Pomona College
Department of Computer Science

Do it! - The Creation of a Decision Making App

Siyao Xie

May 5, 2014

Submitted as part of the senior exercise for the degree of
Bachelor of Arts in Computer Science

Professor Patrick McNally, advisor

Abstract

Making an important decision can be very difficult, especially when the decision has many dimensions of factors that people have to consider and compare. To help decision makers decide more rationally and confidently, an iOS app is built based on theories and practices of decision making. This app utilizes the Pros and Cons to gather information about each alternative and uses Pairwise Comparisons to quantify how important each Factor means to the decision maker. This approach breaks the multi-dimensional problem down into many one-dimensional decisions. Information about the importance of each Factor is extracted and the final decision suggestion is based on the weighted importance of all of the Factors. An User Testing Study was conducted to evaluate the effectiveness of the algorithm and to gather feedbacks for the app. This project explores how decisions can be made rationally based on simple and intuitive user input and how the decision making experience can be improved with a mobile app.

Acknowledgments

I would like to thank my project advisor Professor Patrick McNally for introducing me to the iOS development world and always pointing me to the right directions.

Also special thanks to my advisor Professor Tzu-Yi Chen for her support and patience throughout my time as a CS major, and Professor America Chambers for being excited about the project and answering my questions about decision making and data handling.

Thank you to my wonderful and indecisive friends for the inspiration of the project and also to my friends who help me bounce off ideas and tolerate me talking about decision making for two semesters.

Lastly, I would like to thank those who help me test the app. Thank you for your valuable feedback and your trust in me and my app for your important decisions.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

To be or not to be, that is a difficult decision. Even though our lives might not be as dramatic as Hamlets, we frequently run into difficult decisions too. Which school should I go to? Which company should I work for? Which city should I live in? These decisions are very important. Choosing one way or the other might have a very big impact and serious consequences. These decisions are very complex since they require lots of considerations and weighing of the benefits and costs. Therefore, making a decision rationally and confidently is a problem many people struggle with.

One traditional and intuitive method of decision making is listing Pros and Cons. Simply listing the Pros and Cons already helps decision makers to think about all aspects of the problem and have a holistic view of the decision. [1] And If the decision makers already have a preference, they can probably make a decision at this step. But for decisions that do not have a dominating choice, more sophisticated methods need to be used to actually weigh the Pros and Cons to determine which Choice is the optimal.

There have been many studies and theories on decision making methods, some have been implemented and primarily used for decision making in organizations to prioritize tasks and determine business strategies. However, there is no sufficient and easy to use tool for individuals to make personal big decisions. People still make decisions without tools and often struggle with indecisiveness. Therefore, the goal of the project is to explore the theories and practices of decision making and build an iOS app that help people make important decisions. The app, Do it!, aims to help users think of decisions in an intuitive way yet to provide a rational decision recommendation.

Do it! takes users through a decision making process that includes entering Pros and Cons, doing pairwise comparisons of the Pros and Cons where

users are asked to decide which Factor they care more. And the app will run in the background to compute how important each Factor is to the users and make a recommendation based on the weighted sum. The decision making process involves users in ways that they are already familiar and collects user input by asking questions that have intuitive answers and outputs a rational decision result that takes into consideration of all of the Pros and Cons and their importances.

Chapter 2

Background

2.1 Decision Analysis Models and Methods

Decision Analysis was a term first coined by Ronald A, Howard, a professor at Stanford University in 1964 [2]. Ever since then, a lot of research has been devoted to understanding and modeling how people make decisions as well as to developing a system to help decision makers to reduce complex decisions into simpler ones. According to Howard, there are three steps in the Decision Analysis process: The first step is to formulate a decision basis with regard to the decision maker's situation. The basis consists of three parts: the choices or alternatives the decision-maker faces, the information that is relevant, and the preferences of the decision-maker. The second step is to evaluate the basis with a primarily computational process to recommend an alternative. Lastly, there is an appraisal step that checks with the decision maker if the recommendation is good or the decision basis should be refined [3]. Among the three steps, the second one where evaluation with a computational process takes place is the most studied. Below is a brief overview of some of the most important decision theories and models of the last half-century:

Expected Utility Theory predated Decision Analysis. Daniel Bernoulli first introduced this idea in 1738 as the moral expectation [4], and it is further refined and supported by von Neumann Morgenstern utility theorem in 1947 [5]. This theory is widely adapted in economics and it sufficiently fills the gap of Expected Value Theory where only the values and probabilities of the payouts are multiplied and summed up: it takes into account the decision makers expectation and valuation of the alternatives.

Some researchers have used Cumulative Prospect Theory, which builds

on top of Prospect Theory, to model the decision problems. Cumulative Prospect Theory and Prospect Theory are behavioral economic theories that describe how people make choices between alternatives that involve risks. Kahneman, Daniel, and Amos Tversky believe that people make decisions based on potential values of losses and gains. An order of these outcomes can be obtained by setting an outcome as a reference point and consider the lower outcomes as losses and the higher ones as gains. They provide a formula to calculate expected utility using the values of losses and gains and their according probability [6]. In addition, CPT also allows different weighting functions for the gains and losses [7]. Similar to the traditional Expected Utility Theory, the decision maker wants the alternative that maximizes the expected utility. However, neither theory is directly applicable to decision problems that have gains and losses that cannot be quantified as easily as monetary gains and losses. But its implication is worth considering adapting when calculating the values of gains and losses translated from descriptive decision basis.

Decision Analysis is taught at many universities and MBA programs to educate students how to structure decision problems, develop decision options, quantify uncertainty and preferences and arrive at optimal decisions with the above information [8]. Decision Analysis has also been adopted by organizations and companies in fields like finance, agriculture, automobile and etc. [9] [10]. There have also been efforts in developing computer programs for Intelligent Decision System to help organizations and individuals to make complex decisions [11].

However, despite the usefulness of decision analysis, not that many decision-makers are benefiting from the full power of decision analysis. Howard, in his article *Decision Analysis: Practice and Promise*, mentioned some possible reasons: Decision Analysis requires decision makers to think in a systematic and logical way. This cognitive style is not natural to every decision maker, especially those who make decisions based on feelings rather than thoughts. It is also very hard to communicate and elicit the ideas, feelings and thoughts about a decision to properly construct a decision basis, especially when there is no decision analysis expert to help guide the process [3].

Thus, in order for decision analysis to be more widely adopted by average users and be used in everyday decision making, there needs to be a way to formulate decision makers thoughts and feelings into a decision basis. To reduce the training overhead, the user input experience should be as natural and familiar to the decision makers as possible, therefore the Pros and Cons List is chosen as a bridge from the originally complex situation

in the decision makers mind to a clear decision basis. Pros and Cons list includes information that a decision basis needs, like deciding factors and preferences about each alternative.

Many researchers have proposed techniques and heuristics to evaluate pros and cons in decision making. Dubois, Fargier and Bonnefon believe that the decision process is qualitative and bipolar (positive and negative features) [12]. And humans often make decision base on an ordinal ranking of the pros and cons. They propose a framework for qualitative and bipolar decision making which categorizes features of each option as negative, neutral or positive, and only assign ordinal importance ($++$, $+$, $--$, $-$) to each feature. They define many decision rules to make a decision under such framework, such as canceling pros and cons of the same importance level, comparing the number of pros of the same importance level and the cons of the same importance level. However, their framework is very restricted and is only capable of handling simple qualitative decisions .

On the other end, researchers are also studying ways to quantify decisions and extract weights of the pros and cons from decision makers. Stillwell, Seaver and Edwards discuss about five weighting techniques: Equal, Ratio, Rank Sum, Rank Reciprocal and Rank Exponent [13]. Equal weighting and Ratio weighting are at the extreme ends of attribute weighting. Equal weights simply means all attributes are equal, thus not very accurate. And Ratio weighting is giving each attribute an exact ratio of importance, which is hard to obtain. The other three weighting techniques fall in between these two extreme cases and utilizes ranking of the attributes. Rank Sum weights each attribute in the opposite order. For example, attributes A, B, C are ranked in descending order, therefore their weights are 3, 2, 1 (after normalization 0.5, 0.33 and 0.17) respectively. Rank Reciprocal takes the reciprocal of the ranking and Rank Exponent is similar to Rank Sum, but with an additional exponent value determined by the weight of the most important attribute in an iterative process. The last three weighting techniques are reasonable and yield improvement with respect to Equal Weighting. One caveat of the Rank weighting techniques is that they require users to rank the pros and cons first, which makes the assumption that decision makers can express their feelings in quantitative and orderly fashion. And ranking also may not reflect the attributes' true importances. For example, A is much more important than B and B is only slightly more important than C, but in the ranking order A, B, C, the difference between A and B is the same as between B and C.

