Ottimizzazione dei Garanti accademici

Colli Simone[1] and Merenda Saverio Mattia[1]

[1] simone.colli@studenti.unipr.it
[2] saveriomattia.merenda@studenti.unipr.it

December 19, 2024

Abstract

Questo lavoro presenta l'analisi e l'implementazione di un sistema automatizzato per l'assegnazione dei garanti ai corsi universitari, in conformità ai requisiti ministeriali. L'obiettivo principale è garantire che ogni corso soddisfi i vincoli minimi di docenza, rispettando le regole di distribuzione tra diverse categorie di docenti e ottimizzando l'uso delle risorse disponibili.

Utilizzando la programmazione logica con Answer Set Programming (ASP), abbiamo modellato il problema attraverso fatti, regole e vincoli derivati dai dati ministeriali e universitari. Abbiamo implementato una serie di vincoli per rispettare i minimi richiesti di docenti per corso, evitando sovrapposizioni improprie tra gli incarichi dei docenti e considerando scenari realistici in cui un docente può assumere più ruoli parziali.

L'approccio è stato testato su un dataset reale contenente informazioni su corsi, SSD (Settori Scientifico Disciplinari) e docenti dell'Università degli Studi di Parma. I risultati dimostrano come il sistema possa trovare configurazioni ottimali che soddisfano i requisiti, massimizzando l'efficienza e mantenendo flessibilità nell'assegnazione dei docenti.

1 Introduzione

L'assegnazione dei garanti nei corsi universitari costituisce una questione fondamentale per la gestione ottimale delle risorse accademiche. Nell'ambito universitario, il *garante* è un docente responsabile di rappresentare e tutelare la qualità didattica di un corso, garantendo il rispetto dei requisiti disciplinari e istituzionali. I garanti possono appartenere a diverse categorie contrattuali: docenti a tempo indeterminato, docenti a tempo determinato e, in casi eccezionali, docenti a contratto. La sfida principale consiste nel soddisfare i vincoli ministeriali relativi ai garanti, garantendo al contempo un'allocazione equilibrata e sostenibile delle risorse. Ogni corso deve essere supportato da un numero minimo di garanti, suddivisi tra le diverse fasce contrattuali, per assicurare un livello adeguato di competenza e rappresentatività. Inoltre, è indispensabile che almeno il 50% dei garanti afferisca al Settore Scientifico Disciplinare (SSD) caratterizzante del corso, al fine di garantire la coerenza tra l'offerta formativa e le competenze disciplinari.

Un'ulteriore complessità è rappresentata dall'impiego di docenti a contratto, il cui utilizzo deve essere limitato e subordinato alle sole situazioni in cui non sia possibile soddisfare i requisiti attraverso i docenti strutturati. La necessità di rispettare questi vincoli, combinata con la disponibilità limitata di personale e la necessità di bilanciare il carico di lavoro, rende questo problema una sfida organizzativa e computazionale significativa.

Un primo ostacolo affrontato in questo progetto riguarda la fase di pre-elaborazione dei dati forniti dall'università, descritta nella Sezione 2 di questo elaborato. I dati risultano disomogenei e incompleti, richiedendo una significativa pulizia e riorganizzazione prima di poter essere utilizzati efficacemente nel modello ASP. Questa fase ha richiesto lo sviluppo di strumenti dedicati per uniformare e validare i dati.

La Sezione 3 esplora il cuore del nostro approccio, ovvero la costruzione del modello ASP. Qui abbiamo implementato regole logiche per soddisfare i vincoli ministeriali e massimizzare l'efficienza dell'assegnazione dei garanti, tenendo conto delle limitazioni sulle fasce contrattuali e sull'impiego dei docenti a contratto.

Successivamente, nella Sezione 4, presentiamo i risultati ottenuti su un esempio ridotto, un test che ha permesso di validare il modello in un ambiente controllato e di analizzare la qualità delle soluzioni generate. Infine, la Sezione 4.2 descrive l'applicazione del modello su un dataset completo contenente tutti i corsi universitari, eseguendo un benchmark su larga scala per valutare la capacità del nostro approccio di risolvere problemi reali e complessi.

