Proceedings of the 2020 28th Conference on Nuclear Engineering
Joint With the ASME 2020 Power Conference
ICONE28-POWER2020
August 2-6, 2020, Anaheim, California, USA

ICONE28-POWER2020-XXXXX

BENCHMARKING AN AI-GUIDED REASONING-BASED OPERATOR SUPPORT SYSTEM ON THE THREE MILE ISLAND ACCIDENT SCENARIO

Botros N. Hanna, Tran C. Son,
Computer Science Department, NMSU
Las Cruces, NM, USA

Nam T. Dinh
Nuclear Engineering Department, NCSU
Raleigh, NC, USA

ABSTRACT

In the Nuclear Power Plant (NPP) control room, the operators' performance in emergencies is impacted by the need to monitor many indicators on the control room boards, the limited time to interact with dynamic events, and the incompleteness of the operator's knowledge. Recent research has been directed toward increasing the level of automation in the NPP system by employing modern AI techniques that support the operator's decisions. In previous work, the authors have employed a novel AI-guided declarative approach (namely, Answer Set Programming (ASP)) to represent and reason with human qualitative knowledge. This represented knowledge is structured to form a reasoning-based operator support system that assists the operator and compensates for any knowledge incompleteness by performing reasoning to diagnose failures and recommend executing actions in real time. A general ASP code structure has been proposed and tested against simple scenarios, e.g., diagnosis of pump failures that result in loss of flow transients and generating the needed plans for resolving the issue of stuck valves in the secondary loop.

In this work, we investigate the potential of the previously proposed ASP structure by applying ASP to a realistic case study of the Three Mile Island, Unit 2 (TMI-2) accident event sequence (in particular, the first 142 minutes). The TMI scenario presents many challenges for a reasoning system, including a large number of variables, the complexity of the scenario, and the misleading readings. The capability of the ASP-based reasoning system is tested for diagnosis and recommending actions throughout the scenario. This paper is the first work to test and demonstrate the capability of an automated reasoning system by applying it to a realistic nuclear accident scenario, such as the TMI-2 accident.

Keywords: Operator Support System; Automated Reasoning; Diagnosis; Three Mile Island Accident; Decision Making; Logic Programming.

1. INTRODUCTION

1.1 Background

In the Nuclear Power Plant (NPP) control room, various challenges may impact the operators' performance in emergencies. Among these challenges [1] is the need to monitor many indicators on the control room boards, the limited time to interact with dynamic events, and the incompleteness in the operator's knowledge.

Recent research has been directed toward increasing the level of automation in the NPP control room by employing modern AI methods that support the operator's decisions. Some AI methods rely on statistical learning and the availability of big data from the NPP history or the simulations of the NPP transients. Statistical methods have been previously utilized for different purposes, such as faults' detection (see [2], [3]), faults' diagnosis (see [4], [5]), and ranking the available control actions (see [6]). These statistical methods are limited by data availability and data bias. Besides, many data-driven models are not interpretable and the logic behind these models' predictions cannot be explained. Hence, the operators may not trust these black-box methods. Therefore, these statistical methods need to be supported by other methods that represent and reason with human qualitative knowledge.

Another type of AI methods is reasoning-based qualitative methods. These methods are useful to represent our fundamental understanding of the NPP system, the flow and heat paths, the operating and emergency procedures. Researchers have investigated fault diagnosis reasoning-based systems using a logic programming language (see [7][8]) or the java-based rule engines (see [9]–[11]). Event trees are also a standard method to represent our qualitative knowledge of the NPP system. Besides, they can be considered as a search space for corrective actions. Therefore, event trees have been employed to generate possible corrective control actions [6]. None of those above reasoning-based systems ([7]–[11]) was tested against a realistic dynamic scenario, such as the TMI-2 accident.

1.2 This Work

Along with these efforts for NPP autonomous management, the authors have previously proposed an AI-guided reasoning-based operator support system. In our previous work ([12], [13]), we have developed a novel reasoning system that is based on an AI approach (namely, Answer Set Programming (ASP) [14]). Compared to other knowledge representation and reasoning methods, ASP has some attractive features:

- ASP is fully declarative. Declarativity is a programming paradigm that expresses the logic of a computation without describing its control flow [15]. Although ASP's syntax is similar to that of Prolog, the order of rules is insignificant, unlike Prolog [16].
- ASP-based reasoning system proved to have better performance and scalability compared to the Java-based rules engines [17].
- ASP-based knowledge representation and reasoning system has some advantages over knowledge representation using event trees
  - Each single event tree is constructed to represent the possible scenarios that can occur as a result of an initiating event. ASP code is a set of basic facts and logic rules from which all the scenarios can be inferred.
  - ASP code deals with the dynamic environment. The NPP system indicators and the ASP code outputs are changing over time (ASP-based reasoning system answers are updated each timestep)
  - Because all the rules are implemented in a logic program, conditions guarantee the consistency of all these rules exists, i.e., the correctness of systems developed in ASP can be proven formally.

