Geological Multi-scenario Reasoning

Crystal Chang Din[a], Leif Harald Karlsen[a], Irina Pene[b], Oliver Stahl[a], Ingrid Chieh Yu[a], Thomas Østerlie[c]

[a]Department of Informatics, University of Oslo, Norway
[b]Department of Geosciences, University of Oslo, Norway
[c]Department of computer and information science, NTNU, Norway

Abstract

In the oil and gas industry, during exploration prospect assessment, explorationists rely on ad hoc manual work practices and tools for developing and communicating multiple hypothetical geological scenarios of the prospect. This leaves them with little efficient means to make the fullest use of state of the art digital technologies to communicate and systematically compare and assess different hypothetical geological scenarios before deciding which scenario to pursue. In this paper we present a formal framework for geological multi-scenario reasoning, a novel tool-based method for geologically oriented subsurface evaluation. The methodology applies formal methods and logic-based techniques to subsurface evaluation and expresses interpretive uncertainty as discrete scenarios with branches of potential alternatives. This framework consists of (i) a proto-scenario generator that takes user observations and geological evidence as input and generates semantically valid initial states based on formalized geological knowledge in first-order logic (ii) geological processes formalized as a rewrite theory that are executable in Maude. By applying geological rewrite rules onto the proto-scenarios, we are able to assist explorationists with multi-scenario generation and reasoning beyond human capacity.

1 Introduction

This paper presents a novel method and tool developed with the aim to assist geologists, in particular explorationists in their subsurface evaluation process and in their exploration for new subsurface petroleum resources. This framework is developed as part of a larger interdisciplinary research project on geological multi-scenario reasoning taking place at the SIRIUS research based innovation center.

A key challenge for exploration in particular, but also subsurface evaluation in general, is that geodata are "uncertain, intermittent, sparse, multiresolution, and multi-scale" [6]. Geodata underdetermine concrete models and interpretations [11].

This paper was presented at the NIK-2019 conference; see http://www.nik.no/.

This means that even though geodata may refute particular interpretations of the subsurface, they can never verify the correctness of a single interpretation. As such, it is common for different geoscientists to hold widely differing interpretations of available data [2, 3]. Advances in digital tool support for quantitatively-oriented geodata interpretation over the past decades have improved geoscientists' capacity to delineate closed structures and potential traps indicating possible hydrocarbon deposits in the subsurface [13]. However, available geodata provide limited information about critical factors for exploration; whether source rock is present and mature, the presence and quality of reservoir rock, the presence of an effective seal, and whether or not there are hydrocarbon accumulations in subsurface structures observed in the data. Geoscientists therefore supplement these quantitative methods with qualitatively oriented forms of reasoning we refer to as geological history reasoning. Geological history reasoning makes active use of data underdetermination as a source of knowledge to develop and assess multiple hypothetical scenarios of the subsurface to justify whether, where, and (importantly) why there can be hydrocarbon deposits in a prospect through many-faceted geological scenarios explaining available data in terms of sequences of interrelated geological processes [10]. Despite its centrality in exploring for new subsurface hydrocarbon resources, this qualitative form of reasoning remains almost completely unsupported in the digital tool set currently available for geoscientists [12]. Instead, they use pen and paper along with generic digital drawing and presentation tools for developing and communicating multiple hypothetical geological scenarios. With little to no efficient means to make the fullest use of state of the art digital technologies to systematically compare and assess different hypothetical geological scenarios before deciding which scenario to pursue when assessing exploration prospects, geoscientists often end up limiting themselves to only assessing a few possible scenarios due to the time constraints of commercial petroleum exploration.

Through geological multi-scenario reasoning we apply formal methods and logic-based techniques to express the space of possible interpretations as discrete scenarios with branches of potential alternatives. Through systematic representation, analysis, and comparison of different geological scenarios the goal is to support geoscientists to reason beyond human capacity. In this paper, we present the formal framework behind such tool-supported geological reasoning. This framework consists of (i) a formalization of geological domain knowledge [8] in first-order logic (FOL) that generates *proto-scenarios*, i.e., expanded semantically valid geological descriptions based on geologists' observations and geological evidence (ii) a formal specification of geological processes in rewriting logic (RL) [1, 9] that can be executed in Maude [1]. With RL, we capture geological entities, concurrent geological process, and distributed geological states. With FOL, we address complex geometrical relationships and relative timing. By applying geological rewrite theory that we have developed with geologists onto the proto-scenarios, we are able to assist explorationists with multi-scenario reasoning. The combination of these two reasoning methods having rigorous semantics allow us to explore, generate and analyse multi-scenarios to a depth that is otherwise impossible to achieve. Finally, the methodology and tool chain are designed in such a way that allows deployment on a scalable technology.

