**Probleembeschrijving**:

De opdrachtgever houdt van het maken van sudoku’s die dagelijks in de krant staan. Het probleem is dat de antwoorden voor van die sudoku’s vaak pas de volgende dag in de krant komen. Mijn opdrachtgever heeft niet het geduld om op deze antwoorden te wachten omdat ze wil weten of ze de sudoku goed heeft gemaakt, of ze heeft hulp nodig om de sudoku op te lossen dus wilt even kijken bij de antwoorden voor hulp.

**Algoritme**:

Ik gebruik maken van een backtracking algoritme.

Ik heb al meerdere papers gelezen die een oplossing geven voor een sudoku doormiddel van backtracking en/of recursieve backtracking. Een paper die voor mijn gevoel een goede uitleg geeft is deze: <https://www.ijmter.com/papers/volume-4/issue-12/solving-sudoku-using-backtracking-algorithm.pdf> ik denk ook dat ik deze grotendeels ga gebruiken maar ben nog steeds aan het verder zoeken of ik een betere kan vinden.

**Beschrijving backtracking algoritme.**

Backtracking is een depth-first search in plaats van een breadth-first search. Dit komt omdat bij backtracking het algoritme eerst een tak helemaal afzoekt tot hij op een mogelijk antwoord komt voordat het algoritme door gaat naar een andere tak.

**Minimum** :

Het Algoritme werkend krijgen voor een 9 x 9 sudoku. De gebruiker moet de sudoku invullen in de console, het programma zal hierna de oplossing berekenen hierna kan de gebruiker kiezen of hij alle antwoorden wilt hebben of een paar tips. Als de gebruiker tips wilt moet hij invullen van welke grid hij het getal wilt weten.

**Gemiddeld**:

Alles hierboven plus UI doormiddel van pygame, tkinter of mathplot waar de gebruik zijn sudoku makkelijk kan invullen en makkelijk kan kiezen welke getallen hij wilt zien van de sudoku.

**Optimaal**:

Als ik teveel tijd over heb nog een ander algoritme opzoeken en die ook uitwerken en dan testen welke sneller werkt. Ook zou ik nog een algoritme kunnen opzoeken die sudoku’s kan generen, hierdoor kan mijn opdrachtgever nog meer sudoku’s spelen zonder te wachten op de volgend krant.

**Planning**

**Week 1:**

In week 1 ga ik me compleet focussen op het maken en werkend krijgen van het algoritme hoe sneller ik dat af weet te krijgen hoe makkelijker de rest van de taken kunnen worden. Ik verwacht 4/5 dagen bezig te zijn met het maken/testen van het algoritme.

**Week 2:**

Als ik het algoritme werkend heb gekregen ga ik werken aan de UI zo niet zorg ik dat ik sowieso voor het eind van de week het algoritme compleet werkend heb gekregen. (op het moment dat het algoritme aan het einde van week 2 nog steeds niet werkt ga ik contact zoeken met de docent voor hulp.) Ik heb nog niet veel gedaan met het maken en ontwerpen van UI ’s dus hier verwacht ik wel een tijdje mee bezig te zijn 2/3 dagen. Ik hoop dat ik de UI in 2 tot 3 dagen af kan krijgen zodat ik nog kan proberen een sudoku generator te maken.

**Week 3:**

Ik verwacht dat ik het eerste deel van deze week nog verder zou moeten werken aan de generator. Als er verder geen enorme problemen zijn ontstaan verwacht ik dat ik vanaf woensdag de puntjes op i zou moeten kunnen zetten en alles moeten kunnen afronden. Dus poster maken presentatie/rapport maken zo nodig.

**Risico’s:**

Het risico dat ik zie is dat ik waarschijnlijk weinig motivatie heb om te beginnen aan het project en hierdoor veelte laat ga beginnen aan het project hierdoor kan ik in tijd nood komen. Hierom ga ik naar de HU bibliotheek zodat ik mezelf ga forceren om eraan te beginnen. (dit heeft vaker geholpen)

Een ander risico dat ik voorzie is dat als ik de UI wil maken dat ik mijn code voor het algoritme helemaal overhoop moet gaan gooien om het werkend te krijgen in samenwerking met de UI. Hierom ga ik eerst heel goed kijken wat voor soort programma ik wil gaan gebruiken voor een UI, en om tijdens het schrijven van mijn algoritme hier al rekening mee te houden.