2 Preprocessing dei dati

I dati analizzati, forniti dall'ufficio didattico dell'Università di Parma, erano organizzati in formato Excel con una struttura tabellare complessa, contenente numerosi campi. Per garantire un'analisi efficace e accurata, si è resa necessaria una fase di preprocessing, finalizzata a correggere eventuali incoerenze, migliorare la qualità complessiva del dataset e suddividerlo in sottoinsiemi più gestibili.

Il dataset fornito includeva informazioni relative ai docenti, ai ricercatori e alle coperture degli insegnamenti nei diversi corsi di laurea. Dopo un'attenta analisi preliminare, sono stati selezionati esclusivamente i campi considerati rilevanti per l'analisi. Nel file contenente i dati di docenti e ricercatori, sono state prese in considerazione solo le informazioni riguardanti la matricola, la fascia contrattuale e l'SSD. Per quanto riguarda il file sulle coperture, sono stati mantenuti i campi relativi alla matricola del docente, al codice del tipo di corso, al codice di laurea, all'SSD e alla Tipologia di Attività Formativa (TAF). Questa selezione dei dati è stata essenziale per ridurre la complessità e concentrarsi sugli elementi chiave necessari per costruire il modello ASP e verificare il rispetto dei vincoli ministeriali. Tutto questo è stato realizzato utilizzando diversi script in Python, supportati dai moduli `pandas`[1] e `openpyxl`[2].

La maggior parte delle operazioni di preprocessing ha riguardato il file contenente le coperture degli insegnamenti, articolandosi in diverse attività principali. In primo luogo, abbiamo gestito le righe contenenti una o più colonne vuote, eliminando quelle completamente prive di informazioni, in quanto non rilevanti per l'analisi. Le righe in cui mancavano i dati relativi al docente (i.e., matricola, nome e cognome) sono state invece spostate in un file separato. Questa scelta è stata necessaria per identificare e analizzare in modo specifico gli insegnamenti non ancora assegnati ad alcun docente.

Un ulteriore passo fondamentale ha riguardato la scomposizione del dataset originale, suddividendolo in sottoinsiemi più piccoli e gestibili. Questa operazione ha permesso di affrontare con maggiore precisione i singoli aspetti del problema, semplificando il successivo trattamento dei dati. Infine, sono stati identificati e gestiti in modo univoco e automatizzato i casi particolari, consentendo di creare un dataset uniforme e privo di anomalie, adatto all'analisi e all'implementazione del modello ASP.

Per migliorare ulteriormente la gestibilità dei dati, abbiamo proceduto con la creazione di sottoinsiemi minimali e distinti. Il primo passo riguarda la separazione delle righe relative agli insegnamenti tenuti da docenti o ricercatori da quelle tenute dai docenti a contratto. Questa suddivisione è stata effettuata utilizzando il file dei docenti fornito, sfruttando la colonna contenente la matricola come criterio discriminante. In particolare, le righe associate ai docenti a contratto sono state estratte e archiviate in un file separato, al fine di garantire un maggiore ordine e facilitare eventuali ispezioni manuali. Parallelamente, le righe relative ai docenti a tempo indeterminato e determinato sono state salvate anch'esse in un altro file, per gli stessi identici motivi.

Una volta completata la suddivisione, è stata apportata una modifica ai valori nella colonna relativa al codice del tipo di corso di laurea. In particolare, tutte le occorrenze del valore `L` sono state sostituite con `LT`, così da esplicitare i corsi di laurea triennale e migliorare la chiarezza dei dati. Successivamente, abbiamo proceduto con l'identificazione dei casi particolari utilizzando un controllo incrociato tra due colonne specifiche. Questo processo ha permesso di individuare e gestire in modo univoco i corsi di laurea che presentano requisiti particolari, come illustrato nella Tabella I. Per implementare questa logica, abbiamo adattato il processo di modifica dei dati al corso specifico da verificare. Ad esempio, per la Laurea Triennale in Servizio Sociale, abbiamo sostituito il codice del tipo di corso precedentemente identificato come `LT` con `LTSS`.

3 Answer Set Programming

In questa sezione approfondiamo l'organizzazione del progetto, evidenziando come sia stato strutturato per garantire chiarezza e modularità. Per facilitare l'ordine visivo e semplificare eventuali ispezioni manuali, il progetto è stato suddiviso in più file sorgenti, ciascuno dedicato a un aspetto specifico del dataset e delle regole.

