Causality in DyNetKAT

Amir Hossein Seyhani

September 2022

Contents

1 Preliminaries

1.1 Event Structure [5]

Definition 1.1.1 (Event Structure)**.** An event structure is a triple $\mathrm{E} = (E, \#, \vdash)$ where:

1. E is a set of events
2. $\#$ is a binary symmetric, irreflexive relation on E, the conflict relation. We shall write Con for the set of conflict-free subsets of E, i.e. those finite subsets $X \subseteq E$ for which: $\forall e, e' \in X. \neg(e \# e')$
3. $\vdash \subseteq Con \times E$ is the enabling relation which satisfies: $X \vdash e \;\&\; X \subseteq Y \in Con \Rightarrow Y \vdash e$

Notion 1. In an event structure we shall write $\mathrel{w}$ for the reflexive conflict relation by which we mean that $e \mathrel{w} e'$ in an event structure iff either $e \# e'$ or $e = e'$. With this notion instead of describing the conflict-free sets of an event structure as those sets X such that

$$\forall e, e' \in X. \neg(e \# e')$$

we can say they are those sets X for which:

$$\forall e, e' \in X. e \mathrel{w} e' \Rightarrow e = e'$$

Notion 2. For any event structure we can define the minimal enabling relation $\vdash_{min}$ by:

$$X \vdash_{min} e \iff X \vdash e \wedge (\forall Y \subseteq X. Y \vdash e \Rightarrow Y = X)$$

Then for any event structure:

$$Y \vdash e \Rightarrow \exists X \subseteq Y. X \vdash_{min} e$$

Definition 1.1.2 (Configuration)**.** Let $\mathrm{E} = (E, \#, \vdash)$ be an event structure. Define a configuration of E to be a subset of events $x \subseteq E$ which is

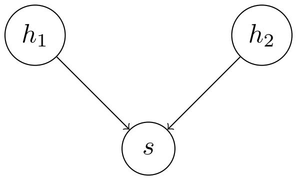
1. conflict-free: $x \in Con$
2. secured: $\forall e \in x. \exists e_0, ..., e_n \in x. e_n = e \;\wedge \forall i \leq n. \{e_0, ..., e_{i-1}\} \vdash e_i$

The set of all configurations of an event structure is written as $\mathcal{F}(E)$. It is helpful to unwrap condition (2) a little. It says an event e is secured in a set x iff there is a sequence of events $e_0, ..., e_n = e$ in x such that:

$$\varnothing \vdash e_0, \{e_0\} \vdash e_1, ..., \{e_0, ..., e_{i-1}\} \vdash e_i, ..., \{e_0, ..., e_{n-1}\} \vdash e_n.$$

We call such a sequence $e_0, e_1, ..., e_n = e$ a *securing* for e in x. We use $X \subseteq_{fin} Y$ to mean X is a finite subset of Y.

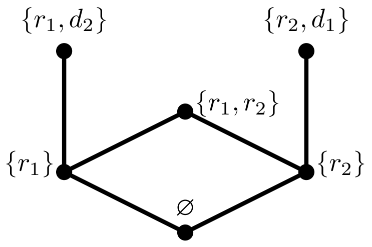
Example 1.1.1. Consider another network in which two hosts are connected to a single switch. This time both hosts are sending a packet concurrently to the host. Once the switch received a packet from either host it will drop the packets received from the other host due to capacity limit.



Let r_1, r_2 represent the events of receiving packet from h_1 and h_2 and d_1, d_2 represent the events of dropping packets from these hosts. We need a conflict relation the least one for which we have $r_1 \# d_1$ and $r_2 \# d_2$ and enabling relation the least one that satisfies:

$$\begin{aligned} \varnothing &\vdash_{min} r_1 \\ \varnothing &\vdash_{min} r_2 \\ \{r_1\} &\vdash_{min} d_2 \\ \{r_2\} &\vdash_{min} d_1 \end{aligned}$$

Thus the configurations have the form:



Stable Event Structure

We look for a special class of event structures for which there is a the partial order of causal dependency on each configuration. This can not be done so obviously for all event structures. Consider the event structure of the previous example in which the event b causally depends not on a unique set of events but rather on either the occurrence of 0 or on the occurrence of 1. It is incorrect to say b causally depends on both 0 and 1 because the occurrence of only one of them enables the occurrence of b. The difficulty arises because there is a configuration $\{0, 1, b\}$ in which there is an event b which is not enabled by a unique minimal set of event occurrences. We can rule out such possibilities by insisting on event structures satisfy the following stability axiom.

Definition 1.1.3 (Stable Event Structure)**.** Let E $= (E, \#, \vdash)$ be an event structure. Say E is stable if it satisfies the following axiom:

$$X \vdash e \ \wedge \ Y \vdash e \ \wedge \ X \cup Y \cup \{e\} \in Con \Rightarrow X \cap Y \vdash e$$

The stability axiom ensures that an event in a configuration is enabled in an essentially unique way. Assume e belongs to a configuration x of a stable event structure. Suppose $X \vdash e$ and $X \subseteq x$. Then $X \cup \{e\} \in Con$, the enabling $X \vdash e$ is consistent. Take

$$X_0 = \bigcap \{Y \mid Y \subseteq X \wedge Y \vdash e\}$$

Because X is finite this is an intersection of a finite number of sets and we see by the stability axiom that $X_0 \vdash e$. Moreover X_0 is the unique minimal subset of X which enables e. Thus for stable event structures, we have:

$$Y \vdash e \wedge Y \cup \{e\} \in Con \Rightarrow \exists! X \subseteq Y . X \vdash_{min} e$$

It follows that for stable event structures

$$X \vdash_{min} e \wedge Y \vdash_{min} \wedge X \cup Y \cup e \in Con \Rightarrow X = Y$$

Definition 1.1.4. Let E$_0 = (E_0, \#_0, \vdash_0)$ and E$_1 = (E_1, \#_1, \vdash_1)$ be event Structures. Define

$$\begin{aligned} \mathrm{E}_0 \trianglelefteq \mathrm{E}_1 \iff & E_0 \subseteq E_1, \\ & \forall e, e'. e \#_0 e' \iff e, e' \in E_0 \ \wedge \ e \#_1 e' \text{ and} \\ & \forall X, e. X \vdash_0 e \iff X \subseteq E_0 \ \wedge \ e \in E_0 \ \wedge \ X \vdash_1 e \end{aligned}$$

In this case say E$_0$ is a substructure of E$_1$.

