A Polymorphic Model for Event Associated Workload Bursts

Matthew Sladescu∗†, Alan Fekete∗†, Kevin Lee∗†, Anna Liu ∗‡
∗National ICT Australia
†The University of Sydney
‡The University of NSW
{firstname.lastname}@nicta.com.au

Abstract—How many cloud resources does an application provider require to manage workload bursts that often accompany events of public interest (like product announcements or sporting events), and when will these resources be required? The availability and performance qualities of systems from numerous domains have often been compromised by such bursts, highlighting the importance of these questions. Earlier work begins to address these concerns by presenting burst models, which are parameterized by a single set of burst feature types, to describe bursts that can be associated with different event types from different domains. In this paper we argue that the profile of a burst can differ for different event types, and will depend on a variable number of feature types that describe the burst’s associated event. We contribute a method for creating a workload model that is polymorphic based on event characteristics. Our evaluation uses real world data sets for two different event types, and compares our event-based model to one of the most recent, state of the art data sets for two different event types, and compares our polymorphic model to the most recent state of the art burst model [6] found in recent literature [5], [6], [7] using real world data sets containing workload bursts from two different domains. Results highlight the polymorphic model’s superior accuracy to the other model assessed, and the dependence of a burst’s profile on its associated event definition.

Keywords—Events, Burst, Spike, Cloud Computing, Dynamic Resource Provisioning, Burst Profile, Flash Crowd, Elasticity

I. INTRODUCTION

How many cloud resources does an application provider require to maintain QoS during their planned events? Sales events on ticketing and deal of the day websites, result announcement events, auction events, and other event types are known to have induced workload bursts that severely impacted on website availability and QoS. Hence, it is valuable to have accurate models of the workload bursts associated to events, which can assist in answering this question.

Earlier work [1] has shown how such a workload burst model can be used to predict workload during events. Prediction is needed to provision resources on time in the cloud, as there is substantial initialization lag in various cloud platforms [2], [3], [4].

How can one estimate the variation in workload through the period of a burst? Earlier work [5], [6], [7] has identified burst ramp up, plateau, and ramp down phases; providing linear [5], [7] and exponential [6] functions to model the profile of these phases; and showing how the load at different time points relates to the peak burst magnitude. In these workload burst models, a few parameters control the scale of the burst and its duration, but each model attempts to fit all events using one set of profile equations and parameter types.

We posit that not all bursts assume a profile that can be described by the same set of profile equations and parameter types, and that the characteristics of an event can influence the profile of a burst. This can be observed in Figures 3 to 5 that show bursts associated with events with differing characteristics.

Earlier work [1], [8], [9] has shown how features of an auction event, like its starting and ending times, can be used in identifying the profile of bursts for auction events. However, the workload model used in [1], [8], [9] was specific to auction events; in this paper we describe how one can define an event, and we address the challenge of using an arbitrary event definition to construct a workload burst model. We contribute a polymorphic workload burst model that adapts to the features given in a burst’s associated event definition; where features can not only include event starting and ending times, but can also include other information such as the teams playing in a sports match, or the name of a product being marketed. Our key idea is to use features to group similar events, and to use the profile of known bursts from events in a group, as the model for other events with those features. When used with the EAP framework [1], this model will allow one to predict the time and profile of bursts associated with future events, based on bursts associated with past events. We compare the accuracy of this polymorphic model to the most recent state of the art burst model [6] found in recent literature [5], [6], [7] using real world data sets containing workload bursts from two different domains. Our evaluation results confirm that the profile of a workload burst can more accurately be modeled when parameterizing the burst model with event-specific features. In the next section we describe the polymorphic workload burst model; in Section III we compare burst model accuracies; and in Section IV we discuss the evaluation results, and present directions for future work.

II. THE POLYMORPHIC WORKLOAD BURST MODEL

A polymorphic workload burst model aims to represent the workload profile of a given event during its eventful
period, based on that event’s definition and based on events with similar definitions that have occurred in the past. Hence, such a workload model depends on how an event is defined, including how the eventful period for an event is determined. We first describe how events can be defined, and then show how to generate the polymorphic workload burst model.

