Binoku Solver
alias de HUMO “waar of niet waar” puzzel

Proefondervindelijke Opbouw van Theorema’s

De puzzelregels (HUMO4173)

1. Vul het raster zo in dat elke rij en kolom gevuld is met zes nullen en zes eentjes.
2. Er mogen niet meer dan twee nullen of twee eentjes naast of onder elkaar staan.
3. Identieke rijen en kolommen zijn niet toegestaan.
4. De stelling (quizvraag) is waar of niet waar wanneer in het rode vakje een 1 of een 0 staat.

			0								
						0				1	1
0										1	
							0	0			0
				1							
		0				0			0	0	
				0							1
		1		0	0					0	
					0	0		0			
	1			0			1		1		
		0		0	0					1	0

Quizvraag: In een triljard zitten eenentwintig nullen

De voor de hand liggende theorema's

1. Tussen 2 gelijke waarden zit het complement (want anders 3 gelijke waarden op rij).
2. Aan beide zijden van 2 gelijke waarden zit het complement (want anders 3 gelijke waarden op rij).
3. Van zodra 6 vakjes een gelijke waarde bevatten, bevatten de overige spaties het complement.

De voorbeeldpuzzel (HUMO4173)

- Q: In een triljard zitten eenentwintig nullen

			0								
						0				1	1
0										1	
							0	0			0
				1							
		0				0			0	0	
				0							1
		1		0	0					0	
					0	0		0			
	1			0			1		1		
		0		0	0					1	0

			0			1			1	0	
						1			1	0	0
						0			0	1	1
0						0		1	0	1	1
				0		1	0	0	1	0	0
				1	0	1		0	1	1	0
		0		1		0		1	0	0	1
	1			0	1	0	1	0	0	1	1
1	0	1	1	0	0	1	0	1	1	0	0
	0			1	0	0	1	0	0	1	
	1			0	1	0	1	0	1	0	
1	1	0	1	0	0	1	0	1	0	1	0

- 32 ingevulde vakjes
- met de 3 stellingen kom je soms al een heel eind:
  88 vakjes ingevuld, 2 rijen & 3 kolommen volledig

De voorbeeldpuzzel (HUMO4173)

- Q: In een triljard zitten eenentwintig nullen

			0								
						0				1	1
0										1	
							0	0			0
				1							
		0				0			0	0	
				0							1
		1		0	0					0	
		1			0	0		0			
	1			0			1		1		
		0		0	0					1	0

			0			1			1	0	0
						1			1	0	0
						0			0	1	1
0						0		1	0	1	1
			1	0		1	0	0	1	0	0
	0	1	0	1	0	1		0	1	1	0
	1	0	0	1		0		1	0	0	1
0	1	0	1	0	1	0	1	0	0	1	1
1	0	1	1	0	0	1	0	1	1	0	0
1	0	1	0	1	0	0	1	0	0	1	1
0	1	0	1	0	1	0	1	0	1	0	1
1	1	0	1	0	0	1	0	1	0	1	0

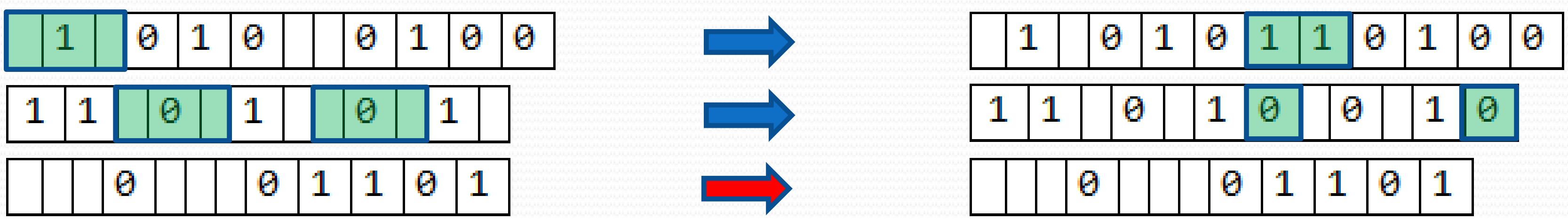
- 32 ingevulde vakjes
- met het antwoord op de quizvraag erbij raak je soms wat verder: 106 vakjes ingevuld, 5 rijen & 4 kolommen volledig

