Grammar Systems - Survey

TID 2007/2008

Filip Goldefus

FIT, VUT BRNO, 2007

Contents of presentation

Contents:

Introduction

Early formal systems:

- Grammars and automata were modelling classic computing devices.
- Centralized devices – computation with one central agent.

Grammar systems:

- Distributed computation – more processors and computers.
- Distribution, parallelism, concurrency and communications.
- Increase of generative power.
- GS are functioning under specific protocols.

Cooperating distributed grammar systen

CD grammar system of degree $n \geq 1$ is $(n+3)$-touple

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

where each $P_1, \ldots, P_n$ is finite set of productions.
Notation:

- i-th CF grammar $G_i = (N, T, S, P_i)$

Modes of derivation ($\Rightarrow$):

- $\Rightarrow^t$ – terminating derivation
- $\Rightarrow^{=k}$ – k-step derivation
- $\Rightarrow^{\leq k}$ – at most k-step derivation
- $\Rightarrow^{\geq k}$ – at least k-step derivation

Hybrid Modes in CDGS

- $\Rightarrow^{(\geq k_1 \wedge \leq k_2)}$ – When enabled, the component has to perform at least k_1 and at most k_2 derivation steps.
- $\Rightarrow^{(t \wedge \geq k)}$ – When enabled, the component has to perform as many derivation steps as possible, and at least k steps.
- $\Rightarrow^{(t \wedge = k)}$ – When enabled, the component has to perform as many derivation steps as possible, and exactly k steps.
- $\Rightarrow^{(t \wedge \leq k)}$ – When enabled, the component has to perform as many derivation steps as possible, and at most k steps.

Hybrid Modes – Examples

Let $G = (N, T, S, P_1, P_2)$ is CD grammar system with

- $N = \{S, A, B, A', B'\}$,
- $T = \{a, b, c\}$,
- $P_1 = \{S \to S, S \to AB, A' \to A, B' \to B\}$ and
- $P_2 = \{A \to aA'b, B \to cB', A \to ab, B \to c\}$.

And e. g.:

- $L_{(t \wedge \geq 1)}(G) = \{a^n b^n c^m | n, m \geq 1\}$
- $L_{(t \wedge = 2)}(G) = \{a^n b^n c^n | n \geq 1\}$
- $L_f(G) = \emptyset$, $f \in \{= k, \geq k | k \geq 3\}$

CDGS - Competence

Domain of i-th component:

- $dom(P_i) = \{X \in N | X \to z \in P_i\}$

Component i is k-competent on word x iff:

- $clev_i(x) = |alph_N(x) \cap dom(P_i)| = k$

Cooperation levels:

- $x \Rightarrow_i^{\leq k-comp.} y$ iff $x = x_1 \Rightarrow_i \ldots \Rightarrow_i x_m = y$ and
  - $clev_i(x_j) \leq k$ for $1 \leq j \leq m-1$ and $clev_i(x_j) = 0$ or $clev_i(x_j) > k$.
- Operations $\Rightarrow_i^{=k-comp.}$ and $\Rightarrow_i^{\geq k-comp.}$ are defined similarly.

CDGS – Competence - Example

$G = (\{A, A', B, B', C, D\}, \{a, b, c\}, AB, P_1, \ldots, P_8)$. Grammar works in ≤ 1-comp. mode $(= 1$-comp. mode).

$$
\begin{aligned}
P_1 &= \{A \rightarrow aA'b, B' \rightarrow B', C \rightarrow C\} \\
P_2 &= \{A \rightarrow A, B \rightarrow B'c, C \rightarrow C\} \\
P_3 &= \{A' \rightarrow A, B \rightarrow B, C \rightarrow C\} \\
P_4 &= \{A' \rightarrow A', B' \rightarrow B, C \rightarrow C\} \\
P_5 &= \{A' \rightarrow C, B \rightarrow B\} \\
P_6 &= \{A \rightarrow A, A' \rightarrow A', B' \rightarrow D\} \\
P_7 &= \{B' \rightarrow B', C \rightarrow \varepsilon\} \\
P_8 &= \{D \rightarrow \varepsilon\}
\end{aligned}
$$

$L(G) =$?

CDGS - generative power

Notation:

- CD$_x$(f) denotes class of CD grammars with x components.

