General scheduler using SMT Solver

Team Name: MORAGS

Team Members:

1. Adithya Kumar Obilisetty
2. Rohith Reddy
3. Jude K Anil
4. Sangadala Gayathri
5. Sooraj Tom
6. Mayank Kumar Gupta

Introduction

Propositional satisfiability problem (SAT) is the problem of deciding if there is a truth assignment under which a given propositional formula (in conjunctive normal form) evaluates to true. It is a canonical NP-complete problem and it holds a central position in the field of computational complexity. SAT solvers have successfully been applied to planning and scheduling problems and there have been some records of using SAT solvers in timetabling applications.

Our strategy is to encode all timetable requirements in propositional logic and generate a CNF formula that describes all timetable constraints. We use SAT solvers to search for the models of the generated formula. Each found model represents a valid teaching timetable. If the formula is found to be unsatisfiable, some requirements have to be relaxed and the process is repeated until a solution is found.

Different institutions impose a range of specific teaching timetable requirements. Many of these requirements are obligatory and must be satisfied in order to have a valid timetable. Therefore, a timetabling system must be general enough and allow users to specify a very rich set of different constraints. Our approach can be easily adapted to fit a wide set of different requirements.

A General Problem Description

The *course scheduling problem* is to assign given lessons to given time slots while obeying some given requirements. The room allocation problem is to additionally assign given lecture rooms to given lessons while obeying some given requirements.

Proposed Solution

Assumptions

1. All working weeks during the semester are considered the same.
2. A week consists of several working days divided into a number of equal-length time-slots (we will call them periods) and lessons are required to fit into these time slots.
3. Students are available on each working day and each working hour.

Requirements

There are two sorts of requirements.

1. Correctness requirements are essential for timetable correctness and all these requirements have to be satisfied.
2. Comfort requirements represent additional wishes of the staff. These requirements can be given as input.

Correctness Requirements

1. Each lesson from a predefined list of lessons has to be scheduled, and it should be scheduled exactly once in the timetable.
2. A teacher cannot teach two different subjects at the same time. It is possible that a teacher is required to teach the same subject to several different groups at the same time.
3. A group cannot attend two or more different lessons at the same time.
4. Only one teacher can occupy one room in one given period.

Comfort Requirements *(Can be mentioned as Input)*

- **Forbidden and requested working hours**
    - Teachers are allowed to state their forbidden hours, i.e., the hours in which they cannot give lessons.
    - Some teachers (e.g., visiting professors, senior professors) are allowed to explicitly state their teaching hours, i.e., the hours in which their lessons have to be scheduled.
    - Groups are also allowed to have forbidden hours.

- **Group and teacher overlapping**
    - It can be required that some different groups do not attend lessons in the same time.
    - Some teachers require not to give lessons in the same days as some of their colleagues (lab constraints etc).

    - Some teachers require to give lessons in the same time (or at least in the same days) as some of their colleagues.

- **Number of teaching days**
    - Some teachers ask to have their lessons scheduled only in a given number of working days (e.g., only in two days in a week)

- **Work day duration**
    - Groups of pupils/students are allowed to have only up to N (e.g., 7) working hours a day, including idle periods.
    - Teachers are allowed to have only N (e.g., 6) teaching hours in a day, including idle periods.

- **Idle periods**
    - Teachers usually ask not to have idle periods. Still, some teachers explicitly require to have idle periods.
    - Groups can/cannot have idle periods.

- **Lessons that have fixed or forbidden periods**
    - It might be required that some more demanding subjects (e.g., Mathematics) can not be scheduled for the last period in a shift, because pupils get tired after a hard days work.
    - Some subjects are required to be scheduled either for first or last period in a day.

- **Consecutive days**
    - Some courses are taught in several lessons during a week (e.g., a course with 4 lessons in a week is required to be scheduled as 2+2, i.e., two lessons one day and two lessons another day). It can be required that these days are not consecutive days.
    - Some teachers who do not teach every day a week want their lessons scheduled in consecutive working days.

- **Changing shifts and buildings**
    - In institutions that work in different shifts, neither pupils nor teachers are allowed to work in different shifts during the same day.
    - In institutions that have different buildings that are far apart, neither teachers nor students should have lessons in different buildings the same day.

SAT Encoding

Variables and Their Relationships

- **Working hours**
  - The set of working days of a given institution will be denoted by days. Each working day is divided in a number of equal periods.
  - The set of periods for a day $d \in days$ will be denoted by $periods(d)$.

