What should I notice? Evaluating unexpectedness of events to generate surprising causal candidates.

*† Étienne Houzé, * Jean-Louis Dessalles, * Ada Diaconescu, † David Menga
* Télécom Paris, IP Paris, *Palaiseau*, France
email: {first}.{second}@telecom-paris.fr
† EDF R&D, *Palaiseau*, France
email: {first}.{second}@edf.fr

***Abstract*—When confronted to an unprecedented situation, humans typically show good performance in quickly identifying noticeable past events and proposing them as possible causal hypotheses. Our research focuses on emulating this behaviour into devices, as to deal with rare, unusual situations where previously acquired knowledge fails to provide hypotheses. We rely on a definition of the unexpectedness of events based on Algorithmic Information Theory and the notion of Kolmogorov Complexity. We illustrate our proposal on a simulated smart homes undergoing various unusual events over the course of more than a year.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

The ability to infer new causal relations is one of the many tasks in which humans prevail but machines still fall short of completing. This ability is prime in making discoveries, or acquiring new knowledge by making educated guesses.

Without any knowledge about how physics or electricty work, it would be possible to infer a possible link between a memorable thunderstorm and a general black-out only by using the noticeability score of the first. While this kind of abductive inference can yield many false-positives, its application can be to propose new hypotheses to be tested out when other possible methods of inference fail to provide any hypothesis (due to a lack of prior similar situations, or to the importance of the context). The difficulty of this approach to reasoning is, however, to quickly identify the relevant candidate hypotheses. In this regards, we may use a basic instinct: without prior knowledge, the most relevant hypothesis might simply be the most memorable recent event. However this definition is highly subjective, and does not seem to fit well into the canvas of computing and AI.

A possible approach would be to rely on the naming complexity of events: memorable events are more likely to be shorter to be named than boring usual events. Think, for instance, of how "last year's hottest day" description appears much simpler than "the 182th day of 7 years ago". Our idea is to rely on this metrics to estimate the description complexity of events. Then, by comparing the description complexity of a given event e to the average complexity of similar events, we induce a measure of unexpectedness that can be used to assess the most memorable events.

To programatically achieve what humans are easily capable to do, we rely on a basic model intuition of how memory works. We assume that a memory is a unordered set of various distinct objects, upon which can be applied boolean predicates to describe them. These predicates thus provide filters that can be applied to the memory. By applying successive filters, we narrow down the size of the memory, until a single event remains. The complexity of this remaining events can thus be estimated as the simplest combination of predicative filter that uniquely describe it.

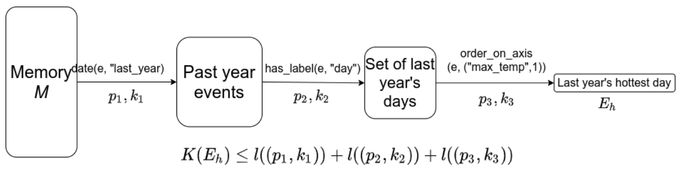


Fig. 1. An example of retrieving an event through successive predicative filters. The "hottest day of last year" is accessed by selecting day-type events, from last year, and looking for the one ranked first by maximum temperature.

The rest of this paper is as follows: we first briefly introduce some notions of Algorithmic Information Theory and its applications in section II, then we explicit our methods of computing the unnexpectedness of past events in cyber-physical systems III. We illustrate this approach by providing an implementation relying on a few predicates on events from a smart home simulation IV. In section V we briefly review other related works using complexity theory or trying to explain smart homes and how our work can be linked with them.

