International Journal of Hybrid Intelligent Systems 5 (2008) 19–29
IOS Press


# Identification of best sets of actions in Influence Nets

Sajjad Haider*, Abbas K. Zaidi and Alexander H. Levis
*System Architectures Lab, George Mason University, Virginia 22030, USA*

**Abstract**. This paper presents a heuristic approach to solve the problem of best set of actions determination in Influence Nets. Influence Nets are a special instance of Bayesian Networks that model uncertain situations by connecting a set of desired effects to a set of actionable events through chains of probabilistic cause and effect relationships. Once an Influence Net is specified, a system analyst is often interested in identifying the set of action which has the highest probability of achieving a desired effect. The existing techniques to solve this problem, such as sensitivity analysis and exhaustive search, have limitations of their own. The proposed algorithm, named SAF, attempts to overcome these limitations. The paper also shows that the problem of best set of actions determination can be formulated as an instance of Mixed Integer Non Linear Programming (MINLP). An empirical study is presented that compares the performance of sensitivity analysis, SAF, and MINLP.

Keywords: Bayesian Networks, sensitivity analysis, Influence Nets, probabilistic reasoning, combinatorial optimization, mixed integer nonlinear programming

## 1. Introduction

Bayesian Networks (BNs) [19,20] have become a popular tool in the Computer Science community for modeling uncertainty. The last two decades have seen an emergence of applications that use BNs for modeling various types of uncertain situations. These include medical diagnosis, human belief systems, forecasting, sensor fusion, system troubleshooting, etc. Despite their popularity, BNs suffer from two major limitations. The first is the amount of data required to completely specify a BN and the second is the intractability of exact belief propagation algorithms [6]. Work is reported in the literature that attempts to overcome these two limitations. Schemes have been proposed that reduce the amount of data that is required to completely specify a BN. Among them are Bayesian Network with Noisy-Or (BN2O) [1,8] and the CAusal STrength (CAST) logic [3,21]. To address the intractability of belief propagation algorithms, several approximation and simulation based algorithms have been proposed. These include the loopy belief propagation [9,17,18,20], logic sampling [16], likelihood weighting [10], backward simulation [11], and importance sampling [2,5]. A detailed comparison of different simulation schemes is given in Shachter and Poet [23] and Cheng [4].

Influence Nets (INs) [21,22] were developed with a similar aim of overcoming the above two limitations of BNs. They use the CAST logic for knowledge elicitation and a variant of loopy belief propagation for probabilistic inference. INs and their extensions, Timed Influence Nets (TINs) [13,14], have been used experimentally to model Effects Based Operations [24–26]. They allow a system analyst to connect a set of desired effects and a set of actionable events through chains of probabilistic cause and effect relationships. Once an IN is specified, an analyst is often confronted with the task of identifying the set (or sets) of actions that maximizes the likelihood of achieving a desired effect. Currently, there are two approaches to solve this problem: sensitivity analysis (SA) and exhaustive search (ES). Both approaches have limitations of their own. SA only looks at the individual impact of an action on a desired effect. It assumes that actions having positive individual impacts would have maximum positive im-

*Corresponding author. E-mail: shaider@gmualumni.org.

pact on the desired effect when considered collectively. Unfortunately, there are two drawbacks with this assumption. Firstly, because of the nonlinearity of the conditional probabilities associated with events in an IN, the combined impact may not always be synergistic. Secondly, it may be the case that actions having individual negative impact may have a positive impact when considered with certain other actions. Thus, despite the fact that SA requires linear number of searches to find the best set of actions, it fails to look at the combined impact of actions on the desired effect. This failure results in achieving a sub-optimal set of actions. Exhaustive search, on the other hand, looks at the combined impact of actions, but it requires an exponential number of searches. If there are ‘n’ actionable events, then ES would require $2^n$ searches.

This paper presents a heuristic approach to the problem of best sets of actions determination in an Influence Net. The approach, named Sets of Actions Finder (SAF) [15], attempts to overcome the limitations of both SA and ES. Unlike SA, SAF looks at the combined impact of actionable events and it does so in polynomial number of searches in contrast to the ES which requires an exponential number of searches. One drawback of SAF is that due to its hill-climbing nature, there is always a possibility that it may stick to local optima. An empirical study is presented in this paper that analyzes the performance of SAF over thousands of IN models.

The paper also shows that the problem of best sets of actions determination can be formulated as an instance of Mixed Integer Nonlinear Programming (MINLP). Besides providing an alternative way of solving the problem of best set of actions determination, the formulation can also be used as a bench mark to the performance of SAF and SA. The empirical study, mentioned above, is conducted for SAF, MINLP, and SA. Each of these approaches is run over thousands of IN models. The optimal value produced by them is compared with the ones obtained through ES. The study provides an estimate of the performance of different approaches over different classes of INs.

The rest of the paper is organized as follows: Section 2 provides a brief introduction of INs. Sensitivity Analysis is presented in Section 3 while the SAF algorithm is explained in Section 4. The MINLP formulation of the problem is described in Section 5. Section 6 presents the empirical study while Section 7 concludes the paper and provides future research directions.

## 2. Influence Nets

Influence Nets are Directed Acyclic Graphs (DAGs) where nodes in the graph represent random variables, while the edges between pairs of variables represent causal relationships. The modeling of the causal relationships is accomplished by creating a series of cause and effect relationships between variables representing desired effect(s) and variables representing set of actionable events. The actionable events are drawn as root nodes (nodes without incoming edges), while the desired effect is modeled as a leaf node (node without outgoing edges). Typically, the root nodes are drawn as rectangles while the non-root nodes are drawn as rounded rectangles. Influence Nets require a system modeler (or a subject matter expert) to specify the CAST logic parameters instead of the probabilities. The required probabilities are internally generated by the CAST logic algorithm with the help of user-defined parameters. The Influence Nets are therefore appropriate for the following situations: i) for modeling situations in which it is difficult to fully specify all conditional probability values, and/or ii) the estimates of conditional probabilities are subjective, and iii) estimates for the conditional probabilities cannot be obtained from empirical data, e.g., when modeling potential human reactions and beliefs. The following items characterize an IN while a formal definition is given in Definition 1.

