Proceedings of the 2008 Winter Simulation Conference
S. J. Mason, R. R. Hill, L. Mönch, O. Rose, T. Jefferson, J. W. Fowler eds.

INTEGRATED HUMAN DECISION MAKING MODEL UNDER BELIEF-DESIRE-INTENTION FRAMEWORK FOR CROWD SIMULATION

Seungho Lee
Young-Jun Son

Dept. of Systems and Industrial Engineering
The University of Arizona
Tucson, AZ 85721, USA

ABSTRACT

An integrated Belief-Desire-Intention (BDI) modeling framework is proposed for human decision making and planning, whose sub-modules are based on Bayesian belief network (BBN), Decision-Field-Theory (DFT), and probabilistic depth first search (PDFS) technique. To mimic realistic human behaviors, attributes of the BDI framework are reverse-engineered from the human-in-the-loop experiments conducted in the Cave Automatic Virtual Environment (CAVE). The proposed modeling framework is demonstrated for human's evacuation behaviors under a terrorist bomb attack situation. The simulated environment and agents (human model) conforming to the proposed BDI framework are implemented in AnyLogic® agent-based simulation software, where each agent calls external Netica BBN software to perform its perceptual processing function and Soar software to perform its real-time planning and decision-execution functions. The constructed simulation has been used to test impact of several factors (e.g. demographics of people, number of policemen) on evacuation performance (e.g. average evacuation time, percentage of casualties).

1 INTRODUCTION

Human's decision behaviors have been studied by various research communities such as artificial intelligence, psychology, cognitive science, and decision science (Lee et al. 2008). As the outcomes of those efforts, several models have been developed to mimic human decision behaviors, where Lee et al. (2008) classified them into three major categories including 1) economical approach, 2) psychological approach, and 3) synthetic engineering-based approach. First, models in the economical approach have concrete foundation, mostly based on the assumption that decision makers are rational (Opaluch and Segerson 1989; Gibson et al. 1997). However, one limitation is their inability to represent human cognitive natures. To overcome this limitation, models in the psychological approach (second category) have been proposed (Einhorn 1970; Payne 1982; Busemeyer and Townsend 1993). While they consider human cognitive natures explicitly, they mainly focus on the human behaviors under simplified and controlled laboratory environments. Finally, the synthetic engineering-based approaches employ a number of engineering methodologies and technologies to help reverse-engineer and represent human behaviors in complex and realistic environments (Laird et al. 1987; Newell 1990; Rao and Goergeff 1998; Konar and Chakraborty 2005; Zhao and Son 2007; Rothrock and Yin 2008; Lee et al. 2008). The human decision-making models in this category consist of the proper engineering techniques employed for each sub-module. However, the complexity of such comprehensive models makes it difficult to validate them against the real human decisions. In this paper, we propose a novel comprehensive human decision behavior model, effectively integrating engineering, psychological, and economical models. Another novelty is its ability to represent both the human decision-making and decision-planning functions in the unified framework.

Among the existing synthetic engineering-based models, Soar, Act-R, and Belief-Desire-Intention (BDI) are popular, allowing us to develop a modular and computational human decision model. Soar and Act-R have their theoretical basis on the unified theories of cognition (Newell, 1990), which is an effort to integrate research outcomes from various disciplines on the single human cognition. Thus, Soar and Act-R focus more on the actual mechanism of the brain during the internal information processing, involving tasks such as reasoning, planning, problem-solving, and learning. Consequently, these models become complex and difficult to understand. On the other hand, the BDI paradigm is originally based on the folk psychology, where the core concepts of the paradigm map easily to the language used to describe human reasoning and actions in everyday life (Norling 2004). Because of this straightforward representation, the BDI paradigm can easily map the extracted human knowledge into its framework. For this reason, we have adopted the BDI as a core modeling and integration framework in our research.

In this paper, the proposed human decision model is illustrated for the scenarios of emergency evacuation from a terrorist bombing attack in large cities. An effective crowd management requires accurate prediction of impact of such incidents. Furthermore, involvement of human lives demands high accuracy of such prediction. For these purposes, the high-fidelity simulation is ideal.

2 BDI AND ENABLING TECHNIQUES

BDI is a model of human's reasoning process, where its mental state is characterized by three components: beliefs, desires, and intentions (Rao and Georgeff 1998). Beliefs are information which human has about the circumstance, and may be incomplete or incorrect due to the nature of human's perception. Desires are the states of affairs which human would wish to be brought about. Intentions are desires which human has committed to achieve. Zhao and Son (2007) extended the original BDI model to include detailed sub-modules such as 1) *deliberator*, 2) *real-time planner*, and 3) *decision executor* in the decision-making (intention) module (see Figure 1). This extension was necessary to accommodate both the decision-making and decision-planning functions in the unified framework. In addition, emotional module containing *confidence index* and *instinct index* has been also appended to represent more psychological human natures. While Zhao and Son (2007) focused only on the conceptual extension of the BDI model, this paper discusses actual algorithms and techniques to realize the sub-modules of the extended model. *Perceptual processor* in the belief module translates the information about the environment and human himself into his beliefs. Then, based on the current *beliefs* (short-term memory), human updates his *instinct index*. If the *instinct index* is below a threshold (normal mode), human decides what states of affairs to achieve (*desires*) through the *desire generator*. Human selects one *desire* and generates *intentions* to achieve the *desire* via the *deliberator*. Human then generates alternative *plans* based on the current *beliefs* to the direction of achieving his *intention*. A *plan* is a sequence of actions in a plan. Once an optimal or satisfactory *plan* is identified, *decision executor* in the decision-making module executes the series of tasks specified in the *plan*. On the other hand, if the *instinct index* is above a threshold (instinct mode), *decision executor* executes tasks based on his instincts retrieved from his *beliefs* (long-term memory) without involving planning. In this work, a *confidence index* is a function of the deviation between what is predicted about the environment during the planning stage and the actual environment during the execution stage. If the *confidence index* is above a threshold (confident mode), *decision executor* executes all the tasks in the plan. Otherwise (suspicious mode), re-planning is performed every time before executing a task.

