Journal of Computer and Communications, 2019, 7, 49-60
http://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

Using Category Theory to Explore and Model Label Event Structures

Ming Zhu, Jing Li

College of Computer Science and Technology, Shandong University of Technology, Zibo, China
Email: zhu_ming@sdut.edu.cn, li_jing@sdut.edu.cn

How to cite this paper: Zhu, M. and Li, J. (2019) Using Category Theory to Explore and Model Label Event Structures. *Journal of Computer and Communications*, **7**, 49-60.
https://doi.org/10.4236/jcc.2019.71005

Received: December 12, 2018
Accepted: January 25, 2019
Published: January 28, 2019

Abstract

The development of a concurrent system poses unique challenges, especially those related to correctness and consistency, as such a system usually involves several interactive processes executing simultaneously. To deal with some of these challenges, we resorted to Labeled Event Structures (LES) and category theory as the formal methods to model concurrent systems. Specifically, in this paper, we proposed an idea to define categories and corresponding constructs, such as product and sum, to model events and relationships among events represented by LES. To explain the idea, several examples are developed. Though a mathematical proof, the proposed idea helped to build a correct-by-construction approach for formalizing LES models of concurrent systems.

Keywords

Concurrent System, LES, Formalization, Category Theory

1. Introduction

A concurrent system that consists of several simultaneously executing components allows carrying out multiple tasks at the same time, which can accelerate the computational work of software substantially. To model concurrent system, Labeled Event Structures (LES) were proposed and evolved in research [1] [2]. However, a concurrent system usually involves many concurrently interactive components; the exhibition of a large number of different behaviors typically occurs, which may introduce difficulties to the development of concurrent systems [3]. In particular, the notable difficulties include the state-space explosion, unpredictable composition, and others [4]. To tackle this kind of challenge, the formal method is considered to be a way, which can provide systems with known safety properties [5]. Usually, a formal specification can be used to check for particular types of errors and as inputs for model checking. Category theory is a formal method, which has been used to model and verify concurrent systems. Category theory has been proposed as a framework to offer specification structure. It has a rich body of theory to reason objects and their relations. Moreover, category theory adopts a correct-by-construction approach by which components can be specified, proved and composed in the way of preserving their properties.

Researches [6] [7] used category theory to model concurrent systems. As a continuation, we propose to use category theory to explore and model LES for concurrent systems in this paper. Specifically, we propose an approach to construct categorical object, morphism, product, and sum for LES. To explain our work, a vending machine example is designed by LES, and modeled by category theory. The rest of this paper is organized as follows. Section 2 introduces related work to this paper. Section 3 introduces background knowledge required to understand the remaining content of the paper. Section 4 provides a vending machine example modeled by LES, which is used for the explanation of the proposed categorical modeling. Section 5 shows how to construct categorical structures from LES. Section 6 illustrates how to use the proposed categorical approach to model the vending machine example. Section 7 concludes this paper.

2. Related Work

2.1. Labeled Event Structure

LES is a mathematical model with true concurrency that describes a process in terms of relations between sets of events it generates. Loogen and Goltz used LES to analyze and model nondeterministic concurrent processes [8]. de León, Haar and Longuet proposed a theoretical framework based on LES for testing and verifying observable behaviors of concurrent systems from true concurrency models like Petri nets or networks of automata [9]. Castellan, Clairambault, Rideau and Winskel introduced a detailed, self-contained update to concurrent games on event structures which were preserved by composition with a copycat strategy, and the construction of a bicategory of these strategies [10]. Bruni, Melgratti and Montanari proposed a definition of a particular class of graph grammars that are expressive enough to model name passing calculi while simplifying the denotational domain construction, and applied this technique to derive event structure semantic [11].

