Game Theory with GamePlan

Chapter Two: The *GamePlan* Software[*]

2.1 Purpose of the Software

One of the greatest challenges in teaching and doing research in game theory is computational. Although there are powerful theoretical results that ensure the existence of equilibria, few methods exist to calculate them reliably and extensively. And when such methods do exist their implementation requires computers in all but the simplest of games. Moreover, techniques that apply to certain kinds of games are usually not applicable to other kinds. And for many kinds of games there are no wholly reliable methods of obtaining equilibria. But the advent of the personal computer has created opportunities for the design of software that can bring some remedy to this challenge. *GamePlan* was created for that purpose and was designed to meet four basic requirements:

• Be User friendly. This means that games should be easy to create, edit and solve. This led to a heavy reliance on color-coding for the players, nodes, moves, information sets, and payoffs.

• Be Flexible. The data structure and solving algorithms are designed to accommodate almost any game structure that meets the simple requirements that there be finitely many players, moves and nodes in the extensive form, and strategies in the normal form. In practice, games cannot exceed some limits due to memory and speed requirements.

• Be Exhaustive. There are two solving options (Pure and Mixed) but only one solution concepts: a generic Perfect Equilibrium that translates into subgame perfect or perfect Bayesian in the extensive form, Markov perfect in the graph form and simply Nash in the normal form. In each case, *GamePlan* attempts to obtain as many equilibria as possible, as well as the corners of equilibrium sets when appropriate.

• Be Fast. Given the need for exhaustive solving, and despite the optimization of the algorithms, the solving of complex games can be quite slow. It is difficult to give a priori estimates of speed given the great variety of games that *GamePlan* can handle. As the solving proceeds, a dialog box displays the percentage of the game solved and allows the user to inspect the solutions already obtained. This gives constant feedback to the user as to how much more time to expect before completion.

[*]Copyright © 2019-2021, Jean-Pierre P. Langlois. All rights reserved. No part of this publication may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without explicit permission from the author.

2.2 Program Set Up

GamePlan runs under Microsoft Windows 32 and 64-bit only. The installer program for version 4.0 (GP4Setup.exe) can be downloaded at:

http://faculty.sfsu.edu/~langlois

There is no need for a license to run *GamePlan*. There is absolutely no cost to you of using it.

The opening screen of *GamePlan* looks like Figure 2.1 (after opening the game file Simplest.gpx):

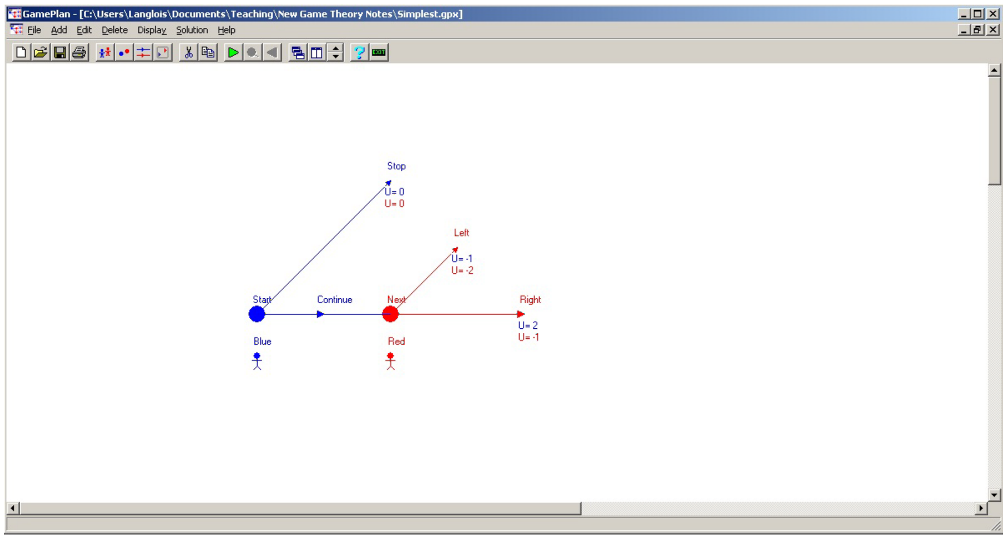
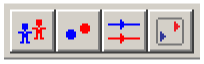


