An in-depth analysis on timetabling at the University of Twente

Thijs Wiefferink
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
t.w.wiefferink@student.utwente.nl

Patrick van Looy
University of Twente
P.O. Box 217, 7500AE Enschede
The Netherlands
p.vanlooy@student.utwente.nl

ABSTRACT

The aim of this paper is to investigate the timetabling data of the University of Twente and give a performance indication on how well-scheduled the timetables are. Furthermore, the key performance indicators (KPIs) of the University itself are matched with the available data to check their compliance. The results were obtained by transforming spreadsheet data to SQL create and insert statements so that a database could be set up. Next, the relevant data was transformed into XML in order to use XQuery for obtaining relevant results. The results have shown that the KPIs of the university itself were not met in many cases. Plus, interesting trends were obtained by trying out different queries.

Keywords

Data Science, Timetable, Scheduling, XML, Databases, XQuery, XPath.

1. INTRODUCTION

With nearly 10.000 students and 3000 staff members, the University of Twente is a very large organisation[1]. Moreover, it is an organisation in which every division has certain privileges and where every single member wants to function as optimal as possible. In achieving this, scheduling plays an enormous role in satisfying the needs of every stakeholder within the organisation. Counting over twenty bachelor studies and over thirty master programmes, not only research facilities, offices, workspaces, communal areas and food courts are of importance to a well-function environment. All these different study programmes need college rooms for lectures, presentations and tuition too.

Since the start of the very first educational year at the university, schedules have been made to provide a more or less regulated way for providing a solid base where students have the possibility to attend lectures in a decent lecture hall. Although this schedule often contained its faults and flaws, it has been accepted as a tolerable way of organising and assigning the available spaces of the university. Intriguingly, in all those years there has never been a thorough analysis on the performance of the schedules. Possibly, many improvements could be made to establish a more beneficiary schedule for every stakeholder or, at least, the most important members of the organisation. The university has come up with a few key performance indicators (KPIs), which represent the most important goals, so it is about time to actually check whether or not they are actually respected.

To perform an analysis most relevant for the next few years, two datasets containing the schedules for the educational years 2013-2014 and 2014-2015 respectively were analysed. The university's own KPIs were checked to see if the schedules actually satisfy their goals. Additionally, multiple investigations were executed on interesting and outstanding facts and figures notable in the timetables.

This paper describes the evident events obtained by in-depth analysing the datasets of schedules from the past two educational years, finding that actually KPIs are not really that well matched at all and that by creating better algorithms and defining better KPIs, there is a lot to gain considering optimisation.

2. RELATED WORK

The research of Burke and Petrovic [2] introduces some approaches to timetabling problems that were developed or under development at that time in the Automated Scheduling, Optimisation and Planning Research Group (ASAP) at the University of Nottingham. Their aim was basically the same as in our research since they specifically concentrated upon analysing and improving university time tabling as well. The paper suggests a number of approaches and comprises three parts. Firstly, recent heuristic and evolutionary timetabling algorithms are discussed. In particular, two evolutionary algorithm developments are described: a method for decomposing large real-world time tabling problems and a method for heuristic initialisation of the population. Secondly, an approach that considers timetabling problems as multi-criteria decision problems is presented. Thirdly, it discusses a case-based reasoning approach that employs previous experience to solve new timetabling problems which basically is similar to our research where we will be analysing past schedules to give indications for improving future schedules.

Also Burke et al. performed another study in 2004 [3] specifically focussing on examination timetabling as a subset of the entire timetabling problem. In their paper, they carry out an investigation of some of the major features of exam timetabling problems with a view to developing a similarity measure. This similarity measure will be used within a case-based reasoning (CBR) system to match a new problem with one from a case-base of previously solved problems. The case base will also store the heuristic or meta-heuristic technique(s) applied most successfully to each problem stored. The technique(s) stored with the matched case will be retrieved and applied to the new case. The CBR assumption in their system is that similar problems can be solved equally well by the same technique.