Generative power of CD grammar systems:

$$L(CF) = L(CD_1(t)) = L(CD_2(t))$$
$$\subset$$
$$L(CD_3(t)) = L(CD_\infty(t)) = L(ET0L)$$

$$L(fRC, CF) = L(CD, CF, = 1)$$
$$L(CD, CF, = 1) = L(RC, CF)$$

Colonies

A colony is a 3-tuple $C = (V, T, F)$, where

- V is finite set of symbols.
- $T \subseteq V$ is set of terminals.
- $F = \{(S_i, F_i) | S_i \in V, F_i \subseteq (V - S_i)^*, F_i$ is finite, $1 \leq i \leq n\}$.

Generally:

- $\Rightarrow^x$ elementary string operation of type x
- $\Rightarrow^{x*}$ stays for a reflexive and transitive closure
- For $C = (V, T, F)$ and axom $w_0 \in V*$,
  $L_x(C, w_0) = \{v | w_0 \Rightarrow^{x*} v, v \in T^*\}$
- COL_x denotes class of all languages generated by colonies with $\Rightarrow^x$.

Sequential Colonies

For $x, y \in V^*$ we define basic derivation step:

- $x \Rightarrow^b y$ iff $x = x_1 S_i x_2$, $y = x_1 z x_2$ and $z \in F_i$ for some i, $1 \leq i \leq n$.

Example:

- $C = (\{A, B, a, b\}, \{a, b\}, \{(A, \{ab, aBb\}), (B, \{A\})), L_b(C, A)?$

Modification:

- $x \Rightarrow^t y$ iff $x = x_1 S_i x_2 S_i \ldots x_m S_i x_{m+1}$, $x_1 x_2 \ldots x_{m+1} \in (V - S_i)^*$, $y = x_1 w_1 x_2 w_2 \ldots w_m x_{m+1}$

Example:

- $C = (\{A, B, a\}, \{a\}, \{(A, \{BB\}), (B, \{A\}), (B, \{a\})\}), L_t(C, A)?$

Parallel Colonies

Informal definition:

- Several components are active in one derivation step of the colony
- If (S, F_i) and (S, F_j) are two components of C and if at least two symbols S appears in current string, then both components must be used.
- If only one S appears in current word and there are at least 2 components then only one *can* be used, there are two possibilities
  - derivation is blocked – *strongly parallel way of derivation* $\Rightarrow^{sp}$
  - derivation continues – *weakly competetive parallel way of derivation* $\Rightarrow^{wp}$

Parallel Colonies – Examples

Strongly parallel:

- $C = (\{A, B, C, D, 0, 1\}, \{0, 1\}, \{(A, \{0B, 1C\}), (A, \{0C, 1B\}),(B, \{A, E\}), (C, \{A, E\}), (E, \{\varepsilon\}), (E, \{\varepsilon\})\}.$
- $L_{sp}(C, AA) =?$

Weakly parallel:

- $C = (\{S, A, B, C, D, E, F, a, b, c\}, \{a, b, c\}, F)$
- $F = \{(S, \{ABC\}), (Y, \{Z\}), (Z, \{Y\}), (A, \{aD, X\}), (B, \{bE, X\}), (C, \{cF, X\}), (D, \{A\}), (E, \{B\}), (F, \{C\}), (X, \{\varepsilon\}),(X, \{\varepsilon\}), (X, \{Y\})\}$
- $L_{wp}(C, S) =?$

Colonies - Generative Power

Generative power:

$$
\begin{aligned}
COL_p &= CF \\
COL_p &\subset COL_{sp} \\
COL_p &\subset COL_{wp} \\
COL_{sp} &\subseteq MAT_{ac} \\
COL_{wp} &\subset 1lET0L
\end{aligned}
$$

Open problems:

- What is relation between COL_{wp} and COL_{sp}?
- Is the inclusion $COL_{sp} \subseteq MAT_{ac}$ proper?

P Systems

- Computability model – distributed and parallel,
- based on notion of a *membrane structure*,
- variable number of components,
  - dissolvation of membrane,
- two modes of functionality
  - generating language
  - accepting word
- Types of P Systems
  - Transition P Systems
  - P System Based on Rewriting

