HARD AND SOFT SENSOR INFORMATION FUSION USING COGNITIVE INJECTION PROCESS FOR DECISION MAKING

A Dissertation in
Industrial Engineering and Operations Research

By
David M. Sudit

© 2012 David M. Sudit

Submitted in Partial Fulfillment
Of the Requirements
For the Degree of

Doctor of Philosophy

May 2012
The dissertation of David M. Sudit was reviewed and approved* by the following:

Soundar Kumara
Allen E. & M. Pearce Distinguished Professor of Industrial Engineering
Dissertation Co-Adviser
Chair of Committee

David L. Hall
Dean of Information Science and Technology
Dissertation Co-Adviser

David A. Nembhard
Associate Professor of Industrial Engineering

Guodong Pang
Marcus Career Professor of Industrial Engineering

Paul M. Griffin
Head of the Department of Industrial Engineering

*Signatures are on file in the Graduate School.
Abstract

Advances in technology have led to the abundant availability of data from heterogeneous sensors. This combined with knowledge about the sensed data has improved our ability to turn data into reliable and concise information. The process of combining data from different sources is termed Data or Information Fusion. In particular, a new area of focus has been on Hard and Soft fusion. The idea of Hard and Soft fusion stems from a classification system that divides sensors into two groups: Hard and Soft sensors. Hard sensors are the traditional electronic sensing sources like satellites and RFIDs that use a programmed system to collect the data. Soft sensing occurs when a human is collecting data. This type of sensing is clearly less predictable, but gives a different dimension to the complete set of data. With the availability of new communication methods (eg. Facebook, Twitter, YouTube) any person in any setting could be a soft sensor. With the advent of crowd sourcing it will be common to encounter the context of soft sensing in the future. With the inclusion of humans as sources of data, Information Fusion algorithms should be able to combine these disparate sources.

In this thesis I focus on including humans in a decision making system to improve the knowledge of a Decision Maker (DM). I accomplish this by using a subject matter expert, whom I call a Human Fuser, to manipulate the reliability values extracted by the Information Fusion process. The Information Fusion process is a hard process as it uses programs to align and associate data to be able to estimate pieces of information. These pieces of information have a reliability or certainty value attached with them that has been extracted by looking at the sources (the hard and soft sensors) error functions. I introduce a soft process, called Cognitive Injection Process into the Information Fusion process.
Including soft sensors will change the DM’s knowledge of the problem scenario because the reliability values of the different pieces of information will be changing. In a decision making context I need to consider several features, such as the color of a car, speed at which it is travelling etc. I should consider these features and select the best feature-estimate set, which can help the decision maker in inferring for example in this case, color and speed. In this thesis I address three sub-problems: 1. Feature-Estimate selection through linear mathematical models, 2. Feature-Estimate selection through complex mathematical models, and 3. Introducing multiple human fusers into the information fusion process. In the first problem I model the feature-estimate selection for soft sensing as a knapsack problem and solve it through heuristics. I consider entropy modeling and minimize entropy, thus leading to best feature-estimate set, in the second problem. Finally I use learning and forgetting models to address the problem of introducing multiple humans in the information fusion process.
# Table of Contents

List of Tables ...................................................................................................................... x  
List of Figures ................................................................................................................... xii  
Acknowledgements ........................................................................................................... xv  

## Chapter 1: Introduction ....................................................................................................... 1  

1.1 Motivation ................................................................................................................. 2  
1.2 Research Problem ...................................................................................................... 3  
1.3 Methodology ............................................................................................................. 7  
1.3.1 Sub-Problem 1: Selecting the Feature-Estimate Set by Introducing the CIP with Knapsack Problems ................................................................................. 7  
1.3.2 Sub-Problem 2: Complex Mathematical Models within the CIP ..................... 8  
1.3.3 Sub-Problem 3: Using Multiple Human Fusers within the CIP ....................... 9  
1.4 Research Contributions ........................................................................................... 10  
1.5 Organization of the Thesis ...................................................................................... 11  

## Chapter 2: Introducing the Cognitive Injection Process with Knapsack Problems ........ 13  

2.1 Introduction ............................................................................................................. 13  
2.1.1 Information Fusion ........................................................................................... 15  
2.1.2 Including a Soft Process ................................................................................... 17  
2.2 Problem Discussion ................................................................................................. 18  
2.3 Background Literature ............................................................................................. 20  
2.3.1 Hard Sensors ..................................................................................................... 21  
2.3.2 Soft Sensors ...................................................................................................... 22
3.3.1 Problem Description ......................................................................................... 58
3.3.2 Mathematical Models and Heuristics ............................................................... 59
3.3.3 CIP Flow ........................................................................................................... 77
3.4 Experimentation ...................................................................................................... 82
  3.4.1 Top of List Analysis ......................................................................................... 82
  3.4.2 Max-Min Analysis ............................................................................................ 85
  3.4.3 Entropy Analysis ............................................................................................. 87
3.5 Summary ................................................................................................................ 89

Chapter 4: Multiple Human Fusers within the Cognitive Injection Process .............. 91
  4.1 Introduction ........................................................................................................... 91
    4.1.1 CIP with Multiple Human Fusers ............................................................... 92
    4.1.2 Choosing a Model ........................................................................................ 94
  4.2 Background Literature ....................................................................................... 100
  4.3 Methodology ....................................................................................................... 103
    4.3.1 Problem Description ...................................................................................... 103
    4.3.2 Cognitive Injection Process ........................................................................ 104
    4.3.3 Mathematical Models and Heuristics ............................................................ 105
  4.4 Experimentation ................................................................................................... 117
    4.4.1 Analysis ......................................................................................................... 117
  4.5 Summary .............................................................................................................. 123

Chapter 5: Conclusion and Future Research ................................................................ 126
Appendix A: OPL Code for All Math Models and Heuristics........................................ 132
Appendix B: Expansion of Decision Tree ................................................................. 136
List of Tables

Table 2.1 IF Output ........................................................................................................... 20
Table 2.2 Empty AHP Table ............................................................................................ 31
Table 2.3 Completed AHP Table ..................................................................................... 31
Table 2.4 Processing Times ............................................................................................. 33
Table 2.5 First Iteration Results ...................................................................................... 33
Table 2.6 Modified Certainty Values ............................................................................... 34
Table 2.7 Initial Ranked List ........................................................................................... 36
Table 2.8 Iteration 1 Ranking .......................................................................................... 36
Table 2.9 Greedy Selection Rules .................................................................................... 40
Table 2.10 Bang-for-Buck Selection Rules ..................................................................... 40
Table 2.11 Metric Values for Model 1 ............................................................................. 42
Table 2.12 Heuristic Performance Compared to Models ................................................. 43
Table 3.1 Hard Sensor Classifications ............................................................................. 57
Table 3.2 Soft Sensor Classifications .............................................................................. 57
Table 3.3 TOL Problem Sizes ........................................................................................... 83
Table 3.4 Average Values over 100 Runs for TOL ......................................................... 84
Table 3.5 Average Values over 100 Runs for Max-Min .................................................. 86
Table 3.6 Entropy Problem Sizes ..................................................................................... 87
Table 3.7 Average Values for Entropy Models ............................................................... 88
Table 4.1 Class Types ...................................................................................................... 94
Table 4.2 Model Choice Guidelines ............................................................................... 96
Table 4.3  Levels of Information Fusion ................................................................. 100
Table 4.4  Modification Parameter Values .............................................................. 117
Table 4.5  Problem Sizes ....................................................................................... 118
Table 4.6  Learning and Forgetting Results .............................................................. 119
List of Figures

Figure 1.1  Inserting a human fuser. ................................................................................... 4
Figure 2.1  An example of information fusion. ................................................................. 15
Figure 2.2  The JDL data fusion model. .......................................................................... 16
Figure 2.3  The addition of soft processes. ..................................................................... 18
Figure 2.4  An example situation. .................................................................................... 19
Figure 2.5  Detailed system flow. .................................................................................... 30
Figure 2.6  AHP computations ....................................................................................... 32
Figure 2.7  A changing ranked list. .................................................................................. 37
Figure 2.8  Knapsack heuristic flow. .............................................................................. 39
Figure 2.9  Number of Variables vs. Solution Time ....................................................... 45
Figure 2.10 First run for Model 1. .................................................................................. 45
Figure 2.11 Model 1 average results .............................................................................. 46
Figure 2.12 First run for Model 2. .................................................................................. 46
Figure 2.13 Model 2 average results .............................................................................. 47
Figure 2.14 First run for Model 3. .................................................................................. 47
Figure 2.15 Model 3 average results .............................................................................. 48
Figure 2.16 Percent gain over time ................................................................................. 49
Figure 3.1  Inclusion of CIP ........................................................................................... 54
Figure 3.2  Computational complexity. .......................................................................... 55
Figure 3.3  Hospital data example. ............................................................................... 59
Figure 3.4  TOL-H ........................................................................................................ 62
Figure 3.5 MM-H .......................................................................................................... 68
Figure 4.13  Problem size 3 results (16,050 variables) ............................................... 122
Figure 4.14  Problem size 4 results (38,850 variables) ............................................... 122
Figure 4.15  Problem size 5 results (51,870 variables) ............................................... 123
Figure 4.16  Run Time vs. Problem Size ..................................................................... 123
Acknowledgements

While I have worked extremely hard on my course work, research, presentations and preparing this document there is no chance I could have reached this point without help, support and guidance from many people. First of all, I would like to thank my Co-advisors Dr. Soundar Kumara and Dr. David Hall. Dr. Hall helped me obtain financial support with the Applied Research Laboratory through a research fellowship. Dr. Hall also spurned the initial idea that has created this research topic. Dr. Kumara has helped me navigate through this incredibly difficult process that is obtaining a Ph.D. He has been supportive and helpful when I have needed him and tough on me when he has wanted me to learn about the research process. Dr. Kumara has made me grow more as a researcher than anyone else and it is going to help me greatly in the future.

I also need to thank my other two committee members. Dr. David Nembhard has given me some great input throughout the research process that has allowed for this research to go into greater detail. Dr. Pang has been extremely supportive throughout the process and have been grateful that he rounded out my committee.

There have been other faculty and staff that while not on my committee has been there for me over my time at Penn State. Dr. Jeya Chandra has answered so many questions for me over the years and gave me the opportunity to teach a class over a summer that was an amazing experience. Dr. and Mrs. Joshi have been great friends to me and have always cheered me during stressful times. The ladies (Erin, Sharon, Olga, Susan and Lisa) in the office have been so helpful and amazing with me. Special thanks to Lisa who I probably bothered once a week and always answered my questions or fixed my problems.

Next I have to thank my friends. I have met some amazing people at Penn State and will never forget what these people have done for me. Aixa Cintron, Lourdes Medina, and Rachel Abrahams have been by my side every step of the way. They have helped me not only through the Ph.D. process but also through some of the hardest times of my life and will be forever grateful. Emily Niman, Dr. Monifa Vaughn-Cooke, Dr. Paul Lynch, the Puerto Rican contingent (Emil, Josue, Liza, Lorna, Nella, and Wilmarie) and many other friends have been extremely supportive. My lab mates, who have been helpful and entertaining while working side by side, also deserve a big thanks (Supreet, Manini, Hui, Akshay and Starry).

Lastly but not least, I am thankful for the support, love, and patience of my entire family. In particular, my sister has handled a tough situation so well and has taught me how strong I need to be when problems arise. I would like to thank my parents for everything and anything one can think of, the list would be too long to write. I love you guys and this work is dedicated to you two.
Chapter 1: Introduction

Technological advances have led to the abundant availability of data from heterogeneous sensors. This, combined with contextual knowledge about sensed data has improved our ability to turn data into reliable and concise information. Converting data into information is extremely important in the decision making process for two reasons. First, information gives a decision maker (DM) a better situational understanding of the problem scenario than raw data alone. For example, data (e.g., the color of a car or the direction that it is heading) must be associated with other data in order to provide contextual understanding (e.g., a blue car is heading north towards a power plant). The conversion of raw data into information occurs in a process called data or information fusion (IF). Among IF scholars, hard and soft fusion has emerged as a new research focus. Hard and soft fusion is based on a classification system that divides sensors into two groups (Pravia, Prsanth, Arambel, Sidner, & Chong, 2008): (a) hard sensors, which are traditional electronic data collection sources such as satellites and RFIDs that rely on programmed systems; and (b) soft sensors, or human data collectors.

With the availability of new communication methods (e.g., Facebook, Twitter, YouTube) any person can serve as a soft sensor in any setting. In crowd sourcing, it is common to encounter the soft sensing context. Clearly, soft sensing is less predictable, but it offers a different dimension to the complete dataset (e.g., when soldiers perform reconnaissance work). As human data collection becomes more common, IF must evolve as well. Hard and soft fusion is used to combine sensors with more uncertainty (soft sensors) with those that are slightly more reliable and driven by programs (hard sensors).

The conversion of raw data into information is also important in order to reduce the amount of information to be processed by a DM before determining a course-of-action (COA).
Due to increased availability, a DM does not have the resources to process all pieces of raw data before determining a COA. By converting raw data into pieces of information, a DM is able to review a smaller set of inputs before choosing his or her action without losing any knowledge of the raw data. This is a key point when considering a decision making system that includes the use of IF, and raises a question that must be answered: Is it possible for the amount information extracted by the IF process to be so extensive that a DM is unable to determine a COA within a reasonable amount of time? The answer is yes. Thus, in this research, I focus on improving a DM’s knowledge base by manipulating the reliability values extracted by the IF process. I consider the idea of hard and soft fusion and expand this idea into hard and soft processes. The IF process is a hard process; it uses programs to align and associate data in order to estimate pieces of information. These pieces of information have reliability or certainty values that are extracted from the source (i.e., hard and soft sensor) error functions. Here, I introduce a soft process called the cognitive injection process (CIP) into a system that includes the IF process. CIP allows a human fuser, who is a subject matter expert on the problem scenario, the dataset or even the sensors used to gather the data, to modify the reliability values of the IF process.

1.1 Motivation

I developed and incorporated the CIP into a decision making system for several reasons. The first and most important is to improve DMs’ knowledge bases. As technology progresses and becomes a larger part of everyday life, data, particularly from human sensors, are integral when decisions must be made. The use of online social networks (i.e., Facebook, YouTube, Twitter, etc.) enables data and information sharing and extraction in a completely new way. With this increase in soft sensor data, it is very beneficial to have human involvement in data processing in a decision making system. Humans offer a different perspective than electronic
automated programs, and can provide a new level of information for a DM. For example, imagine a scenario where a hard sensor sends data that there is a red car at a certain location. This could affect the COA, but there are numerous red cars. Yet, a human in the same scenario may relay data that a red striped car is heading south. This completely changes the situational understanding of the problem scenario and in turn changes the COA, because the DM is more knowledgeable about the situation. By including human fusers in data processing systems, subject matter experts can modify the reliability of questionable data or information.

Another motivation is to determine appropriate contexts (i.e., where in the process and which information to modify) for human fusers in order to achieve the overall goal of improving DMs’ knowledge. Incorporating a human fuser after the IF process allows certainty values to be manipulated. While a case can be made for placing a human fuser before the IF process so as to refine the input data and generate better results, this would also require developing a way to combine and compare certainty values from different error functions. IF already performs that function, and there are advantages to using a process that has already been used and tested. To determine which information the human fuser should modify, an optimization model can help maximize the potential gain in knowledge. Different models provide basic customizability for various problem scenarios, datasets or sensor sets. First, I consider simple mathematical models and then I expand into more complex models that use different certainty calculation techniques (i.e., entropy model) and models that account for multiple human fusers (i.e., learning and forgetting models).

1.2 Research Problem

I consider three sub-problems in this research. The first sub-problem focuses on the methodology of the main research problem, which is how to insert a human fuser into a decision
making system. As shown in Figure 1.1, I place the human fuser between the IF process and the DM instead of between the sources of data and the IF process. In particular, I exploit the fact that the IF process combines not only data from different sensors, but also different error functions to give each piece of information its own reliability or certainty value. Given the location of the human fuser in the process, I consider the question of what should be observed by the human fuser. The Cognitive Injection Process (CIP) allows the human fuser to interact with the output of the IF process in a structured manner.

Within CIP, a method must be included to select the data points a human fuser should modify, because a human fuser cannot observe all data points within a reasonable amount of time. I use mathematical models to select the data points for the human fuser. In order to analyze the overall system with a human fuser, I keep the mathematical models simple so that I can convert them into knapsack problems. Knapsack problems are interesting in that it is possible to convert them into decision problems, thus making them NP-complete problems instead of NP-hard. An NP-complete problem will grow in polynomial or pseudo-polynomial
time (Johnson, 1979) as the problem size increases, as opposed to NP-hard problems, which tend to grow exponentially in time as problem size increases. Converting the models into knapsack problems also provides the benefit of being able to use known heuristics in place of knapsack models, allowing a fundamental question to possibly be answered: Is there a tipping point at which using an accurate model that takes longer to solve is trumped by a quick solution obtained from a heuristic?

The second sub-problem addresses what happens when more complex models are inserted into a system that uses CIP. By including more complex mathematical models, it is possible to further customize how a human fuser will work within a system by changing the selection method. In particular, this provides an additional way to combine certainty values. Knapsack problems combine certainty values by averaging. There are numerous other techniques that have been developed to combine certainty values, such as Dempster-Shafer theory, Bayesian theory, and fuzzy logic. I use entropy as a method to combine certainties instead of a simple averaging method because it has been used in information theory. However, entropy is a non-linear function which takes an extremely long time to solve when formulated in a math model. Thus, the entropy calculation must be converted into a linear form in order to be used in a real time system. Even after the linear conversion, the entropy model takes a long time to solve, so a heuristic methodology must be considered as an alternate solution to an entropy model. The general heuristic methodology is used to create individual heuristics for each of the newly created models (i.e., non-knapsack formulations). In this sub-problem I also develop a new class of problems that allow for the DM to be more involved in the selection method of the human fuser. By allowing a DM to input parameters that could change the eventual options
providing to the human fuser by the selection method, the overall system is more customizable for each problem scenario.

The third sub-problem examines how to include multiple human fusers into the system. With data becoming more available, it is possible to enlist the assistance of multiple experts in the decision making process for a dataset or problem scenario. In this case, the selection method must choose not only which data points to observe, but also which human fuser should evaluate them. Since no two humans are exactly the same, each human fuser may have different areas of expertise for different aspects of the same problem scenario. Learning and forgetting models are used in workforce settings to model different workers with different areas of expertise. In this case, human fusers are the workforce and the data points to be modified are the tasks. Similar to the entropy model, learning and forgetting models tend to be non-linear and require rules to be introduced that linearize the mathematical formulation. As the expertise of a human fuser changes due to learning and forgetting, the notion of data degradation can be incorporated into the math models. Data degradation is the idea that as time progresses, data becomes less and less reliable. For instance, imagine a piece of data stating that person A is at location B at time 0. As time progresses, it is less and less likely that person A will still be at location B. This is true for many datasets, in that the relevance and reliability of data for a problem scenario decreases as more time is taken by the DM to decide on a COA.

To select a model for a problem scenario and dataset, I develop a classification system to determine which model or heuristic best serves the system. I also present a decision tree for model selection.
1.3 Methodology

1.3.1 Sub-Problem 1: Selecting the Feature-Estimate Set by Introducing the CIP with Knapsack Problems

Consider a problem scenario where cars are passing through an intersection. Multiple hard and soft sensors retrieve data on the cars, the directions they are traveling, and any accidents that occur. The hard sensors are still cameras, video cameras and speedometers, while the soft sensors are humans at the intersection. The DM is trying to determine whether to change the yield signs to stop signs. The data from the sensors is put through the IF process which yields:

1. *Estimates*: The pieces of information that come from the IF process based on data from different sensors (e.g., a blue car traveling south almost hit another car).

2. *Features*: Common attributes among the different estimates (e.g., a car was traveling south, an accident did not occur).

3. *Certainty Values*: The level of certainty of every feature within each estimate. The value is determined by using the error functions of the sensors that provided the data for the feature-estimate combination.

Normally, the IF process outputs are used to determine a situational understanding for a DM who then determines a COA. I introduce a human fuser between the IF process and the DM, who modifies the certainty values for the feature-estimate (F-E) combinations. The human fuser does not have the time to look at all F-Es, so the methodology to pick the best F-E set is based on:

1. *Analytical Hierarchical Process (AHP)*: This is used to determine the relative importance of different features. In many problem scenarios, certain features are
more important to the scope of the problem than others. For the intersection example, knowing the direction a car is traveling is more important than if a car has rims.

2. **Linear Math Model**: The math model is used to select the best set of F-Es for a human fuser to modify. It uses the weights from the AHP process to help select the best F-Es.

After the chosen set of F-Es is modified by the human fuser, the DM has a new situational understanding of the problem scenario and can either determine a COA or allow further modification by the human fuser.

### 1.3.2 Sub-Problem 2: Complex Mathematical Models within the CIP

For this sub-problem there is a focus on using more complex mathematical models within the CIP. The models change how the certainty values for features in estimates are brought together, and allow a DM to choose parameters that affect F-E selection. Three new models are presented:

1. **Entropy model**: Estimate certainty values are combined using entropy, a technique used in information theory. Entropy is a nonlinear function, so assumptions are made to linearize the model. The assumptions provide structure to the modification process in order to determine where the certainty values will fall after a modification is made.

2. **Top of List model (TOL)**: This is one of two models that allow a DM to affect the selection process. For the TOL model, a DM specifies how many estimates he or she will be using to determine the COA. This means that the most certain estimates will be used by the DM and all other estimates will not. The TOL model filters out those estimates that cannot reach the top of the list.
3. Max-Min model (MM): This is the other model that allows for DM input. The MM model maximizes the number of estimates that have a minimum certainty value above some threshold set by the DM. Similar to TOL, the MM model filters estimates – in this case, those that cannot reach the minimum certainty threshold.

Since these models require more constraints and more complex objective functions, the solution time is very long for larger problems. To account for this, I present an idea to be used in a heuristic. The idea is to relax all integer variables to be free variables that can be any value between 0 and 1. This allows for a solver to obtain a solution extremely quickly. The heuristic then checks to see if all variable values that were originally assumed to be integers are in fact integers. If they all are integers, then the integer solution can be used to assign data points to the human fuser. If at least one variable is not an integer, then the heuristic determines which variable will provide the most gain to the DM. This gain is unique to each math model, as the objective of each model is different. Once the variable with the largest gain is found, a constraint is added to the relaxation forcing that variable to be 1 if processing time constraints are kept feasible, and 0 if the processing time constraints are violated. The heuristic continues to add constraints until all variables are integers.

1.3.3 Sub-Problem 3: Using Multiple Human Fusers within the CIP

The third sub-problem introduces the idea of using multiple human fusers within the CIP. I use learning and forgetting to model the use of multiple human fusers with different areas of expertise for different features. Like the entropy model, the learning and forgetting model must be linearized in order to be solved in a timely manner. This linearization is achieved by structuring how expertise levels change. I also describe the idea of including data degradation, which reduces the certainty values of data points as time passes. These new models bring the
total to eight (three knapsack, entropy, TOL, MM, learning and forgetting, learning and forgetting with data decay), with a classification system for selecting which one should be used within CIP. The difficulty in choosing among the models lies in determining a common ground on which to compare them. Objective values, for instance, cannot be compared because each model’s objective is completely different. Obviously, the different models can be categorized by specifying problem types. For example, for a dataset with available probability density functions (PDF’s), the entropy model should be used. On the other hand, if a PDF is not available as part of the dataset, then using the entropy model is not feasible because the entropy equation requires a PDF. These guidelines also help when combining two or more of the models. For example, if PDFs are available and multiple human fusers are available, the availability of PDFs implies using the entropy model, but the presence of multiple human fusers means the learning and forgetting model can be used. This leads to the possibility of combining the entropy model and learning and forgetting model to exploit both the PDFs and the multiple human fusers.