Abbiamo separato i fatti e le regole in due file principali: uno dedicato ai docenti (Sezione 3.1) e un altro focalizzato sulle coperture (Sezione 3.2). Per maggiore praticità, è stato creato un file specifico per i docenti a contratto (Sezione 3.3) e un altro per i limiti ministeriali (Sezione 3.4).

Il file principale, il main, integra tutte le regole relative ai garanti, inclusi i vincoli e le priorità definite per l'ottimizzazione del sistema (Sezione 3.5).

[1] `https://pandas.pydata.org/`

[2] `https://openpyxl.readthedocs.io/en/stable/`

Corsi di studio	Docenza di riferimento (**minimo**)	Prof. a tempo indeterm. (**minimo**)	Ricercatori (**massimo**)	Docenti a contratto (**massimo**)
LT	9	5	4	2
LM	6	4	2	1
LMCU 5 anni	15	8	7	3
LMCU 6 anni	18	10	8	4
LT Servizio Sociale LT Scienze Motorie	5	3	2	1
LT Prof. sanitarie LT a orient. profess. LM Servizio Sociale LM Scienze Motorie	4	2	2	1
LM Infermieristica	3	1	2	1

Table I: Tabella che riassume i numeri di docenti per i corsi di studio.

È importante sottolineare che tutti questi file vengono generati dinamicamente dal programma di preprocessing scritto in Python. Questa fase, gestita dal file `main.py` del preprocessing, automatizza la creazione delle regole e dei fatti ASP in base ai dati forniti, garantendo così una configurazione personalizzata e facilmente aggiornabile.

3.1 Docenti

Il file `docenti.lp` contiene una rappresentazione strutturata delle informazioni relative ai docenti, organizzate in diverse sezioni per garantire chiarezza e modularità. Questo file serve come base per definire le caratteristiche dei docenti e il loro SSD.

La prima parte del file definisce gli SSD disponibili, ognuno rappresentato da una coppia di valori: la disciplina e il livello associato (e.g., INF/01, MAT/03). Gli SSD sono utilizzati per verificare la compatibilità tra i docenti e i corsi di laurea.

Successivamente, vengono dichiarate le fasce contrattuali dei docenti (`td` per tempo determinato, `ti` per tempo indeterminato). Queste fasce sono utilizzate per distinguere i docenti in base alla loro tipologia di contratto, aspetto rilevante per rispettare i requisiti ministeriali.

La sezione relativa ai docenti include l'elenco completo dei docenti, ciascuno identificato da una matricola unica. Per ogni docente, vengono indicate la fascia contrattuale e il settore scientifico disciplinare caratterizzante, che ne definisce il dominio di competenza. Queste informazioni sono formalizzate tramite il predicato `docente/3`, che associa la matricola del docente, la fascia, il settore disciplinare e il livello SSD.

```
%  SEZIONE: SSD
ssd(inf).

%  SEZIONE: FASCIE
fascia(td).
fascia(ti).

%  SEZIONE: Docenti
%  Rossi Mario (1234), SSD caratterizzante: inf
matricola_docente(1234).

%  SEZIONE: SSD caratterizzante dei docenti
%  Rossi Mario (1234), SSD caratterizzante: inf
docente(1234, td, inf) :- matricola_docente(1234), fascia(td), ssd(inf).
```

Listing 1: Esempio struttura dati di `docenti.lp`.

3.2 Coperture

Il file `coperture.lp` è fondamentale per rappresentare la struttura dei corsi universitari, le informazioni sui docenti assegnati e le caratteristiche associate a ciascun corso. Questo file contiene

diverse sezioni organizzate per garantire chiarezza e modularità nella definizione delle informazioni.

La prima parte del file introduce i tipi di corso (`laurea/1`) e i relativi SSD utilizzati per caratterizzare i corsi e i docenti. Gli SSD caratterizzanti di un corso vengono estratti dinamicamente dagli SSD degli insegnamenti di quel corso che sono classificati come TAF A. Questa scelta consente di identificare in modo coerente i macrosettori scientifico-disciplinari fondamentali per ciascun corso, basandosi sui suoi insegnamenti di base. Successivamente, vengono definiti i TAF che classificano ulteriormente gli insegnamenti.

Le informazioni sui corsi sono strutturate in due livelli: i codici identificativi di ciascun corso (`codice_corso/1`) e la loro descrizione dettagliata tramite il predicato `corso/3`. Quest'ultimo associa il codice del corso, il tipo di laurea, il SSD caratterizzante e il livello del settore disciplinare.

