ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Lecture with Computer Exercises:
Modelling and Simulating Social Systems with MATLAB

Project Report

A Probabilistic Model For Recommending An Optimal Action To Maximize The Probability Of Achieving a Goal

Sarah K. Curry

Zürich
19 December, 2016

Agreement for free-download

We hereby agree to make our source code of this project freely available for download from the web pages of the SOMS chair. Furthermore, we assure that all source code is written by ourselves and is not violating any copyright restrictions.

Sarah K. Curry

Table of contents

Abstract

When making decisions, humans and machines consider different types of information and have significantly different processes for weighing the importance of the information. Humans often rely on emotions that are swayed toward short term fulfilment, which may compromise the optimality of decisions about long term goals, especially when there are multiple conflicting goals. This article proposes a probabilistic model that, given information about a goal and the characteristics of the individual desiring to achieve the goal, determines the optimal action that should be taken in order to maximize the probability of achieving the goal in a minimal amount of time.

Individual contributions

This model was developed with the help of several people who provided invaluable guidance during brainstorming sessions. These brainstorming opportunities enabled me to better understand the methodology leveraged from Dynamic Programming and Probability, and how to employ it for the specific problem at hand. Please refer to the Acknowledgements section for details on their contributions.

The model leverages several concepts and problem solving methods in the fields of Probability and Dynamic Programming. Examples of these concepts were leveraged in the implementation of the model.

The final design of the model is solely my work.

Introduction and Motivations

Humans constantly deal with the challenge of how to prioritize time spent on various activities, given multiple and often conflicting goals. Goals include long term achievements such as earning an advanced degree, short term desires such as enjoyment of the present moment, and ongoing maintenance such as physical health.

The objective of this current work is to develop a model that recommends an optimal action, given a goal, as well as metrics about the individual that measure the individual's ability to make progress toward that goals. This structure selected for the model allows for it to be expanded to handle multiple, conflicting goals, which is further discussed in the Summary and Outlook section.

Especially in the case of long term goals that require delayed gratification, prioritizing time spent on the activities that contribute to such accomplishments is difficult for humans (footnote for Walter Mischel, marshmallow test). Research in delayed gratification was pioneered by Walter Mischel, who explored self control in children by performing an experiment that has become known as the 'marshmallow test' (footnote). The marshmallow test demonstrated a correlation

between the ability of a preschooler to delay gratification (which in this case, was the enjoyment of a marshmallow), and success later in life. Adults also regularly face challenges of delayed gratification, such as resisting the temptation to smoke or overeat in return for improved long term health.

Decision making in humans is an active area of research[1], though it is clear that emotions play a critical role in decision making by affecting the perception of the situation and the types of information used to determine an optimal action. In contrast to the combination of emotion and logic that humans use to make decisions, this model seeks to determine an optimal action using a mathematical approach; in essence, stripping away illogical influence from the decision. By instead considering decisions as weighted probabilities, such a model may effectively serve as a guide for an individual.

Terminology

There are several terms used throughout this report that are important to clarify.

Individual
The model depends on the specific characteristics and the goal of a single person, who referred to as the individual.

User State
The User State describes all recorded characteristics that may influence an individual's ability to make progress toward a goal. These characteristics describe an individual's emotional and physical state, and include hunger, sleepiness or other fatigue, and motivation to perform tasks involved in achieving the goal. For the current scope of this model, it is assumed that these factors can be accurately measured.

User State Factor
The User State Factor is a value that represents the probability that the individual will accomplish the attempted work in the given User State.

Goal
A goal is any desired accomplishment that has a quantifiable result. The intermediary steps toward the goal must also be measurable; this is to provide feedback to latter versions of the model about whether progress toward the goal was made, given an action taken.

Stage
A goal is divided into n intermediary steps, where each step constitutes a stage. The goal is completed when the nth stage is reached.

---

[1] Lerner, Jennifer S., Li, Ye, Valdesolo, Piercarlo, & Kassam, Karim. Emotion and Decision Making (2014). Manuscript submitted for publication in the Annual Review of Psychology. Retrieved from https://scholar.harvard.edu/files/jenniferlerner/files/annual_review_manuscript_june_16_final.final_.pdf

Node
Each {User State, stage} pair constitutes a node. By describing both the individual's ability to progress toward a goal and the remaining work to be done, nodes describe the complete state of the problem.

Description of the Model

In order to determine an optimal action for a given situation and goal, three components are required. First, it is necessary to have information about the current situation and the impact of selecting a given action. The situation is defined by the User State and the amount of remaining work until the goal is completed. Each possible action has a probability of transitioning the individual to a new User State and stage. Using this information, the Transition Probability Model determines a probability of transitioning between any two states for a given action.

