Solving Clustered Oversubscription Problems for Planning e-Courses

Susana Fernández and Daniel Borrajo

Universidad Carlos III de Madrid. SPAIN

Index

Index

Index

Index

Index

Index

Index

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

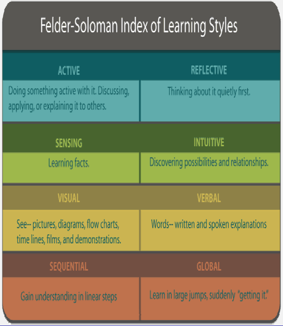
[1] Oversubscription Problem: Given a set of goals, each one with a utility, obtain a plan that achieves some (or all) the goals, maximizing the utility, as well as minimizing the cost of achieving those goals.

E-learning

IMS

LO
<relations> IsBasedOn Disjuction
Requires Conjunction
<time>
<learningsourcetype>
IMS–MD
Course definition

LO1
LO2
....
LON



PEDAGOGICAL THEORY THAT RELATES:

<learningsourcetype>
+
Felder's learning styles
=
Activities Reward

GOAL

GENERATION OF LEARNING DESIGNS ADAPTED TO STUDENTS' PROFILES

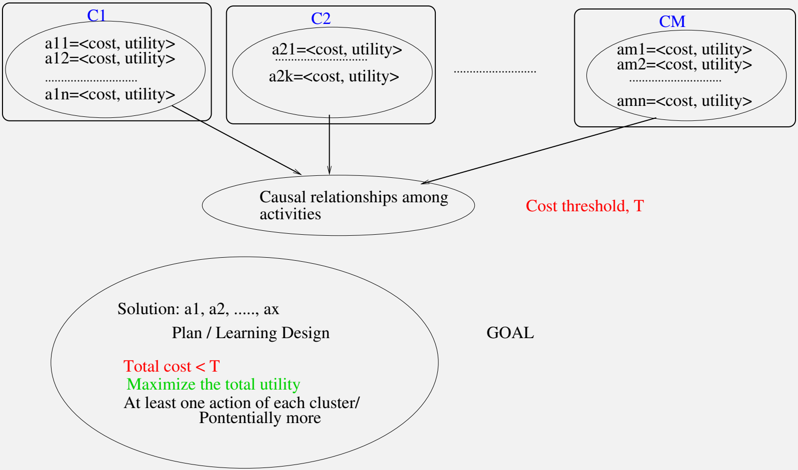
Task1

a11	time	reward
a12	time	reward
a1n	time	reward

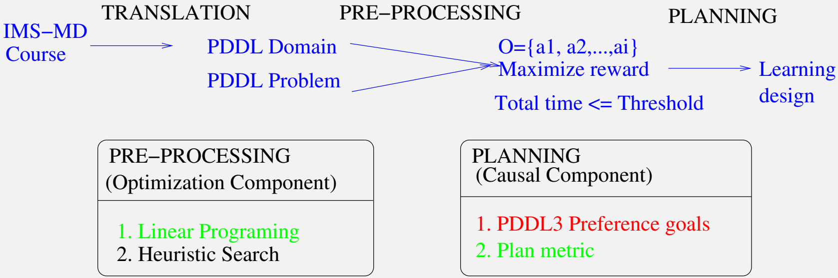
IsBasedOn

Task2
....
Taskn

Clustered Oversubscription Problem



Approach



IMS–MD Course
TRANSLATION
PDDL Domain
PDDL Problem
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

Index

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))))
```

Each LO into an action

Predicates (Requires)

Fluents

Conditional effects (Felner styles)

Modelling Actions

End of course action

```
(: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)))
```

Time constraint

IsBasedOn relation

```
(: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)))
```

Index

Activities selection

- Formalization:
  - $\forall a \in A, a =< t, r >$, the goal $O = \{a_1, \ldots, a_n\}, a_i \in A$, given $\sum_{a_i=<t_i,r_i>\in O} t_i \leq T$, 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 belows 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*/
```

Index

Modelling

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

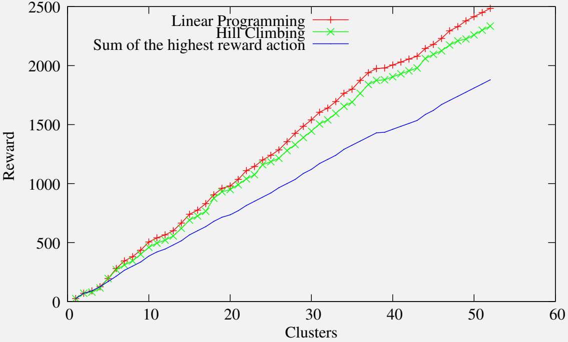
   (:goal (and (preference p0 (<action-name1> st1))
   (preference p1 (<action-name2> st1))
   ...
   (task_course_done student1 st1)))

