujeme $\chi_0 \vdash^* \chi_k$.

Definice 3.2.0.5 (Dosažitelná konfigurace n-PAS)
Nechť $\vartheta = (M_1, \ldots, M_n, Act)$ je n-PAS a nechť $\chi_s = (c_1, \ldots, c_n, \omega)_{|d}$ je počáteční konfigurace a $\chi_{other} = (c'_1, \ldots, c'_n, \omega')_{|d'}$ konfigurace n-PAS. Pak o konfiguraci χ_{other} řekneme, že je *dosažitelná*, pokud $\chi_s \vdash^* \chi_{other}$ pro nějaké ω, $\omega' \in \Sigma^*$.

Definice 3.2.0.6 (Deterministicky řízený n-PAS)
Nechť $\vartheta = (M_1, \ldots, M_n, Act)$ je n-PAS. Dále nechť $\chi_s = (c_1, \ldots, c_n, \omega)_{|d}$ je konfigurace ϑ a funkce $f_{Act}(\chi)$ vrací počet všech komponent $c_i = (q_i, z_i)$ systému, kde $z_i \in \{Act(\Gamma_i \cup Act)^*\}$ v konfiguraci χ, $q_i \in Q_i$ je stav komponenty systému. Pokud platí pro všechna dosažitelná χ v ϑ, $f_{Act}(\chi) \leq 1$, pak n-paralelní automatový systém nazveme *deterministicky řízeným* n-paralelním systémem (n-DPAS).

Definice 3.2.0.7 (Jazyky přijímané n-(D)PAS)
Nechť $\vartheta = (M_1, \ldots, M_n, Act)$ je n-(D)PAS, pak definujeme:

- jazyk přijímaný první komponentou n-(D)PAS jako:

$$L_{first}(\vartheta) = \{ \; \omega | \omega \in \Sigma^*, (c_{1_0}, \ldots, c_{n_0}, \omega)_{|All} \vdash^* (c_1, \ldots, c_n, \varepsilon)_{|All}, \text{ kde } (c_{1_0}, \ldots, c_{n_0}, \omega)_{|All} \text{ je počáteční konfigurace systému, a } M_1 \text{ vstupní řetězec přijal. } \}$$

- jazyk sjednocení n-(D)PAS jako:

$$L_{\cup}(\vartheta) = \{ \omega|\omega \in \Sigma^*, (c_{1_0}, \ldots, c_{n_0}, \omega)_{|All} \vdash^* (c_1, \ldots, c_n, \varepsilon)_{|All}, \text{ kde } (c_{1_0}, \ldots, c_{n_0}, \omega)_{|All} \text{ je počáteční konfigurace systému, a alespoň pro jedno } i, M_i \text{ vstupní řetězec přijal. } \}$$

- jazyk přijímaný celým n-(D)PAS jako:

$$L_{\cap}(\vartheta) = \{ \omega|\omega \in \Sigma^*, (c_{1_0}, \ldots, c_{n_0}, \omega)_{|All} \vdash^* ((c_1, \ldots, c_n), \varepsilon)_{|All}, \text{ kde } (c_{1_0}, \ldots, c_{n_0}, \omega)_{|All} \text{ je počáteční konfigurace systému, a } \forall i, M_i \text{ vstupní řetězec přijal. } \}$$

Příklad 3.2.0.8 (2-PAS)
Mějme:

- $\vartheta = (M_1, M_2, \{\mathcal{A}ct_2\})$,
- $M_1 = (\{q_0, q_1, q_2\}, \Sigma, \{\#, S\}, q_0, S, \delta_1, \{q_2\})$
- $M_2 = (\{q_0, q_1, q_2, q_3\}, \Sigma, \{\#, S\}, q_0, S, \delta_2, \{q_3\})$
- $\Sigma = \{a, b, c\}$
- Hrany:

δ_1:
$(q_1, aSbc\mathcal{A}ct_2\#) \in \delta_1(q_0, S, \varepsilon)$
$(q_1, \varepsilon) \in \delta_1(q_1, a, a)$
$(q_1, aSb) \in \delta_1(q_1, S, \varepsilon)$
$(q_1, \varepsilon) \in \delta_1(q_1, S, \varepsilon)$
$(q_1, \varepsilon) \in \delta_1(q_1, b, b)$
$(q_1, \varepsilon) \in \delta_1(q_1, b, b)$
$(q_2, \varepsilon) \in \delta_1(q_1, \#, \varepsilon)$

δ_2:
$(q_0, aS) \in \delta_2(q_0, S, \varepsilon)$
$(q_0, \varepsilon) \in \delta_2(q_0, a, a)$
$(q_1, bS\#) \in \delta_2(q_0, S, \varepsilon)$
$(q_1, \varepsilon) \in \delta_2(q_1, b, b)$
$(q_1, Sc) \in \delta_2(q_1, S, b)$
$(q_2, \varepsilon) \in \delta_2(q_1, S, c)$
$(q_2, \varepsilon) \in \delta_2(q_2, c, c)$
$(q_2, \varepsilon) \in \delta_2(q_2, \#, \varepsilon)$
$(q_3, \varepsilon) \in \delta_2(q_2, S, \varepsilon)$

ϑ přijímá jazyky:

- $L_{first} = L_{\cap} = L_{\cup} = \{a^i b^i c^i | i \in \mathbb{Z}^+\}$

Výpočet pro řetězec „aabbcc“ by byl následující:

$$\begin{aligned}
((q_0, S), (q_0, S), aabbcc)_{|all} &\vdash ((q_1, aSbc\mathcal{A}ct_2\#), (q_0, aS), aabbcc)_{|all} \vdash \\
((q_1, Sbc\mathcal{A}ct_2\#), (q_0, S), abbcc)_{|all} &\vdash ((q_1, aSbbc\mathcal{A}ct_2\#), (q_0, aS), abbcc)_{|all} \vdash \\
((q_1, Sbbc\mathcal{A}ct_2\#), (q_0, S), bbcc)_{|all} &\vdash ((q_1, bbc\mathcal{A}ct_2\#), (q_1, bS\#), bbcc)_{|all} \vdash \\
((q_1, bc\mathcal{A}ct_2\#), (q_1, S\#), bcc)_{|all} &\vdash ((q_1, c\mathcal{A}ct_2\#), (q_1, Sc\#), cc)_{|all} \vdash \\
((q_1, \mathcal{A}ct_2\#), (q_2, c\#), c)_{|All} &\vdash ((q_1, \#), (q_2, c\#), c)_{|All} \vdash \\
((q_1, \#), (q_2, \#), \varepsilon)_{|2} &\vdash ((q_1, \#), (q_2, \varepsilon), \varepsilon)_{|2} \vdash \\
((q_1, \#), (q_2, S), \varepsilon)_{|All} &\vdash ((q_2, \varepsilon), (q_3, \varepsilon), \varepsilon)_{|All}
\end{aligned}$$

Kapitola 4

Multipřijímací automatové systémy

V této kapitole se zaměříme na tzv. multipřijímací automatové systémy. Jde o semiparalelní, speciálně o paralelní, či sekvenční automatové systémy složené z konečných automatů, zásobníkových automatů, čí Turingových strojů. Ty jsou řízené pomocí množiny stavových, či pravidlových aktivátorů, které určují spouštění a pozastavování dílčích automatů automatového systému.

4.1 Multipřijímací, stavem řízený, automatový systém

Multipřijímací, stavem řízený, automatový systém je prvním z popisovaných multipřijímacích systémů. Skládá se z n automatů, které budeme opět nazývat komponentami systému automatů, a konečné množiny (de)aktivačních, stavem řízených, pravidel systému automatů. V každém kroku se ověří existence pravidla pro změnu aktivit automatů. Pokud takové pravidlo v systému existuje, pak je aplikováno společně s výpočetním krokem systému. V opačném případě zůstává režim výpočtu systému automatů totožný s předchozím režimem. Tedy, dílčí automaty, které jsou aktivní, zůstávají aktivní i po aplikaci následného výpočetního kroku systému. To stejné platí i pro neaktivní komponenty. Následující definice se budou týkat pouze multipřijímacích automatových systémů složených výhradně ze zásobníkových automatů. Definice systému pro konečné automaty, Turingovy stroje, či jiné typy automatů by byla analogická.

Definice 4.1.0.9 (n-Přijímající, stavem řízený, AS (n-MAS))
Nechť $I = I(n)$ pro nějaké $n \geq 1$. Dále nechť $\forall i \in I$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, F_i)$ je zásobníkovým automatem. Pak definujeme n-přijímající, stavem řízený, automatový systém (n-MAS), jako:

$$\vartheta = (M_1, \ldots, M_n, \Psi, S), \text{ kde:}$$

- Ψ je konečná množina přepínacích pravidel tvaru $(q_1, q_2, \ldots, q_n) \rightarrow (d_1, d_2, \ldots, d_n)$, kde $\forall i \in I$:
  - $q_i \in Q_i$
  - $d_i \in \{e, d\}$, přičemž
    - e značí aktivní (enable) komponentu automatového systému

$*$ d značí neaktivní (disable) komponentu automatového systému

- S je n-tice $(\not{e}_1^0, \ldots, \not{e}_n^0)$ a značí počáteční aktivitu komponent n-MAS.

Definice 4.1.0.10 (Multikonfigurace n-MAS)
Nechť $\vartheta = (M_1, \ldots, M_n, \Psi, S)$, kde $\forall i \in I$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, F_i)$ pro $I = I(n)$ a nějaké $n \geq 1$, je n-MAS. Pak definujeme multikonfiguraci jako n-tici

$$\chi = (x_1^{\not{e}_1}, \ldots, x_n^{\not{e}_n}), \text{ kde } \forall i \in I\text{:}$$

- $x_i = (q_i, z_i, \omega_i) \in Q_i \times \Gamma_i^* \times \Sigma^*$,
- $\not{e}_i \in \{d, e\}$, přičemž horní index d_i, resp. e_i označuje konfiguraci neaktivní, resp. aktivní komponentu M_i z n-MAS,
- $\omega_i \in \Sigma^*$ značí dosud nezpracovanou část vstupního řetězce.

Definice 4.1.0.11 (Přechod v n-MAS)
Nechť $I = I(n)$ pro nějaké $n \geq 1$ a nechť $\vartheta = (M_1, \ldots, M_n, \Psi, S)$ je n-MAS, $\forall i \in I$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, F_i)$. Dále:

- Mějme dvě n-MAS multikonfigurace
  - $\chi = ((q_1, z_1, a_1\omega_1)^{\not{e}_1}, (q_2, z_2, a_2\omega_2)^{\not{e}_2}, \ldots, (q_n, z_n, a_n\omega_n)^{\not{e}_n})$,
  - $\chi' = ((q_1', z_1', \omega_1')^{\not{e}_1'}, (q_2', z_2', \omega_2')^{\not{e}_2'}, \ \ldots, (q_n', z_n', \omega_n')^{\not{e}_n'})$,

  přičemž $\forall i \in I$:
  - $q_i, q_i' \in Q_i$; $z_i, z_i' \in \Gamma_i^*$; $\not{e}_i, \not{e}_i' \in \{e, d\}$
  - $\omega_i, \omega_i' \in \Sigma^*$, $a_i \in \Sigma \cup \{\varepsilon\}$
  - $(g_i', z_i') \in \delta_i(q_i, z_i, a_i)$ $\forall i$, pro která $\not{e}_i = e$.

Pak můžeme provést přechod z multikonfigurace χ do χ', zapsáno $\chi \vdash \chi'$, přičemž platí:

- $\forall j \in I$, pro která platí $\not{e}_j = d$, $q_j' = q_j$ a $\omega_j' = a_j\omega_j$,
- $\forall j \in I$, pro která platí $\not{e}_j = e$, $q_j' \in Q_j$ a $\omega_j' = \omega_j$.
- Pokud $(q_1', \ldots, q_n') \to (e_1, \ldots, e_n) \in \Psi$, kde $e_k \in \{e, d\}$ pro všechna $k \in I$, pak $\not{e}_k' = e_k$,
- Pokud $\forall (e_1, \ldots, e_n) \in \overbrace{\{e, d\} \times \ldots \times \{e, d\}}^{n\times} : (q_1', \ldots, q_n') \to (e_1, \ldots, e_n) \notin \Psi$, pak pro všechna $k \in I : \not{e}_k' = \not{e}_k$.

Definice 4.1.0.12 (Sekvence přechodů v n-MAS)
Nechť $\vartheta = (M_1, \ldots, M_n, \Psi, S)$ je n-MAS. Dále:

- Nechť χ je n-MAS multikonfigurace. Pak řekneme, že ϑ provede nula výpočetních kroků z multikonfigurace χ do multikonfigurace χ a zapisujeme $\chi \vdash^0 \chi$.
- Nechť $\chi_0, \chi_1, \ldots, \chi_k$ jsou n-MAS multikonfigurace, přičemž platí, že $\chi_{i-1} \vdash \chi_i$ pro všechna $i \in I(k)$ pro nějaké $k \geq 1$. Pak řekneme, že ϑ provede k přechodů z multikonfigurace χ_0 do multikonfigurace χ_k a zapisujeme $\chi_0 \vdash^k \chi_k$.

- Jestliže $\chi_0 \vdash^k \chi_k$ pro libovolné $k \geq 1$, pak řekneme, že ϑ netriviálně přejde z multikonfigurace χ_0 do multikonfigurace χ_k a zapisujeme $\chi_0 \vdash^+ \chi_k$.
- Jestliže $\chi_0 \vdash^k \chi_k$ pro libovolné $k \geq 0$, pak řekneme, že ϑ přejde z multikonfigurace χ_0 do multikonfigurace χ_k a zapisujeme $\chi_0 \vdash^* \chi_k$.