Wang's paper *Using genetic algorithm methods to solve course scheduling problems* [4] describes that course scheduling at colleges is an optimization problem to be solved under multiple constraints. The most important tasks of course scheduling should consider various constraints, such as conflicts in teaching hours, meeting teacher pref-

erences, and the continuity of teaching hours, etc. These constraints have been made by the University of Twente too and will be checked by our research to see if they really comply. Wang's study utilised genetic algorithm methods to deal with the multiple constraints issue. The results of this study indicated a significant reduction in the amount of time required for course scheduling, and the results were seen as more acceptable by teachers.

Another study conducted in 2003 [5] researches the design and implementation of a course scheduling system. They are using Tabu Search to accomplish this, which allows them to define strategies and procedures for the timetabling process. They distinguish hard requirements, like no room can have two lectures at the same time, and soft requirements. The hard requirements always have to be met, the soft requirements have weights and determine the quality of the timetable. The used algorithm first makes an initial timetable and then optimizes the timetable for the soft requirements. After that, the room assignment is improved, which at first has a very low priority. Finally, the generated timetables were compared to the manual solutions of The Business School. It was proven useful for a problem where students could choose any course they want, which manually was being dealt with in a suboptimal way. The used techniques give a good idea about the timetabling problem and help us understand what KPIs (Key Performance Indicators) are good to implement as requirements for timetabling.

Lastly, a rather humorous study was published in 2007 characterizing college students' daily alcohol consumption patterns and the relation between Thursday drinking and Friday classes overall and for specific vulnerable groups [6]. Excessive drinking on Thursday, relative to other weekdays, was found and was moderated by Friday class schedule: hierarchical linear models indicated that students with no Friday classes drank approximately twice as much on Thursdays as students with early Friday classes. Students who had classes beginning at 12 PM. or later consumed similar amounts as those with no Friday classes. The magnitude of the Friday class effect was comparatively larger among males. Ancillary analyses based on the subset of students who showed within-subject variability in Friday classes across semesters (i.e., had both early and late or no Friday classes) produced findings similar to those based on the entire sample. In our case, it can be interesting for the UT to examine these findings since Friday classes, especially those before 10 AM, may reduce excessive drinking.

3. MATERIALS AND METHODS

3.1 Given data

For this project data consisting of the time tables of the University of Twente is used. This data was provided by the university itself. The data is stored in a couple of `.xlsx` files, which can be used with *Microsoft Excel*. There are separate files for the year 2013-2014 and the year 2014-2015. The following files have been provided:

- **Activities:** Rows with the course name, lecture type, date (day, start/end time), teacher, group size, student sets, room
- **Course codes:** Rows with the activity name, description, course code, lecture type, date (day, start and end time) and group size
- **Teachers:** Rows with course code, course name, teacher code and teacher name
- **Usage counts:** Per activity of a certain day of the year a count of the number of people actually in the room (counted manually for a few days in the year).
- **Rooms:** Information about the available equipment in certain rooms

The activities and course codes files are actually used for the research in an automated way, the other files are only used to provide context.

3.2 Excel to SQL

To work with the provided data, it has to be converted to a format that is easy to loop through or query in. The first step of the conversion is to get the data in an SQL database. In order to do this, the Excel file was saved as a tab-separated file through Excel itself. Then a Java program was written to transform the tab-separated file into an SQL file that contains insert statements for importing the data into a database. This program first prints a `CREATE TABLE` statement to the output, which creates the table in the database with the correct columns. After this, the program loops through the lines in the tab-separated file and performs a couple regular expressions on each line before adding it to an `INSERT` statement in the program's output.

Converting the activities data to SQL consists of just a couple steps. Firstly, filter forbidden characters like `'` and `"`. Secondly, replace all tabs by comma-space. Lastly, add the start and end of the `INSERT` statement.

The courses data required more work. The main reason for this was that the data holds a course code and a module code column. Normally, just one of these should be filled in; the course code if it is an old course or the module code if it is a course in the new TOM model. However, in practice, this seemed not to be true. Some activities contained both a course code and a module code while others had neither one of them. Therefore extra corrections on the data have been made with regular expressions in addition to the corrections that already were made for the activities to merge the module and course codes into one column.

