Presentatie Probleemstelling

**Wat is het probleem?**

Ons doel bestaat erin een didactisch programma te maken dat op een visuele manier combinatorische problemen kan oplossen. Combinatoriek omvat de tak van de wiskunde die de mogelijke verdelingen telt van een groep objecten. Bijvoorbeeld: op hoeveel manieren kan een pak speelkaarten geschut worden? Waarop het antwoord 52! Is. Mensen kunnen dergelijke problemen gemakkelijk oplossen aangezien ze de achterliggende structuren herkennen, waarvoor eenduidige formules bestaan. Voor artificiële intelligentie is dit echter niet vanzelfsprekend. Slechts recent is een techniek binnen de AI, genaamd “Lifted Reasoning” toegepast op de combinatoriek, met een positief resultaat. De zogenaamde CoSo (Combinatorics Solver) is in staat om sneller telproblemen op te lossen dan ooit tevoren. Hij breekt de problemen namelijk op in eenvoudigere subproblemen, vergelijkbaar met hoe mensen redeneren. En juist hierom is het een logische aanvulling om voor de solver een visualisatiesoftware te schrijven die mensen in staat stelt om op een intuïtieve manier telproblemen te begrijpen.

**Waarom is het belangrijk?**

Zoals eerder aangegeven, gaat het over een didactisch programma. Het zal studenten helpen om combinatieproblemen beter begrijpen. Niet alleen kan het oplossingen voor deze problemen berekenen, het geeft ook de strategie weer die het gebruikt heeft om hiertoe te komen. Een student kan zo stap voor stap leren hoe hij het probleem zelf oplost.

(In feite bestaat er al een zeer gelijkaardig programma aan hetgeen dat wij willen bereiken, genaamd *SymboLab*. Dit is een veelgebruikt programma dat voor verschillende wiskunde problemen, uitgebreide stap-voor-stap oplossingen biedt. *SymboLab* is echter veel minder krachtig op het domein van combinatoriek dan wat wij ontwikkelen en kan daarom niet gezien worden als een vervanging.

**Hoe aanpakken?**

Voor de implementatie van de visualisatiesoftware hebben wij gekozen voor de Game Engine *Godot*. *Godot* is een programma waar wij beide al jaren ervaring mee hebben, dus waar we gemakkelijk mee aan de slag kunnen. Het gebruikt een eigen programmeertaal genaamd *GDScript*, die verwant is aan Python, de taal waarin de *CoSo solver* geschreven is. Bovendien kunnen we met *Godot* gemakkelijk externe bestanden gebruiken zoals de *CoSo solver* en andere *Python libraries*.

Eerst zijn we begonnen aan de voorstelling van de domeinverzamelingen. Wij hebben ervoor gezorgd dat we de input van de *solver* kunnen doorgeven aan onze implementatie. Hierdoor kunnen de meegegeven domeinen voorgesteld worden. Voor de voorstelling hebben we gebruikt gemaakt van *Area-proportional Venn Diagrams*. Zoals de naam aangeeft, zijn dit Venndiagrammen waarbij de verzamelingen en de doorsneden in verhouding staan met de domeingroottes. De wiskunde achter dit probleem behoorde echter niet tot ons hoofddoel en hebben we dan ook overgelaten aan een externe *Python library*. Onze volgende stap bestaat uit de voorstelling van simpele telproblemen en daarna kunnen overgaan naar telproblemen met *constraints*. Deze voorstellingen zullen animaties bevatten met bijvoorbeeld pijltjes om de transities tussen de veschillende stappen weer te geven/te verduidelijken.