Un'altra parte importante del file riguarda la relazione tra corsi e docenti, formalizzata tramite il predicato `cattedra/4`. Questo predicato associa un docente (identificato tramite la matricola) a un corso specifico, includendo il tipo di laurea e il TAF relativo.

Di seguito, un esempio rappresentativo del contenuto del file.

```
%  SEZIONE: Tipi di Corso
laurea(lt).
laurea(lm).

%  SEZIONE: TAF
taf(a).
taf(b).
taf(c).

%  SEZIONE: Corsi
%  INFORMATICA (3027)
codice_corso(3027).

%  SEZIONE: Informazioni Corsi
corso(3027, lt, fis) :- codice_corso(3027), laurea(lt), ssd(fis).
corso(3027, lt, mat) :- codice_corso(3027), laurea(lt), ssd(mat).
corso(3027, lt, inf) :- codice_corso(3027), laurea(lt), ssd(inf).

%  SEZIONE: Relazioni Corsi-Docenti
%  Corso: 3027, Docente: Rossi Mario
cattedra(3027, 1234, lt, b) :- codice_corso(3027),
                               matricola_docente(1234),
                               laurea(lt), taf(b).
```

Listing 2: Esempio struttura dati di `coperture.lp`.

3.3 Docenti a contratto

Il file `docenti_a_contratto.lp` contiene informazioni relative ai docenti a contratto, una categoria utilizzata come risorsa di emergenza nei casi in cui i vincoli ministeriali non possono essere soddisfatti con i docenti a tempo determinato o indeterminato. Questi docenti sono identificati dalla fascia `c` e vengono trattati come *jolly*, ovvero risorse flessibili da impiegare per garantire la copertura minima dei corsi universitari.

Essendo considerati come *jolly*, i docenti a contratto non sono associati a un SSD specifico o a un corso particolare, ma possono essere assegnati liberamente per colmare le lacune nei corsi che non soddisfano i requisiti minimi con i docenti delle altre fasce. Il file `docenti_a_contratto.lp` è strutturato in modo minimale, definendo la fascia `c` e un unico predicato `jolly/1`, che indica la possibilità di assegnare tali docenti in modo trasversale a qualsiasi corso.

Nel modello ASP proposto, i docenti a contratto vengono inclusi come possibili garanti, ma il loro utilizzo è regolato e penalizzato nel processo di ottimizzazione, che vedremo meglio nella Sezione 3.7.

```
%  SEZIONE: FASCIE
fascia(c).

jolly(1..5).
```

Listing 3: Esempio struttura dati di `docenti_a_contratto.lp`.

3.4 Limiti ministeriali

Il file `ministeriale.lp` contiene i dati relativi ai requisiti ministeriali per ogni corso di laurea, con particolare riferimento al numero minimo di garanti richiesti per ciascun corso. Ogni corso

ha associato un insieme di valori che determinano i vincoli di assegnazione dei docenti, inclusi i docenti a tempo indeterminato e tempo determinato, e il numero massimo di docenti a contratto.

Le regole `ministeriale/5` sono calcolate dinamicamente durante la fase di preprocessing. In particolare, questi valori sono determinati sulla base del numero di studenti iscritti a ciascun corso e grazie all'applicazione della formula della W, come illustrato nella Tabella II, la quale tiene conto delle specifiche necessità di copertura di ciascun corso. Nello specifico, la W viene calcolata nel seguente modo:

$$W = \frac{\text{Studenti iscritti al corso}}{\text{Massimo teorico di iscritti al corso}} - 1$$

Di seguito un esempio della struttura del file per il corso di Informatica, per il quale sono richiesti almeno 9 garanti, di cui 5 a tempo indeterminato, 4 a tempo determinato, e un massimo di 2 docenti a contratto.

```
%  SEZIONE: Garanti minimi per corso (codice_corso, minimo_complessivo,
%                                     docenti_ti, docenti_td,
%                                     max_docenti_contratto)
ministeriale(3027, 9, 5, 4, 2).
```

Listing 4: Esempio struttura dati di `ministeriale.lp`.

