Chapter 3: Alarm correlation

Part II. Episodes in sequences

- Chapter 3: Alarm correlation
- Chapter 4: Frequent episodes
- Chapter 5: Minimal occurrences of episodes
- Chapter 6: Episode discovery process

3. Alarm correlation: networks and alarms

- network elements: switches, base stations, transmission equipment, etc.
- 10–1000 elements in a network
- an alarm: a message generated by a network element

  `1234 EL1 BTS 940926 082623 A1 Channel missing`

- hundreds of different alarm types
- 200 – 10000 alarms a day
- each contains only local information

Characteristics of the alarm flow

- a variety of situations
- bursts of alarms
- hardware and software change fast

Alarm correlation

"correlating" alarms: combining the fragmented information contained in the alarm sequence and interpreting the whole flow of alarms

- removing redundant alarms
- filtering out low-priority alarms
- replacing alarms by something else
- systems exist
  - knowledge base (correlation rules) constructed manually
  - look at the alarms occurring in a given time window
  - apply actions given in the matching correlation rules

Problem

- how to obtain the information needed for the preparation of an alarm correlation system
- more generally: how to obtain insight into the behavior of the network (alarms)

Solutions

- how to analyze a flow of alarms?
- lots of possibilities: hazard models, neural networks, rule-based representations
- comprehensibility of the discovered knowledge
- simple rule-based representations
- "if certain alarms occur within a time window, then a certain alarm will also occur"

Episodes

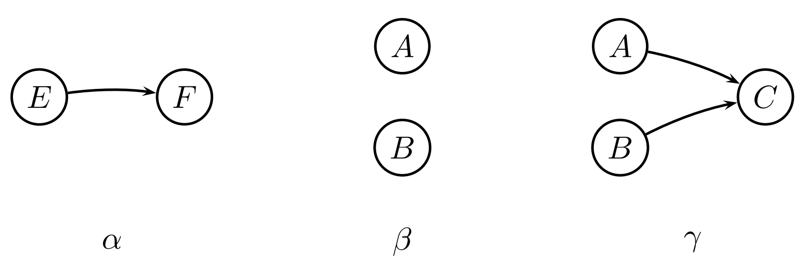

Figure 3.2: Episodes

Basic solution

- look for repeated occurrences of episodes in the alarm flow sequences
- occurrence: alarms of the specified type occur in the specified order
- why this form?
  - comprehensible
  - "standard" for correlation systems
  - represent simple causal relationships
  - insensitive to inaccurate clocks
  - allows analysis of merged, unrelated sequences

Chapter 4: Episodes

4. Frequent episodes

- The framework
- Algorithms
- Experiments

Example sequence

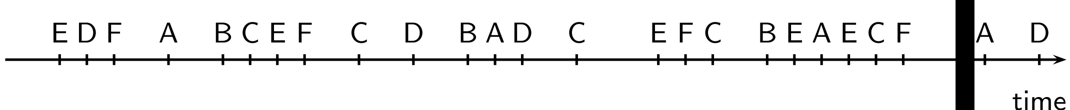

Figure 3.1: A sequence of alarms

Observations:

- whenever E occurs, F occurs soon
- whenever A and B occur (in either order), C occurs soon

Data

- a set R of *event types*
- an *event* is a pair (A, t)
- $A \in R$ is an event type
- t is an integer, the *(occurrence) time* of the event
- *event sequence* s on R: a triple (s, T_s, T_e)
- $T_s < T_e$ are integer (starting and ending time)
- $s = \langle (A_1, t_1), (A_2, t_2), \ldots, (A_n, t_n) \rangle$
- $A_i \in R$ and $T_s \leq t_i < T_e$ for all $i = 1, \ldots, n$
- $t_i \leq t_{i+1}$ for all $i = 1, \ldots, n-1$

Example

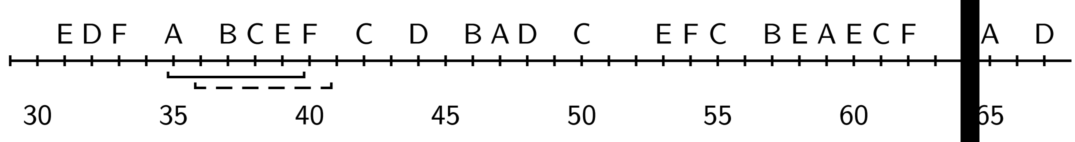

Figure 4.1: The example event sequence s and two windows of width 5.