1. A set of random variables that makes up the nodes of an IN. All the variables in the IN have binary states.
2. A set of directed links that connect pairs of nodes.
3. Each link has associated with it a pair of CAST logic parameters that shows the causal strength of the link (usually denoted as h and g values).
4. Each non-root node has an associated CAST logic parameter (denoted as the baseline probability), while a prior probability is associated with each root node.

**Definition 1.** An *Influence Net* is a four-tuple (**V, E, C, B**) where

**V**: set of Nodes,
**E**: set of Edges,
**C**: represents causal strengths:

$$\mathbf{E} \rightarrow \{ (\mathbf{h}, \mathbf{g}) \text{ such that } -1 < \mathbf{h}, \mathbf{g} < 1 \},$$

**B** represents a Baseline or Prior probability:
$\mathbf{V} \rightarrow [0,1]$

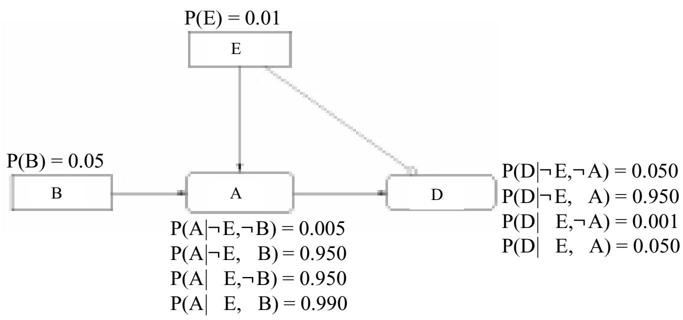


Fig. 1. A sample Influence Net.

Figure 1 shows an example of an Influence Net. Nodes B and E represent the actionable events (root nodes) while node D represents the desired effect (leaf node). The directed edge with an arrowhead between two nodes shows the parent node promoting the chances of a child node being true, while the roundhead edge shows the parent node inhibiting the chances of a child node being true. The text associated with the non-root nodes represent the corresponding conditional probability values obtained from the CAST logic parameters (not shown in the figure) while the text associated with the root nodes represents the prior probabilities. The probability propagation in an IN is based on the "independence of parents" assumptions (similar to the loopy belief propagation) Thus, the marginal probability of a non-root node is computed with the help of its conditional probability table (CPT) and the prior probabilities of its parents. For instance, the marginal probability of variable A is computed as

$$\begin{aligned} P(A) &= P(A|\neg B, \neg E)P(\neg B)P(\neg E) \\ &\quad + P(A|\neg B, E)P(\neg B)P(E) \\ &\quad + P(A|B, \neg E)P(B)P(\neg E) \\ &\quad + P(A|B, E)P(B)P(E) \\ &= 0.005 \times 0.95 \times 0.99 \\ &\quad + 0.95 \times 0.95 \times 0.01 \\ &\quad + 0.95 \times 0.05 \times 0.99 \\ &\quad + 0.99 \times 0.05 \times 0.01 \\ &= 0.06 \end{aligned} \quad (1)$$

The probability of $D$ is then computed by using its CPT and the marginal probabilities of $A$ (computed above) and $E$.

$$\begin{aligned} P(D) &= P(D|\neg E, \neg A)P(\neg E)P(\neg A) \\ &\quad + P(D|\neg E, A)P(\neg E)P(A) \\ &\quad + P(D|E, \neg A)P(E)P(\neg A) \\ &\quad + P(D|E, A)P(E)P(A) \\ &= 0.05 \times 0.99 \times 0.94 \\ &\quad + 0.95 \times 0.99 \times 0.06 \\ &\quad + 0.001 \times 0.01 \times 0.94 \\ &\quad + 0.05 \times 0.01 \times 0.06 \\ &= 0.11 \end{aligned} \quad (2)$$

In this way, marginal probabilities are propagated in the forward direction, i.e., from the root nodes to the leaf nodes. Once an IN is specified, a system analyst is typically interested in identifying a set of actions that maximizes the probability of a desired effect. The following sections discuss approaches to solve this problem.

## 3. Sensitivity analysis

The sensitivity analysis (SA) looks at how sensitive an effect is with respect to the actionable events when actions are considered one at a time. The analysis requires linear number of searches, i.e., if there are 'n' actionable events then 'n' iterations are required to perform the analysis. The SA algorithm is given in Table 1 and is explained with the help of the IN shown in Fig. 2. The CPTs associated with non-root nodes are given in Fig. 3. These values are generated from the CAST logic parameters (not shown in Fig. 2) that a subject matter expert has assigned. The prior probabilities of all the actionable events are set to 0.1.

SA works in the following manner. Each node among the set of actionable nodes $\{A, B, \ldots, H\}$ is selected during an iterative process. Its probability is first set to zero (Step 1 Table 1) and then to one (Step 3) while keeping the probabilities of other actionable events at their initial prior values. The sensitivity of

Table 1
Sensitivity analysis algorithm

Given *A*, *E*
where *A* = Set of Actionable Events
*E* = Desired Effect
Iterate ∀ I where I ∈ *A*

1. Set P(I) = 0.
2. Computé P(E). Set P(E0) = P(E)
3. Set P(I) = 1.
4. Computé P(E). Set P(E1) = P(E).
5. Set Diff(I) = P(E1) − P(E0).
6. Restore the marginal probability of I.
7. Report Diff(I).