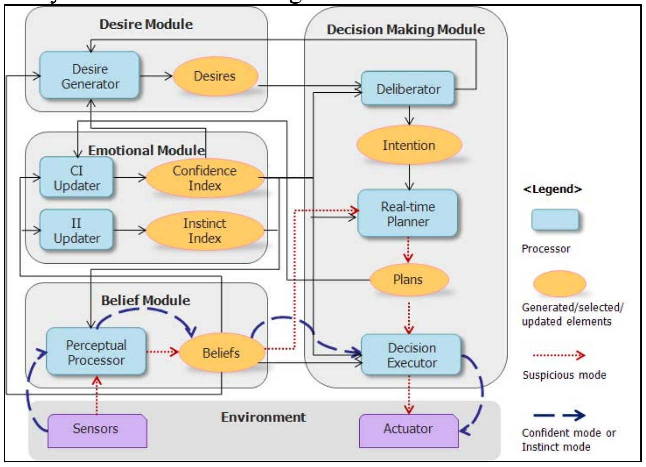


Figure 1: Components of the extended BDI framework

In this paper, we have employed and further developed noble techniques from various disciplines to realize and implement each component of extended BDI, such as Bayesian belief network (BBN), decision field theory (DFT), and probabilistic depth first search (PDFS). Figure 2 depicts a sequence diagram of the overall decision planning and making process, displaying sequential interactions between each component. Whenever an agent needs to plan and makes a decision, it accesses PDFS, DFT, and BBN components to obtain plan/decision, preferences, and environmental evaluation, respectively. Once DFT obtains inferred environmental evaluation from BBN, it gives us the evolved preferences of each option, which will be used to calculate the choice probability of each option. Then PDFS selects an option and makes a plan based on the calculated choice probability. Each component is explained in Section 2.

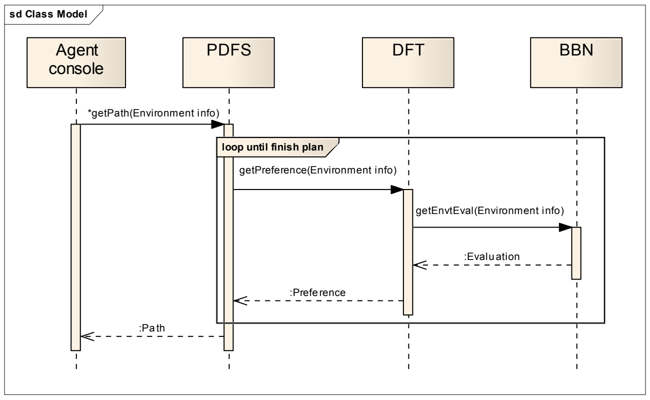

sd Class Model
Agent console
PDFS
DFT
BBN
*getPath(Environment info)
loop until finish plan
getPreference(Environment info)
getEnvtEval(Environment info)
:Evaluation
:Preference
:Path

Figure 2: Sequence diagram of components (corresponding techniques) of the human behavior model

2.1 Application Scenario

As mentioned earlier, the proposed human decision behavior model is illustrated using crowd evacuation behaviors under a terrorist bomb attack in Washington D.C. National Mall area. The map (satellite image) of the area is shown in Figure 9 in Section 4.1. Given the scenario, we have characterized different types of agents based on 1) familiarity with the area, 2) risk taking behavior, 3) confidence index, and 4) guidance by police. Depending on the above characteristics of agents, their evacuation behaviors will be different. After the explosion, the police are informed of it via radio transmission and ask people around them to evacuate from the area.

2.2 BBN for Perceptual Processor

In this research, Bayesian belief network (BBN) is employed to represent the *perceptual processor* (see Figure 1) in the dynamically changing environment. By using BBN, we can capture the probabilistic relationship as well as historical information between variables by containing prior and conditional probabilities to infer the posterior probability through the Bayes' theorem. The major advantage of BBN as a *perceptual processor* is its ability and flexibility to handle uncertain and dynamic environments.

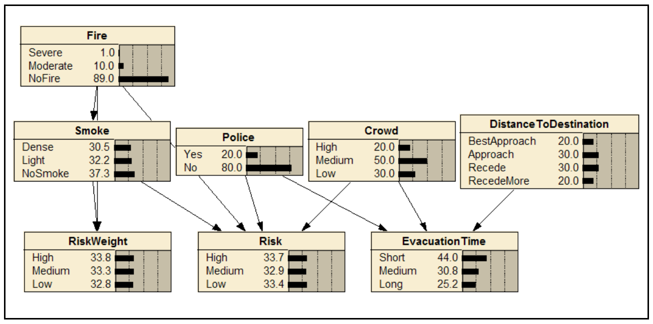


Figure 3: BBN inferring risk, evacuation time, and risk weight for an option

Figure 3 depicts a BBN used to infer the belief of an agent under the evacuation scenario. The beliefs inferred by BBN given environmental information (e.g. smoke, fire, police, crowd, and distance) include 1) evaluation of values for attributes (risk and evacuation time) for the considered option (a path from an intersection) and 2) weights on each attribute. The weights on each attribute at time t, $W(t)=\begin{pmatrix} w_{risk}(t) \\ w_{time}(t) \end{pmatrix}$, is obtained from 'RiskWeight' node of BBN in Figure 3 by defining $w_{risk}(t)$ = 'RiskWeight' and $w_{time}(t)$ = 1- 'RiskWeight'. Similarly, the evaluations of available options on each attribute, $M=\begin{pmatrix} m^1_{risk} & m^1_{time} \\ m^2_{risk} & m^2_{time} \\ \vdots & \vdots \end{pmatrix}$, can be obtained from 'Risk' and 'EvacuationTime' nodes of BBN by assigning m^i_{risk} = 'Risk' and m^i_{time} = 'EvacuationTime' for option i. The inferred belief from BBN is intended to be similar to that of real human. In this research, this similarity can be obtained by constructing BBN based on the data from human-in-the-loop experiments (see Section 4).

