A COLLECTION OF MULTI-CRITERIA DECISION ANALYSES FOR A CHILDBIRTH AFTER CESAREAN DECISION USING TWO DECISION METHODOLOGIES- THE ANALYTIC HIERARCHY PROCESS (AHP) AND DECISION TREES

by

Poonam Sharma

A THESIS

Presented to the Department of Biomedical Informatics & Clinical

Epidemiology and the Oregon Health & Science University

School of Medicine

in partial fulfillment of

the requirements for the degree of

Master of Science

June 2007

School of Medicine

Oregon Health & Science University

Certificate of Approval

This is to certify that the Master's Thesis of

<u>Poonam Sharma</u>

"A collection of multi-criteria decision analyses for a childbirth after cesarean decision using two decision methodologies- the Analytic Hierarchy Process (AHP) and decision trees"

Has been approved

Dr. Karen Eden [redacted]
Thesis Advisor

Dr. Jim Dolan [redacted]
Committee Member

Dr. Jeanne-Marie Guis [redacted]
Committee Member

Dr. Holly Jimison [redacted]
Committee Member

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Karen Eden for her support, advice and teachings throughout the entire duration of this thesis. I also want to thank my wonderful committee members, Dr. Jim Dolan, Dr. Jeanne-Marie Guise, and Dr. Holly Jimison for their abudance of knowledge, expertise and support in bringing together the many facets in medicine and medical decision making that comprised this thesis.

I want to extend my heartfelt gratitude to my fellow classmates and professors for their tremendous support and patience throughout the thesis, especially when putting their work on hold to help facilitate the completion of my thesis. Finally, I want to thank my parents for always instilling in me the importance of education, and to always value and respect it through honesty, patience, and hard work. I would lke to thank my sisters and brother-in-law for their faith and trust in my life pursuits. Lastly, I want to thank my husband for his devoted love, attention and faith not only for this thesis but for all I want to pursue in the future.

Thanks once again everyone!!

ABSTRACT

This thesis presents three decision models for a child-birth after cesarean decision. The first decision model used an AHP approach, the second used a decision tree approach, and the third model used a hybrid AHP-decision tree approach. The AHP model assessed medical risk subjectively while the decision tree and hybrid models assessed medical risk objectively. Decision criteria included both maternal and neonatal outcomes. Maternal outcomes included hysterectomy, numbness/pain near incision, incontinence, and placental abnormalities causing a risk to future pregnancies. Neonatal outcomes included disability and death. Data on 96 women with a prior cesarean were used from a partnering studying that used the AHP decision aid tool. Utilities for the decision tree were derived from a normalization method of AHP criteria weights. Various sensitivity analyses revealed the decision tree was sensitive to all probabilities of maternal outcomes except for hysterectomy. The decision models revealed that the mode of risk assessment plays a big role in determining the decision. Multiple decision models using subjective and objective risk assessments can play an important role in the clinician-patient shared decision making process.

1. Introduction

The question of whether a woman should attempt a vaginal delivery or an elective repeat cesarean after a prior cesarean section has been an area of active research. There is much conflicting evidence as to which birthing strategy is better. Vaginal birth after cesarean (VBAC) is generally associated with shorter hospital stays for the mother and infant, reduced rates of infection, and thromboembolism compared to cesarean. [Guise 04, Guise03] However a failed VBAC is associated with higher rates of infection and hemorrhage, increased risk of symptomatic uterine rupture, and increased neonatal morbidity. [Guise04, Guise 03] Cesarean is more predictable in scheduling and is associated with reduced risk of symptomatic uterine rupture. However, there are risks of operative injury, scarring, and increased risk of placental abnormalities in future pregnancies. Thus, the advantage of VBAC is that when it leads to successful vaginal delivery there is less probability of maternal complications, and the associated health-care costs are also lower. [ACOG99] Regarding both the strategies the risks are similar for asymptomatic uterine rupture, hysterectomy, and maternal death [Guise04, Guise03].

Since cesarean rates continue to increase, the number of women facing the complicated decision of whether to pursue trial of labor or repeat cesarean after a prior cesarean will also continue to increase. Since a child-birth after cesarean decision is sensitive to preference variations, understanding patient preferences is also essential to choosing the best birthing strategy. [Eden04] In a recent systematic literature review performed by Eden et al., it was found that the choice of strategy was related to several patient factors, such as desire for vaginal delivery, previous vaginal delivery, avoidance of labor and feelings about previous cesarean delivery. [Eden04] For example, some of the reasons why women prefer a trial of labor are an easier recovery and being able to quickly return to the care of their children. On the other hand, women who prefer a repeat cesarean may do so because of fear of a long and painful vaginal delivery.

Because of the huge impact this type of decision has on a large number of women, the primary goal of this thesis was to investigate and develop decision methodologies which incorporated **risk information and patient preferences**. No study has been done which incorporates these in one decision aid tool. A literature review was performed in several different areas which include:

- Literature review on decision methodologies in general
- Literature review on various childbirth after cesarean decision models done in the past
- Literature review on maternal and infant risks and outcomes for a trial of labor or repeat cesarean
- Literature review on patient preferences in a childbirth after cesarean decision

In the next section, summaries of these literature reviews will be presented. This will be followed by the motivations for this work, the methods employed, and the results.

2. Literature review

Background on various decision methodologies

This section presents a literature review on some decision methodologies which include the Analytic Hierarchy Process (AHP), decision trees, and Markov models.

1. The Analytic Hierarchy Process (AHP)

AHP establishes priorities for criteria in multi-criteria decision making. One starts with describing the problem (the goal), next proceeding logically to criteria (and possibly sub-criteria) in terms of which outcomes are evaluated. This results in a hierarchic structure consisting of various levels of major criteria and sub-criteria. In order to ascertain the importance of the various criteria, pairwise comparisons are made, by providing the decision maker with verbal statements about the importance of one criterion over another. This generates overall weights for each criterion, represented numerically on an absolute scale. Thus the weights of major criteria are on a numerical scale from 0 to 1 and all weights sum up to 1. The same is true for the sub-criteria. When these matrices of weights are established, one evaluates the outcomes/options available for each criterion in series of pairwise comparisons to generate a separate matrix. Finally, the criteria weight matrix and option weight matrix are multipled to assess the global measure of priority for each option when making the final decision.these matrices are then one proceeds to synthesize the local priorities to derive a global measure of priority used in making the final decision. [Saaty98]

2. <u>Decision Trees</u>

A decision tree is a visual representation of all the possible options and the consequences that may follow each option. The representation consists of branches and nodes. A node can be thought of as a junction which can either be a square or a circle. A square represents a decision node from which all possible health strategies "branch" out. A circle is a chance node from which all risks and outcomes can be described. From the strategy/option branches, subsequent nodes (circles) and branches follow describing risks and outcomes for each option. Each branch in the decision tree eventually ends at an outcome for which a utility is assigned.

Utilities are defined as the quantitative measurements of the strength of a person's preference for an outcome incorporating risk. Standard utilities range from a scale of 0-1 where 0 marks the least preferred, and 1 mark the most preferred outcome. More specifically, utilities reflect the patient's attitudes towards the quality of life for health states. [Hunink01]

There are three critical components to a utility value:

- It should encompass all aspects of the state of health being assessed
- It must be measured on a ratio scale, between extremes of perfect health and death; thus if perfect health is 1 and death is 0, then a value of 0.5 is exactly half as desirable as perfect health
- It must use length of life as the units scale for measuring the subject's preference for the quality of life in a given health outcome

[Hunink01]

There are multiple methods of generating utilities, which include the standard reference gamble, the time trade off method, and the visual analog scale. [Hunink01] [Sox88]

The visual analog scale represents a simple approache to utility assessment. Of all the utility assessments, this is the easiest to administer, however the values generated are not true utilities, since it isn't a ratio scale between perfect health and death. There are many proposed ways of converting rating scale scores to true utilities. One strategy that is used was defined by Torrance et. al (1996) where the authors derived the following relationship between rating scale values and standard gamble utilities: $utility = 1-(1-value)^r$, where r is estimated to be in the range 1.6-2.3. [Hunink01]

The standard reference gamble method was devised by Oskar Morgenstern and John von Neumann (1944), which assigns a utility for a particular health state by asking how high a risk of death one would accept to improve it. One chooses between life in a given health state and a gamble between death and perfect health. The utility is the probability of perfect health in the gamble such that the user is indifferent between the gamble and the certain intermediate outcome.

One only knows the probability of the health states that can occur. Thus the utility reflects one's preferences about life in that state of health and one's attitudes toward risk. [Sox88] [Hunink01]

The time trade-off method assesses a utility for a particular health state by asking a person how much time he/she would give up to improve it. This is accomplished by choosing between a set length of life in a less than perfect health state and a shorter length of life in perfect health. The utility is then given by ratio of the shorter to the longer life expectancy at which the respondent finds the two health states equally desirable. [Hunink01]

3. Markov Models

The difference between traditional decision trees and Markov models is that traditional decision trees model uncertain events at chance nodes, while Markov models represent uncertain events as transitions between defined health states. [Hunink01] Markov modeling defines a set of transitions between a set of states, where a transition is the probability of going from one state to a next state. Alternatively it can model a series of events with a finite number of outcomes, where the outcomes represent health stages when making a medical decision, and the transition rates are placed in a matrix representing the transitions from one state to another. [Carter99]

There are many advantages of using Markov modeling because it allows more freedom in moving between states. For example, it could be used to evaluate future pregnancies because the model would allow for one to go back to state of "Pregnant Woman." Finally, cost can be incorporated along the arrows going from one state to another, describing how much it costs to go from one state to another. This is advantageous because cost can be computed for various scenarios and can be ranked based on patient choices and preferences.

4. Converting AHP preference weights to utilities

Vargas presents a normalization method of converting AHP preference weights into utilities, by normalizing a set of criteria weights relative to one another. The paper presents an extensive mathematical background and set of proofs to present this conversion. The details of this conversion can be found in Appendix A.

[Vargas86]

Background on maternal and neonatal risks for a childbirth after cesarean decision

This section highlights some key papers which have done extensive reviews on the associated maternal and neonatal risks for a child-birth after cesarean decision.

1. A literature review assessing safety of VBAC by Guise et. al

This systematic literature review by Guise et. al presented several studies to assess the benefits and risks of a trial of labor and a repeat cesarean for women with a previous cesarean [Guise04]. Maternal complications studied included excessive bleeding (requiring transfusion or a hysterectomy), maternal infection, wound infection, uterine rupture, incontinence, and maternal death. Neonatal complications studied included infection, respiratory and neurologic sequelae, and death. The proportion of women attempting a trial of labor ranged from 60%-82%.

This review identified two studies which clearly assessed risks of infection regarding the two modes of delivery. In one study, the incidence of infection for women who had attempted a trial of labor was 5.3% while for women who had a repeat cesarean the rate was 6.4%. Within the trial of labor group, the rate of infection for women having a successful trial of labor was 3.5% and for women who had a failed trial of labor (subsequently had a cesarean) was 8%. In the other study, the incidence of infection for women who had attempted a trial of labor was 6.79% and for women who had a repeat cesarean the rate was 9.73%.

Regarding rates of maternal death and hysterectomy, there was no significant difference between the two modes of delivery; however uterine rupture was more frequent in the group attempting a trial of labor. One of the main conclusions of the authors was the lack of literature studying the safety of VBAC and the great need of identifying high-risk and low-risk groups and circumstances for morbidity.
[Guise04]

2. **<u>A study assessing effects of maternal age on trial of labor by Bujold et. al</u>**

The purpose of this study was to study the effect of maternal age on the rate of successful trial of labor and the rate of uterine rupture for patients attempting a trial of labor after a prior cesarean. [Bujold04] This cohort study of 2493 patients, were divided into three groups according to age: less than 30 years old, 30-34 years old, and 35 years or older. For analysis, women with no prior vaginal delivery and women with at least 1 prior vaginal delivery were analyzed separately. Between these two groups, the rate of uterine rupture was similar, and successful trial of labor was inversely related to age. In addition, patients who were 35 years old or greater had lower rates of a successful trial of labor in both groups of women. However, no association between the maternal age and the risk of uterine rupture was found.
[Bujold04]

3. **<u>An observational study assessing risks associated with VBAC by Landon et. al</u>**

The authors of this paper conducted a four-year observational study at several academic centers of women with a prior cesarean. Maternal and perinatal outcomes were compared for women who underwent both modes of delivery for a subsequent pregnancy. Trial of labor was attempted by 17,898 women and 15,801 women had a repeat cesarean. Symptomatic uterine rupture was .7% for women undergoing trial of labor. There was 0% incidence of hypoxic-ischemic encephalopathy in infants whose mothers had a repeat cesarean and there were 12 cases of infants with this complication whose mothers attempted trial of labor. The rate of endometritis was higher in women attempting a trial of labor (2.9%) than for women who had a repeat cesarean (1.8%). The rate of blood transfusion for women attempting trial of labor was 1.7% while for women having a repeat cesarean the rate was 1.0%. The frequency of hysterectomy (.2% vs. .3%) and maternal death (.02% vs. .04%) did not differ significantly between the two groups.

The authors concluded that a trial of labor was associated with greater perinatal risk, however the actual risks were very low. Maternal endometritis and transfusion were both significantly higher in a trial of labor. They also noted that women should be educated on perinatal morbidity and mortality resulting from uterine rupture, however it is not clear from the literature how often uterine rupture directly results in infant death. The overall rate of rupture-related perinantal death was .11 per 1000 trials of labor. The authors also showed that the risk of uterine rupture increased with the induction of labor.
[Landon04]

4. A combined list of risk probabilities from several other sources

In this section, risk probabilities were combined from several other papers into multiple tables. The probabilities of occurrence of were extracted from the raw data given in the literature. The raw data from the literature survey is tabulated in Table 1 through Table 7 below. In these tables the numbers in brackets are the probabilities of occurrence as reported in a particular literature source.

Table 1: Rate of Success in Women Attempting Trial of Labor

Outcome of Trial of Labor	[Landon04]	[Loebel04]	[Mozurkewich00]	[Rageth99]	[McMahon96]	[Wen04]
Successful	13,139	749	20,746	12,986	1963	92455
Failed	4759	178	8067	4627	1287	36505
Failure Probability	0.27	0.2	0.28	0.27	0.4	0.28

Table 2: Maternal Complications for Elective Repeat Cesarean

Outcome	[Landon04] (Out of 15,801)	[Loebel04] (Out of 481)	[Rageth99] (Out of 11,433)	[McMohan96] (Out of 2889)	[Wen04] (Out of 179,795)
Infection	285 (0.02)	11 (0.02)	-	-	837 (0.005)
Operative Injuries	-	2 (0.004)	-	18 (0.006)	-
Uterine Rupture	0	0	22 (0.002)	1 (0)	453 (0.003)
Blood Transfusion	158 (0.001)	3 (0.006)	-	39 (0.01)	268 (0.001)
Maternal Death	7 (0)	-	-	-	10 (0)
Hysterectomy	47 (0.003)	-	52 (0.005)	6 (0.002)	140 (0)

Table 3: Maternal Complications for Successful Trial of Labor

Outcome	[Landon04] (Out of 13,139)	[Loebel04] (Out of 749)	[Rageth99] (out of 12,986)	[McMohan96] (out of 1962)	[Wen04] (Out of 92,455)
Infection	152 (0.01)	14 (0.02)	-	-	177 (0.002)
Operative Injuries	-	0	-	2 (0.001)	-
Uterine Rupture	14 (0.001)	0	18 (0.001)	2 (0.001)	129 (0.001)
Blood Transfusion	152 (0.01)	7 (0.009)	-	18 (0.009)	171 (0.002)
Maternal Death	1 (0)	-	-	-	0
Hysterectomy	19 (0.001)	-	9 (0)	1 (0)	64 (0)

Table 4: Maternal Complications for Failed Trial of Labor

Outcome	[Landon04] (Out of 4,759)	[Loebel04] (Out of 178)	[Rageth99] (Out of 4627)	[McMohan96] (Out of 1287)	[Wen04] (Out of 36,505)
Infection	365 (.08)	9 (.05)	-	-	310 (.008)
Operative Injuries	-	4 (.02)	-	39 (.03)	-
Uterine Rupture	110 (.02)	4 (.02)	52 (.01)	8 (.006)	714 (.02)
Blood Transfusion	152 (.03)	5 (.03)	-	18 (.013)	74 (.002)
Maternal Death	2 (0)	-	-	-	2 (0)
Hysterectomy	22 (.004)	-	20 (.004)	4 (.003)	63 (.001)

Table 5: Complications in Infant for Elective Repeat Cesarean Section

Outcome	[Loebel04] (Out of 481)	[Landon04] (out of 15,014)	[Rageth99] (Out of 11,433)	[Richard05] (Out of 843)	[Hook97] (Out of 497)
Neo-natal Death	1 (.002)	7	10 (0)	0 (0)	-
NICU admission	27 (.06)	-	949 (.08)	70 (.08)	10 (.02)
Suspected Sepsis	17 (.03)	-	-	-	9 (.02)
Respiratory Complications	19 (.04)	-	-	6 (.007)	35 (.07)

Table 6: Complications in Infant for Successful Trial of Labor

Outcome	[Loebel04] (Out of 749)	[Landon04]	[Rageth99] (Out of 12,986)	[Hook97] (Out of 336)
Neo-natal Death	0 (0)		59 (.005)	-
NICU admission	28 (.03)		660 (.05)	6 (.02)
Suspected Sepsis	16 (.02)		-	8 (.02)
Respiratory Complications	10 (01)		-	14 (.04)

Table 7: Complications in Infant for Failed Trial of Labor

Outcome	[Loebel04] (Out of 178)	[Landon04]	[Rageth99] (Out of 4627)	[Hook97] (Out of 156)
Neo-natal Death	1 (.005)		27 (.006)	-
NICU admission	11 (.06)		415 (.09)	11 (.07)
Suspected Sepsis	9 (.05)		-	18 (.11)
Respiratory Complications	8 (.04)		-	12 (.07)

It is clear from the data in the tables above that different literature sources have different sets of data on the various outcomes. In order to combine the data from these different sources, the probability of an outcome was estimated from the raw data by grouping all the literature sources that presented data for that particular outcome and then dividing the total number of reported cases by total number of cases surveyed. As an example when estimating the probability of NICU admissions for repeat cesarean deliveries, four different literature sources reported data on this outcome [Loebel04, Rageth99, Richard05, and Hook97]. So the total number of cases reported for this outcome is 27+949+70+10 = 1056 out of a total of 481+11433+843+497 = 13254 studied cases. Hence, the probability of NICU admission for elective repeat cesarean surgery will be estimated as 1056/13254 = 0.08. Similar calculations for all the probabilities of various outcomes are detailed in Table 8 through Table 13.

Table 8: Probabilities of Maternal Outcomes for Elective Repeat Cesarean

Outcome	Total Number of Reported Cases	Out of a Total Number of Studied Cases	Probability	Probability Range
Infection	1133	196,077	0.0058	(0.02-0.005)
Operative Injuries	20	3370	0.006	(0.004-0.006)
Uterine Rupture	476	210,339	0.0023	(0-0.003)
Blood Transfusion	468	198,966	0.0023	(0.001-0.01)
Hysterectomy	245	209,918	0.0012	(0-0.005)
Total	2342	-	0.0175	-

Table 9: Probabilities of Maternal Outcomes for Successful Trial of Labor

Outcome	Total Number of Reported Cases	Out of a Total Number of Studied Cases	Probability	Probability Range
Infection	343	106,343	0.0032	(0.002-0.02)
Operative Injuries	2	2711	0	
Uterine Rupture	163	121,273	0.0013	
Blood Transfusion	348	108,305	0.0033	
Hysterectomy	93	120,542	0	
Total	949	-	0.0078	-

Table 10: Probabilities of Maternal Outcomes for Failed Trial of Labor

Outcome	Total Number of Reported Cases	Out of a Total Number of Studied Cases	Probability	Probability Range
Infection	684	41,442	0.016	
Operative Injuries	43	1465	0.03	
Uterine Rupture	888	47,356	0.019	
Blood Transfusion	249	42,729	0.0058	
Hysterectomy	109	47,178	0.0023	
Total	1973	-	0.0731	-

Table 11: Probabilities of Infant Outcomes for Elective Repeat Cesarean

Outcome	Total Number of Reported Cases	Out of a Total Number of Studied Cases	Probability	Probability Range
Neo-Natal Death	18	27,771	0	
NICU admission	1056	13,254	0.08	
Suspected Sepsis	26	978	0.026	
Respiratory Complications	60	1821	0.033	
Total	1160	-	0.139	-

Table 12: Probabilities of Infant Outcomes for Successful Trial of Labor

Outcome	Total Number of Reported Cases	Out of a Total Number of Studied Cases	Probability	Probability Range
Neo-Natal Death	59	13,735	0.0043	
NICU admission	694	14,071	0.05	
Suspected Sepsis	24	1085	0.022	
Respiratory Complications	24	1085	0.022	
Total	801	-	0.0983	-

Table 13: Probabilities of Infant Outcomes for Failed Trial of Labor

Outcome	Total Number of Reported Cases	Out of a Total Number of Studied Cases	Probability	Probability Range
Neo-Natal Death	28	4805	0.006	
NICU admission	437	4961	0.088	
Suspected Sepsis	27	334	0.08	
Respiratory Complications	20	334	0.06	
Total	512	-	0.234	-

Background on childbirth after cesarean decision models

This section presents previous decision models for a child-birth after cesarean decision. Each subsection in this section represents a single decision model.

1. A Decision tree by Mankuta et.al

Mankuta et.al, constructed a decision tree based on the reported risks of trial of labor and an elective repeat cesarean. [Mankuta03] The goal was to analyze the decision between choosing a trial of labor or a repeat cesarean after one previous cesarean, including a desire for an additional third pregnancy. For analyzing the future pregnancy, placental complications were taken into account after a delivery by cesarean. Sensitivity analyses were done on probabilities such as uterine rupture, neonatal death, emergency cesarean, and desire for a future pregnancy. Sensitivity analyses were also performed on utilities such hysterectomy and neonatal death.

In the decision tree, the branch for a third future pregnancy was divided into two branches- "normal placenta" and "abnormal placenta". Abnormal placenta was defined as having placenta previa or placenta accreta. The "normal placenta" branch was then further divided into two outcomes- "normal delivery" and "neonatal death". The "abnormal placenta" branch was divided into four outcomes- "normal delivery", "neonatal death", "hysterectomy", and "maternal death".

Utilities were chosen between 0 and 1, where higher values were assigned for delivery of one or two infants alive and lower values were assigned for bad outcomes such as neonatal death, hysterectomy and maternal death. Utilities were assigned by medical experts using the Visual Analog Scale (VAS) method. The model was also analyzed with utilities obtained with the following formula: $TTO = 1 - (1 - VAS)$, where TTO represents utilities obtained from the Time Trade Off (TTO) method.

The authors found that an elective repeat cesarean was the better option only if the probability of an additional pregnancy and the risks from having an abnormal placenta were low. However trial of labor was the better option if the desire for an additional pregnancy was around 10-20%. If the success of the trial of labor was low, then an elective repeat cesarean was the better option regardless of how high the desire was for an additional pregnancy.

The model only included one additional pregnancy, and used only irreversible outcomes. The analysis did not include morbidity and costs and no distinction between spontaneous and induced labor was made because very little data was available on this topic.
[Mankuta03]

2. **A Decision tree and cost analysis by Chuang et. Al**

Chuang et.al, constructed a decision tree to analyze the decision between a trial of labor and repeat cesarean and to perform a cost analysis. [Chuang99] The probabilities were obtained from the literature and disutilities were assigned by the medical expertise of the authors. Complications for the mother were divided into two groups- major and minor. Major complications included hysterectomy, uterine rupture, and operative injury. Minor complications included puerperal fever, a blood transfusion, and infection. Maternal death was not considered in the model because maternal mortality rates are extremely low for both options, and data was not available for the rate of maternal deaths in successful vs. failed trial of labor groups.

In the analysis, disutilities of the procedures and morbidity were also explored. It was found in this decision model that the repeat cesarean was the better option. One-way sensitivity analyses were performed, and it was found that the probabilities and the disutilities of morbidity were insensitive. Other sensitivity analyses showed that the decision was sensitive to the patient's preference for a repeat cesarean, a successful trial of labor of failed trial of labor. The cost analysis showed that if the success rate of trial of labor was greater than .70, it was the less costly and better option. The authors' conclusion was that the best birthing option also depended on patient preferences.

[Chuang99]

3. <u>A decision model examining cost-effectiveness by Chung et. Al</u>

In this decision analysis, the goal of the study was to determine which birthing option, trial of labor or repeat cesarean was the more cost-effective method from society's perspective. The model included both maternal and neonatal outcomes and costs. This model used data from peer-reviewed studies, hospital costs, and utilities measured in QALYs. For the analysis, standard incremental cost-effectiveness methods were used, and sensitivity analyses were performed.

Maternal complications included uterine rupture, severe hemorrhage, operative injury, infection, urinary incontinence and fecal incontinence. Maternal outcomes were "well", "well after hysterectomy" and "death". For each maternal outcome, four neonatal outcomes were possible: none/mild morbidity, moderate morbidity (infection, respiratory distress), severe morbidity (permanent neurologic injury), and mortality (infants who died 30 days within delivery). Utilities for maternal outcomes were 1 for well health, .963 for well after hysterectomy, and 0 for death. Utilities for infant outcomes were 1 for none/mild morbidity and moderate morbidity, .6 for severe morbidity and 0 for death. Below is a table describing the rates of incidence of some of the associated risks with a child-birth after cesarean decision. These values were collected through a literature review done by the authors.

