Personal Scheduling for Concert-Goers at Large-Scale Music Festivals

Eldan Cohen and **Guoyu Huang** and **J. Christopher Beck**
Department of Mechanical & Industrial Engineering
University of Toronto
Toronto, Canada
{ecohen, jcb}@mie.utoronto.ca

1 Introduction

In recent years, music festival have been growing in popularity, generating significant revenue (McIntyre 2015; Mintel 2015). In the U.S. alone, over 30 million people attend music festivals each year, with more than 10 million attending more than one festival each year (Nielsen 2015). Modern music festivals are large-scale events consisting of a set of musical shows, scheduled over the course of a few days at many different venues. The largest festivals exceed 600 shows per day across dozens of venues.

Preparing a personal schedule for a music festival is a challenging task due to the existence of time conflicts between shows and travel times between venues. Festival-goers often spend a significant amount of time deciding which shows to attend, while trying to account for their musical preferences, travel times, and breaks for eating and resting. This problem is often discussed in the entertainment media:

> "The majority of the major conflicts come late in each day–will you dance to HAIM or Flume on Sunday? Will you opt for the upbeat melodies of St. Lucia or Grimes on Saturday?"[1]

> "Just when Coachella is upon us and you couldn't be more excited, a cloud enters – the set times are out, and there are heartbreaking conflicts. Difficult decisions must be made. Do you pass over an artist you love because an artist you love even more is playing all the way across the fest?"[2]

The proposed system addresses the problem of generating an optimal schedule based on user preferences. Our system uses machine learning and combinatorial optimization techniques to learn the user musical preferences and generate a schedule that maximizes the user utility, while taking into account travel times and required breaks. Our system is implemented over a web interface and is able to generate optimal personal schedules in 10 seconds on average.

[1]Zach Long, "Start planning your weekend with the Lollapalooza 2016 schedule," TimeOut Chicago, May 9, 2016, https://www.timeout.com/chicago/blog/start-planning-your-weekend-with-the-lollapalooza-2016-schedule-050916.

[2]Joe Lynch, "Coachella 2016: 10 Heartbreaking Set Time Conflicts (And How to Handle Them)," Billbord, April 14, 2016, http://www.billboard.com/articles/columns/music-festivals/7333891/coachella-2016-set-time-schedule-conflicts.

2 Input

Shows. We consider a set of n festival shows $S := \{s_1, s_2, ..., s_n\}$, each associated with one of the performing artists (or bands) in the festival and taking place in one of the festival venues $V := \{v_1, v_2, ..., v_{|V|}\}$. Each show $s_i \in S$ has a fixed start time, t_i^s, and a fixed end time, t_i^e, such that the show length is $t_i^l = t_i^e - t_i^s$.

Travel Times. We consider an $n \times n$ travel time matrix TT, such that TT_{ij} is the travel time between the venue of show s_i and the venue of show s_j. We do not restrict TT to be symmetric, however, we assume it satisfies the triangle inequality.

Show Preferences. To represent the user's musical preferences, we consider the tuple $\langle f_p, M, N \rangle$. $f_p : S \rightarrow \mathbb{Z}^+ \cup \{\perp\}$ is a mapping from a show to either an integer score or the special value $\perp$ indicating that the user did not provide a score for the show. $M := \{m_1, m_2, ..., m_{|M|}\}$ is a set of show groups, $m_i \subseteq S$, such that the user must attend at least one show in each group. These groups can be used to model a simple list of shows the user has to attend (i.e., if each group is a singleton), as well as more sophisticated musical preferences such as seeking a diversity of musical styles by grouping shows based on style. Finally, $N \subseteq S$ is a set of shows the user is not interested in attending.

Break Preferences. We consider a set of l required breaks $B := \{b_1, b_2, ..., b_l\}$, such that for each $b_k \in B$, w_k^s and w_k^e represent the start and end of a time window in which the break should be scheduled and w_k^t represents the required break length. We assume that breaks are ordered temporally by their index, the time windows are non-overlapping, and at most one break can be scheduled between each pair of consecutive scheduled shows.

3 Architecture

The proposed system architecture is illustrated in Figure 1. Our system implements a web interface, accessible using any web-enabled device.

