CAPIR User Manual

Truong-Huy Dinh Nguyen
truonghuy@gmail.com

Table of Contents

Introduction

Objectives

Simplify the design process of collaborative games by automating the AI authoring phase. The new design process is as follows.

1. Design a level using the popular map editor Tiled
2. Run the game solver to automatically compute the action policy for the helping assistant
3. Playtest the level
4. If not satisfied with the level, go back to 1 to fix.
5. Otherwise, save the level, go back to 1 to design a new one.

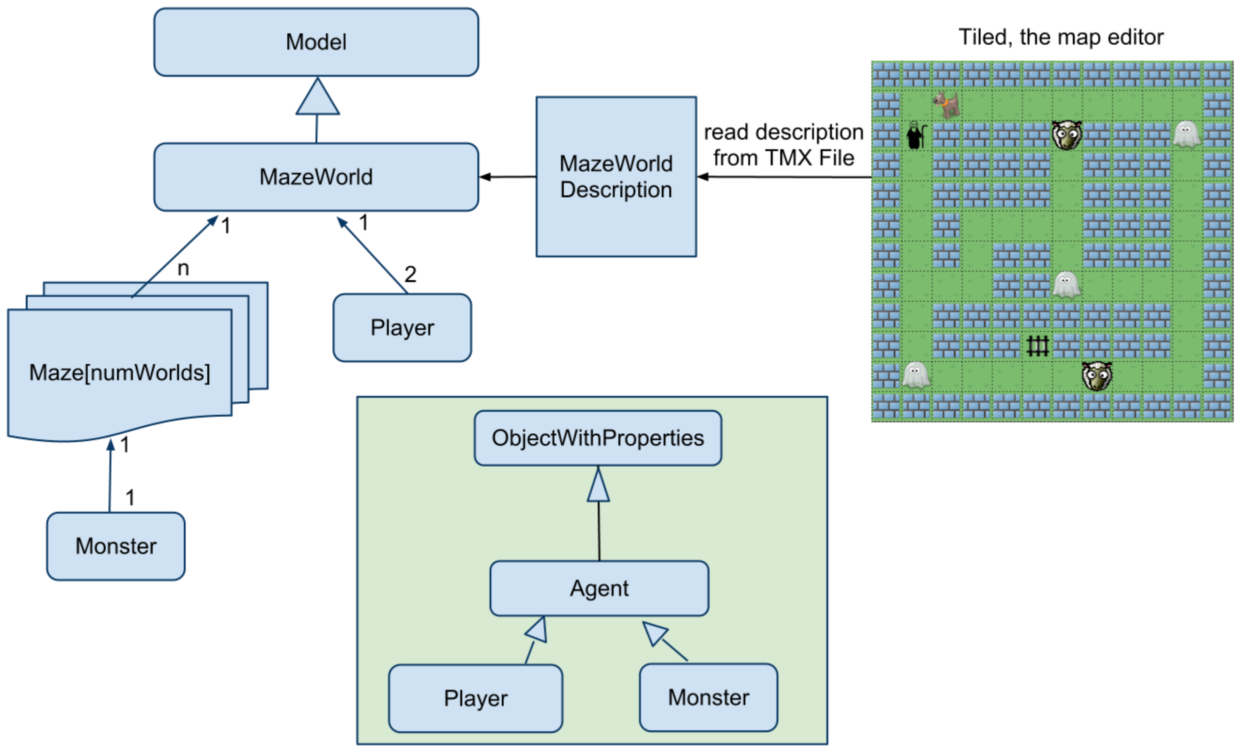
Overview

CAPIR uses a multiple-world model to represent one game level. Intention recognition helps detect the human player's targeted world, which is used by the AI assistant to retrieve the optimal collaborative actions to perform.