Qualifica	**# base**	**# effettivo**
Docenza di riferimento	9	$\lfloor 9 \times (1+w) \rfloor$
Docenti a tempo indeterminato	5	$\lfloor 5 \times (1+w) \rfloor$
Docenti a contratto	2	$\lfloor 2 \times (1+w) \rfloor$

Table II: Formule per il calcolo del numero di docenti di riferimento in base alla numerosità degli studenti.

3.5 Generazione dei garanti

La generazione dei *garanti* rappresenta un elemento chiave nella modellazione ASP sviluppata per questo progetto. In particolare, è stata posta attenzione alla distinzione tra i docenti appartenenti a diverse fasce contrattuali, separando i docenti **non a contratto** da quelli **a contratto**, i quali vengono utilizzati esclusivamente come risorsa di emergenza. Inoltre, ogni docente può essere conteggiato una sola volta oppure, al massimo, essere assegnato a due corsi distinti con un peso pari a 0.5 su ciascun corso.

Per soddisfare quest'ultimo requisito, il modello ASP proposto utilizza una gestione basata sui pesi, i quali rappresentano il livello di impegno del docente come garante:

- Un peso di **10** indica che il docente è assegnato al 100% su un singolo corso.
- Un peso di **5** rappresenta un docente che divide il proprio impegno al 50% su due corsi diversi.

```
% Predicati di base per definire i pesi possibili.
peso(5).
peso(10).
```

Listing 5: Generazione dei pesi.

Docenti non a contratto. I docenti *non a contratto*, appartenenti alle fasce *tempo determinato* (`td`) e *tempo indeterminato* (`ti`), costituiscono la base principale per l'assegnazione dei garanti. La regola ASP proposta genera i possibili garanti per ciascun corso, escludendo esplicitamente i docenti a contratto (`c`).

```
Minimo{
      garante(Docente, Corso, Peso, Fascia) :
            peso(Peso),
            fascia(Fascia),
            Fascia != c,

            % Docente 1
            cattedra(Corso, Docente, _, _),
            docente(Docente, Fascia, _),
```

```
10
11              % Docente 2
12              cattedra(Corso, Docente2, _, _),
13              docente(Docente2, Fascia, _),
14
15              Docente >= Docente2
16  }Massimo :-
17        ministeriale(Corso, MinimoComplessivo, _, _, MassimoDocentiContratto),
18        Minimo = MinimoComplessivo - MassimoDocentiContratto,
19        Massimo = MinimoComplessivo.
```

Listing 6: Generazione dei garanti non a contratto.

In questa regola, il numero minimo di garanti non a contratto per ciascun corso (`Minimo`) viene calcolato sottraendo il numero massimo di docenti a contratto consentiti dal numero minimo complessivo di garanti richiesto per il corso. Questo approccio permette di definire un sottoinsieme minimo di garanti che devono necessariamente appartenere alle fasce `td` e `ti`.

Il numero massimo di garanti non a contratto (`Massimo`), invece, coincide con il valore minimo dei garanti richiesti per quel corso, garantendo che la somma totale dei garanti sia conforme ai requisiti ministeriali. Inoltre, è stata fissata una relazione d'ordine sul numero di matricola (`Docente >= Docente2`): in questo modo, si vanno ad eliminare le possibili permutazioni in fase di generazione.

Docenti a contratto. I docenti *a contratto* vengono considerati come una risorsa ausiliaria, identificati tramite il predicato `jolly/1`. A differenza dei docenti non a contratto, i docenti a contratto possono essere assegnati a qualsiasi corso, indipendentemente dall'SSD. Tuttavia, il loro utilizzo è regolato da un limite massimo per corso, specificato dalla normativa ministeriale, come mostrato nel frammento di codice seguente.

```
1  {
2        garante(Docente, Corso, 10, c) :
3              jolly(Docente),
4              codice_corso(Corso),
5              peso(10),
6
7              jolly(Docente2),
8              codice_corso(Corso),
9              peso(10),
10
11             Docente > Docente2
12 }Massimo :- ministeriale(Corso, _, _, _, Massimo).
```

Listing 7: Generazione dei garanti a contratto.

Questa regola garantisce che il numero di docenti a contratto utilizzati come garanti non ecceda il valore massimo consentito, specificato dal valore `Massimo` associato al corso. Anche in questo caso, viene utilizzata una relazione d'ordine in fase di generazione.

