Rule-Based Complex Event Processing for Food Safety and Public Health

Monica L. Nogueira and Noel P. Greis

Center for Logistics and Digital Strategy, Kenan-Flagler Business School,
The University of North Carolina at Chapel Hill,
Kenan Center CB#3440, Chapel Hill, NC 27599 U.S.A.
{monica_nogueira,noel_greis}@unc.edu

Abstract. The challenge for public health officials is to detect an emerging foodborne disease outbreak from a large set of simple and isolated, domain-specific events. These events can be extracted from a large number of distinct information systems such as surveillance and laboratory reporting systems from health care providers, real-time complaint hotlines from consumers, and inspection reporting systems from regulatory agencies. In this paper we formalize a foodborne disease outbreak as a complex event and apply an event-driven rule-based engine to the problem of detecting emerging events. We define an evidence set as a set of simple events that are linked symptomatically, spatially and temporally. A weighted metric is used to compute the strength of the evidence set as a basis for response by public health officials.

Keywords: Rules engine; complex event processing; answer set programming.

1 Introduction

Even though the U.S. food supply is one of the safest in the world, thousands of foodborne illness cases still occur each year [7]. Surveillance and response to foodborne disease suffer from a number of systemic and other delays that hinder early detection and confirmation of emerging contamination situations. At the onset of an outbreak it is often impossible to link isolated events that may be related to events reported by other data sources. Once distinct events are suspected to be related, public health officials create a cluster and look for confirmatory evidence as part of a lengthy investigatory process. Latencies in the process could be reduced by the earlier availability and synthesis of other confirmatory information, often outside formal public health channels, including information from private companies and consumers.

In practice, local public health departments are usually the first to pick up the signals of foodborne disease. These signals may correspond to reports of illness generated by different types of events, e.g. E1: a patient with symptoms of gastro-intestinal distress seeking medical attention at a hospital emergency room or visiting a private physician's office; E2: laboratory test results for an ill patient which confirm a causative pathogen, e.g. *Salmonella*; and E3: a cluster of ill patients due to a common pathogen. Routinely, a state's syndromic surveillance system collects data from local health care providers about events of type E1, E2 and E3 on a continuous basis, reporting them to the Centers for Disease Control and Prevention (CDC).

However, other events can signal an emerging foodborne disease outbreak. Many public health authorities and food industry operators, e.g. food manufacturers and grocery stores, maintain complaint hotlines (E4) where consumers report foodborne illness or a suspected adulterated food product. Consumer complaints made directly to public agency hotlines, e.g. local health departments (LHDs), state departments of agriculture or departments of environment and natural resources, are officially recorded and may lead to an investigation, at the discretion of the receiving agency.

A public food recall notification (E5) may also signal existing illness cases. Food manufacturers may voluntarily recall of one of their food products due to positive test results for foodborne pathogens, unintentional adulteration, mislabeling, or the presence of an allergen or hazardous material in the food product. Recalls may also be advised by authorities after routine inspections conducted by the U.S. Department of Agriculture (USDA), U.S. Food and Drug Administration (FDA), and state agencies.

Food facility inspection reports (E6), which list violations to the food code applicable to such facilities, provide another signal that may help to identify the root cause of a contamination situation. Evaluation of the type, severity, recurrence, and other characteristics of past code violations for a specific facility and the product(s) it manufactures could help link such operations as a probable source of contamination.

Microblogging and social media networks, i.e. Twitter or Facebook, are nonstandard data sources that hold the potential, yet to be realized, to provide real-time information about emerging food contamination situations. Bloggers posting microblog messages (E7) about illness after eating a certain food product or at a particular restaurant can provide timely warning about an emerging problem.

The paper is organized as follows. Section 2 discusses the motivation and challenges in representing the food safety domain using rules. Section 3 presents our rule-based event model and describes the inference engine developed for our application. An illustrative example of the domain is shown in Section 4. Conclusions and directions for future research are discussed in Section 5.