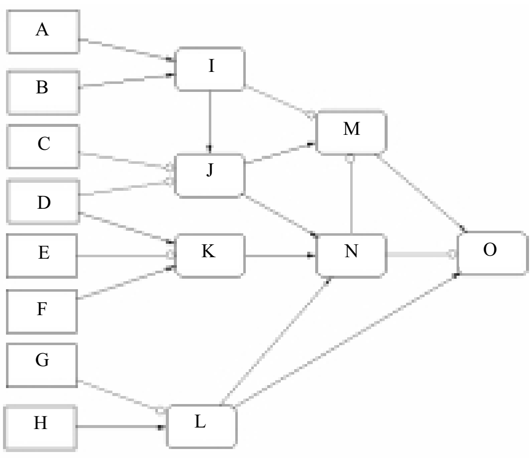


Fig. 2. An Influence Net.

the effect with respect to the selected action is determined by computing the probabilities of the desired effect (Steps 2 and 4) when the action's probability is set to zero and one, respectively. At the end of this iterative process, the events which cause the highest difference in the likelihood of the occurrence of the desired effect are considered for further analysis.

Table 2 shows the results of sensitivity analysis on the IN of Fig. 2. The first column contains the actionable events, while the second column shows the probability of the desired effect when the corresponding action's prior probability is set to zero while keeping the prior probabilities of all other events to their initial values (0.1 in this example). The likelihood of the desired effect is given in the third column when the probability of the corresponding action is set to one. The last column shows the difference between the values in column 3 and column 2. The events having higher absolute values in column 4 are assumed to have greatest individual impact on the desired effect. For instance, if the decision maker is interested in maximizing the probability of 'O', then events C, E, and H would be considered

Table 2
Sensitivity analysis of IN of Fig. 2

| Actions | P(O) when P(Action) = 0 | P(O) when P(Action) = 1 | Difference |
|---|---|---|---|
| A | 0.609503 | 0.483396 | −0.12611 |
| B | 0.601453 | 0.560614 | −0.04084 |
| C | 0.596521 | 0.651012 | 0.05449 |
| D | 0.606377 | 0.491516 | −0.11486 |
| E | 0.594949 | 0.619665 | 0.02472 |
| F | 0.607959 | 0.502733 | −0.10523 |
| G | 0.606466 | 0.578399 | −0.02807 |
| H | 0.589451 | 0.703644 | 0.11419 |

for further analysis as they cause an increase in the probability of 'O'. The other actions cause a decrease in the marginal probability of 'O'. As we will see in the next section, this statement is not accurate. In fact, the combination of events C, D, E, and H causes the highest increase in the probability of 'O'. The reason for this discrepancy is that there are nonlinearities involved in an IN, thus any decision based on the individual impacts of actions may not hold true when actions are considered together.

## 4. The SAF algorithm

This paper presents a heuristic approach, named as Sets of Actions Finder (SAF), for determining the sets of actions that cause the probability of the desired effect to be above (below) a given probability threshold. The algorithm achieves this task in significantly less time than what is required for an exhaustive examination of the actions' search space, which is exponential in terms of the number of actions. The proposed algorithm uses a greedy approach to identify the best (or close-to-best) sets of actions. The algorithm starts with a single action which, when considered individually, causes the highest increase (for a maximization problem) in the probability of the desired effect being true. This is followed by the selection of a second action from the remaining set of actions that together with the first action cause the highest increase in the probability of the desired effect. Other actions are added iteratively in a similar manner. The process stops at a point where (i) the inclusion of an action causes the probability of the objective node to decrease and to fall below the given probability threshold or (ii) there are no more actions to add. Once alternative sets of actions are obtained, they can be grouped together to form more general sets of actions.

The description of the algorithm is given in Table 3 and is explained with the help of the IN of Fig. 2. Event

| Parents | P(I/Parents) |
|---|---|
| ¬A,¬B | 0.005 |
| ¬A, B | 0.147 |
| A,¬B | 0.500 |
| A, B | 0.983 |

| Parents | P(J/Parents) |
|---|---|
| ¬C,¬I,¬D | 0.950 |
| ¬C,¬I, D | 0.170 |
| ¬C, I,¬D | 0.998 |
| ¬C, I, D | 0.950 |
| C,¬I,¬D | 0.170 |
| C,¬I, D | 0.006 |
| C, I,¬D | 0.950 |
| C, I, D | 0.170 |

| Parents | P(K/Parents) |
|---|---|
| ¬D,¬E,¬F | 0.015 |
| ¬D,¬E, F | 0.433 |
| ¬D, E,¬F | 0.000 |
| ¬D, E, F | 0.015 |
| D,¬E,¬F | 0.433 |
| D,¬E, F | 0.980 |
| D, E,¬F | 0.015 |
| D, E, F | 0.433 |

| Parents | P(L/Parents) |
|---|---|
| ¬G,¬H | 0.500 |
| ¬G, H | 0.983 |
| G,¬H | 0.017 |
| G, H | 0.500 |

| Parents | P(M/Parents) |
|---|---|
| ¬I,¬J,¬N | 0.830 |
| ¬I,¬J, N | 0.050 |
| ¬I, J,¬N | 0.994 |
| ¬I, J, N | 0.830 |
| I,¬J,¬N | 0.050 |
| I,¬J, N | 0.002 |
| I, J,¬N | 0.830 |
| I, J, N | 0.050 |

| Parents | P(N/Parents) |
|---|---|
| ¬J,¬K,¬L | 0.006 |
| ¬J,¬K, L | 0.170 |
| ¬J, K,¬L | 0.567 |
| ¬J, K, L | 0.985 |
| J,¬K,¬L | 0.170 |
| J,¬K, L | 0.950 |
| J, K,¬L | 0.985 |
| J, K, L | 0.999 |

| Parents | P(O/Parents) |
|---|---|
| ¬M,¬N,¬L | 0.015 |
| ¬M,¬N, L | 0.830 |
| ¬M, N,¬L | 0.000 |
| ¬M, N, L | 0.050 |
| M,¬N,¬L | 0.830 |
| M,¬N, L | 0.998 |
| M, N,¬L | 0.050 |
| M, N, L | 0.950 |

Fig. 3. Conditional probability values associated with the IN of Fig. 2.