3.6 Vincoli

I vincoli definiti in questo modello costituiscono la struttura portante del sistema, garantendo che le soluzioni generate siano coerenti con i requisiti ministeriali e istituzionali. Tutti i vincoli introdotti sono **strong**, il che significa che ogni soluzione che li viola viene automaticamente esclusa.

Per garantire che ogni corso soddisfi i requisiti ministeriali, il numero di docenti assegnati come garanti è soggetto a controlli rigorosi. Innanzitutto, i garanti a tempo indeterminato (`ti`) devono essere almeno pari al minimo richiesto per ciascun corso. Allo stesso modo, il numero di garanti a tempo determinato (`td`) non deve superare il massimo consentito. Questi vincoli assicurano che il modello rispetti le linee guida istituzionali senza sovraccaricare un'unica tipologia di docenti.

```
1  :- conta_garanti_indeterminato(Corso, Numero),
2        ministeriale(Corso, Minimo, Minimo_ind, _, _),
3        Numero < Minimo_ind.
4
5  :- conta_garanti_determinato(Corso, Numero),
6        ministeriale(Corso, _, _, Massimo_det, _),
7        Numero > Massimo_det.
```

Listing 8: Vincoli su garanti a tempo indeterminato e determinato.

Il modello include vincoli che controllano i pesi assegnati ai garanti. La somma totale dei pesi per ciascun corso deve essere almeno 10 volte il numero minimo di garanti richiesti, poiché un

docente è considerato *effettivo* solo se il suo peso totale raggiunge il valore massimo di 10. Inoltre, tale somma deve essere un multiplo di 10 per garantire la coerenza nell'assegnazione del garante stesso.

```
:- somma_pesi_corso(Corso, Somma),
     ministeriale(Corso, Minimo, _, _, _),
     Min = 10 * Minimo,
     Somma < Min.

:- somma_pesi_corso(Corso, Somma),
     Modulo = Somma \ 10,
     Modulo != 0.
```

Listing 9: Vincoli sui pesi assegnati ai garanti.

Per evitare conflitti e garantire una distribuzione equa del carico di lavoro, ogni docente può essere associato ad un massimo di due corsi, ma con una somma totale di pesi non superiore a 10. Inoltre, un docente non può essere associato allo stesso corso con pesi differenti che, sommati, superano 10.

```
:- garante(Docente, Corso1, Peso1, Fascia),
     garante(Docente, Corso2, Peso2, Fascia),
     Somma = Peso1 + Peso2,
     Somma > 10,
     Fascia != c,
     Corso1 != Corso2.

:- garante(Docente, Corso1, Peso1, Fascia),
     garante(Docente, Corso1, Peso2, Fascia),
     Somma = Peso1 + Peso2,
     Somma > 10,
     Fascia != c,
     Peso1 != Peso2.
```

Listing 10: Vincoli sulla distribuzione dei pesi per docente.

Per mantenere la coerenza tra i garanti e gli SSD caratterizzanti di ciascun corso, almeno il 50% dei garanti deve afferire all'SSD del corso. In caso contrario, la soluzione viene esclusa.

```
:- numero_garanti_che_afferiscono(Aff, Corso),
     numero_garanti_di_riferimento(Tot, Corso),
     2 * Aff <= Tot.
```

Listing 11: Vincolo sui garanti afferenti al SSD caratterizzante.

Infine, il numero totale di docenti a contratto assegnati a ciascun corso non può superare il limite massimo specificato dai requisiti ministeriali.

```
:- jolly_per_corso(Corso, Numero),
     ministeriale(Corso, _, _, _, Massimo_numero_docenti_a_contratto),
     Numero > Massimo_numero_docenti_a_contratto.
```

Listing 12: Vincolo sul numero massimo di docenti a contratto.

3.7 Gestione delle priorità

La gestione delle priorità è stata un aspetto fondamentale nel nostro progetto, poiché miravamo a rispettare scrupolosamente i vincoli ministeriali, garantendo al contempo flessibilità per gestire situazioni particolari in cui tali vincoli non fossero pienamente soddisfacibili.

