Reasoning about Benefits and Costs of Interaction with Users in Real-time Decision Making Environments with Application to Healthcare Scenarios

by

Hyunggu Jung

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2010

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

This thesis examines the problem of having an intelligent agent reasoning about interaction with users in real-time decision making environments. Our work is motivated by the models of Fleming and Cheng, which reason about interaction sensitive to both expected quality of decision (following interaction) and cost of bothering users. In particular, we are interested in dynamic, time critical scenarios. This leads first of all to a novel process known as strategy regeneration, whereby the parameter values representing the users and the task at hand are refreshed periodically, in order to make effective decisions about which users to interact with, for the best decision making. We also introduce two new parameters that are modeled: each user's lack of expertise (with the task at hand) and the level of criticality of each task. These factors are then integrated into the process of reasoning about interaction to choose the best overall strategy, deciding which users to ask to resolve the current task. We illustrate the value of our framework for the application of decision making in hospital emergency room scenarios and offer validation of the approach, both through examples and from simulations. To sum up, we provide a framework for reasoning about interaction with users through user modeling for dynamic environments. In addition, we present some insights into how to improve the process of hospital emergency room decision making.

Acknowledgements

Foremost, I would like to express my deep and sincere gratitude to my supervisor, Dr. Robin Cohen for being an outstanding advisor. Her invaluable guidance, encouragement, and support from the initial to the final level enabled me to make this thesis successful. Furthermore, I am deeply indebted to Dr. Chrysanne Di Marco and Dr. Ian McKillop for serving on the thesis committee with their insightful comments.

I would like to show my gratitude to Diane Doran, R.N. and Ivana Matic of the Lawrence S. Bloomberg Faculty of Nursing for their discussions regarding hospital scenarios in my study. I also gratefully acknowledge the Strategic Research Network hSITE grant of the Natural Sciences and Engineering Research Council (NSERC) of Canada.

I warmly thank Dr. Kate Larson and Dr. Urs Hengartner for their valuable advice and guidance to make my graduate studies successful. I also would like to thank Dr. Kee-Eung Kim, Dr. Jin Hyung Kim and Dr. Wonjoon Kim for their consistent encouragement since my undergraduate studies.

Special thanks to John Champaign for his assistance on proofreading my thesis. Thanks as well to Jimin Hwang, Seong Yong Jang, and Chaenyung Cha for providing me valuable feedback on my writing.

I am very grateful to all of my colleagues: Thomas Ang, Ahmed Ataullah, Simina Branzei, Loky Lachlan Dufton, Kah-Kuen Fu, Jun Carol Fung, Joshua Gorner, Greg Hines, Georgia Kastidou, Reid Kerr, Farheen Omar, Thomas Reidemeister, Elena Ruiz, and Tyrel Russell for creating such a great friendship and making my life much more enjoyable throughout my stay at Waterloo.

Finally, my warmest appreciation goes to my family who has been supporting me throughout my time preparing and writing my thesis.

Dedication

This is dedicated to my parents, Jaesun Chung and Okhe Chyun and my brother, Jonggu Jung for all their unconditional love and support.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In this thesis, we explore the problem of reasoning about interaction between an intelligent agent and a user, in scenarios that are dynamic and time critical. An intelligent agent is a software agent that has been designed to problem solve on behalf of its user, given a user goal and preferences, and knowledge of the environment in which it is operating. In particular, we develop a decision-theoretic framework for deciding when an agent should enlist the problem solving assistance of a user, considering both the expected quality of decision and the possible cost of bothering the user.

As a motivating example, consider the following (from [11]):

> Suppose a user wants to know which path to choose in order to minimize travel time but is unsure whether there are any traffic snarls on the two possible routes. The intelligent agent may be privy to additional knowledge about these routes (though there is uncertainty introduced, since its information is not guaranteed to be accurate).
>
> The user may choose to pick one of the two paths based on his or her knowledge alone, without consulting the intelligent agent. But in cases where the user's confidence in its ability to decide is low and where its confidence in the agent's ability to decide is, at least, higher, it may choose to consult the agent.
>
> In the following scenario the agent faces uncertainty and could benefit from information gathering. An earthquake occurred, and there are fires raging throughout the city. An assistant agent attached to a fire brigade is tasked with finding a route that its fire brigade can take to quickly get to a fire.

Suppose the agent computes two feasible routes, $Route_1$ and $Route_2$. $Route_1$ is the most direct path to the fire, but requires crossing a bridge. Unfortunately, due to the earthquake, the agent is uncertain about traveling over the bridge, as it may have collapsed from the quake. If the bridge did indeed collapse, then the agent will need to take a costly detour to get to the fire. On the other hand, $Route_2$ is an indirect path that does not involve crossing any bridge.

In this scenario, a relevant query that the agent could ask is Q_1 = "Is the bridge condition good enough to travel on?", with the possible answer responses being $r_{1,1} = BG$ (bridge condition good), and $r_{1,2} = BB$ (bridge condition bad). With this information, the agent should be able to make a more informed decision and a wiser choice of route. Without the information, the agent can still choose what it feels is the best action to take, given the information that it has to date.

Previous research has examined what is referred to as mixed-initiative systems: partnerships between users and intelligent agents, where either party may take the initiative to direct the problem solving or the interaction [11]. This research emphasized the importance of user modeling in the determination of whether an intelligent agent, charged with performing the problem solving autonomously, should in fact interrupt the user in order to offload the decision making.

In environments where there are multiple agents representing multiple users, all of whom are cooperating towards the completion of some goal, the challenge is referred to as adjustable autonomy [26]. Now each intelligent agent has a variety of possible users whose assistance may be engaged and the question is not only whether a user should be bothered but instead which user should be approached. One valuable approach to resolving the adjustable autonomy challenge is proposed in the work of Tambe et al. referred to as the Electric Elves project [27]. In this research, each agent calculates a preferred strategy: a series of other entities (users or agents) that may be asked, at which point in time, followed by other entities who will then be asked if the first party does not respond, through to a final strategy state which is typically having the agent perform the decision making itself.

As an example to illustrate the reasoning that an agent may undergo when determining who should be approached to take over the decision making of a current task, consider the following scenario from [6]:

In this scenario, a meeting has been scheduled. *Ed*, the presenter for an upcoming group meeting, has to cancel the meeting. The *agent* is charge of organizing meetings is then tasked with finding an appropriate time slot to reschedule the meeting. It can either make the decision itself, ask *Ed* to make the decision or in fact ask *Bob*, the team leader, to make the decision. Which user to ask would be dependent on expected quality of decision from that user, as well as likelihood of a response. An overall strategy may be as expressed as follows: *Bob(5)Ed(10)Agent* meaning that *Bob* will be asked to make the decision, the agent will wait 5 time units for a response and if one is not received, *Ed* will be asked and another 10 time units will pass, at which time the agent will just make the decision itself.

As an improvement to the approach of Tambe, Cheng developed a framework for adjustable autonomy multiagent systems which integrated not only a method for reasoning about which entity should perform the decision making but also a method for reasoning about interaction: asking users questions first, to then better direct the decision making process [6]. In this framework, overall strategies for asking entities either for information (partial transfers of control) or to perform decision making (full transfers of control) are generated and evaluated, with the strategy that maximizes the overall utility being selected. As with Fleming's approach [11], one critical element in the reasoning is the cost of bothering a user, modeled in terms of a set of formulae that integrate several elements of user modeling (such as the user's inherent willingness to assist and his or her attentional state).

Distinct challenges arise, however, when the environment in which the reasoning is performed is coping with critical tasks that must be completed in a timely manner, and where there is a good deal of dynamic change. One such environment is that of decision making for emergency rooms in hospitals. If an intelligent system were to be running, calculating which experts would be best to contact to assist with the current patients, modeling the expected improvement in decision making as well as the likelihood of response (sensitive to the bother that would be generated), then improved overall decision making could result.

The challenge of effective hospital decision making motivates the development of the models presented in this thesis. Our overall approach consists of two primary components. The first is a general framework for reasoning about strategies to generate and to select, in order to approach the best users to perform decision making. Due to the demands of time critical decision making in the face of dynamic change, novel elements need to be developed, beyond the framework proposed by Cheng [6]. In our research, we focus on the issue of reasoning with up-to-date parameter values and we propose a technique referred to as strategy regeneration. In particular, we outline (in Chapter 3, Section 3.1) how this element may be integrated (in Chapter 4) into an overall framework like that of Cheng's, and we demonstrate through a series of examples (in Chapter 5) the inherent value of this technique, compared to the case where strategies are not regenerated.

The second component of our research is extended user modeling, to enable more effective decision making. For this task, we focus specifically on the scenario of emergency room patient care and outline the kinds of parameters which, when modeled, would lead to improved decisions about which experts to ask to assist for patients (in Chapter 3, Section 3.2). The value of our particular approach is discussed some simulated hospital scenarios (in Chapter 4 and Chapter 5), where patients who are not attended to effectively may become a problem for the hospital.

In Chapter 6, we discuss the value of our research both in advancing research on intelligent interaction for artificial intelligent systems and in assisting with emergency room decision making. In addition, we chronicle a large number of interesting future paths with this research, to expand its scope and to enrich its models. We conclude with a list of the most valuable contributions offered in the thesis.

Chapter 2

Background

In this chapter, we introduce some terminology and present some foundational related work.

2.1 Agents and Multiagent Systems

In artificial intelligence, an intelligent agent (IA) is defined as an autonomous entity that observes and acts upon an environment and directs its activity in order to achieve its goals [25].

Multiagent systems are defined as follows: "... systems in which several interacting, intelligent agents pursue some set of goals or perform some set of tasks."[29] In addition, Weiss [29] emphasizes the value of interaction in multiagent system, as follows:

"To build a multiagent system in which the agents "do what they should do" turns out to be particularly difficult ... The only way to cope with these characteristics is to enable the agents to interact appropriately."

2.2 Mixed-initiative Systems

Mixed-initiative systems are ones in which a system (i.e. an intelligent agent) and users form a problem solving partnership, where either party is able to take the initiative to solve the problem. Haller and McRoy [13] describe mixed-initiative systems as follows: "In a problem-solving situation, the information and abilities needed for the task at hand are often distributed among the collaborators. As a result, direction and control of the interaction shifts among the participants. If future computational systems are to collaborate effectively with users to solve problems, they must have the ability to take and relinquish control of the problem-solving process and the communication about it. The theory and the mechanisms that underlie these behaviors are ... computational models for mixed initiative interaction."

2.3 Fleming's Model for Mixed-Initiative Systems

A model for determining the interaction between the system and the user in a mixed-initiative system is presented in [11]. This work forms the starting point for our own research and is described below.

2.3.1 Reasoning about Interaction

Fleming & Cohen [11, 12] developed a domain-independent decision-theoretic model for an agent to reason about whether or not it should interact with a human user. The model is aimed at solving 'single decision' problems, defined as "from an initial state, the system decides about interacting with the user, then makes a decision about what action to perform and then takes that action to complete the task" [12].

The general algorithm for a system to reason about whether or not it should ask a question is fairly intuitive, and proceeds as follows (as presented by [12]):

1. Determine the expected benefits of interacting with the user. More specifically, determine by how much the system's performance on the task is expected to improve (if at all) after asking the user the question.

2. Determine the expected costs of the interaction.

3. Proceed with the interaction only if the benefits exceed the costs.

The computation of the *benefits* of interaction is simply $Benefits = EU_{ask} - EU_{\neg ask}$, where EU_{ask} represents the expected utility of an agent's decision using information obtained from the user, while $EU_{\neg ask}$ represents the expected utility of an agent's decision made without any more information. Note that the expected utility denoted here does not incorporate the costs incurred, but rather refers only to the value of the decision.

The value of $EU_{\neg ask}$ is the expected utility of the action that the agent believes to be the most promising in the current state, given the information it has without asking the user any further questions and is calculated as follows:

$$EU_{\neg ask} = \max_{a \in Actions} EU(a) \tag{2.1}$$

For each possible action a, the expected utility calculation takes into account the fact that there may be uncertainty about the possible outcomes of the action. For any given action a, suppose there are several possible results, each denoted res_i, with probability $P(res_i)$ and utility $U(res_i)$. Then,

$$EU(a) = \sum_i P(res_i) \cdot U(res_i) \tag{2.2}$$

To compute the value of EU_{ask}, let P_{UR} denote the probability that the user responds and let EU_{UR} be the expected utility that the agent could achieve if it receives an answer from the user. If the user does not respond or says that he does not know the answer, the agent will choose the action it believes to be the best, with expected utility $EU_{\neg ask}$.

$$EU_{ask} = P_{UR} \cdot EU_{UR} + (1 - P_{UR}) \cdot EU_{\neg ask} \tag{2.3}$$

Here, EU_{UR} is computed by considering all possible responses r_j that the user could give and the expected utility of the action a_j that the agent would choose, given each response r_j.

$$EU_{UR} = \sum_{r_j \in Resp} P(r_j) \cdot EU(a_j \mid r_j) \tag{2.4}$$

The computation of interaction *costs* is done through a linear model, where the total cost is a weighted sum of individual costs; i.e., $Costs = \sum_i w_i C_i$. Two costs considered in Fleming & Cohen's work are the cost of the time required to interact with the user, and the cost of bothering the user[1]. This research clearly outlines where a model of bother cost can be introduced into the process of reasoning about interaction.

[1][11] also discusses briefly the cost of carrying out certain queries, such as costs in fetching from databases.

Bother cost is in fact included as a key factor in determining whether or not an agent will interact with a human user. How to model bother cost is discussed in greater detail in Section 2.3.2, below.

2.3.2 Bother Cost Model

There are two main principles to Fleming's bother cost model. The first is the idea that "recent interruptions and diffcult questions should carry more weight than interruptions in the distant past and very straightforward questions." The second is the notion that whether a user is willing to interact with the system is a critical factor to reason about, in order to avoid bothering the user too much. Fleming's model is as follows:

- First estimate how bothersome the dialogue has been so far. This *bother so far (BSF)* is given by $BSF = \sum_I c(I) \times \beta^{t(I)}$, where the system computes the sum over all the past interactions with the user (including the currently considered interaction). $c(I)$ is how bothersome the interaction was (e.g., cognitive effort required by the user to answer the question), $t(I)$ is the amount of time that has passed since that interaction, and β is a discount factor that diminishes the effect of past interactions as time passes.

- Let w represent the user willingness, with a range of 0 to 10, with higher w meaning more willingness.

- Let $\alpha = 1.26 - 0.05w$ and $Init = 10 - w$. Here, $Init$ is to reflect the cost of bothering a user for the first time.

- Then, $BotherCost = Init + \frac{1-\alpha^{BSF}}{1-\alpha}$. From this formulation, a lower willingness w results in a higher $Init$ cost, and also a higher α value (which amplifies the effect of the bother so far BSF). As BSF increases, so too does $BotherCost$, but at different rates, depending on the α value. As shown by [11], for low w values, α will be greater than 1, and we will see an exponential-like increase due to BSF, while for high w values, α will be less than 1, and we see a log-like increase. The values used for the calculation of α are in order to generate these kinds of curves for users with these characterizations of willingness.

2.4 Adjustable Autonomy

Adjustable autonomy multiagent systems are ones in which any agent can offload decision making of its current task to a user or to another agent [14]. In this section, we give an overview of the Electric Elves (E-Elves) project which represents a agent-based adjustable autonomy model. This model inspired that of Cheng [6], on which our own model is based.

2.5 The Electric Elves (E-Elves) Project

Research by Tambe et al. [27] at ISI/USC explored the challenge of adjustable autonomy multiagent systems – allowing agents involved in completing tasks on behalf of users to transfer decision making control to another entity in the environment, where an entity would either be another agent or one of the human users.

Whereas previous research on adjustable autonomy systems led to a decision of the agent to retain decision making control or to transfer it to a single entity in the environment, Tambe et al. proposed the concept of a transfer-of-control strategy: a planned sequence of transfer-of-control actions. In this case, there is a plan to ask a particular entity but to wait a certain period of time before then asking a different entity, through to the end of the planned sequence. For example, the transfer-of-control strategy $U_1(5)\ U_2(10)\ Agent$ would have the Agent attempting to offload decision making to $User_1$, waiting until time 5, then asking $User_2$ and waiting until time 10, at which time the Agent itself would take on the decision making. The central problem to resolve is to determine which of the possible strategies maximizes the expected utility of the overall decision[2]. In order to determine the best strategies, agents reason with a set of parameters and estimations of their values, including EQ_e^d, the expected quality of decision made by the entity being asked and, $P(t)$, the probability that the entity will respond with a decision of that expected quality.

2.5.1 Definitions

At this point, it is useful to record various definitions from Tambe's model [27], as follows:

Transfer-of-Control A transfer-of-control strategy is a planned sequence of transfer-of-control actions, including both those that actually transfer control and those that

[2]Utility is a concept in artificial intelligence aimed at modeling the inherent value of an action for a user, relative to their goals and preferences

simply buy more time to get input.

An agent An agent, A, is responsible for making a decision, d.

Entities There are n entities, $e_1, ..., e_n$, who can potentially make the decision. These entities can be human users, other agents, or the agent itself.

EQ(t) The expected quality (EQ) of a decision, d, made by an entity, e, at time, t, is given by $EQ_e^d(t) : \Re \to \Re$.

P(t) The continuous probability distribution over time that the entity, e, in control will respond with a decision of quality at time, t is given by: $P_\top(t) : \Re \to \Re$.

W(t) The cost of delaying a decision until time t is $W(t) : \Re \to \Re$. $W(t)$ is assumed to be non-decreasing and that there is some point in time, $\lhd$, when the costs of waiting stop accumulating (i.e., $\forall t \geqslant \lhd, W(t) = W(\lhd)$).

2.5.2 Expected Utility

In the Electric Elves model, the expected utility is calculated by multiplying the probability of response by the expected utility at each instant of time and summing the products. Below is a calculation of the expected utility for an arbitrary continuous probability distribution.

$$EU = \int_0^\infty P_\top(t) EU_{e_c}^d(t).dt$$

where e_c denotes the entity currently in decision making control.

