Splicing Operation on Graphs

Meena Parvathy Sankar

Department of Mathematics
SRM University
Kattankulathur – 603203

N. Gnanamalar David

Department of Mathematics
Madras Christian College
Tambaram, Chennai – 600059

D. Gnanaraj Thomas

Department of Mathematics
Madras Christian College
Tambaram, Chennai – 600059

ABSTRACT

In this paper, we introduce the splicing operation on graph P system with the feature of conditional communication. We use the notions of Freund graph splicing and generate the string graph languages. We study the generative power of the splicing graph P system with conditional communication with the other classes of string graph languages and give comparison results of the languages generated by the system.

Keywords

Graph splicing, Conditional communication, Splicing graph P system.

1. INTRODUCTION

The splicing systems were introduced by Head to investigate the power of computing with DNA. The recent development in this field is the generalization of the splicing concept to graphs by Rudolf Freund [2, 6]. It is an accurate model of a system consisting of biochemical units and the interactions between them.

The structure and the functioning of a biological cell were inspired by a class of distributed parallel computing devices called the P system [4, 5]. In this paper, we define graph splicing P system with the feature of conditional communication.

2. PRELIMINARIES

We recall the notions of graph splicing system in this section [1, 2].

2.1 Graph Splicing System

A graph splicing system σ is represented by a pair (Σ, P) where Σ is an alphabet of the system and P is a set of graph splicing rules which is finite and is of the form $((h[1], E'[1]), \dots, (h[k], E'[k]); E)$ where $k \geq 1$ and for all i with $1 \leq i \leq k$, $h[i] = (V[i], E[i], L[i])$, $E'[i] \subseteq E[i]$. The set of cut edges is given by $E'[i]$, $V[i]$ is called the node sets and they are mutually disjoint and E must obey the following rules:

1. Each edge $(n, m) \in E'[i]$ is divided into two parts, that is, the start part $(n, m]$ and the end part $[n, m)$.
2. The elements of E are of the form $((n, m], [n', m'))$, where (n, m) and (n', m') are edges from $\cup E'[i]$.
3. Every element from $\{(n, m], [n, m)/(n, m) \in \cup E'[i]\}$ should appear exactly once in a pair of E .

2.2 Graph Splicing Step

Let the graph splicing system be represented by $\sigma = (\Sigma, P)$ and let the rule in P is given by $p = ((h[1], E'[1]), \dots, (h[k], E'[k]); E)$. Then the set R of graphs derives a set S of graphs through the splicing rule p if there exists graphs $g[1], g[2], \dots, g[k] \in R$ and graphs $g'[1], g'[2], \dots, g'[m] \in S$ such that,

- $h[i]$ is an induced sub graph of $g[i]$ for all i with $1 \leq i \leq k$.
- From $g[1], g[2], \dots, g[k]$ we delete all the edges corresponding to edges in $\cup E'[i]$ but add each edge corresponding to the edge $((n, m], [n', m')) \in E$, which yields the uniquely determined union of m connected graphs $g'[1], g'[2], \dots, g'[m]$.

In this graph splicing, we use implicitly the multisets of graphs; $g[j]$ may be another copy of $g[i]$. The idea of graph splicing rule is to cut the various edges in some graphs and rejoin the edges in another fashion.

2.3 Extended Graph Splicing System

The graph splicing system σ is represented by $\sigma = (\Sigma, P)$ and I be a finite set of graphs, called the set of axioms. The extended graph splicing step is given by quadruple (N, T, P, I) where N and T are disjoint sets of nonterminals and terminals respectively, and $N \cup T = \Sigma$. $\sigma(I)$ is the minimal set of graphs obtain by applying some splicing rules to some subset of I . $\sigma^n(I)$ is defined iteratively as $\sigma(\sigma^{n-1}(I))$; $\sigma^0(I)$ is defined to be I . We also define $\sigma^*(I) = \cup_{n \geq 0} \sigma^n(I)$. The graph language $L(F)$ generated by an extended graph splicing system $F = (N, T, P, I)$ is $L(F) = \{g = (V, E, F) \in \sigma^*(I) / F(n) \in T, \forall n \in V(g)\}$.

3. SPLICING GRAPH P SYSTEM WITH CONDITIONAL COMMUNICATION

In this section, we define a splicing graph P system and examine its generative power by adding the feature conditional communication.