Per organizzare le priorità dei garanti, abbiamo deciso di basarci, in seguito ad un colloquio con l'ufficio didattico, principalmente sulla loro tipologia contrattuale. Il nostro obiettivo era quello di massimizzare l'impiego di docenti a tempo determinato (i.e., ricercatori) e a tempo indeterminato (i.e., professori associati e ordinari), prima di ricorrere ai docenti a contratto, che sono utilizzati solo quando strettamente necessario. Per ottimizzare la gestione, abbiamo preferito che i docenti assumessero il ruolo di garanti per un solo corso, evitando situazioni in cui un docente fosse impegnato come garante per più corsi contemporaneamente al 50%.

```
% Docenti a tempo determinato (ricercatori)
#maximize { 50 : garante(_, _, _, td) }.

% Docenti a tempo indeterminato
#maximize { 40 : garante(_, _, _, ti) }.

% Docenti a contratto
```

```
#maximize { 32 : garante(_, _, _, c) }.

% Docenti con peso 10
#maximize { 25 : garante(_, _, Peso, _), Peso = 10 }.

% Minimizzare i docenti con peso 5
#minimize { 100, Docente : garante(Docente, _, Peso, _), Peso = 5 }.
```

Listing 13: Gestione delle priorità dei docenti.

Inoltre, abbiamo dato priorità ai docenti che insegnano corsi fondamentali per i corsi di laurea, distinguendo tra gli insegnamenti di base, identificati dal TAF A, e gli insegnamenti caratterizzanti, identificati dal TAF B.

```
% Garanti che hanno insegnamenti di base
#maximize { 25 : garante(Docente, Corso, _, _),
                 cattedra(Corso, Docente, _, TAF),
                 TAF = a}.

% Garanti che hanno insegnamenti caratterizzanti
#maximize { 18 : garante(Docente, Corso, _, _),
                 cattedra(Corso, Docente, _, TAF),
                 TAF = b}.

% Garanti che hanno insegnamenti affini
#maximize { 10 : garante(Docente, Corso, _, _),
                 cattedra(Corso, Docente, _, TAF),
                 TAF = c}.
```

Listing 14: Gestione delle priorità del TAF.

Per ottimizzare l'assegnazione dei garanti in base alla loro competenza, sono stati privilegiati i docenti il cui l'SSD corrisponde a quello caratterizzante del corso. In questo modo, ci siamo assicurati che i garanti siano adeguatamente allineati con le competenze richieste dai corsi.

```
% Ottimizzare i garanti con SSD caratterizzante
#maximize { 20 : garante(Docente, Corso, _, _),
                 docente(Docente, _, SettoreSSD, _),
                 corso(Corso, _, SettoreSSD, _) }.
```

Listing 15: Preferenza dei garanti con macrosettore coerente a quello del corso.

Per mantenere l'equilibrio tra la qualità e l'efficacia del sistema, abbiamo minimizzato il numero di garanti assegnati a ciascun corso, penalizzando le situazioni in cui il numero di garanti supera il minimo richiesto. Questo vincolo ci ha permesso di ottimizzare le risorse, evitando l'assegnazione di più docenti di quanto fosse necessario.

```
% Minimizzo il numero di garanti per ogni corso
#minimize { Penalita : garanti_per_corso(Corso, Numero),
                       Penalita = Numero * 10 }.
```

Listing 16: Minimizzazione dei garanti per corso di laurea.

Infine, abbiamo introdotto un ulteriore criterio di ottimizzazione per privilegiare i garanti che ricoprono il ruolo di presidente del loro corso di laurea.

```
% Massimizza i presidenti che sono garanti nel loro corso di laurea
#maximize { 50, Docente, Corso : garante(Docente, Corso, _, _),
                                 presidente(Docente, Corso) }.
```

Listing 17: Massimizzazione dei presidenti come garanti.

4 Validazione della soluzione proposta

Per testare l'efficacia e la correttezza del modello sviluppato, è stato creato un esempio giocattolo basato sui corsi di laurea in Informatica (LT) e Scienze Informatiche (LM). Questo dataset ridotto è stato progettato per verificare la capacità del modello ASP di generare soluzioni valide, rispettando i vincoli definiti e implementando correttamente le ottimizzazioni.