Another method to translate decision makers' feelings into quantifiable values for the computational and logical evaluating process is Pairwise Com-

parison. Pairwise Comparison is first introduced as a method of scientific measurement by prominent psychometrician L. L. Thurstone. Rather than measuring the absolute properties of objects, Pairwise Comparison measures how we perceive the objects in relation to one another in pairs. It can be used to scale a collection of objects based on simple comparisons of two objects at a time [14]. This technique can help decision makers assign an importance value to each attribute more accurately.

2.2 Reviews of Relevant Applications

Professor Ali Abbas is a professor at UIUC teaching Decision Analysis. He recently created Ahoona, a web app that helps people make decision [15]. He provided four different analyses: 1. Pros & Cons Analysis, which filters out bad alternatives if there are more cons than pros for this alternative, 2. Weight & Rate Analysis, which weight and rate how each alternative satisfies the preferences or the goals, 3. Decision Tree Analysis, which helps decisions that have multiple uncertainties, 4. Advanced Decision Tree Analysis, which is for even more complicated cases when there are multiple decisions and multiple uncertainties. His work demonstrates that it is definitely possible to create simple application that helps people make better decisions. However, currently this web app is not very user friendly. Ahoona is built by academic personas who are very familiar with all sorts of decision analysis methods and phrases. But their explanations and guidance on how to use the tool is not very clear to average users. Also currently the process of inputting information is very repetitive. For example, the pros and cons are very closely related and similar to preferences, but the similarity is not utilized to make the process easier.

inDecision is an iOS app developed by Rad Rhino in 2011 [16]. It has a similar concept in that it asks the decision-maker to enter the pros and cons and uses the weights of the pros and cons to compute the final score to make recommendation. But there are some big limitations in this app that might create difficulties for the decision makers: first of all, this app is only designed to handle decisions that have Yes or No as the alternatives. For example, one can only decide If I should study abroad next year? but not Should I work at Google or Facebook?. Another caveat of the app is that decision makers have to decide on the weight of each pros and cons with no relative scale. Decision makers have to directly translate their feelings into a scale of importance, which is difficult and counter-intuitive as discussed in the previous section.

Chapter 3

Methods

3.1 Application Overview

The app consists of six different screens: Decision List, Decision Creation, Pros and Cons, Comparisons, Decision Result and Result Analysis.

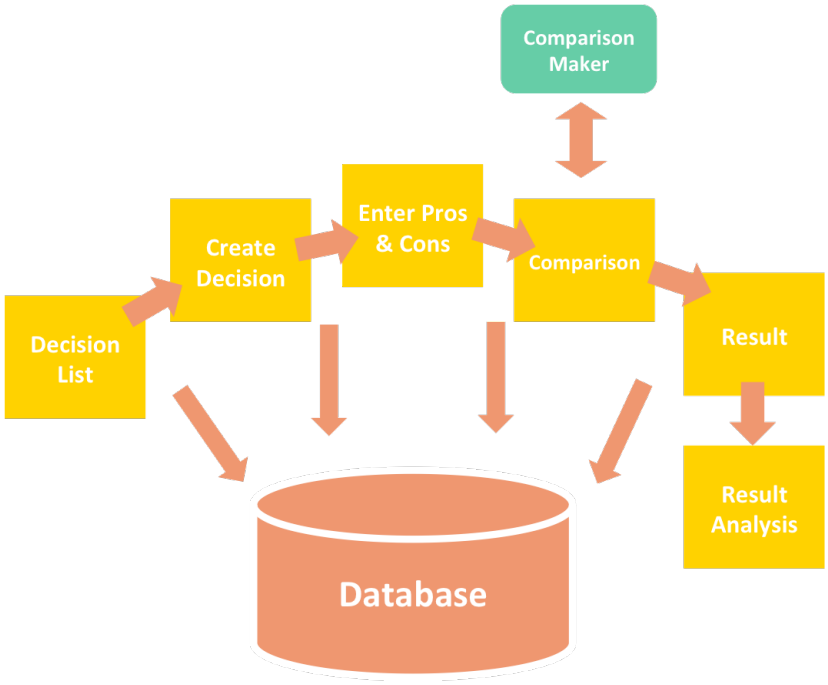

Figure 3.1: App Flow

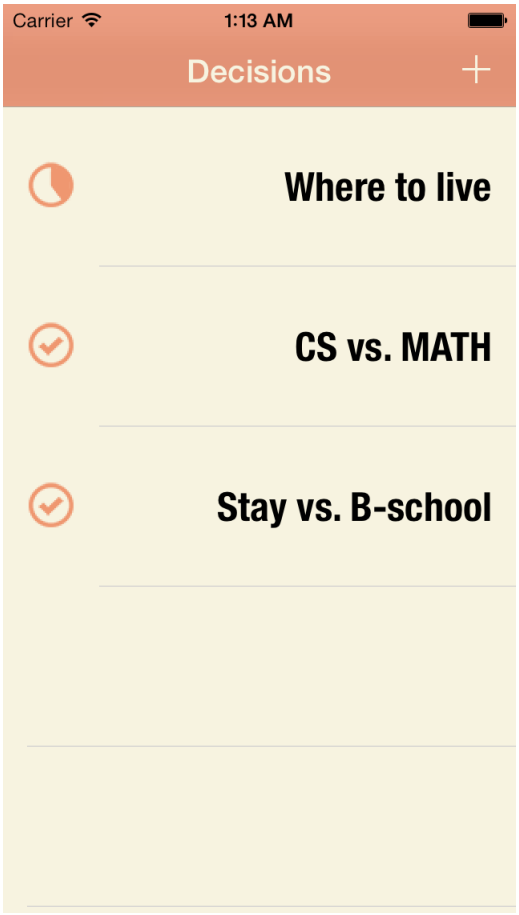

Figure 3.2: Decision List

Decision List is the view presented at launch. It is a list of past decisions the user made with the app. For each decision, there is an icon showing the current stage of the decision, each corresponding to the Pros and Cons, Comparison, and Decision Result screens, depending on where the decision was left off last time. Tapping on any of the past decision will take users to the corresponding screen so they can review the result or continue to make the decision.

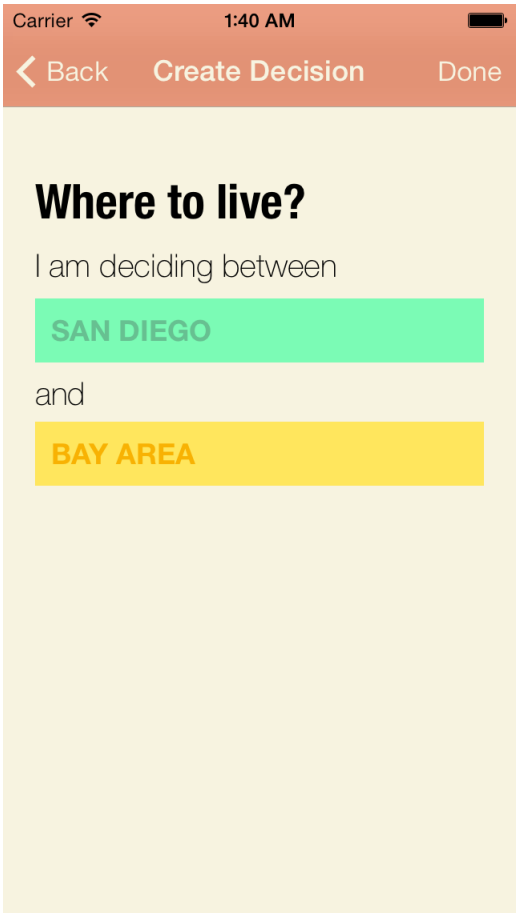

Figure 3.3: Decision Creation

Pressing the Add Button on the top right of the navigation panel will present users the Decision Creation screen. This view asks users to input the two choices they are deciding between in the green and yellow text-field accordingly. Users can also choose to specify a decision title. If the decision title is left blank, the app will generate a title using the two choices as Choice A vs. Choice B to help users identify this decision later.

