Assessing Flood Risk in Functional Zones

A Knowledge Technology System Utilizing Forward Chaining

Jacob Kenning (s4251067), Viktor Milenov (s4310861), Mohamed Hassan (s4327276)
Faculty of Science and Engineering
University of Groningen
The Netherlands
December 2022

Contents

1 Introduction

Knowledge technology or knowledge-based systems are programs designed to mimic the problem-solving capabilities of a human expert in a specific domain, by using a knowledge base of facts and rules and a set of reasoning or inference mechanisms. The problem we chose to tackle by utilizing a knowledge-based expert system was that of assessing the flood risk of a given area. The system consists of 3 paradigms – domain, problem-solving, and rule models – that work in tandem to calculate the flood risk assessment score, to be explained in further detail in Section 2. The inner workings of the whole system and any other granular details will be fully explained throughout this report in their respective sections.

2 Problem Definition

Our system is designed to produce a flood risk score for a given plot of land based on numerous, interwoven factors. A key piece of information in a flood risk assessment is the current vs planned function of the land, also known as the (current/planned) functional type of the land. These types are things such as residential, industrial, recreational, etc., and oftentimes the given plot of land is being repurposed, and a change of functional type can cause an increase in flood risk. Defining a plot of land's functional type is itself based on numerous factors; including details such as building density, minimum building height, minimum green area required on the land, presence of critical infrastructure, etc. The full list of factors can be found in Appendix A, which illustrates all the necessary variables we consider in the knowledge system.

After the lands' current and planned functional types are determined, we account for the exposure the land has to flooding, which is influenced by factors such as how much rainfall the general area receives, and we also consider how vegetated the land is; if the land is non-vegetated (instead covered by building or artificially sealed surfaces), vegetated (woody areas/cultivated land for agricultural purposes etc) or covered in water (permanent, such as a lake/quarry, or temporary such as a fountain/pond). After compiling all of these pieces of information and more, we produce a final vulnerability score that represents the risk of building on this land, which is a symbol on a 5-point scale: VL, L, M, H, VH. The rating of 'VL' is the most ideal; standing for 'very low', it indicates that one can safely build on the piece of land given. 'L' is low, 'M' stands for medium, 'H' for high, and 'VH' for very high. This scale is one used by human experts globally when assessing the flood risk of a potential project and was based on official architectural documents provided to us by our experts.

Ideally, one also must consider factors such as soil and topological details of the land, however including geological factors in our assessment would have significantly increased the overall complexity of the system, and our experts cautioned us against that course of action.

3 Expert

We were able to have two experts aid us in the design of this system, Pavel and Krystian Milenov, s4310861's father and grandfather. Pavel is a civil engineer by trade and has done a lot of work regarding assessing agricultural plots of land, however, his expertise spans the scope of the project. Krystian was an architect, and so combining both of their domains of knowledge we were able to produce a comprehensive overview of how to assess the flood (and other) risks of a given plot of land. The first couple of sessions were knowledge elicitation, where we created a set of flowcharts to help us visualize the garnered knowledge and how it all pieced together (these can be seen in Section 5.1).

4 Role of knowledge technology

The system uses forward chaining to reach its final conclusion by following the results obtained by looking at the individual pieces of information given in order to go through the intertwined web of rules that the rule model of the system is comprised of.

In a system such as the one detailed in this paper, knowledge technology plays a crucial role in several ways. Firstly, it allows us to represent the myriad of variables that go into making an expert decision in the form of ontologies or rule sets and represents the relationships between the variables clearly as well. In our system, we utilize a knowledge base (KB) that contains multiple rule-sets dictating how certain variables interact with one another. Furthermore, using a Graphical User Interface (GUI) to interface with the user allows one to acquire the necessary data in a way that is also convenient for the user, such that the system can make the necessary inferences and come to decisions based on the inputs and KB through the means of forward chaining to come to a conclusion about the flood risk.

5 The knowledge models

In the following sections we will showcase the three models that we have, namely, the problem-solving model, the domain model and the rule model respectively and comment on their functionality and explain the specifics of each model in depth.

5.1 Problem-solving model

The problem-solving model describes the full layout of the expert system showing the individual modules and how they are interconnected with each other. Below you can find the complete problem-solving model depicted in a diagram.

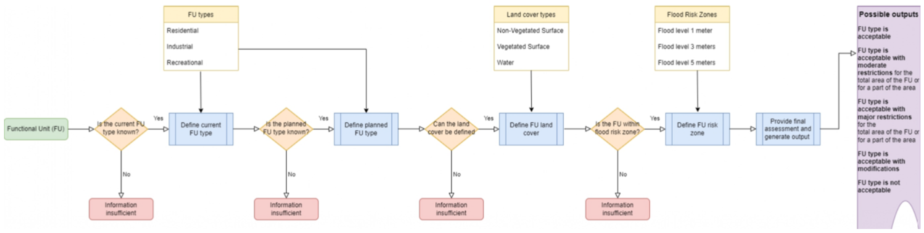

Figure 1: Problem-solving model.

From Fig.1 we can see that the full expert system consists of four individual modules (blue boxes in diagram) that work in tandem in order to produce the final output. They are each comprised of different components and input variables which are used to calculate the different assets throughout the system until we reach a final risk evaluation.

Firstly, we have the modules that are responsible for storing the data concerning the current type of the functional zone as well as the future type that is planned to be at that location. Furthermore, these modules gathers additional information about the functional zones such as the population density, the density of buildings in the area as well as other crucial information about the zone that is necessary in order to assess the final risk. In Fig.2 you can see the first module in more detail. We implemented our problem-solving model in Python, as we did the domain and rule models too, and we used the PySimpleGUI package to present the user with the interface to provide the system with inputs.

The second module in the system is concerned with the planned function of the land. For instance, the current function of a plot of land could be residential, however, that does not necessarily mean that the future function has to be the same, it can be industrial, recreational or even mixed. Therefore, the input for this module is identical to the input for the first module where we define what the current function of the land is.

The third module in the system is concerned with the land cover of the functional zone. In this part of the system, information regarding what is actually already built on the zone is taken as input and it is stored for later use. The land cover type of the functional can be residential (more buildings that allow people to live there), industrial (with factories and/or power plants), parks (with vegetation such as trees, fields, and lakes). Moreover, this module contains the percentage of the land that is covered by a specific type of construction.

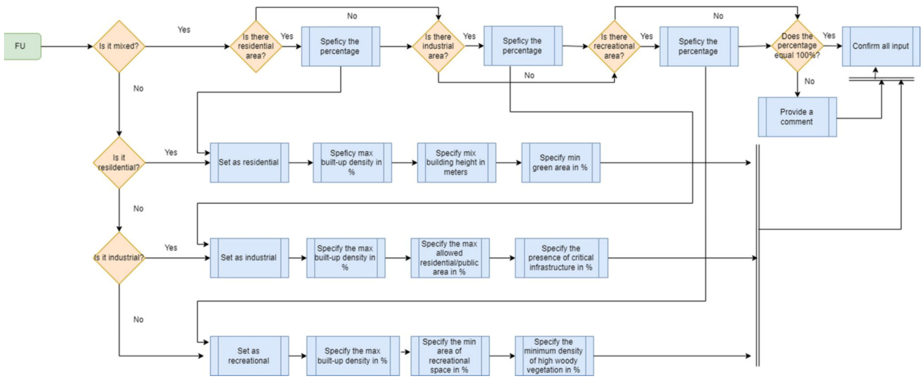

Figure 2: Diagram showing the internal flow of the first and second modules in the system. Note we did not include the 'mixed' FU type (to be discussed in Section 8.2) although it is an FU type considered by human experts in the field.