Second, there must be a cost associated with being in a given User State, since some User States are better for progressing toward the goal. Additionally, there must also be a cost for applying a given action. In this current version of the model, the cost for applying an action is the time required to perform the action. These costs are represented by the Cost Model, which determines a stage cost for each {User State, action} pair.

The third component is the Optimization Strategy Model. There are numerous methods for devising an optimal action; this current version of the model leverages Value Iteration, which is a standard method in Dynamic Programming.

Inputs from the Probability Transition Model and the Cost Model are leveraged by the Optimization Strategy Model, which then calculates the expected cost-to-go from the current node (each User State at any stage of completion, for each possible action, . The action for any given User State at any stage of progress toward the goal, and . These three component models are however independent of each other. Thus so long as they output the same information in the same structure, each of the component models may be developed independently to account for more realistic scenarios.

Implementation

Development of each of the three component models is discussed separately. All code was implemented in matlab.

Development of the Transition Probability Model

For any given node (or {User State, stage} pair) *i* there is a probability of transitioning to a new node *j* given that action *a* is taken. This is represented as a Bayesian conditional probability:

$$P(new\ node\ =\ j \mid current\ node\ =\ i,\ action\ =\ a)$$

The Probability Transition Model describes the probabilities of all possible transitions between nodes, for each allowable action.

The User State is modeled by an N-dimensional space, where N represents the number of measurable factors that may influence the individual's ability to accomplish a goal. If each of the N factors is discretized into M parts, there are $M^N = K$ possible nodes. Thus the Probability Transition Model is represented by an K x K x A matrix.

The Transition Probability Model must be fitted to the individual; otherwise the recommendations will not be suitable. Accurately calculating transitional probabilities is nontrivial, since it is highly dependent both across different individuals' User State spaces and within a single individual's User State space.

First, the same action will result in different outcomes for different individuals. Eating chocolate cake for example may increase one indivudal's User State (e.g. by increasing present moment enjoyment, or decreasing hunger level), while decreasing another's User State. Second, a given action may have different probabilistic outcomes even for the same individual. When an individual is in a state where hunger level is high, an action of eating chocolate cake may increase the individual's User State. Alternatively, eating chocolate cake in a User State where hunger is very low may have little or negative effect.

Given that the Transition Probabilities are so dependent on the individual, it is critical to accurately calculate transition probabilities for each {current state, new state, action} combination. It is neither reasonable nor scalable to test each action at each node a statistically significant number of times. Bayesian learning techniques such as Q-Learning are a much better option, and may be employed for future versions of this model.

For the scope of this current version of the model however, data were collected informally, and the Transition Probabilities were determined based on a best fit to these data. User State data were collected at fifteen minute intervals over a period of several weeks. These data included the activity performed and the work accomplished (as a measure of productivity) for each interval. User State factors for each interval were also collected, including levels of hunger, fatigue, and a combined measure for happiness and motivation. These factors are converted into a 1-dimensional User State measure that represents the individual's ability to accomplish work.

Data were best fit to the transition probabilities by finding the means for a given probability $P(current\ node, new\ node, action)$ when available, and interpolating for probabilities where data were not available. This method is only intended to provide an estimate of the effects of a given action when an individual is in a particular User State. Perfecting the metrics selected for

assessing the effect of a given action, and the data collection method constitute work for a latter version of this model.

For each possible node $i \forall K$, the transitions to every other possible node $j \forall K$ were determined, for each action $a \forall A$. In a simplified implementation of the Transition Probability Model, the User State is discretized into fifteen states and the goal is discretized into ten stages:

$$User\ State = \{1, 2, 3, ...15\}$$
$$Goal\ Stages = \{1, 2, 3, ...10\}$$

Therefore there are $15 \times 10 = 150$ possible nodes:

$$Nodes\ K = \{1, 2, 3, ...150\}$$

For each node *K* there are two allowable actions:

$$Actions\ A = \{work, break\}$$

The transition probabilities based on the best fit estimation of the User State data are presented in Table 1. Selecting the action *break* always resulted in transitioning to new state, either improved, worsened, or the same, with the same probabilities. In the case of selecting the action *work* however, the probability of transitioning to a new state depends on the individual's current User State. If the individual's current user state is high, this increases probability of transitioning to a new stage where work has been accomplished. Alternatively, a low user state makes it less probable that work will be accomplished.