Definition 1.1.5 (Restriction)**.** Let E $= (E, \#, \vdash)$ be an event structure. Let $A \subseteq E$. Define the restriction of E to A to be

$$\mathrm{E} \lceil A = (A, \#_A, \vdash_A)$$

where

$$\begin{aligned} X \in Con_A &\iff X \subseteq A \ \wedge \ X \in Con \\ X \vdash_A e &\iff X \subseteq A \ \wedge \ e \in A \ \& \ X \vdash e \end{aligned}$$

Definition 1.1.6. Let a be an event. For an event structure E $= (E, \#, \vdash)$ define aE to be the event structure $(E', \#', \vdash')$ where:

$$\begin{aligned} & E' = \{(0, a)\} \cup \{(1, e) | e \in E\}, \\ & e'_0 \#' e'_1 \iff \exists e_0, e_1 . e'_0 = (1, e_0) \ \wedge \ e'_1 = (1, e_1) \ \wedge \ e_0 \# e_1 \\ & X \vdash' e' \iff e' = (0, a) \text{ or } [e' = (1, e_1) \ \wedge \ (0, a) \in X \ \wedge \ \{e | (1, e) \in X\} \vdash e_1] \end{aligned}$$

Definition 1.1.7. A labelled event structure consists of $(E, \#, \vdash, L, l)$ where $(E, \#, \vdash)$ is an event structure, L is a set of labels, not including the element *, and l is a function $l : E \to L$ from its events to its labels.

1.2 Causal Model [3]

A signature $\mathcal{S}$ is a tuple $(\mathcal{U}, \mathcal{V}, \mathcal{R})$, where $\mathcal{U}$ is a set of exogenous variables, $\mathcal{V}$ is a set of endogenous variables, and R associates with every variable $Y \in \mathcal{U} \cup \mathcal{V}$ a nonempty set $\mathcal{R}(Y)$ of possible values for Y. A causal model (or structural model) over signature S is a tuple $M = (\mathcal{S}, \mathcal{F})$, where $\mathcal{F}$ associates with each variable $X \in \mathcal{V}$ a function denoted F_X such that $F_X : (\times_{U \in \mathcal{U}} \mathcal{R}(U)) \times (\times_{Y \in \mathcal{V} - \{X\}} \mathcal{R}(Y)) \to \mathcal{R}(X)$.

F_X determines the value of X given the values of all the other variables in $\mathcal{U} \cup \mathcal{V}$. For example, if $F_X(Y, Z, U) = Y + U$ (which we usually write as $X = Y + U$), then if $Y = 3$ and $U = 2$, then $X = 5$, regardless of how Z is set. These equations can be thought of as representing processes (or mechanisms) by which values are assigned to variables. Hence, like physical laws, they support a counterfactual interpretation. For example, the equation above claims that in the context $U = u$, if Y were 4, then X would be $u + 4$ (which we write as $(M, u) \models [Y \leftarrow 4](X = u + 4)$), regardless of what values X, Y, and Z actually take in the real world.

The function $\mathcal{F}$ defines a set of (*modifiable*) *structural equations* relating to the values of the variables.

Example 1.2.1. Suppose that we want to reason about a forest fire that could be caused by either lightning or a match lit by an arsonist. Then the causal model would have the following endogenous variables :

- F for fire
- L for lighting
- ML for match lit

The set $\mathcal{U}$ of exogenous variables includes conditions that suffice to render all relationships deterministic (such as whether the wood is dry, whether there is enough oxygen in the air for the match to light, etc.). Suppose that $\vec{u}$ is a setting of the exogenous variables that makes a forest fire possible (i.e., the wood is sufficiently dry, there is oxygen in the air, and so on). Then, for example, $F_F(\vec{u}, L, ML)$ is such that $F = 1$ if either $L = 1$ or $ML = 1$. Note that although the value of F depends on the value L and ML, the value of L does not depend on the values of F and ML.

1.2.1 Causal Network

We can describe a causal model $\mathcal{M}$ using a causal network. This is a graph with nodes corresponding to the random variables in $\mathcal{V}$ and an edge from a node labeled X to one labeled Y if F_Y depends on the value of X. This graph is a dag that follows from the assumption that the equations are recursive. We occasionally omit the exogenous variables $\vec{U}$ from the causal network. For example, the causal network for example 2.2.1 has the following form:

$$\begin{tikzcd}
\text{L} \arrow[dr] & & \text{F} \arrow[dl] \\
& \text{ML} &
\end{tikzcd}$$

1.2.2 Actual Cause

Given a signature $S = (\mathcal{U}, \mathcal{V}, \mathcal{R})$, a formula of the form $X = x$, for $X \in \mathcal{V}$ and $x \in \mathcal{R}(X)$, is called a *primitive event*. A *basic causal formula* is one of the form $[Y_1 \leftarrow y, ..., Y_l \leftarrow y_k]\varphi$, where φ is a Boolean combination of primitive events, $Y_1, ..., Y_k$ are distinct variables in $\mathcal{V}$, and $y_i \in \mathcal{R}(Y_i)$. Such a formula is abbreviated as $[\vec{Y} \leftarrow \vec{y}]\varphi$. A *causal formula* is a Boolean combination of basic causal formulas. A causal formula ψ is true or false in a causal model, given a context. We write $(M, \vec{u}) \vDash \psi$ if ψ is true in causal model M given context $\vec{u}$. $(M, \vec{u}) \vDash [\vec{Y} \leftarrow \vec{y}](X = x)$ if the variable X has value x in the unique solution to the equation in $M_{\vec{Y} \leftarrow \vec{y}}$ in context $\vec{u}$. The context and structural equations are given. They encode the background knowledge. All relevant events are known. The only question is picking out which of them are the cause of φ or, alternatively, testing whether a given set of events can be considered the cause of φ. The types of events that we allow as actual causes are ones of the form $X_1 = x_1 \wedge ... \wedge X_k = x_k$– that is, conjunctions of primitives events. We abbreviate this as $\vec{X} = \vec{x}$.