I dati di input per il modello sono stati generati attraverso un comando Python, che ha generato i file necessari contenenti i fatti relativi ai docenti, alle coperture, ai docenti a contratto e ai limiti ministeriali.

```
python3 main.py --3027 --5069
```

Una volta generati i file, l'analisi è stata eseguita utilizzando Clingo.

```
clingo -n 0 --parallel-mode 8 --time-limit=180 lp/* main.lp
```

Il modello ha prodotto una soluzione ottima in meno di un *decimo di secondo*, dimostrando tempi di computazione estremamente brevi. Per il corso di Informatica (codice: 3027) sono stati assegnati 9 garanti in conformità ai requisiti, mentre per il corso di Scienze Informatiche (codice: 5069) sono stati assegnati 6 garanti, sempre in conformità ai requisiti.

È stata inoltre verificata la coerenza tra i garanti assegnati e gli SSD caratterizzanti, con 5 docenti afferenti per Informatica e 4 per Scienze Informatiche.

4.1 Dataset ridotto: Dipartimento SMFI

Per valutare la scalabilità del modello, è stato eseguito un test utilizzando un dataset ridotto relativo al Dipartimento di Scienze Matematiche, Fisiche e Informatiche (SMFI), comprendente sei corsi di studio. Anche in questo caso, i file di input sono stati generati tramite Python.

```
python3 main.py --3027 --5069 --3026 --5036 --3030 --5037
```

L'analisi è stata eseguita con Clingo utilizzando lo stesso comando precedentemente indicato, ma con un timer impostato a un'ora. Sebbene non sia stata trovata una soluzione ottimale entro il limite di tempo, il modello ha generato 23 soluzioni in pochissimi secondi, tra cui un modello ottimale rispetto ai vincoli soddisfacibili. Questo modello ottimale, tra quelli generati, ha assegnato zero docenti a contratto come garanti su un totale di 45 garanti, dimostrando la capacità del sistema di rispettare i vincoli più stringenti.

Durante il test, è stata utilizzata la configurazione che prevede solo pesi pari a 10, al fine di ridurre la complessità computazionale. Tuttavia, il modello supporta anche l'opzione di pesi pari a 5, che, pur aumentando significativamente i tempi di computazione, offre maggiore flessibilità nella generazione delle soluzioni.

4.2 Dataset grande: tutti i dipartimenti

Come ultima valutazione del modello, è stato utilizzato un dataset comprendente tutti i corsi di studio dell'ateneo. Prima dell'elaborazione, sono stati esclusi i casi particolari relativi ai corsi inter-atenei, poiché non erano considerati rilevanti per questa analisi. Anche in quest'ultimo caso, i file di input sono stati generati tramite Python.

```
python3 main.py --all
```

Durante questa valutazione, è stato impostato un limite di tempo di un'ora. Sebbene non sia stato trovato un modello ottimale entro il limite stabilito, il modello ha generato 283 soluzioni, tra cui un modello ottimale rispetto ai vincoli soddisfacibili. Questo modello finale includeva solo 11 docenti a contratto su un totale di 869 garanti assegnati, dimostrando una notevole efficienza nell'ottimizzazione delle risorse e una forte preferenza per l'utilizzo di docenti non a contratto.

5 Conclusioni

Il lavoro presentato ha dimostrato l'efficacia di un sistema automatizzato per l'assegnazione dei garanti accademici, basato su Answer Set Programming. Attraverso l'integrazione di tecniche di preprocessing dei dati, la definizione accurata di vincoli ministeriali e la gestione delle priorità, è stato possibile sviluppare un modello flessibile e scalabile, in grado di affrontare le sfide legate alla gestione delle risorse accademiche.

I risultati ottenuti hanno evidenziato la capacità del modello di soddisfare i requisiti istituzionali, rispettando vincoli stringenti e massimizzando l'efficienza nell'uso delle risorse. L'analisi su diversi dataset, dai casi studio ridotti ai benchmark su larga scala, ha permesso di validare il sistema sia in ambienti controllati che su scenari più complessi e realistici. In particolare, i test sul dataset completo hanno evidenziato come il modello possa adattarsi a situazioni critiche, sebbene l'aggiunta di pesi flessibili abbia comportato un aumento esponenziale della complessità computazionale.

Acknowledgments

Questo progetto è stato realizzato durante il corso di Programmazione Dichiarativa (a.a. 2024–25), presso l'Università degli Studi di Parma. Tutti i benchmark sono stati effettuati su un chip Apple Silicon M3Pro avente 11 core e 18 GB RAM.

Il codice sorgente è disponibile su Github:

`https://github.com/simonecolli/ottimizzazione-garanti-accademici`.