	P(improve User State)	*P(worsen User State)*	*P(same User State)*
action = work	0.01	0.2	1-P(improve)-P(worsen)
action = break	0.95	0.01	1-P(improve)-P(worsen)

	P(progression toward goal)	*P(retrogression from goal)*	*P(no change in progress)*
action = work	0.99*UserStateFactor	0	1 - P(progress) - P(retrogression)
action = break	0	0	1

Table 1: Transition probabilities were determined based on the best fit from data collected. The progress toward the goal is dependent on the User State Factor, which is a scalar value representative of the individual's ability to accomplish work.

The User State Factor represents the individual's ability to accomplish work. is a piecewise function that was selected to best match the data collected. Figure (state factor) presents the User State Factor for a given User State.

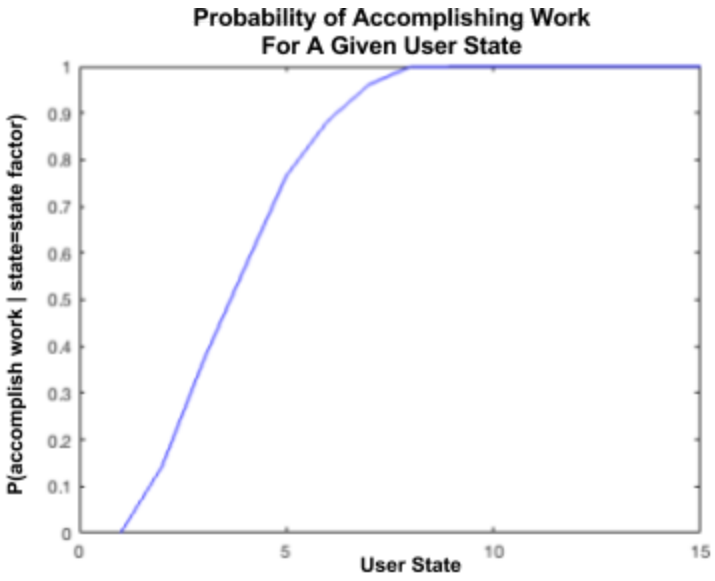

Figure 1. The User State Factor represents the probability that the individual will accomplish work in the next stage, which is dependent on the individual's User State.

Development Of A Cost Model

The objective of the Cost Model is to calculate the stage cost $g_K(i, A)$, which represents the cost of being in a state *i* and selecting action *A* at a given stage *K*. Selecting an action *A* results in a probabilistic transition to a new state j, which also will incur a cost. To account for the cost of selecting an action *A*, the cost of the new state *j* must also be considered. This is achieved by calculating the expected value over all possible future states:

$$g_K(i, A) = E\{\sum_{j=1}^{N} P(j \mid i, A) * g_{K+1}(j, A)\}$$

Where $g_{K+1}(j, A)$ is the cost of the new state *j*, having selected action A, and $P(j \mid i, A)$ is the conditional probability of transitioning to state *j*, given the current state *i* and action *A*.

Because a higher User State corresponds to an increased probability of accomplishing work, there must be a cost for being in any nonideal User State. For this implementation, the state cost of being in a state *i* is the difference between the maximum possible User State value and the current User State value:

$$s(i) = (\ max(User\ State) - UserState(i)\ )$$

Additionally, there is a cost for selecting a given action:

$$c(A) = m,$$

where m = the number of minutes required to perform the action.

Because the costs of being in a state i and selecting an action A are independent of each other, the stage cost for this problem can be stated as:

$$g_K(i, A) = c(A) + E\{\sum_{j=1}^{N} P(j \mid i, A) * s(i)\}$$

Using this equation, the stage cost was calculated for each node *K*, for each action *A*.

Development Of The Optimization Strategy Model

The objective of the Optimization Strategy Model is to identify the optimal action to take for any User State at any stage throughout the process of working toward a goal. The optimal action is defined as the one that maximizes the probability of getting to the goal with the fewest steps, or equivalently, in the least amount of time.

This problem of determining an optimal action for a given state is similar to the Shortest Path Problem, where the objective is to find the shortest distance between two arbitrary locations, or nodes (footnote for bertsekas and/or wikipedia). Having recast the current problem as a Shortest Path Problem, it is now possible to solve it using the Value Iteration Method.

In the Value Iteration method, the cost-to-go *J* is calculated for each node *K* for each action *A*. The cost-to-go represents the expected cost for the entire path from the current node *K* to the terminal node, where the goal has been completed.

$$J(x_0) = E\{\sum_{K=1}^{N-1} g_K(i, A)\}$$

Because the goal has been completed at the terminal node, it is represented by a cost-to-go of zero.