A. Defining Events

An event can be defined as “a scheduled or unscheduled occurrence, (like an online sale or a breaking news event); which can be described by an arbitrary set of features, (like the event starting and ending times); and can induce load for associated resources (like the web pages or database entries associated with a popular online sale).” (adapted from [1]). In this paper we distinguish time features from secondary features. We label the important times during an eventful period, like the starting time and ending time of an auction event, as time features. Time features are closely linked to when changes in the workload profile occur. For example, the surge of bids in an auction is just before the closing time of an auction; and a surge in workload for a product announcement is close to the announcement time. We label the other features that could influence the workload profile associated with an event, as secondary features. Examples for time features and secondary features for four different types of events are shown in Table I. Notice that we can be flexible in how we define an event, where for example, we can define a single soccer match as an event, or we can define two soccer matches that are happening on the same day as a single event. The latter can be useful for example, when the traffic associated with the two soccer matches overlaps in time, and it is not possible to distinguish dedicated traffic for each individual soccer match. As we will see in section II-C, distinguishing traffic for each individual event is required to learn what the workload profile for such an event looks like.

We define an eventful period as the duration of event-associated workload. For the event type examples listed in Table I, we notice that the eventful period for some events can be bound directly by time-features that are specified (e.g.: auction events don’t experience any bids before the auction starting time or after the auction closing time). Other events like product announcements and sporting events can’t be bounded directly by specified time-features. For instance, product announcements and sporting events can exhibit event-associated workload before the earliest specified time feature (e.g.: workload generated in anticipation of a product announcement or the start of a sporting event), or after the latest specified time feature (e.g.: while interest in the announced product or sporting event is still high). For this type of event, we need to derive the time-features that are defined as the limits for the eventful period. For bursts associated with events like product announcements and sporting events, the time features that describe the limits of an eventful period can be derived by using the earliest and latest time features as inputs to the gradient descent technique used in [10] to identify the lower and upper bounds of a workload burst. This will result in the derived time features listed in Table I that describe the event associated burst starting and ending times. Like in Table I, we will classify all events that can be described by the same set of features, as belonging to the same event type.

B. Introducing the Polymorphic Workload Burst Model

The polymorphic workload burst model aims to assist in producing an estimate for the percentage of event associated hits, (also known as web requests), expected at a given time unit during the eventful period, for a given event, by using information given in the event’s definition. The magnitude of expected hits at a given time unit can also be derived, based on what the equivalent hit magnitude of 1% hits was for past events, or based on history for the current event being considered. Before outlining how this model is generated and trained, we give an overview of its structure, which is shown in Figure 1. In this Figure we notice that each event instance is associated with one or more phases. This collection of phases is constructed for each event, based on a chronological list of time feature values for each event. Each phase is bounded by the time values of two chronological time features for a specific event, and hence phases are unique to each event. There are n phases for an event with n + 1 time features when including derived time features. Each phase is associated to a phase model that aims to characterize the workload profile between the two time features that bind the phase. Note that phase models aim to characterize the workload profile across all events of the same type, and hence a phase model is shared across all events of the same type. Since a phase model is not bound to specific time values, we represent position in the phase model using the concept of phase age, which ranges from age at 0% to age at 100%. This lifetime of a phase model is characterized by m chronological and equidistant cells. The number of cells that is used to summarize the workload profile across a phase’s lifetime is determined using the Freedman and Diaconis rule [11], in an effort to capture only the underlying workload profile during a phase,
Table I
EXAMPLES FOR TIME FEATURES AND SECONDARY FEATURES

