Complex Event Processing on Arbitrary Data in Real Time

Eric B Merritt @ericbmerritt - 10io

May 28, 2012

What we will be talking about

- What is Arbitrary Data
- Munging the Data
- Event Description and Processing
- Distributing the Event Processing
- Open Source Implementation Myrmas
- SML as an Erlang Extention Language

What is 10io?

10io keeps CIOs and their teams off the "hot seat" by transforming their data centers, cloud platforms and smart sensor arrays into autonomic computing infrastructures that are easier to operate and have fewer costly, unscheduled failures.

What do we do?

We take existing known failure patterns and large amounts of operational data from places like:

- Log Files
- Sensor Telemetry Streams
- RESTish Documents

and turn them into actions

What we don't do!

- Machine Learning (yet)
- Post Processing
- Batch Querying

Forensic Analysis

A better title might be

Complex Event Processing on Arbitrary (Semistructured) Data in Real Time

Munge the Data

- Structure it into something we can look at
- Make it Uniform; make it trivial to reason about
- Process it

Structuring the Data

Start with a Log File

```
50.57.61.4 1324830675.076 404 "/var/www/no-such-file"
```

Strait Forward Conversion

```
{"50.57.61.4", 1324830675.076, 404,
    "/var/www/no-such-file"}
```

Conversion with type descriptions

```
{{50,57,61,4}, {1337,951613,818581}, 404,
    "/var/www/no-such-file"}
```

Structured Data from Multiple Locations

```
{{127,0,0,1},  {1337,951613,818581}, "GET", "/",
    "HTTP/1.1", 500, 606, "Mozilla/5.0"},
{36304.521571, "e1000e", "0000:00:19.0", irq, 43,
    for, "MSI/MSI-X"},
{{50,57,61,4}, {1337,951613,818581}, 404,
    "/var/www/no-such-file"}
```

Making the Uniform

Triples

Reperesent each element as a list of triples

```
{UnqiueGraphId, ElementId, ElementValue}
```

UniqueGraphId : A generated globally unique id for that element
ElementId : A generated Id Consisting of Positions+Type
Information ElementValue : The actual value of the element

Element Id

Simple Binary Value where values are tagged

```
    04        00001
| 3 Bits | 13 Bits   |
| Tag    | Position |
```

- 16 bits per level, arbitrary number of levels
- 8196 maximum length

An Example

Erlangish Tuple

```
{{50,57,61,4}, {1337,951613,818581}, 404,
  "/var/www/no-such-file"}
```

Expanded Data

```
{GeneratedID1, <path-type-info>, 4}
{GenertadeID1, <path-type-info>, 4}
{GenertadeID1, <path-type-info>, 50}
{GenertadeID1, <path-type-info>, 57}
{GenertadeID1, <path-type-info>, 61}
{GenertadeID1, <path-type-info>, 4}
{GeneratedID1, <path-type-info>, 3}
{GenertadeID1, <path-type-info>, 1337}
{GenertadeID1, <path-type-info>, 951613}
{GenertadeID1, <path-type-info>, 818581}
{GenertadeID1, <path-type-info>, 404}
```

Pattern Description

Base Pattern

```
(defevent  {?ip, _, 404, ?file1}
           {?ip, _, 404, ?file2}
           {?ip, _, 404, ?file3}
           {_, _, 500, _}
           (when (and (!= ?file1 ?file2)
                      (!= ?file2 ?file3)
                      (!= ?file1 ?file3))))
```

Each Pattern Gets Converted To

```
{GeneratedID1, <path-type-info>, ?ip}
{GeneratedID1, <path-type-info>, ?_}
{GeneratedID1, <path-type-info>, 404}
{GeneratedID1, <path-type-info>, ?file1}
```

Becomes a simple case of unification

Properties We Can Exploit

- Cheap Filtering
- Distributed Processing

Filtering Data

```
{GeneratedID1, <path-type-info>, 4}
{GenertadeID1, <path-type-info>, 4}
{GenertadeID1, <path-type-info>, 50}
{GenertadeID1, <path-type-info>, 57}
{GenertadeID1, <path-type-info>, 61}
{GenertadeID1, <path-type-info>, 4}
{GeneratedID1, <path-type-info>, 3}
{GenertadeID1, <path-type-info>, 1337}
{GenertadeID1, <path-type-info>, 951613}
{GenertadeID1, <path-type-info>, 818581}
{GenertadeID1, <path-type-info>, 404}
{GenertadeID1, <path-type-info>, "/var/www/no-such-file"}
```

Cheap Filtering - To This

```
{GeneratedID1, <path-type-info>, 4}
{GenertadeID1, <path-type-info>, 4}
{GenertadeID1, <path-type-info>, 50}
{GenertadeID1, <path-type-info>, 57}
{GenertadeID1, <path-type-info>, 61}
{GenertadeID1, <path-type-info>, 4}
{GenertadeID1, <path-type-info>, 404} <-- Interesting Bit
{GenertadeID1, <path-type-info>, "/var/www/no-such-file"}
```

Distributed Processing

```
(defevent  {?ip, _, 404, ?file1}
           {?ip, _, 404, ?file2}
           {?ip, _, 404, ?file3}
           {_, _, 500, _}
           (when (and (!= ?file1 ?file2)
                      (!= ?file2 ?file3)
                      (!= ?file1 ?file3))))
```

We can separate this into at least four separate, possibly distribute processes.

Myrmas

Implementions

- Data Flattening
- Event Description and Recognition
- Matching/Unification

Limitations

- No distribution
- No autoparsing of data

SML/MLton as an Optimization Language

- Easily controlled native threading
- Hindly-Milner Type System
- Trivial C Integration (indicates trivial Erlang Integration)
- Whole program optimization

Thank You

Figure: 10io.co

- @ericbmerritt - *Technical*
- @ramcsingh - *Business*
- 10io - http://10io.co