Solving Clustered Oversubscription Problems for Planning e-Courses

Susana Fernández and Daniel Borrajo

Universidad Carlos III de Madrid. SPAIN
Index

1 Introduction

2 Translation

3 Pre-processing

4 Planning

5 Experiments

6 Conclusions and Future Work
Index

1. Introduction
2. Translation
3. Pre-processing
4. Planning
5. Experiments
6. Conclusions and Future Work
Index

1. Introduction
2. Translation
3. Pre-processing
4. Planning
5. Experiments
6. Conclusions and Future Work
Index

1 Introduction
2 Translation
3 Pre-processing
4 Planning
5 Experiments
6 Conclusions and Future Work
Index

1. Introduction
2. Translation
3. Pre-processing
4. Planning
5. Experiments
6. Conclusions and Future Work
Motivation

- Current planning:
  - Planning competition drives planning advancement
  - Very powerful domain-independent techniques, but focus on specific aspects of the planning task (tracks)

- Difficult to address real world applications:
  - requires the use of many features of PDDL, . . .
  - requires to compute metrics that use state-dependent fluents

- The application of planning techniques to real problems, sometimes, requires solving interesting associated problems that can be useful in more general contexts
Description

- Application area: the generation of learning designs adapted to students’ profiles
- Associated problem: a variation of OSP\(^1\) that we have called clustered oversubscription

\(^1\)Oversubscription Problem: Given a set of goals, each one with a utility, obtain a plan that achieves some (or all) the goals, maximizing the utility, as well as minimizing the cost of achieving those goals.
E-learning

**GOAL**

GENERATION OF LEARNING DESIGNS ADAPTED TO STUDENTS’ PROFILES

**PEDAGOGICAL THEORY THAT RELATES:**

\[
\text{Activities Reward} = \text{Felder’s learning styles} + <\text{learningsourcetype}>
\]

**Felder-Solomon Index of Learning Styles**

- **ACTIVE**
  - Learning facts
  - Sensing
- **REFLECTIVE**
  - Thinking about it quietly first.
  - Intuitive
- **VISUAL**
  - See—pictures, diagrams, flow charts, time lines, films, and demonstrations.
- **VERBAL**
  - Words—written and spoken explanations.
- **SEQUENTIAL**
  - Gain understanding in linear steps.
- **GLOBAL**
  - Learn in large jumps, suddenly "getting it."

**Solving Clustered Oversubscription Problems for Planning e-Courses**

- **Task1**
  - a11 time reward
  - a12 time reward
  - a1n time reward

- **Task2**
  - .........

- **Taskn**

**Course definition**

- IMS−MD
  - LO1
  - LO2
  - LON

**Time**

- a11
- a12
- a1n

**Reward**

- time
- reward

**IsBasedOn**

- Disjunction
- Conjunction

**I**

**M**

**S**

**IMS**
Clustered Oversubscription Problem

Causal relationships among activities

C1
- a11=<cost, utility>
- a12=<cost, utility>
- a1n=<cost, utility>

C2
- a21=<cost, utility>
- a2k=<cost, utility>

CM
- am1=<cost, utility>
- am2=<cost, utility>
- amn=<cost, utility>

Cost threshold, T

Solution: a1, a2, ..., ax
Plan / Learning Design

GOAL

Total cost < T
Maximize the total utility
At least one action of each cluster/ potentially more
Approach

IMS–MD Course

TRANSITION

PDDL Domain

PRE-PROCESSING

O={a1, a2,...,ai}
Maximize reward
Total time <= Threshold

PLANNING

Learning

design

PRE-PROCESSING
(Optimization Component)

1. Linear Programing
2. Heuristic Search

PLANNING
(Causal Component)

1. PDDL3 Preference goals
2. Plan metric
Learning Activities Actions

(:action simulates-strips-problem
 :parameters (?s - student)
 :precondition (and (task_reads-classical-planning_done ?s) (not (task_simulates-strips-problem_done ?s)))
 :effect (and (is-part-of planning)
 (task_simulates-strips-problem_done ?s
 (increase (reward_student ?s) 5)
 (increase (total_time_student ?s) 30)
 (when (active ?s strong)
 (increase (reward_student ?s) 30)))
 (when (sensitive ?s strong)
 (increase (reward_student ?s) 30)))
 (when (global ?s strong)
 (increase (reward_student ?s) 15))
 (when (visual ?s strong)
 (increase (reward_student ?s) 30))))
Modelling Actions

(:action fictitious-finish-ai-course
 :parameters (?s - student)
 :precondition
   (and (task_performs-test-introduction_done ?s)
        (task_performs-test-representation-others_done ?s)
        (task_performs-test-production-systems_done ?s)
        (task_performs-test-uninformed-search-unit-2_done ?s)
        (< (total_time_student ?s) (time_threshold_student ?s)))
 :effect (and (is-part-of course) (task_ai-course_done ?s)))

(:action OR-fictitious-strips
 :parameters (?s - student)
 :precondition (and (not (task_strips_done ?s))
                (or (task_simulates-strips-problem_done ?s)
                    (task_experiments-strips-problem_done ?s)))
 :effect (and (is-part-of planning) (task_strips_done ?s)))
Activities selection

