دانشگاه تهران
پردیس دانشکده‌های فنی
دانشکده مهندسی برق و کامپیوتر

# توضیح خطا در شبکه‌های مبتنی بر نرم‌افزار با استفاده از استدلال مبتنی بر علیت

پایان‌نامه برای دریافت درجهٔ کارشناسی ارشد در رشتهٔ مهندسی کامپیوتر
گرایش نرم‌افزار

امیرحسین صیحانی

اساتید راهنما

دکتر حسین حجت و دکتر محمدرضا موسوی

شهریور ۱۴۰۱

# چکیده

**واژگان کلیدی**

# فهرست مطالب


**فصل ۱: مقدمه** ۳

**فصل ۲: تعاریف و دانش پیش‌زمینه** ۵

۱.۲ مقدمه . . . . . ۵

۲.۲ شبکه‌های مبتنی بر نرم‌افزار . . . . . ۶

۳.۲ نت‌کت . . . . . ۶

۱.۳.۲ دستور زبان نت‌کت . . . . . ۶

۲.۳.۲ معنای نت‌کت . . . . . ۶

۳.۳.۲ توصیف رفتار شبکه با نت‌کت . . . . . ۸

۴.۳.۲ درستی‌سنجی برنامه‌های نت‌کت . . . . . ۱۱

۴.۲ نت‌کت پویا . . . . . ۱۲

۱.۴.۲ دستور زبان نت‌کت پویا . . . . . ۱۲

۲.۴.۲ معنای عملیاتی نت‌کت پویا . . . . . ۱۳

۳.۴.۲ توصیف برنامه‌ها در نت‌کت پویا . . . . . ۱۵

۵.۲ ساختمان رویداد . . . . . ۱۷

۶.۲ مدل علی . . . . . ۱۹

۱.۶.۲ علت واقعی . . . . . ۲۲

۲.۶.۲ پیدا کردن علت واقعی در مسائل . . . . . ۲۳

**فصل ۳: مروری بر کارهای پیشین** ۲۷

۱.۳ مقدمه . . . . . ۲۷

**فصل ۴: روش و راه‌حل پیشنهادی** ۲۹

۱.۴ مقدمه . . . . . ۲۹

۲.۴ معنای عبارات نت‌کت پویا در قالب ساختمان رویداد . . . . . ۲۹

۱.۲.۴ ترکیب ساختمان رویدادهای برچسب‌دار . . . . . ۳۱

۲.۲.۴ معنای عبارات نت‌کت پویا در قالب ساختمان رویداد برچسب‌دار . . . . . ۳۳

۳.۴ مدل علی برای ساختمان رویداد . . . . . ۳۴

**فصل ۵: نتایج** ۳۷

۱.۵ مقدمه . . . . . ۳۷

۲.۵ آنالیز ویژگی‌های شبکه . . . . . ۳۷

۱.۲.۵ لیست سیاه . . . . . ۳۸

**فصل ۶: جمعه‌بندی و کارهای آینده** ۴۱

۱.۶ مقدمه . . . . . ۴۱

**مراجع** **اول**

**واژه‌نامهٔ فارسی به انگلیسی** **سوم**

**واژه‌نامهٔ انگلیسی به فارسی** **پنجم**

# فهرست کارهای باقیمانده

# فصل ۱

# مقدمه

# فصل ۲

# تعاریف و دانش پیش‌زمینه

## ۱.۲ مقدمه

در این فصل مفاهیم مورد نیاز و استفاده در این پروژه مورد بررسی قرار می‌گیرند. این فصل شامل ۵ بخش است. ابتدا مفاهیم کلی شبکه‌های مبتنی بر نرم‌افزار بررسی می‌شوند. سپس زبان‌های نت‌کت [۱] و نت‌کت پویا [۲] که زبان‌های مورد استفاده در این تحقیق هستند شرح داده می‌شوند. در ادامه تعاریف اولیه ساختمان‌رویداد [۳] که به عنوان معنای زبان مورد استفاده در این تحقیق استفاده می‌شود شرح داده می‌شود. در بخش آخر مدل علی [۴] که در این تحقیق برای پیدا کردن علت واقعی خطا مورد استفاده قرار می‌گیرد توصیف شده است.

[1] NetKAT
[2] DyNetKAT
[3] Event Structure
[4] Causal Model

## ۲.۲ شبکه‌های مبتنی بر نرم‌افزار

## ۳.۲ نت‌کت

نت‌کت [۵] ، یک زبان برای توصیف شبکه‌های مبتنی بر نرم‌افزار است [۱]. این زبان با وجود دستور زبان [۶] ساده‌ای که دارد، بر اساس KAT [۴] بنا شده و به همین دلیل یک سیستم معادلاتی کامل و صحیح دارد. این سیستم معادلاتی کمک می‌کند تا با استفاده از روش‌های جبری و اثبات تساوی برنامه‌های توصیف شده در این زبان، در مورد آن‌ها استدلال کرد.

### ۱.۳.۲ دستور زبان نت‌کت

در نت‌کت هر بسته [۷] به عنوان یک نگاشت از یک مجموعه از فیلدهای $f_1, f_2, ..., f_n$ به اعداد طبیعی با تعداد ارقام ثابت در نظر گرفته می‌شود. آی‌پی [۸] های مبدا و مقصد، نوع بسته، پورت [۹] های مبدا و مقصد مثال‌هایی از این فیلدها هستند. دستور زبان نت‌کت به صورت زیر تعریف می‌شود:

$$a, b ::= 1 | 0 | f = n | a + b | a \cdot b | \neg a$$

$$p, q ::= a | f \leftarrow n | p + q | p \cdot q | p^* | dup$$

### ۲.۳.۲ معنای نت‌کت

برای اینکه امکان استدلال در مورد مسیرهای طی شده توسط یک بسته در شبکه وجود داشته باشد، از مفهومی به نام تاریخچه‌ی بسته [۱۰] استفاده می‌شود. هر تاریخچه‌ی بسته، یک دنباله از بسته‌ها است که بسته نخست دنباله، به عنوان بسته‌ی فعلی در نظر گرفته می‌شود. به صورت شهودی عبارت های ۱ و ۰ به ترتیب به معنای عبور دادن

[5] NetKAT
[6] Syntax
[7] Packet
[8] IP
[9] Port
[10] Packet History

[11] و رها کردن [12] بدون شرط بسته هستند. عبارت $f = n$ در صورتی بسته را عبور می‌دهد که مقدار فیلد f آن برابر با n باشد. عبارت $f \leftarrow n$ مقدار n را به فیلد f بسته اختصاص می‌دهد. عبارت‌های $dup$ باعث می‌شوند تا یک کپی از بسته‌ی فعلی ایجاد شود و به تاریخچه‌ی بسته‌ها اضافه شود. این عبارات در رفتار شبکه تاثیری ندارند اما امکان استدلال در مورد تمامی تغییرات ایجاد شده در حین جابه‌جایی بسته در شبکه را فراهم می‌سازند. به صورت دقیق، معنای هر عبارت [13] نت‌کت با استفاده از معادلات زیر تعریف می‌شود:

$$[\![p]\!]H \in \mathcal{P}(H)$$

$$[\![1]\!]h \triangleq \{h\}$$

$$[\![0]\!]h \triangleq \{\}$$

$$[\![f = n]\!](pk :: h) \triangleq \begin{cases} \{pk :: h\} & \text{if } pk.f = n \\ \{\} & \text{otherwise} \end{cases}$$

$$[\![\neq a]\!] \triangleq \{h\} \setminus (\{a\}h)$$

$$\{f \leftarrow n\}(pk :: h) \triangleq \{pk[f := n] :: h\}$$

$$[\![p + q]\!]h \triangleq [\![p]\!]h \cup [\![q]\!]h$$

$$[\![p \cdot q]\!]h \triangleq ([\![p]\!] \bullet [\![q]\!])h$$

$$[\![p^*]\!]h \triangleq \bigcup_{i \in \mathbb{N}} F^i h$$

$$F^0 h \triangleq \{h\}$$

$$F^{i+1} h \triangleq ([\![p]\!] \bullet F^i)h$$

$$(f \bullet g)x \triangleq \bigcup \{g(y) | y \in f(x)\}$$

$$[\![dup]\!](pk :: h) \triangleq \{pk :: (pk :: h)\}$$

[11] Forward
[12] Drop
[13] Expression

نت‌کت علاوه بر اصول موضوعه‌ی KAT [۱۴] اصول موضوعه‌ی زیر را هم شامل می‌شود تا دستگاه معادلاتی صحیح و کامل [۱۵] داشته باشد:

$$f \leftarrow n \cdot f' \leftarrow n' \equiv f' \leftarrow n' \cdot f \leftarrow n, \text{ if } f \neq f' \tag{۱.۲}$$

$$f \leftarrow n \cdot f' = n' \equiv f' = n' \cdot f \leftarrow n, \text{ if } f \neq f' \tag{۲.۲}$$

$$dup \cdot f = n \equiv f = n \cdot dup \tag{۳.۲}$$

$$f \leftarrow n \cdot f = n \equiv f \leftarrow n \tag{۴.۲}$$

$$f = n \cdot f \leftarrow n \equiv f = n \tag{۵.۲}$$

$$f \leftarrow n \cdot f \leftarrow n' \equiv f \leftarrow n' \tag{۶.۲}$$

$$f = n \cdot f = n' \equiv 0, \text{ if } n \neq n' \tag{۷.۲}$$

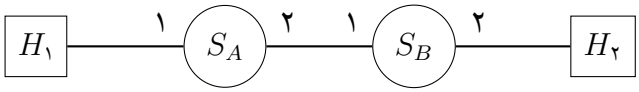
$$\Sigma_i f = i \equiv 1 \tag{۸.۲}$$

اصل‌های ۱.۲، ۲.۲، ۳.۲ خواص جابه‌جایی [۱۶] عملیات‌ها را بیان می‌کنند. اصل ۴.۲ بیان می‌کند که اختصاص مقدار n به یک فیلد و سپس این فیلتر کردن بر روی این فیلد با همین مقدار معادل با عملیات اختصاص [۱۷] به تنهایی است. مشابه همین اصل برای یک فیلتر و سپس یک اختصاص هم در اصل ۵.۲ مشخص شده. اصل ۶.۲ بیان می‌کند که در دنباله‌ای از اختصاص مقادیر به یک فیلد مشخص، تنها آخرین اختصاص تاثیر دارد. در اصل ۷.۲ مشخص شده است که مقدار یک فیلد نمی‌تواند دو مقدار متفاوت داشته باشد. در نهایت اصل ۸.۲ بیان می‌کند که عملیات مجموع فیلترها به ازای هر مقدار ممکن برای یک فیلد مشخص برابر عنصر همانی [۱۸] است.

