Department of Math and Computer Science

MCS 4843 - Senior Project 2
Final Project

Student ID: 000758520

Student Name: Hunter Vallad

Date: 5/8/2022

Outline

1. Introduction

    1.1 Purpose

    1.2 Scope

    1.3 Definitions and Acronyms

    1.4 References

2. Overall Description

    2.1 User Needs

    2.2 Assumptions and Dependencies

3. System Features and Requirements

    3.1 Functional Requirements

    3.2 Nonfunctional Requirements

4. Architecture

    4.1 Visuals and Explanation

5. Setup

5.1 Installing dependencies

5.2 Getting the files from Git

1. Introduction

1.1 Purpose

The purpose of this application is to provide a readily easy way to compute Finite Horizon Markov Decision Processes by importing excel documents that represent various transitions and rewards matrices and generate an excel document that contains the recommended policy. The intended audience of this application is for people who want to fully understand and utilize the processor to compute a Finite Horizon Markov Decision Process.

1.2 Scope

This application will be used by various mathematicians intending to conduct heavy computations through the Markov Decision Process. The application is designed to allow users to input excel sheets that represent transition and reward matrices and maximize the efficiency at which the user adjusts their matrices and generates a policy. This allows the user to easily adjust their matrices on the fly, increasing their productivity and efficiency, and tailor their matrices to their intended values to produce a desired policy. The imported matrices allow mathematicians to easily work in a format that they are familiar with and allow them to understand the processor implicitly.

1.3 Definitions and Acronyms

Term	Description	*Acronym*
Markov Decision Process	The Markov decision process (MDP) is a mathematical model of sequential decisions and a dynamic optimization method (see Littman in references).	MDP
Finite Horizon	A variant of MDP that assumes a terminating time T in which the policy is met.	
Transition Matrix	The matrix representing the transition values, or probability values, that a state transfers to another state	
Reward Matrix	The matrix represents the reward values that an action receives traveling between states.	

1.4 References

Littman, Michael L. “Markov Decision Process.” *Markov Decision Process - an Overview | ScienceDirect Topics*, https://www.sciencedirect.com/topics/computer-science/markov-decision-process.

2. Overall Description

2.1 User Needs

The users of this product will be mathematicians and computer scientists performing Finite Horizon MDP calculations. They are the primary user and will be fully utilizing this processor to handle a lot of the processing and calculations.

2.2 Assumptions and Dependencies

To use this application the user must have the following software installed on their machine:

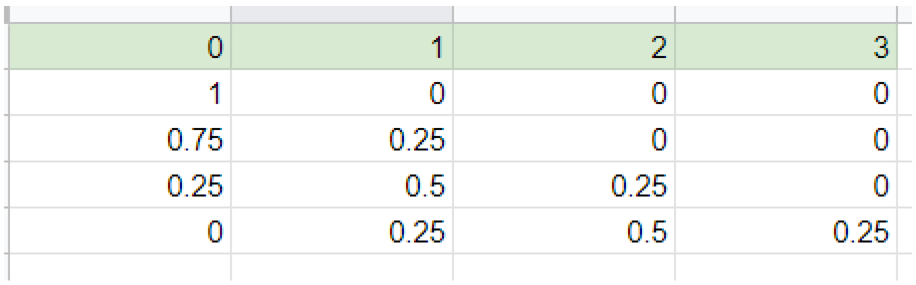
- Windows 7 or higher
- .NET Framework 5.0 or higher
- Python 3.0 or higher
    - Mdptoolbox needs to be installed to computers python library
        - Command line: pip install mdptoolbox
    - Pandas needs to be installed to computers python library
        - Command line: pip install pandas
- Excel (or Google Sheets on the web)

- This is to produce the .xlsx files supported by the application

After having the necessary software installed, the user is also expected to know how to model transition and rewards matrices in excel. The data should have headers at the top with no side headers.

Figure 1: Transition Matrix Data

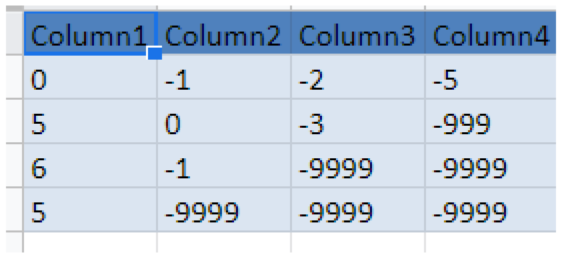
Note: Every sheet in the transition matrix represents an action. The transition matrix is formatted squarely by state space under the header.



0	1	2	3
1	0	0	0
0.75	0.25	0	0
0.25	0.5	0.25	0
0	0.25	0.5	0.25

Figure 2: Reward Matrix Data

Note: the "-9999" values represent states that are unachievable given the current action.



Column1	Column2	Column3	Column4
0	-1	-2	-5
5	0	-3	-999
6	-1	-9999	-9999
5	-9999	-9999	-9999

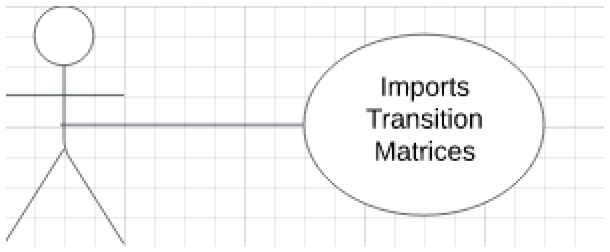
3. System Features and Requirements

3.1 Functional Requirements

3.1.1 User Use Case

Use Case: **Import Transition Matrix on Home Screen**

Diagram:



Brief Description

The user will load in the transition matrix into the processor to be used in the policy generation and value editor from the home screen.