2.2. Category Theory

For modeling concurrency, category theory is used to model, analyze, and compare Transition System, Trace Language, Event Structure, Petri nets, and other classical models of concurrency [12] [13] [14]. Sisiaridis, Kuchta and Markowitch proposed a framework for implementing Godement calculus and cartesian closed comma categories for information security management in the detection of threats and attacks in communication systems [15]. Paper [16] introduces a formal language model which formalized agent-environment interaction in a multi-agent framework called Conversational Grammar Systems (CGS). This system provided a model with a high degree of flexibility. Based on eco-grammar systems, the formal model used in this paper can be defined as an evolutionary multi-agent system. Category theory is applied to study relationships between geometrical models for concurrency and classical models [17]. Abdel Gawad outlined and summarized four new potential applications of category theory to OOP research are presented the use of operads to model Java sub-typing [18].

3. Background

In this section, background and work related to our research are introduced.

3.1. Labeled Event Structure

An event structure expresses how these events are related to each other. In general event structures that are widely used, a concurrent system or a process can be represented as tuples $(E;Con;\vdash)$.

Definition 1. An event structure is a tuple $(E;Con;\vdash)$ consisting of

- a set of events *E*,
- the consistency predicate $Con \subseteq 2^E$, the set of conflict-free finite subsets of *E*, and
- the enabling relation $\vdash \subseteq Con \times E$

which satisfies the following properties:

- consistency of *Con*: $\forall X,Y \subseteq E \wedge X \subseteq Y \Rightarrow Y \in Con$ and
- $\forall e \in E.\ \forall X,Y \subseteq E.\ X \vdash e \wedge X \subseteq Y \in Con \Rightarrow Y \vdash e$, that is, if X enables e so does any conflict-free superset Y of X.

Example 1. There is a transition system with states $S=\{S_0,S_1,S_2,S_3,S_4\}$, where S_0 is the initial state, and transitions $T=\{a,b,c,d\}$. The graphical representation of the transition system is show in **Figure 1**.

This transition system can be represented by LES as follows:

$$E=\{a;b;c;d\}$$

$$Con=\{\varnothing,\{a\},\{b\},\{c\},\{d\},\{a,b\},\{a,d\},\{a,c\},\{b,c\},\{a,b,c\}\}$$

$$\vdash:\varnothing\vdash a,\{a\}\vdash b,\{a\}\vdash d,\{a,b\}\vdash c$$

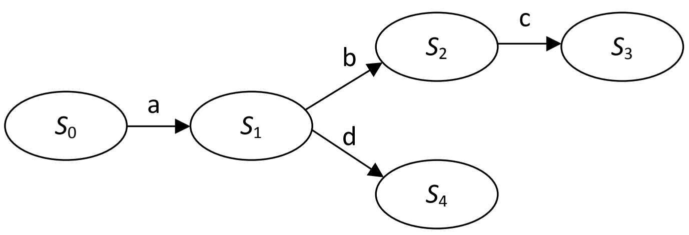


Figure 1. A transition system.

3.2. Category Theory

Category theory focuses on the relationships (morphisms) between objects instead of their representations; the morphisms can determine the nature of interactions established between the objects.

Definition 2. A category consists of the following data:

- *Objects*: A, B, C, etc.
- *Arrows* (*Morphisms*): f, g, h, etc.
- For each arrow f, there are given objects: dom(f), cod(f) called *domain* as well as *codomain* of f, and f: $A \to B$ indicates that A = dom(f), B = cod(f).
- Given arrows f: $A \to B$ and g: $B \to C$ with cod(f) = dom(g), there is an given arrow:

  $g \circ f : A \to C$ called *composite* of f and g.
- For each object A, there is an given arrow: 1_A: $A \to A$ called *identity* arrow of A.

These data need to satisfy the following laws:

- Associativity: $h \circ (g \circ f) = (h \circ g) \circ f$ for all f: $A \to B$, g: $B \to C$, h: $C \to D$.
- Unit: $f \circ 1_A = f = 1_B \circ f$ for all f: $A \to B$.

Example 2. Let (S; ≤) be a partially-ordered set (poset). Define the category C in which: each member x of S is an object of C; and each relation $x \le y$ of (S; ≤) is a morphism $x \to y$ of C.