Windows

- event sequence $\mathbf{s} = (s, T_s, T_e)$
- a window on it: $\mathbf{w} = (w, t_s, t_e)$
- $t_s < T_e, t_e > T_s$
- w consists of those pairs (A, t) from s where $t_s \leq t < t_e$
- $width(\mathbf{w}) = t_e - t_s$: the *width* of the window $\mathbf{w}$
- $\mathcal{W}(\mathbf{s}, \textit{win})$: all windows $\mathbf{w}$ on $\mathbf{s}$ such that $width(\mathbf{w}) = \textit{win}$
- first and last windows!

Episodes

- an *episode* α is a triple $(V, \leq, g)$
- V is a set of nodes
- $\leq$ is a partial order on V
- $g : V \rightarrow R$ is a mapping associating each node with an event type
- intuition: the events in $g(V)$ have to occur in the order described by $\leq$
- *size* of α, denoted $|\alpha|$, is $|V|$
- *parallel episode*: the partial order $\leq$ is trivial
- *serial episode*: $\leq$ is a total order
- *injective*: no event type occurs twice in the episode

Example

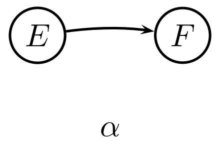

Figure 4.2: An episode

the set V, the mapping g

Example, subepisode

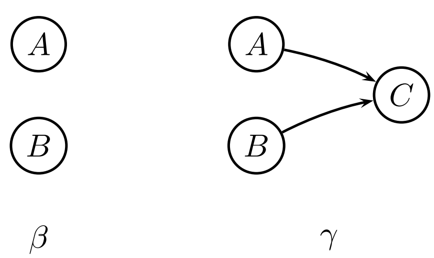

Figure 4.3: A subepisode and episode

Subepisodes

$\beta = (V', \leq', g')$ is a *subepisode* of $\alpha = (V, \leq, g)$, $\beta \preceq \alpha$, if:

there exists an injective mapping $f : V' \to V$ such that

- $g'(v) = g(f(v))$ for all $v \in V'$
- for all $v, w \in V'$ with $v \leq' w$ also $f(v) \leq f(w)$

An episode α is a *superepisode* of β if and only if $\beta \preceq \alpha$

$\beta \prec \alpha$ if $\beta \preceq \alpha$ and $\alpha \not\preceq \beta$

In the example: $\beta \preceq \gamma$

Occurrences of episodes

$\alpha = (V, \leq, g)$ *occurs* in an event sequence
$\mathbf{s} = (\langle (A_1, t_1), (A_2, t_2), \ldots, (A_n, t_n) \rangle, T_s, T_e)$, if there exists an injective mapping $h : V \to \{1, \ldots, n\}$ from nodes to events, such that

- $g(x) = A_{h(x)}$ for all $x \in V$
- for all $x, y \in V$ with $x \neq y$ and $x \leq y$ we have $t_{h(x)} < t_{h(y)}$ (or $h(x) < h(y)$)

$(w, 35, 40)$ on the example sequence: events of types A, B, C, and E

both β and γ occur

Frequency of occurrence

- the *frequency* of an episode α in $\mathbf{s}$ is

$$fr(\alpha, \mathbf{s}, win) = \frac{|\{\mathbf{w} \in \mathcal{W}(\mathbf{s}, win) \mid \alpha \text{ occurs in } \mathbf{w}\}|}{|\mathcal{W}(\mathbf{s}, win)|},$$

- i.e., the fraction of windows on $\mathbf{s}$ in which α occurs.
- a *frequency threshold min_fr*
- α is *frequent* if $fr(\alpha, \mathbf{s}, win) \geq min\_fr$
- $\mathcal{F}(\mathbf{s}, win, min\_fr)$: collection of frequent episodes in $\mathbf{s}$ with respect to *win* and *min_fr*
- size $= l$: $\mathcal{F}_l(\mathbf{s}, win, min\_fr)$.

Pattern discovery task

given an event sequence $\mathbf{s}$, a set $\mathcal{E}$ of episodes, a window width *win*, and a frequency threshold *min_fr*, find $\mathcal{F}(\mathbf{s}, win, min\_fr)$

Algorithms

Algorithm 4.13

Input: A set R of event types, an event sequence $\mathbf{s}$ over R, a set $\mathcal{E}$ of episodes, a window width *win*, and a frequency threshold *min_fr*.