In our previous work ([12], [13]), we have employed ASP to represent our knowledge of the nuclear power plant in the form of logic rules. This represented knowledge is structured to form a reasoning-based operator support system. When an incident occurs, this ASP-based reasoning support system is demonstrated to be capable of fault identification (diagnosis), informing the operator of different scenarios and consequences, and generating the control options (decision making). These efforts ([12], [13]) are part of an ongoing research project designed to develop a Nearly Autonomous Management and Control System for Advanced Reactors (NAMAC [18]).

In this work, we investigate the potential of the previously proposed reasoning system by applying the ASP to a more complex and realistic scenario: The Three Mile Island (TMI-2) accident event sequence (in particular, the first 142 minutes). Compared to the simplified scenarios we considered before ([12], [13]), the TMI-2 scenario presents two significant challenges for a reasoning system:

1. More time-dependent variables

Reasoning about more time-dependent variables implies implementing more rules, increasing the search space, and increases the ASP computational time needed to compute the needed (diagnoses/recommendations) each time step.

2. Misleading readings

One of the complications of the TMI-2 scenario is the fact that the pressurizer Pilot-Operated Relief Valve (PORV) was stuck open. However, the operators believed that it was closed because of the PORV corresponding light (see Sec. 2 for details).

The capability of the ASP-based reasoning system is tested for diagnosis and recommending actions throughout the scenario. To our knowledge, this is the first reasoning system to address the TMI-2 scenario. An overview of the TMI-2 scenario is presented in Sec. 2. The ASP method is briefed in Sec. 3. The ASP-based reasoning method is proposed and applied to the TMI-2 scenario in Sec. 4. The conclusions of this work are discussed in Sec 5.

2. SYNOPSIS OF TMI-2 ACCIDENT

TMI-2 NPP contained a Pressurized Water Reactor (PWR) with a reactor vessel, four reactor coolant pumps, and a pressurizer. The Reactor Coolant System (RCS) consists of 2 flow loops (loops A and B), each with a once-through steam generator. TMI-2 was the newest unit on site, and this unit was operated at 97% of the full power [19]. Two off-normal conditions existed before the initiating event: 1- A small leakage occurred in the pressurizer Pilot-Operated Relief Valve (PORV). This leakage raised the temperature downstream of the PORV. 2- Two emergency feedwater valves, in the secondary loop, were closed (by mistake).

The accident was initiated on March 28th, 1979, by the trips of the condensate and feedwater pumps while the staff was attempting to fix a blockage in one of the condensate polishers (resin filters that clean the secondary loop water). The turbine trip followed the trips of the pumps. Auxiliary emergency feedwater pumps started to provide the steam generators with the feedwater, but the emergency feedwater valves were inadvertently left closed. Because of the lack of water to the steam generators, the primary loop water was heating up, expanding, and flowing to the pressurizer. Pressure in the primary system increased, so the pressurizer PORV was automatically opened.

The reactor tripped because of the high reactor pressure as the RCS was heating up. Pressurizer PORV was opened to release the pressure, but it failed to close (was stuck open) when the pressure decreased. Operators believed that the pressurizer PORV was closed because of the control room light, while this light was only an indication that an electric signal was sent to close the PORV. Checking the downstream temperature at the PORV was the only way to find whether the PORV is closed, but the temperature was already high (before the accident) because of the small PORV leakage. Reactor pressure continued to decrease, the coolant temperature continued to increase, and the High-Pressure Injection System (HPIS) pumps were automatically turned on to re-pressurize the RCS.

Water expansion and pressure decrease in the primary loop led to the generation of steam in the reactor core and the rise of water in the pressurizer. The operators decided to turn off the HPIS (to avoid filling the system with water) while the water level in the reactor was decreasing. Turning off the HPIS led to

the absence of cooling, overheating, and fuel damage. Pressurizer water level continued to increase because of the growth of the steam region in the core. The combination of a rising pressurizer water level, a decreasing primary loop pressure, and rising primary loop temperature was not understandable to the operators. Meanwhile, water escaping through the PORV filled a drain tank, and the primary loop pumps were shut because of the water-steam mixture in the primary loop. It took more than 2 hours before the operators realized that the PORV is stuck open, and they closed the PORV block valve. Primary loop temperatures continued to increase, so the operators decided to turn on the HPIS until the reactor core was finally filled with water [19].

While the TMI-2 occurred as a result of various failures (mechanical failures, human error, and lack of training), the focus of this work is the reasoning process. In such a complex dynamic scenario, it is challenging for the operators to perform reasoning for diagnosis and making timely decisions considering the changes of many variables. For instance, during the accident, many alarms were turned on, and the operator did not notice the containment sump high water level alarm (which indicated the leakage in the primary loop). Also, the operators were concerned about a rising primary water level, and they did not notice that the primary loop water reached saturation pressure [20].

In this work, the potential of an automated reasoning system is investigated by applying an ASP to the TMI-2 accident event sequence (in particular, the first 142 minutes). The sequence of events that are considered when constructing the ASP-based reasoning system is presented in TABLE 1.