De meer geavanceerde theorema's

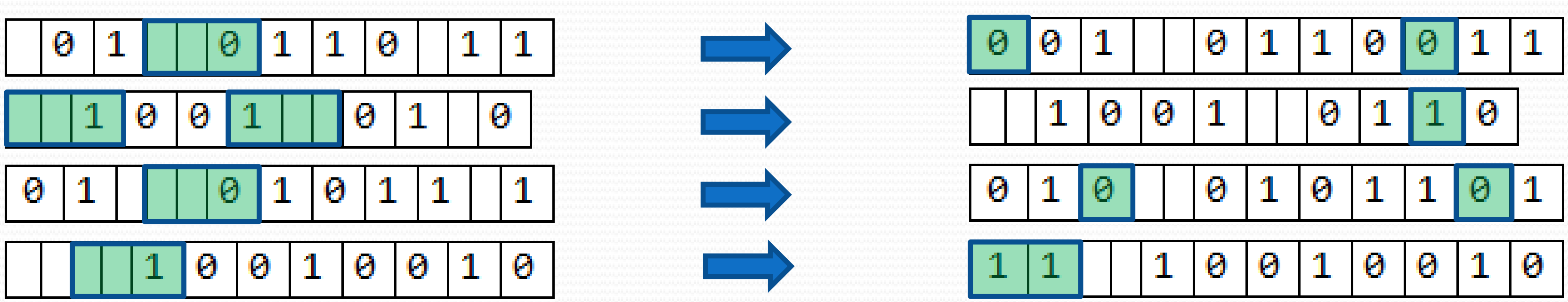
1. (2×0 & 5×1 of 5×0 & 2×1) & 5 spaties naast elkaar:
   ⇒ vul in met 00100 resp. 11011
2. (3×0 & 5×1 of 5×0 & 3×1) & 3 spaties naast elkaar:
   ⇒ resterende spatie bevat minderheidswaarde (0 resp. 1)
3. (4×0 & 5×1 of 5×0 & 4×1) & 3 spaties naast elkaar tweezijdig geflankeerd door minderheidswaarde (0 resp. 1):
   ⇒ vul in met 010 resp. 101
4. (4×0 & 5×1 of 5×0 & 4×1) & 3 spaties naast elkaar links geflankeerd door minderheidswaarde (0 resp. 1):
   ⇒ rechtse van die spaties bevat ook die minderheidswaarde
5. (4×0 & 5×1 of 5×0 & 4×1) & 3 spaties naast elkaar rechts geflankeerd door minderheidswaarde (0 resp. 1):
   ⇒ linkse van die spaties bevat ook die minderheidswaarde

De meer geavanceerde theorema's

6. Spatie aan weerszijden van een minderheidswaarde & aantal dergelijke sets + reeds aanwezige meerderheidswaarden=6:
   $\Rightarrow$ in een van beide flankerende spaties zit telkens 1 meerderheidswaarde, vul de overige spaties met een minderheidswaarde.



7. Dubbele spaties geflankeerd door minstens een minderheidswaarde & aantal dergelijke sets + reeds aanwezige meerderheidswaarden=6:
   $\Rightarrow$ in een van beide spaties zit telkens 1 meerderheidswaarde, vul de overige spaties met een minderheidswaarde.



De meer geavanceerde theorema's

8. (5×1 of 5×0) & 3 spaties naast elkaar:
   overige niet aansluitende spaties bevatten een minderheidswaarde.

1	0	0	1	0	0	1		0			

→

1	0	0	1	0	0	1	1	0			

- Informatie over positie spaties en onderlinge afstand:

```
In[24]:= SeqPos = Flatten[Position[TstRow1, "", {1}, Heads → False]]

Out[24]= {8, 10, 11, 12}

In[25]:= SeqPosPos =
          Block[{n = 12}, MapThread[{#1, #2} &,
            {SeqPos, Partition[Differences[Join[{-1}, SeqPos, {n + 2}]], 2, 1]}]]

Out[25]= {{8, {9, 2}}, {10, {2, 1}}, {11, {1, 1}}, {12, {1, 2}}}

In[26]:= Spc3Pos = Position[Partition[Last /@ SeqPosPos, 3, 1],
           {{l_ /; IntegerQ[l] && l > 1, 1}, {1, 1}, {1, r_ /; IntegerQ[r] && r > 1}}]

Out[26]= {{2}}

In[27]:= SpcPos =
          First /@ Complement[SeqPosPos, SeqPosPos[[Spc3Pos[[1, 1]] ;; Spc3Pos[[1, 1]] + 2]]]

Out[27]= {8}
```