2 Motivation and Challenges

Data associated with event types E1–E7 described above are collected by separate information systems and maintained and managed by distinct governmental agencies. Hence, in responding to the twin challenges of early detection of and rapid response to emerging outbreak situations, a central problem is how to access, process and interpret more events more quickly, thus reducing the time, scale, and scope of an emerging event. Framing the problem of outbreak detection as a complex event addresses a major failure of current surveillance methods. Current syndromic surveillance systems utilize statistics-based cumulative sum algorithms, i.e. CUSUM, to detect increases in illness reporting numbers and to determine that a foodborne disease outbreak may be emerging or is on-going. Alerts are normally generated by the system when the number of illness cases assigned to a certain syndrome, e.g. fever, respiratory, or gastrointestinal distress, exceeds the threshold determined for that particular syndrome for the geographic area originating these events, i.e. county, and the local population baseline. These alerts are typically based solely on reported illness cases, type E1 to E3 events above. Consideration of event types E4 to E7 aids in the detection of emerging outbreaks before they are sufficiently advanced to rise above the threshold of traditional CUSUM statistical methods.

To better understand the investigatory processes for outbreak detection, consider the following situation. A couple experiences severe gastrointestinal ulceration (GIU) symptoms after eating at a local restaurant chain and seeks medical attention at the emergency room of their local hospital. Two separate illness reports are entered into the health care system to be reported to the state's public health syndromic surveillance system. If the number of reported GIU and foodborne-related cases does not exceed the corresponding threshold for GIU syndrome in the area, then no alert will be generated by the CUSUM algorithm and detection of an emerging situation will be delayed. However, consider that another person falls ill after eating at a different branch of the same food chain and calls a consumer complaint hotline to make a report. Currently, this event will be registered in the receiving agency's database but not automatically passed along to public health syndromic surveillance systems. Consider that another person, also ill after eating at that chain, reports the illness on a personal blog. Both these events occur "under the radar" of public health and are not currently picked up as evidence of a possible emerging contamination.

3 Rule-Based Event Modeling

With respect to rule-based event modeling, our work: (1) extracts relevant information from unstructured text, i.e. web-based recall notifications, to generate events that trigger a rule-based inference engine to "reason" about what it knows in light of the new information encoded by this event; (2) semantically links different types of events by employing (simple) ontologies for food, U.S. geographic regions, North Carolina counties, and foodborne diseases; (3) implements a rule-based inference engine using the Answer Set Programming (ASP) paradigm to identify evidence sets that signal an emerging foodborne disease outbreak; (4) computes an Event Evidence Indicator for newly formed evidence sets as a measure of the strength of the evidence in support of such sets; and (5) reduces latency in outbreak detection by identifying emerging outbreaks when the number of cases affecting individual counties falls below the statistical threshold.

3.1 Event Model

An *event* is defined as the acquisition of a piece of information that is significant within a specific domain of interest to the application. In this application the domain of interest is food safety. We distinguish between two different types of events: simple events and materialized complex events. Simple events include both *atomic* events and *molecular* events. Atomic events have a distinct spatio-temporal identity, i.e. they take place at a particular place and time that is relevant to the determination of the complex event. An example of an atomic event would be a single reported case of gastrointestinal illness, an FDA recall, or a consumer complaint. Molecular events can be thought of as atomic events that are "linked together" by evidence, for example that are joined through previous evidence or by public health experts outside the system. Molecular events could include a confirmed cluster of two or more *Salmonella* cases as determined by DNA fingerprinting. An *event stream* is defined as the sequence of simple events received by the complex event processing (CEP) system that are assigned a timestamp from a discrete ordered time domain and a geostamp consisting of a longitude and latitude geocode. An atomic event has a single timestamp and geostamp; molecular events may have multiple timestamps and geostamps. As defined by [4], CEP consists of techniques and tools that enable the understanding and control of event-driven information systems.

A *complex* or materialized event is an event that is inferred by the engine's rules evaluation of the occurrence of other simple events. For example, in our application the materialized event is a foodborne disease outbreak.

3.1 Semantic Model

Our representation allows for incomplete information which is indicated by a unique reserved symbol of the representation language. Sparse data is an inherent characteristic of the problem, and one of our goals is to detect outbreaks when the number of illness cases has not yet exceeded thresholds employed by traditional statistical methods. The events of interest are described by the following concepts.

A *patient illness case* record contains information about an event of type E1 and uniquely identifies a patient; his county of residence; time and date of visit to health care provider; syndrome or diagnosis assigned; and the disease-causing pathogen. This record will be updated to confirm the pathogen identified by a laboratory test when an event of type E2 corresponding to this patient enters the system. A simple ontology of foodborne diseases and related syndromes is employed to enable the semantic link of diagnosis data and pathogen data across different types of events.