Table 14: Rates of maternal risks (Chung01)

	Successful trial of labor		Failed trial of labor leading to cesarean		Repeat cesarean	
	%	Range	%	Range	%	range
Maternal infection	3.4	3.4-3.47	19	11.3-27.1	7.5	2.3-17.3
Hemorrhage (excessive bleeding)	1.2	.92-5.94	2.1	1.4-14.8	2.4	1.4-10.4
Urinary incontinence	2	1.0-4.4	.6	0-2.0	0	0
Uterine rupture	.047	0-1.22	1.9	.6-4.4	.085	0-.36
Hysterectomy	.02	0-1.22	.4	0-.47	.39	0-.62

The authors noted that literature on rates of risks of neonatal outcomes was scarce, and used data from peer-reviewed articles and estimated probabilities of neonatal outcomes. The table below shows the estimated risks for the neonatal outcomes:

Table 15: Rates of neonatal outcomes for maternal risks

	% Incidence			
	None/mild neonatal morbidity	**Moderate neonatal morbidity**	**Severe neonatal morbidity**	**Neonatal morbidity**
Successful trial of labor				
Uncomplicated	95 (90-100)	5 (1-10)	<1 (0-1)	<1 (0-1)
Maternal infection	0	94 (90-100)	5 (0-10)	1 (0-2)
Failed trial of labor, leading to cesarean				
Uncomplicated	80 (75-90)	15 (10-22)	4 (0-10)	1 (0-2)
Maternal infection	0	94 (90-100)	5 (0-10)	1 (0-2)
Maternal hemorrhage	28 (15-40)	50 (35-65)	20 (10-30)	2 (1-5)
Uterine rupture	0	33 (16-40)	33 (5-40)	33 (6-46)
Repeat Cesarean				
Uncomplicated	94 (90-100)	5 (1-9)	1 (0-2)	<1 (0-1)
Maternal infection	0	94 (90-100)	5 (0-10)	1 (0-2)
Maternal hemorrhage	28 (15-40)	50 (33-65)	20 (10-30)	2 (1-5)
Uterine rupture	0	33 (16-40)	33 (5-40)	33 (6-46)

The model was sensitive to the success rate of a trial of labor. At the base case rate of 75% successful of trial of labor, it was found that trial of labor was more cost effective. Thus if the probability was greater than 74%, trial of labor was more cost-effective, and if the probability was less than 65%, then repeat cesarean was more cost-effective. [Chung01]

4. **A decision model examing cost-effectiveness of trial of labor by Grobman et. al**

The goal of this study was to examine the health effects and cost outcomes of a trial of labor versus a repeat cesarean from the point of view of the medical system and third-party payers. The authors constructed a decision tree and a Markov model to study a hypothetical cohort of 100,000 pregnant women who had a previous low transverse cesarean section. The main outcomes for this model included maternal and neonatal morbidity and mortality, total costs to the health care system, and cost per major neonatal complication avoided (death or permanent neurologic damage).

The authors constructed a decision tree, in which the initial decision is whether to choose a repeat cesarean or trial of labor. Then the decision tree was extended with the probabilities of risks and costs associated. After that a Markov model was constructed to extend the analysis beyond a second pregnancy and follow the 100,000 women through the entirety of their reproductive lives. The Markov model included the two birthing strategies along with the health stages (in case the reproductive states) of "subsequent pregnancy" and "no further childbearing". The two birthing strategies could go to either of the two health states. In addition, a woman in the health state "subsequent pregnancy" could go back to the state of "cesarean delivery" or "vaginal delivery".

In the Markov model, after a delivery, a woman has a choice of whether to have a further pregnancy. If the woman chooses to have a further pregnancy, she has a subsequent delivery and the related risks are dependent on the previous childbirth experience. Once childbearing is entirely completed, the model calculates the cumulative probabilities for the outcomes of interest. The probability and cost values were extracted from published data or expert opinion.

The authors found that a repeat cesarean delivery incurred heavier consequences with additional cesarean deliveries, higher maternal morbid events and an additional $179 million. In order to prevent one major adverse neonatal outcome required an additional 1591 cesareans and $2.4 million.

[Grobman00]

5. <u>A decision model assessing future pregnancies by Pare et. Al</u>

This paper presents two decision models to study the downstream maternal implications involved with a child-birth after cesarean decision. The key was to compare the immediate risks of a VBAC attempt, which is mostly uterine rupture against the downstream risks of multiple repeat cesareans, which is mostly placenta accreta. The decision between a trial of labor and an elective repeat cesarean would depend on how many future pregnancies a woman may have- having one additional pregnancy or having multiple future pregnancies. Thus two, decision trees were constructed. The first tree applies to women planning only more future pregnancy, and the second tree applies to women planning two more future pregnancies. Probabilities that were included in the trees were VBAC success rates, risk of uterine rupture, placenta praevia, placenta accreta and hysterectomy. These values were taken from published literature.

Cost and patient preferences, other maternal outcomes such as operative injury, blood transfusion and infertility, and neonatal outcomes were not included in the models. Assumptions for these models included the following: the target population was women with one prior low transverse caesarean who were eligible for a trial of labor, and the risks of uterine rupture and placenta accreta were independent of each other. For the second model, it was assumed that the only additional maternal risk with multiple cesareans was the increased risk of placenta praevia and placenta accreta. Other potential risks such as increased bleeding, transfusion and operative injury were not included.

The authors collected a huge amount of probabilities for the two decision models. Some of these probabilities will be presented in the Methods section of this thesis. However, it is noted here that in the base-case analysis a rate of 70% successful VBAC was used, and the probability was varied from 50-90%.

The authors found that for women planning a single additional pregnancy, a repeat cesarean is better than a trial of labor. However if a women is planning two additional pregnancies then a trial of labor is the more preferred option since the downstream risks of multiple cesareans are higher than for multiple VBACs. In other words, the increase in hysterectomies performed due to placenta accreta with the repeat cesarean section outweighs the increase in hysterectomies performed due to uterine ruptures with a trial of labor.

The authors concluded that the long term reproductive effects of multiple cesarean sections should be examined with making a VBAC decision and noted that patient preferences should also be considered when making a child-birth after cesarean decision.

[Pare06]

Background on patient preferences and patient's role in medical decision making

This section presents a literature review related to women's attitudes and preferences to a child-birth after cesarean decision.

Eden et al., found that the choice of strategy was related to several patient factors, such as desire for vaginal delivery, previous vaginal delivery, avoidance of labor and feelings about previous cesarean delivery. For example, some of the main reasons why women prefer a trial of labor are an easier recovery and being able to quickly return to the care of their children. On the other hand, women who prefer a repeat cesarean may do so because of fear of a long and painful vaginal delivery. [Eden04]

Abitbol et. Al found that patients who were interviewed to assess their attitudes toward VBAC were motivated by reasons different than the medical reasons that were proposed to them. These reasons included a desire to deliver naturally, a fear of having a surgery, and the fear that a cesarean section might harm the infant. In addition, some women wanting a trial of labor felt that a cesarean would inhibit their ability to care for the baby, having to rely on family and hospital staff for help. Women who did not attempt a trial of labor felt that the cesarean section was more convenient because of the ability to schedule and plan the delivery in advance, and were concerned about having a long, strenuous and painful vaginal delivery.
[Abitbol93]

Another study interviewed patients, comparing reactions of experiences of VBAC with experiences of a previous cesarean, to understand the factors that influenced women to attempt a trial of labor, and the factors that women related to the outcome of their birthing experiences. [Fawcett94] The authors found that women were moderately positive about the VBAC experience, and information from hospital staff, family and the media influenced them to attempt a trial of labor. Also, the three most positive things related to a VBAC were a natural childbirth, a shorter recovery time, and the presence of a partner for the birthing experience. The three most negative things were labor pains, perineal discomfort, and technology associated with labor and delivery. With regards to a repeat cesarean, the three most positive things were delivery of a healthy baby, a painless and quick delivery and uncomplicated recovery. The three most negative things were more complicated recovery from surgery, difficulty in caring for a newborn, and the delivery experience.

[Fawcett94]

Similarly, a study by Kline et. al analyzed the motivation behind 241 women with a previous cesarean who had undergone a subsequent childbirth, to understand the factors that determined the selection of each birthing strategy. Patients who had an elective repeat cesarean had their previous cesarean because of failure to progress in labor. Patients who have a VBAC do so because their previous cesarean was a result of fetal distress. The main reasons why women attempt a trial of labor are patient's desire and physician's advice. The main reasons why women decide on a repeat cesarean are because of medical or obstetric indication, patient's desire and physician's advice.

[Kline93]

Two-thirds of women prefer a trial of labor over repeat cesarean. This is due to Hispanic white and Asian ethnicity, teaching hospitals, hospitals with neonatal ICU, shorter recovery time, fear of major surgery, and desire to experience vaginal birth. [Roberts97] A preference of repeat cesarean over trial of labor was attributed to African American and Hispanic ethnicity, schedule concerns, pain of labor and a planned tubal ligation. [Roberts97] Some women change their preference during labor in favor for a repeat cesarean because of labor pain and other factors not related to medical risk. [Roberts97]

From the review it can be seen that women have varied perspectives and attitudes on the strategies available with a child-birth after cesarean decision. In some cases the women may prefer a strategy due to other reasons rather than the associated risks. In general, patients wish to be well informed about medical information and be directly involved in decision making issues, even though they may want their physician to help assist in problem solving issues when deciding medical strategies. Thus it is important that clinicians help and support patients in both problem solving and in decision making. [Deber96]

3. Motivations, Research Question, and General Objectives

Motivations

The literature review yielded a lot of evidence on associated maternal risks for women attempting a trial of labor or repeat cesarean with a prior cesarean. The amount of information available on neonatal outcomes was comparably less. Major maternal risks associated with a child-birth after cesarean decision includes maternal hemorrhage, hysterectomy, incontinence, infection and uterine rupture. Major neonatal risks include sepsis, permanent neurological damage, and death. The study by Guise et. al was one of the most extensive literature reviews available. This paper systematically reviewed many studies and identified the strongest studies to assess the risks associated with a child-birth after cesarean decision. There were limited studies done on examining the maternal risks for future pregnancies; however the study by Pare et. al did significant decision analyses on examining the compounded risks of placental abnormalities and uterine ruptures to future pregnancies.

Regarding decision methodologies, AHP provides an easy and practical method for decision making, allowing patients to weight decision criteria and the decision is based on a subjective risk assessment of the options by the patient. Decision trees and Markov models objectively assess risk for the available options; however, it is not easy to incorporate patient preferences.

From the literature review done on decision aid models constructed for a child-birth after cesarean decision, the majority used decision trees. The probabilities of risks were gathered from literature reviews and/or medical experts. One study by Grobman et.al used both a decision tree followed by a Markov model to understand the costs associated with each type of birthing strategy in women with a prior cesarean. No studies used an AHP based approach. None of the decision analyses included patient preferences, although it was noted in several of the papers that incorporating patient preferences was also vital in determining the best birthing strategy. In addition, information on utility assessments done was very limited. Of the utility assessments that were available, utilities were assigned for only a handful of outcomes, such as perfect health, death of either the mother or infant, or hysterectomy. These utilities were assigned by medical experts, either by using a Visual Analog Scale [Mankuta03] or by using the Time Trade Off method [Chung01]. No utility assessments were done by patients or focus groups.

While the opinion of medical experts is important in understanding utilities of outcomes, patients also need to assess their own personal utilities as seen from literature available on patient preferences, which was emphasized by Mankuta et. al and the systematic literature review done by Eden et. al. The education of women on a child-birth after a cesarean decision, and their perspectives are also extremely vital to assessing appropriate utilities for reaching the best possible birthing decision. [Eden04]

Overall, it is important to integrate patient preferences with risk assessment of health outcomes in the birthing process, in a decision aid methodology for a child-birth after cesarean decision. No study has comprehensively done this, which was a big motivation to do this work.

This thesis focused on two decision methodologies. The first decision methodology was the Analytic Hierarchy Process (AHP) and the second was a decision tree. AHP was decided upon for two reasons. The first is Dr. Karen Eden and Dr. Jeanne-Marie Guise, have developed an AHP decision aid tool in a partnering study, and have collected data on 96 post-natal women. Second, AHP is a decision methodology which is known for its flexibility of including patient preferences and is an easy methodology to administer directly patients. For the second decision methodology, a decision tree was decided upon for its objective assessment of risk for each of the two birthing strategies.

Using these two decision methodologies, three specific decision models were examined. The first decision model was based on a pure AHP methodology- the decision aid tool developed by Dr. Eden and Dr. Guise. The second decision model was a decision tree incorporating the same risk outcomes as in the AHP decision aid tool. At this point it is strongly emphasized that AHP subjectively assesses risk while the decision tree objectively assesses risk. The AHP also includes criteria such as having a good delivery experience. Since no probabilities on this were immediately available, the decision tree lacked this facet when making the decision. This was a motivation to construct a third decision model which assessed risk objectively and included criteria such as having a good delivery experience. The purpose was to create a model that was appropriately comparable to the AHP model since it would include all the criteria presented in the AHP. The only factor that was different was the mode of risk assessment, which also enabled the examination of the mode of risk assessment on the decision. This third decision model used a hybrid AHP and a decision tree approach. A more detailed explanation of each type of decision model can be found in the Methods section.

Research question

As stated in the Motivations section the goal of this thesis was to examine various decision models using AHP and decision trees which incorporated patient preferences, and risk assessment. To facilitate this, the main research question we would like to answer is:

For women considering a child-birth after cesarean decision, do all three decision models: a pure AHP, a decision tree, and a hybrid AHP-decision tree model generate similar decisions?

Objectives

If these approaches generated different results, cases where these were different would be analyzed as to whether one decision model is better for certain types of birthing preferences versus the other. In addition, the other objective was to study how the mode of risk assessment-subjective or objective can influence the decision.

4. Methods

In this thesis three types of decision models were used. The first model was a pure AHP based model already completed in a partnering study and data on 96 postnatal women was available. The second model was a decision tree, in which only medical risks for both the mother and infant were considered. The third model was a hybrid AHP-decision tree based approach, incorporating non-risk criteria from the AHP model and expected values from the decision tree in a collective calculation.

The experimental process will be described in more detail. First the AHP model based decision model is presented, followed by the decision tree and model, and finally how the non-risk criteria from the AHP model and expected values from the decision tree were combined in the third hybrid model.

AHP model for a childbirth after cesarean decision

The AHP model that was used in the partner study is shown below:

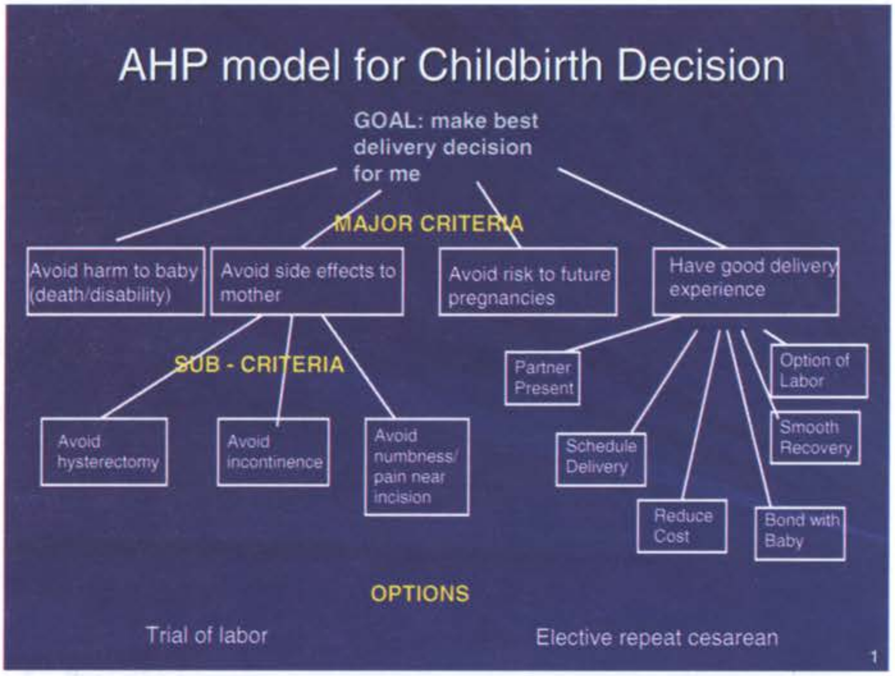

Figure 1: AHP model for childbirth after cesarean decision

The major criteria included avoid harm to baby, avoid side effects to mother, avoid risk to future pregnancies, and have a good delivery experience. These major criteria were separated into two categories: risk and non-risk. Non-risk criteria were defined as criteria not directly related to medical risk, but as criteria related to a personal perspective of various components of the birthing experience. Thus the major criteria in the "risk" category were avoid harm to baby, avoid side effects to mother, and avoid risk to future pregnancies. The non-risk category comprised of "have a good delivery experience". Within the major criteria "avoid side effects to mother" and "have a good delivery experience", sub-criteria were also defined. The main and sub-criteria are listed below.

Risk Factors

1. Avoid complications to infant
2. Avoid complications to mother
    a. Avoid incontinence
    b. Avoid hysterectomy
    c. Avoid numbness/pain near incision
3. Avoid risk to future pregnancy

Non Risk Factors

4. Having an overall good birthing experience
    a. Option to schedule delivery
    b. Partner Involved
    c. Option for labor
    d. Smooth recovery
    e. Bond with Baby
    f. Cost of delivery

For the rest of this document these criteria will be referred to by their number in the list above, e.g. 4.f will refer to the "Cost of Delivery" criteria etc.

The AHP model based decision approach follows the three main steps below:

1. **Determine Preference Weight for each Criterion:** The user does pairwise comparisons of each of the criteria and gives points to the criteria in each pair relative to each other based on her preferences. The pairwise points given by the user are then used to calculate preference weights for each of the criteria. These weights represent how important each of the criteria is to the user. I refer to the weight of a criterion by the symbol w subscripted by the criteria number, e.g. w_1 = preference weight of the criterion "Avoid complications to infant".

2. **Determine Preference Weight for each Option:** The user does a pairwise comparison of the options based on each of the criterion independently and gives points to the options relative to each other based on her perception of the criteria. These points are then used to calculate preference weights for each option. I refer to these preference weights for each option by the symbol o super scripted by the option number and subscripted by the criteria number against which the weight has been evaluated. For example $o_{2.a}^{2}$ = preference weight of option "Repeat Caesarean" based on "Avoid Infection".

3. **Final Decision:** The final decision is made by calculating an overall score for each of the options based on Preference weights calculated above from the points given by the user in pairwise comparisons. The option with the highest overall score becomes the recommended decision for the user based on the AHP model. The overall score is calculated as follows: first we create option weight matrices for each main criteria as shown:

$$O_1 = \begin{bmatrix} o_1^1 \\ o_1^2 \end{bmatrix};\ O_2 = \begin{bmatrix} o_{2.a}^1 & o_{2.b}^1 & o_{2.c}^1 \\ o_{2.a}^2 & o_{2.b}^2 & o_{2.c}^2 \end{bmatrix} \begin{bmatrix} w_{2.a} \\ w_{2.b} \\ w_{2.c} \end{bmatrix};\ O_3 = \begin{bmatrix} o_3^1 \\ o_3^2 \end{bmatrix};\ O_4 = \begin{bmatrix} o_{4.a}^1 & o_{4.b}^1 & \cdots & o_{4.f}^1 \\ o_{4.a}^2 & o_{4.b}^2 & \cdots & o_{4.f}^2 \end{bmatrix} \begin{bmatrix} w_{4.a} \\ w_{4.b} \\ \vdots \\ w_{4.f} \end{bmatrix} \quad (1)$$

The overall score for each option is then calculated as follows:

$$\begin{bmatrix} o_1 \\ o_2 \end{bmatrix} = [O_1 \mid O_2 \mid O_3 \mid O_4] \begin{bmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{bmatrix} \quad (2)$$

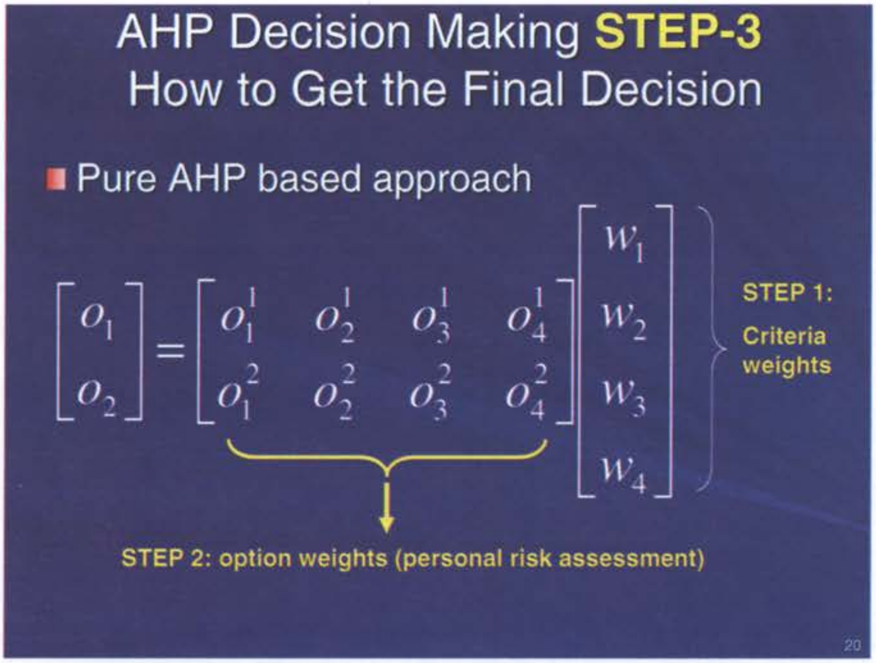

Figure 2: AHP matrices

Decision tree model for a childbirth after cesarean decision

The decision tree model was constructed to model a single decision, containing health states of the mother and the infant. The branches from the two birthing options "Trial of Labor" and "Repeat Cesarean" branched further into the several outcomes which have been described in the AHP model. Each of these sub-branches had the appropriate probability associated with them established by the literature or through an estimate by Dr. Jeanne-Marie Guise. A representation of the decision tree can be found in Appendix B.

A high level view of the decision tree is shown in the figure below. The "Trial of Labor" branch further divided into two sub-trees: "Successful Trial of Labor" and "Emergency Repeat Cesarean". These two sub-trees were identical to the "Elective Repeat Cesarean" sub-tree in terms of outcomes, but only differed in the associated probabilities for each type of strategy.

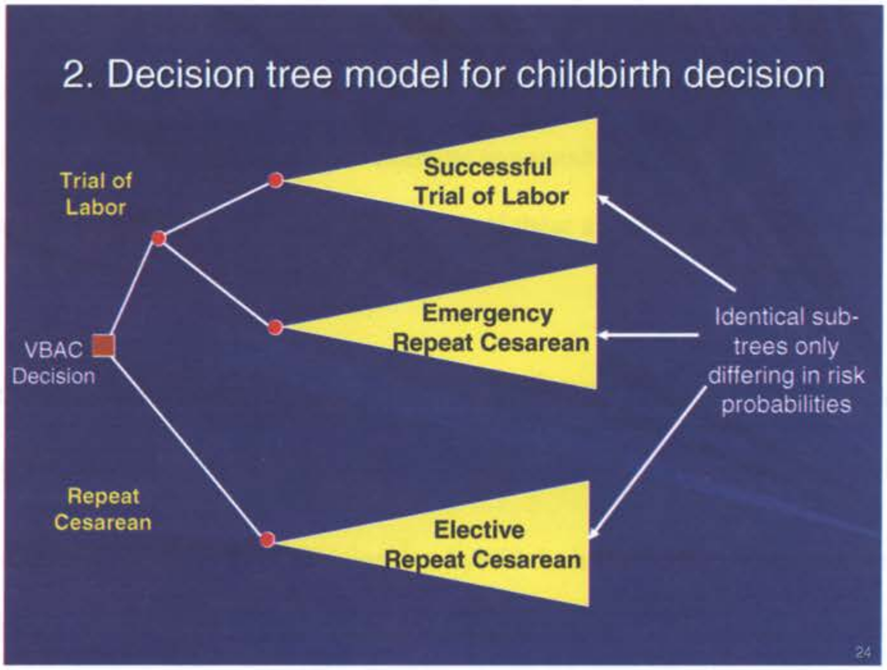

Figure 3: High level view of decision tree

Each of the outcome nodes in the tree also had a utility assigned to it. However, as stated earlier, limited utility assessment was available from the literature, and some utilities were not even available for all the outcomes. One way to have obtained the utilities would have been by administering Time Trade Off or Standard Reference Gamble techniques to clinicians and/or focus groups. However, due to limited resources and the limited timeframe for this thesis, such a path could not be taken.

Thus, utilities were derived from the AHP preference weights using the normalization presented by Vargas et. al. The detailed derivations of these utilities are described in Appendix A. The preference weights (or a combination of them) determined from Step 1 of the pure AHP model were used to estimate utilities. No previous study employed such a technique to estimate utilities for a child-birth after cesarean decision. Thus, *the conversion of preference weights to utilities was also one of the big contributions in this thesis.*

Various sensitivity analyses were also performed on the tree. These were performed on the probabilities, user inputs that were used to generate the criteria weights, and the criteria weights. The goal of the sensitivity analyses was to examine any sensitive areas of the tree which could cause a change in the birthing decision. The results of these analyses are presented in the Results section.

AHP-decision tree hybrid model for a childbirth after cesarean decision

This section presents how the AHP-decision tree hybrid decision model was constructed. The AHP framework presents a nice array of matrices which can also be separated into matrices based on the risk components and non-risk components. Thus another way of viewing the AHP model is that the decision scores for the two options are the addition of components of the risk assessment (in this case subjective) and the non-risk criteria (having a good delivery experience). Therefore, the hybrid replaced the personal subjective risk assessment with the objective risk assessment done by the decision tree.

The next several subsections will present the following: the decision trees, the probability estimates used in the decision trees, the utility derivations and finally using the decision tree along with the non-risk AHP factors in generating a decision.

Decision tree component

The figure of the decision tree was constructed using the TreeAge software. Due to the enormous size of the tree, the tree is presented as two sub-trees- one for trial of labor and one for elective repeat cesarean. These can be found in Appendix B. For the actual analysis, C++ code was written to create the decision tree; the code can be found in Appendix D.

Probability estimates

Below is a table containing all the probability estimates for the health outcomes considered in the decision models.

Table 16: Probability estimates for risks in decision tree

	Successful trial of labor	Failed trial of labor-subsequent cesarean (emergency cesarean)	Elective repeat cesarean
MATERNAL OUTCOMES			
Numbness/pain near incision	.04 (estimate by Dr. Guise)	.14 (estimate by Dr. Guise)	.14 (estimate by Dr. Guise)
Incontinence	.19 (AHP decision aid tool)	.141 (AHP decision aid tool)	.141 (AHP decision aid tool)
Hysterectomy	.0001; range: .00005-.002 (Pare06)	.001; range: .0005-.005 (Pare06)	.0005; range: .0001-.003 (Pare06)
Condition causing risk to future pregnancy	.03 (AHP decision aid tool)	.06 (AHP decision aid tool)	.06 (AHP decision aid tool)
NEONATAL OUTCOMES			
Neonatal death	12.9/10000 (Smith02)	12.9/10000 (Smith02)	1/9014 (Smith02)
Neonatal disability	13/18000 (Landon04)	13/18000 (Landon04)	.00006 (estimate)

The rate of successful trial of labor was set at .76 (based on the AHP decision tool). There were several other papers which also found similar rates in their studies. A table comparing these can be found below:

Table 17: Rate of success in women attempting trial of labormen attempting trial of labor

	[Landon04]	[Loebel04]	[Mozurkewich00]	[Rageth99]	[Wen04]	[Pare06]
Successful VBAC (# patients)	13,139	749	20,746	12,986	92455	--
Failed VBAC (# patients)	4759	178	8067	4627	36505	--
Total number of patients attempting trial of labor	17898	927	28813	17613	128960	--
Successful VBAC rate	0.73	0.8	0.72	0.73	0.72	.7 (range from .5-.9)
Failure VBAC Probability	0.27	0.2	0.28	0.27	0.28	.3

A summary of end-branch outcomes, expected probabilities and multi-attribute utility functions can be found in Appendix A.