Features

1. Objective achieved for maze-like puzzle games (such as GhostBusters). By having a 3D GUI on top of the maze environment, we can design sophisticated games with good collaborative behaviors.
2. Technical: Typically a new game can be created by extending the following set of classes
    - **Player**, the base class of two players in the game.
        i. By default, each player has 5 actions (no_move, N, E, S, W) and no property.
        ii. (optional) a set of special actions (e.g. fight, shoot)
        iii. (optional) properties, e.g. hit points.
    - **Monster**, the base class of NPCs in the game. Note that in our design, every NPC (sheep, ghost, etc.) is attached to one Maze object.
        i. Behaviors when not nearby any player and/or seeing agent/assistant. By default, all NPCs run away from players.
        ii. (optional) a set of special actions
        iii. (optional) properties
    - **Maze**, representing one world/single task.
        i. Terminal conditions
        ii. Reward conditions

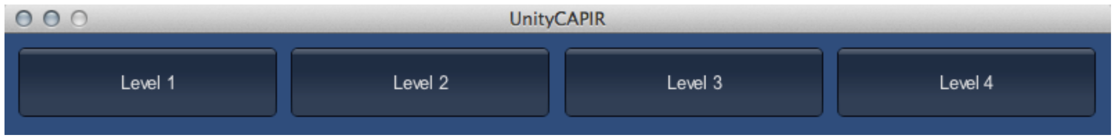
Class Hierarchy



Running SheepSavior, the Demo Game

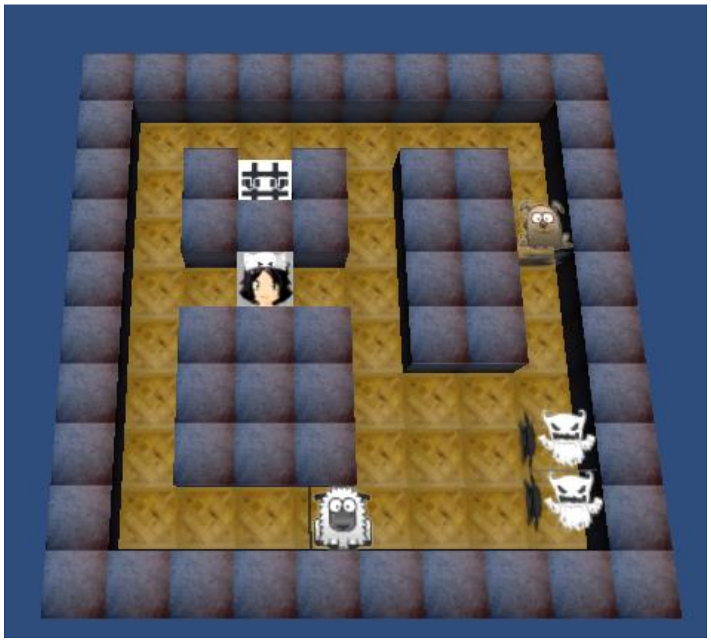
Step 1: Run SheepSaviorUnity

- After Unity Splash Screen, a list of available levels is shown.



Step 2: Select a level (1-4) and start controlling the Girl avatar.

- Arrow buttons are for movement. Space bar is for attacking a nearby ghost (within 3 grids square away).
- The tasks are
  - Herding Sheep into the Pen marked by
  - Killing all Ghosts
  - Herding Fiery into the grid marked by , while preventing them from moving into
- Observe the Dog's collaborative behaviors in the process. Enjoy the game!!



Trying SheepSavior with New Levels

Demo clip: http://www.youtube.com/watch?v=nTYQ2Oo1udA

1. Compile the CAPIR Solver

Step 1: Obtain source code.
Extract the source code tar.gz file to a convenient place.

