CPSC 433 — Assignment I

Daniel Gillson
David Keizer
Ethan Higgs
Zheng Liang
Zibin Mei

October 20th 2017

Problem Description

The problem of assigning courses (resp. course sections) and labs to weekly time slots, in its basic version, can be rather easily described. We have a set,

$$\mathrm{Courses} = \{\mathrm{c}_1, \ldots, \mathrm{c}_\mathrm{m}\}$$

of course sections that a department teaches in a particular semester; and a set,

$$\mathrm{Labs} = \{\mathrm{l}_{11}, \ldots, \mathrm{l}_{1\mathrm{k}_1}, \ldots, \mathrm{l}_{\mathrm{m}_1}, \ldots, \mathrm{l}_{\mathrm{mk}_\mathrm{m}}\}$$

of labs (or tutorials) such that $\mathrm{l}_{\mathrm{i}1}, \ldots, \mathrm{l}_{\mathrm{ik}_\mathrm{i}}$ are the labs for course (section) c_i (which means that k_i can be 0, if the course does not have any labs). And then we have a set,

$$\mathrm{Slots} = \{\mathrm{s}_1, \ldots, \mathrm{s}_\mathrm{n}\}$$

of time slots in which courses and labs have to be fitted.

For each slot s_j, we have a limit coursemax(s_j) (a natural number) of courses that can be put into the slot and also a limit labmax(s_j) (again, a natural number) of labs that can be put into the slot (courses and labs are independent from each other in this regard, i.e. a slot can take coursemax(s_j) courses and labmax(s_j) labs). Naturally, the labs to a particular course can never be in the slot in which the course is. There are also other limitations (so-called *hard constraints*) and wishes that the department has (but they might not be fulfillable, therefore these wishes are called *soft constraints*).

The task that the system described in this paper must fulfill is to find an assignment *assign* of courses and labs to slots that fulfills the hard constraints and optimizes the soft constraints. More formally, *assign* is a function, *assign*: Courses + Labs ⇒ Slots that fulfills two conditions, namely:

1. **Constr**(*assign*) = true, with **Constr** testing the fulfillment of all hard constraints (and being true if and only if every single hard constraint is fulfilled). And,
2. **Eval**(*assign*) is minimal in the set of all possible assignments fulfilling **Constr**.
   Eval is an evaluation function that measures how well an assignment fulfills the soft constraints.

A listing of the hard and soft constraints will follow.

Hard Constraints:

- General Hard Constraints:
  - Not more than coursemax(s) courses can be assigned to slot s.
  - Not more than labmax(s) labs can be assigned to slot s.
  - *assign*(c_i) has to be unequal to *assign*(l_ik) for all k and i.
  - The input for your system will contain a list of *not-compatible*(a, b) statements, with a, b in Courses + Labs. For each of those, *assign*(a) has to be unequal to *assign*(b).
  - The input for your system can contain a partial assignment *partassign*: Courses + Labs ⇒ Slots + {\$}. The assignment *assign* your system produces has to fulfill the condition: *assign*(a) = *partassign*(a) for all a in Courses + Labs with *partassign*(a) not equal to \$.
  - The input for your system can contain a list of *unwanted*(a, s) statements, with a in Courses + Labs and s in Slots. For each of those, *assign*(a) has to be unequal to s.
- Department Hard Constraints:
  - If a course (course section) is put into a slot on Mondays, it has to be put into the corresponding time slots on Wednesdays and Fridays. So, these three time slots are

treated as one abstract slot, which allows us to see our Department problem as an instantiation of the general problem!

  - If a course (course section) is put into a slot on Tuesdays, it has to be put into the corresponding time slots on Thursdays.
  - If a lab/tutorial is put into a slot on Mondays, it has to be put into the corresponding time slots on Wednesdays.
  - If a lab/tutorial is put into a slot on Tuesdays, it has to be put into the corresponding time slots on Thursdays.
  - All course sections with a section number starting LEC 9 are evening classes and have to be scheduled into evening slots.
  - All courses (course sections) on the 500-level have to be scheduled into different time slots.
  - No courses can be scheduled on Tuesdays from 11:00 - 12:30.
  - There are two special "courses" CPSC 813 and CPSC 913 that have to be scheduled Tuesdays/Thursdays 18:00-19:00 and CPSC 813 is not allowed to overlap with any labs/tutorials of CPSC 313 or with any course section of CPSC 313 (and transitively with any other courses that are not allowed to overlap with CPSC 313) and CPSC 913 is not allowed to overlap with any labs/tutorials of CPSC 413 or with any course section of CPSC 413 (and transitively with any other courses that are not allowed to overlap with CPSC 413). These two "courses" are my "tricky" way to deal with the fact of quizzes for CPSC 313 and CPSC 413.