We have organized the remainder of the paper as follows. First we outline petroleum system analysis, a particular form of subsurface assessment that we have

used as use case for developing our framework. We then progress to outlining the formalization of geological processes in rewriting logic, how we represent geological knowledge, and the scalable infrastructure for running multiple geological scenarios. Finally, we outline the use case demonstration, before drawing the conclusion.

2 Petroleum System Analysis

A main objective for explorationists is to make decisions about whether and where to drill given limited and sparse subsurface data. This decision is made based on detailed petroleum system analysis of the area of interest.

Petroleum system is the framework used in substantiating the possibility of hydrocarbons accumulation in an area. It is defined as a unifying concept that

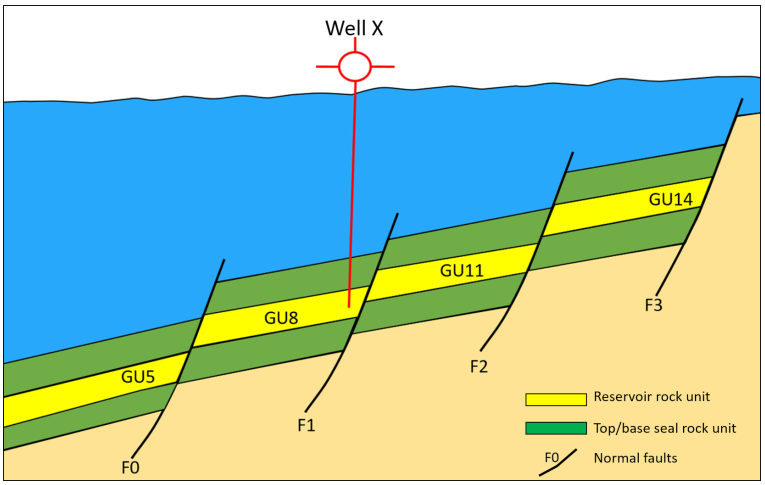

Figure 1: The geological setting for the use case, showing the four rotated fault blocks, with the three geological units that are making them up (reservoir rock units, top/base seal rock units) and the bounding faults

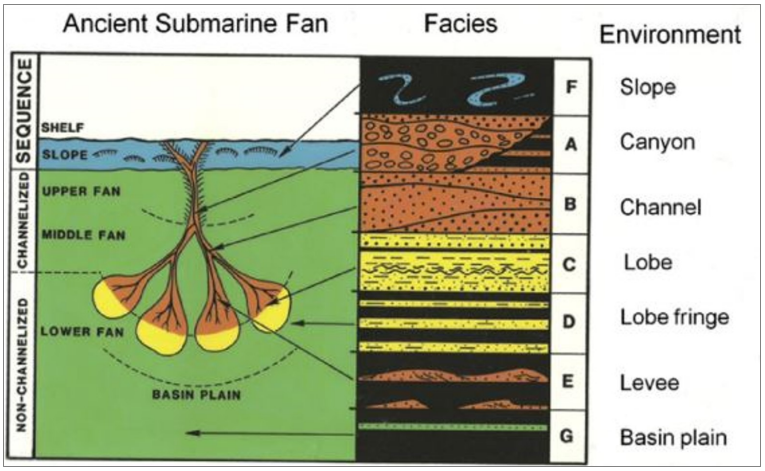

Figure 2: Ancient Submarine Fan, showing the main environments and their corresponding facies, in which the four fault blocks, representing the use case, have been deposited in (from proximal to distal: canyon, channel, levee, lobe, lobe fringe and basin plain). Canyon, channel and lobe represent the reservoir rock units, and basin plain represent the seal rock units.