After the script converted the activities and courses data for both years to SQL statements, they were imported into a PostgreSQL database. This database was chosen because it is capable of generating XML with SQL queries.

3.3 SQL to XML

From the SQL database, the data was transformed in multiple XML databases. For both years, a database of the activities and a database of the courses was generated. The SQL query for generating the activities XML database can be found in Listing 1. This database was used for further querying with XQuery, which is described in Section 3.4. In order to get a valid XML database, a root element had to be added to the output of the SQL query.

As shown in the query of Listing 1, the data was organized by day. This means the XML database contains an element for each day, and this day element contains `<activity>` elements. Activity elements contain the course name/code, teacher, location, lecture type, etcetera. This organisation per day was useful for the questions that were going to be answered. We, for example, investigated the number of wasted hours per day, which is defined as hours between lectures. So, in this case, having all activities of a day together is convenient. Some other questions did not rely on the day grouping, but also did not suffer because of this decision. More about this is explained in Section 3.4. The structure of the generated XML database can be

Listing 1: SQL to XML conversion

```
select
    xmlelement(name "day",
        xmlforest(days.dateGiven, days.daygiven),
        (select
            xmlagg(xmlelement(name "activity",
                xmlforest(t.starttime, t.endtime,
                    t.studentsets, t.room,
                    t.coursename, t.teacher)
            ))
        from
            ut1314 as t
        where
            days.dateGiven = t.dateGiven
        )
    )
from
    (select distinct
        t.dateGiven, t.daygiven
    from
        ut1314 as t
    order by
        t.dateGiven
) as days;
```

found in Listing 2

Listing 2: XML structure

```
<data>
    <day>
        <dateGiven>2013-08-23</dateGiven>
        <daygiven>Friday</daygiven>
        <activity>
            <startTime>08:45:00</startTime>
            <endTime>10:30:00</endTime>
            <studentsets>IDE M 1A A;CEM-CME M 1A
                A</studentsets>
            <room>HB 2A</room>
            <coursename>TW M2 Lineaire
                OptimalisatieZGB/06/01</coursename>
            <teacher>T.W. Wiefferink</teacher>
        </activity>
        ...
    </day>
    ...
</data>
```

3.4 XQuery: gathering statistics

From the initial databases generated by the SQL queries, as discussed in Section 3.3, a couple of databases were generated to specifically check the KPIs. The list below shows details about the generated databases.

The source databases are generated with SQL queries, followed by an XQuery script to transform from these databases the top level items of the list. Then there is another XQuery script for the transformation of each item into its children. In total, there are 14 XQuery scripts for these transformations.

After generating these specific databases, they were used to get the data for the KPIs. For this, there is an XQuery script that loops through these databases and counts the number required for the KPI. The scripts itself can be found on GitHub[1], the data itself cannot be found there since the University of Twente asked to keep the data private.

Activities database:

1. Activities per day, student sets separated as different activities.
   (a) Per day for each student set the contactminutes, collegeminutes and start/end time.
      i. Count of days that student sets have evening classes and next day early classes.
2. Activities per day, student sets separated, classes extended by 15 minutes
   (a) Per day for each student the contactminutes, collegeminutes and start/end time.
      i. 'Wasted' minutes per student set.
3. Activities per day, teachers separated.
   (a) Per day for each student set the contactminutes, collegeminutes and start/end time.
      i. Count of days that student sets have evening classes and next day early classes.
4. Activities per day, teachers separated, classes extended by 15 minutes.
   (a) Per day for each teacher the contactminutes, collegeminutes and start/end time.
      i. 'Wasted' minutes per teacher.
5. Per room the number of minutes it has been used.

Courses database:

1. Per quartile per course the number of planned minutes.

3.5 Per Quartile statistics display

To elaborate on the KPI statistics, an exploration of the data based on per quartile statistics was performed. To generate statistics per quartile instead of per year (as with the KPIs), a Java program was written. The goal is to show these statistics on a website with bar graphs per quartile. To generate files that can be displayed on a website Java is a good choice since it can do a lot more than XQuery. Since Java was already used for data visualisation on a website and the fact that grouping results per quartile in XQuery is complicated, we decided to generate these statistics with Java instead.