### ۳.۳.۲ توصیف رفتار شبکه با نت‌کت

در ادامه نحوه‌ی توصیف یک شبکه با استفاده از نت‌کت بیان می‌شود. در شکل ۱.۲ این شبکه شامل دو

[14] Axiom
[15] Sound and Complete
[16] Commutative
[17] Assignment
[18] Identity




شکل ۱.۲: مثال شبکه

سوییچ [۱۹] A و B و دو هاست [۲۰] می‌باشد. هر سوییچ دو پورت دارد که با شماره‌های ۱ و۲ مشخص شده‌اند. در این شبکه هدف این است که امکان جابه‌جایی همه‌ی بسته‌ها به غیر از بسته‌هایی که از نوع SSH هستند وجود داشته باشد. عبارت نت‌کت زیر را در نظر بگیرید:

$$p \triangleq (dst = H_1 \cdot pt \leftarrow 1) + (dst = H_2 \cdot pt \leftarrow 2)$$

این عبارت همه‌ی بسته‌هایی که مقصد آن‌ها هاست ۱ باشد را به پورت ۱ و همه‌ی بسته‌هایی که مقصد آن‌ها هاست ۲ باشد را به پورت شماره‌ی ۲ می‌فرستد. این سیاست [۲۱] به سادگی رفتار سوییچ‌ها را در نت‌کت تعریف می‌کند. در ادامه می‌توان با اضافه کردن یک فیلتر به این عبارت، ویژگی دسترسی کنترل [۲۲] را به این سیاست اضافه کرد تا همه‌ی بسته‌های از نوع SSH رها شوند:

$$p_{AC} \triangleq \neg(typ = SSH) \cdot p$$

اما استفاده از عبارت بالا به تنهایی برای توصیف رفتار شبکه شکل ۱.۲ کافی نیست. برای تکمیل این عبارت لازم است تا رفتار توپولوژی [۲۳] شبکه هم به آن افزوده شود. در نت‌کت توپولوژی شبکه به عنوان یک گراف جهت‌دار در نظر گرفته می‌شود و رفتار آن در قالب اجتماع رفتار هر یک از پیوندهای [۲۴] آن توصیف می‌شود. برای شبکه‌ی

[19] Switch
[20] Host
[21] Policy
[22] Access Control
[23] Topology
[24] Link

شکل ۱.۲ می‌توان از عبارت زیر برای توصیف توپولوژی شبکه استفاده کرد:

$$t \triangleq (sw = A \cdot pt = 2 \cdot sw \leftarrow B \cdot pt \leftarrow 1)+$$
$$(sw = b \cdot pt = 1 \cdot sw \leftarrow A \cdot pt \leftarrow 2)+$$
$$(sw = b \cdot pt = 2)$$

در نت‌کت در صورتی که سیاست و توپولوژی شبکه در قالب عبارت‌هایی توصیف شده‌باشند، رفتار کل شبکه در واقع دنباله‌ای از اعمال این عبارت‌ها به صورت یکی در میان است. به عنوان مثال در شکل ۱.۲ یک بسته از هاست ۱ ابتدا توسط سوییچ A پردازش شده، سپس روی لینک بین دو سوییچ جا به جا می‌شود و در نهایت توسط سوییچ B پردازش می‌شود. در نت‌کت می‌توان این رفتار را به صورت $p_{AC} \cdot t \cdot p_{AC}$ توصیف کرد. با استفاده از همین شهود، رفتار کل شبکه را می‌توان در قالب عبارت زیر توصیف کرد:

$$(p_{AC} \cdot t)^*$$

در توصیف بالا فرض شده است که بسته‌ها می‌توانند به هر طریق ممکن وارد شبکه و از آن خارج شوند، اما این رفتار همیشه مورد قبول نیست. به عنوان مثال در شبکه شکل ۱.۲ اگر مکان‌های ورودی و خروجی شبکه را در قالب عبارت زیر توصیف کنیم:

$$e \triangleq sw = A \cdot port = 1 \vee sw = B \cdot port = 2$$

می‌توانیم رفتار انتها به انتهای [۲۵] شبکه را به شکل زیر توصیف کنیم:

$$p_{net} \triangleq e \cdot (p_{AC} \cdot t)^* e$$

در حالت کلی‌تر، نیازی به توصیف ورودی و خروجی‌های شبکه در قالب یک عبارت نیست. پس اگر فرض شود که مکان‌های ورودی شبکه توسط عبارت $in$ و مکان‌های خروجی شبکه در قالب عبارت $out$ توصیف شده باشند،

[25] End to End

رفتار یک شبکه در نت‌کت به صورت زیر تعریف می‌شود:

$$in \cdot (p \cdot t)^* \cdot out$$

که عبارت $p$ سیاست شبکه و عبارت $t$ توپولوژی شبکه است.

### ۴.۳.۲ درستی‌سنجی برنامه‌های نت‌کت

درستی‌سنجی یک شبکه و بررسی خواص آن در نت‌کت با استفاده از بررسی تساوی عبارت یک شبکه با عبارت‌های دیگر انجام می‌شود. به عنوان مثال در شبکه‌ی شکل ۱.۲ برای بررسی اینکه همه‌ی بسته‌ها با نوع SSH از هاست ۱ رها می‌شوند کافی است تا تساوی زیر را بررسی کنیم:

$$\begin{pmatrix} type = SSH \cdot sw = A \cdot pt = 1 \cdot \\ (p_{AC} \cdot t)^* \cdot \\ sw = B \cdot pt = 2 \end{pmatrix} \equiv 0$$

از طرفی برای بررسی یک خاصیت در شبکه، مثلا امکان فرستاده شدن همه‌ی بسته‌هایی که از نوع SSH نیستند از هاست ۱ به هاست ۲ می‌توان به جای بررسی تساوی دو عبارت از نامساوی $p \leq q$ استفاده کرد. این نامساوی که خلاصه شده‌ی تساوی $p+q \equiv q$ است بیان می‌کند که رفتار عبارت $p$ بخشی از رفتار عبارت $q$ است. بنابراین برای بررسی این مساله که شبکه‌ی شکل ۱.۲ بسته‌های غیر $SSH$ از هاست ۱ را عبور می‌دهد کافی است تا درستی نامعادله‌ی زیر را بررسی کرد:

$$\begin{pmatrix} \neg(type = SSH) \cdot sw = A \cdot pt = 1 \cdot \\ sw \leftarrow B \cdot pt \leftarrow 2 \end{pmatrix} \leq (p_{AC} \cdot t)^*$$

## ۴.۲ نت‌کت پویا

نک‌کت پویا [۲۶] برای رفع برخی از کاستی‌های نت‌کت ارائه شده است [۲]. به صورت دقیق‌تر نت‌کت پویا، امکان توصیف به‌روزرسانی سیاست‌های شبکه و همچنین رفتار شبکه در مقابل چندین بسته را ممکن می‌سازد.

### ۱.۴.۲ دستور زبان نت‌کت پویا

در نت‌کت پویا، از رفتار انتها به انتهای توصیف‌های شبکه در قالب عبارت‌های نت‌کت استفاده می‌شود. به همین منظور سینتکس نت‌کت پویا به صورت زیر تعریف می‌شود:

$$N ::= \mathrm{NetKAT}^{-dup}$$

$$D ::= \bot | N; D | x?N; D | x!N; D | D \parallel DD \oplus D | X$$

$$X \triangleq D$$

در سینتکس بالا $\mathrm{NetKAT}^{-dup}$ قسمتی از زبان نت‌کت است که عبارت‌های $dup$ از آن حذف شده است. عبارت‌های $dup$ در توصیف‌های نت‌کت تاثیری در رفتار یک عبارت ندارند و هدف از استفاده از آن‌ها ثبت یک اثر از هر بسته پس از پردازش توسط یکی از عناصر شبکه است و امکان استدلال بر روی رفتار شبکه را ممکن می‌سازد. با توجه به این که در نت‌کت پویا رفتار انتها به انتهای یک عبارت نت‌کت مورد استفاده است، عبارت $dup$ از دستور زبان کنار گذاشته شده است. نت‌کت‌پویا یک لیست از بسته‌های ورودی را پردازش می‌کند و یک لیست از مجموعه‌ی بسته‌های خروجی تولید می‌کند. اپراتور ترکیب متوالی [۲۷] $N; D$ باعث می‌شود که یک بسته از لیست بسته‌های ورودی توسط سیاست $N$ پردازش شود و سپس بسته‌ی توسط عبارت $D$ پردازش می‌شود. در نت‌کت پویا امکان ارتباط توسط عبارت‌هایی به شکل $x!N$ و $x?N$ توصیف می‌شوند که به ترتیب ارسال و دریافت یک عبارت نت‌کت را روی کانال $x$ توصیف می‌کنند. ترکیب موازی [۲۸] دو عبارت توسط $D \parallel D$ توصیف می‌شود. در نهایت رفتارهای غیرقطعی [۲۹] توسط عبارت‌هایی به شکل $D \oplus D$ توصیف می‌شوند.