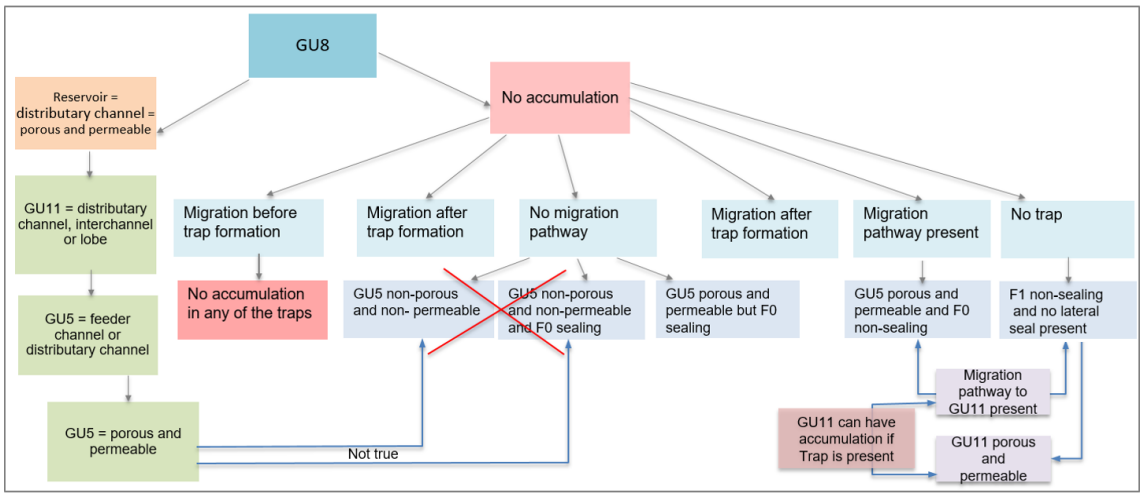

Figure 3: Manual reasoning of the use case

encompasses all of the disparate elements and processes of petroleum geology. The essential elements of a petroleum system include: source rock, reservoir rock, cap rock and overburden rock. Petroleum systems have two processes: trap formation and generation–migration–accumulation of hydrocarbons. If one of these elements or processes are not present than there is no petroleum system.

Use Case The use case chosen is addressed to explorationists in their process of *lead maturation*. The geological settings comprise a series of four rotated fault blocks, consisting of three geological units: base seal, reservoir and top seal (Figure 1), deposited in a marine environment, as a submarine fan (Figure 2). Source rock is present and has generated oil and gas, traps are fault-traps relying on faults being sealing or on lateral seal being present. Useful information is provided by the well X (marked by a red cross), drilled in the area of interest that did not encounter any hydrocarbons but the core and logs acquired tell us that the reservoir is present, of good quality and deposited in a distributary channel.

To demonstrate the necessity and usefulness of a tool for automated reasoning, a human reasoning example is shown in Figure 3. This example will answer the question: "Is it possible to have hydrocarbons accumulation in GU11, while having no hydrocarbons accumulation in GU8? The manual reasoning starts from analysing the data provided by the well, explaining why there is no hydrocarbons accumulation in GU8. After finding out the answer to this first question we proceed explaining under what circumstances there can be an accumulation in GU11. This might be deduced from this decision tree, however, the more complex the use case, the harder the reasoning gets. For certain, the manual reasoning is not always fail-free or complete.

3 Formalization of Geological Processes in Rewriting Logic

Rewriting logic is a computational logic and can be used as a semantic and logical framework. It was introduced in the 1990s and has been applied in various domains, such as hardware, software, logic systems, and computational systems. A rewrite theory $\mathcal{R}$ = (Σ,E,R) consists of a membership equational logic [5] theory (Σ,E)

and a set of (possibly conditional) rewrite rules R. As a semantic framework, (Σ,E) specifies the static aspects of distributed states. Σ is a set of declarations of sorts, subsorts, and function symbols. In Maude, an implementation of rewriting logic capable of executing rewriting logic theories, a function symbol f is declared with the syntax `op` $f : s_1 ... s_n \rightarrow s$, where $s_1 ... s_n$ are the sorts of its arguments, and s is its (value) sort. For example, we can define constructors of different environments in a submarine fan:

```
sort SubmarineFanEnv .
op   feederChannel       : -> SubmarineFanEnv .
op   distributaryChannel : -> SubmarineFanEnv .
op   interChannel        : -> SubmarineFanEnv .
op   lobe                : -> SubmarineFanEnv .
op   lobeFringe          : -> SubmarineFanEnv .
op   basinPlain          : -> SubmarineFanEnv .
```

E is a set of confluent and terminating (possibly conditional) equations. In Maude, equations are written with syntax `eq` $t = t'$ and `ceq` $t = t'$ `if cond`, where the latter expresses conditional equations. Rewrite rules R specify the dynamic aspects of a concurrent and distributed system, i.e. system's transition from a sub-distributed state to a new sub-distributed state. R is a set of (possibly conditional) labeled rewrite rules of the form `rl` $[l] : t \Rightarrow t'$ and `crl` $[l] : t \Rightarrow t'$ `if cond`, where the latter is a conditional rewrite rule. The rule specifies the system's transitions from an instance of t to the corresponding instance of t', where l is a label. Conditional rules apply only if their conditions hold. Operationally, a term is reduced to its normal form modulo a set of equational axioms before any rewrite rule is applied. This dynamic aspect of rewriting logic captures local concurrent transitions. With several local concurrent transitions performed at the same time, true concurrency is represented.