$EU_{e_c}^d(t)$ consists of two factors: the quality of the decision and the cost of waiting as follows:

$$EU_{e_c}^d(t) = EQ_e^d(t) - W(t)$$

2.6 Cheng's Model

2.6.1 Reasoning about Interaction

We begin with a brief overview of Cheng's model for reasoning about interaction [6], before proceding to present our variation for the time-critical environment of hospital decision making. Cheng [6] extends the Electric Elves model to allow each agent to reason about initiating information gathering interaction with a user before determining what to do next. This adjustable autonomy model is described in detail below. In this work, Cheng differentiates between the agent querying an entity for information which he refers to as a partial transfer-of-control or PTOC, and the agent asking an entity to make a decision which we refer to as a full transfer-of-control or FTOC. Both of these cases are considered to be interaction from the agent to the entity. This yields overall what he refers to as a hybrid transfer-of-control strategy.

Below are descriptions of an FTOC and a PTOC node.

FTOC An FTOC node represents the agent fully transferring control to some entity at some time point t_i and waiting until time point t_{i+1} for a response. For simplicity's sake, we regard the case of the agent deciding autonomously as an FTOC to the agent itself. Note that for this special FTOC case, we do not need to plan for any transfers afterwards because the decision will definitely have been made.

PTOC A PTOC node represents the agent partially transferring control by asking some entity a query at some time point t_{i+1} for a response. Each possible response to a query will be represented as a branch from the PTOC node to a strategy subtree representing what the agent should do when it receives that particular response.

There are several terminologies to describe a PTOC node. For example, a particular query is denoted as Q_j, and its possible answer responses are denoted as $r_{j,1}, r_{j,2}, ...,$ and $r_{j,n}$. In addition, $r_{j,?}$ is included to represent "I don't know" and $r_{j,\neg resp}$ is to represent the 'no response' case which occurs when the entity does not respond in time.

Figure 2.1 revisits the meeting scheduling scenario introduced in Chapter 1 and illustrates an example hybrid TOC strategy where the agent is responsible for rescheduling a presentation meeting time. In this interaction strategy, the agent is not sure which factor should be prioritized when selecting a meeting time. Thus, the agent does a PTOC to the group leader *Bob*, asking query Q_1="When rescheduling a meeting time, which factor should be prioritized?", with the possible answer responses being $r_{1,1}$="Prioritize having the meeting earlier", $r_{1,2}$= "Prioritize having the meeting be convenient for the presenter", $r_{1,?}$="I have no idea", and $r_{1,\neg resp}$=No response.

The response from Bob will determine the job the agent will do. If the response is $r_{1,1}$, we simply move to the FTOC node, and the agent figures that it is relatively capable enough to make the decision by itself and decides autonomously. If the response is $r_{1,?}$ or $r_{1,\neg resp}$, then the agent just tries to make the best decision it can.

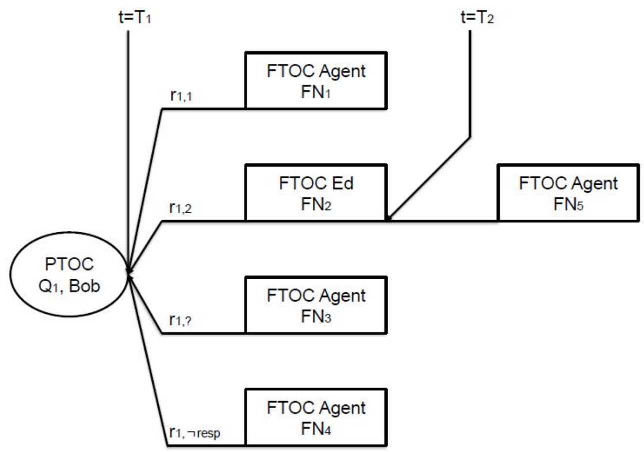

Figure 2.1: Example Hybrid TOC Strategy.

If the response is $r_{1,2}$, then the agent figures that the presenter Ed, is much more capable of making a good decision, and so does an FTOC to Ed, asking Ed to make the meeting time decision and waiting until time T_2 for the response. If time T_2 arrives and Ed still has not responded back, then the agent will just decide by itself.

Reasoning about this interaction in fact requires an effective model of bother cost as well. The challenge is for each agent to determine its optimal TOC strategy, by generating possible strategies, evaluating the expected utility of the strategies and then selecting the one with the highest expected utility. The use of the term "utility" here is consistent with that used in E-Elves and reflects the difference between the benefits and costs.

The expected utility of a strategy s is, in turn, dependent on the expected utility of all the leaf nodes in s.[3]

Formula-wise, $EU(s)$ is calculated as follows:

$$EU(s) = \sum_{LN_l} [P(LN_l) \times (EQ(LN_l) - W(T_{LN_l}) - BC_{LN_l})] \tag{2.5}$$

In the equation, $EQ(LN_l)$ denotes the expected quality of the agent's decision at leaf node LN_l, given the information it has gathered along the path to LN_l. $W(T_{LN_l})$ denotes

[3] Note that, in this Section, only two strategies were considered for the agent: it could ask the user a question (with expected utility EU_{ask}) or it could proceed with its reasoning without the user's help (with expected utility $EU_{\neg ask}$). The benefits of asking were calculated by computing $EU_{ask} - EU_{\neg ask}$, and these benefits were then compared to the costs of interaction. In the model in this section, the expected utility of many possible interaction strategies, with both the benefits and costs incorporated into a single expected utility measure, $EU(s)$.

the costs of waiting until the time of leaf node LN_1 to finish the interaction.[4], and BC_{LN_l} denotes the bother cost accumulated from interacting with entities from all the transfers that the agent has done up to (and including) the current transfer-of-control under consideration.

The expected utility of the overall strategy is in effect the sum of the utility of each of the individual paths in it; thus, one needs to factor in the probability that the particular path will be taken $P(LN_l)$. This in turn will depend as well on the probability of response.

2.6.2 Bother Cost Model

Cheng offers an equation for modeling the cost of bothering a user that is user-specific and incorporates several important elements as follows:

- The difficulty of the interruption query, $TOC\_Base\_Bother\_Cost$. For example, usually asking a user his/her preference is easier (i.e., cognitively less intense) than asking a user to decide on a plan of action.
- The attention state of the user, $Attention\_State\_Factor$. For instance, a user is more interruptible when resting than when he/she is busy with important work.
- The user's unwillingness to interact with the system, $User\_Unwillingness\_Factor(UUF)$ $(0.5 \leq UUF \leq 2)$. This is a measure of how receptive (or rather, unreceptive) the user is towards being TOC'ed, and how disrupted they are by interruptions. Cheng chooses to model user unwillingness factor, rather than willingness in order to make the overall calculations more intuitive. As $User\_Unwillingness\_Factor$ increases, the value of bother cost increases.
- The timings of the interruptions, $t(TOC)$, and the discount factor, β $(0 < \beta < 1)$, which reduces the bother impact of past TOCs as time passes.

The formulae below then specify the bother cost calculation:

- $Init = User\_Unwillingness\_Factor \times Attention\_State\_Factor \times$
  $TOC\_Base\_Bother\_Cost$

[4]Note that $W(t)$ is introduced in the E-Elves model as well and is intended to represent the cost of waiting in order to get the task completed, with respect to the need for the actions to be carried out quickly within the domain of application.

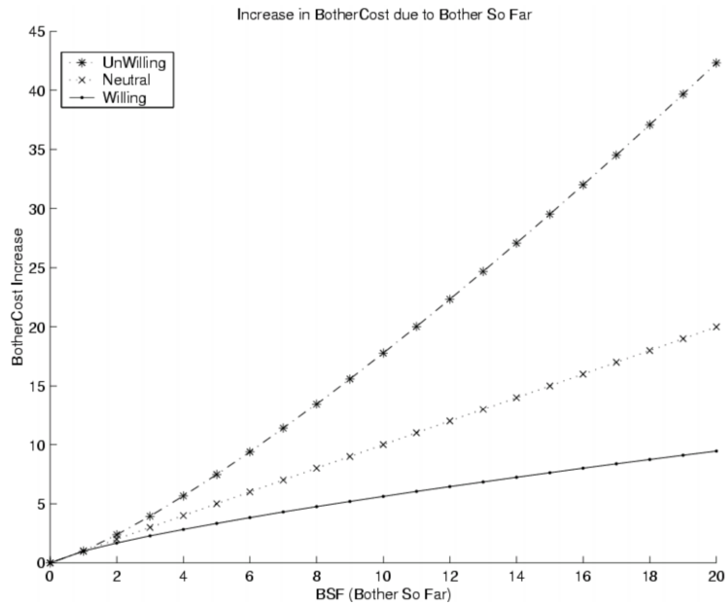

Figure 2.2: Graph showing how much bother cost increases due to bother so far, for the different user willingness types.

- *BSF (Bother So Far)* $= \sum_{toc \in PastTOC} TOC\_Base\_Bother\_Cost(toc) \times \beta^{t(toc)}$, where $PastTOC$ is the set of all the past TOCs experienced by the user, $TOC\_Base\_Bother\_Cost(toc)$ is just the $TOC\_Base\_Bother\_Cost$ of toc, and $t(toc)$ is the time point at which toc occurred.

- To determine the increase to the bother cost due to BSF, there is a function, $BC\_Inc\_Fn(BSF, User\_Unwillingness)$, that maps a BSF value to a bother cost increase, based on the user's unwillingness level.

- *BotherCost (BC)* $= Init + BC\_Inc\_Fn(BSF, User\_Unwillingness)$.

In addition, Cheng suggests possible bother cost factor values as follows:

- [$TOC\_Base\_Bother\_Cost$] Easy=5, Medium=10, Hard=20

- [$Attention\_State\_Factor$] Relaxed=0.75, Neutral=1, Busy=1.25

- [$User\_Unwillingness\_Factor$] Willing=0.5, Neutral=1, Unwilling=2

- [β] 0.90

- [BC_Inc_Fn] For Willing, $BC\_Inc\_Fn(x) = x^{0.75}$, for Neutral, $BC\_Inc\_Fn(x) = x^1$, for Unwilling, $BC\_Inc\_Fn(x) = x^{1.25}$. This gives us roughly the same bother cost shape as used by [11] and [2]. Figure 2.2 shows how the bother cost increases due to bother so far, for the different user willingness types.

2.6.3 Strategy Selection

Cheng provides algorithms to generate all possible strategies, choosing the one with the highest expected utility. This strategy is then executed and various entities are either asked questions or asked to take over decision making. This process is sensitive to both the expected quality of decision and the cost of bothering as the two primary factors to consider.

2.7 Hospital Background

A network of researchers in computer science, engineering, nursing and medicine is currently involved in a project aimed at providing effective decision making support in various healthcare contexts, including that of the hospital setting, in a project known as hSITE (Healthcare Support Through Information Technology Enhancements)[22]. The central aim of this project is to be able to employ the right person, at the right time, with the right information, for more effective healthcare. Specific challenges arise in the emergency room setting, in particular[5].

In general, in hospital emergency room scenarios, a patient arrives and is seen by the ER triage nurse, who determines to what section of the ER the patient should go to. The triage nurse has experience to decide whether the patient needs to be in acute, sub-acute, fast-track section or resuscitation room. After this is determined, the first patient goes to the respective section and is taken care of and assessed again, by an ER nurse, then a Nurse practitioner/ER resident/doctor. If a patient is coming in with a condition that is obvious and needs a specialist, then the nurse would proceed to call the specific specialist right away, e.g., a brain injury triggers a call for a neurologist. If a condition is not obvious, through further assessments with the nurse, nurse practitioner, and ER doctor, it would be determined which specialist service to call.

[5] This scenario was outlined for us by health professionals associated with the strategic research health network known as hSITE [22].

If the patient being brought in by Emergency Medical Services is critical, they are brought into the resuscitation room right away, and nurses assigned to that room attend to the patient, as well as an ER doctor. From there it is determined if other specialities are called. If a patient is being brought in presenting with a stroke, for example, and the triage nurses are notified beforehand, the triage nurses call the specialists for thrombolytic therapy immediately as this therapy is time sensitive. In some hospitals, an urgency level is determined for each patient in the ER and is kept on record for the patient.

Without consideration of the possible bother being incurred when experts are solicited (so merely focusing on who might have the best expertise for the problem at hand) what results is a significant bottleneck in the effective delivery of the care to the patient. Especially with patients in critical conditions, it is important to make very effective decisions about who should be consulted. In addition, the parameters that serve to model the patients are constantly changing in this dynamic setting, and reasoning needs to be sensitive to this as well.

Chapter 3

Our Framework for Reasoning about Dynamic, Time-Critical Interaction

3.1 Decision Making Element

We introduce a model that can be used specifically for scenarios where an agent is reasoning about which human users to enlist to perform decision making, in an environment where decisions need to be made under critical time constraints and where the parameters that serve to model the human users are changing dynamically, to a significant extent.

Transfer-of-control strategies are generated in order for the optimal strategy to be selected for execution. Within the transfer-of-control strategy, one user after another is expected to respond, should there be no response from the previous user after a certain extent of time. In contrast to the approach in Section 2.6, attempts at full transfers of control are in fact framed as PTOCs with the question Q: "Can you take over the decision making?". This then enables both a "yes" response, which results in an FTOC to this user or a "no" response (or silence)[1]. Note that, distinct from Cheng [6], we are not reasoning about which questions to ask a user; we are focused on asking this particular question, to drive the decision making.

The "no" response brings up a new approach to the dynamically changing environment and the challenge of time. There is a new generation of possible strategies in anticipation of other potential experts who may be suitable to help, and the transfer-of-control attempt ends. The regeneration is useful in overcoming possible challenges where choices that are

[1]The case of silence in our model corresponds to the case of $r_{1,\neg resp}$ in Section 2.6.1

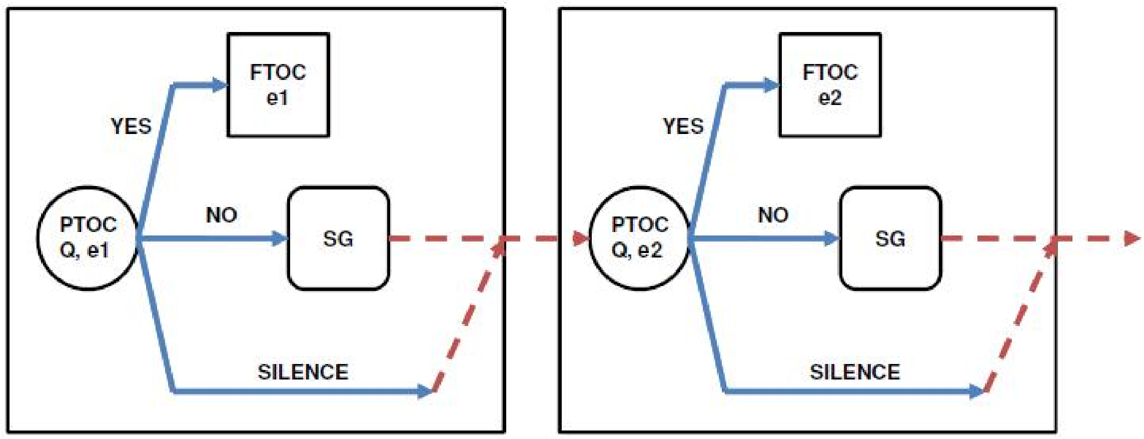

Figure 3.1: Visual representation of strategy with the FTOCs and PTOCs; each world occupies one square.

less than optimal can be unappealing. It helps to reevaluate the users who are available to help in terms of their decisions and bother costs, and the best users to enlist can be recomputed.

The approach that is followed when there is silence projects continued attempts to contact other users. At the end of this chain of attempts, we inject a final decision of strategy regeneration[2]. Strategy regeneration will then allow for an updating of parameter values. Note that in our current model, we make the simplification that the strategies do not involve asking different entities within the same chain. This is because we are limiting ourself to only one question, that of asking the expert to help. We revisit this restriction in Section 6.1.9.

A diagram outlining the FTOCs and the PTOCs that we envisage is presented in Figure 3.1 where an arrow with a solid line means the stream of time, but a dotted line means there is no break by the end of the arrow. In addition, we introduce a concept of *world* to facilitate the computation of the utility of any given strategy. One *world* consists of one PTOC, one FTOC, and one SG (Strategy Regeneration) node and includes all the parameters currently used to calculate benefits and costs to reason about interaction with entities. Therefore, when the current *world* is moved to the next step, our system asks a new entity. The number of worlds is equivalent to the number of entities that will be asked. The SG node is clarified as follows:

[2]This is in contrast to the general approach provided in Section 2.6.1, where the final node in a chain is usually one where there is a full transfer-of-control back to the agent, who must then perform the decision.

SG Node The expected utility of a SG node (sg) is just $EU(sg) = 0$ as the EU of a PTOC node is zero, Cheng [6] clarifies that PTOCs have zero utility because this is instead computed at the final FTOC node of the strategy. This is because a decision is never made in a SG. The power of the SGs is that they allow a strategy chain to be regenerated. We encounter a SG node when the response from an entity is "No" or after an entire chain of silence, to the end of the strategy. Strategy regeneration allows us to then reflect current parameter values.

3.1.1 Algorithm for Finding Optimal Strategy

The procedure for the agent to find the optimal strategy is a basic branch and bound search, where the agent generates all possible strategies, containing one query, evaluates the generated strategies, and then simply selects the one with the highest expected utility value. The strategy generation and evaluation steps are described below. Our procedure differs from that of Cheng [6] in its manner of generating and evaluating strategies containing FTOCs, PTOCs, and SG nodes.

3.1.2 Strategy Generation

The basic idea is that we will generate all possible strategies containing one query. As Cheng mentioned [6], we can visualize a strategy as a tree as displayed in Figure 2.1, composed of FTOC nodes and PTOC nodes, but in our framework SG nodes are inserted into the tree as well.

In our model, we assume that we are not considering strategies which involve the same entity more than once but are considering all possible entities. Thus, using this analogy, the length of the strategy is then the maximum depth of the tree, which is the number of entities.

Let E be the set of all relevant entities in the system. Let FN be the set of all possible FTOC nodes. Each FTOC node (fn) identifies which entity e in E to fully transfer control to. Let SG be the set of all possible SG nodes. Another strategy is generated at the SG node (sg). Let PN be the set of all possible PTOC nodes. The set PN consists of all the possible pairings between a query q and entity e in E. So, each PTOC node (pn) identifies which entity to ask which query. Also, each pn has branches corresponding to the possible responses to query q, and each of these branches will have an attached strategy subtree. With n possible entities, we obtain $n!$ strategies.