2.3 Decision Field Theory (DFT)

Decision field theory (DFT) is a human decision-making model, which is based on psychological rather than economical principles (Busemeyer and Townsend 1993). It provides a mathematical framework to represent the psychological preferences of human on the given choices during his/her deliberation process (Busemeyer and Townsend 1993). In this work, we employ DFT to realize the *real-time planner* sub-module in the decision-making module of the proposed extended BDI (see Figure 1) together with probabilistic depth first search (PDFS) (see Section 2.4). In DFT, the human preference can be described as Equation (1).

$$P(t+h)=SP(t)+CMW(t+h) \tag{1}$$

In Equation (1), $P(t)$ (m-elements vector, where m is the number of options) represents a preference state, where $P_i(t)$ represents a preference value for option i, and h is a time step. The stability matrix S of Equation (1) represents the effect of the preference from the previous state (the memory effect) in the diagonal elements and the effect of the interactions among the options in the off-diagonal elements. For the stability of this linear system, the eigenvalues λ_i of S are assumed to be less than one in magnitude ($|\lambda_i| < 1$). The value matrix M ($m\times n$ matrix, where m is the number of options, and n is the number of attributes) represents the subjective evaluations of a decision-maker for each option on each attribute. For example, given an objective information (e.g. smoke, fire, police, crowd, and distance in the considered scenario), evacuators obtain their own subjective evaluations for each option (e.g. a path from an intersection) on each attribute (e.g. risk, evacuation time), which constitute the M matrix. The weight vector $W(t)$ (n-elements vector) allocates the weights of attention corresponding to each attribute considered at time t. As described in Section 2.2, M and $W(t)$ can be inferred from BBN in our research. The matrix C is the contrast matrix comparing the weighted evaluations of each option, $MW(t)$. Preference increase of one option lowers the preference of alternative options, and the sum of the elements of *CMW*(*t*) is always zero. For example, if we have two options in our evacuation scenario, the corresponding DFT formula, by definition of Equation (1), is following:

$$\begin{pmatrix} p_1(t+h) \\ p_2(t+h) \end{pmatrix} = \begin{pmatrix} s_1 & s_2 \\ s_2 & s_1 \end{pmatrix} \begin{pmatrix} p_1(t) \\ p_2(t) \end{pmatrix} + \begin{pmatrix} 1 & -1 \\ -1 & 1 \end{pmatrix} \begin{pmatrix} m^1_{risk} & m^1_{time} \\ m^2_{risk} & m^2_{time} \end{pmatrix} \begin{pmatrix} w_{risk}(t+h) \\ w_{time}(t+h) \end{pmatrix}.$$

2.4 Probabilistic Depth First Search (PDFS)

In this research, we have employed probabilistic depth first search (PDFS) method to realize *real-time planner* together with DFT in the extended BDI framework (see Figure 1). To this end, we have employed the computer programming aspect of Soar (providing built-in data structures and operators for DFS) to implement the PDFS. Soar searches the problem space with the depth first manner, where a particular branch is selected based on the choice probability of each branch. In order to make series of decisions in a plan, Soar first proposes all the possible options in the current situation and selects one of them as the next task based on their preference values.

As we discussed in Section 2.3, DFT calculates preferences for each option based on the current environment. Then, the choice probability of each option is calculated based on the preference value, which is obtained via multiple replications of DFT evolution. For the binary choice problem, Lee et al. (2008) proved that there is a finite time *t* that the choice probability converges as shown below.

The difference between the expected preference value $E(P(nh))$ and its converging value in DFT becomes less than $\varepsilon > 0$ after n time step where $n = \left\lceil \frac{\log k}{\log D} \right\rceil$, $D = s_{11} - s_{12}$, $E(v_1(h)) = E(w_1(h))(m_{11} - m_{21}) + E(w_2(h))(m_{12} - m_{22})$, and $k = \frac{1-D}{|E(v_1(h))|}\varepsilon$.

Once we obtain a converged choice probability for each option, we feed them into Soar. For example, based on the environmental information (I_{smoke}, I_{fire}, I_{police}, I_{crowd}, and $I_{distance}$ in our scenario of Section 2.1) and the available options (O_{right}, O_{left}, $O_{forward}$, $O_{backward}$ which denote go right, left, forward, and backward, respectively in our scenario of Section 2.1), DFT evolves the preference values for each option until they converge in each of the multiple replications. Let us suppose the numbers of occurrence of each option having the highest preference value are given (F_{right}, F_{left}, $F_{forward}$, $F_{backward}$) based on the converged preferences from the multiple replications ($F = \sum_{i \in Options} F_i$). And suppose further that we select an option with the highest preference value as a final decision in each replication. Then, we can calculate the choice probability of each option ($Pr(O_{right}) = F_{right}/F$, $Pr(O_{left}) = F_{left}/F$, $Pr(O_{forward}) = F_{forward}/F$, $Pr(O_{backward}) = F_{backward}/F$) and feed them into Soar. Then, Soar selects an option based on the given preference (choice probability in this case). The above procedure is repeated for the given number of planning horizon. The number of planning horizon differs for different individuals. For example, the planning horizon for a novice person would be one. However, a commuter may have a plan involving multi-horizon.

