Log-Linear Learning Over Large Iterations in Sudoku Puzzles

Vince Coghlan

May 6, 2014

Sudoku is a puzzle game played on a 9×9 board. The goal is to have a unique occurence of each didget in each row, column, and 3×3 square. This can be modeled as a game with the following parameters:

- Players $N = \{1, 2, \ldots, n\}$ Where n is the number of empty spaces
- Actions $a_i \in \mathcal{A}$ where an action is a number between 1 and 9 and the actions set $\mathcal{A} = a_1 \times a_2 \times \cdots \times a_n$
- Utility Functions $u_i : \mathcal{A} \to \mathbb{R}$ Where the utility will be the number of repetitions of a player's current action in a row, column, or square.

$$u_i(a) = u_{\text{row}_i}(a) + u_{\text{column}_i}(a) + u_{\text{square}_i}(a) \tag{1}$$

where

$$u_{\text{row}_i} = \sum_{j \in i_{row}} (a_j = a_i)$$

- Welfare Function $W(a) : \mathcal{A} \to \mathbb{R}$ where the global welfare of an action represents the global desireability of that action. Let a be the joint action (a_i, a_{-i}) where $a_{-i} = (a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_n)$

We are given the ability to design the welfare function however we would like. Suppose that our objective is captured by a function $\phi : \mathcal{A} \to \mathbb{R}$ where

$$u_i(a_i, a_{-i}) - u_i(a_i^*, a_{-i}) = \phi(a_i, a_{-i}) - \phi(a_i^*, a_{-i}) \tag{2}$$

A ϕ satisfying (2) will be known as a potential function [8]. It can be shown that the game of Sudoku has a potential function when we define the utility as in (1). We begin by assuming that the potential function will take the form

$$\phi(a) = \phi_{\text{row}}(a) + \phi_{\text{column}}(a) + \phi_{\text{square}}(a)$$

We can then show that

$$\phi_{\text{row}}(a) - \phi_{\text{row}}(a^*) = u_{\text{row}_i}(a) - u_{\text{row}_i}(a^*)$$

And so on. When a player changes his number, he increases the amount of repetitions of that number by 1. By doing this, however, he has unintentionally increased the amount of repetitions of any other player preforming the same action, and decreased the amount of repetitions of his old action. One would expect that the potential function is the sum of utilities of each player

$$\phi_{\text{row}}(a) = \sum_{i \in N} u_{\text{row}_i}(a)$$

$$\phi_{\text{row}}(a) - \phi_{\text{row}}(a^*) = \sum_{i \in N} u_{\text{row}_i}(a) - \sum_{i \in N} u_{\text{row}_i}(a^*)$$

$$= \sum_{i \in N} \sum_{j \in i_{row}} (a_j = a_i) - \sum_{i \in N} \sum_{j \in i_{row}} (a_j = a_i^*)$$

$$= \sum_{j \in i_{row}} (a_j = a_i) - \sum_{j \in i_{row}} (a_j = a_i) + \sum_{i \in N \setminus \{i\}} \sum_{j \in i_{row}} (a_j = a_i) - \sum_{i \in N \setminus \{i\}} \sum_{j \in i_{row}} (a_j = a_i^*)$$

Now we will use the fact that an increase in n in the repetitions of a_i corresponds to an increase of 1 of n other players playing a_i. This means that:

$$= \sum_{j \in i_{row}} (a_j = a_i) - \sum_{j \in i_{row}} (a_j = a_i) + \sum_{j \in i_{row}} (a_j = a_i) - \sum_{j \in i_{row}} (a_j = a_i)$$

$$= 2(\sum_{j \in i_{row}} (a_j = a_i) - \sum_{j \in i_{row}} (a_j = a_i))$$

This tells us that we have twice the potential function, meaning our potential function for the row is

$$\phi_{\text{row}}(a) = \frac{1}{2} \sum_{i \in N} u_{\text{row}_i}(a)$$

This can be easily extended to squares and columns to find:

$$\phi(a) = \frac{1}{2} \sum_{i \in N} u_i(a) \tag{3}$$

Since sudoku has a potential function, the minimization of the potential is a pure equilibrium [5]. Since we cannot have a negative potential by our definition of the utility, and a potential of 0 means that there are no repetitions in any row, column, or square, a pure Nash Equilibrium exists, and must be the solution of the puzzle.

