Progress in Artificial Intelligence (2021) 10:245–258
https://doi.org/10.1007/s13748-020-00227-x

REGULAR PAPER

# Fault diagnosis under uncertain situations within a Bayesian knowledge-intensive CBR system

Hoda Nikpour[1] · Agnar Aamodt[1]

Received: 12 April 2020 / Accepted: 6 December 2020 / Published online: 4 March 2021


**Abstract**
This paper presents fault diagnosis and problem solving under uncertainty by a Bayesian supported knowledge-intensive case-based reasoning (CBR) system called BNCreek. In this system, the main goal is to diagnose the causal failures behind the symptoms in complex and uncertain domains. The system's architecture is described in three aspects: the general, structural, and functional architectures. The domain knowledge is represented by formally defined methods. An integration of semantic networks, Bayesian networks, and CBR is employed to deal with the domain uncertainty. An experiment is conducted from the oil well drilling domain, which is a complex and uncertain area as an application domain. The system is evaluated against the expert estimations to find the most efficient solutions for the problems. The obtained results reveal the capability of the system in diagnosing causal failures.

**Keywords** Bayesian network · CBR · Knowledge intensive system · Uncertain domains

## 1 Introduction

Fault diagnosis is a critical task in the problem-solving process for uncertain domains. It helps to predict the failures and prepares a reliable maintenance time. As the study field becomes more uncertain, the fault diagnosis becomes more complex and very costly. For example, in petroleum engineering, sometimes a small fault may cause severe disruptions or damages to the equipment. Therefore, developing practical and effective systems to handle the uncertainties becomes an imperative and critical task.

BNCreek, as a knowledge-intensive system, combines case-based reasoning (CBR) and Bayesian networks (BN) in order to conduct fault diagnosis and problem-solving in uncertain domains.

BNCreek, in parallel with other relevant systems, utilizes an integrated inference and reasoning method consisting of the Bayesian network, Semantic network, and CBR methods to conduct the fault diagnosis and to handle the challenge of problem solving in uncertain domains.

Bayesian networks have shown good efficiency as a diagnosis method. However, it is built based on well-defined parts of the domain, and increasing the network dimensions exponentially increases the complexity of inference [1–3]. The CBR method conducts a similarity-based inference process from the input cases to the stored ones and moderates the BN inference method's limitations while keeping its advantages. Finally, a semantic network that captures taxonomical and other structural relations are combined with these two methods.

The structure of the paper is as follows: Sect. 2 discusses a number of related systems. Section 3 describes the architecture of the system in the three aspects of general, structural, and functional architectures. Section 4 explains how the three knowledge representers, i.e., the semantic network, Bayesian network, and the cases, are employed to represent the knowledge in the system. The problem-solving algorithm and the reasoning methods integration are described in Sect. 5. In the first five sections, a simple domain of cooking faults is used for illustration purposes. Section 6 briefly describes the oil well drilling knowledge model addressing our targeted application domain. In Sect. 7 the conducted experiment, the experimental setups and the results are presented to assess the quality of the developed system. Finally, Sect. 8 discusses

✉ Hoda Nikpour
hoda.nikpour@ntnu.no

Agnar Aamodt
agnar@ntnu.no

[1] Norwegian University of Science and Technology, Trondheim, Norway

the detailed advantages and weaknesses of the system and describes how the new mechanisms can improve it. Section 9 concludes the paper.

## 2 Related works

In the literature, several approaches addressed fault-diagnosis challenges and extensively investigated it. Some of the existing methods are based on probabilistic theory. Their common target is to embed intelligence in the problem solving process and increase its accuracy. A brief presentation of some examples that have proposed hybrid approaches by combining Bayesian network analysis with the CBR method follows.

Tirri et al. [4] presented a probabilistic framework for case-based reasoning. Their study was motivated by handling the challenges of reasoning under uncertainty in weak knowledge domains. They used a case base to form a probabilistic model of the problem domain in a way that the cases are viewed as the component distributions that contribute to the joint probability distribution. They provided a case adaptation algorithm by using the Bayesian model and utilized the probabilities as the measures of similarity.

Silvia et al. [5] presented a technique that integrates case-based reasoning and Bayesian networks aimed to build user profiles incrementally. In their system, case-based reasoning provides a mechanism to acquire knowledge about user actions, and BN is used to model the relationships between the elements of interest employing the stored cases.

Bennacer et al. [6] proposed a self-diagnosis approach in communication networks. They aimed to handle the complexity of fault diagnosis techniques and to reduce human intervention in this process. Their approach employs a combination of Bayesian networks as a flexible knowledge representation tool and case-based reasoning to reduce the inference complexity by emphasizing the prior experiences to solve the current problems and increase the accuracy of root cause identification.

Moghaddass et al. [7] developed a mathematical model that provides explanations for the physicians to reason about cases. They aimed to get insight into medical data and increase the accuracy and efficiency of computations and predictions. They employed the nearest neighbor method and the Bayesian-based Patchworks to generate the cases and conduct a case-based reasoning method to find similar previously visited patients. In their system, the nearest neighbor method is set up with flexibility in the number of neighbors and the distance metric between them.

Essam et al. [8] developed a system for network fault diagnosis, aimed to handle its complexity, and conduct efficient management in computer networks. They proposed a hybrid model consisting of three layers. The first layer detects the online faults by finding possible points of occurrence in the network. The second layer models the faults and indicates the abnormal situations using belief networks probabilistic reasoning techniques. The third layer diagnoses the faults by finding similar fault patterns stored in the case-base as a diagnostic case.

Aamodt et al. [9] developed a knowledge-intensive case-based system called TrollCreek. Their system combines the CBR and a model-based reasoning component that utilizes general domain knowledge in problem-solving and learning in open and weak-theory domains.

Pure CBR considers the syntactical similarity assessment between the cases to retrieve the proper cases. The combination of CBR with a semantic network as a knowledge model added the ability of the semantical similarity assessment and formed the knowledge-intensive CBR system [13]. The underlying value propagation and model-based reasoning abilities are provided by the semantic network, but some of its implicit and not formally defined inference and reasoning methods make the system analysis difficult. CBR relies on the previously solved cases and retrieves the most similar ones to utilize in the problem-solving process. If a similar enough case is not found, either CBR will not have any output, or it will retrieve a less similar case; both situations will cause problems to the system performance.

The formally defined inference methods of Bayesian network make inference in a knowledge-intensive CBR system more analyzable and provides more accurate fault diagnosis possibility. The other methods summarized in this section combine CBR with Bayesian network, but we have found no other work that combine CBR and semantic networks with a Bayesian network. The Bayesian method adds uncertainty handling founded in probabilistic theory. It also adds more flexibility and reliability to different sections of the CBR cycle [14] based on its probabilistic knowledge. The earlier work described in Aamodt et al. [15] laid out a high-level framework for BN-supported retrieval and reuse of past cases. Bruland et al. [16] discussed several ways of integrating CBR and Bayesian networks for clinical decision support. However, those were merely theoretical discussions that did not lead to implemented methods.

BNCreek is inspired by TrollCreek and aims to improve it in terms of accuracy and presents a more formal representation by adding Bayesian network analysis. It suggests an improved architecture for the integration of knowledge-intensive methods and problem-solving in uncertain domains. It is a fully integrated hybrid system in which the Bayesian network component and knowledge-intensive CBR component are tied together semantically and influence each other in the problem-solving process. The term hybrid refers to systems that consist of one or more subsystems integrated with different degrees. In BNCreek, as a fully integrated hybrid system, the subsystems, i.e., the Bayesian network component and the knowledge-intensive CBR component share data

structures and knowledge representations, and the reasoning is accomplished cooperatively. The main goal of integrating the Bayesian analysis with TrollCreek includes developing techniques to increase the efficiency and reasoning power of it as integration of other methods, e.g., neural network, has also attempted [17,18].

This paper focuses on estimating any future failures or diagnosing the causality of the system failures in uncertain domains. The problem-solving process in BNCreek is supported by specialized network data structures that show how the data are represented in the system. The system is evaluated by experiments from the petroleum domain as a complex, uncertain area. The domain's knowledge model has been developed in cooperation with Professor Pål Skalle from the Department of Geoscience and Petroleum, NTNU, Norway [19,20].