2.5 Confidence Index

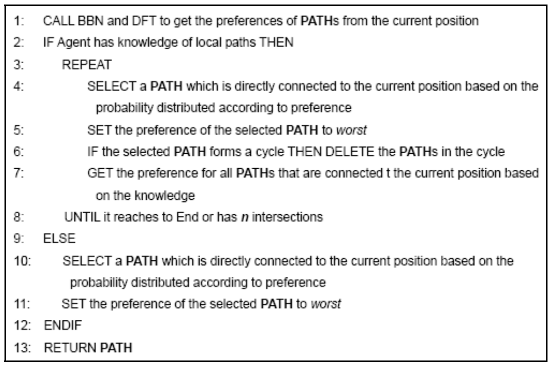
As mentioned before, *confidence index* decides 1) the execution mode (confident or suspicious mode) in the BDI framework and 2) the type of agent (leader or follower). Equation (2) depicts the confidence index.

$$CI_t = \alpha \cdot e^{-d_t} + (1-\alpha)CI_{t-1} \tag{2}$$

where $d_t > 0$, $0 \leq \alpha \leq 1$, and $0 \leq CI_0 = \beta \leq 1$. In Equation (2), d_t denotes the deviation between what is predicted about the environment during the planning stage and the actual environment during the execution stage. Equation (3) depicts d_t used in our research, where m^i_{risk} is the evaluation of risk on the planned option *i*, m^i_{time} is the evaluation of evacuation time on the planned option *i*, and m^{thres}_{risk}, m^{thres}_{time} are the predefined threshold values.

$$d_t = (m^i_{risk} - m^{thres}_{risk}) + (m^i_{time} - m^{thres}_{time}) \tag{3}$$

The parameter α adjusts the effect of previous confidence to the current confidence, which varies depending on the individual human. The initial confidence value (β) is different for each agent. By definition, the range of *confidence index* is between 0 and 1.



```
1:  CALL BBN and DFT to get the preferences of PATHs from the current position
2:  IF Agent has knowledge of local paths THEN
3:      REPEAT
4:          SELECT a PATH which is directly connected to the current position based on the
              probability distributed according to preference
5:          SET the preference of the selected PATH to worst
6:          IF the selected PATH forms a cycle THEN DELETE the PATHs in the cycle
7:          GET the preference for all PATHs that are connected t the current position based
              on the knowledge
8:      UNTIL it reaches to End or has $n$ intersections
9:  ELSE
10:     SELECT a PATH which is directly connected to the current position based on the
          probability distributed according to preference
11:     SET the preference of the selected PATH to worst
12: ENDIF
13: RETURN PATH
```

Figure 4: Planning algorithm

3 REAL-TIME PLANNING ALGORITHM

This section discusses the planning algorithm implemented within Soar in a greater detail. Figure 4 depicts a pseudo code for the proposed planning algorithm for the evacuation application (see Section 2.1), which uses the preference based internal decision operation of Soar. Using this algorithm, an agent (human) develops his evacuation plan (route) dynamically (involving varying number of horizons) until he reaches his destination. The first line of the algorithm in Figure 4 denotes that preferences of paths which are directly accessible from the current position are obtained via BBN and DFT. Then, the algorithm works differently depending on the type of agent.

3.1 Multi-Horizon Planning for Commuter Agent

The commuter agents represent those people who have enough knowledge about the area so that they can plan beyond the current decision point (selecting a path from the current intersection). To illustrate the algorithm for various situations, an exemplary evacuation area (in Washington D.C.) is used (see Figure 5 for its satellite image and its corresponding graph). The graph (G) used here is defined formally as $G = (V, E)$, where V and E represent a set of nodes and edges pertaining to the graph, respectively. The graph (G) in Figure 5 has nodes $V(G) = \{a, b, c, d, e, f, g, h, I, j, k, l, m, n, o\}$ and edges $E(G) = \{ab, bc, be, de, ef, ej, fc, fg, gh, gk, hl, ij, jk, jm, kl, kn, lo, mn\}$.

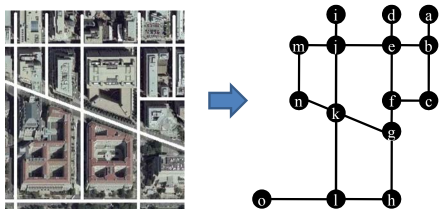
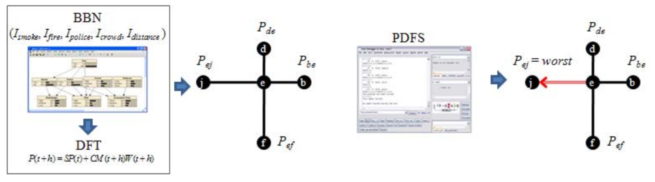


Figure 5: Graph representation of the evacuation area

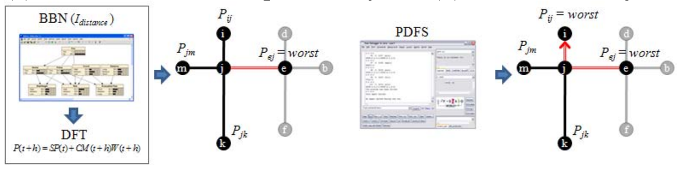
In this example, it is supposed that an agent in node e is searching for a route (series of paths, $R(G)$) to the destination node o. Figure 6 depicts a series of selection process (of a path), which can be described as following:

1. Step 1: At node e (see Figure 6(a)), the agent evaluates each path (be, de, ef, ej) in terms of smoke, fire, police, crowd, and the distance to the destination (distance to node o from nodes b, d, f, j).
2. Step 2: Based on his observation, the agent infers evaluation matrix M, $M = \begin{pmatrix} m_{risk}^{be} & m_{time}^{be} \\ m_{risk}^{de} & m_{time}^{de} \\ m_{risk}^{ef} & m_{time}^{ef} \\ m_{risk}^{ej} & m_{time}^{ej} \end{pmatrix}$, and weight vector $W(t)$, $W(t) = \begin{pmatrix} w_{risk}(t) \\ w_{time}(t) \end{pmatrix}$, via BBN (see Figure 3), where m_{risk}^{be} represents the evaluation of edge be on the risk attribute, and $w_{risk}(t)$ is the weight on the *risk* attribute at time t.
3. Step 3: M and $W(t)$ obtained in Step 2 is provided to Equation (1) (DFT), whose multiple replications generate the choice probabilities P_{be}, P_{de}, P_{ef}, P_{ej} for each path be, de, ef, ej. For each replication, DFT is evolved to reach the convergence of the choice probability (see Section 2.4).
4. Step 4: Now, the choice probabilities are fed into Soar, and Soar selects one edge randomly based on the probabilities (see Figure 6(b)). Suppose edge ej has been selected; then, $R(G)$ is updated to $\{ej\}$. Then, the same process (see Steps 1, 2, 3, and 4) is used to pick a second path from intersection j. The final step of the current iteration before starting the second iteration is to set P_{ej} to *worst* so that path ej (coming back to the intersection e again) will be hardly selected in the second iteration. It is noted that while the second iteration of the planning algorithm starts from intersection j, the agent is still located at intersection e.
5. Step 5: At node j (see Figure 6(c)), the agent repeats Step 1 to evaluate each path (ej, ij, jk, jm). However, this evaluation is only for the updated distance to the destination (distance to node o from nodes e, i, k, m) as other environmental variables (smoke, fire, police, and crowd) are not visible from the current location (intersection e) of the agent. Then, the agent repeats Step 2 to infer evaluation matrix M and weight vector $W(t)$ via BBN (see Figure 3), where BBN uses the updated distance to the destination and expected values for smoke, fire, police, and crowd. Then, the agent repeats Step 3 to obtain the choice probabilities P_{ej}, P_{ij}, P_{jk}, P_{jm}, where the value (*worst*) of P_{ej} (see Step 4) is not updated. Then, the agent repeats Step 4, selecting edge ij and updating $R(G) = \{ej, ij\}$ and $P_{ij} = worst$ (see Figure 6(d)). Note again that the agent is still planning the route without actually moving.
6. Step 6: At node i (see Figure 6(e)), the agent repeats Steps 1, 2, 3, 4. However, as shown in Figure 6(e), the only available path is edge ij, whose P_{ij} was assigned as *worst* in Step 5. In this case, although $P_{ij} = worst$ (involving very small probability instead of zero probability), edge ij is selected and $R(G)$ is updated to $\{ej, ij, ij\}$ (see Figure 6(f)). Then, since edge ij is taken twice, a cycle has been formed and the edges in the cycle are deleted from $R(G)$ (= $\{ej\}$) according to our planning algorithm. Then, Soar selects an edge from intersection j again based on the choice probabilities P_{ej}, P_{ij}, P_{jk}, P_{jm}, where both P_{ej} and P_{ij} are *worst* (see Figure 6(g)). This way, path ij is hardly selected again.
7. Step 7: The agent repeats the above process until it reaches the destination node o or the number of planning horizon is reached (see Figure 6(h)). In other words, the number of elements (edges) in $R(G)$ is same as the number of planning horizon. Then, the multi-horizon planning process is completed, and the agent executes

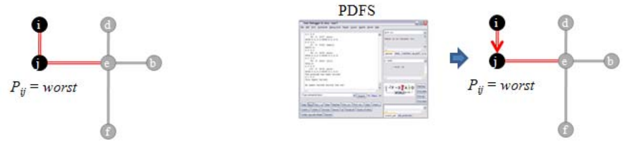
decisions based on the plan ($R(G)$).


BBN
(I_{smoke}, I_{fire}, I_{police}, I_{crowd}, $I_{distance}$)
DFT
$P(t+h) = SP(t) + CM(t+h)W(t+h)$
P_{de}
P_{ej}
P_{be}
P_{ef}
PDFS
P_{ej} = worst

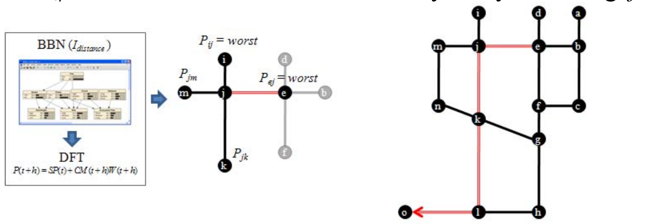
(a) Calculate the choice probability at *e* (b) PDFS selects *ej*



(c) Calculate the choice probability at *j* using only *distance* (d) PDFS selects *ij*



(e) *ij* is the only option with P_{ij} = *worst* (f) PDFS selects *ij* and deletes cycle by removing *ij*



(g) Calculate the choice probability at *j* using only *distance* (h) Repeat above procedure

Figure 6: Illustration of planning algorithm

3.2 Single-Horizon Planning for Novice Agent

For those who do not have knowledge about the area, they cannot make a plan involving multi-horizon as they do not have any information other than what they see for the adjacent paths. Therefore, their planning horizon is one. It is noted that the planning procedure for the novice agent is exactly same as that of the commuter agent (Steps 1 to 4 in Section 3.1) with the planning horizon (n) of 1.

3.3 Meta-model for Commuter Agent

During the multi-horizon planning of the commuter agent (see Section 3.1), the choice probabilities for the paths (even beyond the current intersection) are calculated repeatedly via BBN and DFT (see line 6 of Figure 4), which requires intensive computational power especially when the simulation involves numerous agents. Thus, this section discusses an aggregated meta-model that allows us to obtain the choice probabilities in a significantly shorter time. It is noted that both the original approach (BBN-DFT method) and the meta-model can be used adaptively according to the computational availability.