- **Lessons**
  - All lessons that should be scheduled are denoted by a quadruple of the form $tsgn$ which represents the fact that the teacher t teaches the subject s for the group g for the nth time in a week.
  - Each lesson has its own duration which is denoted by $duration(tsgn)$ and is expressed in number of periods.
  - The list of lessons for a given teacher t will be denoted by $lessons(t)$.
  - The list of lessons for a given group g will be denoted by $lessons(g)$.

- **Idle Periods**
  - We say that a teacher (or a group) has an idle period if he/she does not have lessons in that period, but has lessons before and after it.

- **Notations**
  - $single(\{v_1, \ldots, v_k\})$ -> *only one out of* $v_1, \ldots, v_k$ *is true*
  - $cardinality(\{v_1, \ldots, v_k\}) = m$ -> *m out of* $v_1, \ldots, v_k$ *is true*

- **Variables**
  - x'_{tsgndp}
    - Formed for each $tsgn, d, p$.
    - It represents the fact that the lesson $tsgn$ begins in a day d and a period p.
  - x_{tsgndp}
    - Formed for each $tsgn, d, p$.
    - It represents the fact that the lesson $tsgn$ is given in a day d and a period p.
  - x_{tsgnd}
    - Formed for each $tsgn, d$.
    - It represents the fact that the lesson $tgsn$ is held in the day d.
  - x_{tdp}
    - Formed for each t, d, p.

        - It represents the fact that the teacher t gives a lesson in a day d, in a period p.
    - x_{gdp}
        - Formed for each g, d, p.
        - It represents the fact that the group g attends a lesson in a day d, in a period p.
    - x_{td}
        - Formed for each t, d.
        - It represents the fact that the teacher t teaches during the day d.
    - i^k_{tdp}
        - Formed for each t, d, p.
        - It represent the fact that the teacher t has idle period of length k in the day d, starting with the period p.
    - i^k_{td}
        - Formed for each t, d.
        - It represents the fact that the teacher t has idle period of length k during a day d.
    - i^k_t
        - Represents the fact that a teacher t sometimes has idle period of length k.
    - i_{tdp}
        - Formed for each t, d, p.
        - It represents the fact that the teacher t has an idle period in the day d starting with the period p.
    - l^k_{gd}
        - Formed for each $g, d \text{ and } k < |periods(d)|$.
        - Duration of a working day for student groups.

- **Truth Implications**
    - $x'_{tsgndp} \Rightarrow x_{tsgndp}$
    - $x_{tsgndp} \Rightarrow \bigcup_{p2 - duration(tsgn) + 1 \le p1 \le p2} x'_{tsgndp}$
    - $x_{tsgndp} \Rightarrow x_{tsgnd}$
    - $x_{tsgnd} \Rightarrow \bigcup_{p \in periods(d)} x_{tsgndp}$
    - $x_{tsgndp} \Rightarrow x_{tdp}$

- $x_{tdp} \Rightarrow \bigcup_{tsgn \in lessons(t)} x_{tsgndp}$
- $x_{tsgndp} \Rightarrow x_{gdp}$
- $x_{gdp} \Rightarrow \bigcup_{tsgn \in lessons(g)} x_{tsgndp}$
- $x_{tdp} \Rightarrow x_{td}$
- $x_{td} \Rightarrow \bigcup_{p \in periods(d)} x_{tdp}$
- $i^k_{tdp} \Leftrightarrow (x_{td(p-1)} \wedge (\bigcap_{0 \leqslant j < k} \neg x_{td(p+j)} \wedge x_{td(p+k)}))$
- $i^k_{tdp} \Rightarrow i^k_{td}$
- $i^k_{td} \Rightarrow \bigcup_{min(periods(d)) + 1 \leq p \leq max(periods(d)) - k} i^k_{tdp}$
- $i^k_{td} \Rightarrow i^k_t$
- $i^k_t \Rightarrow \bigcup_{d \in days} i^k_{td}$
- $i^k_{tdp} \Rightarrow i_{tdp}$
- $i_{tdp} \Rightarrow \bigcup_{1 \leq k \leq max(periods(d)) - p} i^k_{tdp}$
- $x_{gdp} \wedge x_{gd(p+k-1)} \Rightarrow l^k_{gd}$
- $l^k_{gd} \Rightarrow \bigcup_{min(periods(d)) \leq p \leq max(periods(d)) - k + 1} (x_{gdp} \wedge x_{gd(p+k-1)})$

- **Correctness Conditions**
  - Each lesson has to be scheduled. The following must hold true for each lesson $tsgn$

    $\bigcup_{d \in days} x_{tsgnd}$

  - Each lesson is scheduled exactly once in the timetable. Therefore, the beginning of each lesson is uniquely determined and the condition below must hold true for every lesson $tsgn$.