Membrane Structures

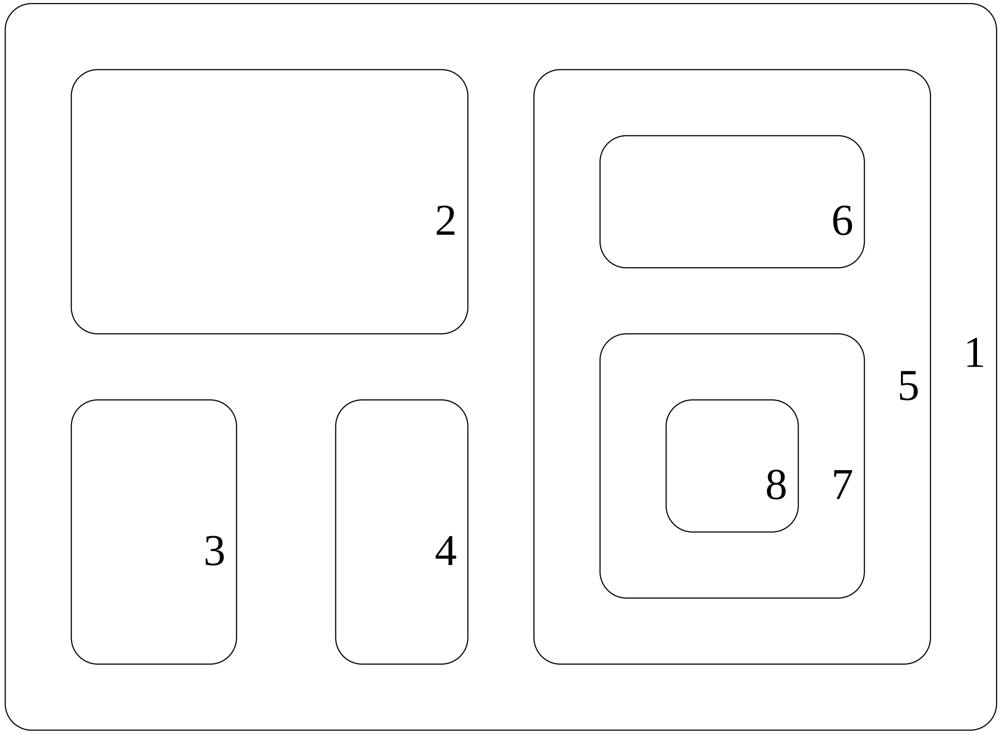
Language of Membrane Structures (MS) is recurrently defined over alphabet $\{[,]\}$:

- $[,] \in MS$,
- if $\mu_1, \ldots, \mu_n \in MS$, $n \geq 1$, then $[\mu_1, \ldots, \mu_n] \in MS$,
- nothing else in MS.

The *depth* of a membrane structure μ, denoted by $dep(\mu)$ is defined recurrently as follows:

- if $\mu = [\,]$, then $dep(\mu) = 1$,
- if $\mu = [\mu_1 \ldots \mu_n]$, for some $\mu_1, \ldots, \mu_n \in MS$ then $dep(\mu) = max\{dep(\mu_i) | 1 \leq i \leq n\} + 1$.

Membrane Structures - Venn diagram



$[_1 [_2]_2 [_3]_3 [_4]_4 [_5 [_6]_6 [_7 [_8]_8]_7]_5]_1$

Transition P Systems

A *transition P system* of degree n, $n \geq 1$, is a construct:

$$\Pi = (V, \mu, w_1, \ldots, w_n, (R_1, \rho_1), \ldots, (R_n, \rho_n), i_0)$$

where

- V is alphabet; its elements are called *objects*
- μ is membrane structure of degree n
- w_i are strings over V^*
- R_i is set of *evolution rules* over V
- ρ_i is partial order relation over R_i, specifying *priority* relation among rules of R_i.
- i_0 is a number between 1 and n – number of output membrane

Transition P Systems II.

Evolution rules:

- evolution rule is pair (u, v), usually written in form $u \to v$
  - u is string over V
  - $v = v'$ or $v = v'\delta$ and

