Compass- Complex Event Processing Enabled By State Space Transformations

Deepak K Gangadhar
IBM, New Delhi
India
deepakg@in.ibm.com

ABSTRACT: *In a world inundated with information, it's becoming increasingly difficult to filter out the unwanted, retain what is needed, and finally garner insight from it. It's not just information in the 'traditional' sense that is adding to this deluge, but with the proliferation of affordable and versatile sensors, information chunks in the form of discrete events are pouring in from places hitherto unthinkable... or unreachable. Compass focuses on taming this fire-hose towards providing a smarter way to analyze and process events. Compass takes the n-dimensional event space through a series of transformation which makes the event space more amenable for analysis. Firstly, it aggregates the discrete events across time and space along the component axis. Secondly, it reduces the dimensionality of the state space by partitioning the state space into several spatio-temporal Regions of Interests (RIs). The RIs are composed of one or more Space- Time Elements (STEs), which are the lowest level units monitoring the incoming events. These RIs themselves cycle through various states like 'hot', 'warn', 'alert', which will enable remedial actions to be triggered. With these two transformations, Compass paves the way for 'emergent' behavior; where the system can be analyzed at macro-level states by making use of the micro-level events.*

Keywords: Internet of things, sensor networks, state space transformation, complex event processing, event stream processing, emergence.

Received: 12 May 2016, Revised 18 June 2016, Accepted 25 June 2016

1. Introduction

We are swamped with information and don't know what to do with it. There are smart devices all around us, all of which are screaming for our attention by sending out important events. Gartner reports that the Internet of Things (IOT) will have close to 26 billion devices connected to it by 2020 [1]. Add to it the smartphones and other tablets and the number reaches a mindboggling 50 billion [2]. All these connected devices are going to continuously generate vents of various types continuously [3]. Even a simple heart-beat signal generated by these devices indicating that they are active and alive will generate a deluge of information, processing of which will be a daunting task to even the most well-configured server by today's standards. To get even a semblance of order with this huge number of devices on the network will need radically new monitoring and administering systems. There is a dire necessity for techniques to filter, correlate and analyze a large number of events in near real-time to arrive at fast decisions. *Compass* is a Complex Event Processing (CEP) technique to address these kinds of problems.

2. What Is Compass?

Compass models the system under study as a dynamical system evolving in an n-dimensional event space. It takes this event space through multiple transformations to isolate critical events from the deluge of events emanating from the system. Like shown in Fig. 1, there are 2 major transformations.

$$F: R_r \rightarrow R_a$$
$$G: R_a \rightarrow R_p$$

where -

R_r is the raw event space, R_a is the aggregated event space, R_p is the partitioned event space.

F is the aggregation function and G is the partition function.

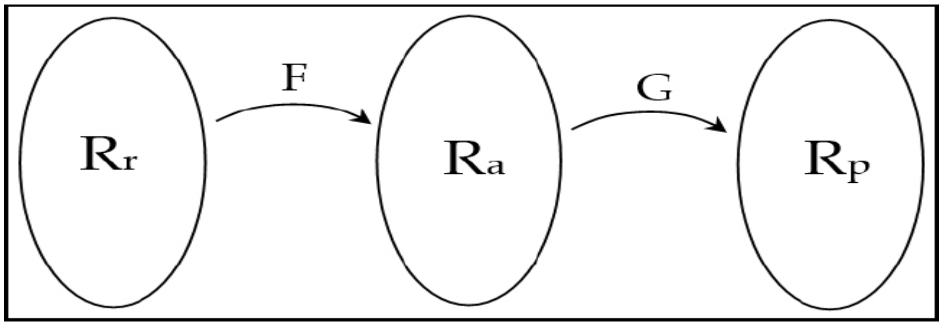


Figure 1. Event space transformations

The aggregation function F works on the n-dimensional event space at two levels:

1. It slices the space along the time axis at periodic intervals (called the Aggregation Time) and collects all the events occurring during that time window.

2. It then aggregates the discrete events belonging to the same component and embeds the components within the event space.

