Modello per la risoluzione di un problema di soddisfacimento di vincoli in MiniZinc

Mohamed Salah Jebali matricola: 5968114

Indice

1 Introduzione

Il programma è stato scritto in linguaggio MiniZinc e costruisce un modello per la generazione di un calendario esami che rispetti i vincoli forniti dal testo dell'esercizio: le aule devono essere abbastanza capienti da ospitare il numero di studenti iscritti per l'esame; gli esami non possono essere spezzati durante la pausa pranzo; ciascun studente può sostenere al più un esame al giorno. Il modello generato è abbastanza generico da poter permettere di risolvere una qualsiasi istanza. I test eseguiti controllano che, data una certa istanza in input, il calendario generato sia quello atteso e che, quindi, rispetti i vincoli prescritti. Inoltre, vengono confrontati i tempi di risoluzione di ciascuna istanza.

2 Descrizione del programma

In questa sezione verrà spiegato come avviare il programma e la struttura del codice. Per prima cosa occorre scaricare ed installare il compilatore MiniZinc e il corrispettivo IDE presso dal sito `https://www.minizinc.org/software.html`. Dopodiché occorre scaricare la cartella contenente il codice e i dati in input da https://github.com/salahjebali/MiniZinc-modelling-for-constraint-satisfaction-problem-.git.

2.1 Come avviare il programma

1. Aprire MiniZinc IDE e aprire il modello *esercizio_ jebali.mzn* contenuto nella cartella *intelligenza_ artificiale_ esercizio.*
2. Aprire un file *data_ *.dzn* a piacere contenuto nella cartella data.
3. Impostare come solver Gecode 6.1.1 [built-in] e premere RUN.

2.2 Breve descrizione del codice

Il codice è suddiviso in tre sezioni: la prima dedicata alla dichiarazioni dei parametri in input, la seconda per la dichiarazione delle variabili decisionali e la terza per la scrittura dei vincoli. Seguono solver e output.
Fra i parametri in input ci sono gli esami del corso, il numero di studenti, il numero di giorni e il numero delle aule. Ci sono inoltre un array che associa ad ogni esame la durata, un array che associa ad ogni aula la capienza e un array bidimensionale che associa ad ogni studente l'esame al quale è prenotato.
Le variabili decisionali sono tutti array che per ogni esame associano numero di studenti prenotati, aula, orario di inizio, orario di fine, giorno e studente che fa l'esame.
L'output fornito restituisce una tabella che ad ogni esame associa: giorno di svolgimento scelto tra la lista dei giorni forniti, aula scelta tra la lista delle aule fornite, orario di inizio e orario di fine.

3 Descrizione dei test eseguiti

Lo scopo dei test era di verificare che, data una certa istanza, con una certa caratteristica, il calendario generato in output rispettasse i vincoli prescritti. Inoltre per ogni istanza è stato riportato il tempo di risoluzione.

3.1 Specifiche d'ambiente

Le specifiche d'ambiente nel quale sono stati condotti i test sono:

- **specifiche di sistema: macchina:** PC Lenovo; **sistema operativo:** Windows 10 Home; **tipologia:** x64-based; **processore:** AMD A12-9720P RADEON R7 12 COMPUTER CORES 4C+8G 2.70 GHz; **RAM:** 11.664 MB.
- **dati utilizzati:** file "*.dzn" contenuti nella cartella "data".
- **solver utilizzato:** Gecode 6.1.1 [built-in].
- **linguaggio e IDE:** MiniZinc e MiniZinc IDE 2.4.1
- **RUN configuration:** "default behaviour" per CSP

3.2 I test eseguiti: INPUT

Di seguito è descritta la tabella dei dati in input che associa ad ogni file data_*.dzn lo schema dei dati in input, eccetto la matrice delle prenotazioni che è possibile visualizzare all'interno di ciascun file di input:

NOME FILE	# esami	# studenti	# aule	# giorni	capienza per aula	durata per esame
data_0.dzn	3	4	3	3	[12,12,12]	[4,3,1]
data_1.dzn	3	10	1	3	[13]	[4,3,4]
data_2.dzn	3	50	2	2	[54,53]	[4,3,4]
data_3.dzn	3	100	1	10	[102]	[1,3,4]
data_4.dzn	3	1000	1	10	[1000]	[4,3,1]
data_5.dzn	3	100	1	2	[102]	[1,3,4]
data_6.dzn	3	100	1	1	[102]	[1,3,4]
data_7.dzn	3	100	1	2	[99]	[1,3,4]