Definition 1.2.1. $\vec{X} = \vec{x}$ is an actual cause of φ in $(M, \vec{u})$ if the following three conditions hold:

- **AC1.** $(M, \vec{u}) \vDash (\vec{X} = \vec{x}) \wedge \varphi$. (both $\vec{X} = \vec{x}$ and φ are true in actual world)
- **AC2.** There exists a partition $(\vec{Z}, \vec{W})$ of $\mathcal{V}$ with $\vec{X} \subseteq \vec{Z}$ and some setting $(\vec{x}', \vec{w}')$ of the variables in $(\vec{X}, \vec{W})$ such that if $(M, \vec{u}) \vDash \vec{Z} = z^*$ for all $Z \in \vec{Z}$, then both of the following conditions hold:

  (a) $(M, \vec{u}) \vDash [\vec{X} \leftarrow \vec{x}', \vec{W} \leftarrow \vec{w}']\neg\varphi$.

  (b) $(M, \vec{u}) \vDash [\vec{X} \leftarrow \vec{x}, \vec{W}' \leftarrow \vec{w}', \vec{Z}' \leftarrow \vec{z}^*]\varphi$ for all subsets $\vec{W}'$ of $\vec{W}$ and all subsets Z' of $\vec{Z}$.
- **AC3.** $\vec{X}$ is minimal; no subset of $\vec{X}$ satisfies conditions $AC1$ and $AC2$.

We call the tuple $(\vec{W}, \vec{w}, \vec{x}')$ a witness to the fact that $\vec{X} = \vec{x}$ is a cause of φ.

Definition 1.2.2. We say $\vec{X} = \vec{x}$ is a but-for cause of φ in $(M, \vec{u})$ if there exists a witness $(\vec{W}, \vec{w}, \vec{x}')$ for $\vec{X} = \vec{x}$ being an actual cause of φ where $\vec{W} = \varnothing$.

Note that, if we consider a witness $(\vec{W}, \vec{w}, \vec{x}')$ for checking whether $\vec{X} = \vec{x}$ is a cause of φ in $(M, \vec{u})$ where $\vec{W} = \varnothing$, then in the AC2(b) condition we only need to check whether $(M, \vec{u}) \vDash [\vec{X} \leftarrow \vec{x}, \vec{Z}' \leftarrow \vec{z}^*]\varphi$ for all subsets $\vec{Z}'$ of $\vec{Z}$. Since we have $(M, \vec{u}) \vDash (\vec{X} = \vec{x})$ and $(M, \vec{u}) \vDash Z = z^*$ for all $Z \in \vec{Z}$, the Interventions $\vec{X} \leftarrow \vec{x}$ and $\vec{Z}' \leftarrow \vec{z}^*$ actually do not change the value of any variable thus checking whether $(M, \vec{u}) \vDash [\vec{X} \leftarrow \vec{x}, \vec{Z}' \leftarrow \vec{z}^*]\varphi$ is true reduces to check whether $(M, \vec{u}) \vDash \varphi$ which must be already satisfied when we have checked AC1 condition. This means that, to check whether $\vec{X} = \vec{x}$ is an actual cuase when using a witness with an empty $\vec{W}$ we only need to check AC1 and AC2(a) conditions.

1.3 Extended Causal Model

An extended causal model is a tuple $(\mathcal{S}, \mathcal{F}, \mathcal{E})$, where $(\mathcal{S}, \mathcal{F})$ is a causal model, and $\mathcal{E}$ is a set of allowable settings for the endogenous variables. That is, if the endogenous variables are $X_1, ..., X_n$ then $(x_1, ..., x_n) \in \mathcal{E}$ if $X_1 = x_1, ..., X_n = x_n$ is an allowable setting. We say that a setting of a subset of the endogenous variables is allowable if it can be extended to a setting in $\mathcal{E}$.

In [3] there is no formal definition of the actual cause in extended causal models. So, here we provide a new definition of the actual cause in the context of extended causal models.

Definition 1.3.1. $\vec{X} = \vec{x}$ is an actual cause of φ in $(M, \vec{u})$ if the following three conditions hold:

- **AC1.** $(M, \vec{u}) \models (\vec{X} = \vec{x}) \wedge \varphi \wedge$.
- **AC2.** There exists a partition $(\vec{Z}, \vec{W})$ of $\mathcal{V}$ with $\vec{X} \subseteq \vec{Z}$ and some setting $(\vec{x}', \vec{w}')$ of the variables in $(\vec{X}, \vec{W})$ such that if $(M, \vec{u}) \models \vec{Z} = z^*$ for all $Z \in \vec{Z}$, then both of the following conditions hold:

  (a) $(M, \vec{u}) \models [\vec{X} \leftarrow \vec{x}', \vec{W} \leftarrow \vec{w}'] \neg\varphi \wedge \vec{V} = \vec{v} \wedge \vec{v} \in \mathcal{E}$.

  (b) $(M, \vec{u}) \models [\vec{X} \leftarrow \vec{x}, \vec{W}' \leftarrow \vec{w}', \vec{Z}' \leftarrow \vec{z}^*] \vec{V} = \vec{v} \wedge (\vec{v} \in \mathcal{E} \Rightarrow \varphi)$ for all subsets $\vec{W}'$ of $\vec{W}$ and all subsets Z' of $\vec{Z}$.
- **AC3.** $\vec{X}$ is minimal; no subset of $\vec{X}$ satisfies conditions $AC1$ and $AC2$.

Where $\vec{v}$ is the value of endogenous variables.

2 Denotational Semantics of DyNetKAT

In this section, first, we introduce Normal DyNetKAT grammar. Then we define different constructions on labeled event structures which will be used to provide denotational semantics for Normal DyNetKAT terms.

2.1 Normal DyNetKAT

We define the Normal DyNetKAT grammar as follows:

$$\begin{aligned} F &::= \alpha \cdot \pi \\ D &::= \bot \mid F; D \mid x?F; D \mid x!F; D \mid D \parallel D \mid D \oplus D \mid \delta_{\mathcal{L}}(D) \\ \mathcal{L} &= \{c \mid c ::= x?F \mid x!F\} \end{aligned}$$

Lemma 9 in [1] says that for a given guarded DyNetKAT term p there exists a DyNetKAT term in normal form q for which we have:

$$E_{DNK} \vdash p \equiv q$$

where E_{DNK} is DyNetKAT axiom system.