Using this model formulation, we show an algorithm for generating strategies, as below:

procedure GENERATESTRATEGY(*int i*) ▷ i represents the length of the strategy chain to generate
  if $(i = 1)$ **then** ▷ Base Case
    create a PTOC node
    create strategy by appending FTOC and SG node to the PTOC node
  else if $(i = \textit{the number of entities})$ **then**
    create a Default node
    create strategy by appending Default node to S_i
  else
    $S_{i-1} \leftarrow \mathit{GenerateStrategy}(i-1)$ ▷ Get the set of strategies of length i-1
    create a PTOC node, a FTOC node, and a SG node
    create strategy by appending the PTOC node to the PTOC in S_{i-1}
    create strategy by appending the FTOC and SG node to the PTOC node which has been just created
  end if

  return set of all newly created strategies
end procedure

3.1.3 Strategy Evaluation

The formulae that would be used to reason about the expected utility derived from a strategy are explained below. As outlined in Section 2.3, the optimal strategy is determined by evaluating the expected utility (EU) of each of the generated strategies and selecting the one with the highest EU value. As explained in [6], a strategy generation phase would begin with the simplest strategies (of length one) and then expand to longer strategies by adding an FTOC or a PTOC node to previously generated strategies. The strategy generation is then limited by bounding the maximal length of strategy. For the model of reasoning presented in this section, we limit the strategy generation based on the number of entities under consideration. If there are k entities, $k!$ strategies are generated, among which we choose the one with the highest EU value. Then, the overall EU of strategy s is computed by taking the sum of the EU of all the leaf nodes in s.

Considering the calculation for the expected utility of a strategy as the sum of the utilities of the leaf nodes in that strategy we then proceed to calculate separately the utility of a) ending in a full transfer-of-control b) ending in a strategy regeneration from a "no" response c) going down a path of "silence"[3] to a final stage of strategy regeneration. Note as well that here the probability that a transfer-of-control is occurring is dependent on the probability that all the PTOC nodes prior to this one are silence and on the probability of the response associated with this node ("yes", "no" or silence).

Below are the equations to calculate EU for the three cases having different a leaf node: "yes" , "no", and silence. If the response is "yes", the leaf node of the path from the initial PTOC node is an FTOC node. The EU of an FTOC node (fn_l) in the jth *world* is computed as follows:

$$EU_j(fn_l) = \prod_{pn_{prev}} P_{e_{prev}}^{\{resp=Silence\}} \times P_{e_i}^{\{resp=Yes\}} \times (EQ_{e_i}^d - W(t_e - t_s) - BC_{fn_l}) \tag{3.1}$$

where $EU_j(fn_l)$ denotes the expected utility in the jth world of full transfer-of-control; pn_{prev} denotes a partial transfer to entity e_{prev}; $P_{e_{prev}}^{\{resp=Silence\}}$ denotes the probability that asking all the previous entities the query will result in silence; $P_{e_i}^{\{resp=Yes\}}$ denotes the probability that asking the entity e_i the query will result in "Yes"; $EQ_{e_i}^d$ denotes the expected quality of decision, d the entity e_i has; $W(t_e - t_s)$ denotes the cost of waiting a decision between time t_s and t_e. Cheng [6] suggested that the cost of waiting is a increasing function. Since our model is for time-critical scenario, the cost of waiting should be increased rapidly as time progresses.

t_e is the ending time of the FTOC (so where the arrow meets the FTOC square in Figure 3.1); t_s is the starting time of the FTOC, (so where the arrow heading into the FTOC square originates); BC_{fn_l} is the accumulated bother cost to entities resulting from all the transfers that agent has done up to the current transfer-of-control under consideration.s

If the response is "no", the leaf node of the path from the initial PTOC node is a SG node. The EU for each SG node in the jth world is calculated as follows:

$$EU_j(sg) = \prod_{pn_{prev}} P_{e_{prev}}^{\{resp=Silence\}} \times P_{e_i}^{\{resp=No\}} \times (EQ_{e_i}^d - W(t_e - t_s) - BC_{sg} - SGC) \tag{3.2}$$

where BC_{sg} is the accumulated bother cost to entities resulting from all the transfers that the agent has done up to the current transfer-of-control under consideration, and SGC denotes the cost of generating a new strategy.

[3] If nobody has been found to answer either "yes" or "no", we define this as a case of silence.

In case of a silence response, we put a virtual node dfl ("default"), into the final world. The EU of the virtual node (dfl) [4] in the final (nth) world is computed as follows:

$$EU_n(dfl) = \prod_{pn_{prev}} P_{e_{prev}}^{\{resp=Silence\}} \times P_{e_i}^{\{resp=Silence\}} \times (EQ_{e_i}^d - W(t_e - t_s) - BC_{sg} - SGC) \quad (3.3)$$

There are n FTOC nodes, n PTOC nodes, and one virtual node in the overall framework with n worlds. We obtain the overall EU of strategy s by summing up n EU values for FTOC nodes, n EU values for SG nodes, and one EU value for the virtual node as follows:

$$EU(s) = EU_n(dfl) + \sum_{j=1}^{n}(EU_j(fn_l) + EU_j(sg)) \quad (3.4)$$

where n represents the number of worlds.

3.2 User Modeling

Projecting the decision making into scenarios of emergency room hospital decision making, we introduce additional assumptions and proceed to develop user models that comprise the important features for this environment.

Fleming [11] proposed that a system should ask for further input precisely when the perceived benefits of this interaction exceed the expected costs. Table 3.1 provides a summary of the main factors identified in his work, where a domain-independent model was developed for reasoning about interaction with users. The second column in Table 3.1 classifies each factor as being relevant to the user model (UM), dialogue model (DM) or task model (TM). Fleming proposed inclusion of all of these factors, but only developed detailed models of some of these parameters,d as shown in his primary formulae (Section 2.3). In our work, factors classified as being relevant to the user model will be designed as follows: a new level of expertise factor, used to help model user knowledge, user unwillingness factor, and task criticality, all of which are described for our scenarios.

In Section 3.1, we had the PTOC nodes reflecting "Can you take over the decision making?". In emergency room scenarios, we now imagine patients arriving, with a current patient selected as the task that needs to be resolved. We would imagine our algorithms for decision making being run in order to determine the medical experts to approach, to attend

[4] The leaf node for the silence response is set to sg.

Table 3.1: Summary of factors to be used in model

Factor	**Model**
The user's knowledge	UM
The user's willingness to interact	UM
The user's preference utility function	UM, TM
Task criticality	TM, UM
Current context and expected understandability of system utterance	DM, UM
Previous interactions The expected improvement of the system's task performance due to interaction	DM TM
Time and time criticality	TM
Resource costs and other task-specific costs	TM

to the patient. The question behind each new PTOC now becomes: "Can you look after this patient?". Following the models of Fleming (Section 2.3) and Cheng (Section 2.6), decision making will be influenced in part by the cost of bothering a user and in part based on the expected quality of decision from this user, requiring effective user modeling to capture these elements.

Below we propose formulae for modeling the cost of bothering users. We assume that the bother cost of users is determined by their willingness.

3.2.1 User Unwillingness Factor

Fleming proposes the user's willingness to interact as one of the domain-independent user modeling factors [11]. In our model, we design a factor called user unwillingness factor which represents the aspect of the user's willingness to interact.

Whereas in Fleming's model [11] each user may have an inherent unwillingness to help, in our model the user unwillingness factor is determined by adding the values for the attentional state factor to the lack of expertise factor. In other words, we introduce a new user modeling parament (expertise) to better model the expected quality of decision and to ensure that attentional state (whether the medical expert is occupied with another patient, for the scenario of hospital decision making) critically influences the calculation of user unwillingness. Table 3.2 shows how the user unwillingness factor is calculated as a combination of the attentional state factor and the lack of expertise factor.

Table 3.2: Calculation of user unwillingness factor

	ASF, *Relaxed*	*ASF*, *Neutral*	*ASF*, *Busy*
LEF, *Low*	0.5	0.75	1
LEF, *Med*	0.75	1	1.25
LEF, *High*	1	1.25	1.5

The user unwillingness factor of an entity is determined by considering attention state factor and level of expertise of the entity. For example, the entity is a willing person if the attention state factor is *Relaxed* and his knowledge is very helpful for a user. If the attention state factor is *Busy*, and she is not good at the specialized area of the current patient, she would not be willing to help a user. There are three different categories: *Med-Willing*, *Medium*, and *Med-Unwilling*. *Med-Willing* is chosen if the attention state factor is *Relaxed* and the lack of expertise factor is *Low*. Also, this state is chosen if the attention state factor is *Neutral* and the lack of expertise factor is *Low*. *Med-Unwilling* is chosen if the attention state factor and the lack of expertise factor is *Neutral* and *High*, respectively or *Busy* and *Med*.

3.2.2 Level of Expertise

Fleming proposes the user's knowledge as one of the domain-independent user modeling factors [11]. Fleming defines the user's knowledge as how likely it is that the user will have the required knowledge to answer the question. In our model, we design a factor called a level of expertise, which includes the same aspect of the user's knowledge but evaluates the knowledge to answer "Yes" for the specific question, "Can you take over the decision making?" in PTOC nodes as described in Section 3.1. If the medical expert as a user thinks that he/she has enough knowledge to treat the selected patient, the experts will be willing to answer "Yes" to the question.

In addition, we propose the addition of level of expertise as part of the user modeling for the bother cost, a *Lack_of_Expertise_Factor*. We choose to model Lack of Expertise, rather than Expertise in order to make the overall calculations more intuitive. A high Lack of Expertise corresponds to a low level of expertise, overall. This parameter is used to help to record the general level of expertise of each doctor, with respect to the kind of medical problem that the patient is exhibiting. This is called *Lack_of_Expertise_Factor* rather than just *Expertise_Factor* to be consistent in adding another factor, $User\_Unwillingness\_Factor$ which increases a *Cost* factor in order to determine bother cost. Therefore, lack of expertise is *High* when the medical expert is lack of expertise and *Low* when the expert is considerable expertise in dealing with the medical problem of the patient.

Note that modeling lack of expertise enables the overall calculations to fall into the desired range specified by Cheng (Section 2.6.2). This is explained in greater detail below.

We also adjust the calculations proposed by Cheng for estimating bother cost, (shown in Section 2.4.2) in order to reduce the number of parameter values that need to be acquired or solicited (for our time-critical scenarios).

Some factors which affect bother cost in hospital settings are thus as follows.

- The difficulty of the query, $TOC\_Base\_Bother\_Cost$. In hospital settings with a streamlined model, this factor is fixed, since we are considering only one question to ask (whether the user can assist with the patient).

- $Attention\_State\_Factor$ reflects how busy the doctor (medical expert) is. A doctor currently without a patient would have a low attentional state value; a doctor currently attending to a patient would have a high attentional state value.

- The lack of expertise of the doctor, $Lack\_of\_Expertise\_Factor$ (LEF). The expertise will then affect the unwillingness of the doctor. That is, as this factor increases, $User\_Unwillingness\_Factor$ increases.

- The user's unwillingness to interact with the system, $User\_Unwillingness\_Factor$. This is a measure of how receptive (or rather, unreceptive) the doctor is towards being TOC'ed, and how disrupted they are by interruptions. We currently present a simplification of this calculation. This factor is obtained by adding $Attention\_State\_Factor$ to $Lack\_of\_Expertise\_Factor$.

- The timings of the interruptions, $t(TOC)$, and the discount factor, β $(0 < \beta < 1)$, which reduces the bother impact of past TOCs as time passes. We choose a relatively high β because hospital settings are under time-critical situations where the time step is 'small'.

With the inclusion of the new parameter, we then propose adjusted formulae for modeling the bother to users with following:

$$User\_Unwillingness\_Factor = Attention\_State\_Factor + Lack\_of\_Expertise\_Factor \tag{3.5}$$

$$Init = User\_Unwillingness\_Factor \times Attention\_State\_Factor \times TOC\_Base\_Bother\_Cost \tag{3.6}$$

$$BSF(\textit{Bother So Far}) = \sum_{toc \in PastTOC} TOC\_Base\_Bother\_Cost(toc) \times \beta^{t(toc)} \tag{3.7}$$

$$BC\_Inc\_Fn(x, bc\_inc\_fac) = x^{bc\_inc\_fac} \tag{3.8}$$

where bc_inc_fac is determined by user unwillingness factors. If user unwillingness factor of the user is $Willing$, bc_inc_fac is 0.75. If $Unwilling$, bc_inc_fac is 1.25. Otherwise, bc_inc_fac is 1 as suggested by Cheng [6].

$$BotherCost(BC) = Init + BC\_Inc\_Fn(BSF, User\_Unwillingness) \tag{3.9}$$

Here are some suggestions for possible bother cost factor values:

- [$TOC\_Base\_Bother\_Cost$] Easy=5, Medium=10, Hard=20
- [$Attention\_State\_Factor$] Relaxed=0.75, Neutral=1, Busy=1.25
- [$Lack\_of\_Expertise\_Factor$] High(i.e., not very expert)=0.25, Medium=0, Low(i.e., very expert)=−0.25
- [β] 0.90
- [BC_Inc_Fn] For Willing, $BC\_Inc\_Fn(x) = x^{0.75}$, for Neutral, $BC\_Inc\_Fn(x) = x^1$, for Unwilling, $BC\_Inc\_Fn(x) = x^{1.25}$.

Note that the $User\_Unwillingness\_Factor$ ends up producing the value of 2 when the user is Busy and the $Lack\_of\_Expertise\_Factor$ is High and a value of 0.5 when the user is Relaxed and the $Lack\_of\_Expertise\_Factor$ is Low. These values in fact correspond to those suggested for Unwilling and Willing by Cheng (from Section 2.6.2).

3.2.3 Task Criticality

We introduce another new parameter, *task criticality (TC)*, to affect the reasoning about interaction. *TC* is used to enable the expected quality of a decision to be weighted more heavily in the overall calculation of expected utility, when the case at hand is very critical. This parameter may also be adjusted dynamically. When a patient has high task criticality,

Table 3.3: Level and increasing rate by the score of task criticality

	[0, 10)	[10, 80)	[80, ∞)
Level of Task Criticality	*Low*	*Med*	*High*
Increasing Rate	2 %	5 %	10 %

Table 3.4: Weights to determine the expected quality of a decision

Task Criticality	*High*			*Med*			*Low*		
LEF	*Low*	*Med*	*High*	*Low*	*Med*	*High*	*Low*	*Med*	*High*
Weight	10 %	0 %	−10 %	5 %	5 %	−5 %	0 %	0 %	0 %

a high level of expertise is required because the patient's condition may become much more serious if not treated intensively.

There are two characteristics of task criticality. First, the TC of a patient who is not treated increases as time passes. We consider different increasing rates for each TC level: high-level, medium-level, and low-level. In other words, high-level TC will increase faster than low-level TC as time passes. Generally speaking, more critical patients usually gets worse quickly than less critical patients. In order to design the dynamic aspect of the status of the patient, the TC curve should be convex so that the increasing rate would grow as time passes. However, each patient has his/her own medical problem so that he/she may have his/her own function of task criticality. We would need to bring the user-customized task criticality function which could be continuous rather than the discrete our model suggests. Table 3.3 shows the sample increasing rate for each type of task criticality. expertise of medical experts.

Second, we propose that the expected quality of a decision is weighted by the TC level and the level of expertise as presented in Equation (3.10). Table 3.4 shows weights for each case, which will be applied in example scenarios in Chapter 4.

$$EQ^d_{e_i} \rightarrow EQ^d_{e_i} + (Weight \times EQ^d_{e_i}) \tag{3.10}$$

If the TC of some patient is low, the patient does not have to consider the expertise of a doctor carefully. Thus, the expertise does not affect determining the expected quality of a decision. However, the TC of some patient is high, the patient should consider the expertise of a doctor seriously. Therefore, the expertise will affect determining the expected quality of a decision. In this case, when the TC is high, the expected quality of a decision

Table 3.5: Probability of entity response to question Q by $User\_Unwillingness\_Factor$

$User\ Unwillingness\ Factor$	Yes	No	$Silence$
$Willing$	60 %	20 %	20 %
$MedWilling$	50 %	30 %	20 %
$Neutral$	40 %	40 %	20 %
$MedUnwilling$	30 %	50 %	20 %
$Unwilling$	20 %	60 %	20 %

needs to be adjusted with more weight. Also, when the TC is low, the expected quality of a decision needs to be adjusted with less weight.

3.2.4 Probability of Response

In our model, we assume that each user has his/her own probability of response. In Tambe's [27] model, we have seen the function of probability of response. Likewise, the probability of response of users is influenced by the user's willingness. A willing person definitely prefers to say, "Yes." rather than "No".

Our model suggests parameter values as shown in Table 3.5. We basically give the same estimate of probability for the case of "Yes" and "No" if the user unwillingness factor of the entity is $Neutral$. For $Willing$ and Med-$Willing$ entities, we increase the probability for the case of "yes" and decrease the probability for the case of "No". On the other hand, for $Unwilling$ and Med-$Unwilling$ entities, we increase the probability for the case of "No" and decrease the probability for the case of "Yes". This simplification is revisited in Section 6.1.2.

As Cheng [6] has a discretized time step function with probability of responses values, different probabilities of responses are given in Table 3.6 [5] which represents the suggested probability of responses for "Yes" and "No". We assume that a willingness person prefer to give a response quickly.

In our strategy chain, we move to the next world if the response from the entity who was asked is silence. We need to check how much time has been passed so that we can find whether the response from the entity can be considered as silence. When you see the Table 3.6, the willingness entity is 33% during the 1st time unit, 27% during the 2nd time

[5] Suggested values in Table 3.6 are initially defined to execute but could be updated by keeping track of behavior of entities.

Table 3.6: Probability of how quickly entity responds to question

User Unwillingness Factor	1 *unit*	2 *unit*	3 *unit*	4 *unit*
Willing	33 %	27 %	13 %	7 %
MedWilling	27 %	23 %	17 %	13 %
Neutral	20 %	20 %	20 %	20 %
MedUnwilling	13 %	17 %	23 %	27 %
Unwilling	7 %	13 %	27 %	33 %

unit, 13% during the 3rd time unit, and 7% during the 4th time unit. The sum of all the probabilities becomes 80% which implies that the entity has 80 percent for the response between "Yes" and "No".

For example, the user unwillingness factor of the entity who is asked is *Willing*, the entity would willing to respond shortly. However, if the user unwillingness factor of the entity is *Unwilling*, he/she would not likely to respond shortly.