$single(\{x'_{tsgndp} \mid d \in days,\ p \in periods(d)\})$

- Each group can attend only one lesson at a time. The following must hold for every $g,\ d,\ p$
  $single(\{x_{tsgndp} \mid tsgn \in lessons(g)\})$

- Since it is required that every teacher can teach only one lesson at a time, for each teacher t, each $t \in days$, and each $p \in periods(d)$
  $single(\{x_{tsgndp} \mid tsgn \in lessons(t)\})$

- **Comfort Conditions**
  - Forbidden working hours are directly encoded by negation of variables x_{tdp}, x_{td}, $and\ x_{tp}$ (Forbidden hours for groups can be done in a similar way).
  - The condition that two groups g_1 and g_2 are not allowed to attend lessons in the same time, is encoded by $x_{g_1dp} \Rightarrow \neg x_{g_2dp}$ and $x_{g_2dp} \Rightarrow \neg x_{g_1dp}$, for each day d and period p. Overlapping of teachers' teaching hours is encoded in a similar way.
  - The condition that a teacher t teaches for exactly n days in a week is encoded by $cardinality(\{x_{td} \mid d \in days\}) \leq n \wedge cardinality(\{\neg x_{td} \mid d \in days\}) \leq |days| - n$
  - The requirement that a work day duration for a group is limited to n periods, is encoded by single literal $\neg l^k_{gd}$ for each $k > n$
  - The requirement that a work day duration for a group is at least n periods is encoded by the constraint $x_{gd} \Rightarrow l^n_{gd}$, Similar for teachers.
  - The requirement that idle periods of length k are not allowed for the teacher t is specified by single literal constraint $\neg i^t_k$.
  - The requirement that a teacher t is not allowed to have more than one idle period per day the condition is specified by
    $single(\{i_{tdp} \mid min(periods(d)) + 1 \leq p \leq max(periods(d)) - 1\})$
  - The requirement that a teacher t is allowed to have at most n idle periods per week is specified by a cardinality constraint:
    $cardinality(\{i_{tdp} \mid d \in days,\ p \in periods(d)\}) \leq n$
  - The requirement that a lesson can begin only in period $p_1,\ p_2,\ \ldots,\ p_k$ is encoded by:
    $x_{tsgndp} \Rightarrow x'_{tsgndp_1} \vee \ldots \vee x'_{tsgndp_k}$
  - The requirement that a lesson $tsgn$ must be the first or last lesson for the group g in a day d, is encoded by :

$$x'_{tsgndp} \Rightarrow \left(\bigcap_{min(periods(d)) \leq p' < p} \neg x_{gdp'}\right) \vee \left(\bigcap_{p + duration(tsgn) \leq p' \leq max(periods(d))} \neg x_{gdp'}\right)$$

- If some lessons for the same course should not occur in consecutive days, the condition below must hold for every d except last working day

  $x_{tsgnd} \Rightarrow \neg x_{tsg(n+1)(d+1)}$

- **Room Allocation**
  - Room Allocation can be considered an isolated problem and can be solved with similar constraints displayed above.

Conclusion

The proposed method is quite general in terms of where it can be used, such as, Universities, High Schools, Departments etc. Since it is an NP-Complete problem the worst case runtime is 2^n. But In practice the runtime is almost polynomial.

Other

1. **SAT Solver:** Z3 (https://z3.codeplex.com)

References

1. An algorithm for constructing University timetables *by Mary Almonds*
2. Solving Timetable Scheduling Problem by Using Genetic Algorithms *by Branimir Sigl, Marin Golub, Vedran Mornar*
3. Timetabling Based on SAT Encoding: A Case Study *by Filip Marić*
4. Automatic Timetable Generation using Genetic Algorithm *by Dipesh Mittal, Hiral Doshi, Mohammed Sunasra, Renuka Nagpure*