By definition, an illness cluster – a molecular event – is formed by a number of patients with a common diagnosis caused by the same pathogen (as identified by laboratory test results or other causal links). The cluster patient with the earliest disease onset date is referred to as "patient#1." Events of type E3 are represented by two different types of records: (a) a *cluster* record provides information that uniquely identifies a specific cluster; and (b) a *cluster illness case* record contains information about a specific patient of the cluster defined by a *cluster* record. A *cluster* record contains a unique identification code, the disease-causing pathogen, number of counties affected by the outbreak, number of patients in the cluster, the unique identification code of patient#1, and date of patient#1 visit to a health care provider. A *cluster illness case* record contains that cluster identification code, and the patient's unique identification code and county of residence. A c*luster illness case* record acts as a pointer to the more complete *patient illness case* record for that patient.

A consumer complaint call, an event of type E4, is represented by three types of records. A *complaint caller* record provides a unique call identification code, date and time of call, and information about the caller, e.g. caller's county of residence; type of illness codified using the responding agency's medical code; and number of people that fell ill because of the product. A *complaint food operator* record informs about the manufacturer or retailer the caller has complained about. A *complaint food product* record lists the food product and its FDA food code, date of manufacturing, and other information provided by the caller. A food ontology semantically links recalled food products to those implicated by consumer complaint calls. Neighboring relations among North Carolina counties and cities are described by a separate ontology. A recall notification, an event of type E5, is represented by two types of records. A *recall* record contains the event unique identification code, the recall-issuing agency, date and time of its release, recalling company, recalled food product, reason for the recall, e.g. presence of allergen or pathogen, known number of illnesses, and number of geographic areas affected. U.S. geographic areas are defined by a simple ontology which includes all U.S. states and regions as defined by the U.S. Census Bureau. An associated *recall area* record defines a geographic area affected.

3.2 ASP Rule-Based Inference Engine

In this work, we use a form of declarative programming known as Answer Set Programming (ASP) [5], to represent the rule-based CEP of the food safety domain and to search for/detect emerging outbreaks and other information of interest to public health officials. ASP has been applied to industrial problems, but to the best of our knowledge it has not been used in food safety applications.

The ASP paradigm is based on the stable model/answer set semantics of logic programs [1, 2] and has been shown to be a powerful methodology for knowledge representation, including representation of defaults and multiple aspects of reasoning about actions and their effects, as well as being useful in solving difficult search problems. In the ASP methodology, search problems are reduced to the computation if the stable models of the problem. Several ASP solvers – programs that generate the stable models of a given problem encoded in the ASP formalism – have been implemented, e.g. Cmodels, DLV, Smodels, etc. In what follows we provide the basic syntactic constructs and the intuitive semantics of the ASP language used in this work. A complete formal specification of the syntax and semantics of the language can be found at [2, 6].

A signature Σ of the language contains constants, predicates, and function symbols. Terms and atoms are formed as is customary in first-order logic. A literal is either an atom (also called a positive literal) or an atom preceded by $\neg$ (classical or strong negation), a negative literal. Literals l and $\neg l$ are called contrary. Ground literals and terms are those not containing variables. A consistent set of literals does not contain contrary literals. The set of all ground literals is denoted by $lit(\Sigma)$. A rule is a statement of the form:

$$h_1 \vee \ldots \vee h_k \leftarrow l_1, \ldots, l_m, \mathit{not}\ l_{m+1}, \ldots, \mathit{not}\ l_n . \tag{1}$$

where h_i's and l_i's are ground literals, *not* is a logical connective called negation as failure or default negation, and symbol $\vee$ corresponds to the disjunction operator. The head of the rule is the part of the statement to the left of symbol $\leftarrow$, while the body of the rule is the part on its right side. Intuitively, the rule meaning is that if a reasoner believes $\{l_1, \ldots, l_m\}$ and has no reason to believe $\{l_{m+1}, \ldots, l_n\}$, then it must believe one of the h_i's. If the head of the rule is substituted by the falsity symbol $\perp$ then the rule is called a constraint. The intuitive meaning of a constraint is that its body must not be satisfied. Rules with variables are used as a short hand for the sets of their ground instantiations. Variables are denoted by capital letters. An ASP program is a pair of ⟨Σ, Π⟩, where Σ is a signature and Π is a set of rules over Σ, but usually the signature is defined implicitly and programs are only denoted by Π. A stable model (or answer set) of a program Π is one of the possible sets of literals of its logical consequences under the stable model/answer set semantics.