We have half of our problem laid out, the next part is to define a learning algorithm that will converge to this equilibrium. Current strategies being studied currently include fictitious play [7], log-linear learning [2], and the cournot best reply process [3] among others. Many of these learning algorithms converge to a Nash equilibrium [6]. It has been shown [9] that

Log-linear learning provides us the fact that, in any exact potential game, log-linear learning will induce a Markov chain with a unique stationary distrobution $\pi \in \Delta(\mathcal{A})$ of the form

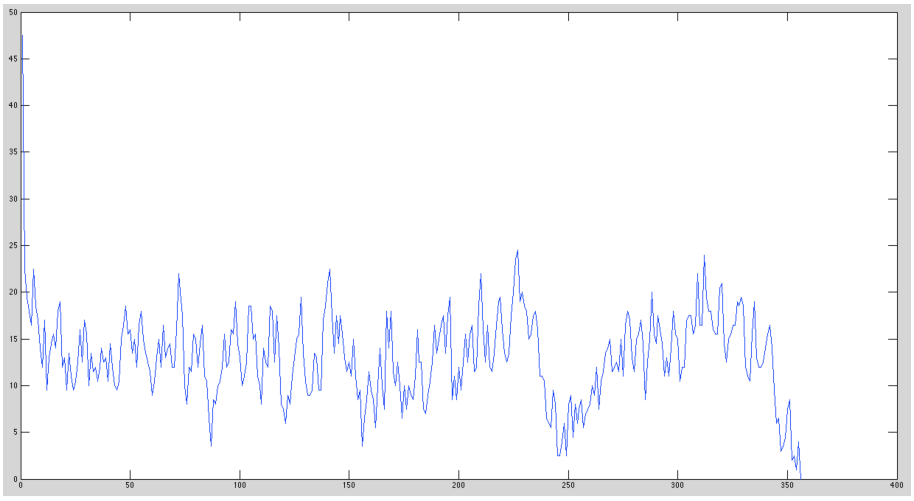
$$\pi_a = \frac{e^{\frac{1}{T}\phi(a)}}{\sum_{\tilde{a}\in\mathcal{A}} e^{\frac{1}{T}\phi(\tilde{a})}} \tag{4}$$

Since ϕ is zero at the solutions, we know that the distrobutions of the system as $T \to 0$ will be distributed on solutions to the sudoku puzzle. We can see this behavior for some simple sudoku puzzles, for example, this puzzle was solved in 28836 iterations, with a temperature of $T = 0.5$.

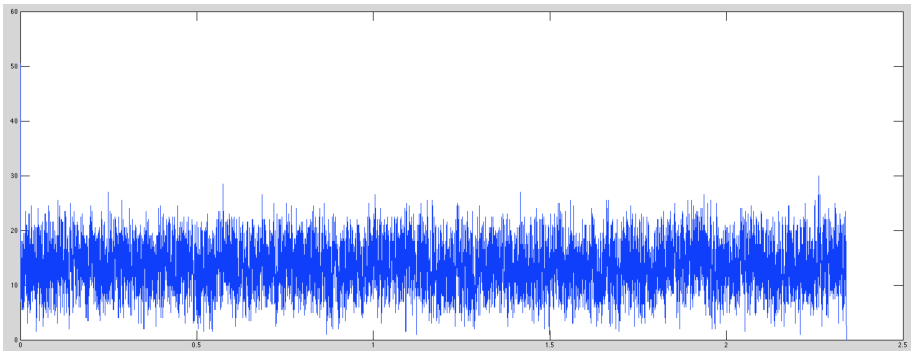
7		3		1	4	6		
			7		9		8	
		8	3					1
5		9					4	
				2		9	1	6
2			9		8			7
	8	7				4	5	
9	1			3		2		
3								9

7	9	3	8	1	4	6	2	5
1	2	6	7	5	9	3	8	4
4	5	8	3	6	2	7	9	1
5	3	9	1	7	6	8	4	2
8	7	4	5	2	3	9	1	6
2	6	1	9	4	8	5	3	7
6	8	7	2	9	1	4	5	3
9	1	5	4	3	7	2	6	8
3	4	2	6	8	5	1	7	9