2.2 Constructions on Labeled Event Structures [5]

We use $(\varnothing, \varnothing)$ to denote an empty labeled event structure with an empty set of events and an empty set of labels. In the following, we define four types of constructions on labeled event structures.

2.2.1 Prefix

Definition 2.2.1. Let (E, L, l) be a labelled event structure. Let α be a label. Define $\alpha(\mathrm{E}, L, l)$ to be a labelled event structure $(\alpha\mathrm{E}, L', l')$ with labels:

$$L' = \{\alpha\} \cup L$$

and

$$l'(e') = \begin{cases} \alpha & \text{if } e' = (0, \alpha) \\ l(e) & \text{if } e' = (1, e) \end{cases}$$

for all $e' \in E'$.

2.2.2 Sum

Definition 2.2.2. Let $\mathrm{E}_0 = (E_0, \#_0, \vdash_0, L_0, l_0)$ and $\mathrm{E}_1 = (E_1, \#_1, \vdash_1, L_1, l_1)$ be labelled event structures. Their sum $\mathrm{E}_0 + \mathrm{E}_1$, is defined to be the structure $(E, \#, \vdash, l)$ with events $E = \{(0, e) | e \in E_0\} \cup \{(1, e) | e \in E_1\}$, the disjoint union of sets E_0 and E_1, with injections $\iota_k : E_k \to E$, given by $\iota_k(e) = (k, e)$, for $k = 0, 1$, conflict relation

$$\begin{aligned} e \# e' \iff & \exists e_0, e_0'. e = \iota_0(e_0) \wedge e' = \iota_0(e_0') \wedge e_0 \#_0 e_0' \\ & \text{or } \exists e_1, e_1'. e = \iota_1(e_1) \wedge e' = \iota_1(e_1') \wedge e_1 \#_1 e_1' \\ & \text{or } \exists e_0, e_1. (e = \iota_1(e_0) \wedge e' = \iota_1(e_1)) \text{ or } (e' = \iota_1(e_0) \wedge e = \iota_1(e_1)) \end{aligned}$$

and enabling relation

$$\begin{aligned} X \vdash e \iff & X \in Con \wedge e \in E \wedge \\ & (\exists X_0 \in Con_0, e_0 \in E_0. X = \iota_0 X_0 \wedge e = \iota_0(e_0) \wedge X_0 \vdash_0 e_0) \text{ or} \\ & (\exists X_1 \in Con_1, e_1 \in E_1. X = \iota_1 X_1 \wedge e = \iota_1(e_1) \wedge X_1 \vdash_1 e_1) \end{aligned}$$

We define the set of labels as $L_0 \cup L_1$ and the labelling function as:

$$l(e) = \begin{cases} l_0(e_0) & \text{if } e = \iota_0(e_0) \\ l_1(e_1) & \text{if } e = \iota_1(e_1) \end{cases}$$

2.2.3 Product

In the product of two event structures, their events of synchronization are those pairs of events (e_0, e_1), one from each event structure; if e_0 is labelled α_0 and e_1 is labelled α_1 the synchronization event is then labelled (α_0, α_1). Events need not synchronize however; an event in one component may not synchronize with any event in the other. We shall use events of the form $(e_0, *)$ to stand for the occurrence of an event e_0 from one component unsynchronized with any event of the other. Such an event will be labeled by $(\alpha_0, *)$ where α_0 is the original label of e_0 and * is a sort of undefined.

Definition 2.2.3. Let $\mathrm{E}_0 = (E_0, \#_0, \vdash_0, L_0, l_0)$ and $\mathrm{E}_1 = (E_1, \#_1, \vdash_1, L_1, l_1)$ be labeled event structures. Define their product $\mathrm{E}_0 \times \mathrm{E}_1$ to be the structure $\mathrm{E} = (E, \#, \vdash, L, l)$ consisting of events E of the form

$$E_0 \times_* E_1 = \{(e_0, *) | e_0 \in E_0\} \cup \{(*, e_1) | e_1 \in E_1\} \cup \{(e_0, e_1) | e_0 \in E_0 \land e_1 \in E_1\}$$

with projections $\pi_i : E \to_* E_i$, given by $\pi_i(e_0, e_1) = e_i$, for $i = 0, 1$, reflexive conflict relation $\mathrel{\mathrm{w}}$ given by

$$e \mathrel{\mathrm{w}} e' \iff \pi_0(e) \mathrel{\mathrm{w}_0} \pi_0(e') \text{ or } \pi_1(e) \mathrel{\mathrm{w}_1} \pi_1(e')$$

for all e, e' we use Con for the conflict-free finite sets, enabling relation $\vdash$ given by

$$X \vdash e \iff X \in Con \land e \in E \land$$
$$(\pi_0(e) \text{ is defined } \Rightarrow \pi_0 X \vdash_0 \pi_0(e)) \land (\pi_1(e) \text{ is defined } \Rightarrow \pi_1 X \vdash_1 \pi_1(e))$$

Its set of labels is

$$L_0 \times_* L_1 = \{(\alpha_0, *) | \alpha_0 \in L_0\} \cup \{(*, \alpha_1) | \alpha_1 \in L_1\} \cup \{(\alpha_0, \alpha_1) | \alpha_0 \in L_0 \land \alpha_1 \in L_1\}$$

with projections: $\lambda_i : E \to_* E_i$ given by $\lambda_i(\alpha_0, \alpha_1) = \alpha_i$, for $i = 0, 1$. Its labeling function is defined to act on an event e so

$$l(e) = (l_0 \pi_0(e), l_1 \pi_1(e))$$

2.2.4 Restriction

Definition 2.2.4. Let $\mathrm{E} = (E, \#, \vdash, L, l)$ be a labelled event structure. Let Λ be a subset of labels. Define the restriction $\mathrm{E}\lceil\Lambda$ to be $(E', \#', \vdash', L \cap \Lambda, l')$ where $(E', \#', \vdash')$ is the restriction of $(E, \#, \vdash)$ to events $\{e \in E | l(e) \in \Lambda\}$ and the labeling function l' is the restriction of the original labeling function to the domain $L \cap \Lambda$.