Table 1. TMI-2 sequence of events.

Time (seconds)	Event
1	Condensate pump trips.
2	The main feedwater pumps trip.
2	The turbine trips.
2	Emergency auxiliary feedwater pumps start.
7	Primary pressure reached PORV setpoint, and the PORV opens.
11	Reactor trips.
11	Primary pressure reduced below PORV setpoint, but the PORV remains open.
122	HPIS starts automatically.
279	Operators throttle HPIS.
499	The block valve on the emergency feedwater pump line is opened (loop A).
500	The block valve on the emergency feedwater pump line is opened (loop B).
4402	The primary loop pumps are tripped offline (loop A).
6036	The primary loop pumps are tripped offline (loop B).
8521	The block valve on the pressurizer drain line is closed.

3. ANSWER SET PROGRAMMING

Answer set programming (ASP) [14] is a high-level and expressive language that is well-suited for knowledge representation, reasoning, and solving combinatorial search as well as optimization problems. ASP can be used to declaratively represent the knowledge about dynamic systems and solve combinatorial search problems such as diagnosis and planning. ASP allows domain knowledge representation, including commonsense knowledge, incomplete knowledge, defaults, preferences, and negation. There is a growing number of ASP applications in various areas [21]. In the area of diagnosis, an ASP system was used to analyze the failures of Google's ads [22]. In the area of planning, an ASP system was employed to generate plans for the space shuttle's maneuvers [23]. ASP is also employed to detect inconsistent information [24] in addition to applications in the robotics domain [25]. ASP became attractive for researchers and industry because of the availability of ASP solvers. In this work, we use *Clingo* [26], a free ASP solver.

In ASP, a program is a set of logic programming rules, facts, and constraints about some problem domain. ASP is oriented toward solving combinatorial search problems where the goal is to find a solution(s) among a finite large number of possibilities. Figure 1 shows briefly the general steps in solving a problem using ASP. As illustrated in Figure 1, modeling the problem leads to creating a formal abstract general problem representation. The Solving process is done by the computer, using ASP solver. We have presented the ASP problem-solving steps in more detail in previous work [13]. It is worth noting that several additional features have been added to ASP (e.g., aggregates, choices, etc.) to enable and simplify the use of ASP. Furthermore, ASP solver with multi-shot capability has been developed, which allows the programmers to change the computation, thus providing a method for implementing a closed-loop system as described in this paper.

We formalized a general structure of an ASP code (that can be utilized to find the diagnoses or the needed actions) in an NPP system (see next section).

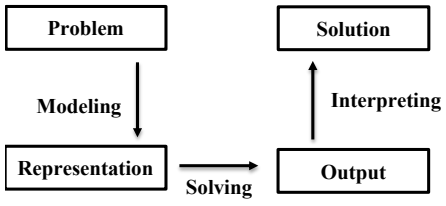

Figure 1. The programming paradigm in problem-solving using ASP code [18].

4. OPERATOR SUPPORT SYSTEM

The structure of the ASP-based reasoning system is depicted in Figure 2.

4.1 Inputs

As shown in Figure 2, the reasoning system has two types of *inputs*:

A. The observed variables (the indicators): among many measured variables, the list of observed variables (16 variables) that are relevant to this scenario are listed in Table 2. Real measurements [27] of seven of these variables (for the first 142 minutes) are depicted in Figure 3. The remaining variables have binary values (for instance, a pump is *on* or *off*).

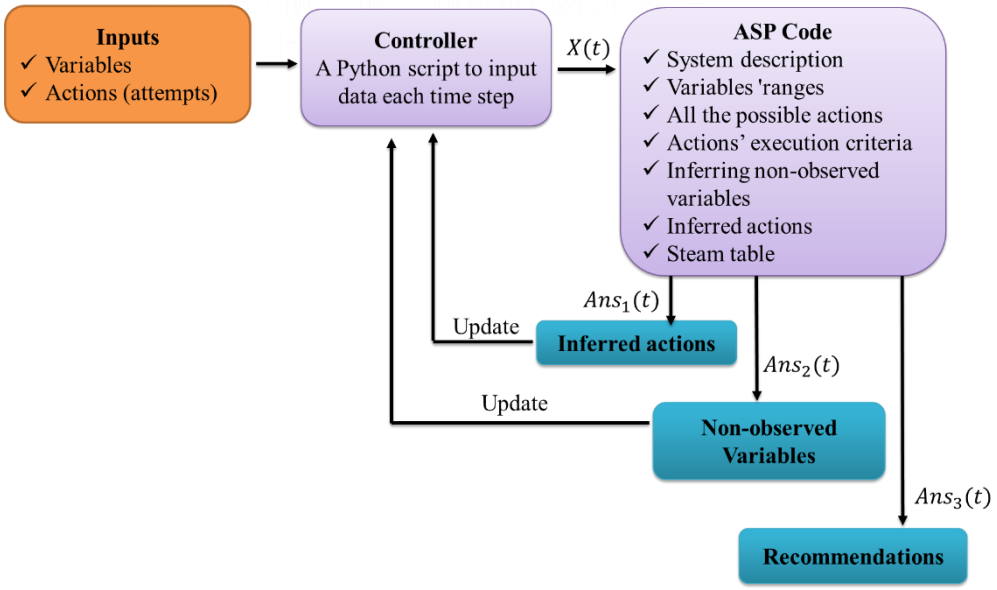

Figure 2. ASP-based reasoning system for diagnoses and recommendations.

