NetworGame tutorial

(Version 05/12/2011)

Game centrality: a novel measure identifying key mediators of cooperation in social dilemma games on real-world networks

Gabor I. Simko[#], Peter Csermely*

Department of Medical Chemistry, Semmelweis University, Tűzoltó str. 37-47, H-1094 Budapest, Hungary

*Corresponding author, E-mail: csermely@eok.sote.hu
[#]Present address: Vanderbilt University, Nashville, TN USA

In this tutorial we show how to use the NetworGame package to build and simulate games. As a "teaching model" we will simulate here hawk-dove games with different parameters on three different small-world networks using the best takes over strategy update rule. From previous data we expect that larger contested resource values of this game will result in higher defection rate (higher rate of hawk strategy), and that a random network will be more advantageous for cooperators than a regular lattice (due to better chances for forming resistant cooperator coalitions using the 'shortcuts' of the random network).

1. Using the simulation engine

1) First, using the supplied Graphical User Interface (GUI) we create a new project (file/new) called example.cng. A default project is created shown below:
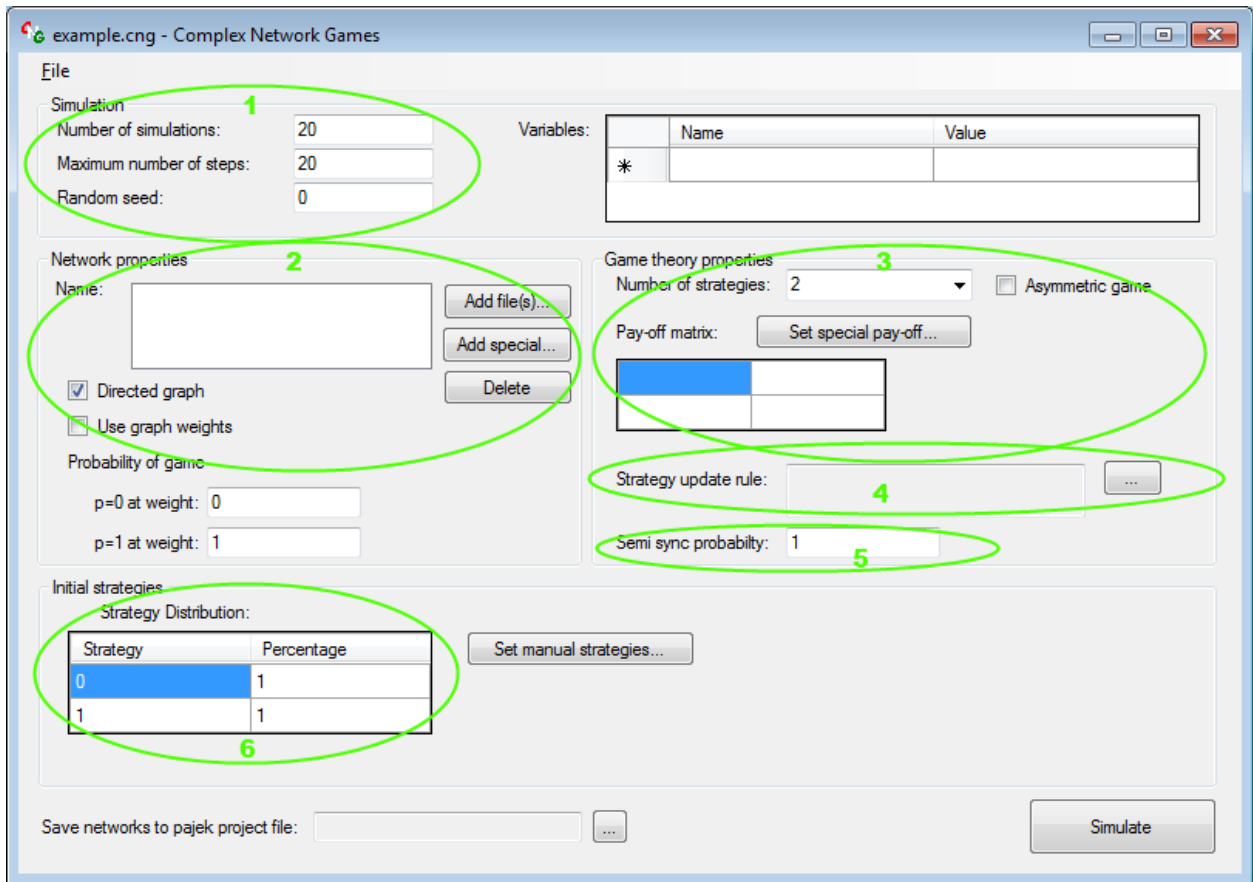
2) Next, we set up the number of simulations (100) and number of steps in each simulation (100; see mark 1 on the Figure). More simulations will result in better (more exact) results, but will require longer processing times. We may also set the default random seed value, which gives the seed value for the random number generator to allow the reproduction of randomized runs.