We mainly studied the hours that teacher and students have lectures. The first statistic shows counts of college hours per day of students sets, divided into groups for college hours between one and eleven. For teachers, the same statistics have been calculated. Next the number of wasted hours compared to lecture hours was plotted. Lecture hours are the actual hours placed into the timetable, wasted hours are hours between the activities in the time table. Both added together is the number of hours required to be present at the university. These statistics can be used to compare the wasted hours with the useful hours. This statistic has been counted for students and teachers. Next, there is a bar graph of the number of courses, student sets and teachers in each quartile. These are there in order to be able to correct the values of the other statistics. Furthermore, the number of times there is a lecture on Friday evening was plotted as well. Lastly, there is a graph that shows the distribution of lecture types for each quartile. This means it is a stacked bar graph with the number of Lectures, Practicals, Tutorials, Exam hours, etcetera. This data is incomplete since the provided course data is (probably by accident) for 2013-2014 and 2015-2016, which means data for 2014-2015 is missing and that the last year is not yet complete.

As input for the Java program, the XML databases as described in Section 3.4 was used. The Java program reads

[1] https://github.com/NLthijs48/UTTimetabling

the XML database it needs for a certain statistic, counts it per quartile and outputs a JavaScript file that contains the graphing code and data. The HighCharts[2] graph library has been used to display the statistics on a website. The result is visible at `http://wiefferink.me/TimeTabling`. This library gives the user the ability to check and uncheck certain values, making it easy to display and compare the data that is of interest.

4. RESULTS

After having translated the datasets into more workable and query-able formats, which on its own is quite a result already, the KPIs could be checked for their compliance. By writing and querying rather complex XQueries, we were able to check the compliance of the KPIs. Table 1 on page 6 displays the obtained results for the KPI checks and the Exploration.

For both educational years, we have calculated the values for the KPIs. The first KPI says that a student should have a minimum of four contact hours on a day (if they have any). In the first year, the KPI was met 75.06% of the time, for the last year this was the case in 71.04% of the time. The second KPI describes that a student should have six or fewer contact hours a day, which is met 58.84% in the first year and 63.07% in the second year. A couple of KPIs cannot easily be measured as a percentage of compliance and, therefore, have an absolute count. For example, the sixth KPI describes that on Friday there should not be evening classes, which has been violated 269 times in the first year and 99 times in the second year. With these numbers, you could still check for trends over the years. KPI nine describes that rooms should have an occupation of at least 70% during educational weeks, this has been met for just one room in the first year, and zero rooms in the second year. For results of the other KPIs see Table 1 on page 6.

Not only did we check the KPIs given by the University itself but we also came up with a few other analyses to point out some interesting discoveries in the data sets. As mentioned before, these results are displayed on the website `http://wiefferink.me/TimeTabling`. The statistics we came up with are results per quartile, this means a statistic has been counted for each quartile in the two studied years.

From the KPIs we learned that ideally a student has 4-6 hours, so we made this visual by splitting it into a couple categories and then display them in a stacked bar graph. Figure 1 shows these results, the activity count is seen in the vertical axis, the quartiles in the horizontal axis. With this graph, the distribution between the chosen categories can be seen. The same analysis has been executed for teachers, which can be found on the website.

The next statistic is about the wasted hours of teachers and students, which was also in interest by the KPIs. The bar graph in Figure 2 shows the distribution between useful and 'wasted' hours.

Furthermore, we investigated the distribution of lecture types among activities. Figure 3 shows the result. For each quartile, the number of activities of a certain lecture type can be seen.

Finally, we have counted the number of teachers, student sets and activities per quartile, these could be used to correct the other data or give insight in the activity on the university.