Output: The collection $\mathcal{F}(\mathbf{s}, \mathit{win}, \mathit{min\_fr})$ of frequent episodes.

Method:

1. compute $\mathcal{C}_1 := \{\alpha \in \mathcal{E} \mid |\alpha| = 1;\}$
2. $l := 1;$
3. **while** $\mathcal{C}_l \neq \emptyset$ **do**
4. // Database pass (Algorithms 4.19 and 4.21):
5. compute $\mathcal{F}_l(\mathbf{s}, \mathit{win}, \mathit{min\_fr}) := \{\alpha \in \mathcal{C}_l \mid \mathit{fr}(\alpha, \mathbf{s}, \mathit{win}) \geq \mathit{min\_fr}\};$
6. $l := l + 1;$
7. // Candidate generation (Algorithm 4.14):
8. compute $\mathcal{C}_l := \{\alpha \in \mathcal{E} \mid |\alpha| = l,$ and $\beta \in \mathcal{F}_{|\beta|}(\mathbf{s}, \mathit{win}, \mathit{min\_fr})$ for all $\beta \in \mathcal{E}$ such that $\beta \prec \alpha$ and $|\beta| < l\};$
9. **for** all l **do** output $\mathcal{F}_l(\mathbf{s}, \mathit{win}, \mathit{min\_fr});$

Basic lemma, once again

Lemma 4.12 If an episode α is frequent in an event sequence $\mathbf{s}$, then all subepisodes $\beta \preceq \alpha$ are frequent. $\square$

Parallel, serial, injective episodes

- *parallel episode*: the partial order $\leq$ is trivial
  (= frequent sets)
- *serial episode*: $\leq$ is a total order
  (= frequent subsequence)
- *injective*: no event type occurs twice in the episode (= proper sets, not multi sets)
- useful cases: (serial or parallel) [injective] episodes
  - reduce redundancy in generated episodes
  - keep episodes comprehensible
  - simpler to implement

Generation of candidate episodes

- parallel episodes, serial episodes (injective or non-injective)
- same idea as for association rules
- a candidate episode has to be a combination of two episodes of smaller size
- very small variations to the candidate generation procedure

Recognizing episodes in sequences

- first problem: given a sequence and an episode, find out whether the episode occurs in the sequence
- finding the number of windows containing an occurrence of the episode can be reduced to this
- successive windows have a lot in common
- how to use this?
- an incremental algorithm

Parallel episodes

- for each candidate α maintain a counter α.*event_count*: how many events of α are present in the window
- When α.*event_count* becomes equal to $|\alpha|$, indicating that α is entirely included in the window
  - save the starting time of the window in α.*inwindow*
- when α.*event_count* decreases again, increase the field α.*freq_count* by the number of windows where α remained entirely in the window

Algorithm

Input: A collection $\mathcal{C}$ of parallel episodes, an event sequence $\mathbf{s} = (s, T_s, T_e)$, a window width *win*, and a frequency threshold *min_fr*.

Output: The episodes of $\mathcal{C}$ that are frequent in $\mathbf{s}$ with respect to *win* and *min_fr*.

Method:

// Initialization:
for each α in $\mathcal{C}$ **do**
    for each A in α **do**
        $A.\mathit{count} := 0$;
        for $i := 1$ **to** $|\alpha|$ **do** $contains(A, i) := \emptyset$;
for each α in $\mathcal{C}$ **do**
    for each A in α **do**
        $a :=$ number of events of type A in α ;
        $contains(A, a) := contains(A, a) \cup \{\alpha\}$;
    $\alpha.\mathit{event\_count} := 0$;
    $\alpha.\mathit{freq\_count} := 0$;

Algorithm Method:

// Recognition:
for $start := T_s - \mathit{win} + 1$ **to** T_e **do**
    // Bring in new events to the window:
    for all events (A, t) in s such that $t = start + \mathit{win} - 1$ **do**
        $A.\mathit{count} := A.\mathit{count} + 1$;
        for each $\alpha \in$ contains($A, A.\mathit{count}$) **do**
            $\alpha.\mathit{event\_count} := \alpha.\mathit{event\_count} + A.\mathit{count}$;
            if $\alpha.\mathit{event\_count} = |\alpha|$ **then** $\alpha.\mathit{inwindow} := start$;
    // Drop out old events from the window:
    for all events (A, t) in s such that $t = start - 1$ **do**
        for each $\alpha \in$ contains($A, A.\mathit{count}$) **do**
            if $\alpha.\mathit{event\_count} = |\alpha|$ **then**
                $\alpha.\mathit{freq\_count} := \alpha.\mathit{freq\_count} - \alpha.\mathit{inwindow} + start$;
            $\alpha.\mathit{event\_count} := \alpha.\mathit{event\_count} - A.\mathit{count}$;
        $A.\mathit{count} := A.\mathit{count} - 1$;
// Output:
for all episodes α in $\mathcal{C}$ **do**
    if $\alpha.\mathit{freq\_count}/(T_e - T_s + \mathit{win} - 1) \geq \mathit{min\_fr}$ **then** output α;