Figure 2.1: The GamePlan Desktop

2.3 Using GamePlan

GamePlan operates just like any other Windows-based program with pull-down menus, speed buttons, and dialog boxes. The File menu, together with the first four buttons (New, Open, Save and Print):

allows file operations, and printing. The Add menu, together with the next four buttons (Player, Node, Move, Table):



allows adding game objects, respectively, players, nodes, moves and tables, to a new or edited game, with tables being used for the normal form. The Edit menu allows the editing of an existing game. The Solution menu together with the four buttons (Solve, Display, Hide and Delete):

allows the solving of a game and the display of its solutions. The Display menu, together with the next three buttons (Cascade, Tile, Zoom In and Out):

allows manipulating the display of games and their solutions, as well as exporting what is displayed to a graphics file for use in other documents. Finally, the help menu, together with the two buttons (Help, Exit):

allows calling the Help file or exiting *GamePlan*.

As in most modern programs, menu items and buttons are activated or deactivated depending on the circumstances. One can open or create several games at the same time but one cannot transfer objects from one game to another.

2.4 Creating, Opening, and Saving Games

The File Menu contains the usual items:

• File|New creates a blank form to create a game.

• File|Open opens the standard Windows File Open dialog box. It opens only files with the .gpx extension.

• File|Save and File|Save As open the standard Windows Save dialog box.

• File|Print opens the standard Windows Print dialog box. It will print the currently active window in GamePlan.

• File|List opens a text window that lists all the objects of the game.

• File|Name allows to give a name to the game that is displayed in the upper-left corner of its window. It doesn't affect the file name.

• File|Exit closes GamePlan.

2.5 Game Objects in GamePlan

As we saw in Chapter 1, classical game theory distinguishes the extensive form from the strategic or normal form. The extensive form is the preferred vehicle in *GamePlan*, especially because of the various "perfect" solution concepts. But *GamePlan* also supports the normal form which is the easiest to grasp and the one that has been most widely advertised. In general, the extensive form contains more details, and is therefore better than the normal form. Also, several extensive forms can usually be associated to a same normal form, while the converse is not true (at least up to a reordering of strategies).

GamePlan also allows structures far more general than the game trees usually considered in standard game theory. In a game tree, there is an initial node from which moves branch out until the final nodes are reached, and payoffs are associated to the final nodes only. *GamePlan* also allows graphs, and payoffs can be associated to final as well as non-final moves. This wider framework is particularly well suited to the representation of many kinds of dynamic games including those that may not allow any final moves.

As we saw, there are five basic objects that make up the extensive form of a game: players, nodes, moves, information sets and payoffs. But the normal form can be described by players, (pure) strategies, the "cells" defined by the choice of one strategy per player, and the associated payoffs. The best way to understand the *GamePlan* framework is to open game files (with an extension .gpx)[1] from the example library. I will refer to Figure 1.1 (Simplest.gpx) for a typical extensive form game and to Figure 1.4 (SimpleNorm.gpx) for a typical normal form game, from Chapter 1.

2.5.1 Players

Theoretically, one may define as many players as one wants in *GamePlan*. In practice, one will run out of colors to define them beyond a total of 14. Chance is a pre-defined player with the reserved dark-gray color. To add a player one simply select the Add|Player menu item or click on the corresponding Player button in the button bar. This opens the Player dialog box shown in Figure 2.2. One must give the player a name and choose its color.

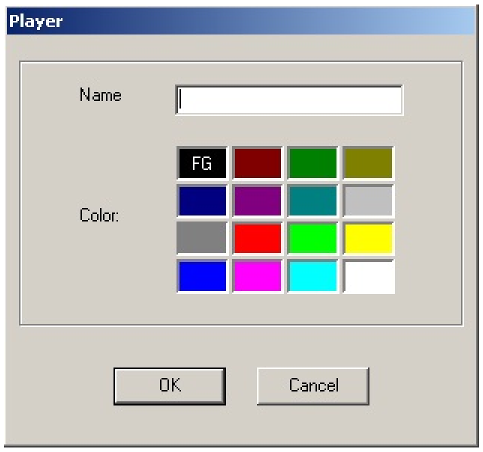


Figure 2.2: The Player Dialog Box

When clicking Ok, a new player icon in the chosen color with the given name appears on the desktop, with the mouse cursor turning to a cross. One can move the player icon around with the mouse and left-click it into the desired position. That position can be changed as follows: move the mouse cursor back to the player icon until it changes to a hand. A right-click opens a floating menu that allows deleting, editing or re-drawing the player.