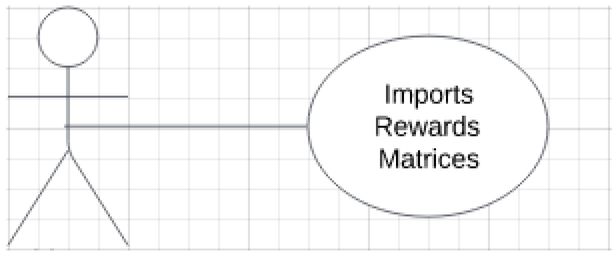
Initial Step-By-Step Description

Before this use case can begin, the user has already launched the application and sees the home screen.

1. The user clicks the “Browse…” button to the right of “Transition File:” in the upper right menu of the home screen
2. User selects .xlsx file that contains their transition matrices on their local machine
3. User presses “Open” or the return key to submit the file to the processor

Use Case: **Import Rewards Matrix on Home Screen**

Diagram:



Brief Description

The user will load in the rewards matrix into the processor to be used in the policy generation and value editor from the home screen.

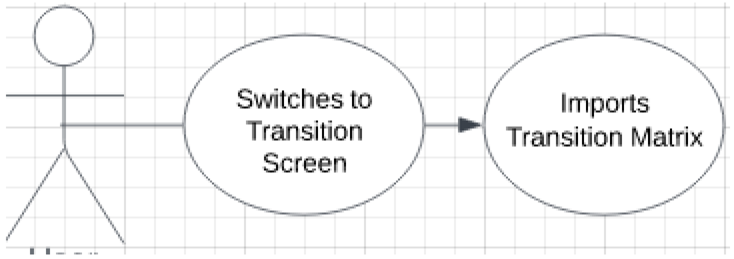
Initial Step-By-Step Description

Before this use case can begin, the user has already launched the application and sees the home screen.

1. The user clicks the “Browse…” button to the right of “Rewards File:” in the upper right menu of the home screen
2. User selects .xlsx file that contains their rewards matrices on their local machine
3. User presses “Open” or the return key to submit the file to the processor

Use Case: **Import Transition Matrix on Transition Screen**

Diagram:



Brief Description

The user will load in the transition matrix into the processor to be used in the policy generation and value editor from the Transition screen.

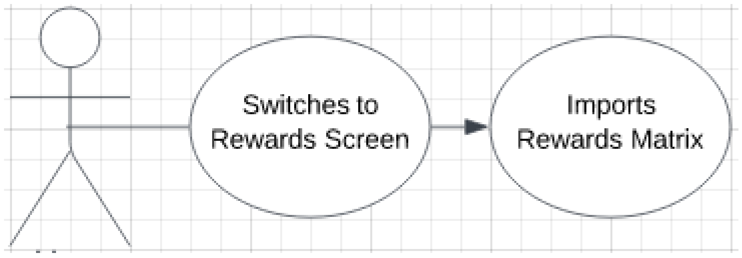
Initial Step-By-Step Description

Before this use case can begin, the user has already launched the application and sees the home screen.

1. The user opens the menu on the left side of the screen by selecting the three bars
2. The user selects the “Transitions” button
3. The user clicks the “Load…” button at the top right of the screen
4. User selects .xlsx file that contains their transition matrices on their local machine
5. User presses “Open” or the return key to submit the file to the processor

Use Case: **Import Rewards Matrix on Rewards Screen**

Diagram:



Brief Description

The user will load in the transition matrix into the processor to be used in the policy generation and value editor from the Rewards screen.

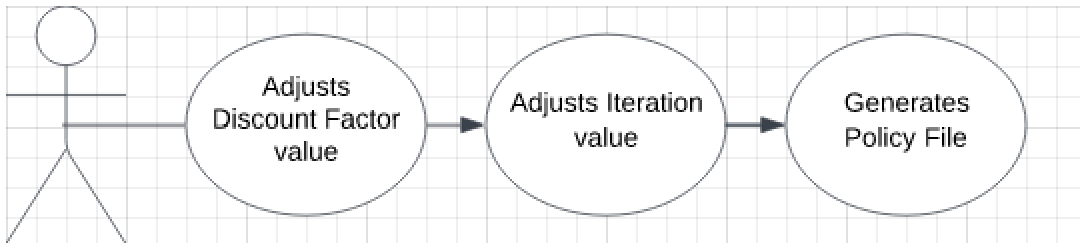
Initial Step-By-Step Description

Before this use case can begin, the user has already launched the application and sees the home screen.

1. The user opens the menu on the left side of the screen by selecting the three bars
2. The user selects the "Rewards" button
3. The user clicks the "Load…" button at the top right of the screen
4. User selects .xlsx file that contains their rewards matrices on their local machine
5. User presses "Open" or the return key to submit the file to the processor

Use Case: **Generate Policy from Home Screen**

Diagram:



Brief Description

The user will generate a policy based on their transition and rewards matrices.