Table 2. A list of TMI-2 scenario relevant variables.

1. The reactor coolant system pressure (in PSI).
2. The pump flow rate of the condensate pump in the secondary loop (loop A).
3. The pump flow rate of the condensate pump in the secondary loop (loop B).
4. The pump flow rate of the main feedwater pump in the secondary loop (loop A).
5. The pump flow rate of the main feedwater pump in the secondary loop (loop B).
6. The pump flow rate of the auxiliary feedwater pump in the secondary loop (loop A).
7. The pump flow rate of the auxiliary feedwater pump in the secondary loop (loop B).
8. The pump flow rate of the high-pressure injection pump
9. The reactor power.
10. The steam generator water level (loop A) in centimeters.
11. The steam generator water level (loop B) in centimeters.
12. The two primary pumps' flow rates (loop A).
13. The two primary pumps' flow rates (loop B).
14. RCS inlet temperature (loop A).
15. RCS inlet temperature (loop B).
16. Turbine power.

B. The attempted actions: actions that are known (to the operator) to be executed (or attempted) by the operator or by any automatic safety system. These actions may or may not have been executed successfully. All these actions are corresponding to some of the events in Table 1. The list of these actions is implemented in the ASP code in the form of: *attempted (procedure, component, time in seconds)*. These actions are:

attempted(open,pressurizer_pilot_operated_relief_valve,7).
attempted(close,pressurizer_pilot_operated_relief_valve,11).
attempted(turn_off,high_pressure_injection_pump,279).
attempted(open,auxiliary_feedwater_pump_a_block_valve,499)
attempted(open,auxiliary_feedwater_pump_b_block_valve,500)
attempted(turn_off,primary_pump_a,4402).
attempted(turn_off,primary_pump_b,6036).
attempted(close,pressurizer_block_valve,8521).

The inputs (variables and actions) are passed to the ASP code each time step (for 8521 timesteps corresponding to 8521 seconds), similar to the process of data streaming in the NPP system. This real-time streaming is enabled by a Python script that extracts the inputs and passes them to the ASP code every time step (the *controller* in Figure 2).

4.2 ASP Knowledge Base

The knowledge base represented by ASP includes the following elements:

A. System description:

The list of components and the connections between these components.

B. Variables' ranges:

The possible range of each variable is known and discretized.

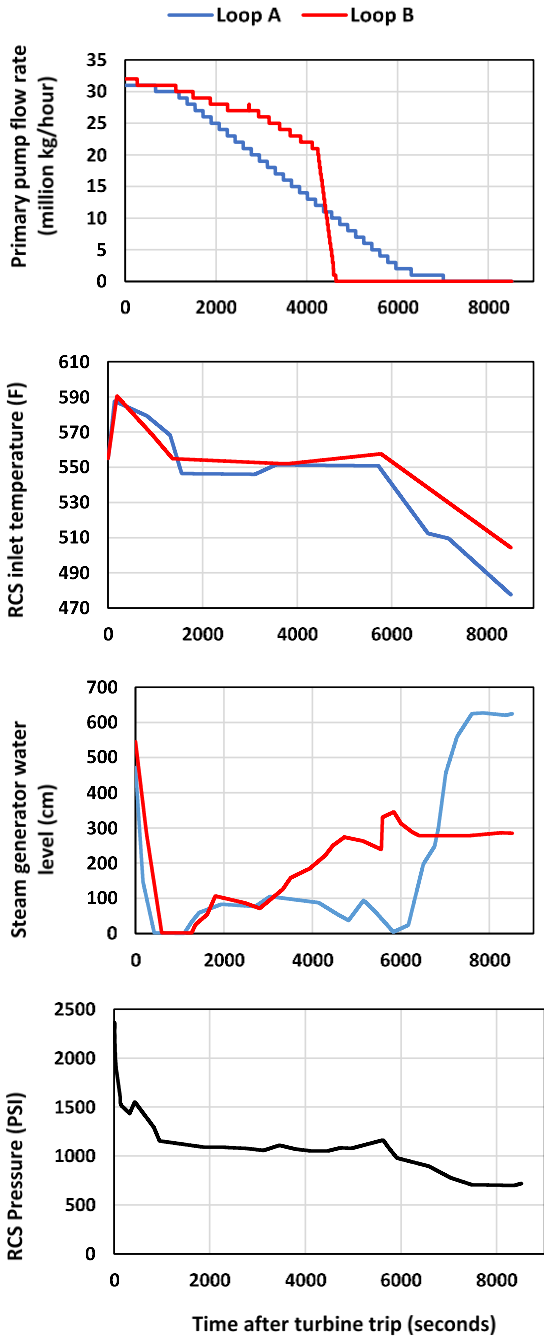

Figure 3. TMI-2 data [27].