2.3 Denotational Semantics of Normal DyNetKAT

Definition 2.3.1. Let $\mathcal{A}$ be an alphabet of letters of the form $\alpha \cdot \pi$, $x?F$, and $x!F$. We define the semantic map of Normal DyNetKAT terms $[\![\ ]\!] : D \to \mathbb{E}$ where $\mathbb{E}$ is the set of all event structures with labels in $\mathcal{A}$ as follows:

$$[\![\bot]\!] = (\varnothing, \varnothing)$$

$$
\begin{aligned}
[\![\alpha; t]\!] &= \alpha[\![t]\!] \\
[\![t_1 \oplus t_2]\!] &= [\![t_1]\!] + [\![t_2]\!] \\
[\![\delta_{\mathcal{L}}(t)]\!] &= [\![t]\!]\lceil(\mathcal{A} \smallsetminus \mathcal{L}) \\
[\![t_1 \parallel t_2]\!] &= [\![t_1]\!] \times [\![t_2]\!]
\end{aligned}
$$

Where $\mathcal{L}$ is a subset of $\mathcal{A}$.

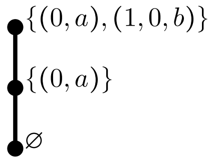
2.4 Examples

In the following, we provide some examples to illustrate how the prefix, sum, and product operators can be used to compose event structures. For simplicity, here we use a, b, c to denote DyNetKAT actions.

Example 2.4.1. Let $a, b \in \mathcal{A}$ be some DyNetKAT actions and $a; b$ a normal DyNetKAT term. E = $[\![a; b]\!]$ is an event structure $(E, \#, \vdash, L, l)$ where:

$$
\begin{aligned}
&E = \{(0,a), (1,0,b)\} \\
&\# = \emptyset \\
&\emptyset \vdash_{min} (0,a), \{(0,a)\} \vdash_{min} (1,0,b) \\
&L = \{a, b\} \\
&l((0,a)) = a, l((1,0,b)) = b
\end{aligned}
$$

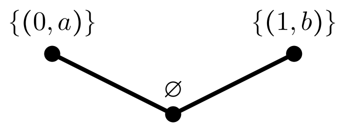
The following diagram shows configurations of E:



Example 2.4.2. Let $a, b \in \mathcal{A}$ be some DyNetKAT actions and $a \oplus b$ a normal DyNetKAT term. E = $[\![a \oplus b]\!]$ is an event structure $(E, \#, \vdash, L, l)$ where:

$$
\begin{aligned}
&E = \{(0,a), (1,b)\} \\
&(0,a)\#(1,b), (1,b)\#(0,a) \\
&\emptyset \vdash_{min} (0,a), \emptyset \vdash_{min} (1,b) \\
&L = \{a, b\} \\
&l((0,a)) = a, l((1,b)) = b
\end{aligned}
$$

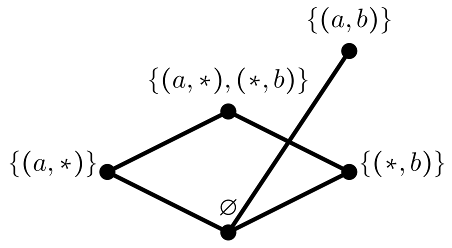
The following diagram shows configurations of E:



Example 2.4.3. Let $a, b \in \mathcal{A}$ be some DyNetKAT actions and $a \parallel b$ a normal DyNetKAT term. E = $[\![a \parallel b]\!]$ is an event structure $(E, \#, \vdash, L, l)$ where:

$$\begin{aligned}
&E = \{(a,*), (*,b), (a,b)\}\\
&\# = \varnothing\\
&\quad \varnothing \vdash_{min} (a,*), \varnothing \vdash_{min} (*,b), \varnothing \vdash_{min} (a,b)\\
&L = \{(a,*), (*,b), (a,b)\}\\
&\quad l((a,*)) = (a,*), l((*,b)) = (*,b), l((a,b)) = (a,b)
\end{aligned}$$

The following diagram shows configurations of E:



3 Causal Model of Unsafe Behavior in Event Structure

3.1 Causal Model of Event Structure

Let E = $(E, \#, \vdash)$ be an event structure where $E = \{e_1, e_2, ..., e_n\}$. We define the causal model $\mathcal{M} = (\mathcal{S}, \mathcal{F}, \mathcal{E})$ of unsafe behavior in E where $\mathcal{S} = (\mathcal{U}, \mathcal{V}, \mathcal{R})$. We define $\mathcal{U}$ to be empty a $\mathcal{V}$ consisting of boolean variables as follows:

$$\begin{aligned}
\mathcal{V} =& \{C_{e_i,e_j} \mid 1 \le i < j \le n. e_i \in E \wedge e_j \in E\}\\
&\cup \{EN_{s,e} \mid s \in \mathcal{P}(E), e \in E. e \notin s\}\\
&\cup \{M_{s,e} \mid s \in \mathcal{P}(E), e \in E. e \notin s\} \cup \{PV\}
\end{aligned}$$

For each variable $X \in \mathcal{V}$ we define $\vec{V}_X$ as a vector of all variables in $\mathcal{V} \setminus \{X\}$. For $x, y \in \mathcal{P}(E)$ we say x is covered by y written $x \prec y$ iff:

$$x \subseteq y \wedge x \neq y \wedge (\forall z. x \subseteq z \subseteq y \Rightarrow x = z \vee y = z)$$

We define the functions in $\mathcal{F}$ as follows:

$$F_{C_{e,e'}}(\vec{V}_{C_{e,e'}}) = \begin{cases} true & \text{if } e \# e' \\ false & \text{otherwise} \end{cases}$$

$$F_{M_{s,e}}(\vec{V}_{M_{s,e}}) = \begin{cases} Min(s,e) \wedge Con(s) & \text{if } s \vdash_{min} e \\ false & \text{otherwise} \end{cases}$$

$$F_{EN_{s,e}}(\vec{V}_{EN_{s,e}}) = \left(M_{s,e} \bigvee \left(\bigvee_{s'<s} EN_{s',e}\right)\right) \bigwedge Con(s)$$

Where we have:

$$Con(s) = \left(\bigwedge_{1\leq j<j'\leq n \wedge e_j, e_{j'} \in s} \neg C_{e_j, e_{j'}} \right)$$

$$Min(s,e) = \left(\bigwedge_{s' \subseteq E.(s' \subset s \vee s \subset s') \wedge e \notin s'} \neg M_{s',e} \right)$$

