Model Based Simulation of Michaelis-Menten Kinetic Reactions By DPO Graph Transformation

J. Philomenal Karoline^1, P. Helen Chandra#2,
S. M. Saroja Theerdus Kalavathy#3, A. Mary Imelda Jayaseeli*4
^Research Scholar in Mathematics, #Associate Professor in Mathematics,
*Associate Professor in Chemistry,
Jayaraj Annapackiam College for Women (Autonomous),
Periyakulam, Theni District, Tamilnadu, India.
1Philoharsh@gmail.com, 2chandrajac@yahoo.com, 3kalaoliver@gamil.com,
4imeldaxavier@gamil.com

Abstract --- **Graph transformation system can model structural transformations of molecules. The Double Pushout (DPO) formulation appears to be the best suited to use graph transformation as a model for chemical reactions, In this paper, a methodology has been developed to model Michaelis-Menten Kinetic reactions networks in terms of DPO Graph Transformation.**

***Keywords* --- Fuzzy Artificial Cell System, Double Pushout, Sequential Dependent Derivation, Partial Composition, Extended Match Matrix.**

I. Introduction

The frame work of graph transformations provide a valuable set of tools to generate and investigate large chemical networks. Such networks require an underlying Artificial Chemistry [1] that describes how molecules and reactions are modeled. If molecules are treated as edge and vertex labeled graphs, where the vertex labels correspond to atom types and the edge labels denote bond types then structural change of molecules during chemical reactions can be modeled as graph rewriting rules [2]. There are many approaches to graph transformations. Here we discuss the algebraic approach which is based on pushout constructions. There are two variants of pushout named as Single Pushout (SPO) and Double Pushout (DPO). Recently, the composition of concrete chemical reactions, i.e., transformations of complete molecules, as a means of reconstructing metabolic pathways is considered [3]. We reframe our conceptual developments on double pushout formulation of graph transformations.

An important issue for the application to chemical reactions is that the graphs involved in the rules are in general not connected. Typical chemical reactions combine molecules, split molecules or transfer groups of atoms from one molecule to another. The transformation rules for all these reactions therefore require multiple connected components [3].

Recently we have proposed a computing device that is based on Abstract Rewriting Systems on Multisets closely related to P system with fuzzy multiset rewriting rules and fuzzy data. As an extension of this model, we have developed a new system called FACSP (FARMS in Artificial Cell System with Proteins on Membranes) and its behavior has been studied in [4].

In this chapter, a mathematical model of the system based on simulation of Michaelis-Menten Kinetic Reactions by DPO graph transformation is established.

II. PRELIMINARIES

We first outline the framework on which the graph transformation is considered.

A. Double Pushout

Given two graphs G and H, G directly derives H through p, denoted by $G \Rightarrow_{p,m} H$ is used when we wish to make $m : L \longrightarrow G$ explicit [6]. Two gluing conditions are used to model a DPO Graph transformation. The DPO formulation of graph transformation considers transformation rules of form $p = (L \xleftarrow{l} K \xrightarrow{r} R)$ where L, R and K are called the left graph, right graph and context graph, (the common interface of L and R) respectively. The maps l and r are graph morphisms. The rule p transforms G to H, that is $G \Rightarrow_{p,m} H$ if there is a pushout graph D and a matching morphism $m : L \longrightarrow G$ such that the diagram in Fig. 1 is valid. The existence of D is equivalent to the so-called gluing condition, which determines whether the rule p is applicable to a match in G [3].

Two gluing constructions are used to model a graph transformation. In the DPO approach, roughly speaking, a production is given by $p = (L, K, R)$, where L and R are the left and right-hand side graphs and K is the common interface of L and R, i.e. their intersection. The left-hand side L represents the preconditions of the rule, while the right-hand side R describes the post conditions. K describes a graph part which has to exist to apply the rule, but which is not changed. $L \setminus K$ describes the part which is to be deleted, and $R \setminus K$ describes the part to be created.

A direct graph transformation with a production p is defined by first finding a match m of the left-hand side L in the current host graph G such that m is structure preserving.

$$\begin{tikzcd}
L \arrow[d, "m"'] \arrow[dr, phantom, "(1)"] & K \arrow[l, "l"'] \arrow[r, "r"] \arrow[d] \arrow[dr, phantom, "(2)"] & R \arrow[d] \\
G & D \arrow[l] \arrow[r] & H
\end{tikzcd}$$

Fig. 1 Double Pushout.

When a direct graph transformation with a production p and a match m is performed, all the vertices and edges which are matched by $L \setminus K$ are removed from G. The removed part is not a graph, in general, but the remaining structure $D := (G \setminus m(L)) \cup m(K)$ still has to be a legal graph, i.e., no edges should be left dangling. This means that the match m has to satisfy a suitable gluing condition, which makes sure that the gluing of $L \setminus K$ and D is equal to G (Fig. 1). In the second step of a direct graph transformation, the graph D is glued together with $R \setminus K$ to obtain the derived graph H (Fig. 1). Since L and R can overlap in K, the sub match occurs in the original graph G and is not deleted in the first step, i.e. it also occurs in the intermediate graph D. For gluing newly created vertices and edges into D, the graph K is used. This defines the gluing items at which R is inserted into D. A graph transformation, or, more precisely, a graph transformation sequence, consists of zero or more direct graph transformations.

More formally, a direct graph transformation with p and m is defined as follows. Given a production $p = (L \longleftarrow K \longrightarrow R)$ and a context graph D, which includes also the interface K, the source graph G of a graph transformation $G \Longrightarrow H$ via p is given by the gluing of L and D via K, written $G = L +_K D$, and the target graph H is given by the gluing of R and D via K, written $H = R +_K D$. More precisely, we shall use graph morphisms $K \longrightarrow L, K \longrightarrow R$, and $K \longrightarrow D$ to express how K is included in L, R, and D, respectively. This allows us to define the gluing constructions $G = L +_K D$ and $H = R +_K D$ as the pushout constructions (1) and (2) in Fig. 1, leading to a double pushout. The resulting graph morphism $R \longrightarrow H$ is called the comatch of the graph transformation $G \Longrightarrow H$.

In order to apply a production p with a match m of L in G, given by a graph morphism $m : L \longrightarrow G$ as shown in Fig. 1, we first have to construct a context graph D such that the gluing $L +_K D$ of L and D via K is equal to G. In the second step, we construct the gluing $R +_K D$ of R and D via K, leading to the graph H and hence to a DPO graph transformation $G \Longrightarrow H$ via p and m. For the construction of the first step, however, a gluing condition has to be satisfied, which allows us to construct D with $L +_K D = G$. In the case of an injective match m, the gluing condition states that all dangling points of L, i.e. the nodes x in L such that $m(x)$ is the source or target of an edge e in $G \setminus L$, must be gluing points x in K.