Utility derivations

A description of the step-by-step process of computing utilities for outcomes in the decision tree from the AHP criteria weights can be found in Appendix A.

For the comprehensive list of health outcomes, C++ code programs were created to compute all the preference weights for each of the 36 health outcomes and subsequently apply the normalization [Vargas86] to compute the actual utility for each outcome. A summary table of the mean, minimum, maximum, standard deviation and confidence intervals for the computed utilities can be found in Appendix C.

The final hybrid approach

By evaluating the decision tree for the options "Trial of Labor" and "Repeat Cesarean", an expected value (in this case expected utility) is generated, and are designated by qo_{all}^{1} and qo_{all}^{2}. The subjective risk assessments made in the AHP model can be replaced by the expected utilities generated when evaluating the decision tree.

Thus the overall decision scores which include the replacement as described above along with the non-risk criteria can be determined as shown below:

$$\begin{bmatrix} o_1 \\ o_2 \end{bmatrix} = \begin{bmatrix} qo_{all}^{1} \\ qo_{all}^{2} \end{bmatrix} + [O_4]w_4 \quad (3)$$

The figure below shows the mathematical representations of the AHP and hybrid models.

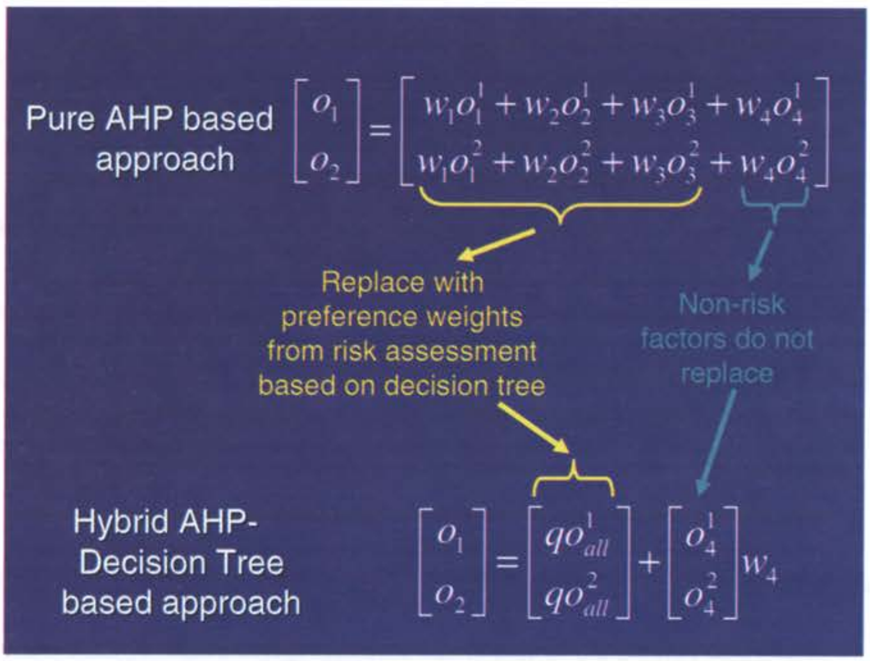

Figure 4: Mathematical representations of AHP and hybrid models

The hybrid model can be summarized as follows. First the criteria weights from the AHP model were converted to utilities for each of the outcomes in the decision tree. Next, the tree was evaluated to obtain the expected value (or expected utility) for each of the two birthing strategies. These values were then multiplied by 100 to put them in a points scale from 0-100. These were further multiplied by the quantity $1 - w_4$, since $1-w_4$ represents the proportion of the entire score devoted to health risks. Once this scaling was complete, these scores were added along with the scores of the non-risk criteria to generate overall scores for each of the two strategies. The higher score among the two strategies corresponded to the better option. The decisions obtained by all the three models were compared and analyzed. These results are presented in the next section.

5. Analyses and Results

Results were gathered and divided into the following areas: the multi-attribute utility assessments, several types of sensitivity analyses, plots of the scores for each birthing strategy in all the three decision models for all 96 women, statistical comparisons between all the three decision models, and a risk assessment analysis.

Each of the next subsections will describe the analyses in more detail.

Utility assessments

The multi-attribute utility assessments for the decision tree were obtained by computing them from the AHP criteria weights based on the 1986 Vargas paper described in the literature review. The specific computational steps are described in Appendix A.

A table of the computed utility estimates can be found in Appendix C. There are a total of 36 health outcomes, each of which comprises both the health of the mother and health of the infant. For each outcome the mean utility estimate, standard deviation and 95% confidence intervals are listed.

Sensitivity analyses

Sensitivity analyses were performed on the decision tree in multiple ways which include

1. One-way sensitivity analyses on each of the following probabilities:
   a. Probabilities of numbness/pain near incision for successful trial of labor, failed trial of labor, and elective repeat cesarean
   b. Probabilities of risk to future pregnancy for successful trial of labor, failed trial of labor, and elective repeat cesarean
   c. Probabilities of hysterectomy for successful trial of labor, failed trial of labor, and elective repeat cesarean
   d. Probabilities of incontinence for successful trial of labor, failed trial of labor, and elective repeat cesarean
2. One-way sensitivity analyses on all pairwise comparisons (Step 1 from AHP model). These sensitivity analyses were computed in the following manner:
   a. For each analysis, the user inputs were varied for the corresponding pairwise comparison, while all other pairwise comparison results were kept constant. The constant pairwise comparisons results were values that had been averaged over all the 96 women.
   b. The computations proceeded in the following order as user inputs were varied for corresponding pairwise comparison:
      i. the preference weights were recalculated
      ii. next utility estimates were recalculated
      iii. finally the overall expected value (expected utility) for each birthing strategy was recalculated for the tree
3. An exhaustive sensitivity analysis on the criteria weights were performed, by varying all three major risk criteria and all three sub-criteria

The sensitivity analyses that were performed in 1 and 2 assumed average utility assessments (over all the 96 women).

The sensitivity analyses that were performed on the probabilities were done in Microsoft Excel. The sensitivity analyses that were performed on the pairwise comparisons of user inputs were done using C code written (Appendix D) and were graphed using Microsoft Excel.

The next several figures show only results where sensitivity occurred for the decision tree. The decision tree was sensitive for the following probabilities: probabilities of risk to future pregnancy, probabilities of numbness/pain near incision, and probabilities of incontinence. The tree was not sensitive to probabilities of hysterectomy.

The tree was sensitive to the following pairwise user comparisons: avoid injury to infant vs. avoid injury to mother, avoid incontinence vs. avoid hysterectomy, and avoid incontinence vs. avoid numbness/pain near incision. The tree was not sensitive to other pairwise comparisons.

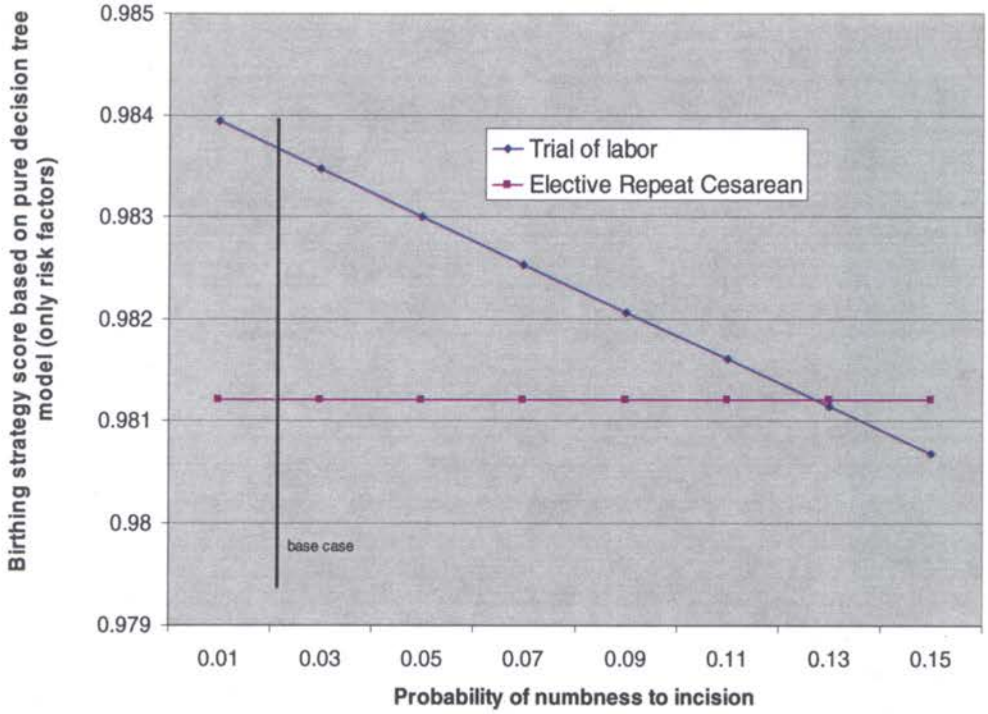

Figure 5: Sensitivity analysis on probability of numbness/pain near incision for successful trial of labor

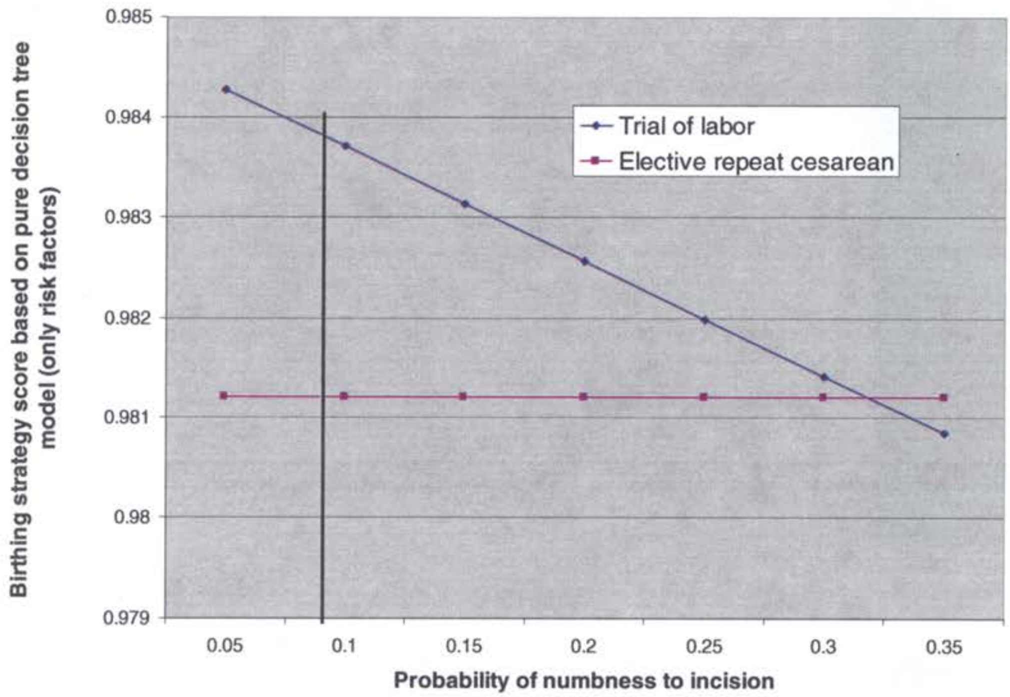

Figure 6: Sensitivity analysis on probability of numbness/pain near incision for failed trial of labor

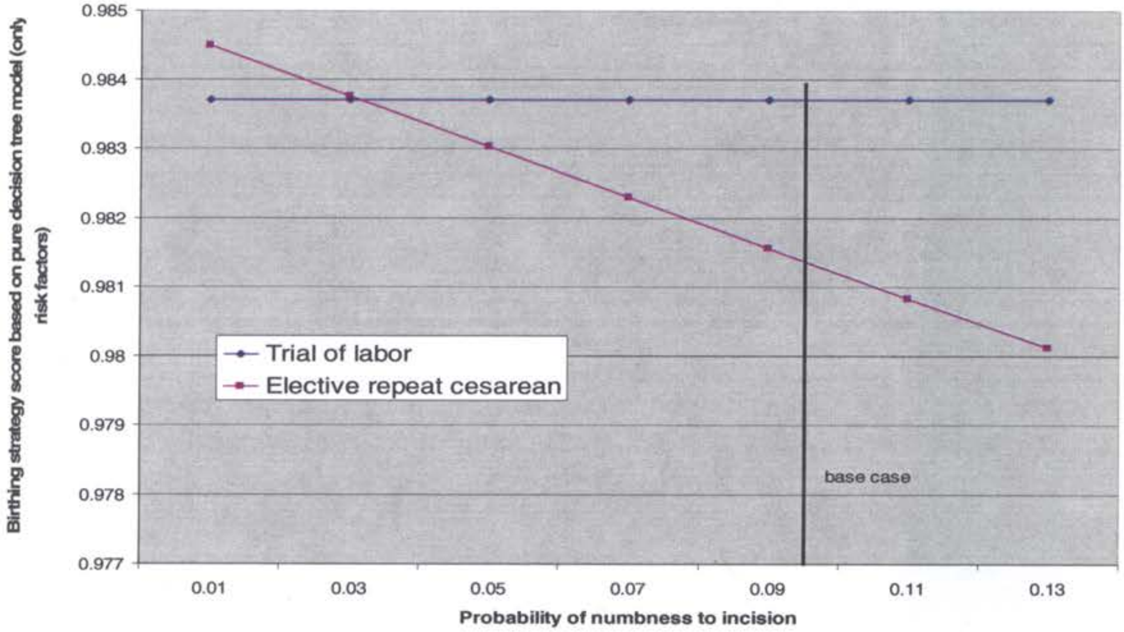

Figure 7: Sensitivity analysis on numbness/pain near incision for elective repeat cesarean

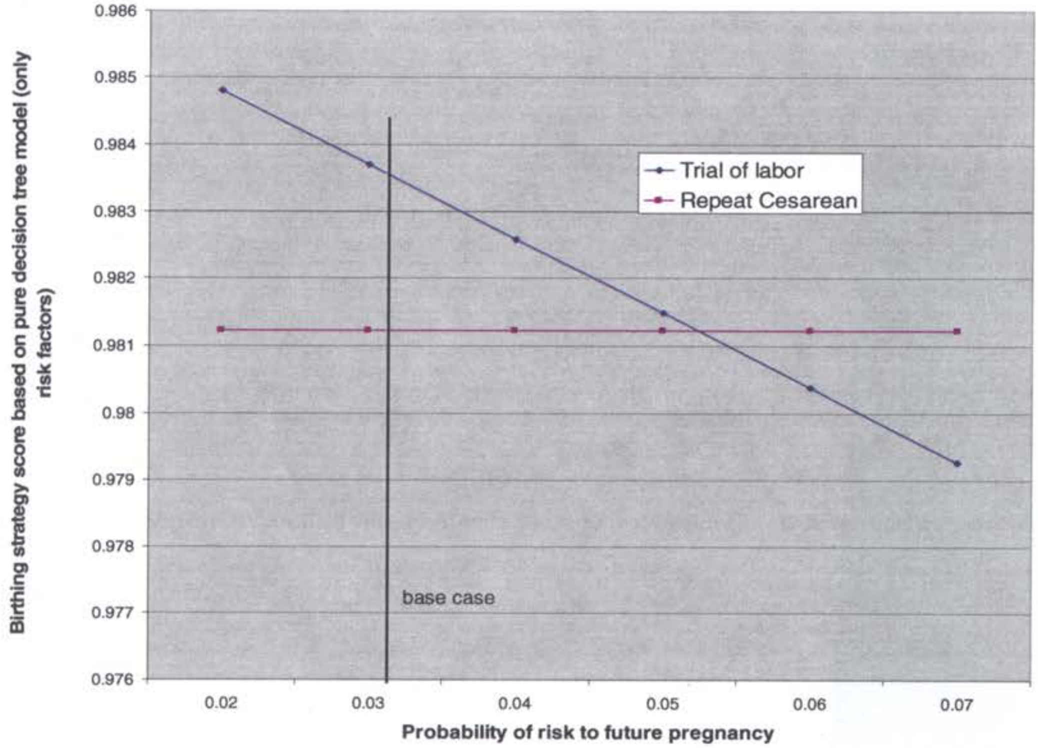

Figure 8: Sensitivity analysis on risk to future pregnancy for successful trial of labor

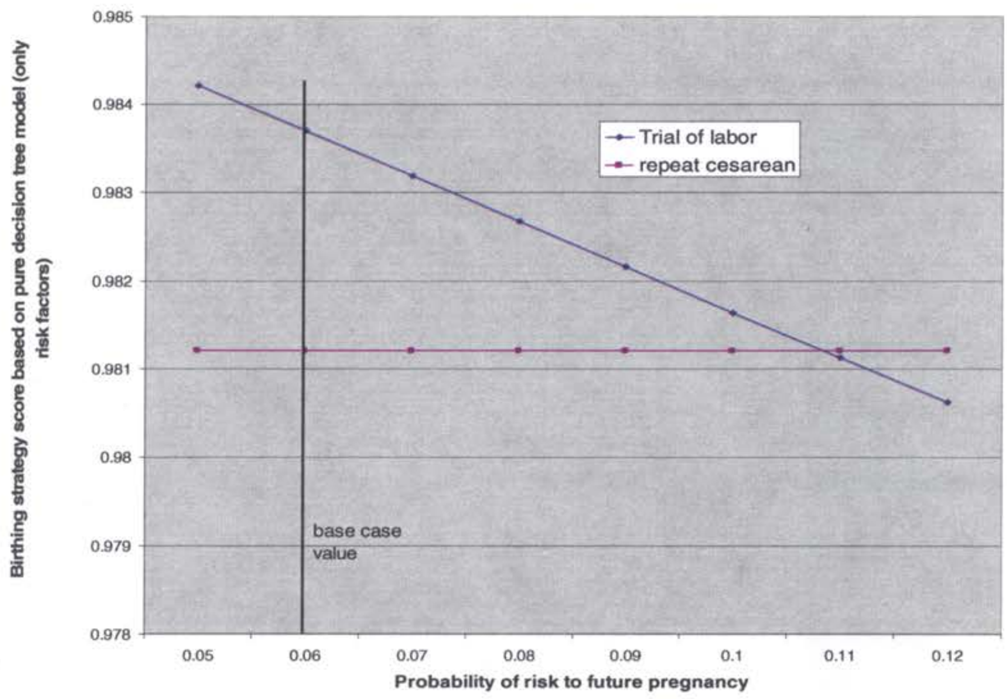

Figure 9: Sensitivity analysis on risk to future pregnancy for failed trial of labor

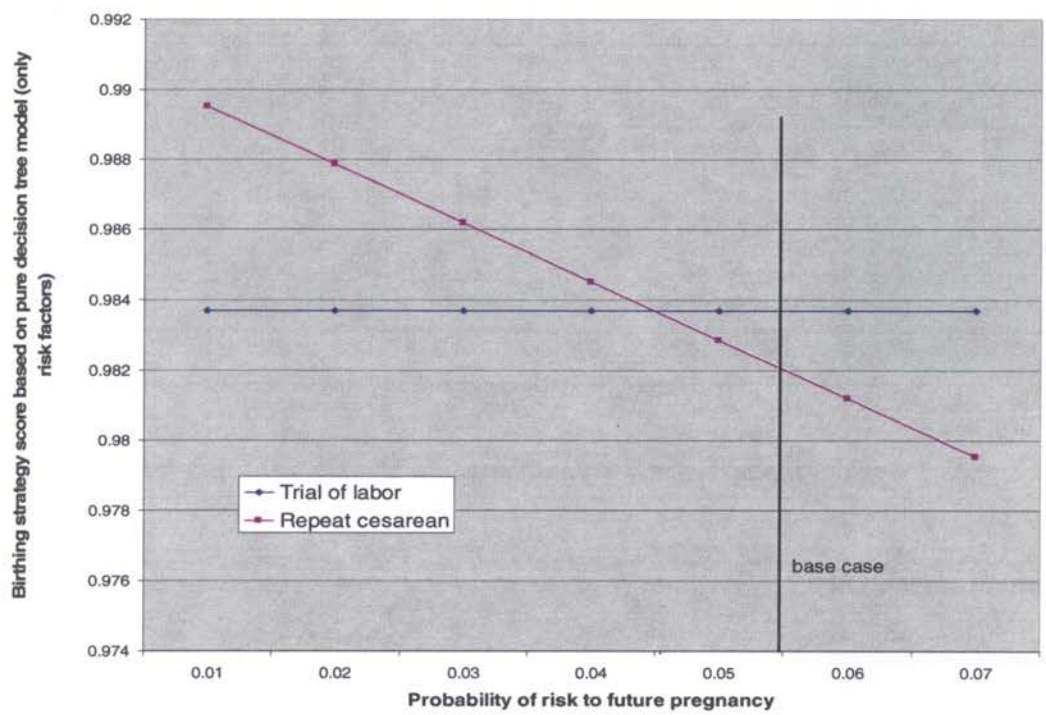

Figure 10: Sensitivity analysis on risk to future pregnancy for elective repeat cesarean

As can be seen from the figures, the decision tree was sensitive for probabilities of risk to future pregnancy and numbness/pain near incision. Although not shown in the figures, the tree was also sensitive for probabilities of incontinence. The tree was not sensitive to probabilities of hysterectomy.

Below are the figures of sensitivity analyses that were sensitive, done on the user inputs in the AHP model. These user inputs corresponded to the sliding bar values (Step 1 from the AHP model) for each of the pairwise comparisons of the decision criteria that would be designated by the user. The birthing scores are the expected values from the decision tree only.

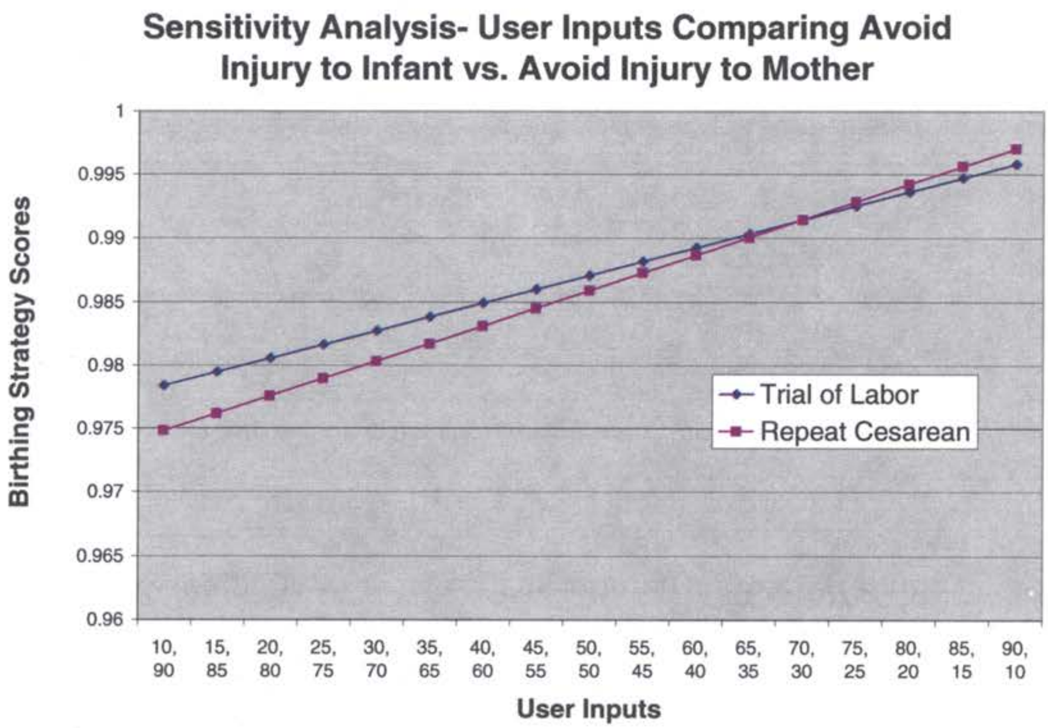

Figure 11: User Input Sensitivity Analysis: Avoid Injury to Infant vs. Avoid Injury to Mother

In the above graph, user inputs of 65 points or less to avoiding injury to infant and user inputs of 35 points or greater to avoiding injury to mother result in a trial of labor decision. User inputs greater than 65 to avoiding injury to infant result in a repeat cesarean decision. This concurs well with the risk probabilities since an elective repeat cesarean has lower risks to the health of the infant when compared to trial of labor.

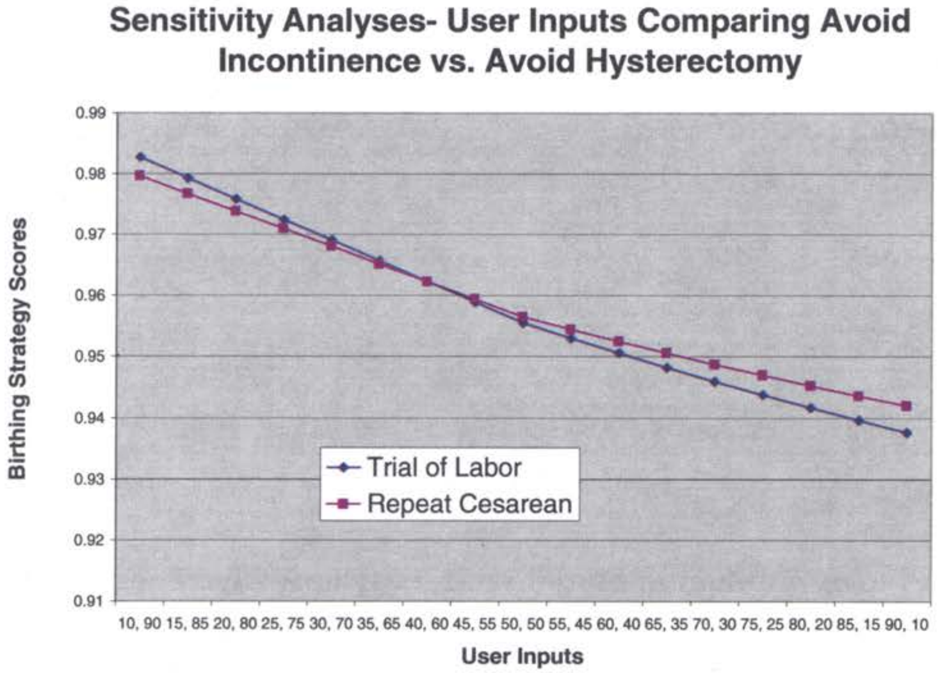

Figure 12: User Input Sensitivity Analysis: Avoid Incontinence vs. Avoid Hysterectomy

In the above graph, user inputs of 40 points or less to avoiding incontinence and user inputs of 60 points or greater to avoiding hysterectomy result in a trial of labor decision. This is as expected since more emphasis is being placed on avoiding hysterectomy, and a trial of labor decision has a slightly lower rate of hysterectomy than an elective repeat cesarean. User inputs greater than 40 to avoiding incontinence result in a repeat cesarean decision. This concurs well with the risk probabilities since an elective repeat cesarean has lower rates of having incontinence and at these inputs more emphasis is being placed on avoiding incontinence than hysterectomy.