3) In the "network properties dialog" box (see mark 2 on the Figure) we can add (directed or undirected) network data using Pajek format, or we may generate model networks. In the tutorial we generate three Tomassini-type small world networks[1] with 100 nodes and 8 neighboring nodes each, having a rewiring probability = 0, 0.1 and 1, respectively. For doing this after pressing the "Add special…" button, we reach the "add special graph" dialog, where we can set up the parameters.

[1]Tomassini, M., Luthi, L. and Giacobini M. (2006) Hawks and doves on small-world networks. Phys. Rev. E 73: 016132.
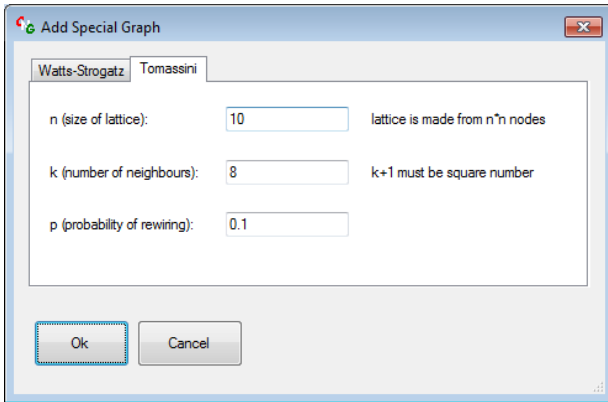
4) In the next step we will set up the payoff matrix (see mark 3 on the Figure on page 2). We will simulate a hawk-dove game with several values for the contested resource. To do so, we add a variable called V (representing the value of the resource), which runs through the values 0.0, 0.1, 0.2,…, 0.9, 1.0, and a C parameter (=1) representing the cost of an escalated fight. Using these parameters, we can fill out the payoff matrix as seen on the figure. Strategy 0 and 1 represent hawk and dove strategies, respectively. If two hawks fight, both of them gain (lose) (V-C)/2, a hawk gains V against a dove, and two doves share the value of the resource (V/2).
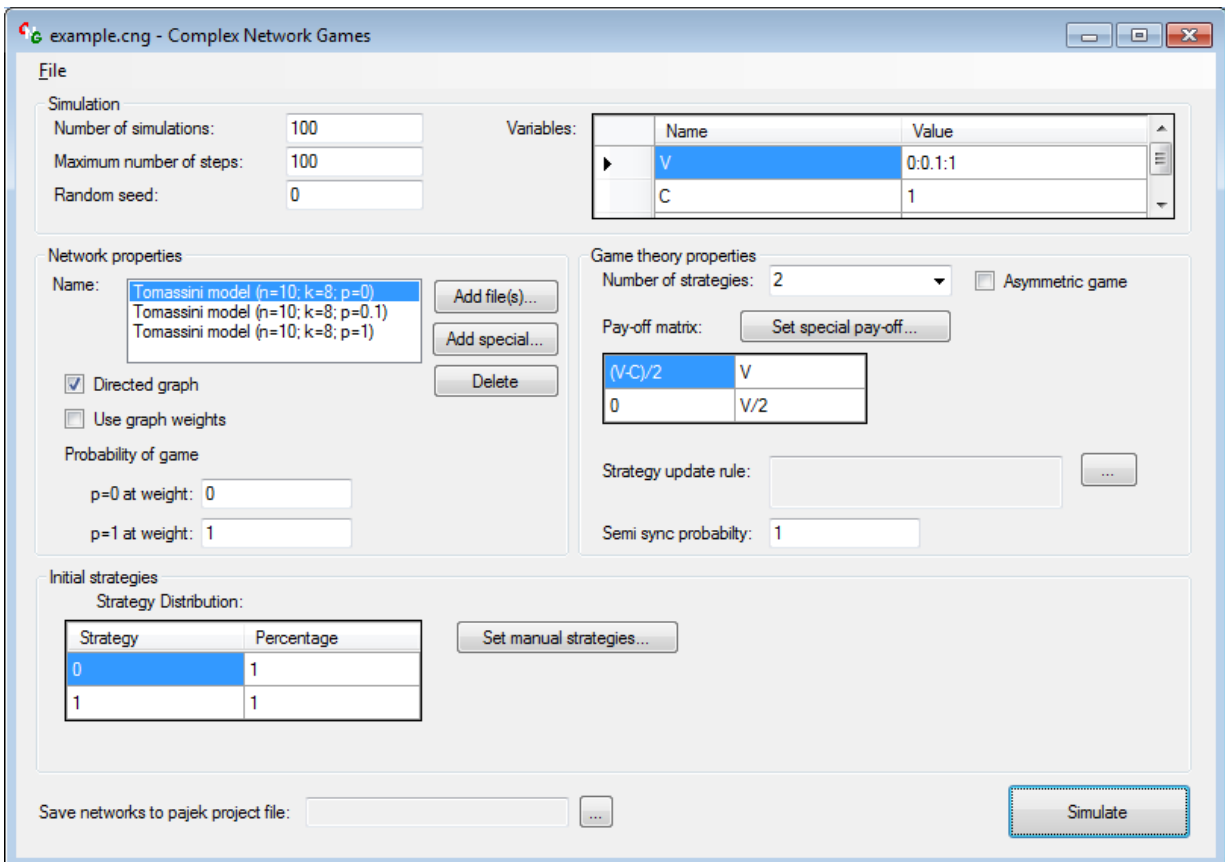
5) Next we have to specify the strategy update rule. After clicking the "…" button next to the strategy update rule label (see mark 4 on the Figure on page 2), we choose the BestTakesOver.dll.