[1]*GamePlan* files are in fact "formated text files". This is especially convenient for multiplatform use. It is also helpful to inspect them in a text app such as Notebook in order to spot some possible errors.

2.5.2 Nodes and Information Sets

Nodes are the most basic states of an extensive form game and are represented by small disks colored according to which players owns them. To create a node select the Add|Node menu item or click the corresponding button in the button bar. This opens the Node dialog box shown in Figure 2.3.

After giving it a name one must either select the player who will own it in the Owner list box or leave it as a chance node as shown. The owner is the player who will make decisions at that node. When clicking Ok, a new node icon in the color of its owner with the given name appears on the desktop, with the mouse cursor turning to a cross. One can move the node icon around with the mouse and left-click it into the desired position. That position can be changed in the same way as already described for players.

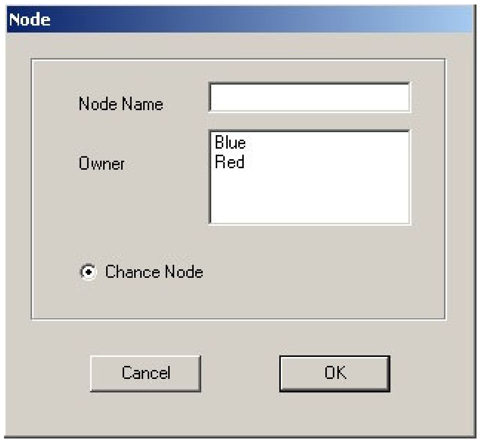


Figure 2.3: The Node Dialog Box

Information sets can only be created from existing nodes. To create, edit or delete an information set associated to a node, right click on the node icon and select the Info Set menu item. This opens the Information dialog box shown in Figure 2.4.

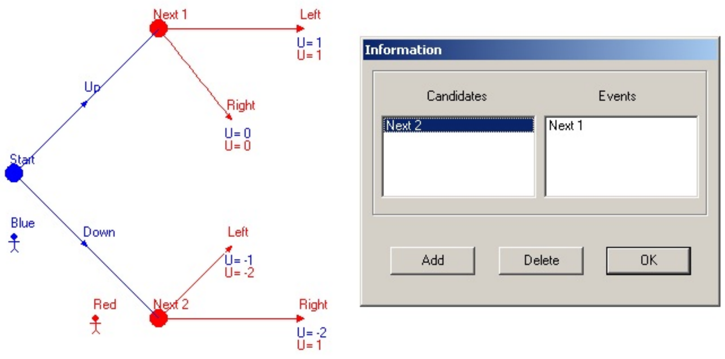


Figure 2.4: The Information Dialog Box

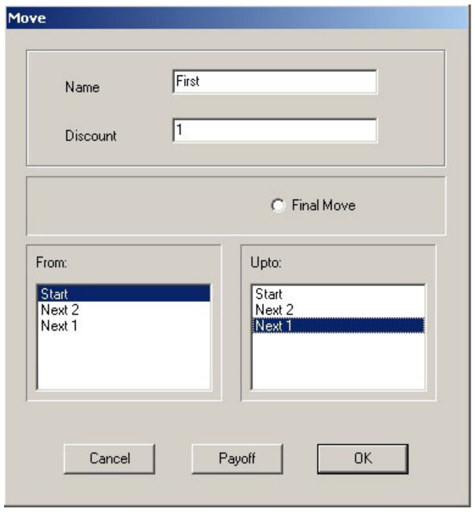
The node you used to access the dialog will always appear in the Events list box on the right. The Candidates list box shows the nodes available for adding to the current information set. To do so, click the Add button after selecting one. You may always delete it once it is in the Events list box by selecting it and clicking the Delete button. You cannot delete the node you used to access the Information dialog box. The nodes that appear in the Candidates box must belong to the same player as the one you accessed it through and they must not already belong to another information set. Once done with editing, click Ok to close the dialog. The created information set, if any, will show up as a thick dotted line between the nodes of the Events list in the same color as these nodes.

