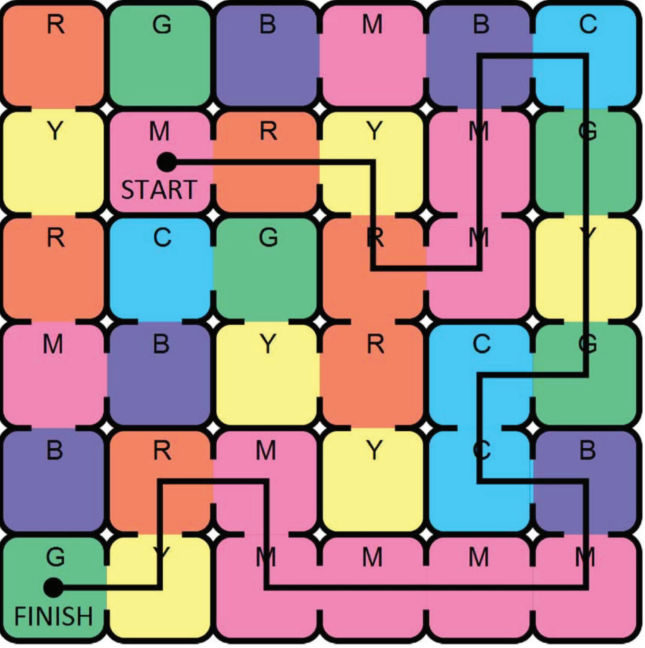


Introductie

Situering

- Generatie van spelinhoud
  - Plattegronden (kerkers, doolhoven ...)
  - Rekwisieten (diamantjes, altaars, wapens ...)



()	==	()	==	()	==	()	==	()	==	()
()		()		()		()		()		
()		()		()		()		()		
()	==	()	==	()	==	()	==	()	==	()
()	==	()		()	==	()	==	()	==	()

- ProbLog
  - Probalisitische programmeertaal
  - Uitbreiding op ProLog
  - Waarheden kansen geven
- Spelinhoud en probabilistisch programmeren
  - AnsProlog (en anderen)
  - Beperkt in probabilliteit


Hypothese


- ProbLog biedt minstens evenveel
  - Niet-uniforme verdelingen
  - Eenvoudigere representaties van bepaalde constraints

Aanpak

Syntactische omzetting

- Omzetten van bestaande code op syntactisch niveau
- Geen algoritmische wijzigingen
  - Hier en daar een optimalisatie
- Kansverdelingen en integriteitsvoorwaarden interessant om nader te bekijken

 Kansverdelingen

 Integriteitsvoorwaarden

AnsProlog: min { predicaat } max

↓

ProbLog: select\_uniform (voor min=1, max=1)

AnsProlog: :- not oplosbaar

↓

ProbLog: evidence(oplosbaar)

Voorbeeld

1 {start(X,Y):dim(X):dim(Y)} 1.  
:- not victory

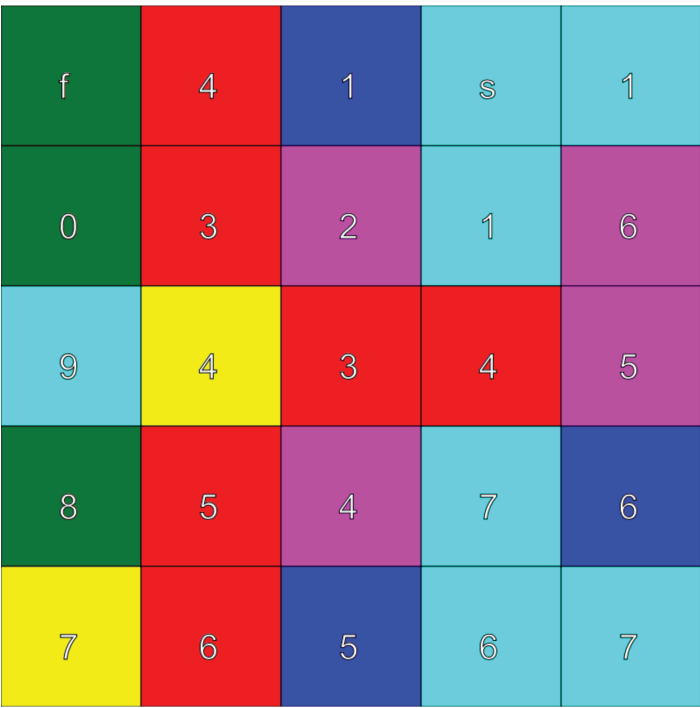
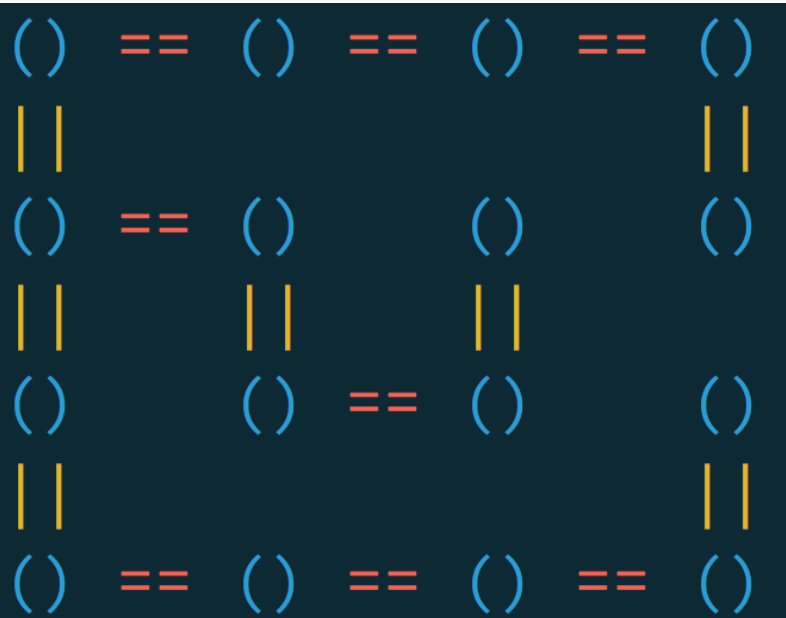
→