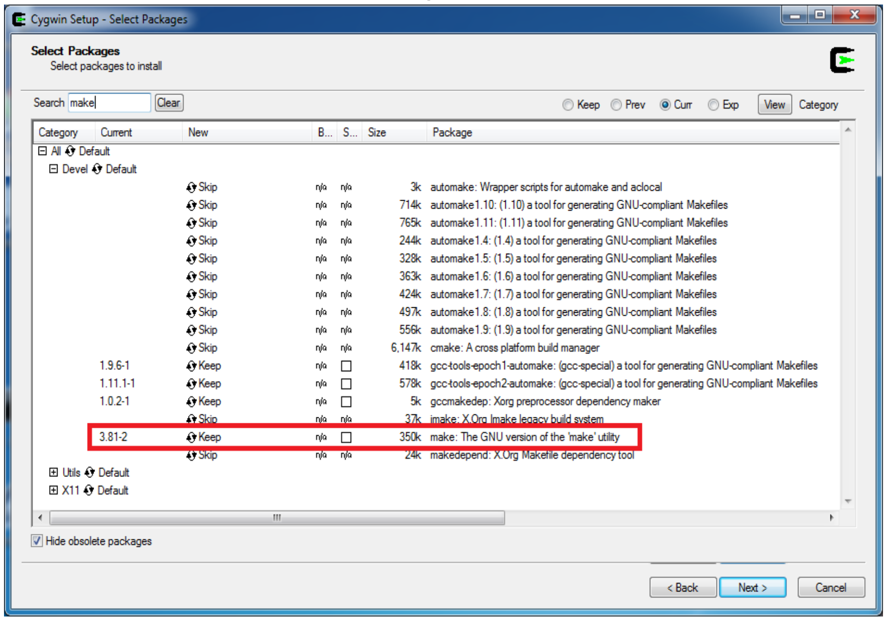
Step 2: Compile.
1. Browse to: CAPIRSolver/games/SheepSavior/bin/
2. Compile: $ touch Makefile.dep && make depend && make

This step compiles the code necessary for running our example game, SheepSavior. The engine is compiled as part of the game. Future games can be written in the same manner.

Note: Ubuntu 10.10 (Maverick Meerkat) and Snow Leopard come with zlib installed, so there should not be any problem yet if you are using any of these OSes. In other cases, if there is any error message regarding the missing of zlib, you should be able to download the newest version of zlib at http://www.zlib.net/ (free) and install it.

Note: Windows users can use Cygwin (http://www.cygwin.com/). To do so:

1. Download and run the cygwin installer (http://cygwin.com/setup.exe)
2. Press 'Next' until reaching the 'Select Packages' screen.
3. Search for 'gcc.' Change the status of 'Devel' from 'Default' to 'Install.' This can be done by clicking the arrow icon
4. Search for 'make.' Expand 'Devel' and change the status of 'make' to 'Install'



Cygwin Setup - Select Packages

Select Packages
Select packages to install

Search: make — Keep / Prev / [✓ Curr] / Exp — View: Category

Category	Current	New	B...	S...	Size	Package
⊟ All Default						
⊟ Devel Default						
		Skip	n/a	n/a	3k	automake: Wrapper scripts for automake and aclocal
		Skip	n/a	n/a	714k	automake1.10: (1.10) a tool for generating GNU-compliant Makefiles
		Skip	n/a	n/a	765k	automake1.11: (1.11) a tool for generating GNU-compliant Makefiles
		Skip	n/a	n/a	244k	automake1.4: (1.4) a tool for generating GNU-compliant Makefiles
		Skip	n/a	n/a	328k	automake1.5: (1.5) a tool for generating GNU-compliant Makefiles
		Skip	n/a	n/a	363k	automake1.6: (1.6) a tool for generating GNU-compliant Makefiles
		Skip	n/a	n/a	424k	automake1.7: (1.7) a tool for generating GNU-compliant Makefiles
		Skip	n/a	n/a	497k	automake1.8: (1.8) a tool for generating GNU-compliant Makefiles
		Skip	n/a	n/a	556k	automake1.9: (1.9) a tool for generating GNU-compliant Makefiles
		Skip	n/a	n/a	6,147k	cmake: A cross platform build manager
	1.9.6-1	Keep	n/a	☐	418k	gcc-tools-epoch1-automake: (gcc-special) a tool for generating GNU-compliant Makefiles
	1.11.1-1	Keep	n/a	☐	578k	gcc-tools-epoch2-automake: (gcc-special) a tool for generating GNU-compliant Makefiles
	1.0.2-1	Keep	n/a	☐	5k	gccmakedep: Xorg preprocessor dependency maker
		Skip	n/a	n/a	37k	imake: X.Org Imake legacy build system
	3.81-2	Keep	n/a	☐	350k	make: The GNU version of the 'make' utility
		Skip	n/a	n/a	24k	makedepend: X.Org Makefile dependency tool
⊞ Utils Default						
⊞ X11 Default						