[26] DyNetKAT
[27] Sequential Composition
[28] Parallel Composition
[29] Non-Deterministic

### ۲.۴.۲ معنای عملیاتی نت‌کت پویا

معنای عملیاتی [۳۰] نت‌کت پویا با استفاده از عبارت‌هایی به شکل $(d, H, H')$ تعریف می‌شوند که $d$ عبارت نت‌کت پویا فعلی است، $H$ لیست بسته‌هایی که در ادامه باید پردازش شوند و $H'$ لیست بسته‌هایی است که به صورت موفقیت‌آمیز توسط شبکه پردازش شده‌اند. در اینجا فرض می شود که $F = \{f_1, ..., f_n\}$ یک مجموعه از فیلدهای بسته‌ها است. یک بسته به شکل یک تابع $F \to \mathbb{N}$ توصیف می‌شود. برای یک بسته مانند $\sigma$ تساوی $\sigma(f_i) = v_i$ بیان می‌کند که مقدار فیلد $f_i$ در بسته‌ی $\sigma$ برابر با $v_i$ است. یک لیست خالی از بسته‌ها با $\langle\rangle$ نمایش داده می‌شود. اگر $l$ یک لیست از بسته‌ها باشد $e :: l$ لیستی است که حاصل از اضافه کردن بسته $\sigma$ به ابتدای لیست به دست می‌آید. برچسب هر قانون که با $\gamma$ مشخص می‌شود به صورت یکی از شکل‌های $(\sigma, \sigma'), x!q, x?q$ یا $rcfg(x, q)$ تعریف می‌شود که $rcfg(x, q)$ به معنی انجام شدن $x!q$ و $x?q$ به صورت همگام [۳۱] است. قوانین

[30] Operational Seamntic
[31] Synchronized

زیر معنای عملیاتی نت‌کت پویا را تعریف می‌کنند:

$$(cpol^{\checkmark}_{;})\frac{\sigma' \in [\![p]\!](\sigma :: \langle\rangle)}{(p;q,\sigma :: H, H') \xrightarrow{(\sigma,\sigma')} (q, H, \sigma' :: H')} \tag{۹.۲}$$

$$(cpol_{X})\frac{(p, H_0, H_1) \xrightarrow{\gamma} (p', H_0', H_1')}{(X, H_0, H_1) \xrightarrow{\gamma} (p', H_0', H_1')} X \triangleq p \tag{۱۰.۲}$$

$$(cpol_{\_\oplus})\frac{(p, H_0, H_0') \xrightarrow{\gamma} (p', H_1, H_1')}{(p \oplus q, H_0, H_0') \xrightarrow{\gamma} (p', H_1, H_1')} \tag{۱۱.۲}$$

$$(cpol_{\oplus\_})\frac{(q, H_0, H_0') \xrightarrow{\gamma} (q', H_1, H_1')}{(p \oplus q, H_0, H_0') \xrightarrow{\gamma} (p', H_1, H_1')} \tag{۱۲.۲}$$

$$(cpol_{\_\|})\frac{(p, H_0, H_0') \xrightarrow{\gamma} (p', H_1, H_1')}{(p \parallel q, H_0, H_0') \xrightarrow{\gamma} (p' \parallel q, H_1, H_1')} \tag{۱۳.۲}$$

$$(cpol_{\|\_})\frac{(q, H_0, H_0') \xrightarrow{\gamma} (q', H_1, H_1')}{(p \parallel q, H_0, H_0') \xrightarrow{\gamma} (p \parallel q', H_1, H_1')} \tag{۱۴.۲}$$

$$(cpol_{?})\frac{}{(x?p;q, H, H') \xrightarrow{x?p} (q, H, H')} \tag{۱۵.۲}$$

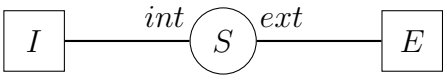
$$(cpol_{!})\frac{}{(x!p;q, H, H') \xrightarrow{x!p} (q, H, H')} \tag{۱۶.۲}$$

$$(cpol_{!?})\frac{(q, H, H') \xrightarrow{x!p} (q', H, H')(s, H, H') \xrightarrow{x?p} (s', H, H')}{(q \parallel, H, H') \xrightarrow{rcfg(x,p)} (q' \parallel s', H, H')} \tag{۱۷.۲}$$

$$(cpol_{?!})\frac{(q, H, H') \xrightarrow{x?p} (q', H, H')(s, H, H') \xrightarrow{x!p} (s', H, H')}{(q \parallel, H, H') \xrightarrow{rcfg(x,p)} (q' \parallel s', H, H')} \tag{۱۸.۲}$$

قانون ۹.۲ انجام یک عملیات مانند $(\sigma, \sigma')$ که به معنای پردازش بسته‌ی ابتدایی لیست ورودی توسط عبارت $p$ و افزودن خروجی حاصل از آن مانند $\sigma'$ به لیست خروجی است را مشخص می‌کند. قانون ۱۰.۲ بیان می‌کند که رفتار متغیر $X$ که برابر با عبارت $p$ است معادل با رفتار عبارت $p$ است. قوانین ۱۱.۲ و ۱۲.۲ رفتار غیرقطعی را توصیف می‌کنند. قوانین ۱۳.۲ و ۱۴.۲ رفتار دو عبارت موازی را توصیف می‌کنند. قوانین ۱۵.۲ و ۱۶.۲ مشخص می‌کنند که ارسال یا دریافت پیام در نت‌کت پویا پردازشی روی بسته‌ها انجام نمی‌دهد. در نهایت همگام‌سازی[۳۲] ارسال و دریافت پیام توسط قوانین ۱۵.۲ و ۱۶.۲ توصیف شده است.

[32] Synchronization




شکل ۲.۲: مثال دیوار آتش

### ۳.۴.۲ توصیف برنامه‌ها در نت‌کت پویا

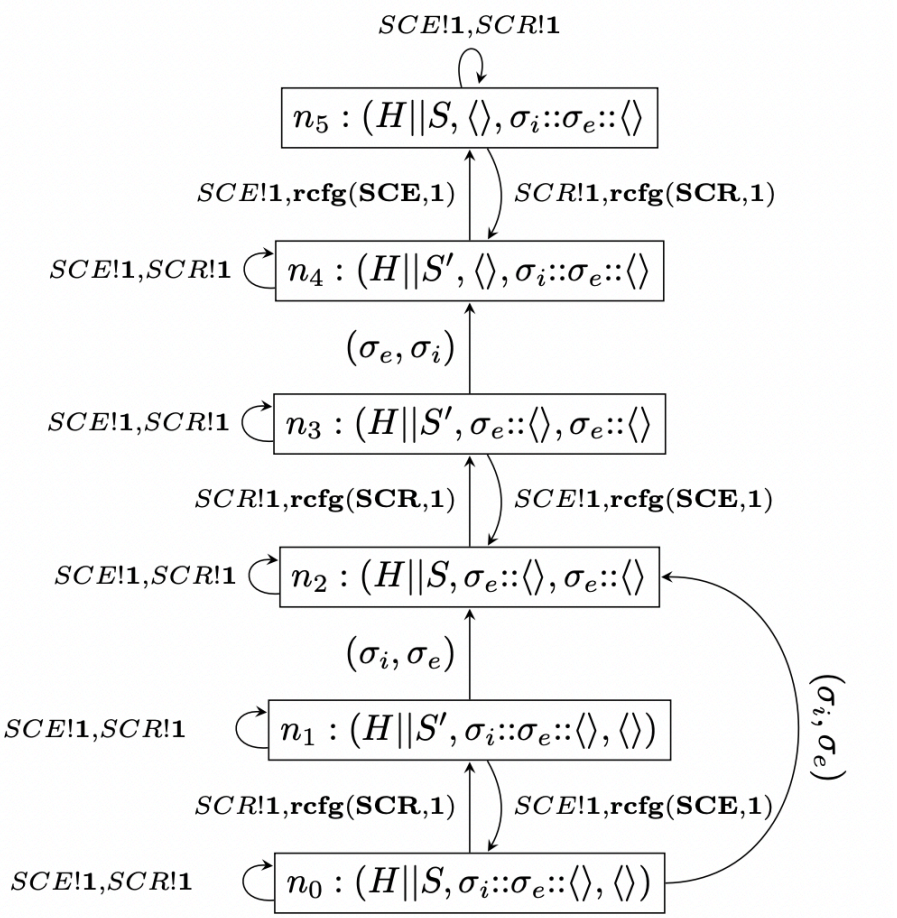
در ادامه چگونگی توصیف یک دیوار آتش [۳۳] وابسته به حالت [۳۴] با استفاده از نت‌کت پویا بیان می‌شود. شبکه‌ی شکل ۲.۲ را در نظر بگیرید. در این شبکه هدف این است که امکان ارتباط از داخل شبکه به بیرون فراهم باشد ولی امکان ارسال بسته از خارج شبکه ممکن نباشد. اما زمانی که یک بسته به خارج شبکه ارسال شد، دیوار آتش باید اجازه‌ی عبور بسته‌ها از بیرون را بدهد تا پاسخ بسته‌ها دریافت شوند. برای توصیف این شبکه می‌توان از عبارت نت‌کت پویای زیر استفاده کرد:

$$
\begin{aligned}
Host \triangleq\ & secConReq!1; Host \oplus secConEnd!1; Host \\
Switch \triangleq\ & (port = int) \cdot (port \leftarrow ext); Switch \oplus \\
& (port = ext) \cdot 0; Switch \oplus \\
& secConReq?1; Switch' \\
Switch' \triangleq\ & (port = int) \cdot (port \leftarrow ext); Switch' \oplus \\
& (port = ext) \cdot (port \leftarrow int); Switch' \oplus \\
& secConEnd?1; Switch \\
Init \triangleq\ & Host \parallel Switch
\end{aligned}
$$