A simple example of a DPO graph transformation step is given in Fig. 2, corresponding to the general scheme in Fig. 1. Note that in the diagram (PO1), G is the gluing of the graphs L and D along K, where the numbering of the nodes indicates how the nodes are mapped by graph morphisms. The mapping of the edges can be uniquely deduced from the node mapping. Note that the gluing condition is satisfied in Fig. 2, because the dangling points (1) and (2) of L are also gluing points. Moreover, H is the gluing of R and D along K in (PO2), leading to a graph transformation $G \Longrightarrow H$ via p. In fact, the diagrams (PO1) and (PO2) are pushouts in the category Graphs of graphs and graph morphisms [5].

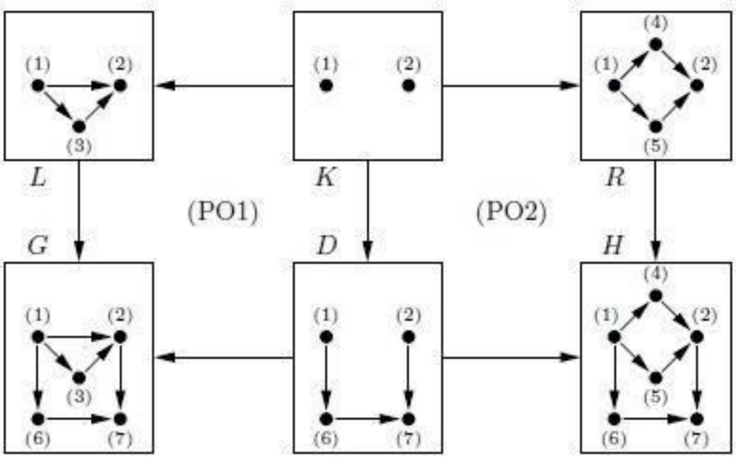


Fig. 2 Example of DPO Graph Transformation.

Given two graphs G, H and a set of rules $\mathfrak{R}$, we write $G \Longrightarrow_{\mathfrak{R}} H$ if there is a rule p in $\mathfrak{R}$ such that $G \Longrightarrow_p H$. A R derivation from G to H over $\mathfrak{R}$ is a sequence of the form $G = G_0 \Longrightarrow_{\mathfrak{R}} G_1 \Longrightarrow_{\mathfrak{R}} \ldots \Longrightarrow_{\mathfrak{R}} G_n \cong H$, which may be denoted by $G \Longrightarrow^*_{\mathfrak{R}} H$ [6].

B. Sequential Independent Derivation

$$\begin{tikzcd}
L_1 \arrow[d, "m_1"'] & K_1 \arrow[l, "l_1"'] \arrow[r, "r_1"] \arrow[d, "k_1"] & R_1 \arrow[dr, "n_1"'] \arrow[drrr, dashed, "i"] & (1) \quad\quad (2) & L_2 \arrow[dl, "m_2"] \arrow[dlll, dashed, "j"'] & K_2 \arrow[l, "l_2"'] \arrow[r, "r_2"] \arrow[d, "k_2"] & R_2 \arrow[d, "n_2"] \\
G & D_1 \arrow[l, "f_1"] \arrow[rr, "g_1"] & & H & & D_2 \arrow[ll, "f_2"'] \arrow[r, "g_2"] & G'
\end{tikzcd}$$

Fig. 3 Sequential Independent Derivation.

Two direct transformations $G \Longrightarrow_{p_1 m_1} H \Longrightarrow_{p_2 m_2} G'$ are sequentially independent if there exist morphisms $i : R_1 \longrightarrow D_2$ and $j : L_2 \longrightarrow D_1$ such that $f_{\,2} \circ i = n_1$ and $g_1 \circ j = m_2$ [5]. Two consecutive direct derivations $G \Longrightarrow_{p_1 m_1} H \Longrightarrow_{p_2 m_2} G'$ are sequentially independent if they may be swapped. i.e., if p_2 can be applied to G and p_1 to the resulting graph. Therefore p_2 at match m_2 cannot delete anything that has been explicitly preserved by the application of p_1 at match m_1 and moreover, it cannot use (neither consuming nor preserving it) any element generated by p_1; this implies that the overlapping of R_1 and L_2 in H must be included in the intersection of the interface graphs K_1 and K_2 [7].

C. Sequential Dependent Derivation

Two direct derivations that are not sequentially independent are called sequentially dependent [8]. No j can be found with $g_1 \circ j = m_2$ because the rule $p_1 = (L_1 \overset{l_1}{\leftarrow} K_1 \overset{r_1}{\rightarrow} R_1)$ creates the nodes and edges required to apply the rule $p_2 = (L_2 \overset{l_2}{\leftarrow} K_2 \overset{r_2}{\rightarrow} R_2)$ with the given match m_2 [5]. If the order in which p_1 and p_2 are applied affects the overall outcome of the application of both rules there is sequential dependence [9].

$$\begin{tikzcd}
L_1 \arrow[d, "m_1"] & K_1 \arrow[l, "l_1"'] \arrow[r, "r_1"] \arrow[d, "k_1"'] \arrow[drr, phantom, "(1)"] & R_1 \arrow[dr, "n_1"'] & & L_2 \arrow[dl, "m_2"] \arrow[dlll, dashed, "\exists j"] \arrow[dr, phantom, "(2)"] & K_2 \arrow[l, "l_2"'] \arrow[r, "r_2"] \arrow[d, "k_2"] & R_2 \arrow[d, "n_2"] \\
G & D_1 \arrow[l, "f_1"] \arrow[rr, "g_1"'] & & H & & D_2 \arrow[ll, "f_2"] \arrow[r, "g_2"'] & G'
\end{tikzcd}$$

Fig. 4 Sequential Dependent Derivation.