A *state* or *configuration* denotes a multi-set of objects and messages. The Maude simulation and execution starts with an initial state/configuration.

Rewriting Logic for Geology The evidence accumulated over the last twenty years strongly supports the claim that rewriting logic can rightfully be said to have "ϵ representational distance" [9] as a semantic and logical framework. That is, what is represented and its representation are often isomorphic structures. We exploit this property and together with the geologists, we apply rewriting logic in the domain of geology and define the geological rewriting theory $\mathcal{R}_{geo}$. Given a subsurface specified as a rewrite theory (Σ,E,R), rewriting logic then allows us to reason about the complex changes that are possible in the geological system, given the basic geological changes modeled as rewrite rules R. That is, we can then use (Σ,E,R) together with Maude and its supporting formal tools to simulate, study, and analyze geological dynamics. In particular, we can study in this way complex processes involving chains of geological changes and leading to multi-scenarios of hydrocarbon migration and accumulation.

In Figure 4 we show a conditional rewrite rule, which captures the geological process of hydrocarbon migration and accumulation through a fault. All the capitalized terms are variables. For example, GU_1 and GU_2 are placeholders for the identity numbers of geological units. RT is the placeholder for the type of

```
crl [migration-through-fault-fillToMaxClosure-accumulation-rule] :

⟨ GU1 : GeoUnit | Type: RT, Permeability: PMT1, Porosity: PRT1, SubmarineFan: SF1,
     DepositedIn: ENV, Hydrocarbon: HC1 ⟩
⟨ GU2 : GeoUnit | Type: sandstone, Permeability: PMT2, Porosity: PRT2,
    SubmarineFan: SF2, DepositedIn: ENV, Hydrocarbon: HC2 ⟩
⟨ N : Pathways | PType: fault(F, GU1, GU2) ⟩
trapformation(GU2, TPT, SPT, TFT)
accumulation(GU2, B)
⟹
⟨ GU1 : GeoUnit | Type: RT, Permeability: PMT1, Porosity: PRT1, SubmarineFan: SF1,
     DepositedIn: ENV, Hydrocarbon: HC1 ⟩
⟨ GU2 : GeoUnit | Type: sandstone, Permeability: PMT2, Porosity: PRT2,
    SubmarineFan: SF2, DepositedIn: ENV, Hydrocarbon: HC1 ⟩
⟨ N : Pathways | PType: fault(F, GU1, GU2) ⟩
trapformation(GU2, TPT, SPT, TFT)
accumulation(GU2, true)
if  HC1 =/= null and PMT2 == permeable and PRT2 == porous
    and (TPT == faultDependent-sealing or TPT == faultDependent-nonSealing)
    and SPT == fillToMaxClosure and isYounger(timeOf(Migration), TFT) .
```

Figure 4: A Rewrite Rule for Hydrocarbon Migration and Accumulation

geological unit such as shale or sandstone. PMT_1 and PMT_2 are the placeholders for the permeability of reservoirs such as permeable or non-permeable. PRT_1 and PRT_2 are placeholders for the porosity of reservoirs such as porous or non-porous. SF_1 and SF_2 are the placeholders for the environments of submarine fan such as feeder channel, distributary channel, inter channel, lobe, lobe fringe, or basin plain. ENV is the placeholder for the depositional environment. HC_1 and HC_2 are the placeholders for the identity numbers of hydrocarbon objects, i.e. **null** if hydrocarbon does not exist. F is a placeholder for the identity number of a fault. TPT is a placeholder capturing the type of traps. SPT is a placeholder capturing the maximum hydrocarbon accumulation level. TFT is the placeholder capturing the trap-formation time. B is a Boolean variable. This is a conditional rule. This rule can only be applied if GU_1 contains hydrocarbon, GU_2 is permeable and porous, the trap for GU_2 is fault-dependent and was formed before the hydrocarbon migration happened, and there is a migration pathway from GU_1 to GU_2 through fault F. After this rule is successfully applied, the state of the configuration expresses that GU_2 accumulates the same type of hydrocarbon as GU_1.