Our encoding – the set of rules of program Π – contains roughly 100 rules, while event records (in ASP, rules with an empty body, also called "facts") and the ontologies describing facts, utilized for experiments, are in the hundreds. We use the DLV system [3] as our ASP solver. To illustrate the ASP methodology, a few (simplified) rules used by our engine to detect emerging clusters are shown below. Rule (2) means that if neighboring counties A and B reported a small number of cases of food-related illnesses, due to pathogen P and/or syndrome S, this constitutes evidence for the engine to create a suspected cluster with associated case records generated by rules of form (3). Thus, an emerging outbreak affecting A and B, due to pathogen P, is computed by rule (4).

suspcluster(A,B,P,S) ← neighbors(A,B),minreached(A,P,S),minreached(B,P,S). (2)

*suspcluster_illness(A,B,Id,P,A) ← suspcluster(A,B,P,S), P != S,
patient_illness(Id,H,M,AmPm,Day,Mon,Y,A,Sys,P).* (3)

susp_outbreak(A,B,P) ← suspcluster(A,B,P,_). (4)

Suspected clusters are linked by (5) to existing recalls of food products affecting the state or the geographic region where it is located. Recalls of food distributed directly to a state or a more specific region, as computed by (5) and (6), are of higher interest.

*more_specif_susprecall_linked(R1,A,B,F1,M1,L1) ←
susprecall(R1,A,B,F1,M1,L1), susprecall (R2,A,B,F2,M2,L2),
subregion(L1,L2), R1 != R2, not other_more_specif(A,B,L1,R2).* (5)

*other_more_specif(A,B,L1,R2) ← susprecall(R2,A,B,_,_,L2),
susprecall(R3,A,B,_,_,L3), subregion(L3,L1), R2 != R3.* (6)

3.3 Evidence Set and Event Evidence Indicator

The set of linked events that provide evidence of the materializing of a complex event is called the *Evidence Set*. An evidence set is associated with a degree of uncertainty as to whether an emerging outbreak event will materialize based on the information in the event data. Rule (7) below exemplifies the computation of elements of the evidence set. Intuitively, the rule meaning is that the engine will conclude that there is evidence that a complaint call C, from county T implicating a food product F1 of type FC–per the FDA Code, is connected to a materialized cluster of illness P affecting neighboring counties A and B, if this call can be linked to an existing recall R of food F2 manufactured by company M at location L, if food F2 is also of type FC.

*evidence(A,B,P,S,R,F2,M,L,F1,FC,T) ← suspcluster(A,B,P,S), nccounty(T),
suspcall(C,A,B,F1,FC,T), susprecall(R,A,B,F2,M,L), type_of(F2,FC).* (7)

The engine computes a measure of the strength of the evidence supporting the conclusion of an emerging complex event through a ranking that ranges from 0, or no evidence, to a maximum of 7, highest evidence rating. The computation of the *Event Evidence Indicator* (EVI) is based on the number and strength of the relationships that connect the events in the evidence set and corresponds to the weighted summation of EVI components calculated for the subsets formed when linking pairs of different types of events. For example, we compute the EVI component for the set of all events corresponding to a suspected cluster and incoming recall notification. The engine also uses EVI to determine what suspected outbreaks to "push" to users and, thus, possibly produce a lower number of false positives, i.e. outbreaks not confirmed later.

4 Illustrative Application

The ASP rule-based inference engine was implemented in the *North Carolina Foodborne Events Data Integration and Analysis Tool* (NCFEDA) shown in Figure 1. In this application, incoming food-related event streams are processed by the *Events Manager* components: (1) a set of databases; and (2) the Event Trigger Module. The databases store all food-related events and geocoded datasets across all contributing NCFEDA public and private sector stakeholders. The Events Manager continuously monitors the databases for new incoming events that are determined by the *Event Trigger Module* as possible triggers. As noted earlier, triggers are events that could include a case related to a foodborne illness, a cluster of illnesses, a recall notification, or a consumer complaint. Inspection reports and microblog messages will be added in the future. The *Event Trigger Module* is composed of *translation units* that convert incoming database event records to ASP *facts* to be evaluated by the ASP inference engine together with all facts describing the domain within a chosen time period.