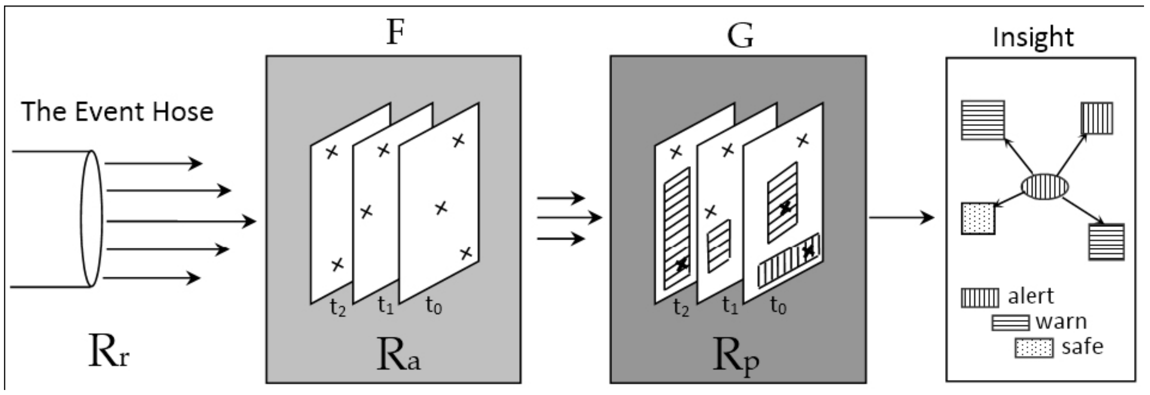


Figure 2. Flow of events in Compass

This transformed space is now passed on to the partition function G, which partitions the event space into sections called *Regions of Interest* (RI). These sections form a small portion of the larger state space and hence monitoring the events within this small space is faster and easier. These sections are created based on a set of conditions that we are interested in monitoring. These sections are attached to state machines which get triggered based on the component and the event states. Any RI that has been triggered by the presence of one or more components is termed as *HOT.* Fig. 2 shows a symbolic flow of events as they first get aggregated and then move through the partitioned state space to finally generate insights.

Alternately, the transformations can be visualized as a set of filters that will split the event stream into two separate streams like shown in Figure 3. Along with these two transformations, the global event set gets split into 2 streams.

E_c is the set of events that are filtered by the Hot RIs. These are the correlated events.

E_r is the set of events that flow undeterred by the filters. These are the raw events.

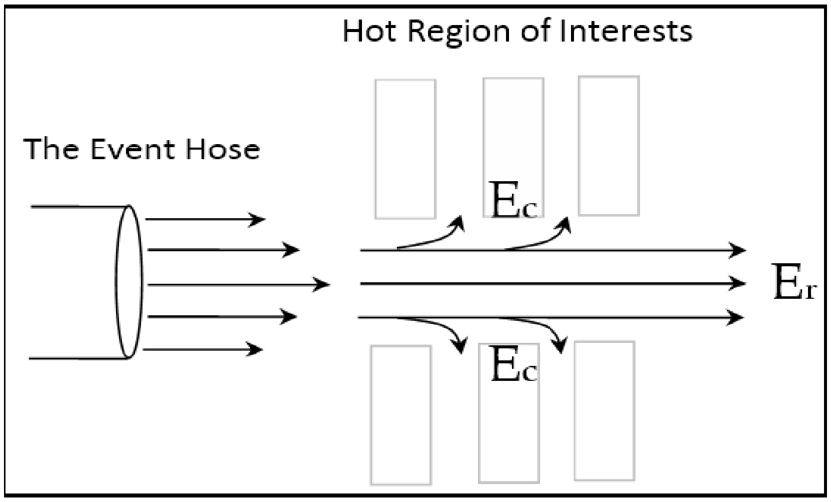


Figure 3. Regions of Interest act as filters

E_c is the set of events for which there are other correlated events manifested as filters, as explained in Section 6. This set is comprised of only those events that are critical. All the other events get passed on to the generic set. As the set become leaner, processing the events gets faster.

3. Modelling The Event Space

Before we take a look at the transformation functions in more detail, let us quickly take a look at how a typical system with multiple components and events is modeled. We will look at a Connected Car – an Internet Of Things scenario. Our simplified model has 4 subsystems generating the specific events as shown below.

Wheels (Component C_1)

Pressure (Event e_1), Temperature (Event e_2)

Engine (Component C_2)

RPM (Event e_3), Pressure (Event e_4)

Camshaft (Component C_3)

Temperature (Event e_5), Speed (Event e_6)

Air Intake (Component C_4)

Temperature (Event e_7), Moisture (Event e_8)

Compass views this as a dynamical system evolving in a 8- dimensional space. At any given point in time, the system can be defined exactly by using the values of the 8 events, or as described later, in a 4-dimensional space by using the macrostates of the 4 components.

Generalizing, any system can be modeled as a space with n dimensions. Each dimension represented by the events.

$E^n = \{ei\}$ where i=1,2,3...n and E^n is the n dimensional event space.

The n dimension space can also be grouped by component states. The pre-requisite is that each of the components should be classifiable into states based on the different values of the events comprising the component. We can define higher level states on the AirIntake component as a complex combination of one or more events along with the time dimension. For eg, state A1 can be defined as

A1 == (Temperature > 45 AND Moisture > 20)