start\_and\_finish((A,B),(C,D)) :-  
pairs(P),  
select\_uniform(I,P,(A,B),R).  
  
victory :- victory\_at(T), time(T)

\* Hierbij is P de lijst van alle mogelijke startcoördinaten

Systeem analyseren

- Puzzels genereren (chromatic maze, perfect maze)
- Uitvoeringstijd analyseren
  - Verschillende dimensies
- Grafische weergave om correctheid te controleren



Probleem & Motivatie

Probleem

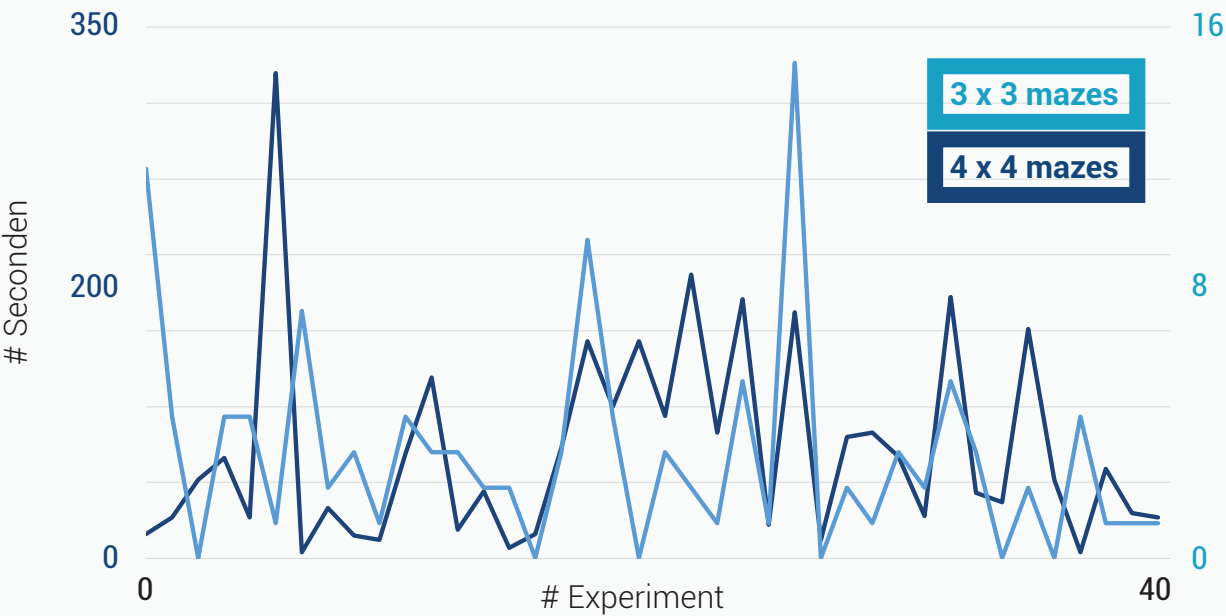
- System
  - Is ProbLog inzetbaar voor genereren van spelinhoud?
  - Zijn de uitvoeringstijden voorspelbaar en beperkt?
- Syntax
  - Predicaten eenvoudig ?

Motivatie

- Onderzoek naar efficiëntie van ProbLog systeem.
- Invloed van andere kansverdeling op uitkomst

Conclusies

- Sampling methode is onvoorspelbaar en traag.
- Gevolg: genereren van puzzels vaak langzaam.
  - Voor grotere dimensies stijgt de tijd nagenoeg exponentieel
- Syntatisch: predicaten eenvoudig voorstellen is geen probleem.
- Grafiek hier naast is voor perfect mazes
  - Dimensie 5: per puzzel min. 400 seconden
- Bij chromatic mazes: lange uitvoeringstijd bij
  - Hoge dimensies
  - Voorwaarden op minimale oploslengte



Sample Methode bij "Perfect Mazes"

Toekomst & Verder onderzoek

Implementatie

- Puzzels nog meer benchmarken
  - Wat met nog hogere dimensies?
  - Code optimaliseren?
  - Zoeken waar het 'mis' gaat



Meer voorbeelden

- Omzetting van nog meer puzzels
  - Levels met nog meer constraints
  - Complexere spelen
  - Procedurale generatie (mogelijk?)



Referenties

M.J. Nelson, A. Smith. ASP with applications to maze and levels. In N. Shaker, J. Togelius en M. J. Nelson, eds. Procedural Content Generation in Games: A Textbook and an Overview of Current Research. Springer. 2015

A.M. Smith. Answer Set Programming for Procedural Content Generation: A Design Space Approach. IEEE Transactions On Computations Intelligence and AI in Games, vol. 3, no. 3, september 2011