We can verify that C is a category as follows:

- For every object x, there is an identity morphism $x \to x$, corresponding to reflexivity, $x \le x$, in the poset.
- The morphisms ($x \to y$) and ($y \to z$) form a composition pair:

  $(y \to z) \circ (x \to y) = (x \to z)$; corresponding to transitivity, $x \le y$, $y \le z$, and $x \le z$, in the poset.
- Composition is associative:

$$((x \to y) \circ (v \to x)) \circ (u \to v) = (v \to y) \circ (u \to v) = u \to y$$

and

$$(x \to y) \circ ((v \to x) \circ (u \to v)) = (x \to y) \circ (u \to x) = (u \to y),$$

because of

$$\begin{aligned} &((x \le y) \circ (v \le x)) \circ (u \le v) \\ &= (x \le y) \circ ((v \le x) \circ (u \le v)) \\ &= (x \le y) \circ (u \le x) \\ &= (v \le y) \circ (u \le v) = (u \le y) \end{aligned}$$

Definition 3. Let A and B be objects in a category *Cat*. Then a *product* of A and B consists of:

- an object, $A \times B$,
- morphisms, often called projections, $A \times B \xrightarrow{\pi_a} A$ and $A \times B \xrightarrow{\pi_b} B$, and

- the property that, for any object C and morphisms $C \xrightarrow{f_a} A$, $C \xrightarrow{f_b} B$, there is a unique morphism $C \xrightarrow{g} A \times B$ such that **Figure 2** commutes. That is $\pi_a \circ g = f_a$ and $\pi_b \circ g = f_b$.

Definition 4. Let A and B be objects in a category *Cat*. Then a *sum* (or *co-product*) of A and B consists of:

- an object, $A + B$,
- morphisms, often called inclusions or canonical projections, $A \xrightarrow{i_a} A + B$ *and* $B \xrightarrow{i_b} A + B$, and
- the property that, for any object C and morphisms $C \xrightarrow{f_a} A$ and $C \xrightarrow{f_b} B$, there is a unique morphism $A + B \xrightarrow{\langle f_a; f_b \rangle} C$ such that **Figure 3** commutes. That is $\langle f_a; f_b \rangle \circ i_a = f_a$ and $\langle f_a; f_b \rangle \circ i_b = f_b$.

4. An Overview of a Vending Machine Example

In this paper, we use a vending machine example to illustrate how to construct categories and the corresponding structures for models specified by LES which is defined and specified in section 3.

Example 3. There is a *vending machine* and a *person*. The *vending machine* can offer *coke* and *pepsi*. Each time, the *vending machine* accepts a *coin* first, then dispenses a bottle of *coke* or *pepsi* according to the *person*'s choice.

In this example, the *vending machine* and the *person* are modeled as two processes. When the *person* inserts a coin and *vending machine* accepts the coin, this interaction can be represented by the shared event *coin* in *person* and *vending machine*. When *person* chooses *coke* or *pepsi* and *vending machine* accepts it, this interaction can be represented by the shared event *coke* or *pepsi* in *person* and *vending machine*. After that, the communications between *person* and *vending machine* terminate.

By using LES, the communications between *person* and *vending machine* in the example can be modeled as follows:

$$\begin{tikzcd}
 & C \arrow[ld, "f_a"'] \arrow[d, "g"] \arrow[rd, "f_b"] & \\
A & A \times B \arrow[l, "\pi_a"] \arrow[r, "\pi_b"'] & B
\end{tikzcd}$$

Figure 2. Product.

$$\begin{tikzcd}
 & C & \\
A \arrow[ru, "f_a"] \arrow[r, "i_a"'] & A + B \arrow[u, "\langle f_a; f_b \rangle"] & B \arrow[lu, "f_b"'] \arrow[l, "i_b"]
\end{tikzcd}$$

Figure 3. Sum (coproduct).