## 3 Architecture

In a general view, the BNCreek architecture has three main components: a semantic network module, a Bayesian network module, and a case base module.

A semantic network is a graph structure for representing knowledge in patterns of interconnected nodes and arcs [22]. The semantic network module models the real domain as a directed acyclic graph with the nodes representing the domain concepts. Bidirectional edges constitute the relationship between the concepts. Different types of relations are defined to represent the relationship between the concepts, for example structural relation types, like subclass-of, part-of, and causal relations types, i.e., causes and caused by. This module enables the system to conduct semantic network inference and reasoning.

The Bayesian network encodes conditional independence relations between random variables using an acyclic directed graph in which its vertices are the random variables [23]. Besides the semantic network module, the Bayesian network module captures and models the concepts with causal relationships, individually. The nodes represent the domain concepts, and the bidirectional edges represent the causal relationships between the concepts. This module enables the system to perform the Bayesian inferences and reasoning.

The semantic network and the Bayesian network of domain concepts and their relationships form the general domain knowledge model of BNCreek.

A case is a previously experienced situation, which has been captured and learned in such a way that it can be reused in the solving of future problems. The cases are represented by nodes and are connected to the other two modules through the case features. A case feature consists of a concept from the knowledge model, a relation type, and a relevance factor that represents the importance of a feature for a stored case

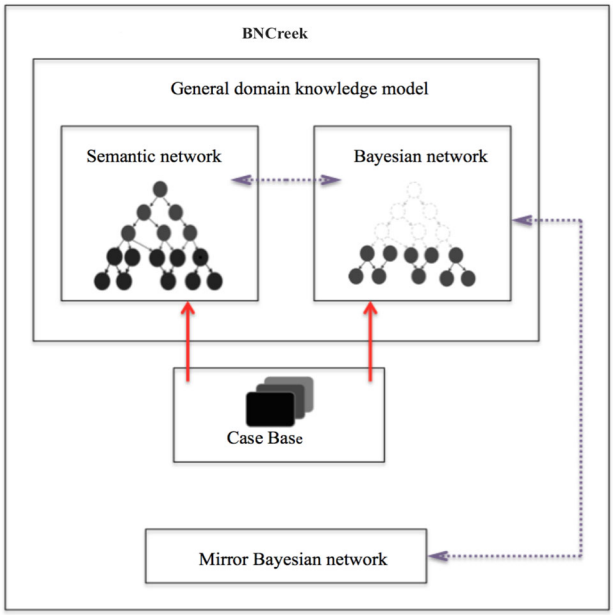


**Fig. 1** The graphical representation of the system's general architecture [21]

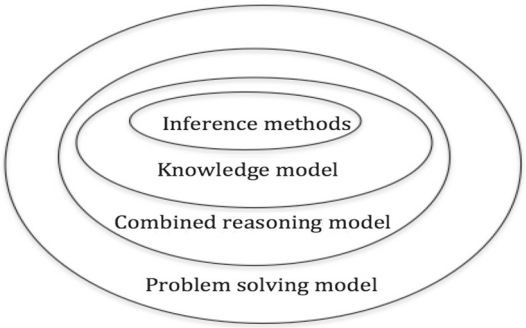


**Fig. 2** The system's structural architecture

[9]. The case base module is a set of cases that are collected to be utilized for inference and reasoning purposes. The tight coupling between the cases and the two networks enables each case to be understood by the system, and the system to perform knowledge intensive case-based reasoning.

Figure 1 illustrates the general architecture. In addition to the three core components of the architecture, the mirror Bayesian network interacts with the Bayesian network module and is responsible for the Bayesian analysis of the computational issues. The mirror Bayesian network is not one of the main components of the structure and is created to keep the implementation complexity low and to provide scalability of the system.

The solid arrows show the connection between the cases and the other concepts. The dotted arrows indicate the information flow between the semantic network, the Bayesian network, and the mirror Bayesian network.

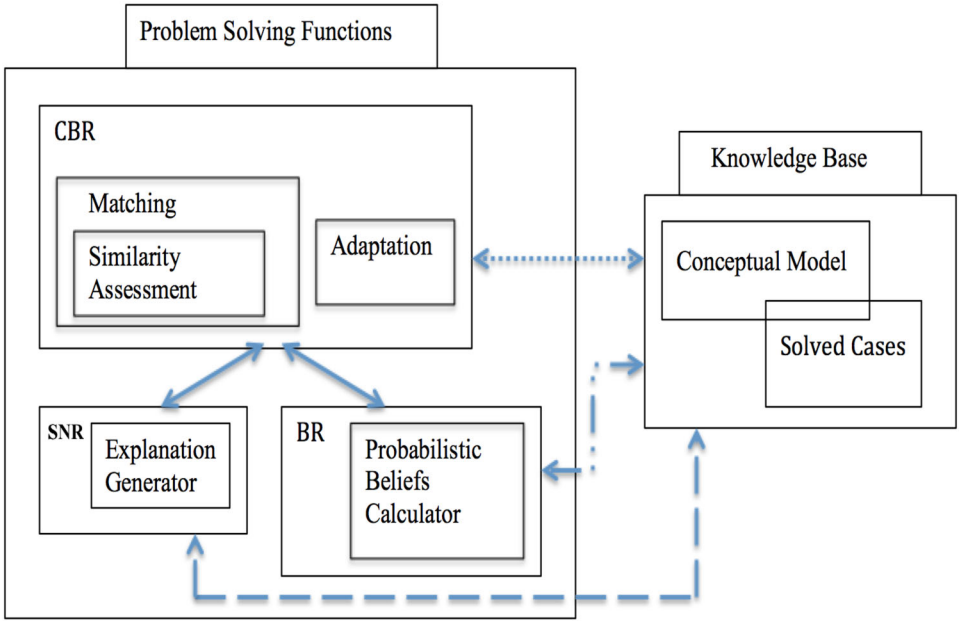


**Fig. 3** The system's functional architecture

### 3.1 Structural architecture

Figure 2 demonstrates the system's structural architecture at the highest level. It consists of four nesting modules: the inference methods, the knowledge model, the combined reasoning model, and the problem-solving model. The inner ellipses prepare the necessary basis for the outer ones.

The innermost ellipsis represents the inference methods. It is triggered by any new observations and derives information by applying a set of inference rules to a knowledge model. The knowledge model includes the general domain knowledge and the previously solved cases.

The inference methods and the knowledge model prepare the basis for the combined reasoning model. This model consists of the three reasoning methods that are integrated to generate knowledge and information produced by the inference methods and the knowledge model.

The problem-solving model is the outermost ellipsis. It represents the problem-solving procedures by a set of actions and rules. The rules are some predetermined regulations from the domain expert, which assist the problem-solving process.

### 3.2 Functional architecture

Figure 3 illustrates the system's functional architecture. The problem-solving functions module employs the inference and reasoning methods and executes the procedures of solving the problems. It is a combination of Bayesian reasoning (BR), case-based reasoning (CBR), and semantic network reasoning (SNR) modules. Two solid bidirectional arrows represent the information flow between the reasoning modules.

The problem-solving functions module communicates with the knowledge base module by bidirectional information passing. The knowledge base module has two sub modules. The underlying model of general domain knowledge is referred to as conceptual knowledge. It represents the general definitions of the domain concepts and relations. The specific experiences (solved cases) are integrated into the conceptual knowledge in the form of the domain concepts.

The information flows between: 1. CBR module and the knowledge base, 2. SNR module and the knowledge base, and 3. BR module and the knowledge base are represented by bidirectional dotted arrows, dashed arrows, and dashed dotted arrows, respectively.

In BNCreek, data are entered into this system as a new case description. There are three main subprocesses to generate a proper solution for an input case: 1. Bayesian subprocess, 2. matching and adaptation subprocesses, and 3. generating explanation subprocess.

The Bayesian subprocess dynamically updates the system's beliefs based on any new information. The process starts from the BR module, with the main function to activate the probabilistic beliefs calculator. The process follows by knowledge passing between the conceptual model and the probabilistic beliefs calculator in different steps of the problem-solving function. In each step, the updated beliefs pass to the explanation generator, matching and adaptation modules.