4 Knowledge Representation

In order to use rewriting logic for simulating these complex geological processes, we need to start with an initial state. In most cases, the end-user does not have complete knowledge about the original state of the system to simulate. This leads to an underdetermined description of the initial state, in which many initial states are possible. To determine which state is a possible state that makes sense for the given domain requires reasoning on a combination of the end-users knowledge about this specific case, and general domain knowledge about what is a meaningful state. However, this knowledge is purely static, i.e. it concerns only individual states, and not the evolution of one state to another. The latter is achieved by the runtime application of $\mathcal{R}_{geo}$.

To automate this task, we formalize both the general domain knowledge and

```
1 above(E1, E2) :- ontopof(E1, E2) .
2 above(E1, E2) :- ontopof(E1, E), above(E, E2) .
```

Figure 5: Example Prolog program defining the `above` relation.

the knowledge specific for this case, in such a way that a machine can reason on this knowledge and determine which state makes sense and which does not. Such formalization of knowledge is normally done using logic, typically first order logic, a modal logic, or a description logic (see e.g. [14] for an overview). For our purposes we have chosen to formalize the geological knowledge in Prolog [4]. Prolog can be viewed as both a declarative general-purpose programming language, and a first-order logical language for formalizing knowledge. In Prolog, knowledge is represented via facts and simple implications, both terminated by a dot. Facts have the form

$$\mathtt{H}(\vec{x}) \ .$$

and the implications have the form

$$\mathtt{H}(\vec{x}) \ \mathtt{:-}\ \mathtt{B}_1(\vec{y_1}), \mathtt{B}_2(\vec{y_2}), \ldots, \mathtt{B}_n(\vec{y_n}) \ .$$

where `H` and each $\mathtt{B}_i$ are relation names and $\vec{x}$ and each $\vec{y_i}$ are tuples of terms (constants or varibles). Furthermore, `:-` denotes left implication (i.e. $\leftarrow$) and comma is conjunction (i.e. $\wedge$). Negation (with negation-as-failure semantics [4]) is written `\+`. Variables are denoted by a capital first letter, and everything else are constants. Figure 5 presents an example Prolog program that defines the `above` relation as the transitive closure of the `ontopof` relation, where `ontopof(A, B)` states that `A` is on top of `B`.

The Prolog formalization is responsible for taking the end-users knowledge about a scenario, and generate all possible initial states with respect to the temporal relationships between the elements. These temporal relationships can be determined from the objects' geometrical relationships. For example, a layer that is geometrically above another layer is also younger. For more complex relations, see the example below.

Our formalization could also have been done in a different logical language, such as OWL [7] (a type of description logic). OWL is more commonly used but less expressive. We plan to investigate the possibility to move (part of) the formalization to OWL in the future, to be able to better reuse already existing formalizations.

Example One important part of determining all possible proto-scenarios is to find the possible temporal relationships one can have between the events forming the different elements of a geological system. For example, we know that if a fault cuts a particular layer, then the layer is formed before the event that produced the fault. As noted above, a fault is younger than another if its cross-cutting topmost layer is above the cross-cutting topmost layer of the other. Accordingly, we define the approximate time of the formation of a fault by finding out the topmost layer it cuts. Both the Prolog-formalization of this relation between a fault and its topmost

```
top_layer(Fault, Layer) :-
    fault(Fault), geological_unit(Layer),
    goes_through(Fault, Layer),
    \+ (ontopof(OtherLayer, Layer),
        goes_through(Fault, OtherLayer)) .

younger_than(F1, F2) :-
    fault(F1), fault(F2),
    top_layer(F1, R1), top_layer(F2, R2),
    above(R1, R2) .
```

Figure 6: Prolog formalization of temporal knowledge.

```
geological_unit(layer1) .    fault(fault1) .
geological_unit(layer2) .    fault(fault2) .
geological_unit(layer3) .    goes_through(fault1, layer3) .
ontopof(layer1, layer2) .    goes_through(fault2, layer3) .
ontopof(layer2, layer3) .    goes_through(fault2, layer2) .
```

Figure 7: Prolog formalization of end-user knowledge.

layer and the temporal order of faults is presented as `top_layer` and `younger_than`, respectively, in Figure 6. The Prolog formalization of the former can be read as: "`Fault` has top-layer `Layer` if `Fault` is a fault, `Layer` is a geological unit, `Fault` goes through `Layer`, and there does not exist a layer that is on top of `Layer` which `Fault` goes through".

In Figure 7 we can see an example knowledge base describing some known facts input by the end-user. From these facts and the definitions of Figure 6 we can e.g. derive that `fault2`'s topmost layer is `layer2` and `fault2` is younger than `fault1`.