Definice 4.1.0.13 (n-Přijímající deterministicky stavem řízený AS (n-DMAS))
Nechť $I = I(n)$ pro nějaké $n \geq 1$ a nechť $\vartheta = (M_1, \ldots, M_n, \Psi, S)$ je n-MAS. Dále $\forall i \in I$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, F_i)$ a navíc $\forall (q_1, \ldots, q_n) \in \overbrace{Q_1 \times \ldots \times Q_n}^{n\times}$ platí:

$$\begin{aligned}&(((q_1, \ldots, q_n) \to (d_1, \ldots, d_n)) \in \Psi\\ &\wedge((q_1, \ldots, q_n) \to (d_1', \ldots, d_n')) \in \Psi)\\ &\Rightarrow \forall i \in I,\ d_i = d_i'.\end{aligned}$$

Pak řekneme, že ϑ je deterministicky stavem řízený automatový systém.

Definice 4.1.0.14 (n-MAS s plně definovaným řízením)
Nechť $I = I(n)$ pro nějaké $n \geq 1$ a nechť $\vartheta = (M_1, \ldots, M_n, \Psi, S)$ je n-MAS, kde $\forall i \in I$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, F_i)$ je zásobníkový automat. n-MAS ϑ nazveme n-MAS s plně definovaným řízením, pokud $\forall (q_1, \ldots, q_n) \in Q_1 \times \ldots \times Q_n$, $(q_1, \ldots, q_n) \to (d_1, \ldots, d_n) \in \Psi$ pro $d_i \in \{d, e\}$.

Definice 4.1.0.15 (n-MAS multi-jazyky)
Nechť $I = I(n)$ pro nějaké $n \geq 1$ a nechť $\vartheta = (M_1, \ldots, M_n, \Psi, S)$ je n-MAS, kde $\forall i \in I$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, F_i)$ je zásobníkový automat přijímající v koncovém stavu[1]. Dále nechť

- $\chi_0 = ((q_1, z_1, \omega_1)^{d_1}, \ldots, (q_n, z_n, \omega_n)^{d_n})$ je jeho počáteční a
- $\chi_f = ((q_1', z_1', \varepsilon)^{d_1'}, \ldots, (q_n', z_n', \varepsilon)^{d_n'})$ jeho koncová

n-MAS multikonfigurace, kde $\forall i \in I$:

- $q_i, q_i' \in Q_i$, $z_i, z_i' \in \Gamma^*$,
- $d_i, d_1' \in \{d, e\}$,
- $\omega, \omega_i \in \Sigma^*$.

Pak definujeme n-MAS *multi-jazyk*:

- první komponenty:

$$n\text{-}L_{first}(\vartheta) = \{(\omega_1, \ldots, \omega_n) | \chi_0 \vdash^* \chi_f; q_1 \in F_1\}$$

- sjednocení:

$$n\text{-}L_{\cup}(\vartheta) = \{(\omega_1, \ldots, \omega_n) | \chi_0 \vdash^* \chi_f; q_j' \in F_j \text{ alespoň pro jedno } j \in I\}$$

[1] Pro zásobníkové automaty přijímající s vyprázdněním zásobníku, nebo s v koncovém stavu s vyprázdněním zásobníku je definice analogická.

- průniku:

$$n\text{-}L_{\cap}(\vartheta) = \{(\omega_1, \ldots, \omega_n) | \chi_0 \vdash^* \chi_f; q'_j \in F_j \text{ pro všechna } j \in I\}$$

- konkatenace:

$$L_{concat}(\vartheta) = \{\omega_1\omega_2 \ldots \omega_n | \chi_0 \vdash^* \chi_f; q'_j \in F_j \text{ pro všechna } j \in I\}$$

Je zřejmé, že n-multipřijímací automatové systémy budou přijímat n-multi-jazyky, které jsou definovány dle režimu přijetí. Snadno lze přijímat multipřijímacím automatovým systémem i jednoduché jazyky, jež budou tvořeny různými operacemi nad přijímanými „n-řetězci“.

Nejdříve budeme uvažovat speciální případ n-MAS multijazyku, kdy všechny vstupní řetězce n-tice jsou v počáteční konfiguraci shodné. Výslednému jazyku pak budeme říkat jazyk omezeného n-MAS systému.

Definice 4.1.0.16 (Jazyk omezeného n-MAS systému)
Nechť $I = I(n)$ pro nějaké $n \geq 1$ a nechť $\vartheta = (M_1, \ldots, M_n, \Psi, S)$ je n-MAS, kde $\forall i \in I$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, F_i)$ je zásobníkový automat přijímající v koncovém stavu. Dále nechť

- $\chi_0 = ((q_1, z_1, \omega)^{d_1}, \ldots, (q_n, z_n, \omega)^{d_n})$ je jeho počáteční a
- $\chi_f = ((q'_1, z'_1, \varepsilon)^{d'_1}, \ldots, (q'_n, z'_n, \varepsilon)^{d'_n})$ jeho koncová

n-MAS multikonfigurace, kde $\forall i \in I$:

- $q_i, q'_i \in Q_i,\ z_i, z'_i \in \Gamma^*$,
- $d_i, d'_1 \in \{d, e\}$,
- $\omega \in \Sigma^*$.

Pak definujeme *jazyk omezeného n-MAS systému*:

- první komponenty:

$$L_{first^1}(\vartheta) = \{\omega | \chi_0 \vdash^* \chi_f; q_1 \in F_1\}$$

- sjednocení:

$$L_{\cup^1}(\vartheta) = \{\omega | \chi_0 \vdash^* \chi_f; q'_j \in F_j \text{ alespoň pro jedno } j \in I\}$$

- průniku:

$$L_{\cap^1}(\vartheta) = \{\omega | \chi_0 \vdash^* \chi_f; q'_j \in F_j \text{ pro všechna } j \in I\}$$

Obecně lze definovat jazyky pomocí operací nad tzv. multiřetězci, které zde bude zastoupeny n-ticí řetězců.

Definice 4.1.0.17 (n-multiřetezec)
Nechť Σ je abeceda a nechť $\omega_1, \ldots \omega_n \in \Sigma^*$ pro nějaké $n > 1$, pak *n-multiřetězcem* nazveme n-tici $(\omega_1, \ldots, \omega_n)$.

Definice 4.1.0.18 (Operace n-konkatenace, n-sjednocení, n-první nad n-multiřetězci)
Nechť $I = I(n)$ pro nějaké $n \geq 1$ a nechť $\vartheta = (M_1, \ldots, M_n, \Psi, S)$ je n-MAS, kde $\forall i \in I$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i,\ s_i, z_{i,0}, F_i)$ je zásobníkový automat. Dále nechť $\omega_1, \ldots, \omega_n \in \Sigma$. Pak můžeme definovat operace:

- *n-konkatenaci multiřetězce* jako $msconcat((\omega_1, \ldots, \omega_n)) = \omega_1 \ldots \omega_n$,
- *n-sjednocení multiřetězce* jako $msunion((\omega_1, \ldots, \omega_n)) = \{\omega_1, \ldots, \omega_n\}$,
- *n-první* nad multiřetěcem jako $msfirst((\omega_1, \ldots, \omega_n)) = \omega_1$.

Definice 4.1.0.19 (Operace n-konkatenace, n-sjednocení, n-první nad n-jazyky)
Nechť $I = I(n)$ pro nějaké $n \geq 1$ a nechť $\vartheta = (M_1, \ldots, M_n, \Psi, S)$ je n-MAS, kde $\forall i \in I$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i,\ s_i, z_{i,0}, F_i)$ je zásobníkový automat. Dále nechť $\omega_1, \ldots, \omega_n \in \Sigma$ a $(\omega_1, \ldots, \omega_n) \in n\text{-}L_x(\vartheta)$ pro nějaké $x \in \{first, \cap, \cup\}$. Pak můžeme nad n-multijazyky definovat jazyky:

- *n-konkatenace* jako $msconcat(n\text{-}L_x(\vartheta))$=$\{\omega_1 \ldots \omega_n |\ (\omega_1, \ldots, \omega_n) \in n\text{-}L_x(\vartheta)\}$,
- *n-sjednocení* jako $msunion(n\text{-}L_x(\vartheta)) = \{\omega_1, \ldots, \omega_n |\ (\omega_1, \ldots, \omega_n) \in n\text{-}L_x(\vartheta)\}$,
- *n-první* nad n-multijazykem jako $msfirst(n\text{-}L_x(\vartheta)) = \{\omega_1 |\ (\omega_1, \ldots, \omega_n) \in n\text{-}L_x(\vartheta)\}$.

Příklad 4.1.0.20 (2-MAS)
Mějme:

- $\vartheta = (M_1, M_2, \Psi, (e, e))$,
- $M_1 = (\{q_0, q_1, q_2\}, \Sigma, \{\#, *, B\}, q_0, \#, \delta_1, \{q_2\})$
- $M_2 = (\{q_0, q_1, q_2\}, \Sigma, \{\#, *, B\}, q_0, \#, \delta_2, \{q_2\})$
- $\Sigma = \{a, b, c\}$
- Hrany:

δ_1:

$(q_0, *) \in \delta_1(q_0, \#, a)$
$(q_0, *B) \in \delta_1(q_0, *, a)$
$(q_0, BB) \in \delta_1(q_0, B, a)$
$(q_1, \varepsilon) \in \delta_1(q_0, B, b)$
$(q_1, \varepsilon) \in \delta_1(q_1, B, b)$
$(q_1, \varepsilon) \in \delta_1(q_1, *, b)$
$(q_1, \varepsilon) \in \delta_1(q_0, *, b)$
$(q_2, \varepsilon) \in \delta_1(q_1, \varepsilon, c)$
$(q_2, \varepsilon) \in \delta_1(q_2, \varepsilon, c)$

δ_2:

$(q_0\#) \in \delta_2(q_0, \#, a)$
$(q_0, *) \in \delta_2(q_0, \#, b)$
$(q_0, *B) \in \delta_2(q_0, *, b)$
$(q_0, BB) \in \delta_2(q_0, B, b)$
$(q_1, \varepsilon) \in \delta_2(q_0, B, c)$
$(q_1, \varepsilon) \in \delta_2(q_1, B, c)$
$(q_1, \varepsilon) \in \delta_2(q_1, *, c)$
$(q_2, \varepsilon) \in \delta_2(q_0, *, c)$

- $\Psi = \{(r, r') \to (d, d) | (r, r') \notin \{(q_0, q_0), (q_1, q_0), (q_1, q_1), (q_2, q_1), (q_2, q_2)\}\}$

ϑ pak pro $\omega_1 = \omega_2 \in \Sigma^*$ přijímá jazyky:

- $L_{first^1} = \{a^i b^i c^j | i \in \mathbb{N}^+, j \in \mathbb{N}\}$

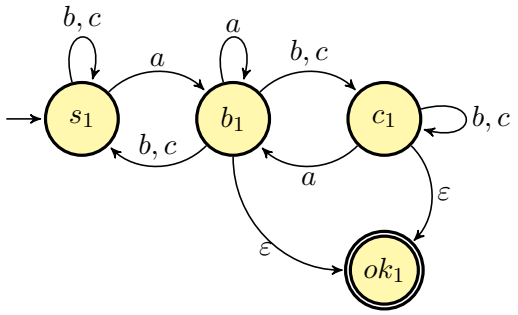
- $L_{\cap^1} = \{a^i b^i c^i | i \in \mathbb{N}^+\}$
- $L_{\cup^1} = \{a^i b^i c^j | a^j b^i c^i : i \in \mathbb{N}^+, j \in \mathbb{N}\}$

Předchozí příklad naznačuje, že multipřijímací automatové systémy složené ze zásobníkových automatů, jsou schopné elegantním způsobem zpracovávat kontextové jazyky. Přesto, že se mi doposud nepodařilo dokázat, že takovéto systémy dosahují síly Turingova stroje, jsem o tomto tvrzení přesvědčen.

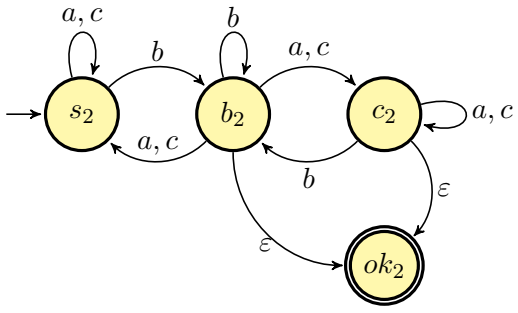
Velmi zajímavé je ale pozorování chování multipřijímacího systému konečných automatů. Podívejme se na následující příklad.