2.5.3 Moves

A move is a decision available at a node to a player who owns it. You can only create a move from an existing node. To create a move, select the Add|Move menu item or click the Move button in the button bar. This opens the Move dialog box shown in Figure 2.5. You must give a name to that move and select the node where it is available in the From list box. The move may be final (if the Final Move radio button is selected) or lead to another distinct node selected from the Upto list box. The default discount factor is d=1 which is the standard case except for the graph form where the moves that can be repeated should be systematically given a discount factor d, strictly between 0 and 1.

It is possible to define payoffs associated to the move, whether it is final or not, by clicking the payoff button. This will access the Payoff dialog box. But this can also be postponed and payoffs assigned later by right-clicking on the arrow icon of the concerned move.

When clicking the Ok button, the move will appear as a line beginning at its From node and extending either to its Upto node or to a point to the right of the From node when the Final Move radio button was selected. The mouse cursor will turn to a cross centered on an arrow that can re-positioned as needed by left-clicking where desired. The move will be in the same color as its From node. The move can be edited or redrawn by right-clicking on its arrow icon.


Move

Name: First

Discount: 1

Final Move

From:
Start (selected)
Next 2
Next 1

Upto:
Start
Next 2
Next 1 (selected)

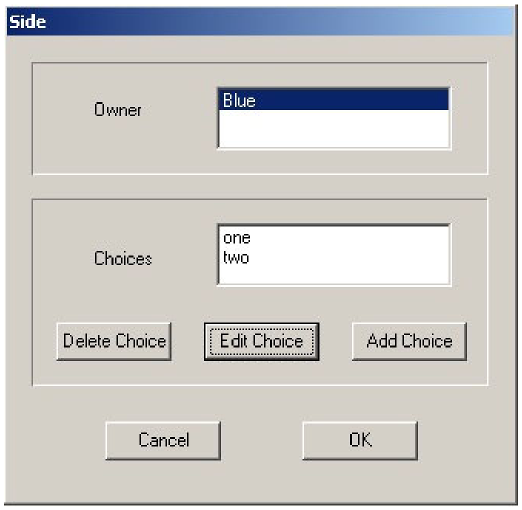
Figure 2.5: The Move Dialog Box

2.5.4 The Normal Form

A normal form game in *GamePlan* is merely a specialized template that can be a stand-alone game or that can become part of an extensive form or a graph form game. There are many advantages to this design choice: one can easily model stochastic games or discounted repeated games with finite memory and solve them in *GamePlan*.

To define a normal form game one must first define the players as in the extensive form. Then one selects the Add|Table menu item or clicks the corresponding button in the button bar. This opens a simple dialog box that asks only for a name for that normal form (the default is Table). After clicking Ok, a round-edged gray rectangle appears which can be positioned like any other game object. This is merely a container for the normal form. To fill it, one needs to first define the Sides involved. They are players taken from the list of existing ones. Not all defined players need to be involved in a normal form if it is only going to be part of a larger game structure. A Side is a player who needs to be given Choices (i.e., Strategies.)

To define a side one may select the Add|Side menu item or move the mouse cursor to the top-left corner of the table until the hand-cursor appears, right click, and select Add Side in the floating menu. This opens the Side dialog box shown in Figure 2.6.


Side

Owner: Blue

Choices: one, two

Figure 2.6: The Side Dialog Box

To define a "side" in the normal form you must select the player who plays it in the Owner list box. Then, you need to give that side Choices. You may add, edit or delete choices at will within that dialog box. When done you can click Ok. The inputs in Figure 2.8 result in the partial normal form shown in Figure 2.7 on the GamePlan desktop:

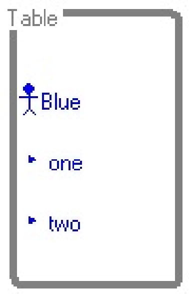


Figure 2.7: One Side in a Normal Form

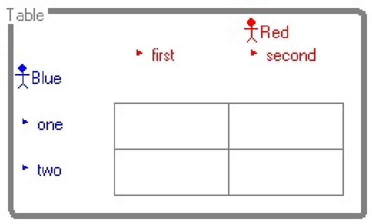
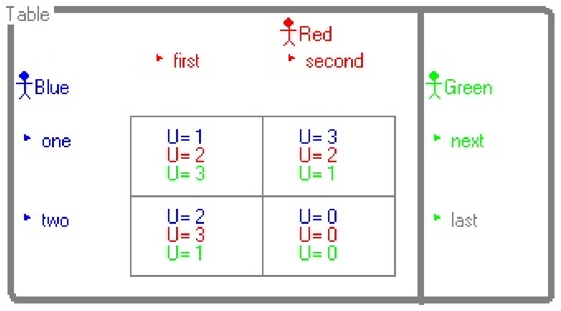


Figure 2.8: A Two-Sided Normal Form

Repeating the process for a second side (for player Red) yields figure 2.8. There now appear four Cells corresponding to each pair of choices for the two sides. These will need to be given payoffs. One can define normal form games with as many sides as wanted in GamePlan, although there are practical limitations of colors and display. Adding a third side Green to Figure 2.8 yields Figure 2.9. Payoffs have also been defined in all four cells shown for all three players.



	first	second

one	U= 1
	U= 2
	U= 3
		U= 3
		U= 2
		U= 1

two	U= 2
	U= 3
	U= 1
		U= 0
		U= 0
		U= 0

Figure 2.9: A Three-Sided Normal Form

Note that the choice (strategy) "next" is highlighted in green while the choice "last" is grayed for Green in the last column of the table. Right-clicking on the strategy icon for "last" (the arrow) will open a floating menu allowing you to select "Show Cells." This will replace the cells shown by those corresponding to the strategy "last" for Green which will become highlighted in green while "next" will become grayed. The addition of further sides will follow the same pattern in additional columns to the right of the table with the same rules.

In order to define payoffs in each cell of a table one needs to open the Payoff dialog box. In order to do so, you must right-click on the desired cell and select Payoff. The next section describes the definition of payoffs for both extensive and normal forms.

2.5.5 Payoffs

At the end of each final move in the extensive form and in each cell in the normal form, there must be payoffs defined for all players of the game. Payoofs can also be assigned to non-final moves, an option that is critical in defining dynamic games. The Payoff dialog box can be accessed from either a move or a cell by right clicking on the move arrow icon or inside the cell and selecting the Payoff item in the floating menu that opens. That dialog box is pictured in Figure 2.10.

Here, the Blue player has been selected and given payoff U=0 in the corresponding move or cell. Payoffs can be edited or deleted entirely if need be. Once defined, the payoffs immediately appear next to the move arrow or inside the cell they are associated to in the color of the player they belong to.

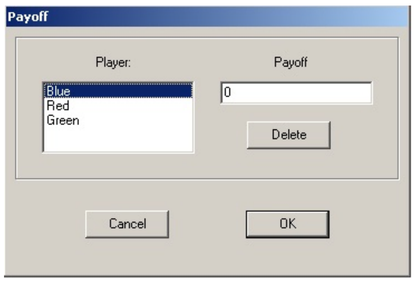


Figure 2.10: The Payoff Dialog Box

2.5 Editing Games

All the dialog boxes accessed to create game object can be used to edit them. One may either use the Edit or Delete main menu or, more practically, right click on the icon of the object to be edited or deleted in order to access its floating menu. Deleting must respect the natural hierarchy of game objects: a player that owns nodes cannot be deleted: either the nodes must be deleted first or their ownership must be transferred to another player. Nodes cannot be deleted if they own moves or belong to an information set. Such attachments must be removed before deleting the node.

Perhaps the most attractive feature of *GamePlan* is that it allows the design and solving of complex game structures such as incomplete information games, repeated games and stochastic games. These often involve duplicating a basic structure while creating new links between the duplicated structures. There is a powerful tool for doing just that in very little time: one simply selects the structure to be duplicated and copies it elsewhere. In order to do so, left-click on a corner of the area to be duplicated and, while holding the left-button down, move to the other extreme corner of that area. There appears a rectangle with dotted edges that can be adjusted to the area to be copied. When the left-button is released, the mouse cursor turns to a hand-cursor and jumps to one corner of the rectangle. Right-clicking opens a floating menu that allows deleting, copying or redrawing that entire area. Equivalently, one may use the delete or copy buttons in the button bar. When copying, the copied area will appear with the cross cursor to be positioned at will (by a left-click in the desired position.)