[2]http://highcharts.com

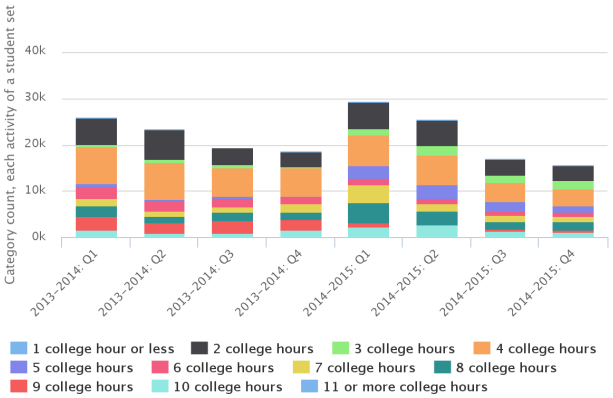

Figure 1: Day lengths for students

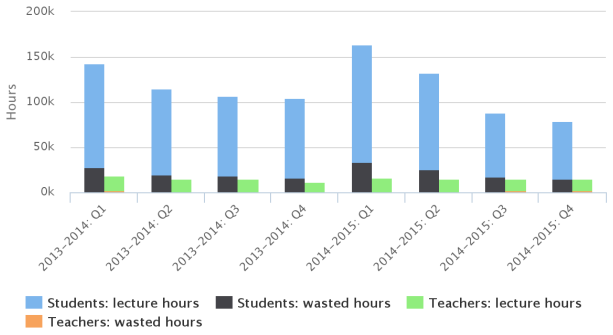

Figure 2: Wasted hours for students and teachers

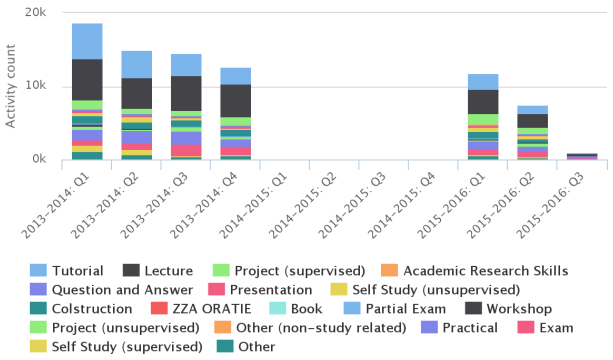

Figure 3: Lecture types

5. DISCUSSION

Although the results seem quite accurate with such large and inconsistent data sets, they are always skewed. This is certainly not different in our case. The datasets show many inconsistencies which already made the first step of translating the Excel data into an insert query for the database a very complex job. Firstly, many filters needed to be applied to obtain a working insert query. It is possible that some of these filters have altered data in a few cases, however, this is quite impossible to validate.

The workflow for transforming this dataset and extracting the statistics we were interested in might not have been the best choice. The problem with the current chain of conversions is that there are too many, and some are hard to automate. Both of these points contribute to the fact that discovering mistakes in earlier stages lead to significantly more work for correcting them since all stages need to run again. It might be better to cut the SQL step out of the chain, or possibly even load the data directly with Java and perform all processing there.

5.1 Assumptions

We were given a set of KPIs and questions to perform analyses with on our datasets. However, it was not obvious in all cases how to interpret these. Therefore, we had to make certain assumptions:

- We defined a contact hour or college hour as 45 minutes (rather than a clock hour)
- We assumed that student sets and teacher were separated with a semicolon
- Interpretation of KPI 4 in table 1 on page 6: End of last college hour minus start of first college hour is less than 495 minutes (11 college hours).
- Interpretation of KPI 6 in table 1 on page 6: Evening classes are classes after the 9^{th} (college) hour (after 17:30 or 5.30 PM)
- For KPI 9 in table 1 on page 6: We defined the total minutes of educational hours as;
  - minutes a day: $9 * 45 = 405$ minutes
  - five days in a week: $5 * 405 = 2025$ minutes
  - 4 quartiles counting ten weeks each (eight weeks of regular education and two exam weeks): $4 * 10 * 2025 = 81000$ minutes

5.2 Corrections

In order to come up with more or less accurate results we had to perform multiple corrections, namely:

- With teachers we discovered that it could occur in some circumstances that a teacher actually had more contact minutes a day then there was between the start of his/hers first activity and last activity. This means that the teacher has overlapping activities in the dataset. In order to correct this, when we observed this behaviour we set it to the actual time between the first and the last activity.
- There were many cases in which no date was given. The only possible option for us was to neglect this entry in our results.
- Also, in many cases the start and end time were missing, so these entries were not accounted for either.