Alcune considerazioni sugli output attesi sono che in **data_0.mzn** c'è uno studente che è prenotato a tutti e 3 gli esami, perciò ci si aspetta che il calendario generato sia suddiviso in almeno 3 giorni per il vincolo che uno studente non può eseguire più di un esame al giorno. Le stesse considerazioni valgono per **data_1.mzn**, **data_3.mzn**, **data_4.mzn**.
Inoltre, per **data_6.mzn** ci si aspetta che sia **non soddisfacibile** perché ci sono più studenti iscritti a 2 esami, ma un solo giorno disponibile.
Infine, anche per **data_7.mzn** ci si aspetta un output **non soddisfacibile**, perché la capienza delle aule disponibile non è sufficiente ad ospitare il numero di iscritti.

3.3 I test eseguiti: OUTPUT

Di seguito sono mostrate le immagini degli output generati per ciascun input. Esse rappresentano i calendari generati e i tempi di esecuzione. Le singole immagini sono reperibili nella cartella *output*.

```
Compiling esercizio_jebali.mzn, with additional data data_0.dzn
Running esercizio_jebali.mzn
ESAME: AI | GIORNO: 3 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 12
ESAME: IC | GIORNO: 2 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 11
ESAME: CN | GIORNO: 1 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 9
----------
Finished in 802msec
```

```
Compiling esercizio_jebali.mzn, with additional data data_0.dzn
Running esercizio_jebali.mzn
ESAME: AI | GIORNO: 3 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 12
ESAME: IC | GIORNO: 2 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 11
ESAME: CN | GIORNO: 1 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 9
----------
Finished in 802msec
```

```
Compiling esercizio_jebali.mzn, with additional data data_1.dzn
Running esercizio_jebali.mzn
ESAME: AI | GIORNO: 3 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 12
ESAME: IC | GIORNO: 2 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 11
ESAME: CN | GIORNO: 1 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 12
----------
Finished in 1s 70msec
```

```
Compiling esercizio_jebali.mzn, with additional data data_2.dzn
Running esercizio_jebali.mzn
ESAME: AI | GIORNO: 2 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 12
ESAME: IC | GIORNO: 1 | AULA: 1 | ORA INIZIO: 14 | ORA FINE: 17
ESAME: CN | GIORNO: 1 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 12
----------
Finished in 990msec
```

```
Compiling esercizio_jebali.mzn, with additional data data_3.dzn
Running esercizio_jebali.mzn
ESAME: AI | GIORNO: 3 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 9
ESAME: IC | GIORNO: 2 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 11
ESAME: CN | GIORNO: 1 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 12
----------
Finished in 1s 45msec
```

```
Compiling esercizio_jebali.mzn, with additional data data_4.dzn
Running esercizio_jebali.mzn
ESAME: AI | GIORNO: 3 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 12
ESAME: IC | GIORNO: 2 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 11
ESAME: CN | GIORNO: 1 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 9
----------
Finished in 1s 142msec
```

```
Compiling esercizio_jebali.mzn, with additional data data_5.dzn
Running esercizio_jebali.mzn
ESAME: AI | GIORNO: 2 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 9
ESAME: CN | GIORNO: 1 | AULA: 1 | ORA INIZIO: 8 | ORA FINE: 12
ESAME: IC: nessun studente iscritto per questo esame
----------
Finished in 1s 51msec
```

```
Compiling esercizio_jebali.mzn, with additional data data_6.dzn
Running esercizio_jebali.mzn
=====UNSATISFIABLE=====
Finished in 1s 64msec
```

```
Compiling esercizio_jebali.mzn, with additional data data_7.dzn
Running esercizio_jebali.mzn
=====UNSATISFIABLE=====
Finished in 956msec
```

4 Conclusioni

In conclusione, gli output restituiti dal solver Gecode 6.1.1[buil-in], per il modello creato, rispecchiano le attese e rispettano i vincoli prescritti. Gli esami non vengono spezzati durante la pausa pranzo. Gli studenti possono fare al più un esame al giorno. Le aule scelte sono sufficientemente capienti per ospitare il numero di iscritti all'esame. Laddove questi vincoli non vengono rispettati, il problema risulta non soddisfacibile.
Inoltre, i tempi di esecuzione sono molto buoni ed è possibile osservare che pur passando da 10 studenti a 1000 studenti l'ordine di grandezza del tempo di risoluzione rimane intorno al secondo.