Příklad 4.1.0.21 (3-MAS)
Mějme 3-multipřijímací automatový systém konečných automatů:

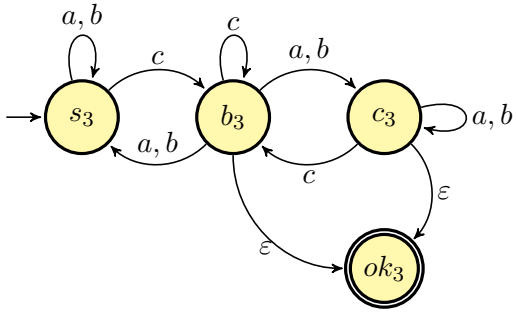
- $\vartheta = (M_1, M_2, M_3, \Psi, (e, e, e))$,
- $M_1 = (Q_1, \{a, b, c\}, \delta_1, s_1, \{ok_1\})$
- $M_2 = (Q_2, \{a, b, c\}, \delta_2, s_2, \{ok_2\})$
- $M_3 = (Q_3, \{a, b, c\}, \delta_2, s_3, \{ok_3\})$
- hrany:
  - M_1 :



  - M_2 :



  - M_3 :



- $P = \{(s_1, s_2, s_3) \to (e, e, e), (b_1, s_2, s_3) \to (d, e, e), (s_1, b_2, s_3) \to (e, d, e), (s_1, s_2, b_3) \to (e, e, d), (b_1, b_2, s_3) \to (d, d, e), (b_1, s_2, b_3) \to (d, e, d), (s_1, b_2, b_3) \to (e, d, d), (b_1, b_2, b_3) \to (e, e, e), (ok_1, c_2, c_3) \to (d, e, e), (ok_1, ok_2, c_3) \to (d, d, e), (c_1, ok_2, c_3) \to (e, d, e), (c_1, ok_2, ok_3) \to (e, d, d), (ok_1, c_2, ok_3) \to (d, e, d)\}$
- $\Psi = P \cup \{(q_1, q_2, q_3) \to (d, d, d) | q_1 \in Q_1, q_2 \in Q_2, q_3 \in Q_3$, kde $(q_1, q_2, q_3) \to (a_1, a_2, a_3) \notin P; \forall (a_1, a_2, a_3) \in \{e, d\} \times \{e, d\} \times \{e, d\}\}$

Je snadné ukázat, že např. unárně definovaný jazyk průniku je pak definován jako

$$L_{\cap^1}(\vartheta) = \{\omega : |\omega|_a = |\omega|_b = |\omega|_c, \}$$

nebo obecně

$$L_{\cap}(\vartheta) = \{(\omega_1, \ldots, \omega_n) : |\omega_1|_a = |\omega_2|_b = |\omega_3|_c, \}$$

Pomocí n-MAS nad konečnými automaty jsme schopni rozhodovat o příslušnosti řetězců jazyků, určených vzájemnou korespondencí symbolů abecedy. Některé Dickovy jazyky však dělají n-MAS konečných automatů problémy. Je nutné vzít na vědomí, že konečné automaty nemají k dispozici žádnou paměť. Nebudou tedy v uvažovaném systému schopny rozhodovat například o jazyku daným předpisem $L = \{ww^R | \ w \in \Sigma^*\}$, což je typický bezkontextový jazyk. Třída bezkontextových jazyků a jazyků daných n-MAS systémem konečných automatů pro nějaké $n > 1$ jsou tedy nesrovnatelné.

4.2 Multipřijímací automatový přechodem řízený systém

Multipřijímací, přechodem řízený, automatový systém je logickou alternativou k popsanému multipřijímacímu, stavem řízenému, automatovému systému. Opět se skládá z n automatů (komponent) a konečné množiny (de)aktivačních, tentokrát přechodem řízených, pravidel systému automatů. V každém kroku se ověřuje existence pravidla pro změnu aktivit automatů. Pokud takové pravidlo v systému existuje, pak je stejně, jako u předchozího systému, aplikováno společně s výpočetním krokem systému. I zde se budou definice týkat výhradně multipřijímacích automatových systémů složených ze zásobníkových automatů, s vědomím, že definice systému pro konečné automaty, nebo Turingovy stroje, popřípadě využívající kombinaci různých strojů (automatů), by byla analogická.

Definice 4.2.0.22 (n-Přijímající přechodem řízený AS (n-MAT))
Nechť $I = I(n)$ pro nějaké $n \geq 1$. Dále nechť $\forall i \in I$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, F_i)$ je zásobníkovým automatem. Pak definujeme n-přijímací, přechodem řízený automatový systém jako $(n + 2)$-tici:

$$\vartheta = (M_1, \ldots, M_n, \Psi, S), \text{ kde } \forall i \in I\text{:}$$

- $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, F_i)$ je zásobníkový automat,
- Ψ je konečná množina přepínacích pravidel tvaru $(r_1, \ldots, r_n) \to (\text{ď}_1, \ldots, \text{ď}_n)$, kde $\forall j \in I$:
  - $r_j \in \{(q_j, q'_j, a, \gamma_j, \omega_j) \mid ((q'_j, \omega_j) \in \delta_j(q_j, \gamma_j, a)) : q_j, q'_j \in Q_j, \ \omega_j \in \Gamma_j^*, \ \gamma_j \in \Gamma_j \cup \{\varepsilon\}, \ a \in \Sigma \cup \{\varepsilon\}\} \cup \{l\}$,
  - $\text{ď}_j \in \{e, d\}$, přičemž

  - $*$ e značí aktivní komponentu automatového systému
  - $*$ d značí neaktivní (blokovanou) komponentu automatového systému
- S je n-tice $(\mathscr{d}_1^0, \mathscr{d}_2^0, \ldots, \mathscr{d}_n^0)$ a značí počáteční aktivitu komponent n-MAT.

Definice 4.2.0.23 (Multikonfigurace n-MAT)
Nechť $\vartheta = (M_1, \ldots, M_n, \Psi, S)$, kde $\forall i \in I$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, F_i)$ pro $I = I(n)$ a nějaké $n \geq 1$, je n-MAT. Pak definujeme multikonfiguraci jako n-tici

$$\chi = (x_1^{\mathscr{d}_1}, \ldots, x_n^{\mathscr{d}_n}) \text{ , kde } \forall i \in I\text{:}$$

- $x_i = (q_i, z_i, \omega_i) \in Q_i \times \Gamma_i^* \times \Sigma^*$,
- $\mathscr{d}_i \in \{d, e\}$, přičemž horní index d_i, resp. e_i označuje konfiguraci neaktivní, resp. aktivní komponentu M_i z n-MAT,
- $\omega_i \in \Sigma^*$ značí dosud nezpracovanou část vstupního řetězce.

Definice 4.2.0.24 (Přechod v n-MAT)
Nechť $I = I(n)$ pro nějaké $n \geq 1$ a nechť $\vartheta = (M_1, \ldots, M_n, \Psi, S)$ je n-MAT, $\forall i \in I$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, F_i)$. Dále:

- Mějme dvě n-MAT multikonfigurace
  - $\chi = ((q_1, z_1, a_1\omega_1)^{\mathscr{d}_1}, \ldots, (q_n, z_n, a_n\omega_n)^{\mathscr{d}_n})$,
  - $\chi' = ((q_1', z_1', \omega_1')^{\mathscr{d}_1'}, \ldots, (q_n', z_n', \omega_n')^{\mathscr{d}_n'})$,

  přičemž $\forall i \in I$:
  - $q_i, q_i' \in Q_i$; $z_i, z_i' \in \Gamma_i^*$; $\mathscr{d}_i, \mathscr{d}_i' \in \{e, d\}$
  - $\omega_i, \omega_i' \in \Sigma^*$, $a_i \in \Sigma \cup \{\varepsilon\}$
  - $(g_i', z_i') \in \delta_i(q_i, z_i, a_i)$ $\forall i$, pro která $\mathscr{d}_i = e$.

Pak můžeme provést přechod z multikonfigurace χ do χ', zapsáno $\chi \vdash \chi'$, přičemž platí:

- $\forall j \in I$, pro která platí, že $\mathscr{d}_j = d$, $q_j' = q_j$ a $\omega_j' = a_j\omega_j$,
- $\forall j \in I$, pro která platí, že $\mathscr{d}_j = e$, $q_j' \in Q_j$ a $\omega_j' = \omega_j$.
- Pokud $\forall j \in I : r_j \in \{(q_j, q_j', a_j, z_j, z_j')|\ (g_j', z_j') \in \delta_i(q_j, z_j, a_j)\} \cup \{l\} \wedge ((r_j \in \{l\}) \Leftrightarrow (\mathscr{d}_j = d))$ platí, že $(r_1', \ldots, r_n') \to (\mathscr{e}_1, \ldots, \mathscr{e}_n) \in \Psi$, kde $\mathscr{e}_k \in \{e, d\}$ pro všechna $k \in I$, pak $\mathscr{d}_k' = \mathscr{e}_k$,
- Pokud $\forall j \in I$, $r_j \in \{(q_j, q_j', a_j, z_j, z_j')|\ (g_j', z_j') \in \delta_i(q_j, z_j, a_j)\} \cup \{l\} \wedge ((r_j \in \{l\}) \Leftrightarrow (\mathscr{d}_j = d))$ platí, že $\forall(\mathscr{e}_1, \ldots, \mathscr{e}_n) \in \overbrace{\{e, d\} \times \ldots \times \{e, d\}}^{n\times} : (r_1', \ldots, r_n') \to (\mathscr{e}_1, \ldots, \mathscr{e}_n) \notin \Psi$, kde $\mathscr{e}_k \in \{e, d\}$ pro všechna $k \in I$, pak $\mathscr{d}_k' = \mathscr{d}_k$.

Definice 4.2.0.25 (Sekvence přechodů v n-MAT)
Nechť $\vartheta = (M_1, \ldots, M_n, \Psi, S)$ je n-MAT. Dále:

- Nechť χ je n-MAS multikonfigurace. Pak řekneme, že ϑ provede nula výpočetních kroků z multikonfigurace χ do multikonfigurace χ a zapisujeme $\chi \vdash^0 \chi$.

- Nechť $\chi_0, \chi_1, \ldots, \chi_k$ jsou n-MAT multikonfigurace, přičemž platí, že $\chi_{i-1} \vdash \chi_i$ pro všechna $i \in I(k)$ pro nějaké $k \geq 1$. Pak řekneme, že ϑ provede k přechodů z multikonfigurace χ_0 do multikonfigurace χ_k a zapisujeme $\chi_0 \vdash^k \chi_k$.
- Jestliže $\chi_0 \vdash^k \chi_k$ pro libovolné $k \geq 1$, pak řekneme, že ϑ netriviálně přejde z multikonfigurace χ_0 do multikonfigurace χ_k a zapisujeme $\chi_0 \vdash^+ \chi_k$.
- Jestliže $\chi_0 \vdash^k \chi_k$ pro libovolné $k \geq 0$, pak řekneme, že ϑ přejde z multikonfigurace χ_0 do multikonfigurace χ_k a zapisujeme $\chi_0 \vdash^* \chi_k$.

Definice 4.2.0.26 (n-přijímající deterministicky přechodem řízený AS (n-DMAT))
Nechť $I = I(n)$ pro nějaké $n \geq 1$ a nechť $\vartheta = (M_1, \ldots, M_n, \Psi, S)$ je n-MAS. Dále $\forall(r_1, \ldots, r_n)$, kde $\forall j \in I : r_j \in \{(q_j, q'_j, a, \gamma_j, \omega_j) |\ ((q'_j, \omega_j) \in \delta_j(q_j, \gamma_j, a_j)),\ q_j, q'_j \in Q_j,\ \omega_j \in \Gamma_j^*,\ \gamma_j \in \Gamma_j \cup \{\varepsilon\},\ a_j \in \Sigma \cup \{\varepsilon\}\} \cup \{l\}$, platí:

$$\begin{gathered}((r_1, \ldots, r_n) \to (\mathcal{d}_1 \ldots, \mathcal{d}_n)) \in \Psi \\ \wedge((r_1, \ldots, r_n) \to (\mathcal{d}'_1, \ldots, \mathcal{d}'_n)) \in \Psi \\ \Rightarrow \mathcal{d}_i = \mathcal{d}'_i;\ \forall i \in I.\end{gathered}$$

Pak řekneme, že ϑ je deterministicky přechodem řízený automatový systém.

Definice 4.2.0.27 (n-MAT s plně definovaným řízením)
Nechť $I = I(n)$ pro nějaké $n \geq 1$ a nechť $\vartheta = (M_1, \ldots, M_n, \Psi, S)$ je n-MAS, kde $\forall i \in I$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i,\ s_i, z_{i,0}, F_i)$ je zásobníkový automat. n-MAT ϑ nazveme n-MAT s plně definovaným řízením, pokud $\forall(r_1, \ldots, r_n) \in R_1 \times \ldots \times R_n$, kde $\forall i \in I$, $R_i = \{(q_i, q'_i, a, \gamma_i, \omega_i) |\ q_i, q'_i \in Q_i,\ a \in \Sigma \cup \{\varepsilon\},\ \gamma_i \in \Gamma_i \cup \{\varepsilon\},\ \omega_i \in \Sigma^*\}$, $(r_1, \ldots, r_n) \to (\mathcal{d}_1, \ldots, \mathcal{d}_n) \in \Psi$ pro $\mathcal{d}_i \in \{d, e\}$.

Definice 4.2.0.28 (n-MAT n-multijazyky a jazyky)
Jazyky, resp. n-multijazyky jsou definované analogicky s definicí jazyků, resp. n-multijazyků nad n-MAS systémy.