II. Theoretical Background

The notion of complexity formally quantifies the amount of information required for the computation of a binary string (or any object representable by a finite bynary string) [1], [2]. For such a binary string s, its complexity $K(s)$, is the bit length of the shortest program p which, if given as input to a universal Turing Machine U, outputs s.

$$K_U(s) = \min_p \{l(p) | U(p) = s\} \tag{1}$$

The first notable property of this definition is its universality: given that a Turing machine U' can be encoded into U using a program p_U, we have the following inequality:

$$K_U(s) \leq l(p_u) + K_U(s) \tag{2}$$

since the additional constant does not depend on the string s, we can drop the requirement on a given machine U and use a universal complexity $K(s)$, such that:

$$\forall U \in \text{TM}, \forall s, |K(s) - K_u(s)| \leq C_U \tag{3}$$

It is important to note that the notion of Kolmogorov complexity is not related to the computation complexity, as there is no requirement on the execution time of the programs, only their length in bits matters for the computation of complexity. In fact, it can be shown that Kolmogorov complexity is not computable [2]. The sketch of the proof is that computing the Kolmogorov complexity would require to run all programs, up to a certain length, which includes non-terminating ones. As the termination of programs is non-decidable, the computation of complexity is not.

While the introducion of Kolmogorov complexity was purely theoretical, many applications were found, notably because of its close relation with the intuitive notions of the complexity of items, events, situations. For instance, [3] studied the use of Kolmogorov Complexity to identify and solve analogies.

Algorithmic Information Theory is also used to define the notion of unexpectedness: in [4], the unexpectedness $U(e)$ of an event is defined as the difference between an a-priori expected causal complexity $C_w(e)$ and the actual observed complexity $C(e)$

$$U(e) = |C_w(e) - C(e)| \tag{4}$$

Using this measure, it is possible to model phenomenons such as coincidences: imagine that you happen to run into a friend in a park. The unnexpectedness will be much larger if this park is in Singapore than if it is in your common neighbourhood: the causal complexity C_w is much higher if you a priori have no knowledge of your friend being around, while the description complexity remains the same.

III. Computing the unexpectedness of events

A. A memory model

We model the memory as an unordered set M of events e. Individual events can be seen as data points, augmented with a label indicating their nature (temperature events, failure events, addition/removal events), and a timestamp:

$$e \equiv (l, t, \mathcal{D}) \tag{5}$$

where l is the label, t a timestamp and $\mathcal{D}$ a multi-dimensional datapoint representing various characteristics of the events (e.g. its duration, the maximum temperature reached, the sensor name, its position).

We define *predicates* as boolean function operating on these events, and taking an additional argument, encoded as a finite binary string

$$\pi : \begin{cases} \mathcal{M} \times \{0,1\}^* & \mapsto \{0,1\} \\ (e,k) & \mapsto \pi_k(e) \end{cases} \tag{6}$$

Using this construction allows for structures such as *max_temperature(e, T)*, where e is the event argument and T a temperature described by the finite binary string.

From these predicates, we can construct filter operations on the memory. A filter operates by simply selecting the events satisfying a given predicate π_k and constructing a resulting memory from these events.

$$f_{\pi,k}(\mathcal{M}) = \{e \in \mathcal{M} | \pi_k(e)\} \tag{7}$$

For example, the filter "last year" corresponds to $\pi = date$ and $k = \texttt{lastyear}$.

B. Description complexity of events

From the previous definitions, we then estimate the complexity of events as the "simplest" way of retrieving them from the memory, by applying successive filters. A retrieval path composed of various filters $p = f_1 = f_{\pi_1,k_1}, f_2 = f_{\pi_2,k_2}, \dots, f_n = f_{\pi_n,k_n}$ is said to *retrieve* an event e from a memory $\mathcal{M}$ if and only if $p(\mathcal{M}) = f_1 \circ f_2 \circ \cdots \circ f_n(\mathcal{M}) = \{e\}$. For an event e, if such a path exist, its complexity can be bounded by $K(p) = \sum_{f_{\pi,k} \in p} l(\pi) + l(k)$, the complexity of the retrieval path being the sum of the bit lengths required to describe the predicates π and their arguments k.

The description complexity of an event e can thus be defined as the minimum complexity of a retrieval path retrieving e:

$$K(e) = \min_{p|p(\mathcal{M})=e} K(p) = \min_{p|p(\mathcal{M})=e} \sum_{f_{\pi,k} \in p} l(\pi) + l(k) \tag{8}$$

C. Computing an unexpectedness measure

Equation 4 gives a universal description of unexpectedness, as the difference between the complexity of the phenomenon generating an event and the complexity required to describe this event. In our particular application case, we may directly use the description complexity $K(e)$ as defined in 8.