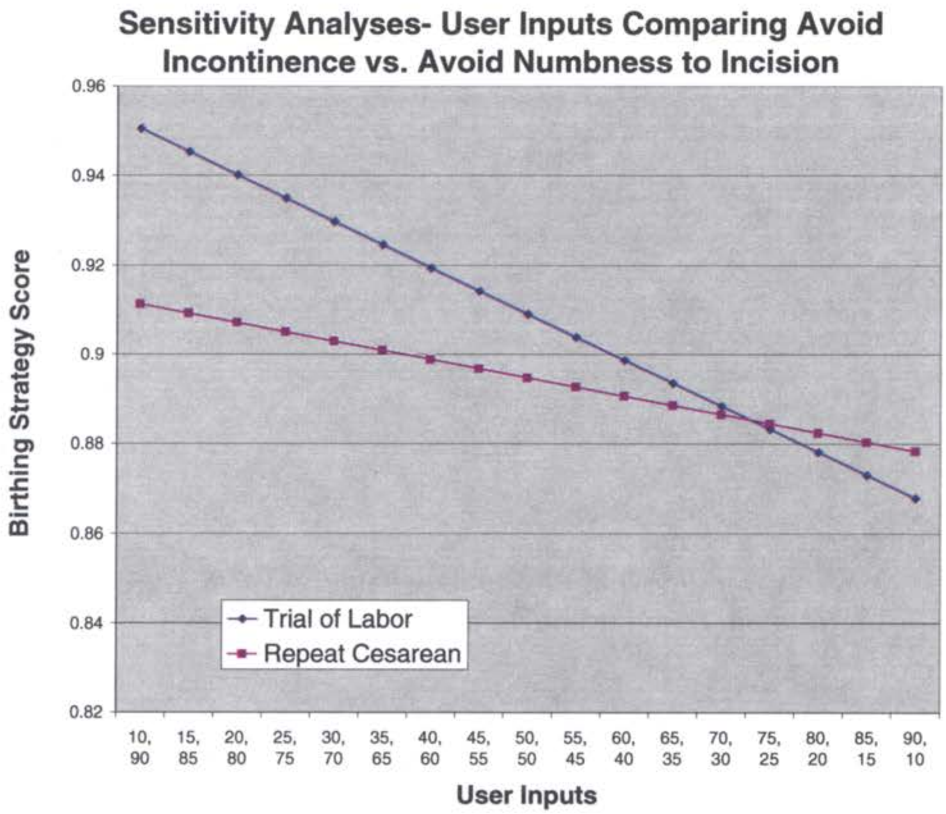

Figure 13: User Input Sensitivity Analysis: Avoid Incontinence vs. Avoid Numbness to Incision

In the above graph, user inputs of 72 points or less to avoiding incontinence and user inputs of 28 points or greater to avoiding numbness/pain near incision result in a trial of labor decision. User inputs greater than 72 to avoiding incontinence result in a repeat cesarean decision. This is as expected, because a very high emphasis on avoiding numbness/pain near incision would result in a trial of labor decision since the probability of having numbness/pain near incision is 1/5 the probability of the value it would be for an elective repeat cesarean. Only for a very high emphasis on avoiding incontinence does the decision model favor an elective repeat cesarean, since the probability of incontinence is lower for cesarean than trial of labor. In that case, avoiding incontinence dominates over avoiding numbness/pain near incision.

Decisions for all models

Evaluating just the decision tree for each of the women in the study yielded a total of 83 trial of labor decisions and 13 elective repeat cesarean decisions. The probability of successful trial of labor was then set at .6 and .5 (base case was .7) to study if changing that probability changed the overall decision. A value of .5 reflects the lowest value of success in the range defined in the paper by Pare et al [Pare06]. At a value of .6, the number of trial of labor decisions decreased to 71 while the number of elective repeat cesarean decisions increased to 25. At a value of .5, the number of trial of labor decisions increased to 50 while the number of elective repeat cesarean decisions decreased further to 46. Thus, as the probability of successful trial of labor decreases, the number of elective repeat cesarean decisions increases. This is expected since a lower probability of successful trial of labor would push the decision more towards an elective repeat cesarean.

Statistical comparisons between decision models

A Chi-Square test was performed between the decision models to compare concordance/discordance. Three series of tests were performed for the following comparisons: pure AHP & pure decision tree models, pure AHP & hybrid models, and decision tree & hybrid models. In general, the null and alternative hypotheses for any of the tests were:

H_0: The proportions of women having a trial of labor decision are the same for both models.

H_A: The proportions of women having a trial of labor decision are different for both models.

The value of alpha was set at .05, above which a value of p would accept the null hypothesis and below which a value of p would reject the null hypothesis.

The goal of the Chi-Square tests was to compare rates of generating a certain birthing strategy versus the other between any two models. The Chi-Square test does not however take into account any differences between decisions for a specific patient. For example, assume that 50% of women had a trial of labor decision and 50% had an elective repeat cesarean decision in the pure AHP model. Next assume that 50% of women had a trial of labor decision and 50% had an elective repeat cesarean decision in the decision tree model. However the group of women who had a trial of labor decision in one model wasn't the same group of women in the other model. Performing a Chi-Square test in this case would still generate a non-significant decision, in other words the decision models are similar since they generate the same rate of one type of decision, even if the decisions differed among specific women between the two models.

For the first comparison, the pure AHP and pure decision tree models were compared. The value of the Chi-Square was equal to 68.95 and the degrees of freedom was 95, subsequently yielding a value of $p < .001$. This means that the null hypothesis is rejected, and that the two models yield different proportions of women deciding one birthing strategy over another. Below is a table showing the number of decisions for each birthing strategy among the two models.

Table 18: Pure AHP vs. Pure decision tree Chi-Square test

	AHP	**Dec Tree**	**Total**
TOL	26	83	109
RC	70	13	83
Total	96	96	192

For the second comparison, the pure AHP and hybrid models were compared. The value of the Chi-Square was equal to 59.03 and the degrees of freedom was 95, subsequently yielding a value of $p < .001$. This means that the null hypothesis is rejected, and that the two models yield different proportions of women deciding one birthing strategy over another. Below is a table showing the number of decisions for each birthing strategy among the two models.

Table 19: Pure AHP vs. Hybrid Chi-Square test

	AHP	Hybrid	Total
TOL	26	79	105
RC	70	17	87
Total	96	96	192

For the third comparison, the pure decision tree and hybrid models were compared. The value of the Chi-Square was equal to .63 and df (degrees of freedom) was 95. However at a significance level of .05, Chi-Square should be greater than or equal to 3.84, for significance. Since Chi-Square is less than 3.84, the null hypothesis is not rejected and there isn't a statistical difference between the two models. This means that the two models yielded similar proportions of women deciding one birthing strategy over another. Below is a table showing the number of decisions for each birthing strategy among the two models.

Table 20: Pure decision tree vs. hybrid Chi-Square test

	Hybrid	Dec Tree	Total
TOL	79	83	162
RC	17	13	30
Total	96	96	192

Comparison of risk assessments between decision models

After generating the decisions for all the three models a series of comparisons was done comparing decisions between the models for different case scenarios of preference weights using specific women as examples. The goal of this set of analyses was to study similarities and differences between decisions among the various models for specific women and understand how risk assessments between the AHP and decision tree models compared.

From the statistical analyses it can be seen that there are differences in many cases for women when comparing AHP & decision tree models and comparing AHP & hybrid models. In order to understand the behavior of the models in these two comparisons, an analysis was done of the extreme cases of women who had obvious decisions based on their criteria weights to ensure that the models were behaving as they should for those cases. Subsequently analysis were done on cases of women who assessed their risk by either over-emphasizing or under-emphasizing risk which led to differences in the final decision between the models. These analyses are described in more detail in the following sections.

Analysis of women with strong birthing strategy decisions

First, an analysis was done of women that had *obvious* birthing strategy decisions based on their preferences. The goal was to validate the decision models using extreme cases as examples. For this women were selected from two different groups: those who had the largest difference in birthing strategy scores in the decision tree and those who had the largest difference in birthing strategy scores based on the AHP model.

Among women who had strong decisions for a particular birthing strategy in the decision tree, subject-14 and subject-87 were selected. A table below shows the main criteria weights for both subject-14 and subject-87.

Table 21: Criteria weights for two women having strong decisions in decision tree model

	Child's Health	**Mother's Health**	**Future Pregnancy**	**Delivery Experience**
Subject-14	0.197	0.305	**0.252**	0.244
Subject-87	**0.680**	0.223	0.0479	0.0479

Subject-87 had extreme scores in the decision tree model, which yielded a decision of elective repeat cesarean. This woman gave a very high preference to avoiding injury to the infant, which resulted in both the decision tree and AHP models to yield an elective repeat cesarean decision. This makes sense since an elective repeat cesarean would result in a lower risk to the infant.

Subject-14 had extreme scores in the decision tree model, and the decision tree yielded a trial of labor decision. However, the AHP model yielded a decision of elective repeat cesarean. On close examination, this woman gave a high preference weights to avoiding risk to future pregnancy, which resulted in a trial of labor decision in the tree since trial of labor has a lower risk to future pregnancy as compared to elective repeat cesarean. The AHP model also indicated a higher preference for trial of labor just based on the risk factors, however this woman gave a higher preference to an elective repeat cesarean for non-risk factors. Since the AHP model combines both the risk and non-risk factors, the final AHP result was an elective repeat cesarean.

Among women who had strong decisions for a particular birthing strategy in the AHP model, subject-83 and subject-85 were selected. Decisions were compared between the AHP and hybrid models. Both models yielded the same result for these two women. Subject-85 gave a high preference to avoiding injury to mother and avoiding risk to future pregnancy, resulting in a trial of labor decision for both models. Subject-83 put a lot of emphasis on an elective repeat cesarean for the non-risk factors, resulting in an elective repeat cesarean for both models.

These examples help to validate what decision one would expect to see based on strong criteria weights for one more criteria among the models. A analysis was not done between the decision tree and the hybrid models since most decisions were exactly the same.

Analysis of women with women having different decisions among models

Next women who had different decisions in all the three models were analyzed. Since most of the decision tree and hybrid decisions matched, the comparison was essentially between differences in decision tree/hybrid decisions and AHP decisions. For these analyses, subject-46, subject-35, and subject-25 were chosen to illustrate various behaviors. A table for each woman containing her decisions for all models and criteria weights is followed by a discussion.

Table 22: Decisions and criteria weights for subject-46

	Decision Tree Result	AHP Result	Hybrid Result
Subject-46	TOL	RC	TOL

Major Criteria

Avoid injury to infant	Avoid injury to mother	Avoid risk to future pregnancy	Satisfactory Experience
0.690	0.108	0.148	0.055

Sub-criteria

Avoid incontinence	Avoid numbness/pain near incision	Avoid hysterectomy
0.118	**0.537**	0.345

Subject-46 put a high emphasis on avoiding numbness around incision amongst all the sub-criteria which led to a decision tree/hybrid result of trial of labor since numbness around incision is lower for trial of labor. However, when actually assessing the risk in the AHP model (to generate the second set of weights based on risk), this woman incorrectly answered that elective repeat cesarean was "more risky" than TOL for numbness or pain around incision leading to a repeat cesarean decision.

Table 23: Decisions and criteria weights for subject-35

	Decision Tree Result	AHP Result	Hybrid Result
Subject-35	TOL	RC	TOL

Major criteria

Avoid injury to infant	Avoid injury to mother	Avoid risk to future pregnancy	Satisfactory Experience
0.44	0.28	**0.158**	0.116

Sub-criteria

Avoid incontinence	Avoid numbness around incision	Avoid hysterectomy
0.2689	**0.0617**	0.6692

Subject-35 gave a high preference to avoiding hysterectomy (which has a slightly lower rate in trial of labor) and a fairly solid emphasis on avoiding risk to future pregnancy. Both hysterectomy and risk to future pregnancy rates are lower for a trial of labor strategy, thus resulting in a trial of labor decision for the decision tree/hybrid models. However, when actually assessing the risk in the AHP model (to generate the second set of weights based on risk), this woman incorrectly answered that elective repeat cesarean was "more risky" than TOL for avoiding risk to future pregnancy. This resulted in an AHP decision of elective repeat cesarean.

Table 24: Decisions and criteria weights for subject-25

	Decision Tree Result	AHP Result	Hybrid Result
Subject-25	TOL	TOL	RC
Avoid injury to infant	**Avoid injury to mother**	**Avoid risk to future pregnancy**	**Satisfactory Experience**
0.32	0.22	0.22	0.23

Subject-25 gave overall similar weights to all the main criteria. Based on the weights, the decision tree yielded a decision of trial of labor. When performing the risk assessment in the AHP model, this woman over-emphasized risks in favor of a trial of labor, thus resulting in an AHP decision of trial of labor as well. However the hybrid model yielded a decision of elective repeat cesarean. This decision can be broken down into the two components: risk factors and non-risk factors. In this model the assessment of risk factors is coming from the decision tree. Since the decision tree scores are still quite close overall, a trial of labor is not strong decision based on the tree. When examining the non-risk factors, this woman gave a high preference for an elective repeat cesarean based on the non-risk factors. This coupled with the high criteria weight on having a satisfactory experience then pushed the decision towards a repeat cesarean. This example shows, had the woman not over-emphasized her risks in favor of a trial of labor decision in the AHP model, the AHP model would have also resulted in an elective repeat cesarean decision.

It can be seen from these examples that different decisions between models resulted mostly from the differences in risk assessment. Subjective assessment in the AHP model resulted in different decisions from the objective risk assessment done in the decision tree and hybrid models.

Overall, the discrepancies in decisions between models fell into one of three categories:

1. Risk assessments between models were comparable, but the non-risk factors pushed the decision
2. Risk assessment was done incorrectly, such as preferring a strategy with a higher risk
3. Subjective and objective risk assessments led to different decisions. It is noted that most women with different decisions fell in this category.

About 10% of the women fell into the second category, where they incorrectly preferred a strategy having a higher risk (e.g. risk to future pregnancy). There were some women who fell into more than one of the categories above, however a detailed analysis was not done to quantify these proportions. Doing this type of analysis would be an area of future research.

Regarding risk assessments, since the AHP model multiplies the matrix of criteria preference weights with the matrix of option weights (the matrix of risk assessment between the two strategies), an incorrect or over-emphasis in the matrix of option weights can be propagated depending on the criteria preference weights in the first matrix.

6. Conclusions

In this thesis a hybrid decision model was developed, incorporating aspects of both AHP and decision tree approaches. The hybrid model performed objective risk assessment of various birthing strategies based on a decision tree, at the same time incorporating user preferences of various health outcomes from an AHP model. To summarize, the main contributions of this thesis are listed below:

- Developing a hybrid decision aid methodology combining advantages of AHP and decision trees and comparing the results from these different models
- Evaluating a method to estimate utilities of various health outcomes from AHP criteria weights
- Performing sensitivity analyses based on variations of risk probabilities and user input preferences
- Performing rigorous risk assessment evaluation and uncovering the issue that in some cases women either over-emphasize or incorrectly assess risk

The main research question that this thesis set out to answer was:

For women considering a child-birth after cesarean decision, do all three decision models: a pure AHP, a decision tree, and a hybrid AHP-decision tree model generate similar decisions?

From the results it can be seen that the decision tree and hybrid models yielded similar decisions while the AHP yielded different decisions when compared to the other two models. When comparing the decision tree and hybrid models it can be seen that the non-risk factors led to some differences in decisions, but predominantly did not have a huge impact in the decision since most of the decisions were the same. Differences where mainly due to subjective perceptions of risk by women in the AHP decision methodology, while the hybrid and decision tree models assessed risk objectively.

As can be seen from the literature review, information on utilities for health outcomes is limited, and only limited to certain types of outcomes. Furthermore, previous utility assessments were made by healthcare professionals only, and several studies noted that incorporating patient preferences and possibly having patients define their own utilities is important to reaching an accurate decision. [Mankuta03] However, doing good utility estimates using either the Time Trade Off or Standard Reference Gamble method can be cumbersome because of limited resources, since most resources are directed towards research in studying medical risks. In addition, these traditional methods are useful for estimating utilities for a limited number of outcomes, and for outcomes limited to a single individual's health. In this thesis however, the computed utilities reflect a large number of outcomes which incorporate both the health of the mother and health of the infant. Thus, the method of using AHP criteria weights to compute the utilities is a reasonable approach to estimate utilities, since an AHP decision model is easily administered and the subject can do a reasonable amount of pairwise comparisons, which in turn can be used to estimate utilities for a much larger set of health outcomes.

Sensitivity analyses revealed that the decision tree was sensitive for probabilities of risk to future pregnancy, incontinence, and numbness/pain near incision. However it was not sensitive to probabilities of hysterectomy. This is due to the fact that the chance of hysterectomy is 100-1000 times smaller than the rest of the probabilities. The tree was sensitive to the following pairwise user comparisons: avoid injury to infant vs. avoid injury to mother, avoid incontinence vs. avoid hysterectomy, and avoid incontinence vs. avoid numbness/pain near incision. The tree was not sensitive to other pairwise comparisons.

An interesting outcome of the decision tree analysis was that the overall difference in the risk scores between the two birthing strategies was small. This leads to interesting interpretations and differences between *objective* and *subjective* risk assessment. From a purely objective (evidence from the literature), i.e. based purely on health outcome probabilities, point of view the overall risks are small to begin with so neither birthing strategy is much better than the other. From a patient's subjective point of view, the perception could be that even though the risks are small, the ratios between probabilities are still significant and small numerical differences can still indicate significant differences between birthing strategies. Studying the risk assessment of the women unveiled that the decision depended on the mode of risk assessment done. In other words, personal risk assessments done in the AHP model led to different decisions when compared to the objective risk assessments done in the other two models.

The results of this thesis also suggest several modifications that can be made to the AHP decision aid tool, and, that is another area for future work. For example one modification can be to first have the women do the pairwise comparisons which generate the option criteria weights based on risk, and then do the pairwise comparisons which generate the criteria preference weights. Currently, the model first does the criteria preference weight comparisons, followed by the option weights based on each risk. This modification may allow women to first see the risks and then appropriately weight criteria later based on the rates of the risks involved with each option.

All this leads to an important conclusion: the way the risk assessment is done can influence the final decision. Even though the evidence from the literature might indicate a certain strategy, the patient's perception of those risks can influence the decision for another strategy. These results suggest that when coming to medical decisions, both types of assessments should be done because individuals can perceive and react to outcomes differently, can have different thresholds of tolerating discomfort, pain, and can have different perspectives on which outcomes they are willing to accept or not. Thus, reaching a decision based solely on objective assessments dictated by literary evidence and clinician recommendations is not enough to reach the best decision. A subjective risk assessment must be also done to understand the patient's perspectives and concerns. By doing a subjective risk assessment, the clinician can then also understand the patient's preferences and a subsequent shared decision making process can ensue between the physician and patient to address areas of gap between clinical recommendations and patient perspectives to come to an overall decision.

Another area of future work would be to construct a decision tree which also included criteria such as having a good delivery experience as in the AHP model. By constructing such a tree, it would give another model which could be compared against both the AHP model and hybrid. The comparison would then be made for three models: the pure AHP, the hybrid, and the pure decision tree (with delivery experience included). Thus there would be one model showing using a subjective risk assessment (the pure AHP), while the other two models would use an objective risk assessment (the hybrid and pure decision tree). In addition, a comparison could also be made between the hybrid and pure decision tree to see whether the way of incorporating the criteria of delivery experience influences the decision.

References

1. [Guise04] Guise J-M, Berlin M, McDonagh M, Osteweil P, Nygren P, Helfand M. Safety of vaginal birth after cesarean: a systematic review. Obstetrics & Gynecology. 2004;103(3):420-429.

2. [Guise03] Guise JM, McDonagh MS, Hashima J, Kraemer DF, Eden KB, Berlin M, Nygren P, Osterweil P, Krages KP, Helfand M. Vaginal birth after cesarean (VBAC). Evid Rep Technol Assess (Summ). 2003 Mar;(71):1-8.

3. [Landon04] Landon MB, et.al. Maternal and perinatal outcomes associated with a trial of labor after prior cesarean delivery. The New England Journal of Medicine 2004 December;351(25):2581-2589.

4. [Loebel04] Loebel G, Zelop CM, Egan JF, Wax J. Maternal and neonatal morbidity after elective repeat Cesarean delivery versus a trial of labor after previous Cesarean delivery in a community teaching hospital. J Matern Fetal Neonatal Med. 2004 Apr;15(4):243-6.

5. [ACOG99] Vaginal birth after previous cesarean delivery:clinical management guidelines for obstetricians-gynecologists. ACOG practice bulletin no. 5. Washington D.C.: American College of Obstetricians and Gynecologists, July 1999.

6. [Eden04] Eden KB, Hashima JN, Osterweil P, Nygren P, Guise JM. Childbirth preferences after cesarean birth: a review of the evidence. Birth. 2004 Mar;31(1):49-60.

7. [Saaty98] Saaty L Thomas, Vargas G Luis. Diagnosis with Dependent Symptoms: Bayes Theorem and the Analytic Hierarchy Process. Operations Research. 1998, July;46(4):491-502

8. [Carter99] Carter KH, Ritchey NP, Castro F, Caccamo LP, Kessler E, Erickson BA. Analysis of three decision-making methods: a breast cancer patient as a model. Med Decis Making. 1999 Jan-Mar;19(1):49-57.

9. [Carter98] Carter KJ, Ritchey NP, Castro F, et al. Treatment of early stage breast cancer in the elderly: a health outcome based approach. Med Decis Making. 1998;18:213-9.

10. [Hunink01] Hunink Myriam, Glasziou Paul. Decision making in health and medicine. Cambridge University Press 2001. pp. 305-336.

11. [Sonnenberg93] Sonnenberg FA, Beck JR. Markov models in medical decision making: a practical guide. Med Decis Making. 1993 Oct-Dec;13(4):322-38.

12. [Chung01] Chung A, Macario A, El-Sayed YY, Riley ET, Duncan B, Druzin ML. Cost-effectiveness of a trial of labor after previous cesarean. Obstet Gynecol. 2001 Jun;97(6):932-41

13. [Chuang99] Chuang JH, Jenders RA. Trial of labor versus elective repeat cesarean section for the women with a previous cesarean section: a decision analysis. Proc AMIA Symp. 1999;226-30.

14. [Bujold04] Bujold E, Hammoud AO, Hendler I, Berman S, Blackwell SC, Duperron L, Gauthier RJ. Trial of labor in patients with a previous cesarean section: does maternal age influence the outcome? *Am J Obstet Gynecol.* 2004 Apr;190(4):1113-8.

15. [Abitbol93] Abitbol MM, Castillo I, Taylor UB, et al. Vaginal birth after cesarean section: The patient's point of view. *Am Fam Physician* 1993;47:129–134.

16. [Gregory99] Gregory KD, et al. Vaginal birth after cesarean and uterine rupture rates in California. *Obstetrics & Gynecology* 1999 December;94(6):985-989.

17. [Grobman00] Grobman, WA, et al. Cost-effectiveness of elective cesarean delivery after one prior low transverse cesarean. Obstetrics and Gynecology 2000;95(5):745-751.

18. [Rageth99] Rageth JC, Juzi C, Grossenbacher H. Delivery after previous cesarean: a risk evaluation. Obstet Gynecol 1999;93:332-7.

19. [McMahon96] McMahon MJ, Luther ER, Bowes WA, Olshan AF. Comparison of a trial of labor with an elective second cesarean section. N Engl J Med 1996;335:689-95.

20. [Wen04] Wen SW, Rusen ID, Walker M, Liston R, Kramer MS, Baskett T, Heaman M, Liu S; Maternal Health Study Group, Canadian Perinatal Surveillance System. Comparison of maternal mortality and morbidity between trial of labor and elective cesarean section among women with previous cesarean delivery. Am J Obstet Gynecol. 2004 Oct;191(4):1263-9.

21. [Hook97] B. Hook, R. Kiwi, S.B. Amini, A. Fanaroff and M. Hack, Neonatal morbidity after elective repeat cesarean section and trial of labor, Pediatrics 100 (1997), pp. 348–353

22. [Mozurkewich00] Mozurkewich EL, Hutton EK. Elective repeat cesarean delivery versus trial of labor: a meta-analysis of the literature from 1989 to 1999. Am J Obstet Gynecol 2000;183: 1187-97.

23. [Richard05] Richardson BS, Czikk MJ, daSilva O, Natale R. The impact of labor at term on measures of neonatal outcome. Am J Obstet Gynecol. 2005 Jan;192(1):219-26.

24. [Fawcett94] Fawcett J, Tulman L, Spedden J. Responses to vaginal birth after cesarean section. Journal of Obstetric, Gynecologic, & Neonatal Nursing. 1994;23(3):253-259.

25. [Roberts97] Roberts RG, Bell HS, Wall EM, Moy JG, Hess GH, Bower HP. Trial of labor or repeated cesarean section. The woman's choice. *Arch Fam Med.* 1997 Mar-Apr;6(2):120-5.

26. [Kline93] Kline J, Arias F. Analysis of factors determining the selection of repeated cesarean section or trial of labor in patients with histories of prior cesarean delivery. *J Reprod Med* 1993; 38:289–292.

27. [Deber96] Deber RB, Kraetschmer N, Irvine J. What role do patients wish to play in treatment decision making? *Arch Intern Med.* 1996 Jul 8;156(13):1414-20.

28. [Flamm94] Flamm BL, Goings JR, Liu Y, Wolde-Tsadik G. Elective repeat cesarean delivery versus trial of labor: a prospective multicenter study. Obstet Gynecol 1994;83:927–32.

29. [Vargas86] Vargas LG. Utility theory and reciprocal pairwise comparisons: the eigenvector method. Socio-Economic Planning in Science 1986;20(6):387-391.

30. [Pare06] Pare E, Quinones JN, Macones GA. Vaginal birth after caesarean section versus elective repeat caesarean section: assessment of maternal downstream health outcomes. BJOG. 2006 Jan;113(1):75-85.

31. [Smith02] Smith GCS, Pell JP, Cameron AD, Dobbie R. Risk of perinatal death associated with labor after previous cesarean delivery in uncomplicated term pregnancies. JAMA 2002;287:2684 –90.

Appendix A- all health outcomes, expected probabilities and multi-attribute utility functions

The table below shows the overall probability expressions and the probability values based on table 16 for each of the health outcomes in the decision tree. The key to the symbols used in the probability expressions is as below:

$p^{\{i\}}_{bhth}$ = Probability that baby is healthy (no injury to baby)
$p^{\{i\}}_{bdth}$ = Probability of death to baby
$p^{\{i\}}_{bdis}$ = Probability of disability to baby
$p^{\{i\}}_{mhth}$ = Probability that mother is healthy (no injury to mother)
$p^{\{i\}}_{mrfp}$ = Probability of risk to future pregnancy in mother
$p^{\{i\}}_{mnmb}$ = Probability of numbness around incision in mother
$p^{\{i\}}_{mhst}$ = Probability of hysterectomy in mother

The super-script *{i}* takes on the values: elective repeat cesarean, successful TOL and emergency repeat cesarean depending on the birthing process which results in the health outcomes. In other words it indicates that the values of the above probabilities will vary based on the birthing process (as shown in Table 16), however the basic probability expression remains the same for a given health outcome.