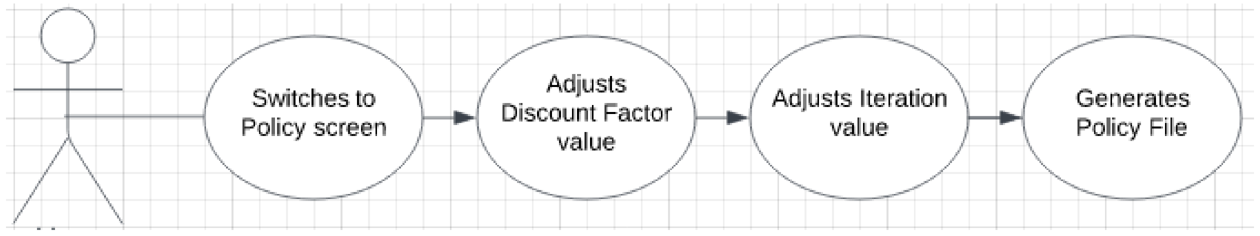
Initial Step-By-Step Description

Before this use case can begin, the user has already launched the application and loaded in their matrices to the application.

1. After loading in the transition and rewards matrix files, the user will adjust the discount factor located beneath the rewards file upload on the home screen
2. The user will adjust the iteration count below the discount factor on the home screen
3. The user will click the "Generate…" button located at the bottom right the menu located at the top right of the home screen
4. The user will name the policy file and click "Save"

Use Case: **Generate Policy from Policy Screen**

Diagram:



Brief Description

The user will generate a policy based on their transition and rewards matrices.

Initial Step-By-Step Description

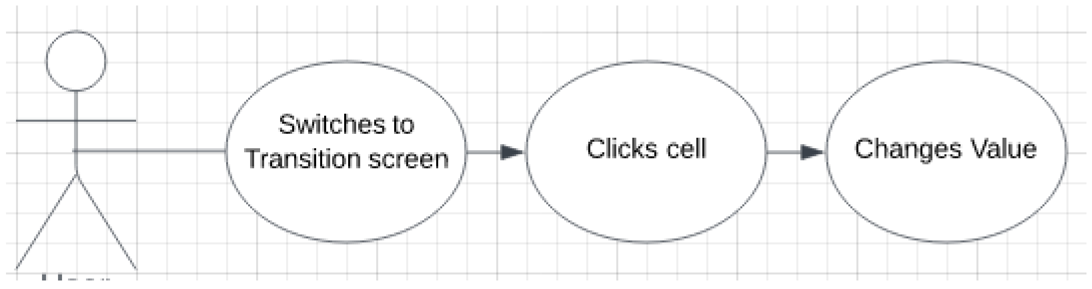
Before this use case can begin, the user has already launched the application and loaded in their matrices to the application.

1. The user opens the menu on the left side of the screen by selecting the three bars
2. The user selects the "Policy" button

3. After loading in the transition and rewards matrix files, the user will adjust the discount factor located to the left of the "Generate…" button at the top right
4. The user will adjust the iteration count to the right of the discount factor
5. The user will click the "Generate…" button located at the bottom right the menu located at the top right of the home screen
6. The user will name the policy file and click "Save"

Use Case: **Edit Transition values in Transition Screen**

Diagram:



Brief Description

The user will be able to edit their transition matrices in the application

Initial Step-By-Step Description

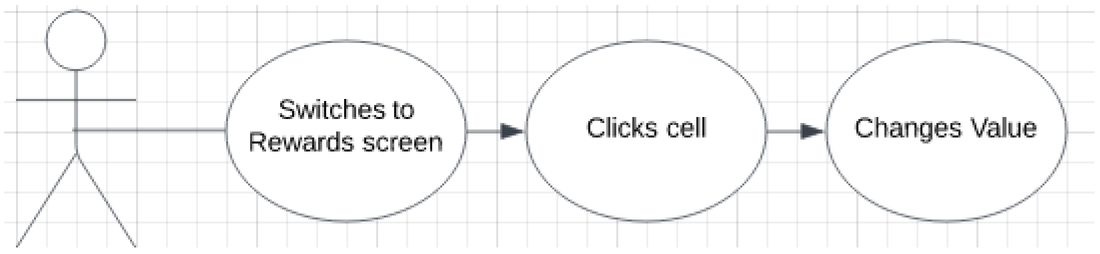
Before this use case can begin, the user has already launched the application and loaded in their matrices to the application.

1. The user will open the menu on the right side by clicking on the three bars
2. The user will select the "Transitions" option from the expanded window
3. The user will be able to select a cell from the grid and type in new values

4. The user will then save the changes to both the application and the file by clicking off the cell or by clicking the return key

Use Case: **Edit Rewards values in Transition Screen**

Diagram:



Brief Description

The user will be able to edit their rewards matrices in the application

Initial Step-By-Step Description

Before this use case can begin, the user has already launched the application and loaded in their matrices to the application.