Chapter 4

Example

In this chapter, we present a series of examples to demonstrate the value of our proposed framework. In particular, we anticipate having our algorithm running, determining the best strategy (which experts to ask, how long to wait) which would then inform the first clinical assistant (FCA) of how to address the needs of the current patient. Below is an overview of the examples we introduce and the parts of our model that they illustrate:

Section 4.1 Fixed initial parameters for entities and patient - strategy chain is shown

- **Scenario 1** high critical patient
- **Scenario 2** medium critical patient
- **Scenario 3** low critical patient

Section 4.2 Variable initial parameters for entities - strategy chain is shown

- **Scenario 1** high critical patient - best expert is *Relaxed*
- **Scenario 2** high critical patient - best expert is *Busy*
- **Scenario 3** low critical patient - best experts are all *Busy*

Section 4.3 executing a strategy - varying responses from entities

- **Scenario 1** of Section 4.2

4.1 Different Task Criticality Values

This first set of examples briefly illustrates how different strategies are selected to execute, when the criticality of the patient is different. We assume 4 medical experts and keep their parameter values the same for all three scenarios in this subsection. We vary the task criticality of the current patient: high in scenario 1, medium in scenario 2 and low in scenario 3.

We illustrate how effective choices are made to enable the coordination of medical professionals and the resolution of the decision making regarding patient care in hospital emergency room scenarios.

In our example, the 4 possible medical experts to approach, each has differing expected quality of decision making, differing attentional state (e.g. attending to other patients currently or not), and different inherent willingness to assist.

The model parameters used in our scenarios are as follows:

- [$TOC\_Base\_Bother\_Cost$] 15
- [$Time\ discount\ factor\ \beta$] 0.90
- [$initial\ EQ$] 150
- [$Cost\ of\ Waiting,\ W(t)$] $t^{1.5}$
- [$the\ Number\ of\ Worlds\ (n)$] 4
- [SGC] 0 cost

Note that we assume that the cost of regenerating a strategy is simply zero. We also assume that the expected quality of decision for all specialists begins with the same initial base value (which is then adjusted according to the user's expertise level as discussed earlier). The *TOC_Base_Bother_Cost* is set "somewhat high" (using the range of values listed in Section 2.6.2 and 3.2.2). This is because our one question has the purpose of getting a user to agree to carry out a decision.

As shown in Table 3.3, there are levels of task criticality and increasing rates for each level of task criticality. We assume that the patient's criticality will be assessed by the first clinical assistant attending to the patient (and it may be periodically updated, as the patient continues to be unattended).

Table 4.1: Probability of entity response to question Q by Attentional State Factor

Attentional State Factor	*Yes*	*No*	*Silence*
Relaxed	0.45	0.45	0.1
Busy	0.1	0.6	0.3

Table 4.2: Elapsed time (in time units)

Attentional State Factor	*Lack-of-expertise Factor*	*Yes*	*No*	*Silence*
Relaxed	*Low*	2	4	5
Relaxed	*High*	3	3	5
Busy	*Low*	3	3	5
Busy	*High*	4	2	5

The probability of response depends on the user unwillingness factor of each doctor. Table 4.1 represents a probability of response for "Yes", "No", and "Silence", which includes two cases: relaxed and busy with respect to the attentional state factor.

The time by which a response to a question will be generated from the doctors will be referred to as the elapsed time. We have this elapsed time determined by the attentional state factor and the expertise level of the doctor, according to predefined values as provided in Table 4.2. The units of time are left unspecified. Note that we assume a fixed elapsed time for all the cases of "silence".

In our scenario, we divide task criticality into three levels: high, medium, and low level. Given the level of task criticality, we determine the expected quality of a decision by adding a weight represented in Table 3.4. The expected quality of a decision is dynamically changed by the change of the value of task criticality as time progresses.

Table 4.3 shows profiles of available doctors in a hospital currently (experts e_1, e_2, e_3, and e_4), for our sample scenarios below.

Scenario 1.
A patient has just arrived at the emergency room who is assessed as highly critical. The FCA tries to search for the right doctor for the current patient with the decision-support system. In this hospital, as in Table 4.3, there are four doctors, e_1, e_2, e_3, and e_4. Our system checks the profile of each doctor and begins finding the optimal strategy by calculating an expected utility for each generated strategy. Since there are four doctors, we obtain 4! strategies. By evaluating each strategy, we obtain 24 expected utility values for each strategy shown in Table 4.4. This is computed using Equation 3.4, which in turn relies on

Table 4.3: Profiles of entities at the time the patient arrives

	e_1	e_2	e_3	e_4
Attentional State Factor	Relaxed	Busy	Busy	Relaxed
Lack-of-expertise Factor	Low	High	Low	High
Probability of Response for Yes	0.45	0.1	0.1	0.45
Probability of Response for No	0.45	0.6	0.6	0.45
Probability of Response for Silence	0.1	0.3	0.3	0.1
Elapsed time for Yes	2	4	3	3
Elapsed time for No	4	2	3	3
Elapsed time for Silence	5	5	5	5

calculation of bother cost, sensitive to each particular expert who might be asked. The greatest expected utility is $EU(s) = 198.39$, the strategy chain: e_1 - e_3 - e_4 - e_2.

The strategies that ask e_4 first do not have high EU values, even though the expert is not Busy and can attend to the patient. This is because the High Criticality of the patient has raised the weight of the EQ value in the calculation. The maximal EU of a strategy that asks e_4 first is 49.87. Likewise, strategies that select e_2 first have very low EU values, as this expert is both Busy and has High $Lack\_of\_Expertise$.

Scenario 2.
A patient has just arrived at the emergency room, assessed at medium criticality. The FCA tries to search for the right doctor for the current patient with the decision-support system. In this hospital, there are again four doctors, e_1, e_2, e_3, and e_4. Our system checks the profile of each doctor and begins finding the optimal strategy by calculating an expected utility for each generated strategy. As before, since there are four doctors, we obtain 4! strategies. By evaluating each strategy, we obtain 24 expected utility values for each strategy shown in Table 4.5. The greatest expected utility is $EU(s) = 163.03$ whose strategy chain is e_1 - e_3 - e_4 - e_2.

Scenario 3.
A patient has just arrived at the emergency room, assessed at low criticality (but still in need of specialized assistance). The FCA tries to search for the right doctor for the current patient with the decision-support system. In this hospital, there are four doctors, e_1, e_2, e_3, and e_4. Our system checks the profile of each doctor and begins determining the optimal strategy by calculating an expected utility for each generated strategy. And since there are four doctors, we obtain 4! strategies once more. By evaluating each strategy, we obtain

Table 4.4: Expected utility of strategies at the time the patient arrives

No.	Expected Utility (EU)	Strategy Chain
1	189.055932	e_1 - e_2 - e_3 - e_4
2	186.222260	e_1 - e_2 - e_4 - e_3
3	197.846134	e_1 - e_3 - e_2 - e_4
4	198.391167	e_1 - e_3 - e_4 - e_2
5	189.251631	e_1 - e_4 - e_3 - e_2
6	188.493741	e_1 - e_4 - e_2 - e_3
7	49.190011	e_2 - e_1 - e_3 - e_4
8	46.356339	e_2 - e_1 - e_4 - e_3
9	44.739695	e_2 - e_3 - e_1 - e_4
10	32.315057	e_2 - e_3 - e_4 - e_1
11	4.896448	e_2 - e_4 - e_3 - e_1
12	5.150315	e_2 - e_4 - e_1 - e_3
13	138.701167	e_3 - e_2 - e_1 - e_4
14	126.276528	e_3 - e_2 - e_4 - e_1
15	177.340219	e_3 - e_1 - e_2 - e_4
16	177.885253	e_3 - e_1 - e_4 - e_2
17	136.679229	e_3 - e_4 - e_1 - e_2
18	133.046656	e_3 - e_4 - e_2 - e_1
19	35.955365	e_4 - e_2 - e_3 - e_1
20	36.209232	e_4 - e_2 - e_1 - e_3
21	44.745566	e_4 - e_3 - e_2 - e_1
22	48.378139	e_4 - e_3 - e_1 - e_2
23	49.867505	e_4 - e_1 - e_3 - e_2
24	49.109615	e_4 - e_1 - e_2 - e_3

Table 4.5: Expected utility of strategies at the time the patient arrives

No.	Expected Utility (EU)	Strategy Chain
1	157.143432	e_1 - e_2 - e_3 - e_4
2	156.109760	e_1 - e_2 - e_4 - e_3
3	162.258634	e_1 - e_3 - e_2 - e_4
4	163.028667	e_1 - e_3 - e_4 - e_2
5	158.614131	e_1 - e_4 - e_3 - e_2
6	158.381241	e_1 - e_4 - e_2 - e_3
7	64.527511	e_2 - e_1 - e_3 - e_4
8	63.493839	e_2 - e_1 - e_4 - e_3
9	60.077195	e_2 - e_3 - e_1 - e_4
10	53.727557	e_2 - e_3 - e_4 - e_1
11	40.483948	e_2 - e_4 - e_3 - e_1
12	40.512815	e_2 - e_4 - e_1 - e_3
13	117.288667	e_3 - e_2 - e_1 - e_4
14	110.939028	e_3 - e_2 - e_4 - e_1
15	141.752719	e_3 - e_1 - e_2 - e_4
16	142.522753	e_3 - e_1 - e_4 - e_2
17	119.541729	e_3 - e_4 - e_1 - e_2
18	117.709156	e_3 - e_4 - e_2 - e_1
19	71.542865	e_4 - e_2 - e_3 - e_1
20	71.571732	e_4 - e_2 - e_1 - e_3
21	76.658066	e_4 - e_3 - e_2 - e_1
22	78.490639	e_4 - e_3 - e_1 - e_2
23	79.980005	e_4 - e_1 - e_3 - e_2
24	79.747115	e_4 - e_1 - e_2 - e_3

24 expected utility values for each strategy as shown in Table 4.6. The greatest expected utility is $EU(s) = 128.20$ whose strategy chain is e_1 - e_4 - e_3 - e_2.

4.2 Conflicting Parameter Values

This series of examples serves to show which strategies are selected when there is a tension between certain parameter values. The first scenario is a base case, where the expert chosen to be first in the strategy is the one who will deliver the best expected quality of decision and is also enduring the least bother. The second scenario is of a highly critical patient, where there is a tension between choosing the best expert for this important task against the cost of bother, since this expert is currently at a high bother level as well. The last scenario is one of a patient with low criticality, where again there is a best expert who is at a high state of bother, but where perhaps an expert with low bother and lower expertise will be adequate to approach.

The examples in this subsection are also described at a greater level of detail. This is done in order to further clarify all the steps that are proposed in our framework in determining the appropriate strategy chain, along with various processes to update and manage the parameters that serve to model the environment.

4.2.1 Scenario 1

In this example, there are five patients waiting for treatment and four medical experts in the emergency room. The FCA tries to search for the right doctor for the current patient. Below are profiles of medical experts:

Entity	ASF
e_1	*Relaxed*
e_2	*Relaxed*
e_3	*Busy*
e_4	*Relaxed*

Entity	Specialized Area	Number of Patients	LEF
e_1	*Cardio*	7	*Med*
e_2	*Cardio*	100	*Low*
e_3	*Cardio*	0	*High*
e_4	*Cardio*	0	*High*

Table 4.6: Expected utility of strategies at the time the patient arrives

No.	Expected Utility (EU)	Strategy Chain
1	125.905932	e_1 - e_2 - e_3 - e_4
2	125.772260	e_1 - e_2 - e_4 - e_3
3	127.346134	e_1 - e_3 - e_2 - e_4
4	127.891167	e_1 - e_3 - e_4 - e_2
5	128.201631	e_1 - e_4 - e_3 - e_2
6	128.043741	e_1 - e_4 - e_2 - e_3
7	80.540011	e_2 - e_1 - e_3 - e_4
8	80.406339	e_2 - e_1 - e_4 - e_3
9	76.089695	e_2 - e_3 - e_1 - e_4
10	74.465057	e_2 - e_3 - e_4 - e_1
11	75.396448	e_2 - e_4 - e_3 - e_1
12	75.650315	e_2 - e_4 - e_1 - e_3
13	96.551167	e_3 - e_2 - e_1 - e_4
14	94.926528	e_3 - e_2 - e_4 - e_1
15	106.840219	e_3 - e_1 - e_2 - e_4
16	107.385253	e_3 - e_1 - e_4 - e_2
17	102.629229	e_3 - e_4 - e_1 - e_2
18	101.696656	e_3 - e_4 - e_2 - e_1
19	106.455365	e_4 - e_2 - e_3 - e_1
20	106.709232	e_4 - e_2 - e_1 - e_3
21	107.895566	e_4 - e_3 - e_2 - e_1
22	108.828139	e_4 - e_3 - e_1 - e_2
23	110.317505	e_4 - e_1 - e_3 - e_2
24	110.159615	e_4 - e_1 - e_2 - e_3

Entity	Specialized Area	Number of Patients	LEF
e_1	$Neuro$	0	$High$
e_2	$Neuro$	0	$High$
e_3	$Neuro$	15	Med
e_4	$Neuro$	120	Low

1. The FCA identifies the most serious patient from the waiting list. Below is a waiting list in this scenario. Our system chooses p_2 since the task criticality of p_2 is highest among patients. Since p_2's task criticality is greater than 80 according to the Table 3.3, he is assessed as highly critical.

No.	Patient	Medical Problem	Task Criticality
1	p_1	$Cardio$	70
2	p_2	$Cardio$	90
3	p_3	$Neuro$	63
4	p_4	$Cardio$	82
5	p_5	$Neuro$	70

2. Then, the waiting list is updated by eliminating the selected patient which has been assessed as the most critical patient. Thus, the number of patients remains becomes four. Below is a table where you can find a updated waiting list in our scenario.

No.	Patient	Medical Problem	Task Criticality
1	p_1	$Cardio$	70
2	p_3	$Neuro$	63
3	p_4	$Cardio$	82
4	p_5	$Neuro$	70

3. We generated strategies by following the process introduced in Chapter 3. In our scenario, 4! strategies are generated since there are four entities attending in our scenario.

4. We evaluate the expected utility of each strategy generated in step 3. There are several steps to evaluate the expected utility as follows:

    (a) We set values of parameters for each entity based on the profile of the current patient. We already know some information about each entity such as attentional state factor, specialized area, and the number of patients the entity has treated. We would like to set the following parameters: lack of expertise factor, probability of response for answer, and probability of response, and how quickly the entity will respond, bc_inc_fac, and $Init$. These parameter values are used to determine the bother cost of each entity.

The current patient we picked at step 1 has a medical what classified as Cardio. The specialized area of e_1 is $Cardio$, which is what the current patient requires, but e_1 has treated only 7 patients which have the same medical problem. Thus, we consider this entity as Med person for the level of expertise factor. In case of e_2, his specialized area is the same as the problem the current patient has and the number of patients is 100, which represents his expertise for the medical problem. Therefore, the level of expertise factor becomes Low (lack of expertise). However, e_3 and e_4 are assessed as entities whose lack of expertise factor is $High$ because they do not have any specialized area for the specific medical problem the current patient has.

Now we can obtain the user unwillingness value by adding the value of attentional state factor to the value of lack of expertise factor as we specified in formula 3.5. With the user unwillingness factor, we define a value of probability of response for each case of "Yes," "No," and "Silence" according to the table below.

Entity	ASF	LEF	UUF	Yes	No	Silence
e_1	*Relaxed*	*Med*	*Med-Willing*	50%	30%	20%
e_2	*Relaxed*	*Low*	*Willing*	60%	20%	20%
e_3	*Busy*	*High*	*Unwilling*	20%	60%	20%
e_4	*Relaxed*	*High*	*Medium*	40%	40%	20%

We can also determine the probability of response of each entity based on his/her user unwillingness factor as shown in Table 3.5.

Entity	ASF	LEF	UUF	1 unit	2 unit	3 unit	4 unit
e_1	*Relaxed*	*Med*	*Med-Willing*	27%	23%	17%	13%
e_2	*Relaxed*	*Low*	*Willing*	33%	27%	13%	7%
e_3	*Busy*	*High*	*Unwilling*	7%	13%	27%	33%
e_4	*Relaxed*	*High*	*Medium*	20%	20%	20%	20%

We finally obtain the value of bc_inc_fac, the exponent used for the bother increasing function based on the willingness values as below and calculate another variable $Init$ by following the formulae introduced in Section 3.2.2.

Entity	UUF	bc_inc_fac	Init
e_1	$Med\text{-}Willing$	1	8.4375
e_2	$Willing$	0.75	5.625
e_3	$Unwilling$	1.25	28.125
e_4	$Medium$	1	11.25

where the calculations are:

- $Init(e_1) = User\ Unwillingness\ Factor \times Attention\ State\ Factor \times TOC\ BaseBotherCost = 0.75 \times 0.75 \times 15 = 8.4375$
- $Init(e_2) = 0.5 \times 0.75 \times 15 = 5.625$
- $Init(e_3) = 1.5 \times 1.25 \times 15 = 28.125$
- $Init(e_4) = 1 \times 0.75 \times 15 = 11.25$

(b) We update the task criticality(TC) of the current patient with different TC weights as time passes. TC increases by 10% if it is highly critical as shown in Table 3.3. The TC of the current patient is initially 90. If the strategy chain is e_3 - e_2 - e_1 - e_4, e_3 will meet the patient whose task criticality is 90. However, e_2 will meet the patient whose task criticality is 99, which is increased by 10%. The task criticality goes up to 108.9 and 119.79 for e_1 and e_4.

(c) The EQ values are also set with different EQ weights as time passes. According to Table 3.4, the EQ value of e_1 is weighted by zero so that the EQ value is the same as the initial value. However, the EQ value of e_2 becomes 165 because e_2 has expertise. On the other hand, the EQ values of e_3 and e_4 become 135 which is less than the initial value, 150, because they do not have any expertise for the medical problem of the patient. That is, e_3 and e_4 cannot treat the highly critical patient very well due to the lack of expertise for the specific medical problem of the patient.

	e_1	e_2	e_3	e_4
EQ	150	165	135	135

(d) We calculate the expected utility for each FTOC node for "Yes", SG node for "No", and Default node for "Silence" as we described in Section 3.1.3.

(e) After aggregating the expected utility value for each node as in formula 3.4 we evaluate the final expected utility value of the current strategy chain.