The matching and adaptation subprocesses carry the main goal of the system. They aim to generate a proper solution for the new input case by adapting the solutions of the best matched cases. It is triggered by the CBR module and followed by passing the relevant information between the

conceptual model and the two modules of matching and adaptation in different steps of problem-solving function. In each step, the information from the probabilistic beliefs calculator and the explanation generator is employed.

The generating explanation subprocess aims to explain the partial similarity or relevance between concepts of the knowledge base. This subprocess follows one of the two scenarios, depending on whether the CBR task is retrieve or reuse: 1. It is initiated by the SNR module and passes the relevant information to the conceptual model, explanation generator, and similarity assessment. 2. It is initiated by the SNR module and passes the relevant information to the conceptual model, explanation generator, and adaptation. This subprocess works in different steps of problem solving function; in each step, the updated information from the probabilistic beliefs calculator is employed.

An expressive and flexible knowledge representation language is required to achieve a common consensus between the expert and the system about the knowledge interpretation.

The representation language based on combining the semantic network and Bayesian network was assessed to give the highest degree of flexibility, expressiveness, and user transparency while avoiding the limitations imposed by logic and rule-based representation formalisms. The semantic network and Bayesian network are well-defined extendible representation languages in which the properties of a new concept can be added into the network without imposing a heavy change to the rest. Also, for big domains, the networks could split up and distribute between the individual systems [21]. This makes the network representation language a good candidate for a knowledge-based designer to model the knowledge.

A run-through toy example from the cooking domain is presented through this and the following chapters to clarify the system processes and methods. The problem is fault diagnosis in making a dinner course, and the knowledge model focuses on the ingredients and failures. Figure 4 is part of the cooking domain knowledge model. It consists of 46 cooking domain concepts and more than 43 relations between them. The knowledge model is divided into the semantic network layer, Bayesian network layer, and partial descriptions of three cases that are connected to the networks (dashed edges). The relation types and their strengths are written along the edges. HS and HSt stand for Has subclass and Has status, respectively. In this example, for simplicity, the reverse relationships are not displayed and the strengths of a relationship and its reverse are considered to be 0.9. The causal relationships are exceptions. The causal relations present the failures of using an inappropriate amount of ingredients. To keep the figure simple, most of the concepts' names are abbreviated. The ShCE, BCE, and FCE stand for shrimp cooked enough, beef cooked enough, and fish cooked enough, respectively. The UC, OC, L, M, and E stand for undercooked, overcooked, little, much, and enough, respectively, and followed by the first letter of its parent. For example, UCSh stands for undercooked shrimp and MP stand for much pepper.

## 4 Knowledge representations

### 4.1 Representing knowledge with the semantic network

The semantic network as a declarative graphical representation language is introduced as a specialized approach for representing deep and shallow domain knowledge in a structured way. The general definition of a concept is part of the semantic knowledge representation. It enables an explicit definition for each type of concept in the domain. Also, it allows for the definition of any type of relation and provides a general structural inference mechanism.

The semantic network knowledge representation is partly developed on the basis of the CreekL representation language with some modifications and extension. CreekL is a frame-based language with some implemented basic inference methods, such as frame-matching, default inheritance, spreading activation, and constraint satisfaction [24,25].

The network is represented as an edge labeled directed graph such that nodes are domain concepts, and edges demonstrate the relationships between concepts. The nodes and edges are associated with labels that name the relation type and the strength that reflects the frequency of holding the relationship.

Formally, given a finite set of domain concepts (C), relation types (T), and the relationships (R), the BNCreek semantic network (BSN) is defined by a triple of BSN = (C, T, R). Definition 1 demonstrates the formalism of the BSN. At this level of formalization, all the domain objects, including the relation types, are considered as concepts. The set of concepts is denoted by the upper-case letter C, and its instances are denoted by lower-case letters, c. The relationships are identified by a quadruple of (C, T, S, C) consisting of two concepts and a relation type that connects them and S representing the relation strength. The instances of the relationships are denoted by the lower-case letter r and its identification is $(c_s, t, s, c_e)$, such that $c_s$ is the start concept, $t$ is the relation type, $c_e$ is the end concept and $s$ is the strength of the relationship.

**Definition 1** (*BNCreek semantic network*) The BNCreek semantic network (BSN) is a triple of BSN = (C, T, R), such that:

- C is a finite set of domain concepts.
- $T \subseteq C$ is a set of relation types.

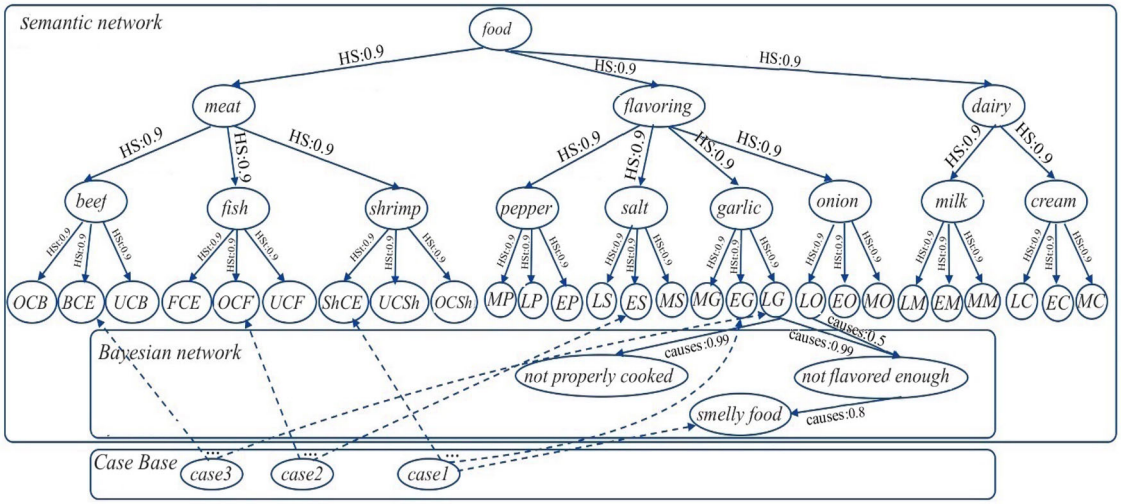


**Fig. 4** The figure illustrates parts of the cooking area knowledge model. The relation types and strengths are written along the edges. Reverse relationships are not included. HS and HSt stand for has subclass and has status, respectively. *BCE, FCE, and ShCE* stand for *beef cooked enough, fish cooked enough, and shrimp cooked enough,* respectively. The OC, UC, L, M, and E stand for overcooked, undercooked, little, much, and enough, respectively, and are followed by the first letter of its parent. For example, LS stands for *little salt* [https://doi.org/10.1007/s13748-020-00223-1]

– $R \subseteq (C, T, S, C)$ is the set of relationships in the network where S demonstrates the relation strength by a number between [0...1].

A visual example of a semantic network as a part of the cooking domain is presented in Fig. 4. The formal syntax description of its leftmost branch is as follows:

C = {$food$, $meat$, $beef$, $OCB$, $BCE$, $UCB$, Has subclass, Has status}
T = {Has subclass, Has status}
R = {$r_1$, $r_2$,$r_3$, $r_4$, $r_5$}, where

$r_1$ ={$food$, Has subclass: 0.9, $meat$}
$r_2$ = {$meat$, Has subclass: 0.9, $beef$}
$r_3$ = {$beef$, Has status: 0.9, $OCB$}
$r_4$ = {$beef$, Has status: 0.9, $BCE$}
$r_5$ = {$beef$, Has status: 0.9, $UCB$}

For each relationship, there is an inverse that demonstrates a kind of relationship in the opposite direction between the concepts. The inverse relationship of $(c_s, t, c_e, x)$ will be $(c_e, t', c_s, y)$, such that the $t'$ and $y$ are relation type and strength, respectively, which are predetermined by the expert during constructing the domain semantic network.