[✓] Hide obsolete packages

< Back | Next > | Cancel

5. Make the following changes to CAPIRSolver/games/SheepSavior/bin/Makefile

```
# name of dependency file
DEPFILE = Makefile.dep

# solver, WorldModels and problem directories
UTILS = ../../utils/
WORLDMODELS = ../../WorldModels/
GAMESRC = ../src/
# the following line is included to compile on Cygwin
#- include folder in cygwin, this may be different on your system
ZLIB = /cygdrive/c/cygwin/usr/include
#- folder where cygz.dll resides, this may be different on your system
LDFLAGS=-L/cygdrive/c/cygwin/bin/

# include directories, now with ZLIB
INCDIR = -I$(UTILS) -I$(GAMESRC) -I$(WORLDMODELS) -I$(ZLIB)

# Change this line if you want a different compiler
# CXX = g++ -ggdb -Wall -W -lz $(INCDIR)
# CXX = g++ -ggdb -lz $(INCDIR)
CXX = g++ -O2 -lz $(INCDIR) $(LDFLAGS)

...// here onwards everything is the same
```

2. Install Tiled for level editor

Download newest version of Tiled at http://www.mapeditor.org/

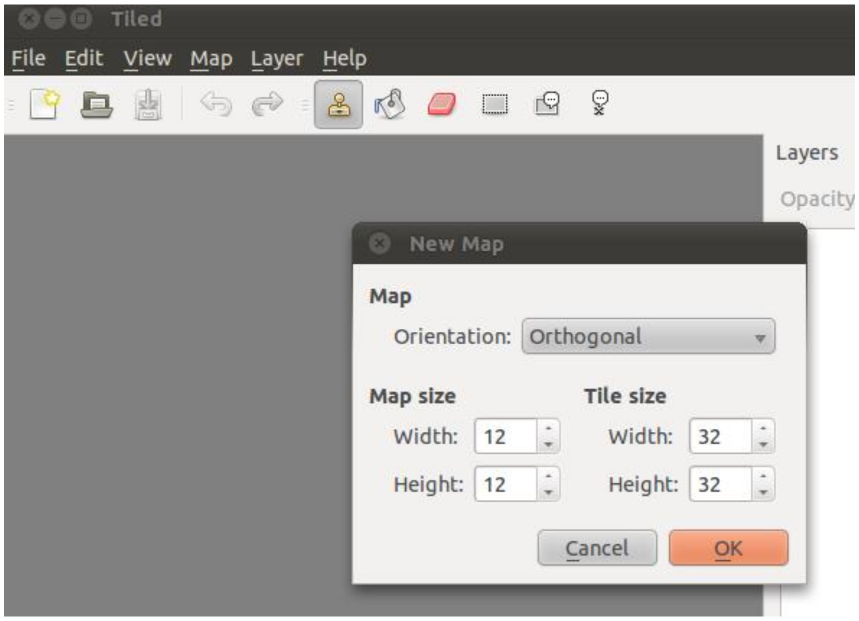
- Mac OS X (Snow Leopard and above).
  Download the disk image, mount it, drag and drop the application to Applications folder.
- Ubuntu (10.10 Maverick and above) This may still work for Ubuntu 9.10 (Karmic)
  a. Add the Tiled's PPA to Software Source:
     $ sudo add-apt-repository ppa:duck-wallace/tiled
  b. Refresh software list:
     $ sudo apt-get update
  c. Install Tiled via apt-get
     $ sudo apt-get install tiled
  d. You can find Tiled on the menu Applications/Graphics/Tiled