6) We set the semi-synchronous probability to 0.01 (see mark 5 on the Figure on page 2), to make the simulations not synchronous. This is highly beneficial for a higher level of cooperation when using the best takes over strategy update rule, because it introduces some stochasticity in the simulations.[2]

7) We do not change the initial strategy distribution (see mark 6 on the Figure on page 2) describing a 50% to 50% distribution of hawk (defector) and dove (cooperator) strategies, respectively.

The final settings are on the following figure:
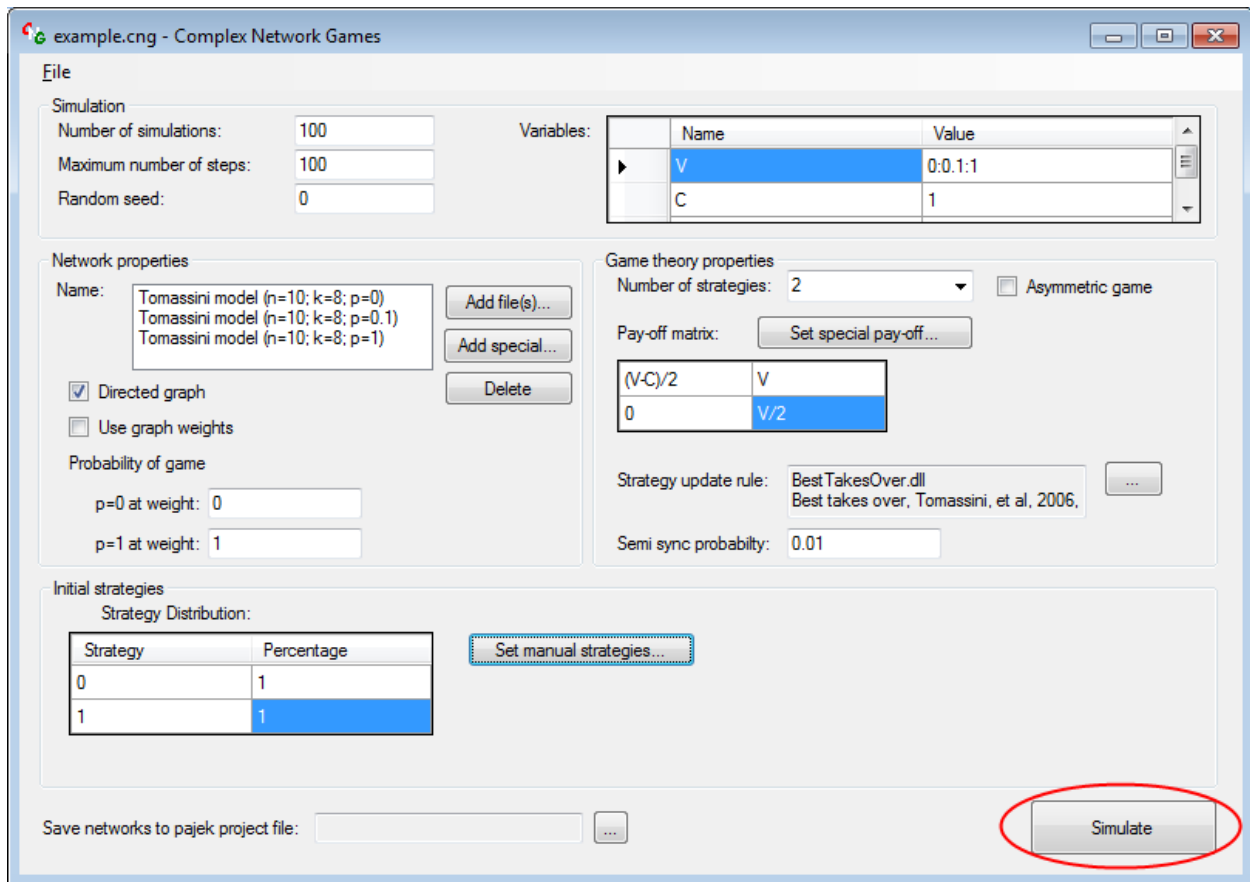
Next, we can run the simulation by pressing the "Simulate" button (see red circle on the Figure).