$E^m = \{c_j\}$ where j=1,2,3...m and E^m is the m dimensional component space

4. Aggregation Function (F)

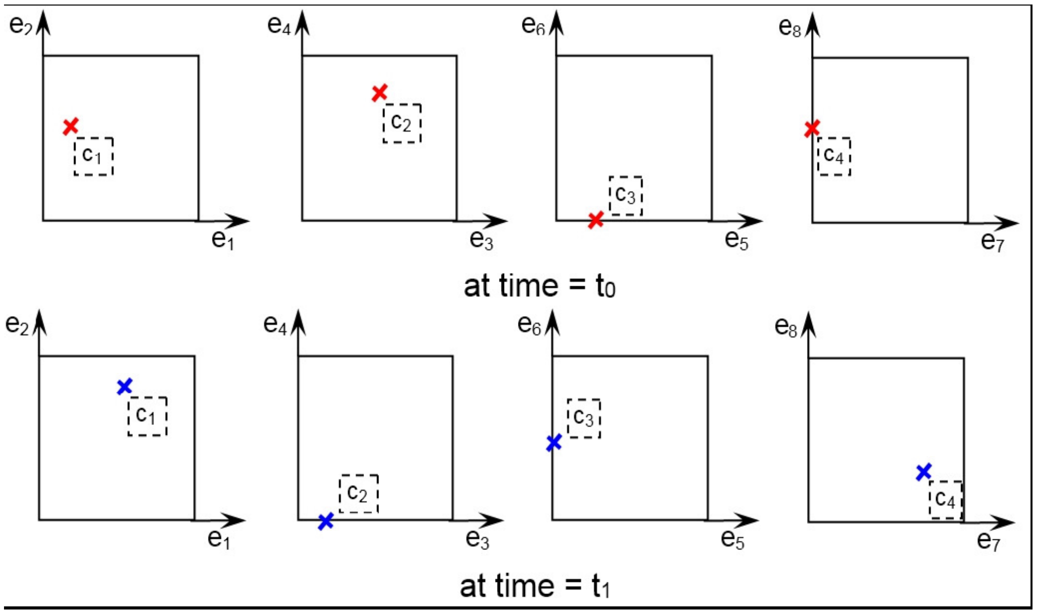


Figure 4. Event aggregation: Embedding components in the n-d space

As the events keep pouring in, rather than deal with discrete individual events, the aggregation function makes use of the knowledge of the components present in the system and the events that each of them generate. When a snapshot of the system is taken at regular intervals of time, F groups all events by components to obtain a component level view. This time interval is called the ‘Aggregation Time’. At the end of the transformation, the event space has the components embedded in an n-dimensional space. Let us look at an illustration referring to the model discussed in Section 3. For ease of visualization, we look at the component level view using 2 axes at a time as shown in Fig. 4.

At time t_0, 7 events are generated like shown below:

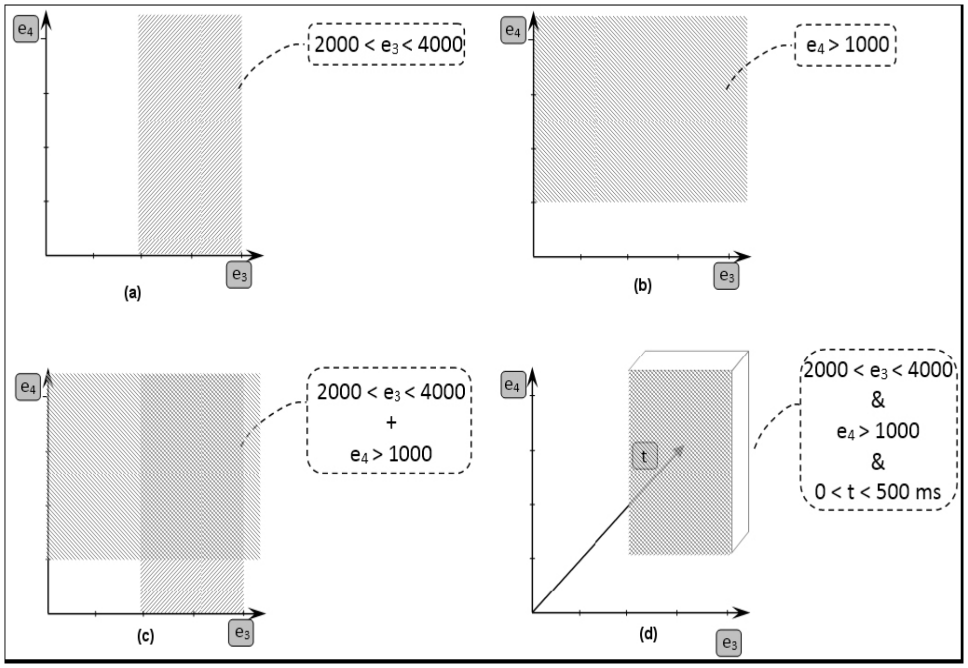
e1 (generated by c1), e3 (generated by c2)
e2 (generated by c1), e4 (generated by c2)
e1 (generated by c1), e5 (generated by c3)
e8 (generated by c4)

Notice that e1 is generated twice in this time slice and hence the latter value is retained. After aggregation at a component level, the event data is pivoted about the component axis, and we obtain the following component level view:

c1(e1,e2), c2(e3,e4), c3(e5), c4(e8)

Fig. 4 also shows how the system will look after the aggregation of the next time slice.

5. Partition Function (G)



e_4
$2000 < e_3 < 4000$
e_3
(a)
e_4
$e_4 > 1000$
e_3
(b)
e_4
$2000 < e_3 < 4000$ + $e_4 > 1000$
e_3
(c)
e_4
t
$2000 < e_3 < 4000$ & $e_4 > 1000$ & $0 < t < 500$ ms
e_3
(d)

Figure 5. Evolution of the RI for the conditions listed in section 3-A

After the first transformation, the various components are embedded within the event space and it is ready to be acted upon by G - the partition function. In essence, G uses space partitioning techniques to partition the global event space by carving out Regions of Interest (RI) which also play the role of event filters once they are created and primed. The event space thus partitioned helps in separating the critical events from the deluge of events that are present in the event space. The space is partitioned based on user defined rules or implicit operating parameters of the various components. By doing this, we can view the system at well-defined slices across specific dimensions of our interest, including time.

5.1 Regions of Interest (RI)

A Region of Interest is an m-dimensional space embedded within the n-dimensional aggregated event space Ra (m << n). These RIs correspond to rules or conditions that we are interested in monitoring. Let's look at how a RI is created taking an example in Section 3.