C. All the possible actions:

We hypothesize that all the executable actions are known, and the reasoning system searches for the needed action each time step.

D. The actions' executability conditions:

For each action, we defined the executability condition so the action cannot be executed (or recommended) if the execution criteria are not satisfied. The execution criteria for the actions relevant to the TMI-2 scenario can be listed as follows:

1. Tripping the reactor
   - The reactor is tripped *if* the pressure is higher than a predefined value, and the reactor is not already tripped.
2. Turning on pumps
   - Auxiliary pumps are started *if* another pump in the same loop was tripped, and the auxiliary pumps are not already running.
   - The HPIS pump is turned on *if* the pressure is lower than a predefined value, and the HPIS pump is not already running.
3. Opening valves
   - The auxiliary pump block valve is opened *if* the auxiliary pump is turned on, and its block valve is not.
   - The auxiliary pump block valve is opened *if* the auxiliary pump is turned on, and a lack of water supply is detected (inferred) in the flow loop to which this pump belongs.
   - The pressurizer PORV is opened *if* the pressure is higher than a predefined value.
   - The pressurizer PORV is closed *if* the pressure is lower than a predefined value
   - The pressurizer backup block valve is closed *if* the pressurizer valve is detected to be stuck open.

E. The inferred non-observed variables:

These are the variables that cannot be measured but are inferred based on the observed variables. The relevant non-observed variables are:

1. Lack of water supply
   - Lack of water supply is inferred *if* one of the pumps in the secondary loop is tripped, and the corresponding steam generator water level is below a minimum value, or the water level is continuously decreasing.
2. Stuck valve
   - The PORV is considered stuck open *if* the valve was opened, and the pressure continued to decrease below a predefined minimum value at which the valve should have closed.
   - The PORV is considered stuck open *if* the drain tank temperature is higher than a predefined temperature, and the pressure in the primary loop is lower than a specific value.
3. The status of the coolant:
   - Depending on the coolant inlet temperature and pressure, the status of the coolant can be inferred (whether the coolant is still pressurized water or saturated or superheated steam).

F. The Inferred actions:
These are the actions that are implied even if not attempted/executed by the operator. The action effect is detected by watching the observed variables. For example, the reasoning system infers that a pump is tripped if its flow rate changes from any value above zero to zero.

G. Steam table:
A simple steam table is implemented (the water saturation temperature and its corresponding saturation pressure). The existence of steam in the coolant can be inferred using the steam table.

4.3 ASP-Based Reasoning System Output

As shown in Figure 2, the ASP-based reasoning system gives three time-dependent sets of answers (outputs). The first output (Ans_1) is the *inferred actions* (the actions that were inferred) even if not executed by the operator. These actions may be similar or different from the inputted attempted actions. The occurrence of these actions is verified by detecting the action effect and watching the observed variables. Listing 1 shows the inferred actions (if any) at each time step. Each action has the form: *happened(procedure, component, time in seconds)*.

```
% TIME =  0 Second
% TIME =  1 Second
happened(trip,condensate_pump_a,1).
happened(trip,condensate_pump_b,1).
% TIME =  2 Second
happened(trip,feedwater_pump_a,2).
happened(trip,feedwater_pump_b,2).
happened(trip,turbine1,2).
happened(start,auxiliary_feedwater_pump_a,2).
happened(start,auxiliary_feedwater_pump_b,2).
% TIME =  11 Second
happened(trip,reactor1,11).
% TIME =  122 Second
happened(start,high_pressure_injection_pump,122).
% TIME =  278 Second
happened(trip,high_pressure_injection_pump,278).
% TIME =  4403 Second
happened(trip,primary_pump_a,4403).
% TIME =  6037 Second
happened(trip,primary_pump_b,6037).
```

Listing 1. The first output (all the inferred actions).