De meer geavanceerde theorema's

- Pas ontdekt als noodzakelijk voor het oplossen van puzzel uit HUMO4194 …

									0		
							0		1	1	
						0		0	0		0
	0		0		0					0	
						1				0	
		0									
		0				0		0			0
0					0		1				
	0	0							1	1	
	0				0	0		0		1	1

						0	1	1	0	0	1
						1	0	0	1	1	0
						1	0	1	1	0	1
						0	1	0	0	1	0
		0				1	0	1	0	1	0
1	0	1	0	1	0	0	1	0	1	0	1
		1				1	0	1	1	0	
		0				1	0	1	0	1	
		0				0	1	0	1	0	0
0	1	1	0	1	0	0	1	1	0	0	1
1	0	0	1	0	1	1	0	0	1	1	0
1	0			1	0	0	1	0	0	1	1

- Overige opvulregels lopen al snel vast bij deze puzzel …

		0				0	1	0	1	0	0

1	1	0				0	1	0	1	0	0

Het uitsluitingsprincipe

9. Vergelijk rijen (resp. kolommen) die 5*0 & 5*1 bevatten met reeds volledige rijen (resp. kolommen) die 6*0 & 6*1 bevatten.

Als de reeds ingevulde vakjes overeenstemmen, vul de resterende 2 spaties in met de ontbrekende 1 & 0 teneinde identieke rijen of kolommen te vermijden (basisregel 3).

0	0	1	1	0			1	0	1	1	0
0	0	1	1	0	0	1	1	0	1	1	0

0	0	1	1	0	1	0	1	0	1	1	0
0	0	1	1	0	0	1	1	0	1	1	0

De voorbeeldpuzzel (HUMO4173)

- Q: In een triljard zitten eenentwintig nullen

			0								
						0				1	1
0										1	
							0	0			0
				1							
		0				0			0	0	
				0							1
		1		0	0					0	
					0	0		0			
	1			0			1		1		
		0		0	0					1	0

			0	1	0	1	0	1	1	0	0
						1			1	0	0
						0			0	1	1
0				1		0		1	0	1	1
				0		1	0	0	1	0	0
				1	0	1		0	1	1	0
		0		1		0		1	0	0	1
0	1			0	1	0	1	0	0	1	1
1	0	1	1	0	0	1	0	1	1	0	0
1	0			1	0	0	1	0	0	1	1
0	1			0	1	0	1	0	1	0	1
1	1	0	1	0	0	1	0	1	0	1	0

- Met de set van opgestelde regels komt je al een heel eind
- Volstaat niet voor oplossing: modulerecursie nodig (zie verder)

De voorbeeldpuzzel (HUMO4173)

- Q: In een triljard zitten eenentwintig nullen

			0								
						0				1	1
0										1	
							0	0			0
				1							
		0				0			0	0	
				0							1
		1		0	0					0	
		1			0	0		0			
	1			0			1		1		
		0		0	0					1	0

0	1	1	0	1	0	1	0	1	1	0	0
1	0	0	1	0	1	1	0	1	1	0	0
0	0	1	0	1	1	0	1	0	0	1	1
0	1	0	0	1	0	0	1	1	0	1	1
1	0	1	1	0	1	1	0	0	1	0	0
0	0	1	0	1	0	1	1	0	1	1	0
1	1	0	0	1	1	0	0	1	0	0	1
0	1	0	1	0	1	0	1	0	0	1	1
1	0	1	1	0	0	1	0	1	1	0	0
1	0	1	0	1	0	0	1	0	0	1	1
0	1	0	1	0	1	0	1	0	1	0	1
1	1	0	1	0	0	1	0	1	0	1	0

- Invullen van het antwoord op de quizvraag leidt tot oplossing!