D. E - Composition

Concurrency theory provides a canonical framework for the composition of two graph transformations [3]. Given two production rules $p_1 = (L_1 \overset{l_1}{\leftarrow} K_1 \overset{r_1}{\rightarrow} R_1)$ and $p_2 = (L_2 \overset{l_2}{\leftarrow} K_2 \overset{r_2}{\rightarrow} R_2)$ an E - dependency relation (E, e_1 , e_2) is given by a graph E and injective morphisms $e_1 : R_1 \longrightarrow E$ and $e_2 : L_2 \longrightarrow E$, which are jointly surjective. The E - concurrent production $p = p_1 *_E p_2$, ie., a production rule $p = (L \overset{l}{\leftarrow} K \overset{r}{\rightarrow} R)$ is defined whenever a dependency graph E exists. It is computed based on the diagram (Fig. 5), where double squares (1)(2) and (3)(4) form double pushouts and (5) is a pullback. p is also called the composite rule or the E-based composition of p_1 and p_2 [4].

$$\begin{tikzcd}
L_1 \arrow[d, "m_1"'] \arrow[dr, phantom, "(1)"] & K_1 \arrow[l, "l_1"'] \arrow[r, "r_1"] \arrow[d, "k_1"] \arrow[drr, phantom, "(2)"] & R_1 \arrow[dr, "e_1"] & & L_2 \arrow[dl, "e_2"'] \arrow[dr, phantom, "(3)"] & K_2 \arrow[l, "l_2"'] \arrow[r, "r_2"] \arrow[d, "k_2"] \arrow[dr, phantom, "(4)"] & R_2 \arrow[d, "n_2"] \\
L & K_1' \arrow[l, "f_1"'] \arrow[rr, "g_1"'] & & E \arrow[d, phantom, "(5)"] & & K_2' \arrow[ll, "f_2"] \arrow[r, "g_2"'] & R \\
& & & K \arrow[ulll, "l"] \arrow[ull, "k_1'"'] \arrow[urr, "k_2'"] \arrow[urrr, "r"'] & & &
\end{tikzcd}$$

Fig. 5 E-Composition.

E. Partial Composition [3]

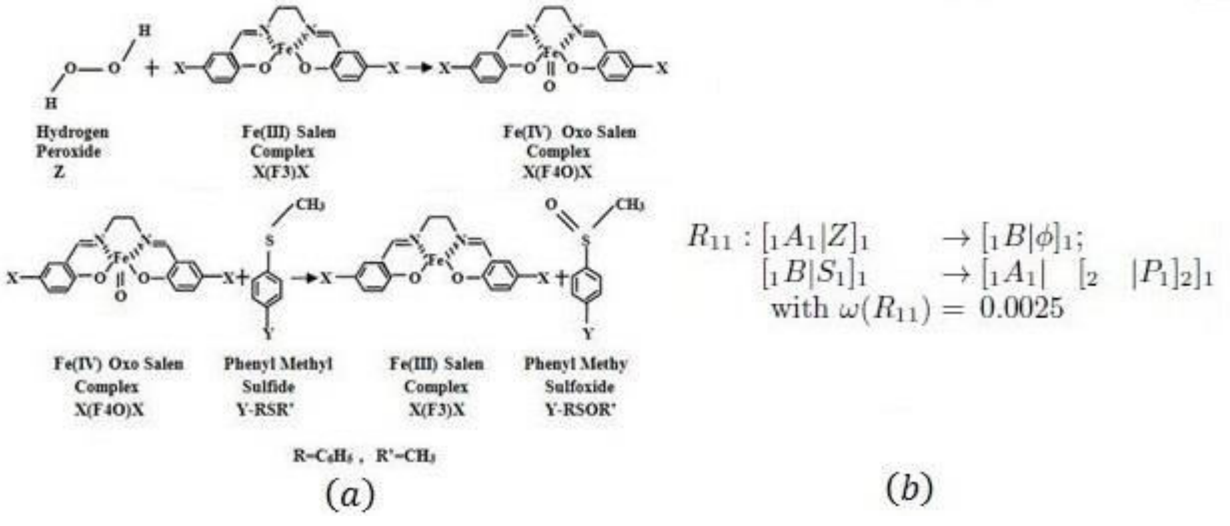
Let Q be a graph with $\#Q$ connected components $Q_i, i = 1,\ldots,\#Q$. It will be convenient to treat Q as the multiset of its components. A typical chemical graph derivation corresponding to a bi-molecular reaction can be written in the form $\{G^1, G^2\} \Rightarrow_{p,m} \{H^1, H^2, H^3\}$, where we take the notation to imply that all graphs G^i and H^j are connected.

$$\begin{tikzcd}
& & & \overline{L_2} \arrow[dl, "\overline{m}"'] \arrow[dr] & L_2' \arrow[d] & & \\
L_1 \arrow[d, "m_1"'] & K_1 \arrow[l, "l_1"'] \arrow[r, "r_1"] \arrow[d, "k_1"'] \arrow[dr, phantom, "(1)"] & R_1 \arrow[dr, "e_1"'] & & L_2 \arrow[dl, "e_2"] \arrow[dr, phantom, "(2)"] & K_2 \arrow[l, "l_2"'] \arrow[r, "r_2"] \arrow[d, "k_2"'] & R_2 \arrow[d, "n_2"] \\
L & K_1' \arrow[l, "f_1"] \arrow[rr, "g_1"] & & E \arrow[d, phantom, "(3)"] & & K_2' \arrow[ll, "f_2"'] \arrow[r, "g_2"] & R \\
& & & K \arrow[ull, "k_1'"] \arrow[urr, "k_2'"'] & & &
\end{tikzcd}$$

Fig. 6 Partial Composition.

Given two production rules $p_1 = (L_1 \overset{l_1}{\leftarrow} K_1 \overset{r_1}{\rightarrow} R_1)^2$ and $p_2 = (L_2 \overset{l_2}{\leftarrow} K_2 \overset{r_2}{\rightarrow} R_2)$, we consider a partition of the components of L_2 into two parts $\overline{L_2}$ and $L_2{}'$ (Fig. 6) and we require that E is isomorphic to a disjoint union of a copy of R_1 and $L_2{}'$, while $\overline{L_2}$ must be isomorphic to a subgraph of R_1. As a consequence, every connected component $L_2{}^i$ of L_2 satisfies either $e_2(L_2{}^i) \subseteq e_1(R_1)$ or $e_2(L_2{}^i)$ is a connected component of E isomorphic to $L_2{}^i$. For a rule composition of this type to be well defined we need that $\exists\, i$ such that $e_2(L_2{}^i) \subseteq e_1(R_1)$ holds, i.e., $\overline{L_2}$ must be non-empty [3].

F. Oxidation of Sulfides



$R_{11} : [{}_1A_1|Z]_1 \rightarrow [{}_1B|\phi]_1;$
$[{}_1B|S_1]_1 \rightarrow [{}_1A_1|\;[{}_2\;|P_1]_2]_1$
with $\omega(R_{11}) = 0.0025$

Fig. 7 (***a***) Oxidation of Sulfides (***b***) Evalution Rule

In [10] oxidation of arylmethyl sulfides using iron-salen complexes as catalyst in presence of hydrogen peroxide as oxidant is followed kinetically and is described. The authors carried out catalytic reactions of aryl methyl sulfides varying the substitution at Y as $H, Cl, Br, CH_3, OCH_3, F$ and NO_2 groups.