<table>
<thead>
<tr>
<th>Event Type</th>
<th>Time Features</th>
<th>Secondary Features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auction Event</td>
<td>- Auction Starting Time</td>
<td>- Auction Category</td>
</tr>
<tr>
<td></td>
<td>- Auction Ending Time</td>
<td>- Seller Reputation</td>
</tr>
<tr>
<td>Product Announcement Event</td>
<td>- Announcement Time</td>
<td>- Product Name</td>
</tr>
<tr>
<td>Sporting Event (defined as a single soccer match)</td>
<td>- Sporting Event Starting Time</td>
<td>- Teams Playing</td>
</tr>
<tr>
<td>Sporting Event (defined as two soccer matches happening on the same day)</td>
<td>- First soccer match Starting Time, Ending Time</td>
<td>- Game Round (eg: quarter final, semi final or final)</td>
</tr>
</tbody>
</table>

and avoid capturing noise, when training the phase model from the workload history of past events. Each cell includes a hypothesis that reports the expected proportion of event associated hits that will occur within that cell, relative to the total number of hits expected within the phase containing the cell. The hypothesis reports this proportion of hits based on the secondary feature values given in an event’s definition. When generating the workload model, a number of machine learning techniques like linear regression, or support vector regression can be used to train the cell hypotheses based on secondary features and workload history of past events. In our implementation, we form a simple string signature that concatenates the secondary feature values of an event, and use a map of event secondary feature signatures to maintain running averages of % hits expected within each cell based on event secondary features. A hypothesis can then use this map to determine the proportion of hits expected based on an upcoming event’s secondary feature signature, or report a global average of expected % hits where a secondary feature signature has not been encountered before. We label the set of phase models that can characterize the workload profile of a given event type, based on specific feature values, as the polymorphic workload model.

When a request is received to report the expected proportion of event-associated hits, for event $E$, at time unit $i$, (for example minute $i$), the following would occur:

1) Locate minute $i$ within one of the event’s chronological phases based on phase bounding time features and their values. We label the identified phase as $X$.
2) Locate minute $i$ within one of the equidistant, chronological cells, that exist within the phase model for $X$. We label the identified cell as $Y$.
3) Use the hypothesis associated to cell $Y$, based on the secondary feature signature of event $E$, to determine the proportion of event-associated hits expected at cell $Y$ relative to the total event-associated hits expected at phase $X$.
4) Based on the length of cells within phase $X$, and using information from step 3, determine the expected proportion of event-associated hits per minute in cell $Y$, and cells adjacent to $Y$.
5) Use linear interpolation, similarly to [1] to determine the expected proportion of event-associated hits at minute $i$.

C. Generating the Polymorphic Workload Burst Model

Events will not only have different values for time features and secondary features, but may also have a different schema of event features, and hence instead of having a single common model for workload bursts, we define a common process, (learning algorithm), for generating the polymorphic workload model. Figure 2 shows how this common learning algorithm must be able to take in repositories of past event occurrences, which may have different feature definition schemas, and use these repositories to generate the corresponding polymorphic workload models.

This common process, shown in algorithm 1, creates a separate phase model for each pair of time features that are defined in the feature schema definition $S$ for a given repository of past events $P$. The set of generated phase models represents the polymorphic workload model for the event type conforming to $S$.

The contents of the outer-most loop in this algorithm is delegated the responsibility of creating a Phase Model that can characterize the workload between a given pair
of time features across all events in the given repository of past events. It does this by dividing the Phase age lifetime discussed earlier into $m$ cells. For each event in the repository of past events, we can then count how many hits exist in each of these $m$ cells. A hypothesis is updated with information about the hit count for a given event at a given cell, the hit count for the phase containing that cell, and the corresponding event’s secondary feature values. Using this information, our implementation of the cell hypothesis can capture variations in the % event associated hits reported, based on the specific secondary feature values of an event, using the map of secondary feature signatures discussed in Section II-B.

In summary, all time features that are specified for an event will be used to define the starting and ending point for phases, where there are $n$ phases for $n+1$ time features. We note that the $n+1$ time features may also include derived time features if there is event associated workload to the left of the earliest time feature, or to the right of the latest time feature. We make use of secondary time features to allow us to model variations in the workload model that may be influenced by such features. We use multiple phases bounded by time features since this allows important workload characteristics like workload surges, (which are usually found close to these time features), to be more accurately located. We will illustrate this with a concrete example after showing the phases that compose the workload model for some of the event types discussed in Table I.