3.1 Definition

A splicing graph P system with conditional communication (SGPCC) is a construct

$$\Pi = (V, T, \mu, A_1, A_2, \dots, A_n, (R_1, P_1, F_1), \dots, (R_n, P_n, F_n), (n, d))$$

where V is a finite set of nonterminal and terminal symbols; T is a set of terminal symbols available in V ; μ is membrane structure with n membranes and depth d (maximum no of membranes in the nesting of membranes in the whole system);

each $A_i \subseteq V^*$, $i \in \{1, 2, \dots, n\}$ are the finite languages

associated with the compartments $1, 2, \dots, n$ of μ ; R_i is a finite set of graph rules associated with the region i , $i = 1, 2, \dots, n$ which is of the form (V, P) where V is an alphabet and P is a finite set of graph splicing rules as given in the definition of graph splicing. The rules are used in the regions in parallelism mode as follows:

A rewriting step with parallelism involves substitution of all occurrences of one edge according to a rule that can be applied to that edge. P_i and F_i are permitting and forbidding

conditions associated with the region i , $i = 1, 2, \dots, n$ of the form empty or symbol checking given as follows.

1. Empty: no restriction is imposed on graphs; they either exit the current membrane or enter any of the directly inner membranes freely; we denote an empty permitting condition by (true, in), (true, out), and an empty forbidding condition by (false, notin), (false, notout).
2. Symbols checking: each P_i is a set of pairs (A, α) , $\alpha \in \{\text{in}, \text{out}\}$ for $A \in V$ and each F_i is a set of pairs $(B, \text{not } \alpha)$, $\alpha \in \{\text{in}, \text{out}\}$ for $B \in V$; a graph W_i can go to a lower membrane only if there is a pair $(A, \text{in}) \in P_i$ with $A \in \text{lab}(W)$ and for each $(B, \text{notin}) \in F_i$, $B \notin \text{lab}(W)$; similarly the graph goes out of the membrane i , if there is at least one pair $(A, \text{out}) \in P_i$ and $(B, \text{notout}) \in F_i$ for all $B \notin \text{lab}(W)$.

The system works in the following way. The graph is rewritten in each region according to the rule from that region. The rules and the nonterminal are nondeterministically chosen. Each graph obtained is checked against the conditions P_i , F_i from that region. If the graph fulfills the requested conditions, then it will be sent out of the membrane or it enters into the inner membrane if any exists. If both the conditions 'in' and 'out' are satisfied by choosing nondeterministically, the graph either enters into the membrane or it is sent out of the membrane. If the graph fulfills the 'in' conditions and if there is no inner membrane, then the graph remains in the same region. A graph which is rewritten and which is sent to another membrane is 'consumed', then no copy of it will be available in the same membrane for the next rewriting step. If a graph which cannot be rewritten is directly checked against the communication conditions, and depending upon the result of checking, it remains or leaves the membrane. The rewriting rule has a priority over communication. The language generated by the splicing graph P system is denoted by $L(II)$. The family of languages generated by a splicing graph P system with conditional communication is denoted by $SGPCC_n^d(\alpha, \beta)$. Here n denotes the total number of membranes in the system, and d is the depth of the membranes. $\alpha, \beta \in \{\text{empty, symbol}\}$ denote the permitting and forbidding conditions for communication. If there is no upper bound on the number of membranes then we replace n by $*$.

3.2 Theorem

$$\text{SGPSCC}_3^3(\text{empty}, \text{empty}) - \text{CF}_{\text{string}} \neq \emptyset$$

Proof:

The non-context free string graph language

$L_1 = \{(a^{2n}b^{2n})^\bullet / n \geq 0\}$ generated by a splicing graph P system with three membranes with the conditional communication empty in both permitting and forbidding cases.

