Interactive Course Timetabling

Tomáš Müller*, Hana Rudová◊ and Keith Murray*

*Purdue University, West Lafayette, USA
◊Masaryk University, Brno, Czech Republic
Comprehensive university timetabling system

- used for generation of timetables at Purdue University (USA)
- course timetabling
- exam timetabling
- event scheduling
- student scheduling under development

Course timetabling

- decentralized problem solving
- about 70 problems of different characteristics and complexity
  - about 40,000 students
  - about 7,700 classes per term in total
  - about 1,000 classes in the largest problem
- automated computing of timetables
- interactive changes of generated timetables
Interactive Changes of Course Timetable
Changes with class "POL 101 Lec 3" are considered

<table>
<thead>
<tr>
<th>Score</th>
<th>Class</th>
<th>Date</th>
<th>Time</th>
<th>Room</th>
<th>Students</th>
</tr>
</thead>
<tbody>
<tr>
<td>+15.2</td>
<td>POL 101 Lec 3</td>
<td>Full Term</td>
<td>TTh 12:00p → TTh 7:30a</td>
<td>BRNG 2280</td>
<td>+11</td>
</tr>
<tr>
<td>+31.7</td>
<td>POL 101 Lec 3</td>
<td>Full Term</td>
<td>TTh 12:00p → TTh 10:30a</td>
<td>BRNG 2280</td>
<td>+36 (h+3)</td>
</tr>
<tr>
<td></td>
<td>HIST 342 Lec 1</td>
<td>Full Term</td>
<td>TTh 10:30a → TTh 1:30p</td>
<td>BRNG 2280 → BRNG 2290</td>
<td></td>
</tr>
<tr>
<td>+36.6</td>
<td>POL 101 Lec 3</td>
<td>Full Term</td>
<td>TTh 12:00p → TTh 10:30a</td>
<td>BRNG 2280</td>
<td>+36 (h+4)</td>
</tr>
<tr>
<td></td>
<td>HIST 342 Lec 1</td>
<td>Full Term</td>
<td>TTh 10:30a → TTh 7:30a</td>
<td>BRNG 2280</td>
<td></td>
</tr>
<tr>
<td>+44.1</td>
<td>POL 101 Lec 3</td>
<td>Full Term</td>
<td>TTh 12:00p → TTh 10:30a</td>
<td>BRNG 2280</td>
<td>+34 (h+2)</td>
</tr>
<tr>
<td></td>
<td>HIST 342 Lec 1</td>
<td>Full Term</td>
<td>TTh 10:30a → TTh 3:00p</td>
<td>BRNG 2280 → BRNG 2290</td>
<td></td>
</tr>
<tr>
<td></td>
<td>OBHR 330 Lec 4</td>
<td>Full Term</td>
<td>TTh 3:00p</td>
<td>BRNG 2290 → LWSN B155</td>
<td></td>
</tr>
</tbody>
</table>

(all 1571 possibilities up to 3 changes were considered, top 4 of 17 suggestions displayed)
Interaction Process: Variables

**Timetabling problem** $P$: weighted constraint satisfaction problem
- hard constraints must be satisfied
- soft constraints are satisfied to a certain degree/weight
- objective function $F$ summerizes weights of soft constraints

**Initial solution** $\delta$
- initial timetable of the interaction process
Interaction Process: Variables

Timetabling problem $P$: weighted constraint satisfaction problem
- hard constraints must be satisfied
- soft constraints are satisfied to a certain degree/weight
- objective function $F$ summarizes weights of soft constraints

Initial solution $\delta$
- initial timetable of the interaction process

Selected assignments $\mu$: changes made with the timetable $\delta$
during current interaction

Selected class $v$
- to modify its placement or to be placed into the timetable
Interaction Process: Variables

Timetabling problem $P$: weighted constraint satisfaction problem
- hard constraints must be satisfied
- soft constraints are satisfied to a certain degree/weight
- objective function $F$ summarizes weights of soft constraints

Initial solution $\delta$
- initial timetable of the interaction process

Selected assignments $\mu$: changes made with the timetable $\delta$
during current interaction

Selected class $\nu$
to modify its placement or to be placed into the timetable

Suggestions $\Omega$: set of generated assignments $\omega$
making the timetable feasible (all hard constraints are satisfied)

Conflicting assignments $\gamma$
set of assignments conflicting with selected assignments $\mu$
Simplified Interaction Process

procedure INTERACTION(P, δ, ν)