De ultieme, maar tijdrovende uitweg: recursie op alternatieven (1 & 2)

- Zoek 1 geïsoleerde spatie op een kruising van een rij en een kolom
- Voer vervolgens 2 afzonderlijke iteraties van de invulstellingen uit:
  1. Eerst met een 0
  2. Alternatief met een 1
- Doe dit recursief voor alle resterende dergelijke geïsoleerde spaties.

- Indien die strategie niet tot 1 of meer oplossingen leidt:
- Zoek een rij of kolom met slechts 1 enkele geïsoleerde spatie
- Voer vervolgens 2 afzonderlijke iteraties van de invulstellingen uit:
  1. Eerst met een 0
  2. Alternatief met een 1
- Doe dit recursief voor alle resterende enkele geïsoleerde spaties.

De ultieme, maar tijdrovende uitweg: recursie op alternatieven (3 & 4)

- Indien die strategie nog niet tot 1 of meer oplossingen leidt:
- Zoek een rij of kolom met 5×0 & 5×1.
- Voer vervolgens 2 afzonderlijke iteraties van de invulstellingen uit;
  1. Vul de ene spatie eerst in met 0, de andere met een 1
  2. Alternatief in omgekeerde volgorde
- Doe dit recursief voor alle resterende groepen van 2 spaties.

- Indien die strategie nog steeds niet tot 1 of meer oplossingen leidt:
- Zoek een rij of kolom met 4×0 & 5×1 of 5×0 & 4×1
- Voer vervolgens 3 afzonderlijke iteraties van de invulstellingen uit voor elk van resterende combinaties 001, 010 & 100 resp. 011, 101, 110.
- Doe dit recursief voor alle resterende dergelijke rijen en kolommen

Verificatie en Eliminatie van conflicten

- Controleer elk van de uit de modulerecursie resulterende tabellen op conflicten met de basisregels en verwijder deze die er niet aan voldoen.
- Slechts 1 combinatie van alle genomen opties zou conflictvrij tot de unieke oplossing moeten leiden

- Verificatie op eventuele conflicten na elke toepassing van een van de invulregels leidt tot een substantiële reductie van de uitvoeringstijd door overbodige invuloperaties te vermijden.

De voorbeeldpuzzel (HUMO4173)

- Q: In een triljard zitten eenentwintig nullen

			0								
						0				1	1
0										1	
							0	0			0
				1							
		0				0			0	0	
				0							1
		1		0	0					0	
					0	0		0			
	1			0			1		1		
		0		0	0					1	0

0	1	1	0	1	0	1	0	1	1	0	0
1	0	0	1	0	1	1	0	1	1	0	0
0	0	1	0	1	1	0	1	0	0	1	1
0	1	0	0	1	0	0	1	1	0	1	1
1	0	1	1	0	1	1	0	0	1	0	0
0	0	1	0	1	0	1	1	0	1	1	0
1	1	0	0	1	1	0	0	1	0	0	1
0	1	0	1	0	1	0	1	0	0	1	1
1	0	1	1	0	0	1	0	1	1	0	0
1	0	1	0	1	0	0	1	0	0	1	1
0	1	0	1	0	1	0	1	0	1	0	1
1	1	0	1	0	0	1	0	1	0	1	0

- Set van opgestelde regels in combinatie met modulerecursie vindt unieke oplossing zelfs zonder het antwoord op de vraag in te vullen.

De gemakkelijke opgave (HUMO4162)

- Q: Karel de Grote was de zoon van Pepijn de Korte
- Ook deze puzzel raakt reeds opgelost zonder module recursie en zonder het antwoord op de vraag!

											1
				1							0
		0				1		0		1	
0											1
		1					0	0			
										0	0
		0		1	1			0			0
	0		0	0					0		
						1					
		1		1		1		0		1	1
		0		1	1				1		1

0	0	1	1	0	1	0	0	1	0	1	1
1	0	1	0	1	1	0	0	1	1	0	0
1	1	0	1	0	0	1	1	0	0	1	0
0	1	0	0	1	0	0	1	1	0	1	1
0	0	1	1	0	1	1	0	0	1	0	1
1	0	1	1	0	0	1	0	1	1	0	0
1	1	0	0	1	1	0	1	0	0	1	0
0	1	0	1	1	0	1	0	0	1	0	1
1	0	1	0	0	1	0	1	1	0	1	0
1	1	0	1	0	0	1	0	1	1	0	0
0	0	1	0	1	0	1	1	0	0	1	1
0	1	0	0	1	1	0	1	0	1	0	1