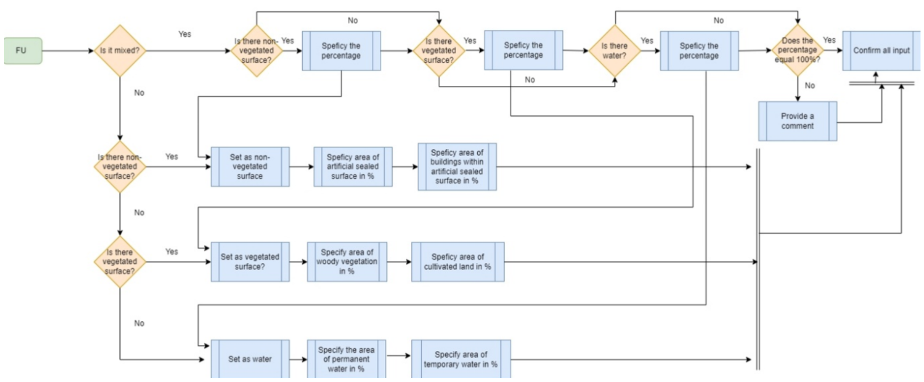

Figure 3: Diagram showing the internal flow of the third module. Note we did not include the 'mixed' FU type (to be discussed in Section 8.2) although it is an FU type considered by human experts in the field.

The final module in the system takes into account information that depicts the expected flooding risk of that specific functional zone. The flooding risk can be either 1 meter (light), 3 meters (moderate), 5 meters (heavy). This unit in the system is responsible for providing an estimate of how likely the zone is susceptible to a flood by using rules that have been predetermined in the Master Plan of the city - which is the complete overview of how a cities transport, economy, housing, public buildings etc should be implemented - and doesn't serve as a concrete estimate of probability of whether the land will flood or not.

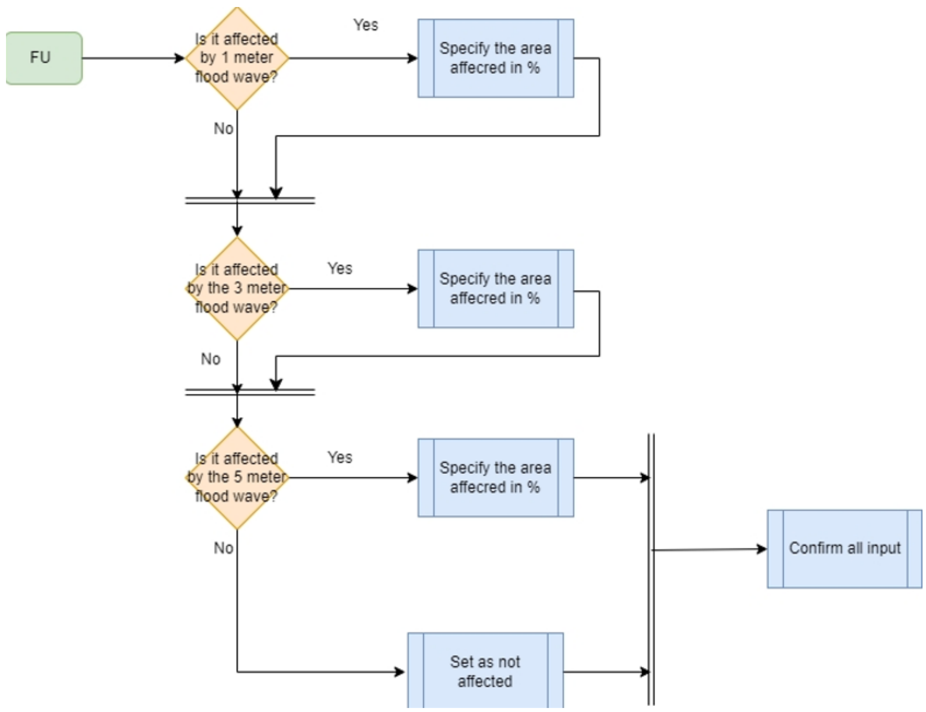

Figure 4: Diagram showing the internal flow of the fourth module. Note we did not include the 'mixed' FU type (to be discussed in Section 8.2) although it is an FU type considered by human experts in the field.

All of the data stored in these individual modules will be used in order to assess the overall flooding risk of the functional zone as well as provide to which degree the planned type and its specifications are acceptable, and whether it requires modifications. Furthermore, a general diagram showing where all the assets are being calculated can be seen below for the purpose of better understanding the flow of the system.

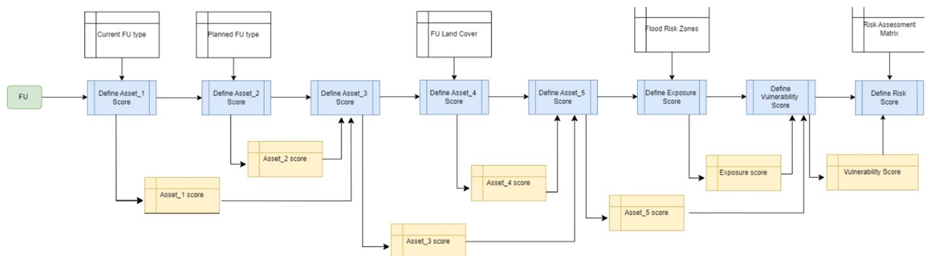

Figure 5: General diagram showing where all the assets are calculated and how that leads to the final risk score. Note we did not include the 'mixed' FU type (to be discussed in Section 8.2) although it is an FU type considered by human experts in the field.

5.2 Domain model

Our domain model consists of a few classes/knowledge units: functional units (FU) & its' properties, the land cover of the FU, the exposure risk of the land, and then combining these classes to calculate other scores called 'assets'. Assets are intermediate variables and are combinations of the component variables of the given FU. As the data collection goes on, we use previous asset scores in the calculations for new ones. This follows our expert knowledge(see appendix A). As a result, the final asset score (asset 5) is a culmination of all the information about the land type, and this value is then used to determine the final risk assessment for the plot of land. The data contained within the functional unit depends on what purpose the land holds, and all variations are displayed below. The FU class can be used for both the current and expected functional type of the land.

Class: Functional Unit (FU) If the functional type = 'Residential', we require the user to input the values of 3 variables: built-up density, which is how covered by buildings the land is (given as a percentage). Next, we need the average building height, which is merely indicated - as per industry standards - as either <15m or >15m. We then ask for the minimum green area required on the land, which is also given as a percentage.

If the functional type = 'Industrial', we first require the built-up density (as before). We then ask for the maximum allowed residential area, meaning the maximum allowed space for things like apartments/other communal buildings in the industrial zone on the land. Then we ask for the measure of presence of critical infrastructure: things such as highways, connecting bridges and tunnels, railways, utilities and buildings. These variables are all given as percentages.

If the functional type = 'Recreational', we again require the maximum built-up density. Next is the minimum desired area of recreational space, followed by the minimum density of high woody vegetation, which is the minimum density needed of trees and forested areas on the land. Again, all of these data fields are given as percentages.

Class: FU Land Cover If the user indicates the land is a 'non-vegetated surface', we require a couple of pieces of information: how much of the land is an artificially sealed surface. In other words, how much of the total area of land covered by artificial constructions (buildings, roads, car-parks etc), and then we need the percentage of that area that is purely covered by buildings. Both, as usual, are given by percentages.

If instead the user says that the land is a 'vegetated surface', we first ask the area of woody vegetation present on the land plot, and then we ask how much of the land is cultivated for agricultural purposes.

Lastly, the user could indicate that the land contains some areas of water on it, such as lakes, pools, reservoirs etc. If this is the case and the user chooses 'water', we only require them to tell us how much of the land is taken up by permanent water features such as a lake or a quarry, and then how much of the land is covered by temporary water installments such as public fountains or artificial hot springs etc.