$$\Pi_1 = (V, T, \mu, A_1, A_2, A_3, (R_1, P_1, F_1), (R_2, P_2, F_2), (R_3, P_3, F_3), (3, 2))$$

where $V = \{X, A, B, a, b\}, T = \{a, b\}, \mu = [_1[_2[_3]_3]_2]_1$

$$P_1 = \{(\text{true}, \text{in}), (\text{true}, \text{out})\}$$

$$F_1 = \{(\text{false}, \text{notin}), \text{false}, \text{notout}\}$$

$$P_2 = \{(true, in), (true, out)\}$$

$$F_2 = \{(false, notin), (false, notout)\}$$

$$P_3 = \{(true, out)\}$$

$$F_3 = \{(false, notout)\}$$

In this example, the derived string graphs are of the form $(awb)^*$, but only the string graph $(w)^*$ is taken as the output. Initially, the string graph with the non terminal X is in the skin membrane. If the rewriting rule of the first membrane is used, the node labeled with X is replaced with two nodes A and B. In the second membrane, the terminating rule of the node A is used and the string graph is sent to the first membrane and there the terminating rule of the node B is used and therefore the first member of the language is derived. In the same way in the second membrane if the rewriting rule where the node A is doubled is applied and in the third membrane the rewriting rule where the node B is doubled is used and finally the terminating rules are applied to generate the second member of the language. Proceeding this way, the output will be the string graphs of the form $(w)^*$ where $w = a^{2^n}b^{2^n}$. Hence the language collected is $L_1 = \{(a^{2^n}b^{2^n})^* / n \geq 0\}$.

3.3 Theorem

$SGPSCC^1_1(\text{empty, symbol}) \subset SGPSCC^3_3(\text{empty, symbol})$

Proof:

The non-context free string graph language

$L_2 = \{(a^{2^n}b^{2^n}c^{2^n})^* / n \geq 0\}$ is generated by a splicing graph

P system with the conditional communication empty and symbol with three membranes.

$\Pi_2 = (V, T, \mu, A_1, A_2, A_3, (R_1, P_1, F_1), (R_2, P_2, F_2), (R_3, P_3, F_3), (3,2))$ where $V = \{X, A, B, a, b\}, T = \{a, b\}, \mu = [{}_1[{}_2[{}_3]_3]_2]_1$

$P_1 = \{(true, in), (true, out)\}$

$F_1 = \{(A, notin), (B, notin), (C, notin)\}$

$P_2 = \{(true, in), (true, out)\}$

$F_2 = \{(A', notin), (B', notin), (C', notin)\}$

$P_3 = \{(true, out)\}$

$F_3 = \{(A, notout), (B, notout), (C, notout)\}$

The language L_2 is in $SGPSCC^3$ (empty, symbol) but it cannot be in $SGPSCC^1$ (empty, symbol), for one membrane is not enough to generate this string-graph language L_2 as by parallelism mode, 'a' is generated 2^n times, but not b and c.

3.4 Theorem

$LIN_{string} \subset SGPSCC$ (empty, symbol).

Proof:

Consider a splicing graph P system with the conditional communication empty, symbol with one membrane.

$\Pi_2 = (V, T, \mu, A_1, (R_1, P_1, F_1), (1, 1))$

where $V = \{X, A, B, a, b, c, d\}, T = \{a, b, c, d\}, \mu = [1]_1$.

$P_1 = \{(true, out)\}$

$F_1 = \{(A, notout), (B, notout)\}$

The application of rules is in parallelism mode, since it contains at most one nonterminal string in any sentential form. All the values in a linear string graph grammar can be kept in a single membrane with empty permitting condition and symbol in forbidding condition for every nonterminal in a linear string graph grammar. Hence the language

$L_3 = \{w \bullet / w = (a^n cb^n a^m db^m), m, n \geq 0\}$ is the output of Π_3 .

4. CONCLUSION

Conditional communication with splicing is an interesting variant in the study of rewriting graph P system with graph as an object. Here, Freund graph splicing is used. Our further work is towards this graph splicing.

4. REFERENCES

- [1] P. Bottoni, A. Labella, C. Martin-Vide and Gh. Paun, "Rewriting P systems with conditional communication", Lecture Notes in Computer Science, 2300 (2002), pp. 352-353.
- [2] R Freund, "Splicing systems on graphs", in : proceedings of Intelligence in neural and Biological Systems, IEEE Press, New York, May 1995, pp 189-194.
- [3] C. Martin-vide and Gh. Paun, "Computing with membranes (P systems): Universality results", Lecture notes in computer science, 2055 (2001) pp. 82-101.
- [4] G. Paun, "A guide to membrane computing", Theoretical computer science, vol 287 (2002), 73-100.
- [5] G.Paun, "Computing with membranes", Journal of computer and system sciences, 61(2000), pp 108-143.
- [6] Rahul Santhanam, Kamala Krithivasan, "Graph splicing systems", Discrete applied mathematics, 154(2006) 1264-1278.