1.4 Research Contributions

The motivation of this research is to improve a DM’s knowledge so that he or she can determine an optimal COA for a given problem scenario. The first contribution from this research is the introduction of the Cognitive Injection Process (CIP), which improves decision making systems. This research also introduces a way for humans to be more involved in the data processing portion of such systems. While computers and automated processes are typically faster, a human can provide a different viewpoint and add another layer of knowledge to a dataset. This leads to the second contribution: determining where a human should be injected into a decision making system. A human fuser can be placed at many different points in a decision making system, but by placing a human between the IF process and the DM, the DM’s
knowledge can be improved. Another major contribution of this research is the wide variety of mathematical models that can be used for different problem scenarios. This is important, because optimal mathematical models may not be usable for certain problem scenarios within the CIP. Furthermore, I introduce a new heuristic that can be used in place of the mathematical models when time is an issue. This heuristic can translate to other systems that use mathematical models to solve problems substantially faster while yielding reasonable results. Finally, I present a classification for the math models that helps a system implementer choose the model that best fits the scope of the problem or the dataset available. Along with the classification, a decision tree is presented that provides a structured way to determine whether combining models is the best solution for a problem scenario or dataset.

1.5 Organization of the Thesis

The introduction described the overall thesis by providing an overview of my motivation, methodology and anticipated research contributions. This thesis is a collection of three papers that address the three sub-problems. Chapter 2 is the first of three papers, titled Introducing the Cognitive Injection Process with Knapsack Problems. In Chapter 2, I introduce the methodology for the CIP and show, using knapsack problems and known knapsack heuristics, that a DM’s knowledge can be improved when a human fuser modifies the certainty values of data points. Chapter 3 is the second paper, titled Complex Mathematical Models within the Cognitive Injection Process. I present three new models; two use the DM to set a parameter, and the other combines the certainty values using entropy. I also present a new heuristic technique that can be used when a quick solution is needed. Chapter 4 is the final paper, titled Using Multiple Human Fusers within the Cognitive Injection Process. In this chapter, I discuss the use of learning and forgetting to bring multiple human fusers together to examine one dataset. I also present a model
that includes data degradation to represent how data reliability decreases with time. In Chapter 4, I also present a classification system for all the different models in order to structure the model selection process for a given problem scenario. I also include a decision tree, which allows users to combine models based on the problem scenario and datasets. Finally, in Chapter 5 I conclude the thesis and highlight some general contributions of this research, as well as areas outside the scope of this research that should be explored in the future.
Chapter 2: Introducing the Cognitive Injection Process with Knapsack Problems

2.1 Introduction

Technological advances have led to the abundant availability of data from heterogeneous sensors, which, when combined with knowledge about the sensed data, has improved our ability to turn data into reliable and relevant information. The process of combining data from different sources is termed data or information fusion (IF). In particular, a new area of focus has been on what the IF world calls hard and soft fusion, which stems from a classification system that divides sensors into two groups (Pravia, Prsanth, Arambel, Sidner, & Chong, 2008): hard (i.e., traditional electronic sensing sources such as satellites and RFIDs that use programmed systems to collect data) and soft (i.e., humans). Although soft sensing is clearly less predictable, it provides a different dimension to a dataset. Soldiers performing reconnaissance work traditionally are thought of as soft sensors, but with the availability of new communication methods (e.g., Facebook, Twitter, YouTube) any person could be a soft sensor in any setting. With the advent of crowd sourcing it is common to encounter the context of soft sensing. With the inclusion of humans as data sources, IF has had to be modified in order to combine sources with more uncertainty (soft sensors) with those that are slightly more reliable and driven by programs (hard sensors) through hard and soft fusion.

Many data fusion models have been developed and presented in the literature; among them, the JDL data fusion model (Steinberg & Bowman, 1999) is the most widely accepted. According to Steinberg, Bowman and White (1999), IF uses overlapping information to determine relationships among data, which can improve the estimates for a situation and therefore improve how one determines a COA. The goal of IF is to provide a DM with a better understanding of a situation by combining meaningful data from myriad sources. Furthermore,
with increased knowledge about data sources, data certainty values can provide DMs with additional knowledge. In particular, if the sensors have error models associated with them, IF can assign an error or uncertainty value for combined data from different sources.

Since IF is a process that uses mathematical models and rules to combine data from different sources in order to obtain information about a scenario, it is considered a hard process. The process yields a list of estimates, or pieces of knowledge that have been extracted by combining data from different sensors, that are used by the DM to determine a COA. Examples include a list of coordinates that help a military analyst determine where to deploy soldiers, or patient information that can help a nurse in a hospital to determine which patients need immediate attention. Although helpful, the lists of estimates still can contain too much information for a DM to process in the time he or she has available. By using the uncertainty values from the IF process in an uncertainty model (e.g., average and entropy) the estimate list can be sorted from most certain to least certain so the DM can focus on more certain estimates. Sorting estimates is also a hard process, as it uses the result of an uncertainty calculation to compare the estimates. Since both IF and the sorting process are hard processes, I want to explore what will happen if I add a soft process to the system. A soft process, much like soft sensors, is something that is not controlled only by formulas and rules. In this case, a soft process is induced by a subject matter expert (SME), who modifies and adds uncertainty data in a highly structured manner to the IF uncertainty results for different estimate attributes. The SME also determines the relative importance of the different attributes, which, when combined with modifications of the uncertainty values, changes the order of the estimate list, thereby helping the DM make a more informed decision.
2.1.1 Information Fusion

Information Fusion is the process of combining data from different sources to provide more concise and reliable information to DMs. In Figure 2.1, the heterogeneous sensors include: a video camera, a still camera and a human. These sensors are all retrieving vehicle data from an intersection. The objective is for the sensors to observe the make, model, size, and color of each car that drives through. The sensors are also collecting data on the number of passengers in each vehicle and their genders. These are the features, which are common categories or attributes that are seen across the different datasets that are being retrieved by each sensor. Each sensor has an error model included that decides if the reading for a piece of feature data is poor, fair or good. IF, which includes data alignment, association and state estimation, yields estimates with greater reliability than the original datasets including error values for each feature. IF can create many estimates by combining different datasets from different sensors.

![Figure 2.1 An example of information fusion.](image)

Advances in information technology (IT) have led to increased data collection and availability, which in turn necessitates improved and robust IF processes. Unfortunately, this has
caused terminological confusion and has led to the development of many different models (Salerno, 2002) in attempts to concisely explain IF. One of the most widely accepted models is the JDL data fusion model, shown in Figure 2.2 (Steinberg & Bowman, 1999). The JDL Data Fusion Group decided to use the term fusion “levels” to describe the different fusion processes that deal with refinement of objects, situations, threats and processes (White, 1988).

Figure 2.2  The JDL data fusion model (Steinberg, 1999).

The JDL model consists of four levels:

1. **Object Assessment**: Tracking, detecting, recognizing and identifying the objects.
2. **Situation Assessment**: Estimating and predicting the relationships between the objects.
3. **Impact Assessment**: Estimating and predicting the effects of the estimated and predicted actions.
4. **Process Refinement**: Modifying sensors (e.g., location) in order to acquire new data.

### 2.1.2 Including a Soft Process

Including a soft process (i.e., human intervention) helps a DM to select a particular COA. Figure 2.3 shows the interaction between hard and soft processes and hence hard and soft sensors. In the figure, the green areas are the hard processes, fusion and sensors, and the blue areas are the soft processes and sensors. The assumption is that many different sensors are retrieving data with error values on a problem that has already been recognized. The data is streaming in from hard sensors (H) and soft sensors (S) and is being sent to the IF process to obtain estimates, features and the corresponding certainty values. After the IF process, two different soft processes, namely feature weighting and certainty adjustments, are included. Feature weighting involves an SME using the Analytical Hierarchical Process (AHP) to assign relative importance values to the features extracted from the data during IF. AHP is a soft process because it requires the SME to do a pairwise comparison of all feature combinations. The calculated weights are used in conjunction with the certainty values to develop a ranked estimate list. The mathematical model takes the feature weights and the certainty values from the IF process and selects which features the SME should investigate. The SME then adjusts the certainty values for the features in the estimates if he or she believes that the current values are incorrect. The adjusted certainty values and the feature weights are combined to obtain the ranked estimate list that the DM uses to determine the COA. If the DM does not decide on a COA, the certainty adjustment process continues until the DM has determined the proper COA for the situation. This complete system is the Cognitive Injection Process, or CIP.
In the following sections, I explore the idea of adding a soft process to a system in order to improve decision making. In section 2.2, I illustrate the use of CIP in an example problem. In section 2.3, I explore the literature on IF, different sensor types, and systems that attempt to incorporate humans. In section 2.4, I discuss methodology and present a numerical example and three mathematical models that I developed. In section 2.5, I test the results of the mathematical models using different metric calculations. Finally, in section 2.6 I provide a summary for this chapter and discuss future research opportunities.

2.2 Problem Discussion

Assume that an intersection has five sensors (Figure 2.4), four of which are hard sensors: a video camera, a still photo camera, a speedometer, and a panoramic photo camera. The fifth (soft) sensor is a human. The objective is for the sensors to relay data on the cars that pass through the intersection so that a DM can determine if yield signs should be changed to stop signs. The DM determines a COA by examining data on actual accidents and near misses. Each sensor retrieves data on accidents and near-miss accidents, each with its own error function. All
of the data is sent to the IF process, where the features, estimates and corresponding certainties are realized.

Figure 2.4 An example situation.

For this example problem, the output generated by the IF process is shown in Table 2.1. The output includes 20 estimates (E1 … E20) and three features (F1, F2, and F3). The center columns of Table 2.1 represent the five sensors (H1, H2, H3, H4, and S1) and when the data they send is used to create each estimate. For example in E12, data from H2 and S1 are used by the IF process to determine the estimates and certainty values for F1 and F2 within E12. The feature columns of Table 2.1 represent the certainty values for each feature in each estimate. These values are all between 0 and 1, where higher values indicate more certainty of the feature-estimate (F-E). The F-E combinations with the value 0.00 were not realized by the IF process; the F-E is not completely uncertain. The certainty values are created when the IF process aligns, associates and estimates error functions to represent each sensor.
2.3 Background Literature

Previous research in this field is limited, as the idea of including humans in data processing systems is a fairly new concept. A substantial amount of research exists on hard sensors and their application areas. Most of the research on soft sensing is focused on cognition, specifically the psychology of how a human senses. A large body of literature also exists on IF, including many different types of data fusion models and application areas for their use.
2.3.1 Hard Sensors

A hard sensor is any device that relays information about an event using physics-based equations. Hard sensors can be radar devices, cameras, telephones, etc. Due to advancements in wireless communication, wireless sensor networks (WSNs) have become a focus of hard sensor research and are being used in various applications. Area monitoring is among the most common; it involves placing sensors in a region to track events that occur. Early uses included tracking seismic and animal activity in a body of water (Nishimura & Conlon, 1994). Later, WSNs were used to track vessels such as submarines in bodies of water. Environmental monitoring is a second type of application using WSNs (Chong & Kumar, 2003; Hart, 2006). Sensors in this application area have been used to monitor temperature and moisture levels in greenhouses or even to monitor soil to detect possible landslides. A third area in which WSNs are used is machine health monitoring (Low, 2005; Tiwari, Balla & Lewis, 2005). Sensors are placed within machines to monitor how they are functioning and to signal if one needs to be repaired. With sensor networks being applied to so many different areas, research has begun to focus on sensor placement and selection (Isler, 2005; Gupta, Chung, Hassibi & Murray, 2006). This is important, since hard sensors do not have an infinite detection range and can be expensive.

Although hard sensors are continuously evolving and becoming more reliable, uncertainty still exists. Uncertainty about retrieved data and resultant incorrect actions are of major concern. Fault tolerance has been a major focus in trying to address hard sensor uncertainty (Shen, 2001; Hoblos, 2000). Techniques such as transfer belief functions, Bayesian modeling and information entropy have been used to try to filter out uncertain data and will be discussed further in the following sections.
2.3.2 Soft Sensors

While soft sensing has existed since the dawn of humanity, it is a fairly new focus area for research. Soft sensing involves using people as sensors to relay data on a situation. Examples of situations in which soft sensing is used are interrogations, military reconnaissance or even during a poker match when a person is trying to guess the values of the other players’ cards. Another example is when a crowd relays information about a disaster situation that can help FEMA to reconstruct it. Much like hard sensors, soft sensors have an uncertainty associated with the retrieved data; however, since soft sensors do not rely on physics-based formulas, the uncertainty can be much greater and harder to quantify. Scholars have attempted to determine how to calculate the uncertainty of a human’s situation report. Simons’ (1999) study involved a person watching a video of three people passing a basketball around followed by a random event, such as a person in a gorilla suit walking through the court, while the observer was trying to count the number of passes that had occurred. He varied the random event, the passing sequence performed by the players and the color of the basketball players’ shirts. In the end, he found that 46% of the observers did not sense the random event. One area where human uncertainty sensing has become a focus is legal settings. Psychologists have found that a human’s sensing ability is affected by stressful events (Kassin, 2001; Christianson, 1992). For example, when people are asked to recall events in situations involving guns, their sensing abilities decreased when compared to situations without weapons. Many factors can affect how humans sense situations (Liu, Lieberman, & Selker, 2003) making quantifying the associated uncertainty value a difficult task. Liu (2010) illustrated different techniques such as Dempster-Shafer theory, entropy and human modeling using fuzzy logic, to calculate human uncertainty.
2.3.3 Data Fusion

Data fusion is the process of combining data to make inferences about a situation while improving accuracy (Hall & Llinas, 1997). Data fusion was developed in the 1980s primarily for military applications, but terminological differences among researchers created confusion that was eventually sorted out by the Data Fusion Subpanel of the Joint Directors of Laboratories (JDL) (White, 1991). The JDL reconvened later to further solidify specifics of data fusion, in particular, its five levels (Steinberg, Bowman & White, 1999). Data fusion remains an important tool in military settings for tasks such as recognizing targets (Hall, Linn, & Llinas, 1991), obtaining situational awareness and assessing impact. Although a number of other fusion frameworks were proposed (i.e., Endsley, 1995; Salerno, Hinman & Boulware 2004), the fundamental question remained: Where and how should a human fuser be injected into the overall IF system? With increased connectivity and technological advancements, scholars began to examine hard and soft data fusion (Hall, Llinas, McNeese, & Mullen, 2008), creating new literature streams on subjects such as how to associate uncertainty values with heterogeneous data.

2.4 Methodology

Considering the example shown in Figure 2.4 and Table 2.1, assume that F-E certainty values are known. The responsibility for maximizing the overall certainty of the F-Es lies with the SME (human fuser), who must first identify the F-Es to be considered and modified. However, it may not be feasible to evaluate all possible F-Es due to time constraints. Instead, F-E selection can be formulated as an optimization problem. In this section, I present three mathematical models and provide a detailed explanation of system mechanics. The objective of these models is to determine which F-E combinations should be inspected and modified by the
human fuser to improve estimate certainty. All of the models incorporate similar variables, constraints and parameters, but vary significantly in the objective function. After introducing the sets, variables, and parameter notations used in the models, I provide details on each, including their transformations into knapsack problems. Finally, I provide an example of a system implementation.

2.4.1 Sets, Variables, and Parameters

2.4.1.1 Sets

The sets for the formulations represent the features and estimates obtained from the IF process and include the number iterations.

\[ i = 1, 2, \ldots, F, \text{ where } F \text{ is the number of features seen among all the estimates;} \]
\[ j = 1, 2, \ldots, E, \text{ where } E \text{ is the number of estimates extracted by the fusion process; and} \]
\[ k = 0, 1, \ldots, IT, \text{ where } IT \text{ is the maximum number of iterations and } k=0 \text{ is the initial data.} \]

2.4.1.2 Decision Variables

The decision variables \( X \) and \( Y \) represent two different actions. \( X_{ij} \) represents a feature \( i \) in estimate \( j \) that the DM should evaluate that does not have a certainty value assigned to it. Any F-E combination with a value of 0.00 in Table 2.1 is a candidate for \( X_{ij} \). \( Y_{ij} \) represents a feature \( i \) in estimate \( j \) that has already been assigned a certainty value from the IF process, but needs to be revised. Any F-E combination with a value greater than 0.00 in Table 2.1 is a candidate for \( Y_{ij} \).

\[ X_{ij} = \begin{cases} 1, & \text{If feature } i \text{ in estimate } j \text{ should be evaluated by the human fuser for the first time} \\ 0, & \text{Otherwise} \end{cases} \]

\[ Y_{ij} = \begin{cases} 1, & \text{If feature } i \text{ in estimate } j \text{ should be reevaluated by the human fuser} \\ 0, & \text{Otherwise} \end{cases} \]

2.4.1.3 Parameters

Model parameters \( U_{ijk} \), \( W_{ij} \) and \( P_{ij} \) are the input values that drive the models. Parameters \( D_{ijk} \), \( H_{ijk} \), \( C_{jk} \) and \( R_{jk} \) are all determined by model outputs or mathematical formulas. The last
parameter, \( T \), is the only parameter determined by the model user (i.e., a human fuser or DM).

Three of these parameters, \( U_{ijk} \), \( D_{ijk} \) and \( H_{ijk} \), change after each iteration.

\( U_{ijk} = \) The certainty of feature \( i \) in estimate \( j \) for iteration \( k \), where \( 0 < U_{ijk} < 1 \)

\( W_i = \) The weight of importance of feature \( i \), where \( \sum_{i=1}^{F} W_i = 1 \)

\( P_{ij} = \) The processing time for the evaluation of feature \( i \) in estimate \( j \) by the human fuser

\[
D_{ijk} = \begin{cases} 
1, & \text{If feature } i \text{ in estimate } j \text{ for iteration } k \text{ has a certainty value assigned to it} \\
0, & \text{Otherwise}
\end{cases}
\]

\[
H_{ijk} = \begin{cases} 
1, & \text{If feature } i \text{ in estimate } j \text{ for iteration } k \text{ has been evaluated by the human fuser} \\
0, & \text{Otherwise}
\end{cases}
\]

\[
C_{jk} = \frac{\sum_{i=1}^{F} (X_{ij}Y_{ij} + Y_{ij}(1-U_{ijk}))}{F}, \text{ where the certainty formula is the average of all features for an estimate}
\]

\[
R_{jk} = \sum_{i=1}^{F} W_i (X_{ij}), \text{ where reliability is the sum of the weights of realized features for an estimate}
\]

\( T = \) Amount of time available for the human fuser to update certainty values

### 2.4.2 Constraints

Four constraints are used in each of the three models. The first constraint (2.A) is a processing time constraint for the human fuser.

\[
\sum_{j=1}^{E} \sum_{i=1}^{F} (X_{ij} + \frac{Y_{ij}}{2}) P_{ij} \leq T \tag{2.A}
\]

It sums the processing time for every first time observation and halves the processing time for every existing observation that is modified. The assumption here is that when a human fuser modifies an existing certainty value \( Y_{ij} \) it will take half the time of assigning a certainty value for the first time \( X_{ij} \).
The second constraint (2.B) ensures that a new observation ($X_{ij}$) can only occur if a certainty value has not been assigned to that F-E combination ($D_{ijk}$). If feature $i$ in estimate $j$ for iteration $k$ has a certainty value assigned to it ($D_{ijk} = 1$) then $X$ will automatically be set to 0; otherwise, $X$ can be set to 1. Similarly, $Y$ can only exist when a certainty value has been assigned to an F-E combination (2.C). If feature $i$ in estimate $j$ for iteration $k$ does not have a certainty value assigned to it ($D_{ijk} = 0$), then $Y$ will automatically be set to 0; otherwise, $Y$ can be set to 1.

$$X_{ij} \leq 1 - D_{ijk} \quad \forall \ i, j \quad 2.B$$

$$Y_{ij} \leq D_{ijk} \quad \forall \ i, j \quad 2.C$$

The last constraint found in all four models (1.D) is one that only allows the model to choose an F-E combination that has not yet been evaluated by the human fuser. Assume that when a human fuser has modified or assigned a certainty value for an F-E combination, he or she has used all the information and knowledge available to determine the new certainty value. This means that unless the human fuser changes, there is no reason to evaluate an F-E combination more than once.

2.4.3 Why Propose Three Models?

I propose three models to be used within the system because each model allows the DM to customize what is important to him or her when evaluating the estimates. Model 1 is used when the DM believes that all features are equally important. This means that the feature weights are all equal and do not need to be included in the formulation. Model 2 allows the importance of the features to be prioritized over certainty values. Thus, whether the certainty value is 0.01 or 1.00, the feature weight value does not affect the outcome. In Model 3, the
certainty values and feature weights are equally important, thus the weights are combined in the certainty calculation.

2.4.4 Model 1 - Estimate Certainty Model (ECM)

The first model is the simplest case and only uses the certainty of each estimate. The objective is to maximize the total certainty for all estimates. The only constraints are the four mentioned in the previous section, plus the certainty gain calculation \( C_j \), which is the average of the feature certainties for each estimate.