The second output ($Ans2$) is the inferred non-observed variables. These are the significant variables that are inferred based on the observed variables. Non-observed variables were inferred at each timestep, but we show the answers only at some of these timesteps in Listing 2. For instance, the lack of water supply was inferred very early ($t=2$) because of the low steam generator water level. Starting from $t=16$, the PORV was inferred to be stuck open (because of the continuous decrease in the RCP pressure below the PORV setpoint pressure). The existence of steam, in the primary loop, was also inferred.

```
% TIME =  2 Second
lack_of_water_supply(secondary_loop_A,2).
lack_of_water_supply(secondary_loop_B,2).
% TIME =  3 Second
lack_of_water_supply(secondary_loop_A,3).
lack_of_water_supply(secondary_loop_B,3).
% TIME =  4 Second
lack_of_water_supply(secondary_loop_A,4).
lack_of_water_supply(secondary_loop_B,4).
% TIME =  5 Second
lack_of_water_supply(secondary_loop_A,5).
lack_of_water_supply(secondary_loop_B,5).
% TIME =  16 Second
lack_of_water_supply(secondary_loop_A,16).
lack_of_water_supply(secondary_loop_B,16).
stuck_open(pressurizer_pilot_operated_relief_valve, 16).
% TIME =  1000 Second
lack_of_water_supply(secondary_loop_A,1000).
lack_of_water_supply(secondary_loop_B,1000).
steam(primary_loop_B,1000).
steam(primary_loop_A,1000).
stuck_open(pressurizer_pilot_operated_relief_valve,
1000).
% TIME =  4000 Second
stuck_open(pressurizer_pilot_operated_relief_valve,
4000).
% TIME =  6000 Second
lack_of_water_supply(secondary_loop_A,6000).
steam(primary_loop_B,6000).
stuck_open(pressurizer_pilot_operated_relief_valve,
6000).
% TIME =  7000 Second
steam(primary_loop_B,7000).
stuck_open(pressurizer_pilot_operated_relief_valve,
7000).
% TIME =  8000 Second
steam(primary_loop_B,8000).
stuck_open(pressurizer_pilot_operated_relief_valve,
8000).
% TIME =  8521 Second
stuck_open(pressurizer_pilot_operated_relief_valve,
8521).
```

Listing 2. The second output (some of the non-observed variables).

The third output (Ans_3) is the recommendations. The recommendations are the actions that are recommended if their execution criteria are satisfied at any timestep. The same recommendation can be given repeatedly until its execution criteria are no longer satisfied. Listing 3 shows the

recommendations that the reasoning system suggests each timestep. We have chosen some timesteps to show their corresponding recommendations in Listing 3).

```
% TIME =  1 Second
recommendation(start,auxiliary_feedwater_pump_a,1).
recommendation(start,auxiliary_feedwater_pump_b,1).
% TIME =  2 Second
recommendation(open,auxiliary_feedwater_pump_a_block_valve,2).
recommendation(open,auxiliary_feedwater_pump_b_block_valve,2).
% TIME =  30 Second
recommendation(close,pressurizer_block_valve,30).
recommendation(close,
pressurizer_pilot_operated_relief_valve,30).
recommendation(open,auxiliary_feedwater_pump_a_block_valve,30).
recommendation(open,auxiliary_feedwater_pump_b_block_valve,30).
% TIME =  2000 Second
recommendation(close,pressurizer_block_valve,2000).
recommendation(close,
pressurizer_pilot_operated_relief_valve,2000).
recommendation(start,high_pressure_injection_pump,2000).
% TIME =  6000 Second
recommendation(close,pressurizer_block_valve,6000).
recommendation(close,
pressurizer_pilot_operated_relief_valve,6000).
recommendation(open,auxiliary_feedwater_pump_a_block_valve,6000).
recommendation(start,high_pressure_injection_pump,6000).
% TIME =  7000 Second
recommendation(close,pressurizer_block_valve,7000).
recommendation(close,
pressurizer_pilot_operated_relief_valve,7000).
recommendation(start,high_pressure_injection_pump,7000).
% TIME =  8521 Second
recommendation(close,pressurizer_block_valve,8521).
recommendation(close,
pressurizer_pilot_operated_relief_valve,8521).
recommendation(start,high_pressure_injection_pump,8521).
```

Listing 3. The third output (some of the recommendations).

For example, a recommendation to start the emergency feedwater pumps, at *t=1,* is suggested because of the tripped pumps in the secondary loop. Because these emergency feedwater pumps started automatically at *t=2,* this recommendation is no longer proposed in the next timesteps. Next, the block valve corresponding to each pump is recommended to be opened. At *t=30*, the pressurizer block valve is recommended to be closed, and so on.

4.4 Computational Expense and Data Accumulation

ASP problem solving includes searching for solutions that satisfy logic rules/constraints. The longer the scenario and the more timesteps are accounted for by the ASP-based reasoning system, the larger the search space is. If the reasoning system "remembers" all the variables' values and the occurrences at each timestep, the computational expense will exacerbate over time. Therefore, it is vital to decide which data are "remembered" and which data are ignored. To resolve this issue, the observed variables and the attempted actions, that occurred more than one minute before the current timestep, are ignored. On the other side, all the inferred actions and inferred non-observed variables are "remembered."

In this work, the computational expense of ASP-based reasoning, about variables and actions within 142 minutes (8521 seconds) of the accident scenario, is 96 minutes (on a four-processor machine). One of the significant challenges that face the proposed ASP-reasoning system is the large number of variables in the NPP system (a higher computational expense). Besides, the values of all variables were updated each second, which may not be needed.

Another challenge that faces this knowledge representation system is the difficulty of verifying whether the implemented qualitative knowledge is complete. Therefore, the reasoning system needs to be reviewed and tested.