2.6 Display Options

The Display menu and buttons allows some variations in the way a game and its solutions are displayed.

• Display|Export is a powerful graphics utility that copies the game display into a .bmp file that is automatically opened in Windows Paint. It can be manipulated, edited and saved to various graphics formats for use in other documents.

• Display|Grid displays a grid that helps the precise positioning of game objects to produce clean and elegant game displays.

• Display|Values opens a submenu that helps viewing various aspects of a game solution.

• The tile buttons allow the standard tiling of windows, when several games are open at once or when several solution windows are open.

• The Zoom In and Out button allows zooming in and out for more or less precise viewing.

2.7 Solving a Game

The main purpose of GamePlan is to allow the solving of vast classes of games without even telling the program what method to use. It offers only two solution methods (Pure and Mixed) and only one solution concept (a generic Perfect Equilibrium.) This automatically reduces to a subgame perfect equilibrium (SPE) or a perfect Bayesian equilibrium (PBE) in extensive form games, to a Markov perfect equilibrium (MPE) in graph form games and to the standard Nash equilibrium in normal form games.

A game cannot be solved until it has been saved. In order to solve it one must select the Solution|Solve menu item or click the corresponding button in the button bar. Before solving a game, *GamePlan* automatically audits it. The Solution|Audit menu item also allows the user to monitor the solvability of a game while editing. If any defect is found, *GamePlan* displays an Audit window listing the possible flaws. If it finds no flaw, it begins solving and displays the Solve dialog box. A typical example is shown in Figure 2.11.

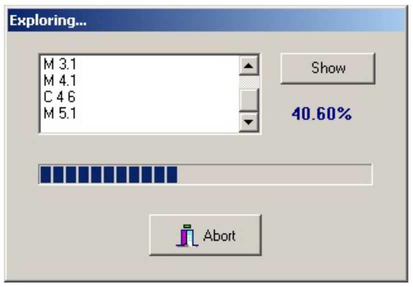


Figure 2.11: The Solve Dialog Box

The title of the box changes as the solving proceeds:

• "Profiling" is a preliminary step of checking pure strategy profiles for equilibria. In this step, *GamePlan* also evaluates pure profiles and stores them for further use. In games with relatively small sizes, this step will be so brief that the title will not even appear.

• "Eliminating" evaluates which profiles will be relevant to the elimination of locally dominated strategies. Again, this title will be skipped if the step is brief enough.

• "Exploring" seeks all mixed strategy equilibria by a systematic search of the regions of the action space that are not locally dominated for one player or another.

• "Organizing" is a systematic classification of all solutions found according to a simple criterion: solutions with the same optimal strategies, although with distinct probabilities of use, are placed in a same set. Set #1, for instance, may contain:

- P 1.1: meaning "pure, set #1, solution #1 in set";
- M 1.2: meaning "mixed, set #1, solution #2 in set";
- C 1 2: meaning "corner of set #1 and set #2".

A corner solution is one that belongs to more than one set. This typically happens when a strategy is optimal and has probability zero in one solution while it is suboptimal (and therefore has probability zero) in another solution. Organizing is valuable to identify entire polyhedra or more complex non-linear solution sets.

• "Cleaning Up" is simply de-allocating computer memory space.

For a complex enough game the complete solving can take quite some time. The progress bar gives the user feedback on the current progress (here 40.60%.) The solving can be aborted and that will not jeopardize the solutions already found. As the solving proceeds, any existing solution can be displayed by selecting it in the list box and clicking the Show button. An arbitrary number of solutions can be displayed at once. Once the solving is complete, the Abort button will turn to a Close button that simply closes the dialog box but does not affect the solutions found. The dialog box can be reopened with the Solution|Show menu item or the corresponding button in the button bar. The Solution|Delete All menu item or corresponding button will dump all existing solutions. Open solution windows can be closed at will. This will not dump the corresponding solution. An individual solution can be dumped by selecting it in the Solution dialog box and clicking Delete.

2.8 Viewing Solutions

The solution window contains a lot of information: in the extensive form, it shows the game itself as well as the probabilities of optimal moves, the beliefs consistent with the optimal moves when appropriate, the expected payoffs for all players at all nodes and the expected payoffs of all moves for the deciding player. In addition, moves with zero probability are shown in dotted lines and nodes with zero beliefs are shown as disks empty of color. A typical example is shown in Figure 2.12.