Table 3
The SAF algorithm

Given *A, E, S, t*
where *A* = Set of Actions
*S* = Set of Selected Actions
*E* = Desired Effect
*t* = Threshold

1. Initialize *S* = null.
2. ∀ x Set P(x) = 0, where x ∈ *A*.
3. Compute P(E). Set P'(E) = P(E).
4. Set P(I) = 1 where I ∈ *A*
   a. ∀ J Set P(J) = 0 where J ∈ *A* \ {I}
   b. Compute P(E).
   c. Set Diff(I) = P (E) – P'(E).

Iterate for all members of A.

1. Select the highest Diff(I) obtained in Step 4.
   a. If Diff(I) > 0 OR if the corresponding P(E) is greater than the threshold t
      i. Remove I from *A*.
      ii. Insert I into *S*.
      iii. Set P'(E) = P(E).
      iv. Go to Step 4.
   b. Else Stop.

O represents the desired effect and it is of interest to identify a set which maximizes the probability of O, i.e., P(O). Besides identifying a set that maximizes P(O), suppose a system analyst is also interested in identifying other sets that satisfy a given threshold. For instance, in the given situation, the analyst is interested in identifying a few of those sets that cause the probability of O to be greater than or equal to 0.8.

At the start of the algorithm (Table 3), the set of the selected actionable events is set to null. At Step 2, the prior probabilities of all the actionable nodes are set to 0. The marginal probability of 'O', P(O), is then computed with this combination in Step 3 and is found to be 0.63. In an incremental fashion in Step 4, each action is set to true (the prior probability is set to 1) while keeping the state of the remaining actions to be false. An increase or decrease in the probability of node 'O' is recorded. For instance, when P(A) is set to 1 while the prior probabilities of the remaining actions are set to 0, the P(O) is computed as 0.51. Thus, action A decreases the marginal probability of 'O' by 0.12 (0.63–0.51). Similarly, when B is set to true, while the other actions are set to false, the P(O) is computed as 0.60. Thus, B causes a decrease of 0.03 in the marginal probability of O. The remaining computations are shown in the first column of Table 4.

It can be seen from the first column of the table that node H causes the highest increase in the probability of node 'O', making it 0.78. As a result, H is included in the set of selected actions, as described in the Step 5(a) of the algorithm. Step 4 of the algorithm is then executed again, but this time with the probability of H occurring set to 1. The process is shown in the second column of Table 4. For instance, keeping H as true, the marginal probability of A is set to 1 while the marginal probabilities of the remaining nodes are set to 0. The probability of O is then computed and is found to be 0.50. Thus, there is a decrease of 0.28 (0.78–0.50), if A is taken together with H as compared to when H is taken alone. The same process is repeated for the other actionable events. During this iteration, action C causes

Table 4
Working of SAF algorithm

| All Actions are False P(O) = 0.63 | H is True, Other Actions are False P(O) = 0.78 | H,C are True, Other Actions are False P(O) = 0.84 | H,C,E are True, Other Actions are False P(O) = 0.84 | H,C,E,D are True, Other Actions are False P(O) = 0.89 | H,C,E,D,B are True, Other Actions are False P(O) = 0.85 |
|---|---|---|---|---|---|
| A: 0.51 (−0.12) | A: 0.50 (−0.28) | A: 0.62 (−0.22) | A: 0.62 (−0.22) | A: 0.76 (−0.13) | A: 0.64 (−0.21) |
| B: 0.60 (−0.03) | B: 0.70 (−0.08) | B: 0.77 (−0.07) | B: 0.78 (−0.06) | **B: 0.85** (**−0.04**) | F: 0.62 (−0.23) |
| C: 0.71 (0.08) | **C: 0.84** (**0.06**) | D: 0.65 (−0.19) | **D: 0.89** (**0.05**) | F: 0.65 (−0.24) | G: 0.70 (−0.15) |
| D: 0.51 (−0.12) | D: 0.63 (−0.15) | **E: 0.84** (≈ **0.0**) | F: 0.84 (≈ 0.0) | G: 0.75 (−0.14) | |
| E: 0.63 (≈ 0.0) | E: 0.78 (≈ 0.0) | F: 0.63 (−0.01) | G: 0.72 (−0.12) | | |
| F: 0.54 (−0.09) | F: 0.77 (−0.01) | G: 0.71 (−0.13) | | | |
| G: 0.66 (0.03) | G: 0.63 (−0.15) | | | | |
| **H: 0.78** (**0.15**) | | | | | |