The potential can be seen



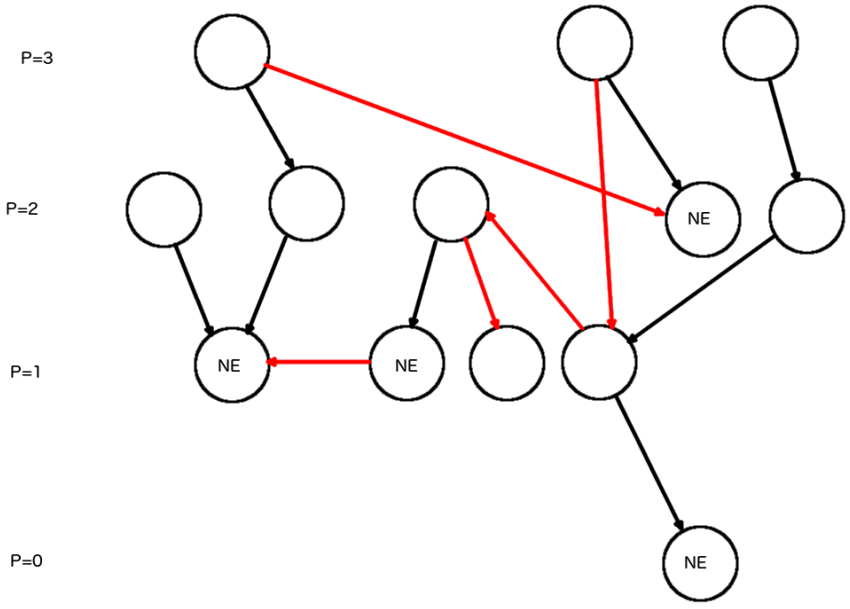
You can see that once the potential is 0, we are done, and the puzzle is solved. We can see a different story if we run from a different initial condition.



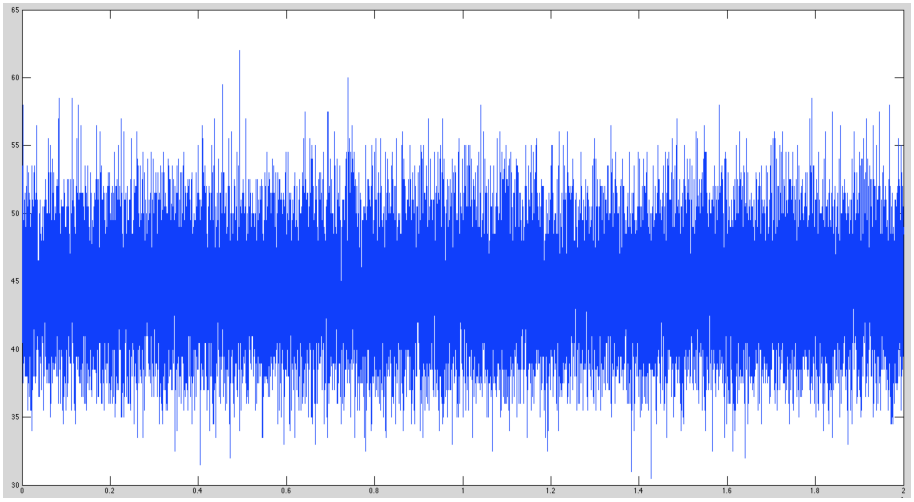
When we start from a different initial condition, we get stuck in a puzzle configuration that is not bad, but not a solution to the puzzle. There is sufficiently high resistance between this state and the actual solution state so that moving to the solution requires many iterations. It took 500000+ iterations to solve, since we spend most of out time in the following puzzle configuration:

7	9	3	2	1	4	6	9	5
6	4	1	7	5	9	3	8	2
2	5	8	3	8	6	4	7	1
5	3	9	1	6	7	8	4	2
8	7	4	5	2	3	9	1	6
2	6	1	9	4	8	5	3	7
4	8	7	6	9	2	4	5	3
9	1	5	4	3	7	2	6	8
3	2	6	8	7	5	1	2	9