The probability expressions for a health outcome representing a combination of health states were determined assuming independence of individual health states.

Table 25: Health outcomes and expected probabilities

Health Outcomes	Probability Expression	Probability Values		
		Elective Repeat Cesarean	Successful TOL	Emergency Repeat Cesarean
Mother healthy, infant healthy	$p^{\{i\}}_{mhth} \times p^{\{i\}}_{bhth}$	6.9838E-01	7.7434E-01	6.6366E-01
Mother healthy, neonatal death	$p^{\{i\}}_{mhth} \times p^{\{i\}}_{bdth}$	7.6835E-05	1.0009E-03	8.5785E-04
Mother healthy, neonatal disability	$p^{\{i\}}_{mhth} \times p^{\{i\}}_{bdis}$	4.1910E-05	5.5865E-04	4.7880E-04
Risk to future pregnancy, infant healthy	$p^{\{i\}}_{mrfp} \times p^{\{i\}}_{bhth}$	5.9990E-02	2.9940E-02	5.9879E-02
Risk to future pregnancy, neonatal death	$p^{\{i\}}_{mrfp} \times p^{\{i\}}_{bdth}$	6.6000E-06	3.8700E-05	7.7400E-05
Risk to future pregnancy, neonatal disability	$p^{\{i\}}_{mrfp} \times p^{\{i\}}_{bdis}$	3.6000E-06	2.1600E-05	4.3200E-05
Numbness around incision, infant healthy	$p^{\{i\}}_{mnmb} \times (1 - p^{\{i\}}_{mrfp} - p^{\{i\}}_{minc} - p^{\{i\}}_{mhst}) \times p^{\{i\}}_{bhth}$	7.9836E-02	1.5886E-02	7.6346E-02
Numbness around incision, neonatal death	$p^{\{i\}}_{mnmb} \times (1 - p^{\{i\}}_{mrfp} - p^{\{i\}}_{minc} - p^{\{i\}}_{mhst}) \times p^{\{i\}}_{bdth}$	8.7835E-06	2.0534E-05	9.8685E-05
Numbness around incision, neonatal disability	$p^{\{i\}}_{mnmb} \times (1 - p^{\{i\}}_{mrfp} - p^{\{i\}}_{minc} - p^{\{i\}}_{mhst}) \times p^{\{i\}}_{bdis}$	4.7910E-06	1.1461E-05	5.5080E-05
Numbness around incision, risk to future pregnancy, infant healthy	$p^{\{i\}}_{mrfp} \times p^{\{i\}}_{mnmb} \times p^{\{i\}}_{bhth}$	5.9990E-03	5.9879E-04	5.9879E-03
Numbness around incision, risk to future pregnancy, neonatal death	$p^{\{i\}}_{mrfp} \times p^{\{i\}}_{mnmb} \times p^{\{i\}}_{bdth}$	6.6000E-07	7.7400E-07	7.7400E-06
Numbness around incision, risk to future pregnancy, neonatal disability	$p^{\{i\}}_{mrfp} \times p^{\{i\}}_{mnmb} \times p^{\{i\}}_{bdis}$	3.6000E-07	4.3200E-07	4.3200E-06

Numbness around incision, incontinence, infant healthy	$p^{(i)}_{mnmb} \times p^{(i)}_{minc} \times (1 - p^{(i)}_{mrfp} - p^{(i)}_{mhst}) \times p^{(i)}_{bhth}$	1.3245E-02	3.3685E-03	1.6306E-02
Numbness around incision, incontinence, neonatal death	$p^{(i)}_{mnmb} \times p^{(i)}_{minc} \times (1 - p^{(i)}_{mrfp} - p^{(i)}_{mhst}) \times p^{(i)}_{bdth}$	1.4572E-06	4.3541E-06	2.1077E-05
Numbness around incision, incontinence, neonatal disability	$p^{(i)}_{mnmb} \times p^{(i)}_{minc} \times (1 - p^{(i)}_{mrfp} - p^{(i)}_{mhst}) \times p^{(i)}_{bdis}$	7.9482E-07	2.4302E-06	1.1764E-05
Numbness around incision, incontinence, hysterectomy, infant healthy	$p^{(i)}_{mnmb} \times p^{(i)}_{minc} \times p^{(i)}_{mhst} \times p^{(i)}_{bhth}$	7.0488E-06	3.4730E-07	1.7365E-05
Numbness around incision, incontinence, hysterectomy, neonatal death	$p^{(i)}_{mnmb} \times p^{(i)}_{minc} \times p^{(i)}_{mhst} \times p^{(i)}_{bdth}$	7.7550E-10	4.4892E-10	2.2446E-08
Numbness around incision, incontinence, hysterectomy, neonatal disability	$p^{(i)}_{mnmb} \times p^{(i)}_{minc} \times p^{(i)}_{mhst} \times p^{(i)}_{bdis}$	4.2300E-10	2.5056E-10	1.2528E-08
Numbness around incision, incontinence, risk to f+A20uture pregnancy, infant healthy	$p^{(i)}_{mrfp} \times p^{(i)}_{mnmb} \times p^{(i)}_{minc} \times p^{(i)}_{bhth}$	8.4586E-04	1.0419E-04	1.0419E-03
Numbness around incision, incontinence, risk to future pregnancy, neonatal death	$p^{(i)}_{mrfp} \times p^{(i)}_{mnmb} \times p^{(i)}_{minc} \times p^{(i)}_{bdth}$	9.3060E-08	1.3468E-07	1.3468E-06
Numbness around incision, incontinence, risk to future pregnancy, neonatal disability	$p^{(i)}_{mrfp} \times p^{(i)}_{mnmb} \times p^{(i)}_{minc} \times p^{(i)}_{bdis}$	5.0760E-08	7.5168E-08	7.5168E-07
Numbness around incision, hysterectomy, infant healthy	$p^{(i)}_{mnmb} \times p^{(i)}_{mhst} \times p^{(i)}_{bhth}$	4.9992E-05	1.9960E-06	9.9799E-05
Numbness around incision, hysterectomy, neonatal death	$p^{(i)}_{mnmb} \times p^{(i)}_{mhst} \times p^{(i)}_{bdth}$	5.5000E-09	2.5800E-09	1.2900E-07
Numbness around incision, hysterectomy, neonatal disability	$p^{(i)}_{mnmb} \times p^{(i)}_{mhst} \times p^{(i)}_{bdis}$	3.0000E-09	1.4400E-09	7.2000E-08
Incontinence, infant healthy	$p^{(i)}_{minc} \times (1 - p^{(i)}_{mrfp} - p^{(i)}_{mhst}) \times p^{(i)}_{bhth}$	1.3245E-01	1.6842E-01	1.6306E-01
Incontinence, neonatal death	$p^{(i)}_{minc} \times (1 - p^{(i)}_{mrfp} - p^{(i)}_{mhst}) \times p^{(i)}_{bdth}$	1.4572E-05	2.1770E-04	2.1077E-04
Incontinence, neonatal disability	$p^{(i)}_{minc} \times (1 - p^{(i)}_{mrfp} - p^{(i)}_{mhst}) \times p^{(i)}_{bdis}$	7.9482E-06	1.2151E-04	1.1764E-04
Incontinence, risk to future pregnancy, infant healthy	$p^{(i)}_{mrfp} \times p^{(i)}_{minc} \times p^{(i)}_{bhth}$	8.4586E-03	5.2095E-03	1.0419E-02
Incontinence, risk to future pregnancy, neonatal death	$p^{(i)}_{mrfp} \times p^{(i)}_{minc} \times p^{(i)}_{bdth}$	9.3060E-07	6.7338E-06	1.3468E-05
Incontinence, risk to future pregnancy, neonatal disability	$p^{(i)}_{mrfp} \times p^{(i)}_{minc} \times p^{(i)}_{bdis}$	5.0760E-07	3.7584E-06	7.5168E-06
Incontinence, hysterectomy, infant healthy	$p^{(i)}_{minc} \times p^{(i)}_{mhst} \times p^{(i)}_{bhth}$	7.0488E-05	1.7365E-05	1.7365E-04
Incontinence, hysterectomy, neonatal death	$p^{(i)}_{minc} \times p^{(i)}_{mhst} \times p^{(i)}_{bdth}$	7.7550E-09	2.2446E-08	2.2446E-07
Incontinence, hysterectomy, neonatal disability	$p^{(i)}_{minc} \times p^{(i)}_{mhst} \times p^{(i)}_{bdis}$	4.2300E-09	1.2528E-08	1.2528E-07
Hysterectomy, infant healthy	$p^{(i)}_{mhst} \times p^{(i)}_{bhth}$	4.9992E-04	9.9799E-05	9.9799E-04
Hysterectomy, neonatal death	$p^{(i)}_{mhst} \times p^{(i)}_{bdth}$	5.5000E-08	1.2900E-07	1.2900E-06
Hysterectomy, neonatal disability	$p^{(i)}_{mhst} \times p^{(i)}_{bdis}$	3.0000E-08	7.2000E-08	7.2000E-07

Table 26 below contains the multi attribute utility functions for each of the health outcomes in the decision tree. The utility functions for a health outcome are the same irrespective of which of the three birthing processes lead to it. All the multi attribute utility functions are composed of the following basic user preference weights as defined in the AHP decision model:

Risk Factors (Main Criteria)	Preference Weights
Avoid Injury to Infant	w_1
Avoid Injury to Mother	w_2
Avoid Risk to Future Pregnancy	w_3

Risk Factors (Sub-Criteria of "Avoid Injury to Mother")	Preference Weights
Avoid Incontinence	$w_{2.a}$
Avoid Hysterectomy	$w_{2.b}$
Avoid Numbness around Incision	$w_{2.c}$

The multi attribute utility functions are determined from the above user preference weights for each criteria and sub-criteria using the following process:

Step 1: First we flatten out the hierarchy in the AHP model by scaling the sub-criteria weights by the weight of the corresponding main criteria. With this we get the following list of weights:

Risk Factors (Flattened Hierarchy)	Preference Weights
Avoid Injury to Infant	w_1
Avoid Incontinence to Mother	$w_2 w_{2.a}$
Avoid Hysterectomy to Mother	$w_2 w_{2.b}$
Avoid Numbness around Incision to Mother	$w_2 w_{2.c}$
Avoid Risk to Future Pregnancy	w_3

Step 2: The weight for a complementary health state is assumed to be the negative of the weight corresponding to the health state as shown in the table below:

Complementary Health State	Preference Weights
Injury to Infant	0- w_1
Incontinence to Mother	0- $w_2 w_{2.a}$
Hysterectomy to Mother	0- $w_2 w_{2.b}$
Numbness around Incision to Mother	0- $w_2 w_{2.c}$
Risk to Future Pregnancy	0- w_3

Step 3: The weights corresponding to a combination of health states are determined assuming linearity. In other words the weight of a health outcome which is a combination of two health states is assumed to be the sum of the weights corresponding to the two health states. The weight for a health outcome becomes its multi attribute utility function. These functions for all the health outcomes in the decision tree are shown in Table 26 below:

Table 26: Multi-attribute preference functions

Health States	Multi-attribute preference function
Mother healthy, infant healthy	$W_1 = w1 + w_2 + w_3$
Mother healthy, neonatal death	$W_2 = (0\text{-}w_1) + w_2 + w_3$
Mother healthy, neonatal disability	$W_3 = (0\text{-}w_1) + w_2 + w_3$
Risk to future pregnancy, infant healthy	$W_4 = w_1 + w_2 + (0\text{-}w_3)$
Risk to future pregnancy, neonatal death	$W_5 = (0\text{-}w_1) + w_2 + (0\text{-}w_3)$
Risk to future pregnancy, neonatal disability	$W_6 = (0\text{-}w_1) + w_2 + (0\text{-}w_3)$
Numbness around incision, infant healthy	$W_7 = w_1 + (0\text{-}(w_2w_{2.a})) + (w_2w_{2.b}) + (w_2w_{2.c}) + w_3$
Numbness around incision, neonatal death	$W_8 = (0\text{-}w_1) + (0\text{-}(w_2w_{2.a})) + (w_2w_{2.b}) + (w_2w_{2.c}) + w_3$
Numbness around incision, neonatal disability	$W_9 = (0\text{-}w_1) + (0\text{-}(w_2w_{2.a})) + (w_2w_{2.b}) + (w_2w_{2.c}) + w_3$
Numbness around incision, risk to future pregnancy, infant healthy	$W_{10} = w_1 + (0\text{-}(w_2w_{2.a})) + (w_2w_{2.b}) + (w_2w_{2.c}) + (0\text{-}w_3)$
Numbness around incision, risk to future pregnancy, neonatal death	$W_{11} = (0\text{-}w_1) + (0\text{-}(w_2w_{2.a})) + (w_2w_{2.b}) + (w_2w_{2.c}) + (0\text{-}w_3)$
Numbness around incision, risk to future pregnancy, neonatal disability	$W_{12} = (0\text{-}w_1) + (0\text{-}(w_2w_{2.a})) + (w_2w_{2.b}) + (w_2w_{2.c}) + (0\text{-}w_3)$
Numbness around incision, incontinence, infant healthy	$W_{13} = w_1 + (0\text{-}(w_2w_{2.a})) + (0\text{-}(w_2w_{2.b})) + (w_2w_{2.c}) + w_3$
Numbness around incision, incontinence, neonatal death	$W_{14} = (0\text{-}w_1) + (0\text{-}(w_2w_{2.a})) + (0\text{-}(w_2w_{2.b})) + (w_2w_{2.c}) + w_3$
Numbness around incision, incontinence, neonatal disability	$W_{15} = (0\text{-}w_1) + (0\text{-}(w_2w_{2.a})) + (0\text{-}(w_2w_{2.b})) + (w_2w_{2.c}) + w_3$
Numbness around incision, incontinence, hysterectomy, infant healthy	$W_{16} = w_1 + (w_2w_{2.a}) + (0\text{-}(w_2w_{2.b})) + (w_2w_{2.c}) + w_3$
Numbness around incision, incontinence, hysterectomy, neonatal death	$W_{17} = (0\text{-}w_1) + (w_2w_{2.a}) + (0\text{-}(w_2w_{2.b})) + (w_2w_{2.c}) + w_3$
Numbness around incision, incontinence, hysterectomy, neonatal disability	$W_{18} = (0\text{-}w_1) + (w_2w_{2.a}) + (0\text{-}(w_2w_{2.b})) + (w_2w_{2.c}) + w_3$
Numbness around incision, incontinence, risk to f+A20uture pregnancy, infant healthy	$W_{19} = w_1 + (w_2w_{2.a}) + (w_2w_{2.b}) + (0\text{-}(w_2w_{2.c})) + w_3$
Numbness around incision, incontinence, risk to future pregnancy, neonatal death	$W_{20} = (0\text{-}w_1) + (w_2w_{2.a}) + (w_2w_{2.b}) + (0\text{-}(w_2w_{2.c})) + w_3$
Numbness around incision, incontinence, risk to future pregnancy, neonatal disability	$W_{21} = (0\text{-}w_1) + (w_2w_{2.a}) + (w_2w_{2.b}) + (0\text{-}(w_2w_{2.c})) + w_3$
Numbness around incision, hysterectomy, infant healthy	$W_{22} = w_1 + (w_2w_{2.a}) + (w_2w_{2.b}) + (0\text{-}(w_2w_{2.c})) + (0\text{-}w_3)$
Numbness around incision, hysterectomy, neonatal death	$W_{23} = (0\text{-}w_1) + (w_2w_{2.a}) + (w_2w_{2.b}) + (0\text{-}(w_2w_{2.c})) + (0\text{-}w_3)$
Numbness around incision, hysterectomy, neonatal disability	$W_{24} = (0\text{-}w_1) + (w_2w_{2.a}) + (w_2w_{2.b}) + (0\text{-}(w_2w_{2.c})) + (0\text{-}w_3)$
Incontinence, infant healthy	$W_{25} = w_1 + (0\text{-}(w_2w_{2.a})) + (w_2w_{2.b}) + (0\text{-}(w_2w_{2.c})) + w_3$
Incontinence, neonatal death	$W_{26} = (0\text{-}w_1) + (0\text{-}(w_2w_{2.a})) + (w_2w_{2.b}) + (0\text{-}(w_2w_{2.c})) + w_3$
Incontinence, neonatal disability	$W_{27} = (0\text{-}w_1) + (0\text{-}(w_2w_{2.a})) + (w_2w_{2.b}) + (0\text{-}(w_2w_{2.c})) + w_3$
Incontinence, risk to future pregnancy, infant healthy	$W_{28} = w_1 + (0\text{-}(w_2w_{2.a})) + (0\text{-}(w_2w_{2.b})) + (0\text{-}(w_2w_{2.c})) + w_3$
Incontinence, risk to future pregnancy, neonatal death	$W_{29} = (0\text{-}w_1) + (0\text{-}(w_2w_{2.a})) + (0\text{-}(w_2w_{2.b})) + (0\text{-}(w_2w_{2.c})) + w_3$
Incontinence, risk to future pregnancy, neonatal disability	$W_{30} = (0\text{-}w_1) + (0\text{-}(w_2w_{2.a})) + (0\text{-}(w_2w_{2.b})) + (0\text{-}(w_2w_{2.c})) + w_3$
Incontinence, hysterectomy, infant healthy	$W_{31} = w_1 + (0\text{-}(w_2w_{2.a})) + (w_2w_{2.b}) + (0\text{-}(w_2w_{2.c})) + (0\text{-}w_3)$
Incontinence, hysterectomy, neonatal death	$W_{32} = (0\text{-}w_1) + (0\text{-}(w_2w_{2.a})) + (w_2w_{2.b}) + (0\text{-}(w_2w_{2.c})) + (0\text{-}w_3)$
Incontinence, hysterectomy, neonatal disability	$W_{33} = (0\text{-}w_1) + (0\text{-}(w_2w_{2.a})) + (w_2w_{2.b}) + (0\text{-}(w_2w_{2.c})) + (0\text{-}w_3)$
Hysterectomy, infant healthy	$W_{34} = w_1 + (w_2w_{2.a}) + (0\text{-}(w_2w_{2.b})) + (0\text{-}(w_2w_{2.c})) + w_3$
Hysterectomy, neonatal death	$W_{35} = (0\text{-}w_1) + (w_2w_{2.a}) + (0\text{-}(w_2w_{2.b})) + (0\text{-}(w_2w_{2.c})) + w_3$
Hysterectomy, neonatal disability	$W_{36} = (0\text{-}w_1) + (w_2w_{2.a}) + (0\text{-}(w_2w_{2.b})) + (0\text{-}(w_2w_{2.c})) + w_3$

The utility of each of the above health outcomes can now be estimated using the following normalization:

$$U_i = \frac{W_i - W_{\min}}{W_{Max} - W_{Min}}$$

Where W_{max} and W_{min} are the maximum and minimum weights (among W_i) respectively. [Vargas86]

Appendix B- decision trees

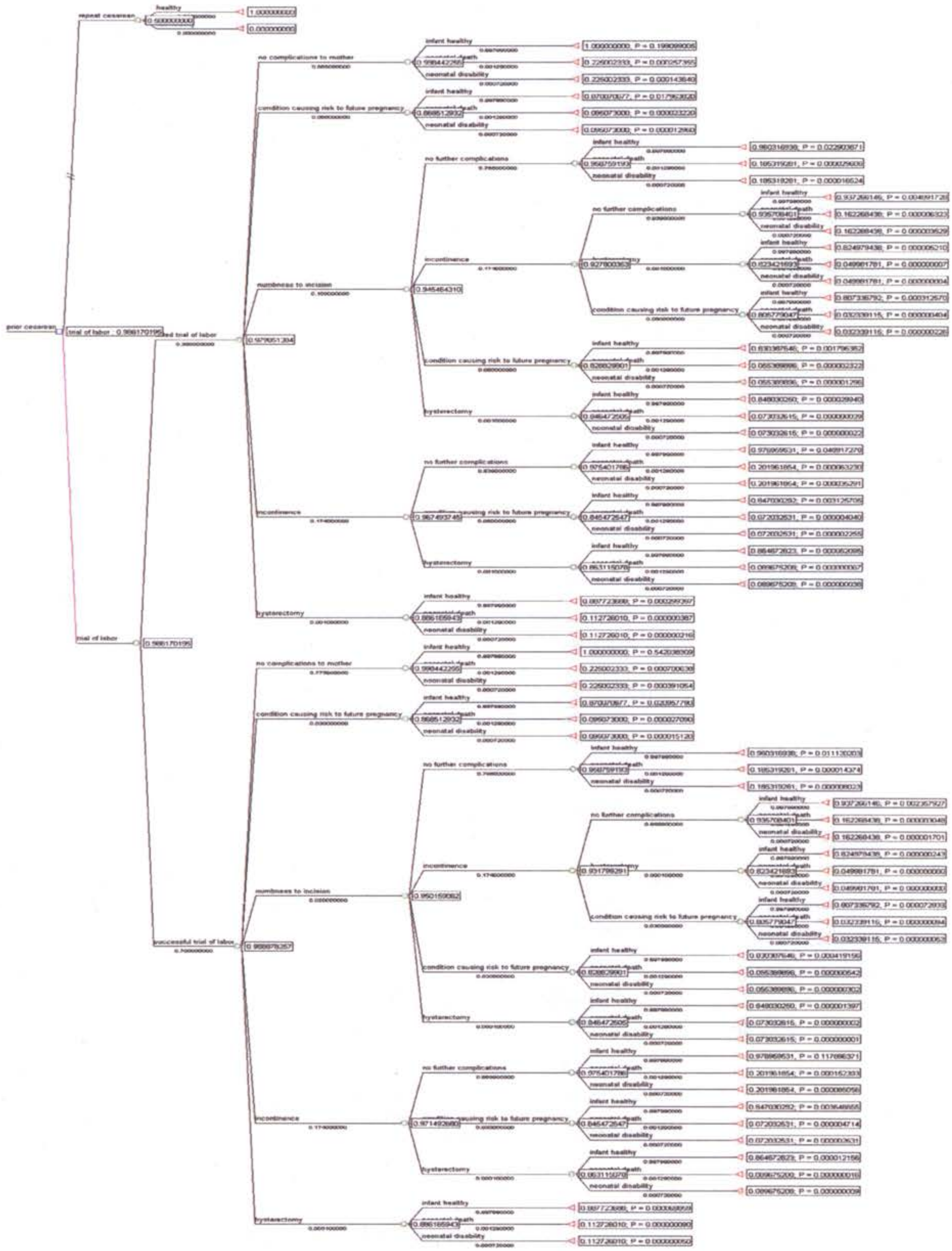

Figure 14: Sub-tree for trial of labor

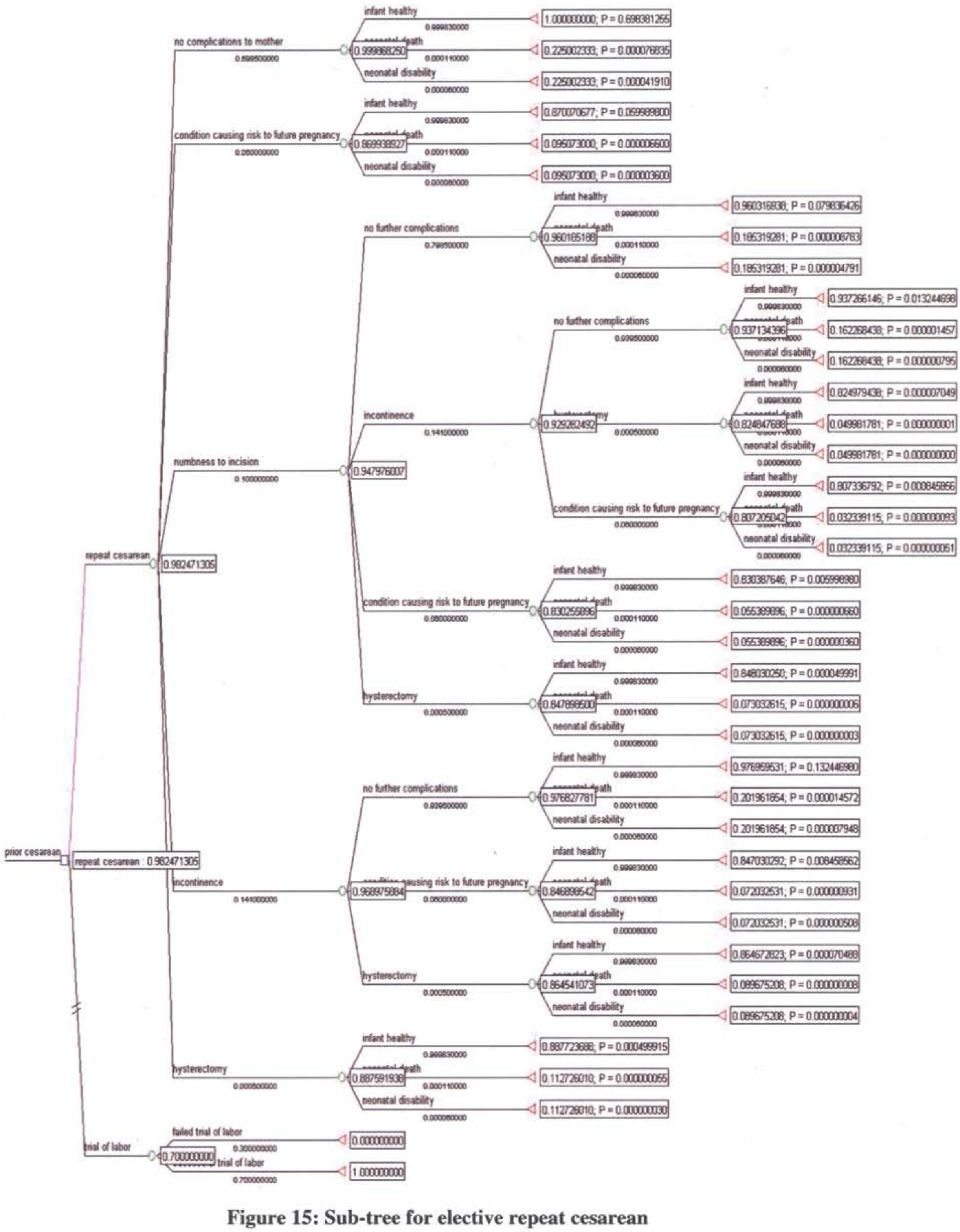

Figure 15: Sub-tree for elective repeat cesarean

Appendix C- utility assessments

Health States	Average Utility	Minimum	Maximum	Std. deviation	95% Confidence Interval	
					lower	upper
Mother healthy, infant healthy	1	1	1	0	1	1
Mother healthy, neonatal death	0.225002333	0.102486	0.715277	0.119458	0.200798	0.249207
Mother healthy, neonatal disability	0.225002333	0.102486	0.715277	0.119458	0.200798	0.249207
Risk to future pregnancy, infant healthy	0.870070677	0.385714	0.961272	0.091436	0.851544	0.888597
Risk to future pregnancy, neonatal death	0.095073	0.010675	0.443925	0.085084	0.077833	0.112313
Risk to future pregnancy, neonatal disability	0.095073	0.010675	0.443925	0.085084	0.077833	0.112313
Numbness around incision, infant healthy	0.960316938	0.813919	0.997547	0.034815	0.953263	0.967371
Numbness around incision, neonatal death	0.185319281	0.057279	0.694155	0.110063	0.163018	0.20762
Numbness around incision, neonatal disability	0.185319281	0.057279	0.694155	0.110063	0.163018	0.20762
Numbness around incision, risk to future pregnancy, infant healthy	0.830387646	0.376482	0.944254	0.09709	0.810715	0.85006
Numbness around incision, risk to future pregnancy, neonatal death	0.055389896	0.002506	0.398425	0.070261	0.041154	0.069626
Numbness around incision, risk to future pregnancy, neonatal disability	0.055389896	0.002506	0.398425	0.070261	0.041154	0.069626
Numbness around incision, incontinence, infant healthy	0.937266146	0.647098	0.989325	0.058547	0.925403	0.949129
Numbness around incision, incontinence, neonatal death	0.162268438	0.048684	0.614286	0.087082	0.144624	0.179913
Numbness around incision, incontinence, neonatal disability	0.162268438	0.048684	0.614286	0.087082	0.144624	0.179913
Numbness around incision, incontinence, hysterectomy, infant healthy	0.824979438	0.488098	0.959784	0.09947	0.804825	0.845134