The Mechanism for Sulfides Oxidation is analysed by the theory of computation. The general reaction rule is presented in (a) and the structure is shown in Fig. 7 (a). In (a) the formation of intermediate Oxo compound of the catalyst is first described and in second step, the oxidation of substrate following regeneration of catalyst.

$(a).\ Z + X(F\ 3)X \rightarrow X(F\ 4O)X;$

$X(F\ 4O)X + Y - RSR' \rightarrow X(F\ 3)X + Y - RSOR'$

In case $X = H$ and Y varying as seven substitutions, (a) consists of seven reaction rules. The first reaction rule (R_{11}) is shown in Fig. 7 (b). The behaviour of FACSP-A is developed [11] in which R_p consists of seven evolution rules. In R_p, A_i, Z, B, S_i and P_i represents the substrate, oxidant, complex, sulfide and product respectively.

III. MMK Reactions by DPO Graph Transformation for FACSP-A

In this section, we consider the reaction rule (R_{11}) in FACSP-A to be described by the graph transformation rules in the double pushout (DPO) formalism that encode specific reaction mechanisms and describe in detail how transformation rules are composed in a chemically relevant manner. Composition of chemical reaction rule is exemplified by graph grammar rule composition operators.

A. DPO for FACSP-A

Consider the evolution rule R_{11} in FACSP-A which consists of two production rules: $p_1 = [_1\, A_1 \,|\, Z\,]_1 \longrightarrow [_1\, B \,|\, \phi\,]_1$ and $p_2 = [_1\, B \,|\, S_1\,]_1 \longrightarrow [_1\, A_1 \,| \quad [_2 \quad |\, P_1\,]_2\,]_1$. We now describe the reaction rule R_{11} in DPO formulation of graph transformation.

Let the initial graph G contains connected components A_1, Z and S_1. The DPO formulation of graph transformation considers the first rule in the evolution rule R_{11} as $p_1 = \left(L_1 \overset{l_1}{\leftarrow} K_1 \overset{r_1}{\rightarrow} R_1\right)$ $= \left(A_1\, Z \overset{l_1}{\leftarrow} N\ (Fe) \overset{r_1}{\rightarrow} B\right)$ where, $A_1\, Z,\ N\ (Fe)$ and B denote the left graph, context graph and right graph respectively. The maps l_1 and r_1 are graph morphisms. The rule p_1 transforms G to H, that is $G \Longrightarrow_{p_1 m_1} H$ as there is a pushout graph D_1 which contains S_1 and $N\ (Fe)$ and a matching morphism $m_1 : L_1 \longrightarrow G$. The existence of D_1 is equivalent to the gluing condition which determines whether the rule p_1 is applicable to a match in G.

The application of the rule $p_1 = \left(A_1\, Z \overset{l_1}{\leftarrow} N\ (Fe) \overset{r_1}{\rightarrow} B\right)$ to a graph $G = A_1 Z\, S_1$, that is, the construction of the double pushout (1) (2) in Fig. 8 amounts to the following.

The gluing graph K_1 is created by choosing a node $N\ (Fe)$ from L_1 and R_1. The context graph $D_1 = N\ (Fe)\, S_1$ is formed using the relation $D_1 = (G \setminus m_1(L_1\ )) \cup m_1(K_1)$ and the gluing condition is checked. The transformed graph H is built by adding the elements of $R_1 \setminus K_1$ (that is, $B \setminus N(Fe)$) to the built context graph D_1.

Since $p_1 = \left(L_1 \overset{l_1}{\leftarrow} K_1 \overset{r_1}{\rightarrow} R_1\right) = \left(A_1\, Z \overset{l_1}{\leftarrow} N\ (Fe) \overset{r_1}{\rightarrow} B\right)$, $K_1 = N\ (Fe)$ is in L_1, R_1 and D_1, we express that there are morphisms $K_1 \longrightarrow L_1$, $K_1 \longrightarrow R_1, K_1 \longrightarrow D_1$ and pushout constructions $G = L_1 +_{K_1} D_1$, $H = R_1 +_{K_1} D_1$. H is the gluing of R_1 and D_1 along K_1 in (2), leading to a graph transformation $G \Longrightarrow H$ via p_1. In fact, the diagrams (1) and (2) are pushouts in the category Graphs of graphs and graph morphisms.

Now the graph H contains connected components B and S_1. The DPO formulation of graph transformation considers the second rule in the evolution rule R_{11} as $p_2 = \left(L_2 \overset{l_2}{\leftarrow} K_2 \overset{r_2}{\rightarrow} R_2\right)$ $=$ $\left(BS_1 \overset{l_2}{\leftarrow} N(Fe,S) \overset{r_2}{\rightarrow} A_1 P_1\right)$ where $B\, S_1$, $N\ (Fe,\ S)$ and $A_1\, P_1$ denote the left graph, context graph and