Theorem 1 *Algorithm 102 works correctly.*

Proof We consider the following two invariants. (1) For each event type A that occurs in any episode, the variable $A.\mathit{count}$ correctly contains the number of events of type A in the current window. (2) For each episode α, the counter $\alpha.\mathit{event\_count}$ equals $|\alpha|$ exactly when α occurs in the current window. □

Complexity

Assume that exactly one event takes place every time unit.

Assume candidate episodes are all of size l, and let n be the length of the sequence.

Theorem 2 *The time complexity of Algorithm 102 is* $\mathcal{O}((n + l^2)|\mathcal{C}|)$.

Proof Initialization takes time $\mathcal{O}(|\mathcal{C}|l^2)$.

How many accesses to $\alpha.\mathit{event\_count}$ on lines 7 and 14.

In the recognition phase there are $\mathcal{O}(n)$ shifts of the window. In each shift, one new event comes into the window, and one old event leaves the window. Thus, for any episode α, $\alpha.\mathit{event\_count}$ is accessed at most twice during one shift.

The cost of the recognition phase is thus $\mathcal{O}(n|\mathcal{C}|)$. □

Serial episodes

- use state automata that accept the candidate episodes
- example: episode A B A B

General episodes

different alternatives

Window width (s)	Serial episodes Count	Serial episodes Time (s)	Injective parallel episodes Count	Injective parallel episodes Time (s)
10	16	31	10	8
20	31	63	17	9
40	57	117	33	14
60	87	186	56	15
80	145	271	95	21
100	245	372	139	21
120	359	478	189	22

Table 4.1: Results of experiments with s_1 using a fixed frequency threshold of 0.003 and a varying window width

Frequency threshold	Serial episodes Count	Serial episodes Time (s)	Injective parallel episodes Count	Injective parallel episodes Time (s)
0.1	0	7	0	5
0.05	1	12	1	5
0.008	30	62	19	14
0.004	60	100	40	15
0.002	150	407	93	22
0.001	357	490	185	22

Table 4.2: Results of experiments with $\mathbf{s}_1$ using a fixed window width of 60 s and a varying frequency threshold

Episode size	Number of episodes	Number of candidate episodes	Number of frequent episodes	Match
1	287	287.0	30.1	11 %
2	82 369	1 078.7	44.6	4 %
3	$2 \cdot 10^7$	192.4	20.0	10 %
4	$7 \cdot 10^9$	17.4	10.1	58 %
5	$2 \cdot 10^{12}$	7.1	5.3	74 %
6	$6 \cdot 10^{14}$	4.7	2.9	61 %
7	$2 \cdot 10^{17}$	2.9	2.1	75 %
8	$5 \cdot 10^{19}$	2.1	1.7	80 %
9	$1 \cdot 10^{22}$	1.7	1.4	83 %
10–		17.4	16.0	92 %

Table 4.3: Number of candidate and frequent serial episodes in $\mathbf{s}_1$ with frequency threshold 0.003 and averaged over window widths 10, 20, 40, 60, 80, 100, and 120 s

Experiences in alarm correlation

Useful in

- finding long-term, rather frequently occurring dependencies,
- creating an overview of a short-term alarm sequence, and
- evaluating the consistency and correctness of alarm databases
- discovered rules have been applied in alarm correlation
- lots of rules are trivial

Chapter 5: Minimal occurrences of episodes

5. Minimal occurrences of episodes

- an alternative approach to discovery of episodes
- no windows
- for each potentially interesting episode, find out the exact occurrences of the episode
- advantages: easy to modify time limits, several time limits for one rule ("if A and B occur within 15 seconds, then C follows within 30 seconds")
- disadvantages: uses lots of space

Definitions

- an episode α and an event sequence $\mathbf{s}$
- interval $[t_s, t_e)$ is a *minimal occurrence* of α in $\mathbf{s}$, if
  - α occurs in the window $\mathbf{w} = (w, t_s, t_e)$ on $\mathbf{s}$
  - α does not occur in any proper subwindow on $\mathbf{w}$
- *set of (intervals of) minimal occurrences* of an episode α:
  mo$(\alpha) = \{ [t_s, t_e) \mid [t_s, t_e)$ is a minimal occurrence of $\alpha\}$.