1. The user will open the menu on the right side by clicking on the three bars
2. The user will select the "Rewards" option from the expanded window
3. The user will be able to select a cell from the grid and type in new values
4. The user will then save the changes to both the application and the file by clicking off the cell or by clicking the return key

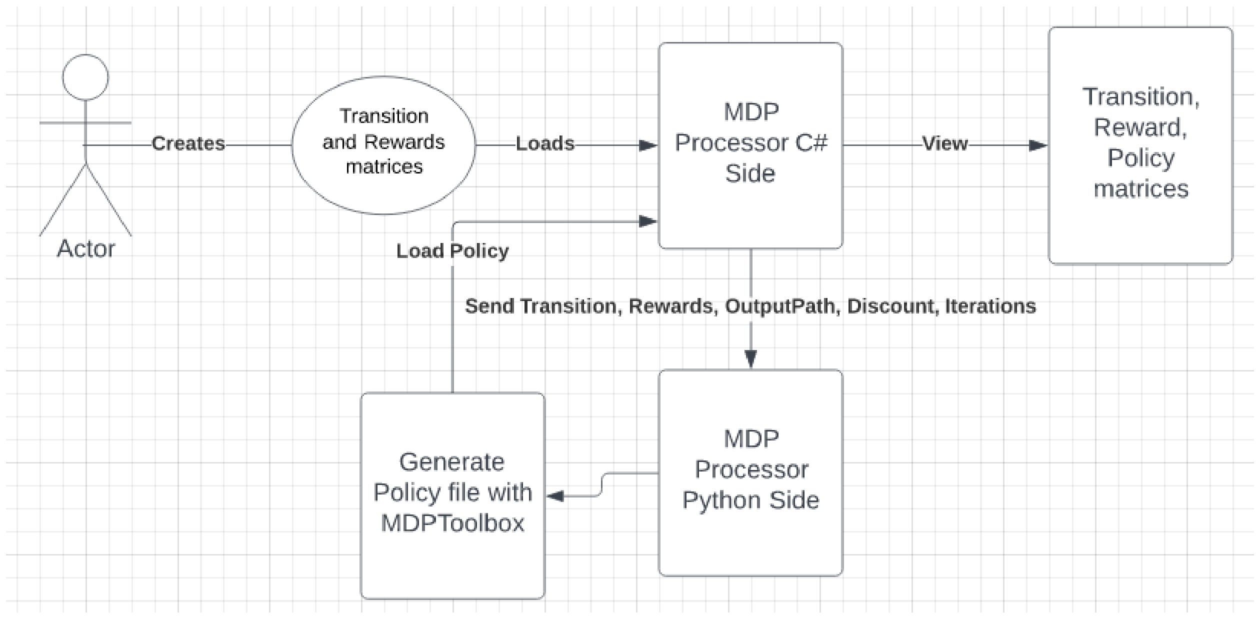
3.2 NonFunctional Requirements

- **Performance:** The application should remain fully functional unless computing a policy. Due to the complexity of computing policies, this may freeze the application.
- **Reliability:** Users can launch and run excel files in the correct format.
- **Availability:** Users can launch the application if they have it downloaded
- **Maintainability:** The application is offline and compiled
- **Recoverability:** If there is a bug, the developer will be able to whip up a fix within a couple days
- **Capacity:** One user per computer, it can only support loading of one rewards file and one transition file along with loading one viewable policy file
- **Manageability:** The code will require a new compilation since it is compiled
- **Data integrity:** All files need to be formatted with headers in the first row. The file may have multiple sheets if the spreadsheet represents a Sheet by Sheet by Action matrix, otherwise it must only have one sheet to represent a State by Action matrix

4. Architecture

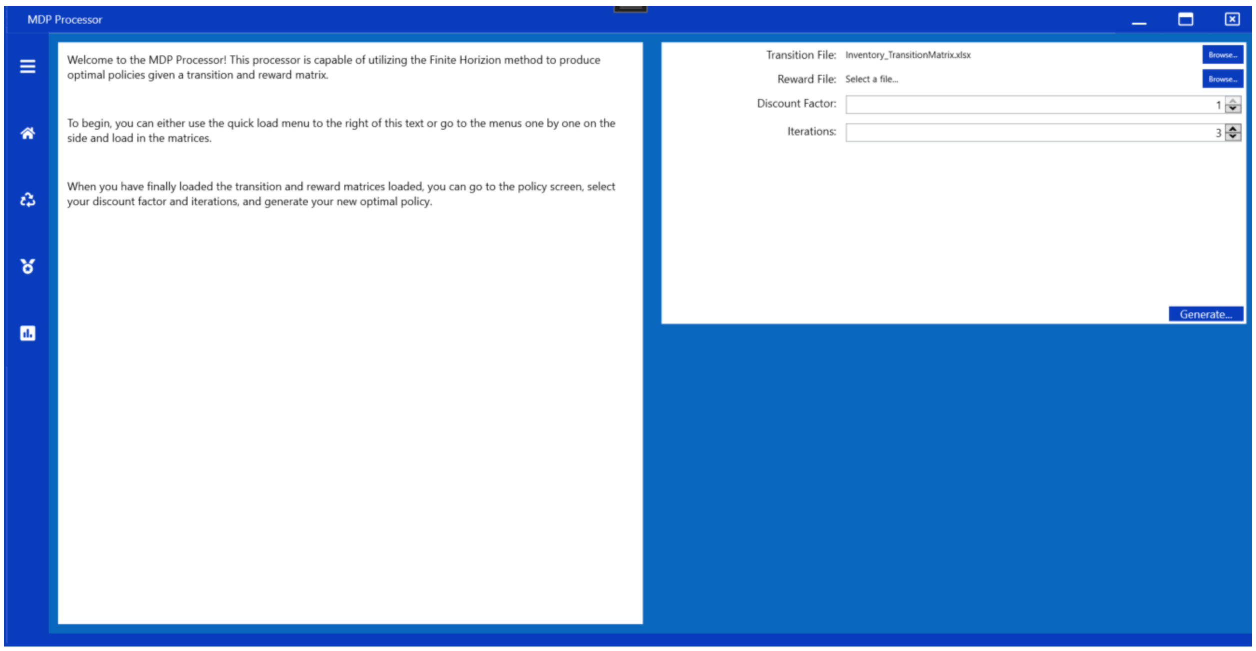
4.1 Visuals and Explanation

Figure 1: Program Architecture design


Actor
Creates
Transition and Rewards matrices
Loads
MDP Processor C# Side
View
Transition, Reward, Policy matrices
Load Policy
Send Transition, Rewards, OutputPath, Discount, Iterations
Generate Policy file with MDPToolbox
MDP Processor Python Side

The application has two sections: A C# side for most of the GUI and basic file processing and a Python side for the MDP processing side. The c# loads the files into the GUI for the user to interact with and then execute and dynamically adjust the python code to accurately compute the desired policy and policy file location.