For planning from the current intersection to an adjacent one (all the considered environmental variables are available to the agent), both the original approach and the meta-model work exactly same, obtaining choice probabilities using BBN and DFT. However, they work differently for planning beyond the current intersection. The original method uses only knowledge on the distance (distance from the considered node to the destination) to infer *M* (evacuation time and risk) via BBN and obtain the choice probabilities. On the other hand, the proposed meta-model utilizes knowledge on the number of connected paths from an intersection (which is related with risk) in addition to knowledge on the distance (which is related with evacuation time). For example, considering four nodes *b, d, f, j* connected to node *e* in Figure 5, nodes *b* and *f* have three paths from each of them, node *d* has one path from it, and node j has four paths from it. Here, we consider the number of connected edges because going to an intersection connected to more paths may be safer under the emergency evacuation situation. Based on knowledge about the distance and the number of connected paths, its preference is calculated using Equation (4).

$$P_i = CP_i + \left|\frac{x_{curr} - x_{dest}}{x_i - x_{dest}}\right| \cdot \left|\frac{y_{curr} - y_{dest}}{y_i - y_{dest}}\right| \cdot w_{dist} \qquad (4)$$

where P_i = preference of path i, CP_i = number of paths connected from the node connected to path i, (x_i, y_i) = x, y coordinates of the node connected to path i, (x_{curr}, y_{curr}) = x, y coordinates of the current node, (x_{dest}, y_{dest}) = x, y coordinates of the destination node, w_{dist} = weight on the distance factor. Once the calculated preference values are fed into Soar, Soar selects one path using the choice probabilities converted from the preference values.

To illustrate the proposed meta-model, we consider the same example used in Section 3.1. For the decision from the current intersection (e) to an adjacent intersection, the agent uses the original (BBN-DFT) method. It is assumed that path *ej* is selected ($R(G)$ = {*ej*}). Now, the agent is at node *j*. Each of their number of connected paths are CP_e = 4, CP_i = 1, CP_k = 4, and CP_m = 2, respectively, and each of their coordinates are (x_e, y_e) = (5, 7), (x_i, y_i) = (3, 8), (x_k, y_k) = (3, 4), and (x_m, y_m) = (1.5, 7). It is assumed that the current node ((x_{curr}, y_{curr}) = (x_j, y_j)) is at (3, 7), the destination node ((x_{dest}, y_{dest}) = (x_o, y_o)) is at (0, 0), and w_{dist} is set to 2. Then, preference values for each node can be calculated using Equation (4): 1) P_e = 4+3/5·2 = 5.2, 2) P_i = 1+7/8·2 = 2.75, 3) P_k = 4+7/4·2 = 6.8, and 4) P_m = 2+3/1.5·2 = 6. Here, as path *ej* has been already selected,

P_e is set to *worst*. Then, Soar uses these preference values to calculate the choice probabilities: 1) P_i = 2.75 / (2.75 + 6.8 + 6) = 0.18, 2) P_k = 6.8 / (2.75 + 6.8 + 6) = 0.44, and 3) P_k = 6 / (2.75 + 6.8 + 6) = 0.38, and selects a path randomly based on those choice probabilities. This planning procedure is repeated until it reaches the destination node *o* or the number of planning horizon is reached. The experimental results, which will be discussed in the next section, in the current work are based on the meta model.

4 EXPERIMENT AND VALIDATION

This section discusses a crowd simulation model that mimics the considered emergency evacuation scenario, human-in-the-loop experiments for behavioral data collection, and testing of impacts of several factors (e.g. demographics of people, number of policemen) on evacuation performance.

4.1 Simulation Model Development

The environment (paths and intersections information of the National Mall in Washington D.C.) has been implemented in AnyLogic® 6.0 agent-based simulation software. As discussed in Section 2, an agent plans and makes decisions via BBN, DFT, and PDFS techniques. For the implementation purposes, we employed various software packages: Netica for BBN, JAMA (A Java Matrix Package) for DFT, and Soar for PDFS. Figure 7 depicts an exemplary rule written in Tcl (a scripting language) for Soar.

```
sp {next-intersection*propose*intersection-anywhere
        (state <s> ^intersection <i> ^active.name <ain>)
        (<i> ^name <in>)
        (<i> ^near <ain>)
        (<i> ^pref <pf>)
        (<i> ^pref <ipf>)
        (<s> ^prefPremium.dist <fc> ^max <max>)
-->
        (<s> ^operator <o> = (+ <ipf>)+   )
        (<o> ^name <in>)
        }
```

Figure 7: An exemplary rule written in Tcl in Soar

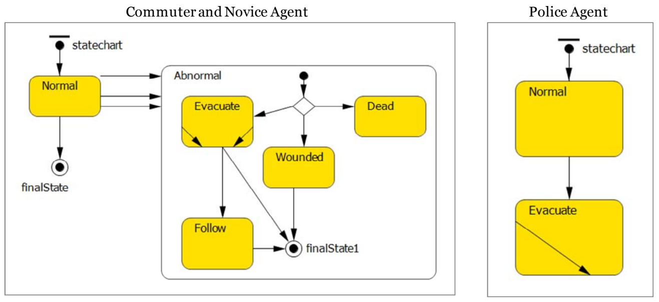


Figure 8: State charts of the agent behaviors

In our simulation, three types of agents are considered, including 1) commuter, 2) novice, and 3) police agent. The commuter and novice agents are differentiated further into leader and follower agents, respectively. As mentioned before, depending on the *confidence index*, some of the commuter agents act as a leader who leads the follower agents to the exits, and some of the novice agents follow the leader agents. Figure 8 depicts behaviors of each agent using state charts.