Finally, We have some last pieces of information that we gather, such as the exposure risk, estimated flood risk, and intermediate calculations using all of these variables to generate new asset scores that can then be used further down the inference line in order to produce the final flood risk assessment score.

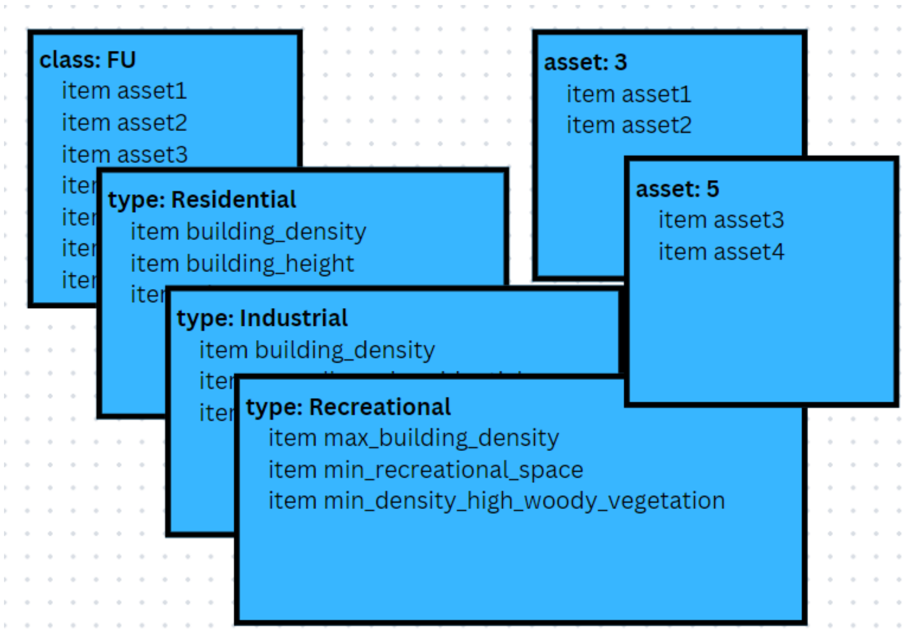

Figure 6: A representation of how different types, assets and classes are defined in our domain model.

5.3 Rule model

Calculating Asset1 and Asset2 is done with pre-defined functions given to us by our experts that can be found in the `functions_trial.py` file in the code. Then these values are used to calculate Asset3 by using the rules in the full rule model given in Appendix C by the knowledge base. Then Asset4 is calculated using similarly defined functions, afterwards, the values for Asset3 and Asset4 are used to calculate Asset5. Following this, is the calculation of the vulnerability score for the different levels of flooding (1 meter, 3 meters, or 5 meters). In order to calculate this value, Asset5 and the value of the exposure score are needed. Finally, we calculate the overall risk level of the land by using the vulnerability score and flooding levels.

5.4 User interface, functionality

We used the Python package PYSimpleGUI to build the user interface and posit all the necessary questions to the user. The back-end parts of the system (the KB, inference engine, and the functions) were also coded in Python, and in order to run the program we included all of these files into a main file and run it from there. The user interface consists of separate windows for each stage of the questioning, and only relevant windows will pop up that are based on the previously inputted values. This follows our expert knowledge. The type of inference used in our expert system is forward chaining, as all of the data that is gathered from the user, is used in order to evaluate the final risk of the functional zone, which means that from the data available, we reach our conclusions and not the other way around.

6 Walk through of a Session

6.1 External

Upon running the code, the user is presented with the first question which asks what the current functional type of the land in question is. Depending on their answer, the window then changes to ask them for the respective 3 variables that provide more insight into what is currently situated on the land (as explained in Section 5.2 and Appendix A). After this, questions regarding the planned functional type are asked, and again depending on what type the user inputs a corresponding set of questions pops up, asking the user the relevant questions about the planned functional type.

Next, the user inputs the land cover information, describing whether the land is vegetated, non-vegetated, or submerged in water. Each option has a unique set of follow-up questions, and after indicating which of the 3 possible land covers describes the land they are asked the respective follow-ups in a new window. After this, they are asked to input the exposure to flood risk, where the options are 1 meter, 3 meters, or 5 meters of flooding. After this, the system outputs its final response, telling the user whether it is acceptably safe to build on the land and if any modifications are needed to increase the safety, or it outputs that the flood risk is too high with accompanying information about how to reduce this risk.

6.2 Internal

The system alternates between merely taking in inputs and calculating asset scores along the way. After reading the input for the current and planned functional types it calculates asset 3 according to the table visible in the spreadsheet (Appendix A), which is a culmination of the asset scores for the current and planned FUs respectively. Next, it takes in the land cover information and uses this in conjunction with the Asset 3 score to calculate Asset 4, which in turn is used with the following exposure information to calculate Asset 5 (the naming of these variables is arbitrary, however, we followed the conventions as given to us by our experts). It then defines the final vulnerability score of the plot of land, and depending on the flood risk and vulnerability the system outputs the appropriate label for how large the flood risk is, along with any suggestions about modifications to make the land safer to build on.

The system is coded entirely in Python, and is a modular system; we have separate files for the GUI, the functions to calculate asset scores etc., the knowledge base, and the main driver file. The GUI was created using the Python package PYSimpleGUI, which includes clickable buttons and customizable color settings to make the user experience more friendly and accessible to all age groups within our target demographic. This modular structure of the code allowed for easier debugging and making the task more granular inherently made it easier to code, as we could focus all of our attention on one section of the task at a time.

The inference engine has facts added to it as the inputs go on, and the system infers the values of the related assets by reading the facts currently known to it in the array and performing the corresponding calculations. Any new variables it calculates then also get added to the facts array, and in this way, the KB evolves over time and gathers new information to inform future decisions.

7 Validation

We organized a meeting with our expert to validate our system after we had a working system, and we went through many permutations of variable inputs and tested many cases on the system, all of which held up to their scrutiny. The way we did it was by creating a hypothetical situation with certain values for the required data and presenting them to Pavel. After consulting the data, he would give a rating of VL-VH for the scenario, after which we ran the same scenario through the system and compared its output to Pavels'. As mentioned, we had a 100% correlation with our expert's recommendations, confirming the system made the correct informed decisions for any case. Pavel did point out in early implementations that our output messages were quite primitive, in the sense that they offered no explanation as to what the 'modifications' to a planned build should be in the case the system deemed the risk 'M' or above, We took this into consideration and implemented proper feedback in the final version.

However, overall he was impressed with the systems' efficacy and said that such a system has a lot of potential to be adopted into workplaces similar to his (with some further development of course), as he said it "vastly reduces the time we have to spend deciding on whether we can move forward with a project - being able to ask this system for risk assessments and focus our attention on the project itself would boost productivity and improve construction time-scales.".

8 Reflection

8.1 Evaluation of used techniques

Overall we are quite pleased with the methods we chose to use, as although not familiar to us we managed to maneuver the intricate Python documentation and the complex relationships between all the variables in the task. Choosing to use forward chaining worked well for us because of its flexibility; as successive questions were asked the system was able to generate a more solid risk assessment and gather the relevant information when it needed it. One way to illustrate how we made the right choice of inference method is to think about what would happen in a backward chaining system; it would be significantly harder to discern the starting variables (for example, the built-up density of the land or how much critical infrastructure is present on the land) from just the final risk assessment.