Above each move one reads its optimal probability. For instance, $p = 0.5$ for submit at Node D2. One also reads its expected payoff $e = 0$ above the probability. This is calculated from the expected payoff of the node it leads to (if any) for the deciding player, discounted if appropriate, as well as any possible immediate payoff for that player of choosing that move. If its From node belongs to an information set, the expected payoff is weighted by the belief of the move for each of the nodes at which it is available. So, it very easy to verify that one is looking at a true solution: since the equilibrium must be perfect any move with positive probability must involve an optimal expected payoff. And indeed, submit at D2 is optimal since $e = 0$ is maximum (and so is resist at D2.) At C1 instead, challenge involves $e = 0.25$ that is obtained by weighting, according to the beliefs $b = 0.5$ at each of the two nodes C1 and C2, the expected payoffs for Blue $\mathbb{E} = 0$ at node D1 and $\mathbb{E} = 0.5$ at node D2 that the move challenge can lead to. The move challenge is indeed optimal since stay at C1 and C2 yields an expected payoff $e = 0$ according to a similar calculation. Of course, since that move stay is not optimal it has zero probability and is dotted. Node C5 has no disk coloring because the belief at that node is zero, which is consistent with the fact that it cannot be reached according to the solution. One says that the node is "off the equilibrium path."

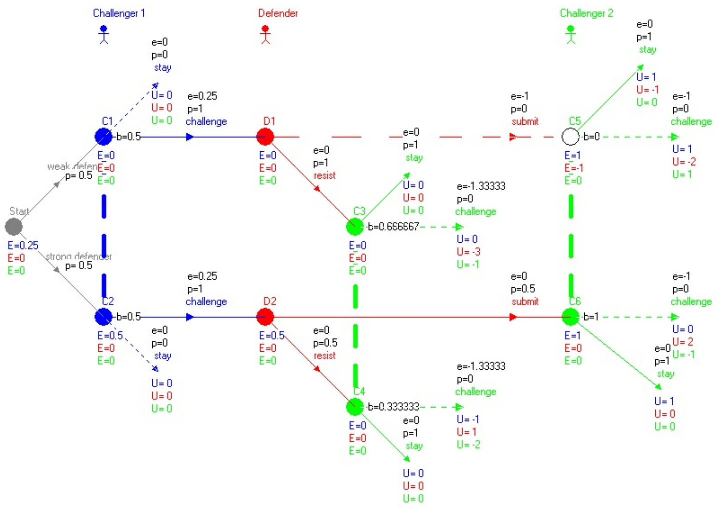
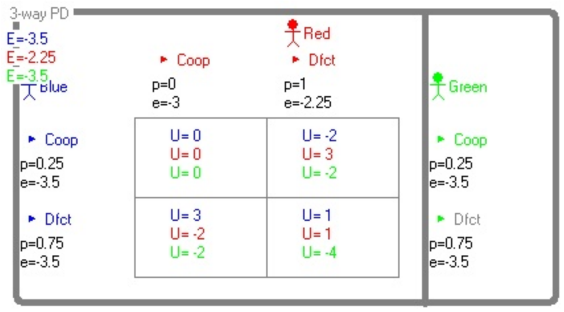


Figure 2.12: The Solution Window for an Extensive Form Game

The solution window for normal form games is shown similarly in Figure 2.13.



3-way PD
E=-3.5
E=-2.25
E=-3.5

	Red: Coop p=0 e=-3	Red: Dfct p=1 e=-2.25	Green

Blue: Coop p=0.25 e=-3.5	U= 0
	U= 0
	U= 0
		U= -2
		U= 3
		U= -2
			Coop p=0.25 e=-3.5

Blue: Dfct p=0.75 e=-3.5	U= 3
	U= -2
	U= -2
		U= 1
		U= 1
		U= -4
			Dfct p=0.75 e=-3.5

Figure 2.13: The Solution Window of a Normal Form Game

The strategy Dfct for Red has probability $p = 1$ and expected payoff $e = -2.25$, strictly better than the strategy Coop for Red. For Blue, the two strategies have same expected payoffs and positive probabilities. The expected payoffs for all three players of playing accordingly in this table are shown at the top left. They will be particularly useful in more advanced game structures where the table can be entered with a move from another part of the game.