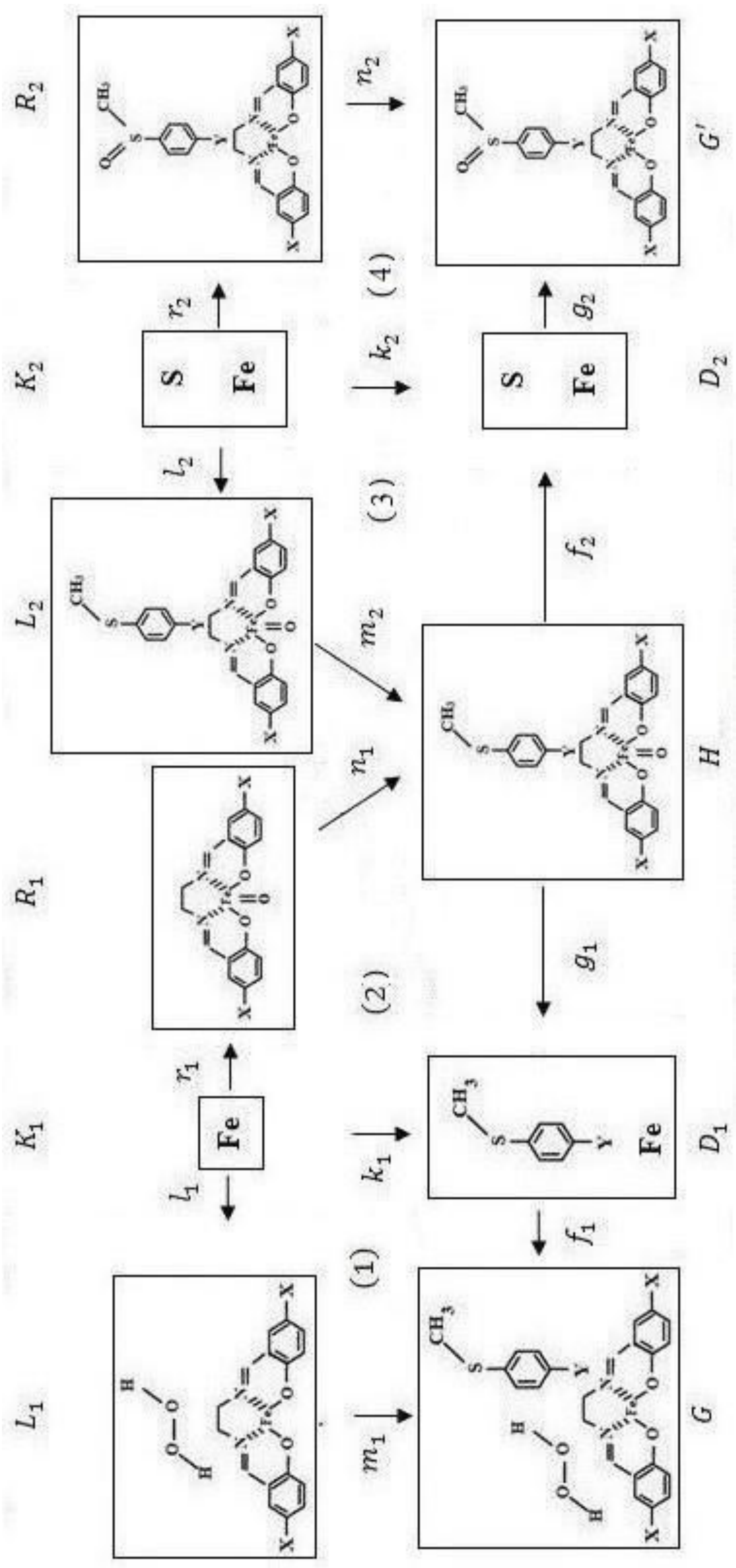


Fig. 8 DPO for FACSP-A.

right graph respectively. The maps l_2 and r_2 are graph morphisms. The rule p_2 transforms H to G', that is $H \Longrightarrow_{p_2,m_2} G'$ as there is a pushout graph D_2 which contains $N\ (Fe,\ S)$ and a matching morphism $m_2 : L_2 \longrightarrow H$. The existence of D_2 is equivalent to the gluing condition which determines whether the rule p_2 is applicable to a match in H.

The application of the rule $p_2 = \left(BS_1 \overset{l_2}{\leftarrow} N(Fe,S) \overset{r_2}{\rightarrow} A_1P_1\right)$ to a graph $H = B\,S_1$, that is, the construction of the double pushout (3) (4) is shown in Fig. 8 amounts to the following.

The gluing graph K_2 is created by choosing the nodes $N\ (Fe)$ and $N\ (S)$ from L_2 and R_2. The context graph $D_2 = N\ (Fe,S)$ is formed using the relation $D_1 = (G \setminus m_2(L_2)) \cup m_2(K_2)$ and the gluing condition is checked. The transformed graph G' is built by adding the elements of $R_2 \setminus K_2$ (that is, $A_1P_1 \setminus N\ (Fe)\ N\ (S)$) to the built context graph D_2.

Since $p_2 = \left(L_2 \overset{l_2}{\leftarrow} K_2 \overset{r_2}{\rightarrow} R_2\right) = \left(B\,S_1 \overset{l_2}{\leftarrow} N\,(Fe,\ S) \overset{r_2}{\rightarrow} A_1\,P_1\right)$, $K_2 = N(Fe,S)$ is in L_2, R_2 and D_2, we express that there are morphisms $K_2 \longrightarrow L_2$, $K_2 \longrightarrow R_2$, $K_2 \longrightarrow D_2$ and pushout constructions $H = L_2 +_{K_2} D_2$, $G' = R_2 +_{K_2} D_2$. G' is the gluing of R_2 and D_2 along K_2 in (4), leading to a graph transformation $H \Longrightarrow G'$ via p_2. In fact, the diagrams (3) and (4) are pushouts in the category Graphs of graphs and graph morphisms.

Since there is a sequence of derivations $G \Longrightarrow_{p_1} H \Longrightarrow_{p_2} G'$, a derivation from G to G' over R_{11} exists. We write $G \Longrightarrow^{*}_{R_{11}} G'$.

B. Sequential Dependent Derivation for FACSP-A

Consider the rule R_{11} in FACSP-A which consists of two production rules $p_1 = \left(A_1\,Z \overset{l_1}{\leftarrow} N(Fe) \overset{r_1}{\rightarrow} B\right)$ and $p_2 = \left(B\,S_1 \overset{l_2}{\leftarrow} N\,(Fe,S) \overset{r_2}{\rightarrow} A_1\,P_1\right)$. Here A_1, Z, B, S_1 and P_1 respectively represents Iron III Salen Complex, Hydrogen Peroxide, Iron IV Salen Complex, Phenyl Methyl Sulfide and Phenyl Methyl Sulfoxide. As the chemical reaction takes place through formation of intermediate, the dependency relation is verified. There is a sequence of derivation $G \Longrightarrow_{p_1} H \Longrightarrow_{p_2} G'$. They are not sequentially independent as the order in which p_1 and p_2 are applied affects the overall outcome of the application of both rules. We observe that there is a dependency relation between R_1 and L_2. That is, R_1 and L_2 have the common element B. No j can be found with $g_1 \circ j = m_2$, because p_1 creates B required to apply the rule p_2 with the given match m_2. This means that the rule p_2 is applicable only when p_1 is applied. The sequential dependent derivation of FACSP is shown in Fig. 9.