در این توصیف عملیات $secConReq$ برای شروع ارتباط امن و $secConEnd$ برای خاتمه‌ی ارتباط امن در نظر گرفته شده‌است. بنابراین برنامه‌ی سوییچ پس از دریافت پیام برای شروع ارتباط امن تبدیل به برنامه‌ی $Switch'$ می‌شود که در آن اجازه‌ی ارسال بسته از پورت خارجی به پورت داخلی را دارد. پس از دریافت پیام برای خاتمه‌ی ارتباط امن، برنامه به حالت اولیه‌ی خود بر می‌گردد و دوباره تمامی بسته‌های ورودی از پورت خارجی را رها

[33] Firewall
[34] Stateful

می‌کند. در نهایت رفتار کل شبکه با استفاده از ترکیب موازی یک هاست و یک سوییچ در حالت اولیه خود توصیف می‌شود. نمودار نمایش داده شده در شکل ۳.۲ سیستم انتقال برچسب‌دار [۳۵] این شبکه را در حالتی




شکل ۳.۲: سیستم انتقال برچسب‌دار برای شبکه‌ی دیوار آتش

که یک بسته روی پورت ورودی و یک بسته روی پورت خروجی شبکه وجود دارد نشان می‌دهد. همانطور که در نمودار مشخص است، عملیات $(\sigma_e, \sigma_i)$ که به معنای ارسال بسته از پورت ورودی به پورت خروجی است تنها در قسمتی از این سیستم انتقال قابل دسترسی است که پیش از آن یکی از عملیات‌های $SCR?۱$ یا $rcfg(SCR, ۱)$ انجام شده باشند. بنابراین در این حالت شبکه تنها در صورتی که بسته خارجی را به داخل ارسال می‌کند که پیش از آن پیام آغاز ارتباط امن دریافت کرده باشد.

[35] Labeled Transition System

## ۵.۲ ساختمان رویداد

ساختمان رویداد [۳۶] [۶] یک مدل محاسباتی [۳۷] برای پردازه‌های هم‌روند [۳۸] است. ساختمان رویداد یک مدل غیر جای‌گذاری [۳۹] شده است. در این مدل، برخلاف مدل‌های جایگذاری شده مانند سیستم‌انتقال که هم‌روندی پردازه‌های موازی با انتخاب غیرقطعی مدل می‌شود، هم‌روندی پردازه‌ها به صورت صریح در مدل توصیف می‌شوند.

**تعریف ۱.۵.۲.** ساختمان رویداد یک ساختمان رویداد یک سه‌تایی $(E, \#, \vdash)$ است که در آن:

۱. $E$ یک مجموعه از رویدادها است

۲. $\#$ رابطه‌ی تعارض [۴۰] ، یک رابطه‌ی دودویی متقارن و غیربازتابی بر روی مجموعه‌ی $E$ است

۳. $\vdash \subseteq Con \times E$ رابطه‌ی فعال سازی [۴۱] است که شرط زیر را برقرار می‌کند:

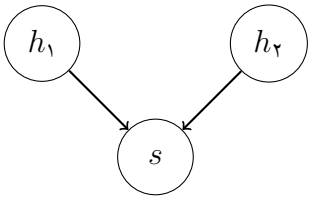
$$X \vdash e \wedge X \subseteq Y \in Con \Rightarrow Y \vdash e$$

در رابطه‌ی بالا $Con$ زیرمجموعه‌ای از مجموعه‌ی توانی رویدادها است که اعضای آن فاقد تعارض باشند. به صورت دقیق‌تر داریم:

$$Con = \{X \subseteq E | \forall e, e' \in X. \neg(e \# e')\}$$

برای مشخص کردن وضعیت یک سیتم در هر لحظه از مفهومی به نام پیکربندی [۴۲] استفاده می‌شود و و یک مجموعه شامل رویدادهایی است که تا آن لحظه در سیستم رخ داده‌اند.

[36] Event Structure
[37] Computational Model
[38] Concurrent
[39] Non-Interleaving
[40] Conflict
[41] Enabling
[42] Configuration




شکل ۴.۲

**تعریف ۲.۵.۲.** پیکربندی اگر $\mathrm{E} = (E, \#, \vdash)$ یک ساختمان رویداد باشد، یک پیکربندی آن یک زیرمجموعه از رویدادها $x \subseteq E$ است که شرایط زیر را داشته باشد:

۱. $x \in Con$

۲. $\forall e \in x \exists e_0, ..., e_n \in x. e_n = e \,\&\, \forall i \leq n. \{e_0, ..., e_{i-1}\} \vdash e_i$

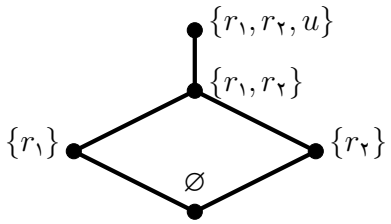
مجموعه‌ی همه‌ی پیکربندی‌های یک ساختمان رویداد مانند $\mathrm{E}$ با $\mathcal{F}(\mathrm{E})$ نمایش داده می‌شود.

شبکه‌ی موجود در شکل ۴.۲ را در نظر بگیرید. در این شبکه دو هاست ۱ و ۲ به صورت هم‌روند یک بسته را به سوییچ ارسال می‌کنند. این بسته‌ها شامل اطلاعات برای به روزرسانی مسیرهای دیگر در شبکه هستند، بنابراین سوییچ پس از دریافت هر دوی این بسته‌ها آن ها را پردازش کرده و مسیرهای خود را به روزرسانی می‌کند. برای مدل کردن این شبکه می‌توانیم از یک ساختمان رویداد به صورت زیر استفاده کنیم:

$$\mathrm{E} = (\{r_1, r_2, u\}, \varnothing, \{(\varnothing, r_1), (\varnothing, r_2), (\{r_1, r_2\}, u)\})$$

در این ساختمان رویداد، رویدادها به ترتیب دریافت یک بسته از هاست ۱، دریافت یک بسته از هاست ۲ و به روز رسانی سوییچ را مدل می‌کنند. یکی از روش‌های رسم نمودار برای ساختمان رویداد، رسم نمودار هس [۴۳] برای مجموعه‌ی پیکربندی‌های این ساختمان رویداد بر اساس رابطه‌ی زیرمجموعه است. برای مثالی که بیان شد می‌توان نموداری مطابق شکل ۵.۲ زیر را رسم کرد:

[43] Hasse




شکل ۵.۲

## ۶.۲ مدل علی

پیدا کردن تعریفی برای علت واقعی [۴۴] مبحثی است که مورد مطالعه و تحقیق بسیاری قرار گرفته است. این مساله به طور خاص در متون فلسفه مورد توجه قرار گرفته است. یکی از تعاریف علت واقعی که مورد توجه بسیاری قراری گرفته است، تعریفی مبتنی بر وابستگی خلاف واقع [۴۵] است. مطابق این تعریف، رویداد الف علت رویداد ب است اگر در شرایطی که رویداد الف اتفاق نیافته باشد، رویداد ب هم اتفاق نیافتند. در اینجا اتفاق نیفتادن رویداد الف خلاف واقع است، چون در سناریوی واقعی (سناریو ای که واقعا اتفاق افتاده و مشاهده شده است) رویداد الف اتفاق افتاده است و در نظر گرفتن شرایطی که در آن رویداد الف اتفاق نیفتاده باشد بر خلاف واقعیت موجود است. اما این مدل به تنهایی امکان پیدا کردن علت مناسب را در همه‌ی موارد ندارد. به عنوان مثال سناریوی زیر را در نظر بگیرید که در آن سارا و بهرام هر کدام یک سنگ را برداشته و به سمت یک بطری شیشه‌ای پرتاب می‌کنند. در این سناریو، سنگ سارا زودتر از سنگ بهرام به بطری برخورد کرده و در نتیجه آن را می‌شکند. در این سناریو واضح است که پرتاب سنگ توسط سارا علت شکسته شدن بطری است. فرض کنید بخواهیم از علیت مبتنی بر خلاف واقع برای پیدا کردن این علت استفاده کنیم. بنابراین باید شرایطی را در نظر بگیریم که سارا سنگ خود را پرتاب نکند. اما مشکل اینجاست که در این شرایط همچنان بطری شکسته می‌شود، چون اگر سارا سنگ خود را پرتاب نکند، بهرام همچنان سنگ خود را پرتاب می‌کند و در نتیجه این بار سنگ بهرام به بطری برخورد کرده و آن را می‌شکند. بنابراین در این سناریو امکان تعریف پرتاب سنگ سارا به عنوان علت شکسته شدن بطری با استفاده از استدلال مبتنی بر خلاف واقع وجود ندارد. هالپرن [۴۶] و پرل [۴۷] برای حل کردن مشکلاتی از این دست، تعریف جدیدی از علت واقعی [۳] ارائه کردند. مدل ارائه شده توسط آن‌ها به دلیل اینکه

[44] Actual Cause
[45] Counterfactual
[46] Halpern
[47] Pearl

بر پایه ریاضی بنا شده است امکان استفاده از آن را در آنالیز و تحلیل سیستم‌های محاسباتی فراهم می‌کند. به همین دلیل این تعریف در مقالات زیادی در حوزه‌ی دانش کامپیوتر مورد استفاده قرار گرفته است.