Numbness around incision, incontinence, hysterectomy, neonatal death	0.049981781	0	0.474633	0.078592	0.034058	0.065906
Numbness around incision, incontinence, hysterectomy, neonatal disability	0.049981781	0	0.474633	0.078592	0.034058	0.065906
Numbness around incision, incontinence, risk to future pregnancy, infant healthy	0.807336792	0.284723	0.922657	0.12018	0.782986	0.831688
Numbness around incision, incontinence, risk to future pregnancy, neonatal death	0.032339115	0	0.352925	0.056631	0.020865	0.043814
Numbness around incision, incontinence, risk to future pregnancy, neonatal disability	0.032339115	0	0.352925	0.056631	0.020865	0.043814
Numbness around incision, hysterectomy, infant healthy	0.84803025	0.5455	0.989	0.08443	0.830923	0.865137
Numbness around incision, hysterectomy, neonatal death	0.073032615	0.00617	0.60703	0.100783	0.052612	0.093453
Numbness around incision, hysterectomy, neonatal disability	0.073032615	0.00617	0.60703	0.100783	0.052612	0.093453
Incontinence, infant healthy	0.976959531	0.66822	0.997518	0.03998	0.968859	0.98506
Incontinence, neonatal death	0.201961854	0.09301	0.623608	0.095522	0.182607	0.221316
Incontinence, neonatal disability	0.201961854	0.09301	0.623608	0.095522	0.182607	0.221316
Incontinence, risk to future pregnancy, infant healthy	0.847030292	0.305845	0.946967	0.112953	0.824144	0.869917
Incontinence, risk to future pregnancy, neonatal death	0.072032531	0.002453	0.398425	0.069949	0.05786	0.086205
Incontinence, risk to future pregnancy, neonatal disability	0.072032531	0.002453	0.398425	0.069949	0.05786	0.086205
Incontinence, hysterectomy, infant healthy	0.864672823	0.5455	0.965257	0.084432	0.847565	0.88178
Incontinence, hysterectomy, neonatal death	0.089675208	0.00676	0.483956	0.082351	0.072989	0.106361

Incontinence, hysterectomy, neonatal disability	0.089675208	0.00676	0.483956	0.082351	0.072989	0.106361
Hysterectomy, infant healthy	0.887723688	0.591	0.995923	0.07057	0.873425	0.902022
Hysterectomy, neonatal death	0.11272601	0.02256	0.628151	0.106674	0.091112	0.13434
Hysterectomy, neonatal disability	0.11272601	0.02256	0.628151	0.106674	0.091112	0.13434