3. Design new levels for SheepSavior

Step 1: Open Tiled to design a sample level

1. Open Tiled
2. Click File/New...
3. In the New Map dialog
   a. Choose Map size around 12 by 12. Larger map takes the AI engine longer to solve, so for a quick demo, 12 by 12 is reasonably fast.

b. Leave Tile size at default values of 32 by 32



Map

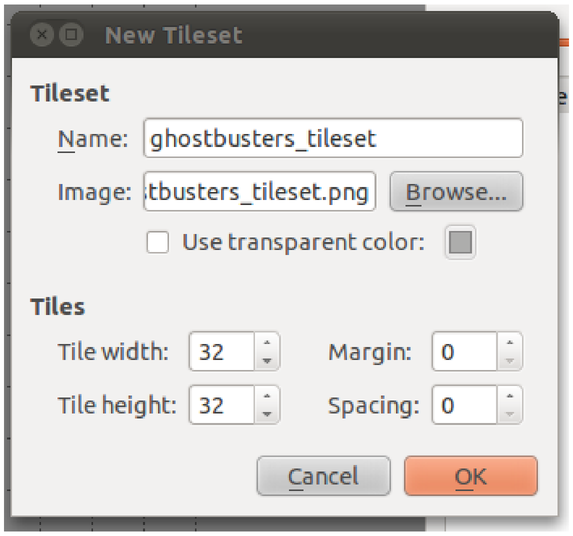
Orientation: Orthogonal

Map size		Tile size	
Width:	12	Width:	32
Height:	12	Height:	32

4. Press OK

Step 2: Load the predesigned Tileset

1. Click Map/New Tileset...
2. Click Browse
   - Browse to folder CAPIRSolver/games/SheepSavior/levels/ghostbusters_tileset.png



Tileset

Name: ghostbusters_tileset

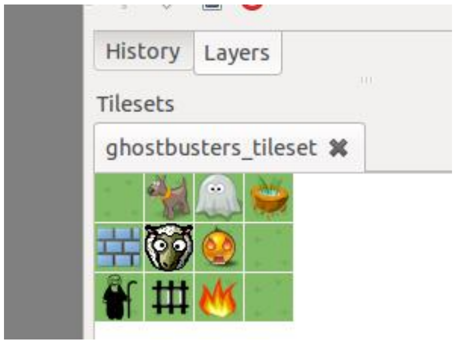
Image: :tbusters_tileset.png

Use transparent color: ☐

Tiles

Tile width:	32	Margin:	0
Tile height:	32	Spacing:	0

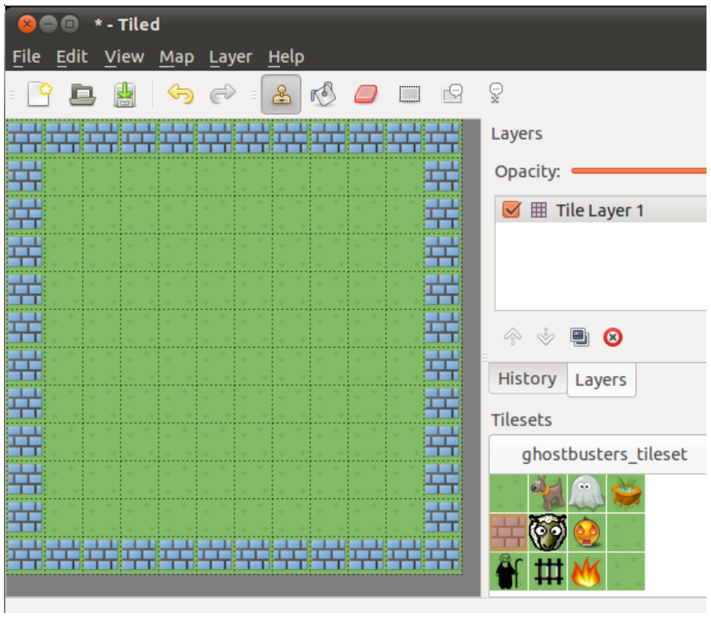
3. Press OK
   - On the right, the tileset is shown



Note: To denote free space, only use the grass tile in the upper-left corner of the tile set. Using the others (last two) for that purpose will cause an error.