Consider the following conditions that need monitoring:
In the Engine,
RPM (e_3) is between 2000 and 4000
Pressure (e_4) exceeds 1000 over a 500 millisecond window

Since the aggregated space is pivoted along the component dimension (refer to Section 4), *Compass* further splits the RI into discrete Space-Time Elements called STEs, each one to monitor a specific component. To begin with, Fig. 5 shows the evolution of a STElement for the above condition, spanning across 2 spatial dimensions and 1 temporal one.

In (a), the region defined over the e_3 dimension demarcates $2000 < e_3 < 4000$.

In (b), the region defined over the e4 dimension demarcates $e_4 > 1000$.

In (c), the hashed region demarcates the intersection of the two regions.

In (d), a new time dimension has been added to the existing region to form STElement.

Let us look at a more complex RI containing 2 STElements. Fig. 6 shows 2 components *(B1,B2)* that are captured in the STEs by virtue of the generated events, thus triggering the RI. In this case, both the components are of a similar kind since they are generating the same kind of events. This scenario makes more sense in a power generation plant for instance, where a multitude of sensors are deployed to monitor various equipments like boilers, generators, turbines, etc. Compass models each of these equipment families as components. Each component is modeled as a point in a 3 dimensional space *S(P,T,t)* where P is Pressure, T is temperature and t is time. Consider the following conditions that need monitoring:

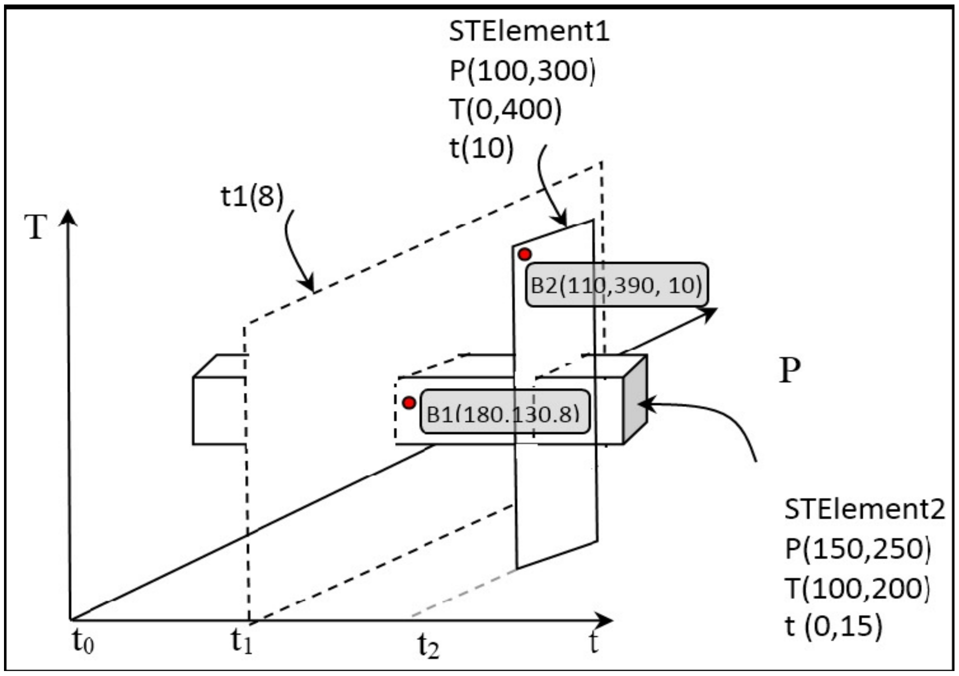


Figure 6. A more complex RI

In a Boiler, it is unsafe if
• 8 seconds after start of the system (at t_1)
Pressure is between 100 and 300
Temperature is between 0 and 400

• Anytime within 15 seconds of observation (within t0 to t_2)
Pressure is between 150 and 250
Temperature is between 100 and 200

Effectively, the ranges provided are the unsafe operating conditions for the device. Fig. 6 shows how compass has modeled the Region of Interest for the above conditions. There are 2 STEs corresponding to the 2 sets of conditions listed above, both of which collectively span across 3 dimensions including a temporal one. You will notice that there are devices B1(180,130,8) and B2(110,390,10), both of which made the RI HOT because the STEs corresponding to the conditions got triggered. Once the event space is partitioned using these Regions of Interest, it provides a channel for the event surge to efficiently flow through. The next section explains how Compass channels the events through these regions with the aid of filters. The separation of these events will help in processing these critical events faster.