the highest increase in the marginal probability of O. As a consequence, C is included in the set of selected actions along with H as described in Step 5(a). The important thing to note is that a solution has already been found because the probability of node ‘O’ has met the given threshold of 0.8. In other words, when both H and C are true, while other actions are false, there is an 84% chance of achieving the desired effect. The iterative process of Step 4 is repeated again for the rest of the nodes. Columns 3 to 6 show the results of these iterations. Column 5 contains an interesting situation that requires explanation. The iteration shown in Column 5 starts with actions H, C, E, and D set permanently to true while the marginal probability of node O is 0.89. It can be observed that none of the actions among A, B, F, and G further increases the probability of node O. The interesting case is that of node B. Despite the fact that it decreases the probability of node ‘O’, the resultant probability (0.85) is still above the given threshold. Hence the combination where H, C, E, D, and B are true while A, F, and G are false also constitutes a feasible solution. Thus B is included in the set of selected actions along with H, C, E, and D. Column 6 shows the results of incrementally setting the rest of the actionable variables as true and computing the probability of O for each combination. There is not a single combination that either causes an increase in the probability of ‘O’ or keeps the resultant probability of ‘O’ above the given threshold. The algorithm is terminated at this point (Step 5(b)). During the iterative process of the SAF algorithm, five feasible solutions are produced. Each solution represents a rule having binary predicates. These rules are shown in Table 5. Rule 1 (R1) for instance says that the desired effect could be achieved with a certain likelihood, if actions C and H are in state true, while actions A, B, D, E, F, and G are in state false. Other rules can be read in a similar manner.

Table 5
Solutions set found by SAF

| | |
|---|---|
| R1: | ¬A, ¬B, C, ¬D, ¬E, ¬F, ¬G, H |
| R2: | ¬A, ¬B, C, ¬D, E, ¬F, ¬G, H |
| R3: | ¬A, ¬B, C, D, E, ¬F, ¬G, H |
| R4: | ¬A, B, C, D, E, ¬F, ¬G, H |
| R5: | ¬A, ¬B, C, ¬D, E, F, ¬G, H |

When an exhaustive search is done over the solution space of the model shown in Fig. 2, six feasible solutions are determined. The SAF algorithm is able to find 5 out of those 6 solutions (83%) but at a significantly reduced computational cost. Instead of generating $2^8$ = 256 possible combinations, the algorithm is able to generate 5 solutions by searching only 33 combinations (13%). Thus 83% of the feasible solutions are obtained at only 13% of the computational cost of the exhaustive search. In general, if the algorithm finds a solution, it takes $n(n+1)/2$ iterations at most where n is the number of actions. The drawback, on the other hand, is that the proposed technique is a greedy approach so there would be cases when it fails to find the global optimum. Section 6 presents a detailed analysis of how many times the approach converges to the actual solution, and when it does not, how far is it from the actual solution.

Table 5 shows that there are four actions that keep their states intact in all the rules. A and G are always false while C and H are always true. This suggests that the rules generated by the SAF algorithm can be grouped together to obtain a smaller number of rules. Since all the rules have binary predicates, techniques similar to the Karnaugh-Map (or the Quine-McClusky method) (Dewey (1996) for logic reduction can be used to reduce the number of rules. The reduced rule set is given in Table 6. The symbol ‘?’ in rules R1‘-R4‘ represents indifference to the state of the corresponding actionable event. For instance, the rule R1‘ says that in order to maximize the likelihood of achieving the objective, actions C and H must be true and A, B, D, F,

Table 6
Generalized solutions set

| | |
|---|---|
| R1‘: | ¬A, ¬B, C, ¬D, ?, ¬F, ¬G, H |
| R2‘: | ¬A, ¬B, C, ?, E, ¬F, ¬G, H |
| R3‘: | ¬A, ¬B, C, D, E, ?, ¬G, H |
| R4‘: | ¬A, ?, C, D, E, ¬F, ¬G, H |

and G must be false while the state of E does not really matter.

## 5. MINLP formulation

This section demonstrates how the problem of best set of actions determination can be formulated as an instance of Mixed Integer Non Linear Programming. The non linear constraints are due to the conditional probabilities associated with non-root nodes in an Influence Net while the integer constraints are due to the binary state actionable events. It is generally believed that "MINLP is still a new field, and not all the problems that naturally fall within this area can be solved".[1]

Consider again the IN of Fig. 1. If the prior probabilities of B and E are unknown and the objective is to find the marginal probabilities P(B) and P(E) that maximize the probability of D, P(D), then this problem can be formulated as a mixed integer non linear programming problem. Equations (1) and (2) can be rewritten as active non linear constraints.

$$\begin{aligned} &P(A) - P(A|\neg B,\neg E)P(\neg B)P(\neg E) \\ &+P(A|\neg B,E)P(\neg B)P(E) \\ &+P(A|B,\neg E)P(B)P(\neg E) \\ &+P(A|B,E)P(B)P(E) = 0 \end{aligned} \tag{3}$$

$$\begin{aligned} &P(D) - P(D|\neg E,\neg A)P(\neg E)P(\neg A) \\ &+P(D|\neg E,A)P(\neg E)P(A) \\ &+P(D|E,\neg A)P(E)P(\neg A) \\ &+P(D|E,A)P(E)P(A) = 0 \end{aligned} \tag{4}$$

Events B and E can have probability of either one or zero. Thus, these probabilities can be written as integer constraints.

$$P(B) \in \{0,1\} \tag{5}$$

$$P(E) \in \{0,1\} \tag{6}$$

In this example, we are interested in maximizing the probability of D. Thus, the objective function becomes:

$$\max P(D) \tag{7}$$

[1] The statement is taken from the MINLP World website: http://www.gamsworld.org/minlp/ (Date: 08/10/07).