C. E-Composition for FACSP-A

The graph transformation $G \Longrightarrow_{p_1} H \Longrightarrow_{p_2} G'$ is sequentially dependent, the E-dependency relation $(E, n_1, m_2) = (E, e_1, e_2)$ is defined by a Graph E and injective morphisms $e_1 : B \longrightarrow B\,S_1$ and $e_2 : B\,S_1 \longrightarrow B\,S_1$, which are jointly surjective. The E-concurrent production

$$p = \left(A_1Z\,S_1 \overset{l}{\leftarrow} N\,(Fe,\ S) \overset{r}{\rightarrow} A_1\,P_1\right)$$

$$= \left(A_1\,Z \overset{l_1}{\leftarrow} N\,(Fe) \overset{r_1}{\rightarrow} B\right) *_E \left(B\,S_1 \overset{l_2}{\leftarrow} N\,(Fe,\ S) \overset{r_2}{\rightarrow} A_1\,P_1\right)$$

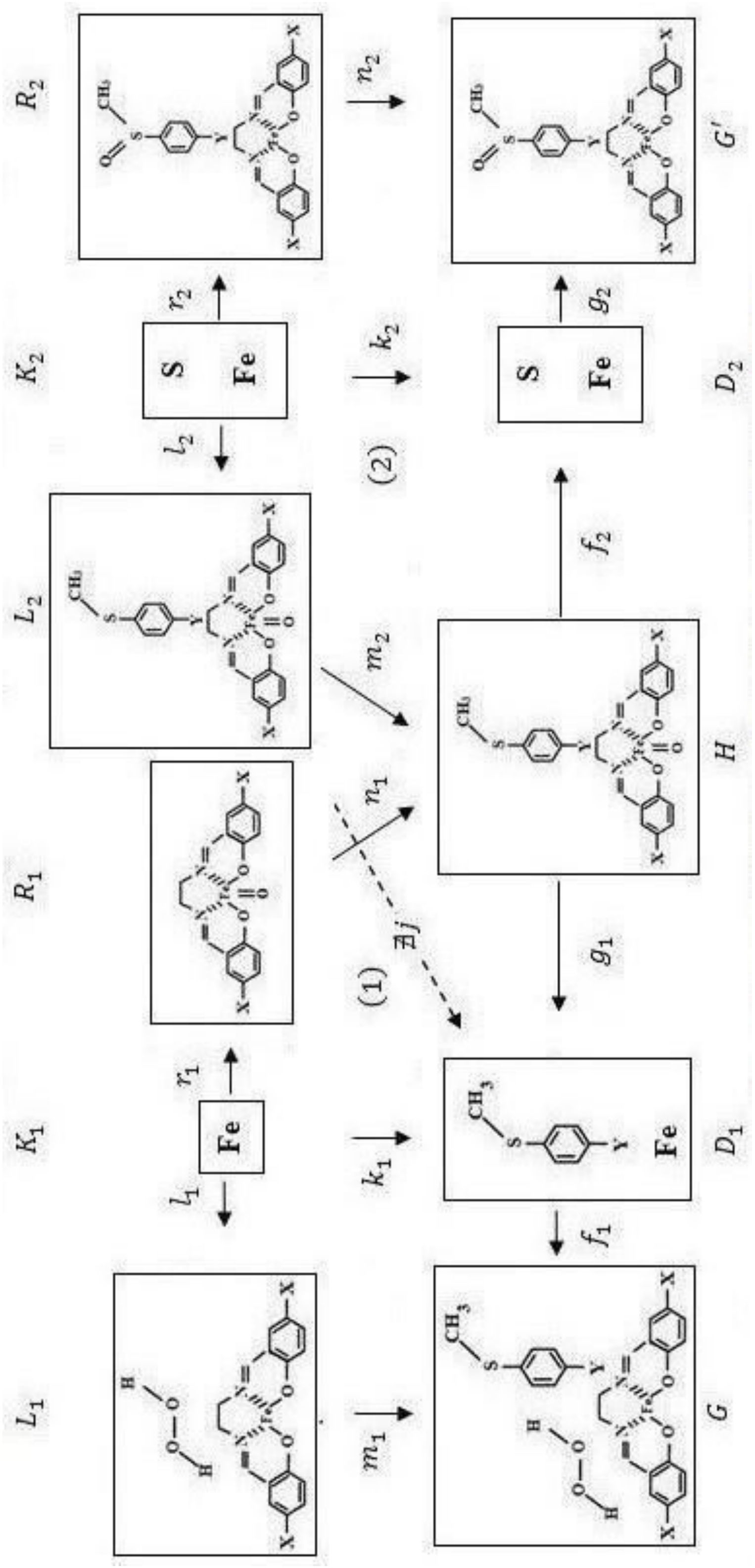


Fig. 9 Sequential Dependent Derivation for FACSP-A.