6. Filtering The Event Flow

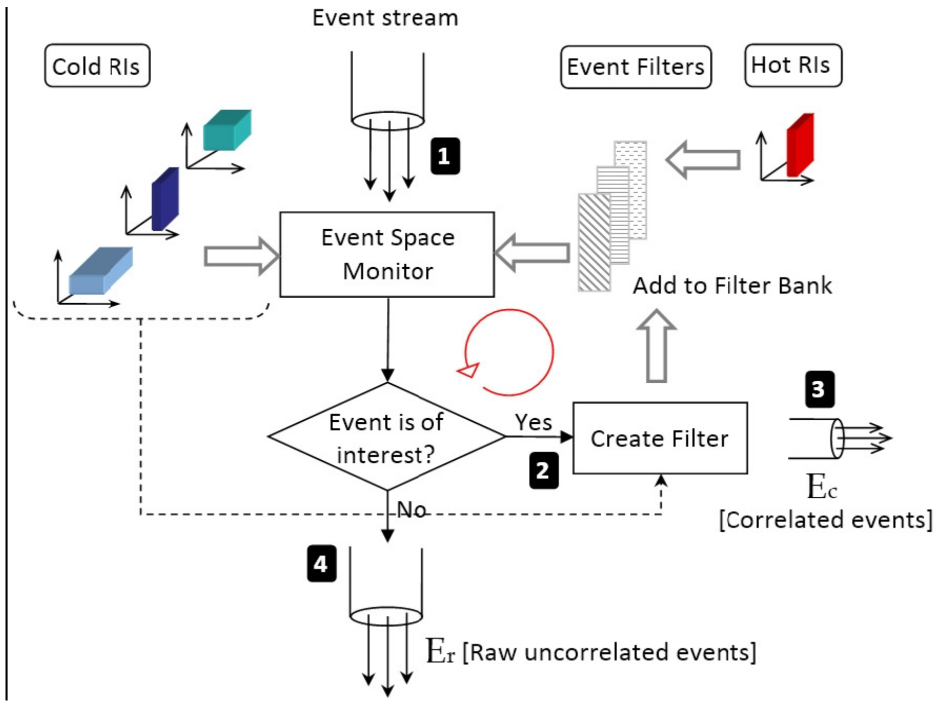


Figure 7. The event flow in more detail. Filtering using Hot RIs

In the beginning of Section 2, we briefly discussed how the RIs help in filtering the global event set into correlated and raw events. In this section, we delve into more details on how this happens (with reference to Fig. 7).

1. The events stream in from the event hose and pass through a filtering mechanism. In the beginning, there are no event filters. Only the RIs are in place. The Event Space Monitor analyses whether the incoming event falls into any of the RI.

2. If an event happens to fall in any of the demarcated regions, then it triggers the RI to be marked as HOT. These HOT RIs act like event filters that separate the other events that are present in the same RI (in the form of other STEs). These STEs act like a bank of filters that will help trap the critical events for faster processing. Once all the STEs are triggered, the RI changes state accordingly to generate alerts or other suitable actions. Section 7. A talks more about RIs as a state machine.

3. The presence of the filters effectively creates a separate channel for the events that are critical and needs to be processed faster. Since these events are correlated to past events, this event channel is called the correlated event channel and the event set is represented as E_c.

4. The events that are not of any interest flow away unhindered. This is the raw event channel E_r.

6.1 Pivoting data - The Partition Function (G)

At the end of Section 6, we saw how Compass aggregates the event data and presents it to the partition function (G). The partition function is primarily responsible for partitioning the event state space into Regions of Interest. Initially, it compares the aggregated events against the Regions of Interest. It marks those RIs as HOT which have been triggered on by the presence of one or more events within their space. Subsequently, it compares the events against the HOT RIs and filters the events into a separate correlated event channel E_c.

The comparison and filtering functions are brought about by pivoting the data slices to match the RIs and filters. At the end of the aggregation function (F) (refer Section 4), the aggregated event slices are grouped by components, followed by axes and corresponding values like shown in Fig. 8(a). It shows a time slice taken at 10:00:25 comprising of 3 components each having data along different dimensions. The number inside the braces indicates the value of the component in the specified dimension. The same figure in (b) shows how the slice looks after it is pivoted along the dimensions. The pivoted data in Fig. 8(b) will enable Compass to easily match the incoming event slice against the RIs.