IV. Example

A. Setup

In order to provide an acceptable setup to our experiments, we used the scenario of smart homes. This kind of scenario is a prime example of situations where innovative methods of abduction can prove useful, for various reasons: i) as the number of sensors grows with the number of equipped devices in the house, not all recorded events are useful and should be remembered over long period of time; ii) atypical situations, which are the ones where abduction is most likely to be used (to explain situations to the user, or to solve a conflict), are also where the lack of past data makes knowledge acquisition hard.

We evaluate our approach in these two prominent use cases: recognizing memorable events from a wide array of past measures, and finding relevant hypotheses for abductive reasoning in unusual situations. To achieve both these situtations, we rely on simulations provided by a smart home simulator: iCasa [5].

Using this basis, we implemented a scenario spanning over 420 days, and comprising a daily cycle of outdoor weather

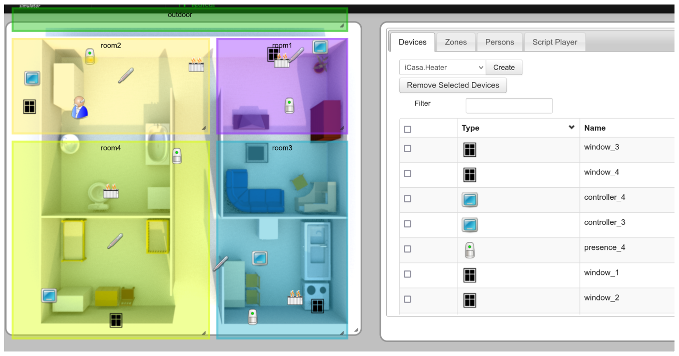
```
$\texttt{current}_{\texttt{explore}} \leftarrow [\mathcal{M}]$ ;
$\texttt{future}_{\texttt{explore}} \leftarrow [\,]$ ;
**while** $\texttt{current}_{\texttt{explore}} \neq [\,]$ ***and*** $\texttt{pass} < \texttt{max\_pass}$ **do**
    **for** $(\texttt{M}_{\texttt{prev}}, \texttt{K}_{\texttt{prev}}) \in \texttt{current}_{\texttt{explore}}$ **do**
        **for** $\texttt{p} \in \mathcal{P}$ **do**
            **for** $\texttt{k} \in \{\texttt{0}, \texttt{1}\}^*$ **do**
                $\texttt{K}_{\texttt{current}} \leftarrow \texttt{l}(\texttt{p}) + \texttt{l}(\texttt{k}) + \texttt{K}_{\texttt{prev}}$ ;
                **if** $\texttt{K}_{\texttt{current}} > \texttt{max}_{\texttt{complex}}$ **then**
                    **break** ;
                **end**
                $\texttt{M}' \leftarrow f_{p,k}(\texttt{M}_{\texttt{prev}})$ ;
                **if** $\texttt{M}' = \{\texttt{e}\}$ **then**
                    $\texttt{K}(\texttt{e}) \leftarrow \min(\texttt{K}(\texttt{e}), \texttt{K}_{\texttt{current}})$ ;
                **else**
                    $\texttt{future}_{\texttt{explore}}.\texttt{append}((\texttt{M}', \texttt{K}_{\texttt{current}}))$;

                **end**
            **end**
        **end**
    **end**
    $\texttt{current}_{\texttt{explore}} \leftarrow \texttt{future}_{\texttt{explore}}$ ;
    $\texttt{future}_{\texttt{explore}} \leftarrow [\,]$ ;
    $\texttt{pass} \leftarrow \texttt{pass} + 1$;
**end**
```

Algorithm 1: Iterative computation of the complexity

(temperature and sunlight), as well as user's movements. All these daily changes create non-noticeable events, serving as a background noise for our experiments. To produce outstanding events, we randomly generated around twenty events, spanning over the whole duration of the simulation, of different kinds:

- Unusual weather: the outdoor conditions are set to unusually high or low temeperatures.
- Heater failures: heater can fail, making them turning off regardless of the command they receive.
- User's vacation: the user go out of the building for an extended period of time.