(f) We repeat step 3 and 4 in order to evaluate the expected utility value for each strategy chain and choose to execute the strategy with the highest expected utility. (Note that we would repeat from step 1 to 4 until there is no patient on the waiting list, in order to process all the patients.)

5. We ask the first entity in the strategy we have chosen to execute the query, "Can you take on this patient?". The entities are ordered on the optimal strategy chain which has been obtained at the step of strategy generation and evaluation.

   (a) First, we set values for parameters of each entity based on the profile of the current patient. There are parameters such as attentional state factor and the number of patients the medical expert treated so far for the specific medical problem.

       We would like to change the parameter of each entity based on the type of responses such as Yes, No, or Silence after asking him/her a question, "Can you come over and take the patient right now?".

       i. If the response from the entity is "Yes," we assign the entity to the patient. At this time, the attentional state of the entity is changed to $BUSY$. Then, we go to step 1 in order to treat the next patient[1].

       ii. If the response from the entity is "No," we simply go to step 3 in order to regenerate the strategy chain.

       iii. If there is no response from the entity and the entity was not the person positioned as a last one on the strategy chain, we simply ask the next entity and repeat step 5(a) through step 5(c). If the response from the entity is silence and the entity was the last person who was asked on the strategy chain, we go to step 3 in order to regenerate the strategy chain.

Table 4.7 shows the list of expected utility of strategies at the time the patient arrives in Scenario 1. It shows that the optimal strategy chain is e_2 - e_1 - e_4 - e_3 whose expected utility is 130.080393.

Note that as housekeeping, we update task criticality (TC) of the patients as time passes. As explained, we increase the TC of the current patient who is looking for a medical expert. Second, we increase the TC of the patients on the waiting list. Finally, we decrease the TC of the patients who are currently being treated by medical experts[2]. The task criticality of patients on the waiting list are increased by the increasing rate corresponding to the TC of each patient.

[1] If the entity answering "Yes" is the last person on the strategy chain, we end up the process for finding the best entity.

[2] We assume that patients medical experts are treating get better, even though it might get worse in real-life situations.

Table 4.7: Expected utility of strategies at the time the patient arrives

No.	Expected Utility (EU)	Strategy Chain
1	113.707732	e_1 - e_2 - e_3 - e_4
2	114.581042	e_1 - e_2 - e_4 - e_3
3	103.964326	e_1 - e_3 - e_2 - e_4
4	102.780795	e_1 - e_3 - e_4 - e_2
5	106.715640	e_1 - e_4 - e_3 - e_2
6	108.785957	e_1 - e_4 - e_2 - e_3
7	129.207084	e_2 - e_1 - e_3 - e_4
8	130.080393	e_2 - e_1 - e_4 - e_3
9	123.195320	e_2 - e_3 - e_1 - e_4
10	122.913160	e_2 - e_3 - e_4 - e_1
11	126.859259	e_2 - e_4 - e_3 - e_1
12	128.025983	e_2 - e_4 - e_1 - e_3
13	75.616428	e_3 - e_2 - e_1 - e_4
14	75.334268	e_3 - e_2 - e_4 - e_1
15	71.884786	e_3 - e_1 - e_2 - e_4
16	70.701255	e_3 - e_1 - e_4 - e_2
17	68.646845	e_3 - e_4 - e_1 - e_2
18	69.552732	e_3 - e_4 - e_2 - e_1
19	99.140234	e_4 - e_2 - e_3 - e_1
20	100.306958	e_4 - e_2 - e_1 - e_3
21	89.412599	e_4 - e_3 - e_2 - e_1
22	88.506712	e_4 - e_3 - e_1 - e_2
23	94.495967	e_4 - e_1 - e_3 - e_2
24	96.566284	e_4 - e_1 - e_2 - e_3

4.2.2 Scenario 2

In our example, there are currently five patients who are waiting for treatment and these are four medical experts in the emergency room. The FCA tries to search for the right doctor for the current patient as we have seen in Scenario 1. Below are profiles of medical experts. The status of each entity is the same as in Scenario 1 except the attentional state of e_2 is now $BUSY$.

Entity	ASF
e_1	*Relaxed*
e_2	*Busy*
e_3	*Busy*
e_4	*Relaxed*

Entity	Specialized Area	Number of Patients	LEF
e_1	*Cardio*	7	*Med*
e_2	*Cardio*	100	*Low*
e_3	*Cardio*	0	*High*
e_4	*Cardio*	0	*High*

Entity	Specialized Area	Number of Patients	LEF
e_1	*Neuro*	0	*High*
e_2	*Neuro*	0	*High*
e_3	*Neuro*	15	*Med*
e_4	*Neuro*	120	*Low*

1. The FCA chooses the most serious patient from the waiting list as described in Scenario 1. Below is a table where you can find a waiting list in our scenario. Our system chooses p_2 since the task criticality of p_2 is highest among patients. Since p_2's task criticality is greater than 80 according to the table 3.3, he is assessed as highly critical.

No.	Patient	Medical Problem	Task Criticality
1	p_1	*Cardio*	70
2	p_2	*Cardio*	90
3	p_3	*Neuro*	63
4	p_4	*Cardio*	82
5	p_5	*Neuro*	70

2. Then, the waiting list is updated by eliminating the selected patient which has been assessed as the most critical patient. Thus, the number of patients remaining becomes four. Below is a table where you can find a updated waiting list in our scenario.

No.	Patient	Medical Problem	Task Criticality
1	p_1	*Cardio*	70
2	p_3	*Cardio*	63
3	p_4	*Cardio*	82
4	p_5	*Neuro*	70

3. We generated strategies by following the process introduced in Chapter 3. In our scenario, 4! strategies are generated since there are four entities attending in our scenario.

4. We evaluate the expected utility of each strategy generated in step 3. There are several steps to evaluate the expected utility with following:

   (a) We set values for parameters of each entity based on the profile of the current patient. We already know some information about each entity, such as attentional state factor, specialized area, and the number of patients the entity has treated. We would like to set the following parameters: lack of expertise factor, probability of response for answer, and probability of response how quickly the entity will response, *bc_inc_fac*, and $Init$. These parameter values are used to determine the bother cost of each entity.

   Now we can obtain user unwillingness factor by adding attentional state factor to lack of expertise factor and have a slightly new table, as below. With user unwillingness factor, we set a value of probability of response for each cases such as "Yes," "No," and "Silence" according to the Table 4.1.

Entity	ASF	LEF	UUF	Yes	No	Silence
e_1	*Relaxed*	*Med*	*Med-Willing*	50%	30%	20%
e_2	*Busy*	*Low*	*Medium*	40%	40%	20%
e_3	*Busy*	*High*	*Unwilling*	20%	60%	20%
e_4	*Relaxed*	*High*	*Medium*	40%	40%	20%

   We can also find the probability of response, which represents how quickly the entity will respond.

Entity	ASF	LEF	UUF	1 unit	2 unit	3 unit	4 unit
e_1	*Relaxed*	*Med*	*Med-Willing*	27%	23%	17%	13%
e_2	*Busy*	*Low*	*Medium*	20%	20%	20%	20%
e_3	*Busy*	*High*	*Unwilling*	7%	13%	27%	33%
e_4	*Relaxed*	*High*	*Medium*	20%	20%	20%	20%

The values of bc_inc_fac and $Init$ are as below:

Entity	UUF	bc_inc_fac	Init
e_1	*Med-Willing*	1	8.4375
e_2	*Medium*	1	18.75
e_3	*Unwilling*	1.25	28.125
e_4	*Medium*	1	11.25

where the calculations are:

- $Init(e_1) = User\ Unwillingness\ Factor \times Attention\ State\ Factor$ $\times\ TOC\ BaseBotherCost = 0.75 \times 0.75 \times 15 = 8.4375$
- $Init(e_2) = 1 \times 1.25 \times 15 = 18.75$
- $Init(e_3) = 1.5 \times 1.25 \times 15 = 28.125$
- $Init(e_4) = 1 \times 0.75 \times 15 = 11.25$

(b) We update the task criticality(TC) of the current patient with different TC weights as time passes. TC increases by 10% if it is highly critical as shown in Table 3.3. TC of the current patient is initially 90. If the strategy chain is e_3 - e_2 - e_1 - e_4, e_3 will meet the patient whose task criticality is 90. However, e_2 will meet the patient whose task criticality is 99, which is increased by 10%. The task criticality goes up into 108.9 and 119.79.

(c) The EQ values are also set with different EQ weights as time passes. According to Table 3.4, the EQ value of e_1 is weighted by zero so that the EQ value is same as the initial value. However, the EQ value of e_2 becomes 165 because e_2 has expertise. On the other hand, the EQ values of e_3 and e_4 become 135 which is less than the initial value, 150 because they do not have any expertise for the medical problem of the patient. That is, e_3 and e_4 cannot treat the highly critical patient very well due to the lack of expertise for the specific medical problem of the patient.

	e_1	e_2	e_3	e_4
EQ	150	165	135	135

(d) We calculate the expected utility for each FTOC node for “Yes”, SG node for “No”, and Default node for “Silence” as we described in Section 3.1.3.

(e) After aggregating the expected utility value for each node, we evaluate the final expected utility value of the current strategy chain.

(f) We repeat step 3 and 4 in order to evaluate the expected utility value for each strategy chain.

(g) We repeat from step 1 to 4 until there is no patient on the waiting list

5. We ask the entity with a query. The entities are ordered on the optimal strategy chain which has been obtained at the step of strategy generation and evaluation.

Table 4.8 shows the list of expected utility of strategy chain at the time the patient arrives in Scenario 2. We find that the optimal strategy chain is e_1 - e_2 - e_4 - e_3 whose expected utility is 110.031364.

4.2.3 Scenario 3

In this example, there are currently five patients waiting for treatment and four medical experts in the emergency room. The FCA tries to search for the right doctor for the current patient as we have seen in Scenario 1. Below are profiles of medical experts.

Entity	ASF
e_1	*Busy*
e_2	*Busy*
e_3	*Busy*
e_4	*Relaxed*

Entity	Specialized Area	Number of Patients	LEF
e_1	*Cardio*	7	*Med*
e_2	*Cardio*	100	*Low*
e_3	*Cardio*	2	*High*
e_4	*Cardio*	2	*High*

Entity	Specialized Area	Number of Patients	LEF
e_1	*Neuro*	2	*High*
e_2	*Neuro*	2	*High*
e_3	*Neuro*	15	*Med*
e_4	*Neuro*	120	*Low*

Table 4.8: Expected utility of strategies at the time the patient arrives

No.	Expected Utility (EU)	Strategy Chain
1	109.158054	e_1 - e_2 - e_3 - e_4
2	110.031364	e_1 - e_2 - e_4 - e_3
3	103.040701	e_1 - e_3 - e_2 - e_4
4	102.608701	e_1 - e_3 - e_4 - e_2
5	106.566054	e_1 - e_4 - e_3 - e_2
6	107.871364	e_1 - e_4 - e_2 - e_3
7	107.738381	e_2 - e_1 - e_3 - e_4
8	108.611691	e_2 - e_1 - e_4 - e_3
9	101.726617	e_2 - e_3 - e_1 - e_4
10	101.444457	e_2 - e_3 - e_4 - e_1
11	105.390556	e_2 - e_4 - e_3 - e_1
12	106.557280	e_2 - e_4 - e_1 - e_3
13	71.066750	e_3 - e_2 - e_1 - e_4
14	70.784590	e_3 - e_2 - e_4 - e_1
15	70.961161	e_3 - e_1 - e_2 - e_4
16	70.529161	e_3 - e_1 - e_4 - e_2
17	68.474750	e_3 - e_4 - e_1 - e_2
18	68.624590	e_3 - e_4 - e_2 - e_1
19	94.590556	e_4 - e_2 - e_3 - e_1
20	95.757280	e_4 - e_2 - e_1 - e_3
21	88.484457	e_4 - e_3 - e_2 - e_1
22	88.334617	e_4 - e_3 - e_1 - e_2
23	94.346381	e_4 - e_1 - e_3 - e_2
24	95.651691	e_4 - e_1 - e_2 - e_3

1. The FCA identifies the most serious patient from the waiting list. Below is a waiting list in this scenario. Our system chooses p_2 since the task criticality of p_2 is highest among patients. Since p_2's task criticality is less than 10 according to the table 3.3, he is assessed as low critical, while the most critical patient was assessed as high critical in Scenario 1 and 2.

No.	Patient	Medical Problem	Task Criticality
1	p_1	$Cardio$	7
2	p_2	$Cardio$	9
3	p_3	$Neuro$	6
4	p_4	$Cardio$	8
5	p_5	$Neuro$	7

2. Then, the waiting list is updated by eliminating the selected patient which has been assessed as the most critical patient. Thus, the number of patients remained becomes four. Below is a table where you can find a updated waiting list in our scenario.

No.	Patient	Medical Problem	Task Criticality
1	p_1	$Cardio$	7
2	p_3	$Neuro$	6
3	p_4	$Cardios$	8
4	p_5	$Neuro$	7

3. We generated strategies by following the process introduced in Chapter 3. In our scenario, 4! strategies are generated since there are four entities attending in our scenario.

4. We evaluate the expected utility of each strategy generated in step 3. There are several steps to evaluate the expected utility as follows:

   (a) We set values for parameters of each entity based on the profile of the current patient. We already know some information about each entity, such as attentional state factor, specialized area, and the number of patients the entity has treated. We would like to set the following parameters: lack of expertise factor, probability of response for answer, and probability of response for how quickly the entity will response, bc_inc_fac, and $Init$. These parameter values are used to determine the bother cost of each entity.

   Now we can obtain user unwillingness factor by adding attentional state factor to lack of expertise factor as we described in Section 3.2.2. With user unwillingness factor, we set values of probability of response for each case such as "Yes," "No," and "Silence" according to the Table 4.1.

Entity	ASF	LEF	UUF	Yes	No	Silence
e_1	*Busy*	*Med*	*Med-Unwilling*	30%	50%	20%
e_2	*Busy*	*Low*	*Medium*	40%	40%	20%
e_3	*Busy*	*High*	*Unwilling*	20%	60%	20%
e_4	*Relaxed*	*High*	*Medium*	40%	40%	20%

We can also find the probability of response, which represents how quickly the entity will response.

Entity	ASF	LEF	UUF	1 unit	2 unit	3 unit	4 unit
e_1	*Busy*	*Med*	*Med-Unwilling*	13%	17%	23%	27%
e_2	*Busy*	*Low*	*Medium*	20%	20%	20%	20%
e_3	*Busy*	*High*	*Unwilling*	7%	13%	27%	33%
e_4	*Relaxed*	*High*	*Medium*	20%	20%	20%	20%

The values of bc_inc_fac and $Init$ are as below.

Entity	UUF	bc_inc_fac	Init
e_1	*Med-Unwilling*	1	8.4375
e_2	*Medium*	1	18.75
e_3	*Unwilling*	1.25	28.125
e_4	*Medium*	1	11.25

where the calculations are:

- $Init(e_1) = User\ Unwillingness\ Factor \times Attention\ State\ Factor$ $\times\ TOC\ BaseBotherCost = 1.25 \times 1.25 \times 15 = 23.4375$
- $Init(e_2) = 1 \times 1.25 \times 15 = 18.75$
- $Init(e_3) = 1.5 \times 1.25 \times 15 = 28.125$
- $Init(e_4) = 1 \times 0.75 \times 15 = 11.25$

(b) We update the task criticality (TC) of the current patient with different TC weights as time passes. TC increases by 2% if it is highly critical as shown in Table 3.3. TC of the current patient is initially 9. If the strategy chain is e_2 - e_3 - e_4 - e_1, e_2 will meet the patient whose task criticality is 90. However, e_3 will meet the patient whose task criticality is 9.18, which is increased by 2%. The task criticality goes up into 9.3636 and 9.5509.

(c) The EQ values are also set with different EQ weights as time passes. According to the table 3.4, however, the EQ value of any entities is not weighed because the task criticality of the patient is low. That is, the EQ value of e_1, e_2, e_3, and e_4 is 150 which is the initial value. Therefore, every entity has equally likely ability to treat the patient.

	e_1	e_2	e_3	e_4
EQ	150	150	150	150

(d) We calculate the expected utility for each FTOC node for "Yes", SG node for "No", and Default node for Silence as we described in Section 3.1.3.

(e) After aggregating the expected utility value for each node, we evaluate the final expected utility value of the current strategy chain.

(f) We repeat step 3 and 4 in order to evaluate the expected utility value for each strategy chain.

(g) We repeat from step 1 to 4 until there is no patient on the waiting list

5. We ask the entity with a query. The entities are ordered on the optimal strategy chain which has been obtained at the step of strategy generation and evaluation.

Table 4.9 displays the list of expected utility of strategy chain at the time the patient arrives in Scenario 3. We find that the optimal strategy chain is e_4 - e_1 - e_2 - e_3 whose expected utility is 105.848998.

4.3 Executing a Strategy

This example revisits Scenarios 1 in Section 4.2, in order to illustrate the possible outcomes in executing a particular strategy. In particular, we examine a few distinct cases, showing different possible responses from the experts to whom decision making control is transferred, within the overall strategy.

Case 1. The first clinical assistant(FCA) obtained the optimal strategy chain and asked the first expert in strategy a query, "Can you help with the current patient?". The expert provided a "Yes" response to the FCA in 3 unit time and the FCA assigned the patient to the expert. The expert that was relaxed became busy after taking the patient.