The optimal cost-to-go *J** identifies the policy that selects the actions that minimize the total costs incurred until a terminal stage is reached:

$$J^*(x_0) = \min_{a \,\forall\, A} [\, E\{\sum_{K=1}^{N-1} g_K(i, A)\}\,]$$

To find the optimal cost-to-go, the algorithm is initialized with guesses for the costs-to-go at each stage. If the guess is close to the optimal value, then the algorithm will converge faster; however it is guaranteed to converge for any initial value of $J(x_0)$. For each state, the cost

associated with being in that state and taking a given action is known. For each iteration of the cost-to-go J, the action that has the minimal cost is selected. Upon the next iteration of *J*, the minimized cost of a state j is factored in to the sum of the future cost for its preceding state *i*. Thus with each new iteration of cost-to-go *J*, we approach an optimal cost-to-go.

Simulation Results and Discussion

The results shown in Figure 2 correspond to the parameters described in Table 2. These parameters were selected because they best match the data collected. In this scenario, an arbitrary goal is divided into fifteen stages. At each stage, the individual may be in any one of ten possible user states. Each of these state-stage pairs represents a node in the shortest path problem, and for each of these nodes, the Value Iteration algorithm is applied to determine which actions result in completing the goal with the fewest steps. The expected value of the cost-to-go is plotted for each node, as well as the optimal action that should be taken if the individual is at that node.

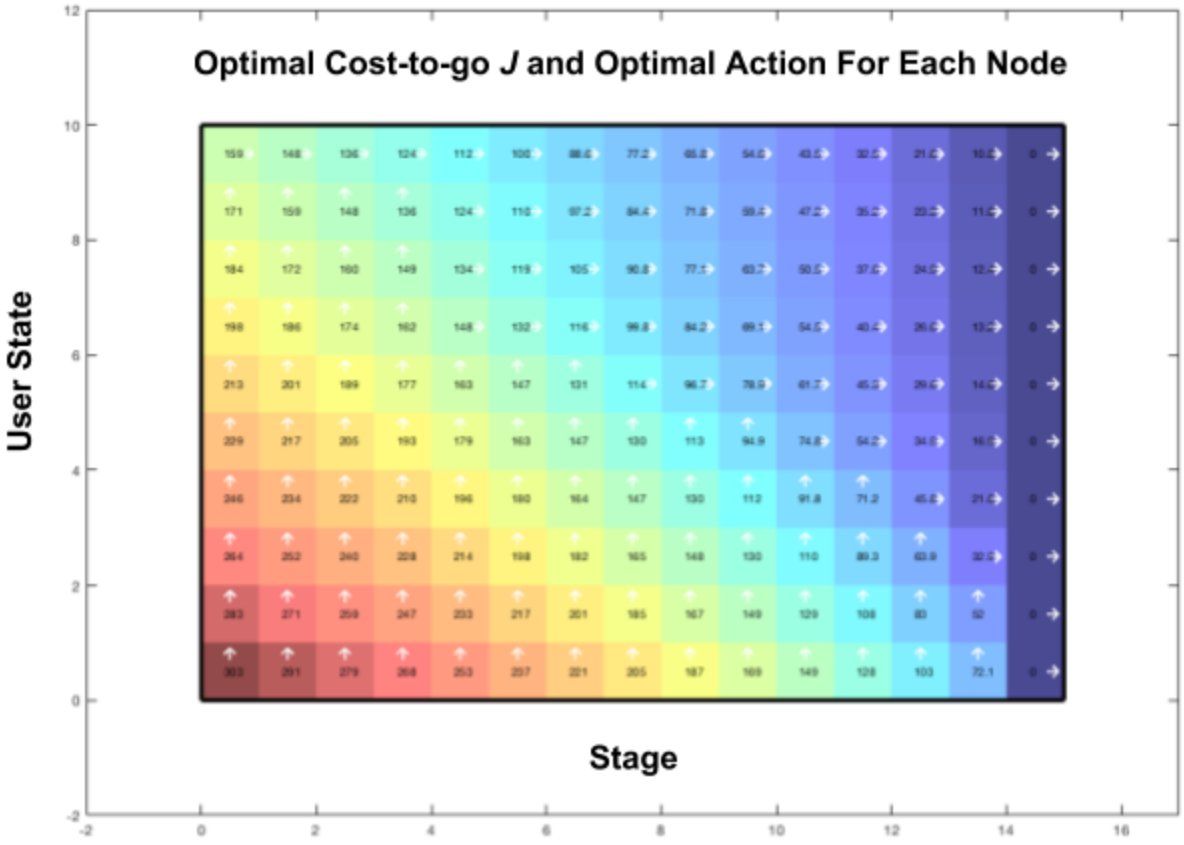

Figure 2. The optimal cost-to-go function is solved for each state at each stage of completion toward the goal; this value is printed for each node *K*. The optimal action, represented by a white arrow, indicates the recommended action to take in order to minimize the cost-to-go.