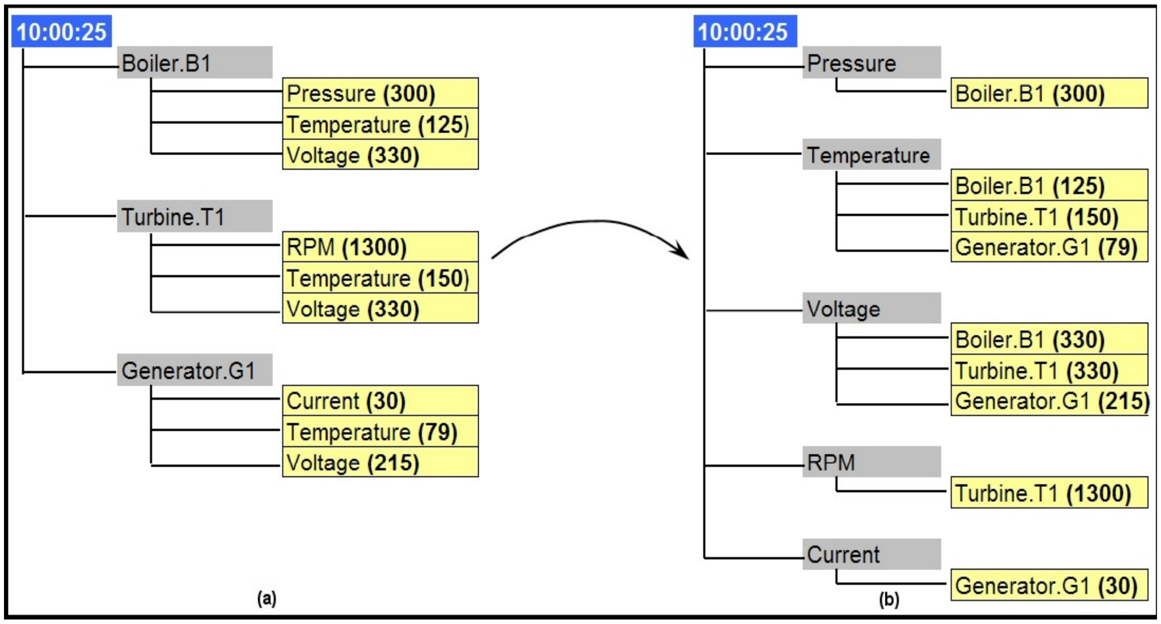


Figure 8. Pivoting the Event slice

Given an Event Slice defined over ‘m’ dimensions (E^m) and a Region of Interest RI defined over ‘n’ dimensions (R^n).

The set of components in E^m is defined as C_E
$C_E = \{C_1,C_2,C_3...C_i\}$ where *i* is the number of components present in the slice. For eg, with reference to Figure 8, *i*=3.

The set of components in R^n is defined as CR
$C_R = \{C1,C2,C3...Cj\}$ where j is the number of components defined in the RIs.

R^n is comprised of several RIs each of which can be represented as the set
$\{R1, R2, ... , Rm\} \subset R^n$

Each component is represented with the number of dimensions in which the component exists – like
C_i^k where K is the number of dimensions that the component has.

Referring to Figure 8, Boiler.B1 is a component that is defined over 3 dimensions - Pressure, Temperature and Voltage.

Each component can also have a pre-defined upper and lower bounds defined as C_i^k L and C_i^k U. A component is considered as operating under safe working conditions as long as the value of the component denoted by X along that dimension (C_i^k X) is between the upper and lower bounds, i.e., $C^k_i L \leq C_i^k X \leq C_i^k U$

Given the above conditions, a comparison of event slice (E^m) with the RI (R^n) can be represented as the intersection of the 2 spaces.

6.2 Marking a RI as HOT

If the intersection of E^m with R^n is not null, then it implies that there is at least one RI belonging to the set $\{R_1, R_2, ... , R_m\}$ that has been triggered and this RI is marked as HOT.

$$E^m \cap R^n \neq \phi \Rightarrow$$

$$\forall\, C_i^k \in E^m, C_j^i \in R^n, \exists\, C_p^q : (C_p^q \in E^m, C_p^q \in R^n) \neg\, {C_p^q}^L \leq {C_p^q}^X \leq {C_p^q}^U \tag{1}$$

What the above equation represents is that if the intersection of the event slice with the RIs is not null, then it implies that there is at least one component C_p^q that exists in both the event slice and a particular RI such that the current value *X* of the component C_p, along the dimension *q*, (${C_p^q}^X$) is out of the upper and lower bounds for that particular component. The RI to which that component belongs is marked as HOT.

6.3 Filtering events based on HOT RIs

Like discussed in Section 6, the RIs that are already marked as HOT will filter the event stream and separates those events that are already correlated (Ec). The set of RIs that are marked hot is given byR_H^n.

The correlated event set Ec contains those events that belong to those components which the RIs in R_H^n are comprised of. The conditions of Equation (1) still hold for filtering out the events from the event stream.

7. Emergence

One of the fundamental concepts of Complex Event Processing is the principle of Emergence. In this case, Compass behaves as an emergent system in terms of being able to compose higher order semantics from the lower order raw events emitted by the individual components. The first step towards creating macrolevel semantics is by partitioning the event space into explicit Regions of Interest (RI). The next stage of emergence is achieved by modeling the RIs as Finite State Machines (FSM). FSMs have earlier been used to observe discrete event systems [4].

The FSM in the RI plays multiple roles, all of which allows Compass to exhibit strong emergent behavior.

- The FSM is used to control and monitor the states that the RI is in. We have already encountered one such state in Section 6 B, where in we qualified a RI as HOT. Section 7.1 elaborates this aspect of the FSM in more detail.
- The FSMs also help Compass to compose higher order states by aggregating the lower order states of the individual RIs. Section 7.2 provides more details on how this is done.

7.1 FSM to model RI states

Like we saw in Section 5.1, each RI is composed of several STElements. These STElements are the partitioned spaces within the RI that will detect if any of the events belonging to the component they are monitoring are a cause of concern. Since several such STElements will simultaneously be monitoring the subspace, Compass provides the flexibility to monitor the RI at various states. For e.g., let us take a look at the scenario where in we are monitoring the following parameters with a RI:
- Pressure in a Boiler should be between 100 and 500 - Temperature of the Turbine should be between 300 and 500 - Voltage of the Generator should be between 2200 and 2400

Compass will create 3 separate STElements in the RI to monitor each separate device. Now, we can define the criticality of the system at various levels. Earlier, in section 5, we had marked the RI as HOT. This is just one of the possible states. We can get more elaborate with the states.

- If all the devices comprising the RI are functioning under safe operating limits, then the RI is marked *GREEN*.
- If one of the devices is outside the safe operating limit, then the RI is marked *AMBER*.

If all the devices are functioning beyond their safe operating conditions, then the RI can be marked *RED*.