---

[2]Saif, M. A. and Gade, P. M. (2009) Prisoner's dilemma with semi-synchronous updates: evidence for a first order phase transition. J. Stat. Mech. Theor. Exp. 7: P07023; Wang, S., Szalay, M.S., Zhang, C. and Csermely, P. (2008) Learning and innovative elements of strategy update rules expand cooperative network topologies. PLoS ONE 3: e1917.

The results will be stored as an xml file in the same directory as the simulation settings file is. The xml files may be processed with any programs, but the NetworGame program package also provides a data mining tool, which can be used to generate csv files from the result.

2. Using the data mining tool

In the tutorial we want to compare the three types of small world networks (see Step 3) and the different contested resource (V) values (see Step 4) to each other. The rows of the csv table will represent the V values, and the columns of the table will be the three networks. Currently the NetworGame data mining tool is not supported through the GUI yet, so we need to copy the simulation's result file (example.cngr) to the directory, where cng.exe is found. Then using the command prompt we execute the following command:

cng.exe example.cngr 1 V 10 networks

This tells the data mining tool that we want to work with example.cngr, we want to extract the proportion of strategy 1 (the cooperating dove strategy), the rows are the different V values, each cell is calculated by averaging the result of the last 10% of steps (currently 10 steps), and the columns are the different networks.

The resultant table will have 3 x 2 columns of data. The first two columns give the average dove proportion during the last 10% of steps (as specified in the command above, currently 10 steps) in each network. The third and fourth columns give the standard deviation of the mean of the last 10% of steps, i.e. 10 steps for the 100 runs total. The sixth and seventh columns give the average of the maximal difference (for a more detailed explanation of this value: see below) during the last 10% of steps (which is again 10 steps, since we had 100 steps total).

The meaning of the average and deviation values is easy to understand, but the third value, the "average of the maximal difference" needs some more explanation. Because the standard deviation of columns 3 and 4 is calculated using the mean of the last 10% steps (i.e. 10 steps of the currently 100 simulations total), it does not give any information about the time-course of the individual runs; rather it describes the deviation of the runs from each other. On the other hand, the "average of the maximal difference" value is calculated by subtracting the minimum value of dove proportion from the maximum value of dove proportion in the last 10% of the steps for each run, and by averaging these differences for the currently 100 simulations total. This metric is more useful, if we are interested to know, whether the simulation converged to a "steady state" (quasi-equilibrium) in the last 10% of the steps.

3. Interpreting the results

The tutorial produces the following output (example.csv, columns renamed and keeping only the first 3 columns – the mean cooperator proportions):

V	Average:	(n=10, k=8, p=0)	(n=10, k=8, p=0.1)	(n=10, k=8, p=1)
0		0.815	0.816	0.812
0.1		0.781	0.765	0.778
0.2		0.656	0.659	0.71
0.3		0.528	0.551	0.611
0.4		0.433	0.458	0.526
0.5		0.345	0.359	0.416
0.6		0.282	0.288	0.332
0.7		0.243	0.249	0.273
0.8		0.229	0.233	0.248
0.9		0.226	0.228	0.236
1		0.226	0.227	0.23

The first column shows the value of the contested resource (V). The 3rd, 4th and 5th columns show the final mean cooperation level in the three networks. The results clearly show that the increasing value of the competed resource also decreases the proportion of players playing the cooperating dove strategy. The comparison of the three networks shows that the small-world network (4th column, p=0.1) is more advantageous for the cooperators (players with dove strategy) than the lattice (3rd column, p=0), and that the random network (5th column, p=1) is even more advantageous for cooperators than the small world network (4th column, p=0.1).