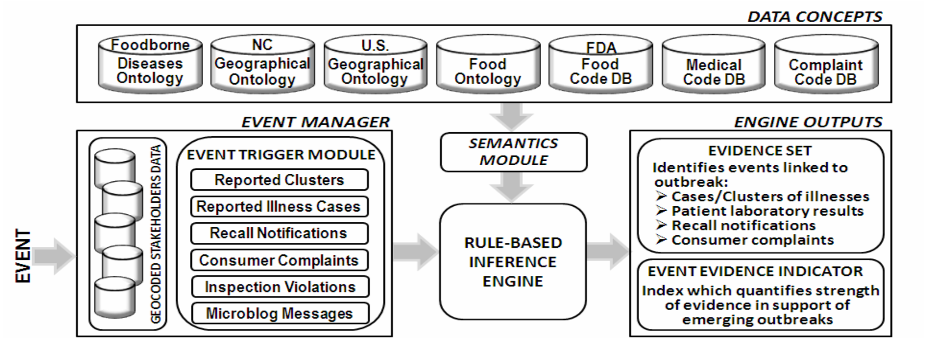

Fig. 1. NCFEDA CEP High-level Diagram

NCFEDA's web scraping tool (not shown in Figure 1) retrieves recall notifications issued by the FDA and USDA directly from their websites generating recall events which are sent to the Events Manager for storing and processing. Upon arrival of a new such event, the Events Manager sends the free-text record to the Event Trigger Module's recall translation unit which (1) parses the recall record text; (2) utilizes the

Semantics Module and ontologies for extraction of any relevant recall information from the text; and (3) generates the recall's corresponding ASP facts.

NCFEDA's *Rule-based Inference Engine* consists of the ASP program containing (a) 100 inference rules for the food safety domain; (b) new and (selected) previously stored facts describing the current situation being monitored; and (c) the DLV solver. Arrival of new events triggers computation of new stable models of the assembled program which will determine whether there is an emerging outbreak event occurring.

5 Conclusions and Future Directions

A primary contribution of this paper has been to frame the outbreak detection problem as a complex event where events include not only structured event data (e.g. case information) but also unstructured event data (e.g. recall or complaint data). We develop semantic models that are able to extract meaningful information from unstructured text data that can serve as event triggers. Using ontologies and rules we are able to discover semantic links between events that provide evidence of an emerging outbreak event. Identification of events that comprise the evidence set is accomplished using ASP. Finally, we introduce a novel concept, the Event Evidence Indicator, which quantifies the strength of evidence in support of an emerging event as a basis for response by public health officials. We successfully implemented these concepts in the NCFEDA prototype. This work is on-going and we are continuing to further develop the rule-based inference engine and the computational strategy for the Event Evidence Indicator.

Acknowledgments. This work was funded by the Department of Homeland Security and Institute of Homeland Security Solutions under Contract# HSHQDC-08-C-00100.

References

1. Gelfond, M., Lifschitz, V.: The stable model semantics for logic programming. In: Kowalski, R., Bowen, K. (eds.) Intl. Logic. Progr. Conf. Symposium, pp. 1070–1080. MIT Press, Cambridge (1988)
2. Gelfond, M., Lifschitz, V.: Classical negation in logic programs and disjunctive databases. New Generation Computing 9, 365–385 (1991)
3. Leone, N., Pfeifer, G., Faber, W., Calimeri, F., Dell'Armi, T., Eiter, T., Gottlob, G., Ianni, G., Ielpa, G., Koch, C., Perri, S., Polleres, A.: The DLV System. In: Flesca, S., Greco, S., Ianni, G., Leone, N. (eds.) JELIA 2002. LNCS (LNAI), vol. 2424, pp. 537–540. Springer, Heidelberg (2002)
4. Luckham, D.C.: The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems. Addison-Wesley Longman Publishing, Boston (2001)
5. Marek, V.W., Truszczynski, M.: Stable models and an alternative logic programming paradigm. In: The Logic Programming Paradigm: a 25-Year Perspective, pp. 375–398. Springer, Berlin (1999)
6. Niemela, I., Simons, P.: Extending the Smodels System with Cardinality and Weight Constraints. In: Logic-Based Artificial Intelligence, pp. 491–521. Kluwer Academic Publishers, Dordrecht (2000)
7. Scallan, E., et al.: Foodborne illness acquired in the United States–Major pathogens. Emerg. Infect. Dis. 17(1), 7–15 (2011)