To enable such versatility in modeling the states and creating and managing filters, Compass models each RI as a FSM. Formally, the state machines encoding the RIs can be represented by the 5-tuple,

$(\boldsymbol{\Sigma}, \boldsymbol{S}, \boldsymbol{s_0}, \boldsymbol{\delta}, \boldsymbol{F})$, where:

Σ is the input alphabet,

S is the set of states of that cell,

s_0 is the initial state and $s_0 \in S$

δ is the state transition function, such that $\delta: S \times \Sigma \rightarrow S$

F is the final set of states and $F \subset S$

Fig. 9 shows a simple 3 state FSM modeling a typical RI.

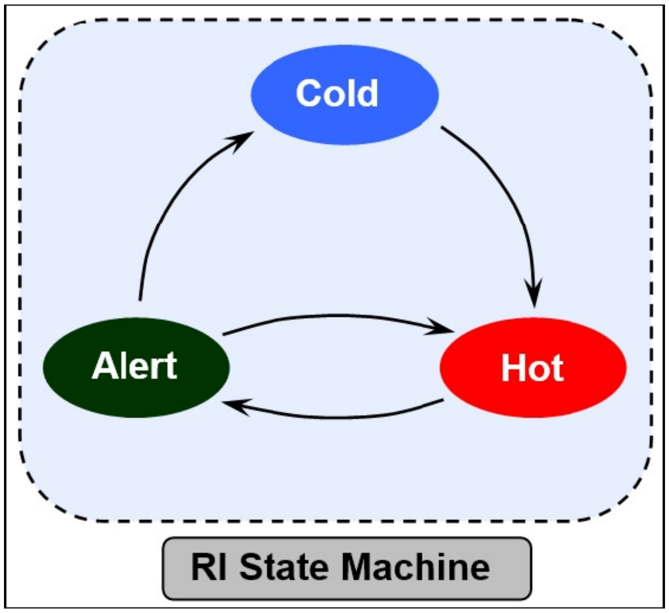


Figure 9. Finite State Machine to model the RI states

7.2 Using RI states to compose higher order states

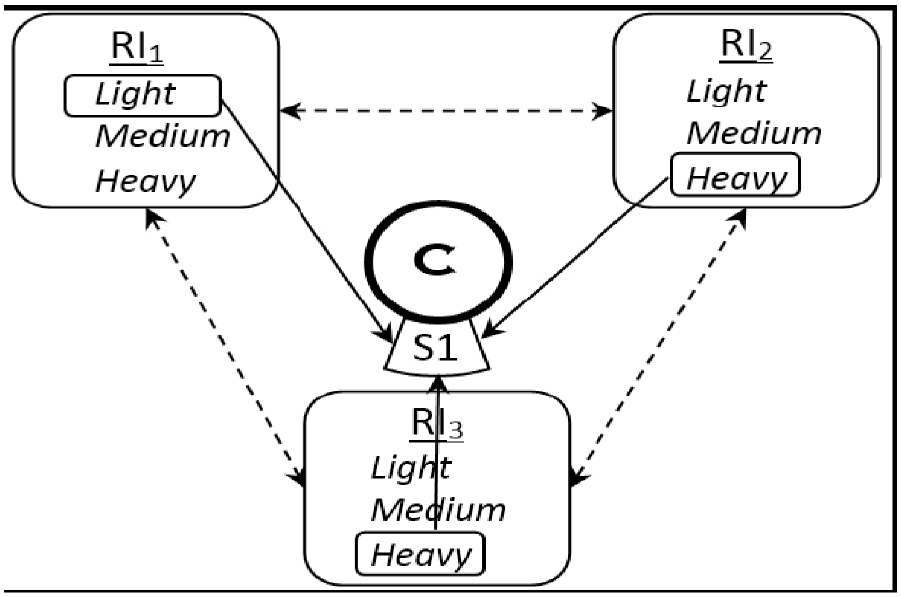


Figure 10. Interconnecting RIs to compose higher order states

The individual states of the RIs (as controlled by the FSMs contained within) can be aggregated to compose higher order states as well. For e.g., looking at Fig. 10, there are 3 RIs defined for the system. They could be modeled as RIs for smarter traffic management. The three RIs shown could be monitoring 3 traffic junctions leading to the city center. The RIs have 3 states based on the traffic density at the junctions; *light, moderate and heavy*. The traffic density at the city center can be controlled by routing the traffic to alternate roads. So, the city center 'state' is composed of the states of the RIs surrounding the city center. For e.g., if RI1 is *light*, and RI2 and RI3 are *heavy*, then the city center can be in a state like '*divert traffic'*, which can be lead to alerts like '*open Lincoln Underpass for eastbound traffic*' and so on.