	Type	Name
☐	▦	window_3
☐	▦	window_4
☐	▭	controller_4
☐	▭	controller_3
☐	▯	presence_4
☐	▦	window_1
☐	▦	window_2

Fig. 2. View of the simulator's web interface provided by iCasa. The four rooms are visible, with their equipment and the user.

The values of all devices and zones variables was regularly monitored throughout the simulation run, and the resulting

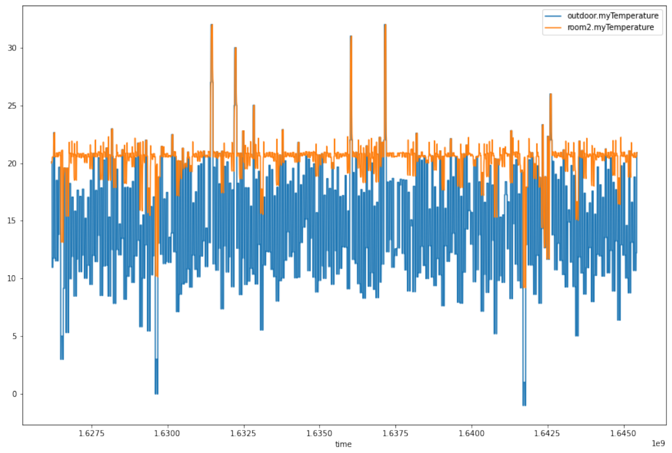


Fig. 3. Time series data from the simulation: outdoor temperature and controller temperature of a room. We can see two temperature drops, one for each room. What could be the causes of these drops?

data, which an excerpt is shown in figure **??** can later be used as a basis for our experiments.

B. Implementation

For the implementation of our method, we first needed to identify and characterize events from the time series data generated by the iCasa simulation. Since this is not the focus point of our present work (see sec. V), we simply apply threshold and precompted conditions based detections to create a base of events.

This base of events consists the basis of the initial memory $\mathcal{M}$ used for computations.

C. Results

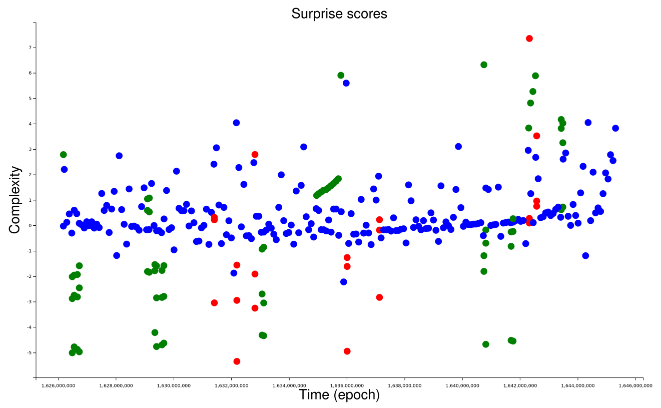


Fig. 4. Unexpectedness score for events in the memory

V. Related Works

VI. Perspectives

VII. Conclusion

References

[1] A. N. Kolmogorov, "Three approaches to the definition of the concept "quantity of information"," *Problemy peredachi informatsii*, vol. 1, no. 1,

pp. 3–11, 1965, publisher: Russian Academy of Sciences, Branch of Informatics, Computer Equipment and . . . .

[2] M. Li, P. Vitányi, and others, *An introduction to Kolmogorov complexity and its applications*. Springer, 2008, vol. 3.

[3] P.-A. Murena, M. Al-Ghossein, J.-L. Dessalles, A. Cornuéjols, and others, “Solving Analogies on Words based on Minimal Complexity Transformations,” in *International Joint Conference on Artificial Intelligence*. IJCAI, 2020.

[4] J.-L. J.-L. Dessalles, “Coincidences and the encounter problem: A formal account,” *arXiv preprint arXiv:1106.3932*, 2011.

[5] P. Lalanda, E. Gerber-Gaillard, and S. Chollet, “Self-Aware Context in Smart Home Pervasive Platforms,” in *2017 IEEE International Conference on Autonomic Computing (ICAC)*, 2017, pp. 119–124.