5 Scalable Infrastructure

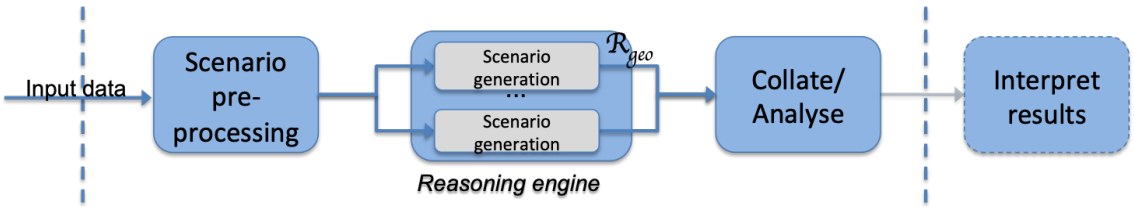

Figure 8: Work flow

In order to run the sequence of tools in a reliable and efficient way, we need to establish a pipeline, see Figure 8, that encapsulates and automates all the different steps of the process and at the same time is able to scale the simulations to shorten computing time. As a secondary priority the preparation and installation time on systems supposed to run this pipeline should be kept as small as possible.

The decision to use rewriting logic in the simulation part of the pipeline enables us to scale and run the simulations on different systems in parallel. This allows using affordable standard or cloud-based hardware while still keeping the required time per use case within reasonable bounds. The chosen approach is to establish a computation framework which consists of a controller and several worker nodes. The controller dictates and schedules the workflow while being agnostic of any domain knowledge. All computation work is done by the worker nodes which have the necessary domain knowledge and all required tools installed. Each worker node that is started automatically registers with its controller. The controller exposes a few webservices to allow user interactions such as starting a computation with a given set of data, collecting results afterwards, as well as giving access to some status information. The first step taken in the pipeline is the creation of proto-scenarios from the input data which was given into the system when submitting the use case. After this is successfully done, the number of simulations that have to be run is known to the system for the first time. According to this number, work packages for the proto-scenarios are prepared and queued. The controller then starts to asynchronously distribute these work packages to all available and currently idle worker nodes in set intervals. Each of the nodes then runs a simulation based on the proto-scenario it received with the work package from the controller and returns a final configuration as result after finishing the simulation. Each result/final configuration is reported back to the controller and also persisted into a database for later evaluation. After all work packages (all simulations) are completed, the resulting data can be collected by the user by calling the appropriate webservice.

The controller as well as the worker nodes are implemented as Java Spring applications and use webservices to communicate to each other and the outside world. The database currently used is PostgreSQL, but could basically be any relational database that can be used in connection with the Spring Framework and JPA (Java Persistence API). To increase portability and reduce the effort for installation and configuration, controller and worker nodes are encapsulated into Docker images (with the worker node image also containing everything necessary to create the proto-scenarios and run simulations). All worker nodes use the same implementation and Docker image, only being different in the configuration of the Docker container created when first started (namely container name and port claimed on the host machine, plus IP address in cases where different machines are used). This allows us to keep administrative work when deploying the pipeline environment to a minimum.

6 Use Case: multi-scenario analysis of hydrocarbon migration and accumulation

This section shows the analysis results of the use case presented in Section 2. Note that a geological unit GU defined in Section 2 is presented as GeoUnit in our tool. Based on the facts about depositional environment, the observation of geometrical relations, and the assumptions of sealing capacity, hydrocarbon migration time, and reservoir locations, our reasoning engine generates 3024 proto-scenarios for our use case as the input configurations (i.e., initial states) for the Maude engine. Multi-scenarios of geological processes are computed by Maude simulation. Currently, the total execution time, including the time for generating all 3024 proto-scenarios plus the complete simulation time, using 4 worker nodes in parallel on the same

```
**********************************************************************************************************************************************************
GeoUnit 4 is the source rock.
--HYDROCARBON--
The type of hydrocarbon is oil&gas.
--MIGRATION PATHWAY--
GeoUnit 4 -> GeoUnit 5 -> Fault 0 -> GeoUnit 8
--SUBMARINE FAN--
GeoUnit 5 was deposited in feederChannel.
GeoUnit 5 is permeable and porous.
GeoUnit 8 was deposited in distributaryChannel.
GeoUnit 8 is permeable and porous.
--FAULT TYPE--
Fault 0 is non-sealing.
Fault 1 is non-sealing.
--MIGRATION TIME--
Migration happened after Fault 0 was ceased.
Migration happened after Fault 1 was ceased.
--TRAP--
GeoUnit 5 can be trapped and the trapped was formed when Fault 0 was ceased.
<Reason>: There is fault-dependent trap-formation for GeoUnit 5 by #topSeal and #lateralSeal formed completely by shale.
Besides, GeoUnit 5 is permeable and porous.
GeoUnit 8 cannot be trapped.
<Reason>: There is no trap-formation for GeoUnit 8 because Fault 1 is non-sealing and neighboring GeoUnit 11 is permeable and porous.
Even though GeoUnit 8 is permeable and porous.
--ACCUMULATION--
GeoUnit 5 has accumulation.
  The accumulation in GeoUnit 5 can be filled to maximum closure.
GeoUnit 8 does not have accumulation.
--HISTORY OF HYDROCARBON MIGRATION AND ACCUMULATION--
oil&gas was generated by Kerogen Type II in GeoUnit 4
pathway formation from GeoUnit 4 to GeoUnit 5 through rocks in vertical contact
oil&gas migrated through rocks in-contact from GeoUnit 4 to GeoUnit 5 and there was fill-to-Max-closure fault-dependent accumulation in GeoUnit 5
pathway formation through fault 0 from GeoUnit 5 to GeoUnit 8 when rocks are not in-contact
oil&gas migrated through fault 0 from GeoUnit 5 to GeoUnit 8 but there was no accumulation in GeoUnit 8 because NO TRAP COULD BE FORMED FOR GEOUNIT 8
**********************************************************************************************************************************************************
```