## 4.2 Representing knowledge with a Bayesian network

A Bayesian network as a probability network in the form of a directed acyclic graph is able to represent a large uncertain domain probability distribution compactly. In a Bayesian representation, the variables of interest are represented as the nodes, and the direct causal relationships among them are represented by the edges [10–12]. The semantic of a Bayesian network represents a unique probability distribution over its variables, and the conditional probabilities of the domain variables quantify the strength of the dependencies between the variables and their parents in the Bayesian network [26,27].

Bayesian network knowledge representation is a novel extension to the Creek-type knowledge-intensive systems. The representation language and formalisms for representing causal knowledge by a Bayesian network and taking advantage of its inference methods were inspired by Pearl [28] and Darwiche [29].

Formally, a Bayesian network is a directed acyclic graph over variables BC. BC stands for Bayesian concepts and is a finite set of the domain concepts that are the Bayesian network variables. The roots of the Bayesian networks represent the basic faults in the domain, and the target nodes could be the mid-level faults or the probable symptoms.

Definition 2 defines the Bayesian Variables (BC) in a way that each variable $BC_i$ has a finite number of values $Bc_i$ and $\omega = \{Bc_1, Bc_2, \ldots, Bc_n\}$ is a possible instantiation of BC for the Bayesian network. Definition 3 defines the BNCreek Bayesian network(BBN) over variables BC as a quadruple, i.e., the Bayesian variables, the set of Bayesian relation types (BT) consisting of causes and caused by, Bayesian relationships (BR) and the conditional probabilities. The BR is identified by a quadruple of (BC, BT, BC, [0...1]) and con-

sists of two Bayesian concepts, a Bayesian relation type that connects them, and a degree for the Bayesian relation that in this context is called Bayesian relation strength. The conditional probabilities from each variable BC to its parent U are sets of conditional probability tables (CPTs), which is denoted by $\Theta_{X|U}$.

Finally, Definition 4 defines the probability distribution over variables $BC$, $Pr(BC) = \prod_{\Theta_{Bc_i|u:Bc_i u\sim u}} \Theta_{Bc|u}$.

**Definition 2** (*Bayesian variables*) $BC \subseteq C$ is the set of Bayesian variables $\{BC_1, BC_2, \ldots, BC_n\}$, such that:

- Each variable $BC_i$ has a finite number of values $Bc_i$.
- $\omega = \{Bc_1, Bc_2, \ldots, Bc_n\}$ is a possible instantiation of BC.
- $\Omega$ is the set of all possible $\omega$.

**Definition 3** (*BNCreek probabilistic Bayesian network*) The BNCreek Bayesian network(BBN) over variables BC is a quadruple of $BBN = (BC, BT, BR, \Theta)$, such that:

- $BC = \{BC_1, BC_2, \ldots, BC_n\}$ is a finite set of the Bayesian variables.
- $BT \subseteq BC, BT = \{causes, causedby\}$ is a set of Bayesian relation types.
- $BR = (BC, BT, BC, [0...1])$ is the set of the Bayesian relationships in the network.
- $\Theta$ is a set of conditional probability tables (CPTs), from each variable $BC_i$ to its parent $BC_j$, denoted by $\Theta_{BC_i|BC_j}$.

**Definition 4** (*Probability distribution*) $Pr(BC) = \prod_{\Theta_{Bc|u:Bcu\sim u}} \Theta_{Bc|u}$ is a probability distribution over variables $BC$.

The causal relations in Fig. 4 show part of a simplified Bayesian network from a knowledge model. The formal syntax description of the sample domain is as follows:

BC = {$LG$, $LO$, $not\ properly\ cooked$, $not\ flavored\ enough$, $smelly\ food$, causes}
BT = {causes}
BR = {$BR_1$, $BR_2$,$BR_3$, $BR_4$}, where

$BR_1$ = {$LG$, causes: 0.99, $not\ properly\ cooked$}
$BR_2$ = {$LG$, causes: 0.99, $not\ flavored\ enough$}
$BR_3$ = {$LO$, causes: 0.5, $not\ flavored\ enough$}
$BR_4$ = {$not\ flavored\ enough$, causes: 0.8, $smelly\ food$}

For each Causal relationship $(Bc_s, causes, Bc_e, x)$, there is an inverse relationship $(Bc_e, causedby, Bc_s, y)$ that is not shown in Fig. 4. The $x$ and $y$ are the relation strengths that are predetermined by the expert during constructing the domain Bayesian network and are not necessarily identical numbers. The Bayesian network layer is formed by the causal relations and their relevant concepts. The Bayesian network layer is considered as an individual component, and at the same time, it is part of the semantic network component, as well. This overlap enables interaction between the two networks.

### 4.3 Representing knowledge with cases

A case illustrates a specific situation of the domain. It is described by a list of features (F), Case = ($F_1$, $F_2$,…, $F_n$). Features are the triple of relationships (R), values (V), and relevance factors (RF) as a number that indicates the importance of a feature for a stored case, feature = (R, V, RF).

There are three specific relation types for the cases: has status, has symptom, and has failure. These relationships link the case concept into the values represented by concepts in the knowledge model and do not have any reverse, and then a case can be viewed as a subgraph of the domain model.

There is no limit on the number of case features. Any number of observations, measurement results, or related concepts and values could be reported as a feature. The features of a case are in two types:

1. The observed features that are entered into the case by the user (the raw case features).
2. The inferred features that are entered into the case by the system.

An observed or an inferred feature could be part of the symptom features (symptoms), the status features (status), or the failure features (failures).

The dashed relationships in Fig. 4 illustrate parts of $case_1$, $case_2$, and $case_3$. The full descriptions of the three cases are illustrated in Fig. 5.

## 5 Problem solving

The problem-solving process in BNCreek is described from the CBR point of view in a two-step model of retrieve–reuse. The details of the process are presented based on the three main subprocesses, which are explained in the functional architecture (Fig. 3). Algorithm 1 presents the process in detail. The input of the algorithm is a new failed case, and the output is the list of most probable failures as a solution for that case. The run-through example from the cooking domain elaborates the process further.

The process starts with the retrieve part of the matching and adaptation subprocesses, which attempts to gather and infer sufficient information about the problem. The reuse part of the matching and adaptation subprocesses uses the retrieve results to achieve the main goal of the problem-solving pro-

*Name: Roasted Shrimp (case1) (raw)*

| Relation | Value | RF |
|---|---|---|
| has normal st. | shrimp cooked enough | 0.7 |
| has normal st. | enough flour | 0.5 |
| has normal st. | enough garlic | 0.5 |
| has normal st. | enough salt | 0.5 |
| has normal st. | enough pepper | 0.5 |
| has normal st. | enough butter | 0.5 |
| has symptom | smelly food | 0.9 |
| has case st. | Unsolved case | |
| has solution | | |

(a)

*Name: Roasted Shrimp (case1) (pre-processed)*

| Relation | Value | RF |
|---|---|---|
| has normal st. | shrimp cooked enough | 0.7 |
| has normal st. | enough flour | 0.5 |
| has inferred failure | little garlic | 0.68 |
| has normal st. | enough salt | 0.5 |
| has normal st. | enough pepper | 0.5 |
| has normal st. | enough butter | 0.5 |
| has inferred failure | not flavored enough | 0.55 |
| has symptom | smelly food | 0.9 |
| has case st. | Unsolved case | |
| has solution | | |

(b)

*Name: Roasted Shrimp (case1) (solved)*

| Relation | Value | RF |
|---|---|---|
| has normal st. | shrimp cooked enough | 0.7 |
| has normal st. | enough flour | 0.5 |
| has failure | little garlic | 0.68 |
| has normal st. | enough salt | 0.5 |
| has normal st. | enough pepper | 0.5 |
| has normal st. | enough butter | 0.5 |
| has failure | not flavored enough | 0.55 |
| has symptom | smelly food | 0.9 |
| has case st. | solved case | |
| has solution | Little garlic, not flavored enough | |

(c)

*Name: Baked fish (case2) (solved)*

| Relation | Value | RF |
|---|---|---|
| has failure | overcooked fish | 0.55 |
| has normal st. | enough lemon | 0.5 |
| has normal st. | enough garlic | 0.5 |
| has normal st. | enough salt | 0.5 |
| has normal st. | enough oil | 0.5 |
| has symptom | overcooked | 0.9 |
| has case st. | solved case | |
| has solution | overcooked fish | |