$$VMP = (E; Con; \vdash)$$

$$E = \{coin; coke; pepsi; stop\}$$

$$Con = \{\varnothing, \{coin\}, \{coke\}, \{pepsi\}, \{coin, coke\}, \{coin, pepsi\},$$
$$\{coin, pepsi, stop\}, \{coin, coke, stop\}\}$$

$$\vdash : \varnothing \vdash coin, \{coin\} \vdash coke, \{coin\} \vdash pepsi, \{coin, coke\} \vdash stop, \{coin, pepsi\} \vdash stop$$

5. Construct Categorical Structures for LES Models

In this section, we use category theory to construct structures for LES models. We first define categorical object and morphism for constructing categories for LES models, then we construct product and sum based on the categories, and we use the vending machine example in Section 3 to explain the categorical structures.

5.1. Object

An object is like an event structure, but simpler. Formally, an object (*E*, *R*) consists of a set of events, *E*, and an ordering relation $R \subseteq E \times E$. If events e_1 and e_2 are related by *R* (that is, $(e_1, e_2) \in R$), $\{e_1\} \vdash e_2$ that indicates "e_1 precedes e_2" or "e_1 happens before e_2".

The relation *R* is a partial order: reflexive, antisymmetric, and transitive. If $e_1 \neq e_2$, it is not possible to have both $e_1 \vdash e_2$ and $e_2 \vdash e_1$, but it is possible that neither is true.

An object can be represented by a forest diagram such as **Figure 4**. For example:

For this object,

$$E = \{a, b, c, d, e\},$$

$$R = \{\varnothing \vdash a, \varnothing \vdash c, \{a\} \vdash b, \{c\} \vdash d, \{c\} \vdash e\}$$

R is reflexive, transitive. In **Figure 3**, we omit reflexive and transitive arrows. Note that some pairs of events, e.g., *a* and *c*, are unrelated: this means simply that there are no constraints on their order of occurrence.

5.2. Morphism

Let $O_1 = (E_1, R_1)$ and $O_2 = (E_2, R_2)$ be objects. There is a morphism $O_1 \xrightarrow{m} O_2$ if and only if $E_1 \subseteq E_2$ and $O_1 \subseteq O_2$. In words:

- Every event *e* that belongs to E_1 also belongs to E_2.
- Every ordered pair $\{e_1\} \vdash e_2$ in O_1 also belongs to O_2.

There is an identity morphism for every object (*E*, *R*) because $E \subseteq E$ and $R \subseteq R$. Similarly, compositions of morphism exist by transitivity of $\subseteq$. Consequently, the objects and morphisms combined forma category **CP**.

If there is a morphism $O_1 \xrightarrow{m} O_2$, which indicates O_1 is contained by O_2 or simply O_2 contains O_1. This terminology is consistent with a set "containing" its subsets.

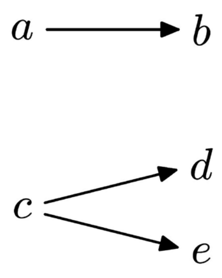


Figure 4. Graphical representation of object.

Also, if there is a morphism $O_1 \xrightarrow{m} O_2$, then it must be unique, because there is only one way that a set can be a subset of another set.

There exist some special cases:

- Traces are objects: the trace $\{a\} \vdash b$, $\{a,b\} \vdash c$ can be represented as object $a \rightarrow b \rightarrow c$.
- If trace t_1 is a prefix of trace t_2, there is a morphism $t_1 \xrightarrow{m} t_2$.
- Event trees are objects: the object shown above in **Figure 3** has a tree with c as root.
- There is a morphism from any path in a tree to the tree itself.