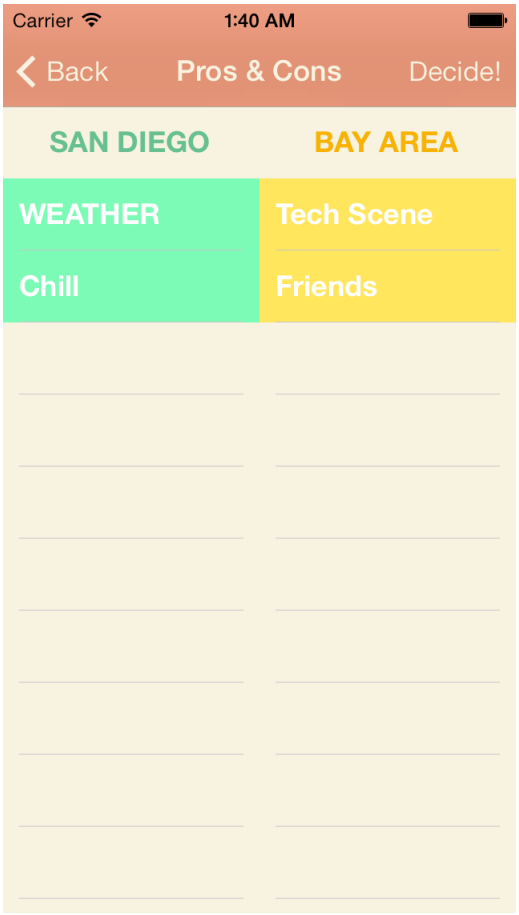

Figure 3.4: Pros and Cons

Pressing the Done Button on the top right corner of the navigation panel will take users to the Pros and Cons Screen. This is the place where users will take some time to think about what they like and not like about each of the choices. Upon arrival of the screen, there will be two blank lists on each side under each choice. To enter a Factor (Pro or Con), users can tap on the list or on the choice label. Users will then type in the factor on the popped up keyboard and select whether it is a pro or a con.

Figure 3.5: Comparisons

After the users feel they have entered enough Pros and Cons for the two Choices, pressing the Decide Button will take tthem to the Comparisons screen. This screen displays a pair of Factors, one from each choice in the form of two bubbles with the Factor labels attached. The bubbles also show the current weight users assign to each one of the Factors, with the total weight being 100. Users need to decide which Factor matters more and tap on the corresponding bubble to adjust the Factors' relative weights. Once the weights match users' expectation, user can press the Confirm Button to go to the next pair of Comparison. There is a progress bar near the top showing users how far they are in the process.

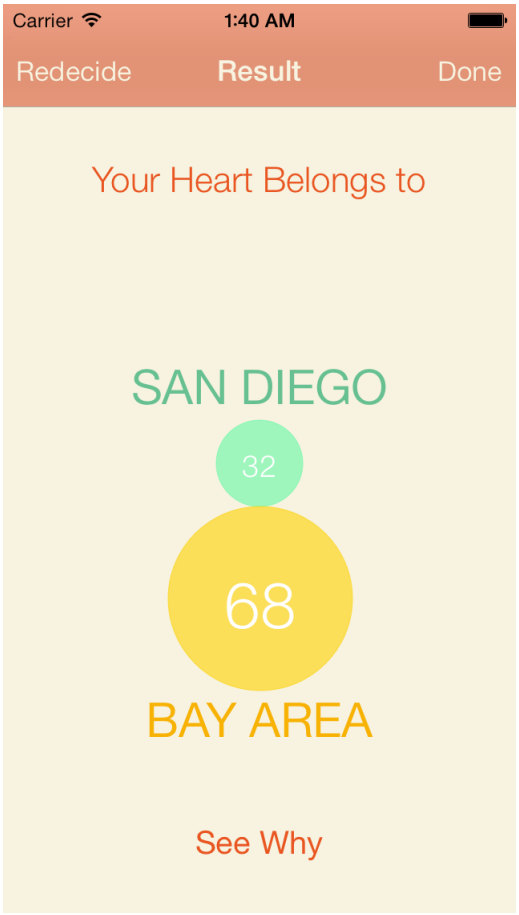

Figure 3.6: Decision Result

When enough pairs (discussed in Section 3.2.3) are compared, a dialog box will pop up and notify users that the decision is ready. Pressing See Result Button will take user to the Decision Result Screen. This screen displays the result in a similar format as the Comparisons screen. Each choice gets a percentage score that indicates the users preference. The one with higher score and bigger bubble is the app's recommendation.

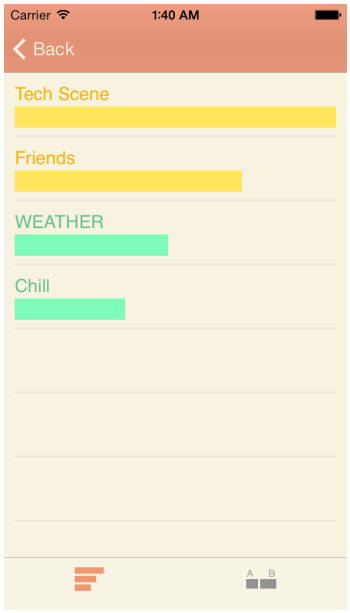

Figure 3.7: Factor Ranking

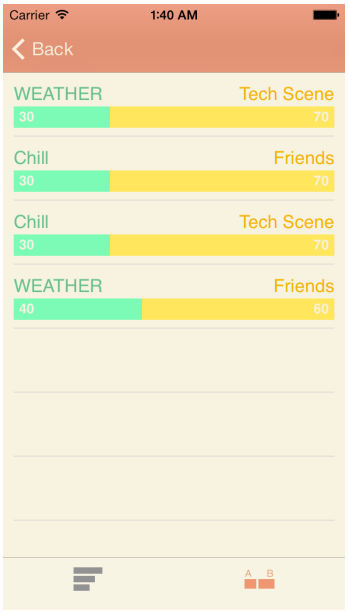

Figure 3.8: Comparison History

Users can press the See Result Button to go to the Result Analysis screen. This screen has two components. The left tab shows a ranking of the Factors from the most important to the least important and the right tab shows a history of all the Comparisons the user made for the Decision. The ranking shows users what the deciding Factors are and also help them quantify how much they care about the Factors. To visualize the ranking, each Factor is accompanied by a rectangle bar which shows the importance graphically. The same color coding is used to reflect which Choice the Factor belongs to and whether it is a Pro or a Con.

3.2 Implementation

3.2.1 User Experience Design

The ultimate goal of this app is to simplify and rationalize the users decision-making process. One way to make an app easy and intuitive to learn and use is to make the user experience mimic one that the user is already familiar with [17]. Since many decision-makers already use Pros and Cons to help them with the decision-making process, users are already familiar with the experience of coming up Pros and Cons. To make the experience of entering

Pros and Cons also familiar, the screen is divided into two halves, each titled with a Choice label and has a list of Pros and Cons for the corresponding Choice. Users can tap on the list to enter or edit a Factor, which is similar to writing notes on the built-in Notes application on iOS that most targeted users of the app have used.