In Figures 3 to 5 we show the number of phases used to model workload bursts for product announcement, weekend soccer match, and weekday soccer match events respectively, as well as the time features that are used to bind these phases. When present, the derived time features that represent the lower and upper burst bounds are shown in italics.

The workload shown in Figure 3 represents the hit-rate of a Wikipedia page around the time of the announcement for a popular product. The lower and upper bounds of the announcement event associated burst are identified by running the gradient descent algorithm described in [10] to the left and right of the announcement time. We note that the tail to the left of the announcement time can be longer or shorter, based on how much anticipation there is for the product being announced. If only one phase was used to model this workload, the location of the burst, quantified by %Age in this phase, would vary based on how long the left tail is. Hence, using a single phase would not allow one to consistently and accurately locate the position of the spike. From the workload traces we have seen in our work, the announcement time has always been relatively close to the workload burst, hence it would be beneficial to use this time feature to help in locating the burst more accurately. This is achieved when using multiple phase models that are bounded by time features, since this would locate the important workload characteristics close to the start or end of a phase model.
Figure 5. Weekday Soccer Match: Actual Trace, Polymorphic Model Phase Bounds, and Exponential Model

Algorithm 1: CreatePolymorphicModel: An algorithm to compute the Polymorphic model for a given event type.

**Input**: A repository \( P \) of past events that includes: an event feature schema definition \( S \); a set of events \( P.E \) that conforms to \( S \); a set of workload history \( W \). Here \( S \) contains a set \( T \) of time feature definitions with \(|T| > 1\), and \( T_{i} < T_{j} \) iff \( i < j \); and a set \( F \) of secondary feature definitions.

**Output**: The polymorphic model \( \mu \), containing a set of phase models that will be associated with corresponding phases as shown in Figure 1, and as described in II-B.

1. initialise \( \mu \);
2. for \( i \leftarrow 1 \) to \(|T| - 1\) do
   3. \( H \leftarrow \emptyset \);
   4. \( m \leftarrow \text{ComputeCellCount}(T_{i}, T_{i+1}) \);
   5. \( C_{1}, \ldots, C_{m} \leftarrow \text{Split}(T_{i}, T_{i+1}, m) \);
   6. for \( j \leftarrow 1 \) to \( m \) do
      7. initialise \( H_{j} \);
      8. \( H \leftarrow H \cup H_{j} \);
   7. for \( e \in P.E \) do
      8. for \( j \leftarrow 1 \) to \( m \) do
         9. \( W_{e} \leftarrow \text{Select}(W, e) \);
         10. \( \text{cell_hit_count} \leftarrow \text{CountCellHits}(C_{j}, W_{e}) \);
         11. \( \text{phase_hit_count} \leftarrow \text{CountPhaseHits}(T_{i}, T_{i+1}) \);
         12. UpdateHypothesisForPhaseCell( \( C_{j}, H_{j}, e, \text{hit_count}, \text{phase_hit_count} \);
      13. Add(\( \mu, H \));
5. return \( \mu \);
iPad” page around the time of product announcements of iPad generations 1 through to 4.

C. Evaluation Method

The accuracy of workload burst models is assessed by comparing each model to a reference model for each event. We define the reference model for an event as a model consisting of $n$ relative frequency histograms, where each of these histograms represents a summary of the actual page hit counts that occurred during a phase in the event associated workload burst. The number of cells used in each histogram is determined using the Freedman and Diaconis rule [11]. The reference model takes the form of multiple histograms to allow for a straightforward assessment of the Polymorphic multi-phase workload burst model accuracy. We identify the lower and upper bounds of each event-associated burst, $x$ and $y$ respectively, using the same gradient descent technique mentioned in Section II-A.