Figure 2: Home Screen



Welcome to the MDP Processor! This processor is capable of utilizing the Finite Horizion method to produce optimal policies given a transition and reward matrix.

To begin, you can either use the quick load menu to the right of this text or go to the menus one by one on the side and load in the matrices.

When you have finally loaded the transition and reward matrices loaded, you can go to the policy screen, select your discount factor and iterations, and generate your new optimal policy.

Transition File: Inventory_TransitionMatrix.xlsx

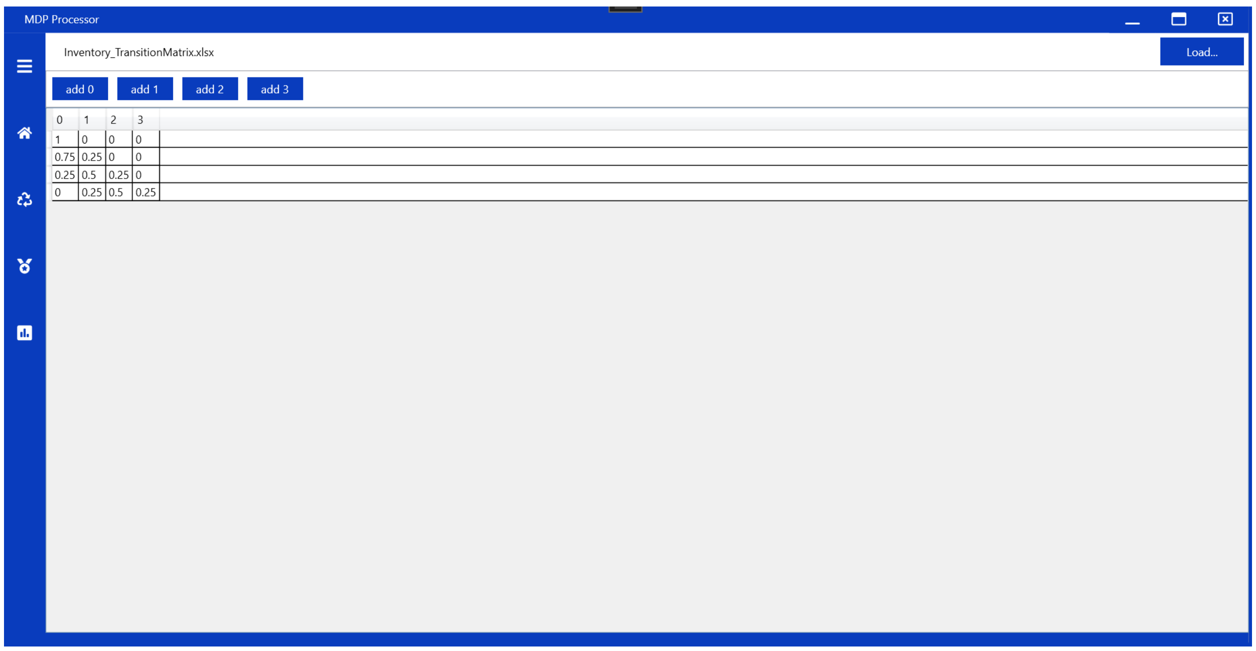
Reward File: Select a file...

Discount Factor: 1

Iterations: 3

On the left of Figure 2 you see the menu with the different menu options. To the right of the menu is the screen. In the screen, on the left, is a description of the application and how to get started. On the right side, you see a quick start menu to quickly load the matrices, adjust the discount factor and iterations, and generate a policy.