5.3 Inconsistencies

- It can occur that a student is actually present in multiple student sets. Therefore a student set does not represent an individual student.
- It came to our eye that there were some cases in which teachers were separated by a space; this could not be accounted for in the results since it is impossible to build an accurate filter for that.
- In the dataset used for analysing the different lecture types, data from 2014 was missing.

6. CONCLUSION

To conclude, we have seen that the KPIs set as a target by the University itself are not that well defined and certainly not achieved in most cases. Elaborating on this, the results show that the University, at least, tries some form of implementing the KPIs in the timetable algorithms. However, we suspect that there are many improvements possible to boost the compliance rates of the KPI with correct feedback about the final schedules in the form of compliance rates.

Moreover, with more data and specifically more accurate and consistent data, it becomes possible to do a more in-depth analysis on different aspects. For example, it could be interesting to see how the different locations for a particular student are spread out around the campus. With the current data set, this is nearly impossible but when more data is available (possibly a dataset with distances between different locations) this becomes more doable.

7. APPENDICES

7.1 KPI and Exploration results

See table 1 on page 6 of this paper.

8. ACKNOWLEDGEMENTS

We would like to use this opportunity to thank the University of Twente for kindly sharing the datasets of the past two educational years concerning the timetabling schedules with us. Special thanks to Djoerd Hiemstra and Rudy Oude Vrielink for introducing the topic of XML and timetabling.

9. REFERENCES

[1] University Twente, "Ut facts and figures 2014/2015," 2015.

[2] E. K. Burke and S. Petrovic, "Recent research directions in automated timetabling," *European Journal of Operational Research*, vol. 140, no. 2, pp. 266–280, 2002.

[3] E. Burke, A. Eckersley, B. McCollum, S. Petrovic, and R. Qu, "Analysing similarity in exam timetabling," 2004.

[4] Y.-Z. Wang, "Using genetic algorithm methods to solve course scheduling problems," *Expert Systems with Applications*, vol. 25, no. 1, pp. 39–50, 2003.

[5] R. Alvarez-Valdes, E. Crespo, and J. M. Tamarit, "Design and implementation of a course scheduling system using tabu search," *European Journal of Operational Research*, vol. 137, no. 3, pp. 512 – 523, 2002.

[6] P. K. Wood, K. J. Sher, and P. C. Rutledge, "College student alcohol consumption, day of the week, and class schedule," *Alcoholism: Clinical and Experimental Research*, vol. 31, no. 7, pp. 1195–1207, 2007.

Table 1: KPI and Exploration results

#	Key Performance Indicator (KPI)	2013-2014	2014-2015
1	Students have a minimum of 4 contact hours on any day	75.06%	71.04%
2	Students have a maximum of 6 contact hours on any day	58.84%	63.07%
3	Students have a maximum of 2 free hours in 1 series on any day	81.29%	70.93%
4	The timetable of students have a maximum of 11 college hours on any day. This means 8:15 clock hours, which is the time between start of the first college and the end of the last college on any day)	71.69%	70.44%
5	If a student has a class at the 11th and 12th college hour, then that student has no class at the 1st and 2nd college hour the next day	Violated 269 times	Violated 99 times
6	At Fridays there are no evening classes	Violated 232 times	Violated 255 times
7	A teacher has a maximum of 8 contact hours per day	94.39%	93.64%
8	If a teacher has a class at the 11th and 12th college hour, then that teacher has no class at the 1st and 2nd college hour the next day	Violated 23 times	Violated 35 times
9	Rooms must have an occupation of at least 70%. Occupation is defined as follows: occupying a space (room) by the timetabling process during educational weeks	0.37% (just one room)	0.00%
10	Student set with the most 'wasted' time	ATLAS B1 SEM1 A 10320 minutes	ITC AES-ERE 01 8850 minutes
11	Teacher with the most 'wasted' time	WW Wits 10320 minutes	G Meinsma 7860 minutes