Let $\mathbb{E}$ be the all triples of the form $(E, \#', \vdash')$ on the set of events E where $\#' \subseteq E \times E$ and $\vdash' \subseteq \mathcal{P}(E) \times E$. We define a function $ES : \times_{V \in \mathcal{V} \setminus \{PV\}} \mathcal{R}(V) \to \mathbb{E}$ which intuitively returns the event structure that can be derived from the values of the endogenous variables in the model excluding PV. Let $\vec{v}$ be a vector of the values of the variables in $\mathcal{V} \setminus \{PV\}$ and $\vec{v}(V)$ be the value of V in $\vec{v}$ for each $V \in \mathcal{V} \setminus \{PV\}$. We define the function ES so that if $ES(\vec{v}) = (E, \#', \vdash')$ we have:

$$\forall e, e' \in E. e \#' e' \wedge e' \#' e \iff \vec{v}(C_{e,e'}) = \text{True}$$
$$\forall s \in \mathcal{P}(E), e \in E. s \vdash' e \iff \vec{v}(EN_{s,e}) = \text{True}$$

We define $\mathcal{E}$, the set of all allowable settings of the endogenous variables, to be the vector of values such as $\vec{v}'$ for which $ES(\vec{v})$ is an event structure.

Finally, we encode the property violation or unsafe behaviors as a predicate on the set of configurations of the event structure returned by ES.

3.2 Actual Cause of Unsafe Behavior

Using the definition of the actual cause in the extended causal model, we can rewrite the definition for unsafe behaviors in event structures as follows:

Definition 3.2.1. Let E $= (E, \#, \vdash)$ be an event structure and $\mathcal{M}$ be the causal model of an unsafe behavior in E where unsafe behavior is specified as the function F_{PV} in $\mathcal{M}$. We say $\vec{X} = \vec{x}$ is an actual cause of the unsafe behavior in E if the following conditions hold:

- **AC1.** $M \vDash \vec{X} = \vec{x} \wedge PV = \text{True}$
- **AC2.** There exists a partition $(\vec{Z}, \vec{W})$ of $\mathcal{V}$ with $\vec{X} \subseteq \vec{Z}$ and some setting $(\vec{x}', \vec{w}')$ of the variables in $(\vec{X}, \vec{W})$ such that if $(M, \vec{u}) \vDash \vec{Z} = z^*$ for all $Z \in \vec{Z}$, then both of the following conditions hold:

  (a) $M \vDash [\vec{X} \leftarrow \vec{x}', \vec{W} \leftarrow \vec{w}']PV = \text{False} \wedge \vec{V} = \vec{v} \wedge \vec{v} \in \mathcal{E}$.

  (b) $M \vDash [\vec{X} \leftarrow \vec{x}, \vec{W}' \leftarrow \vec{w}', \vec{Z}' \leftarrow \vec{z}^*]\vec{V} = \vec{v} \wedge (\vec{v} \in \mathcal{E} \Rightarrow PV = \text{True})$ for all subsets $\vec{W}'$ of $\vec{W}$ and all subsets Z' of $\vec{Z}$.
- **AC3.** $\vec{X}$ is minimal; no subset of $\vec{X}$ satisfies conditions $AC1$ and $AC2$.

Where $\vec{v}$ is the value of endogenous variables.

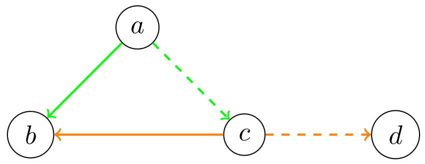


Figure 1:

3.3 Examples

In the following assume that packets have a field sw representing the packet's switch and we have:

$$x \to y \triangleq sw = x \cdot sw \leftarrow y$$

Example 3.3.1. Blacklisted nodes are nodes in the network that must not be reachable [4]. Let's assume the network in figure 1 as an example where the node d is blacklisted. For simplicity we assume that we require d not reachable only from a. Consider the following DyNetKAT program for this network:

$$\begin{aligned} P &= p!1 \\ Q &= q!1 \\ N &= F \oplus p?1; N_p \oplus q?1; N_q \\ N_p &= F_p \oplus q?1; F_{pq} \\ N_q &= F_q \oplus p?1; F_{pq} \\ F &= a \to b \oplus c \to b \end{aligned} \qquad \begin{aligned} F_p &= a \to c \oplus c \to b \oplus a \to b \\ F_q &= a \to b \oplus c \to d \\ F_{pq} &= a \to c \oplus c \to d \oplus a \to d \\ SDN &= \delta_{\mathcal{L}}(N \parallel P \parallel Q) \\ \mathcal{L} &= \{p!1, p?1, q?1, q?1\} \end{aligned}$$

We assume that there are two concurrent processes for updating the switches a and c. Let we use p and q to denote $rcfg(p,1)$ and $rcfg(q,1)$ respectively. p and q replace the solid green and orange paths in the network with dashed paths respectively. Figure 2 shows an excerpt of the LTS of SDN when there is a packet σ_a where $\sigma_a(sw) = a$. Obviously, executing both $rcfg_{p,1}$ and $rcfg_{q,1}$ leads to a state where we can forward the σ_a to d. To find the cause of error, let E = $[\![SDN]\!]$ and $\mathcal{M}$ be the causal model of E where we encoded the unsafe behavior as:

$$F_{PV}(\vec{V}_{PV}) = \exists c \in \mathcal{F}(ES(\vec{v})).\exists e \in c.l(e) = a \to d$$

The function above sets the value of PV to true if $ES(\vec{v})$ contains a configuration with an event labeled $a \to d$. In SDN there are two order execution for p and q, so, there are two events for each of these labels in E and thus, there are two events with the label $a \to d$ as well. So, let's consider events p_1, p_2 with label p, events q_1, q_2 with label q and events ad_1, ad_2 with label $a \to d$ in E. Figure 3 shows a portion of E consist of configurations leading to events labeled with $a \to d$.