Example

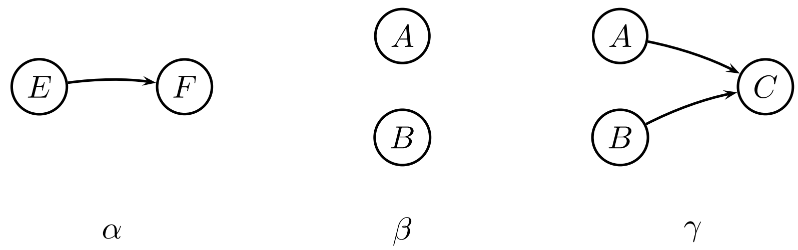

Figure 1: Episodes.

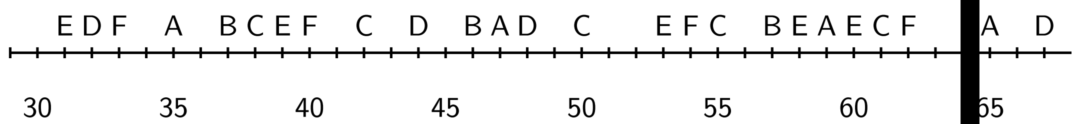

Figure 2: The example event sequence s.

β consisting of event types A and B has four minimal occurrences in s: *mo*$(\beta) = \{[35, 38), [46, 48), [47, 58), [57, 60)\}$.

The partially ordered episode γ has the following three minimal occurrences: $[35, 39), [46, 51), [57, 62)$.

Episodes rules, new version

- *episode rule*: $\beta\,[\mathit{win}_1] \;\Rightarrow\; \alpha\,[\mathit{win}_2]$,
- β and α are episodes such that $\beta \preceq \alpha$
- win_1 and win_2 are integers
- if episode β has a minimal occurrence at interval $[t_s, t_e)$ with $t_e - t_s \leq \mathit{win}_1$, then episode α occurs at interval $[t_s, t'_e)$ for some t'_e such that $t'_e - t_s \leq \mathit{win}_2$

- formally: $\mathit{mo}_{\mathit{win}_1}(\beta) = \{[t_s, t_e) \in \mathit{mo}(\beta) \mid t_e - t_s \leq \mathit{win}_1\}$
- given α and an interval $[u_s, u_e)$, define $occ(\alpha, [u_s, u_e)) =$ true if and only if there exists a minimal occurrence $[u'_s, u'_e) \in \mathit{mo}(\alpha)$ such that $u_s \leq u'_s$ and $u'_e \leq u_e$
- The confidence of an episode rule $\beta\,[\mathit{win}_1] \;\Rightarrow\; \alpha\,[\mathit{win}_2]$ is now

$$\frac{|\{[t_s, t_e) \in \mathit{mo}_{\mathit{win}_1}(\beta) \mid occ(\alpha, [t_s, t_s + \mathit{win}_2))\}|}{|\mathit{mo}_{\mathit{win}_1}(\beta)|}.$$

Example, cont.

- $\beta\,[3] \Rightarrow \gamma\,[4]$
- three minimal occurrences $[35, 38), [46, 48), [57, 60)$ of β of width at most 3 in the denominator
- Only $[35, 38)$, has an occurrence of α within width 4, so the confidence is $1/3$.
- rule $\beta\,[3] \Rightarrow \gamma\,[5]$ the confidence is 1.

Rule forms

- temporal relationships can be complex

Frequency and support

- previously: frequency = fraction of windows containing the episode
- no fixed window size
- several minimal occurrences within a window
- support of an episode: the number of minimal occurrences of an episode, $|mo(\alpha)|$

Rule discovery task

- an event sequence $\mathbf{s}$
- a class $\mathcal{E}$ of episodes
- a set W of time bounds
- find all frequent episode rules of the form $\beta\,[win_1] \Rightarrow \alpha\,[win_2]$
- $\beta, \alpha \in \mathcal{E}$ and $win_1, win_2 \in W$.