5.3. Sum (Coproduct)

If S and T are objects, we define their *sum*(*coproduct*) $S + T$ to be the smallest object that contains both S and T. The sum can be defined by set union as follows:

Definition 5. Let $S = (E_1, R_1)$ and $T = (E_2, R_2)$, then

$$S + T = (E_1, R_1) + (E_2, R_2) = (E_1 \cup E_2, R_1 \cup R_2).$$

- By definition, $S + T$ contains S. Consequently, there is a morphism $S \xrightarrow{i_S} S + T$. Similarly, there is a morphism $T \xrightarrow{i_t} S + T$.
- Let X be another object and suppose there is a morphism $S \xrightarrow{s} X$, then X contains S. Similarly, if there is a morphism $T \xrightarrow{t} X$, then X contains T. Thus X contains both S and T. Since $S + T$ is the smallest object containing both S and T, it follows that X must contain $S + T$ and therefore there is a morphism $S + T \xrightarrow{h} X$. The morphism h is unique.

To illustrate the sum $S + T$, **Figure 5** describes the sum and it commutes.

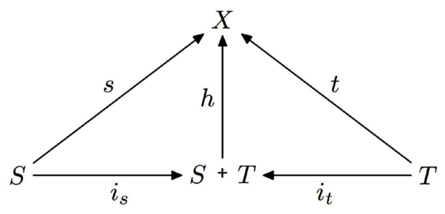
We can use sum to build *branching* structures, which can be explained by using Example 4 as follows:

Example 4. Given (E_1, R_1) and (E_2, R_2), where

- $E_1 = \{a,b\}$, $R_1 = \{\varnothing \vdash a, \{a\} \vdash b\}$.
- $E_2 = \{a,c\}$, $R_2 = \{\varnothing \vdash a, \{a\} \vdash c\}$.

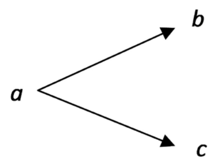
Then, there is a sum $(E_1 \cup E_2, R_1 \cup R_2) = (\{a,b,c\}, \{\varnothing \vdash a, \{a\} \vdash b, \{a\} \vdash c\})$ which can be represented by **Figure 6**.

In general, when $S + T$ is formed, we assume that the event structures S and T belong to the same process. If e occurs in both structures, it refers to the same event.



$$\begin{tikzcd} & X & \\ S \arrow[ur, "s"] \arrow[r, "i_s"'] & S+T \arrow[u, "h"] & T \arrow[ul, "t"'] \arrow[l, "i_t"] \end{tikzcd}$$

Figure 5. The sum $S+T$.



$$\begin{tikzcd} & b \\ a \arrow[ur] \arrow[dr] & \\ & c \end{tikzcd}$$

Figure 6. The sum $(E_1 \cup E_2, R_1 \cup R_2)$.

5.4. Product

The purpose of the product is to combine concurrent processes, each defined by an object, *i.e.*, event structure. To do this, two kinds of event must be distinguished:

Let $P_1 = (E_1, R_1)$ and $P_2 = (E_2, R_2)$ be objects corresponding to concurrent processes and let $e \in E_1$. Then:

- If $e \in E_2$, it indicates that e is a communication event and e must occur simultaneously in both processes.
- Otherwise, $e \notin E_2$, and e belongs to process P only. In this case, e is an independent event.

Product of P_1 and P_2 can be constructed as follows:

- Let $E^{\times} = E_1 \cap E_2$ be the set of events that communicate, namely the *communication set*. The communication set is as large as possible, which means that the only events that cannot be added to it are not communication events.
- The ordering relation $R^{\times}$ is the union $R_1 \cup R_2$ restricted to elements of $E^{\times}$. Formally,

$$R^{\times} = \left\{ (e_1, e_2) \mid e_1 \in E^{\times} \text{ and } e_2 \in E^{\times} \text{ and } (e_1, e_2) \in R_1 \cup R_2 \right\}$$

So, the definition of product is provided as follows:

Definition 6. Let $S = (E_1, R_1)$ and $T = (E_2, R_2)$, then

$$\begin{aligned} S \times T &= (E_1, R_1) \times (E_2, R_2) \\ &= \left(E_1 \cap E_2, \left\{ (e_1, e_2) \mid e_1 \in E^{\times} \text{ and } e_2 \in E^{\times} \text{ and } (e_1, e_2) \in R_1 \cup R_2 \right\} \right) \end{aligned}$$

- By construction, $S \times T$ contains only events that occur in S and T and is therefore contained in both of them. Thus the projections $S \times T \xrightarrow{\pi_s} S$ and $S \times T \xrightarrow{\pi_t} T$ are well-defined.
- Assuming that there is an object X and morphisms $X \xrightarrow{s} S$ and $X \xrightarrow{t} T$. Then X must be contained in both S and T. Therefore it must also be contained in $S \times T$ which is the *largest* object contained in both S and T. Hence the morphism $X \xrightarrow{h} S \times T$ exists and is unique.