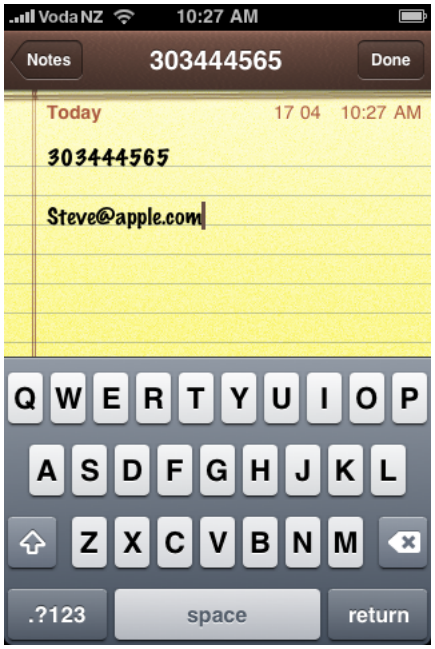

Figure 3.9: Notes in iOS 6 and before

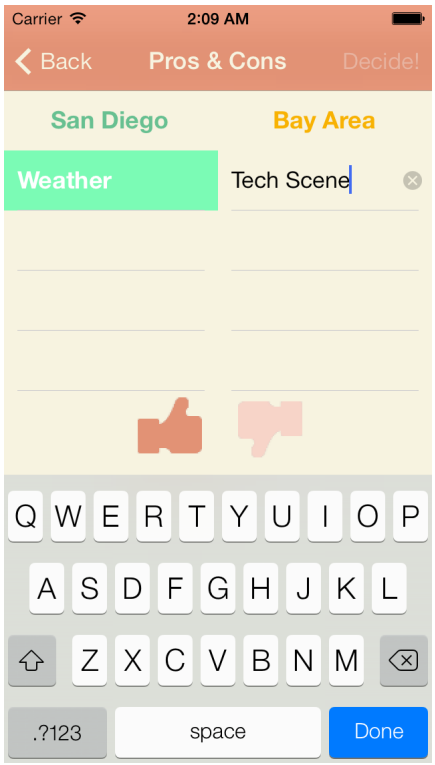

Figure 3.10: Inputting Pros and Cons

One difficult design decision to make is whether to keep the Pros and Cons of both Choices on the same screen. If the Factors are on the same screen, the users can refer back to what has been entered for both Choices when they think about new ones. And it also makes adding or editing factors of both choices smoother and easier. Otherwise, having to switch back and forth between the two Choices lists will distract users from the actual task of coming up and entering Factors. But displaying and entering the large amount of information on one small iPhone screen creates some problems too: 1. There is no space to have separate Pros and Cons columns for each choice. 2. There is not enough space to display the full text of long Pros

and Cons if they exceed a width limit of 160 points (12 letters during input mode and 14 letters during display).

In the current version, I choose to have both Choices Pros and Cons lists on the same screen because I want the process of entering factors be as simple and fast as possible and eliminate any redundant operations that might distract users from their main task.

Similar design decision was made for the Comparisons screen. To make sure users focus on the comparison of the current pair of Factors without considering other Factors and the rest of the Decision, I display only one pair of Factors at a time, as opposed to displaying a list of pair-wise comparisons all at once and have users go down the list to compare each one. The latter might make the comparison process go faster, but users will not be as focused and careful. And the fact that users can see and modify all past and future comparisons will likely have an influence on how they compare the current pair.

As the number of Factors grows, the number of Comparisons also grows linearly. To encourage users to continue and complete the task to reach their decisions, a completeness meter is necessary. A progress bar has become the common practice of a completeness meter in numerous software applications. Myers found that people prefer to have percent-done progress indicators through experiments [18]. Users are more patient because they are clear what the goal is (finishing the Comparisons) and how far it is until achieving the goal. Even though no controlled experiment is performed to test the effect of the progress bar specifically, in the first round alpha-testing I performed with my friends, a common reaction is the lack of progress indication. People were not sure how many more Comparisons they had to do and hence complained about having to do too many Comparisons. However, after the progress bar is added, no one in the second official testing had similar complaints. And even when asked about whether the process felt too long, peoples reactions were No and It went by faster than I thought.

3.2.2 App Architecture

The underlying architecture of the app consists of 1 SQLite Database, 5 object classes, 7 ViewControllers and a number of customized views.

Classes and Objects

The essential information that is created, modified, passed around and stored in the application is the Decision object. A Decision object keeps

track of information about the decisions result, as well as about the decisions current status: stage, round and number of Comparisons that have been compared so far. The Decision class also has methods that resets the statistics of the Decision and computes the scores for the Decisions Factors and its final result.

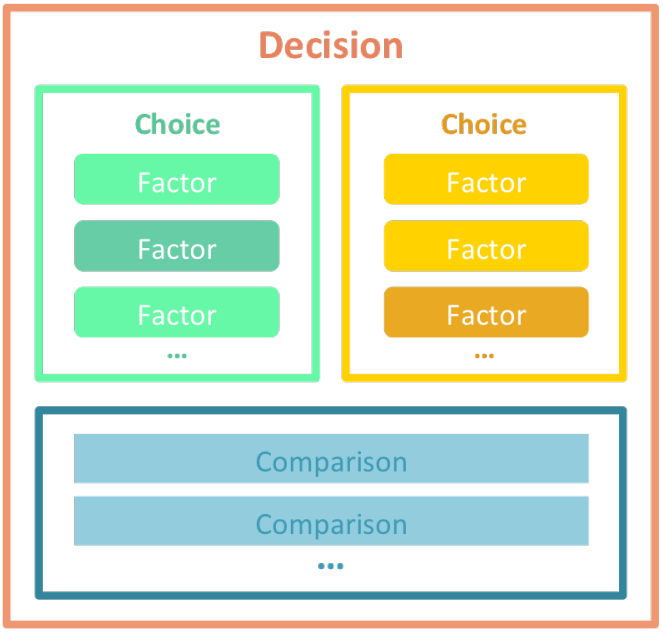

Figure 3.11: Decision Object

A Decision also contains two Choice objects and a list of Comparison objects that are generated by a ComparisonMaker object. A Choice object is much simpler: it contains the title of the Choice and a list of Factor Objects. A Factor is more complicated in that it contains its title and whether it is a Pro or a Con. And in order to keep tracks of the list of opponent Factors that it has been compared with and the list of weights it receives in the Comparisons. There are also other properties such as average weight and score that are relevant to score keeping and computation in a Factor object. A Comparison object contains two Factor objects, one from each Choice, along with the weights they received in the Comparison. Choice, Factor and Comparison all have resetStats methods that are called by the Decision when it needs to be reset.

An object class separate from the Decision hierarchy is the ComparisonMaker. Given a Decision, it generates sets of unique Comparisons at each

round of comparison based on a specified rule: e.g. compare with opponent Factor that has a similar current weight.

Database

Data that needs to persist between each use of the app needs to be saved in a Database. This app uses an SQLite Database to store all of the information about the decisions made and in the process of making. SQLite is chosen because it can be easily integrated with iOS and it is one of the most popular Databases so this database can be used by applications on other platforms if I decide to make this app cross-platform.

An SQLite database can store the basic datatypes such as INTEGER, TEXT, REAL and BLOB, which is just a blob of data, as how it was inputted. Since a Decision is an object that contains other objects, it can not be simply stored using the existing datatype. Using the NSKeyArchiver and NSKeyUnarchiver classes, the Decision objects can be turned into a blob of data which can be stored in the Database. When a NSKeyArchiver archives a Decision, the encodeWithCoder method in the Decision class is called to encode all of the primitive objects and called the non-primitive objects corresponding endCodeWithCoder method to encode themselves to prepare for an archival. Similarly in the unarchival process, the decodeWithCoder method is called through the layers of objects to decode the data and create a Decision. Therefore the encoding and decoding methods are needed for all of the customized objects including Decision, Choice, Factor and Comparison.

3.2.3 Pairwise Comparison

It is natural and easy for decision makers to come up with a list of Pros and Cons, but it is difficult for them to say specifically how important each of these factors is to them [12]. People can give a qualitative assessment of the importance of a pro or a con by stating: "I care about A more than B" or "C is very important to me", but stating "A is at an importance level of 55 and B is at 50, whereas C is at 90" is much harder and it is almost impossible to be accurate.