Regarding knowledge elicitation, we feel like the techniques we used worked well and we were able to acquire all the necessary information from our experts without major hurdles. Initially, we let Pavel go where his mind wandered; he explained the ins and outs of how these decisions are made in the real world, which aspects of assessing flood risk are more important than others, how city planners designate functional zones to certain areas of the city, etc. We were initially quite confused simply due to bombardment of information we received as we weren't sure which bits were fully relevant to creating this expert system and which bits were background knowledge/knowledge to help us understand the nature of the task but not how to complete the task itself. Later on, we narrowed our scope, asking him direct questions about what considerations he would make if he was to tackle a task like this himself. Through, this line of questioning we were able to create the spreadsheet that became the basis for our entire system (Appendix A). Utilizing the spreadsheet, along with the flowcharts that illustrated how sections of the code should run, proved very helpful. Proving their efficacy, these visualization techniques would be useful in future projects.

8.2 Problems

The first problem we encountered was finding an expert with enough time and expendable resources to aid us on this project. We had no trouble coming up with ideas for the system - we had 5 ideas with solid foundations in our first meeting - however after contacting the experts, only one ended up being doable. Furthermore, as Pavel worked full-time and was stationed in Bulgaria, there were issues in determining meeting times and how to organize everything online. As well as an issue with translating the material given to us by Pavel (and later Krystian), as the city plans highlighting how functional zones were demarcated by the Sofia municipality and their associated flood risk assessment documents they used in their jobs were all in Bulgarian, so Viktor had the arduous task of translating many pages of intense jargon into English that Jacob and Mohamed could understand and get a grasp of the steps necessary to assess flood risk.

The main problem we faced then was actually coding the system; frankly, none of us are exceptionally proficient at coding, and it was a struggle to get over the magnitude of the task. We spent a fair amount of time breaking it down into more digestible chunks and figuring out the best language and IDE to code it in. Jacob had decent experience in Java and OOP techniques, however, Viktor and Mohamed didn't, and so we had to settle for Python which was the fairest middle ground we could find that suited our needs, and then we trawled through the PySimpleGUI documentation (among other GUI Python packages) that were possibilities for implementing the graphical interface of the system.

Furthermore, we spent a while trying to figure out how to properly represent the knowledge in our knowledge base. The first proper implementation that we decided to use consisted of individual rules compiled into an array. However, some of the rules were written in such a way that caused problems when it came to the brittleness of the entire system. For instance, some of the rules contained a range (i.e. the value of the specific variable could be between two discrete values, but the variable itself could be continuous). This created a problem during the fact matching process since we could not determine in between which two values the variable is without the use of string matching, which meant that in order to add new rules to the knowledge base we also needed to adjust the string matching procedure accordingly. To avoid this, we decided to use placeholder characters that represented different ranges. For instance, as a trivial example, if the variable that we have stored in the facts array is between the values 1 and 3 then the placeholder value would be the character "b", otherwise if the variable is below the value of 1 then the placeholder character is "a" and so on. This allowed us to be able to write specific rules for adding new facts to the facts array after a rule has been found that matches in the knowledge base. These "mappings" as we call them are found in the knowledge base itself, and so if a new rule is added then we only certain parts in the knowledge base have to be adjusted while all the other files remain untouched.

Lastly, as can be seen in Fig.2 the system that our expert had planned contained a mixed FU type which consisted of co-dominant land cover types. For instance, a certain plot of land might not be solely residential, but might also consist a percentage of industrial land or recreational. We attempted to implement this functionality in our model, however, we quickly realised that this was going to be quite challenging since the mixed rules required continuous values and as we mentioned previously the solution to this was not trivial, therefore, we decided to exclude this from our model but still mention it since in real world scenarios it is more likely that a plot of land has different functionalities.

8.3 Lessons and learning curve

The biggest issue that we had to overcome (which had a considerably steep learning curve) was that our initial design of the system was not using a proper inference engine with forward chaining; it was merely an if-else system that had hard-coded rule-sets and the system wasn't really inferring new facts as it went. This required a massive revamp of the system, and through a lot of trial and error and a meeting with the teacher, we realized what we had to fix. Aside from this, we had issues creating the GUI as it is not something any of us had done before this task, however, figuring out how to create the windows buttons and store the inputs into their corresponding variables are things that have a lot of practical use elsewhere and we were glad to have learned these skills. We ultimately also gained a more in-depth understanding of the inner working of knowledge systems, their possible applications, their pitfalls, and how they can be implemented.

8.4 Improvements

Although fully functional and efficient, there are some improvements we thought of that would benefit the overall experience of using this system. For instance, in the case of a high-risk assessment, we inform the user of some basic modifications they could make to their plan of what they want to build on the land. These could be more fleshed out, however. We could for example include specific variables that the user could consider optimizing or even provide some plan they could follow in order to safely build on the land. Furthermore, we could attach a confidence rating of how serious the given assessment is; for example, if the system decides that there is a high risk of building on the land, we could output 'H, x%' instead of simply 'H'. This would require a lot more research into current evaluation practices though, as this is something that our experts did not mention as a possibility. Only a flood risk assessment score does not seem descriptive enough for a flood risk assessment - some measure of how 'H' or 'L' the decision is could be a massive addition to the system and allow the users to make a more informed decision about whether to go through with the plan or not.

Moreover, including the mixed FU type functionality in the model would greatly improve the applicability of our system and relate it more accurately to the situations encountered in the real world, allowing for more specific inquiries about a certain plot of land.

In terms of aesthetic improvements, despite the user interface being simple to use and relatively self-explanatory, the definitions of some terms used in the questions could be hidden behind a [?] symbol that you hover your cursor over to view, as for experienced users using this application they will not need to see the definitions alongside the terms, and in this case, the definitions' clutter the text window. We could also have standardized the size of each window to make the system look neater, but this isn't a massive concern and is subjective to each user.

9 Task Division

Each of the group members had a 'specialty' or one task they focused on more than the others; Viktor played the largest role in knowledge elicitation, as outside of the meetings we collectively had with our experts he had a couple of private ones in which the language used was Bulgarian, as it was sometimes easier for our experts to go into the required detail in their native tongue, which Viktor then translated and relayed back to Mohamed and Jacob. He also worked closely with Mohamed to ensure that our system hit all the necessary requirements, and they worked on the code. Jacob wrote the report, and worked closely with Viktor and Mohamed for Section 6 ensuring that we could convey the technicalities involved in the system code, as well as ensuring the vast amount of information that went into this assignment was explained clearly in each other section.

10 Appendix

10.1 Appendix A - Spreadsheet

Define FU Asset_1 score from data on the "current FU type"

Residential (Rs)							
Built-up density (Bd)			Building height (Bh)		Min Green Area (Mg)		
<20%	40-60%	>60%	<15m	>15m	<20%	40-60%	>60%
1	2	3	1	3	3	2	1

Calculation of asset_1 score for Residential = (1.2*Bd+1.4*Bh+1.1*Mg)/3

Industrial (In)								
Built-up density (Bd)			Max Allowed residental/public area (Ra)			Presence of critical infrastructure (Ci)		
<20%	40-60%	>60%	<20%	40-60%	>60%	<20%	40-60%	>60%
1	2	3	1	2	3	1	2	3

Calculation of asset_1 score for Industrial = (1.2*Bd+1.4*Ra+2*Ci)/3

Recreational (Rc)								
Max Built-up density (Bd)			Min area of recreational space (Mr)			Minimum Density of high woody Vegetation (Wv)		
<20%	40-60%	>60%	<20%	40-60%	>60%	<20%	40-60%	>60%
1	2	3	3	2	1	3	2	1

Calculation of asset_1 score for Recreational=(1.2*Bd+Mr+0.5*Wv)/3

Calculation of asset_1 score for functional units of mixed type - Asset_1=PercRs*Rs+PercIn*In+PercRc*Rc

Define FU Asset_2 score from data on the "planned FU type"

Residential (Rs)							
Built-up density (Bd)			Building height (Bh)		Min Green Area (Mg)		
<20%	40-60%	>60%	<15m	>15m	<20%	40-60%	>60%
1	2	3	1	3	3	2	1