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

Index

Computing Set O

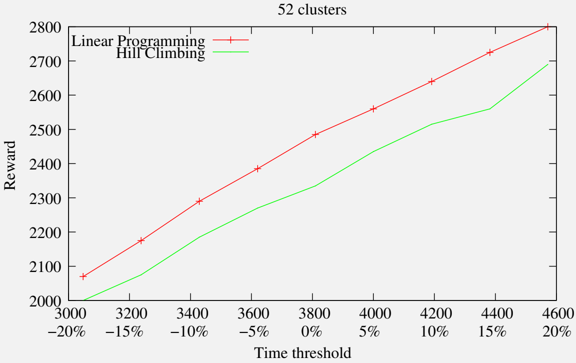
Reward when time limit is the sum of the time of the highest-time activity in each cluster



(LP always found the solution in less that 0.1s while the search-algorithm execution time steady 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 %

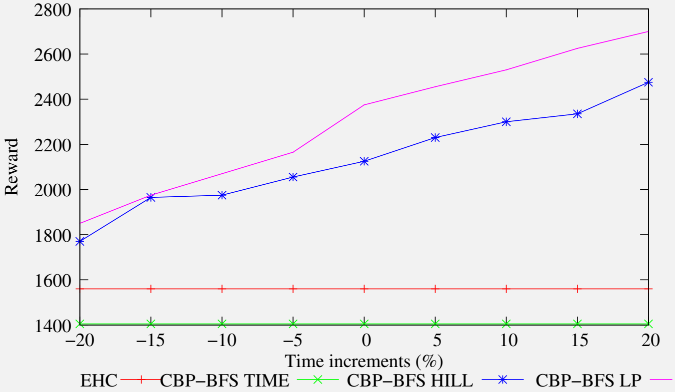


(The execution time was never higher than 18s)

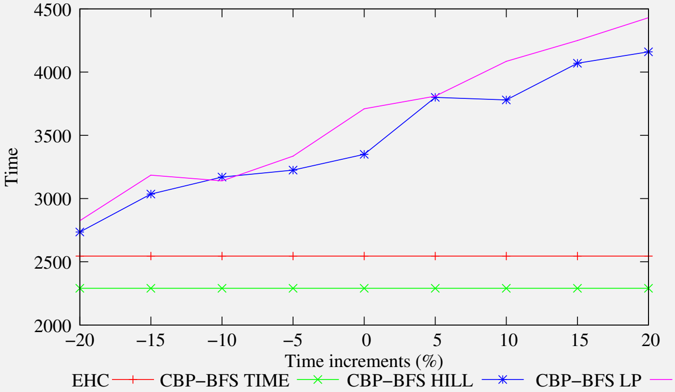
Planning Results. Configurations

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. Hil-climbing selection (unfeasible plans)

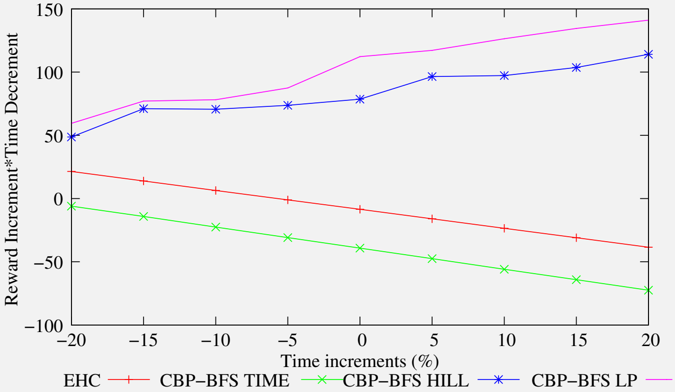
Planning Results. Reward



Planning Results. Time



Planning Results. Both



Index

Conclusions

- E-learning planning application for generating learning designs adapted to different students' profiles
- Modelled as a *clustered-oversubscription problem*
- Hybrid approach:
  - LP/Heuristic search solves the optimization component
  - Planning solves the causal component:
- Integration:
  - PDDL3 preference-goals (SGPLAN6 unfeasible plans)
  - As plan metric: CBP (*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)