(d)

*Name: Beef steak (case3) (solved)*

| Relation | Value | RF |
|---|---|---|
| has normal st. | beef cooked enough | 0.5 |
| has inferred failure | little onion | 0.6 |
| has inferred failure | little garlic | 0.68 |
| has normal st. | enough salt | 0.5 |
| has normal st. | enough pepper | 0.5 |
| has normal st. | enough flour | 0.5 |
| has normal st. | enough butter | 0.5 |
| has inferred failure | not flavored enough | 0.55 |
| has symptom | smelly food | 0.9 |
| has case st. | solved case | |
| has solution | little garlic, little onion | |

(e)

**Fig. 5** The three changing steps for the *case*1 consist of the raw new *case*1 (**a**), the pre-processed *case*1 (**b**), the solved *case*1 (**c**) and the solved *case*2 and *case*3 in (**d**, **e**), respectively

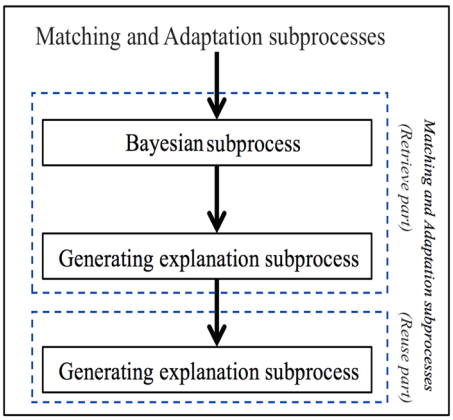


**Fig. 6** The dependencies and information flow between the three subprocesses from the functional architecture

cess. Figure 6 shows the dependencies and information flow between the three subprocesses. In an example of the cooking domain, the goal of the retrieve step could be diagnosing the causes of a *smelly food* symptom, and the goal of the reuse step to present a solution for the case utilizing the diagnosed causes.

The first part of the retrieve process focuses on the problem (case) description in order to establish the initial understanding of it. The initial understanding of a case is performed in two steps: First, the Bayesian subprocess is triggered and information from the case description (raw case) transfers into the knowledge model and updates its beliefs. Second, the processed information inferred from the knowledge model updates the case description and the Bayesian subprocess is concluded.

The second part of the retrieve process focuses on a similarity assessment aimed to retrieve the most similar cases. This step triggers the Generating explanation subprocess, which explains the partial similarity between the coupled features, retrieves the most similar cases, and concludes the Generating explanation subprocess.

Algorithm 1 (lines 1 to 11) describes the retrieve process as the first part of the Matching and Adaptation subprocesses. The system considers the symptoms of the raw input case and computes the Bayesian posterior beliefs given the symptoms. The raw input case description is modified by adding the extracted causes as the inferred features and adjusting the relevance factors of the existing features, if necessary. Finally, the causal strengths in the semantic network module are adjusted. As the next step, the similarity between the input case and the cases from the case base are assessed and a list of the matched cases are presented as the output of the retrieve phase.

As an example, consider part (a) from Fig. 5 as a raw input case. The case describes a failed *Roasted Shrimp* dish. It consists of the ingredients and a symptom that indicates the dish status. The matching and adaptation subprocesses consider the *smelly food* symptom as an input. It triggers the Bayesian subprocess, propagates the symptom, and updates the knowledge model beliefs. Then, the matching and adaptation subprocesses in an interaction with the conceptual model extract the *not flavored enough* concept as the cause of the *smelly food* and adds it to the case description. The other features of the case are modified based on the updated beliefs, and the result is presented in part (b) of Fig. 5 under the name of the pre-processed case. In order to find the most similar case, the matching and adaptation subprocesses interacts with the solved cases module and examines all the cases. Parts (d) and (e) of Fig. 5 demonstrate the solved *case*2 and solved *case*3, which are the best matched cases to *case*1. The generating explanation subprocess is triggered and explains partial similarities between the features of the pre-processed *case*1 and the solved *case*2 and solved *case*3. Consider $LG$–>$garlic$–>$EG$ and $LG$–>$garlic$–>$flavoring$–>$garlic$–>$EG$, which are two examples of the explanation paths between ($EG$, $LG$) coupled features. The problem-solving process continues and the total similarities between (*case*1, *case*2) and (*case*1,

*case*3) becomes 56% and 71%, respectively. *Case*3 is the most similar case to *case*1 that is retrieved by the first part of the matching and adaptation subprocesses.

The first part of the reuse process generates or gathers the potential solutions. This step employs the most similar cases' solutions, triggers the second round of the Generating explanation subprocess, and infers more potential solutions from the Knowledge model.

The second part of the reuse process refines the potential list and identifies the final solution. This step utilizes some predefined rules from the expert regarding the domain characteristics and some systematic thresholds and generates the final solution.

Algorithm 1 (line 12 to 18) describes the reuse process as the second part of the matching and adaptation subprocesses. The system considers the solutions of the first n best similar cases as the potential solutions and modifies the relevance factors of the cases' features by the corresponding causal strengths from the knowledge model. Then, it generates the causal explanation path with maximum length m from each feature of the new case to its nearest failure, infers the failures, and adds them into the potential solution list. Finally, the potential solution list is modified by applying the removing rules from an expert and the less probable failures are removed from the list. The finalized failure list is presented as the reuse phase output.

The matching and adaptation subprocesses in the reuse part follow by considering {*little garlic*, *little onion*} which is the solution of *case*3 as a potential solution for *case*1. The generating explanation subprocess is triggered, and the *not flavored enough* concept is inferred from the knowledge model and is added to the list. Then, the less probable failures are removed and the modified solution is presented as {*little garlic*, *little onion*, *not flavored enough*}. Then, the expert predefined rules removes the little onion from the list, as it is not even one of the ingredients. The finalized list is presented as the solution of *case*1: {*little garlic*, *not flavored enough*}.

**Algorithm 1:** Problem solving in BNCreek

**Input** : An input raw case.
**Output**: A solution for the input case consisting of a list of most probable failures.

1 Consider the symptoms of the input raw case.
2 Compute the Bayesian posterior beliefs given the symptoms.
3 Extract the symptoms' causes.
4 Modify the raw input case description.
- Add the extracted causes as the inferred features.
- Adjust the existing features if necessary.
5 Employing the posterior distribution, adjust the causal strengths in the semantic network module.
6 **while** *not all the case-base is tested* **do**
7 Consider one case from the case-base.
8 Compute the explanation strength between any pair of input and retrieved case features.
9 Compute the similarity between input and retrieved case.
10 **end**
11 List the matched cases.
12 Consider the solutions of the first n best similar cases as the potential solutions.
13 Modify the relevance factors of the cases' features by the corresponding causal strengths from the Knowledge model.
14 Generate a causal explanation with maximum length m from each feature of the new case to its nearest failure.
15 Add the inferred failures into the potential solution list.
16 Modify the potential solution list by applying the removing rules from an expert.
17 Remove the less probable failures from the list.
18 Present the modified list as the input case solution.

## 6 Application domain

The real application domain focused in this study is oil well drilling. The oil and gas industry is a complicated area that faces challenges and uncertainties to keep its efficiency as high as possible. The natural complication of this area is accompanied by the geological factors, temperature, and pressure. This makes understanding the process and achieving the objectives very complex for drilling engineers and causes drilling operations to encounter failures frequently.

Oil wells are holes with an approximate diameter of 12 cm to 1 meter drilled into the earth aimed to extract petroleum oil hydrocarbons and natural gas and bring them to the surface. Usually, the main goal in this process is to optimize production from the well and to manage the potential failures.

These facts make the oil well drilling process a good case study for us to test our system capabilities in diagnosing the potential failures of new cases in this area.

The feasibility of combining CBR with some form of general domain knowledge for oil well drilling was demonstrated by the DrillEdge system [30]. The drilling domain model we are using describes the drilling process concepts, properties, and relationships, such as hierarchical structures, functional relations, and causalities. It constitutes the knowledge fundament of the domain and gives detailed knowledge and understanding of the system that helps an efficient fault diagnosis [31].