In this example we can consider C_{p_1,q_2} = False as a cause of PV = True using the witness (C_{p_2,q_2}, True, True). Let $\vec{v}'$ be the values of variables when we set

$$\begin{tikzcd}
 & \boxed{n_0 : (SDN, \sigma_a, \langle\rangle)} \arrow[dl, "rcfg(p,1)"'] \arrow[dr, "rcfg(q,1)"] & \\
\boxed{n_1 : (\delta_{\mathcal{L}}(N_p \parallel Q), \sigma_a, \langle\rangle)} \arrow[dr, "rcfg(q,1)"'] & & \boxed{n_2 : (\delta_{\mathcal{L}}(N_q \parallel P), \sigma_a, \langle\rangle)} \arrow[dl, "rcfg(p,1)"] \\
 & \boxed{n_3 : (F_{pq}, \sigma_a, \langle\rangle)} \arrow[d, "(\sigma_a, \sigma_d)"'] & \\
 & \boxed{n_4 : (\checkmark, \langle\rangle, \sigma_d)} &
\end{tikzcd}$$

Figure 2:

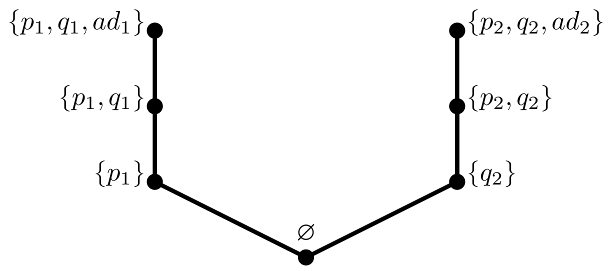


Figure 3:

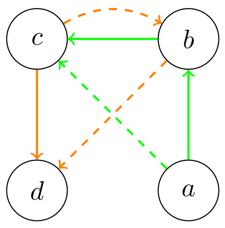


Figure 4:

both C_{p_1,q_1} and C_{p_2,q_2} to True. Thus we have:

$$\mathcal{M} \models [C_{p_1,q_1} \leftarrow \text{True}, C_{p_2,q_2} \leftarrow T]\vec{V} = \vec{v}'$$

Obviously, neither $\{p_1, q_1, ad_1\}$ nor $\{p_2, q_2, ad_2\}$ are not configurations of $ES(\vec{v}')$ because we have $p_1 \# q_1$ and $p_2 \# q_2$. So, we can claim that:

$$\mathcal{M} \models [C_{p_1,q_1} \leftarrow \text{True}, C_{p_2,q_2} \leftarrow T]PV = \text{False}$$

Which satisfies the AC2.a condition.

Now, for the AC2.b condition, we need to consider contingencies where we set C_{p_1,q_1} to False, C_{p_2,q_2} to True and resetting values of any subset of other variables to their original values. If we set C_{p_1,q_1} to False it means that we removed the conflict between p_1 and q_1, so, $\{p_1, q_1, ad_1\}$ becomes a configuration of the $ES(\vec{v})$ again. Note that value of C_{p_2,q_2} has no effect on this. Formally, this means that:

$$\mathcal{M} \models [C_{p_1,q_1} \leftarrow \text{False}, C_{p_2,q_2} \leftarrow T, \vec{Z}' \leftarrow \vec{z}^*]\vec{V} = \vec{v} \wedge (\vec{v} \in \mathcal{E} \Rightarrow PV = \text{True})$$

Thus, we can conclude that the AC2.b condition is also satisfied and since the cause is a single conjunct, the AC3 condition is satisfied too. So, we can conclude that the lack of conflict between p_1 and q_1 is an actual of the blacklist property violation.

Example 3.3.2. Loop-freedom property requires the network to not contain loop at any given moment [4]. For example, consider the network in figure 4. Initially, there exists a path from a to d. Assume that we wish to update this route to make it first visit c. Again, here we assume two updates replacing green and orange solid paths with dashed ones. We can encode this network using the following DyNetKAT terms:

$$\begin{aligned} P &= p!1 \\ Q &= q!1 \\ N &= F \oplus p?1; N_p \oplus q?1; N_q \\ N_p &= F_p \oplus q?1; F \\ N_q &= F_q \oplus p?1; F \\ SDN &= \delta_{\mathcal{L}}(N \parallel P \parallel Q) \\ \mathcal{L} &= \{p!1, p?1, q!1, q?1\} \end{aligned}$$

$$\begin{aligned} F =& a \to b \oplus a \to c \oplus a \to d \\ &\oplus b \to c \oplus b \to d \oplus c \to d \\ F_p =& a \to c \oplus a \to d \oplus c \to d \\ F_q =& a \to b \oplus a \to c \oplus a \to d \\ &\oplus b \to c \oplus b \to b \oplus b \to d \\ &\oplus c \to b \oplus c \to c \oplus c \to d \end{aligned}$$

Here we assume two concurrent processes for P and Q for updating the green and orange paths respectively. It is obvious that if $rcfg(q,1)$ happens while $rcfg(p,1)$ has not happened yet, then a loop including b and c would be created. Let E = $[\![SDN]\!]$ and $\mathcal{M}$ be the causal model of E. Here we can assume that forwarding rules of the form $x \to x$ indicate the existence of a loop in the network. So, we can define unsafe behavior in the event structure as the existence of a configuration that includes an event with a label of the form $\alpha \cdot \pi$ where $\alpha(sw) = \pi(sw)$:

$$F_{PV}(\vec{V}_{PV}) = \exists c \in \mathcal{F}(ES(\vec{v})).l(c) = \alpha \cdot \pi \Rightarrow \alpha(sw) = \pi(sw)$$

We can see that in SDN there are only two such actions: $b \to b$ and $c \to c$. Like the previous example, there are two orders of execution for $rcfg(p,1)$ and $rcfg(q,1)$ thus we need two events to represent them. So, let's consider events p_1 and p_2 with label $rcfg(p,1)$ and events q_1 and q_2 with label $rcfg(q,1)$. We also consider events bb and cc with labels $b \to b$ and $c \to c$ respectively. Figure 5 shows a part of the E where events bb and cc are reachable. In this model we can introduce $M(\{p_2\},q_2) = \text{False}$ as a cause of loop considering $(\varnothing,\varnothing,\text{True})$ as a witness. In the $\mathcal{M}$, functions of $M_{\varnothing,q_2}$ and $EN_{\varnothing,q_2}$ are defined as follows:

$$\begin{aligned}
F_{M_{\varnothing,q_2}}(\vec{V}_{M_{\varnothing,q_2}}) &= Min(\varnothing,q_2) \wedge Con(\varnothing)\\
&= Min(\varnothing,q_2)\\
&= \bigwedge_{q_2 \notin s'} \neg M_{s',q_2}\\
F_{EN_{\varnothing,q_2}}(\vec{V}_{EN_{\varnothing,q_2}}) &= M_{\varnothing,q_2}
\end{aligned}$$