Figure 9: An Example of Scenario Explanation

machine is about 77 minutes. We observe that the number of scenarios for such a small use case is already pretty high. This shows that the comprehension of the geologically oriented subsurface evaluation is likely beyond human capacity and requires digital support. Therefore, we design an interactive tool interface that provides users various options to constrain and aggregate the scenarios computed by Maude. The more constraints given, the less scenarios remained for investigation. Take an example of our use case, by giving three constraints: (i) GeoUnit 8 is in distributary channel, (ii) GeoUnit 8 does not have accumulation, and (iii) migration happened *after* the trap was formed, the number of scenarios remained is reduced from 3024 to 14. Based on the observations, evidence (i) and (ii), and assumption (iii), these 14 scenarios explain all the possible reasons of *why* GeoUnit 8 in our use case is a dry well. Figure 9 shows one of the scenario explanations by our engine among these 14 scenarios. The yellow part highlights the observation, which shows the location of the source rock and the type of the hydrocarbon. The green parts highlight the evidence (i) and (ii). The blue part highlights the assumption (iii), i.e. a trap is formed when the corresponding fault is ceased. In addition, the engine is capable of providing users the following information: the hydrocarbon migration pathway, the sealing capacity of faults, how a reservoir is trapped or why a reservoir cannot be trapped, whether a reservoir has hydrocarbon accumulation or not and the maximum accumulation volume if any, and the simulation history of hydrocarbon migration and accumulation. By continuing with the same approach of constraining scenarios, our tool shows not only that it is possible to find hydrocarbon in the surrounding area of the dry well GeoUnit 8 but also explains *why* it is possible and *how* the hydrocarbon can be accumulated. Figure 10 is a screenshot of the tool that shows under which constraints we may find accumulation in GeoUnit 11 and the number of scenarios is reduced from 14 to 8. The corresponding explanations of these 8 scenarios are provided by our tool and the format is similar to the one shown in Figure 9. Note that during the process of constraining scenarios, if none of the remaining scenarios satisfies the given constraint, the tool shows *"No cases are matched"*.

In order to assist the users during the process of constraining scenarios, our tool provides multiple options for selecting and seeing the variations of a specific

```
8 scenarios.

############## What you have chosen to keep in the search space: ##############
(1) GeoUnit 8 is in distributaryChannel.
(2) GeoUnit 8 does not have accumulation.
(3) Migration happened AFTER the chosen fault 1 was ceased.
(4) GeoUnit 11 has accumulation.
################################################################################
```

Figure 10: The tool shows that it is possible to find accumulation in GeoUnit 11

```
**************************************************
GeoUnit 8 was deposited in distributaryChannel.
GeoUnit 8 is permeable and porous.
**************************************************

1 variation.
```

```
**************************************************
GeoUnit 11 was deposited in distributaryChannel.
GeoUnit 11 is permeable and porous.
**************************************************

**************************************************
GeoUnit 11 was deposited in interChannel.
GeoUnit 11 is non-permeable and non-porous.
**************************************************

**************************************************
GeoUnit 11 was deposited in lobe.
GeoUnit 11 is permeable and porous.
**************************************************

3 variations.
```

Figure 11: An example of how the tool shows the variations of a target category

target category, for example the user can choose to see only the submarine fan environment and the corresponding permeability and porosity of a reservoir, or all the possible migration pathways up to a certain reservoir, or the sealing capacity of a specific fault, or the trap-formation of a reservoir, etc. These supplementary information pinpoint the evidence across scenarios based on the already given constraints. Figure 11 shows an example of how our tool presents these variations. While GeoUnit 8 is in the distributary channel, GeoUnit 11 can only be in either distributary channel, inter-channel or lobe.