A load trace is also produced using the exponential burst model, described in Section III-A1, between the boundaries $x$ and $y$ identified in the reference model. In the same way that the actual workload burst was summarized for the reference model, this load trace is also summarized using $n$ relative frequency histograms. To allow for comparison we set the number of cells in each of these histograms to correspond to the number of cells used in each corresponding histogram of the reference model. The polymorphic model is also constructed, as described in II-C, where the cell count for each phase is forced to the same number used in the reference model. As mentioned earlier, unlike the exponential model, the polymorphic model has no knowledge of specific workload burst parameters, but rather uses information about workload bursts for similar events present in the repository of past events. In this evaluation we will assess the accuracy of models when applied to event-associated bursts in our data sets, where the repository of past events will contain all events except the one being assessed.

Based on the evaluation process described above, for each event-associated burst, there will be one set of relative frequency histograms for each model, and one reference model that captures the true profile; we then calculate the mean absolute error (MAE) of each model relative to the reference model, for this burst. We report the average MAE across all bursts for a given model. We also calculate a breakdown, showing the average MAE from the histograms of the model that fall within a given phase, across all bursts.

D. Results

The average MAE, and associated 95% confidence intervals, across all workload bursts from the Wikipedia dataset, were calculated for each event burst model assessed relative to the reference model for a burst. These results are reported in Figure 6, where the error component associated with each phase is also shown.

The same metrics were also calculated for bursts present within the world cup data set. Figure 7 shows these metrics for bursts associated with single-match events (semi-finals, the final, and the third place game). For the results shown in the Polymorphic series, the polymorphic model is parameterized by match starting and ending time features, and by a secondary feature to describe if a match took place on the weekend. The series Restricted Polymorphic indicates the impact of not using any secondary features (in particular, ignoring the weekend/weekday distinction).

IV. DISCUSSION

The results of Section III provide evidence that the profile of a burst can be influenced by the associated event’s characteristics. Although the general purpose burst model evaluated was given an advantage in being given the explicit burst peak time and location, this model was still outperformed by a polymorphic model that was parameterized by time and secondary features. Figures 4 and 5 confirm
that even burst profiles associated with the same event type can differ: as already described in [16], the burst profile for weekend matches differs from that of weekday matches (perhaps because some people watch matches on television on the weekend instead of checking scores online). Hence, a polymorphic model that can adapt with knowledge of which matches are on weekends can more accurately represent such workload than previous models, which use the same profiles for all events. In the series Restricted polymorphic of Figure 7 we confirm increased error when the weekend secondary feature is not used. We also experimented with restricting the polymorphic model to using only the starting time for the world cup data set. In removing the match ending time from the soccer event definition, one is also removing information about: extensions to a soccer match like extra time; how long people would be interested in the event; and the burst peak position relative to the match ending time. This led to increased error of phase 1 (which now also includes modeling for a time range that would have been covered by phase 2 had match ending time been used as a time feature), which grows to 6.76±4.56 %Hits, whilst phase 0 remains at 1.14±0.40 %Hits.

In this paper we have shown how the profile of a burst can depend on the associated event’s characteristics, and that the event’s definition should be considered when developing an accurate model for event-associated bursts. This paper has introduced a polymorphic burst model that considers such event definitions, which has exhibited superior accuracy to the most recent state of the art burst model [6] out of [5], [6], [7], when applied to two real-world workload burst data sets. Such accurate models can assist in: understanding how an event’s features will influence an event’s popularity; and in predicting the time, magnitude and profile of upcoming event associated bursts. Foreknowledge of such bursts can allow application providers to provision compute resources on time for associated events, and to understand the types of workload profiles to use in their performance tests. In future work we will consider techniques for effective feature selection, consider the interplay of seasonal effects with our model, and assess prediction accuracy of the EAP framework introduced in [1] when used with the polymorphic model to predict event-associated bursts in numerous domains where burst prediction is important.

V. ACKNOWLEDGEMENTS

NICTA is funded by the Australian Government as represented by the Department of Broadband, Communications and the Digital Economy and the Australian Research Council through the ICT Centre of Excellence program.

REFERENCES