برای تعریف علت واقعی ابتدا برخی مفاهیم اولیه مورد استفاده در این تعریف توضیح داده می‌شوند.

به صورت کلی فرض می‌شود که دنیای مورد تحلیل توسط تعدادی متغیر تصادفی مدل شده است. اگر $X$ یک متغیر تصادفی باشد، یک رویداد به شکل $X = x$ تعریف می‌شود. برخی از این متغیرها بر روی یکدیگر تاثیر گذارند. این وابستگی‌ها در قالب مجموعه‌ای از معادلات ساختاری [۴۸] مدل می‌شوند. هر یک از این معادلات در واقع یک مکانیزم یا قانون مشخص در این دنیا را مدل می‌کنند. متغیرها به دو دسته درونی [۴۹] و برونی [۵۰] تقسیم می‌شوند. متغیرهای برونی متغیرهایی در نظر گرفته می‌شوند که مقدار آن‌ها توسط عواملی که درون مدل نیستند تعیین می‌شوند. بنابراین در مدل فرض می‌شود که مقدار این متغیرها از قبل مشخص است. اما متغیرهای درونی متغیرهایی هستند که مقدار آن‌ها بر اساس معادلات ساختاری تعیین می‌شود. به صورت دقیق‌تر، امضای [۵۱] یک مدل یک سه‌تایی $\mathcal{S} = (\mathcal{U}, \mathcal{V}, \mathcal{R})$ است که در آن $\mathcal{U}$ مجموعه‌ی متغیرهای بیرونی $\mathcal{V}$ مجموعه‌ی متغیرهای درونی و $\mathcal{R}$ دامنه‌ی مقادیر ممکن برای هر یک از متغیرها را مشخص می‌کند. در این مدل فرض می‌شود که مجموعه‌ی متغیرهای درونی محدود است. مدل علی بر روی یک امضای $\mathcal{S}$ یک دوتایی $\mathcal{M} = (\mathcal{S}, \mathcal{F})$ است که در آن $\mathcal{F}$ به هر متغیر داخلی $X \in \mathcal{V}$ یک تابع $F_X : (\times_{U \in \mathcal{U}} \mathcal{R}(U)) \times (\times_{Y \in \mathcal{V}-\{X\}} \mathcal{R}(Y)) \to \mathcal{R}(X)$ اختصاص می‌دهد. هر تابع، معادله‌ی یک متغیر را به ازای مقادیر تمام متغیرهای دیگر مشخص می‌کند. به عنوان مثال اگر فرض کنیم $F_X(Y, Z, U) = Y + U$ اگر داشته باشیم $Y = ۳, U = ۲$ آنگاه مقدار $X$ برابر ۵ خواهد شد. این معادلات امکان تفسیر آن‌ها بر اساس شرایط خلاف واقع را می‌دهند. به عنوان مثال در همین مدل اگر فرض کنیم که $U = u$ می‌توانیم نتیجه بگیریم که اگر مقدار متغیر $Y$ برابر ۴ باشد آنگاه مستقل از اینکه مقدار بقیه‌ی متغیرها در دنیای واقعی چه مقداری دارند، مقدار متغیر $X$ برابر $u + ۴$ خواهد بود که به صورت $(M, u) \models [Y \leftarrow ۴](X = u + ۴)$ نوشته می‌شود. توابع ذکر شده فقط برای متغیرهای درونی تعریف می‌شوند و همانطور که پیش‌تر اشاره شد، برای متغیرهای بیرونی تابعی تعریف نمی‌شود و فرض می‌شود که مقدار آن‌ها از قبل مشخص شده است.

**مثال ۱.۶.۲.** یک جنگل را در نظر بگیرید که می‌تواند توسط رعد و برق یا یک کبریت رها شده دچار آتش سوزی

[48] Structural Equations
[49] Endogenous
[50] Exogenous
[51] Signature

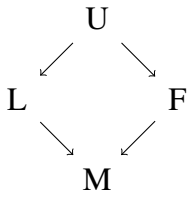
شود. برای مدل کردن این سناریو از سه متغیر بولی [۵۲] استفاده می‌کنیم:

- متغیر $F$ که اگر جنگل دچار آتش سوزی شود مقدار آن درست است و در غیر این صورت مقدار آن غلط است
- متغیر $L$ که اگر رعد و برق اتفاق افتاده باشد مقدار آن درست است و در غیر این صورت غلط است
- متغیر $M$ که اگر یک کبریت در جنگل رها شده باشد مقدار آن درست است و در غیر این صورت غلط است

در این مثال فرض می کنیم که مقادیر متغیرهای برونی به گونه‌ای است که تمام شرایط لازم برای آتش سوزی جنگل در صورتی که رعد و برق اتفاق بیافتد یا کبریتی در جنگل رها شود را دارد (به عنوان مثال درختان جنگل به اندازه‌ی کافی خشک هستند و اکسیژن کافی در هوا وجود دارد). در این مدل تابع متغیر $F$ را به گونه‌ای تعریف می‌کنیم که داشته باشیم: $F_F(\vec{u}, L, M) = L \vee M$. همانطور که پیش‌تر بیان شد، این مدل علی امکان بررسی معادلات بر اساس شرایط خلاف واقع را می‌دهد. به صورت دقیق‌تر اگر $M = (\mathcal{S}, \mathcal{F})$ یک مدل علی، $\vec{X}$ یک بردار از متغیرهای درونی و $\vec{x}, \vec{u}$ برداری از مقادیر متغیرهای $\vec{X}, \mathcal{U}$ باشند مدل $M_{\vec{X}\leftarrow\vec{x}}$ را با امضای $S_{\vec{X}}$ = $(\mathcal{U}, \mathcal{V} - \vec{X}, \mathcal{R}|_{-\vec{X}})$ یک زیرمدل [۵۳] از $M$ تعریف می‌کنیم. به صورت شهودی این مدل حاصل مداخله [۵۴] ای در مدل $M$ است که در آن مقادیر $\vec{x}$ را به متغیرهای $\vec{X}$ اختصاص داده‌ایم. به صورت دقیق‌تر تعریف می‌کنیم $M_{\vec{X}\leftarrow\vec{x}} = (\mathcal{S}_{\vec{X}}, \mathcal{F}^{\vec{X}\leftarrow\vec{x}})$ که $F_Y^{\vec{X}\leftarrow\vec{x}}$ از تابع $F_Y$ که در آن مقادیر $\vec{x}$ را به متغیرهای $\vec{X}$ اختصاص داده‌ایم به دست می‌آید. به عنوان مثال اگر $M$ مدل مثال باشد آنگاه در مدل $M_{L\leftarrow \mathrm{F}}$ معادله‌ی متغیر $F$ به $F = M$ تبدیل می‌شود. این معادله دیگر به متغیر $L$ وابسته نیست بلکه با توجه به مقدار آن که در اینجا غلط است معادله‌ی جدیدی دارد. علاوه براین توجه کنید که در مدل $M_{L\leftarrow \mathrm{F}}$ دیگر معادله‌ای برای متغیر $L$ وجود ندارد. توجه کنید که در حالت کلی ممکن است یک بردار یکتا از مقادیر متغیرها برای یک مدل وجود نداشته باشد که همزمان تمامی معادلات را حل کند. در مدل علی یک بردار از مقادیر متغیرهای برونی $\vec{u}$ یک هم‌بافت [۵۵] نامیده می‌شود. در مدل‌های بازگشتی به ازای یک هم‌بافت مشخص همیشه یک راه‌حل یکتا برای تمامی معادلات مدل وجود دارد. در ادامه فرض می‌شود که مدل‌ها بازگشتی هستند. تعمیم مدل علی برای مدل‌های غیربازگشتی در [۳] توضیح

[52] Boolean
[53] Sub-Model
[54] Intervention
[55] Context

داده است. برای یک مدل می‌توان یک شبکه‌ی علی ترسیم کرد. این شبکه یک گراف جهت‌دار است که به ازای هر متغیر یک گره در آن وجود دارد و یک یال بین دو گره وجود دارد اگر تابع متغیر دوم به متغیر اول وابسته باشد. به عنوان مثال شکل زیر شبکه‌ی علی مثال ۱.۶.۲ را نشان می‌دهد:




در ادامه برای سادگی رسم متغیرهای علی، متغیرهای برونی را از آن‌ها حذف می‌کنیم.

### ۱.۶.۲ علت واقعی

در ادامه فرمول‌های لازم برای تعریف علت واقعی توصیف می‌شوند. اگر $\mathcal{S} = (\mathcal{U}, \mathcal{V}, \mathcal{R})$ یک امضا باشد فرمول $X = x$ یک رویداد بدوی [۵۶] نامیده می‌شود که $X \in \mathcal{V}, x \in \mathcal{R}(X)$ فرمول $[Y_1 \leftarrow y_1, ..., Y_k \leftarrow y_k]\varphi$ یک فرمول علی پایه [۵۷] نامیده می‌شود که در آن:

- $\varphi$ یک ترکیب بولی از رویدادهای بدوی است
- $Y_1, ..., Y_k$ متغیرهای متمایز در $\mathcal{V}$ هستند
- $y_i \in \mathcal{R}(Y_i)$

این فرمول به صورت خلاصه به شکل $[\vec{Y} \leftarrow \vec{y}]\varphi$ نوشته می‌شود و اگر $k = 0$ باشد آنگاه به صورت $\varphi$ نوشته می‌شود. به صورت شهودی یک فرمول به شکل $[\vec{Y} \leftarrow \vec{y}]\varphi$ بیان می‌کند که در شرایط خلاف واقع ای که در آن مقادیر $\vec{y}$ به متغیرهای $\vec{Y}$ اختصاص داده شده است فرمول $\varphi$ برقرار است. یک فرمول علی به صورت یک ترکیب بولی از فرمول‌های علی پایه تعریف می‌شود. برقراری فرمول علی $\psi$ در مدل $M$ تحت هم‌بافت $\vec{u}$ را به صورت $(M, \vec{u}) \models \psi$ نشان می‌دهیم. به عنوان مثال $(M, \vec{u}) \models [\vec{Y} \leftarrow \vec{y}](X = x)$ برقرار است اگر مقدار متغیر $X$ در راه حل معادلات مدل $M_{\vec{Y} \leftarrow \vec{y}}$ تحت هم‌بافت $\vec{u}$ برابر $x$ باشد.