Case 2. The FCA obtained the optimal strategy chain and asked the first expert in strategy a query, "Can you help with the current patient?". The expert did not provide

Table 4.9: Expected utility of strategies at the time the patient arrives

No.	Expected Utility (EU)	Strategy Chain
1	92.028054	e_1 - e_2 - e_3 - e_4
2	92.901364	e_1 - e_2 - e_4 - e_3
3	90.230701	e_1 - e_3 - e_2 - e_4
4	90.182701	e_1 - e_3 - e_4 - e_2
5	94.140054	e_1 - e_4 - e_3 - e_2
6	95.061364	e_1 - e_4 - e_2 - e_3
7	94.415688	e_2 - e_1 - e_3 - e_4
8	95.288998	e_2 - e_1 - e_4 - e_3
9	92.469555	e_2 - e_3 - e_1 - e_4
10	92.230492	e_2 - e_3 - e_4 - e_1
11	96.199099	e_2 - e_4 - e_3 - e_1
12	97.300218	e_2 - e_4 - e_1 - e_3
13	81.009687	e_3 - e_2 - e_1 - e_4
14	80.770625	e_3 - e_2 - e_4 - e_1
15	81.158468	e_3 - e_1 - e_2 - e_4
16	81.110468	e_3 - e_1 - e_4 - e_2
17	83.121687	e_3 - e_4 - e_1 - e_2
18	82.930625	e_3 - e_4 - e_2 - e_1
19	104.599099	e_4 - e_2 - e_3 - e_1
20	105.700218	e_4 - e_2 - e_1 - e_3
21	102.790492	e_4 - e_3 - e_2 - e_1
22	102.981555	e_4 - e_3 - e_1 - e_2
23	104.927688	e_4 - e_1 - e_3 - e_2
24	105.848998	e_4 - e_1 - e_2 - e_3

any response even after 5 unit time had passed. Our system classified him as Silence and suggested the FCA to ask the second best expert. The expert provided a "Yes" response to the FCA in 2 unit time and the FCA assigned the patient to the expert.

Case 3. The FCA obtained the optimal strategy chain and asked the first expert in strategy a query, "Can you help with the current patient?". The expert did not provide any response even after 5 unit time had passed. Thus, the FCA tried to ask the second best expert but he also did not give any response for 5 unit time. The FCA asked the third expert in the strategy. She said "Yes" immediately The patient is finally assigned to the third expert. This might happen, for instance, if the first two experts, thought to be *Relaxed*, actually became *Busy* and the third expert, thought to be *Busy*, became free by the time she was asked.

Case 4. The FCA obtained the optimal strategy chain and asked the first expert in strategy a query, "Can you help with the current patient?". The expert did not provide any response even after 5 unit time had passed. Thus, the FCA tried to ask the second best expert but he also did not give any response for 5 unit time. The FCA asked the third and fourth experts in the strategy but they also provided only Silence. Meanwhile, time has been passed and the expert who had the greatest quality of decision was no longer busy because the patient left the expert as she got better due to the treatment by that expert. Strategy regeneration would put this expert as the first entity to whom control was transferred and then we might be back in case 1 above.

Chapter 5

Validation

5.1 Experimental Setup

Our validation measures performance of our model reflecting dynamic and time critical aspects. Our simulation used Matlab (R2010a) on a machine with the following settings: AMD athlon(tm) 64 X2 Dual, Core Processor 5600+, 2.91 GHz, and 3.25 GB of RAM. In the setting of our validation simulating hospital emergency scenarios, there are four entities on the entity list and five patients on the waiting list. Below are profiles of patients and entities.

No.	Patient	Medical Problem	Task Criticality
1	p_1	*Cardio*	70
2	p_2	*Cardio*	90
3	p_3	*Neuro*	63
4	p_4	*Cardio*	82
5	p_5	*Neuro*	70

Entity	ASF	Specialized Area	Number of Patients
e_1	*Relaxed*	*Cardio*	7
e_2	*Relaxed*	*Cardio*	100
e_3	*Relaxed*	*Neuro*	15
e_4	*Relaxed*	*Neuro*	120
$*e_5$	*Relaxed*	*Neuro*	240
$*e_6$	*Relaxed*	*Cardio*	98

Note that entities e_5 and e_6 on the entity profiles were not included during the simulation with 4 entities case.

Every patient has a task criticality for his/her specific medical problem, and the task criticality of each patient is changed dynamically as time progresses.

Our simulation considers all the patients in the emergency room, beginning with the most critical patient first and then sequentially processing the remaining patients on the waiting list, always processing the most critical patient first.

5.2 Process

The first clinical assistant picks up the most serious patient from the waiting list. Then, the waiting list is updated by eliminating the selected patient which has been assessed as the most critical patient. Thus, the number of patients remaining becomes four.

Strategies are generated by following the process introduced in Chapter 3. In our validation, 4! strategies are generated since there are four entities available in our scenario. We evaluate the expected utility of each strategy and choose the one with the optimal utility. There are several steps to evaluate the expected utility with following.

We set values of parameters of each entity based on the profile of the current patient. We already know some information of each entity such as attentional state factor, specialized area, and the number of patients the entity has treated. Then, we set the following parameters: lack of expertise factor, probability of response for answer, and probability of response for how quickly the entity will respond, bc_inc_fac, and $Init$. Those parameter values affect to determine the bother cost of each entity.

Obtain Parameter Values We obtain lack of expertise factor of the entity based on the medical problem of the patient and the history of the entity.

We obtain user unwillingness factor by adding attentional state factor to lack of expertise factor. With user unwillingness factor, we determine a value of probability of response for each case such as "Yes," "No," and "Silence" according to the Table 3.5.

We can also find the probability of response timing, which represents how quickly the entity will respond.

We finally obtain the value of bc_inc_fac and calculate another variable $Init$ by following the formulae introduced in Section 3.2.2.

Update task criticality We update the task criticality(TC) of the current patient with the increasing rate based on how critical he/she is as time passes according to Table 3.3.

Setting EQ The EQ values are also set with different EQ weights as time passes according to Table 3.4.

Calculation of EU We calculate the expected utility for each FTOC node for "Yes", SG node for "No", and Default node for "Silence". After aggregating the expected utility value for each node, we evaluate the final expected utility value of the current strategy chain. This results in the strategy that is best to execute.

5.3 Execution

Obtain the answer and response time
We ask the best entity on the best strategy chain. He/she would have a function of probability of response as shown in Table 3.5 based on his/her willingness.

The entity will answer "Yes", "No" or "Silence" based on probability of response as shown in Table 3.5. With the current user unwillingness factor, we get information about when the entity will answer the question and which answer will be given. For example, if the user unwillingness factor is Willing, the probability to answer "Yes" is 60%. We use a uniform distribution. In other words, our simulation generates a random number between 0 and 10. If the random number is between 0 and 6, our simulation considers that the answer is "Yes". If the random number is between 6 and 8, the answer is considered as "No". Otherwise, we consider that the response from the entity is silence.

After obtaining the type of answer, we use another uniform distribution to simulate when the entity responded to the question. If the answer is silence, the response time becomes 5 unit time. Otherwise, we generate a random number between 0 and 80 and see which number is generated. If the entity is a $Willing$ person, the entity would give a response which may be either "Yes" or "No" in 1 unit time with the probability of 33% as designed in Table 3.6. Thus, if the random number generated from the uniform distribution is between 0 and 33, we consider that the entity responded in 1 unit time after being asked. Also, the entity a has probability of 27% to respond in 2 unit time, 13% in 3 unit time, and 7% in 4 unit time. Therefore, if the random number drawn from the uniform distribution between 0 and 80 [1] is between 33 and 60, 2 unit time is given to the entity as response

[1] We modeled that the probability of response for the case of silence as 20% for any type of user

time. If it is between 60 and 73, 3 unit time is given. Otherwise, 4 unit time is given as the entity response time to the question.

Update task criticality of the current patient
We learned about when the entity will respond from the last step. Since the current patient should wait for a response from the entity, we increase the task criticality of the current patient during the response time. Examples are as follows:

1. If the expected response is "Yes" in 3 unit time and the current task criticality of the patient is 85, the expected task criticality in 3 unit time is calculated as follows:

   - In 1 unit time,
     $TC = current\ TC + current\ TC \times Increasing\ Rate = 85 + 85 \times 0.1 = 93.5$
   - In 2 unit time,
     $TC = 93.5 + 93.5 \times 0.1 = 102.85$
   - In 3 unit time,
     $TC = 102.85 + 102.85 \times 0.1 = 113.135$

   If the task criticality of the a patient increased over 100, we model this as a problem patient. In this case, the patient becomes dead before he/she gets a response from the entity. We put the patient on the Dead List.

2. If the expected response is "Yes" in 2 unit time, and the current task criticality of the patient is 78, the expected task criticality in 2 unit time is calculated as follows:

   - After 1 unit time,
     $TC = current\ TC + current\ TC \times Increasing\ Rate = 78 + 78 \times 0.05 = 81.9$
   - After 2 unit time,
     $TC = 81.9 + 81.9 \times 0.1 = 90.09$

   After 1 unit time, 5% was applied as an increasing rate as task criticality of the patient is $Medium$, but 10% was applied in 2 unit time since task criticality of the patient became $High$. In this case, the patient is still alive before he/she get a response from the entity. Since the entity's answer to the question is "Yes", the patient will be taken by the entity.

unwillingness factor. In other words, the probability of response for the case of "Yes" and "No" is always 80%. Thus, we arranged the range of the distribution to be between 0 and 80 for convenience in our calculation.

3. If the expected response is "No" in 2 unit time, and the current task criticality of the patient is 78, the expected task criticality in 2 unit time is calculated as follows:

   - In 1 unit time,
     $TC = current\ TC + current\ TC \times Increasing\ Rate = 78 + 78 \times 0.05 = 81.9$
   - In 2 unit time,
     $TC = 81.9 + 81.9 \times 0.1 = 90.09$

   After 1 unit time, 5% was applied as an increasing rate as task criticality of the patient was *Medium*, but 10% was applied in 2 unit time since task criticality of of the patient became *High*. In this case, the patient is still alive before he/she get a response from the entity. However, the patient still need to wait for a while to see a doctor because the expected doctor answered "No". In our model, we regenerate a strategy chain reflecting current parameter values and repeat asking an entity on the new strategy. Meanwhile, task criticality of other patients on the waiting list is increased by the increasing rate corresponding to the task criticality of them.

4. If the expected response is silence, and the current task criticality of the patient is 45, we need to calculate the the expected task criticality in 5 unit time.

   - In 1 unit time,
     $TC = current\ TC + current\ TC \times Increasing\ Rate = 45 + 45 \times 0.05 = 47.25$
   - In 2 unit time,
     $TC = 47.25 + 47.25 \times 0.05 = 49.6125$
   - In 3 unit time,
     $TC = 49.6125 + 49.6125 \times 0.05 = 52.0931$
   - In 4 unit time,
     $TC = 52.0931 + 52.0931 \times 0.05 = 54.6978$
   - In 5 unit time,
     $TC = 54.6978 + 54.6978 \times 0.05 = 57.4327$

   5% was applied as an increasing rate as task criticality of the patient was *Medium*. In this case, the patient is still alive before he/she get a response from the entity. However, the patient still needs to wait for a while to see a doctor because the expected doctor answered nothing rather than "Yes". In our model, we ask the second best doctor on the strategy chain and wait for the response from the doctor. If there is no next doctor left, we regenerate a strategy chain as is done in the case of "No".

Check the status of patients on the waiting list
As time passes, task criticality of patients on the waiting list increases. If task criticality of any patients become more than 100, we consider that the patient is dead at that point. Our simulation first selects the patient whose task criticality is highest.

5.4 Simulations

5.4.1 Time Cost and Bother Cost

This validation measures performance of our model reflecting dynamic and time critical aspects by comparing it with one that is missing the calculation of bother cost. In the setting of our validation simulating hospital emergency scenarios, there are four entities on the entity list and five patients on the waiting list.

Every patient has a task criticality for his/her specific medical problem, and the task criticality of each patient is changed dynamically as time progresses. Our simulation first selects the patient whose task criticality is highest.

The number of patients the entity has treated so far is used to determine weather the entity has experience for a particular medical problem for the profile recorded. For example, we consider the entity as an expert if he has treated more than 100 patients for the specific medical problem. Otherwise, the entity is considered as novice for the specific medical problem.

We then obtain a strategy chain by calculating formulae (Equation 3.4, 3.9, 3.10) reflecting our model based on the patient's profile (medical problem and criticality). After choosing a entity in the chain, we ask him/her to treat the current patient and update the criticality of patients who have been treated by entities, as well as those remaining on the waiting list. If a patient has not been attended to (i.e. no doctor has replied "yes"), the task criticality of the patient increases as time passes. If the task criticality of the a patient increased over 100, we model this as a problem patient. When there are no more patients on the waiting list, we finally count the number of problem patients. By comparing the number of problem patients simulated by our model with bother cost and without bother cost, we can validate whether our model reflects dynamic and time critical domains effectively.

Figure 5.1 illustrates the distribution generated by our model with Bother Cost and without Bother Cost. The graph on the left represents the case of 4 entities and 5 patients

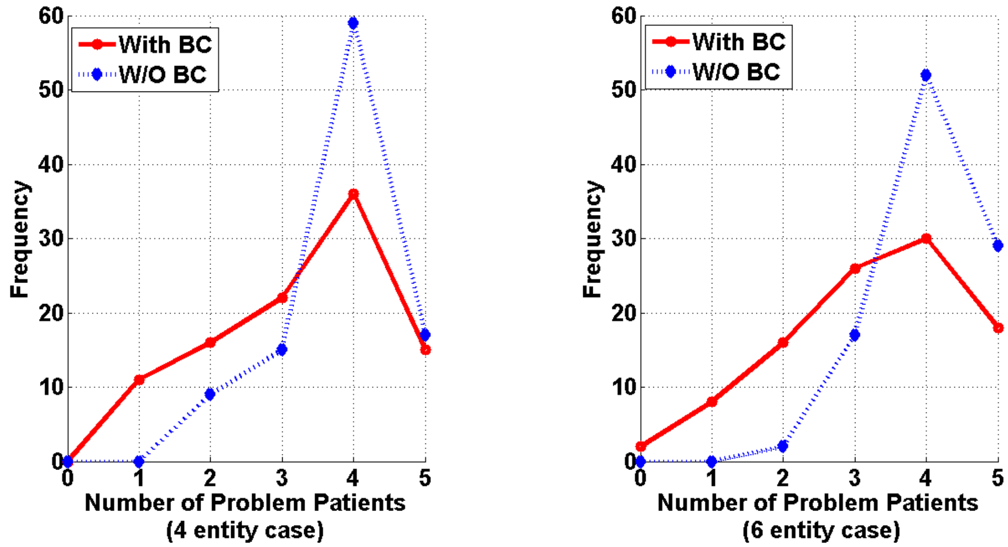

Figure 5.1: Our model with and without Bother Cost

and one on the right represents the case of 6 entities and 5 patients. The x-axis of each graph denotes the number of problem patients, and the y-axis the frequency of each value on x-axis after running our simulation 100 times. The solid line represents the version including Bother Cost, and the dotted line represents the version excluding Bother Cost. In Figure 5.1, we can find the peak of the dotted line located in a higher position than the peak of the line at 4 on the x-axis and inclined to the right. This implies that there have been more problem patients during simulations with the version without Bother Cost (dotted line) than one with Bother Cost (solid line). In other words, the version calculating Bother Cost outperforms the one which does not calculate Bother Cost by comparing the number of problem patients on the graphs.

5.4.2 Strategy Regeneration

In this experiment, we compare the version with a SG node for strategy regeneration to the one without the SG node. As shown in Section 3.1.3, there is a SG node where a new strategy chain is generated if the response from the entity is "No" to reflect the aspect of real-time and dynamic environments. For the version excluding the SG node, we simply moved to the next world and asked the next entity instead of strategy regeneration.

Figure 5.2 illustrates the distribution generated by our model with strategy regeneration

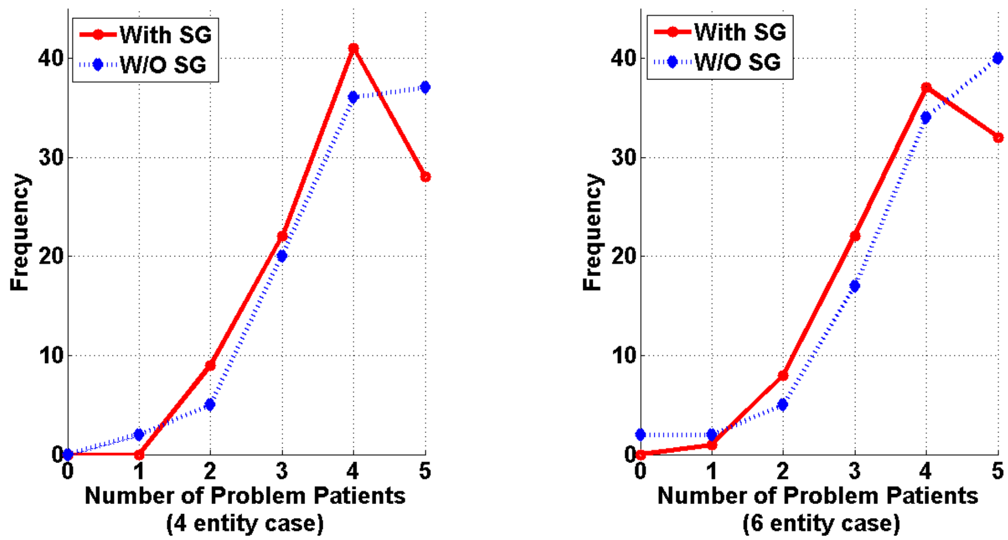

Figure 5.2: Our model with and without a SG node

and without strategy regeneration. The graph on the left represents the case of 4 entities and 5 patients and one on the right represents the case of 6 entities and 5 patients. The x-axis of each graph denotes the number of problem patients, and the y-axis the frequency of each value on x-axis after running our simulation 100 times. The solid line represents the version including a SG node, and the dotted line represents the version excluding the SG node.

In Figure 5.2, we can find the peak of the solid line at 4 on the x-axis and inclined to the right. However, the peak of the dotted line is spotted at 5 on the x-axis. This implies that 4 problem patients are mostly found under the version with strategy regeneration but 5 problem patients under the version without strategy regeneration. In other words, the version including a SG node outperforms the one which does not regenerate a strategy chain.