The drilling operation is a widespread process containing approximately 300 properties described by observable or measurable descriptors. Some of the concepts describe simple internal properties (e.g., cuttings on shaker); the others represent non-normal situations, i.e., symptoms (e.g., cuttings initial concentration high) and their causes/failures (e.g., accumulated cuttings). There are about 20 or so significant single causes for about 100 non-normal drilling operation situations and many relevant combinations. Diagnosing failures is a complicated challenge because the values of drilling properties are interdependent. Besides, one symptom may have more than one cause that led to the diagnosis of more than one failure. This situation introduces a level of complexity that is difficult to handle with traditional methods. Figure 7 illustrates a pre-processed oil well drilling case.

| *A pre-processed drilling case* | | |
|---|---|---|
| **Relation** | **Value** | **RF** |
| has symptom | activity of tripping in | 0.7 |
| has internal property | cavings on shaker | 0.7 |
| has symptom | fm fault expected | 0.7 |
| has symptom | fm hard | 0.7 |
| has inferred symptom | mse high | 0.5 |
| has symptom | fm laminated | 0.7 |
| has symptom | fm soft | 0.7 |
| has symptom | fm unstable expected | 0.7 |
| has inferred symptom | cavings blocky | 0.6 |
| has symptom | hkl high | 0.7 |
| has inferred symptom | build/drop section inside openhole | 0.62 |
| has inferred symptom | activity of tripping out | 0.67 |
| has inferred symptom | dls high | 0.54 |
| has inferred symptom | hoisting speed high | 0.52 |
| has symptom | losses seepage | 0.7 |
| has inferred failure | stuck pipe mechanically | 0.9 |
| has inferred symptom | ecd - frac d low | 0.66 |
| has inferred failure | overpull | 0.7 |
| has symptom | time of stillstand long | 0.7 |
| has symptom | time of tripping long | 0.7 |
| has symptom | torque high | 0.7 |
| has inferred symptom | wob spikes | 0.89 |
| has symptom | well inclination high | 0.7 |
| has symptom | well openhole long | 0.7 |
| has symptom | well p high | 0.7 |
| has inferred symptom | torque erratic | 0.8 |
| has case st. | unsolved case | |
| has solution | | |

**Fig. 7** A pre-processed oil well drilling case

## 7 System evaluation

The main goal of BNCreek is diagnosing the failures that result in a new case and presenting them as its solution during the problem-solving process. The process is based on the first two phases of the CBR cycle [14]: 1. retrieve phase and 2. reuse phase.

In the retrieve phase, the system aims to retrieve the most relevant cases from the case base. The results of the retrieve phase are evaluated in two perspectives: 1. The NDCG and precision–recall metrics are utilized to measure the system's ability to retrieve similar cases in the correct rank. 2. The root-square error (RSE) and weighted error (WE) are applied to measure the accuracy of the similarity degrees. The results of the Retrieve phase evaluation are presented in [21] [https://link.springer.com/article/10.1007/s13748-020-00223-1].

In the reuse phase, the system aims to generate a solution for the input case employing the previous phase results, which is the final goal of the problem solving process in our system. The current publication has focused on evaluating the BNCreek results of the Reuse phase.

### 7.1 Evaluation metrics

In the conducted experiment, a set of failures, which are the causes of the input case's symptoms, form a solution. A completely correct solution, which is the ideal situation, is when the system presents the same failure list with the expert's list for the input case. As most of the solutions partially match the expert solutions, we investigate the correctness of a solution by considering a kind of confusion matrix. In a way, each member of the solution could take four situations: 1. It is also a member of the expert solution (TP), 2. it is not a member of the expert solution (FP), 3. failure is neither a member of the system solution nor the expert solution (TN), and 4. failure is not a member of the system solution, but it is in the expert solution (FN).

For more clarification, suppose $\{f1, f2, f3, f4, f5\}$ is the set of all possible failures in our domain. Now assume that the BNCreek solution for case 1 is $\{f1, f2, f3\}$ and the expert solution for case 1 is $\{f1, f2, f4\}$. Therefore, the presented solution for case 1 is partially correct. The confusion matrix terms for case 1 are TP = 2, TN = 1, FP = 1, FN = 1. The confusion matrix of the 45 solutions for the cases being studied is calculated and presented in Table 1.

To evaluate the reuse phase's accuracy in generating the closest solution to the expert solutions, the $F_1 - score$, sensitivity, and specificity are applied. The metrics calculate the accuracy and stability of the system. The same experiment is conducted on a related system, TrollCreek, and the results are compared.

Among the evaluation metrics that have been derived by the confusion matrix, sensitivity (recall), specificity, and precision metrics are utilized to evaluate the validity of BNCreek. Sensitivity and specificity metrics both address how often the system presents reliable answers.

Sensitivity shows the ability of the system to detect actual failures. In other words, sensitivity shows how often the system measures what it claims to measure. In Eq. 1, TPR stands for the true positive rate that shows the system's Sensitivity. It investigates the degree of the actual failures that are correctly diagnosed as failures (TP) or mistakenly considered as a not failure (FN).

$$TPR = \frac{TP}{FN + TP} \tag{1}$$

Specificity shows the ability of the system to detect the failures that are not part of the current case's solution. Equation 2 shows the system Specificity. It investigates the number of not relevant failures that are correctly diagnosed as not relevant failures (FP) or mistakenly considered as a relevant failure (FP).

$$Specificity = \frac{TN}{TN + FP} \tag{2}$$

Precision in a classification model is the number of items that are correctly labeled as the positive class divided by the total number of items labeled as the positive class. Equation 4 illustrates Precision. The TP stands for the number of true positives, and the FP stands for the number of false positives. A Precision score of 1.0 illustrates all the items labeled correctly as the positive class.

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

The $F_1 - score$ measures the accuracy of a system utilizing precision and recall. Precision could be considered as the probability that a retrieved failure is a relevant one. Therefore, the recall is the probability that a relevant failure is retrieved. Taking the (weighted) harmonic average of the precision and recall specifies the $F_1 - score$. Thus, the $F_1 - score$ measures a balance between precision and recall by taking both false positives and false negatives into account. The highest possible value of $F_1$ is 1, indicating perfect precision and recall, and the lowest possible value is 0, if either the precision or the recall is zero [32–34].

$$F_1 = 2.\frac{Precision * Recall}{Precision + Recall} \tag{4}$$

## 7.2 Experimental setup

The oil well drilling knowledge model utilized in this experiment is a detailed ontology consisting of 350 drilling domain concepts and more than 1000 relationships between them. Forty-five artificial drilling failure cases are generated by the expert and are utilized as the queries (input cases) for testing the system.

We conducted an experiment from the oil well drilling application domain utilizing LOOCV (leave-one-out cross-validation) to test the performance of BNCreek. In each turn of the experiment, one case was picked out as the test case. The other cases were regarded as training cases and their solutions were utilized to generate a solution for the testing case. The generated solutions were presented as the BNCreek results and evaluated against the expert predictions.

The experiment follows two goals. The main goal is to evaluate the BNCreek ability to solve fault diagnosis problems. The other goal is to find out the best system tuning to achieve the best performance, based on the current knowledge model.

In the current version of BNCreek, there are two parameters to set; both depend on the targeted application domain.

The first parameter addresses the boundaries for the explanations' strengths. The second parameter relates to the number of retrieved cases that would be involved in the reuse process. In order to investigate different possibilities, three situations called Types 1, 2, and 3 are investigated. Type 1 utilizes the first most similar case in the reuse process. The Types 2 and 3 utilize the two and three most similar cases for the reuse process, respectively. The experiment results are presented in the form of bar charts.