Calculation of asset_2 score for Residential = (1.2*Bd+1.4*Bh+1.1*Mg)/3

Industrial (In)								
Built-up density (Bd)			Max Allowed residental/public area (Ra)			Presence of critical infrastructure (Ci)		
<20%	40-60%	>60%	<20%	40-60%	>60%	<20%	40-60%	>60%
1	2	3	1	2	3	1	2	3

Calculation of asset_2 score for Industrial = (1.2*Bd+1.4*Ra+2*Ci)/3

Recreational (Rc)								
Max Built-up density (Bd)			Min area of recreational space (Mr)			Minimum Density of high woody Vegetation (Wv)		
<20%	40-60%	>60%	<20%	40-60%	>60%	<20%	40-60%	>60%
1	2	3	3	2	1	3	2	1

Calculation of asset_2 score for Recreational=(1.2*Bd+Mr+0.5*Wv)/3

Calculation of asset_2 score for functional units of mixed type - Asset_1=PercRs*Rs+PercIn*In+PercRc*Rc

Define the combined FU Asset_3 score from the combination of the results for Asset_1 and Asset_2

		Asset_1 (Current FU)		
		>=3	(3-1)	*<=1*
Asset_2 (Planned FU)	*>3*	3	3	2
	[3-1]	3	3	2
	<1	2	2	1

Define the Asset_4 score from the data on "FU land cover"

Non-Vegetated Surface (Ns)					
Area of artificially sealed surface (As)			Area of buildings (Ab)		
<20%	40-60%	>60%	<20%	40-60%	>60%
1	2	3	1	2	3

Non-Vegetated Surface = (1.2*Ass+1.4*Ab)/3

Vegetated Surface (Vs)					
Area of Woody Vegetation (Wv)			Area of cultivated land (Cl)		
<20%	40-60%	>60%	<20%	40-60%	>60%
3	2	1	1	2	3

Vegetated Surface = (0.2*Wv+1.1*Cl)/3

Water (Wt)					
Area of permanenet water (Pw)			Area of temporary water (Tw)		
<20%	40-60%	>60%	<20%	40-60%	>60%
1	2	3	3	2	1

Water=(1.5*Pw+0.9*Tw)/3

Asset=PercNs*Ns+PercVs*Vs+PercWt*Wt

Define the combined Asset_5 score from the combination of the results for Asset_3 and Asset_4

		Asset 3		
		>=3	(3-1)	<=1
Asset 4 (Land Cover)	>3	3	3	2
	[3-1]	2	2	1
	<1	2	1	1

Define the Exposure score from the data on each of the "Flood Risk Zones"

Flooding levels								
1m (once in 20 years)			3m (once in 100 years)			5m (once in 1000 years)		
<20%	40-60%	>60%	<20%	40-60%	>60%	<20%	40-60%	>60%
1	2	3	1	2	3	1	2	3

Define the Vulnerability score from the combination of the results for Asset 5 and Exposure scores, on each of the "Flood Risk Zones"

Vulnerability for 1 meter flood

		Asset 5		
		>=3	(3-1)	<=1
Exposure for 1 m	>3	5	4	4
	[3-1]	4	3	3
	<1	2	2	1

Vulnerability for 3 meter flood

		Asset 5		
		>=3	(3-1)	<=1
Exposure for 3 m	>3	5	4	4
	[3-1]	5	4	4
	<1	3	3	2

Vulnerability for 5 meter flood

		Asset 5		
		>=3	(3-1)	<=1
Exposure for 5 m	>3	5	5	4
	[3-1]	5	4	4
	<1	4	3	3

Define the Risk score from the Vulnerability score for each of the flood levels

		Vulnerability				
		5	4	3	2	1 (Negligible)
Probability	1m - High	VH	VH	VH	H	M
	3 m -	VH	H	M	L	L
	5 m - Low	H	M	L	VL	VL

VL	VERY LOW RISK
L	LOW RISK
M	MEDIUM
H	HIGH RISK
VH	VERY HIGH RISK

Define the Risk output for the assessend functional unit and for the PLANNED FU type

IF RISK score < L, for ANY flooding level, THEN	**FU type is acceptable**
IF RISK score < M AND RISK score = L, for ANY flooding level, THEN	**FU type is acceptable with moderate restrictions** for the total area of the FU or for a part of the area
IF RISK score < H AND RISK score = M, for ANY flooding level, THEN	**FU type is acceptable with major restrictions** for the total area of the FU or for a part of the area
IF RISK score < VH AND RISK score = H, for ANY flooding level, THEN	**FU type is acceptable with modifications**
IF RISK score = VH, for ANY flooding level, THEN	**FU type is not acceptable**

10.2 Appendix B - Score evaluation

1. The flood risk is low. The planned type of the functional unit/zone (FU) is acceptable. For the administrator/end-user this means that there is no need of additional indicators of risk impact monitoring, nor additional measures for strengthening resilience (administrative, technical, financial, infrastructure, etc.);

2. The flood risk is low to medium. The planned type of the functional unit/zone (FU) is acceptable with additional soft measures as moderate restrictions and additional risk assessment parameters for the total area of the FU or for a part of the area. These moderate restrictions refer to – a) regular monitoring of territorial management of the FU territory with respect to the applied restrictions; b) application of certain limitation with respect to the build-up density, the type of constructions applied and c) application of potential risk warning system .

3. The flood risk is medium to high. The planned type of the functional unit/zone is acceptable for clearly defined part of the FU with low or medium risk level with additional soft measures as defined under risk level 2. For the other part of the FU area major restrictions must be applied; in cases of uncertainty, such major restrictions must be applied for the whole FU area. These moderate restrictions refer to – a) certain limitation in territorial planning/management parameters, with respect to the build-up density, the type of constructions applied, and the establishment of buffer zones where construction will be limited; b) regular monitoring of changes; c)application of potential risk warning system; d) construction of elements protective infrastructure on vulnerable areas; e) adoption of a real-time emergency response and prevention plan.

4. The flood risk is high to very high. The planned type of the functional unit/zone is acceptable with strong modifications. In addition to the presented above measures, this implies modification of the parameters of the built-up density and building height, establishment of buffer zones, restriction rules and construction of the protection infrastructure (ditched, walls/dykes, drainage channels, prevention - network of sensors and additional emergency infrastructure).

5. The flood risk is high to very high. The planned type of the functional zone is not acceptable. Other functional type having lower risk score should be proposed or the area may be used as a emergency sector, bearing the pressure of the flood. Any significant elements of the urban infrastructure should be not allowed.