[56] Prime Event
[57] Basic Causal Formula

**تعریف ۲.۶.۲.** علت واقعی فرمول $\vec{X} = \vec{x}$ علت واقعی $\varphi$ (که تاثیر [۵۸] نامیده می‌شود) در $(M, \vec{u})$ اگر شرایط زیر برای آن برقرار باشد:

۱. $(M, \vec{u}) \vDash (\vec{X} = \vec{x}) \wedge \varphi$

۲. یک افراز مانند $(\vec{Z}, \vec{W})$ از مجموعه‌ی متغیرهای $\mathcal{V}$ با شرط $\vec{X} \subseteq \vec{Z}$ و مقادیر $(\vec{x}, \vec{w}')$ برای متغیرهای $(\vec{X}, \vec{W})$ وجود داشته باشد که داشته باشیم $(M, \vec{u}) \vDash \vec{Z} = \vec{z}^*$ و شرایط زیر را برآورده کند:

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

(ب) $\forall \vec{W}' \subseteq \vec{W}, \vec{Z}' \in \vec{Z}. (M, \vec{u}) \vDash [\vec{X} \leftarrow \vec{x}, \vec{W}' \leftarrow \vec{w}', \vec{Z}' \leftarrow \vec{z}^*]\varphi$

۳. $\vec{X}$ مینیمال باشد.

در این تعریف شرط اول بیان می‌کند که علت و تاثیر هر دو در شرایط واقعی برقرار هستند. شرط دوم به دنبال پیدا کردن شرایطی است که تحت آن تاثیر به صورت غیر واقع به علت وابسته باشد. این شرایط متغیرهای $\vec{W}$ و مقادیری مانند $\vec{w}'$ برای آن‌ها هستند. شرط ۲.الف بررسی می‌کند که تحت شرایطی که توسط $\vec{W} \leftarrow \vec{w}'$ به وجود می‌آید اگر علت مقداری متفاوت از مقدار خود در هم‌بافت واقعی داشته باشد اثر در مدل دیده نمی‌شود. شرط ۲.ب بررسی می‌کند که شرایط از بین رفتن اثر در ۲.الف نباشند. برای این منظور در شرایطی که علت مقدار واقعی خود را دارد در تمامی حالت‌هایی که متغیرهای شرایط می‌توانند داشته باشند بررسی می‌شود که اثر همچنان برقرار باشد. شرط سوم در واقع بیان می‌کند که زیرمجموعه‌ای از علت وجود نداشته باشد که همزمان شرایط ۱ و ۲ را برقرار کند. در تعریف بالا $(\vec{W}, \vec{w}', \vec{x}')$ یک شاهد [۵۹] بر اینکه $\vec{X} = \vec{x}$ علت $\varphi$ است تعریف می‌شود.

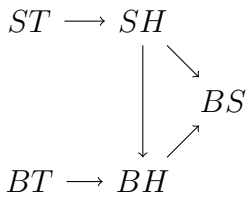
## ۲.۶.۲ پیدا کردن علت واقعی در مسائل

در ادامه مثال سارا و بهرام که در ابتدای این بخش ذکر شده بود را بررسی می‌کنیم.

برای مدل کردن این مساله متغیرهای زیر را در نظر می‌گیریم:

- $BT$: پرتاب سنگ توسط بهرام

[58] Effect
[59] Witness




شکل ۶.۲

- $BH$ برخورد سنگ بهرام به بطری
- $ST$: پرتاب سنگ توسط سارا
- $SH$: برخورد سنگ سارا به بطری
- $BS$: شکسته شدن بطری

ابتدا فرض می‌کنیم که متغیرهای $BT, ST$ تنها به متغیرهای برونی وابسته اند. بطری در صورتی شکسته می‌شود که هر یک از سنگ‌های سارا یا بهرام با آن برخورد کنند. بنابراین برای شکسته شدن بطری معادله‌ی $BS = BH \vee SH$ را در نظر می‌گیریم. نکته‌ی اصلی در این مساله این است که سنگ سارا زودتر از سنگ بهرام به شیشه برخورد می‌کند، به همین دلیل لازم است تا این موضوع در مدل لحاظ شود. یک راه برای مدل کردن این مساله این است که معادله‌ی برخورد سنگ بهرام به شیشه را به گونه‌ای تعریف کنیم که تنها در صورتی که سنگ سارا به بطری برخورد نکرده باشد آنگاه سنگ بهرام به بطری برخورد کند. بنابراین می‌توانیم معادله‌ی $BH = BT \wedge \neg SH$ را تعریف کنیم. علاوه بر این معادله‌ی برخورد سنگ سارا را بدون وابستگی به برخورد سنگ بهرام تعریف می‌کنیم: $SH = ST$. با توجه به این تعاریف برای معادلات می‌توانیم گراف علی شکل ۶.۲ را برای این مدل رسم کنیم در این مدل می‌توانیم $ST = \mathrm{T}$ را به عنوان علت $BS = \mathrm{T}$ تعریف کنیم. برای برقراری شرط ۲ در تعریف علت واقعی شرایط $\vec{W} = \{BT\}$ و $w' = \mathrm{F}$ را در نظر می‌گیریم. در این شرایط چون مقدار $BH$ برابر F می‌شود، مقدار $BS$ تنها وابسته به مقدار $SH$ و در نتیجه $ST$ می‌شود. همچنین در این مدل $BT = \mathrm{T}$ علت شکسته شدن شیشه نیست. مثلا فرض کنید که شرایط $\vec{W} = \{ST\}, w' = \mathrm{F}$ را در نظر بگیریم. در این شرایط اگر مقدار $BT$ را به F تغیر دهیم مقدار $BS$ هم غلط می‌شود. بنابراین شرط ۲.الف برقرار است. اما به ازای $\vec{Z}' = \{BH\}$ شرط ۲.ب برقرار نمی‌شود. در این حالت داریم: $(M, \vec{u}) \models [BT \leftarrow \mathrm{T}, ST \leftarrow \mathrm{F}, BH \leftarrow F]BS = \mathrm{F}$ توجه کنید با وجود اینکه مقدار درست به $BT$ اختصاص یافته اما چون مقدار $BH$ به مقدار آن در هم‌بافت واقعی

برگردانده می‌شود در نتیجه مقدار $BS$ همچنان غلط می‌ماند.

مثال بالا نشان می‌دهد که این تعریف از علت واقعی برخی از مشکلات موجود در تعاریف ساده مبتنی بر خلاف واقع را برطرف می‌کند و می‌توان توضیح مناسبی در برخی از این مثال‌ها پیدا کند. نکته‌ای که باید به آن توجه شود این است که هنوز روش یا معیاری برای این که چه تعریفی از علت واقعی تعریف مناسب است وجود ندارد. تنها روش مقایسه تعاریف مختلف استفاده از آن‌ها در مساله‌ها و سناریوهای مختلف و بررسی تطابق علت به دست آمده با استفاده از این تعریف‌ها با شهود موجود از مساله است.

# فصل ۳

# مروری بر کارهای پیشین

## ۱.۳ مقدمه

در این فصل با استفاده از مدل علی تعریف شده در فصل پیشین، علت نقض چند رسته از ویژگی‌ها در شبکه را مورد بررسی قرار می‌دهیم.

# فصل ۴

# روش و راه‌حل پیشنهادی

## ۱.۴ مقدمه

در این فصل روش پیدا کردن علت خطا در شبکه‌های نرم‌افزاری توضیح داده می‌شود. در بخش اول معنای عبارات نت‌کت پویا با استفاده از ساختمان رویداد تعریف می‌شود. در بخش دوم یک مدل علی برای توصیف ساختمان رویداد مطرح می‌شود. در نهایت بخش سوم شامل استفاده از این روش‌ها برای توضیح خطا در شبکه‌های نرم‌افزاری با استفاده از چند مثال بیان می‌شود.

## ۲.۴ معنای عبارات نت‌کت پویا در قالب ساختمان رویداد

در این بخش ابتدا شیوه‌ی اعمال چند نوع عملیات برای ترکیب ساختمان‌های رویداد تعریف می‌شود. سپس با استفاده از این عملیات‌ها معنای عبارات نت‌کت پویا توصیف می‌شود که برگرفته از [۶] می‌باشد.

**تعریف ۱.۲.۴.** محدودیت فرض کنید $\mathrm{E} = (E, \#, \vdash)$ یک ساختمان رویداد باشد. به ازای یک مجموعه‌ی $A \subseteq E$ محدودیت[1] $\mathrm{E}$ به $A$ یک ساختمان رویداد به صورت زیر است:

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

که داشته باشیم:

$$X \subseteq Con_A \iff X \subseteq A \wedge X \in Con$$
$$X \vdash_A e \iff X \subseteq A \wedge e \in A \wedge X \vdash e$$

**تعریف ۲.۲.۴.** فرض کنید $\mathrm{E} = (E, \#, \vdash)$ یک ساختمان رویداد و $a$ یک رویداد باشد. ساختمان رویداد $a\mathrm{E} = (E', \#', \vdash')$ به گونه‌ای تعریف می‌شود که داشته باشیم:

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

**تعریف ۳.۲.۴.** ساختمان رویداد برچسب‌دار یک ساختمان رویداد برچسب‌دار[2] یک پنج‌تایی به شکل $(E, \#, \vdash, L, l)$ است که در آن $(E, \#, \vdash)$ یک ساختمان رویداد است، $L$ یک مجموعه از برچسب‌ها (فاقد عنصر *) و $l$ یک تابع به فرم $l : E \to L$ است که به هر رویداد یک برچسب اختصاص می‌دهد. یک ساختمان رویداد برچسب‌دار را به اختصار به صورت $(\mathrm{E}, L, l)$ نشان می‌دهیم.