To illustrate the sum $S \times T$, **Figure 7** describes the product and it commutes.

We can use product to build *largest communication* structures, which can be explained by using Example 5 as follows:

Example 5. Let $P = (E_P, R_P)$ and $Q = (E_Q, R_Q)$ defined as in **Figure 8**, in which the c's are communicating events and the e's are independent events:

Then the product $P \times Q$ has the events $E_P \cap E_Q = \{c_1, c_2\}$ and the ordering is just $\{c_1\} \vdash c_2$. The processes execute as shown, with time increasing from left to right, synchronizing at c_1 and c_2. The execution independent events, such as e_3, are ignored, as they don't affect the communications. So, independent events are abstracted out of the system.

6. Use Categorical Structures to Model the Vending Machine

In Section 4, this paper provided a vending machine in Example 3 which is modeled by LES. In this section, we use categorical structures defined in Section 5 to model the vending machine example.

The objects of the vending machine are listed as follows:

$$A = (\{\}, \{\}),$$

$$B = (\{coin\}, \{\varnothing \vdash coin\}),$$

$$C = (\{coin, coke\}, \{\varnothing \vdash coin, \{coin\} \vdash coke\}),$$

$$D = (\{coin, pepsi\}, \{\varnothing \vdash coin, \{coin\} \vdash pepsi\}),$$

$$E = (\{coin, coke\}, \{\varnothing \vdash coin, \{coin\} \vdash coke, \{coke\} \vdash stop\}),$$

$$F = (\{coin, pepsi\}, \{\varnothing \vdash coin, \{coin\} \vdash pepsi, \{pepsi\} \vdash stop\}),$$

$$G = (\{coin, coke\}, \{\varnothing \vdash coin, \{coin\} \vdash coke, \{coin\} \vdash pepsi\}),$$

$$H = (\{coin, coke\}, \{\varnothing \vdash coin, \{coin\} \vdash coke, \{coin\} \vdash pepsi, \{coke\} \vdash stop\}),$$

$$I = (\{coin, coke\}, \{\varnothing \vdash coin, \{coin\} \vdash coke, \{coin\} \vdash pepsi, \{pepsi\} \vdash stop\}),$$

$$J = (\{coin, coke\}, \{\varnothing \vdash coin, \{coin\} \vdash coke, \{coin\} \vdash pepsi,$$
$$\{coke\} \vdash stop, \{pepsi\} \vdash stop\}),$$

The morphisms between objects are listed as follows, where identities and composites are ignored:

$$A \to B, B \to C, B \to D, C \to E, D \to F, C \to G, D \to G,$$
$$E \to H, F \to I, G \to H, G \to I, H \to J, I \to J$$

The diagram of the category of the vending machine is illustrated in **Figure 9**.

In **Figure 9**, there exist some products and sums, where a product represents the largest communication structure of two objects and a sum represents the branching structure of two objects.

$$\begin{tikzcd}
 & X \arrow[dl, "s"'] \arrow[d, "h"] \arrow[dr, "t"] & \\
S & S \times T \arrow[l, "\pi_s"] \arrow[r, "\pi_t"'] & T
\end{tikzcd}$$

Figure 7. The sum $S \times T$.

$$\begin{tikzcd}
P: & e_1 \arrow[r] & c_1 \arrow[rr] \arrow[d, dash, dashed] & & c_2 \arrow[r] \arrow[d, dash, dashed] & e_2 \\
Q: & & c_1 \arrow[r] & e_3 \arrow[r] & c_2 \arrow[r] & e_4
\end{tikzcd}$$

Figure 8. Communication events.