Maximize \[ Z = \sum_{j=1}^{E} C_j \] \hspace{1cm} 2.E

Subject to:

\[ C_j = \frac{\sum_{i=1}^{F} (X_{ij} + Y_{ij}(1 - U_{ijk}))}{F} \hspace{1cm} \forall \ j \] \hspace{1cm} 2.F

\[ \sum_{j=1}^{E} \sum_{i=1}^{F} (X_{ij} + \frac{Y_{ij}}{2}) P_{ij} \leq T \] \hspace{1cm} 2.A

\[ X_{ij} \leq 1 - D_{ijk} \hspace{1cm} \forall \ i, j \] \hspace{1cm} 2.B

\[ Y_{ij} \leq D_{ijk} \hspace{1cm} \forall \ i, j \] \hspace{1cm} 2.C

\[ X_{ij} + Y_{ij} \leq 1 - H_{ijk} \hspace{1cm} \forall \ i, j \] \hspace{1cm} 2.D

2.4.5 Model 2 - Estimate Certainty and Reliability Model (ECRM)

The second model is exactly like the first except it incorporates reliability into the objective function. This allows the model to take into account the importance of each feature assigned by the DM (the calculated feature weight). The reliability gain calculation \( R_j \) only includes new observations, because if an existing certainty is chosen to be observed \( Y \) there is
no gain in reliability; the gains are realized in certainty values only, because certainty and reliability are summed in this model.

\[
\text{Maximize } Z = \sum_{j=1}^{E} (C_j + R_j) \quad 2.G
\]

Subject to:

\[
R_j = \sum_{i=1}^{F} W_i X_{ij} \quad \forall j \quad 2.H
\]

\[
C_j = \frac{\sum_{i=1}^{F} \left( X_{ij} + Y_{ij} (1 - U_{ijk}) \right) F}{\sum_{i=1}^{E} \sum_{j=1}^{F} (X_{ij} + \frac{Y_{ij}}{2}) P_{ij} \leq T} \quad \forall j \quad 2.F
\]

2.4.6 Model 3 – Weighted Estimate Certainty Model (WECM)

The third model combines the certainty and reliability values into one by adding the weight values into the certainty formula. Thus, more importance is assigned to the certainty values of the F-E combinations, because the certainty value impacts the weights as a multiplier.

\[
\text{Maximize } Z = \sum_{j=1}^{E} C_j \quad 2.E
\]

Subject to:

\[
X_{ij} \leq 1 - D_{ijk} \quad \forall i, j \quad 2.B
\]

\[
Y_{ij} \leq D_{ijk} \quad \forall i, j \quad 2.C
\]

\[
X_{ij} + Y_{ij} \leq 1 - H_{ijk} \quad \forall i, j \quad 2.D
\]
Converting to a Knapsack Problem

All of these models can be converted into knapsack problems which allows the results to be compared to known heuristics, thereby decreasing the solution time needed by CPLEX by reducing the number of constraints to be processed. First, the four constraints must be condensed into one, as the traditional knapsack problem has only one constraint. This is accomplished by adding conditions to the summation terms:

\[
\sum_{i,j} P_{ij} \times X_{ij} + \sum_{i,j} \frac{P_{ij} \times Y_{ij}}{2} \leq T
\]  

2.J

By adding the conditions, constraints 2.B through 2.D (which ensured that X could be set to 1 only if it was a new observation, and Y could be set to 1 only if it was modification) do not need to be included. The certainty and reliability formulas (equations 2.E – 2.I) are used for the objective function instead of the certainty and reliability parameters, including the same conditional summations. Objective functions for Models 1, 2 and 3 are shown below:

Model 1  
\[
\max Z = \sum_{i,j} X_{ij} + \sum_{i,j} \left(1 - U_{ij}\right) \times Y_{ij}
\]

2.K
Model 2 \[ \max Z = \sum_{i,j} (1 + W_i) \cdot X_{ij} + \sum_{i,j} (1 - U_{ij}) \cdot Y_{ij} \]  

Model 3 \[ \max Z = \sum_{i,j} W_i \cdot X_{ij} + \sum_{i,j} W_i \cdot (1 - U_{ij}) \cdot Y_{ij} \]

2.4.8 Example System Description

Figure 2.5 shows the system after the data has been transformed by the IF process in the following steps:

1. The list of features is sent to the human fuser to determine their weights;
2. The certainty values are input into the mathematical model; and
3. An initial list of estimates is provided to the DM in case a COA needs to be implemented immediately without human fuser adjustments.

Figure 2.5 Detailed system flow.

2.4.8.1 Feature Weights
During the IF process, a list of features is sent to the human fuser who performs pairwise comparisons, and then the AHP process is used to calculate the weights of each feature. AHP was chosen because it also considers a consistency value to ensure that the human fuser does not give confounding comparisons such as (F1 > F2, F2 > F3, F3 > F1). AHP does have its critics that focus on problems with rank reversal, but the focus of this research is not using AHP but using any weighting system to differentiate between the importances of each feature. In AHP, values between 1 and 9 are used to represent relative importance, with 1 indicating no preference and 9 indicating a significant preference for one of the two features. The diagonals of the comparison matrix represent a comparison of a feature to itself so those values are all 1. Table 2.2 shows a comparison matrix that has not been completed and Table 2.3 is a comparison matrix that has been filled out by a human fuser.

<table>
<thead>
<tr>
<th>Features</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>1</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td>F2</td>
<td>---</td>
<td>1</td>
<td>---</td>
</tr>
<tr>
<td>F3</td>
<td>---</td>
<td>---</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2.2 Empty AHP Table

<table>
<thead>
<tr>
<th>Features</th>
<th>F1</th>
<th>F2</th>
<th>F3</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>1</td>
<td>0.333</td>
<td>5</td>
</tr>
<tr>
<td>F2</td>
<td>3</td>
<td>1</td>
<td>7</td>
</tr>
<tr>
<td>F3</td>
<td>0.2</td>
<td>0.142857</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2.3 Completed AHP Table

In the table cell at the intersection of row F2 and column F1, there is a value of 3. This means that the human fuser prefers Feature 2 three times as much as Feature 1. Likewise, the inverse value appears in the cell at the intersection of row F1 and column F2; that is, the human
fuser prefers Feature 1 one-third as much as Feature 2. To calculate the weight values, the steps shown in

Figure 2.6 must be followed.

Let \( M_{ij} \) be the matrix value of row \( i \) in column \( j \):

**Step 1:** Find the total sum of each column (\( COL_j = \sum_{i=1}^{3} M_{ij} \)).

**Step 2:** Divide the values of that column by the column’s sum to normalize the matrix

\[
(M_{ij}^N = \frac{M_{ij}}{COL_j}).
\]

**Step 3:** Calculate the Eigen vector \( (W_l = \sum_{j=1}^{3} M_{ij}^N / 3) \), representing the weights of each feature.

**Step 4:** Calculate \( \lambda_{\text{max}} \) value for the consistency index (CI). This is accomplished by summing the product of the weights and the reciprocal of the normalized diagonals

\[
(\lambda_{\text{max}} = \sum_{i=1}^{3} W_l \times \frac{1}{M_{ii}^N}).
\]

**Step 5:** Calculate CI \( (CI = \frac{\lambda_{\text{max}} - \text{Number of Features}}{\text{Number of Features} - 1}) \).
Step 6: Calculate the consistency ratio (CR). This is done by dividing the CI by the random index value (RI) which is found by using a look-up table and the number of features.

Step 7: If the CR value is lower than 0.1 (which in this example is true), the inputs into the AHP are considered consistent; otherwise, modifications must be made and the process starts again at Step 1.

2.4.9 Human Fusion Iteration

Once the weights are determined, the model can be run to select F-Es for the human fuser to evaluate. In this example, the first model described in section 2.4.4 is being used with a $T$-value of 15 units, the inputs from Table 2.1, the weights from above, and a set of processing times which are random values between 1 and 10 (Table 2.4). Based on the inputs, the model generates the F-E combinations that will most improve the overall certainty values of the estimates. In Table 2.5, the yellow highlighted cells represent required new observations ($X$) and green highlighted cells represent required modifications to current certainty values ($Y$).

<table>
<thead>
<tr>
<th>Processing Times</th>
<th>Estimates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Features</td>
<td>E1</td>
</tr>
<tr>
<td>F1</td>
<td>5</td>
</tr>
<tr>
<td>F2</td>
<td>7</td>
</tr>
<tr>
<td>F3</td>
<td>9</td>
</tr>
</tbody>
</table>

| Table 2.4 Processing Times |

To validate that the model is working properly, the results must be evaluated in order to determine why the model selected the specific F-E combinations. Since the model assumes that the human fuser will increase the certainty of an F-E to 1, then the biggest gain for Model 1
would be to make a new observation (X) rather than a modification (Y) if the processing times are similar. This is because Model 1’s objective function only deals with the certainty function; by increasing a certainty value of a feature by 1, the estimate certainty and in turn the objective function, are increased by 1/3. After the first iteration, the model suggests making eight new observations and modifying the certainty value for F1-E20. At first, this does not seem correct, as the model has chosen to modify a certainty value; however, upon further inspection, the eight new observations require a total processing time of 14 units, which leaves 1 unit available. Since none of the remaining F-E combinations requiring new observations have a processing time of 1 unit, the model chooses one of the combinations with an existing certainty value. Since the processing time for a modification (Y) is half of what is listed in Table 2.4, any F-E combination with a processing time of 1 or 2 that already has a certainty value can also be included in the solution. This leaves two choices: F2-E8 and F1-E20. Both have processing times of 2, and both already have certainty values. As with new observations, the model assumes that the human fuser will increase the certainty to 1. With current certainty values of 0.70 and 0.62 respectively, increasing F1-E20 increases the objective the most. This validation was performed for all three models at all iteration stages to ensure accuracy.

Using Table 2.5, the human fuser modifies the certainty values of the F-E combinations by evaluating the estimates and their sources, the different sensors. The resulting certainty table after the modifications is presented in Table 2.6. These modification values are called ground truth values. Assume that the ideal certainty value for each F-E is 1, or 100% certainty that the information is correct. When the mathematical model suggests modifying or making a new observation, the certainty value is changed to this ground truth value of 1.

Table 2.6 Modified Certainty Values
The next step in Figure 2.5 is calculating the ranks for each estimate by using the updated certainty values from Table 2.6. The ranks are determined by calculating certainty and reliability for each estimate. The certainty of an estimate $j$ for iteration $k$, $CER_{jk}$, is the average over all the features. The computation for the certainty of each estimate $j$ and iteration $k$ is given by:

$$CER_{jk} = \frac{\sum_{i=F1}^{F3} U_{ijk}}{\text{Number of Features}} \quad \forall \ j = E1, ..., E20 \text{ and } k = 1, ..., IT$$

where $U_{ijk}$ are the certainty values. Reliability values ($REL_{jk}$) represent how much of the feature weights estimate $j$ covers for iteration $k$. Therefore,

$$REL_{jk} = \sum_{i=F1}^{F3} W_i * D_{ijk} \quad \forall \ j = E1, ..., E20 \text{ and } k = 1, ..., IT$$

where $W_i$ represents the weight of feature $i$ found in section 0, and $D_{ij}$ represents a feature $i$ in estimate $j$ with an associated uncertainty value. The reliability term is included in the ranking process to allow a DM to see uncertain estimates including all features that otherwise are eliminated in favor of more certain estimates with less features. The rank of the estimates is then determined by ordering them based on the sums of the certainty and reliability values. Higher values correspond to more certain and reliable estimates, which are assigned lower rankings (the best estimate is assigned rank 1). The certainty, reliability and ranks for each estimate based on initial data from the IF process and after the human fuser’s first iteration are shown in Table 2.7 and
Table 2.8, respectively.

Table 2.7 Initial Ranked List

|     | E1 | E2 | E3 | E4 | E5 | E6 | E7 | E8 | E9 | E10 | E11 | E12 | E13 | E14 | E15 | E16 | E17 | E18 | E19 | E20 |
|-----|----|----|----|----|----|----|----|----|----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| REL | 0.64| 0.00| 0.00| 0.00| 0.64| 0.07| 0.28| 0.00| 0.64| 0.00| 0.00| 0.64| 0.93| 0.00| 0.64| 0.28| 0.00| 0.28| 0.07| 0.07| 0.28|
| CER | 0.33| 0.00| 0.00| 0.11| 0.17| 0.14| 0.00| 0.23| 0.00| 0.00| 0.06| 0.32| 0.00| 0.27| 0.18| 0.00| 0.14| 0.17| 0.08| 0.21|
| SUM | 0.98| 0.00| 0.00| 0.76| 0.24| 0.42| 0.00| 0.88| 0.00| 0.00| 0.70| 1.25| 0.00| 0.91| 0.46| 0.00| 0.43| 0.24| 0.15| 0.49|
| RANK|  2 |   -|   -|   5|   12| 10  |   -|   4 |   - |   -| 6   |   1 |   -|   3 |   8 |   - |   9 |  11 |  13 |   7 |

Table 2.8 Iteration 1 Ranking

|     | E1 | E2 | E3 | E4 | E5 | E6 | E7 | E8 | E9 | E10 | E11 | E12 | E13 | E14 | E15 | E16 | E17 | E18 | E19 | E20 |
|-----|----|----|----|----|----|----|----|----|----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| REL | 0.64| 0.00| 0.28| 0.64| 0.07| 0.28| 0.28| 0.72| 0.00| 0.93| 0.93| 0.93| 0.00| 0.72| 0.36| 0.00| 0.28| 0.07| 0.07| 0.28|
| CER | 0.33| 0.00| 0.33| 0.11| 0.17| 0.14| 0.33| 0.37| 0.00| 0.07| 0.39| 0.32| 0.00| 0.60| 0.51| 0.00| 0.14| 0.17| 0.08| 0.33|
| SUM | 0.98| 0.00| 0.62| 0.76| 0.24| 0.42| 0.62| 1.28| 0.00| 1.59| 1.32| 1.25| 0.00| 1.32| 0.87| 0.00| 0.42| 0.24| 0.15| 0.62|
| RANK|   6|   -|   9 |   8 |  15|  13 |   9 |   4 |   - |   1|  3   |  5   |   2 |   7 |   - |  12 |  14 |  16 |   9 |

The estimates without certainty values assigned to any of their features are not ranked. The DM does not see these tables, but rather sees the estimates in rank order (Figure 2.7), which is updated for the DM after each iteration until an action is performed by the DM or until every F-E combination has been inspected by the human fuser.
2.5 Metrics and Analysis

The goal of injecting soft processes into a system that typically is comprised of only hard processes is to show that a DM’s knowledge of a situation can be improved before determining a COA. Improvement is demonstrated in the following ways:

1. Mathematical formulations will be closer to optimality in terms of largest knowledge gain when comparing them to known heuristic selection techniques for knapsack type problems. This will show that using the proposed math models is ideal rather than using random, greedy or bang-for-buck approaches.

2. The list received by the DM contains human fuser input and provides more knowledge than the list with no human fuser input (i.e., the initial list from the IF process). There is always a gain in information, proving that there is a change in the available knowledge for the DM when determining the COA.
In this section, I introduce the three comparison heuristics for the proposed model. Then, I discuss the metric calculations and analyze the results of the three models, proving the two statements above.

2.5.1 Knapsack Heuristics

The mathematical formulations all can be converted to knapsack problems, allowing them to be compared to known heuristics, including random (Vose, 1999), greedy (Stinson, 1999), and bang–for-buck (Nauss, 1976) selection. Using a heuristic as opposed to an optimization model presents a trade-off between a better solution and a faster solution time. Comparing each model to the three heuristics, the mathematical models yield better results. It is also important to assess solution time, since as the variable size increases, the solution time for the mathematical models grows in polynomial time (Zemel, 1980); thus, there may be a point at which using a heuristic becomes more time efficient.

All three heuristics have the same flow (Figure 2.8), each with a different selection rule. The selection rule changes based not only on the heuristic, but also on the comparison model. Since each model has a different objective function, the rules must change in the heuristics in order to be able to compare them. This will be further explained in each heuristic subsection.
The heuristic starts by initializing the available time to inspect F-Es (T) and then an F-E is selected by one of the selection rules (random, greedy or bang-for-buck). It proceeds if the F-E is new and appears on the list to be inspected by the human fuser. If the chosen F-E has a certainty of 1, then another F-E must be selected, as the human fuser cannot improve that value. If the F-E has a certainty value that is not 1, it must be determined whether or not the certainty value is 0 in order to subtract the correct amount of processing time for either a new observation (X) or a modification (Y). It must be confirmed that the available time is greater than 0 to ensure that the human fuser will have enough time to check that F-E. If T is positive, then the F-E is stored, and another random F-E is selected; otherwise, the processing time must be added. If all F-E combinations have already been checked, the list of chosen F-Es are sent to the human fuser to be updated; otherwise a new F-E is selected.

2.5.1.1 Random Selection Rule

An estimate and feature are randomly chosen. Prioritization is not given to lower processing times or those that could potentially provide the most gain. The random selection rule does not change for each model since the selection process does not take anything into account and is completely random.

2.5.1.2 Greedy Selection Rule

For the greedy selection rule, the F-E with the largest gap from the maximum certainty value is chosen. That is, when compared to Model 1, the F-E that is furthest away from a value of 1 would provide the largest gain. The optimal value is 1 because the best any F-E can be is 100% certain. When compared to Model 2, the greedy selection rule requires the feature weights
to be included, since the objective is to incorporate both certainty and reliability. A comparison with Model 3 involves multiplying the certainty distance by the feature weight.

<table>
<thead>
<tr>
<th>Model</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 1</td>
<td>Choose the F-E with the largest ((1 - U_{ijk}))</td>
</tr>
<tr>
<td>Model 2</td>
<td>Choose the F-E with the largest (((1 - U_{ijk}) + W_i))</td>
</tr>
<tr>
<td>Model 3</td>
<td>Choose the F-E with the largest (((1 - U_{ijk}) \times W_i))</td>
</tr>
</tbody>
</table>

This selection process searches down through the estimates (start at E1) and across through the features (start at F1). If two F-Es are tied for the largest gap, the first F-E found is selected.

2.5.1.3 Bang-for-Buck Selection Rule

When using the bang–for-buck rule, both the largest gap and the processing time are taken into account. First, the largest gaps are found and then divided by the processing times. The \(P\) values in the rule correspond to the processing times: \(P_{ij}\) for all new observations \((X)\) and \(P_{ij} / 2\) for all modifications \((Y)\). The largest values result when \(U_{ijk}\) is closest to 0 and \(P_{ij}\) is small.

<table>
<thead>
<tr>
<th>Model</th>
<th>Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model 1</td>
<td>Choose the feature-estimate with the largest ((1 - U_{ijk}) / P)</td>
</tr>
<tr>
<td>Model 2</td>
<td>Choose the feature-estimate with the largest (((1 - U_{ijk}) + W_i) / P)</td>
</tr>
<tr>
<td>Model 3</td>
<td>Choose the feature-estimate with the largest (((1 - U_{ijk}) \times W_i) / P)</td>
</tr>
</tbody>
</table>

2.5.2 Metrics

Metric variables represent the distance from the current solution to the ground truth solution where all \(U_{ijk}\) are assigned a value of 1. These variables are defined as \(V_N^{MM}\), where \(N\)
represents the objective of Model 1, 2 or 3 (ECM, E CRM, and WECM, respectively) and MM represents one of the four different methods that can be used to solve the problem (math formulation (*), random (R), greedy (G) or bang for buck (B)). The variable definitions are as follows:

\[ V_{1MM} = 60 - \sum_{i=1}^{F} \sum_{j=1}^{E} U_{ij} \]

\[ V_{2MM} = 80 - \left[ \sum_{i=1}^{F} \sum_{j=1}^{E} U_{ij} + \sum_{i,j} W_i \right] \]

\[ V_{3MM} = 20 - \sum_{i=1}^{F} \sum_{j=1}^{E} U_{ij} \times W_i \]

\[ V_{1MM} \] is used when the objective is the sum of all the certainties (Model 1). The maximum objective value is equivalent to all F-E certainties \((U_{ij})\) being set to a value of 1. In this example, the maximum objective value is 60 because there are 60 \(U_{ij}\)’s (20 estimates x 3 features). This means the distance from the maximum objective value is 60 minus the sum of all the certainty values. Similarly, when using the objective from Model 2 (sum of certainty and reliability) the maximum becomes 80 because reliability is included in the objective function. Since the sum of the weights must equal 1 \((\sum_{i=1}^{f} W_i = 1)\) and there are 20 estimates, there is an increase of 20 in the maximum objective value. So, for the Model 2 objective, the distance from the maximum is 80 minus the sum of all the certainties and reliabilities where the certainty value is not 0 \((1V_{2MM})\). Finally, for the objective from Model 3 (weighted certainty) the maximum is 20 \((V_{3MM})\). Since the formulation multiplies the weights by the certainties, each estimate has a maximum value of 1; there are 20 estimates in this example problem.
While these $V$ values provide a sense of distance from the optimal situation and gain from each iteration, it is also necessary to compare the mathematical models to the heuristics. To do this, the percent difference between the models ($V_N^*$) and the three heuristics must be found:

$$p_{MM}^N = \frac{V_N^{MM} - V_N^*}{V_N^{MM}} \times 100$$

This calculation quantifies how much closer the mathematical models are to the three heuristics and shows that for solvable problem sizes, using the mathematical models is always superior to using a heuristic.

2.5.3 Analysis

To analyze the model, I ran it 20 times while randomizing the certainty values, processing times and feature weights for each run. These inputs were used to compare the mathematical models and the heuristics. Table 2.11 shows the results of the 20 runs using the objective from Model 1 (sum of all certainty values). I ran the random heuristic 10 times for each set of inputs and calculated the average, since different results were obtained each time.

Table 2.11 Metric Values for Model 1
Conversely, the greedy and bang-for-buck heuristics yielded the same results as long as the inputs did not change. Table 2.11 shows the average performance of the heuristic selection rules for each of the models. The $V$ values from the distance equations are shown on the left side of the table for each run and the percent difference values are shown on the right side. As expected, the models always outperform the heuristics in terms of the objective value ($V$). I can also conclude that the mathematical model outperforms bang-for-buck selection, which outperforms greedy selection, which outperforms random selection.

<table>
<thead>
<tr>
<th>RUN</th>
<th>AVG VR</th>
<th>VG</th>
<th>VB</th>
<th>VSTAR</th>
<th>RUN</th>
<th>AVG VR</th>
<th>VG</th>
<th>VB</th>
<th>VSTAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>49.30281</td>
<td>47.6856</td>
<td>43.53499</td>
<td>43.54399</td>
<td>1</td>
<td>31.08426</td>
<td>32.68751</td>
<td>0.960579</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>43.79003</td>
<td>40.39084</td>
<td>34.2789</td>
<td>34.2789</td>
<td>2</td>
<td>41.96245</td>
<td>35.07203</td>
<td>0.807496</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>43.7881</td>
<td>43.29306</td>
<td>37.29102</td>
<td>37.29705</td>
<td>3</td>
<td>36.39601</td>
<td>28.46941</td>
<td>1.780848</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>41.75789</td>
<td>40.63919</td>
<td>34.80829</td>
<td>34.80829</td>
<td>4</td>
<td>43.90207</td>
<td>40.99241</td>
<td>4.27916</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>38.85038</td>
<td>38.16687</td>
<td>30.55591</td>
<td>30.55951</td>
<td>5</td>
<td>43.29482</td>
<td>33.02739</td>
<td>4.741574</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>42.2063</td>
<td>42.90045</td>
<td>37.60123</td>
<td>37.46136</td>
<td>6</td>
<td>37.98099</td>
<td>37.17617</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>39.40127</td>
<td>38.54733</td>
<td>32.72693</td>
<td>32.64914</td>
<td>7</td>
<td>41.37619</td>
<td>44.06528</td>
<td>2.714898</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>40.22541</td>
<td>38.80176</td>
<td>34.02951</td>
<td>34.02951</td>
<td>8</td>
<td>41.99229</td>
<td>39.19813</td>
<td>4.037126</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>39.70618</td>
<td>36.94943</td>
<td>31.84702</td>
<td>31.3967</td>
<td>9</td>
<td>43.0586</td>
<td>41.35232</td>
<td>3.179175</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>45.35556</td>
<td>44.33801</td>
<td>37.42084</td>
<td>37.42084</td>
<td>10</td>
<td>43.71619</td>
<td>34.57878</td>
<td>3.814399</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>44.15336</td>
<td>43.97446</td>
<td>37.48103</td>
<td>37.41678</td>
<td>11</td>
<td>38.87684</td>
<td>33.60866</td>
<td>4.446862</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>44.60714</td>
<td>42.48232</td>
<td>38.58789</td>
<td>38.58789</td>
<td>12</td>
<td>48.50979</td>
<td>40.90673</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>38.1555</td>
<td>36.94022</td>
<td>31.66695</td>
<td>31.6355</td>
<td>13</td>
<td>40.71246</td>
<td>39.25928</td>
<td>2.965667</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>42.22523</td>
<td>42.28708</td>
<td>34.06418</td>
<td>33.83414</td>
<td>14</td>
<td>46.24049</td>
<td>31.77909</td>
<td>3.565826</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>42.17351</td>
<td>40.09573</td>
<td>34.68365</td>
<td>34.68365</td>
<td>15</td>
<td>42.03184</td>
<td>36.73308</td>
<td>0.0</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>35.9473</td>
<td>33.32217</td>
<td>27.64667</td>
<td>27.64667</td>
<td>16</td>
<td>38.22136</td>
<td>37.84947</td>
<td>1.078024</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>43.21686</td>
<td>40.3479</td>
<td>34.80785</td>
<td>34.78763</td>
<td>17</td>
<td>47.75557</td>
<td>44.98192</td>
<td>2.073604</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>42.67504</td>
<td>42.27808</td>
<td>36.75189</td>
<td>36.68726</td>
<td>18</td>
<td>42.54073</td>
<td>37.63262</td>
<td>0.561416</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>39.79976</td>
<td>38.26526</td>
<td>32.95023</td>
<td>32.62848</td>
<td>19</td>
<td>46.63552</td>
<td>39.9189</td>
<td>3.67605</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>45.45098</td>
<td>43.81131</td>
<td>37.40796</td>
<td>37.40796</td>
<td>20</td>
<td>37.62191</td>
<td>37.79449</td>
<td>0.0</td>
<td></td>
</tr>
</tbody>
</table>

AVERAGE 42.13943 40.7748 35.01028 34.93561

### Table 2.12 Heuristic Performance Compared to Models

Conversely, the greedy and bang-for-buck heuristics yielded the same results as long as the inputs did not change. Table 2.11 shows the average performance of the heuristic selection rules for each of the models. The $V$ values from the distance equations are shown on the left side of the table for each run and the percent difference values are shown on the right side. As expected, the models always outperform the heuristics in terms of the objective value ($V$). I can also conclude that the mathematical model outperforms bang-for-buck selection, which outperforms greedy selection, which outperforms random selection.
Based on the percent difference values, there is always a large gain to be realized by using the mathematical models compared to the random and greedy heuristics. However, the bang–for-buck heuristic performs only 2 - 5% worse than the mathematical models and provides an alternate solution method when the number of variables is too large for the mathematical models to provide a solution in a reasonable amount of time. I generated datasets for a different number of variables, and in Figure 2.9 I show the solution time versus the number of variables including a trend line, which follows a polynomial path. At around 500,000 variables, the mathematical model solution time increases dramatically, while the bang-for-buck heuristic solution time remains under a minute. Furthermore, the bang-for-buck heuristic yields an objective value that is within 8% of the optimal solution from the mathematical model, providing an alternate solution method when a quick solution is required for large scale problems.
Figure 2.10 through Figure 2.15 show that for all models, as iterations continue, knowledge continues increase, eventually converging on the ideal solution. The graphs on the left side show the decrease in $V$ for each of the three models as the iterations continue, compared to the three different heuristics. The graphs on the right hand side show the decrease in the average $V$ values over the 20 runs for each model compared to the three heuristics. In both cases, the $V$ value always decreases, which means that the solution is always getting closer to the ideal. Also, the mathematical model always has a $V$ value that is better than the random and greedy heuristics, and better than or equal to the bang-for-buck heuristic. This proves that by including a human fuser in the process, the DM’s knowledge about estimates improves. In other words, a system with a soft process is always better than one incorporating only a hard process in terms of assisting a DM.

