Benchmarking Semantic Availability of Dynamic Data Feeds

Orna Raz, Philip Koopman, Mary Shaw
Carnegie Mellon University
{orna.raz@cs, koopman@ece, mary.shaw@cs}.cmu.edu

Abstract

Many of the software systems we use for everyday purposes incorporate elements developed and maintained by third parties. These elements include not only code components and data bases but also dynamic data feeds from online data sources. Even though everyday software is not mission critical it must be dependable enough for its intended use. This is limited by the dependability of its constituting elements.

It is especially difficult to assess the dependability of dynamic data feeds because they exhibit not only "fail-silent" behavior but also semantic failures—delivery of unreasonable yet well structured results by a responsive data feed. Further, it is normal for the behavior of such data feeds to change. Unfortunately, the specifications of these data feeds are often too incomplete and sketchy to support failure detection.

We propose an approach for benchmarking the semantic availability of redundant data feeds. The fault model is defined as violations of inferred invariants about the usual behavior of a data feed.

1. Introduction

Everyday software is usually not mission critical, yet it must be dependable enough for its intended use. Assessing dependability requires a model of proper and improper behavior. However, specifications for everyday software are often incomplete and imprecise. When the software incorporates third party elements, such as code components, data bases, and dynamic data feeds from online data source, this situation is exacerbated. Assessing the dependability of dynamic data feeds is especially challenging, because a data feed remains under the control of its proprietor, who might change its format, semantics, or even remove it, as it is being used.

Examples of data sources include stock quotes, weather forecasts and airline ticket prices. A data feed captures a particular usage of a data source: for example, stock quotes for a specific company, the weather forecast for a specific city and airfare for specific origin and destination.

It is especially hard to automatically detect changes in the semantics of a data feed, since the data feed might superficially appear to be delivering the required service. This is the availability facet of dependability, under a *semantic fault model*: the data is delivered, it is syntactically correct, but it is inconsistent, out of range, incorrect, or otherwise unreasonable.

1.1. Semantic availability

Availability is defined as "readiness for correct service", "a measure of the delivery of correct service with respect to the alternation of correct and incorrect service" [1]. We, therefore, define *semantic availability* of a data feed to be its readiness for usage, indicated by whether the data feed delivers reasonable results. We assume the data feed is responsive (no connectivity failures) and delivers parsable results (no syntax/form failures). The availability of the data feed directly affects the availability of the system using it. To measure and assess this availability, the delivery of semantically correct service needs to be estimated, with respect to the alternation of semantically correct and semantically incorrect service. Detecting semantic failures would enable us to estimate the semantic correctness of a service.

Fault tolerance approaches to detection often use state-space methods [4]. This requires specifications of states and transition probabilities between states (left hand side of Figure 1). Masking, which does not require detection, requires specifications of outputs and their selection.

However, a particular problem in the domain of dynamic data feeds is that their specifications are sketchy and incomplete. Unfortunately, the analysis simplicity of the state-space model is not applicable in our setting. In [2] we noted state-space models are difficult to work with when the specifications are inaccurate and suggested an alternative gradient view. The gradient view, depicted on the right hand side of Figure 1, emphasizes the direction of the transitions rather than the precise distinction among states: transitions may degrade or improve performance, though the distinction between working and broken may be fuzzy.

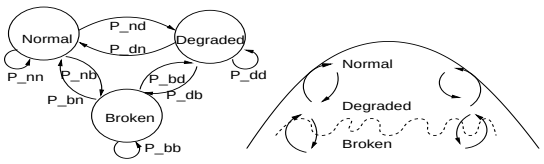

Figure 1. Degradation and failure described by a state-transition diagram and by a gradient view

In [3] we introduced an approach for detecting semantic anomalies in dynamic data feeds, following the gradient view. Rather than demanding better specifications, we infer invariants about the behavior of a data feed using and adapting existing statistical and machine learning techniques. We then use these invariants as proxies for missing specifications. Initial feasibility results indicate these invariants suffice for good-enough detection of semantic anomalies (in the context of stock market tickers).

We believe our approach of inferring proxies for missing specifications, in the form of invariants, can be used to create benchmarks for the semantic availability of redundant data feeds.

2. Benchmarking redundant data feeds

We propose to define benchmarks for evaluating and comparing redundant data feeds (data feeds that provide similar service) based on invariants about the behavior of the data feeds. The measures we propose for the benchmark are the number and nature of violations of invariants (anomalies).

Fault injection approaches often use bit flips to emulate failures and assume a fail-fast, fail-silent behavior. However, for semantic failures in data feeds, it is not clear what a bit-flip fault model would measure. It may test a subset of correctness failures, but we believe there is a need for a stronger fault model. Instead of bit flips we propose violation of invariants as a fault model.

Unfortunately, not only are invariants about the behavior of a data feed rarely provided but also the behavior of the data feed may change. We suggest determining, periodically, a standard set of invariants to be used as a benchmark. These invariants may be not only stationary, but often *adaptive*: the invariants may change as the behavior of the data feed changes.

Our approach of inferring proxies for missing specifications could be used to automate parts of both creating the standard set of invariants and producing benchmark measurements, as follows: periodically,

1. use our invariant inference framework and tools to synthesize a list of candidate adaptive invariants, then
2. have a certification authority, composed of domain experts, select the standard set of invariants from the list (selection through a social process). Constantly:
3. use the standard set of invariants for anomaly detection in the redundant data feeds under test.

Anomalies are detected by evaluating each invariant in the standard set over fresh observations of each of the redundant data feeds and reporting an anomaly when an invariant evaluates to false. We assume it is possible to synchronize the redundant data feeds.

Various comparison metrics of redundant data feeds are possible. For example: (1) the number of detected anomalies and (2) the nature of the anomalies; a larger weight should probably be given to anomalies that are more severe. These metrics could be combined with metrics that measure connectivity and syntax/form availability, for a more complete picture regarding the availability of a data feed.

3. Summary and challenges

We propose: (1) a reference model for how to benchmark the semantic availability of redundant data feeds and (2) tools to aid certification authorities in creating a standard set of invariants. Our premise is that choosing from a list of inferred invariants is easier than creating this list, so having a machine synthesize the list is helpful.

The process of deciding what to benchmark and how to do so is inherently subjective. The task of a certification authority should be made feasible. We believe a fruitful direction is to limit the human intervention and level of expertise required. A possible direction is to require experts to approve only templates of invariants.

Our invariant inference engine cannot guarantee to detect all anomalies. This is true for any technique that does not demand complete specifications. Further, data feeds are dynamic: they are often expected to change. The benchmarks we suggest are, therefore, adaptive. An open issue is when the benefits of adapting to data feed changes are greater than the risks of having a drifting benchmark.

4. Acknowledgments

This research is supported by the National Science Foundation under Grant CCR-0086003 and by the Software Industry Center at Carnegie Mellon University.

References

[1] A. Avizienis et al. Fundamental concepts of dependability. Technical report, UCLA CSD Report no. 010028, 2001.

[2] O. Raz et al. An approach to preserving sufficient correctness in open resource coalitions. In *IWSSD-10*, 2000.

[3] O. Raz et al. Semantic anomaly detection in online data sources. In *ICSE'02*, 2002.

[4] A. Villemeur. *Reliability, Availability, Maintainability, and Safety Assessment*. Jon Wiley & Sons, 1992.