Příklad 4.2.0.29 (2-MAT)
Mějme:

- $\vartheta = (M_1, M_2, \Psi, (e, e))$,
- $M_1 = (\{q_0, q_1, q_2\}, \Sigma, \{\#, *, B\}, q_0, \#, \delta_1, \{q_2\})$
- $M_2 = (\{q_0, q_1, q_2\}, \Sigma, \{\#, *, B\}, q_0, \#, \delta_2, \{q_2\})$
- $\Sigma = \{a, b, c\}$
- Hrany:

δ_1**:**

$r_1 = (q_0, q_0, a, \#, *)$	pro $(q_0, *) \in \delta_1(q_0, \#, a)$
$r_2 = (q_0, q_0, a, *, *B)$	pro $(q_0, *B) \in \delta_1(q_0, *, a)$
$r_3 = (q_0, q_0, a, B, BB)$	pro $(q_0, BB) \in \delta_1(q_0, B, a)$
$r_4 = (q_0, q_1, b, B, \varepsilon)$	pro $(q_1, \varepsilon) \in \delta_1(q_0, B, b)$
$r_5 = (q_1, q_1, b, B, \varepsilon)$	pro $(q_1, \varepsilon) \in \delta_1(q_1, B, b)$

$r_6 = (q_1, q_1, b, *, \varepsilon)$ pro $(q_1, \varepsilon) \in \delta_1(q_1, *, b)$
$r_7 = (q_0, q_1, b, *, \varepsilon)$ pro $(q_1, \varepsilon) \in \delta_1(q_0, *, b)$
$r_8 = (q_1, q_2, c, \varepsilon, \varepsilon)$ pro $(q_2, \varepsilon) \in \delta_1(q_1, \varepsilon, c)$
$r_9 = (q_2, q_2, c, \varepsilon, \varepsilon)$ pro $(q_2, \varepsilon) \in \delta_1(q_2, \varepsilon, c)$

δ_2:

$r_1 = (q_0, q_0, a, \#, \#)$ pro $(q_0\#) \in \delta_2(q_0, \#, a)$
$r_2 = (q_0, q_0, b, \#, *)$ pro $(q_0, *) \in \delta_2(q_0, \#, b)$
$r_3 = (q_0, q_0, b, *, *B)$ pro $(q_0, *B) \in \delta_2(q_0, *, b)$
$r_4 = (q_0, q_0, b, B, BB)$ pro $(q_0, BB) \in \delta_2(q_0, B, b)$
$r_5 = (q_0, q_1, c, B, \varepsilon)$ pro $(q_1, \varepsilon) \in \delta_2(q_0, B, c)$
$r_6 = (q_0, q_0, c, B, \varepsilon)$ pro $(q_1, \varepsilon) \in \delta_2(q_1, B, c)$
$r_7 = (q_0, q_0, c, *, \varepsilon)$ pro $(q_1, \varepsilon) \in \delta_2(q_1, *, c)$
$r_8 = (q_0, q_0, c, *, \varepsilon)$ pro $(q_2, \varepsilon) \in \delta_2(q_0, *, c)$

- $P = (\{r_1, r_2, r_3, r_4, r_5, r_6, r_7\} \times \{r_1, r_2, r_3, r_4\}) \cup (\{r_5, r_6, r_7, r_8, r_9\} \times \{r_5, r_6, r_7\}) \cup (\{r_8, r_9\} \times \{r_8\})$
- $\Psi = \{(r, r') \to (d, d) | (r, r') \notin P\}$

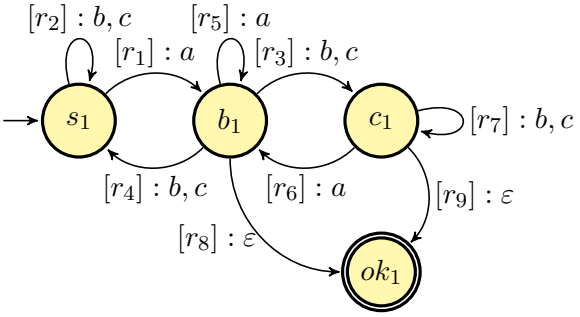
ϑ pak pro $\omega_1 = \omega_2 \in \Sigma^*$ přijímá jazyky:

- $L_{first^1} = \{a^i b^i c^j | i \in \mathbb{N}^+, j \in \mathbb{N}\}$
- $L_{\cap^1} = \{a^i b^i c^i | i \in \mathbb{N}^+\}$
- $L_{\cup^1} = \{a^i b^i c^j | a^j b^i c^i : i \in \mathbb{N}^+, j \in \mathbb{N}\}$

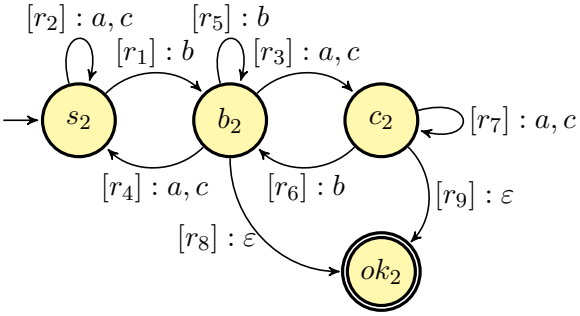
Z předchozího příkladu (tj. 4.2.0.29 a z příkladu 4.1.0.20 lze snadno vypozorovat, že se řízení pouze přesunulo ze stavů na přechodové hrany, které do stavů vstupují, nebo z něj vystupují. To bylo zcela postačující k tomu, aby jazyky, přijímané automatovým systémem z příkladu 4.1.0.20, byly ekvivalentní jazykům odpovídajících módů předešlého příkladu. Lépe lze vidět závislost mezi multipřijímajícími automatovými systémy v následujícím příkladu. Pro přehlednost zde budou některé přechodové hrany, lišící se pouze čteným vstupním symbolem, označeny za shodnou.

Příklad 4.2.0.30 (3-MAT)
Mějme 3-multipřijímací automatový systém konečných automatů:

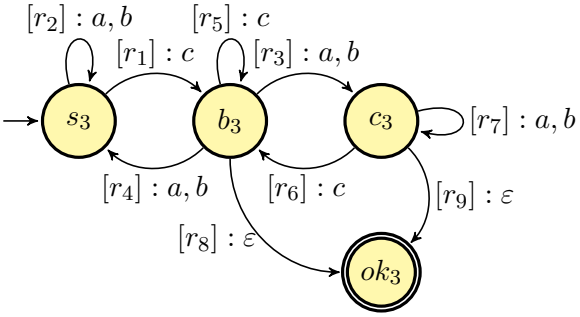
- $\vartheta = (M_1, M_2, M_3\Psi, (e, e, e))$,
- $M_1 = (Q_1, \{a, b, c\}, \delta_1, s_1, \{ok_1\})$
- $M_2 = (Q_2, \{a, b, c\}, \delta_2, s_2, \{ok_2\})$
- $M_3 = (Q_3, \{a, b, c\}, \delta_2, s_3, \{ok_3\})$
- hrany:



– M_1 :



– M_2 :



– M_3 :

- Ψ :
  - $P = \{(p_1, p_2, p_3) \to (d_1, d_2, d_3) | \forall i \in \{1, 2, 3\} : p_i \in \{r_1, r_2, r_4, r_5\} \wedge \exists j \in \{1, 2, 3\} : p_j \in \{r_2, r_4\} \wedge \forall k = 1, 2, 3 : p_k \in \{r_1, r_5\} \Leftrightarrow d_k = d\} \cup \{(p_1, p_2, p_3) \to (e, e, e) | p_1, p_2, p_3 \in \{r_1, l\}\} \cup \{(r_5, r_5, r_5) \to (e, e, e)\}$,
  - $P_d = \{(p_1, p_2, p_3) \to (d_1, d_2, d_3) | \forall i \in \{1, 2, 3\} : p_i \in \{r_1, r_2, r_5, l\} \wedge \exists j \in \{1, 2, 3\} : p_j \in \{r_2\} \wedge \forall k \in \{1, 2, 3\} : p_k \in \{r_1, l\} \Leftrightarrow d_k = d\}$
  - $P_{sp} = \{(p_1, p_2, p_3) \to (d_1, d_2, d_3) | \forall i \in \{1, 2, 3\} : p_i \in \{r_5, r_3, r_7, r_6, r_9, r_8, l\} \wedge \exists j \in \{1, 2, 3\} : p_j \in \{r_3, r_7\} \wedge \forall k \in \{1, 2, 3\} : p_k \in \{r_5, r_6, r_8, r_9, l\} \Leftrightarrow d_k = d\}$
  - $\Psi = P \cup P_d \cup P_{sp}$,

Opět je snadné ukázat, že např. unárně definovaný jazyk průniku je pak definován jako

$$L_{\cap^1}(\vartheta) = \{\omega : |\omega|_a = |\omega|_b = |\omega|_c, \}$$

nebo obecně

$$L_{\cap}(\vartheta) = \{(\omega_1, \ldots, \omega_n) : |\omega_1|_a = |\omega_2|_b = |\omega_3|_c, \}$$

V automatech z pravidla bývá daleko více hran než stavů, tedy i množina přepínacích pravidel Ψ, oproti stavy řízeného systému, zde výrazně narůstá. Na druhou stranu lze dříve

odhalovat nepřijatelné řetězce, potažmo i dříve zastavit výpočet vedoucí k chybě. Mezi řízením podle stavů automatů a hran automatů existuje pevná souvislost. Z příkladů je na první pohled vidět, že převod ze stavy řízeného systému na systém řízený hranami lze pouze pomocí jiné množiny přepínacích pravidel, beroucí v úvahu vstupní a výstupní hrany odpovídajícího stavu. O něco hůře by se převáděl n-MAT na odpovídající n-MAS. Zde už jiná množina Ψ nemusí plně postačovat a musí se zasáhnout i do struktury dílčích automatů.

Definice 4.2.0.31 (R_{In} nad zásobníkovým automatem)
R_{In} je zobrazení definované vztahem

$$\begin{aligned} R_{In}(M,q) = \quad \{ \quad & (q,q',a,\gamma,\omega) \mid \\ & M = (Q,\ \Sigma,\ \Gamma,\ \delta,\ s_i,\ z_{i,0}, F_i),\ q,q' \in Q,\ a \in \Sigma \cup \{\varepsilon\}, \\ & \gamma \in \Gamma \cup \{\varepsilon\},\ \omega \in \Gamma^*,\ \neg(q = q' \ \wedge\ a \in \{\varepsilon\} \ \wedge\ \gamma \in \{\varepsilon\}), \\ & (q',\omega) \in \delta(q,\gamma,a) \\ \} & \end{aligned}$$

Definice 4.2.0.32 (R_{Loop} nad zásobníkovým automatem)
R_{Loop} je zobrazení definované vztahem

$$\begin{aligned} R_{Loop}(M,q) = \quad \{ \quad & (q,q,a,\gamma,\omega) \mid \\ & M = (Q,\ \Sigma,\ \Gamma,\ \delta,\ s_i, z_{i,0},\ F_i),\ q \in Q,\ a \in \{\varepsilon\}, \\ & \gamma \in \{\varepsilon\},\ \omega \in \{\varepsilon\},\ (q,\omega) \in \delta(q,\varepsilon,\varepsilon) \\ \} & \end{aligned}$$

Množina $R_{In}(M,q)$ je tedy množinou všech hran automatu M, které vstupují do stavu q, přičemž jsou z této množiny vyloučeny vlastní smyčky (tj. hrany z q do q, které nečtou žádný symbol ze vstupního řetězce a neberou ohled na obsah zásobníku). Naopak množina $R_{Loop}(M,q)$ obsahuje právě vlastní smyčku, pokud taková pro daný stav q existuje. Definice R_{In} a R_{Loop} pro konečné automaty, resp. Turingovy stroje by byla analogická.

Algoritmus 4.2.0.33
Převod n-MAS zásobníkových automatů na n-MAT zásobníkových automatů.[2]

- **Vstup:** n-MAS $\vartheta = (M_1, \ldots, M_n, \Psi, S)$ s plně definovaným řízením, kde $\forall i \in \{1, \ldots, n\}$ pro nějaké $n \geq 1$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, F_i)$ je zásobníkový automat.
- **Výstup:** n-MAT $\vartheta = (M'_1, \ldots, M'_n, \Psi', S')$ s plně definovaným řízením, kde $\forall i \in \{1, \ldots, n\}$ pro nějaké $n \geq 1$, $M'_i = (Q'_i, \Sigma', \Gamma'_i, \delta'_i, s'_i, z'_{i,0}, F'_i)$ je zásobníkový automat.
- **Metoda:** $\forall i \in \{1, \ldots, n\}$:
  - $Q'_i = Q_i$,
  - $\Sigma' = \Sigma$,
  - $\Gamma'_i = \Gamma_i$,
  - $(q_i, \omega_i) \in \delta_i(r_i, \gamma_i, a) \Leftrightarrow (q_i, \omega_i) \in \delta'_i(r_j, \gamma_i, a)$, $\forall q_i, r_i \in Q_i \cap Q'_i$, $\forall \gamma_i \in \Gamma_i \cap \Gamma'_i$, $\forall a \in \Sigma \cap \Sigma'$, $\forall \omega_i \in \Gamma_i^* \cap \Gamma_i'^* \wedge \forall q_i \in Q_i,\ (q_i, \varepsilon) \in \delta_i(q_i, \varepsilon, \varepsilon)$,

[2] Analogicky pro konečný automat, či Turingův stroj.