Figure 2.10 First run for Model 1.

Figure 2.9 Number of Variables vs. Solution Time
Figure 2.11  Model 1 average results.

Figure 2.12  First run for Model 2.
Figure 2.13  Model 2 average results.

Figure 2.14  First run for Model 3.
Examining gain over time can also help identify the point at which modifying the certainty values results in minimum gain.

Figure 2.16 shows the percent gain or percent from optimality (all certainty values have a value of 1) over time. The blue line shows how much gain the human fuser adds over time. These results show that at about 45 seconds into the process, gain is less than 0.5% of the total knowledge for every additional 2 seconds of time. By the same token, the red line represents human fuser’s level of optimality, which reaches 90% at the same point (45 seconds). This means that there is a break point in this system at which allowing the human fuser to continue to change observations results in minimum gain.
In this chapter we have presented a novel methodology to add a soft process to a system that is typically only hard processes even though the inputs into the system are both hard and soft sensors. Along with presenting this new data flow for the system we have demonstrated through an example how the soft process would work with the hard processes that are already in place. Through the metrics defined we have shown that the addition of a soft process (Human Fuser) can actually improve the Decision Makers knowledge of the problem, which should improve the accuracy of the Course-Of-Action that he or she determines. We show that over time our system converges towards optimality (Figure 2.11, Figure 2.13 and Figure 2.15). We have also presented three different linear mathematical models that are used in conjunction with the Human Fuser. We compare each of them to three heuristics (Random, Greedy and Bang for Buck) and demonstrated that by using the math models we can outperform the heuristics (Conversely, the greedy and bang-for-buck heuristics yielded the same results as long as the inputs did not change. Table 2.11 shows the average performance of the
heuristic selection rules for each of the models. The $V$ values from the distance equations are shown on the left side of the table for each run and the percent difference values are shown on the right side. As expected, the models always outperform the heuristics in terms of the objective value ($V$). I can also conclude that the mathematical model outperforms bang-for-buck selection, which outperforms greedy selection, which outperforms random selection.

Table 2.12). We also found that while the mathematical models outperform the best heuristic (Bang-for-Buck) the amount of time increases dramatically after 500,000 variables and switching to Bang-for-Buck allows us to obtain a solution much faster while only being 7-8% away from the optimal solution. We also have demonstrated that as the system continues to run more iterations, which represents time, we always saw a gain in knowledge until we reached ground truth or when all certainty values for all F-E’s had a value of 1 (Figure 2.10 - Figure 2.15). This allowed us to see that there is a point in time that allowing the human fuser to continue modifying the certainty values adds very little to the overall knowledge of the DM.

We need to further explore many areas to be able to completely understand this sot process addition. First we need to explore a few more elaborate models. In particular on that includes a more well-known uncertainty calculation rather than just averaging the values together as we did here. Entropy theory, which has been mentioned in the literature, seems like a prime candidate. Lastly, we would also like to explore having more than one human fuser in the system and include a time degradation function as information or data typically lose credibility as time continues.
Chapter 3: Complex Mathematical Models within the Cognitive Injection Process

3.1 Introduction

Technological advancements have contributed to high levels of data availability for DMs. This makes it increasingly important to push the most reliable and useful data to the forefront, because DM’s do not have the time to review all of the data available to them for a particular problem. In Chapter 2, I demonstrated how injecting a human into the data processing component and allowing the human to interact with the data improves a DM’s knowledge base. This is accomplished using CIP, which allows a human fuser to interact with data output from the IF process. Just as a DM does not have the time to evaluate all data points, a human fuser has limited time to manipulate them. Thus, the driving force behind CIP is the mathematical models or heuristics that select a small subset of data points for the human fuser to manipulate. The mathematical models previously used in CIP were in the form of knapsack problems because they tend to be easier to solve than other mixed-integer problem formulations. For both mathematical models and known knapsack heuristics, DM knowledge always increases, and that gain decreases over time.

In this chapter, I build on CIP and develop three new models that are more intricate and complex. In particular, I move away from the knapsack problem and add constraints that allow for more customization, which may be more appropriate for certain problem types. This includes changing the way certainty is calculated by adding another dimension to the data and using entropy to calculate the uncertainty. Previously, I simply averaged together the certainty values for each piece of information received from the IF process. While averaging the values is one way to determine the overall certainty, entropy is a more comprehensive way to combine the uncertainty values. Entropy measures the uncertainty associated with a random variable.
(Shannon, 1949) and has been used in many algorithms and information theory applications (Coifman, 1992; Cornwell, 1984; Barbara, Li & Couto, 2002). Entropy allows another dimension of data to be included, providing the human fuser with a more specific piece of data to modify and the DM with more specific pieces of information. Using more complex mathematical models requires heuristics that are used when problem sizes become too large to solve in a reasonable amount of time.

3.1.1 Cognitive Injection Process

The CIP is a selection algorithm that helps the human fuser (subject matter expert) modify the certainty values of specific data points. Figure 3.1 illustrates the inclusion of CIP and a human fuser in the IF process. Data sources include two different types of sensors: hard sensors, which are controlled by programs and rules; and soft sensors, which are humans who relay what they see, smell, feel, hear or taste. According to Pravia (2008) the sensor sources can be classified into one of the six displayed in Figure 3.1. These classifications are explained in section 3.2. Normally, the sensors retrieve data which are converted into pieces of information during the IF process. These pieces of information are estimates comprised of features, or general similarities among the different estimates. For example, sensors at a traffic intersection may relay data on car color; one estimate may specify a car’s color to be blue and another may specify green, but both refer to the feature “color.” Normally, these estimates and features are then used to determine situational awareness (SA). Endsley’s (1988) definition of SA is the most widely accepted; SA is “the perception of elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future” (p. 97). Combining SA with impact assessment (IA) gives the DM a situational understanding, which is comprehension not only of the problem space, but also of what will
happen when COAs are applied to the problem space. Refining the data by including a human fuser and CIP will allow for an improvement in SA, IA, and situational understanding, providing the DM with a more accurate idea of the problem space and likely COA effects.

![Figure 3.1 Inclusion of CIP (Steinberg & Bowman, 1999).](image)

### 3.1.2 Model Complexity

Previously, I applied CIP to knapsack problems only. Knapsack problems are considered to be some of the “easier” NP-hard problems to solve from a practical perspective (Pisinger, 2005) and I wanted to see how CIP would perform on these types of problems before using more complex models. Figure 3.2 represents the different classifications of computational problems based on their resolution difficulty. In particular, I examine the decision problem space, which represents problems with yes or no answers. An example of a decision problem could be deciding whether or not to purchase an apple. The knapsack problems I presented are function problems with complex single outputs (i.e., more than yes or no). Knapsack problems are interesting in that it is possible to convert them into decision problems, thus making them NP-
complete problems. An NP-complete problem will grow in polynomial or pseudo-polynomial time (Johnson, 1979) as the problem size increases, which I observed when applying CIP to knapsack problems. Known heuristics (i.e., random, greedy, bang-for-buck) that have been used successfully to solve knapsack problems solved the problems more quickly than the optimization solvers but yielded worse results. The models I present in this paper (TOL, Max-Min, entropy) are classified as NP-hard. Unlike the knapsack problems, the new models contain more than one constraint, and their objective functions are computationally more complex. Because the new models are not known problems I developed heuristics (P problems) to solve the same datasets faster and yield results relatively close to the objective values obtained using the mathematical models.

Figure 3.2 Computational complexity.

In the following sections, I further discuss the addition of the human fuser and the difficulties associated with solving more complex models. In section 3.2, I explore the literature
on information fusion, model complexity and entropy. In section 3.3, I describe a problem area that could benefit from CIP and present the developed models and heuristics. In section 3.4, I discuss model experimentation and analyze them by comparing solution time and solution optimality. Finally, I summarize the findings and discuss future work that can be performed to further enhance the notion of adding a human fuser to the data processing section of a system.

3.2 Background Literature

Data fusion is the combining of data to make inferences about a situation while improving its accuracy (Hall & Llinas, 1997). Data fusion was developed in the 1980s primarily for military projects, but terminological differences among researchers created confusion, which was eventually sorted out by the Data Fusion Subpanel of the Joint Directors of Laboratories (JDL) (White, 1991). The JDL reconvened later to further solidify specifics of data fusion, particularly its five levels (Steinberg, Bowman, & White, 1999). Data fusion remains an important tool in military settings for tasks such as recognizing targets (Hall, 1991), obtaining situational awareness and assessing impact. Over the years, a number of other fusion frameworks have been proposed (Endsley, 1995; Salerno, 2002), but the fundamental question remains in terms of where and how a human fuser should be injected into the overall IF system. As connectivity increased and technology evolved, scholars began exploring hard and soft data fusion (Hall, Llinas, McNeese, & Mullen, 2008), creating new research areas such as how to associate uncertainty values with data when the data types are very different. Data sources have been classified into two sensing types: hard and soft. Pravia (2008) further classified the sensors into six different types. Hard sensor types are presented in Table 3.1 and soft sensor types are presented in Table 3.2.
Table 3.1 Hard Sensor Classifications

<table>
<thead>
<tr>
<th>Type</th>
<th>Acronym</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Geospatial</td>
<td>GEOINT</td>
<td>Radars, Infrared and Visible Light Imaging</td>
</tr>
<tr>
<td>Signal</td>
<td>SIGINT</td>
<td>Electronic Interception of a Signal</td>
</tr>
<tr>
<td>Measurement and Signature</td>
<td>MASINT</td>
<td>All Other Hard Sensors</td>
</tr>
</tbody>
</table>

Table 3.2 Soft Sensor Classifications

<table>
<thead>
<tr>
<th>Type</th>
<th>Acronym</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open Source</td>
<td>OSINT</td>
<td>Facebook, Twitter, Newspapers</td>
</tr>
<tr>
<td>Communication</td>
<td>COMINT</td>
<td>Human Interception of Information</td>
</tr>
<tr>
<td>Human</td>
<td>HUMINT</td>
<td>Interrogations or Conversations</td>
</tr>
</tbody>
</table>

Rudolf Clausius first introduced the thermodynamic concept of entropy in 1865, using it to describe the lost energy in a system due to dissipation or friction (Mendoza, 1988). Claude Shannon (1948) then used this idea of entropy to develop a concept called information entropy. He discussed uncertainty and entropy associated with signal processing and noise. Entropy theory is based on a set of possible events and their associated occurrence probabilities. Entropy, $H(X)$, of a random variable $X$ with a probability mass function $p(x)$ is defined by:

$$H(X) = -\sum_{x} p(x) \log p(x)$$  

The probability mass function of $X$ is the probability that a random variable is equal to a specific value within the set of all observations in $X$. Since the formula only uses the probability mass function values, entropy can be extremely important in calculating sensor uncertainty if the mass functions can be obtained, because the uncertainty calculation does not depend on the actual value of the random variable. This could make determining the uncertainty of qualitative data easier, since there is no need for a set of possible observation outcomes. For example, let $X$
represent the value of a die that is being rolled. Since it is a fair die, the probability mass
function $p(X)$ is $1/6$ for any of the possible outcomes (i.e., the die landing on 1, 2, 3, 4, 5 or 6).
The entropy of $X$ with a log base of 2 is:

$$H(X) = -\left\{ p(X = 1) \log(p(X = 1)) \right\} - \cdots - \left\{ p(X = 6) \log(p(X = 6)) \right\} = 2.585 \quad 3.B$$

Using a base of 2 in the entropy equation allows for the entropy value to represent the number of
binary questions it would take to determine the value of the random variable $X$.

Entropy’s usefulness can be seen further when examining the extensions of the technique
that consider more than one event at a time. Joint entropy (Arthur, 2003) is the total entropy of
two variables (or sensors) describing the same event. Conditional entropy (Arndt, 2001) is the
uncertainty of one random variable or a piece of sensor data, given that a different sensor’s
observation of that same event is known. A third measurement is relative entropy (Cover &
Thomas, 1991), which provides the distance between two distributions. Mutual information
(Guiasu, 1977) is the last measurement that can be found using entropy, and it describes the
amount of information that one random variable or sensor contains about another. Shannon
(1949) discussed how entropy can be used to determine the uncertainty of a message that has
been encoded and then decoded. In addition to calculating uncertainty values for information,
entropy has also been used in algorithms to help with decision making (Coifman, 1992;
Cornwell, 1984; Barbara, Li & Couto, 2002).

3.3 Methodology

3.3.1 Problem Description

Assume the proposed algorithm is to be implemented in a hospital, in which both hard
(e.g., thermometer, blood pressure monitor, heart rate monitor) and soft (i.e., healthcare
providers) sensors are used for data collection. All five sensors retrieve data on patient Z and that data is used in the IF process to obtain estimates, features and certainty values (Figure 3.3).

![Figure 3.3 Hospital data example.](image)

An estimate is a piece of information that is extracted from the raw data from the sensors. For example, estimate 6 could state that the patient has a fever and above normal blood pressure, information that was extracted from the numerical data of the thermometer and blood pressure monitor (BPM). Features are commonalities among the estimates such as blood pressure and temperature that can be represented in different forms. For example, while estimate 6 could result in an inference that the patient has a fever, estimate 5 could claim that the patients head felt warm when touched. The IF process also assigns a certainty value for each feature in every estimate. The certainty values represent how reliable the data in a feature-estimate (F-E) is. Imagine that in estimate 6 the temperature is described as being 102.1 with a certainty value of 0.62 or 62%. This means that the probability that the temperature of patient Z is actually 102.1 is 62%. These values are the data points that will be modified by the human fuser to improve the DM’s knowledge of patient Z. In this scenario, the human fuser could be an additional doctor who is an expert at diagnosing patients.

### 3.3.2 Mathematical Models and Heuristics
In this section, I present three mathematical models that can be used in the CIP. Each model allows for the algorithm to accomplish different objectives specified by a DM. I also present three heuristics, one for each of the mathematical models, that can be used once the problem size becomes too large to be solved by CPLEX within a reasonable amount of time.

### 3.3.2.1 Top of List Model

The first model is what I call the Top of List (TOL) model. TOL provides the human fuser with estimate data points to modify that have the potential to make the estimates optimally certain and reliable. This means that if a human fuser can modify the certainty values of the data points, the estimates will be placed at the top of the ranked estimate list when the DM is determining the COA. TOL accomplishes two things: first, it selects data points from estimates with the largest potential, which provides the DM the most certain and reliable set of estimates; and second, it allows estimate data points to be examined that would otherwise never reach the top of the ranked estimate list and would likely never be evaluated by the DM.

The mathematical formulation of the TOL model can be seen below:

Maximize \( Z = \sum_i C_i \) \hspace{1cm} 3.C

Subject to: \( \sum_{u_{ij} > 0} \left[ W_j \left( A_i \cdot U_{ij} + Y_{ij} \cdot (1 - U_{ij}) \right) \right] + \sum_{u_{ij} = 0} \left[ W_j \cdot X_{ij} \right] \leq P \) \hspace{1cm} \( \forall \ i \) 3.D

\[ = C_i \]

\[ \sum_{u_{ij} > 0} \sum_j \left[ P_{ij}^Y \cdot Y_{ij} \right] + \sum_{u_{ij} = 0} \sum_j \left[ P_{ij}^X \cdot X_{ij} \right] \leq P \]

\[ \sum_i l_i \leq TL, \] 3.G

where the variables are \( X, Y, I, A \) and the parameters are \( P^X, P^Y, P, W, U, C \) and TL.
The purpose of the TOL model is to find the most certain and reliable estimates, which means the objective function must be the sum of only the estimates that can potentially end up in the top TL spots of the ranked estimate list. This is done by including the $I_i$ and $A_i$ variables in the mathematical formulation. By adding the $A_i$ variable to the weighted certainty calculation (3.D), the overall certainty of an estimate can be set to 0 even if the IF process has assigned a certainty value to an F-E ($U_{ij} > 0$). The $I_i$ variable is used in 3.F to prevent $C_i$ and $A_i$ from being greater than zero for an estimate, and 3.G limits the number of $I$’s that can be selected. The addition of 3.F and 3.G allows for the objective to sum only $TL$ number of estimate certainty values. Equation 3.E is a processing time constraint that is used to limit the number of data values.
points that are selected for the human fuser to observe based on processing time of the F-Es \( P_X \) and \( P_Y \) and the available time of the human fuser \( P \).

The TOL model is an Integer Program (IP), because all of the variables are restricted to integer values. This means solving TOL is NP-hard and may require an excessive amount of time to solve when the number of variables is large, which I will show in the experimentation section. The proposed system is meant to run in real time, which requires the data points selected by the math model to be discovered quickly. To account for this increase in resolution time for the TOL model when using CPLEX, I developed the TOL-Heuristic (Figure 3.4) to function as an adequate replacement in the overall system.

![Figure 3.4 TOL-H](image)

The key to the TOL-Heuristic (TOL-H) is to solve the relaxation of the original TOL model. I accomplish this by allowing all of the variables \( X, Y, I \) and \( A \) to be floating values between 0 and 1. The relaxed formulation is then just a Linear Program (LP), which can be solved by CPLEX much more quickly. The first step in the heuristic is to solve the LP relaxation.
relaxation. This yields the values for the four different variable types, which may or may not be integer. The heuristic then checks if all estimate $I_i$’s are integer values. If at least one $I_i$ is not an integer, then the TOL-H must find the estimate $k$ with the largest $\frac{C_k}{(1-I_k)}$ ratio. This value represents the estimate with the largest certainty and smallest distance for $I_i$ to be a value of 1. By adding a constraint forcing the $I_k$ value to be 1, the next LP solution must include that particular estimate $k$ at the top of the list. This cycle repeats until all $I_i$’s are either 0 or 1, and then the TOL-H evaluates the specific values of the $X$ and $Y$ variables representing the data points the human fuser should inspect. As before, the TOL-H checks to see if all $X_{ij}$ and $Y_{ij}$ variables are integers (0 or 1). If at least one is not an integer value, the heuristic finds the variable that is not an integer and has the smallest impact on the objective value. This is done by using the value of $W_{j^*} \left(1 - \frac{U_{i^*j^*}}{\frac{X_{i^*j^*} \lor Y}{P_{i^*j^*}}}\right)$. The fractional ratio represents the possible gain by having the human fuser observe feature $j^*$ in estimate $i^*$ over the processing time for that F-E. This yields the variable that, if chosen, will possibly yield the smallest gain with the largest processing time. The weight value ($W_{j^*}$) is included to take into account the importance of feature $j^*$. As with the $I_i$’s the TOL-H then adds a constraint to the LP by setting the chosen $X_{i^*j^*}$ or $Y_{i^*j^*}$ to 0 and then resolves the LP relaxation. This continues until all the variables are 0 or 1 and an integer solution is found for the LP relaxation. It is important to note that the TOL-H never checks the values of the $A$ variable because when all the $I_i$’s are set to 0 or 1 and the objective function $A_i$’s are maximized, the values will always be integers once the TOL-H is complete.

There are two theoretical results of the TOL-H, which in conjunction with the empirical experiments discussed in section 4.1, demonstrate the value of the method. While the experimental section shows great qualitative performance in a computationally efficient manner,
two rigor questions arise. First, does the TOL-H guarantee a feasible solution to the original integer program? And second, does the running time have a polynomial bound? The following two theorems show that the power of the heuristic is that it both guarantees feasibility and polynomial running time.

**Theorem 3.1:** If the TOL model has a feasible solution, then the TOL-H will terminate with a feasible solution to the original model.

*Proof:* The heuristic is composed of two sequential processes in which the first subprocess (hereafter, the I-Process) sets a subset of $I_k$ variables to 1, and the second subprocess (hereafter, the XY-Process) sets a subset of the $X_{ij}$’s and $Y_{ij}$’s to 0. It suffices to show that the I-Process first guarantees that the $I_k$ variables will all be set to 0 or 1, and the XY-Process will obtain feasible solutions for the remaining variables, such that all constraints are satisfied.

*I-Process:* It should be clear that the solution to the first LP relaxation has a bounded feasible solution. Feasibility stems from the fact that the original program has a feasible solution (as stated in the premise of Theorem 3.1) and boundlessness stems from the fact that the objective function value cannot exceed the number of estimates $N$ since $N C_j$’s are summed, each with a maximum value of 1. The number of iterations required to set $I_k$’s to 1 is equal to the value of the $TL$ parameter. At this point the desired $TL$ number of variables is set to 1, and the remaining unassigned $I_k$’s are set to 0 when running the LP relaxation, at which time the conditions of the I-Process stops and the sequential process is sent to the XY-Process. It is important to note that at this point, setting all other variables (i.e., $X_{ij}$, $Y_{ij}$, $A_j$ and $C_j$) to 0 guarantees a feasible solution not only to the LP relaxation, but also to the original problem with an optimal value of 0.
**XY-Process:** In the worst case scenario, this sub-process will run $2*N*F$ times whenever either the $X_{ij}$ or $Y_{ij}$ variable is set to 0. The only way the heuristic will stop is if at some point the LP relaxation yields integer values for all $X_{ij}$’s and $Y_{ij}$’s. If all estimates and features have been considered and no integer solution has been achieved by solving the LP relaxation, all the $X_{ij}$’s and $Y_{ij}$’s are set to 0, which in turn allows all $A$’s and $C$’s also to be set to 0. As discussed before, an all 0 solution guarantees a feasible solution to the original problem.