**Table 1** The true positive (TP), false positive (FP), false negative (FN), and true negative (TN) values resulted from the conducted experiment on BNCreek system in the Types 1, 2, and 3, respectively

| Type 1 | Actual results: P | Actual results: N |
|---|---|---|
| Predicted results: P | TP: 233 | FP: 18 |
| Predicted results: N | FN: 62 | TN: 498 |
| Type 2 | Actual results: P | Actual results: N |
| Predicted results: P | TP: 256 | FP: 38 |
| Predicted results: N | FN: 39 | TN: 477 |
| Type 3 | Actual results: P | Actual results: N |
| Predicted results: P | TP: 267 | FP: 63 |
| Predicted results: N | FN: 28 | TN: 452 |

## 7.3 BNCreek results

The results of a fault diagnosis test can be evaluated by measuring its attributes utilizing confusion matrixes' terms. The confusion matrixes for Types 1, 2, and 3 are illustrated in Table 1. The TP term for Types 1, 2, and 3 is {233, 256, 267}, respectively, which show the number of true positive failures. The TN term for Types 1, 2, and 3 is {498, 477, 452}, which show the number of true negative failures. The FN term for Types 1, 2, and 3 is {62, 39, 28}, which show the number of false negative failures. Finally, the FP term for Types 1, 2, and 3 is {18, 38, 63}, which show the number of false positive failures. The true positive and true negative terms show the consistency of the fault diagnostic test.

Along with the description of the confusion matrix terms, sensitivity and specificity are utilized to quantify how good the test is at detecting a positive failure and estimating how likely not failures can be correctly ruled out, respectively. The upper side of Fig. 8 illustrates the sensitivity and specificity of BNCreek for Types 1, 2, and 3 in the form of bar charts. The sensitivity value for Types 1, 2, and 3 are {0.789, 0.867, 0.905}, respectively. The specificity value for Types 1, 2, and 3 is {0.965, 0.926, 0.877}, respectively. A fault diagnosis test with both high sensitivity and specificity values is a more reliable test.

We also tested the system's accuracy by utilizing the $F_1 - score$. The $F_1 - score$ measures the balance between precision and recall. The precision and recall measure the trade-off between the number of relevant identified failures against the sensitivity. The precision and recall degrees for Types 1, 2, and 3 are {0.92, 0.87, 0.80 } and {0.78, 0.86, 0.90}, respectively. Outcome of the precision and recall, the $F_1 - score$ for Types 1, 2, and 3 is {0.84, 0.86, 0.84}. In the upper side of Fig. 8, the third bars in each category of Types 1, 2, and 3 demonstrate the $F_1 - scores$.

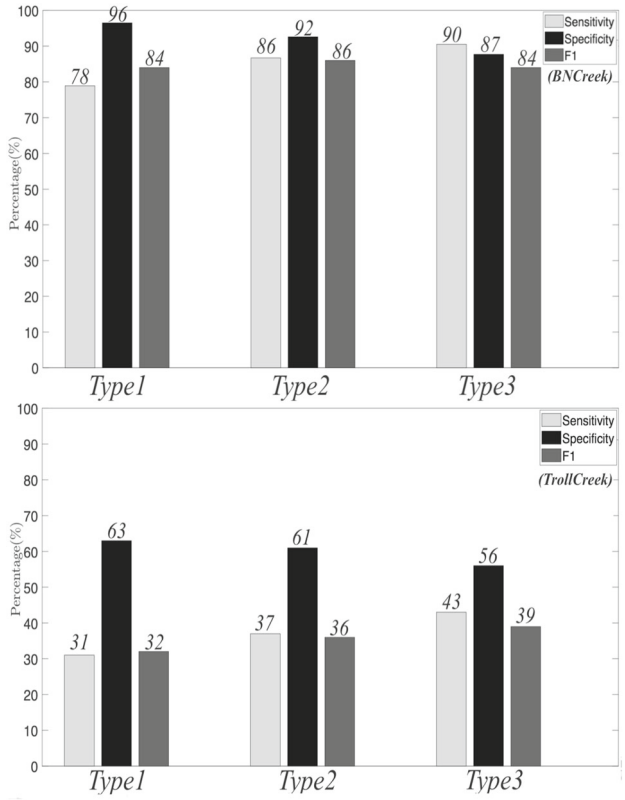


**Fig. 8** The bar charts in the upper side demonstrate the sensitivity, specificity and the $F_1 - score$ in percent resulted from the conducted experiment on the BNCreek system for Types 1, 2, and 3. The bar charts in the lower side demonstrate the same metrics for the TrollCreek system

### 7.4 TrollCreek results

The experiment is repeated with the TrollCreek system, and the confusion matrixes' terms for Types 1, 2, and 3 are illustrated in Table 2. The TP term for Types 1, 2, and 3 is {93, 111, 129}, respectively, which show the number of true positive failures. The TN term for Types 1, 2, and 3 is {328, 316, 295}, which show the number of true negative failures. The FN term for Types 1, 2, and 3 is {202, 184, 166}, which show the number of false negative failures. Finally, the FP term for Types 1, 2, and 3 is {187, 200, 223}, which show the number of false positive failures.

The sensitivity and specificity of the experiment on the TrollCreek system for Types 1, 2, and 3 are illustrated in the lower side of Fig. 8. The sensitivity value for Types 1, 2, and 3 is {0.789, 0.867, 0.905}, respectively. The specificity value for Types 1, 2, and 3 is {0.965, 0.926, 0.877}, respectively.

The system's accuracy is tested by utilizing the $F_1 - score$. The $F_1 - score$ for Types 1, 2, and 3 is {0.32, 0.36, 0.39}. The third bars in each category of Types 1, 2, and 3 demonstrate the $F_1 - scores$.

**Table 2** The true positive (TP), false positive (FP), false negative (FN), and true negative (TN) values resulted from the conducted experiment on TrollCreek system in the Types 1, 2, and 3, respectively

| Type 1 | Actual results: P | Actual results: N |
|---|---|---|
| Predicted results: P | TP: 93 | FP: 187 |
| Predicted results: N | FN: 202 | TN: 328 |
| Type 2 | Actual results: P | Actual results: N |
| Predicted results: P | TP: 111 | FP: 200 |
| Predicted results: N | FN: 184 | TN: 316 |
| Type 3 | Actual results: P | Actual results: N |
| Predicted results: P | TP: 129 | FP: 223 |
| Predicted results: N | FN: 166 | TN: 295 |

## 8 Discussion

In BNCreek, we have utilized a combination of the semantic network reasoning, the Bayesian network reasoning, and the case-based reasoning for problem solving purposes. This integration enables the system to cover the different aspects of the domain complexities in problem solving, in such a way that the more detailed and well-understood aspects of the domain are analyzed by the Bayesian network, and the incommensurable aspects of the problems are solved by case-based reasoning method. The system is tested by a classification model evaluation metrics.

Sensitivity provides information regarding the related failures which are truly introduced as the failures of the current case, while specificity focuses on the failures that are not related to the solution of the study case and are truly introduced as the not related failures for it. Therefore, it is desirable to have a system that represents the results with high sensitivity and specificity, which shows true detection of related failures and not related failures at the same time. The upper side of Fig. 8 illustrates the sensitivity and specificity of the conducted fault diagnosis experiment. The system provided acceptable values for sensitivity and specificity in the three investigated conditions, i.e., Types 1, 2, and 3, which shows its reliability.

Precision shows the system's ability to label an unrelated failure as it truly is. Recall shows the system's ability to find all relevant failures. An ideal system with a high precision and a high recall returns many correctly detected failures as the case solution.

The $F_1 - score$ shows the overall performance of the model as the average rate of precision and recall. Conversely, having a low precision or recall score will dominate the $F_1 - score$ and push it to the lower score. The high value for the $F_1 - score$ in the three investigated conditions, i.e., Types 1,

2, and 3, shows the high accuracy of the system in generating proper solutions for the new cases.

The lower side of Fig. 8 illustrates the sensitivity and specificity of the conducted fault diagnosis experiment on the TrollCreek system. The reuse phase, as part of the CBR cycle, is not implemented in the available versions of the TrollCreek system. The reuse process in these versions is done in a manual process based on the user's decisions. Therefore, to compare the final result of TrollCreek and BNCreek, we focused on investigating the effect of Bayesian analysis and considered the same reuse process for TrollCreek while we have removed all Bayesian related parts. The sensitivity, specificity, and $F_1 - score$ in the three investigated conditions, i.e., Types 1, 2, and 3, are not so high in comparison with the same values in BNCreek.