**تعریف ۴.۲.۴.** در یک ساختمان رویداد رابطه‌ی $\curlyvee$ را به صورت زیر تعریف می‌کنیم:

$$e \curlyvee e' \iff e \# e' \vee e = e'$$

[1] Restriction
[2] Labeled Event Structure

### ۱.۲.۴ ترکیب ساختمان رویدادهای برچسب‌دار

**تعریف ۵.۲.۴.** فرض کنید $(\mathrm{E}, L, l)$ یک ساختمان رویداد برچسب دار باشد و $\alpha$ یک برچسب باشد. $\alpha(\mathrm{E}, L, l)$ را به صورت یک ساختمان رویداد برچسب دار به شکل $(\alpha\mathrm{E}, L', l)$ تعریف می‌کنیم که در آن $L' = \{\alpha\} \cup L$ و به ازای هر $e' \in E'$ داریم:

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

**تعریف ۶.۲.۴.** فرض کنید $\mathrm{E}_0 = (E_0, \#_0, \vdash_0, L_0, l_0)$ و $\mathrm{E}_1 = (E_1, \#_1, \vdash_1, L_1, l_1)$ دو ساختمان رویداد برچسب‌دار باشند. مجموع این دو ساختمان رویداد $\mathrm{E}_0 + \mathrm{E}_1$ را به صورت یک ساختمان رویداد برچسب‌دار $(E, \#, \vdash, L, l)$ تعریف می‌کنیم که در آن داشته باشیم:

$$E = \{(0, e) | e \in E_0\} \cup \{(1, e) | e \in E_1\}$$

با استفاده از این مجموعه از رویدادها توابع $iota_k : E_k \to E$ را به صورت زیر و به ازای $k = 0, 1$ تعریف می‌کنیم:

$$\iota_k(e) = (k, e)$$

رابطه‌ی تعارض را به گونه‌ای تعریف می‌کنیم که داشته باشیم:

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

رابطه‌ی فعال‌سازی را به گونه‌ای تعریف می‌کنیم که داشته باشیم:

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

و مجموعه‌ی برچسب‌ها را به صورت $L = L_0 \cup L_1$ و تابع برچسب‌گذاری را به صورت زیر تعریف می‌کنیم:

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

**تعریف ۷.۲.۴.** فرض کنید که $\mathrm{E}_0 = (E_0, \#_0, \vdash_0, L_0, l_0)$ و $\mathrm{E}_1 = (E_1, \#_1, \vdash_1, L_1, l_1)$ دو ساختار رویداد برچسب‌گذاری شده باشند. حاصلضرب آن‌ها $\mathrm{E}_0 \times \mathrm{E}_1$ را به صورت یک ساختمان رویداد برچسب‌گذاری شده $\mathrm{E} = (E, \#, \vdash, L, l)$ تعریف می‌کنیم که در آن رویدادها به صورت زیر تعریف می‌شوند:

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

با توجه به این مجموعه رویدادها توابعی به شکل $\pi_i : E \to_* E_i$ تعریف می کنیم که به ازای $i = 0, 1$ داشته باشیم: $\pi_i(e_0, e_1) = e_i$. در اینجا رابطه‌ی تعارض را به کمک رابطه‌ی $\Cup$ که پیش‌تر تعریف شد، به شکل زیر به ازای تمامی رویدادهای $e, e' \in E$ توصیف می‌کنیم:

$$e \Cup e' \iff \pi_0(e) \Cup_0 \pi_0(e') \vee \pi_1(e) \Cup_1 \pi_1(e')$$

رابطه‌ی فعال‌سازی را به صورت زیر تعریف می‌کنیم:

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

مجموعه‌ی برچسب‌های حاصلضرب را به صورت زیر تعریف می‌کنیم:

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

در انتها تابع برچسب‌گذاری را به صورت زیر تعریف می‌کنیم:

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

**تعریف ۸.۲.۴.** فرض کنید که $\mathrm{E} = (E, \#, \vdash, L, l)$ یک ساختمان رویداد برچسب‌دار باشد. فرض کنید $\Lambda$ یک زیرمجموعه از $L$ باشد. محدودیت $\mathrm{E}$ به $\Lambda$ را به صورت $\mathrm{E}\Lambda$ و به شکل یک ساختمان رویداد برچسب‌گذاری شده به شکل $(E', \#', \vdash', L \cap \Lambda, l')$ که در آن $(E', \#', \vdash') = (E, \#, \vdash)\lceil\{e \in E | l(e) \in \Lambda\}$ است و تابع برچسب‌گذاری معادل محدودسازی تابع $l$ به دامنه‌ی $L \cap \Lambda$ است.

## ۲.۲.۴ معنای عبارات نت‌کت پویا در قالب ساختمان رویداد برچسب‌دار

در ادامه چگونگی تعریف معنا برای عبارت‌های نت‌کت پویا بیان می‌شود. در نت‌کت پویا رفتار عبارت‌های نت‌کت فقط به صورت انتها به انتها در نظر گرفته می‌شوند. با توجه به همین موضوع در ادامه قسمتی از نت‌کت پویا مورد استفاده قرار می‌گیرد که عبارت‌های نت‌کت به صورت نرمال در آن ظاهر می‌شوند. اگر فرض کنیم که مجموعه‌ی فیلدها $f_1, f_2, ..., f_k$ باشد یک فیلتر کامل [۳] به صورت $\alpha = f_1 = n_1 ... f_k = n_k$ و یک اختصاص کامل [۴] به صورت $\pi = f_1 \leftarrow n_1 ... f_k \leftarrow n_k$ تعریف می‌شود. می‌گوییم یک عبارت $q$ در $NetKAT^{-dup}$ به فرم نرمال است اگر به شکل $\sigma_{\alpha \cdot \pi \in \mathcal{A}} \alpha \cdot \pi$ باشد که داشته باشیم $\mathcal{A} = \{\alpha_i \cdot \pi_i | i \in I\}$. در عبارت قبل $I$ مدل

[3] Complete Test
[4] Complete Assignment

زبانی $NetKAT^{-dup}$ می‌باشد. بر اساس لم ۵ در [۲] به ازای هر عبارت $p$ در $NetKAT^{-dup}$ یک عبارت $p'$ به فرم نرمال وجود دارد که داشته باشیم: $p \equiv p'$

در ادامه از گرامر زیر برای توصیف عبارات شبکه استفاده می‌کنیم:

$$F ::= \alpha \cdot \pi$$

$$D ::= \bot | F; D | x?F; D | x!F; D | D \parallel D | D \oplus D$$

بیانگری [۵] گرامر بالا همچنان به اندازه‌ی نت‌کت پویا است. در ادامه فرض کنیم که $\mathcal{A}$ مجموعه‌ی الفبا شامل تمامی حروف به شکل $\alpha \cdot \pi, x?F, x!F$ باشد و داشته باشیم $\alpha \in \mathcal{A}$. معنای عبارت‌های بر روی زبان را به صورت زیر تعریف می‌کنیم:

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

$$[\![\alpha; t]\!] = \alpha([\![t]\!])$$

$$[\![t_1 \oplus t_2]\!] = [\![t_1]\!] + [\![t_2]\!]$$

$$[\![t_1 \parallel t_2]\!] = [\![t_1]\!] \times [\![t_2]\!]$$

$$[\![\delta_L(t)]\!] = [\![t]\!] \lceil \mathcal{A} \setminus L$$

## ۳.۴ مدل علی برای ساختمان رویداد

در ادامه نحوه‌ی توصیف یک ساختمان رویداد در قالب یک مدل علی را بیان می کنیم.

فرض کنیم که $\mathrm{E} = (E, \#, \vdash)$ یک ساختمان رویداد باشد. مدل علی این ساختمان رویداد را به صورت $\mathcal{M} = (, \mathcal{F}, \mathcal{E})$ تعریف می‌کنیم که در آن $\mathcal{S} = (\mathcal{U}, \mathcal{V}, \mathcal{R})$. در این مدل فرض می‌کنیم همه‌ی متغیرها از نوع بولی هستند. همچنین در این مدل متغیر برونی در نظر نمی‌گیریم بنابراین داریم $\mathcal{U} = \varnothing$. اگر فرض کنیم مجموعه

[5] Expressiveness

رویدادها به صورت $E = \{e_1, e_2, ..., e_n\}$ باشد مجموعه‌ی متغیرهای درونی را به صورت زیر تعریف می‌کنیم:

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

به صورت شهودی به ازای هر عضو از رابطه‌های $\#, \vdash, \vdash_{min}$ یک متغیر درونی در نظر می‌گیریم که درست بودن این متغیر به معنای وجود عنصر متناظر با آن در رابطه است. به ازای $x, y \in \mathcal{P}(E)$ پوشیده شدن [۶] $x$ توسط $y$ را که با $x \prec y$ نمایش می‌دهیم به صورت زیر تعریف می‌کنیم:

$$x \subseteq y \wedge x \neq y \wedge (\forall z . x \subseteq z \subseteq y \Rightarrow x = z \text{ or } y = z)$$