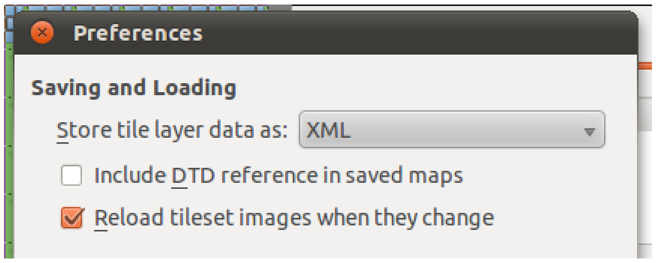
Step 3: Fill the map with Grass tile and build Wall around the map

1. Click icon on the menu bar.
   - This Fill icon fills the whole map with a selected tile.
2. Click , the Grass tile, the first tile in the tileset loaded.
3. Click on the map.
   - The whole map is fully populated by the Grass tile.
4. Click icon on the menu bar
   - This Stamp icon fills one map grid square with a selected tile.
5. Click , the Wall tile, second tile in the tileset.
6. Click along the edge grid squares of the map to form a boundary wall around it. You can click and drag to draw faster.
   - If things go wrong and you accidentally draw a wall at unwanted places, click the Grass tile and place it on locations you want the Green back.



Step 4 (Important!!!): Set Preferences to save the map as XML

1. Click Edit/Preferences...
2. Choose XML from the drop-down menu of "Store tile layer data as:"



Preferences

Saving and Loading

Store tile layer data as: XML

☐ Include DTD reference in saved maps

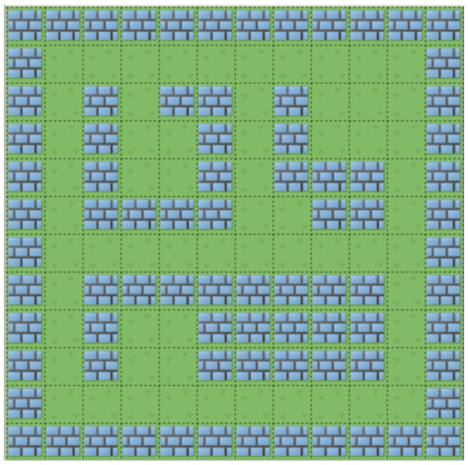
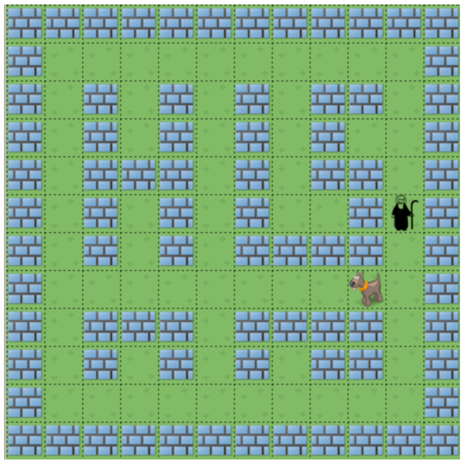
☑ Reload tileset images when they change

3. Click Close

Step 5: Save the map to folder CAPIRSolver/games/SheepSavior/levels/

1. Click
2. Browse to **CAPIRSolver/games/SheepSavior/levels/**
3. Save the map with .tmx extension, for instance, level1.tmx (I recommend saving the tmx file as level1 to level4.tmx)

Step 6: Continue to draw inner walls for the map (i.e. let your creativity go wild).
Choose the Stamp icon and use Wall tile to draw walls. Note that as the number of free grids (grass tiles) increases, the time taken to solve a level increases polynomially as well. Some sample layouts are