Regarding the definition of these functions if we set $M_{\{p_2\},q_2}$ to True then $M_{\varnothing,q_2}$ and subsequently $EN_{\varnothing,q_2}$ becomes False. More formally we can say:

$$M \models [M_{\{p_2\},q_2} \leftarrow \text{True}]M_{\varnothing,q_2} = \text{False} \wedge EN_{\varnothing,q_2} = \text{False}$$

Thus, in $ES(\vec{v})$ under this setting, $\{q_2\}$ and all other configurations in the right branch of the figure 5 are no longer a configuration of $ES(\vec{v})$ which means that the property is no longer violated. Since we have considered an empty $\vec{W}$ in the witness and the AC2.a condition is satisfied we can conclude that $M_{\{p_2\},q_2}$ is an actual cause of a loop in this network. Intuitively $M_{\{p_2\},q_2} = \text{False}$ means p_2 not happening before q_2 is an actual cause of the loop.

Example 3.3.3. The blackhole freedom property requires that no packet being lost in the network [2]. One way to interpret this property in the DyNetKAT programs is to require that all packets entering network must exit the network eventually. For example, consider the network in figure 6 and let a and c be the network input locations and s be its output location. To enforce the blackhole freedom in this network we may require that any packet entering the network from either a or c must reach s. Again, assume that there are two updates, one for replacing bs with bd and another for replacing ds with db. We can encode

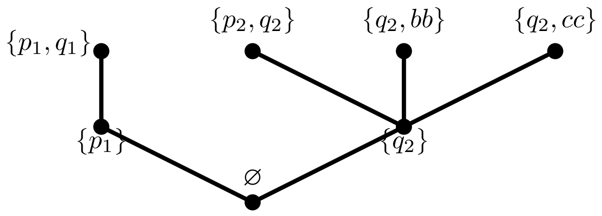


Figure 5:

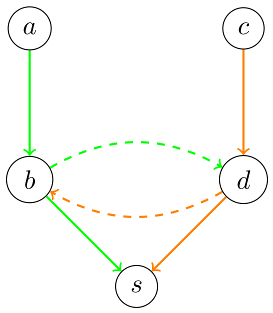


Figure 6:

this network using the following DyNetKAT terms:

$$
\begin{aligned}
P &= p!1 \\
Q &= q!1 \\
N &= F \oplus p?1; N_p \oplus q?1; N_q \\
N_p &= F \oplus q?1; F_{pq} \\
N_q &= F \oplus p?1; F_{pq}
\end{aligned}
\qquad
\begin{aligned}
F &= a \to s \oplus c \to s \\
F_{pq} &= a \to d \oplus c \to b \\
SDN &= \delta_{\mathcal{L}}(N \parallel P \parallel Q) \\
\mathcal{L} &= \{p!1, p?1, q?1, q?1\}
\end{aligned}
$$

We encoded the network such that packets are forwarded along the longest possible loop-free paths. We define the property to require that s be the destination of all forwarding actions. Thus, the $a \to d$ and $c \to b$ actions are unsafe behaviors. Again we have two events for each of the actions $rcfg(p, 1)$, $rcfg(q, 1)$, $a \to d$, and $c \to b$. So we consider the events p_1, p_2 with label $rcfg(p, 1)$, events q_1, q_2 with label $rcfg(q, 1)$, events ad_1, ad_2 with label $a \to d$ and events cb_1, cb_2 with label $c \to b$. Figure 7 shows a part of the event structure of SDN. Let $\mathcal{M}$ be the causal model of E. We can encode the unsafe behavior as follows:

$$F_{PV}(\vec{V}_{PV}) = \exists c \in \mathcal{F}(ES(\vec{v})), \exists e \in c.l(e) = \alpha \cdot \pi \wedge \pi(sw) \neq s$$

Using the same arguments as in the blacklist example we can introduce the C_{p_1,q_1} = False as a cause of blackhole using the $(C_{p_2,q_2}, \text{True}, \text{True})$ as the witness. C_{p_1,q_1} = False intuitively says that the possibility of both p_1 and q_1 happening is a cause of blackhole.

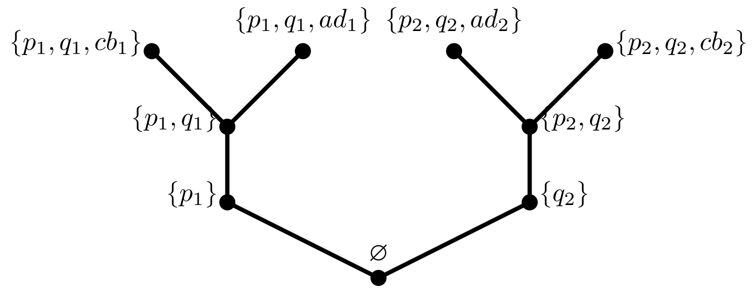


Figure 7:

References

[1] Georgiana Caltais, Hossein Hojjat, Mohammad Mousavi, and Hunkar Can Tunc. Dynetkat: An algebra of dynamic networks. *arXiv preprint arXiv:2102.10035*, 2021.

[2] Klaus-Tycho Foerster, Stefan Schmid, and Stefano Vissicchio. Survey of consistent software-defined network updates. *IEEE Communications Surveys & Tutorials*, 21(2):1435–1461, 2018.

[3] Joseph Y. Halpern and Judea Pearl. Causes and explanations: A structural-model approach, part i: Causes. *arXiv:cs/0011012*, Nov 2005. arXiv: cs/0011012.

[4] Mark Reitblatt, Nate Foster, Jennifer Rexford, Cole Schlesinger, and David Walker. Abstractions for network update. *ACM SIGCOMM Computer Communication Review*, 42(4):323–334, 2012.

[5] Glynn Winskel. *Event structures*, volume 255 of *Lecture Notes in Computer Science*, page 325–392. Springer Berlin Heidelberg, Berlin, Heidelberg, 1987.