5. CONCLUSIONS

Recently, AI methods that support the operator decisions in the control room have been proposed. Statistical AI methods rely on the availability of big data from the NPP history or the simulations of the NPP transients. The data-driven methods are limited by data availability and data bias. Besides, many data-driven models are not interpretable, and the operator may not trust the logic behind these black-box models' predictions. Hence, these statistical methods need to be supported by other reasoning-based methods that represent human qualitative knowledge, such as the Answer Set Programming (ASP) based reasoning method proposed in this work. Compared to other reasoning-based methods that have been proposed in the literature to support the operator decisions, the proposed ASP reasoning system has better performance, scalability, and declarativity. Additionally, the previously proposed reasoning systems have not been tested against a realistic dynamic scenario.

In this work, we applied an ASP-based reasoning system to the Three Mile Island Unit 2 (TMI-2) accident event sequence (in particular, the first 142 minutes). The ASP-based Reasoning system accounted for 16 variables that change over 8521 timesteps to give faster than real-time answers each timestep. These answers include inferring non-observed variables (such as leakage and stuck valves) and giving recommendations (suggested actions) to the operator.

Based on this study, the proposed ASP-based reasoning system has the potential to assist the operator in making timely decisions because:

1. ASP can perform reasoning about many variables while the human capacity to monitor and reason about many variables is limited. In the TMI-2 scenario, the operator was overwhelmed by many alarms and monitoring many dynamic events. For instance, the operator did not notice the containment sump high water level alarm (which indicated the leakage in the primary loop). In this work, the computational expense of ASP-based reasoning, accounting for 16 variables and eight actions (attempted by the operator) within 142 minutes of the accident scenario, is 96 minutes only. Although the number of variables, 16, is small compared to hundreds of indicators in the NPP, it is worth noting that we only used a four-processor machine, and there is an opportunity of reasoning about more variables by exploiting the high-performance machines. Additionally, we assumed that NPP variables are updated each one second, so new answers (diagnoses and recommendations) are required each second. Depending on the variables' time scale, answers can be generated at a slower rate to decrease the ASP computational time.
2. The proposed ASP code structure could distinguish between the executed actions and the attempted actions by watching the action effect. This feature is significant for the TMI-2 scenario because of the confusion about the PORV (an action to close the PORV was attempted, but the valve was not closed, and the primary loop pressure kept decreasing).
3. Because of the long duration of the TMI-2 scenario (and NPP transients), more timesteps are accounted for by the ASP-based reasoning system, and the computational cost exacerbates, To resolve this issue, the observed variables and the attempted actions, that occurred more than one minute before the current timestep, are ignored. On the other side, all the inferred actions and the inferred non-observed variables are "remembered." Hence, the ASP reasoning system could provide timely answers.

This work is the first automated reasoning system to be applied to a complex, realistic case study such as the TMI-2 scenario. We demonstrated the potential of applying an ASP-based reasoning operator support system to a real-life scenario.

In future work, we will demonstrate that rational explanations for each ASP answer can be computed. Generating explanations for any diagnosis or recommendation, suggested by the reasoning system, allows the operator to check the rationality of each explanation and decide whether to trust the ASP reasoning system answers.

ABBREVIATIONS

AI	Artificial Intelligence
Ans	Answer
ASP	Answer Set Programming
HPIS	High-Pressure Injection System
NAMAC	Nearly Autonomous Management and Control
NPP	Nuclear Power Plant
PORV	Pilot-Operated Relief Valve
PWR	Pressurized Water Reactor
RCS	Reactor Coolant System
TMI	Three Mile Island

ACKNOWLEDGMENTS

This research is supported by the US Department of Energy's Advanced Research Project Agency-Energy (ARPA-E) MEITNER Program through award DE-AR0000976. The authors would like to thank Dr. Robert Youngblood (Idaho National Laboratory) for his stimulating and insightful comments.

REFERENCES

[1] R. L. Boring, K. D. Thomas, T. A. Ulrich, and R. T. Lew, "Computerized Operator Support Systems to Aid Decision Making in Nuclear Power Plants," Procedia Manuf., vol. 3, pp. 5261–5268, 2015.

[2] B. R. Upadhyaya, K. Zhao, S. R. P. Perillo, X. Xu, and M. G. Na, "Autonomous Control of Space Reactor Systems," Knoxville, TN, 2007.

[3] M. Peng et al., "An intelligent hybrid methodology of on-line system-level fault diagnosis for nuclear power plant," Nucl. Eng. Technol., vol. 50, no. 3, pp. 396–410, Apr. 2018.

[4] K. Groth, M. Denman, J. Cardoni, and T. Wheeler, "'Smart Procedures' : Using dynamic PRA to develop dynamic, context-specific severe accident management guidelines (SAMGs)," in PSAM International Conference on Probabilistic Safety Assessment and Management, 2014.

[5] D. Lee, P. H. Seong, and J. Kim, "Autonomous operation algorithm for safety systems of nuclear power plants by using long-short term memory and function-based hierarchical framework," Ann. Nucl. Energy, vol. 119, pp. 287–299, Sep. 2018.