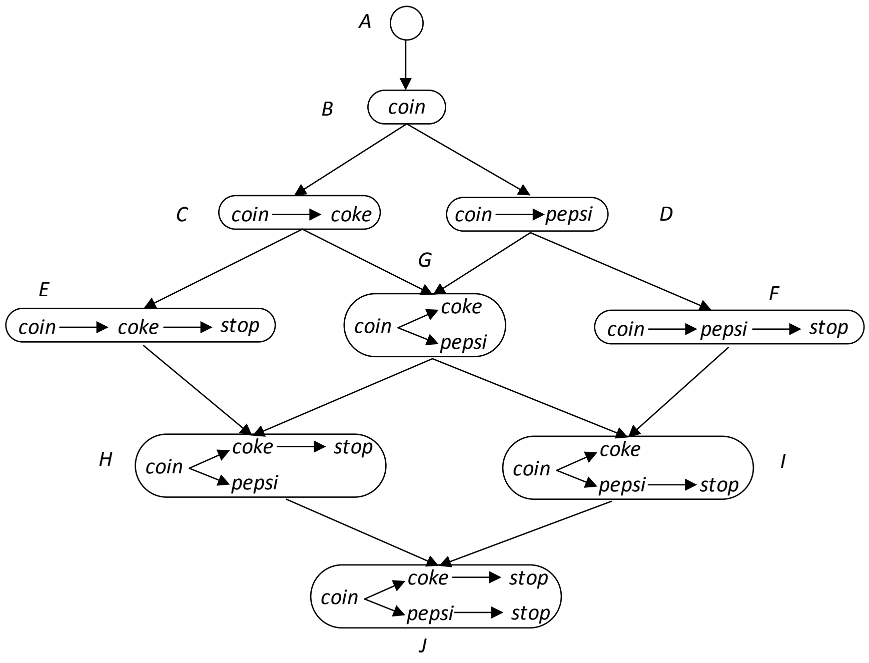


Figure 9. Category of the vending machine.

- Products of the category of the vending machine are listed as follows:

$$G \times F = D, H \times F = D, H \times I = G, E \times G = C, H \times F = D, E \times F = B$$

The above-mentioned products are just some in the category, while there are other products that are not listed specifically in above.

- Sums of the category of the vending machine are listed as follows:

$$C + D = G, E + G = H, G + F = I, H + I = J, E + D = H, E + F = J, C + F = I$$

The above-mentioned sums are just some in the category, while there are other sums that are not listed specifically in above.

7. Conclusion

In view of the difficulties in the development of concurrent systems, the present work proposes a new approach to model LES based on the category theory, which helps to explore the communication events and relationship among them. Specifically, categorical object, morphism, product, and sum are constructed for events and relationships. To explain the work, a vending machine example is designed by LES, and the communications between vending machines and persons are modeled by category theory. By adopting the categorical approach, we can explore and model concurrent systems designed by LES with categorical structures, which can be specified, proved and composed formally with preserving their properties. In Future, we will explore the usage of more categorical structures, such as limit/colimit and natural transformation in LES which may be useful for the formalization and verification of communications.

Acknowledgements

We thank the editor and the referee for their comments. Research of M. Zhu and J. Li is funded by the Shandong University of Technology grants 4041-416069 and 4041-417010. The support is greatly appreciated.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Winskel, G. (1989) An Introduction to Event Structures. *Proceeding of Linear, Time, Branching Time and Partial Order in Logics and Models for Concurrency*, Berlin, 30 May-3 June 1989, 364-397.

[2] Baldan, P., Busi, N., Corradini, A. and Pinna, G.M. (2000) Domain and Event Structure Semantics for Petri Nets with Read and Inhibitor Arcs. *Theoretical Computer Science*, **323**, 129-189. https://doi.org/10.1016/j.tcs.2004.04.001

[3] Kuang, H. (2013) Towards a Formal Reactive Autonomic Systems Framework Using Category Theory. Ph.D. Thesis, Concordia University, Montreal.