Table 7
Steps required in MINLP formulation of an Influence Net

| |
|---|
| Given **A**, **E**, **X** |
| Where **A** = Set of Actions |
| **E** = Desired Effect |
| **X** = Set of events excluding actions |
| 1. The marginal probabilities of **X** become active non linear constraints. |
| 2. The marginal probabilities of **A** become integral constraints. |
| 3. P(**E**) becomes the objective function. |

The above steps are presented in Table 7 in the form of a transformation algorithm. In this research, we have formulated the problems using AMPL, which is one of the most popular modeling languages among the OR (Operations Research) community. It is described as a comprehensive and powerful algebraic modeling language for linear and nonlinear optimization problems, in discrete or continuous variables.[2]

Once a problem is formulated using AMPL, it can be solved using an algorithm developed for mixed integer nonlinearly constrained optimization problems. This paper uses a particular implementation of such algorithms named MINLP. It implements a branch-and-bound algorithm searching a tree whose nodes correspond to continuous nonlinearly constrained optimization problems. The continuous problems are solved using filterSQP, a Sequential Quadratic Programming solver which is suitable for solving large nonlinearly constrained problems.[3]

When applied on the IN of Fig. 2, the MINLP produces the same optimal solution as obtained through SAF and ES, i.e., {H, C, E, D}. Besides providing an alternative way of accomplishing the task of best set of actions determination, the results obtained through MINLP can be used to compare the performance of the SAF algorithm. The comparison is presented in the following section. It is worth noting that the SAF algorithm not only provides a single best solution, but it also provides a set of solutions that are close enough in terms of making the probability of a desired effect above/below the specified threshold.

## 6. Comparisons of different approaches

This section applies SAF, SA, and MINLP algorithms on eight large and realistic INs as compared to

[2] http://www.ampl.com/.
[3] http://www-neos.mcs.anl.gov/neos/solvers/MINCO:MINLP-AMPL/.

Table 8
Description of INs used in the experiment

| Model Name | # of Nodes | # of Links | Ratio (# of Links to # of Nodes) | # of Actionable Events |
|---|---|---|---|---|
| M1 | 40 | 49 | 1.23 | 11 |
| M2 | 21 | 39 | 1.86 | 10 |
| M3 | 35 | 74 | 2.11 | 11 |
| M4 | 39 | 98 | 2.51 | 5 |
| M5 | 30 | 45 | 1.50 | 11 |
| M6 | 25 | 42 | 1.68 | 11 |
| M7 | 23 | 41 | 1.78 | 5 |
| M8 | 26 | 36 | 1.38 | 11 |

the illustrative examples used so far. Each model is instantiated 1000 times (with different CAST logic parameters) to have a sufficiently diverse pool of models. The objective is to compare the performance of these algorithms on different types of models and to build an approximate error bound on the solutions produced by them. Table 8 describes some of the important characteristics of the eight models[4] used in the experiments. These include number of nodes, links, actionable events, and the ratio of number of links to number of nodes. For instance model M1 has 11 actionable events. Thus the exhaustive search needs to run $2^{11}$ times to determine the set of actions that maximizes the probability of the effect node. The graphical representation of these models can be found in Haider [12]. There are two models in Table 8 which have only five actionable events. Thus there are only 32 possible combinations of actions which can easily be explored in real time by an exhaustive search. The reason for including these models in the experiment is to analyze whether the number of actions is the only important factor affecting the performance of the tested approaches, or the performance is also dependent upon the structure of an IN. Model M4 is of particular interest because it is very strongly connected (having a links to nodes ratio of 2.5).

During the experiment, each model is instantiated 1000 times. In each run, an IN is initialized by randomly assigned CAST logic parameters (h and g values). The baseline probabilities are set to 0.5. All the models have one effect node and we are interested in identifying the set of actions which maximizes the probability of the effect node. Thus, there are 8 structures (and 1000 unique set of parameters for each structure) used in this experiment. SAF, MINLP, and SA are applied on these 8000 INs. The optimal values produced by each of these approaches are compared with the global optimum obtained from the exhaustive search. The number of times a particular approach fails to produce a global optimum is recorded. There were cases when the difference between the optimal values produced by a particular approach and the exhaustive search is less than 0.00005. In such cases, it is assumed that the approach has achieved the global optimum. For instance, suppose that in an IN there are eight actionable events (from A to H). The best sets of actions and the corresponding probabilities of achieving a desired effect as obtained from the exhaustive search and MINLP are as follows (1 represents true, 0 false):

Actions : $A\ B\ C\ D\ E\ F\ G\ H\ P$(Effect)

ES : $1\ 0\ 1\ 1\ 0\ \mathbf{1}\ 1\ 0\ 0.9421582$

MINLP $1\ 0\ 1\ 1\ 0\ \mathbf{0}\ 1\ 0\ 0.9421564$

The two sets are different (F has different values in two sets), but the difference in the probabilities of achieving the desired effect is almost negligible (0.0000018). When such situations arise in the experiment, it is assumed that MINLP has reached the global maximum. The reason for taking this assumption is that different algorithms are implemented in different programming languages. Thus, the final solution produced by each of them may depend upon how a particular programming language rounds off a real number. Secondly, the kind of problem being considered does not require accuracy of up to five significant digits. Thus, the assumption should not have any impact on the analysis that follows.

Table 9 shows the percentage of inferior solutions produced by a particular approach. For instance, SA fails 69% of the times when it is applied on models having structure similar to M2 ($2^{nd}$ row of the table) while SAF fails 17% of the time. The other entries in the table can be read in a similar manner. The last row shows the overall percentage of inferior solutions produced by particular approaches when tested on 8000 different Influence Nets. SAF has the lowest error rate

[4]The authors are very grateful to Dr. Lee Wagenhals and Mr. Raymond Janssen for providing these models.