Pairwise Comparison is an approach to quantify the importance of each Pro and Con with only the information from a qualitative assessment as asking questions like "Which factor do you care more?". This approach helps break down the multi-dimensional decision into many one-dimensional questions that are easy to answer. And by doing the ComparisonS, users provide

information about relative importance between the two Factors which can be later collected and collated to explore the absolute importance of each Factor.

Comparison Generation

A ComparisonMaker class was used to generate Comparisons. To make sure the Pair-wise Comparison is a constructive and pleasant experience for the users, there are two problems that need to be solved: 1) users dont want to waste time and patience over the same Comparisons, therefore each Comparison needs to be unique, 2) In many cases, the exhaustive set of Comparisons is too much for any user to go through, so the subset of Comparisons chosen needs to be able to elicit enough useful information while stay at a reasonable size.

To solve the first problem, each Factor is made to keep track of the list of opponent Factors it has compared with. Every time when the ComparisonMaker generates a new Comparison of Factor A and B, it asks A whether it has compared with B. If so, the ComparisonMaker will discard this Comparison and will repeat the generation until a unique Comparison is generated or the list of possible opponents has been exhausted.

In order to tackle the second problem, a question needs to be answered first: what kinds of Comparisons will provide the most information gain? The task of assigning weights to each Factor can be framed as determining the locations of some points given the distance between every two points.

There are two types of Comparisons: Competitive ones that are between two factors which are close in terms of importance level, and Dominant ones where one Factor is clearly more important than the other one. The Competitive Comparisons describe and adjust the distances between points that are close to one another, which form clusters. A Dominant Comparison provides information that will set two points far apart from each other, which separates the clusters from one another. Therefore both competitive and dominant comparisons are important for finding and identifying the clusters.

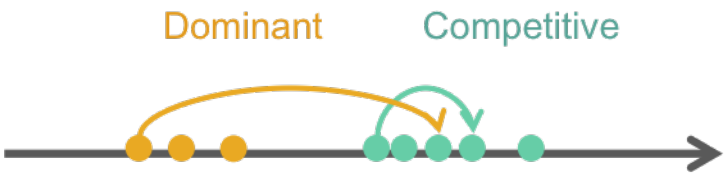

Figure 3.12: Dominant and Competitive Comparisons

A Dominant Comparisons separates 2 clusters, but if these two clusters have already been identified by another Dominant Comparison, a new Dominant Comparison does not add any new information. So after the clusters are found, new information gains mainly come from Competitive Comparisons adjustment of points within each clusters. But since we dont know the factors actual weights, we need to generate Comparisons based on our hypothesis that is built on top of the current information.

In order to gain more information, hypothesized Competitive Comparison is preferred because it will provide big information gain (identify new clusters) if it turns out to be a Dominant Comparison and it will still provide decent information gain (adjust positions) if it is actually a Competitive Comparison. On the other hand, if a hypothesized Dominant Comparison (Factors with very different weights currently) turns out to be a Competitive Comparison, it will provide decent information gain and help adjust the positions of the two existing clusters, dragging them closer together. But it will bring no new information gain if it actually is a Dominant Comparison because these clusters have already been identified.

To generate sets of Comparisons that provide the most information, the Comparison generation is separated into three rounds. The ComparisonMaker generates Comparisons in the beginning of every round. In the first round, no information is available yet, so the input order is used to form hypotheses about the importance similarity between Factors and generate Comparisons as following: $A1 : B1$, $A2 : B2$, $A3 : B3 \ldots$ Some users tend to input the most important and obvious Factors first while some start with the more trivial ones and saves the big ones for later. And if users enter their Pros and Cons in a more casual and random way, generate Comparisons by input order will just generate a set of random Comparisons which will likely generate some Dominant and some Competitive Comparisons. In the second round, since the information from the first round is available, Factors are matched up based on their current weights to form hypothesized Competitive Comparisons. And in the last round, a tint of randomness is added, in the hope of discovering missing cluster and fine tuning the weights.

For each round, the ComparisonMaker goes through the longer list of Factors of the two Choices to make sure each Factor in that list is only compared once in each round to ensure fairness. And since the number of Factors from different Choices is very likely to differ, to make sure each Factor gets exposure to at least one Comparison each round, the extra Factors are compared with opponent Factors randomly and uniquely. Therefore the number of Comparisons each round depends on the Choice with more Factors. For example, if A has 6 Factors and B has 8, each round has 8 Comparisons and there will be $8 \times 3 = 24$ Comparisons in total.

Comparison Custom User Interface

In order to present each Comparison in a clean and focused way and allow direct and intuitive interaction, a custom BubbleView is created. A BubbleView has two "bubbles" and two title labels, each bubble-label set displays a Factor from each side. The title labels font is made to auto-shrink when the title text becomes too long to display. Inside each bubble, there is a label for the Factors current weight in this Comparison. The bubbles size and the weight labels font size changes proportional to the Factors current weight. Since the "bubbles" are custom views instead of buttons, they dont have the nice touch detection feature built in. A touchesBegan:withEvent method is implemented to detect the coordinates of the touch and to update the Factors weights depending on which bubble the touch takes place within.

3.2.4 Factor Scoring and Result Computation

Once users have compared three rounds of Comparisons or all the Comparisons available (both Choices have less than three Factors), the information about the relative importance of the Factors were collected from those Pairwise Comparisons. To actually make a recommendation about a decision, the Choices need to be rated by evaluating the absolute importance of their Factors. So how can the relative importance be accurately converted to absolute importance?

Average Scoring

The most obvious and easiest way is to simply compute the average of the Factors weights in the Comparisons. For example, Factor A was compared to B and C (70 : 30 and 40 : 60 respectively), so its final weight is

$(70 + 40) \div 2 = 55$. This method is very easy and fast, but it also has a major limitation. Because of the averaging effect, the Dominant Comparisons information gains are neutralized by Competitive Comparisons. Therefore it can underestimate some important Factors and overestimate some less important ones, causing the Decision result to depend more on the number of Factors rather than the importance of each. For example, if Choice A has more Pros than Choice B, even though As Pros are less important, because all of the Factors weights are within a relatively small range, A will win because there are more of them.

Network Scoring

To compensate this limitation and make sure the result is as fair as possible for all kinds of user input, another scoring method is developed. This method first converts the Decision to a network of Factors. Each Factor is a node and the Comparisons are the edges that connect the nodes. Weights are shifted between factors through comparisons. Similar to the PageRank algorithm, which estimates a websites importance using the number and quality of links to the website, this scoring method make estimations about Factors based on the edges between them.[19] The underlying idea is that the score a Factor receives from a Comparison should be proportional to how important its opponent Factor is. For example, the statement "A is important" is more valid and more valuable coming from an important Factor B than from a trivial Factor C.

Each Factor starts out with 10 points and these points will be redistributed via Comparisons to reflect the estimated importance of the Factors. The Factors points first need to be fairly distributed through all the Comparisons it has been involved:

$$ComparisonPoints = TotalPoints \times \frac{CurrentOpponentWeight}{TotalOpponentWeights}$$

Each Comparison gets a portion of the Factors Total Points based on how much weight a specific Comparison carries with regard to the Factor's total contribution to opponents. And within each Comparison, the Factor contributes a portion of of the Comparison Points to the opponent Factor, while keeping the rest to itself:

$$\text{Gives: } ComparisonPoints \times \frac{OpponentWeight}{100}$$
$$\text{Keeps: } ComparisonPoints \times \frac{OwnWeight}{100}$$

Going through each Factor and each Comparison following the above distribution formula, the scores of the Factors are updated. Weights are transferred from the less important Factors to the more important ones. This process is repeated until the resulting scores of the Choices converge.

For example, Figure 3.13 shows the Network of Factors of a decision with two Factors for each Choice. The Comparisons are shown as edges between the Factors A1, A2, B1, B2. All Factor start out with equal scores.