Appendix D- C++ Code for Thesis

```
/*****************************************************************/
/* This file contains all the code for decision trees           */
/* 1/20/07                                                       */
/*                                                               */
/* Author: Poonam Sharma                                         */
/*****************************************************************/

#include "dec_tree.h"

// GLOBAL FUNCTIONS
bool decode_user_arguments(int argc, char*argv[]);
void user_shell();
void run_ahp_weight_sensitivity_analysis(void);
void run_user_criteria_sensitivity_analysis(void);

// GLOBAL VARIABLES
string decision_tree_fname;
string ahp_wt_fname;
bool start_shell;
bool verbose_mod=false;

/***************************************/
/* The main starting point of the code */
/***************************************/
int main(int argc,char *argv[]) {

   // READ IN THE USER ARGUMENTS
   start_shell = decode_user_arguments(argc,argv);

   // CREATE THE TREE DATA STRUCTURES
   initialize_utility_function_map();
   ahp_calc = new AHPCalculations(ahp_wt_fname);
   decision_tree = new DecisionTree(decision_tree_fname);
   util_calc = new UtilityCalculations(decision_tree,ahp_calc);


   // USER SHELL
   if(start_shell) user_shell();
   // OTHERWISE RUN BATCH MODE
   else {

      /* Print AHP weights corresponding to the user input preferences */
      ahp_calc->determine_AHP_weights_from_user_preferences();
        ahp_calc->print_all_ahp_weights();

      string fname = "trees/prob.txt";
      decision_tree->read_in_probabilities(fname);
      ahp_calc->restore_user_value_for_all_weights();
      decision_tree->evaluate_tree(util_calc,0);
      /* Experiment Specific Messaging */
      if(decision_tree->tree_nodes[1]->get_value()>decision_tree->tree_nodes[2]-
>get_value()) printf("Trial of Labor\n");
      else printf("Repeat Cesarean\n");
   }


   // DELETE THE TREE DATA STRUCTURE CREATED
   delete ahp_calc;
   delete decision_tree;
   delete util_calc;
}


/*******************************************/
/* This function decodes the user arguments */
/*******************************************/
```

```
bool decode_user_arguments(int argc, char*argv[]) {
   bool run_shell=true;
   if((argc!=3) && (argc!=4)) {
      cout << "Usage: " << argv[0] << " <AHP weight values file> <decision tree file> [-
noshell]\n";
      exit(1);
   }
   ahp_wt_fname = (string)(argv[1]);
   decision_tree_fname = (string)(argv[2]);

   if(argc==4) {
      string option = (string)(argv[3]);
      string::size_type loc = option.find("-noshell");
      if(loc==0) run_shell=false;
   }
   return(run_shell);
}

/*************************************/
/* This function creates a user shell */
/*************************************/
void user_shell(void) {
   bool done=false;

   while(!done) {
      printf("DecTree> ");
      string user_input;
      std::getline(cin,user_input);


      // HELP
      string::size_type loc = user_input.find("h");
      if(loc==0) {
         printf("(q)uit\n");
         printf("(a)hp weight sensitivity analysis\n");
         printf("(u)ser criteria sensitivity analysis\n");
         printf("(e)valuate the decision tree\n");
         printf("(v)erbose\n");
         printf("(h)elp\n");
         printf("(p)rint utilities\n");
         printf("(n)ew probabilities\n");
         continue;
      }

      // NEW PROBABILITIES
      loc = user_input.find("n");
      if(loc==0) {
         cout << "Probability File Name: ";
         string fname;
         std::getline(cin,fname);
         decision_tree->read_in_probabilities(fname);
         continue;
      }

      // QUIT
      loc = user_input.find("q");
      if(loc==0) {
         cout << "All Done. Bye!\n";
         done=true;
         continue;
      }

      // RUN AHP WEIGHT SENSITIVITY ANALYSIS
      loc = user_input.find("a");
      if(loc==0) {
         run_ahp_weight_sensitivity_analysis();
         continue;
      }

      // RUN USER CRITERIA SENSITIVITY ANALYSIS
      loc = user_input.find("u");
```

```
      if(loc==0) {
         run_user_criteria_sensitivity_analysis();
         continue;
      }

      // EVALUATE TREE
      loc = user_input.find("e");
      if(loc==0) {
         ahp_calc->restore_user_value_for_all_weights();
         decision_tree->evaluate_tree(util_calc,0);
         continue;
      }

      // VERBOSE
      loc = user_input.find("v");
      if(loc==0) {
         if(verbose) {
            verbose=false;
            printf("Verbose OFF\n");
         } else {
            verbose=true;
            printf("Verbose ON\n");
         }
         continue;
      }

      // PRINT UTILITIES
      loc = user_input.find("p");
      if(loc==0) {
         if(utility_print) {
            utility_print=false;
            printf("Utility Print OFF\n");
         } else {
            utility_print=true;
            printf("Utility Print ON\n");
         }
         continue;
      }

      // COMPLETE UTILITY ANALYSIS
      loc = user_input.find("c");
      if(loc==0) {
         ahp_calc->perform_matrix_size_3_sensitivity_analysis(0,decision_tree,util_calc);
         continue;
      }

      // ILLEGAL
      cout << "Unrecognized option: " << user_input << " Type h for Help\n";
   }
}

/**********************************************************/
/* This function runs a sensitivity analysis on AHP weight */
/**********************************************************/
void run_ahp_weight_sensitivity_analysis(void) {
   printf("AHP Weight Index: ");
   int wt_ix;
   string inp;
   std::getline(cin,inp);
   wt_ix = atoi(inp.c_str());


   ahp_calc->start_sensitivity_analysis(wt_ix,0.1);
   while(ahp_calc->increment_weight_being_analyzed())
      decision_tree->evaluate_tree(util_calc,0);

}

/**********************************************************************/
/* This function runs sensitivity analysis on user criteria preference */
/**********************************************************************/
```

```
void run_user_criteria_sensitivity_analysis(void) {
   string inp;
   int uc1, uc2;
   int mtx_ix;
   printf("User Criteria 1: ");
   std::getline(cin,inp); uc1 = atoi(inp.c_str());

   printf("User Criteria 2: ");
   std::getline(cin,inp); uc2 = atoi(inp.c_str());

   printf("Matrix Number: ");
   std::getline(cin,inp); mtx_ix = atoi(inp.c_str());

   /* Debug: Modified User Input Sensitivity Analysis */
   ahp_calc->start_user_preference_sensitivity_analysis_mod(mtx_ix,uc1,uc2,0.05);
   while(ahp_calc->increment_user_preference_being_analyzed_mod())
      decision_tree->evaluate_tree(util_calc,0);
   /* End */

   /*ahp_calc->start_user_preference_sensitivity_analysis(mtx_ix,uc1,uc2,5);
     while(ahp_calc->increment_user_preference_being_analyzed())
     decision_tree->evaluate_tree(util_calc,ahp_calc);*/
}


/*********************************/
/* Decision tree class functions *
/*********************************/

/*********************************************/
/* Constructor: Read in tree from user file */
/*********************************************/
DecisionTree::DecisionTree(string &tree_fname) {
   // Read in the decision tree from the user given file
   ifstream tree_file;
   tree_file.open(tree_fname.c_str(), ios::in);

   // KEYWORD STRINGS
   const string number_keywd = "Number of Nodes: ";
   const string node_keywd = "Node: ";
   const string child_keywd = "Child: ";
   const string parent_keywd = "Parent: ";
   const string type_keywd = "Type: ";
   const string utility_keywd = "Utility: ";

   // READ IN THE NUMBER OF NODES
   string number_line;
   std::getline(tree_file,number_line);
   string::size_type loc = number_line.find(number_keywd);
   if((loc == string::npos) || (loc!=0)) {
      cout << "ERROR: Illegal file format: Number of nodes not specified " << number_line
<< "\n";
      exit(1);
   }
   string number_str = number_line.substr(number_keywd.length(),number_line.length());
   int no_of_nodes = atoi(number_str.c_str());
   if(no_of_nodes==0) {
      cout << "ERROR: Illegal number of nodes " << number_str << "\n";
   }
   if(start_shell) { cout << "Number of nodes=" << no_of_nodes << "\n"; }
   tree_nodes.clear();
   tree_nodes.resize(no_of_nodes);


   // PARSE ALL THE LINES IN THE CODE
   int curr_node_number=-1;
   TreeNode *new_tree_node=0;
   bool child_read=true;
   bool parent_read=true;
   bool type_read=true;
   bool utility_read=true;
```

```
    while(!tree_file.eof()) {
        string nextLine;
        std::getline(tree_file, nextLine);

        // IGNORE COMMENT
        loc = nextLine.find("//");
        if(loc == 0) { continue; }

        // NEW NODE
        loc = nextLine.find(node_keywd);
        if((loc != string::npos) && (loc==0)) {
            if(verbose) cout << "New Node :: " << nextLine << "\n";

            // CHECK FOR THE PREVIOUSLY READ NODE
            if((!parent_read) && (!child_read)) {
                cout << "ERROR: Node " << curr_node_number << " has neither children or
parent\n";
                exit(1);
            }
            if(!type_read) {
                cout << "ERROR: Node " << curr_node_number << " does not have a type
specified\n";
                exit(1);
            }
            if(!utility_read) {
                if(new_tree_node->get_type()==OUTCOME_NODE) {
                    cout << "ERROR: Utility not specified for Outcome Node " <<
curr_node_number << "\n";
                    exit(1);
                }
            }
            child_read = false;
            parent_read = false;
            type_read=false;
            utility_read=false;

            // CREATE A NEW NODE FOR THE CURRENT NODE
            curr_node_number++;
            new_tree_node = new TreeNode(curr_node_number);
            tree_nodes[curr_node_number]=new_tree_node;

            // VERIFY NODE NUMBER
            string node_number_str = nextLine.substr(node_keywd.length(),nextLine.length());
            int node_number = atoi(node_number_str.c_str());
            if((node_number != curr_node_number)) {
                cout << "ERROR: Illegal node number string " << node_number_str << "\n";
                exit(1);
            }
        }

        // NODE TYPE
        loc = nextLine.find(type_keywd);
        if((loc != string::npos) && (loc==0)) {
            if((curr_node_number==-1) || (new_tree_node==0)) {
                cout << "ERROR: Illegal Type Format in " << tree_fname << "\n";
                exit(1);
            }
            string type_str = nextLine.substr(type_keywd.length(),nextLine.length());

            loc = type_str.find("DecisionNode");
            if((loc != string::npos) && (loc==0)) {
                if(verbose) cout << "Decision Node\n";
                new_tree_node->set_type(DECISION_NODE);
                type_read=true;
            }
            loc = type_str.find("OutcomeNode");
            if((loc != string::npos) && (loc==0)) {
                if(verbose) cout << "Outcome Node\n";
                new_tree_node->set_type(OUTCOME_NODE);
                type_read=true;
            }
```

```
         loc = type_str.find("ChanceNode");
         if((loc != string::npos) && (loc==0)) {
            if(verbose) cout << "Chance Node\n";
            new_tree_node->set_type(CHANCE_NODE);
            type_read=true;
         }
         if(!type_read) {
            cout << "ERROR: Illegal node type string " << type_str << " (Node Number " <<
curr_node_number << ")\n";
            exit(1);
         }
      }

      // CHILDREN
      loc = nextLine.find(child_keywd);
      if((loc != string::npos) && (loc==0)) {
         // CHECK ORDER
         if(!type_read) {
            cout << "ERROR: Node type not specified before chiden (Node Number " <<
curr_node_number << ")\n";
            exit(1);
         }
         if(new_tree_node->get_type() == OUTCOME_NODE) {
            cout << " ERROR: Outcome nodes cannot have children (Node Number " <<
curr_node_number << ")\n";
         }

         // READ PROBABILITY FOR CHANCE NODE
         float prob = 0;
         if(new_tree_node->get_type() == CHANCE_NODE) {
            loc = nextLine.find(", ");
            if(loc == string::npos) {
               cout << "ERROR: Illegal child probability " << nextLine << " (Node Number
" << curr_node_number << ")\n";
               exit(1);
            }
            string prob_str = nextLine.substr((loc+2),nextLine.length());
            nextLine = nextLine.substr(0,loc);
            prob = atof(prob_str.c_str());
            if(verbose) cout << "Probability (" << prob_str << ") " << prob << " (" <<
nextLine << ")\n";
         }

         loc = nextLine.find(", ");
         if(loc != string::npos) {
            cout << "ERROR: Illegal child string " << nextLine << " (Node Number " <<
curr_node_number << ")\n";
            exit(1);
         }


         string child_str = nextLine.substr(child_keywd.length(),nextLine.length());
         int child_id = atoi(child_str.c_str());
         if((child_id<0) || (child_id<=curr_node_number)) {
            cout << "ERROR: Illegal child string " << child_str << " (Node Number " <<
curr_node_number << ")\n";
            exit(1);
         }
         if(verbose) cout << "Child=" << child_id << "\n";

         if((curr_node_number==-1) || (new_tree_node==0)) {
            cout << "ERROR: Illegal Child Format in " << tree_fname << " (Node Number "
<< curr_node_number << ")\n";
            exit(1);
         }
         child_read = true;
         new_tree_node->add_child(child_id,prob);

      }

      // PARENT
```

```
      loc = nextLine.find(parent_keywd);
      if((loc != string::npos) && (loc==0)) {
         // CHECK ORDER
         if(!type_read) {
            cout << "ERROR: Node type not specified before parent (Node Number " <<
curr_node_number << ")\n";
            exit(1);
         }

         string parent_str = nextLine.substr(parent_keywd.length(),nextLine.length());
         int parent_id = atoi(parent_str.c_str());
         if((parent_id<0) || (parent_id>=curr_node_number)) {
            cout << "ERROR: Illegal parent string " << parent_str << " (Node Number " <<
curr_node_number << ")\n";
            exit(1);
         }
         if(curr_node_number==0) {
            cout << "ERROR: Root node cannot have parent\n";
            exit(1);
         }
         if(verbose) cout << "Parent=" << parent_id << "\n";

         if((curr_node_number==-1) || (parent_read) || (new_tree_node==0)) {
            cout << "ERROR: Illegal Parent Format in " << tree_fname  << " (Node Number "
<< curr_node_number << ")\n";
            exit(1);
         }
         parent_read = true;
         new_tree_node->set_parent(parent_id);
      }

      // UTILITY
      loc = nextLine.find(utility_keywd);
      if((loc != string::npos) && (loc==0)) {
         // CHECK ORDER
         if(!type_read) {
            cout << "ERROR: Node type not specified before utility (Node Number " <<
curr_node_number << ")\n";
            exit(1);
         }
         if(new_tree_node->get_type()!=OUTCOME_NODE) {
            cout << "ERROR: Utility specified for non outcome node (Node Number " <<
curr_node_number << ")\n";
            exit(1);
         }
         string utility_str = nextLine.substr(utility_keywd.length(),nextLine.length());

         // Make sure a utility function exists corresponding to the ulitility expression
name
         if(util_function_map.find(utility_str)==util_function_map.end()) {
            cout << "ERROR: Illegal Utility String " << utility_str << " (Node Number "
<< curr_node_number << ")\n";
            exit(1);
         }
         UtilityFunction *new_func = new UtilityFunction(util_function_map[utility_str]);

         new_tree_node->set_utility_function(new_func);
         utility_read=true;
      }
   }
   // CHECK FOR THE PREVIOUSLY READ NODE
   if((!parent_read) && (!child_read)) {
      cout << "ERROR: Node " << curr_node_number << " has neither children or parent\n";
      exit(1);
   }
   if(!type_read) {
      cout << "ERROR: Node " << curr_node_number << " does not have a type specified\n";
      exit(1);
   }
   if(!utility_read) {
      if(new_tree_node->get_type()==OUTCOME_NODE) {
```

```
            cout << "ERROR: Utility not specified for Outcome Node " << curr_node_number <<
"\n";
            exit(1);
        }
    }
    if((curr_node_number+1)!=no_of_nodes) {
        cout << "ERROR: Number of nodes read does not match number of nodes specified\n";
        exit(1);
    }

    if(start_shell) { cout << "Number of tree nodes read " << (curr_node_number+1) <<
"\n"; }
    check_tree();

    tree_file.close();
}


/****************************************************************************/
/* This function runs some basic checks on the tree read in from user file */
/****************************************************************************/
void DecisionTree::check_tree(void) {

    // CHECK THE ROOT NODE
    if(tree_nodes.size()==0) {
        cout << "ERROR: Empty Tree\n";
        exit(1);
    }
    if(tree_nodes[0]->get_parent()!=-1) {
        cout << "ERROR: Node Zero is not root node\n";
        exit(1);
    }

    // CHECK THE TREE STRUCTURE
    for(unsigned int node_id=0; node_id<tree_nodes.size(); node_id++) {
        TreeNode *curr_tree_node = tree_nodes[node_id];
        if(curr_tree_node->get_id()!=node_id) {
            cout << "ERROR: Node id mismatch on " << node_id << "\n";
            exit(1);
        }

        // CHECK THE PARENT
        int parent_id=curr_tree_node->get_parent();
        if(parent_id!=-1) {
            TreeNode *parent_node = tree_nodes[parent_id];
            if(!parent_node->is_child(node_id)) {
                cout << "ERROR: Node " << node_id << " not parent's child\n";
                exit(1);
            }
        }

        // CHECK THE CHILDREN
        for(unsigned int ix=0; ix<curr_tree_node->get_no_of_children(); ix++) {
            int child_id = curr_tree_node->get_child(ix);
            TreeNode *child_node = tree_nodes[child_id];
            if(child_node->get_parent()!=node_id) {
                cout << "ERROR: Child's parent not current node (" << child_id << " " <<
node_id << ")\n";
                exit(1);
            }
        }

        // CHECK THE PROBABILITIES
        if(!curr_tree_node->check_probabilities()) {
            cout << "ERROR: Incorrect probabilities for chance node " << node_id << "\n";
            exit(1);
        }

        // CHECK THE UTIITIES
        if(!curr_tree_node->check_utility()) {
            cout << "ERROR: Incorrect utility for outcome node " << node_id << "\n";
```

```
          exit(1);
        }

    }
    if(start_shell) { cout << "Tree Check Complete Without Errors!\n"; }
}

// This function reads in a new set of probabilities for the chance node children
void DecisionTree::read_in_probabilities(string &prob_fname) {
  ifstream prob_file;
  prob_file.open(prob_fname.c_str(), ios::in);

  while(!prob_file.eof()) {
     string nextLine;
     std::getline(prob_file, nextLine);

     string::size_type loc = nextLine.find(",");
     if(loc == string::npos) continue;
     if((loc != string::npos) && (loc<1)) {
        cout << "ERROR: Illegal probability line " << nextLine << "\n";
        exit(1);
     }

     string node_number_str = nextLine.substr(0,loc);
     string prob_str = nextLine.substr(loc+1,nextLine.length());

     if(verbose) { cout << node_number_str << " " << prob_str << "\n"; }

     int node_id = atoi(node_number_str.c_str());
     if((node_id<0) || ((unsigned int)node_id>=tree_nodes.size())) {
        printf("ERROR: Illegal node id %d\n",node_id);
        exit(1);
     }
     float new_prob = atof(prob_str.c_str());
     if((new_prob<0) || (new_prob>1)) {
        printf("ERROR: Illegal probability value %f\n",new_prob);
        exit(1);
     }

     TreeNode *curr_node = tree_nodes[node_id];
     TreeNode *parent_node = tree_nodes[curr_node->get_parent()];
     if(parent_node->get_type()!=CHANCE_NODE) {
        printf("ERROR: Parent of node %d not chance node\n",node_id);
        exit(1);
     }
     bool found=parent_node->set_probability(node_id,new_prob);
     if(!found) {
        printf("ERROR: Probability cound not be set\n"); exit(1);
     }
     if(start_shell) { printf("Node: %d, New Probability = %f\n",node_id,new_prob); }
  }

  // RE-CHECK THE PROBABILITIES
  for(unsigned int ix=0; ix<tree_nodes.size(); ix++) {
     TreeNode *curr_tree_node = tree_nodes[ix];
      if(!curr_tree_node->check_probabilities()) {
         cout << "ERROR: Incorrect probabilities for chance node " << ix << "\n";
         exit(1);
      }
  }
  cout << "Probability Update Successful!\n";

  prob_file.close();
}


/*****************************************/
/* End of Decision tree class functions */
/*****************************************/
```

```
/**********************************/
/* Functions for Tree Node class */
/**********************************/

// This function evaluates a tree node
void TreeNode::evaluate_node(vector<TreeNode *> &nodes_list) {
   bool sum_performed=false;
   switch(type) {
      case OUTCOME_NODE:
         value = utility_func->get_normalized_value();
         if(verbose) printf("Outcome Node %d, %f\n",node_id, value);
         break;

      case CHANCE_NODE:
         if(children.size()==0) {
            printf("ERROR: Chance node %d has no children\n",node_id);
            exit(1);
         }
         value=0;
         sum_performed=false;
         for(unsigned int i=0; i<children.size(); i++) {
            if((children[i]<0) || ((unsigned int)children[i]>=nodes_list.size())) {
               printf("ERROR: Illegal child %d of node %d\n",children[i],node_id);
               exit(1);
            }
            float child_value = nodes_list[children[i]]->get_value();
            if(child_value==-1) {
               cout << "ERROR: Child value uninitialized\n";
               exit(1);
            }
            value += (child_value * probabilities[i]);
            sum_performed=true;
         }
         if(!sum_performed) {
            cout << "ERROR: In Chance node calculation\n";
            exit(1);
         }
         if(verbose) printf("Chance Node %d, %f\n",node_id, value);
         break;

      case DECISION_NODE:
         if(children.size()==0) {
            printf("Decision node %d has no children\n",node_id);
            exit(1);
         }
         value=-1000;
         for(unsigned int i=0; i<children.size(); i++) {
            if((children[i]<0) || ((unsigned int)children[i]>=nodes_list.size())) {
               printf("ERROR: Illegal child %d of node %d\n",children[i],node_id);
               exit(1);
            }
            float child_value = nodes_list[children[i]]->get_value();
            if(child_value==-1) {
               cout << "ERROR: Child value uninitialized\n";
               exit(1);
            }
            if(value<child_value) value=child_value;

            /* Debug - Print */
            /*if((verbose) || (!utility_print)) printf("Node(%d) = %f
",children[i],child_value);*/
            /* End */
            if((verbose) || (!utility_print)) printf("%f,",children[i],child_value);
            if(verbose) printf("\n");
         }
         if(value==-1000) {
            cout << "ERROR: In Decision node calculation\n";
            exit(1);
         }
         if(verbose) printf("Decision Node %d, %f\n",node_id, value);
         else if(!utility_print) printf("\n");
```

```
         break;
   }
}

/*************************************/
/* End Functions for Tree Node class */
/*************************************/


/****************************************/
/* Functions for AHP calculations class */
/****************************************/
AHPCalculations::AHPCalculations(string &wt_fname) {
   ahp_weights.clear();
   sensitivity_anal=false;

   // Read in the AHP weights from the user given file
   ifstream wt_file;
   wt_file.open(wt_fname.c_str(), ios::in);

   // KEYWORD STRINGS
   const string value_keywd = "Value: ";
   const string row_keywd = "Row: ";
   const string weights_keywd = "Weights: ";

   // READ IN ALL THE LINES
   vector<string> matrix_weight_strs;
   vector<vector<string> > matrix_row_strs;
   matrix_weight_strs.clear();
   matrix_weight_strs.resize(MAX_NO_OF_MATRICES);
   matrix_row_strs.clear();
   matrix_row_strs.resize(MAX_NO_OF_MATRICES);
   string::size_type loc;
   int weight_number=0;
   int matrix_number=0;
   while(!wt_file.eof()) {
      string nextLine;
      std::getline(wt_file, nextLine);

      // IGNORE COMMENT
      loc = nextLine.find("//");
      if(loc == 0) { continue; }

      // WEIGHT VALUE
      loc = nextLine.find(value_keywd);
      if((loc != string::npos) && (loc==0)) {
         string value_str = nextLine.substr(value_keywd.length(),nextLine.length());

         float value = atof(value_str.c_str());
         AHPWeight *new_wt = new AHPWeight(value,weight_number);
         ahp_weights.push_back(new_wt);
         weight_number++;
         if(verbose) cout << "Weight " << weight_number << " Value=" << value << "\n";
      }

      // WEIGHTS FOR MATRIX
      loc = nextLine.find(weights_keywd);
      if((loc != string::npos) && (loc==0)) {
         matrix_number++;
         if(matrix_number>=MAX_NO_OF_MATRICES) {
            printf("ERROR: Too many matrices in weights file\n");
            exit(1);
         }
         string weights_str = nextLine.substr(weights_keywd.length(),nextLine.length());
         matrix_weight_strs[matrix_number-1] = weights_str;
      }

      // MATRIX ROW
      loc = nextLine.find(row_keywd);
      if((loc != string::npos) && (loc==0)) {
```

```
        string row_str = nextLine.substr(row_keywd.length(),nextLine.length());
        matrix_row_strs[matrix_number-1].push_back(row_str);
      }
    }

    // CHECK THE WEIGHTS READ IN
    if(ahp_weights.size()==1) {
      printf("ERROR: A single AHP weight not allowed!\n");
      exit(1);
    }
    if(ahp_weights.size()==0) {
      printf("ERROR: No AHP weights read in\n");
      exit(1);
    }

    // DECODE THE MATRIX WEIGHTS
    user_pref_matrix_weights.resize(matrix_number);
    for(int mtx_ix=0; mtx_ix<matrix_number; mtx_ix++) {
      string weight_str = matrix_weight_strs[mtx_ix];

      loc=0;
      while(true) {
        loc = weight_str.find(", ");
        if(loc != string::npos) {
          string user_value_str = weight_str.substr(0,loc);
          int user_value = atoi(user_value_str.c_str());
          weight_str = weight_str.substr(loc+2,weight_str.length());
          user_pref_matrix_weights[mtx_ix].push_back(user_value);
        } else {
          int user_value = atoi(weight_str.c_str());
          user_pref_matrix_weights[mtx_ix].push_back(user_value);
          break;
        }
      }
    }
    for(int mtx_ix=0; mtx_ix<matrix_number; mtx_ix++) {
      for(int i=0; i<user_pref_matrix_weights[mtx_ix].size(); i++) {
        if(user_pref_matrix_weights[mtx_ix][i]>=ahp_weights.size()) {
          printf("ERROR: Matrix Weight Index %d larger than Number of Weights for
Matrix %d\n",user_pref_matrix_weights[mtx_ix][i],mtx_ix);
          exit(1);
        }
      }
    }


    // DECODE THE MATRIX VALUES
    user_pref_matrix.resize(matrix_number);
    for(int mtx_ix=0; mtx_ix<matrix_number; mtx_ix++) {
      if(matrix_row_strs[mtx_ix].size()!=user_pref_matrix_weights[mtx_ix].size()) {
        printf("ERROR: Number of matrix rows does not match Number of weights\n");
        exit(1);
      }

      user_pref_matrix[mtx_ix].resize(matrix_row_strs[mtx_ix].size());
      for(unsigned int row_ix=0; row_ix<matrix_row_strs[mtx_ix].size(); row_ix++) {
        string row_str = matrix_row_strs[mtx_ix][row_ix];

        loc=0;
        while(true) {
          loc = row_str.find(", ");
          if(loc != string::npos) {
            string user_value_str = row_str.substr(0,loc);
            float user_value = atof(user_value_str.c_str());
            row_str = row_str.substr(loc+2,row_str.length());
            user_pref_matrix[mtx_ix][row_ix].push_back(user_value);
          } else {
            float user_value = atof(row_str.c_str());
            user_pref_matrix[mtx_ix][row_ix].push_back(user_value);
            break;
          }
```

```
            }
            if(user_pref_matrix[mtx_ix][row_ix].size()!=user_pref_matrix[mtx_ix].size()) {
               printf("ERROR: Row size does not match column size for row %d\n",row_ix);
               exit(1);
            }
         }
      }
      if(verbose) {
         for(int mtx_ix=0; mtx_ix<matrix_number; mtx_ix++) {
            printf("Matrix Number %d\n",mtx_ix);

print_user_pref_matrix(user_pref_matrix_weights[mtx_ix],user_pref_matrix[mtx_ix]);
         }
      }

      // COMPELTE THE USER PREFERENCE MATRIX BASED ON THE SCALE SIZE
      complete_user_pref_matrix();
      if(verbose) {
         for(int mtx_ix=0; mtx_ix<matrix_number; mtx_ix++) {
            printf("Matrix Number %d\n",mtx_ix);

print_user_pref_matrix(user_pref_matrix_weights[mtx_ix],user_pref_matrix[mtx_ix]);
         }
      }


      // CHECK THE VALUES
      for(unsigned int i=0; i<user_pref_matrix_weights.size(); i++) {
         float sum=0;
         for(unsigned int j=0; j<user_pref_matrix_weights[i].size(); j++) {
            sum+=ahp_weights[user_pref_matrix_weights[i][j]]->get_value();
         }
         /* Debug */
         /*if((fabs(sum-1))>TOLERANCE) {
           cout << "ERROR: Weight values do not add up to 1\n";
           exit(1);
           }*/
         /* End */
      }

      wt_file.close();
}

// This function is the starting point of sensitivity analysis
void AHPCalculations::start_sensitivity_analysis(int wt_ix, float step) {
   if((unsigned int)wt_ix>=ahp_weights.size()) {
      cout << "ERROR: Illegal weight index " << wt_ix << " for sensitivity analysis\n";
      exit(1);
   }
   if(sensitivity_anal) {
      cout << "ERROR: Sensitivity analysis already started\n";
      exit(1);
   }

   // Sensitivity analysis always done on user weight values, not on calculated weight
values
   restore_user_value_for_all_weights();

   sensitivity_anal=true;
   weight_ix_being_analyzed=wt_ix;
   ahp_weights[wt_ix]->save_value();
   ahp_weights[wt_ix]->set_value(0);

   if((step>0.5) || (step<0.01)) {
      cout << "ERROR: Step " << step << " too large or too small\n";
      exit(1);
   }
   increment_step = step;
}

// This function determines the next incremental point in sensitivity analysis
```

```
bool AHPCalculations::increment_weight_being_analyzed(void) {
   bool ret_stat=false;
   if(!sensitivity_anal) {
      cout << "ERROR: Sensitivity analysis not started\n";
      exit(1);
   }

   float wt_val = ahp_weights[weight_ix_being_analyzed]->get_value();
   wt_val += increment_step;
   if(wt_val>=1.0) {
      sensitivity_anal=false;
      ahp_weights[weight_ix_being_analyzed]->restore_value();
      ret_stat = false;
   } else {
      printf("Weight %d, Value=%f: ",weight_ix_being_analyzed,wt_val);
      if(verbose) printf("\n");

      ahp_weights[weight_ix_being_analyzed]->set_value(wt_val);
      ret_stat = true;
   }
   return(ret_stat);
}

// This function determines AHP weights from user input preferences
void AHPCalculations::determine_AHP_weights_from_user_preferences_for_matrix(int mtx_ix)
{

   if(verbose) {
      printf("*****************(%d)*******************\n",mtx_ix);
      print_user_pref_matrix(user_pref_matrix_weights[mtx_ix],user_pref_matrix[mtx_ix]);
   }

   // (1) Determine the criteria ratios
   vector<vector <float> > calc_mtx;
   calc_mtx.resize(user_pref_matrix[mtx_ix].size());
   for(unsigned int row=0; row<user_pref_matrix[mtx_ix].size(); row++) {
      if(user_pref_matrix[mtx_ix].size()!=user_pref_matrix[mtx_ix][row].size()) {
         printf("ERROR: User preference matrix not square\n");
         exit(1);
      }

      calc_mtx[row].resize(user_pref_matrix[mtx_ix][row].size());
      for(unsigned int col=0; col<user_pref_matrix[mtx_ix][row].size(); col++) {
         calc_mtx[row][col] =
user_pref_matrix[mtx_ix][row][col]/user_pref_matrix[mtx_ix][col][row];
      }
   }
   if(verbose) {
      printf("****************************************\n");
      print_user_pref_matrix(user_pref_matrix_weights[mtx_ix],calc_mtx);
   }

   // (2) Determine the criteria relations in terms of base criteria
   //     (last criteria is chosen as base criteria
   for(unsigned int col=0; col<(calc_mtx.size()-1); col++)
      for(unsigned int row=0; row<calc_mtx.size(); row++) {
         calc_mtx[row][col] = calc_mtx[row][col]/calc_mtx[row][col+1];
   }

   if(verbose) {
      printf("****************************************\n");
      print_user_pref_matrix(user_pref_matrix_weights[mtx_ix],calc_mtx);
   }

   // (3) Determine mean relations and solve for criteria
   vector<float> avgs;
   avgs.resize(calc_mtx.size());
   avgs[calc_mtx.size()-1]=1.0;
   for(unsigned int col=0; col<(calc_mtx.size()-1); col++) {
      avgs[col]=0;
```

```
      for(unsigned int row=0; row<calc_mtx.size(); row++) avgs[col] +=
calc_mtx[row][col];
      avgs[col] = avgs[col]/(calc_mtx.size());
      if(verbose) {
         printf("%f ",avgs[col]);
      }
   }
   if(verbose) {
      printf("\n");
   }

   float normalization_sum=1;
   for(unsigned int col=(avgs.size()-1); col>0; col--) {
      avgs[col-1] = avgs[col]*avgs[col-1];
      normalization_sum += avgs[col-1];
      if(verbose) {
         printf("(%d x %d) %f\n",col,col-1,avgs[col-1]);
      }
   }
   if(verbose) {
      printf("Normalization Sum = %f\n",normalization_sum);
   }

   // (4) Determine the final weights
   for(unsigned int col=0; col<avgs.size(); col++) {
      avgs[col] = avgs[col]/normalization_sum;
      if(verbose) {
         printf("(Weigth %d) %f\n",col,avgs[col]);
      }
   }


   // (5) Store the calculated weight values
   if(avgs.size()!=user_pref_matrix_weights[mtx_ix].size()) {
      cout << "ERROR: Number of calculated weights not equal to weights in the matrix\n";
      exit(1);
   }
   for(unsigned int i=0; i<avgs.size(); i++) {
      if(user_pref_matrix_weights[mtx_ix][i]>=ahp_weights.size()) {
         cout << "ERROR: Matrix Weight Index greater than number of weights\n";
         exit(1);
      }
      ahp_weights[user_pref_matrix_weights[mtx_ix][i]]->set_value(avgs[i]);
      if(verbose) {
         printf("Setting Weight %d value to
%f\n",user_pref_matrix_weights[mtx_ix][i],avgs[i]);
      }
   }

   avgs.clear();
   calc_mtx.clear();

   if(verbose) {
      printf("********************END***************\n");
   }
}

// This function starts a sensitivity analysis on user input preferences
void AHPCalculations::start_user_preference_sensitivity_analysis(int mtx_ix, int wt_ix1,
int wt_ix2, float step) {
   if(verbose) {
      printf("\nStarting sensitivity analysis***********\n");
   }
   if(((unsigned int)wt_ix1>=ahp_weights.size()) || ((unsigned
int)wt_ix2>=ahp_weights.size())) {
      cout << "ERROR: Illegal criteria index " << wt_ix1 << " " << wt_ix2 << " for
sensitivity analysis\n";
      exit(1);
   }
   if((unsigned int)mtx_ix>=user_pref_matrix.size()) {
```

```
      cout << "ERROR: Illegal user preference matrix " << mtx_ix << " " << " for
sensitivity analysis\n";
      exit(1);
   }
   if(sensitivity_anal) {
      cout << "ERROR: Sensitivity analysis already started\n";
      exit(1);
   }
   sensitivity_anal=true;
   save_user_pref_matrix();

   if((step>50) || (step<1)) {
      cout << "ERROR: Step " << step << " too large or too small\n";
      exit(1);
   }
   increment_step = step;

   // FIND WHERE THE WEIGHT INDICES OCCUR IN THE USER PREFERENCE MATRIX
   int tmp_wt_ix1=-1;
   int tmp_wt_ix2=-1;
   for(unsigned int i=0; i<user_pref_matrix_weights[mtx_ix].size(); i++) {
      if(user_pref_matrix_weights[mtx_ix][i]==wt_ix1) tmp_wt_ix1=(int)i;
      if(user_pref_matrix_weights[mtx_ix][i]==wt_ix2) tmp_wt_ix2=(int)i;
   }
   if((tmp_wt_ix1==-1)||(tmp_wt_ix2==-1)) {
      cout << "ERROR: Weight indices " << wt_ix1 << ", " << wt_ix2 << " not found in user
preference matrix " << mtx_ix << "\n";
      exit(1);
   }
   wt_ix1=tmp_wt_ix1; wt_ix2=tmp_wt_ix2;

   // RESET THE USER PREFERENCE MATRIX FOR THE GIVEN WEIGHT INDICES
   if(wt_ix1>wt_ix2) {
      int temp_wt_ix1=wt_ix1;
      wt_ix1 = wt_ix2;
      wt_ix2 = temp_wt_ix1;
   }