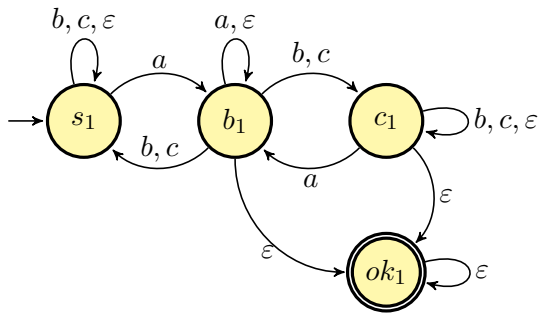
- $s'_i = s_i$,
- $z'_{i,0} = z_{i,0}$,
- $\forall q_i \in Q_i,\ q_i \in F_i \Leftrightarrow q_i \in F'_i$,
- Ψ' :
  - $P = \{(r_1, \ldots, r_n) \rightarrow (\overbrace{e, \ldots, e}^{n\times}) |\ (q_1, \ldots, q_n) \rightarrow (\acute{d}_1, \ldots, \acute{d}_n) \in \Psi,\ \forall i \in \{1, \ldots, n\},\ q_i \in Q_i,\ \acute{d}_i \in \{d, e\},\ r_i \in R_{In}(M_i, q_i) \cup R_{Loop}(M_i, q_i),\ \exists j \in \{1, \ldots, n\}, \acute{d}_i = e \ \wedge\ \acute{d}_i = d \Leftrightarrow r_i \in R_{Loop}(M_i, q_i) \cup R_{In}(M_i, q_i)\}$
  - $P' = \{(r_1, \ldots, r_n) \rightarrow (\overbrace{d, \ldots, d}^{n\times}) |\ \forall (\acute{d}_1, \ldots, \acute{d}_n) \in \overbrace{\{d, e\} \times \ldots \times \{d, e\}}^{n\times},\ (r_1, \ldots, r_n) \rightarrow (\acute{d}_1, \ldots, \acute{d}_n) \notin R_{In}\}$
  - $\Psi' = P \cup P'$
- $S' = (\overbrace{e, \ldots, e}^{n\times})$

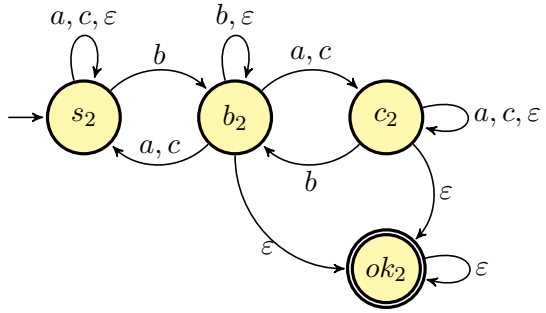
Neformálně řečeno, převod se provede tak, že se použijí automaty z n-MAS, přičemž se ke každému stavu dílčí komponenty přidá přechodová hrana, která přes daný stav cyklí bez toho, aniž by nějak brala v potaz vstupní symbol, či zásobník. K systému s rozšířenými automaty pak přidáváme aktivační pravidla tak, že pro každé takové pravidlo, tvořené n-ticí stavů v n-MAS, bude množina pravidel zahrnující všechny hrany vstupující do těchto stavů, přičemž bude zařízeno, že automaty, které by v n-MAS měly být blokovány, zde budou aktivní, ale budou cyklit přes algoritmem přidanou hranu. Výjimku vytváří pravidla, která měla blokovat všechny komponenty systému. Ty vedou na blokaci komponent všech automatů i zde. n-tice použitých hran, které už v množině aktivačních pravidel nejsou zahrnuty, apriori vedou na blokaci všech komponent také. Odsud je patrné, proč algoritmus bezpečně pracuje pouze nad plně specifikovanými n-MAS. Nevylučuji však ani lepší algoritmy, které by mohly být použity obecněji. Tuto problematiku ale ponechávám otevřenou pro další možný výzkum nad navrženými systémy.

Přes fakt, že zde uvedený algoritmus je uveden pro zásobníkové automaty, kvůli úspoře místa si ukážeme aplikaci algoritmu pouze na příkladu 4.1.0.21 reprezentujícím 3-MAS konečných automatů.

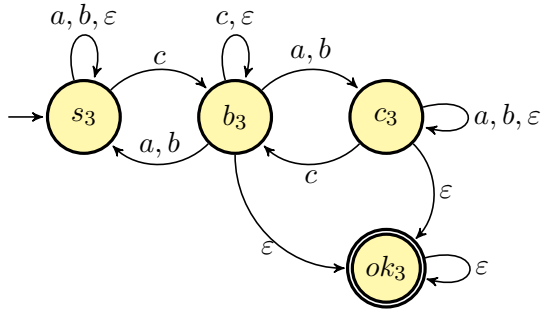
Příklad 4.2.0.34 (Převod 3-MAS konečných automatů z příkladu 4.1.0.21 na ekvivalentní 3-MAT konečných automatů)
Uvažujeme tedy 3-MAS ϑ z příkladu 4.1.0.21. Rozšířením komponent dle algoritmu 4.2.0.33 tedy dostaneme automaty:

- M'_1 :





- M_2' :



- M_3' :

Dále uvažujme zobrazení $R_{In} : M \times Q \to Q \times Q \times (\Sigma \cup \{\varepsilon\})$, kde M je konečný automat, definované analogicky k R_{In} nad zásobníkovými automaty (def. 4.2.0.31).

Množiny P a P' zkonstruované dle algoritmu pak budou:

$\overline{P} = \{(i_s, i_s, i_s), (l_2, i_s, i_s), (i_b, i_s, i_s), (i_s, l_2, i_s), , (i_s, i_b, i_s), (i_s, i_s, l_2), (i_s, i_s, i_b), (l_2, l_2, i_s), (i_b, i_b, i_s), (l_2, i_s, l_2), , (i_b, i_s, i_b), (i_s, l_2, l_2), (i_s, i_b, i_b), (i_b, i_b, i_b), (l_4, i_c, i_c), (i_{ok}, i_c, i_c), (l_4, l_4, i_c), (i_{ok}, i_{ok}, i_c), (i_c, l_4, i_c), (i_c, i_{ok}, i_c), (i_c, l_4, l_4)$, $(i_c, i_{ok}, i_{ok}), (l_4, i_c, l_4), (i_{ok}, i_c, i_{ok})$ | pro

$M_1' : l_1 = (s_1, s_1, \varepsilon), l_2 = (b_1, b_1, \varepsilon), l_3 = (c_1, c_1, \varepsilon), l_4 = (ok_1, ok_1, \varepsilon), i_s \in R_{In}(M_1, s_1), i_b \in R_{In}(M_1, b_1), i_c \in R_{In}(M_1, c_1), i_{ok} \in R_{In}(M_1, ok_1),$

$M_2' : l_1 = (s_2, s_2, \varepsilon), l_2 = (b_2, b_2, \varepsilon), l_3 = (c_2, c_2, \varepsilon), l_4 = (ok_2, ok_2, \varepsilon), i_s \in R_{In}(M_2, s_2), i_b \in R_{In}(M_2, b_2), i_c \in R_{In}(M_2, c_2), i_{ok} \in R_{In}(M_2, ok_2),$

$M_3' : l_1 = (s_3, s_3, \varepsilon), l_2 = (b_3, b_3, \varepsilon), l_3 = (c_3, c_3, \varepsilon), l_4 = (ok_3, ok_3, \varepsilon), i_s \in R_{In}(M_3, s_3), i_b \in R_{In}(M_3, b_3), i_c \in R_{In}(M_3, c_3), i_{ok} \in R_{In}(M_3, ok_3)$ },

$P = \{r \to (e, e, e) |\ r \in \overline{P}\},$

$P' = \{r \to (d, d, d) |\ r \notin \overline{P}\},$

Konečně množina Ψ' je definována jako sjednocení množin P a P'.

Vzniklý automatový systém můžeme tedy zapsat jako $\vartheta' = (M_1', M_2', M_3', (e, e, e), \Psi')$, kde $M_1,'$ M_2' a M_3' jsou konečné automaty definované grafem ze začátku příkladu.

Stejným způsobem lze převést jakýkoliv n-MAS na n-MAT. Přes fakt, že je ekvivalence odpovídajících si typů jazyků, resp. multijazyků zřejmá, formální důkaz ponechávám pro budoucí studia, či zájemcům o tuto problematiku.

Multipřijímací automatové systémy jsou přímou návazností na práci Ing. Romana Lukáše, PhD., který zavedl kanonické n-gramatické systémy. Úzkou souvislost mezi těmito systémy lze lépe vidět na algoritmu 4.2.0.35, který převádí KGN na MAS zásobníkových automatů.

Algoritmus 4.2.0.35
Převod kanonického n-generativního non-terminálově synchronizovaného gramatického systému na ekvivalentní n-přijímající stavem řízený automatový systém nad zásobníkovými automaty.

Vstup: Kanonický n-generativní non-terminálově synchronizovaný gramatický systém, zapsáno $\widehat{\Gamma} = (G_1, \ldots, G_n, \widehat{Q})$, kde $\forall i = 1, \ldots, n$ $G_i = (\widehat{N}_i, \widehat{T}_i, \widehat{P}_i, \widehat{S}_i)$ je bezkontextová gramatika generující řetězce pomocí nejlevějších derivací.

Výstup: n-MAS $\vartheta = (M_1, \ldots, M_n, \Psi, S)$, kde $\forall i = 1 \ldots n$, $M_i = (Q_i, \Sigma, \Gamma_i, \delta_i, s_i, z_{i,0}, F_i)$ je zásobníkový automat přijímající s přečtením vstupního řetězce s vyprázdněním zásobníku.

Metoda:

- $\forall 1 \leq i \leq n : Q_i = \{q_j^i | 0 \leq j \leq m, m = card(\widehat{N}_i) + 1\}$,
- $\Sigma = \bigcup_{i=1}^{n} \widehat{T}_i$,
- $\forall 1 \leq i \leq n$: $\Gamma^i = \widehat{T}_i \cup \widehat{N}_i \cup \{*_j | 0 \leq j \leq m, m = card(\widehat{N}_i) - 1\} \cup \{\Delta, \Delta'\}$,
- $s_i \in NSMap_i(\widehat{S}_i)$,
- $z_{i,0} = \Delta'$,
- $F_i = \emptyset$,
- δ_i: Zaveďme pomocné množiny a funkce potřebné k popisu algoritmu:
  - $\mathbf{Spec_i} = \{x \in \Gamma_i | x \notin \widehat{T}_i \wedge x \notin \widehat{N}_i\}$,
  - $\mathbf{NTSpecMap_i}$: $\widehat{N}_i \to Spec_i\Gamma_i$, kde $\forall x \in Spec_i, \forall A, B \in \widehat{N}_i$:
    - $(NTSpecMap_i(A) = xB) \Rightarrow (A = B)$,
    - $((NTSpecMap_i(A) = xA) \wedge (NTSpecMap_i(B) = xB)) \Rightarrow (A = B)$,
  - $\mathbf{SpecSMap_i}$: $Spec_i \to Q_i \backslash \{q^i_{card(\widehat{N}_i)}, q^i_{card(\widehat{N}_i)+1}\}$, kde:
    $\forall q \in Q_i \backslash \{q^i_{card(\widehat{N}_i)}, q^i_{card(\widehat{N}_i)+1}\}$ a $\forall x, y \in Spec_i$:
    - $((SpecSMap_i(x) = q) \wedge (SpecSMap_i(y) = q)) \Rightarrow (x = y)$,
  - $\mathbf{NSMap_i}$: $\widehat{N}_i \to Q_i \backslash \{q^i_{card(\widehat{N}_i)}, q^i_{card(\widehat{N}_i)+1}\}$:
    $\forall q \in Q_i \backslash \{q^i_{card(\widehat{N}_i)}, q^i_{card(\widehat{N}_i)+1}\}$, $x \in Spec_i$, $A \in \widehat{N}_i$:
    - $(NSMap_i(A) = q) \Rightarrow (SpecSMap_i(x) = q \wedge NTSpecMap_i(A) = xA)$,
  - $\mathbf{SStream_i}$: $\omega \in (\widehat{N}_i \cup \widehat{T}_i)^* \to \omega' \in (\{NTSpecMap_i(\widehat{N}_i)\} \cup \widehat{T}_i)^*$, kde ω' vznikne nahrazením všech výskytů nonterminálních symbolů, $\forall A \in \widehat{N}_i$ v ω, za řetězec $NTSpecMap_i(A)$,
  - $\forall a \in \widehat{T}_i, \forall b \in Spec_i$, $\forall q \in Q_i \backslash \{q^i_{card(\widehat{N}_i)}, q^i_{card(\widehat{N}_i)+1}\}$, $r \in \{q^i_{card(\widehat{N}_i)}\}$:
    - $(r, a) \in \delta_i(q, a, \varepsilon)$,
    - $(r, b) \in \delta_i(q, b, \varepsilon)$,
    - $(r, \varepsilon) \in \delta_i(r, a, a)$,
    - $(SpecSMap_i(b), \varepsilon) \in \delta_i(r, b, \varepsilon)$,
    - $(q^i_{card(\widehat{N}_i)+1}, \varepsilon) \in \delta_i(q, \Delta, \varepsilon)$,

- $(SpecSMap_i(\widehat{S}_i), \Delta\widehat{S}_i) \in \delta_i(SpecSMap_i(\widehat{S}_i), \Delta', \varepsilon)$,
- $(q^i_{card(\widehat{N}_i)+1}, \varepsilon) \in \delta_i(q_{card(\widehat{N}_i)}, \Delta, \varepsilon)$
- $\forall(A \to \alpha) \in P_i$, $(NSMap_i(A), SStream_i(\alpha)) \in \delta_i(NSMap_i(A), A, \varepsilon)$,

- Ψ :
  - $Q_{j_{work}} = Q_j \backslash \{q^j_{card(\widehat{N}_j)}, q^j_{card(\widehat{N}_j)+1}\}$,
  - $EN = \{(NSMap_1(A_1), \ldots, NSMap_n(A_n)) \to (\overbrace{e, \ldots, e}^{n\times}) | (A_1, \ldots, A_n) \in \widehat{Q}\}$,
  - $PD = \{(q^1, \ldots, q^n) \to (d_1, \ldots, d_n) | \forall 1 \leq j \leq n : q^j \in (Q_{j_{work}} \cup \{q^j_{card(\widehat{N}_j)}\}) \wedge d_j \in \{e_j, d_j\} \wedge ((q^j \in Q_{j_{work}}) \Leftrightarrow (d_j = e_j)) \wedge \exists 1 \leq l \leq n : q^l \notin \{q^l_{card(\widehat{N}_l)}\} \wedge \exists((q'^1, \ldots, q'^n) \to (\overbrace{e, \ldots, e}^{n\times})) \in EN : (\forall 1 \leq k \leq n((q^k \in Q_{k_{work}}) \Leftrightarrow (q'^k = q^k)))\}$,
  - $PA = \{(q^1, \ldots, q^n) \to (d_1, \ldots, d_n) | \forall 1 \leq j \leq n : q^j \in \{q^j_{card(\widehat{N}_j)}, q^j_{card(\widehat{N}_j)+1}\} \wedge d_j \in \{e_j, d_j\} \wedge ((q^j \in \{q^j_{card(\widehat{N}_j)}\}) \Leftrightarrow (d_j = e_j)) \wedge \exists 1 \leq l \leq n : q^l \notin \{q^l_{card(\widehat{N}_l)+1}\}\}$,
  - $BL = \{(q^1, \ldots, q^n) \to (d_1, \ldots, d_n) | \forall 1 \leq j \leq n : q^j \in Q_j \wedge d_j \in \{d_j, e_j\} \wedge (q^1, \ldots, q^n) \to (d_1, \ldots, d_n) \notin (EN \cup PD \cup PA \cup \{(q^1, \ldots, q^n) \to (d_1, \ldots, d_n) | \forall 1 \leq j \leq n : q^j \in Q_j \backslash Q_{j_{work}}\})\}$,
  - $\Psi = (EN \cup PD \cup PA \cup BL)$,
- $S = (\overbrace{e, \ldots, e}^{n\times})$.