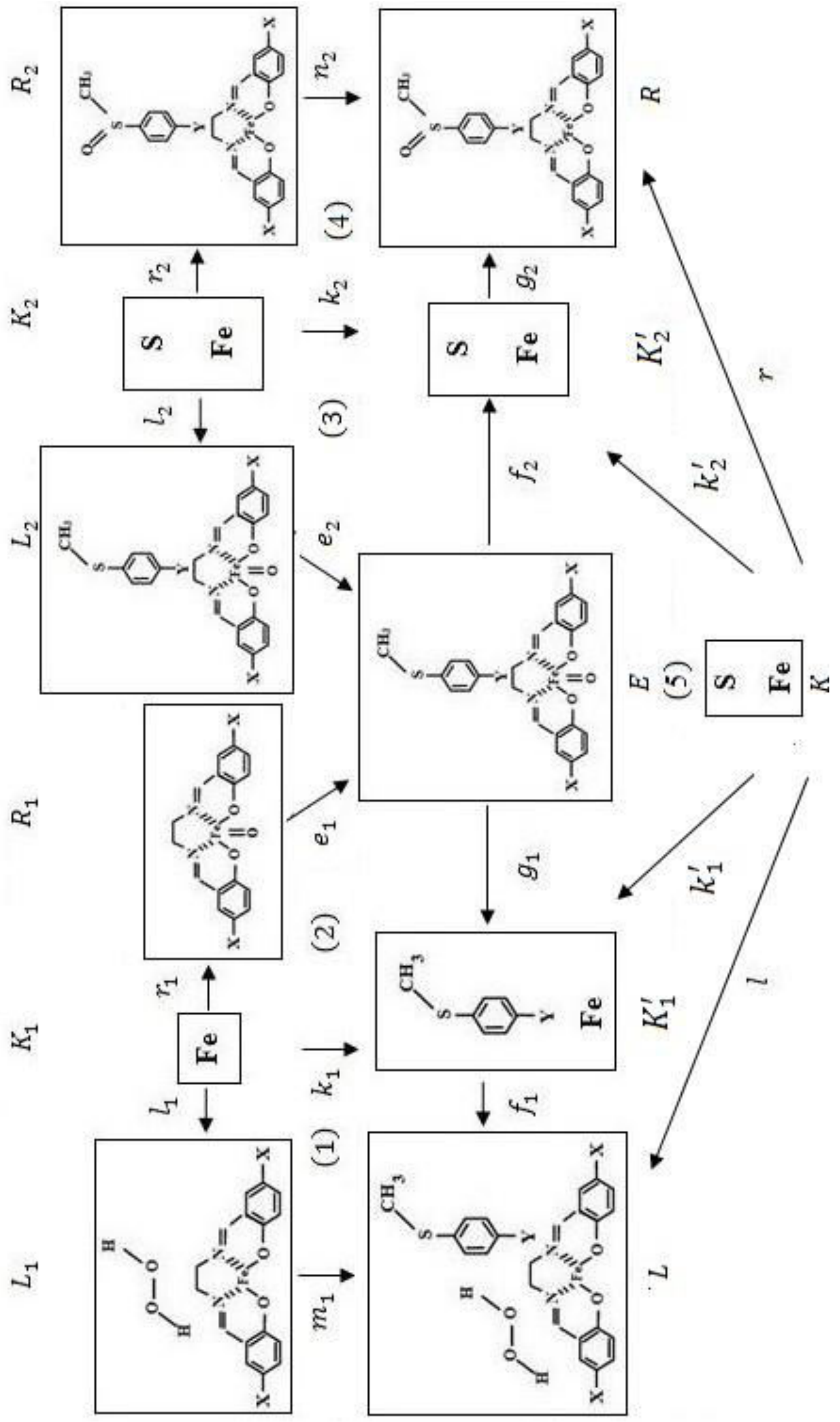


Fig. 10 E-Composition for FACSP-A.

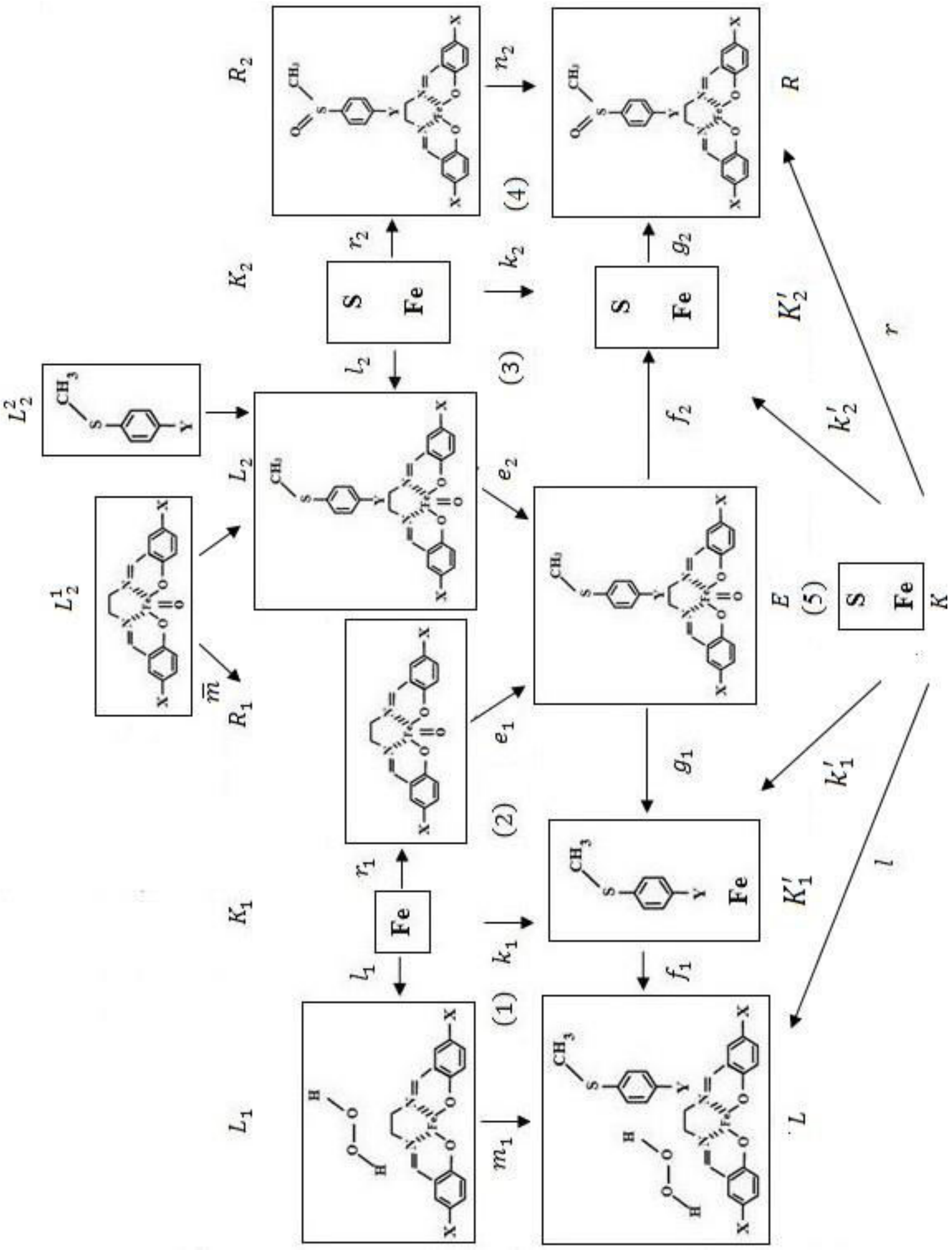


Fig. 11 Partial Composition for FACSP-A.

ie., a production rule $p = \left(A_1Z\, S_1 \overset{l}{\leftarrow} N\,(Fe,\ S) \overset{r}{\rightarrow} A_1\, P_1 \right)$ is defined whenever a dependency graph E exists. It is computed based on the diagram (Fig. 10), where double squares (1)(2) and (3)(4) form double pushouts and (5) is a pullback. $p = \left(A_1Z\, S_1 \overset{l}{\leftarrow} N\,(Fe,\ S) \overset{r}{\rightarrow} A_1\, P_1 \right)$ is also called the composite rule or the E-based composition of $p_1 = \left(A_1\, Z \overset{l_1}{\leftarrow} N\,(Fe) \overset{r_1}{\rightarrow} B\right)$ and $p_2 = \left(B\, S_1 \overset{l_2}{\leftarrow} N\,(Fe,\ S) \overset{r_2}{\rightarrow}\ A_1\, P_1 \right)$.