Table 9
Percentage of inferior solutions produced by different approaches

| Models | MINLP | SAF | SA |
|---|---|---|---|
| M1 | 2.4 | 0.8 | 26.0 |
| M2 | 15.4 | 17 | 69.4 |
| M3 | 11.7 | 5.8 | 46.7 |
| M4 | 6.9 | 5.4 | 34.0 |
| M5 | 4.6 | 3.0 | 25.7 |
| M6 | 6.8 | 5.7 | 37.6 |
| M7 | 4.9 | 5.2 | 40.1 |
| M8 | 14.0 | 10.0 | 59.8 |
| Mean | 8.34 | 6.61 | 42.41 |

Table 10
Mean difference between global and local optima

| Models | MINLP | SAF | SA |
|---|---|---|---|
| M1 | 0.0056 | 0.0164 | 0.0130 |
| M2 | 0.0315 | 0.0489 | 0.0616 |
| M3 | 0.0205 | 0.0370 | 0.0272 |
| M4 | 0.0107 | 0.0053 | 0.0099 |
| M5 | 0.0234 | 0.0151 | 0.0220 |
| M6 | 0.0147 | 0.0241 | 0.0300 |
| M7 | 0.0142 | 0.0157 | 0.0173 |
| M8 | 0.0230 | 0.0284 | 0.0336 |
| Mean | 0.01795 | 0.023863 | 0.026825 |

of 6.6% while SA has the highest error rate of 42.4%. MINLP is not very far from SAF as its error rate is 8.3%.

Table 10 shows the average difference between the global optima (produced by the exhaustive search) and the local optima produced by different approaches (when they fail to converge to global optima). For instance, when MINLP fails in models having structure M1, on average there is a difference of 0.0056 between the global optimum and its solution. Similarly, when the SAF algorithm fails to reach the global maximum in models having structure similar to M6, the average difference between its solution and the global maximum is 0.0241. The last row of the table shows the overall differences of corresponding approaches when tested on 8000 models. Surprisingly there is not a significant difference between the mean difference values. Even though SA fails to produce global maxima 42% of the time (as shown in Table 9), it has a mean difference of 0.0268. This small mean difference suggests that even though there are nonlinearities in the conditional probability values generated by the CAST logic parameters, these nonlinearities are not very strong. MINLP and SAF have mean differences of 0.0180 and 0.0239, respectively. Thus, MINLP has the lowest mean difference between its optima and global optima, but for practical purposes all approaches have similar mean difference.

Table 11 presents the $95^{th}$ percentile and the maximum difference produced by different approaches. The columns for maximum difference show the most inferior solution produced by a particular approach for a certain class of models. For instance, in 1000 instances of model M3, the most inferior result produced by SAF has a difference of 0.305 ($3^{rd}$ row and $4^{th}$ columns) from the optimal value; while 95% of the time the inferior solutions are within a distance of 0.1422 ($3^{rd}$ row and $3^{rd}$ column) to the global optimum. The last row of the table shows the mean $95^{th}$ percentile. Thus, reading Tables 9–11 together, it can be said that SAF produces global optima 93.4% of the time (Table 9). Out of 6.6% of the time when it does not produce a global optimum, 95% of the time the difference is less than 0.0734 (Table 11). The mean error rate in those 6.6% cases is 0.0239 (Table 10). It can be seen from the columns having heading "Max." that at times SA produces very inferior solutions. For instance, it produces inferior solutions having difference, from corresponding global maxima, of 0.53, 0.30, and 0.48 for model types M2, M3, and M8, respectively.

It is worth mentioning that the CAST logic subsumes the Noisy-Or modeling techniques for Bayesian Networks. Thus, if a BN is modeled using Noisy-Or parameters then any of the techniques presented in this paper (either SA or SAF or MINLP) are guaranteed to produce global optima. This is due to the fact that individual impacts in BN2O have always synergistic effect when considered collectively. However, the application of SA, SAF, and MINLP on a general class of BNs, where all possible types of interactions among parent nodes are possible, is not discussed in this paper and is the focus of the future research work.

## 7. Conclusions

The paper presented approaches to solve the problem of best set of actions determination in uncertain situations modeled using Influence Nets. A heuristic approach is developed that attempts to overcome the limitations of existing approaches, such as sensitivity analysis and exhaustive search, by considering the combined impact of actionable events on a desired effect in polynomial number of searches. The approach not only provides the best set of actions, but it also provides alternative sets of actions.

The paper also showed that the problem of finding best set of actions can be formulated as an instance of Mixed Integer Non Linear Programming. The formu-

Table 11
$95^{th}$ percentile and the maximum difference in the optimal values

| Models | MINLP | | SAF | | SA | |
|---|---|---|---|---|---|---|
| | $95^{th}$ Per. | Max. | $95^{th}$ Per. | Max. | $95^{th}$ Per. | Max. |
| M1 | 0.0141 | 0.0203 | 0.0197 | 0.0750 | 0.0548 | 0.2111 |
| M2 | 0.1212 | 0.2427 | 0.1740 | 0.3874 | 0.2167 | 0.5334 |
| M3 | 0.0780 | 0.3051 | 0.1422 | 0.3050 | 0.0964 | 0.3036 |
| M4 | 0.0489 | 0.0610 | 0.0132 | 0.0589 | 0.0422 | 0.0908 |
| M5 | 0.0640 | 0.1109 | 0.0487 | 0.0799 | 0.0771 | 0.1931 |
| M6 | 0.0438 | 0.1224 | 0.0826 | 0.1511 | 0.1039 | 0.2667 |
| M7 | 0.0597 | 0.1296 | 0.0914 | 0.1187 | 0.0620 | 0.1533 |
| M8 | 0.0900 | 0.2096 | 0.0157 | 0.1917 | 0.1334 | 0.4834 |
| Mean | 0.0650 | | 0.0734 | | 0.0983 | |

lation not only provides an alternative way of solving the problem but it can also be used as a benchmark to SAF or other algorithms that would be developed in the future. An empirical study is performed over thousands of Influence Net (INs) models to analyze the effectiveness of these approaches over different classes of INs. It was shown that SAF performance over the experimental models was almost similar to that of MINLP, with the additional advantage that instead of producing a single set of actions, SAF produces several sets of action that are above a pre-define threshold. These multiple solutions may be very useful during the decision making stage as it gives more flexibility and options to a decision maker.