- **Formalization:**
  - $\forall a \in A, a = \langle t, r \rangle$, the goal $O = \{a_1, \ldots, a_n\}, a_i \in A$, given $
  \sum a_i = \langle t_i, r_i \rangle \in O \quad t_i \leq T$, maximizing $\sum r_i$
  - Activities are grouped into a set of clusters, $C = \{c_1, \ldots, c_m\}$, $c_i = \{a_1, \ldots, a_{c_i}\}$ that can perform the same learning task.
  - $\forall c_i \in C$ at least one $a_j \in c_i$ should be in $O$
  - Similar to the well-known knapsack problem in combinatorial optimization, but with the addition of clusters

- **Solution:**
  - Using Linear Programming: optimal
  - Using hill-climbing algorithm with backtracking
Linear Programming

set A; /* list of activities*/
set T; /*list of tasks*/
param t{a in A}; /* time of each activity in A */
param r{a in A}; /* reward of each activity in A */
param c{a in A, j in T}, binary; /* activity i belongs to task j */
param tt; /* bound time */
var x{a in A}, binary;
maximize treward: sum{a in A} x[a]*r[a];
s.t. time: sum{a in A} x[a]*t[a] <= tt;
s.t. cluster{j in T}: sum{a in A} c[a,j]*x[a] >= 1;
/* there is at least one action per task*/
<table>
<thead>
<tr>
<th></th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
</tr>
<tr>
<td>2</td>
<td>Translation</td>
</tr>
<tr>
<td>3</td>
<td>Pre-processing</td>
</tr>
<tr>
<td>4</td>
<td>Planning</td>
</tr>
<tr>
<td>5</td>
<td>Experiments</td>
</tr>
<tr>
<td>6</td>
<td>Conclusions and Future Work</td>
</tr>
</tbody>
</table>
Modelling

1. Including the actions in $O$ as PDDL3 preference-goals:

\[
\text{:goal (and (preference p0 (<action-name1> st1))}
\]
\[
\text{(preference p1 (<action-name2> st1))}
\]
\[
\ldots
\]
\[
\text{(task_course_done student1 st1))}
\]

2. Using selection as plan metric:
   - **Domain:**
     - add conditional effects to the actions
       \[
       \text{(when (not (action-in-plan ?s <action-name>))}
       \]
       \[
       \text{(increase (penalty ?s) 1))}
       \]
     - Add precondition in end of course action:
       \[
       \text{(<= (reward_student ?s) (reward_threshold_student ?s))}
       \]
   - **Problem:**
     - Initial state: including actions in $O$ as action-in-plan predicates
     - Metric: (\text{:metric (minimize (penalty student1)))}
Index

1. Introduction
2. Translation
3. Pre-processing
4. Planning
5. Experiments
6. Conclusions and Future Work
Computing Set $O$

Reward when time limit is the sum of the time of the highest-time activity in each cluster

(LP always found the solution in less than 0.1s while the search-algorithm execution time steadily increased from 0.1 up to 8s)
Computing Set $O$. All clusters

Reward of 52 clusters when time limit varies from -20% up to 20%

(The execution time was never higher than 18s)
Planning Results. Configurations

1. **EHC**: original Enforced Hill-climbing algorithm in Metric-FF
2. **CBP-BFS**: CBP planner with BFSearch+Lookahead algorithm
   - Time: minimizing the \((total\_time\_student)\)
   - LP: minimizing \((penalty\_student)\). LP selection
   - Hill: minimizing \((penalty\_student)\). Hill-climbing selection
3. **SGPlan6**: SGPlan6 planner
   - Without preference goals
   - LP: preferences. LP selection (unfeasible plans)
   - Hill: preferences. Hill-climbing selection (unfeasible plans)
Planning Results. Reward

- The diagram illustrates the Reward over Time increments (%). The x-axis represents time increments in percentage, and the y-axis represents reward.
- Several methods are compared, including CBP-BFS, HILL, LPEHC, and TIME, among others.
- The graph shows how each method performs over time, with different markers indicating different algorithms.
- The performance varies, with some methods consistently outperforming others as indicated by the trend lines and markers.

---

Solving Clustered Oversubscription Problems for Planning e-Courses
Planning Results. Time

![Graph showing planning results over time for different methods: EHC, CBP-BFS TIME, CBP-BFS HILL, CBP-BFS LP. The x-axis represents time increments (%), and the y-axis represents time in minutes. The graph compares the performance of these methods over time, indicating CBP-BFS LP performs better than the others.]
Planning Results. Both

![Graph showing planning results for different methods]

- EHC
- CBP–BFS TIME
- CBP–BFS HILL
- CBP–BFS LP

Solving Clustered Oversubscription Problems for Planning e-Courses
Conclusions

- E-learning planning application for generating learning designs adapted to different students’ profiles
- Modelled as a \textit{clustered-oversubscription problem}
- Hybrid approach:
  - LP/Heuristic search solves the optimization component
  - Planning solves the causal component:
- Integration:
  - PDDL3 preference-goals (\texttt{SGPLAN6} unfeasible plans)
  - As plan metric: \texttt{CBP}
    \texttt{(penalty, action-in-plan, reward\_threshold\_student)}
Future Work

- Test the approach in other domains
- Include causal relations in the LP model (without OR relations)