µ = ∅

A = \{ ν ≠ dν \}

while true do

Ω = BB(P ∪ A, δ, µ, ν)

end while

end procedure
Simplified Interaction Process

procedure INTERACTION($P, \delta, \nu$)

$\mu = \emptyset$

$A = \{ \nu \neq d_v \}$

while true do

$\Omega = \text{BB}(P \cup A, \delta, \mu, \nu)$

$S = \text{COMMUNICATION}(\Omega)$

end procedure
procedure INTERACTION(\(P, \delta, \nu\))

\(\mu = \emptyset\)

\(A = \{\nu \neq d_\nu\}\)

while true do

\(\Omega = \text{BB}(P \cup A, \delta, \mu, \nu)\)

\(S = \text{COMMUNICATION}(\Omega)\)

\text{case } (S)\text{ commit}(\omega \in \Omega): \delta = \text{join}(\delta, \mu \cup \omega); \text{return}

\text{abort}: \text{return}

\text{selectAssignment}(d_n): \mu = \mu \cup \{\nu/d_n\}

\text{selectFilter}(\alpha): A = \alpha \nu

\text{end case}

\text{end while}

end procedure
Simplified Interaction Process

procedure INTERACTION($P$, $\delta$, $v$)

$\mu = \emptyset$
$A = \{v \neq d_v\}$

while true do

$(\Omega, \gamma) = \text{BB}(P \cup A, \delta, \mu, v)$
$S = \text{COMMUNICATION}(\Omega, \gamma)$

case ($S$)

commit($\omega \in \Omega$): $\delta = \text{join}(\delta, \mu \cup \omega)$; return

abort: return

selectAssignment($d_n$): $\mu = \mu \cup \{v/d_n\}$

selectFilter($\alpha$): $A = \alpha v$

selectClass($c \in \{\mu \cup \gamma \cup \Omega\}$): $v = c$; $A = \{v \neq d_v\}$

removeClass($c \in \mu$): $\mu = \mu \backslash \{c/d_c\}$

end case

end while

end procedure
Branch and Bound (BB) \( \Omega = \text{BB}(P \cup A, \delta, \mu, \nu) \)

**Variables**
- weighted constraint satisfaction problem \( P \)
- filter \( A \)
- initial timetable \( \delta \)
- selected assignments \( \mu \)
- class to be (re-)placed \( \nu \)

**Initialization**
- compute conflicting assignment caused by \( \mu \)

**Run BB to find assignments of variables for**
- class \( \nu \)
- classes involved in conflicting assignments
Branch and Bound (continues)

Run BB

- $n$ best suggestions $\omega$ are given to user
- search with timeout
- best values (based on contribution to $F$) explored first

Bounds

- limited search depth
  - to allow changes of small number of variables only
  - to include changes of one new class it does make sense to change too many other classes
- $F$ must be better than the $n$-th best found suggestion
Branch and Bound (continues)

Run BB
- \( n \) best suggestions \( \omega \) are given to user
- search with timeout
- best values (based on contribution to \( F \)) explored first

Bounds
- limited search depth
  - to allow changes of small number of variables only
  - to include changes of one new class it does make sense to change too many other classes
- \( F \) must be better than the \( n \)-th best found suggestion

Repeat BB: process another run of BB with
- increased search depth or
- increased timeout
## Experiments

<table>
<thead>
<tr>
<th>Problem</th>
<th>pu-fal07-llr</th>
<th>pu-spr07-llr</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classes</td>
<td>891</td>
<td>803</td>
</tr>
<tr>
<td>Time limit (s)</td>
<td>– 5</td>
<td>– 5</td>
</tr>
<tr>
<td>Time spent (s)</td>
<td>128.6 4.7</td>
<td>39.9 4.2</td>
</tr>
<tr>
<td>Number of backtracks</td>
<td>66367.9 2886.9</td>
<td>13949.1 2592</td>
</tr>
<tr>
<td>Optimal suggestion found (%)</td>
<td>98.4 51.5</td>
<td>99.2 67.0</td>
</tr>
<tr>
<td>Improvements in objective function (%)</td>
<td>+1.1 +0.8</td>
<td>+0.9 +0.7</td>
</tr>
</tbody>
</table>
Demonstration

See http://www.unitime.org/uct_demo.php for online demo