10.3 Appendix C - Knowledge Base

```
    ## rules for appending asset3 facts
## the two numbers represent the lower and upper boundaries of asset3
## the last number represents a placeholder value used for appending the facts
mappingAsset3 = {
    (0, 0): "b",
    (0, 1): "b",
    (1, 1): "b",
    (1, 2): "c",
    (1, 3): "c",
    (2, 2): "c",
    (2, 3): "c",
    (3, 3): 3,
    (3, 4): 3,
    }

## rules for appending asset5 facts
## the two numbers represent the lower and upper boundaries of asset5
## the last number represents a placeholder value used for appending the facts
mappingAsset5 = {
    (0, 0): "b",
    (0, 1): "b",
    (1, 1): "b",
    (1, 2): "c",
    (1, 3): "c",
    (2, 2): "c",
    (2, 3): "c",
    (3, 3): 3,
    (3, 4): 3,
    }
```

```
## rules for appending vulnerability facts
## the two numbers represent the lower and upper boundaries of the vulnerability score
## the last number represents a placeholder value used for appending the facts
mappingVulnerability = {
    (0, 1): "b",
    (1, 2): "c",
    (0, 0): "b",
    (2, 2): "c",
    (1, 1): "b",
    (3, 3): 3,
    (1, 3): "c",
    (2, 3): "c",
    (3, 4): 3,
    }

from functions_trial import (
    calculateAsset1Residential,
    calculateAsset1Industrial,
    calculateAsset1Recreational,
    calculateAsset2Residential,
    calculateAsset2Industrial,
    calculateAsset2Recreational,
    calculateNonVegetatedSurface,
    calculateVegetatedSurface,
    calculateWaterSurface,
)

#fill in with [premise, conclusion] for all cases
KB = [

    ## rules for asset1 residential
    ## index 0 = 1 represents that it is residential (asset1) and the following 3 indices re
    [1, 1, 1, 1, ["asset1", calculateAsset1Residential(1,1,1)]],
    [1, 1, 1, 2, ["asset1", calculateAsset1Residential(1,1,2)]],
    [1, 1, 1, 3, ["asset1", calculateAsset1Residential(1,1,3)]],
    [1, 1, 2, 1, ["asset1", calculateAsset1Residential(1,2,1)]],
    [1, 1, 2, 2, ["asset1", calculateAsset1Residential(1,2,2)]],
    [1, 1, 2, 3, ["asset1", calculateAsset1Residential(1,2,3)]],
    [1, 1, 3, 1, ["asset1", calculateAsset1Residential(1,3,1)]],
    [1, 1, 3, 2, ["asset1", calculateAsset1Residential(1,3,2)]],
    [1, 1, 3, 3, ["asset1", calculateAsset1Residential(1,3,3)]],
    [1, 2, 1, 1, ["asset1", calculateAsset1Residential(2,1,1)]],
    [1, 2, 1, 2, ["asset1", calculateAsset1Residential(2,1,2)]],
    [1, 2, 1, 3, ["asset1", calculateAsset1Residential(2,1,3)]],
    [1, 2, 2, 1, ["asset1", calculateAsset1Residential(2,2,1)]],
    [1, 2, 2, 2, ["asset1", calculateAsset1Residential(2,2,2)]],
```

```
[1, 2, 2, 3, ["asset1", calculateAsset1Residential(2,2,3)]],
[1, 2, 3, 1, ["asset1", calculateAsset1Residential(2,3,1)]],
[1, 2, 3, 2, ["asset1", calculateAsset1Residential(2,3,2)]],
[1, 2, 3, 3, ["asset1", calculateAsset1Residential(2,3,3)]],
[1, 3, 1, 1, ["asset1", calculateAsset1Residential(3,1,1)]],
[1, 3, 1, 2, ["asset1", calculateAsset1Residential(3,1,2)]],
[1, 3, 1, 3, ["asset1", calculateAsset1Residential(3,1,3)]],
[1, 3, 2, 1, ["asset1", calculateAsset1Residential(3,2,1)]],
[1, 3, 2, 2, ["asset1", calculateAsset1Residential(3,2,2)]],
[1, 3, 2, 3, ["asset1", calculateAsset1Residential(3,2,3)]],
[1, 3, 3, 1, ["asset1", calculateAsset1Residential(3,3,1)]],
[1, 3, 3, 2, ["asset1", calculateAsset1Residential(3,3,2)]],
[1, 3, 3, 3, ["asset1", calculateAsset1Residential(3,3,3)]],

## rules for asset1 industrial
## index 0 = 2 represents that it is industrial (asset1) and the following 3 indices rep
[2, 1, 1, 1, ["asset1", calculateAsset1Industrial(1,1,1)]],
[2, 1, 1, 2, ["asset1", calculateAsset1Industrial(1,1,2)]],
[2, 1, 1, 3, ["asset1", calculateAsset1Industrial(1,1,3)]],
[2, 1, 2, 1, ["asset1", calculateAsset1Industrial(1,2,1)]],
[2, 1, 2, 2, ["asset1", calculateAsset1Industrial(1,2,2)]],
[2, 1, 2, 3, ["asset1", calculateAsset1Industrial(1,2,3)]],
[2, 1, 3, 1, ["asset1", calculateAsset1Industrial(1,3,1)]],
[2, 1, 3, 2, ["asset1", calculateAsset1Industrial(1,3,2)]],
[2, 1, 3, 3, ["asset1", calculateAsset1Industrial(1,3,3)]],
[2, 2, 1, 1, ["asset1", calculateAsset1Industrial(2,1,1)]],
[2, 2, 1, 2, ["asset1", calculateAsset1Industrial(2,1,2)]],
[2, 2, 1, 3, ["asset1", calculateAsset1Industrial(2,1,3)]],
[2, 2, 2, 1, ["asset1", calculateAsset1Industrial(2,2,1)]],
[2, 2, 2, 2, ["asset1", calculateAsset1Industrial(2,2,2)]],
[2, 2, 2, 3, ["asset1", calculateAsset1Industrial(2,2,3)]],
[2, 2, 3, 1, ["asset1", calculateAsset1Industrial(2,3,1)]],
[2, 2, 3, 2, ["asset1", calculateAsset1Industrial(2,3,2)]],
[2, 2, 3, 3, ["asset1", calculateAsset1Industrial(2,3,3)]],
[2, 3, 1, 1, ["asset1", calculateAsset1Industrial(3,1,1)]],
[2, 3, 1, 2, ["asset1", calculateAsset1Industrial(3,1,2)]],
[2, 3, 1, 3, ["asset1", calculateAsset1Industrial(3,1,3)]],
[2, 3, 2, 1, ["asset1", calculateAsset1Industrial(3,2,1)]],
[2, 3, 2, 2, ["asset1", calculateAsset1Industrial(3,2,2)]],
[2, 3, 2, 3, ["asset1", calculateAsset1Industrial(3,2,3)]],
[2, 3, 3, 1, ["asset1", calculateAsset1Industrial(3,3,1)]],
[2, 3, 3, 2, ["asset1", calculateAsset1Industrial(3,3,2)]],
[2, 3, 3, 3, ["asset1", calculateAsset1Industrial(3,3,3)]],

## rules for asset1 recreational
## index 0 = 3 represents that it is recreational (asset1) and the following 3 indices r
```

```
[3, 1, 1, 1, ["asset1", calculateAsset1Recreational(1,1,1)]],
[3, 1, 1, 2, ["asset1", calculateAsset1Recreational(1,1,2)]],
[3, 1, 1, 3, ["asset1", calculateAsset1Recreational(1,1,3)]],
[3, 1, 2, 1, ["asset1", calculateAsset1Recreational(1,2,1)]],
[3, 1, 2, 2, ["asset1", calculateAsset1Recreational(1,2,2)]],