   user_pref_matrix[mtx_ix][wt_ix1][wt_ix2]=5;
   complete_user_pref_matrix();
   user_wt_ix1_being_analyzed = wt_ix1;
   user_wt_ix2_being_analyzed = wt_ix2;
   user_mtx_ix_being_analyzed = mtx_ix;

   if(verbose) {
      print_user_pref_matrix(user_pref_matrix_weights[mtx_ix],user_pref_matrix[mtx_ix]);
      printf("***********************************************\n\n");
   }
}

// This function returns the net incremental point in the user input preferences
sensitivity analysis
bool AHPCalculations::increment_user_preference_being_analyzed(void) {
   bool ret_stat=false;
   int wt_ix1 = user_wt_ix1_being_analyzed;
   int wt_ix2 = user_wt_ix2_being_analyzed;
   int mtx_ix = user_mtx_ix_being_analyzed;
   if(!sensitivity_anal) {
      cout << "ERROR: Sensitivity analysis not started\n";
      exit(1);
   }

   float user_pref_wt_val = user_pref_matrix[mtx_ix][wt_ix1][wt_ix2];
   user_pref_wt_val += increment_step;
   if(user_pref_wt_val>=AHP_USER_SCALE-5) {
      sensitivity_anal=false;
      restore_user_pref_matrix();
      ret_stat=false;
   } else {
      /* Debug - Print */
      /*printf("User Preference (%d-vs-%d), Value=%f:
",wt_ix1,wt_ix2,user_pref_wt_val);*/
```

```
      /* End */
      printf("(%d-vs-%d),%f,",wt_ix1,wt_ix2,user_pref_wt_val);
      if(verbose) printf("\n");

      user_pref_matrix[mtx_ix][wt_ix1][wt_ix2] = user_pref_wt_val;
      complete_user_pref_matrix();
      if(verbose) {
         printf("Incremented Matrix\n");

print_user_pref_matrix(user_pref_matrix_weights[mtx_ix],user_pref_matrix[mtx_ix]);
         printf("End Incremented Matrix\n");
      }
      ret_stat=true;
   }

   return(ret_stat);
}

// This function starts a sensitivity analysis on user input preferences (a modified
version)
void AHPCalculations::start_user_preference_sensitivity_analysis_mod(int mtx_ix, int
wt_ix1, int wt_ix2, float step) {
   if(verbose_mod) {
      printf("\nStarting sensitivity analysis***********\n");
   }
   if(((unsigned int)wt_ix1>=ahp_weights.size()) || ((unsigned
int)wt_ix2>=ahp_weights.size())) {
      cout << "ERROR: Illegal criteria index " << wt_ix1 << " " << wt_ix2 << " for
sensitivity analysis\n";
      exit(1);
   }
   if((unsigned int)mtx_ix>=user_pref_matrix.size()) {
      cout << "ERROR: Illegal user preference matrix " << mtx_ix << " " << " for
sensitivity analysis\n";
      exit(1);
   }
   if(sensitivity_anal) {
      cout << "ERROR: Sensitivity analysis already started\n";
      exit(1);
   }
   sensitivity_anal=true;

   // Sensitivity analysis always done on user weight values, not on calculated weight
values
   restore_user_value_for_all_weights();
   user_wt_ix1_being_analyzed = wt_ix1;
   user_wt_ix2_being_analyzed = wt_ix2;
   user_mtx_ix_being_analyzed = mtx_ix;
   ahp_weights[wt_ix1]->save_value();
   ahp_weights[wt_ix1]->set_value(0.05);


   // FIND WHERE THE WEIGHT INDICES OCCUR IN THE USER PREFERENCE MATRIX
   int tmp_wt_ix1=-1;
   int tmp_wt_ix2=-1;
   for(unsigned int i=0; i<user_pref_matrix_weights[mtx_ix].size(); i++) {
      if(user_pref_matrix_weights[mtx_ix][i]==wt_ix1) tmp_wt_ix1=(int)i;
      if(user_pref_matrix_weights[mtx_ix][i]==wt_ix2) tmp_wt_ix2=(int)i;
   }
   if((tmp_wt_ix1==-1)||(tmp_wt_ix2==-1)) {
      cout << "ERROR: Weight indices " << wt_ix1 << ", " << wt_ix2 << " not found in user
preference matrix " << mtx_ix << "\n";
      exit(1);
   }

   float sum_value=0;
   for(unsigned int i=0; i<user_pref_matrix_weights[mtx_ix].size(); i++) {
      if(user_pref_matrix_weights[mtx_ix][i]!=wt_ix2)
         sum_value += ahp_weights[user_pref_matrix_weights[mtx_ix][i]]->get_value();
   }
```

```
    ahp_weights[wt_ix2]->save_value();
    ahp_weights[wt_ix2]->set_value(1-sum_value);

    if((step>0.5) || (step<0.01)) {
       cout << "ERROR: Step " << step << " too large or too small\n";
       exit(1);
    }
    increment_step = step;

    if(verbose_mod) {
       print_all_ahp_weights();
       printf("************************************************\n\n");
    }
}

// This function returns the net incremental point in the user input preferences
sensitivity analysis (modified version)
bool AHPCalculations::increment_user_preference_being_analyzed_mod(void) {
    bool ret_stat=false;
    int wt_ix1 = user_wt_ix1_being_analyzed;
    int wt_ix2 = user_wt_ix2_being_analyzed;
    int mtx_ix = user_mtx_ix_being_analyzed;
    if(!sensitivity_anal) {
       cout << "ERROR: Sensitivity analysis not started\n";
       exit(1);
    }

    float wt_val1 = ahp_weights[wt_ix1]->get_value();
    wt_val1 += increment_step;
    int int_wt_val1 = (int)floor(wt_val1*100);
    ahp_weights[wt_ix1]->set_value(wt_val1);
    float sum_value=0;
    for(unsigned int i=0; i<user_pref_matrix_weights[mtx_ix].size(); i++) {
       if(user_pref_matrix_weights[mtx_ix][i]!=wt_ix2)
          sum_value += ahp_weights[user_pref_matrix_weights[mtx_ix][i]]->get_value();
    }
    float wt_val2 = 1 - sum_value;
    ahp_weights[wt_ix2]->set_value(wt_val2);

    if((wt_val1>=0.95)||(wt_val2<0)) {
       sensitivity_anal=false;
       ahp_weights[wt_ix1]->restore_value();
       ahp_weights[wt_ix2]->restore_value();
       ret_stat=false;
    } else {

       printf("(%d-vs-%d),%d,",wt_ix1,wt_ix2,int_wt_val1);
       if(verbose_mod) printf("\n");

       if(verbose_mod) {
          printf("Incremented Weights\n");
          print_all_ahp_weights();
          printf("End Incremented Weights\n");
       }
       ret_stat=true;
    }

    return(ret_stat);
}


// This function performs sensitivity analysis for all possible user values for matrix
sizes of 3
void AHPCalculations::perform_matrix_size_3_sensitivity_analysis(int mtx_ix, DecisionTree
*dec_tree, UtilityCalculations *utility_calc) {
    save_user_pref_matrix();

    if(user_pref_matrix[mtx_ix].size()!=3) {
       printf("ERROR: Function cannot be used\n"); exit(1);
    }
```

```
   for(float val1=10; val1<=90; val1=val1+10) {
      for(float val2=10; val2<=90; val2=val2+10) {
         for(float val3=10; val3<=90; val3=val3+10) {

            // Fill in the matrix values
            user_pref_matrix[mtx_ix][0][1]=val1;
            user_pref_matrix[mtx_ix][0][2]=val2;
            user_pref_matrix[mtx_ix][1][2]=val3;
            complete_user_pref_matrix();

            if(verbose)
print_user_pref_matrix(user_pref_matrix_weights[mtx_ix],user_pref_matrix[mtx_ix]);


            // Evaluate the tree
            if(!verbose) printf("%f,%f,%f: ",val1,val2,val3);
            dec_tree->evaluate_tree(utility_calc,this);

         }
      }
   }


   restore_user_pref_matrix();
}

/*****####**********************************/
/* End Functions for AHP calculations class */
/**********###*******************************/

/**********************###*****************/
/* Functions for Utility calculations class */
/**************************###*************/

// Constructor
UtilityCalculations::UtilityCalculations(DecisionTree *dec_tree, AHPCalculations
*_ahp_calc_ptr) {
   utility_funcs.clear();
   ahp_calc_ptr = _ahp_calc_ptr;

   for(unsigned int i=0; i<dec_tree->tree_nodes.size(); i++) {
      TreeNode *curr_node = dec_tree->tree_nodes[i];
      if(curr_node->get_type()==OUTCOME_NODE)
         utility_funcs.push_back(curr_node->get_utility_function());
   }
}


// This function normalizes the values of utility functions
void UtilityCalculations::calculate_normalized_utilities(void) {
   float max_value=-1000;
   float min_value=1000;

   // Determine the maximum and minimum utility function values
   for(unsigned int i=0; i<utility_funcs.size(); i++) {
      UtilityFunction *utility_func = utility_funcs[i];
      float value = utility_func->get_unnormalized_value(ahp_calc_ptr);
      if(value<min_value) min_value=value;
      if(value>max_value) max_value=value;
   }
   if((max_value==-1000) || (min_value==1000) || (min_value>max_value)
      ||((max_value==0) && (min_value==0))) {
      cout << "ERROR: Maximum, Minimum values not found\n";
      exit(1);
   }

   // Normalize the values using maximum, minimum
   if(verbose) {
      printf("Max Value = %f, Min Value = %f\n",max_value,min_value);
   }
   for(unsigned int i=0; i<utility_funcs.size(); i++) {
```

```
      UtilityFunction *utility_func = utility_funcs[i];
      float value = utility_func->get_unnormalized_value(ahp_calc_ptr);
      float normalized_value=1;
      if(max_value!=min_value)
         normalized_value = (value - min_value)/(max_value - min_value);
      utility_func->set_normalized_value(normalized_value);
      if(verbose) {
         printf("Normalized Value = %f\n",normalized_value);
      }
   }
}

/*****####****************####*****************/
/* End Functions for Utility calculations class */
/*********####***************####**************/

/*********************************/
/* Experiment specific functions */
/*********************************/
void print_utilities(void) {
   if(decision_tree->tree_nodes.size()<=58) {
      printf("ERROR: Undefined use of print_utilities function\n");
      exit(1);
   }

   for(int ix=23; ix<=58; ix++) {
      if(decision_tree->tree_nodes[ix]->get_type()!=OUTCOME_NODE) {
         printf("ERROR: Undefined use of print_utilities function\n");
         exit(1);
      }

      float value1 = decision_tree->tree_nodes[ix]->get_value();
      printf("%f,",ix,value1);

   }
   printf("\n");
}


/****************************/
/* Generic Global Functions */
/****************************/

// Utility Functions
float utility0(vector<float> &wt) {
   float result=0;

   // Calculation: r = w0
   result = wt[0];

   if(verbose) {
      cout << "Utility0 " << result << "\n";
   }

   return(result);
}

// Utility Functions
float utility1(vector<float> &wt) {
   float result=0;

   // Calculation: r = w1
   result = wt[1];

   if(verbose) {
      cout << "Utility1 " << result << "\n";
   }

   return(result);
}
```

```
// Utility Functions
float utility2(vector<float> &wt) {
   float result=0;

   // Calculation: r = w2
   result = wt[2];

   if(verbose) {
      cout << "Utility2 " << result << "\n";
   }

   return(result);
}

// Utility Functions
float utility3(vector<float> &wt) {
   float result=0;

   // Calculation: r = w3
   result = wt[3];

   if(verbose) {
      cout << "Utility3 " << result << "\n";
   }

   return(result);
}

// W1
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W1(vector<float> &wt) {
   float result=0;

   // Calculation r = w1 + w2 + w3
   result = wt[0] + wt[1] + wt[2];

   if(verbose) {
      cout << "W1 " << result << "\n";
   }

   return(result);
}

// W2
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W2(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + w2 + w3
   result = (0-wt[0]) + wt[1] + wt[2];

   if(verbose) {
      cout << "W2 " << result << "\n";
   }

   return(result);
}

// W3
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W3(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + w2 + w3
   result = (0-wt[0]) + wt[1] + wt[2];

   if(verbose) {
      cout << "W3 " << result << "\n";
   }

   return(result);
```

```
}

// W4
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W4(vector<float> &wt) {
   float result=0;

   // Calculation r = w1 + w2 + (0-w3)
   result = wt[0] + wt[1] + (0-wt[2]);

   if(verbose) {
      cout << "W4 " << result << "\n";
   }

   return(result);
}

// W5
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W5(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + w2 + (0-w3)
   result = (0-wt[0]) + wt[1] + (0-wt[2]);

   if(verbose) {
      cout << "W5 " << result << "\n";
   }

   return(result);
}

// W6
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W6(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + w2 + (0-w3)
   result = (0-wt[0]) + wt[1] + (0-wt[2]);

   if(verbose) {
      cout << "W6 " << result << "\n";
   }

   return(result);
}

// W7
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W7(vector<float> &wt) {
   float result=0;

   // Calculation r = w1 + (0-(w2*w2.a)) + (w2*w2.b) + (w2*w2.c) + w3
   result = wt[0] + (0-(wt[1]*wt[3])) + (wt[1]*wt[4]) + (wt[1]*wt[5]) + wt[2];

   if(verbose) {
      cout << "W7 " << result << "\n";
   }

   return(result);
}

// W8
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W8(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (0-(w2*w2.a)) + (w2*w2.b) + (w2*w2.c) + w3
   result = (0-wt[0]) + (0-(wt[1]*wt[3])) + (wt[1]*wt[4]) + (wt[1]*wt[5]) + wt[2];

   if(verbose) {
```

```
      cout << "W8 " << result << "\n";
   }

   return(result);
}

// W9
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W9(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (0-(w2*w2.a)) + (w2*w2.b) + (w2*w2.c) + w3
   result = (0-wt[0]) + (0-(wt[1]*wt[3])) + (wt[1]*wt[4]) + (wt[1]*wt[5]) + wt[2];

   if(verbose) {
      cout << "W9 " << result << "\n";
   }

   return(result);
}

// W10
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W10(vector<float> &wt) {
   float result=0;

   // Calculation r = w1 + (0-(w2*w2.a)) + (w2*w2.b) + (w2*w2.c) + (0-w3)
   result = wt[0] + (0-(wt[1]*wt[3])) + (wt[1]*wt[4]) + (wt[1]*wt[5]) + (0-wt[2]);

   if(verbose) {
      cout << "W10 " << result << "\n";
   }

   return(result);
}

// W11
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W11(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (0-(w2*w2.a)) + (w2*w2.b) + (w2*w2.c) + (0-w3)
   result = (0-wt[0]) + (0-(wt[1]*wt[3])) + (wt[1]*wt[4]) + (wt[1]*wt[5]) + (0-wt[2]);

   if(verbose) {
      cout << "W11 " << result << "\n";
   }

   return(result);
}

// W12
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W12(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (0-(w2*w2.a)) + (w2*w2.b) + (w2*w2.c) + (0-w3)
   result = (0-wt[0]) + (0-(wt[1]*wt[3])) + (wt[1]*wt[4]) + (wt[1]*wt[5]) + (0-wt[2]);

   if(verbose) {
      cout << "W12 " << result << "\n";
   }

   return(result);
}

// W13
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W13(vector<float> &wt) {
   float result=0;
```

```
    // Calculation r = w1 + (0-(w2*w2.a)) + (0-(w2*w2.b)) + (w2*w2.c) + w3
    result = wt[0] + (0-(wt[1]*wt[3])) + (0-(wt[1]*wt[4])) + (wt[1]*wt[5]) + wt[2];

    if(verbose) {
       cout << "W13 " << result << "\n";
    }

    return(result);
}

// W14
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a    wt[4]=w2.b    wt[5]=w2.c
float W14(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (0-(w2*w2.a)) + (0-(w2*w2.b)) + (w2*w2.c) + w3
   result = (0-wt[0]) + (0-(wt[1]*wt[3])) + (0-(wt[1]*wt[4])) + (wt[1]*wt[5]) + wt[2];

   if(verbose) {
      cout << "W14 " << result << "\n";
   }

   return(result);
}


// W15
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a    wt[4]=w2.b    wt[5]=w2.c
float W15(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (0-(w2*w2.a)) + (0-(w2*w2.b)) + (w2*w2.c) + w3
   result = (0-wt[0]) + (0-(wt[1]*wt[3])) + (0-(wt[1]*wt[4])) + (wt[1]*wt[5]) + wt[2];

   if(verbose) {
      cout << "W15 " << result << "\n";
   }

   return(result);
}

// W16
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a    wt[4]=w2.b    wt[5]=w2.c
float W16(vector<float> &wt) {
   float result=0;

   // Calculation r = w1 + (w2*w2.a) + (0-(w2*w2.b)) + (w2*w2.c) + w3
   result = wt[0] + (wt[1]*wt[3]) + (0-(wt[1]*wt[4])) + (wt[1]*wt[5]) + wt[2];

   if(verbose) {
      cout << "W16 " << result << "\n";
   }

   return(result);
}

// W17
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a    wt[4]=w2.b    wt[5]=w2.c
float W17(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (w2*w2.a) + (0-(w2*w2.b)) + (w2*w2.c) + w3
   result = (0-wt[0]) + (wt[1]*wt[3]) + (0-(wt[1]*wt[4])) + (wt[1]*wt[5]) + wt[2];

   if(verbose) {
      cout << "W17 " << result << "\n";
   }

   return(result);
}
```

```
// W18
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W18(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (w2*w2.a) + (0-(w2*w2.b)) + (w2*w2.c) + w3
   result = (0-wt[0]) + (wt[1]*wt[3]) + (0-(wt[1]*wt[4])) + (wt[1]*wt[5]) + wt[2];

   if(verbose) {
      cout << "W18 " << result << "\n";
   }

   return(result);
}

// W19
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W19(vector<float> &wt) {
   float result=0;

   // Calculation r = w1 + (w2*w2.a) + (w2*w2.b) + (0-(w2*w2.c)) + w3
   result = wt[0] + (wt[1]*wt[3]) + (wt[1]*wt[4]) + (0-(wt[1]*wt[5])) + wt[2];

   if(verbose) {
      cout << "W19 " << result << "\n";
   }

   return(result);
}

// W20
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W20(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (w2*w2.a) + (w2*w2.b) + (0-(w2*w2.c)) + w3
   result = (0-wt[0]) + (wt[1]*wt[3]) + (wt[1]*wt[4]) + (0-(wt[1]*wt[5])) + wt[2];

   if(verbose) {
      cout << "W20 " << result << "\n";
   }

   return(result);
}

// W21
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W21(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (w2*w2.a) + (w2*w2.b) + (0-(w2*w2.c)) + w3
   result = (0-wt[0]) + (wt[1]*wt[3]) + (wt[1]*wt[4]) + (0-(wt[1]*wt[5])) + wt[2];

   if(verbose) {
      cout << "W21 " << result << "\n";
   }

   return(result);
}

// W22
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W22(vector<float> &wt) {
   float result=0;

   // Calculation r = w1 + (w2*w2.a) + (w2*w2.b) + (0-(w2*w2.c)) + (0-w3)
   result = wt[0] + (wt[1]*wt[3]) + (wt[1]*wt[4]) + (0-(wt[1]*wt[5])) + (0-wt[2]);

   if(verbose) {
      cout << "W22 " << result << "\n";
   }
```

```
   return(result);
}

// W23
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W23(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (w2*w2.a) + (w2*w2.b) + (0-(w2*w2.c)) + (0-w3)
   result = (0-wt[0]) + (wt[1]*wt[3]) + (wt[1]*wt[4]) + (0-(wt[1]*wt[5])) + (0-wt[2]);

   if(verbose) {
      cout << "W23 " << result << "\n";
   }

   return(result);
}

// W24
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W24(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (w2*w2.a) + (w2*w2.b) + (0-(w2*w2.c)) + (0-w3)
   result = (0-wt[0]) + (wt[1]*wt[3]) + (wt[1]*wt[4]) + (0-(wt[1]*wt[5])) + (0-wt[2]);

   if(verbose) {
      cout << "W24 " << result << "\n";
   }

   return(result);
}

// W25
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W25(vector<float> &wt) {
   float result=0;

   // Calculation r = w1 + (0-(w2*w2.a)) + (w2*w2.b) + (0-(w2*w2.c)) + w3
   result = wt[0] + (0-(wt[1]*wt[3])) + (wt[1]*wt[4]) + (0-(wt[1]*wt[5])) + wt[2];

   if(verbose) {
      cout << "W25 " << result << "\n";
   }

   return(result);
}

// W26
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W26(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (0-(w2*w2.a)) + (w2*w2.b) + (0-(w2*w2.c)) + w3
   result = (0-wt[0]) + (0-(wt[1]*wt[3])) + (wt[1]*wt[4]) + (0-(wt[1]*wt[5])) + wt[2];

   if(verbose) {
      cout << "W26 " << result << "\n";
   }

   return(result);
}

// W27
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W27(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (0-(w2*w2.a)) + (w2*w2.b) + (0-(w2*w2.c)) + w3
   result = (0-wt[0]) + (0-(wt[1]*wt[3])) + (wt[1]*wt[4]) + (0-(wt[1]*wt[5])) + wt[2];
```

```
   if(verbose) {
      cout << "W27 " << result << "\n";
   }

   return(result);
}

// W28
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W28(vector<float> &wt) {
   float result=0;

   // Calculation r = w1 + (0-(w2*w2.a)) + (0-(w2*w2.b)) + (0-(w2*w2.c)) + w3
   result = wt[0] + (0-(wt[1]*wt[3])) + (0-(wt[1]*wt[4])) + (0-(wt[1]*wt[5])) + wt[2];

   if(verbose) {
      cout << "W28 " << result << "\n";
   }

   return(result);
}

// W29
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W29(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (0-(w2*w2.a)) + (0-(w2*w2.b)) + (0-(w2*w2.c)) + w3
   result = (0-wt[0]) + (0-(wt[1]*wt[3])) + (0-(wt[1]*wt[4])) + (0-(wt[1]*wt[5])) +
wt[2];

   if(verbose) {
      cout << "W29 " << result << "\n";
   }

   return(result);
}

// W30
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W30(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (0-(w2*w2.a)) + (0-(w2*w2.b)) + (0-(w2*w2.c)) + w3
   result = (0-wt[0]) + (0-(wt[1]*wt[3])) + (0-(wt[1]*wt[4])) + (0-(wt[1]*wt[5])) +
wt[2];

   if(verbose) {
      cout << "W30 " << result << "\n";
   }

   return(result);
}

// W31
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a   wt[4]=w2.b   wt[5]=w2.c
float W31(vector<float> &wt) {
   float result=0;

   // Calculation r = w1 + (0-(w2*w2.a)) + (w2*w2.b) + (0-(w2*w2.c)) + (0-w3)
   result = wt[0] + (0-(wt[1]*wt[3])) + (wt[1]*wt[4]) + (0-(wt[1]*wt[5])) + (0-wt[2]);

   if(verbose) {
      cout << "W31 " << result << "\n";
   }

   return(result);
}

// W32
```

```
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a    wt[4]=w2.b    wt[5]=w2.c
float W32(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (0-(w2*w2.a)) + (w2*w2.b) + (0-(w2*w2.c)) + (0-w3)
   result = (0-wt[0]) + (0-(wt[1]*wt[3])) + (wt[1]*wt[4]) + (0-(wt[1]*wt[5])) + (0-
wt[2]);

   if(verbose) {
      cout << "W32 " << result << "\n";
   }

   return(result);
}

// W33
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a    wt[4]=w2.b    wt[5]=w2.c
float W33(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (0-(w2*w2.a)) + (w2*w2.b) + (0-(w2*w2.c)) + (0-w3)
   result = (0-wt[0]) + (0-(wt[1]*wt[3])) + (wt[1]*wt[4]) + (0-(wt[1]*wt[5])) + (0-
wt[2]);

   if(verbose) {
      cout << "W33 " << result << "\n";
   }

   return(result);
}

// W34
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a    wt[4]=w2.b    wt[5]=w2.c
float W34(vector<float> &wt) {
   float result=0;

   // Calculation r = w1 + (w2*w2.a) + (0-(w2*w2.b)) + (0-(w2*w2.c)) + w3
   result = wt[0] + (wt[1]*wt[3]) + (0-(wt[1]*wt[4])) + (0-(wt[1]*wt[5])) + wt[2];

   if(verbose) {
      cout << "W34 " << result << "\n";
   }

   return(result);
}

// W35
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a    wt[4]=w2.b    wt[5]=w2.c
float W35(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (w2*w2.a) + (0-(w2*w2.b)) + (0-(w2*w2.c)) + w3
   result = (0-wt[0]) + (wt[1]*wt[3]) + (0-(wt[1]*wt[4])) + (0-(wt[1]*wt[5])) + wt[2];

   if(verbose) {
      cout << "W35 " << result << "\n";
   }

   return(result);
}

// W36
// wt[0]=w1  wt[1]=w2   wt[2]=w3  wt[3]=w2.a    wt[4]=w2.b    wt[5]=w2.c
float W36(vector<float> &wt) {
   float result=0;

   // Calculation r = (0-w1) + (w2*w2.a) + (0-(w2*w2.b)) + (0-(w2*w2.c)) + w3
   result = (0-wt[0]) + (wt[1]*wt[3]) + (0-(wt[1]*wt[4])) + (0-(wt[1]*wt[5])) + wt[2];

   if(verbose) {
      cout << "W36 " << result << "\n";
```

```
    }

    return(result);
}


// This function create a mapping from a utility expression name to a utility function
that calculates that expression
void initialize_utility_function_map(void) {
    util_function_map.clear();
    util_function_map["utility0"] = utility0;
    util_function_map["utility1"] = utility1;
    util_function_map["utility2"] = utility2;
    util_function_map["utility3"] = utility3;

    util_function_map["W1"] = W1;
    util_function_map["W2"] = W2;
    util_function_map["W3"] = W3;
    util_function_map["W4"] = W4;
    util_function_map["W5"] = W5;
    util_function_map["W6"] = W6;
    util_function_map["W7"] = W7;
    util_function_map["W8"] = W8;
    util_function_map["W9"] = W9;
    util_function_map["W10"] = W10;
    util_function_map["W11"] = W11;
    util_function_map["W12"] = W12;
    util_function_map["W13"] = W13;
    util_function_map["W14"] = W14;
    util_function_map["W15"] = W15;
    util_function_map["W16"] = W16;
    util_function_map["W17"] = W17;
    util_function_map["W18"] = W18;
    util_function_map["W19"] = W19;
    util_function_map["W20"] = W20;
    util_function_map["W21"] = W21;
    util_function_map["W22"] = W22;
    util_function_map["W23"] = W23;
    util_function_map["W24"] = W24;
    util_function_map["W25"] = W25;
    util_function_map["W26"] = W26;
    util_function_map["W27"] = W27;
    util_function_map["W28"] = W28;
    util_function_map["W29"] = W29;
    util_function_map["W30"] = W30;
    util_function_map["W31"] = W31;
    util_function_map["W32"] = W32;
    util_function_map["W33"] = W33;
    util_function_map["W34"] = W34;
    util_function_map["W35"] = W35;
    util_function_map["W36"] = W36;

}

/********************************/
/* End Generic Global Functions */
/********************************/



/*****************************************************************/
/* This file contains all the data structures for decision trees */
/* 1/20/07                                                       */
/*                                                               */
/* Author: Poonam Sharma                                         */
/*****************************************************************/

#include <math.h>
#include <stdlib.h>
#include <stdio.h>
#include <string>
```

```
using std::string;
#include <vector>
using std::vector;
#include <list>
using std::list;
#include <fstream>
using std::ifstream;
using std::ios;
#include <map>
using std::multimap;
using std::map;
#include <iostream>
using std::cout;
using std::cin;
using std::cerr;
using std::endl;
#include <iomanip>
using std::setw;
using std::setfill;
using std::pair;

// Constant Definitions
#define TOLERANCE 0.005
#define AHP_USER_SCALE 100
#define MAX_NO_OF_MATRICES 10

// Class Declaration
class DecisionTree;
class TreeNode;
class AHPCalculations;
class AHPWeight;
class UtilityFunction;
class UtilityCalculations;
typedef float (* UtilFuncPtrType)(vector<float> &wt);

// Global Generic Functions
void initialize_utility_function_map(void);


// EXPERIMENT SPECIFIC FUNCTIONS
void print_utilities();

// Global Variables
DecisionTree *decision_tree=0;
AHPCalculations *ahp_calc=0;
UtilityCalculations *util_calc=0;
map<string, UtilFuncPtrType> util_function_map;
bool verbose=false;
bool utility_print=false;

// This class represents an AHP weight
class AHPWeight {
   protected:
      float value;
      float value_bkp;
      float user_defined_value;
      int wt_id;

   public:
      AHPWeight(float _value, int _wt_id) { value = _value; user_defined_value = _value;
wt_id = _wt_id; }
      ~AHPWeight(void) {}
      inline float get_value(void) { return(value); }
      inline void set_value(float _value) { value=_value; }
      inline int id(void) { return(wt_id); }
      inline void save_value(void) { value_bkp = value; }
      inline void restore_value(void) { value = value_bkp; }
      inline void restore_user_defined_value(void) { value = user_defined_value; }
};

// This class represents the calculation of AHP weights
```

```
class AHPCalculations {
   protected:
      vector<AHPWeight *> ahp_weights;
      vector<float> weight_values;
      bool sensitivity_anal;
      int weight_ix_being_analyzed;
      int user_wt_ix1_being_analyzed;
      int user_wt_ix2_being_analyzed;
      int user_mtx_ix_being_analyzed;
      float increment_step;
      vector<vector <int> > user_pref_matrix_weights;
      vector<vector<vector <float> > > user_pref_matrix;
      vector<vector<vector <float> > > bkp_user_pref_matrix;

   public:
      AHPCalculations(string &wt_fname);
      ~AHPCalculations(void) {
         for(unsigned int i=0; i<ahp_weights.size(); i++) { delete ahp_weights[i]; }
         ahp_weights.clear();
         user_pref_matrix.clear();
         bkp_user_pref_matrix.clear();
      }
      inline AHPWeight *get_weight(int ix) {
         if((ix<0) || (ix>=ahp_weights.size())) return(0);
         return(ahp_weights[ix]);
      }
      inline void restore_user_value_for_all_weights(void) {
         for(unsigned int ix=0; ix<ahp_weights.size(); ix++) ahp_weights[ix]-
>restore_user_defined_value();
      }

      void complete_user_pref_matrix(void) {
         for(unsigned int mtx=0; mtx<user_pref_matrix.size(); mtx++) {
            for(unsigned int row=0; row<user_pref_matrix[mtx].size(); row++) {
               for(unsigned int col=0; col<user_pref_matrix[mtx][row].size(); col++) {
                  if(row==col) user_pref_matrix[mtx][row][col]=1;
                  if(row>col) user_pref_matrix[mtx][row][col] = AHP_USER_SCALE -
user_pref_matrix[mtx][col][row];
               }
            }
         }
      }

      void start_sensitivity_analysis(int wt_ix, float step);
      bool increment_weight_being_analyzed(void);

      inline void determine_AHP_weights_from_user_preferences(void) {
         for(unsigned int mtx_ix=0; mtx_ix<user_pref_matrix.size(); mtx_ix++)
            determine_AHP_weights_from_user_preferences_for_matrix(mtx_ix);
      }
      void determine_AHP_weights_from_user_preferences_for_matrix(int mtx_ix);

      void start_user_preference_sensitivity_analysis(int mtx_ix, int wt_ix1, int wt_ix2,
float step);
      bool increment_user_preference_being_analyzed(void);

      void start_user_preference_sensitivity_analysis_mod(int mtx_ix, int wt_ix1, int
wt_ix2, float step);
      bool increment_user_preference_being_analyzed_mod(void);

      void perform_matrix_size_3_sensitivity_analysis(int mtx_ix, DecisionTree *dec_tree,
UtilityCalculations *utility_calc);

      inline void save_user_pref_matrix(void) {
         bkp_user_pref_matrix.clear();
         bkp_user_pref_matrix = user_pref_matrix;

         /*bkp_user_pref_matrix.resize(user_pref_matrix.size());
           for(unsigned int mtx=0; mtx<user_pref_matrix.size(); mtx++) {
           bkp_user_pref_matrix[mtx].resize(user_pref_matrix[mtx].size());
```

```
            for(unsigned int row=0; row<user_pref_matrix[mtx].size(); row++) {
            bkp_user_pref_matrix[mtx][row].resize(user_pref_matrix.size());
            for(unsigned int col=0; col<user_pref_matrix[row].size(); col++)
            bkp_user_pref_matrix[row][col] = user_pref_matrix[row][col];
            }*/
      }
      inline void restore_user_pref_matrix(void) {
         for(unsigned int mtx=0; mtx<user_pref_matrix.size(); mtx++) {
            for(unsigned int row=0; row<user_pref_matrix[mtx].size(); row++) {
               for(unsigned int col=0; col<user_pref_matrix[mtx][row].size(); col++)
                  user_pref_matrix[mtx][row][col] = bkp_user_pref_matrix[mtx][row][col];
            }
         }
         bkp_user_pref_matrix.clear();
      }

      void print_user_pref_matrix(vector <int> &mtrx_wts, vector<vector <float> > &mtrx)
{
         for(unsigned int row=0; row<mtrx_wts.size(); row++) printf("%d ",mtrx_wts[row]);
         printf("\n");

         for(unsigned int row=0; row<mtrx.size(); row++) {
            for(unsigned int col=0; col<mtrx[row].size(); col++) {
               printf("%f  ",mtrx[row][col]);
            }
            printf("\n");
         }
      }

      vector<float> &get_all_weight_values(void) {
         weight_values.clear();
         for(unsigned int i=0; i<ahp_weights.size(); i++)
weight_values.push_back(ahp_weights[i]->get_value());
         return(weight_values);
      }

      inline void print_all_ahp_weights(void) {
         for(unsigned int ix=0; ix<ahp_weights.size(); ix++) {
            float wt_value = ahp_weights[ix]->get_value();
            printf("Weight %d, Value = %f\n",ix,wt_value);
         }
      }
};

// This class represents a utility function in terms of AHP weights
class UtilityFunction {
   protected:
      UtilFuncPtrType util_func_ptr;
      float normalized_value;

   public:
      UtilityFunction(UtilFuncPtrType _util_func_ptr) { util_func_ptr = _util_func_ptr; }
      ~UtilityFunction(void) {}
      inline float get_unnormalized_value(AHPCalculations *ahp_calc_ptr) {
         float value = util_func_ptr(ahp_calc_ptr->get_all_weight_values());
         return(value);
      }
      inline void set_normalized_value(float _normalized_value) { normalized_value =
_normalized_value; }
      inline float get_normalized_value(void) { return(normalized_value); }
};


// This class represents the calculation of utility values from AHP weights
class UtilityCalculations {
   protected:
      vector<UtilityFunction *> utility_funcs;
      AHPCalculations *ahp_calc_ptr;

   public:
      UtilityCalculations(DecisionTree *dec_tree, AHPCalculations *_ahp_calc_ptr);
```

```
      ~UtilityCalculations(void) { utility_funcs.clear(); }
      void calculate_normalized_utilities();
};


// This class represents a node in the decision tree
enum Node_Type {
   OUTCOME_NODE = 0,
   CHANCE_NODE = 1,
   DECISION_NODE = 2
};
class TreeNode {
   protected:
      int node_id;
      int parent;
      vector<int> children;
      vector<float> probabilities;
      Node_Type type;
      float value;
      UtilityFunction *utility_func;

   public:
      TreeNode(int _node_id) {
         node_id = _node_id;
         children.clear();
         probabilities.clear();
         utility_func=0;
         value=-1;
         parent=-1;
      }
      ~TreeNode(void) { children.clear(); probabilities.clear(); delete utility_func; }
      inline int get_id(void) { return(node_id); }
      inline void set_parent(int _parent) { parent = _parent; }
      inline int get_parent(void) { return(parent); }
      inline void add_child(int _child, float _probability) { children.push_back(_child);
probabilities.push_back(_probability); }
      inline void set_type(Node_Type _type) { type = _type; }
      inline unsigned int get_no_of_children(void) { return(children.size()); }
      inline int get_child(unsigned int ix) {
         if(ix>=children.size()) return(-1); else return(children[ix]);
      }
      inline void reset_value(void) { value=-1; }
      inline float get_value(void) { return(value); }
      inline Node_Type get_type(void) { return(type); }
      inline int number_of_children(void) { return(children.size()); }
      inline bool is_child(int _child_id) {
         for(unsigned int ix=0; ix<children.size(); ix++) {
            if(children[ix]==_child_id) return(true);
         }
         return(false);
      }
      inline bool set_probability(int _child_id, float prob) {
         bool found=false;
         for(unsigned int ci=0; ((ci<children.size())&&(!found)); ci++) {
            if(children[ci] == _child_id) {
               probabilities[ci]=prob;
               found=true;
            }
         }
         return(found);
      }
      inline bool check_probabilities() {
         float sum=0;
         if(probabilities.size()!=children.size()) return(false);
         else {
            for(unsigned int ix=0; ix<probabilities.size(); ix++) sum+=probabilities[ix];
         }
         if(type==CHANCE_NODE) {
            if((fabs(sum-1))>TOLERANCE) return(false);
            else return(true);
         } else {
```

```
            if(sum!=0) return(false);
            else return(true);
         }
         return(false);
      }
      inline void set_utility_function(UtilityFunction *_utility_func) {
         if(utility_func!=0) {
            cout << "ERROR: Utility Function already set\n";
            exit(1);
         }
         utility_func = _utility_func;
      }
      inline UtilityFunction *get_utility_function(void) { return(utility_func); }
      inline bool check_utility(void) {
         if(type==OUTCOME_NODE) {
            if(utility_func==0) return(false);
         } else {
            if(utility_func!=0) return(false);
         }
         return(true);
      }

      void evaluate_node(vector<TreeNode *> &node_list);
};


// This class represents a decision tree
class DecisionTree {
   public:
      vector<TreeNode *> tree_nodes;

   public:
      DecisionTree(string &tree_fname);
      ~DecisionTree(void) {
         for(unsigned int i=0; i<tree_nodes.size(); i++) {
            TreeNode *node = tree_nodes[i];
            delete node;
            tree_nodes[i]=0;
         }
         tree_nodes.clear();
      }
      void check_tree(void);
      void reset_tree_node_values(void) { for(unsigned int i=0; i<tree_nodes.size(); i++)
tree_nodes[i]->reset_value(); }

      void evaluate_tree(UtilityCalculations *u_calc, AHPCalculations *a_calc) {
         if(a_calc!=0) a_calc->determine_AHP_weights_from_user_preferences();
         /* Debug */
         if(a_calc!=0) a_calc->print_all_ahp_weights();
         /* End */
         u_calc->calculate_normalized_utilities();
         reset_tree_node_values();
         for(unsigned int i=tree_nodes.size(); i>0; i--)
            tree_nodes[i-1]->evaluate_node(tree_nodes);

         // EXPERIMENT SPECIFIC CODE
         if(utility_print) print_utilities();
      }

      void read_in_probabilities(string &prob_fname);
};
```