Although this current implementation of the model is highly simplified, there are several conclusions that can be drawn by the recommended actions given. These results indicate the length of time that the individual can work before requiring a break, for any given user state. For example, in the highest user state possible, the individual may work continuously for at least fifteen time units (150 minutes) without requiring a break. At the other end of the spectrum, the individual is advised not to attempt work at all in the lowest two user states. For the user states ranging in the middle, the length of time recommended to work continuously is indicated by the number of rightward pointing arrows in that row. In the row where the user state is equal to seven for example, there are eleven rightward pointing arrows; thus indicating a recommendation that the individual work continuously for no more than eleven time units (110 minutes).

Each of the parameters listed in Table 1 affect the recommendations made by the simulation. Of these however, two probabilities have the most significant influence: the probability of accomplishing work given a user state, and the probability that doing work lowers the user state.

As previously described, the User State Factor is the probability of accomplishing work given a user state. Increasing the User State Factor therefore also increases the total length of continuous work time recommended. Similarly, decreasing the User State Factor decreases the total length of continuous work time recommended.

Summary and Outlook

A model for recommending the optimal action given an individual's state and a goal has been developed. The results of this version of the model demonstrate that the problem of selecting an optimal action in a given situation can be recast as a Shortest Path Problem.

The Transition Probability Model and the Cost Model are inputs to the Optimization Strategy Model. The three components of the model have been structured in a way that allows for each to be independently expanded upon. As the ultimate objective of this model is to provide guidance in scenarios that are difficult for humans to assess, it is critical to provide the model with sufficient information. In particular, it is critical to have accurate information about the probabilistic effect of a given action for a given User State, which is captured by the Transition Probability Model. In this current version, the Transition Probability Model was based on data collected for a given individual. However, future versions of this model would significantly benefit from being able to learn about the effects of an action in real time.

The transition probabilities may instead be modeled as a Bayesian Network, where the structure between the nodes must be learned. Given this representation for the transition probabilities, methods such as Q-Learning may be employed to learn the structure between the nodes, given feedback from the individual. Furthermore, although this version of the model is designed to maximize the probability of achieving a single goal, a future version of the model may be expanded to consider multiple goals using a multiple-goal maximization method that is standard in Dynamic Programming.

There are also several improvements that could be made to the Optimal Strategy Model. In this version, the Value Iteration method is employed to determine the optimal action for each node by calculating the cost-to-go for each action, then selecting the action that minimizes the cost-to-go until the terminal stage is reached. One downside to this method however is that it is difficult to know when the algorithm has converged, and thus how many iterations should be performed. Policy Iteration is an alternative method that identifies a policy, or a set of actions $\{A_1,A_2,...A_N\}$. With each iteration, the policy is improved. This method requires more computation per step, but fewer steps than the Value Iteration Method. It is easy to determine when the Policy Iteration method has converged; this occurs when two consecutive policies are exactly equal.

In summary, the model proposed here determines an optimal action given information about an individual and a specified goal. The results of the model indicate the ratio of time that an individual should spend working versus taking a break for a given User State. The architecture of the model was selected to enable each of the component models, the Transition Probability Model, the Cost Model, and the Optimal Strategy Model, to be expanded upon independently.

References

Bertsekas, Dimitri P., Dynamic Programming and Optimal Control (3rd ed.), 2005.

Russell, Stuart J.; Norvig, Peter (2003), *Artificial Intelligence: A Modern Approach* (3rd ed.), Upper Saddle River, New Jersey: Prentice Hall, ISBN 0-13-790395-2

Lerner, Jennifer S., Li, Ye, Valdesolo, Piercarlo, & Kassam, Karim. Emotion and Decision Making (2014). Manuscript submitted for publication in the Annual Review of Psychology. Retrieved from https://scholar.harvard.edu/files/jenniferlerner/files/annual_review_manuscript_june_16_final.final_.pdf

Acknowledgements

Sebastian Tschiatschek, lecturer of Probabilistic Foundations of Artificial Intelligence for the fall 2016 semester, provided helpful guidance on development of the transition probability model.

Perry Franklin and Sandeep Menta, both master's students of robotics at ETH, provided helpful guidance on development of the Value Iteration optimization solution.

As part of the Dynamic Programming and Optimal Control course, a stochastic shortest path problem is provided as an example. This example was leveraged for this work.