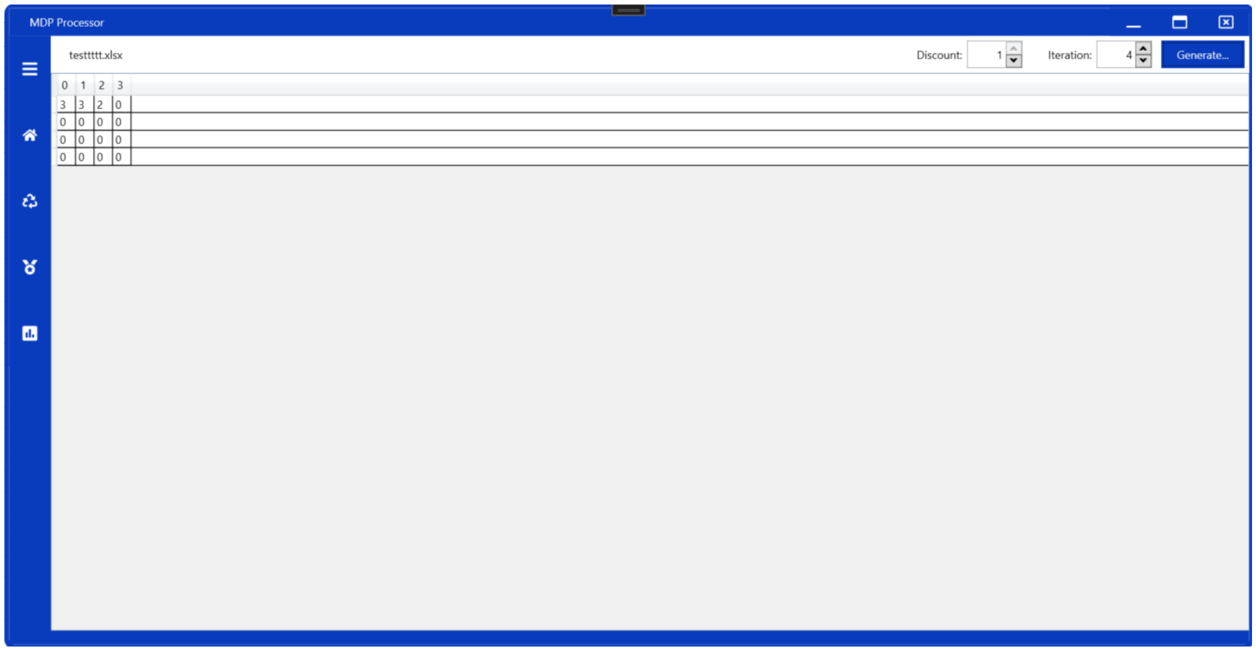
Figure 3: Transition and Rewards Screen



0	1	2	3
1	0	0	0
0.75	0.25	0	0
0.25	0.5	0.25	0
0	0.25	0.5	0.25

In figure 3, you see the transition/rewards screen. Loaded is an example from a MDP book related to stochastic inventory. You can see the loaded data in the first sheet to the right of the menu. You can choose to reload the data or swap it out at the top right using the “load” button. You can switch between the sheets with the buttons under the file name at the top left of the screen. The buttons are labeled by whatever the sheet name was in the excel file.

Figure 4: Policy Screen



testttttt.xlsx

Discount: 1

Iteration: 4

0	1	2	3
3	3	2	0
0	0	0	0
0	0	0	0
0	0	0	0

Figure 4 shows the policy screen, really similar to the rewards/transition screen. The only difference for this screen is that the load button is replaced with a generation button. The datagrid is also not editable.

5. Setup

5.1 Installing Dependencies

The application requires to be launched on a windows machine and uses the .NET framework. To install the .NET framework please navigate to the following URL and follow the site's instructions to download it for your machine…

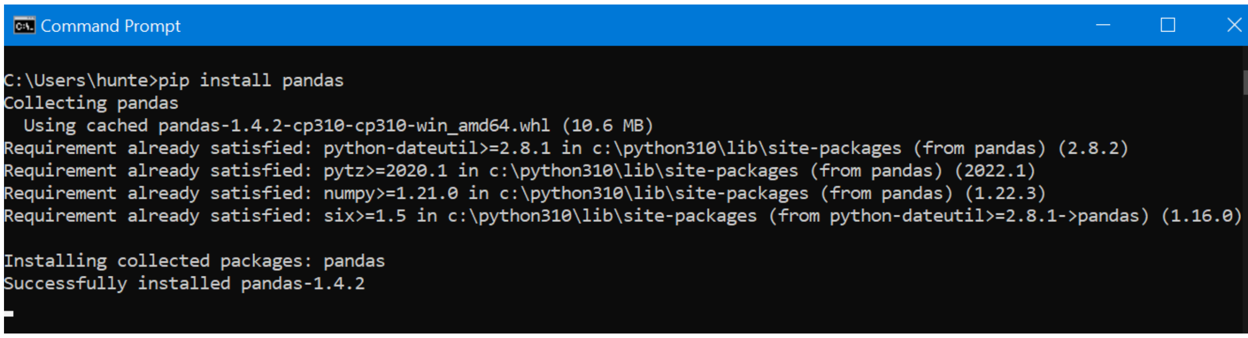
https://dotnet.microsoft.com/en-us/download/dotnet/6.0

After installing the framework, next is to install python. To install python, navigate to the following URL and follow the instructions to install python3…

https://www.python.org/downloads/

After installing python, you need to install the necessary libraries used by the MDP Processor on the python side. Open up a command prompt and type in the following commands:

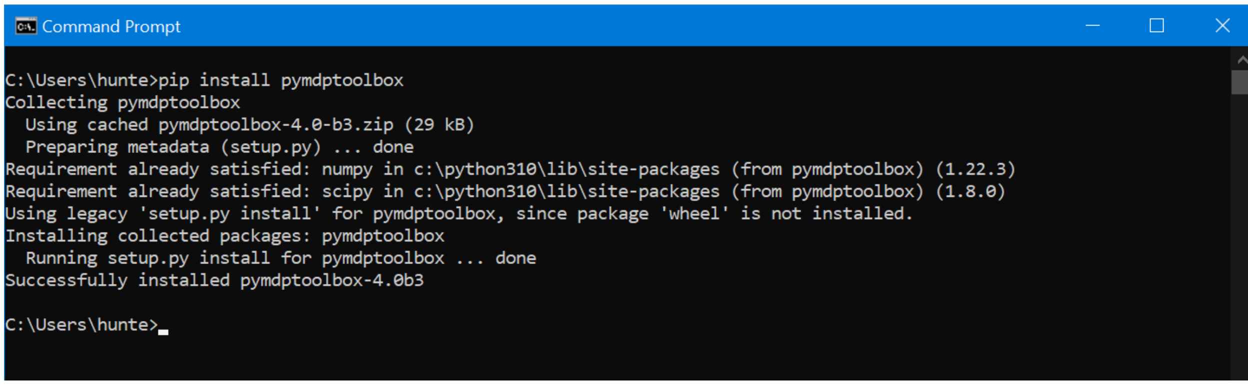
- pip install pandas



```
C:\Users\hunte>pip install pandas
Collecting pandas
  Using cached pandas-1.4.2-cp310-cp310-win_amd64.whl (10.6 MB)
Requirement already satisfied: python-dateutil>=2.8.1 in c:\python310\lib\site-packages (from pandas) (2.8.2)
Requirement already satisfied: pytz>=2020.1 in c:\python310\lib\site-packages (from pandas) (2022.1)
Requirement already satisfied: numpy>=1.21.0 in c:\python310\lib\site-packages (from pandas) (1.22.3)
Requirement already satisfied: six>=1.5 in c:\python310\lib\site-packages (from python-dateutil>=2.8.1->pandas) (1.16.0)

Installing collected packages: pandas
Successfully installed pandas-1.4.2
```

- pip install pymdptoolbox



```
C:\Users\hunte>pip install pymdptoolbox
Collecting pymdptoolbox
  Using cached pymdptoolbox-4.0-b3.zip (29 kB)
  Preparing metadata (setup.py) ... done
Requirement already satisfied: numpy in c:\python310\lib\site-packages (from pymdptoolbox) (1.22.3)
Requirement already satisfied: scipy in c:\python310\lib\site-packages (from pymdptoolbox) (1.8.0)
Using legacy 'setup.py install' for pymdptoolbox, since package 'wheel' is not installed.
Installing collected packages: pymdptoolbox
  Running setup.py install for pymdptoolbox ... done
Successfully installed pymdptoolbox-4.0b3

C:\Users\hunte>_
```

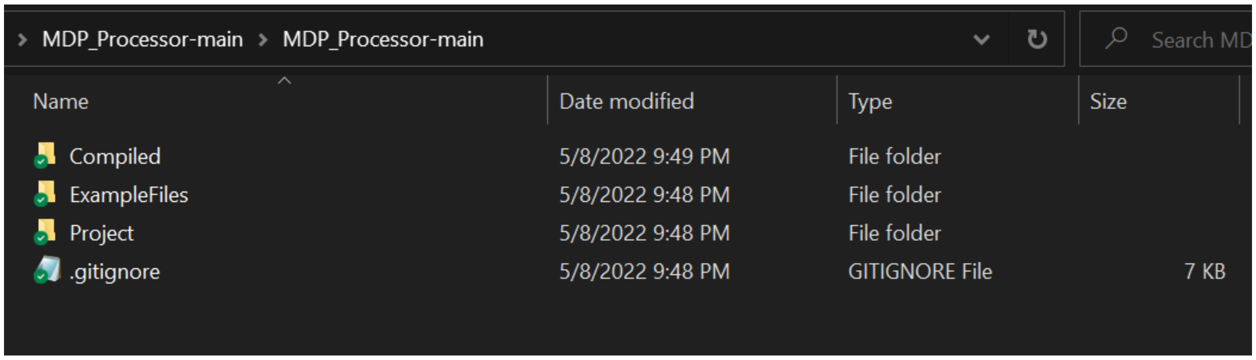
After installing these python libraries, you should be able to fully use the MDP Processor.

5.2 Getting files from git

To install all the files for the project, navigate to the following URL and download the project:

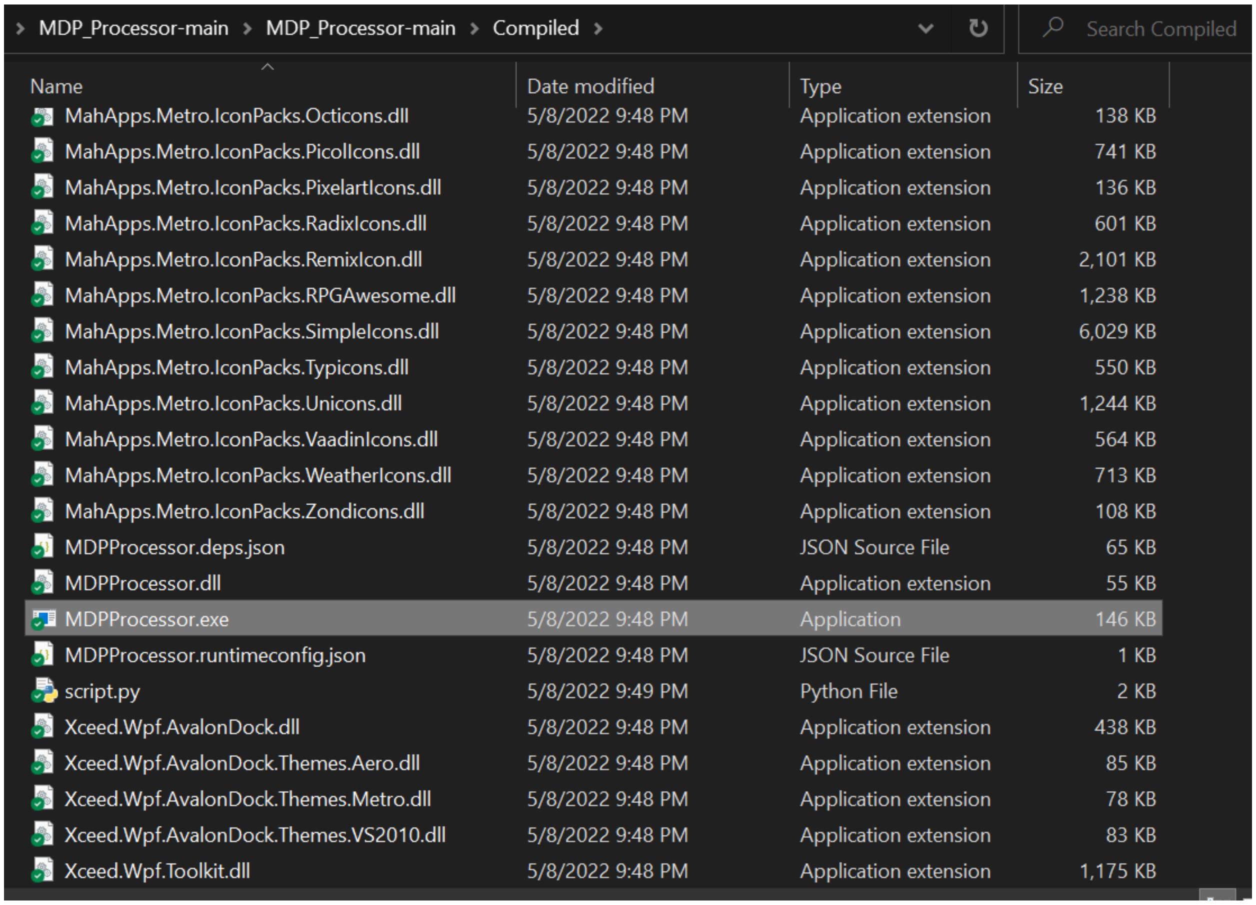
https://github.com/HVallad/MDP_Processor

After downloading the project, unzip the folders and navigate to the compiled folder to run the application.



Name	Date modified	Type	Size
Compiled	5/8/2022 9:49 PM	File folder	
ExampleFiles	5/8/2022 9:48 PM	File folder	
Project	5/8/2022 9:48 PM	File folder	
.gitignore	5/8/2022 9:48 PM	GITIGNORE File	7 KB

In this directory you will find a MDPProcessor.exe. Double click to launch.



› MDP_Processor-main › MDP_Processor-main › Compiled ›

Name	Date modified	Type	Size
MahApps.Metro.IconPacks.Octicons.dll	5/8/2022 9:48 PM	Application extension	138 KB
MahApps.Metro.IconPacks.PicolIcons.dll	5/8/2022 9:48 PM	Application extension	741 KB
MahApps.Metro.IconPacks.PixelartIcons.dll	5/8/2022 9:48 PM	Application extension	136 KB
MahApps.Metro.IconPacks.RadixIcons.dll	5/8/2022 9:48 PM	Application extension	601 KB
MahApps.Metro.IconPacks.RemixIcon.dll	5/8/2022 9:48 PM	Application extension	2,101 KB
MahApps.Metro.IconPacks.RPGAwesome.dll	5/8/2022 9:48 PM	Application extension	1,238 KB
MahApps.Metro.IconPacks.SimpleIcons.dll	5/8/2022 9:48 PM	Application extension	6,029 KB
MahApps.Metro.IconPacks.Typicons.dll	5/8/2022 9:48 PM	Application extension	550 KB
MahApps.Metro.IconPacks.Unicons.dll	5/8/2022 9:48 PM	Application extension	1,244 KB
MahApps.Metro.IconPacks.VaadinIcons.dll	5/8/2022 9:48 PM	Application extension	564 KB
MahApps.Metro.IconPacks.WeatherIcons.dll	5/8/2022 9:48 PM	Application extension	713 KB
MahApps.Metro.IconPacks.Zondicons.dll	5/8/2022 9:48 PM	Application extension	108 KB
MDPProcessor.deps.json	5/8/2022 9:48 PM	JSON Source File	65 KB
MDPProcessor.dll	5/8/2022 9:48 PM	Application extension	55 KB
MDPProcessor.exe	5/8/2022 9:48 PM	Application	146 KB
MDPProcessor.runtimeconfig.json	5/8/2022 9:48 PM	JSON Source File	1 KB
script.py	5/8/2022 9:49 PM	Python File	2 KB
Xceed.Wpf.AvalonDock.dll	5/8/2022 9:48 PM	Application extension	438 KB
Xceed.Wpf.AvalonDock.Themes.Aero.dll	5/8/2022 9:48 PM	Application extension	85 KB
Xceed.Wpf.AvalonDock.Themes.Metro.dll	5/8/2022 9:48 PM	Application extension	78 KB
Xceed.Wpf.AvalonDock.Themes.VS2010.dll	5/8/2022 9:48 PM	Application extension	83 KB
Xceed.Wpf.Toolkit.dll	5/8/2022 9:48 PM	Application extension	1,175 KB

Note: If you use the input files provided in the repository, you should see a policy screen similar to that of Figure 4 in Section 4.1

"I have neither given nor received unauthorized aid in completing this work, nor have I presented someone else's work as my own." -Hunter Vallad