Hlavní myšlenkou algoritmu je reprezentovat každý non-terminál vlastním stavem, ze kterého může být příslušný non-terminál expandován. Do systému jsou navíc přidány dva stavy. První z nich se stará o přijímání vstupních symbolů ze vstupu (tj. případ, kdy je na vrcholu zásobníku terminální symbol z KGN). Druhý z nich naopak reprezentuje stav, kdy všechny znaky ze vstupu byly přečteny a zásobník zůstal prázdný (tj. automat řetězec přijal). Na příkladu 4.2.0.36 si opět ukážeme aplikaci algoritmu.

Příklad 4.2.0.36
Mějme 2-KGN z příkladu 3.1.9 v [4] definovaný trojicí $\widehat{\Gamma} = (G_1, G_2, \widehat{Q})$, kde:

- $G_1 = (\{S_1, A_1\}, \{a, b, c\}, \{S_1 \to aS_1,\ S_1 \to aA_1,\ A_1 \to bA_1c,\ A_1 \to bc\}, S_1)$,
- $G_2 = (\{S_2, A_2\}, \{d\}, \{\ S_2 \to S_2A_2,\ S_2 \to A_2,\ A_2 \to d\}, S_2)$
- $\widehat{Q} = \{(S_1, S_2), (A_1, A_2)\}$.

K němu zkonstruovaný 2-MAS ϑ dle algoritmu 4.2.0.35 bude reprezentován čtveřicí 2-MAS $\vartheta = (M_1, M_2, (e, e), \Psi)$, kde:

- $M_1 = (\{q_{S_1}, q_{A_1}, q_c, q_f\}, \{a, b, c, d\}, \{a, b, c, *_{A_1}, *_{S_1}, A_1, S_1, \Delta, \Delta'\}, \delta_1, q_{S_1}, \Delta', \emptyset)$,
- $M_2 = (\{q_{S_2}, q_{A_2}, q_c, q_f\}, \{a, b, c, d\}, \{d, *_{A_2}, *_{S_2}, A_2, S_2, \Delta, \Delta'\}, \delta_2, q_{S_2}, \Delta', \emptyset)$,
- Ψ :

$$P = \{(q_{S_1}, q_{S_2}) \to (e,e),\ (q_{A_1}, q_{A_2}) \to (e,e),\ (q_{S_1}, q_c) \to (d,e), (q_c, q_{S_2}) \to (e,d),\ (q_{A_1}, q_c) \to (d,e),\ (q_c, q_{A_2}) \to (e,d),\ (q_c, q_c) \to (e,e),\ (q_c, q_f) \to (e,d),\ (q_f, q_c) \to (d,e),\ (q_f, q_f) \to (e,e)\},$$

$$P' = \{(q_1, q_2) \to (d,d) |\ (q_1, q_2) \to (\not{d}_1, \not{d}_2) \notin P \text{ pro žádné } \not{d}_1, \not{d}_2 \in \{e,d\}\},$$

$$\Psi = P \cup P',$$

- hrany:

δ_1:	δ_2:
$(q_{S_1}, S_1\Delta) \in \delta_1(q_{S_1}, \Delta', \varepsilon)$	$(q_{S_2}, S_2\Delta) \in \delta_2(q_{S_2}, \Delta', \varepsilon)$
$(q_{S_1}, a *_{S_1} S_1) \in \delta_1(q_{S_1}, S_1, \varepsilon)$	$(q_{S_2}, a *_{S_2} S_1 *_{A_2}) \in \delta_2(q_{S_2}, S_2, \varepsilon)$
$(q_{S_1}, a *_{A_1} A_1) \in \delta_1(q_{S_1}, S_1, \varepsilon)$	$(q_{S_2}, *_{A_2} A_2) \in \delta_2(q_{S_2}, S_2, \varepsilon)$
$(q_{A_1}, b *_{A_1} A_1 c) \in \delta_1(q_{A_1}, A_1, \varepsilon)$	$(q_{A_2}, d) \in \delta_2(q_{A_2}, A_2, \varepsilon)$
$(q_{A_1}, bc) \in \delta_1(q_{A_1}, A_1, \varepsilon)$	
$(q_c, \varepsilon) \in \delta_1(q_c, a, a)$	$(q_c, \varepsilon) \in \delta_2(q_c, a, a)$
$(q_c, \varepsilon) \in \delta_1(q_c, b, b)$	$(q_c, \varepsilon) \in \delta_2(q_c, b, b)$
$(q_c, \varepsilon) \in \delta_1(q_c, c, c)$	$(q_c, \varepsilon) \in \delta_2(q_c, c, c)$
$(q_f, \varepsilon) \in \delta_1(q_c, \Delta, c)$	$(q_f, \varepsilon) \in \delta_2(q_c, \Delta, c)$
$(q_f, \varepsilon) \in \delta_1(q_{S_1}, \Delta, \varepsilon)$	$(q_f, \varepsilon) \in \delta_1(q_{S_2}, \Delta, \varepsilon)$
$(q_f, \varepsilon) \in \delta_1(q_{A_1}, \Delta, \varepsilon)$	$(q_f, \varepsilon) \in \delta_1(q_{A_2}, \Delta, \varepsilon)$
$(q_{A_1}, \varepsilon) \in \delta_1(q_{S_1}, *_{A_1}, \varepsilon)$	$(q_{A_2}, \varepsilon) \in \delta_2(q_{S_2}, *_{A_2}, \varepsilon)$
$(q_{A_1}, \varepsilon) \in \delta_1(q_{A_1}, *_{A_1}, \varepsilon)$	$(q_{A_2}, \varepsilon) \in \delta_2(q_{A_2}, *_{A_2}, \varepsilon)$
$(q_{A_1}, \varepsilon) \in \delta_1(q_c, *_{A_1}, \varepsilon)$	$(q_{A_2}, \varepsilon) \in \delta_2(q_c, *_{A_2}, \varepsilon)$
$(q_{S_1}, \varepsilon) \in \delta_1(q_{S_1}, *_{S_1}, \varepsilon)$	$(q_{S_2}, \varepsilon) \in \delta_2(q_{S_2}, *_{S_2}, \varepsilon)$
$(q_{S_1}, \varepsilon) \in \delta_1(q_{A_1}, *_{S_1}, \varepsilon)$	$(q_{S_2}, \varepsilon) \in \delta_2(q_{A_2}, *_{S_2}, \varepsilon)$
$(q_{S_1}, \varepsilon) \in \delta_1(q_c, *_{S_1}, \varepsilon)$	$(q_{S_2}, \varepsilon) \in \delta_2(q_c, *_{S_2}, \varepsilon)$

Ve vytvořeném 2-MAS existují například tyto sekvence přechodů pro řetězce generované původním 2-KGN:

- $(S_1, S_2) \Rightarrow (aA_1, A_2) \Rightarrow (abc, d)$:
  $((q_{S_1}, \Delta', abc)^e, (q_{S_2}, \Delta', d)^e) \vdash ((q_{S_1}, S_1\Delta, abc)^e, (q_{S_2}, S_2\Delta, d)^e)$
  $\vdash ((q_{S_1}, a *_{A_1} A_1\Delta, abc)^e, (q_{S_2}, *_{A_2} A_2\Delta, d)^e)$
  $\vdash ((q_c, a *_{A_1} A_1\Delta, abc)^e, (q_{A_2}, A_2\Delta, d)^d) \vdash$
  $\vdash ((q_c, *_{A_1} A_1\Delta, bc)^e, (q_{A_2}, A_2\Delta, d)^d)$
  $\vdash ((q_{A_1}, A_1\Delta, bc)^e, (q_{A_2}, A_2\Delta, d)^e)$
  $\vdash ((q_{A_1}, bc\Delta, bc)^e, (q_{A_2}, d\Delta, d)^e)$
  $\vdash ((q_c, bc\Delta, bc)^e, (q_c, d\Delta, d)^e)$
  $\vdash ((q_c, c\Delta, c)^e, (q_c, \Delta, \varepsilon)^e)$
  $\vdash ((q_c, \Delta, \varepsilon)^e, (q_f, \varepsilon, \varepsilon)^d)$
  $\vdash ((q_f, \varepsilon, \varepsilon)^e, (q_f, \varepsilon, \varepsilon)^e)$

- $(S_1, S_2) \Rightarrow (aS_1, S_2A_2) \Rightarrow (aaA_1, A_2A_2) \Rightarrow (aabA_1c, dA_2) \Rightarrow (aabbcc, dd)$
  $((q_{S_1}, \Delta', aabbcc)^e, (q_{S_2}, \Delta', dd)^e) \vdash ((q_{S_1}, S_1\Delta, aabbcc)^e, (q_{S_2}, S_2\Delta, dd)^e)$
  $\vdash ((q_{S_1}, a *_{S_1} S_1\Delta, aabbcc)^e, (q_{S_2}, *_{S_2} S_2 *_{A_2} A_2\Delta, dd)^e)$
  $\vdash ((q_c, a *_{S_1} S_1\Delta, aabbcc)^e, (q_{S_2}, S_2 *_{A_2} A_2\Delta, dd)^d)$
  $\vdash ((q_c, *_{S_1} S_1\Delta, abbcc)^e, (q_{S_2}, S_2 *_{A_2} A_2\Delta, dd)^d)$
  $\vdash ((q_{S_1}, S_1\Delta, abbcc)^e, (q_{S_2}, S_2 *_{A_2} A_2\Delta, dd)^e)$

$\vdash ((q_{S_1}, a *_{A_1} A_1\Delta, abbcc)^e, (q_{S_2}, *_{A_2} A_2 *_{A_2} A_2\Delta, dd)^e)$
$\vdash ((q_c, a *_{A_1} A_1\Delta, abbcc)^e, (q_{A_2}, A_2 *_{A_2} A_2\Delta, dd)^d)$
$\vdash ((q_c, *_{A_1} A_1\Delta, bbcc)^e, (q_{A_2}, A_2 *_{A_2} A_2\Delta, dd)^d)$
$\vdash ((q_{A_1}, A_1\Delta, bbcc)^e, (q_{A_2}, A_2 *_{A_2} A_2\Delta, dd)^e)$
$\vdash ((q_{A_1}, b *_{A_1} A_1c\Delta, bbcc)^e, (q_{A_2}, d *_{A_2} A_2\Delta, dd)^e)$
$\vdash ((q_c, b *_{A_1} A_1c\Delta, bbcc)^e, (q_c, d *_{A_2} A_2\Delta, dd)^e)$
$\vdash ((q_c, *_{A_1} A_1c\Delta, bcc)^e, (q_c, *_{A_2} A_2\Delta, d)^e)$
$\vdash ((q_{A_1}, A_1c\Delta, bcc)^e, (q_{A_2}, A_2\Delta, d)^e)$
$\vdash ((q_{A_1}, bcc\Delta, bcc)^e, (q_{A_2}, d\Delta, d)^e)$
$\vdash ((q_c, bcc\Delta, bcc)^e, (q_c, d\Delta, d)^e)$
$\vdash ((q_c, cc\Delta, cc)^e, (q_c, \Delta, \varepsilon)^e)$
$\vdash ((q_c, c\Delta, c)^e, (q_f, \varepsilon, \varepsilon)^d)$
$\vdash ((q_c, \Delta, \varepsilon)^e, (q_f, \varepsilon, \varepsilon)^d)$
$\vdash ((q_f, \varepsilon, \varepsilon)^e, (q_f, \varepsilon, \varepsilon)^d)$

- ...

Při bližší analýze lze snadno usoudit vztah mezi původním 2-KGN a výsledným 2-MAS:

- 2-$L_\cap(\vartheta) = \{(a^n b^n c^n, d^n) |\ n \geq 1\} = 2\text{-}L(\widehat{\Gamma})$,
- $msunion(\ 2\text{-}L_\cap(\vartheta)\ ) = \{a^n b^n c^n |\ n \geq 1\} \cup \{d^n |\ n \geq 1\} = L_{union}(\widehat{\Gamma})$,
- $msconcat(\ 2\text{-}L_\cap(\vartheta)\ ) = \{a^n b^n c^n d^n |\ n \geq 1\} = L_{conc}(\widehat{\Gamma})$,
- $msfirst(\ 2\text{-}L_\cap(\vartheta)\ ) = \{a^n b^n c^n |\ n \geq 1\} = L_{first}(\widehat{\Gamma})$.

Odsud je vidět, že pro ekvivalenci MAS s KGN připadá v úvahu pouze pro n-jazyky sjednocení, potažmo s jazyky nad těmito n-jazyky definovanými. Nicméně z teoretického i praktického hlediska mohou být zajímavé i ostatní módy přijímaní multiřetězců, kdy například bude plně postačovat přijetí řetězce jednou komponentou, přičemž ostatní komponenty budou pouze řídícími prvky systému.