[6] S. M. Cetiner et al., "Supervisory Control System for Multi-Modular Advanced Reactors," Oak Ridge, TN, 2016.

[7] S. H. Chang, K. S. Kang, S. S. Choi, H. G. Kim, H. K. Jeong, and C. U. Yi, "Development of the On-Line Operator Aid System OASYS Using A Rule-Based Expert System and Fuzzy Logic for Nuclear Power Plants," Nucl. Technol., vol. 112, no. 2, pp. 266–294, Nov. 1995.

[8] H. Qudrat-Ullah, "QES-shell: An expert system for nuclear power plant operator's training," in Proceedings - 3rd International Conference on Intelligent Systems Modelling and Simulation, ISMS 2012, 2012, pp. 151–156.

[9] M. Lind and X. Zhang, "Functional modeling for fault diagnosis and its application for NPP," Nucl. Eng. Technol., vol. 46, no. 6, pp. 753–772, Dec. 2014.

[10] J. Reifman and T. Y. C. Wei, "PRODIAG: A Process-Independent Transient Diagnostic System - I: Theoretical Concepts," Nucl. Sci. Eng., vol. 131, no. 2–3, pp. 329–347,

1999.
[11] Y. S. Park and R. Vilim, “Implementation of new PRODIAG algorithm and simulation-based acceptance test,” in 10th International Topical Meeting on Nuclear Plant Instrumentation, Control, and Human-Machine Interface Technologies, NPIC and HMIT 2017, 2017, vol. 2, pp. 884–893.
[12] B. Hanna, T. C. Son, and N. Dinh, “An Artificial Intelligence-Guided Decision Support System for The Nuclear Power Plant Management,” in 18th International Topical Meeting on Nuclear Reactor Thermal Hydraulics (NURETH 2019), 2019, pp. 394–406.
[13] B. Hanna, T. C. Son, and N. Dinh, “AI-GUIDED REASONING-BASED OPERATOR SUPPORT SYSTEM FOR THE NUCLEAR POWER PLANT MANAGEMENT,” Under Prep., 2020.
[14] M. Gebser, R. Kaminski, B. Kaufmann, and T. Schaub, “Answer set solving in practice,” Synth. Lect. Artif. Intell. Mach. Learn., vol. 19, pp. 1–240, 2012.
[15] J. Lloyd, “Practical advantages of declarative programming,” Jt. Conf. Declar. Program. GULP-PRODE, pp. 1–15, 1994.
[16] W. F. Clocksin and C. S. Mellish, Programming in Prolog : Using the ISO Standard. Berlin Heidelberg: Springer Science & Business Media., 2003.
[17] S. Liang, P. Fodor, H. Wan, and M. Kifer, “OpenRuleBench: An analysis of the performance of rule engines,” in WWW’09 - Proceedings of the 18th International World Wide Web Conference, 2009, pp. 601–610.
[18] “Development of a Nearly Autonomous Management and Control System for Advanced Reactors,” ARPA-E, 2018. [Online]. Available: https://arpa-e.energy.gov/?q=slick-sheet-project/management-and-control-system-advanced-reactors.
[19] Nuclear Safety Analysis Center, “Analysis of Three Mile Island - Unit 2 Accident,” Palo Alto, California, 1980.
[20] M. Derivan, “The Davis Besse Nuclear Power Plant Three Mile Island Accident Precursor Event.” 2014.
[21] A. Falkner, · Gerhard Friedrich, K. Schekotihin, · Richard Taupe, and E. C. Teppan, “Industrial Applications of Answer Set Programming,” KI - Künstliche Intelligenz, vol. 32, pp. 165–176, 2018.
[22] A. Brik and J. Remmel, “Diagnosing automatic whitelisting for dynamic remarketing ads using hybrid ASP,” in International Conference on Logic Programming and Nonmonotonic Reasoning, 2015, vol. 9345, pp. 173–185.
[23] M. Nogueira, M. Balduccini, M. Gelfond, R. Watson, and M. Barry, “An a-prolog decision support system for the space shuttle,” in International symposium on practical aspects of declarative languages, 2001, vol. 1990, pp. 169–183.
[24] M. Albanese, M. Broecheler, J. Grant, M. V. Martinez, and V. S. Subrahmanian, “PLINI: A probabilistic logic program framework for inconsistent news information,” in Logic programming, knowledge representation, and nonmonotonic reasoning, vol. 6565 LNAI, Berlin Heidelberg: Springer, 2011, pp. 347–376.
[25] E. Erdem, E. Aker, and V. Patoglu, “Answer set programming for collaborative housekeeping robotics: Representation, reasoning, and execution,” Intell. Serv. Robot., vol. 5, no. 4, pp. 275–291, 2012.
[26] M. Gebser et al., A User’s Guide to gringo, clasp, *clingo, and iclingo*. 2008.
[27] J. L. Rempe and D. L. Knudson, “Instrumentation Performance during the TMI-2 Accident,” *IEEE Trans. Nucl. Sci.*, vol. 61, no. 4, p. 1963, 2014.