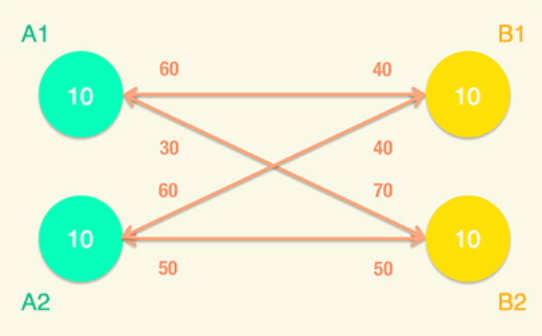

Figure 3.13: Network of Factors

Start with Factor A1, its score is redistributed among its opponent Factors (B1 and B2) and itself. B1 gets $10 \times \frac{40}{40+70} \times \frac{40}{100} = 1.45$ and B2 gets $10 \times \frac{70}{40+70} \times \frac{70}{100} = 4.45$. A1 gets to keep $10 \times \frac{40}{40+70} \times \frac{60}{100} = 2.18$ from Comparison with B1 and $10 \times \frac{70}{40+70} \times \frac{30}{100} = 1.91$ from Comparison with B2, 4.09 in total. Similarly, A2's score is distributed among A2, B1 and B2.

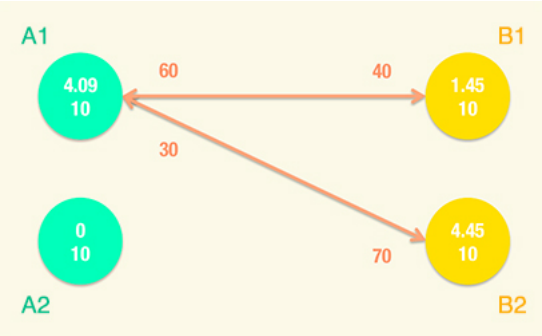

Figure 3.14: After Distributing A1's Score

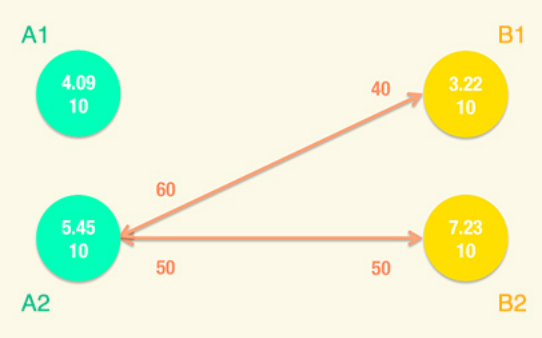

Figure 3.15: After Distributing A2's Score

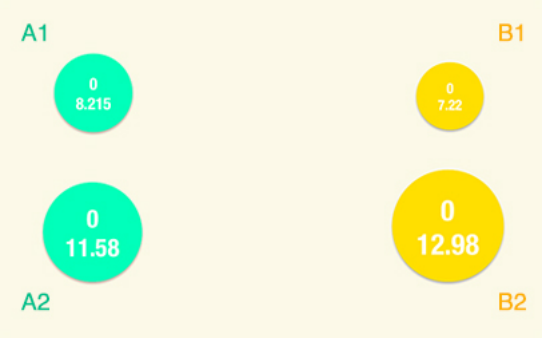

Figure 3.16: After Distributing All Factors' Scores

After going through all the Factors to redistribute scores via Comparisons, every Factor is updated with a new score. The new scores are then redistributed in the same way repeating above steps until the scores converge when overall score of each Choice stops fluctuating.

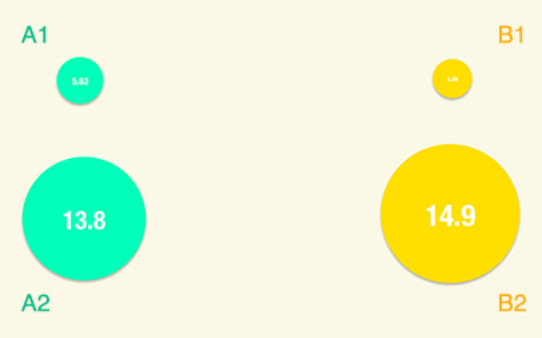

Figure 3.17: After Converging

Because the two Choices might have an uneven number of Factors, there will be Factors that were compared more than others. Therefore Network Scoring, modeling after PageRank, will favor the Factors that have more connecting edges (Comparison) because having more edges means there will be more points being transferred over. To counterbalance this problem, the weight of each Factor is divided by the number of Comparisons it was involved in.

$$FactorWeight = NetworkScore \div NumberofComparisons$$

The Factors final weights are summed together to compute the "goodness" or score of each Choice, where Pros have positive weights while Cons have negative weights.

$$Score = \sum_{Pro} FactorWeight - \sum_{Con} FactorWeight$$

The summation results in two floating point real numbers. However, it might be hard for users to interpret the result just looking at the "goodness", especially in situations when one is positive and the other is negative or when both are negative. To make the scores more comparable, the following adjustments are performed:

If the scores are of different signs, both scores are adjusted rightwards on the numeric scale by twice of the absolute value of the negative score. After the adjustment, both scores become positive so they are more comparable while their difference stays constant.

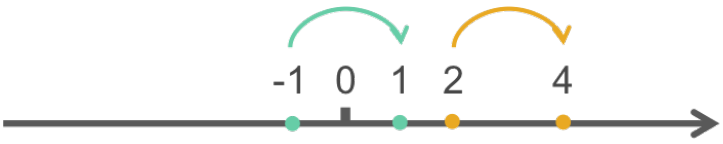

Figure 3.18: Normalize Scores of Different Signs

Similarly for scores that are both negative: to make both scores positive, they are adjusted rightwards on the numeric scale by the sum of the absolute values of both scores.

Figure 3.19: Normalize Two Negative Scores

When both scores are positive, they can be normalized to reflect relative "goodness", which is more straightforward and easier to understand:

$$resultA = 100 \times \frac{scoreA}{scoreA+scoreB}$$
$$resultB = 100 - resultA = 100 \times \frac{scoreB}{scoreA+scoreB}$$

3.2.5 Error Prevention

"Nothing is more likely to invoke the ire of users than to have an app crash", says Apples iOS Developer Guide[20]. Error prevention is critical to the user experience, and it is important to make sure the app won't fail no matter what the users do. Although it is not likely to develop perfect error prevention mechanism upfront, special attentions were paid in the development of this app to address some of the common user operation errors found in and prior to the testing.

In the Pros and Cons screen, the Decide button is disabled before there is at least one Factor for each Choice. This is reasonable because: 1. the Comparison step cannot be initiated because at least two Factors form one Comparison, 2. the lack of Factors indicates that users have not finished the input and it was probably an accidental operation.

Also during Factor input, users need to indicate whether the Factor is a Pro or a Con. Since the app cannot decide for the user and no default option is reasonable, this step is mandatory and the app will crash if no action is

taken. To prevent crashing, an alertView will pop up if the user forget to select one of the two options and ask the user to make a selection in the alertView. The alertView ensures user input and reminds user to make a selection next time around.

Chapter 4

Evaluation

After the completion of this app, user testing sessions are conducted to evaluate the app. The most important goal of the testings is to find out whether this app is actually helpful for decision makers. Along with this question, information about how people make decisions and feedbacks for improvements are also collected.

4.1 Decision Making Patterns

Prior to the actual testing, a list of background questions about decision making is asked: "How do you normally make decisions?". "Have you ever feel regret about a decision you make?" and etc..

According to the answers, a majority of the participants think about Pros and Cons (11 out of 13) in the decision making process. And most people also talk to other people to get input about the options or to talk through the Pros and Cons of each option (9 out of 13). Some people like to wait until the last moment to make a decision (4 out of 13), which gives them time to think about the options as long as it is allowed and also forces a quick and decisive decision because it leaves no time to change their minds. One very interesting and opposite method that was brought up is to make the decision way before the deadline so there is time to live with the decision for a while and evaluate whether it is in fact a good choice.

And according to almost all of the participants (12 out of 13), they rarely or never feel regret about their decisions. Most people don't think about the decision once it is made and contempt about where their decisions brought them to be. While this discovery might not be generalizable, it is encouraging to know that after careful research and consideration, it is likely

that no matter what the final decision is, people are happy with the result.