همچنین به ازای هر متغیر $X \in \mathcal{V}$ بردار $\vec{V}_X$ را بردار شامل همه‌ی متغیرهای درونی به غیر از $X$ تعریف می‌کنیم. با استفاده از این تعاریف توابع متغیرهای درونی را به صورت زیر تعریف می‌کنیم:

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

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

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

[6] Covering

که در آن‌ها داریم:

$$Con(s) = \left( \bigwedge_{1 \leq j < j' \leq n \wedge e_j, e_{j'} \in s} \neg C_{e_j, e_{j'}} \right)$$
$$Min(s, e) = \left( \bigwedge_{s' \subseteq E. (s' \subset s \vee s \subset s') \wedge e \notin s'} \neg M_{s', e} \right)$$

فرض کنید که $\mathbb{E}$ مجموعه‌ی تمامی سه‌تایی‌ها به فرم $(E, \#', \vdash')$ باشد که داشته باشیم:

$$\#' \subseteq E \times E$$
$$\vdash' \subseteq \mathcal{P} \times E$$

یک تابع به فرم $ES : \times_{V \in \mathcal{V} \setminus \{PV\}} \mathcal{R}(V) \to \mathbb{E}$ تعریف می‌کنیم که به صورت شهودی ساختمان رویداد حاصل از مقدار فعلی متغیرها در مدل علی را به دست می‌دهد. فرض کنیم $\vec{v}$ برداری شامل مقادیر متغیرهای $\mathcal{V} \setminus \{PV\}$ باشد. به ازای هر متغیر مانند $V \in \mathcal{V}$ مقدار آن در $\vec{v}$ را با $\vec{v}(V)$ نمایش می‌دهیم. تابع $ES$ را به گونه‌ای تعریف می‌کنیم که اگر $ES(\vec{v}) = (E, \#', \vdash')$ آنگاه داشته باشیم:

$$\forall e, e' \in E. e \#' e' \wedge e' \#' e \iff \vec{v}(C_{e,e'}) = \mathrm{T}$$
$$\forall s \in \mathcal{P}(E), e \in E. s \vdash' e \iff \vec{v}(EN_{s,e}) = \mathrm{T}$$

در نهایت فرض می‌کنیم که رفتار بد سیستمی که در قالب ساختمان رویداد مدل شده است، در قالب تابع متغیر $PV$ توصیف شده است و در صورتی که رفتار بد در سیستم وجود داشته باشد مقدار آن درست و در غیر این صورت غلط است. با استفاده از مدل علی که به این شکل توصیف شود برای پیدا کردن علت خطا کافی است علت واقعی $PV = \mathrm{T}$ را در مدل علی و مطابق تعریف پیدا کنیم.

# فصل ۵

# نتایج

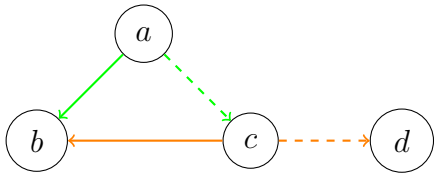
## ۱.۵ مقدمه

در این فصل با استفاده از مدل علی تعریف شده در فصل پیشین، علت نقض چند رسته از ویژگی‌ها در شبکه را مورد بررسی قرار می‌دهیم.

## ۲.۵ آنالیز ویژگی‌های شبکه

در ادامه فرض می‌کنیم که فیلد $sw$ در همه‌ی توصیف‌های نت‌کت پویا وجود دارد. همچنین برای ساده‌تر شدن توصیف‌ها از اصل زیر استفاده می‌کنیم:

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




شکل ۱.۵

### ۱.۲.۵ لیست سیاه

ویژگی لیست سیاه، یک لیست سیاه [1] از مکان‌هایی در شبکه وجود دارد که نباید در شبکه به آن‌ها دسترسی وجود داشته باشد [۵] به عنوان مثال شبکه‌ی رسم شده در شکل ۱.۵ را در نظر بگیرید. در این شبکه سوییچ $d$ در لیست سیاه قرار دارد، بنابراین در هیچ لحظه نباید از $a$ که ورودی شبکه است در دسترس باشد. در شبکه‌ی بالا ابتدا مسیرهایی که با خط پررنگ مشخص شده‌اند وجود دارند. در ادامه هر یک از مسیرها با مسیرهای خط‌چین جایگزین می‌شوند. فرض کنید به روز رسانی این مسیرها توسط دو پردازه هم‌روند انجام می‌شود. واضح است که اگر هر دوی این به‌روز رسانی‌ها انجام شوند دسترسی به سوییچی که در لیست سیاه قرار دارد ممکن می‌شود. اکنون فرض کنید که از عبارات زیر برای توصیف این شبکه در نت‌کت پویا استفاده کنیم:

$$P = p!1$$

$$Q = q!1$$

$$N = F \oplus p?1; N_p \oplus q?1; N_q$$

$$N_p = F_p \oplus q?1; F_{pq}$$

$$N_q = F_q \oplus p?1; F_{pq}$$

$$F = ab \oplus cb$$

$$F_p = ac \oplus cb \oplus ab$$

$$F_q = ab \oplus cd$$

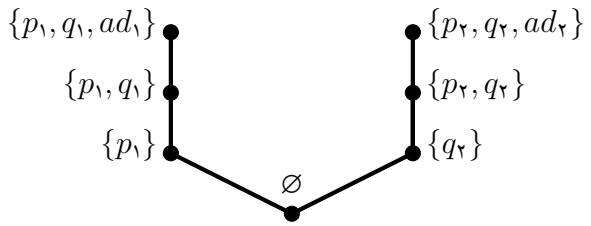
$$F_{pq} = ac \oplus cd \oplus ad$$

$$SDN = \delta_{\mathcal{L}}(N \parallel P \parallel Q)$$

$$\mathcal{L} = \{p!1, p?1, q?1, q?1\}$$

در توصیف بالا امکان اجرای هر دو به روز رسانی وجود دارد. اکنون از مدل علی این توصیف استفاده می‌کنیم تا علت خطا را در آن پیدا کنیم. برای توصیف مدل علی این شبکه لازم است تا ابتدا ویژگی را در قالب تابع متغیر

[1] Blacklist




شکل ۲.۵

$PV$ توصیف کنیم. برای این مثال تابع را به صورت زیر تعریف می‌کنیم:

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

تابع بالا رفتار نا ایمن را حالتی تعریف می‌کند که در آن امکان ارسال بسته از $a$ به $d$ وجود داشته باشد.

شکل ۲.۵ قسمتی از نمودار ساختمان رویداد این شبکه را نشان می‌دهد که در آن تمام حالت‌هایی که $ad$ قابل دسترس است وجود دارد. با استفاده از مدل علی در این مثال می‌توانیم $C(p_1, q_1) = \mathrm{F}$ را به عنوان یک علت برای نقض ویژگی لیست سیاه بیان معرفی کنیم در صورتی که از $(C(p_2, q_2), \mathrm{T}, \mathrm{T})$ به عنوان شاهد استفاده کنیم.

# فصل ۶

# جمعه‌بندی و کارهای آینده

## ۱.۶ مقدمه

در این فصل با استفاده از مدل علی تعریف شده در فصل پیشین، علت نقض چند رسته از ویژگی‌ها در شبکه را مورد بررسی قرار می‌دهیم.

# مراجع


[1] Anderson, Carolyn Jane, Foster, Nate, Guha, Arjun, Jeannin, Jean-Baptiste, Kozen, Dexter, Schlesinger, Cole, and Walker, David. Netkat: Semantic foundations for networks. *Acm sigplan notices*, 49(1):113–126, 2014. 6

[2] Caltais, Georgiana, Hojjat, Hossein, Mousavi, Mohammad, and Tunc, Hunkar Can. Dynetkat: An algebra of dynamic networks. *arXiv preprint arXiv:2102.10035*, 2021. 12, 34

[3] Halpern, Joseph Y. and Pearl, Judea. Causes and explanations: A structural-model approach, part i: Causes. *arXiv:cs/0011012*, Nov 2005. arXiv: cs/0011012. 19, 21

[4] Kozen, Dexter. Kleene algebra with tests. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 19(3):427–443, 1997. 6

[5] Reitblatt, Mark, Foster, Nate, Rexford, Jennifer, Schlesinger, Cole, and Walker, David. Abstractions for network update. *ACM SIGCOMM Computer Communication Review*, 42(4):323–334, 2012. 38

[6] Winskel, Glynn. *Event structures*, volume 255 of *Lecture Notes in Computer Science*, page 325–392. Springer Berlin Heidelberg, Berlin, Heidelberg, 1987. 17, 29

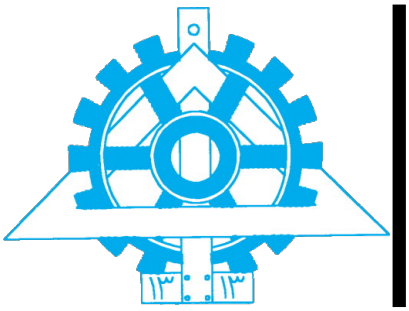
# واژه‌نامهٔ فارسی به انگلیسی

# واژه‌نامهٔ انگلیسی به فارسی

# Abstract

This thesis studies on writing projects, theses and dissertations using tehran-thesis class. It ...

**Keywords** Writing Thesis, Template, LaTeX, XeLaTeX, Persian



**University of Tehran**
**College of Engineering**
**Faculty of Electrical and Computer Engineering**

# Explaining Failures in Software-Defined Networks Using Casual Reasoning

A Thesis submitted to the Graduate Studies Office
In partial fulfillment of the requirements for
The degree of Master of Science
in Computer Engineering - Software Engineering

**By:**

**Amir Hossein Seyhani**

Supervisors:

**Dr. Hossein Hojjat and Dr. Mohammad Reza Mousavi**

Advisor:

**First Advisor**

September 2022