[3, 1, 2, 3, ["asset1", calculateAsset1Recreational(1,2,3)]],
[3, 1, 3, 1, ["asset1", calculateAsset1Recreational(1,3,1)]],
[3, 1, 3, 2, ["asset1", calculateAsset1Recreational(1,3,2)]],
[3, 1, 3, 3, ["asset1", calculateAsset1Recreational(1,3,3)]],
[3, 2, 1, 1, ["asset1", calculateAsset1Recreational(2,1,1)]],
[3, 2, 1, 2, ["asset1", calculateAsset1Recreational(2,1,2)]],
[3, 2, 1, 3, ["asset1", calculateAsset1Recreational(2,1,3)]],
[3, 2, 2, 1, ["asset1", calculateAsset1Recreational(2,2,1)]],
[3, 2, 2, 2, ["asset1", calculateAsset1Recreational(2,2,2)]],
[3, 2, 2, 3, ["asset1", calculateAsset1Recreational(2,2,3)]],
[3, 2, 3, 1, ["asset1", calculateAsset1Recreational(2,3,1)]],
[3, 2, 3, 2, ["asset1", calculateAsset1Recreational(2,3,2)]],
[3, 2, 3, 3, ["asset1", calculateAsset1Recreational(2,3,3)]],
[3, 3, 1, 1, ["asset1", calculateAsset1Recreational(3,1,1)]],
[3, 3, 1, 2, ["asset1", calculateAsset1Recreational(3,1,2)]],
[3, 3, 1, 3, ["asset1", calculateAsset1Recreational(3,1,3)]],
[3, 3, 2, 1, ["asset1", calculateAsset1Recreational(3,2,1)]],
[3, 3, 2, 2, ["asset1", calculateAsset1Recreational(3,2,2)]],
[3, 3, 2, 3, ["asset1", calculateAsset1Recreational(3,2,3)]],
[3, 3, 3, 1, ["asset1", calculateAsset1Recreational(3,3,1)]],
[3, 3, 3, 2, ["asset1", calculateAsset1Recreational(3,3,2)]],
[3, 3, 3, 3, ["asset1", calculateAsset1Recreational(3,3,3)]],

##asset2
## rules for asset2 residential
## index 0 = 4 represents that it is residential (asset2) and the following 3 indices re
[4, 1, 1, 1, ["asset2", calculateAsset2Residential(1,1,1)]],
[4, 1, 1, 2, ["asset2", calculateAsset2Residential(1,1,2)]],
[4, 1, 1, 3, ["asset2", calculateAsset2Residential(1,1,3)]],
[4, 1, 2, 1, ["asset2", calculateAsset2Residential(1,2,1)]],
[4, 1, 2, 2, ["asset2", calculateAsset2Residential(1,2,2)]],
[4, 1, 2, 3, ["asset2", calculateAsset2Residential(1,2,3)]],
[4, 1, 3, 1, ["asset2", calculateAsset2Residential(1,3,1)]],
[4, 1, 3, 2, ["asset2", calculateAsset2Residential(1,3,2)]],
[4, 1, 3, 3, ["asset2", calculateAsset2Residential(1,3,3)]],
[4, 2, 1, 1, ["asset2", calculateAsset2Residential(2,1,1)]],
[4, 2, 1, 2, ["asset2", calculateAsset2Residential(2,1,2)]],
[4, 2, 1, 3, ["asset2", calculateAsset2Residential(2,1,3)]],
[4, 2, 2, 1, ["asset2", calculateAsset2Residential(2,2,1)]],
[4, 2, 2, 2, ["asset2", calculateAsset2Residential(2,2,2)]],
[4, 2, 2, 3, ["asset2", calculateAsset2Residential(2,2,3)]],
```

```
[4, 2, 3, 1, ["asset2", calculateAsset2Residential(2,3,1)]],
[4, 2, 3, 2, ["asset2", calculateAsset2Residential(2,3,2)]],
[4, 2, 3, 3, ["asset2", calculateAsset2Residential(2,3,3)]],
[4, 3, 1, 1, ["asset2", calculateAsset2Residential(3,1,1)]],
[4, 3, 1, 2, ["asset2", calculateAsset2Residential(3,1,2)]],
[4, 3, 1, 3, ["asset2", calculateAsset2Residential(3,1,3)]],
[4, 3, 2, 1, ["asset2", calculateAsset2Residential(3,2,1)]],
[4, 3, 2, 2, ["asset2", calculateAsset2Residential(3,2,2)]],
[4, 3, 2, 3, ["asset2", calculateAsset2Residential(3,2,3)]],
[4, 3, 3, 1, ["asset2", calculateAsset2Residential(3,3,1)]],
[4, 3, 3, 2, ["asset2", calculateAsset2Residential(3,3,2)]],
[4, 3, 3, 3, ["asset2", calculateAsset2Residential(3,3,3)]],

## rules for asset2 industrial
## index 0 = 5 represents that it is industrial (asset2) and the following 3 indices rep
[5, 1, 1, 1, ["asset2", calculateAsset2Industrial(1,1,1)]],
[5, 1, 1, 2, ["asset2", calculateAsset2Industrial(1,1,2)]],
[5, 1, 1, 3, ["asset2", calculateAsset2Industrial(1,1,3)]],
[5, 1, 2, 1, ["asset2", calculateAsset2Industrial(1,2,1)]],
[5, 1, 2, 2, ["asset2", calculateAsset2Industrial(1,2,2)]],
[5, 1, 2, 3, ["asset2", calculateAsset2Industrial(1,2,3)]],
[5, 1, 3, 1, ["asset2", calculateAsset2Industrial(1,3,1)]],
[5, 1, 3, 2, ["asset2", calculateAsset2Industrial(1,3,2)]],
[5, 1, 3, 3, ["asset2", calculateAsset2Industrial(1,3,3)]],
[5, 2, 1, 1, ["asset2", calculateAsset2Industrial(2,1,1)]],
[5, 2, 1, 2, ["asset2", calculateAsset2Industrial(2,1,2)]],
[5, 2, 1, 3, ["asset2", calculateAsset2Industrial(2,1,3)]],
[5, 2, 2, 1, ["asset2", calculateAsset2Industrial(2,2,1)]],
[5, 2, 2, 2, ["asset2", calculateAsset2Industrial(2,2,2)]],
[5, 2, 2, 3, ["asset2", calculateAsset2Industrial(2,2,3)]],
[5, 2, 3, 1, ["asset2", calculateAsset2Industrial(2,3,1)]],
[5, 2, 3, 2, ["asset2", calculateAsset2Industrial(2,3,2)]],
[5, 2, 3, 3, ["asset2", calculateAsset2Industrial(2,3,3)]],
[5, 3, 1, 1, ["asset2", calculateAsset2Industrial(3,1,1)]],
[5, 3, 1, 2, ["asset2", calculateAsset2Industrial(3,1,2)]],
[5, 3, 1, 3, ["asset2", calculateAsset2Industrial(3,1,3)]],
[5, 3, 2, 1, ["asset2", calculateAsset2Industrial(3,2,1)]],
[5, 3, 2, 2, ["asset2", calculateAsset2Industrial(3,2,2)]],
[5, 3, 2, 3, ["asset2", calculateAsset2Industrial(3,2,3)]],
[5, 3, 3, 1, ["asset2", calculateAsset2Industrial(3,3,1)]],
[5, 3, 3, 2, ["asset2", calculateAsset2Industrial(3,3,2)]],
[5, 3, 3, 3, ["asset2", calculateAsset2Industrial(3,3,3)]],

## rules for asset2 recreational
## index 0 = 6 represents that it is recreational (asset2) and the following 3 indices r
[6, 1, 1, 1, ["asset2", calculateAsset2Recreational(1,1,1)]],
```

```
[6, 1, 1, 2, ["asset2", calculateAsset2Recreational(1,1,2)]],
[6, 1, 1, 3, ["asset2", calculateAsset2Recreational(1,1,3)]],
[6, 1, 2, 1, ["asset2", calculateAsset2Recreational(1,2,1)]],
[6, 1, 2, 2, ["asset2", calculateAsset2Recreational(1,2,2)]],
[6, 1, 2, 3, ["asset2", calculateAsset2Recreational(1,2,3)]],
[6, 1, 3, 1, ["asset2", calculateAsset2Recreational(1,3,1)]],
[6, 1, 3, 2, ["asset2", calculateAsset2Recreational(1,3,2)]],
[6, 1, 3, 3, ["asset2", calculateAsset2Recreational(1,3,3)]],