4.2 User Testing Plan

To assess the effectiveness of the app, participants are asked to make a decision of their choice. First the participants will make the decision without the app, in ways they normally make decisions in. Some people brought a list of factors they are considering, some made a spreadsheet of all the criteria and scores for each category, some talked to me about what they are considering about the decision and some just think in their heads. After they made the decision, they are asked to evaluate their confidence level about the decision on a scale of 1 to 10 prior to using the app. Then participants proceed to make a decision with this app. Afterwards they are asked to evaluate their confidence level again. They are also asked to rate their trust in this app and whether they will use it to make their decisions in the future.

The original research plan asks participants to make two decisions with the app, one fixed decision that every participant should be able to relate to. However, because the group of participants was very diverse, there is no one question that will fit everyone. For example, "Study abroad or not?", "What should I major in?" and "Should I go to grad school after graduation?" are all relevant questions for underclassmen. But these questions do not apply to upperclassmen or recent alums. As the testing continues, this problem becomes more prominent so the first decision is eventually dropped from the plan as well as from the result analysis.

See Appendix for the final version of User Testing Research Plan.

4.3 Testing Results

13 people participated in this User Testing Study. Data for each decision is collected and shown below.

The average Difficulty (`d`) of the decisions is 6.79, with a 2.14 standard deviation and ranges from 3 to 10. The participants' reported confidence before using the app (`c1`) and after using the app ($c2$) are recorded on a scale of 1 to 10, with the average being 4.62 and 6.81 respectively. However, both confidence measures vary greatly across decisions, each with a standard deviation of 2.59 and 2.49. Participants confidence in the decision also seem to be negatively correlated with the decisions difficulty. This makes intuitive sense because people tend to be less sure about more difficult decisions.

Participant	Difficulty	Confidence Before	Confidence After	Trust
1	6	0	4	5
2	7	2	7	7
3	10	2	8	7
4	8	4	6	7
5	8	6	9	5
6	4	9	10	7
7	3	9	10	7
8	7	6	8	7
9	7.5	5	9	7.5
10	8	2	4	8.5
11	7	4	7.5	8
12	4	6	4	8
13	10	5	2	7

Table 4.1: Table of Collected Data

Participant	Difficulty	Confidence Before	Confidence After	Trust
Average	6.88	4.62	6.81	7.00
MAX	10	9	10	8.50
MIN	3	0	2	5.00
Std Dev	2.08	2.59	2.49	0.98

Table 4.2: Table of Analyzed Data

Participants showed generally high Trust in the app, which is on average 6.88 (out of 10), ranging from 5 to 8, with a relatively small standard deviation 0.92. And all participant indicated that they will use this app for future decisions.

And among the 13 decisions, there were two cases where the recommendation given by the app contradicted with what the decision maker originally chose. They were surprised at first, but still reacted positively about the app (trust level at 7 and 8). One of them thought the opposite result really influenced her decision and the other one actually ended up following the apps recommendation when the decision moment came. In both cases, the participants indicated a decrease in confidence in their original choice, which by definition, should be different from confidence in the decision when the original and recommended choice differ. However, this subtle difference was not observed prior to the study, thus only one general measure of confidence in the decision is collected.

Participant	Difference	Abs. Difference	Influence	Abs. Influence
1	4	4	0.67	0.67
2	5	5	0.71	0.71
3	6	6	0.60	0.60
4	2	2	0.25	0.25
5	3	3	0.38	0.38
6	1	1	0.25	0.25
7	1	1	0.33	0.33
8	2	2	0.29	0.29
9	4	4	0.53	0.53
10	2	2	0.25	0.25
11	3.5	3.5	0.50	0.50
12	-2	2	-0.50	0.50
13	-3	3	-0.30	0.30

Table 4.3: Measurement of Effectiveness

Table 4.3 shows 4 kinds of measurement for evaluating the effectiveness of the app, the Difference, Absolute Difference, Influence and Absolute Influence. Difference is simply the difference between confidence after and confidence before: $c2 - c1$. Influence can be interpreted as Difference per unit of Difficulty. Since if a decision is relatively easy, the decision makers are quite sure already so original confidence is quite high, which leaves little room for growth and the decision makers are less likely to be influenced. Therefore Influence is defined as $\frac{c2-c1}{d}$ to normalize Difference for fair evaluation. The result shows an average of 21.92% Difference and 30.45% Influence.

However the Differences and Influences do not take into account the contradicting results, thus the positive and negative values offset each other out, whereas Absolute Differences and Absolute Influences capture the actual influence more accurately. After adjustment, the Absolute Difference is 29.62% and Absolute Influence is 42.76% on average.

In general, participants are positively influenced after using the app and participants have high trust in this app regardless of the difficulty or the result of the decision.

4.4 User Feedback

When participants were using the app to make decisions, I observed their interactions with the app and took notes on usage patterns and common

places that people were having trouble with. I also asked them to think aloud about their interactions with the app to help me understand better whether something was more difficult than it needed to be. At the end of the testing session, I also asked them for general feedback about the experience and suggestions for improvement.

Figure 4.1: Comparisons Screen Redesign

Many useful feedbacks were collected. One common thing people brought up is that they want more guidance during Pairwise Comparison. When they are brought to the Comparison screen, some participants were hesitant about how to proceed. But with a little time and poking around, everyone figured out how to do the comparisons without guidance. The difficult part is when participants were asked to compare a Pro versus a Con. Close to half of the participants did not know how to make such comparison at first. To help them with the comparison, I asked them: "Which one do you care more?" With this hint, they were all able to proceed and make the Comparison. Later when they encountered similar situations, they still showed hesitation and needed to remind themselves to compare "Which one do I care more" instead of "Which one is more important". This difficulty suggests that more guidance need to be provided for the Comparison screen, especially during the Comparison of a Pro and a Con. Such guidance should be easily

accessible, for example, as a static part of the UI as shown in Figure 4.1, because it is likely that users will need it more than once.

Participants also wanted more guidance for the Pros and Cons input step. The app provides the functionalities to help users formalize the process of listing Pros and Cons and rationalize the process of weighing the Factors, but the quality of the decision still comes down to the quality and the completeness of the decision makers input. One source of suboptimal results is skewed distribution of Factors. Even though the Network Scoring algorithm is developed to extract information from Comparisons fairly, if the distribution of Factors is really skewed, that is if one Choice has 5 Factors while the other one has only 1, there will only be 5 Comparisons and not much information can be elicited. Another problem with uneven Factors is that the app generates rounds of Comparisons based on the Choice with more Factors (e.g. If Choice A has 10 Factors and Choice B has 3 Factors, each round will have 10 Comparisons which sums up to a total of 30 comparisons. Whereas if the Factors can be adjusted to be 7 and 6 for Choice A and B respectively, only 21 comparisons are needed and will likely give a more accurate result because each Factor is compared for more or less the same number of times. Different from the guidance for the Comparison screen though, it is not necessary to provide guidance for the Pros and Cons screen every time users input Factors. But having an old-fashioned FAQ does not help in this case either. Since users will not automatically realize they might be on the path to a suboptimal result, they will not bother to check the FAQ. One solution is to display tips as an alertView when users are done with the input and are ready to proceed to Comparisons. This can help target the applicable situations where users has entered a skewed set of Factors.

Chapter 5

Future Work

5.1 App Submission

This app currently has all of the functionalities implemented and tested. And it has been shown helpful for decision makers through user testings. But there is some additional work to be done before I publish this app in the App Store.

First of all, I will have to fix some small edge case bugs found during user testing. I will also implement the additional guidance according to user feedback. Additional polishing will also be done to improve the User Experience before app submission.

For example, the app should be more reliable so users will never lose any decision data without willingly deleting it. Therefore each Decision should be able to store histories of comparisons, if users want to redecide, either to modify the Pros and Cons or just to redo the Comparisons, the older version of Decision will be stored in history and can be viewed later for reference.

In addition, I would like to add sharing feature so users can post the results of their decision making to Facebook and/or Twitter to share the decision and app with their friends.

5.2 Future Iteration and Improvement

After submitting the first version to the App Store, I also plan to iterate to make improvements based on user feedback and fix bugs that come up during actual uses.