[4] Sampson, A.T. (2008) Process-Oriented Patterns for Concurrent Software Engineering. Ph.D. Thesis, University of Kent, Kent.

[5] Hinchey, M.G., Rouff, C.A., Rash, J.L. and Truszkowski, W.F. (2005) Requirements of an Integrated Formal Method for Intelligent Swarms. *Proceedings of the 10th International Workshop on Formal Methods for Industrial Critical Systems*, Lisbon, 5-6 September 2005, 125-133. https://doi.org/10.1145/1081180.1081196

[6] Zhu, M., Grogono, P. and Ormandjieva, O. (2017) Exploring Relationships between Syntax and Semantics of a Process-Oriented Language by Category Theory. *Proceedings of the 8th International Conference on Ambient Systems, Networks and Technologies*, Madeira, 16-19 May 2017, 241-248. https://doi.org/10.1016/j.procs.2017.05.342

[7] Zhu, M., Grogono, P., Ormandjieva, O. and Kuang H. (2016) Using Failures and Category Theory to Verify Process Communications between Design and Implementation of Concurrent Systems. *Proceedings of the 7th International Conference on Ambient Systems, Networks and Technologies*, Madrid, 23-26 May 2016, 700-704. https://doi.org/10.1016/j.procs.2016.04.155

[8] Loogen, R. and Goltz, U. (1991) Modelling Nondeterministic Concurrent Processes with Event Structures. *Fundamenta Informaticae*, **14**, 39-73.

[9] León, H.P.D., Haar, S. and Longuet, D. (2012) Conformance Relations for Labeled Event Structures. *Proceedings of the 6th International Conference on Tests and Proofs*, Prague, 31 May-1 June 2012, 83-98. https://doi.org/10.1007/978-3-642-30473-6_8

[10] Castellan, S. and Clairambault, P., Rideau, S. and Winskel, G. (2016) Concurrent Games. *Logical Methods in Computer Science*, **13**, 1-51.

[11] Bruni, R., Melgratti, H. and Montanari, U. (2006) Event Structure Semantics for Nominal Calculi. *Proceedings of 17th International Conference on Concurrency Theory*, Bonn, 27-30 August 2006, 295-309.

[12] Winskel, G. and Nielsen, M. (1995) Models for Concurrency. Handbook of Logic in Computer Science, Vol. 4, 1-148.

[13] Nielsen, M., Sassone, V. and Winskel, G. (1996) Models for Concurrency: Towards a Classification. *Theoretical Computer Science*, **170**, 297-348. https://doi.org/10.1016/S0304-3975(96)80710-9

[14] Hildebrandt, T.T. (2000) Categorical Models for Fairness: Completion vs. Delay. *Proceedings of the First Irish Conference on the Mathematical Foundations of Computer Science and Information Technology*, Cork, 20-21 July 2000, 188.

[15] Sisiaridis, D., Kuchta, V. and Markowitch, O. (2016) A Categorical Approach in Handling Event-Ordering in Distributed Systems. *Proceedings of IEEE 22nd International Conference on Parallel and Distributed Systems*, Wuhan, 13-16 December 2016, 1145-1150. https://doi.org/10.1109/ICPADS.2016.0150

[16] Enguix, G.B. and Lopez, M.D.J. (2007) Agent-Environment Interaction in a Multi-Agent System: A Formal Model. *Proceedings of the GECCO Conference on Genetic and Evolutionary Computation*, London, 7-11 July 2007, 2607-2612.

[17] Goubault, E. and Mimram, S. (2010) Formal Relationships between Geometrical and Classical Models for Concurrency. *Proceedings of the Workshop on Geometric and Topological Methods in Computer Science*, Denmark, 11-15 January 2010, 77-109.

[18] Abdel Gawad, M.A. (2017) Novel Uses of Category Theory in Modeling OOP. CoRR, abs/1709.08056, 1-8.