7 Conclusion

In this paper we have shown how our method and tool can support geological reasoning through historical narratives. By exploring, explaining and constraining scenarios based on observations, evidence and assumptions, we are able to assist explorationists in communicate and systematically compare and assess different hypothetical geological scenarios before deciding which scenario to pursue when assessing exploration prospects. Our methodology provides a qualitative form of reasoning and addresses a gap that is currently existing between geologically oriented work processes and the digital tools available.

The framework offers a novel application of rewriting logic that holds the potential of extending the scope of formal methods beyond the conventional domain of computing.

Acknowledgment The authors would like to thank Hallgrim Ludvigsen from Schlumberger, Michael Heeremans from the Department of Geosciences at University of Oslo, Adnan Latif from SIRIUS, and Vegar Skaret for their fruitful discussions and constructive feedbacks.

References

[1] L. Bachmair, editor. *Rewriting Techniques and Applications, 11th International Conference, RTA 2000, Norwich, UK, July 10-12, 2000, Proceedings*, volume 1833 of *Lecture Notes in Computer Science*. Springer, 2000.

[2] C. E. Bond. Uncertainty in structural interpretation: Lessons to be learnt. *Journal of Structural Geology*, 74:185 – 200, 2015.

[3] C. E. Bond, A. Gibbs, Z. Shipton, and S. Jones. What do you think this is? "Conceptual uncertainty" In geoscience interpretation. *GSA Today*, 17(11):4–10, 12 2007.

[4] I. Bratko. *Prolog programming for artificial intelligence*. Pearson education, 2001.

[5] M. Clavel, F. Durán, S. Eker, P. Lincoln, N. Martí-Oliet, J. Meseguer, and C. L. Talcott, editors. *All About Maude - A High-Performance Logical Framework, How to Specify, Program and Verify Systems in Rewriting Logic*, volume 4350 of *Lecture Notes in Computer Science*. Springer, 2007.

[6] Y. Gil, S. A. Pierce, H. Babaie, A. Banerjee, K. Borne, G. Bust, M. Cheatham, I. Ebert-Uphoff, C. Gomes, M. Hill, J. Horel, L. Hsu, J. Kinter, C. Knoblock, D. Krum, V. Kumar, P. Lermusiaux, Y. Liu, C. North, V. Pankratius, S. Peters, B. Plale, A. Pope, S. Ravela, J. Restrepo, A. Ridley, H. Samet, and S. Shekhar. Intelligent systems for geosciences: An essential research agenda. *Commun. ACM*, 62(1):76–84, Dec. 2018.

[7] P. Hitzler, M. Krötzsch, B. Parsia, P. F. Patel-Schneider, and S. Rudolph. Owl 2 web ontology language primer. *W3C recommendation*, 27(1):123, 2009.

[8] L. Mastella, M. Perrin, Y. Aït-Ameur, M. Abel, and J.-F. Rainaud. Formalizing geological knowledge through ontologies and semantic annotation. 4:2473–2477, 06 2008.

[9] J. Meseguer. Twenty years of rewriting logic. *The Journal of Logic and Algebraic Programming*, 81(7-8):721–781, 2012.

[10] E. Monteiro, T. Østerlie, E. Parmiggiani, and M. Mikalsen. Quantifying quality: Towards a post-humanist perspective on sensemaking. pages 48–63, Cham, 2018. Springer International Publishing.

[11] N. Oreskes, K. Shrader-Frechette, and K. Belitz. Verification, validation, and confirmation of numerical models in the earth sciences. *Science*, 263(5147):641–646, 1994.

[12] T. Østerlie, E. Parmiggiani, and E. Monteiro. Information infrastructure in the face of irreducible uncertainty. In *In 5th Innovation in Information Infrastructures (III) Workshop, 7th-9th November, Rome.*, 11 2017.

[13] A. K. Turner and C. W. Gable. A review of geological modeling. 01 2007.

[14] F. Van Harmelen, V. Lifschitz, and B. Porter. *Handbook of knowledge representation*, volume 1. Elsevier, 2008.