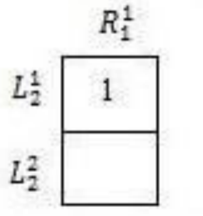
D. Partial Composition for FACSP-A

Consider the two reaction rules $p_1 = \left(A_1\ Z \overset{l_1}{\leftarrow} N\,(\,Fe\,) \overset{r_1}{\rightarrow}\ B \right)$ and $p_2 = \left(B\ S_1 \overset{l_2}{\leftarrow} N\,(Fe,\ S) \overset{r_2}{\rightarrow}\ A_1\, P_1 \right)$ and the DPO graph transformation, $G \Rightarrow_{p_1} H \Rightarrow_{p_2} G'$ (Fig. 11) where R_1 represents B, the Iron IV Salen Complex. L_2 has two components B and S_1, the iron IV Salen Complex and Phenyl Methyl Sulfide. Let B and S_1 in L_2 be denoted by ${L_2}^1$ and ${L_2}^2$ respectively. Now the graph E contains the components B and S_1 which is isomorphic to the disjoint union of a copy of R_1 and ${L_2}^2$ while, ${L_2}^1$ is isomorphic to a subgraph of R_1. As a consequence, there exist ${L_2}^1$ which has the component B, such that $e_2({L_2}^1)\ \subseteq\ e_1(R_1)$ holds. ie., ${L_2}^1$ is non empty. Hence partial composition exists. Since, ${L_2}^2$ contains the component S_1, the Phenyl Methyl Sulfide, it is non empty and so the partial composition is not a full composition. The partial composition of $p_1 = (L_1,\ K_1,\ R_1)$ and $p_2\ =\ (\{{L_2}^1,\ {L_2}^2\},\ K_2,\ R_2\,)$ yields $p_2\ \circ\ p_1\ =\ (\{L_1, {L_2}^2\}, K, R)$, which is shown in Fig. 11.

The partial composition of the two rules $p_1 = \left(A_1\, Z \overset{l_1}{\leftarrow} N(Fe) \overset{r_1}{\rightarrow} B\right)$ and $p_2 = \left(BS_1 \overset{l_2}{\leftarrow} N(Fe, S) \overset{r_2}{\rightarrow} A_1P_1\right)$ is mediated by the dependency graph E and the two matching morphisms e_1 and e_2. Since these are subgraph isomorphisms, E consists B and S_1 which is simply the union $e_1(R_1)\ \cup\ e_2(L_2)$. The partial match $e_1(R_1)\ \cap\ e_2(L_2)$ is B which can be understood as a matching μ between R_1 and L_2.

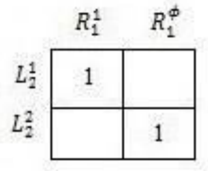
*E. Enumerating the matchings **μ** for FACSP-A*

The key to find all compositions is the enumeration of all matchings μ that respect our restrictions on overlaps between connected components. As partial composition of FACSP exists, we consider the sets $\{R_1^1\}$ and $\{L_2^1, L_2^2\}$ of connected components of R_1 and L_2 respectively. In the first set we find all subgraph matches $L_2^i \subseteq R_1^j$ (represented the corresponding matchings μ_{ij}) and arrange the result in a matrix of lists of subgraph matches. Since $L_2^1 \subseteq R_1^1$, we arrange the result in the first row and first column of R_1^1. As L_2^2 is not matched in any component of R_1, the matching matrix is extended by a virtual column. The match matrix and the extended match matrix is shown in Fig. 12 and Fig. 13.



	R_1^1
L_2^1	1
L_2^2	

Fig. 12: Match Matrix for FACSP-A.



	R_1^1	R_1^ϕ
L_2^1	1	
L_2^2		1

Fig. 13: Extended Match Matrix for FACSP-A.

IV. Conclusions

As graph transformation system can model structural transformations of molecules, a DPO graph transformation methodology has been developed to model Michaelis-Menten Kinetic reactions networks. We observe that there exists a sequential dependent derivation and partial composition of FACSP. This result will be utilizable to predict the progress of reactions and is stepped into analyse the quantitative dynamics of the reactions.

Acknowledgment

The author Dr. Sr. P. Helen Chandra, Principal Investigator of UGC Major Research Project (F.No. -43-412/2014 (SR) dated 05-09-2015) is grateful to UGC, New Delhi, for the award of the project which enabled to execute this research work in Jayaraj Annapackiam College for Women (Autonomous), Periyakulam, Theni District, Tamilnadu.

References

[1] P. Dittrich, J. Ziegler and W. Banzhaf, “Artificial chemistries - a review”, *Artificial Life*, 7(3), 225-275, 2001.

[2] G. Benko, C. Flamm and P.F. Stadler, “A graph-based toy model of chemistry”, *J. Chem. Inf. Comput. Sci.*, 43(4), 1085-1093, 2003.

[3] J. L. Andersen, C. Flamm, D. Merkle and P. F. Staler, “Inferring Chemical Reaction Patterns Using Rule Composition in Graph Grammars”, *Journal of Systems Chemistry*, 4(4), 1-14, 2013.

[4] M. Astalos, E. syriyani, M. Wimmer and M. Kessentini, “Towards transformation rule composition”, *Proceedingsofthefourth international workshoponMulti- paradigm Modeling*, 10, 1-13, 2010.

[5] H. Ehrig, K. Ehrig, U. Prange and G. Taenthzer, *Fundamentals of Algebraic Graph Transformation, EATCS Monographs of theoretical computerscience,* Springer, Berlin, 2006.

[6] A. Habel, J. Muller and D. Plump, “Double Pushout Graph Transformation Re- visited”, *Mathematical Structures in Computer Science*, 11, 637-688, 2001.

[7] A. Corradani, U. Montanari, F. Rossi, H. Ehrig, R. Heckel and M. Lowe, Hand-book of Graph Grammars and Computing by Graph Transformation: “Algebraic Approaches to Graph Transformation, Part I: Basic Concepts and Double Pushout Approach”, *World Scientific*, 1, 163-245, 1996.

[8] H. Ehrig, A. Habel and L. Lambers, “Parallelism and Concurrency Theorems for Rules with Nested Application Conditions”, F. Drewes, A. Habel, B. Hoffmann, D. Plump, Eds.: *Manipulation of Graphs, Algebras and Pictures*, 109-133, 2009.

[9] M. Bapordra, “Chemical Reaction Rate Analysis Using Graph Transformations”, BSc Computer Science with Management Project, Final Report, 2009.

[10] A.M.I. Jayaseeli and S. Rajagopal: “[Iron(III)-salen] Ion Catalyzed H_2O_2 Oxidation of Organic Sulfides and Sulfoxides”, *J. Mol. Catal, A: Chem*, 309, 103(10), 2009.

[11] P. Helen Chandra, S.M. Saroja Theerdus Kalavathy, A.M.I. Jayaseeli and J. Philomenal Karoline, “ Fuzzy ACS with Biological Catalysts on Membranes in Chemical Reactions”, *Journal of Network and Innovative Computing, MIR Labs, USA, 4, 143-151, 2016.*