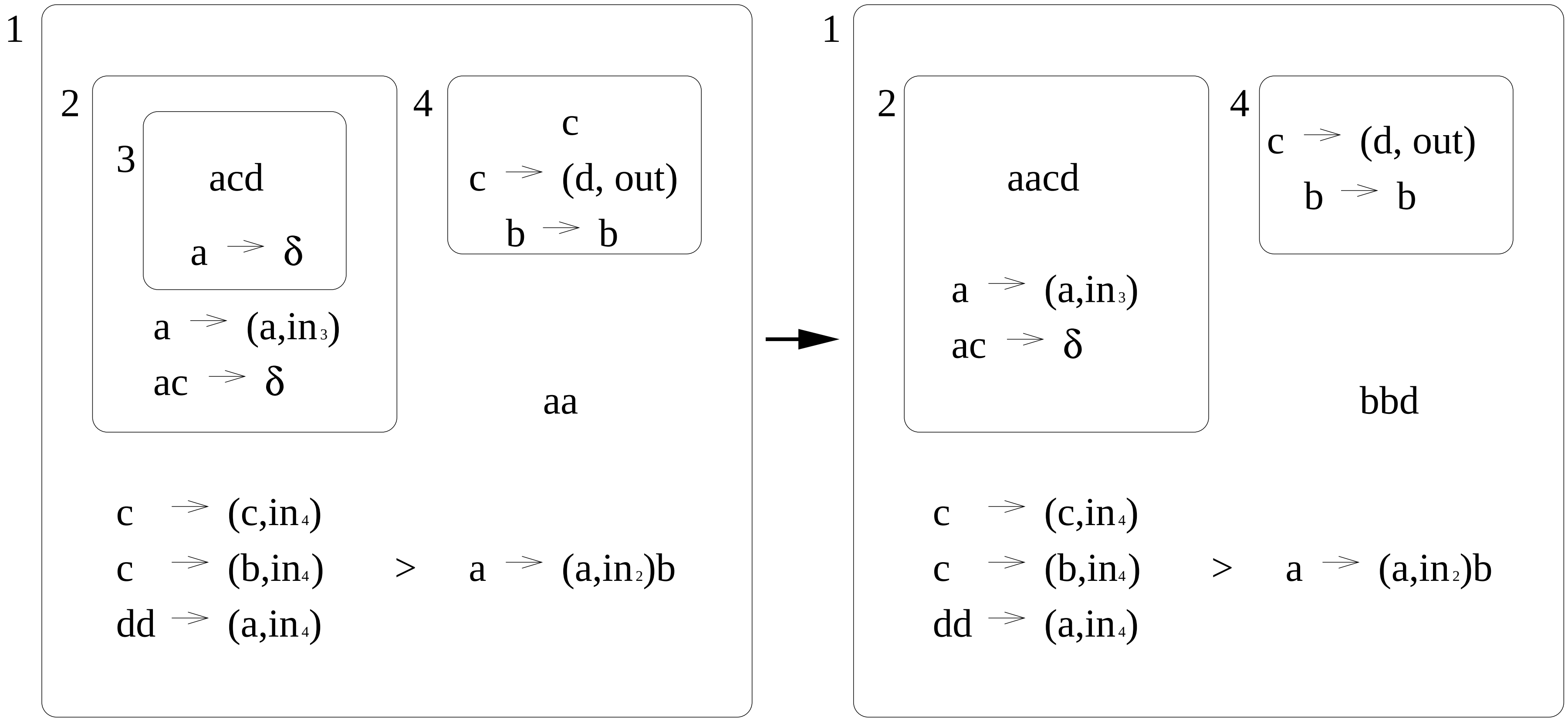
$$v' \in (V \times \{here, out\}) \cup (V \times \{in_j | 1 \leq j \leq n\})$$

  and δ is special symbol not in V. The length of u is called the *radius* of the rule $u \to v$.

Note:

- Computation is succesful if no rule can be applied.
- Some words w_i can be ε.
- Some sets of productions R_i can be empty.

Transition P System - Example



Generative Power of P Systems

Types of P Systems:

- Non-cooperative – radius of rules is 1 - $nCoo$
- Cooperative – radius of rules is at least 2 - Coo

Notation:

- $TP_n(\alpha, \delta)$ is class of languages computed by P systems with at most n components.
  - $\alpha \in \{nCoo, Coo\}$
  - δ if present denotes P System with δ rules

Generative Power of P Systems

Hierarchy of $TP_n(\alpha)$ system families

$$L(E0L) \subseteq TP_1(nCoo) \subseteq TP_2(nCoo) \subseteq \ldots \subseteq TP(nCoo).$$

For every $i = 1, 2, \ldots,$

$$TP_i(nCoo) \subseteq TP_i(nCoo, \delta).$$

And

$$TP_2(Coo) = TP_2(Coo, \delta) = TP(Coo) = TP(Coo, \delta) = RE.$$

Conclusion

Grammar Systems

- usage in many practical fields
  - parallel compilers
  - biology
  - chemistry

Increase of generative power using paralelism e. g. left-forbidding CD grammar systems.

References

- J. Dassow, Gh. Paun and G. Rozenberg. Grammar Systems. In Handbook of formal languages, Vol. 2. Springer-Verlag, Berlin, 1997.