Several solution windows can be viewed at once and tiled for better comparison. It is sometime useful to not show some features such as the expected payoffs. One needs to select the Display|Values menu item to adjust such features. One can also zoom in and out using the zoom button in the button bar. Moves can also be redrawn in the solution window, for better viewing, by right clicking on the arrow and selecting Redraw in the floating menu.

Solution windows can be closed individually. They can be reopened by selecting Solution|Show in the menu, or clicking the corresponding button in the button bar. A solved game can be saved with or without solution. The solution will then be stored in the game file and be reopened later. A solved game cannot be edited without first deleting all solutions. In order to edit a solved game one simply needs to copy it without solutions to another file (using Save As in the File menu) and proceed with editing in that other file. But all solutions can also be dumped at once by selecting Solution|Delete All in the main menu or clicking the corresponding button in the button bar.

2.9 Some Advanced Features

GamePlan offers great flexibility in designing and solving advanced game structures. For instance, Figure 2.14 shows a model of a one-shot Prisoner's Dilemma that is repeated with discount factor $d = 0.9$, and with memory of the past limited to three states: No Guilt, Row Guilty and Column Guilty. Solving in Pure mode will instantly produce the only two pure strategy Markov perfect equilibria: Defect (Dfct) all the time and the well known Contrite Tit for Tat.

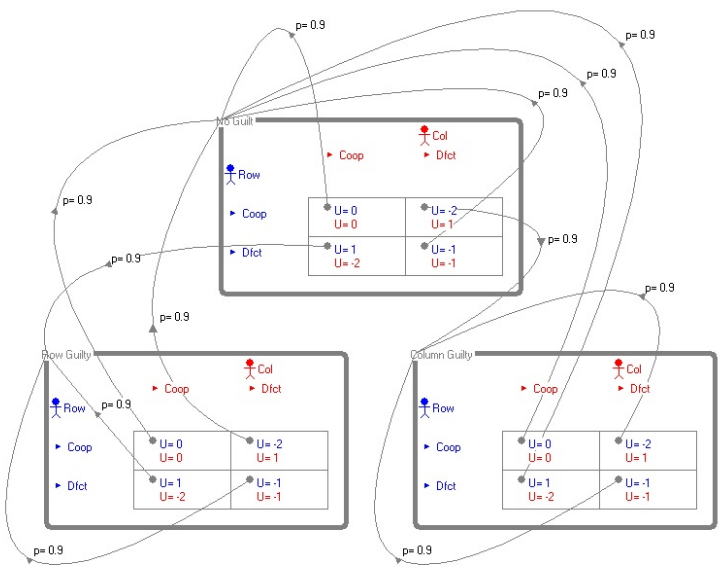


Figure 2.14: A Repeated Prisoner's Dilemma with Three Memory States

In order to design such a structure, one may simply construct the one-shot normal form game, then make two copies using the editing features described above and finally define the transitions from state to state by simply editing all cells from final to non-final. To do so, right click on any cell, define a non-trivial discount factor ($d = 0.9$ in Figure 2.16) and select the table that the cell should lead to when it is the outcome of one turn of play. For instance, when Row is guilty (according to previous play), we are in the bottom-left table. Blue should then cooperate (Coop) while Red can defect (Dfct) while not becoming guilty. So, both upper cells of that table lead to the upper No Guilt table. In the No Guilt table, a unilateral defection will lead to the corresponding guilt state while joint cooperation or joint defection will lead back to the No Guilt table. Once a the appropriate Upto has been selected, the move can be drawn in the same way as any other extensive form move, although they are gray-colored since they do not belong to a single player.

Extensive, normal and graph features can cohabitate in the *GamePlan* framework, thus allowing the design of sophisticated game structures. They may, of course, yield long solving times. The auditing function of *GamePlan* will provide feedback on what are valid or invalid features. But it cannot entirely rule out non-meaningful ones. It is assumed that the user has good knowledge of proper game theory.

2.10 Help

The *GameHelp* help file can be accessed with Help|Help in the menu or the question mark button. It only offers quick references to using *GamePlan*.