Soft Constraints:

- General Soft Constraints:
  - Since there are usually time slots that are less liked than others, there is a certain pressure to also put courses and labs into the more unwanted slots. To facilitate this pressure, we have for each slot s minimal numbers coursemin(s) and labmin(s) that indicate how many courses, resp. labs, should at least be scheduled into the slot s. Your system should be able to accept as input penalty values pen_coursemin and pen_labsmin (as system parameters) and for each course below coursemin we will get pen_coursemin and for each lab pen_labsmin added to the **Eval**-value of an assignment.
  - Certain professors that often teach certain courses have certain preferences regarding in which time slots their courses and labs should be scheduled. Naturally, we see this as something that should be treated as soft constraint. Depending on a to-be-determined ranking scheme, each professor will be awarded a certain set of ranking points and he/she can distribute these points over pairs of (course/lab, time slots). Formally, we assume a function preference: (Courses + Labs) x Slots ⇒ Natural numbers that reports those preferences. For each assignment in *assign*, we add up the preference-values for a course/lab that refer to a different slot as the penalty that is added to the **Eval**-value of *assign*.
  - For certain courses and/or labs, a department might know that there are never any students that take these courses/labs in the same semester. And therefore the department might find it convenient to have such courses/labs scheduled at the same time (this can also be used to keep students from taking certain courses prematurely). To facilitate this, there will be a list of *pair*(a, b) statements in the input for your system, with a, b in Courses + Labs, and a parameter pen_notpaired for your system. For every *pair*(a, b) statement, for which *assign*(a) is not equal to *assign*(b), you have to add pen_notpaired to the **Eval**-value of *assign*.
- Department Soft Constraints:
  - Different sections of a course should be scheduled at different times. For each pair of sections that is scheduled into the same slot, we add a penalty pen_section to the **Eval**-value of an assignment *assign*.

Additional Information:

At the University of Calgary, courses are identified by a department/program indicator, a course number and a section number. For example: CPSC 433 LEC 02. Labs/tutorials add to the course identification the lab/tutorial number. If a lab is intended to be associated with a specific section of the course, for example CPSC 433 LEC 02 TUT 01 (or LAB instead of TUT, **we will treat TUT and LAB as synonymous for this assignment**). If a lab is open to students from all sections of a course, the section number is dropped. For example: CPSC 433 TUT 01.

The available time slots depend on the day of the week and whether we look at lectures or labs/tutorials:

- On Mondays, Wednesdays, and Fridays, the slots available for lectures are: 8:00-9:00, 9:00-10:00, 10:00-11:00, 11:00-12:00, 12:00-13:00, 13:00-14:00, 14:00-15:00, 15:00-16:00, 16:00-17:00, 17:00-18:00, 18:00-19:00, 19:00-20:00, and 20:00-21:00.
- On Mondays, Tuesdays, Wednesdays, and Thursdays, the slots available for labs/tutorials are: 8:00-9:00, 9:00-10:00, 10:00-11:00, 11:00-12:00, 12:00-13:00, 13:00-14:00, 14:00-15:00, 15:00-16:00, 16:00-17:00, 17:00-18:00, 18:00-19:00, 19:00-20:00, and 20:00-21:00.
- On Fridays, the slots available for labs/tutorials are: 8:00-10:00, 10:00-12:00, 12:00-14:00, 14:00-16:00, 16:00-18:00, and 18:00-20:00.
- On Tuesdays and Thursdays, the slots available for lectures are: 8:00-9:30, 9:30-11:00, 11:00-12:30, 12:30-14:00, 14:00-15:30, 15:30-17:00, 17:00-18:30, and 18:30-20:00.
- All slots beginning at 18:00 or later are so-called evening slots.

Search Paradigm I — Set-Based Search

Set-Based Search Model

$A_{\text{Set}} = (S_{\text{Set}}, T_{\text{Set}})$:

F = set of facts

$\text{Ext} \subseteq \{A \rightarrow B \mid A, B \subseteq F\}$: set of extension rules

$S_{\text{Set}} \subseteq 2^F$ (power set of F)

$T_{\text{Set}} = \{(s, s') \mid \exists (A \rightarrow B) \in \text{Ext}$ such that $A \subseteq s$ and $s' = (s - A) \cup B\}$: transition relation between states

The Search Model defines the main data structure(s) and the search space. It tells us what the Search Control can work with and limits its choices.