Although the proof that the TOL-H will yield a feasible solution is mathematically correct, it could give the impression that a value of 0 (which could be arbitrarily bad) is the outcome of the method. This is absolutely not the case, as shown in the empirical results in section 4.1. Moreover, it can be seen that the method allows a maximum number of estimates to be verified by the human fuser, since the number of $I_k$’s defines the upper bound for the sum of $C_k$’s, which is at the maximum value allowed. Furthermore, the greedy ratios chosen both for the $I_k$’s in the I-Process and the $X_{ij}$’s / $Y_{ij}$’s in the XY-Process promote those estimates and features that use the least time but provide the most potential gain in minimizing uncertainty.

**Theorem 3.2:** The TOL-H runs in $O((N+2*N*F)LP)$, where $N$ is the number of estimates considered by the $TL$ parameter, $F$ is the number of features, and $LP$ is the running time to solve a linear program.

**Proof:** Use the same definitions as before for the I-Process and the XY-Process.

**I-Process:** With a feasible solution, this means that all $I_k$’s will be assigned a value of 0 or 1 by the heuristic. The important part is setting the $I_k$’s to a value of 1, as this will force the other $I_k$’s to a value of 0. With a maximum number of $N$ estimates (determined by the $TL$ parameter) selected as the Top of List, there will be at most $N I_k$’s assigned a value of 1. This
means that the I-Process can iterate \( N \) times at most; with a single LP solved during every iteration, \( N \) LP’s will be run at most.

**XY-Process:** For a set of \( l_k \)’s determined by the I-Process, there is a maximum number of \( N \) estimates that can be considered by the XY-Process. With each estimate containing \( F \) number of features and every iteration in the XY-Process setting one \( X_{ij} \) or \( Y_{ij} \), the maximum number of iterations needed to obtain a feasible solution would be \( 2 \times N \times F \). This is based on the fact that there are two variables (\( X_{ij} \) and \( Y_{ij} \)) in every F-E combination. Like the I-Process, the XY-Process solves 1 LP during each iteration, meaning that the maximum number of LP’s solved would be \( 2 \times N \times F \).

### 3.3.2.2 Max-Min Model

The second proposed model is the Max-Min model. The purpose of this model is to maximize the number of estimates with a minimum certainty value over a threshold assigned by the DM. This will allow the system to potentially provide the DM with the largest number of estimates that he or she has deemed certain and reliable enough for use in determining the COA. Using the Max-Min model encourages the human fuser to only observe data points that will bring an estimate’s overall weighted certainty over the threshold. This means that once an estimate \( i \)’s certainty reaches this threshold, there is no reason for the human fuser to spend his or her time on any other features for estimate \( i \). Like the TOL model, the Max-Min model can also filter out estimates; but, unlike the TOL model, the Max-Min model filters out any estimates that cannot reach the threshold value rather than those that cannot reach the top of the ranked estimate list.

The mathematical formulation for the Max-Min model is shown below:
Maximize \( Z = \sum_i T_i \) \( \quad \) 3.H

Subject to: \( \sum_{u_{ij} > 0} \left[ W_j \ast \left( U_{ij} + Y_{ij} \ast (1 - U_{ij}) \right) \right] + \sum_{u_{ij} = 0} \left[ W_j \ast X_{ij} \right] = C_i \quad \forall \ i \) 3.I

\[ \sum_{u_{ij} > 0} \sum_j \left[ P_{ij}^Y \ast Y_{ij} \right] + \sum_{u_{ij} = 0} \sum_j \left[ P_{ij}^X \ast X_{ij} \right] \leq P \] 3.J

\[ T_i \leq 1 + (C_i - TH) \quad \forall \ i \] 3.K

\[ M \ast T_i \geq \sum_j (X_{ij} + Y_{ij}) \quad \forall \ i \] 3.L

where the variables are \( X, Y \) and \( T \) and the parameters are \( P^X, P^Y, P, W, U, C \) and \( TH \). All the variables and parameters from the TOL model are the same in the Max-Min model, with the addition of \( T \) and \( TH \):

\[ T_i = \begin{cases} 1, & \text{If estimate } i \text{ can potentially have a weighted certainty above a threshold (TH)} \\ 0, & \text{Otherwise} \end{cases} \]

\( TH = \) Minimum certainty value set by the DM and considered to be a reliable estimate

For the Max-Min model, the objective function simply sums the number of estimates meeting the threshold requirement. To determine if an estimate has met this threshold, the certainty calculation (3.I) and equation 3.K are used in the formulation. If the certainty value (\( C_i \)) is above the threshold, then the right hand side (RHS) of 3.K will yield a value greater than 1, meaning that \( T_i \) can be set to 1. Otherwise the RHS of 3.K will yield a value smaller than 1, forcing the \( T_i \) variable to be 0. The model also limits an \( X \) or \( Y \) variable from being selected if the estimate cannot potentially reach the threshold value through 3.L. The processing time constraint from the TOL model remains unchanged in the Max-Min model (3.J).
The Max-Min model is an IP, meaning it also is NP-hard and developing a heuristic is beneficial for larger problems. I took the same approach by using the LP relaxation of the mathematical model to obtain variable values between 0 and 1 and then added constraints to the formulation to force the free variables to become integers. The Max-Min heuristic (MM-H) is shown in Figure 3.5.

![Figure 3.5 MM-H](image)

The first step is to solve the original LP relaxation. The heuristic then checks whether or not the resulting $T_i$ values are integers (0 or 1). If at least one $T_i$ is not an integer value, the MM-H searches for the estimate $k$ with smallest $T_k$ value that is also greater than 0. The smallest $T_k$ represents the estimate $k$ with the lowest overall weighted certainty value. Thus, of the estimates with certainty values, estimate $k$ is least likely to reach the threshold value. The next step is to add a constraint to the LP relaxation by forcing $T_k$ to be 0. Then the heuristic solves the new LP relaxation and again checks all the $T_i$ values until they are all 0 or 1. Once the $T_i$'s have been set to integer values, the heuristic checks the $X_{ij}$ and $Y_{ij}$ variable values to determine if all of them
are integer values as well. If one $X_{ij}$ or $Y_{ij}$ is not an integer, the heuristic then proceeds to find the one $X_{ij^*}$ or $Y_{ij^*}$ with the smallest impact. The smallest impact value is found by looking at the calculation of $W_{ij^*} \left( \frac{1-U_{ij^*}}{p_{X_{ij^*}} \lor Y_{ij^*}} \right)$. The heuristic then adds another constraint to the LP relaxation that forces the $X_{ij^*}$ or $Y_{ij^*}$ variable to stay at a value of 0 and resolves the LP relaxation.

Unlike the TOL-H, for the MM-H there is a need to go back and check the $T_i$ values because setting one of the $X_{ij^*}$ or $Y_{ij^*}$ to 0 may reduce the certainty of an estimate below the threshold limit. Once all the $T_i$’s, $X_{ij}$’s and $Y_{ij}$’s are integer values of 0 or 1, the heuristic is complete and the selected data points are given to the human fuser to modify.

3.3.2.3 Entropy Model

The final proposed model is the entropy model. I have decided to include this model because it allows for a more comprehensive and appropriate way to combine certainty values over the estimates. Unlike the TOL and Max-Min models, the entropy model does not allow for the system to filter estimates based on potential certainty values, nor does it use a parameter that is determined by one of the humans in the system ($TL$ and $TH$). The entropy model modifies how the certainty of an estimate is calculated and allows for a new dimension of data to be included, because now there is a certainty value for every possible state within each feature for every estimate. In Figure 3.3, there are two dimensions to the data: estimates and features. Entropy uses state probability to determine the certainty of an estimate, so possible states for each feature (i.e., options) must be included. This changes the output of the IF process to include the possible options for each feature, which can be seen in Figure 3.6.
The entropy formula $H(X)$ is nonlinear if used in a mathematical model and requires that all the probabilities of all possible states of $X$ sum to a value of 1 to work properly. This makes the formulation of entropy not only a nonlinear program (NLP), but forces the use of both free variables and integer variables, making it a mixed integer problem (MIP). NLPs and MIPs are NP-hard to begin with, so combining them makes solving this type of formulation incredibly difficult and time consuming. The NLMIP for entropy is:

Minimize $Z = -\sum_i \sum_j \sum_k (X_{ijk} + Y_{ijk} + U_{ijk})$

$+ \left(1 - \left(X^c_{ijk} + Y^c_{ijk}\right)\right) \frac{B_{ij}}{K-1}$ \hspace{1cm} 3.M

$* \log \left(X_{ijk} + Y_{ijk} + U_{ijk} + \left(1 - \left(X^c_{ijk} + Y^c_{ijk}\right)\right)\frac{B_{ij}}{K-1}\right)$

Subject to: $|X_{ijk}| \leq X^c_{ijk}$ \hspace{1cm} \forall i, j, k \hspace{1cm} 3.N

Figure 3.6 Hospital entropy data.
\[ |Y_{ijk}| \leq Y_{ijk}^C \quad \forall i, j, k \quad 3.0 \]
\[ X_{ijk}^C \leq 1 - \left| U_{ijk} - \frac{1}{O} \right| \quad \forall i, j, k \quad 3.3 \]
\[ Y_{ijk}^C \leq M \cdot \left| U_{ijk} - \frac{1}{O} \right| \quad \forall i, j, k \quad 3.7 \]
\[ X_{ijk}^C + Y_{ijk}^C \leq 1 \quad \forall i, j, k \quad 3.8 \]
\[ \sum_{k} [X_{ijk} + Y_{ijk} + U_{ijk}] + B_{ij} = 1 \quad \forall i, j \quad 3.9 \]
\[ X_{ijk} + Y_{ijk} + U_{ijk} + \left( 1 - (X_{ijk}^C + Y_{ijk}^C) \right) \frac{B_{ij}}{K-1} \geq 0 \quad \forall i, j \quad 3.10 \]
\[ \sum_{k} X_{ijk}^C + Y_{ijk}^C \geq \left| B_{ij} \right| \quad \forall i, j \quad 3.11 \]
\[ \sum_{i} \sum_{j} \sum_{k} (X_{ijk}^C \cdot P_{ijk}^X + Y_{ijk}^C \cdot P_{ijk}^Y) \leq P \quad \forall i, j \quad 3.12 \]

where variables \( X_{ijk}, Y_{ijk}, X_{ijk}^C, Y_{ijk}^C, B_{ij} \) and parameters \( U_{ijk}, P_{ijk}^X, P_{ijk}^Y, P \) represent:

\( X_{ijk} = \) Amount the HF should modify the uncertainty for option \( k \) in feature \( j \) for estimate \( i \) where \(-1 \leq X_{ijk} \leq 1\)

\( Y_{ijk} = \) Amount the HF should modify the uncertainty for option \( k \) in feature \( j \) for estimate \( i \) where \(-1 \leq Y_{ijk} \leq 1\)

\( X_{ijk}^C = \begin{cases} 1, & \text{Human fuser should assign an uncertainty for option } k \text{ in feature } j \text{ for estimate } i \\ 0, & \text{Otherwise} \end{cases} \)

\( Y_{ijk}^C = \begin{cases} 1, & \text{Human fuser should modify the uncertainty of option } k \text{ in feature } j \text{ for estimate } i \\ 0, & \text{Otherwise} \end{cases} \)

\( B_{ij} = \) Buffer for feature \( j \) in estimate \( i \), where \(-1 \leq B_{ij} \leq 1\)

\( U_{ijk} = \) The uncertainty given to option \( k \) in feature \( j \) for estimate \( i \) by the IF process

\( P_{ijk}^X = \) Processing time to perform task \( X \) on option \( k \) in feature \( j \) for estimate \( i \)

\( P_{ijk}^Y = \) Processing time to perform task \( Y \) on option \( k \) in feature \( j \) for estimate \( i \)

\( P = \) Processing time available to the human fuser
The entropy model has a nonlinear objective function (3.M) and constraint (3.T). Equations 3.N and 3.O are used to set the $X_{ijk}^C$ and $Y_{ijk}^C$ binary variables if the best case involves using the free variables $X_{ijk}$ and $Y_{ijk}$. These constraints are required in order to fully account for the processing time by using the binary variables. As before, two different operations may be performed using the uncertainty value: assign a new value, or modify an uncertainty value from the IF process ($X_{ijk}$ and $Y_{ijk}$, respectively). The two tasks are differentiated by equations 3.P and 3.Q. If $U_{ijk}$ has not been assigned a value by the IF process, it is given the worst possible uncertainty value, which for entropy is $(1/O)$, where $O$ is the total number of options for a feature. So in a case with five options, a value of 0.2 is the largest uncertainty value for the entropy equation. This means any $U_{ijk} = \frac{1}{O}$ can result in task $X$ being performed and any $U_{ijk} \neq \frac{1}{O}$ can result in task $Y$ being performed. Equation 3.R allows for only one task, $X$ or $Y$, to be performed per option $k$ in feature $j$ for estimate $i$. Equation 3.S ensures that the property of entropy is followed and the buffer variable ensures that the sum of uncertainties for all options in a feature equals 1. It is also necessary to ensure that the uncertainty value for all option-feature_estimate combinations is greater than 0 (3.T), as an uncertainty value can never be negative. Equation 3.U is included to ensure that variables $X$ and $Y$ invoke the change of the uncertainty values and that the buffer variable can only be used if an $X$ or $Y$ task are chosen to be performed. The last constraint limits the number of data points chosen to be changed by the processing time available to the human fuser.

The entropy model is both an NLP and a MIP, making it extremely hard for any solver to obtain a feasible solution. Thus, I propose a mathematical model and a heuristic to solve larger problems in a timely fashion. The mathematical model linearizes the entropy formula by making two assumptions. The first is that only one option can be chosen in a feature, or all must be
chosen. This assumption makes it possible to generalize the entropy formula linearly and to hold the properties required for entropy to be true. If more than one option in a feature is chosen, then all possible terms in the objective function must be enumerated (e.g., 2 options chosen, 3, 4, etc.). This makes the objective function grow exponentially as the number of options in a feature increases. The second assumption is that when the human fuser modifies a certainty value, he or she can perform one of the following actions:

1. **Increase**: The chosen option of a feature in an estimate is increased by 1 and then normalized so that the sum over all the options satisfies the entropy constraint of the probabilities summing to 1. Since the original sum of all the options is 1 and the value 1 is added to an option to normalize it, always divide by a value of 2. Figure 3.7 shows how this occurs:

   ![Figure 3.7 Increasing options in an estimate.](image)

2. **Decrease**: The options of a feature in an estimate that are not chosen are increased by $1/(O-1)$, where $O$ is the total number of options for a feature. Then, normalize over the options to obtain a sum of 1. Just as before, a value of 1 is added, which means that again normalization is accomplished by dividing by a value of 2. Figure 3.8 shows how this occurs:
With these two assumptions, a linear mathematical model can be developed that is easier to solve and a feasible integer solution can be obtained. The entropy heuristic math model can be seen below:

Minimize \( Z \) = \[
\sum_{j} \left[ \sum_{l} \left( \sum_{k} \left( X_{ijk}^{+} + Y_{ijk}^{+} \right) \left( \frac{U_{ijk} + 1}{2} \right) \log \left( \frac{U_{ijk} + 1}{2} \right) \right) \right]
\]
\[+ \sum_{m \neq k} \left( X_{ijm}^{+} + Y_{ijm}^{+} \right) \left( \frac{U_{ijm}}{2} \right) \log \left( \frac{U_{ijm}}{2} \right) \]
\[+ \left( \sum_{k} \left( X_{ijk}^{-} + Y_{ijk}^{-} \right) \left( \frac{U_{ijk}}{2} \right) \log \left( \frac{U_{ijk}}{2} \right) \right) \]
\[+ \sum_{m \neq k} \left( X_{ijm}^{-} \right) \left( \frac{U_{ijm} + \frac{1}{(K - 1)}}{2} \right) \log \left( \frac{U_{ijm} + \frac{1}{(K - 1)}}{2} \right) \]
\[+ \left( M_{ij} - ALL_{ij} \right) \cdot CUR_{ij} \]

Subject to: \[
\sum_{k} \left( X_{ijk}^{+} + Y_{ijk}^{+} + X_{ijk}^{-} + Y_{ijk}^{-} \right) = 1 - M_{ij} \quad \forall i, j \quad 3.X
\]
\[
\sum_{k} \left( X_{ijk}^{+} + Y_{ijk}^{+} + X_{ijk}^{-} + Y_{ijk}^{-} \right) + ALL_{ij} \leq 1 \quad \forall i, j \quad 3.Y
\]
\[ X_{ijk}^+ + X_{ijk}^- \leq 1 - \left| U_{ijk} - \frac{1}{K} \right| \quad \forall i, j, k \quad 3.Z \]

\[ Y_{ijk}^+ + Y_{ijk}^- \leq M^B \cdot \left| U_{ijk} - \frac{1}{K} \right| \quad \forall i, j, k \quad 3.AA \]

\[
\sum \sum \sum \left( \left( \sum_{k}^{(X_{ijk}^+ + X_{ijk}^-)} \cdot P_{ijk}^X + (Y_{ijk}^+ + Y_{ijk}^-) \cdot P_{ijk}^Y + ALL_{ij} \cdot P_{ij}^{ALL} \right) \right) \leq P \quad 3.BB
\]

where variables \( X_{ijk}^+ \), \( X_{ijk}^- \), \( X_{ijk}^+ \), \( M_{ij} \), \( ALL_{ij} \) and parameters \( P_{ijk}^X \), \( P_{ijk}^Y \), \( P_{ij}^{ALL} \), \( P \), \( U_{ijk} \), \( M^B \), \( CUR_{ij} \) represent:

\[ X_{ijk}^+ = \begin{cases} 1, & \text{If option } k \text{ in feature } j \text{ in estimate } i \text{ should increase from a value of } \frac{1}{K} \\ 0, & \text{Otherwise} \end{cases} \]

\[ X_{ijk}^- = \begin{cases} 1, & \text{If option } k \text{ in feature } j \text{ in estimate } i \text{ should be decreased from a value of } \frac{1}{K} \\ 0, & \text{Otherwise} \end{cases} \]

\[ Y_{ijk}^+ = \begin{cases} 1, & \text{If option } k \text{ in feature } j \text{ in estimate } i \text{ should be increased from the IF assigned value} \\ 0, & \text{Otherwise} \end{cases} \]

\[ Y_{ijk}^- = \begin{cases} 1, & \text{If option } k \text{ in feature } j \text{ in estimate } i \text{ should be decreased from the IF assigned value} \\ 0, & \text{Otherwise} \end{cases} \]

\[ M_{ij} = \begin{cases} 1, & \text{If no option } k \text{ in feature } j \text{ for estimate } i \text{ was chosen} \\ 0, & \text{Otherwise} \end{cases} \]

\[ ALL_{ij} = \begin{cases} 1, & \text{If all } k \text{ options in feature } j \text{ for estimate } i \text{ were chosen} \\ 0, & \text{Otherwise} \end{cases} \]

\[ P_{ijk}^X = \text{Processing time to perform task } X \text{ on option } k \text{ in feature } j \text{ for estimate } i \]

\[ P_{ijk}^Y = \text{Processing time to perform task } Y \text{ on option } k \text{ in feature } j \text{ for estimate } i \]

\[ P_{ij}^{ALL} = \text{Processing time to perform a task on every option } k \text{ in feature } j \text{ for estimate } i \]

\[ P = \text{Processing time available to the human fuser} \]

\[ U_{ijk} = \text{The uncertainty given to option } k \text{ in feature } j \text{ for estimate } i \text{ by the IF process} \]
\[ M^B = \text{Large number} \]

\[ CUR_{ij} = \text{The current entropy value for feature } j \text{ in estimate } i \]

The objective function for the entropy heuristic is now linear. It includes what the entropy for an F-E would be if one of the four tasks was performed on one of the options using the \( U_{ijk} \) parameter. The objective function also considers whether all or none of the options in an F-E are evaluated. Thus, the current F-E uncertainty value must be included. Equation 3.X is used to set the \( M_{ij} \) variable to represent whether an option in the F-E is going to be modified. If a change is to be made, then the \( M_{ij} \) variable must be set to 0 to satisfy the change. Likewise, if no option is to be modified, the \( M_{ij} \) variable is set to 1 to satisfy 3.X. Equation 3.Y ensures the second assumption that either one option is chosen, all options are chosen or none are chosen to be modified. Equations 3.Z and 3.AA are used to regulate the \( X \) and \( Y \) tasks to be selected when the uncertainty rules for the tasks are followed. Finally, equation 3.BB is the processing constraint limiting the number of data points selected by considering the time the human fuser can allot to modifying uncertainty values.

The Entropy heuristic takes the same approach as the other models; first, the LP relaxation of the mathematical model is solved and then constraints are added until an integer solution is obtained (Figure 3.99). Once the LP relaxation is solved, the heuristic determines whether all values of the five different variable types are 0 or 1. If at least one is not 0 or 1, then the next step is to find the variable that provides the smallest gain per time unit. The gain values are determined by using equations 3.CC, 3.DD and 3.EE below:

\[
\frac{(1-ALL_{ij}) \cdot CUR_{ij}}{Q_{ij}^{LL}}
\]

(3.CC)
\[
\frac{\{X^+_{ijk} \text{ or } Y^+_{ijk}\} \cdot (INC_{ijk} - CUR_{ij})}{p_{ijk}(X \text{ or } Y)} \quad (3.DD)
\]

\[
\frac{\{X^-_{ijk} \text{ or } Y^-_{ijk}\} \cdot (DEC_{ijk} - CUR_{ij})}{p_{ijk}(X \text{ or } Y)} \quad (3.EE)
\]

where \( INC_{ijk} \) and \( DEC_{ijk} \) represent how much the entropy value increases or decreases, respectively, if option \( k \) in feature \( j \) for estimate \( i \) is chosen to be modified by the human fuser.

The heuristic then takes the variable associated with the smallest gain larger than 0 and adds it to the LP relaxation, setting the upper bound of the variable to 0. This forces that variable to be set to an integer and frees up processing time units for the other chosen variables. This new LP relaxation is solved and the process is repeated until all variables are set to either 0 or 1.