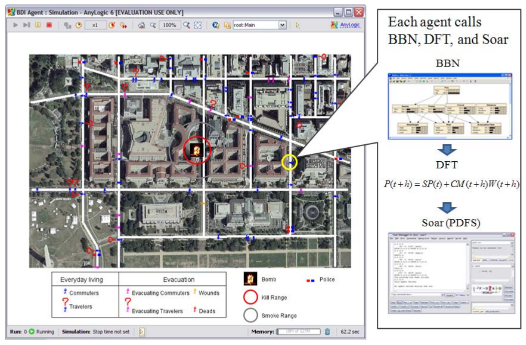


Figure 9: Emergency evacuation simulation in AnyLogic interacting with BBN, DFT and Soar

Figure 9 depicts a snapshot of the AnyLogic simulation. When the simulation is started, the given number of agents for each type are generated and placed randomly within the simulating area heading to their destinations of everyday life (under the normal situation). After 15 seconds from the simulation start, an explosion occurs in the middle of the area. Based on the distance from the explosion, the agents 1) who were within 'fatal range' (as marked with circle in Figure 9) from the explosion at that moment will be dead, 2) who were within 'wound range' will be wounded, and who were within 'notice range' will hear and notice the explosion. Smoke will go up and be diffused from the explosion. In addition to the perception of sound and smoke, the agent can notice the explosion via communication with other agents and police. When an agent approaches to the other agent within certain proximity, they can communicate and exchange the information about the explosion. When the agents notice the explosion, they start to move faster and head to one of the four exits placed in the area. The constructed simulation allowed us to observe agents' behaviors that mimic human in the given scenario, using which we were able to evaluate various evacuation policies.

4.2 Virtual Reality Human Experiments

As discussed in Sections 2.2 and 2.3, BBN infers *M* and *W*(*t*), and DFT calculates preference values of the considered options based on them. Thus, constructing an accurate BBN for a human is a critical task to mimic the human behavior. To this end, we need to conduct human experiments to extract their behaviors and capture them into a BBN. In this research, the Cave Automatic Virtual Environment (CAVE) is employed to allow human experiments. The hardware system used is FakeSpace Inc. CAVE simulator. Figure 10 depicts a human-in-the-loop experiment in CAVE and the CAVE system. The 3D model projected within the CAVE system is developed using Google SketchUp 3D modeling software. The individual 3D images were collected from Google SketchUp component library and Google 3D Warehouse. Figure 11 depicts snapshot of a virtual cityscape of an intersection developed by Google SketchUp 3D modeling software.

In the human-in-the-loop experiment, each subject is asked to evaluate various situations (fire, smoke, police, and crowd) of the intersection. The collected data on the relationship between the situation and the subject's evaluation is used to construct a BBN.

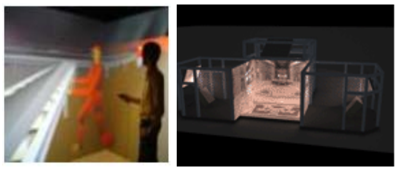


Figure 10: Human-in-the-loop experiment and CAVE

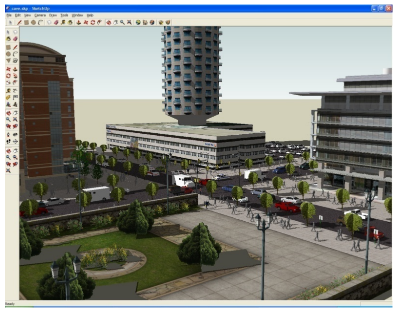
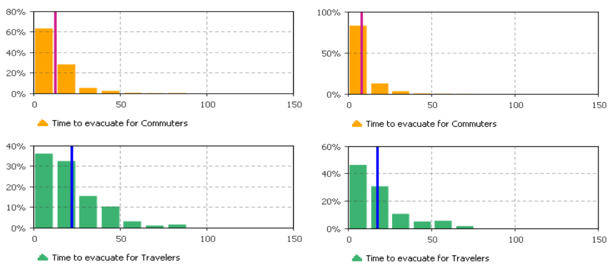


Figure 11: VR model (Google SketchUp)

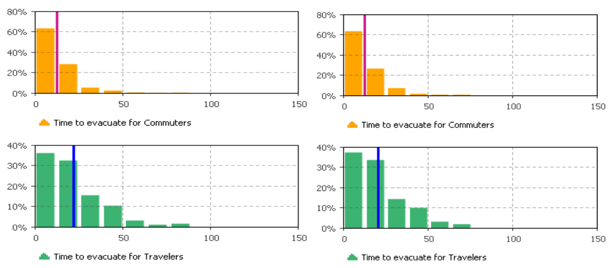
4.3 Simulation Results

Using the constructed crowd simulation model, we conducted various experiments to test an impact of several factors on evacuation performance. Figures 12 and 13 depict impacts of the number of police officers and the number of leaders, respectively. In each figure, the top histogram represents the evacuation time for the commuter agents, and the second histogram represents the evacuation time for the novice agents (e.g. travelers). When we conducted the simulation with only 50 police officers deployed (see Figure 12(a)), the average evacuation times for the commuter agents and the novice agents were 13 minutes and 35 minutes, respectively. When we increased the number of police officers to 500 (see Figure 12(b)), the average evacuation times for the commuter agents and the novice agents were 8 minutes and 15 minutes, respectively. In both simulations, 2000 commuter agents and 200 novice agents were distributed. The results reveal that we can reduce the evacuation times for both agents by increasing the number of police officers in the field.