7.3 Reductionist v/s Global Interactions

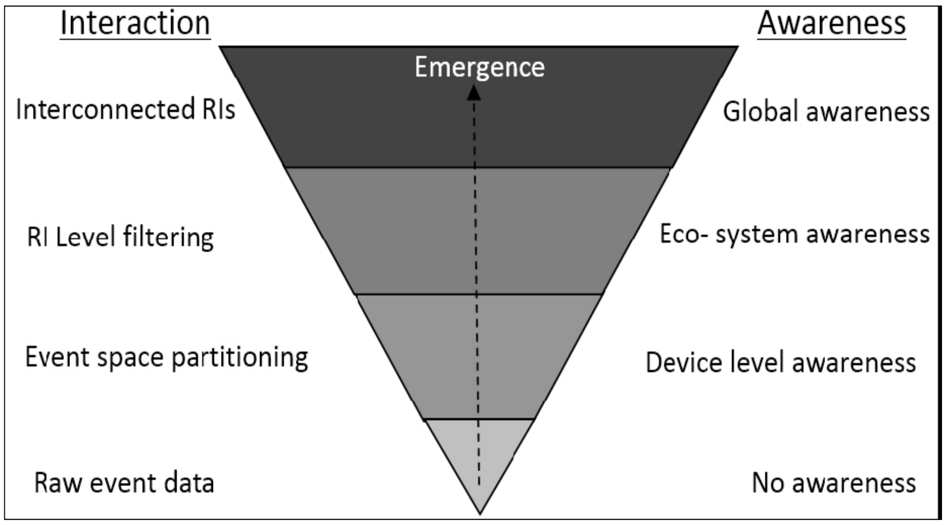


Figure 11. More event correlation leads to higher information dissemination,
thus enabling Compass to exhibit higher emergence

As we have seen earlier, Compass partitions the global state space is into smaller regions amenable for analysis. Each such RI represents a small island which is oblivious to the changes happening around it. This reductionist approach might be restrictive when it comes to understanding the system as a whole. We have seen how Compass overcomes this by using the RI states as shown in Sections 7.1 and 2. When RIs are connected with each other to compose higher order states, the input alphabet to the state machines comprises of both events and the states of other RIs. If 2 events are correlated, but not captured as part of an explicit RI, then these 2 events can be connected by linking the RIs to which they belong. This way, events can be correlated through explicit RIs or implicit links.

As we move up from the individual RI towards interconnected RIs, this dissemination of information across RIs leads to more global interactions which enable emergent behavior. This 'convergence of information' was first presented in [5]. Fig. 11 shows the emergence of higher level awareness in Compass as the amount of interactions increase from bottom to top.

8. Results

Compass simulations were run with various number of devices and number of rules. Almost all the time, the devices were made to run at extreme conditions so that all the events that were generated were beyond the normal operating conditions.

Fig. 12 shows a Compass simulation with 20 devices and 40 rules. The Aggregation Time (refer to section 2.2 for the definition of Aggregation Time) is equal to 20 seconds. The blue line (with square markers) shows the total number of raw events entering the Compass server. The red line (with diamond markers) shows the number of raw events that are not correlated and leave the system (Er). The green line (with round markers) shows the number of events that are filtered by Compass (Ec). You will notice that after the first event slice comes in, there is a small time delay while the RIs get triggered and the filters are created before the filtering of events starts to take place.

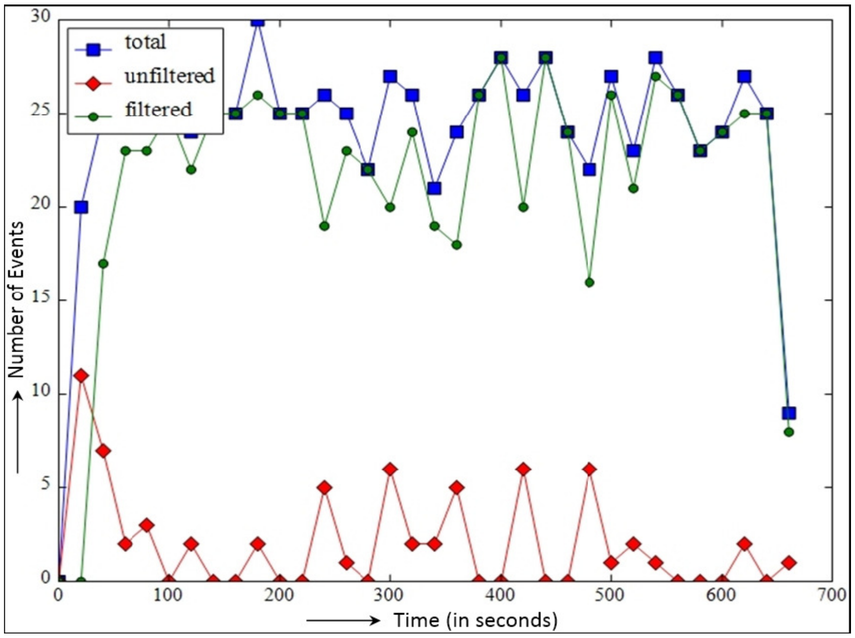


Figure 12. Compass Simulation. Devices=20, Rules=40, Aggregation Time=20

References

[1] http://www.gartner.com/newsroom/id/2636073

[2] http://www.theconnectivist.com/2014/05/infographic-the-growth-of-theinternet- of-things/

[3] Zouganeli, E., Svinnset, I.E. (2009). Connected objects and the Internet of things — A paradigm shift, *In*: Photonics in Switching, 2009. PS ’09. International Conference on , 1 (4) 15-19 Sept. 2009

[4]. Ramadge, P. J (1986). Observability of discrete event systems, *In*: Proc. 25th IEEE Conf. Decis. Contr., 1986, p. 1108–1112

[5] Gangadhar, D.K. (2007). Emergence of spatial awareness in self configuring sensor networks, *In*: International Symposium on Communications and Information Technologies, 2007.