5.4.3 Task Criticality

In this experiment, we compare the version with weights by task criticality of the patients to the one without weights. The expected quality of decision of each entity is determined by his/her lack of expertise factor as presented in Formula 3.10. In this section, we compare

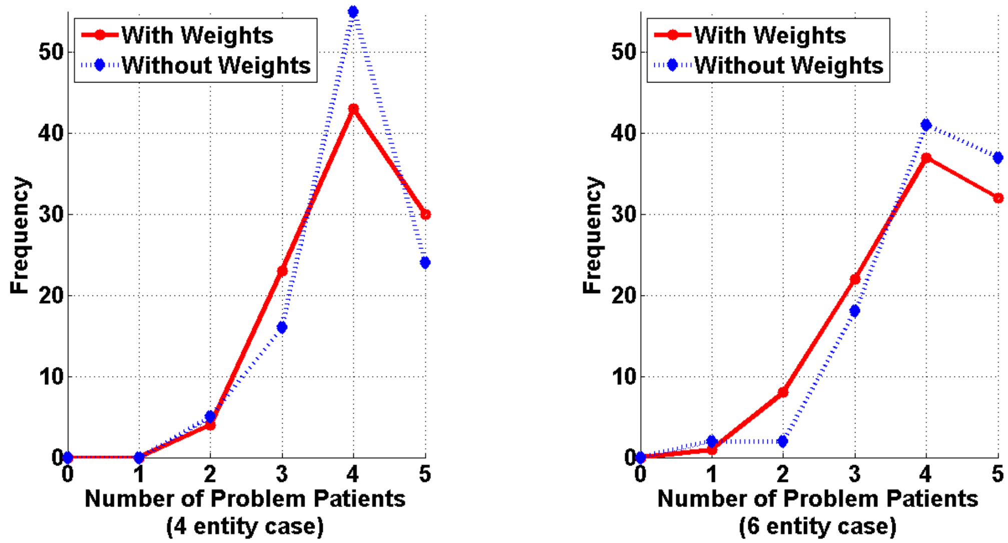

Figure 5.3: Our model with and without weights

the version with weights and the one without weights. The version without weights implies that every entity has equal expected quality of decision.

Figure 5.3 illustrates the distribution generated by our model with strategy regeneration and without weights. The graph on the left represents the case of 4 entities and 5 patients and one on the right represents the case of 6 entities and 5 patients. The x-axis of each graph denotes the number of problem patients, and the y-axis the frequency of each value on x-axis after running our simulation 100 times. The solid line represents the version reflecting weights, and the dotted line represents the version excluding weights. In Figure 5.3, we can find the peak of the dotted line located in a higher position than the peak of the line at 4 on the x-axis and inclined to the right. This implies that there have been more problem patients during simulations with the version without weights (dotted line) than one reflecting weights (solid line). In other words, the version with weights outperforms the one which does not reflect weights by comparing the number of problem patients on the graphs.

Table 5.1: Profiles of entities at the time the patient arrives

	e_1	e_2	e_3	e_4
Attentional State Factor	Relaxed	Busy	Busy	Relaxed
Lack-of-expertise Factor	Low	High	Low	High

5.5 Level of Expertise

We illustrate the value of the $Lack\_of\_Expertise$ factor through an example. Table 5.1 displays the Attentional State and Lack of Expertise of each of the four entities, at the beginning.

5.5.1 With Level of Expertise

In our example, a patient has just arrived at the emergency room. The first clinical assistant (FCA) assesses him as highly critical, whose task criticality is 85 for the medical problem on his Neuro. The FCA tries to search for the right doctor for the current patient. Our system checks the profile of each doctor and begins finding out the optimal strategy by calculating an expected utility for each generated strategy. Since there are four doctors, we obtain 4! strategies. By evaluating each strategy, we obtain 24 expected utility values for each strategy. The greatest expected utility is $EU(s) = 110.03$ whose strategy chain is e_1 - e_2 - e_4 - e_3.

The strategies that choose to ask e_4 first do not have high EU values, even though the expert is not Busy and can attend to the patient. This is because the High Criticality of the patient has raised the weight of the EQ value in the calculation. The maximal EU of a strategy that asks e_4 first is 95.75 as represented in Table 5.2. Likewise, strategies that select e_2 first have very low EU values, as this expert is both Busy and with High $Lack\_of\_Expertise\_Factor$.

5.5.2 Without Level of Expertise

Each entity has its own $Lack\_of\_Expertise\_Factor$ of either $High$, Med or Low. However, we set $Lack_of_Expertise_Factor$s of all the entities into Med. The greatest expected utility is $EU(s) = 110.37$ whose strategy chains are e_1 - e_4 - e_3 - e_2, e_1 - e_4 - e_2 - e_3, e_4 - e_1 - e_3 - e_2, and e_4 - e_1 - e_2 - e_3 as shown in Table 5.3. We find that there are four different

Table 5.2: Expected utility of strategies at the time the patient arrives

No.	Expected Utility (EU)	Strategy Chain
1	109.158054	e_1 - e_2 - e_3 - e_4
2	110.031364	e_1 - e_2 - e_4 - e_3
3	103.040701	e_1 - e_3 - e_2 - e_4
4	102.608701	e_1 - e_3 - e_4 - e_2
5	106.566054	e_1 - e_4 - e_3 - e_2
6	107.871364	e_1 - e_4 - e_2 - e_3
7	107.738381	e_2 - e_1 - e_3 - e_4
8	108.611691	e_2 - e_1 - e_4 - e_3
9	101.726617	e_2 - e_3 - e_1 - e_4
10	101.444457	e_2 - e_3 - e_4 - e_1
11	105.390556	e_2 - e_4 - e_3 - e_1
12	106.557280	e_2 - e_4 - e_1 - e_3
13	71.066750	e_3 - e_2 - e_1 - e_4
14	70.784590	e_3 - e_2 - e_4 - e_1
15	70.961161	e_3 - e_1 - e_2 - e_4
16	70.529161	e_3 - e_1 - e_4 - e_2
17	68.474750	e_3 - e_4 - e_1 - e_2
18	68.624590	e_3 - e_4 - e_2 - e_1
19	94.590556	e_4 - e_2 - e_3 - e_1
20	95.757280	e_4 - e_2 - e_1 - e_3
21	88.484457	e_4 - e_3 - e_2 - e_1
22	88.334617	e_4 - e_3 - e_1 - e_2
23	94.346381	e_4 - e_1 - e_3 - e_2
24	95.651691	e_4 - e_1 - e_2 - e_3

strategy chains which were generated with the same EU because there is indifferent between $High$ expertise medical experts and Low expertise experts. For example, e_1 and e_4 are exchangeable on the strategy chain since their user unwillingness factor is same each other.

However, if e_4 is chosen as the first entity, our patient would not be treated properly because e_4 has no experience for the medical problem.

Table 5.3: Expected utility of strategies at the time the patient arrives

No.	Expected Utility (EU)	Strategy Chain
1	108.657664	e_1 - e_2 - e_3 - e_4
2	108.700761	e_1 - e_2 - e_4 - e_3
3	108.657664	e_1 - e_3 - e_2 - e_4
4	108.700761	e_1 - e_3 - e_4 - e_2
5	110.366391	e_1 - e_4 - e_3 - e_2
6	110.366391	e_1 - e_4 - e_2 - e_3
7	95.250357	e_2 - e_1 - e_3 - e_4
8	95.293454	e_2 - e_1 - e_4 - e_3
9	93.584726	e_2 - e_3 - e_1 - e_4
10	93.584726	e_2 - e_3 - e_4 - e_1
11	95.250357	e_2 - e_4 - e_3 - e_1
12	95.293454	e_2 - e_4 - e_1 - e_3
13	93.584726	e_3 - e_2 - e_1 - e_4
14	93.584726	e_3 - e_2 - e_4 - e_1
15	95.250357	e_3 - e_1 - e_2 - e_4
16	95.293454	e_3 - e_1 - e_4 - e_2
17	95.293454	e_3 - e_4 - e_1 - e_2
18	95.250357	e_3 - e_4 - e_2 - e_1
19	108.657664	e_4 - e_2 - e_3 - e_1
20	108.700761	e_4 - e_2 - e_1 - e_3
21	108.657664	e_4 - e_3 - e_2 - e_1
22	108.700761	e_4 - e_3 - e_1 - e_2
23	110.366391	e_4 - e_1 - e_3 - e_2
24	110.366391	e_4 - e_1 - e_2 - e_3

Chapter 6

Discussion and Conclusions

In this chapter, we begin by presenting future work, commenting further on how to make the calculations of our framework more complex. We then include a contrast with other related work, including other efforts on designing mixed-initiative systems and other research on reasoning about interaction with users, sensitive to bother. We end with a summary of the contributions of the thesis.

6.1 Future Work

6.1.1 Sensor and Learning Techniques

One valuable topic for future work is how best to set the various parameter values. Some parameter values could possibly obtained from sensors attached to the medical experts being modeled. We would need to determine how often the sensor check the status of the doctors and send information to the system. If the status has been checked by sensor and sent to the system frequently, our system could reflect the aspect of dynamic and real-time settings more effectively. However, the heavy loads might be required to work out frequently.

The hSITE project ([22]) aims to integrate sensor readings into the overall decision making in hospital environments and to integrate as well an effective networking of the various devices used within the hospital. One could imagine, for instance, being able to assess the attentional state of the medical experts based on devices which register patient status. In addition, the time and location of the medical experts could be known and this could be another influence in determining the expected quality of decision, if asking a particular expert (e.g. experts who are very far away may be less able to quickly assist a critical patient). Integrating sensor data into the determination of parameter values is in general an interesting topic for future research.

We may also use learning techniques to obtain parameter values instead of using information from the sensor. The parameter values obtained by using learning techniques might not reflect the current situation as effectively as using the sensor, but it may give the system less burden to use the learning techniques.

In order to acquire the values of some of the existing parameters in the model described in Chapters 3 and 4, we are interested in using machine learning methods. We can classify different machine learning algorithms, based on the desired outcome of the algorithm as follows [25]. First, supervised learning is for learning a function from training data. The training data consist of pairs of input objects and desired outputs. Second, unsupervised learning models a set of inputs: labeled examples are not available. Third, semi-supervised learning combines both examples to generate an appropriate function or classifier. Finally, reinforcement learning learns how to act given an observation of the world. Every action has some impacts in the environment, and the environment provides feedback in the form of rewards that guides the learning algorithms.

We are especially interested in the active learning method [17] in the category of supervised learning since this method is more appropriate for dynamic and real-time situations. By the active learning, we may determine the value of variables which are changed dynamically. As seeing the characteristic of each variable, we can find the fact that Attention State Factor can be determined by this learning method. That is, we may classify the degree of how busy doctors are by the active learning method, and each classified cluster may represent the value of the parameter, Attention State Factor.

A useful starting point for investigating active learning for reasoning about interruption is the work of Kapoor and Horvitz [17] which advocates the use of active learning through experience sampling to determine the benefit of predictive models against the cost of the probes to obtain parameter values.

6.1.2 Probability of Response

Our current calculation for the probability of response uses an estimate based on the user unwillingness factor of the medical expert. Various default values are employed; for example, we assume that the probability of silence is the same, for all medical experts. For future work, it would be valuable to integrate into the estimate for probability of response a calculation of how much stress the doctor has been under, due to workload with patients that day. For example, we could measure how long the doctor has been with a status of *Relaxed* and determine the parameter value of $Stress\_Level\_Factor$. If the doctor has been relaxed longer than other doctors at this point, we could say that more relaxed doctor would have less stress. Sensor information could be sent to our system and this can help to determine the parameter value. Alternatively, we could simply be recording in time spent with patients for that particular medical expert and reduce our expectations of a successful probability of response for those who have been overworked that day.

We could also be modeling more carefully whether the expert's attention state factor is likely to be reduced soon, based on when the expert became busy and how much time has past.

6.1.3 Attention State Factor

The Attention State Factor represents the attention state of the user. For instance, a user is more interruptible when resting than when he/she is busy with important work [7]. In our scenario, this variable will determine how busy doctors are currently. We can measure the value of this variable by checking how many hours are needed to finish a current task the doctors are undertaking. This factor changes over time under the dynamic situation in our hospital setting.

For simplicity, if the doctor is currently handling any patient, we consider him/her as the doctor with a status of *Busy*. If not, we consider him/her as the one with a status of *Relaxed*. We could potentially obtain this parameter value by checking a sensor attached to the doctor. The sensor will check whether the doctor is currently having a patient.

6.1.4 Lack of Expertise Factor

In order to obtain the value of $Lack\_of\_Expertise\_Factor$, we need to keep track of the history of behavior of the medical expert from database in hospital. We can count the number of patients who have been treated by the doctor to figure out how the doctor has expertise for the specific patient. If the doctor has handled more than one thousand patients having the specific symptom, the doctor would be considered as more skilled expert than other doctors having treated five hundreds patients. In our model, we classify medical experts as three groups: less skilled experts, medium skilled experts, and more skilled experts. More skilled experts are given a status, *Low*, and the others are given a status, *High* for the parameter of $Lack\_of\_Expertise\_Factor$.

This current model can become more sophisticated. A more precise representation of the kind of problem that the current patient is exhibiting may better determine whether the medical expert in question has sufficient expertise. In addition, a class of problems should all be addressible equally by any medical expert (i.e. do not require considerable specific previous experience). In these cases, it may simply be the status of the medical expert (e.g intern compared to longterm practitioner) that makes the most difference. We note as well that as medical experts elect to take on more patients, their level of expertise will need to be adjusted.

6.1.5 User Unwillingness Factor

The user unwillingness Factor is the user's unwillingness to interact with the system. This is a measure of how receptive the user is towards being TOC'ed, and how disrupted they are by interruptions[Cheng, Fleming, and Cohen 05']. This value is related to the expertise of each doctor. If one doctor is good at the medical problem of the current patient because of his/her expertise, the doctor may be willing to do treat the patient. Otherwise, he/she will not help the patient and the doctor's User unwillingness Factor may become higher. If we view this variable in respect of the expertise of each doctor, we are able to conclude the fact that this variable is not directly related to the dynamic and real-time situations.

Currently, user unwillingness is simply calculated in terms of attentional state and lack of expertise. We could instead construct detailed user models more carefully, determining just how inherently willing this particular expert appears to be, when asked to assist (from previous scenarios). The inherent user willingness would then have to be integrated with the current calculation to determine the most appropriate value.

6.1.6 Expected Quality of Decision

To decide which doctor has a higher quality of decision, we considered his/her specialized area for the medical problem to figure out which expert is more knowledgable for the specific patient. In other words, we assumed that a doctor is knowledgable when the specialized area of the expert is matched with the medical problem of the patient. However, there might be various methods to evaluate knowledge of doctors and several factors to affect evaluating expected quality of decision after evaluating knowledge of the doctors.

Experiences of each doctor in hospital such as how long they have worked or which education they have obtained would be beneficial to gain information for estimating the expected quality of decision.

6.1.7 Enhancing the PTOC Question

In our current model, the question that each medical expert is asked is "Can you help with the current patient?". We can imagine making this communication more informative, indicating at least the basic area of the patient's problem or perhaps a very specific initial diagnosis. In this case, the expert's response may be altered (for example a very busy doctor who realizes he is the most expert may decide to say "Yes"). If more information were flowing, then our modeling which estimates probability of response would have to be adjusted as well.

6.1.8 Calculating the Timing in the Strategy Chains

Currently, the time to wait for the first entity in a strategy chain before then asking the next is determined on the basis of fixed values and estimations of probability of response. For future work, we could in fact explore more detailed algorithms to try to optimize the times at which each strategy moves on to consider the next entity in its chain. Our starting point for this research would be the methods proposed by Cheng [6] motivated by those introduced in the E-Elves project [27] where one uses an expected utility equation with an integral as in Section 2.4 and then differentiates solving for the t value that makes the equation equal to zero.

6.1.9 Revisiting Strategy Regeneration

At the moment, we assume there is no cost for strategy regeneration. While it is definitely valuable to proceed with updated parameter values, this will at least consume some time. We leave open for future work a more careful consideration of the value of the strategy regeneration cost.

We have also limited the length of strategies being generated, allowing the same entity to be consulted only once in a given strategy chain. For future work, we could extend the model to give up this restriction. Continuing to include strategy regeneration would then require an algorithm for when that generation should take place. In addition, it may make sense to reconsider the same expert within a chain, if that person is changing their attentional status (busy back to relaxed) or expertise level (to be increased).

It may also be interesting to vary the need for strategy regeneration based on the task at hand. Very critical patients may introduce an even greater need to regenerate strategy values frequently.

Finally, when the number of possible experts becomes quite high we may consider techniques for reducing the number of strategies that are generated, as discussed in [6]. For example, not considering all the possible p experts or grouping experts into types where parameters are all modeled similarly.

6.1.10 Task and Resource Allocation Problem

At the moment, our framework is developed in order to determine the best strategy (i.e. which entities to ask, how long to wait before asking another entity, etc.) to address the current task. In hospital emergency scenarios, this is the task of caring for the current patient.

In general, in dynamic environments there will be multiple tasks that need to be addressed at once and multiple resources that can be brought to bear in order to address those tasks (e.g. the different medical experts, for the hospital application).

In Chapter 5, we simply processed each new patient sequentially. For future work, we would explore a multiagent extension to our current framework. As in the work of Cheng [6], one approach we could investigate is effectively coordinating the requests for assistance that each task requires. In particular, Cheng proposes that each agent have a proxy who handles requests for the agent to assume decision making, which may arrive from any number of other agents in the system. In the hospital decision making scenario, this would involve having one medical expert being asked to assist with several different current patients at once (simply because he is considered to be the agent with the best expected quality of decision, relative to bother cost). The challenge is in effectively modeling the cost of bother, if the estimates for bother cost are possibly stale (unaware of any simultaneous requests for assistance with this expert or having a stale model of the expert's current attentional state). In Cheng, proxy agents could use a verification procedure, requiring each agent asking the expert for assistance to also indicate their estimate of the expert's current bother cost and being provided with a more accurate estimate from the proxy, if this is in fact quite out of sync with the expert's current state. The agent who had put the expert into its transfer-of-control strategy could then recompute, possibly asking a different expert.

With real-time settings, it would be important not to lose significant time approaching experts who are unlikely to assist, due to other current committments. But it would also be challenging to properly model the precise attentional state of each expert, because simply being asked to assist in a strategy being executed for a current patient would not ensure that the expert will say “Yes” and would actually be assisting.

What would ultimately be needed is some kind of effective task and resource allocation scheme, whereby tasks that are executed simultaneously do not try to make use of exactly the same resources at the same time. Research which may be of value to examine for insights includes that of Decker and Li [9] that views the problem of patient scheduling in hospitals as a multiagent coordination question.

6.1.11 Exploring Hospital Scenarios

In Chapter 5, we presented experimental results from simulations which monitored the number of problem patients that would arise, due to various algorithms. We had originally planned to label these as “dead patients”, since they were in a critical state and were unattended. In reality, hospitals will try hard to have these patients attended to in some manner, but as their level of criticality rises beyond a certain point, we have simply indicated that they are a problem because they may begin to incur a significant cost to the hospital. The certain point can be determined by measuring the value of blood pressure, breadth, and purse patients have. For future work, we could interact further with medical professionals to gain greater insights into how very critical patients are managed while waiting for their primary care.