(a) With 50 police officers (b) With 500 police officers

Figure 12: Impact of police on average time to evacuate



(a) With 50 leaders (b) With 200 leaders

Figure 13: Impact of leaders on average time to evacuate

Similarly, when we conducted the simulation with 50 leaders (see Figure 13(a)), the average evacuation times for the commuter agents and the novice agents were 13 minutes and 34 minutes, respectively. When we increased the number of leaders to 200 (see Figure 13(b)), the average evacuation times for the commuter agents and the novice agents were 13 minutes and 22 minutes, respectively. In both of simulations, 2000 commuter agents and 200 novice agents were distributed. The results reveal that we can reduce the evacuation times for the novice agents by increasing the number of leaders in the field. It is noted that the constructed simulation is flexible, so it can be used to test impacts of other factors (e.g. impact of information-sharing via speakers or text-messaging) on various other security metrics (e.g. percentage of casualties).

5 CONCLUSION AND FUTURE RESEARCH

In this research, we have proposed promising techniques to realize each sub-module of the extended BDI architecture. The techniques employed in this research have been selected to represent the characteristics of the corresponding steps of human decision planning and making process. Successful implementation of these techniques allowed the extended BDI architecture to be used to mimic human behaviors even in the intricate situation. Furthermore, the proposed techniques and the extended BDI framework have been demonstrated using the agent-based simulation, which has provided various dynamic environments. The developed simulation allowed us to simulate and observe the crowd behaviors under various conditions. The proposed simulation has a potential to allow the responsible governmental and law-enforcement agencies to evaluate different evacuation and damage control policies beforehand, which in turn allows the execution of the most effective crowd evacuation scheme during an actual emergency situation. As part of the proposed research, we have conducted human-in-the-loop experiments in the Virtual Reality system to collect more realistic human behaviors. Currently, the learning effect of agents is not considered. To validate the proposed model against decisions from real human, we plan to conduct an additional human-in-the-loop experiment. These tasks are left as future work.

ACKNOWLEDGMENTS

This work was supported by the Air Force Office of Scientific Research under AFOSR/MURI F49620-03-1-0377.

REFERENCES

Busemeyer, J. R., and J. T. Townsend. 1993. Decision Field Theory: A Dynamic-Cognitive Approach to Decision Making in an Uncertain Environment. *Psychological Review* 100(3): 432-459.

Earnshaw R. A., J. A. Vince, and H. Jones. 1995. *Virtual reality Applications*. Orlando, Florida:Academic Press Limited.

Einhorn, H. J. 1970. The Use of Nonlinear, Noncompensatory Models in Decision Making. *Psychological Bulletin* 73: 221-230.

Gibson, F. P., M. Fichman, and D. C. Plaut. 1997. Learning in Dynamic Decision Tasks: Computational Model and Empirical Evidence. *Organizational Behavior and Human Decision Processes* 71: 1-35.

Konar, A., and U. K. Chakraborty. 2005. Reasoning and unsupervised learning in a fuzzy cognitive map. *Information Sciences* 170: 419-441.

Laird, J. E., A. Newell, and P. S. Rosenbloom. 1987. SOAR: An Architecture for General Intelligence. *Artificial Intelligence* 33: 1-64.

Lee, S., Y. Son, and J. Jin. 2008. Decision Field Theory Extensions for Behavior Modeling in Dynamic Environment using Bayesian Belief Network. *Information Sciences* 178(10): 2297-2314.

Newell, A. 1990. *Unified Theories of Cognition*. Cambridge, Massachusetts: Harvard University Press.

Norling, E. 2004. Folk psychology for human modeling: extending the BDI paradigm. In *International Conference on Autonomous Agents and Multi-Agent System (AAMAS).* New York.

Opaluch, J. J., and K. Segerson. 1989. Rational Roots of Irrational Behavior: New Theories of Economic Decision-Making. *Northeastern Journal of Agricultural and Resource Economics* 18(2): 81-95.

Payne, J. W. 1982. Contingent Decision Behavior. *Psychological Bulletin* 92: 382-402.

Rao, A. S., and M. P. Georgeff. 1998. Decision procedures for BDI logics. *Journal of logic and computation* 8: 293-342.

Rothrock, L., and J. Yin. 2008. Integrating Compensatory and Noncompensatory Decision Making Strategies in Dynamic Task Environments. In *Decision Modeling and Behavior in Uncertain and Complex Environments,* ed. T. Kugler, C. Smith, T. Connolly, and Y. Son, 123-138. Springer (forthcoming).

Shendarkar, A., K. Vasudevan, S. Lee, and Y. Son. 2006. Crowd Simulation for Emergency Response using BDI Agents Based on Immersive Virtual Reality. *Simulation Modelling Practice and Theory* (submitted).

Zhao, X., and Y. Son. 2007. BDI-based Human Decision-Making Model in Automated Manufacturing Systems. *International Journal of Modeling and Simulation* (accepted).

AUTHOR BIOGRAPHIES

SEUNGHO LEE is a Ph.D. student in the department of Systems and Industrial Engineering at the University of Arizona. He received his Bachelor of Engineering degree in Industrial Engineering from Korea University in Korea in 1999 and his M.S. degree in Industrial Engineering from Texas A&M University in 2005. His research focuses on application of distributed simulation and simulation of human decision making. He is a student member of IIE and INFORMS. He can be reached by email at <mountlee@email.arizona.edu>.

YOUNG-JUN SON is an associate professor in the Department of Systems and Industrial Engineering at The University of Arizona. He received his BS degree in Industrial Engineering with honors from POSTECH in Korea in 1996 and his M.S. and Ph.D. degrees in Industrial and Manufacturing Engineering from Penn State University in 1998 and 2000, respectively. His research focuses primarily on the application of distributed and hybrid simulation to the analysis and control of automated manufacturing system and integrated enterprise. He is a recipient of 2004 SME Eugene Merchant Outstanding Young Manufacturing Engineer Award and 2005 IIE Outstanding Young Industrial Engineer Award. He is an associate editor of the International Journal of Modeling and Simulation and the International Journal of Simulation and Process Modeling. He can be reached by email at <son@sie.arizona.edu >.