## Acknowledgements

This research was sponsored by the Office of Naval Research under Grant No. N00014-06-1-0081 and the Air Force Research Laboratory Information Directorate under Grant No. FA8750-05-C-0145.

## References

[1] J.M. Agosta, Conditional Inter-Causally Independent Node Distributions, a Property of Noisy-OR Models, in *Proceedings of 7th Conference on Uncertainty in Artificial Intelligence,* 1991.

[2] J.E. Cano, L.D. Hern'andez and S. Moral, Importance Sampling Algorithms for the Propagation of Probabilities in Belief Networks, *International Journal of Approximate Reasoning* **15** (1996), 77–92.

[3] K.C. Chang, P.E. Lehner, A.H. Levis, S.A.K. Zaidi and X. Zhao, On Causal Influence Logic, Technical Report, Center of Excellence for C3I, George Mason University, 1994.

[4] J. Cheng, *Efficient Stochastic Sampling Algorithms for Bayesian Networks*, PhD Dissertation, University of Pittsburgh, 2001.

[5] J. Cheng and M.J. Druzdzel, *AIS-BN: An Adaptive Importance Sampling Algorithm for Evidential Reasoning in Large Bayesian Networks*, *Journal of Artificial Intelligence Research* **13** (2000), 155–188.

[6] G.F. Cooper, The Computation Complexity of Probabilistic Inference Using Bayesian Belief Networks, *Artificial Intelligence* **42**(2–3) (1990), 393–405.

[7] A. Dewey, *Analysis and Design of Digital Systems with VHDL,* PWS Publishing Company, 1996.

[8] M.J. Druzdzel and M. Henrion, Intercausal Reasoning with Uninstantiated Ancestor Nodes, in *Proceedings of the $9^{th}$ Conference on Uncertainty in Artificial Intelligence,* 1993, 317–325.

[9] W.T. Freeman and Y. Weiss, On the Optimality of Solutions of the Max-Product Belief Propagation Algorithm in Arbitrary Graphs, *IEEE Transaction on Information Theory* **47** (2001).

[10] R. Fung and K.C. Chang, Weighting and Integrating Evidence for Stochastic Simulation in Bayesian Networks, in: *Uncertainty in Artificial Intelligence,* Vol. 5, 1990.

[11] R. Fung and B.D. Favero, Backward simulation in Bayesian networks, in *Proceedings of the 10th Conference on Uncertainty in Artificial Intelligence,* San Francisco CA, 1994, 227–234.

[12] S. Haider, *On Finding Effective Courses of Action in Dynamic Uncertain Situations*, PhD Dissertation, School of IT&E, George Mason University, Fairfax, VA, 2005.

[13] S. Haider and A.H. Levis, An Approximate Technique for Belief Revision in Timed Influence Nets, in *Proceedings of Command and Control Research and Technology Symposium,* San Diego, 2004.

[14] S. Haider and A.K. Zaidi, Transforming Timed Influence Nets into Time Sliced Bayesian Networks, in *Proceedings of Command and Control Research and Technology Symposium,* San Diego, 2004.

[15] S. Haider, A.K. Zaidi and A.H. Levis, A Heuristic Approach for Best Set of Actions Determination in Influence Nets, in *Proceedings of IEEE International Conference on Information Reuse and Integration,* Las Vegas, 2004.

[16] M. Henrion, Propagating Uncertainty in Bayesian Networks by Probabilistic Logic Sampling, in: *Uncertainty in Artificial Intelligence,* (vol. 2), J.F. Lemmer and L.N. Kanal, eds, 1988, pp. 317–324.

[17] F.R. Kschischang and B.J. Frey, Iterative Decoding of Compound Codes by Probability Propagation in Graphical Models, *IEEE Journal on Selected Areas in Communication* **16**(2) (1998), 219–230.

[18] K. Murphy, Y. Weiss and M. Jordan, Loopy-belief Propagation for Approximate Inference: An Empirical Study, in *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence,* 1999.

[19] R.E. Neapolitan, *Learning Bayesian Networks,* Prentice Hall, 2003.

[20] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Network of Plausible Inference*, Morgan Kaufmann, 1987.

[21] J.A. Rosen and W.L. Smith, Influence Net Modeling with Causal Strengths: An Evolutionary Approach, in *Proceedings of the Command and Control Research and Technology Symposium*, Naval Post Graduate School, Monterey CA, 1996.

[22] J.A. Rosen and W.L. Smith, Influence Net Modeling for Strategic Planning: A Structured Approach for Information Operations, *Phalanx* **33**(4) (2000).

[23] R.D. Shachter and M. Poet, Simulation Approaches Weighting and Integrating Evidence for Stochastic Simulation in Bayesian Networks, in: *Uncertainty in Artificial Intelligence,* (vol. 5), M. Henrion, R.D. Shachter, L.N. Kanal and J.F. Lemmer, eds, 1990.

[24] L.W. Wagenhals and A.H. Levis, Course of Action Development and Evaluation, in *Proceedings of the 2000 Command and Control Research and Technology Symposium*, 2000.

[25] L.W. Wagenhals and A.H. Levis, Modeling Effects Based Operations in Support of War Games, in *Proceedings of 15th International Symposium on Aerospace / Defense Sensing,* Internal Society for Optical Engineering, 2001.

[26] L.W. Wagenhals and A.H. Levis, Modeling Support of Effect-Based Operations in War Games, in *Proceedings of the 2002 Command and Control Research and Technology Symposium,* 2002.