Set-Based Search Process

$P_{\text{Set}} = (A_{\text{Set}}, K_{\text{Set}})$:

A_{Set} is the Search Model.

$K_{\text{Set}}(s) = s' \Rightarrow (s, s') \in T_{\text{Set}}$. (Search Control)

Where,

$A \rightarrow B \in \text{Ext}$

$A \subseteq s$

For all $A' \rightarrow B' \in \text{Ext}$ with $A' \subseteq s$, then: $f_{\text{Wert}}(A, B) \leq f_{\text{Wert}}(A', B')$

$A \rightarrow B = f_{\text{Select}}(\{A' \rightarrow B' \mid f_{\text{Wert}}(A', B') \leq f_{\text{Wert}}(A'', B'')\ \forall (A'' \rightarrow B'') \in \text{Ext}, A'' \subseteq s\})$

The Search Process defines how to deal with indeterminism of Search Model. It must deal with all states and all searches. Env is ignored for the purposes of the assignment — therefore it does not appear in any of the above formulae.

$f_{\text{Wert}}(A, B) = ???$: evaluates the transition from A to B and returns an integer value.

$f_{\text{Select}}(?, ?) = ???$: acts as a tie-breaker between all minimal values returned by f_{Wert}

Set-Based Search Instance

$\text{Ins}_{\text{Set}} = (s_0, G_{\text{Set}})$:

$s_0, s_{\text{Goal}} \in S_{\text{Set}}$ = Start state for the instance

$G_{\text{Set}}(s)$ = yes, if and only if $s_{\text{Goal}} \subseteq s$ or there is no extension rule applicable in s.

We need to define:

- F
- Ext
- f_{Wert}
- f_{Select}

Search Paradigm II — And-Tree-Based Search

And-Tree-Based Search Model

$A_\wedge = (S_\wedge, T_\wedge)$:

Prob = set of problem descriptions

Div $\subseteq$ Prob$^+$ = division relation for generating sub-problems out of problem instances

$S_\wedge \subseteq A_{\text{Tree}}$

$T_\wedge = \{(s_1, s_2) \mid s_1, s_2 \in S_\wedge \text{ and } \text{Erw}_\wedge(s_1, s_2)\}$

Where A_{tree} is recursively defined by:

(pr, sol) $\in A_{\text{Tree}}$ for pr $\in$ Prob, sol $\in$ {yes, ?}

(pr, sol, $b_1, \ldots, b_n$) $\in A_{\text{Tree}}$ for pr $\in$ Prob, sol $\in$ {yes, ?}, $b_1, \ldots, b_n \in A_{\text{Tree}}$

$\text{Erw}_\wedge$ is a relation on A_{Tree} defined by:

$\text{Erw}_\wedge$((pr, ?)) = (pr,yes) if pr is solved

$\text{Erw}_\wedge$((pr, ?)) = (pr, ?, (pr$_1$, ?), …, (pr$_n$, ?)) if Div(pr, pr$_1$, …, pr$_n$) holds

$\text{Erw}_\wedge$((pr, ?, $b_1, \ldots, b_n$)) = (pr, ?, b_1', …, b_n'), if for an i:

$\text{Erw}_\wedge(b_i, b_i')$ and $b_j = b_j'$ for $i \neq j$

$\text{Erw}^*_\wedge$ is ignored here, as back-tracking is unnecessary when considering the problem at hand.

And-Tree-Based Search Process

$P_\wedge = (A_\wedge, K_\wedge)$:

$A_\wedge$ = the Search Model

$K_\wedge$ = the Search Control

$K_\wedge$ uses two functions: f_{Leaf}, and f_{Trans}, which compare all leaves of the tree representing the state and select one, and select one of the transitions available to the selected leaf, respectively.

And-Tree-Based Search Instance

$\text{Ins}_\wedge = (s_0, G_\wedge)$:

s_0 = (pr, ?)

$G_\wedge(s)$ = yes, if and only if:

- s = (pr', yes) or,
- s = (pr', ?, $b_1, \ldots, b_n$), $G_\wedge(b_1) = \ldots = G_\wedge(b_n)$ = yes and the solutions to b_1, …, b_n are compatible with each other. Or,
- there is no possible transition that has not already been attempted and analyzed

We need to define:

- Prob
- Div
- f_{Leaf}
- f_{Trans}