Since this app is currently only useful for important and complex decisions, an average user will only use this app once in a while. To prevent

this app from being buried in pages of rarely used apps on the device and to make it an useful tool for daily life, this app should also be able to help with small daily decision making as well. For example, there have been many requests for deciding "where to eat today?" For such decisions, people usually have more than two options and simply want a quick answer. Therefore, adding a simple random decision maker, which will do the job by randomly picking one option, will make this app a more complete and practical tool for daily use.

I would also like to explore better user interaction designs. For the Pros and Cons screen, I want to explore ways to display all the information while retaining the ease of adding and editing Factors. And there is definitely room for improvement for the Comparisons screen: way to regret and redo a Comparison, more dynamic and interactive bubble and etc..

One feature I would like to add is to ask for decision feedback after the decision has been made. Collect and record information about users final choice, and how they feel about the decision in a week, a month or a year. This not only helps people reflect on their decisions but also provide useful information for the app to learn the user and potentially improve decision making in the future.

In addition, since not all Factors are necessarily facts, the app will be more complete and more powerful if it can take uncertainties into consideration. For example, if a Factor is might get a better job, it is not certain if this Factor will actually happen, therefore it should not be counted equally as other Factors that are facts.

Chapter 6

Conclusion

Decision making is an important and difficult process. In order to make a good decision, decision makers first need to know their choices. This step can be done through research, talking to other people and self-reflection. But even with enough information, it is still hard to compare across options and consider all the different dimensions. This project is an exploration to develop an iOS app that help users compare their choices and make rational and confident decisions. Learning from the decision making theories and methods, this app utilizes Pros and Cons weighing and Pairwise Comparisons to guide users to choose the optimal option.

To make a decision, the app lets users create a Decision with two Choices and asks them to enter the Pros and Cons for each Choice. The next screen asks users to do Pairwise Comparisons to answer the question, "Which factor do you care more?". The comparisons elicit information about the Factors relative importance to the decision maker and Network Scoring quantifies the Factors absolute importance with the information. The end result is presented as relative "goodness" of each each Choice, which is a weighted sum computed and normalized using the importance score of the Factors.

Based on the user testing results and user feedbacks, this app does help to improve decision makers confidence in their decisions. Users felt the app really helped to make them think deeply about their decisions and the Result Analysis also helped them reflect on the decisions and on themselves.

This app will be published in the App Store for a wider audience to test and use for decision making after some minor bug fixes and feature enhancements. Additional work will be done to improve the User Interface and add features according to user feedbacks in the future.

Overall, this has been a successful project where an effective decision

making app is built from ground up, starting as a conceptual solution to a problem shared by my friends and myself. It has been a great learning experience for me in terms of mobile development and decision making. Some of my friends and myself have benefited from making decision with the app and I hope it will be able to help make many other people's decision making process easier and happier.

Appendix A

Research Plan

Overview

10 usability study sessions will be conducted with students and faculties at the Claremont Colleges. Participants will make a decision of their choice, with and without the decision-making app and then evaluate their confidence and satisfaction with the decision. This studys goal is to find out whether this decision-making app is a useful and effective aid of decision-making. Participants interactions and feedback will also be taken into account for improvement of the User Experience of the app.

Usability Testing Outline

Introduction

Thank you very much for agreeing to help out with the study! I will ask you a couple questions about how you make decisions and then ask you to make a decision. The decision can be anything that you are deciding recently or one that you have recently decided. I will be taking notes throughout the study so I can remember later. Please dont be distracted by me and continue to talk or do whatever you are doing.

So before getting into the actual decision-making, lets start off with some questions about how you make decisions.

Background Questions

1. How do you make decisions?
2. How would you describe yourself as a decision maker? Decisive? Indecisive? Confident?
3. Are you happy with the decisions you make? Do you ever have regret about your decisions?
4. Whats the most recent important decision you make? How did you decide?

Decision Making

Ok, now lets make a decision! Do you have an important decision to make recently? Or in the recent past? (Job/grad school offers? Internship? Summer plan? Study abroad?)
Do all the things you do when you make a decision. Here are some tools that might help you (pen, paper and coin).

How difficult do you think this decision is? (Scale 1- 10)
(Observe decision-making process)
How confident (rational?) are you about this decision? (Scale 1-10)

Now make the same decision with this decision-making app.
I would like you to try to think aloud as you use the app. If there is something you like, dislike, something you think is more difficult than it needs to be, let me know those things.
(Observe decision-making process)
How confident (rational?) are you about this decision now? (Scale 1-10)
(If the result contradicts with the previous result), how much do you think this app influences your decision? (Scale 1-10)

How much do you trust this app? (Scale 1-10)
Will you use this app? (Yes or No)

Feedback and Suggestions

1. How did you think about the experience of making a decision with this app?

2. Do you have any suggestions for improvement?

Bibliography

[1] Bonnefon, Jean-Franois (08/2008). "Qualitative Heuristics For Balancing the Pros and Cons". Theory and decision (0040-5833), 65 (1), p. 71.

[2] Howard, Ronald A. (1966). "Decision Analysis: Applied Decision Theory" (PDF). Proceedings of the 4th International Conference on Operational Research. pp. 5577.

[3] Howard, Ronald A. (Jun., 1988), Decision Analysis: Practice and Promise, Management Science, Vol. 34, No. 6. , pp. 679-695.

[4] Bernoulli, Daniel; Originally published in 1738; translated by Dr. Louise Sommer. (January 1954). "Exposition of a New Theory on the Measurement of Risk". Econometrica (The Econometric Society) 22 (1): 22-36. doi:10.2307/1909829. JSTOR 1909829.

[5] Neumann, John von, and Morgenstern, Oskar (1953), Theory of Games and Economic Behavior, Princeton, NJ, Princeton University Press, 1944, second ed. 1947, third ed. .

[6] Daniel Kahneman; Amos Tversky (Mar 1979), Prospect Theory: An Analysis Of Decision Under Risk. Econometrica (Pre-1986); ; 47, 2; ABI/INFORM Global Pg. 263

[7] Tversky, A., Kahneman, D. (1992). Advances in prospect theory: Cumulative representation of uncertainty. Journal of Risk and uncertainty, 5(4), 297-323.

[8] Decision Analysis and Decision Support Systems, Marek J. Druzdzel

[9] Poh, K. L. (2000), An Intelligent Decision Support System for Investment Analysis, Springer-Verlag London Limited, 0219-1377

[10] KAMARUDIN SAADAN, ABDUL RAZAK HAMDAN, Development Of Intelligent Decision Support System For Crop Management

[11] Dong-Ling Xu, Jian-Bo Yang, Intelligent decision system based on the evidential reasoning approach and its applications

[12] Dubois, D., Fargier, H., Bonnefon, J. F. (2008). On the Qualitative Comparison of Decisions Having Positive and Negative Features. J. Artif. Intell. Res.(JAIR), 32, 385-417.

[13] Stillwell, W. G., Seaver, D. A., Edwards, W. (1981). A comparison of weight approximation techniques in multiattribute utility decision making. Organizational behavior and human performance, 28(1), 62-77.

[14] Thurstone, L.L. (1959). The Measurement of Values. Chicago: The University of Chicago Press.

[15] Ali Abbas, Ahoona, `https://www.ahoona.com`

[16] Rad Rhino, inDecision,
`https://itunes.apple.com/us/app/indecision/id458265246`

[17] Dmitry Fadeyev (2009). "8 Characteristics Of Successful User Interfaces". Usability Post. http://usabilitypost.com/2009/04/15/8-characteristics-of-successful-user-interfaces/

[18] Myers, Brad A. (1985) "The importance of percent-done progress indicators for computer-human interfaces." ACM SIGCHI Bulletin. Vol. 16. No. 4. ACM.

[19] Page, L., Brin, S., Motwani, R., Winograd, T. (1997). PageRank: Bringing order to the web (Vol. 72). Stanford Digital Libraries Working Paper.

[20] Apple iOS Developer Guide. "Improving Your Customers' Experience".
`https://developer.apple.com/library/ios/documentation/LanguagesUtilities/Conceptual/iTunesConnect_Guide/Chapters/ImprovingCustomersExperience.html`