![Figure 3.9 Entropy heuristic.](image)

### 3.3.3 CIP Flow

Using the mathematical models described above, the proposed Cognitive Injection Process (CIP) can now be explained in detail. Figure 3.10 illustrates the flow of data and information in CIP.
The process begins when data are fused together from different sources and estimates are created. These estimates are assigned certainty or uncertainty values for each feature based on the reliability of the sources used to create the estimates. With these original certainty values, an initial ranked estimate list can be created for the DM based on the output of the IF process. As long as the DM has not determined a COA, the human fuser interacts with the data from the IF process. At time $T = 0$, the human fuser uses AHP on the list of all features extracted by the IF process to assign weighted importance values to each. Next, a mathematical model or heuristic determines which data point certainty value on the list of F-Es should be modified by the human fuser in order to most enhance the DM’s knowledge. These new certainty values are then processed to determine a new ranking for each estimate and the order of the DM’s ranked estimate list is adjusted. The human fuser may modify certainty values and the mathematical model may be used to reorder the ranked estimate list multiple times until the DM has chosen a COA.
3.3.3.1 Determine Feature Weights

Using AHP to determine the importance of features begins with a pairwise comparison of all the features. A value between 1 and 9 is assigned to each comparison, where a value of 9 means that the row feature is preferred nine times more than the column feature. The comparison matrix is then normalized using the column sums. The weights for each feature are calculated by averaging the normalized values across the rows. AHP was chosen as the weighting technique because there is a built in consistency check to ensure that the human fuser is not being inconsistent. This prevents a human fuser from assigning random or cyclical weights (e.g., $W_1 > W_2 > W_3 > W_1$). If the consistency ratio (CR) is greater than 0.1, then the human fuser must reassign the pairwise comparison values.

![AHP flow diagram](image)

Figure 3.11 AHP flow.

3.3.3.2 Modify Certainty Values

Once the feature weight values are determined, the mathematical model can be used in conjunction with the processing times ($P^x$ and $P^y$) and the certainty values from the IF process
(U) (see Figure 3.12). The mathematical model or heuristic can then determine the data points that should be observed by the human fuser (purple cells in Figure 3.12). The human fuser modifies the certainty values of the chosen data points based on his or her knowledge of the scenario, previous data and the sources that created the data points. In this case, the human fuser has increased the certainty values of all chosen data points to 1 (green cells in Figure 3.12), which means the human fuser believes that the values of those F-Es are 100% certain. The modified values are used to calculate new metrics to rank the estimates.

3.3.3.3 Modify Ranked List

Initially, the algorithm yields a ranked list for the DM with unmodified certainty values from the IF process. The estimates are ranked by calculating their weighted certainty values (WCV_i).
This allows the algorithm to place the most important and most certain estimates at the top of the list of estimates used by the DM to determine his or her COA. Since the weights for the features are not developed by the IF process, the original estimate list assigns equal weighted importance to all features. Figure 3.133 shows the \( WCV \) values for each estimate and the list of estimates which is ranked based on those values. In this algorithm, the DM only sees the list of ranked estimates and never interacts with any of the processes or data. If the DM does not need to determine a COA immediately, the certainty values can be modified. New metrics are calculated and a new estimate list is given to the DM. In this scenario, estimate 2 moved up the list because the human fuser was tasked by the mathematical model to attempt to modify the second and third features of estimate 2, while nothing in estimate 3 was chosen to be modified. The DM then determines a COA, or if time is not a constraint he can allow for the human fuser to continue to modify data points in the hopes that it will give him a more reliable and impactful set of estimates. While this example only contains five estimates, in real world problems the amount of data is extremely large, making it impossible for the DM to process all of them. Thus, placing the best estimates at the top of the list greatly assists the DM.
3.4 Experimentation

To analyze the algorithm, simulated datasets were used as input values and optimization Programming Language (OPL) was used for the mathematical programs and heuristics, except for the entropy NLP case where GAMS was used because OPL cannot handle non-linear problems. Both OPL and GAMS use the CPLEX solver. For experimentation purposes, the algorithm was run for each math model and heuristic on at least three dataset sizes with 100 different simulated datasets for each.

3.4.1 Top of List Analysis

The TOL model and heuristic require two parameters to be set: available processing time \((P)\) and the number of estimates that the DM will consider before determining a COA \((TL)\). For the sake of consistency in the analysis, the available processing time parameter is always 5\% of the total expected processing time to evaluate all F-E combinations. The processing time for

![Diagram](image.png)

Figure 3.13 What the DM sees.
each F-E is a discrete uniform distribution for new observations \( P^X \in (1, ..., 20) \) and for a modification \( P^Y \in (1, ..., 10) \). Assuming that half the F-Es were assigned values during the IF process and the other half were not, the expected value of the total processing time for all F-Es is:

\[
E(\sum_i \sum_j (P^X_i + P^Y_j)) = \frac{1}{2} [\mu_{px} \ast E \ast F] + \frac{1}{2} [\mu_{py} \ast E \ast F] = \frac{1}{2} [10 \ast E \ast F] + \frac{1}{2} [5 \ast E \ast F]
\]

\[
= \frac{15}{2} (E \ast F)
\]

where \( E \) is the total number of estimates and \( F \) is the total number of features. For consistency, the number of estimates the DM will consider is assumed to be a quarter of the total number of estimates \((\frac{1}{4} E)\).

Five different problem sizes were used for the TOL model and heuristic (Table 3.3). The number of variables is double the \( E \ast F \) value because each F-E combination has an \( X \) and a \( Y \) associated with it in the formulations. For each problem size, 100 different simulated datasets were solved.

<table>
<thead>
<tr>
<th>Problem Size (# of Variables)</th>
<th>Number of Estimates (E)</th>
<th>Number of Features (F)</th>
<th>Processing Time (P)</th>
<th>Top of List Parameter (TL)</th>
</tr>
</thead>
<tbody>
<tr>
<td>120</td>
<td>20</td>
<td>3</td>
<td>20</td>
<td>5</td>
</tr>
<tr>
<td>1,000</td>
<td>50</td>
<td>10</td>
<td>180</td>
<td>12</td>
</tr>
<tr>
<td>5,000</td>
<td>50</td>
<td>50</td>
<td>900</td>
<td>12</td>
</tr>
<tr>
<td>10,000</td>
<td>100</td>
<td>50</td>
<td>1800</td>
<td>25</td>
</tr>
<tr>
<td>50,000</td>
<td>250</td>
<td>100</td>
<td>9000</td>
<td>63</td>
</tr>
</tbody>
</table>

Solving times were recorded for the mathematical models, the upper bound (UB) of the optimal solution, the incumbent solution from the mathematical models and the heuristic, along
with the heuristic solution. The solving time is the number seconds that the algorithm took to obtain a solution when using the TOL model. In some cases, the algorithm was stopped after a long time had passed, because this algorithm is meant to be used in a real-time system. If termination of the algorithm was forced, the UB of the optimal solution and the incumbent solution were recorded after 1 hour of runtime. When the algorithm solved the problem in a reasonable amount of time, the incumbent solution reached the UB of the optimal solution. Thus, for the values in Table 3.4, when the incumbent and UB solution are equal, it means that the TOL model reached optimality. Included in Table 3.4 are two metrics used to compare the TOL model to the TOL heuristic: percent distance from the optimal UB ($P^{UB}$) and percent difference in the solution time ($P^{TIME}$).

Table 3.4 Average Values over 100 Runs for TOL

<table>
<thead>
<tr>
<th>Problem Size</th>
<th>Model Time (s)</th>
<th>UB of Optimal</th>
<th>Incumbent Solution</th>
<th>Heuristic Time (s)</th>
<th>Heuristic Solution</th>
<th>$P^{UB}$</th>
<th>$P^{TIME}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Model</td>
<td>Heuristic</td>
</tr>
<tr>
<td>120</td>
<td>3.20</td>
<td>4.8610</td>
<td>4.8610</td>
<td>4.140</td>
<td>4.50300</td>
<td>0%</td>
<td>7.5%</td>
</tr>
<tr>
<td>1,000</td>
<td>5.43</td>
<td>8.4690</td>
<td>8.4690</td>
<td>5.935</td>
<td>7.83345</td>
<td>0%</td>
<td>7.5%</td>
</tr>
<tr>
<td>5,000</td>
<td>43200.00</td>
<td>8.9290</td>
<td>8.2844</td>
<td>15.804</td>
<td>8.10071</td>
<td>7.3%</td>
<td>9.1%</td>
</tr>
<tr>
<td>10,000</td>
<td>86400.00</td>
<td>17.2060</td>
<td>14.7197</td>
<td>19.004</td>
<td>14.46500</td>
<td>14.5%</td>
<td>15.9%</td>
</tr>
<tr>
<td>50,000</td>
<td>86400.00</td>
<td>49.9898</td>
<td>33.9750</td>
<td>106.000</td>
<td>41.55200</td>
<td>32.1%</td>
<td>16.9%</td>
</tr>
</tbody>
</table>

The $P^{UB}$ value represents how far away the model and heuristic are from the optimal solution, or if the algorithm was terminated, the best current solution, which represents the upper bound for the optimal solution. We found that for smaller problems, using the mathematical model in the algorithm yields much better results than the heuristic with short run times (see Table 3.4). So for smaller problems, using the mathematical model yields better results than the heuristic and takes approximately the same amount of time to obtain a solution. For problems with 5,000 and 10,000 variables, the mathematical model still performs slightly better than the heuristic, but the amount of run time required for the TOL model is excessive. The heuristic, on the other hand,
performs well for these two problem sizes and can yield a solution in less than 15 seconds, on average. Thus, the user must decide whether he or she wants a slightly better solution (TOL model) or a quick solution (TOL heuristic). When the problem size increases to 50,000 variables, the heuristic outperforms the TOL model in terms of both time spent and percent difference from the UB of the optimal solution. Also included in Figure 3.14 are trend lines, which illustrate general projections for both the TOL model and heuristic if the problem size was increased further.

![Figure 3.14 Percent difference from optimal UB.](image)

### 3.4.2 Max-Min Analysis

Just as with TOL, the Max-Min model and heuristic require two parameters to be set. As in the TOL case, the $P$ parameter was set to 5% of the expected total processing time for all F-Es. The threshold parameter ($TH$) was assigned a value of 0.80 for every problem size, representing that the DM wanted as many estimates as possible with a weight certainty value of at least 80%. As with the TOL, the algorithm was run for 100 different datasets for both the Max-Min model and the heuristic, and the values from Table 3.3 were used for the different
problem sizes (not including the $TL$ value, as it is not part of the Max-Min model). The results of the Max-Min algorithm runs are shown in Table 3.5.

Table 3.5 Average Values over 100 Runs for Max-Min

<table>
<thead>
<tr>
<th>Problem Size</th>
<th>Model Time(s)</th>
<th>UB of Optimal</th>
<th>Incumbent Solution</th>
<th>Heuristic Time(s)</th>
<th>Heuristic Solution</th>
<th>$p^{UB}$</th>
<th>$p^{TIME}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Heuristic</td>
<td></td>
</tr>
<tr>
<td>120</td>
<td>3.57</td>
<td>5.000</td>
<td>5</td>
<td>9.93</td>
<td>4</td>
<td>0.0%</td>
<td>0.3595</td>
</tr>
<tr>
<td>1,000</td>
<td>4.16</td>
<td>20.000</td>
<td>20</td>
<td>21.00</td>
<td>17</td>
<td>0.0%</td>
<td>0.1981</td>
</tr>
<tr>
<td>5,000</td>
<td>6.20</td>
<td>7.000</td>
<td>7</td>
<td>21.36</td>
<td>6</td>
<td>0.0%</td>
<td>0.2900</td>
</tr>
<tr>
<td>10,000</td>
<td>707.28</td>
<td>15.000</td>
<td>15</td>
<td>35.01</td>
<td>14</td>
<td>0.0%</td>
<td>20.2020</td>
</tr>
<tr>
<td>50,000</td>
<td>86400.00</td>
<td>36.121</td>
<td>35</td>
<td>64.81</td>
<td>34</td>
<td>3.1%</td>
<td>1333.130</td>
</tr>
</tbody>
</table>

Unlike the TOL model, the Max-Min model outperforms the heuristic for all problem sizes. Even so, as the problem size increases, the percent difference from the optimal UB for the heuristic decreases. This means that the mathematical model seemingly always finds a better solution than the heuristic; however, as the problem size increases, the heuristic solution approaches that of the mathematical model and is retrieved in less time. Figure 3.15 displays how the runtimes for both the model and heuristic increase as the problem size increases. The heuristic runtime steadily increases as the problem size increases, whereas the model runtime increases dramatically once the problem size reaches 50,000 variables.
3.4.3 Entropy Analysis

The entropy models require only one parameter to be set: the total available processing time. The $P$ parameter is set to 5% of the sum of all the processing times for each option $k$ in each feature $j$ for each estimate $i$. The processing time for each option-feature-estimate combination is a uniform distribution where $P_{ijk}^X \in (1,2,\ldots,20)$ and $P_{ijk}^Y \in (1,2,\ldots,10)$. The processing time to look at all options in a feature $j$ for estimate $i$ is determined by using the $P_{ijk}^X$ and $P_{ijk}^Y$ values,

$$P_{ij}^{ALL} = \sum_{k} P_{ijk}^X + \sum_{k} P_{ijk}^Y$$

where $O$ is the number of options. The uncertainty values are set to a value of $\frac{1}{O}$ if the IF process did not assign an uncertainty value to the option-feature-estimate, otherwise $U_{ijk} \in (0,\ldots,1)$. For entropy, a value of $\frac{1}{O}$ is assigned instead of a value of a 0, because as described before, when using entropy a value of 0 implies complete certainty that the value is incorrect, not that the value is completely uncertain.

The MINLP, MILP and Heuristic were run for five different problem sizes (Table 3.6). For every option-feature-estimate there are four variables $\left(X_{ijk}^+,X_{ijk}^-,Y_{ijk}^+,Y_{ijk}^-\right)$ and for every F-E there is one variable $\left(ALL_{ij}\right)$. The problem size in terms of number of variables is then determined by calculating $4 \times (E \times F \times O) + (E \times F)$, where $E$ is the total number of estimates, $F$ is the total number of features and $O$ is the number of options per feature.

<table>
<thead>
<tr>
<th>Table 3.6 Entropy Problem Sizes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Problem Size</td>
</tr>
</tbody>
</table>

87
For each problem size, 100 different simulated datasets were solved for both the MILP and Heuristic entropy models. DUMA was run using the MINLP model for 100 different simulated datasets when the problem size was 195 variables. Once the problem size was increased, GAMS would not provide an integer solution after 24 hours when trying to solve the MINLP model. Testing this with five different datasets yielded the same result of no integer solution. Table 3.7 shows the average objective value results and runtime values over the all runs for each of the models.

<table>
<thead>
<tr>
<th>Problem Size</th>
<th>NLP Time(s)</th>
<th>MINLP Solution</th>
<th>MILP Time(s)</th>
<th>Heuristic Time(s)</th>
<th>$P_{TIME}$</th>
<th>MILP Solution</th>
<th>Heuristic Solution</th>
<th>$P_{UB}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>195</td>
<td>300</td>
<td>2.386</td>
<td>3.22</td>
<td>3.98</td>
<td>0.81</td>
<td>2.700</td>
<td>2.744</td>
<td>1.60%</td>
</tr>
<tr>
<td>780</td>
<td>86,400</td>
<td>N/A</td>
<td>7.79</td>
<td>4.98</td>
<td>1.56</td>
<td>18.225</td>
<td>18.365</td>
<td>0.99%</td>
</tr>
<tr>
<td>5,250</td>
<td>86,400</td>
<td>N/A</td>
<td>16.08</td>
<td>13.59</td>
<td>1.18</td>
<td>110.910</td>
<td>111.729</td>
<td>0.73%</td>
</tr>
<tr>
<td>41,000</td>
<td>86,400</td>
<td>N/A</td>
<td>342.33</td>
<td>35.03</td>
<td>9.77</td>
<td>537.191</td>
<td>542.132</td>
<td>0.95%</td>
</tr>
<tr>
<td>102,500</td>
<td>86,400</td>
<td>N/A</td>
<td>1518.72</td>
<td>144.74</td>
<td>10.49</td>
<td>1339.485</td>
<td>1347.267</td>
<td>0.58%</td>
</tr>
</tbody>
</table>

The heuristic objective values are within 2% of the MILP solution for all problem sizes tested, which means if the solution time for the MILP model starts to become too large, the heuristic model is a viable option. Figure 3.166 demonstrates how the time of each model increases along with the objective values as the problem size increases. The time of the MILP model starts to grow exponentially, whereas the run time when using the Heuristic model grows in a more linear fashion. The graph also shows how close the objective value is when using either model. For the $P_{TIME}$ statistic, both models run in a similar time frame for smaller

88
problems, but once the number of variables is increased to 41,000, the heuristic model runs almost 10 times faster than the MILP model. This means that when the number of variables increases past 5,250, the heuristic model should be used in CIP.

3.5 Summary

In this chapter, I further examined CIP and tested it with a more complicated model. I presented a model that combines uncertainty values from the IF process using Shannon entropy. Using entropy required including another dimension to the data from the IF process that was not used in the previous models. Since entropy is nonlinear, I developed a rule-based system in order to build a linear model so that it can be solved in a reasonable amount of time for larger problem sizes. Even so, the IP model required a large amount of time to obtain a feasible solution, so I also developed a heuristic that could be used in CIP. The heuristic relaxed the variables of the IP model and recursively added constraints to the relaxed IP until all variables reached integer values. I found that the heuristic solved significantly more quickly once the
number of variables increased beyond 5,250; the heuristic solution was also within 2% of the IP solution in all cases. Thus, more complex models can be solved in CIP, but heuristics are needed in order to use CIP in real time.

Based on these results, further exploration into using complex models in CIP is warranted. First, it is important to explore the use of multiple human fusers with different areas of expertise. This allows for the inclusion of what is called learning and forgetting of performing tasks, which means that as a human fuser performs a task repeatedly, the time it takes to complete the task will decrease; conversely, as a human fuser performs a task less often, the amount of processing time will increase. Second, including a data degradation function will decrease the reliability of data points the longer they remain untouched.
Chapter 4: Multiple Human Fusers within the Cognitive Injection Process

4.1 Introduction

With increased data availability due to technological advances, DMs must rely on tools that can focus their attention on specific pieces of data. IF has been a useful tool for DMs that takes data from different types of sensors and combines them to create pieces of information. These different sensors can be hard sensors (traditional) or soft sensors (humans). The IF process also attaches a reliability value to the pieces of information that are extracted from the raw data. The reliability of data is an attribute that can help a DM determine which data to use and which to discard. As introduced in previous chapters, CIP provides a way for a system to focus a DM’s attention on the most reliable pieces of data. CIP is driven by a mathematical model or heuristic that interacts with a human fuser who is a subject matter expert on the dataset, problem scenario or sensors that are retrieving data. This allows for a human to modify the reliability values provided by the IF process. By using CIP, the DM can use a better dataset to determine a COA.

Previously, I presented mathematical models and heuristics that can be used within CIP. In this chapter, I address Class III problems, which deal with multiple human fuses in a system. The idea is that each human fuser can have a different area of expertise. For example, one human fuser may be a data analyst who knows datasets better than anyone, another may have a Ph.D. in electrical engineering and have written a dissertation on sensor characteristics, and a third may have a Ph.D. in psychology and be able to predict how soft sensors will react in different situations. If these three human fusers are all working on the same problem scenario, they will each have a different comfort level associated with modifying different pieces of data. This will result in each human fuser modifying the same data point in a different amount of time.
and changing the reliability value of that data point by a different amount. This type of model using multiple human fusers who have different areas of expertise can correlate directly to what is known as learning and forgetting in a workforce setting. The idea is that as someone performs a task more often, they become more comfortable with it and can complete it more quickly the next time they perform the same task. Learning and forgetting also adds another dimension to the models – time. By including time as a dimension in the models, the idea of data degradation can be incorporated, which allows for the reliability of data to decrease as time passes.

Incorporating two new models (learning and forgetting, with and without data degradation) and defining a new class of problems makes it important to revisit all models presented in the previous chapters and provide guidelines on when to use each model. Clearly, each model has its own parameters in terms of problem type. For example, when there are multiple human fusers, one of the learning and forgetting models is likely most appropriate. But there are some instances when a choice between models must be made. To address this problem, I introduce a decision tree for CIP implementers to help them determine which model to use.

4.1.1 CIP with Multiple Human Fusers

The Cognitive Injection Process (CIP) is a selection algorithm that interacts with human fusers. Since there is more data available, a human fuser, much like a DM, needs a tool to limit the amount of data he or she has to evaluate; that tool is CIP. Figure 4.1 shows how a system that uses the IF process would be modified to include CIP.
The data sources include both hard sensors (MASINT, SIGINT, and GEOINT) and soft sensors (OSINT, COMINT, and HUMINT). The IF process then takes the data and combines them to create pieces of information called estimates. These estimates have common attributes, termed features. The IF process extracts the features and assigns reliability values to them within the estimates. Typically, the estimates from the IF process are used to develop a situational understanding, which is then used by the DM to determine a COA. This situational understanding is a combination of situational awareness (SA) and impact assessment (IA). Situational awareness is an understanding of the problem space and the elements within it, including projecting the elements into the near future (Endsley, 1995). Impact assessment, on the other hand, is understanding what will happen if a COA is implemented in a given situation. CIP and the human fusers have been incorporated into the system between the IF process and creating a situational understanding for the DM. This allows the reliability values of data points flagged as being extremely important by the CIP process to be modified by human fusers before they are used to determine the situational understanding. By modifying the reliability values, the
situational understanding will change for the DM and will provide the DM with more accurate information about the problem space and the potential effects of each COA.

4.1.2 Choosing a Model

I have presented eight different mathematical models that can be used in CIP. While introducing many mathematical models provides flexibility, it becomes difficult to decide which model to use for each problem scenario. In particular, there may be instances where two or more models fit a single problem scenario. I have split the eight models up into three different classes of problems (Table 4.1).