Převést na ekvivalnetní n-MAS lze libovolný n-KGN. Důkazem tohoto tvrzení bude dokázáno i to, že n-přijímající automatové systémy mají sílu Turingova stroje. Formální důkaz tvrzení opět ponechávám budoucímu vývoji, popřípadě zájemcům o tuto problematiku.

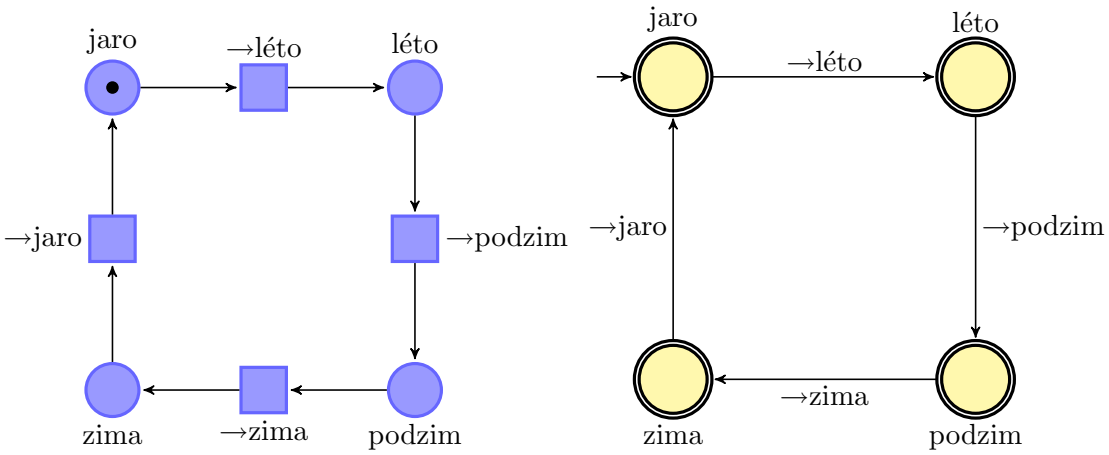
Kapitola 5

Multipřijímací automatové systémy jako C/E systém

Multipřijímací automatové systémy mohou mít veliký potenciál v různých oblastech nejen informačních technologií. Jednou z mnoha možností využití může být například popis Petriho sítí, které jsou dnes hojně využívané v mnoha směrech.

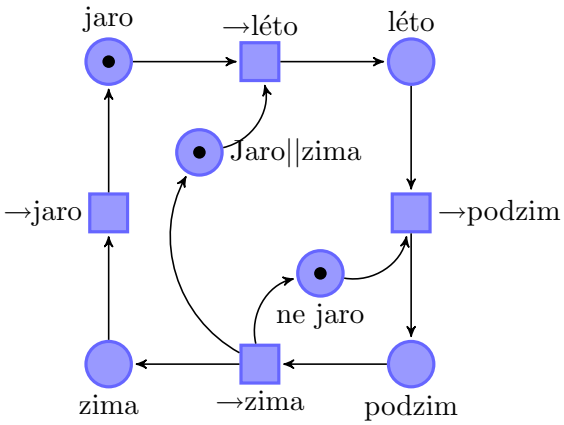
C/E Petriho sítě se skládají z množiny míst (graficky znázorněných kolečkem), množiny událostí (značených čtvercem) a tokové relace mezi těmito množinami takové, že žádná událost nesmí být v relaci s žádnou událostí a žádné místo nesmí být v tokové relaci žádným místem sítě. Každé z míst má jednu, nebo žádnou značku. Událost je proveditelná, pokud všechna místa, která jsou v tokové relaci s touto událostí (vstupní místa, podmínky) mají značku a zároveň ji nemá ani jedno z míst, s nimž je v relaci uvažovaná událost (výstupní místa události, důsledky). Po provedení události není značka v žádném ze vstupních míst a naopak ji mají všechna místa výstupní. Přesnější definice jsou uvedeny níže nebo v [11].

K realizaci lze využít například n-MAS, nebo n-MAT systémy konečných automatů. Podíváme-li se na levou část obrázku 5.1, reprezentujícího jednoduchou C/E síť, lze v něm snadno nalézt konečný automat, který je znázorněn v pravé části obrázku 5.1.



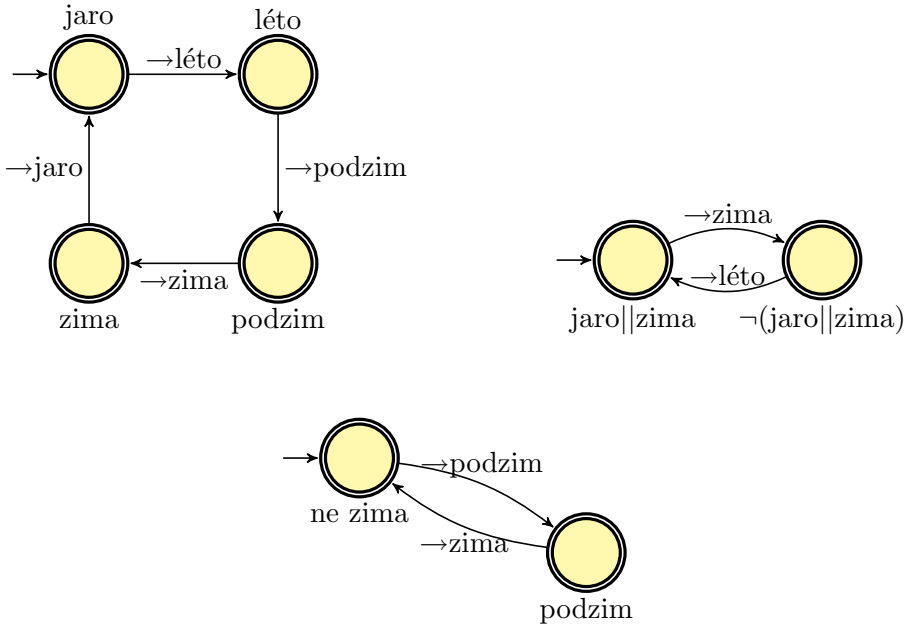
Obrázek 5.1: C/E síť ročního období s odpovídajícím automatem

Uvažujme ale například síť z obrázku 5.2. Zde by už vykonstruování konečného automatu bylo složité. V síti mohou být značky až ve třech místech současně. Konečný automat



Obrázek 5.2: druhá C/E síťě reprezentující běh ročního období

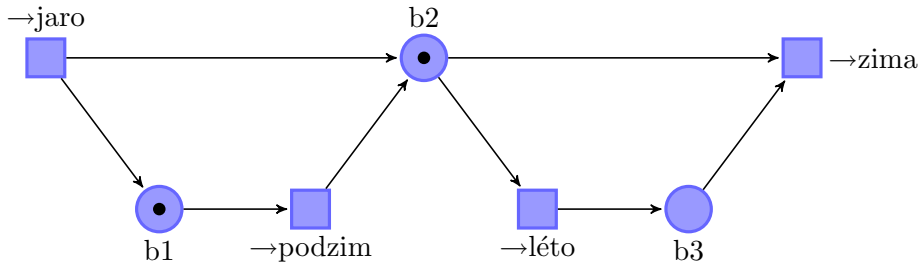
může být v jeden okamžik pouze v jednom z jeho stavů. Abychom pokryli všechny možné případy sítě, potřebovali bychom zkonstruovat automat s často mnohem větším počtem stavů, než je míst v C/E síti. Síť ovšem můžeme rozdělit na podsítě a každou z těchto podsítí reprezentovat menším konečným automatem, viz obr. 5.3. Pokud se automaty složí



Obrázek 5.3: Tři konečné automaty odpovídající podcestám C/E sítě obr. 5.2

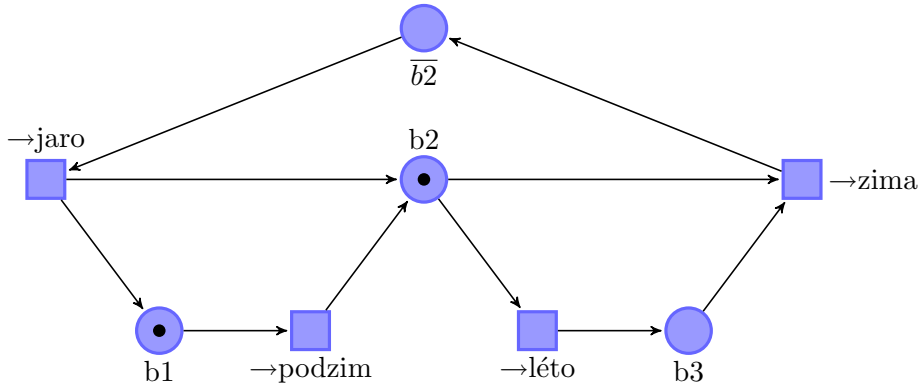
do systému s vhodným n-MAS, resp. n-MAT řízením tak, aby stavy se stejným názvem reprezentovali jedno místo C/E sítě, pak systém přijímá jazyk sekvencí událostí, proveditelných C/E systémem.

Obrázek 5.4 znázorňuje další variantu C/E sítě opět reprezentující změny ročních období. I pro tuto síť je nutné k reprezentaci Konečnými automaty použít více strojů. Na první pohled zde nemusí být souvislost s n-MAS, resp. n-MAT zřejmá, proto rozšiřme síť o



Obrázek 5.4: opět jiná možnost C/E sítě reprezentující cyklus ročních období

stav $\overline{b2}$, viz obr. 5.5. Rozšiřující stav pak bude označovat případ, kdy není v systému žádná značka. Nyní je snadné vydedukovat, že by se k popisu sítě využil 2-MAS, resp. 2-MAT.



Obrázek 5.5: opět jiná možnost C/E sítě reprezentující cyklus ročních období

Obecně bude platit, že minimální počet automatů, popisujících nějaký C/E systém, bude odpovídat maximálnímu počtu značek, který se může v systému objevit v jeden okamžik.

5.1 Konstrukce analyzátoru C/E systémů z C/E sítí

Než přistoupíme k samotné konstrukci analyzátoru jazyka, definovaného nějakým C/E systémem, je nezbytné zavést základní definice, které budou podkladem pro samotnou konstrukci. Pro více informací o C/E systémech opět doporučuji [11].

5.1.1 Základní definice

Definice 5.1.1.1 (C/E síť, toková relace)
Trojici $N = (B, E, F)$ nazveme C/E Petriho sítí jestliže:

- $B \cap E = \emptyset$, kde B je konečná množina míst a E je konečná množina událostí,
- $F \subseteq (B \times E) \cup (E \times B)$ je binární relací, již budeme nazývat relací toku, nebo-li tokovou relací,

Množinou B se chápe množina míst (chcete-li stavů) sítě, která se graficky znázorňuje kolečkem. Naproti tomu množina e je množina událostí (přechodů) graficky označovaná čtverečkem. Toková relace umožňuje propojit pouze kolečko se čtvercem a naopak. Nemůže tedy nikdy dojít ke stavu, kdy jsou přímo propojená dvě kolečka, nebo dva čtverce.

Definice 5.1.1.2 (Preset, postset)
Nechť $N = (B, E, F)$ je C/E síť. Pak řekneme, že:

- ${}^\bullet x = \{y |\ yFx,\ x, y \in (B \cup E)\}$ je *presetem* prvku x,
- $x^\bullet = \{y |\ xFy,\ x, y \in (B \cup E)\}$ je *postsetem* prvku x.

Presetem je tedy množina všech prvků vcházejících do prvku x a naopak postsetem se chápou prvky z x vycházející.

Definice 5.1.1.3 (Případ, proveditelnost, následník, izolovaný prvek)
Nechť $N = (B, E, F)$ je C/E síť.

a) Podmnožinu $c \subseteq B$ nazveme *případem* sítě N,

b) Nechť $e \in E$ a $c \subseteq B$. Událost je *proveditelná*, nebo-li *c-proveditelná*, pokud ${}^\bullet e \subseteq c$ a současně $e^\bullet \cap c = \emptyset$.

c) Nechť $e \in E$, $c \subseteq B$ a e je c-proveditelná. Případ $c' = (c \backslash {}^\bullet e) \cup e^\bullet$ nazveme *následníkem* případu c při události e a zapisujeme $c[e > c'$.

d) Prvek $x \in (B \cup E)$ se nazývá *izolovaný prvek*, pokud ${}^\bullet x \cup x^\bullet = \emptyset$.

Definice 5.1.1.4 (Nezávislá množina, krok)
Nechť $N = (B, E, F)$ je C/E síť.

- Množina událostí $G \subseteq E$ nazveme *nezávislou*, pokud $\forall e_1, e_2 \in G : {}^\bullet e_1 \cap {}^\bullet e_2 = \emptyset = e_1^\bullet \cap e_2^\bullet$.
- Nechť c, c' jsou případy sítě N a nechť G je nezávislou množinou událostí. G nazveme *krokem*, pokud každá událost $e \in G$ je c-proveditelná a $c' = (c \backslash {}^\bullet G) \cup G^\bullet$, přičemž ${}^\bullet G = \bigcup_{e \in G} {}^\bullet e$ a $G^\bullet = \bigcup_{e \in G} e^\bullet$.

Definice 5.1.1.5 (C/E systém)
C/E systém definujeme jako čtveřici $\Sigma_{C/E\ sys} = (B, E, F, C)$, kde:

- (B,E,F) je jednoduchou (tj. $\forall x, y \in (B \cup E),\ xFy \Leftarrow \neg(yFx)$) bez izolovaných prvků a $B \cup E \neq \emptyset$,
- $C \subseteq 2^B$ je ekvivalenční třídou vzhledem k relaci dosažitelnosti danou vztahem $R_\Sigma = (r_\Sigma \cup r_\Sigma^{-1})$, kde $r_\Sigma \subseteq 2^B \times 2^B$ a $c_1 r_\Sigma c_2 \Leftrightarrow \exists G \subseteq E :\ c_1[G > c_2$. C nazveme případovou třídou sítě $\Sigma_{C/E\ sys}$,
- $\forall e \in E\ \exists c \in C$, takové, že e je c-proveditelná.