[6, 2, 1, 1, ["asset2", calculateAsset2Recreational(2,1,1)]],
[6, 2, 1, 2, ["asset2", calculateAsset2Recreational(2,1,2)]],
[6, 2, 1, 3, ["asset2", calculateAsset2Recreational(2,1,3)]],
[6, 2, 2, 1, ["asset2", calculateAsset2Recreational(2,2,1)]],
[6, 2, 2, 2, ["asset2", calculateAsset2Recreational(2,2,2)]],
[6, 2, 2, 3, ["asset2", calculateAsset2Recreational(2,2,3)]],
[6, 2, 3, 1, ["asset2", calculateAsset2Recreational(2,3,1)]],
[6, 2, 3, 2, ["asset2", calculateAsset2Recreational(2,3,2)]],
[6, 2, 3, 3, ["asset2", calculateAsset2Recreational(2,3,3)]],
[6, 3, 1, 1, ["asset2", calculateAsset2Recreational(3,1,1)]],
[6, 3, 1, 2, ["asset2", calculateAsset2Recreational(3,1,2)]],
[6, 3, 1, 3, ["asset2", calculateAsset2Recreational(3,1,3)]],
[6, 3, 2, 1, ["asset2", calculateAsset2Recreational(3,2,1)]],
[6, 3, 2, 2, ["asset2", calculateAsset2Recreational(3,2,2)]],
[6, 3, 2, 3, ["asset2", calculateAsset2Recreational(3,2,3)]],
[6, 3, 3, 1, ["asset2", calculateAsset2Recreational(3,3,1)]],
[6, 3, 3, 2, ["asset2", calculateAsset2Recreational(3,3,2)]],
[6, 3, 3, 3, ["asset2", calculateAsset2Recreational(3,3,3)]],

## rules for asset3
## index 0 = 13 represents that it is asset3 and the following 2 indices represent the v
[13, 3, 3, ["asset3", 3]],
[13, 3, "c", ["asset3", 3]],
[13, 3, "b", ["asset3", 2]],

[13, "c", 3, ["asset3", 3]],
[13, "c", "c", ["asset3", 3]],
[13, "c", "b", ["asset3", 2]],

[13, "b", 3, ["asset3", 2]],
[13, "b", "c", ["asset3", 2]],
[13, "b", "b", ["asset3", 1]],

## rules for asset4 nonvegetatedsurface
## index 0 = 7 represents that it is asset4 (non-vegetated) and the following 2 indices
[7, 1, 1, ["asset4", calculateNonVegetatedSurface(1, 1)]],
[7, 1, 2, ["asset4", calculateNonVegetatedSurface(1, 2)]],
[7, 1, 3, ["asset4", calculateNonVegetatedSurface(1, 3)]],
```

```
[7, 2, 1, ["asset4", calculateNonVegetatedSurface(2, 1)]],
[7, 2, 2, ["asset4", calculateNonVegetatedSurface(2, 2)]],
[7, 2, 3, ["asset4", calculateNonVegetatedSurface(2, 3)]],
[7, 3, 1, ["asset4", calculateNonVegetatedSurface(3, 1)]],
[7, 3, 2, ["asset4", calculateNonVegetatedSurface(3, 2)]],
[7, 3, 3, ["asset4", calculateNonVegetatedSurface(3, 3)]],

## rules for asset4 vegetatedsurface
## index 0 = 8 represents that it is asset4 (vegetated) and the following 2 indices repr
[8, 1, 1, ["asset4", calculateVegetatedSurface(1, 1)]],
[8, 1, 2, ["asset4", calculateVegetatedSurface(1, 2)]],
[8, 1, 3, ["asset4", calculateVegetatedSurface(1, 3)]],
[8, 2, 1, ["asset4", calculateVegetatedSurface(2, 1)]],
[8, 2, 2, ["asset4", calculateVegetatedSurface(2, 2)]],
[8, 2, 3, ["asset4", calculateVegetatedSurface(2, 3)]],
[8, 3, 1, ["asset4", calculateVegetatedSurface(3, 1)]],
[8, 3, 2, ["asset4", calculateVegetatedSurface(3, 2)]],
[8, 3, 3, ["asset4", calculateVegetatedSurface(3, 3)]],

## rules for asset4 watersurface
## index 0 = 9 represents that it is asset4 (water) and the following 2 indices represen
[9, 1, 1, ["asset4", calculateWaterSurface(1, 1)]],
[9, 1, 2, ["asset4", calculateWaterSurface(1, 2)]],
[9, 1, 3, ["asset4", calculateWaterSurface(1, 3)]],
[9, 2, 1, ["asset4", calculateWaterSurface(2, 1)]],
[9, 2, 2, ["asset4", calculateWaterSurface(2, 2)]],
[9, 2, 3, ["asset4", calculateWaterSurface(2, 3)]],
[9, 3, 1, ["asset4", calculateWaterSurface(3, 1)]],
[9, 3, 2, ["asset4", calculateWaterSurface(3, 2)]],
[9, 3, 3, ["asset4", calculateWaterSurface(3, 3)]],

## rules for asset5
## index 0 = 12 represents that it is asset5 and the following 2 indices represent the v
[12, 3, 3, ["asset5", 3]],
[12, 3, "c", ["asset5", 2]],
[12, 3, "b", ["asset5", 2]],

[12, "c", 3, ["asset5", 3]],
[12, "c", "c", ["asset5", 2]],
[12, "c", "b", ["asset5", 1]],

[12, "b", 3, ["asset5", 2]],
[12, "b", "c", ["asset5", 1]],
[12, "b", "b", ["asset5", 1]],

## rules for vulnerability
```

```
## index 0 = 10 represents that it is vulnerability and the following 3 indices represen
[10, 1, 3, 3, ["vulnerability", 5]],
[10, 1, 3, "a", ["vulnerability", 4]],
[10, 1, 3, 0, ["vulnerability", 2]],

[10, 1, "c", 3, ["vulnerability", 4]],
[10, 1, "c", "a", ["vulnerability", 3]],
[10, 1, "c", 0, ["vulnerability", 2]],

[10, 1, "b", 3, ["vulnerability", 2]],
[10, 1, "b", "a", ["vulnerability", 3]],
[10, 1, "b", 0, ["vulnerability", 1]],

[10 , 3, 3, 3, ["vulnerability", 5]],
[10 , 3, 3, "a", ["vulnerability", 5]],
[10 , 3, 3, 0, ["vulnerability", 3]],

[10, 3, "c", 3, ["vulnerability", 4]],
[10, 3, "c", "a", ["vulnerability", 4]],
[10, 3, "c", 0, ["vulnerability", 3]],

[10, 3, "b", 3, ["vulnerability", 4]],
[10, 3, "b", "a", ["vulnerability", 4]],
[10, 3, "b", 0, ["vulnerability", 2]],

[10, 5, 3, 3, ["vulnerability", 5]],
[10, 5, 3, "a", ["vulnerability", 5]],
[10, 5, 3, 0, ["vulnerability", 4]],

[10, 5, "c", 3, ["vulnerability", 5]],
[10, 5, "c", "a", ["vulnerability", 4]],
[10, 5, "c", 0, ["vulnerability", 3]],

[10, 5, "b", 3, ["vulnerability", 4]],
[10, 5, "b", "a", ["vulnerability", 4]],
[10, 5, "b", 0, ["vulnerability", 3]],

## rules for risk
## index 0 = 11 represents that it is risk and the following 2 indices represent the va
[11, 5, 1, ["risk", "VH"]],
[11, 5, 3, ["risk", "VH"]],
[11, 5, 5, ["risk", "H"]],

[11, 4, 1, ["risk", "VH"]],
[11, 4, 3, ["risk", "H"]],
[11, 4, 5, ["risk", "M"]],
```

```
    [11, 3, 1, ["risk", "VH"]],
    [11, 3, 3, ["risk", "M"]],
    [11, 3, 5, ["risk", "L"]],

    [11, 2, 1, ["risk", "H"]],
    [11, 2, 3, ["risk", "L"]],
    [11, 2, 5, ["risk", "VL"]],

    [11, 1, 1, ["risk", "M"]],
    [11, 1, 3, ["risk", "L"]],
    [11, 1, 5, ["risk", "VL"]],
]
```