The low performance of TrollCreek could stem from three reasons. 1. The problem solving in both of the systems is based on the retrieve phase of the CBR cycle. The low performance in retrieving the correct cases leads to not generating the proper solutions by the reuse phase. 2. The main structure of the reuse process in TrollCreek is considered the same in BNCreek. In one aspect, this similarity could be considered a positive point, which makes the two systems more comparable. In another aspect, the specific techniques that could be used based on the TrollCreek specifications and would improve the results are missing. 3. The higher performance of BNCreek indicates the positive effect of Bayesian analysis in BNCreek in comparison with TrollCreek, which does not use the Bayesian component.

For the BNCreek results in the upper side of Fig. 8, although the $F_1 - score$ of Type 1 and Type 3 is the same, Type 1 has higher precision but lower recall, which means the more relevant failures are truly retrieved. Both Type 1 and Type 3 would be promising. Type 2 has the highest $F_1 - score$ despite its precision being lower than Type 1, and its recall is lower than Type 3, but in terms of the $F_1 - score$, Type 2, which includes the first two best-matched cases in the reuse phase with both high values of precision and recall, is a more balanced tuning for the system.

The conducted experiment evaluates the system's abilities utilizing 45 artificially generated test cases. Although the cases are well studied and generated by an experienced expert, they cover limited types of failures in the oil well drilling domain. An increased number of cases, including cases from real specific drilling operations, would increase the reliability of our results. Testing the system using real cases would be an interesting future work for our group.

## 9 Conclusions

This study has focused on problem solving under uncertainty. The developed and implemented knowledge-based system employs Bayesian analysis to reason on the well understood parts of the knowledge domain. The CBR method is employed to handle the uncertainty of the incommensurable parts while it incorporates the effects of Bayesian analysis in its reasoning process. A similarity model is proposed for assessing the similarity of the cases.

The system is evaluated by an experiment from the oil well drilling area, which was our targeted application domain. The results showed BNCreek acceptable performance to generate a proper solution for an input failure case.

**Acknowledgements** The authors would like to thank Professor Pål Skalle for preparing drilling cases, Professor Helge Langseth, Doctor Kerstin Bach, Doctor Frode Sørmo, and Nur Suriani Mamat for their useful suggestions. Funding was provided by NTNU.



## References

1. Fabre, E.: Bayesian networks of dynamic systems. Habilitation à diriger des recherches, Université de Rennes1 (2007)
2. Kirsch, H., Kroschel, K.: Applying Bayesian networks to fault diagnosis. In: Proceedings of the Third IEEE Conference on Control Applications, Vol. 2, pp. 895–900 (1994)
3. Yongli, Z., Limin, H., Jinling, L.: Bayesian networks-based approach for power systems fault diagnosis. IEEE Trans. Power Deliv. **21**(2), 634–639 (2006)
4. Tirri, H., Kontkanen, P., Myllymäki, P.: A Bayesian framework for case-based reasoning. In: European Workshop on Advances in Case-Based Reasoning, pp. 413–427. Springer (1996)
5. Schiaffino, S.N., Amandi, A.: User profiling with case-based reasoning and Bayesian networks. In: IBERAMIA-SBIA 2000 Open Discussion Track, pp. 12–21 (2000)
6. Bennacer, L., Amirat, Y., Chibani, A., Mellouk, A., Ciavaglia, L.: Self-diagnosis technique for virtual private networks combining bayesian networks and case-based reasoning. IEEE Trans. Autom. Sci. Eng. **12**(1), 354–366 (2014)
7. Moghaddass, R., Rudin, C.: Bayesian patchworks: an approach to case-based reasoning. arXiv preprint arXiv:1809.03541 (2018)
8. Ali, E.S., Darwish, M.: Diagnosing network faults using Bayesian and case-based reasoning techniques. In: 2007 International Conference on Computer Engineering & Systems, pp. 145–150. IEEE (2007)
9. Aamodt, A.: Knowledge-intensive case-based reasoning in creek. In: European Conference on Case-Based Reasoning, pp. 1–15. Springer (2004)
10. Lacave, C., Díez, F.J.: A review of explanation methods for Bayesian networks. Knowl. Eng. Rev. **17**(2), 107–127 (2002)

11. Velasco, F.J.M.: A Bayesian network approach to diagnosing the root cause of failure from trouble tickets. Artif. Intell. Res. **1**(2), 75–85 (2012)
12. Jensen, F.V., et al.: An Introduction to Bayesian Networks, vol. 210. UCL Press, London (1996)
13. Sørmo, F.: Plausible inheritance; semantic network inference for case-based reasoning. Department of Computer and Information Science. Trondheim: Norwegian University of Science and Technology, Vol. 102 (2000)
14. Aamodt, A., Plaza, E.: Case-based reasoning: foundational issues, methodological variations, and system approaches. AI Commun. **7**(1), 39–59 (1994)
15. Aamodt, A., Langseth, H.: Integrating Bayesian networks into knowledge-intensive cbr. In: AAAI Workshop on Case-Based Reasoning Integrations, pp. 1–6 (1998)
16. Bruland, T., Aamodt, A., Langseth, H.: Architectures integrating case-based reasoning and Bayesian networks for clinical decision support. In: International Conference on Intelligent Information Processing, pp. 82–91. Springer (2010)
17. Corchado, J.M., Lees, B.: Adaptation of cases for case based forecasting with neural network support. In: Soft Computing in Case Based Reasoning, pp. 293–319. Springer (2001)
18. Medsker, L.R., Bailey, D.L.: Models and guidelines for integrating expert systems and neural networks. In: Hybrid Architectures for Intelligent Systems, pp. 153–171 (1992)
19. Skalle, P., Aamodt, A., Gundersen, O.E., et al.: Detection of symptoms for revealing causes leading to drilling failures. SPE Drill. Completion **28**(02), 182–193 (2013)
20. Nikpour, H., Aamodt, A., Skalle, P.: Diagnosing root causes and generating graphical explanations by integrating temporal causal reasoning and cbr. CEUR Workshop Proceedings (2017)
21. Nikpour, H., Aamodt, A., Bach, K.: Bayesian-supported retrieval in bncreek: a knowledge-intensive case-based reasoning system. In: International Conference on Case-Based Reasoning, pp. 323–338. Springer (2018)
22. Sowa, J.F.: Principles of Semantic Networks: Explorations in the Representation of Knowledge. Morgan Kaufmann (2014)
23. Cussens, J.: Bayesian network learning with cutting planes. *arXiv preprint* arXiv:1202.3713 (2012)
24. Sørmo, F.: Case-based tutoring with concept maps (2007)
25. Aamodt, A.: A knowledge representation system for integration of general and case-specific knowledge. In: Proceedings Sixth International Conference on Tools with Artificial Intelligence. TAI 94, pp. 836–839. IEEE (1994)
26. Van Harmelen, F., Lifschitz, V., Porter, B.: Handbook of Knowledge Representation. Elsevier (2008)
27. Langseth, H., Portinale, L.: Bayesian networks in reliability. Reliab. Eng. Syst. Saf. **92**(1), 92–108 (2007)
28. Judea, P.: Causality: models, reasoning, and inference. Cambridge University Press. ISBN 0, Vol. 521, No. 77362, p. 8 (2000)
29. Darwiche, A.: Modeling and Reasoning with Bayesian Networks. Cambridge University Press (2009)
30. Gundersen, O.E., Sørmo, F., Aamodt, A., Skalle, P.: A real-time decision support system for high cost oil-well drilling operations. AI Mag. **34**(1), 21–21 (2013)
31. Skalle, P., Aamodt, A., Laumann, K.: Integrating human related errors with technical errors to determine causes behind offshore accidents. Saf. Sci. **63**, 179–190 (2014)
32. Sasaki, Y., et al.: The truth of the f-measure. 2007 (2007)
33. Goutte, C., Gaussier, E.: A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In: European Conference on Information Retrieval, pp. 345–359. Springer (2005)
34. Tague-Sutcliffe, J., Blustein, J.: A statistical analysis of the trec-3 data. NIST SPECIAL PUBLICATION SP, pp. 385–385 (1995)