**Datanose**

Een rooster voor iedereen

Bas Chatel  
Universiteit van Amsterdam  
bchatel@student.uva.nl

Job Huisman  
10119647  
Universiteit van Amsterdam  
jhuisman@student.uva.nl

Bram Sloots  
Universiteit van Amsterdam  
bdsloots@student.uva.nl

1. **Inleiding**

*Hier schrijf je een inleiding die in elk geval kort, bondig en compleet de hele vraagstelling bevat. Ook moet er een inschatting komen van de toestandsruimtegrootte (belangrijk, daar deden we het voor), en eventuele restricties op transities in beschreven worden, maar nog niet de methodes die je gebruikt om de toestandsruimte te doorzoeken*

**-** Roosteren is lastig, veelgebruikt, veelvoorkomend probleem

- uitleg probleem, compact maar duidelijk

- stroomschema voor puntentelling(zeer inzichtelijk)

- Toestandsruimt 140! – 124! / 16!

1. **Methodes**

*In deze paragraaf beschrijf je de methodes die je hebt gebruikt, liefst in de volgorde waarin je ze hebt toegepast (in dit geval drie). Die hoeft niet per sé te kloppen met de tijdsvolgorde, echte wetenschappers proberen ook vanalles door elkaar. Wat wél moet kloppen zijn de details van de algoritmes, en de bijbehorende resultaten in de volgende paragraaf. Je moet zoveel informatie geven dat je experiment in principe herhaalbaar is, en je resultaten reproduceerbaar, ook je algoritme een stochastisch element (randomfunctie) bevat. Echte stoere mensen maken al hun resultaten, data en sourcecode ook online beschikbaar. Dat is nu nog niet overal gangbaar, maar gaat het wel worden (mijn inschatting)*

* 1. **Simulated Annealing**

*Gebruik meerdere subparagrafen als je dat nodig vindt. Bij meerdere methodes is dat vaak een goed idee.*

* 1. **Genetische algoritmes**

*In tegenstelling tot eerdergebruikte methodes…*

1. **Resultaten**

*Hier bespreek je heel droog je resultaten. Als je statistieken hebt: toevoegen. Als je vergelijkingen hebt met randomposities: toevoegen. Alles is woord en getal, alle details en het liefst ook het één en ander in grafieken, plaatjes of anderzins.*

1. **Conclusies**

*Hier schrijf je je conclusies, eventuele overdenkingen (hoe zou het nog beter kunnen, is het algoritme ook in andere gebieden toepasbaar).*

1. **Dankwoord**

*Wat ook nog kan is een dankwoord, bijvoorbeeld voor mensen die wel geholpen hebben maar geen auteur zijn, mensen die je een inzicht hebben gegeven, of administrators die je even hun supercomputer hebben laten gebruiken. Altijd naam en bedrijf noemen en zorgen dat de bedankte persoon zich er goed over voelt.*

1. **Referenties**

*Als je literatuur hebt gebruikt, hier toevoegen. Als je eraan refereert in de tekst, zet je op die plek alleen [1], zodat mensen achterin de details kunnen vinden. Als je geen literatuur gebruikt, weglaten.*

Algorithms

* Welk algoritme past er bij het geïntroduceerde probleem.

Notation

Before presenting the general model we will introduce some notation and conventions to simplify the problem statement.

* The amount of courses which need to be scheduled is denoted by *C.* C = 1,…., *C*
* The amount of lessons per course to be scheduled, is denoted by *L*. l = 1,…., *L*
* The lecture rooms are denoted by *R,* = {r1, r2, r3, r4, r5, r6, r7}
* The students are denoted by *S,* {s1, s2, s3, …., s610}
* The scheduling period will be 5 days = {d1, d2, d3, d4, d5}
* Each day has 4 timeslots (09:00 – 11:00; 11:00 – 13:00; 13:00 – 15:00; 15:00 – 17:00) denoted by T = {t1, t2, t3, t4}, optionally = {t5} (17:00 – 19:00)\*
* The schedule can be represented in multiple ways, i.e. per student, per location-slot, per course. Because of our scheduling algorithm and the scheduling priorities the schedule is made per … . Resulting in the following notation: ….. .

\*Optionally t5 can be used, but only for c 0.1110. So this will be one slot per day extra at max.

State Space

Formulation

Assumptions

Counstraints

Implementation

* Hoe gaat het in z’n werk

Analysis of sample instance

Conclusion

Literature

Toestandsruimte, zowel absoluut als ook in formule

29 vakken

610 studenten

39 hoorcolleges

7 zalen

5 \* 4 \* 7 slots in een week = 140 zaalslots

Prioriteiten:

Datastructuur

Random rooster

Testmodule

1000 punten voor een geldig rooster

Bonus: spreiding vakken over de week

Malus:

* voor ieder vak van x activiteiten 10 minpunten voor x-1, 20 voor x-2, 30 voor x-3.
* Per student teveel per ruimte 1 minpunt per conflict
* Dubbelroostering geeft 1 minpunt per conflict

\* Als suggestie wordt gegeven dat verschillende groepen van een werkcollege geen verschillende activiteiten zijn, en de weekafstand tussen een hoor- en een werkcollege is de \*kortste\* weekafstand tot een groep.

Escape:

* 17-19 is er in c0.110 een escape. Kosten 50 minpunten.

Advanced:

* Misschien is een aardige 'advanced'-opdracht om de boekingsbezetting (bezette tijdsslots tegen vrije tijdsslots) per zaal te registreren.
* Misschien is het ook aardig om de zetelbezetting (studenten tegen max. capaciteit) per zaal te registreren.