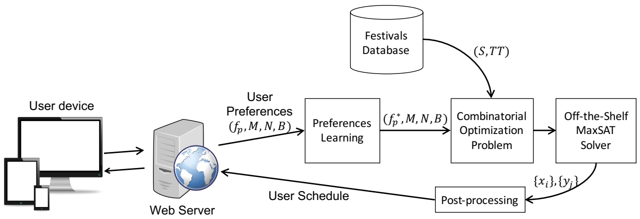

Figure 1: The system architecture

Given an input of user preferences (f_p, M, N, B), provided over a web interface, we start by populating the missing scores in f_p using our preference learning algorithm. Then, we formulate the scheduling problem as a MaxSAT problem and solve it using an off-the-shelf MaxSAT solver. The details of the shows, S, and the travel time matrix, TT, for all festivals are stored in a database on the server. The results are processed and a schedule is produced and displayed over the web interface.

4 Implementation

4.1 Preference Learning

Given a function $f_p : S \rightarrow \mathbb{Z}^+ \cup \{\perp\}$ that maps shows to scores, our preference learning problem consists of replacing each $\perp$ value by an integer value to produce a full mapping f_p^* that is consistent with the user's preferences. To do so, we formulate a regression problem that consists of finding a function $g : S \rightarrow \mathbb{Z}^+$ that minimizes the mean squared error, a common measure of fit, over the set of shows for which a score was provided $Q = \{s_i \mid f_p(s_i) \neq \perp\}$:

$$min \ \frac{1}{|Q|} \sum_{s_i \in Q} (g(s_i) - f_p(s_i))^2$$

The function g will then be used to predict the missing scores:

$$f_p^*(s) = \begin{cases} f_p(s), & \text{if } f_p(s) \neq \perp \\ g(s), & \textit{if } f_p(s) = \perp \end{cases}$$

Our approach is to use the tags assigned to each artist on Last.fm,[3] a popular music website, as a feature set for a regression model that predicts the user score. The tags typically describe the artist musical style and origin (e.g., *pop*, *indie rock*, *punk*, *australian*, *spanish*, etc).

Due to the properties of this problem, notably a large number of features compared to a small training set, we choose to use Elastic Nets (Zou and Hastie 2005), which employ a convex combination of ridge regression (Hoerl and Kennard 2000) and lasso regression (Tibshirani 1996). We use 5-fold cross-validation on the training set to choose the α value from a set of 10 values in $[0, 1]$.

4.2 Scheduling

Our scheduling subproblem consists of finding an assignment of values for a set of boolean variables $\{x_i \mid i \in [1..n]\}$, representing whether or not the festival-goer attends show s_i, and a set of integer values $\{y_j \mid j \in [1..l]\}$, specifying the start time of break b_j. The assignment has to satisfy the user preference w.r.t. M, N, and B (i.e., groups, shows not attended, and break time-windows). Our objective is to maximize the sum of the user-specified scores for the attended shows:

$$max \sum_{s_i \in S} x_i \cdot f_p^*(s_i)$$

To solve the problem, we develop a MaxSAT formulation of the problem and solve it using MaxHS v2.9, a state-of-the-art MaxSAT solver that employs a hybrid SAT and MIP approach (Davies and Bacchus 2013).

5 Evaluation

To evaluate the preference learning method, we used an external dataset of user musical preferences, and showed that our algorithm significantly outperforms a baseline algorithm that is based on the mean score.

To evaluate our scheduling method, we performed an empirical evaluation based on 34 instances of different size. The instances are based on the timetables of seven popular music festivals in recent years.

Our system evaluation showed that the use of preference learning allows us to provide more accurate results and the use of a MaxSAT model allows us to provide an efficient online service, with most instances taking less than 5 seconds and the hardest instances reaching 15 seconds for learning and optimization together.

References

Davies, J., and Bacchus, F. 2013. Exploiting the power of MIP solvers in MaxSAT. In *International conference on theory and applications of satisfiability testing*, 166–181. Springer.

Hoerl, A. E., and Kennard, R. W. 2000. Ridge regression: biased estimation for nonorthogonal problems. *Technometrics* 42(1):80–86.

McIntyre, H. 2015. America's top five music festivals sold $183 million in tickets in 2014. [Online; posted Mar 21, 2015].

Mintel. 2015. Music concerts and festivals market is star performer in the uk leisure industry as sales grow by 45% in 5 years. [Online; posted Dec 9, 2015].

Nielsen. 2015. For music fans, the summer is all a stage. [Online; posted Apr 14, 2015].

Tibshirani, R. 1996. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)* 267–288.

Zou, H., and Hastie, T. 2005. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B* 67:301–320.

[3]http://www.last.fm