<table>
<thead>
<tr>
<th>Problem Type</th>
<th>Description</th>
<th>Mathematical Models</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class I</td>
<td>These types of problems are the most basic of the three. They deal solely with how to combine the certainty values of the features for the estimates. Any new uncertainty calculation technique would be included in this problem class.</td>
<td>Total Certainty Knapsack, Total Certainty + Importance Knapsack, Total Weighted Certainty Knapsack, Entropy</td>
</tr>
<tr>
<td>Class II</td>
<td>These types of problems are more complicated than Class I. They are problems where the Decision Maker is available and has input specifications to help customize the model to be used.</td>
<td>Top Of List (TOL), Max-Min (MM)</td>
</tr>
<tr>
<td>Class III</td>
<td>These types of problems are the most complex. They are problem instances where there are multiple human fusers at the disposal of the system. Also problems that use time as an index will be classified here as well.</td>
<td>Learning and Forgetting, Learning and Forgetting with Data Decay</td>
</tr>
</tbody>
</table>

In general, Class I problems focus on how to combine the different certainty values from the IF process. Class II problems allow for the DM to be more involved in the system than just...
deciding on a COA. Finally, Class III problems involve multiple human fusers. The difficulty in choosing between models lies in determining a common ground by which to compare them. Objective values, for instance, cannot be compared because each model’s objective is completely different. Obviously, the different models can be categorized by specifying the problem type (i.e., average or entropy technique, multiple human fusers or one, etc.). Table 4.2 presents some guidelines for choosing a model. The green color indicate that the problem or data description fits very well within a model. For example, when probability density functions (PDFs) are available for a dataset, one should use the entropy model. On the other hand, if PDFs are not available, then using the entropy model is not feasible (indicted by red). The letters within the cells represent the complexity of the model used with those attributes included. Cases with yellow cells indicate that the model can be used, but it may not be ideal. For example, for both Class II problem types (TOL and MM), one could estimate the DM’s input and still use these models within CIP, but the estimation may not be accurate and provide the DM with an incomplete set of information.
Table 4.2  Model Selection Guidelines

<table>
<thead>
<tr>
<th>Equal Feature Importance</th>
<th>L</th>
<th>M</th>
<th>M</th>
<th>L</th>
<th>L</th>
<th>M</th>
<th>M</th>
</tr>
</thead>
<tbody>
<tr>
<td>New Data Point</td>
<td>M</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>L</td>
<td>M</td>
</tr>
<tr>
<td>Different Feature Importance</td>
<td>M</td>
<td>M</td>
<td>L</td>
<td>M</td>
<td>L</td>
<td>L</td>
<td>M</td>
</tr>
<tr>
<td>PDF Available</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>H</td>
<td>M</td>
<td>M</td>
<td>H</td>
</tr>
<tr>
<td>PDF not Available</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>MH</td>
<td>L</td>
<td>L</td>
<td>M</td>
</tr>
<tr>
<td>DM Available</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>H</td>
<td>MH</td>
<td>MH</td>
<td>MH</td>
</tr>
<tr>
<td>DM not Available</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>MH</td>
<td>L</td>
<td>L</td>
<td>M</td>
</tr>
<tr>
<td>Single HF</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>M</td>
<td>L</td>
<td>L</td>
<td>L</td>
</tr>
<tr>
<td>Multiple HF’s</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>H</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>Decay Occurs</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>H</td>
<td>M</td>
<td>M</td>
<td>MH</td>
</tr>
<tr>
<td>Decay doesn’t Occur</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>M</td>
<td>L</td>
<td>L</td>
<td>M</td>
</tr>
<tr>
<td>HF’s have Expertise</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>H</td>
<td>M</td>
<td>M</td>
<td>H</td>
</tr>
<tr>
<td>HF’s don’t have Expertise</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>M</td>
<td>L</td>
<td>L</td>
<td>MH</td>
</tr>
<tr>
<td>KTC</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>M</td>
<td>L</td>
<td>L</td>
<td>MH</td>
</tr>
<tr>
<td>KTCW</td>
<td>KTC Total Certainty + Weight Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KTWC</td>
<td>Knapsack Total Weighted Certainty Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOL</td>
<td>Top of List Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LF</td>
<td>Learning and Forgetting Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KTC</td>
<td>Knapsack Total Certainty Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>KTWC</td>
<td>Knapsack Total Weighted Certainty Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>TOL</td>
<td>Top of List Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LF</td>
<td>Learning and Forgetting Model</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

These guidelines also help when combining two or more of the models. The knapsack problem implements a technique that averages the certainty values together, which is actually used within the TOL, MM and both learning and forgetting models. Assume that the probability distribution functions required in order to use the entropy model are available, along with a DM who only wants to look at the top of the estimate list. In this situation, using both the TOL model and the entropy model would be ideal. To do this, the objective function must be
modified and then both sets of constraints from each of the two models must be included. The ability to combine techniques adds a new level of decisions for a system user. The only common statistic that can be used to compare the models and different model combinations is solution time. Although all the models technically could be combined, the long solution times make it unrealistic for real-time decision making contexts. Figure 4.2 shows an example of a decision tree based on solution time that can be used to select the mathematical model or heuristic method for a specific problem scenario. The idea behind this decision tree is that a problem scenario that can fit into one of the more complex classes (Class III) can be simplified to also fit the simpler class of problems (Class I); that is, the complex classes contain the simpler class sets. The decision tree therefore begins by assessing whether there are multiple human fusers available for the system. If multiple human fusers are available with different areas of expertise, the decision tree will then check whether or not further traits should be added. These traits are ways for ideas described in the different mathematical models to be combined. Traits include:

1. Combining certainty values using average or entropy technique;
2. Including feature importance or not; and
3. Including data decay or not.

Once the traits are defined, the solution time is checked. If it is not acceptable, the recommendation is to use the heuristic defined for the problem type. If it is still not acceptable, the traits are removed one at a time until the solution time is deemed acceptable. The feature importance trait is not removed because it does not add any additional complexity to a model. If the solution time is still unacceptable after: (a) all options on the decision tree have been

1 A larger divided version of Figure 4.2 can be seen in the Appendix.
explored, (b) the problem type has been converted to the heuristic, and (c) all traits have been removed, the next step is to switch to a different class of problems that is less complex.
Figure 4.2 Decision tree.
In the following sections, I further discuss the addition of multiple human fusers and the difficulties associated with solving complex models. In section 4.2, I explore the literature related to information fusion, learning and forgetting, and data decay. In section 4.3, I describe a problem area that could benefit from CIP and present the models and heuristics for situations with multiple human fusers. In section 4.4, I discuss experimentation and analyze the different models by comparing solution time and closeness to the optimal solution or UB. Finally, I summarize the findings and discuss future work that could further enhance the notion of adding a human fuser to the data processing section of a system.

4.2 Background Literature

IF is the process of combining data to make inferences about a situation while improving data accuracy (Hall & Llinas, 1997). It was developed in the 1980s primarily for use on military projects, but soon thereafter, IF was applied to other research areas. Terminological differences among researchers caused confusion, and a Data Fusion Subpanel within the Joint Directors of Laboratories (JDL) (White, 1991) was tasked with making a cohesive set of terminology for IF. The JDL reconvened later to further solidify specifics, in particular the five “levels” of data fusion (Steinberg, Bowman & White, 1999). The five levels are presented in Table 4.3.

Table 0.3 Levels of Information Fusion

<table>
<thead>
<tr>
<th>Level</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Data Assessment</td>
<td>Estimate and predict signal or feature states.</td>
</tr>
<tr>
<td>1</td>
<td>Object Assessment</td>
<td>Estimate and predict object parametric and attributive states.</td>
</tr>
<tr>
<td>2</td>
<td>Situation Assessment</td>
<td>Estimate and predict the relationships among the different objects and how they affect the states of related objects.</td>
</tr>
<tr>
<td>3</td>
<td>Impact Assessment</td>
<td>Estimate and predict the impact that a course of</td>
</tr>
</tbody>
</table>
action will incur on a system.

**Process Refinement**

Estimate and predict a system performance and could sensors be moved to give better results.

Information fusion remains an important tool in military settings for tasks such as recognizing targets (Hall, Linn & Llinas, 1991), obtaining situational awareness and measuring impact assessment. A number of other IF frameworks have been proposed (Endsley, 1995; Salerno, 2002), but where and how a human fuser can be injected in the overall IF system remains an open question. As connectivity increased and technology advanced, researchers began to explore hard and soft data fusion (Hall, Llinas, McNeese & Mullen, 2008), creating a whole new area of research on how to associate uncertainty values with heterogeneous data types.

Learning curves were first described in 1885 when Hermann Ebbinghaus found that the time required to memorize a word increased as the length of the word increased (Ebbinghaus, 1885). Subsequently, in 1936 Theodore Paul Wright found a learning effect in the building of aircrafts and developed a mathematical model for the learning curve (Wright, 1936). He developed two equations based on the same concept. The first equation (4.A) yields a value, $Y_x$, which represents the number of hours it would take to make the $x^{th}$ unit. The second equation (4.B) yields a cumulative average or the average number of hours it would take to make the $x^{th}$ unit.

$$Y_x = K \cdot x^{\log_2 b} \quad 4.A$$

$$Y_x = K \cdot \frac{1}{1 + \log_2 b} \cdot x^{1 + \log_2 b} \quad 4.B$$

The parameter $K$ represents the amount of time required to make the first unit, $x$ is the unit number and $b$ is the learning percentage. These equations result in a curved graph that starts
at $K$ and eventually levels off at a maximum potential. Figure 4.3 illustrates what a learning curve would look like for someone who can produce the first unit in 10 time units and learns at a rate of 10%. The amount of time to produce the next unit continues to decrease, but the change in production time becomes smaller.

![Learning Curve](image)

Figure 4.3 Example learning curve.

In the same publication, Ebbinghaus also discovered a forgetting curve. In other words, when a person stops performing a task, he or she will forget how to do it. He developed a formula for forgetting:

$$R = e^{-\frac{t}{s}}$$

where $R$ represents the amount of memory retained, $S$ is the forgetting parameter and $t$ is time. As research in the field progressed, people began combining the idea of learning and forgetting into one expression and developed learning and forgetting curves. Numerous different models have been developed to account for learning and forgetting. The learn-forget curve model (LFCM) developed by Jaber and Bonney (1996) assumes that at some point, everything that has
been learned is completely forgotten and the production rate drops back down to the initial production rate of the first unit. It also assumes that the performance time of the learning curve and forgetting curve is equal if an interruption occurs, which assures a continuous curve when combining the two. Jaber (2003) later defined seven attributes of forgetting that should be considered when developing learning and forgetting model. Jaber found that his LFCM included 6 of the 7 attributes with the seventh attribute left untested. Another learning and forgetting model can be found in Chapter 4 of a book authored by Nembhard (2007) that includes both learning and forgetting coefficients in one formula (4.D) to determine the production rate of a worker on a task during a time period.

\[ P_{ijt} = l_{ij} + K_{ij} \left[ 1 - \exp\left(-\frac{1}{L_{ij}} \sum_{k=1}^{t} X_{ijk}\right) \right] \exp\left[\frac{1}{F_{ij}} \left( \sum_{k=1}^{t} X_{ijk} - t \right) \right] \]  

The learning coefficient is represented by the \( L_{ij} \) value and the first bracketed term represents the learning that has occurred. The forgetting coefficient is represented by the \( F_{ij} \) value and the forgetting is calculated by the second bracketed term. The \( K_{ij} \) parameter represents the steady-state production rate, and the \( l_{ij} \) value represents the initial production rate and provides a lower bound for the production rate potential. This representation of learning and forgetting will be the basic idea used to develop the learning and forgetting mathematical model.

4.3 Methodology

4.3.1 Problem Description

Assume that the problem scenario has a context in which the sensors are retrieving data on a person of interest. The sensor set consists of two hard sensors, satellite images and facial recognition software; and two soft sensors, a sniper and a pilot. The data from the sensors is sent to the IF process, which in turn extracts estimates, features and certainty values. Figure 4.4
shows what the IF process extracts from the sensor data. The upper portion of the figure describes which sensor data is used to create each estimate. For example, the information in estimate 1 (E1) is based on data from both hard sensors. Each estimate (E1 … E20) represents a piece of information that is used by a DM to determine the COA. The features (F1 … F4) are commonalities among the different pieces of information. For example, one estimate could refer to a blue car and another could refer to a tank, but both would be categorized as vehicle features. The IF process also assigns a certainty value for each feature in every estimate. A value of 0.79, as seen in F1 – E11, for example, means that the data describing the vehicles has a 79% probability of being accurate. These values are the data points that will be modified by the human fuser to improve the DM’s knowledge of the area of interest.

4.3.2 Cognitive Injection Process

The output of estimates, features and certainty values from the IF process are the inputs into CIP (Figure 4.5). The certainty values are used by the mathematical model or the heuristic
to determine which F-E data point each human fuser should evaluate. The human fusers then modify the certainty values of the selected points. The certainty values of all the features in each estimate are combined to give each estimate a total certainty. The estimates are then rank ordered on a list for the DM with the most certain estimates at the top, allowing a DM to see the most reliable information first and avoid using uncertain information when determining the COA. If the DM wants more input from the human fuser or if he or she is not ready to determine a COA, the mathematical model or heuristic can proceed, providing the human fusers with another set of data points to modify.

![Figure 4.5 CIP flow.](image)

### 4.3.3 Mathematical Models and Heuristics

In this section, I present two mathematical models that can be used in the algorithm described above. I also present a heuristic procedure for each of the mathematical models, which can be used once the problem size becomes too large to be solved by CPLEX in a reasonable amount of time. These models are considered to be Class III problems and really should be used in scenarios in which multiple human fusers are available or when human fusers have varying expertise to perform different tasks.

#### 4.3.3.1 Learning and Forgetting Model (LF-Model)
Learning and forgetting allows the model to consider a person’s expertise when evaluating different features. It also allows for a change in expertise based on repeatedly performing the same task or not performing the task as explained in section 4.2. To apply learning and forgetting to the IF context, the worker is a human fuser, the task is observing a data point (F-E combination), and time is different time periods. Computing expertise levels using a nonlinear model results in a Mixed Integer Non Linear Problem (MINLP) formulation. For arbitrarily large problem sizes, these problems become computationally intractable.

Due to the complexity of the MINLP I have made assumptions to linearize the model so that it can be used in real-time scenarios. The assumptions are:

1. The expertise of a human fuser will increase for a feature if he or she was assigned to modify it during the previous time period.
2. The expertise of a human fuser will decrease for a feature if he or she was not assigned to modify it during the previous two time periods.
3. Expertise levels not only affect the processing time of a human fuser, but also the quality of the certainty value modifications.

The first two assumptions are used to determine how a human fuser’s expertise increases or decreases. This enables us to forego the nonlinear learning and forgetting expression and use constraints to represent how a human fuser learns or forgets how to do a task. Let $\delta$ represent the expertise level that a human fuser can have where $\delta = 1$ signifies lost expertise and $\delta = L$ represents the highest expertise level ($L$). For the learning assumption, a human fuser’s expertise level for a feature will increase to the next level if he or she observed that feature in at least one estimate during the previous time period. For the forgetting assumption, a human fuser’s expertise level for a feature will decrease to the next level if he or she did not observe the feature
in any of the estimates during the previous two time periods. Figure 4.6, Figure 4.7, and Figure 4.8 represent how a human fuser’s expertise could increase or decrease over time for different features. In this example, there are three different levels and three different features. The human fusers’ expertise levels always remain at $\delta = 1$ for feature 1, meaning that the human fuser never observed feature 1 for any of the estimates during any time period. The second feature is evaluated by the human fuser at time $t = 5$. This causes the expertise to jump to level 2 ($\delta = 2$). The feature is never evaluated again, so after two time periods ($t = 6, t = 7$) the expertise level drops back to level 1.
Another assumption states that learning and forgetting not only affect a human fuser’s processing time, but also change how much a human fuser can increase a certainty value. This assumption makes the model more realistic, because a human fuser who is not an expert on certain features is going to have uncertainty associated with modifying those values. With the inclusion of the assumptions, the LF-Model can be formulated as a Mixed Integer Problem (MIP), which is linear and will solve much faster than the original MINLP. The general formulation is shown below:

\[
X_{ijht} = \begin{cases} 
1, & \text{If human fuser } h \text{ should observe } F-E \text{ } i-j \text{ during time period } t \\
0, & \text{Otherwise} 
\end{cases}
\]

\[
Q_{iht}^\delta = \begin{cases} 
1, & \text{If human fuser } h \text{ is at expertise level } \delta \text{ for feature } i \text{ during time period } t \\
0, & \text{Otherwise} 
\end{cases}
\]

\[
0 \leq V_{ijht} \leq 1
\]

\[
0 \leq U_{ijt} \leq 1
\]
Maximize $Z = \sum_i \sum_j \sum_t U_{ijt}$

Subject To: \(\sum_h X_{ijht} \leq 1\) \(\forall i, j, t\) \(4.A\)

\(2 - (U_{ijt-1} + 0.001) \geq X_{ijht}\) \(\forall i, j, t\) \(4.B\)

\(U_{ijt-1} + \sum_h V_{ijht} \geq U_{ijt}\) \(\forall i, j, t\) \(4.C\)

\(U_{ijt} \geq U_{ijt-1}\) \(\forall i, j, t > 0\) \(4.D\)

\(P_{MOD}^\delta * P_{ij} * (Q_{iht}^\delta + X_{ijht} - 1) \leq PT_{ijht}\) \(\forall i, j, h, t, \delta\) \(4.E\)

\(\sum_i \sum_j PT_{ijht} \leq P\) \(\forall h, t\) \(4.F\)

\(U_{MOD}^\delta (1 - U_{ijt-1}) + (2 - (X_{ijht} + Q_{iht}^\delta)) \geq V_{ijht}\) \(\forall i, j, t\) \(4.G\)

\(V_{ijht} \leq X_{ijht}\) \(\forall i, j, h, t\) \(4.H\)

\(\sum_\delta Q_{iht}^\delta = 1\) \(\forall i, h, t\) \(4.I\)

\(Q_{iht}^1 \geq Q_{iht-1}^1 - \sum_j X_{ijht-1}\) \(\forall i, h, t > 1\) \(4.J\)

\(Q_{iht}^\delta \geq (Q_{iht-1}^{\delta-1} + X_{ijht-1}) - 1\) \(\forall i, h, t > 1, \delta > 1\) \(4.K\)

\(Q_{iht}^\delta \geq \left(\left(Q_{iht-1}^\delta + Q_{iht-2}^{\delta-1} + X_{ijht-2}\right) + (1 - X_{ijht-1})\right) - 3\) \(\forall i, h, t > 2, \delta > 1\) \(4.L\)

\(Q_{iht}^\delta \geq \left(\left(Q_{iht-1}^{\delta+1} + Q_{iht-2}^{\delta+1}\right) - 2\right) + \left(1 - (X_{ijht-1} + X_{ijht-2})\right)\) \(\forall i, h, t > 2, L > \delta\) \(4.M\)

\(Q_{iht}^L \geq (Q_{iht-1}^L + X_{ijht-1}) - 1\) \(\forall i, h, t\) \(4.N\)

\(Q_{iht}^\delta \geq (Q_{iht-1}^\delta + X_{ijht-1}) - 1\) \(\forall i, h, t\) \(4.O\)

The processing time required for human fuser $h$ to observe F-E $i-j$ during time period $t$ based on his or her expertise.
with the following parameters:

\[ P^\delta_{MOD} \quad \text{Processing time modifier at expertise level } \delta, \text{ where } P^1_{MOD} \geq \cdots \geq P^\delta_{MOD} \geq \cdots \geq P^L_{MOD} \]

\[ P_{ij} \quad \text{Processing time required to observe F-E } i-j \]

\[ 0 \leq U^\delta_{MOD} \leq 1 \quad \text{Certainty value modifier at expertise level } \delta, \text{ where } U^1_{MOD} \leq \cdots \leq U^\delta_{MOD} \leq \cdots \leq U^L_{MOD} \]

The objective of the L-F model is to maximize the sum of all of the certainty values over all of the time periods (4.A). This is the objective instead of just the sum of all the certainty values so that all of the tasking can be pushed into the first time period. Constraint 4.B ensures that at most one human fuser will be tasked to modify a certainty value of an F-E during a time period. This means that multiple human fusers cannot observe the same F-E during a time period, which limits wasted tasking. It is obvious that if one human fuser is tasked to modify an F-E and another human fuser with better expertise is going to modify the same one, the first human’s time will be wasted. The next set of constraints (4.C-4.E) deal with updating the certainty values and solving the numerical issues that can occur. It is possible that none of the modified certainty values \((U^\delta_{MOD})\) will equal 1, in which case the human fusers’ expertise will experience marginal gains with the progression of time. Therefore, the model will never terminate. Constraint 4.C fixes this problem, where if the sum of the last certainty value \((U_{ijt-1})\) with a very small number (0.001) is greater than 1, then no observation should be made on that F-E, as the gain would be minimal. Constraint 4.D updates the certainty value by considering the previous time period’s value. Lastly, constraint 4.E simply states that the certainty value can never decrease from the previous time’s value for an F-E.

The constraints (3.3.1.F and 3.3.1.G) deal with the processing times. Assume that there are three levels of expertise (1, 2, and 3). This results in:
\[ P_{MOD}^1 \cdot P_{ij} \cdot (Q_{ijht}^1 + X_{ijht} - 1) \leq PT_{ijht} \quad \forall i, j, h, t \]
\[ P_{MOD}^2 \cdot P_{ij} \cdot (Q_{ijht}^2 + X_{ijht} - 1) \leq PT_{ijht} \quad \forall i, j, h, t \]
\[ P_{MOD}^3 \cdot P_{ij} \cdot (Q_{ijht}^3 + X_{ijht} - 1) \leq PT_{ijht} \quad \forall i, j, h, t \]
\[ \sum_i \sum_j PT_{ijht} \leq P \quad \forall h, t \]

The first three of the expanded 4.F constraints set the processing time values for each F-E for all the human fusers and for each time period. Since only one of the \( Q \) variables can be set to a value of 1 (4.K), only one of the three constraints will have a left hand side (LHS) value greater than 0. This means that only one of the \( P \) values will be set to a value greater than 0, which represents the processing time required to perform an observation on an F-E during a time period for a human user’s appropriate expertise level. The LHS also includes the \( X \) variable, which allows the LHS to be set to 0 when no observation is made. This in turn sets the processing time appropriately to a value of 0, as no observation is made. Each constraint also has a different processing time modification value for each expertise level where the higher the expertise, the lower the modification value, resulting in less required processing time.

Constraints 4.H and 4.I are used to control how much the certainty value of an F-E will increase. Again, with the example of a three level expertise problem, the constraints are expanded to:

\[ U_{MOD}^1(1 - U_{ijt-1}) + \left(2 - (X_{ijht} + Q_{ijht}^1)\right) \geq V_{ijht} \quad \forall i, j, h, t \]
\[ U_{MOD}^2(1 - U_{ijt-1}) + \left(2 - (X_{ijht} + Q_{ijht}^2)\right) \geq V_{ijht} \quad \forall i, j, h, t \]
\[
U_{MOD}^3 (1 - U_{ijt-1}) + \left(2 - (X_{ijht} + Q_{iht}^3)\right) \geq V_{ijht} \quad \forall i, j, h, t
\]
\[
V_{ijht} \leq X_{ijht} \quad \forall i, j, h, t
\]

The first three constraints determine the maximum increase value by setting an upper bound for \( V \). Only one of the RHS’s is set to a value that is less than 1 because just as before, only one \( Q \) variable is set to 1 (4.J). The RHS that is binding will have a value that represents the increase in certainty allowed for the given expertise, where the higher the expertise, the larger the possible increase in certainty. The last constraint, 4.I, ensures that no increase can be made if no observation is made. Constraint 4.J limits the number of \( Q \) variables that can be set, but ensures that a human fuser for a feature cannot be said to have more than one expertise level during a time period.

The next constraint (4.K) deals with the lowest level of expertise (level 1). It states that if a human fuser was at an expertise level of 1 during the previous time period and nothing was observed, then he or she remains at expertise level 1. This is required, because one cannot drop below an expertise level of 1. Constraints 4.L through 4.N control the flow of all expertise and the expansion of those constraints for a three level system can be seen below:

\[
Q_{iht}^2 \geq (Q_{iht-1}^1 + X_{ijht-1}) - 1 \quad \forall i, h, t
\]
\[
Q_{iht}^3 \geq (Q_{iht-1}^2 + X_{ijht-1}) - 1 \quad \forall i, h, t
\]
\[
Q_{iht}^2 \geq \left((Q_{iht-1}^2 + Q_{iht-2}^1 + X_{ijht-2}) + (1 - X_{ijht-1})\right) - 3 \quad \forall i, h, t
\]
\[
Q_{iht}^3 \geq \left((Q_{iht-1}^3 + Q_{iht-2}^2 + X_{ijht-2}) + (1 - X_{ijht-1})\right) - 3 \quad \forall i, h, t
\]
\[
Q_{iht}^2 \geq \left((Q_{iht-1}^3 + Q_{iht-2}^3 - 2) + (1 - (X_{ijht-1} + X_{ijht-2}))\right) \quad \forall i, h, t
\]
The first two constraints represent 4.L and are used to increase the expertise level of a human fuser when an observation was made during the previous time period. The third and fourth constraints represent 4.M where the expertise level remains the same if the expertise level increased previously and an observation occurred two time periods ago but not in the last time period. This prevents the expertise level from dropping if nothing was observed during last time period, as per the assumptions made for this model. Constraint 4.N is used to decrease the expertise level if the human fuser was at the same expertise level during the past two time periods and did not observe anything at those times. The last generalized constraint (4.O) is used to keep the expertise level of a human fuser at the maximum level \( L \) if an observation is made. This ensures that the expertise level is bounded at \( L \) and does not increase beyond it.