Nyní si uvedeme jeden z možných algoritmů pro C/E systémů na ekvivalentní n-MAS konečných automatů, který bude přijímat proveditelné sekvence událostí.

Algoritmus 5.1.1.6
Konstrukce analyzátoru jazyku událostí definovaného C/E systémem.

- **Vstup:** C/E systém $\Sigma_{C/E\ sys} = (B, E, F, C)$,
- **Výstup:** n-MAS $\vartheta = (M_1, \ldots, M_n)$.
- **Metoda**

1. Vytvořme třídu rozkladu H pro P takovou, že pro všechna $h \in H$ a $c \in C$ platí:
   (a) $p_1, p_2 \in h \Leftrightarrow p_1, p_2 \notin c$,
   (b) pro každé $p_1 \in h$, $\exists p_2 \in h$ takové, že (p_1, t), $(t, p_2) \in F$ nebo (p_2, t), $(t, p_1) \in F$ pro nějaké $t \in T$, nebo $card(h) = 1$.
2. Pro každé $h_i \in H$ vytvořme konečný automat $M_i = (Q_i, \Sigma, \delta_i, s_i, F_i)$ následovně:
   (a) Q_i :
      – pro každé $p \in h_i$ existuje $p \in Q_i$,
      – pokud existuje $c \in C$ takové, že $p \notin c$ pro všechna $p \in h_i$, pak do Q_i přidej stav r, jenž bude reprezentovat, že podsíť neobsahuje značku.
   (b) $\Sigma = T$,
   (c) δ_i:
      – pro každé $p_1, p_2 \in h_i$, $(p_2) \in \delta_i(p_1, t)$, kde $t \in T$ takové, že $(p_1, t), (t, p_2) \in F$,
      – pro každé $p_1, p_2 \in h_i$, $(p_2) \in \delta_i(p_1, \varepsilon)$[1],
      – pokud existuje $r \in Q_i$ takové, že $r \notin h_i$, pak $(r) \in \delta_i(p_1, t_1) \wedge (p_2) \in \delta_i(r, t_2)$ pro $p_1, p_2 \in h_i$, $t_1, t_2 \in T$: $(t_2, p_2) \in F$, $(p_1, t_1) \in F$ a současně $card({}^\bullet t_1) > 1$ a $card(t_2^\bullet) > 1$,
   (d) s_i - dle vybraného případu $c \in C$,
   (e) $F_i = Q_i$.
3. Množinu Ψ zkonstruujeme tak, aby zachovávala současný přechod přes hrany beroucí ze vstupu shodnou událost, přičemž nesmí docházet ke kontaktní situaci (tj. přechod je proveditelný, pokud by byl proveditelný i ve vstupním C/E systému). V opačném případě jsou komponenty splňující podmínku přechodu blokovány, dokud ji nesplňují i ostatní komponenty.

Uvedený algoritmus je spíše ukázkový, přičemž má poukázat na souvislost mezi C/E systémy a multipřijímacími systémy.

Příklad 5.1.1.7
Uvažujme C/E systém z obrázku 5.4. Je zřejmé, že případová třída je

$$C = \{\{b_1, b_2\}, \{b_1, b_3\}, \{b_2, b_3\}, \{\}\}.$$

Třída rozkladu H bude

$$H = \{\{b_1\}, \{b_2\}, \{b_3\}\}.$$

MAS ϑ tedy bude tvořit trojice konečných automatů:

[1] Tvoří doplňující přechodové hrany, bez kterých by systém prováděl pouze maximální kroky

- $M_1 = (\{b_1, r_1\}, \{\rightarrow\text{jaro}, \rightarrow\text{podzim}\}, \delta_1, s_1, \{b_1, r_1\}\}$,
- $M_2 = (\{b_2, r_2\}, \{\rightarrow\text{jaro}, \rightarrow\text{zima}\}, \delta_2, s_2, \{b_2, r_2\}\}$,
- $M_3 = (\{b_3, r_3\}, \{\rightarrow\text{léto}, \rightarrow\text{zima}\}, \delta_3, s_3, \{b_3, r_3\}\}$,
- Hrany:

  δ_1:
  $b_1 \in \delta_1(r_1, \rightarrow\text{jaro})$
  $b_1 \in \delta_1(b_1, \varepsilon)$
  $r_1 \in \delta_1(b_1, \rightarrow\text{podzim})$
  $r_1 \in \delta_1(r_1, \varepsilon)$

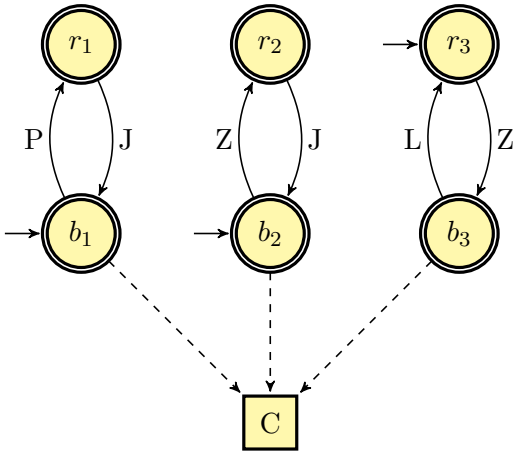
  δ_2:
  $b_2 \in \delta_2(r_2, \rightarrow\text{jaro})$
  $b_2 \in \delta_2(b_2, \varepsilon)$
  $r_2 \in \delta_2(b_2, \rightarrow\text{zima})$
  $r_2 \in \delta_1(r_2, \varepsilon)$

  δ_3:
  $b_3 \in \delta_3(r_3, \rightarrow\text{zima})$
  $b_3 \in \delta_3(b_3, \varepsilon)$
  $r_3 \in \delta_3(b_3, \rightarrow\text{léto})$
  $r_3 \in \delta_3(r_3, \varepsilon)$

- Ψ :
  - $P = \{(b_1, b_2, r_3) \rightarrow (d, d, e), (b_1, b_2, b_3) \rightarrow (e, d, d), (r_1, b_2, b_3) \rightarrow (d, e, e), (r_1, r_2, r_3) \rightarrow (e, e, d)\}$,
  - $P' = \{(q_1, q_2, q_3) \rightarrow (d, d, d) |\ \forall i = 1, 2, 3 : \quad q_i \in Q_i, (q_1, q_2, q_3) \rightarrow (d_1, d_2, d_3) \notin P$ pro všechna $d_i \in \{e, d\}\}$
  - $\Psi = P \cup P'$.

Odtud je vidět, že vzniklý 3-MAS přijímá multijazyk:
3-$L_\cap(\vartheta) = \{(\rightarrow\text{podzim}\rightarrow\text{jaro})^m, (\rightarrow\text{zima}\rightarrow\text{jaro})^m, (\rightarrow\text{zima}\rightarrow\text{léto})^m\}$
Systém pak bude ze vstupů postupně odebírat vstupní symboly →léto, →podzim, →zima, →jaro, ...

Petriho sítě, pro které platí, že každé místo sítě má maximálně jednu značku a přechody z místa vždy jen jednu značku odebírají, resp. přidávají, dosahují modelovací schopnosti pouze konečných automatů. Už z příkladu 4.1.0.21 je ale patrné, že pomocí multipřijímacích automatových systémů je možné popisovat některé kontextové jazyky, přičemž pro místa a přechody, resp. hrany platí tytéž podmínky. Multipřijímací automatové systémy tak mohou ověřovat i takové otázky, jako například zda systém pro daný vstup uvolnil všechny alokované zdroje ap., přičemž právě příklad 4.1.0.21 může být abstrakcí podobného systému. Množina Ψ pak nemusí být jen výčtem přepínacích pravidel, ale lze na ni nahlížet jako na logický nebo matematický předpis pro stavy, popř. přechodové hrany. Aby se modelování zjednodušilo i na grafické úrovni, lze do systému zavádět nová místa (značená například čtvercem), do nichž budou vstupovat stavy. Nová místa pak budou obsahovat blokovací pravidla nad vstupními stavy a tím budou definovat množinu Ψ. Zobrazení systému z příkladu 5.1.1.7 by pak odpovídalo obrázku 5.6. Podobným způsobem lze zavést grafickou reprezentaci i pro ostatní automatové systémy. Toto téma už ale ponechávám dalšímu případnému rozšíření této práce.



C**:**

$(b_1 \wedge b_2 \wedge \neg b_3) \rightarrow (d, d, e)$
$(b_1 \wedge b_2 \wedge b_3) \rightarrow (e, d, d)$
$(\neg b_1 \wedge b_2 \wedge b_3) \rightarrow (d, e, e)$
$(\neg b_1 \wedge \neg b_2 \wedge \neg b_3) \rightarrow (e, e, d)$

Obrázek 5.6: Grafické znázornění výsledného 3-MAS z příkladu 5.1.1.7

Kapitola 6

Závěr

V diplomové práci byly zavedeny čtyři základní systémy automatů. První ze systémů byl navržen obecně tak, aby mohl být složen z jakéhokoliv typu automatu. Všechny komponenty systému zde pracují nad jedním vstupním řetězcem. Automaty jsou spuštěny sekvenčně a jejich aktivita končí například po předem stanoveném počtu kroků. Když skončí aktivita jedné komponenty, je aktivována libovolná jiná, která může pokračovat ve zpracování. Tento sekvenční systém nad zásobníkovými automaty s jedním společným zásobníkem je pak schopen pracovat s řetězci, které jsou generovatelné pomocí PC-gramatického systému v nejlevější derivaci. Obecně jsou pravděpodobně sekvenční systémy automatů stejně silné jako PC-gramatické systémy. Převod z gramatik do přechodů automatů není triviální záležitostí a provádí se ad-hoc.

Druhý z navrhovaných systémů rozšiřuje předchozí systém o možnost paralelismu. Je definován pouze nad zásobníkovými automaty. Komponenty zde pracují opět nad společným řetězcem. Nyní ale výpočet provádí všechny komponenty současně. Pokud se na vrcholu zásobníku některého z automatů objeví speciální aktivační symbol, je ze zásobníku vyjmut a další výpočet provádí pouze určená komponenta. Pokud komponenta vyřeší svou část řetězce a „ohlásí“ ji za přijatou, vrátí řízení zpět celému systému.

Poslední dva systémy jsou nejzajímavější. Jde o multipřijímací automatové systémy, které jsou opět definovány obecně nad jakýmkoliv typem formálních automatů. Každý z automatů zde má vlastní vstupní řetězec, nad kterým provádí výpočet. Běh každé aktivní komponenty je ovlivňován ostatními podle stavů, ve kterém se nachází, nebo přechodovými hranami, přes které automaty přešly do další konfigurace. Stav, resp. přechodové hrany těchto komponent mohou aktivovat, resp. deaktivovat, kteroukoliv z komponent systému na základě předurčených pravidel. Systém pak přijímá řetězce přijaté první komponentou, přijaté alespoň jednou komponentou, nebo přejaté všemi komponentami sytému. Komponentám lze vložit na vstup i různé řetězce. Jazyk, který pak automat přijímá je jazyk konkatenací řetězců přijatých dílčími automaty. Takto definovaný přístup pracuje ruku v ruce s kanonickými n-multigenerativními gramatickými systémy.

Diplomová práce současně otevírá spoustu nevyřešených problematik nejen z teorie formálních modelů. Další práce nad těmito systémy by se měla týkat formálního vymezení jazyků, které systémy dokáží zpracovávat. Navrhované systémy také uvažují všechny své komponenty shodného typu. Z praktického hlediska by mohlo být zajímavé kombinovat například zásobníkové a konečné automaty. U paralelních automatových systému je vhodné také uvažovat jeho minimální formu. Zavedené automatové systémy jsou zcela novou problematikou, čímž nabízí spousty neprozkoumaných oblastí.

Literatura

[1] *Turingův stroj* [online]. c2004, poslední revize 10.9.2007 [cit.2007-09-11]. Dostupné z: <http://cs.wikipedia.org/wiki/Turing%C5%AFv_stroj>

[2] Beneš, M., Hruška, T., Češka, M.: *Překladače*. Brno, CZ, VUT, 1994.

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

[4] Lukáš, R.: *Multigenerativní gramatické systémy*. Dizertační práce, Brno, CZ, FIT VUT, 2006.

[5] Meduna, A.: *Automata and Languages: Theory and Applications*. Springer, London, 2000. ISBN 1-85233-074-0.

[6] Meduna, A.: *Formální jazyky: Modely a jejich aplikace*. Habilitační práce, Brno, CZ, FIT VUT, 2005. ISBN 80-214-2939-9. ISSN 1213-418X.

[7] Meduna, A., Lukas, R.: *Multigenerative Grammar Systems*. , Schedae Informaticae, Volumes 15, str. 175-188, 2006.

[8] Čermák, M.: *Proceedings of the 12th conference student EEICT 2006, díl 1.:Syntax analysis based on several methods*. Brno, CZ, FEKT a FIT VUT, 2006. ISBN 80-214-3160-1.

[9] Čermák, M.: *Syntaxtická analýza založená na kombinaci několika metod*. Bakalářská práce, Brno, CZ, FIT VUT, 2006.

[10] Češka, M.: *Teoretická informatika 1*. Učební texty, 2002.

[11] Češka, M., a kol.: *Petriho sítě*. Učební texty, 2006.