We could as well learn how to model in some respects the economic considerations of patient care for emergency room scenarios. In particular, currently hospitals in Ontario are being judged according to their patient throughput and this is to impact the funding that they will receive. For this reason, it will become even more important for the most effective medical experts to be brought in to care for these patients.

The examples that we introduced in Chapter 4 indicated the best strategy chain to execute, in order to result in the highest overall utility. Another direction for future research would be to determine the importance of running the strategy with the top utility, compared to other choices where the utility value is only somewhat lower. The side-effects of operating at a lower level of utility could be investigated, to shed some light on the significance of the actual utility values. This might be done for instance through simulations.

Ultimately, it would be most valuable to conduct a thorough user study with medical professionals who operate in hospital settings. As a first step, we plan to learn more about the challenges to hospital workflow faced by these medical experts through the research that is currently being conducted by Diane Doran and her team, as part of the hSITE project. At that stage of the project, the use of sensors may also be involved as part of the information gathering during hospital scenarios.

Another element of the modeling of hospital scenarios that would be valuable to explore is the extent to which each patient's care is in fact reliant on multiple resources, working together. As each current patient relies not only on multiple experts but perhaps multiple kinds of equipment (which may or may not be available), there may be interesting challenges in reasoning about whether the group that is required to attend to the patient can in fact be assembled appropriately.

6.1.12 Exploring other Application Areas

Currently, we have projected our framework into the application area of hospital decision making. For future work, it would be worthwhile to explore the use of this framework for distinct applications where it is important to be reasoning about interaction with users, sensitive to bother and where there may be time critical conditions. One application of particular relevance is that of handling real-world emergencies such as fires, to be attending to by fire brigades. While often the solutions for addressing these problems are resolved by algorithms for dispatching available resources[1], one can imagine modeling more effectively the tendency for certain units to respond quickly or not (reflecting a kind of lack of expertise) or to be modeling the extent to which the current units are currently busy (so not simply a binary state of being busy or not but a continuum of values).

[1] I have spent time as working on dispatching vehicles for a fire brigade in South Korea

6.2 Contrast with Related Work

6.2.1 Modeling Bother

Our formulae for modeling the cost of bothering users are extensions of those used in Cheng's research [6], making use of an Initial bother cost reflective of the kind of question being asked, a bother increasing function reflective of this particular user, the bother endured by the user so far. In addition, we discuss how these parameters should be updated after strategy chains are executed and how our new lack of expertise factor would influence the user's unwillingness which forms part of the bother cost calculation.

This formulation is distinct from various other efforts by artificial intelligence researchers to model the cost of interrupting users. For example Bauer [2] proposes the inclusion of differing kinds bother increases so that users who are inherently unwilling to be bothered may have this factor increased exponentially whereas more relaxed users can have a log-like increase in their bother. But the nature of the penalty function for bother is not formulated explicitly. In this framework, users do indeed get more bothered with each successive query, in what they refer to as an annoy factor.

Raskutti and Zukerman [23] look at what is referred to as a nuisance factor when reasoning about which disambiguating query to issue, when interacting with a user. We are less focused on the choice of query in our current framework. They also examine the the number of additional queries that will likely come to the same user, once the initial interaction is generated. As we discussed in Section 6.1.7 above, if we extend our framework to be providing additional information when we approach an expert to take on the current task, we might then consider the cost that would likely arise from an additional ongoing dialogue, as generating bother. The suggestion that some questions carry more bother than others is at least reflected in part in our decision to have a high base bother cost with our query that asks users to assume the decision making.

Various researchers are considering the interruptability of users as critical to the design of any intelligent interface. Horvitz et al.'s COORDINATE system [16] allows users to indicate beforehand how interruptible they are during certain meetings. While we currently consider a user's unwillingness to be a reflection of their recorded attentional state and lack of expertise, we could additionally allow, for instance in the medical application, experts to declare themselves as highly disposed against interruption as they head into handling critical tasks, themselves (and to reset this value when their current task is complete). Horvitz's follow up research [15] does move on to consider an attentional state factor and this motivated the inclusion of such a factor in Cheng's work [6], which has carried over to our own model.

Bailey et al. [1] also have interesting research to confirm that a user's interruptability is reflective of their mental load at the time of interruption. This again suggests a more careful modeling, for future work, of the precise current task of each expert who is recorded currently as busy, distinguishing those experts who may be less able to cope with a possible interruption (and thus adjusting our parameter values when calculating strategies). We note that Bailey's work is more experimental and psychological, to compliment the development of models by artificial intelligence researchers.

6.2.2 Mixed-Initiative Systems

Although our current framework reasons about which entities to ask within a strategy chain and is therefore concerned with the multi-user case, we focus on a decision for how to handle the current task and as such are offering a process for whether to interact with a particular user, similar to what is examined by researchers designing mixed-initiative systems. A Good overview of mixed-initiative system is provided in [10] and [13].

As in the work of Fleming [11], our approach is one of deciding whether to interact with a user based on various user modeling parameters and on the basis of the expected quality of decision. This is in contrast with other efforts concerned with whether to interact with users, which address differing subproblems.

The work of Cesta and D'Aloisi [5] advocates a task driven control of how the initiative will shift within the system. This aligns well with our decision to focus on resolving the current task and determining the appropriate strategy of who to ask, at that point. This work also suggests that the user be allowed to control the decision making. It would be interesting for us to explore this possibility, within the context of our framework; for example, experts who became free could request that tasks be assigned to them, this signalling the value of a regeneration of strategies currently underway. Rich and Sidner [24] discuss the value of a shared plan known to both the user and the system, while the system reasons about interacting with the user. Future directions for our research might also explore how best to convey the current set of tasks that are underway, when interaction is initiated with an expert.

6.2.3 Adjustable Autonomy Systems

Our framework specifies the best strategy chain to execute in an effort to bring the most appropriate expert to attend to the task at hand. It integrates a process of asking other experts if a positive response is not received from the entities earlier in the chain. As such, this is a proposal for adjustable autonomy of the agent handling the task that is modeled on the framework provided in the E-Elves project [27].

Distinct from Cheng's approach, we do not consider differing questions to ask our entities, but we instead provide a kind of middle-ground solution whereby, as with Cheng's work, we allow a question to be asked of an expert, we allow for a response to that question (either yes or no, or the case of silence). The question acts as an initiation of the full transfer-of-control. This contrasts with Cheng's separation of PTOCs and FTOCs as two distinct nodes in the strategies that are considered. This is also distinct from the proposal of Tambe et al. who do not allow for users to be asked questions per se.

There is a variety of other research in the design of adjustable autonomy multiagent systems which generally focuses on differing concerns. For example, Schreckenghost et al. [28] view the challenge of adjustable autonomy in multi-user scenarios as best addressed by methods employed for coordination and communication in multiagent systems. Similar to Cheng, they introduce proxy agents who assist in the coordination, including an overall Crew Proxy (for the application of space crew management) to assist in notifying agents of incoming events and selecting the best methods for offloading the autonomy, for the resolution of tasks. This research suggests further exploration of proxy agents to be understanding the current limitations of each entity that may be approached to take on a current task.

Berry et al. [3] comment on the challenges of honouring preferences of users for tasks such as meeting scheduling. This provides some backing for approaches such as ours where which experts are consulted is simply driven by the predominant needs of the current task.

Martin et al. [21] point out that in multi-user environments there may be challenges when different agents are soliciting the assistance of the same, other agents (or when two parties may be asking each other to take on their current task). They refer to this as the problem of interfering with each other's commands. This work may provide some insights into how to manage the case of multiple patients needing assistance simultaneously, outlined in Section 6.1.10 above.

6.2.4 User Modeling for Healthcare Applications

Our research is aimed at contributing to the mandate of the hSITE project [22]: delivering the right information to the right people at the right time. In particular, we determine who the right people are and what the right time to ask each person is. Our procedure forms part of the modeling required in hSITE's Theme 1: determining the overall workflow (outlined for the particular use of in-hospital decision making).

Other researchers have been exploring the value of user modeling towards the improvement of healthcare services. A special issue of the UMUAI journal is forthcoming [8]. Some of the other work that is relevant to this topic includes efforts to personalize the delivery of health information to users (e.g. [4]) and projects to give users control over their user model, for more effective health promotion (e.g. [19]).

6.2.5 Real-Time Decision Making

One of the primary challenges that we examine in this thesis is that of coping with a need for quick decisions, in real-time dynamic environments. Altering the algorithms that have been designed for intelligent automated reasoning to be more time sensitive is another topic that other researchers have explored.

Included here are efforts by multiagent systems researchers in applications such as RoboCup Search and Rescue [20] where algorithms need to be designed to coordinate the activities of emergency medical, police and firefighting robots to rescue civilians after an earthquake. In particular, the model of Micacchi, Cheng and Cohen [18] leverages interaction between agents to ensure that parameters are up to date for more effective real-time decision making, for this application. This suggests that our strategy regeneration procedure, to determine current parameter values at periodic intervals is of general value.

6.3 Conclusions

This thesis has examined the challenge of having agents reason about whether to interact with users, in multiagent, multi-user scenarios where each agent has been tasked with operating autonomously to solve a problem on behalf of its user, and in environments where the tasks may have critical time constraints and the parameter values may be dynamically changing. In contrast with previous work on formulating transfer-of-control strategies for multiagent adjustable autonomy systems, we propose a strategy regeneration process that limits the lengths of the strategy chains and results in updated parameter values for the decision-theoretic reasoning. We demonstrate that this strategy regeneration is effective, through a simulation in the application of hospital emergency room decision making, showing the benefits of the regeneration of values, towards successfully completing the tasks that are being transferred to the entities in each strategy chain. This begins to provide some insights into how the dynamic nature of the environment can be considered as part of the determination of the most effective transfer-of-control strategies. We restrict our focus to effectively addressing each new task, sequentially and discuss possible extensions to this procedure that address multiple tasks simultaneously.

We also explore in greater detail the user modeling requirements for designing effective transfer-of-control strategies in scenarios that are time and task critical. In particular, we first introduce a parameter that models the level of expertise of each entity, of use in more effectively modeling the expected quality of decision from this entity. We integrate this parameter into our calculation of a user's willingness to accept a transfer-of-control (equating less expertise with less willingness), of use in applications where there is a clear understanding of the users' expertise towards the possible tasks at hand. This provides a richer modeling of the user unwillingess factor than that employed in previous models, which was largely predicated on deriving an initial estimate for each user based on their stereotype or through explicit acquisition (e.g. a survey).

We also introduce an explicit modeling of the criticality of the task at hand and propose an weighting adjustment for the balance between the expected quality of decision and the cost of bother, relative to the task criticality. We are thus able to place greater demands on the quality of decision for more critical tasks and to reduce the focus on the cost of bother. This proposal is in contrast to other approaches in the field which offer varying methods for modeling bother, but do not reason about this factor relative to the importance of the quality of decision provided by the user with whom interaction has been initiated.

In order to demonstrate that our proposed formulae for modeling users are effective, we offer some detailed examples where there is tension between the need for an effective decision and the possible cost of bother and discuss the value of our particular results. We also provide a simulation to show that our modeling of bother in particular is effective, compared to a case where bother cost has not been modeled when reasoning about transfering control for addressing current tasks in an environment.

As such, we begin to outline how the modeling of the task and of the user as part of the overall process for reasoning about interaction can be designed effectively, for dynamic environments with time critical constraints.

We also project our model into the specific application of hospital emergency room decision making, outlining how the various parameters serve to model the current patients and the medical experts and exploring appropriate qualitative values to employ, to distinguish the user models in this environment (e.g. Attentional State ranging from relaxed to busy). We propose to be running our algorithm for reasoning about transfers of control to have a current patient attended to, driving the actual interaction with existing medical professionals. Our proposal for constructing a strategy chain provides for a quick change to eliciting the assistance of a different expert, if the initial expert is unwilling to assist. Our strategy regeneration also allows for up to date modeling of the environment, where tasks arise with great frequency (e.g. the attentional state of each expert may change a great deal). As our validations are also projected in particular into this specific application, this provides further evidence of the value of our approach for this particular medical challenge.

References

[1] B.P. Bailey, J.A. Konstan, and J.V. Carlis. The effects of interruptions on task performance, annoyance, and anxiety in the user interface. In *Proceedings of Human-Computer Interaction – INTERACT'01*, pages 593–601, 2001. 75

[2] M. Bauer, D. Dengler, and G. Paul. Instructible agents for web mining. In *Proceedings of International Conference on Intelligent User Interfaces*, pages 21–28, 2000. 15, 74

[3] Pauline M. Berry, Melinda Gervasio, Toma's E. Uribe, Karen Myers, and Ken Nitz. A personalized calendar assistant. In *Working notes of the AAAI Spring Symposium Series*, 2004. 76

[4] Alison Cawsey, Floriana Grasso, and Ray Jones. A conversational model for health promotion on the world wide web. In *AIMDM '99: Proceedings of the Joint European Conference on Artificial Intelligence in Medicine and Medical Decision Making*, pages 379–388, London, UK, 1999. Springer-Verlag. 77

[5] A. Cesta, D. D'Aloisi, and M. Collina. Adjusting autonomy of agent systems. In *Proceedings of AAAI'99 Spring Symposium on Agents with Adjustable Autonomy*, pages 17–24, 1999. 75

[6] M. Cheng. A hybrid transfer of control approach to designing adjustable autonomy multiagent systems. Master of Mathematics thesis, University of Waterloo, Waterloo, Ontario, 2005. 2, 3, 9, 10, 17, 19, 20, 21, 26, 28, 70, 71, 74, 75

[7] M.Y.K. Cheng and R. Cohen. A hybrid transfer of control model for adjustable autonomy multiagent systems. In *Proceedings of AAMAS'05*, pages 1149 – 1150, 2005. 68

[8] R. Cohen, H. Jung, M.W. Fleming, and M.Y.K. Cheng. A user modeling approach for reasoning about interaction sensitive to bother and its application to hospital decision scenarios. *Journal of User Modeling and User-Aapted Interaction*, 2010. 77

[9] K. Decker and J. Li. Coordinated hospital patient scheduling. In *ICMAS '98: Proceedings of the 3rd International Conference on Multi Agent Systems*, page 104, Washington, DC, USA, 1998. IEEE Computer Society. 72

[10] G. Ferguson, J. Allen, and B. Miller. Trains-95 : Towards a mixed-initiative planning assistant. In *Proceedings of the Third Conference on Artificial Intelligence Planning Systems*, pages 70–77, 1996. 75

[11] M. Fleming. *Reasoning about Interaction in Mixed-Initiative Artificial Intelligence Systems*. PhD thesis, University of Waterloo, 2003. 1, 2, 3, 6, 7, 8, 15, 22, 23, 24, 75

[12] M. Fleming and R. Cohen. A decision procedure for autonomous agents to reason about interaction with humans. In *Proceedings of the AAAI 2004 Spring Symposium on Interaction between Humans and Autonomous Systems over Extended Operation*, pages 81–86, 2004. 6

[13] Susan Haller, Susan McRoy, and Alfred Kobsa. *Computational Models of Mixed-Initiative Interaction*. Springer Publishing Company, Incorporated, 2007. 5, 75

[14] H. Hexmoor, R. Falcone, and C. Castelfranchi, editors. *Agent Autonomy*. Kluwer Publishers, 2003. 9

[15] E. Horvitz and J. Apacible. Learning and reasoning about interruption. In *Proceedings of the 5th International Conference on Multimodal Interfaces (ICMI'03)*, pages 20–27, 2003. 74

[16] E. Horvitz, P. Koch, C.M. Kadie, and A. Jacobs. Coordinate: Probabilistic forecasting of presence and availability. In *Proceedings of the 18th Conference in Uncertainty in Artificial Intelligence (UAI'02)*, pages 224–233, 2002. 74

[17] Ashish Kapoor and Eric Horvitz. Experience sampling for building predictive user models: a comparative study. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 657–666, New York, NY, USA, 2008. ACM. 67

[18] Balázs Kégl and Guy Lapalme, editors. *Advances in Artificial Intelligence, 18th Conference of the Canadian Society for Computational Studies of Intelligence, Canadian AI 2005, Victoria, Canada, May 9-11, 2005, Proceedings*, volume 3501 of *Lecture Notes in Computer Science*. Springer, 2005. 77

[19] Stefanie Kethers, Peter Lamb, Shijian Lu, Ce'cile Paris, Ross Wilkinson, and Kathleen Griffiths. Combining personalization and privacy to deliver remote care to people with depressive illnesses. In *Proceedings of CHI 2006 Workshop on Privacy-Enhanced Personalization PEP06*, 2006. 77

[20] Hiroaki Kitano. Robocup rescue: A grand challenge for multi-agent systems. In *Proceedings of ICMAS 2000*, pages 5–12, 2000. 77

[21] C. Martin, D. Schreckenghost, and P. Bonasso. Augmenting automated control software to interact with multiple humans. In *Proceedings of the AAAI04 Workshop on Interaction between Humans and Autonomous Systems over Extended Operation*, 2003. 76

[22] D. Plant. hsite: Healthcare support through information technology enhancements. *NSERC Strategic Research Network Proposal*, 2008. 15, 66, 77

[23] B. Raskutti and I. Zukerman. Generating queries and replies during informationseeking interactions. *International Journal of Human Computer Studies*, 47(6):689–734, 1997. 74

[24] Charles Rich and Ace L. Sidner. Diamondhelp: A generic collaborative task guidance system. In *AI Magazine 28(2). Special Issue on Mixed-Initiative Assistants.* 75

[25] S.J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2003. 5, 67

[26] P. Scerri, D. Pynadath, and M. Tambe. Towards adjustable autonomy for the real world. *Journal of AI Research*, 17:171–228, 2002. 2

[27] P. Scerri, D.V. Pynadath, and M. Tambe. Why the elf acted autonomously: Towards a theory of adjustable autonomy. In *Proceedings of AAMAS'02*, 2002. 2, 9, 28, 70, 76

[28] D. Schreckenghost, C. Martin, P. Bonasso, D. Kortenkamp, T. Milam, and C. Thronesbery. Supporting group interaction among humans and autonomous agents. In *Proceedings of AAAI02 Workshop on Autonomy, Delegation and Control: From Inter-Agent to Groups*, 2002. 76

[29] G. Weiss, editor. *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, 1999. 5