Even after the conversion from a MINLP to a MIP, this model is still considered NP-hard and can take a large amount of time to generate an optimal feasible solution. Thus, a heuristic is necessary in order to use the idea of learning and forgetting in real time systems. I exploit solving numerous relaxations of the MIP model by continuously adding constraints, resulting in a mixed integer solution. The heuristic flow can be seen in Figure 4.9 and is split up into two processes: the X-Process and Q-Process. The X-Process is used to adjust the lower bounds of the \( X \) variables, which forces them to be a value of 1 and essentially makes the solution an integer solution. The heuristic first sets a counter variable \( T \), which represents the time periods, to 1. Then the LP relaxation of MIP from above is solved for all time periods. The resulting \( X \) values from the LP relaxation for time period \( T \) are checked to see if they are integer values (0 or 1) for all features \((i)\), estimates \((j)\) and human fusers \((h)\). If at least one \( X \) for time period \( T \) is not an integer, then the \( i-j-h \) combination must be found where \( X \) is not an integer and has the largest
\[
\frac{v_{ijhT}}{\max_{\delta}(p^\delta_{ijhT})}
\]
This ratio represents the feature-estimate-human fuser that provides the most gain in certainty \((V)\) with the shortest processing time. The heuristic then sets the \(X\) variable’s lower bound corresponding to the largest ratio to 1 rather than to 0. This continues until all the \(X\)’s are set to 0 or 1, at which time the Q-Process ensues.

The Q-Process begins by checking if all the \(Q\) variables are integer values. If they are, then there is no need to solve an additional LP relaxation; otherwise, the lower bounds of the \(Q\) variables must be set appropriately in order to obtain an all integer solution. Setting the lower bounds of the \(Q\) variables is accomplished by using the \(Q\) values and \(X\) values from previous time periods. The step in Figure 4.9 that represents determining the lower bounds of the \(Q\)’s is expanded and can be seen in Figure 4.10. Because of constraint 4.J, only one of the \(Q\)s must be set to 1, forcing the \(Q\)s that represent the other expertise levels for the same \(i-h-T\) to be set to 0.
4.3.3.2 Learning and Forgetting with Data Decay

As described, learning and forgetting deals with the abilities of the human fusers to perform a task. However, data also becomes less and less reliable with the progression of time. For instance, imagine a piece of data stating that person A is at location B at time 0. As time progresses, it becomes less and less likely that person A is still at location B. This is true for many datasets, in that the relevance and reliability of data for a problem scenario decreases as more time is taken by the DM to decide on a COA. Including this data decay element can be accomplished by simply building on the learning and forgetting model presented in the previous section. As with the nonlinear learning and forgetting models, assumptions are made to include data decay in the mathematical model. It is assumed that if an F-E certainty value has not been evaluated by any human fuser for three consecutive time periods, the certainty value decreases by a quarter; after a fourth, the certainty decreases by half; and after a fifth consecutive time period, the certainty decreases by 100%. To accomplish this, constraint 4.D is modified,
constraint 4.E is removed and the following constraints are included in the complete mathematical model from before:

\[ U_{ijt-1} + \sum_{h} V_{ijht} - D_{ijt} \leq U_{ijt} \quad \forall i, j, t > 1 \quad 4.P \]

\[ D_{ijt} \leq 1 - X_{ijht-1} \quad \forall i, j, h, t > 3 \quad 4.Q \]

\[ D_{ijt} \leq 1 - X_{ijht-2} \quad \forall i, j, h, t > 3 \quad 4.R \]

\[ D_{ijt} \leq 1 - X_{ijht-3} \quad \forall i, j, h, t > 3 \quad 4.S \]

\[ D_{ijt} \geq \frac{1}{4} U_{ijt-1} - \sum_{h} (X_{ijht-1} + X_{ijht-2} + X_{ijht-3}) \quad \forall i, j, t > 3 \quad 4.T \]

\[ D_{ijt} \geq \frac{1}{2} U_{ijt-1} - \sum_{h} (X_{ijht-1} + X_{ijht-2} + X_{ijht-3} + X_{ijht-4}) \quad \forall i, j, t > 4 \quad 4.U \]

\[ D_{ijt} \geq U_{ijt-1} - \sum_{h} (X_{ijht-1} + X_{ijht-2} + X_{ijht-3} + X_{ijht-4} + X_{ijht-5}) \quad \forall i, j, t > 5 \quad 4.V \]

where \( 1 \geq D_{ijt} \geq 0 \) and \( D_{ijt} \) represents the level of certainty that an F-E will decrease for time period \( t \). The modification of 4.D to 4.P is done to include the \( D \) variable so that the certainty values can be decreased when data decay occurs. Similarly, 4.E must be removed since certainty values now decrease as time passes. Constraints 4.Q through 4.S set the \( D \) variable to 0 if any observations are made in the past three time periods for any of the human fusers. If no observations are made and decay is going to occur, the decrease for the certainty value must be determined. This is accomplished by including constraints 4.T through 4.V.

Since the changes made to the learning and forgetting math model do not include adding any additional integer variables, the heuristic that is used for the learning and forgetting model discussed previously can also be used for learning and forgetting that includes data decay.
4.4 Experimentation

To analyze CIP, simulated datasets were used as input values and Optimization Programming Language (OPL) was used for the mathematical programs and heuristics. OPL uses the CPLEX as its solver. For experimentation purposes, CIP was run for each mathematical model and heuristic on five sizes of datasets and 100 different simulated datasets were used for each different dataset size. All problems were run on an Asus laptop with an Intel Core I5 processor (2.30GHz), 8 GB of RAM and a 64-bit operating system.

### 4.4.1 Analysis

The learning and forgetting model requires the user to set three different parameters. The $P$ parameter, which represents the processing time available for each human fuser for every time period, is arbitrarily set to a value of 10, which is maintained throughout the experimentation for all problem sizes. The other parameters that must be set are the processing time ($P_{MOD}^\delta$) and certainty ($U_{MOD}^\delta$) modification values. The values of the modification parameters are shown in Table 4.4, and were maintained for all problem sizes so that the analysis would not be skewed by the values of the parameters. The five different problem sizes are presented in Table 4.0.5.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P_{MOD}^1$</td>
<td>1.00</td>
</tr>
<tr>
<td>$P_{MOD}^2$</td>
<td>0.75</td>
</tr>
<tr>
<td>$P_{MOD}^3$</td>
<td>0.50</td>
</tr>
<tr>
<td>$U_{MOD}^1$</td>
<td>0.50</td>
</tr>
<tr>
<td>$U_{MOD}^2$</td>
<td>0.75</td>
</tr>
<tr>
<td>$U_{MOD}^3$</td>
<td>1.00</td>
</tr>
</tbody>
</table>
Table 4.0.5 Problem Sizes

<table>
<thead>
<tr>
<th>Problem Size (# of Variables)</th>
<th>Estimates (E)</th>
<th>Features (F)</th>
<th>Human Fusers (H)</th>
<th>Time Periods (T)</th>
</tr>
</thead>
<tbody>
<tr>
<td>369</td>
<td>2</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>1,305</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>16,050</td>
<td>12</td>
<td>5</td>
<td>5</td>
<td>10</td>
</tr>
<tr>
<td>38,850</td>
<td>15</td>
<td>10</td>
<td>5</td>
<td>9</td>
</tr>
<tr>
<td>51,870</td>
<td>20</td>
<td>10</td>
<td>7</td>
<td>7</td>
</tr>
</tbody>
</table>

For each problem size, 30 different datasets were run through CIP with a mathematical model or heuristic. The reason behind the decrease in time periods for the largest two problem sizes is because if the number of time periods increased further, no incumbent solutions would be found when using the mathematical model. A few different metrics were averaged across the 30 different runs. First, the UB of the objective was recorded, along with the incumbent solution, which represents the best integer feasible solution that CPLEX has found at the point when the process was terminated. The solution times for both the mathematical model and heuristic were also recorded. The average over the 30 datasets of these values is shown in Table 4.0.6, along with the $P^{UB}$ metric. The $P^{UB}$ value represents the difference between the mathematical model or heuristic and the UB of the optimal solution. For only the smallest problem size did the mathematical model ever reach optimality, and that was also the only case in which the mathematical model outperformed the heuristic. But the mathematical model arrived at the solution after 824 seconds, whereas the heuristic arrived at a solution just 0.003% worse and over 150 times faster. For all other problem sizes, the heuristic outperformed the math model by
anywhere from 3% to 15%. It is interesting to note that as the problem size gets larger the distance from the UB does not get larger. This is due to the number of time periods remaining static for the last two problem sizes. The number of time periods greatly affects the overall model, because every time period interacts with every other time period during the solving process.

**Table 4.0.6 Learning and Forgetting Results**

<table>
<thead>
<tr>
<th>Problem Size</th>
<th>Model Time (s)</th>
<th>UB of Optimal</th>
<th>Incumbent Solution</th>
<th>Heuristic Time(s)</th>
<th>Heuristic Solution</th>
<th>$P_{UB}$</th>
<th>$p_{UB}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Model</td>
<td>Heuristic</td>
</tr>
<tr>
<td>369</td>
<td>824.68</td>
<td>15.3425</td>
<td>15.3425</td>
<td>4.88</td>
<td>15.342</td>
<td>0.00%</td>
<td>0.003%</td>
</tr>
<tr>
<td>1,305</td>
<td>3600.00</td>
<td>94.1500</td>
<td>65.6175</td>
<td>8.71</td>
<td>68.594</td>
<td>30.31%</td>
<td>27.140%</td>
</tr>
<tr>
<td>16,050</td>
<td>3600.00</td>
<td>883.2600</td>
<td>519.5000</td>
<td>10.60</td>
<td>564.943</td>
<td>41.18%</td>
<td>36.040%</td>
</tr>
<tr>
<td>38,850</td>
<td>3600.00</td>
<td>1405.3600</td>
<td>903.9400</td>
<td>35.38</td>
<td>1114.260</td>
<td>35.68%</td>
<td>20.710%</td>
</tr>
<tr>
<td>51,870</td>
<td>3600.00</td>
<td>1491.3900</td>
<td>1036.0300</td>
<td>89.24</td>
<td>1182.600</td>
<td>30.53%</td>
<td>20.700%</td>
</tr>
</tbody>
</table>

The smallest problem size (369 variables) is the only case in which the mathematical model outperformed the heuristic, so naturally the graph for this problem size is different than the rest (Figure 4.11). In the graph, the time axis is plotted on a logarithmic scale so as to more clearly illustrate the change in the incumbent solution. The first incumbent solution was found at around 2 seconds and CPLEX continued to find better integer feasible solutions until the 13 minute and 45 second mark when it reached the optimal solution. The heuristic solution was solved in 4.88 seconds and is less than 0.01% different from the optimal value. The green bar represents the heuristic objective value but also demonstrates at what time the math model became equal to or better than heuristic in terms of the objective value. In other words, the math model took 9.734 seconds to obtain the same objective value that the heuristic did in 4.88
seconds. This demonstrates that even for smaller problems, using the heuristic is just as beneficial in terms of yielding a good objective value faster.

For all other problem sizes, the heuristic outperformed the mathematical model’s incumbent solution. Figure 4.12 presents the results from the second problem size (1,305 variables). The heuristic retrieved a value of 68.594 after 8.71 seconds. Interestingly, an incumbent solution was also found after about 8 seconds of run time for the mathematical model with an objective of 63.8065. This clearly shows that if the mathematical model and heuristic are run for the same amount of time, a better solution will be obtained from the heuristic. Since the heuristic discovers a solution so quickly, the mathematical model provides no benefit as its objective value was 75% worse and only provided a solution 6 seconds sooner, even when it used the first incumbent solution, which was found after just 2 seconds.
Figure 4.12 Problem size 2 results (1,305 variables).

Figure 4.13, Figure 4.14 and Figure 4.15 show the average results for problem sizes with 16,050, 38,850 and 51,870 variables, respectively. In all of these cases and for all problem sizes larger than 1,305 variables, the heuristic yielded a solution 10 to 35 times faster than the CPLEX solver. Even after allowing the mathematical model to run for an hour, for all problems, the best integer solution found (final incumbent solution) was no better than the heuristic. This provides further evidence that using the presented heuristic trumps the mathematical model in all cases for all problem sizes.

To further compare the mathematical model to the heuristic, the time at which the first incumbent solution was found is plotted against the run time for the heuristic (Figure 4.16). For smaller problem sizes, run times were similar for both; but, as was explained previously, the objective values were close or the heuristic outperformed the mathematical model. Once the number of variables increases beyond 1,305 variables, the amount of time it took CPLEX to find the first integer feasible solution of the mathematical model increased dramatically. As the
number of variables increased further, these increases became more pronounced, whereas the heuristic still took minimal time for all problem sizes.

Figure 4.13  Problem size 3 results (16,050 variables).

Figure 4.14  Problem size 4 results (38,850 variables).
4.5 Summary

In this chapter, I introduced the concept of using multiple human fusers in CIP. In particular, I developed a formal methodology to incorporate multiple human fusers into the IF process. This involved developing a model that used the concept of learning and forgetting to
differentiate between different human fusers with varying levels of expertise. The learning and forgetting model, which is typically an NLP, is transformed to a linear formulation. Since solving the linear model for optimality is time consuming, I developed a heuristic methodology for generating feasible solutions. Through experimentation, I was able to show that the heuristic outperformed or only slightly underperformed the mathematical model in every case, and was much faster, with its longest run time being about a minute and a half for the largest problem size, whereas the mathematical model failed to provide an integer feasible solution until the 52\textsuperscript{nd} minute.

Data decay also added a new concept to the model and changed the model formulation slightly. The inclusion of the data decay constraints yielded virtually the same exact results when comparing objective values and solution times for the mathematical model and the heuristic. In the data decay model, the total certainty of all the estimates increases at each interval until it starts to fluctuate and level off. The leveling off of the total certainty is directly related to the size of the problem versus the number of human fusers available. If all of the human fusers can cover all of the F-E combinations over two time periods, then the system will max out when all the certainty values are 100\%. Otherwise, the final certainty values will hover around a level for the remaining time periods as the human fusers are assigned observations to maintain certainty at its maximum level.

Clearly, other methods for combining multiple human fusers must be explored (i.e., goal programming, etc.). The way in which the human fusers are brought together depends on the problem scenario. Even with the method of learning and forgetting, further test sets that increase the number of expertise levels are needed to make the mathematical model mimic how learning
and forgetting actually works. Further exploration of data decay is also warranted, specifically to formulate its function more accurately to determine how it actually works in the real world.
Chapter 5: Conclusion and Future Research

In Chapter 2, I presented a novel methodology to add a soft process to a system that typically includes only hard processes, even though the inputs into the system include both hard and soft sensors. In addition to presenting this new data flow for the system, I demonstrated through an example how the soft process would work with the hard processes that are already in place. Through defined metrics, I showed that the addition of a soft process (i.e., human fuser) can actually improve a DM’s knowledge about a problem, which should improve the accuracy of the COA that he or she determines. I show that over time our system converges towards optimality, meaning the knowledge of the DM improves. I also found that while the mathematical models outperforms the best heuristic (bang-for-buck) the amount of time increases dramatically after 500,000 variables and switching to a bang-for-buck heuristic allows a solution to be obtained much faster that is only 7-8% worse than the optimal solution.

In Chapter 3, I further explored CIP by testing it with more complex models. I presented a model that combines uncertainty values from the IF process using Shannon entropy. Since entropy is nonlinear, I developed a rule-based system in order to build a model that is linear so that it can be solved in a reasonable amount of time for larger problem sizes. Even so, the linear model began to require a large amount of time to obtain a feasible solution, so I also developed a heuristic that could be used in CIP. The heuristic relaxed the variables of the linear model and recursively added constraints to the relaxed linear model until all variables reached integer values. I found that the heuristic solved significantly quicker once the number of variables increased beyond 5,250; the heuristic solution was also within 2% of the linear model solution in all cases.
In Chapter 4, I introduced the concept of using multiple human fusers in CIP. In particular, I developed a formal methodology to incorporate multiple human fusers into the information fusion process. This involved developing a model that used the concept of learning and forgetting to differentiate between different human fusers with varying levels of expertise. The learning and forgetting model, which is typically an NLP, is transformed to a linear formulation. As solving the linear model for optimality is time consuming, I developed a heuristic methodology for generating feasible solutions. Through experimentation, I was able to show that the heuristic outperformed or only slightly underperformed the mathematical model in every case and was much quicker, with its longest run time being about a minute and a half for the largest problem size, whereas the math model failed to yield an integer feasible solution until the 52\textsuperscript{nd} minute. Data decay also added a new concept to the model and changed the model formulation slightly. The inclusion of the data decay constraints yielded virtually the same exact results when comparing the objective values and solution times of the mathematical model and the heuristic.

Future research can go in many different directions. It is most important to acquire real data that can be tested in CIP with the given models, observe how a human fuser would actually function, and be able to modify the models to account for how human fusers act. There are also many other models that can be developed for different problem scenarios. In particular, some of the other ways certainty values are combined (i.e., Bayesian theory) can be explored and implemented. There are also other methods to combine multiple human fusers that must be explored (i.e., goal programming, etc.). The way in which the human fusers are brought together depends on the problem scenario. Even with the method of learning and forgetting, further test sets that increase the number of expertise levels should be evaluated to mimic the mathematical
model and how learning and forgetting actually works. It is also important to further examine data decay and try to formulate its function more accurately in order to determine how it actually works in the real world.
REFERENCES


Nauss, R. M. (1976). An Efficient Algorithm for the 0-1 Knapsack Problem. (pp. 27-31). INFORMS.


Appendix A: Sample OPL Code for Math Models and Heuristics

/*********************************************
* OPL 12.2 Model
* Author: David Sudit
* Creation Date: Apr 27, 2011 at 2:19:29 PM
*********************************************/

{string} Estimates = ...;
{string} Features = ...;

int FEAT_EST_LIST [Estimates,Features] = ...;
float UNCERTVALS [Estimates,Features] = ...;
float PROCESSINGTIME [Estimates,Features] = ...;
float FEATWEIGHTS [Features] = ...;
int XY[Estimates,Features] = ...;
float GT[Estimates,Features] = ...;

dvar int X[Estimates][Features] in 0..1;
dvar int Y[Estimates][Features] in 0..1;
dvar float C[Estimates];

maximize
(sum( e in Estimates, f in Features : UNCERTVALS[e][f] == 0)(X[e][f])) +
(sum( e in Estimates, f in Features : UNCERTVALS[e][f] > 0)(Y[e][f]*(1-UNCERTVALS[e][f])));

subject to {

ProcessingTime:
(sum( e in Estimates, f in Features : UNCERTVALS[e][f] == 0)(PROCESSINGTIME[e][f]*X[e][f])) +
(sum( e in Estimates, f in Features : UNCERTVALS[e][f] > 0)((PROCESSINGTIME[e][f]/2)*Y[e][f])) <= 15;

};

main{

thisOplModel.generate();
var MODEL1 = thisOplModel;
var M1OUTPUT = new IloOplOutputFile();
M1OUTPUT.open("Model1_Output[GT=1].txt");
for (var i = 1; i <= 20; i++){

}
cplex.solve();
var XO = MODEL1.X;
var YO = MODEL1.Y;

M1OUTPUT.writeln("Iteration := ",i);
M1OUTPUT.writeln();
M1OUTPUT.writeln("X");
M1OUTPUT.writeln();
M1OUTPUT.writeln(MODEL1.X);
M1OUTPUT.writeln("Y");
M1OUTPUT.writeln();
M1OUTPUT.writeln(MODEL1.Y);

var def = MODEL1.modelDefinition;
var data = MODEL1.dataElements;

for (var a in thisOplModel.Estimates){
  for (var b in thisOplModel.Features){

    data.XY[a][b] = MODEL1.XY[a][b] + MODEL1.X[a][b] + MODEL1.Y[a][b];
    data.FEAT_EST_LIST[a][b] = data.FEAT_EST_LIST[a][b] + MODEL1.X[a][b];
    if (XO[a][b] + YO[a][b] == 1){
      data.UNCERTVALS[a][b] = MODEL1.GT[a][b];
    }
  }
}
MODEL1 = new IloOplModel(def,cplex);
MODEL1.addDataSource(data);
MODEL1.generate();
}
M1OUTPUT.close();
0;
}
float PROCESSINGTIME [Estimates,Features] = ...;
float FEATWEIGHTS [Features] = ...;
int XY[Estimates,Features] = ...;
float GT[Estimates,Features] = ...;

dvar boolean X[Estimates][Features];
dvar boolean Y[Estimates][Features];
dvar float C[Estimates];
dvar float R[Estimates];

maximize
 (sum( e in Estimates ) ( C[e] + R[e] ));

subject to {
   forall ( e in Estimates )
      EstimateWeight:
         sum( f in Features )
            (FEATWEIGHTS[f]*(X[e][f] + FEAT_EST_LIST[e][f])) == R[e];

   forall( e in Estimates )
      EstimateCalc:
         sum( f in Features )
            (X[e][f] + UNCERTVALS[e][f] + Y[e][f]*(1 - UNCERTVALS[e][f]))*(1/3) == C[e];

   ProcessingTime:
      sum( e in Estimates, f in Features )
         (X[e][f]*PROCESSINGTIME[e][f] + Y[e][f]*(PROCESSINGTIME[e][f]/2)) <= 15;

   forall( e in Estimates, f in Features ){
      NewOrOld:
         X[e][f] + Y[e][f] <= 1;
   };

   forall( e in Estimates, f in Features ){
      Rereviewed:
         Y[e][f] <= FEAT_EST_LIST[e][f];
   };

   forall( e in Estimates, f in Features ){
      Newreview:
         X[e][f] <= 1 - FEAT_EST_LIST[e][f];
   };

}
forall( e in Estimates, f in Features){
    CheckOnce:
    X[e][f] + Y[e][f] <= 1 - XY[e][f];
};

main{

    thisOplModel.generate();
    var MODEL2 = thisOplModel;
    var M2OUTPUT = new IloOplOutputFile();
    M2OUTPUT.open("Model2_Output[GT=1].txt");
    for (var i = 1; i <= 20; i++){
        cplex.solve();
        var XO = MODEL2.X;
        var YO = MODEL2.Y;

        M2OUTPUT.writeln("Iteration := ",i);
        M2OUTPUT.writeln();
        M2OUTPUT.writeln("X");
        M2OUTPUT.writeln();
        M2OUTPUT.writeln(MODEL2.X);
        M2OUTPUT.writeln("Y");
        M2OUTPUT.writeln();
        M2OUTPUT.writeln(MODEL2.Y);

        var def = MODEL2.modelDefinition;
        var data = MODEL2.dataElements;

        for (var a in thisOplModel.Estimates){
            for (var b in thisOplModel.Features){
                data.XY[a][b] = MODEL2.XY[a][b] + MODEL2.X[a][b] + MODEL2.Y[a][b];
                data.FEAT_EST_LIST[a][b] = data.FEAT_EST_LIST[a][b] + MODEL2.X[a][b];
                if (XO[a][b] + YO[a][b] == 1){
                    data.UNCERTVALS[a][b] = MODEL2.GT[a][b];
                }
            }
        }
        MODEL2 = new IloOplModel(def,cplex);
        MODEL2.addDataSource(data);
        MODEL2.generate();
    }
    M2OUTPUT.close();
    0;
};
Appendix B: Expansion of Decision Tree