Step 7: Place predefined Agents on the map

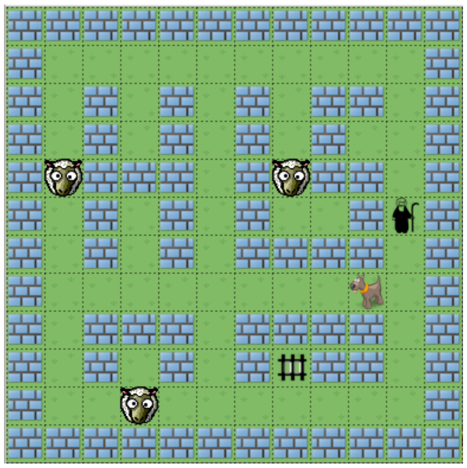
1. One Human (the black shepherd)
2. One Dog ( )

You can place them anywhere in the free space (non-wall).

Step 8: Place Monster NPCs on the map

In this SheepSavior game, we have designed three sample types of monsters, Sheep, Ghost and Fiery (experimental!! Dod does not behave very well with Fiery) with different behaviors to play around with. For example, if we place sheep in the map, the goal of the game is to herd all the sheep to a designated pen.

1. Place Sheep tiles ( ) at various places.
   - The amount of monsters of any kind is not restricted, but the more monsters there are the more confused the dog can be about the player's targeted NPC.
2. Place one Pen tile ( ) at one place.



Important!! When you have some Sheep in the map, you must place one and only one Pen somewhere.

With Fiery ( ), there must be one Fire ( ) and one Water Pot ( ) in the map. The Fiery have the tendency to rush to the Fire. The Shepherd and Dog need to herd them to the water pot.

Step 9: Save the map

Click

4. Solve the Level

Suppose the level was saved with name "level1.tmx" in folder CAPIRSolver/games/SheepSavior/levels/. From the folder CAPIRSolver/games/SheepSavior/bin, run

```
$ ./GhostBustersSolver -m ../levels/level1.tmx
```

It may take up to 4, 5 minutes to solve, depending on how many types of NPCs there are in the map and the number of free grids. After solving, the program produces

- level1.tmx.sheep.0.Ftn if there is a Sheep in the map
- level1.tmx.ghost.0.Ftn if there is a ghost in the map

- level1.tmx.fiery.0.Ftn if there is a Fiery in the map.

Note that the number of NPCs does not affect the solving time because a NPC type only gets solved once.

5. Run the Level for Playtest

Copy the .tmx file and all corresponding *.Ftn files into the Resources folder of the companying Mac executable (SheepSaviorUnity).

1. Right click on the executable, select "Show Package Contents"
2. Browse to Contents/Resources
3. Copy the TMX file that you designed together with its Ftn files into this folder.
    a. Note that since the menu only has option to choose level from 1 to 4, you should have the new level and its Ftn files named as "level1" to 4.
    b. You may find level1.tmx to level4.tmx and their accompanying Ftn files already residing in the folder. Just overwrite, perhaps after backing up the overwritten files elsewhere.

6. Play the newly designed level.

Refer to Section II above.

Create new NPCs with different behaviors

This is where the codes of the game need to be altered. Previous tasks do not need any code change.

- As mentioned in the Introduction, the solver computes the optimal collaborative policy for the Assistant, i.e. Dog in SheepSavior, based on how the game state changes (*dynamics*) and corresponding rewards with respect to each executed action. The goal of the Assistant is to select actions that maximize its long-term reward with respect to the dynamics and rewarding system of the game.
- Therefore, in order for the Assistant to cooperate successfully with the Player when dealing with new NPCs, the ***dynamics*** and ***reward function*** of this NPC must be specified in C++ for the Game Solver.
- The same functions need to be reproduced in the Frontend (Unity) in the form of C# or JavaScript. While the same reward function does not need exact replication, the NPC's dynamics does. If there is an implementation mismatch in the NPC's dynamics between the solver and frontend, it is similar to training an Assistant to behave in one way and expecting it to excel in a different environment. The assistant will fail or exhibit bizarre behavior.

Details on Classes and Routines to customize are in the Tech Doc.