Een moeilijke opgave (HUMO4152)

- Q: de Titanic werd gebouwd in New York
- Zonder modulerecursie raak je met deze puzzel niet ver:

0		1	1						1		
		0									1
0		0	0		0		1				
			0								
				0	0			1			
1							1			0	
				0			1		1		
			1								
1	1				0			0	0		
			1	1						1	1

0	0	1	1	0					1		
		0									1
	0	1	1	0							
0	1	0	0	1	0		1				
			0		1			0			
			1	0	0	1	0	1			
1				1		0	1	1	0	0	1
				0			1	0	1	1	0
	0	1	1	0	1		0	1	1	0	
1	1	0	0	1	0		1	0	0	1	
		0	1	1	0				0	1	1

- Slechts 68 van de 144 vakjes ingevuld, geen rij of kolom volledig!

Een moeilijke opgave (HUMO4152)

- Q: de Titanic werd gebouwd in New York

➢ Recursieve toepassing van module vindt unieke oplossing zelfs zonder het antwoord op de vraag in te vullen (na 1.2 s).

0		1	1						1		
		0									1
0		0	0		0		1				
			0								
				0	0			1			
1							1			0	
				0			1		1		
			1								
1	1				0			0	0		
			1	1						1	1

➡

0	0	1	1	0	1	1	0	0	1	0	1
0	0	1	0	1	1	0	1	1	0	1	0
1	1	0	0	1	0	1	0	1	0	0	1
1	0	1	1	0	1	1	0	0	1	0	0
0	1	0	0	1	0	0	1	1	0	1	1
0	1	1	0	0	1	1	0	0	1	1	0
1	0	1	1	0	0	1	0	1	1	0	0
1	1	0	0	1	0	0	1	1	0	0	1
0	1	0	1	0	1	0	1	0	1	1	0
0	0	1	1	0	1	1	0	1	1	0	0
1	1	0	0	1	0	0	1	0	0	1	1
1	0	0	1	1	0	0	1	0	0	1	1

De uitdagende opgave uit HUMO 4192

- Q: De oorspronkelijke kleur van Coco-Cola was groen
- Zonder modulerecursie raak je met deze puzzel niet ver:

			0			1					
											1
0				0	0						
						1					
	0				0			0	0		
	0		0								1
1		1								1	
											0
									1	1	
	1				1						
			0	0				1	1		1

			0			1			0	1	0
1				1		0		1	0	0	1
0			1	0	0	1	1	0	1	1	0
0	1			0	1	1	0	1	1	0	0
1	0			1	0	0	1	0	0	1	1
	0	1	0								1
	1	0									0
1	0	1							0	1	1
									1	0	0
					0			0	1	1	0
	1				1				0		1
		1	0	0	1		0	1	1	0	1

- ➢ Slechts 76 van de 144 vakjes ingevuld, slechts 1 kolom volledig!

De uitdagende opgave uit HUMO 4192

- Q: De oorspronkelijke kleur van Coco-Cola was groen
- ➢ Recursieve toepassing van module vindt unieke oplossing zelfs zonder het antwoord op de vraag in te vullen (na 2.1 s).

			0			1					
											1
0				0	0						
						1					
	0				0			0	0		
	0		0								1
1		1								1	
											0
									1	1	
	1				1						
			0	0				1	1		1

→

0	1	1	0	0	1	1	0	1	0	1	0
1	0	0	1	1	0	0	1	1	0	0	1
0	1	0	1	0	0	1	1	0	1	1	0
0	1	1	0	0	1	1	0	1	1	0	0
1	0	0	1	1	0	0	1	0	0	1	1
1	0	1	0	0	1	1	0	1	0	0	1
0	1	0	1	1	0	1	1	0	1	0	0
1	0	1	0	1	0	0	1	0	0	1	1
1	0	1	1	0	1	0	0	1	1	0	0
0	1	0	1	1	0	1	0	0	1	1	0
0	1	0	0	1	1	0	1	0	0	1	1
1	0	1	0	0	1	0	0	1	1	0	1