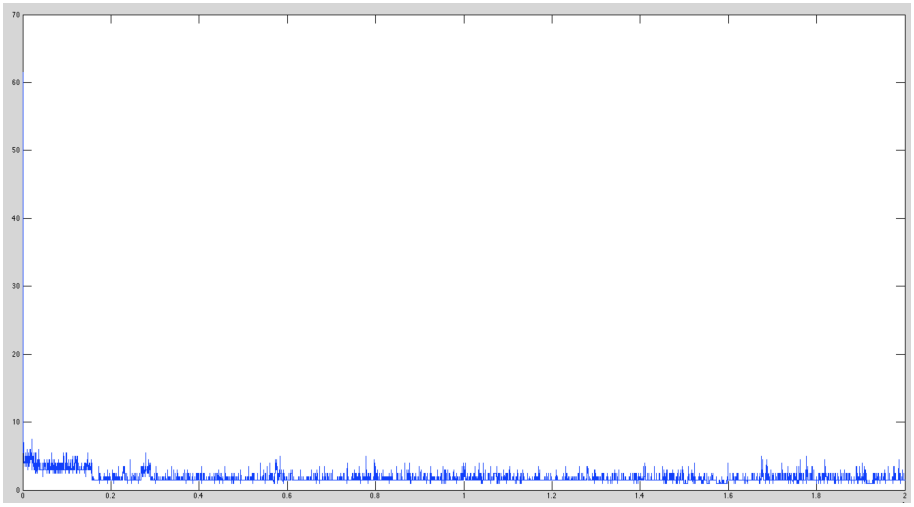
Repeated numbers are highlighted in red. You can see that this solution is close, with no more than one repitition for any numbers. If any of these repeated players chose to change their configuration, than they would either maintain their cost, or increase it. This fact means that it is sufficiently difficult to leave this configuration, and requires many levels of ϵ to overcome. It is clear from this that any learning method will not do. We need a learning method that has some sort of kickout probability so that we can leave an innefficient Nash Equilibrium. In the above case we are in equilibrium and staying there wont solve the puzzle. A learning algorithm like fictitious play or cournot adjustment process would remain at this equilibrium. This is where the value of log linear learning lies, the potential graph shows that we are constantly in a state of finding a low potential solution, then leaving it to find another. We can visualize what this would look like in the following figure



Where the circles are states of the puzzle, each level of the picture is a level of potential. There is only one solution, meaning that there is only one configuration with potential of zero. The black arrows are paths of probability $1 - \epsilon$ and the red arrows are paths of ϵ. In reality this graph extends to include all possible configurations, a number as large as 10^{81}. The resistance between the left nash equilibrium and the solution of the puzzle is 6 (note that any black arrow will have a red arrow going in the opposite direction). The resistance from the far right equilibrium would be 3. This is what makes finding a solution so difficult, and the reason why our learning algorithm needs a sort of "kick-out" probability that allows us to leave a state. If we increase the temperature to a higher value, say 2, we will more easily leave these nodes, but the potential shows that we have too much chaos in the system



This is not right, and wont find a solution in sufficient time. You can see the potential arrive at values as low as two, maybe these were on the correct path, but the large temperature forced these to go away from this equilibrium. A small temperature is no good either



We get stuck at another equilibrium. There are, in fact, a multitude of equilibria, only one of which has a potential of zero, the solution. The probability epsilon becomes too small to solve this in a reasonable amount of time. We can see that these wont be solved in a reasonable time, but we know that in any exact potential game, the only stochastically stable state is the potential minimizer [6]. Games with multiple solutions provide more insight into the problem at hand. take for example [4]

9		6		7		4		3
			4			2		
	7			2	3		1	
5						1		
	4		2		8		6	
		3						5
	3		7				5	
		7			5			
4		5		1		7		8

9	2	6	5	7	1	4	8	3
3	5	1	4	8	6	2	7	9
8	7	4	9	2	3	5	1	6
5	8	2	3	6	7	1	9	4
1	4	9	2	5	8	3	6	7
7	6	3	1			8	2	5
2	3	8	7			6	5	1
6	1	7	8	3	5	9	4	2
4	9	5	6	1	2	7	3	8

The empty squares could be filled in with $\{9, 4, 4, 9\}$ or $\{4, 9, 9, 4\}$. It can be show that any square of the form $\{a, b, b, a\}$ in a puzzle can be reversed and another solution found [4]. For any of these cases, it will take a probability of ϵ to move to any other configuration, for example $\{a, b, a, b\}$. Once this change occurs, the probability of moving to one solution or the other becomes a 50/50 chance, making the probability of changing equilibria to be

$\epsilon/2$. For sufficiently large epsilon, it follows that the probability of landing in any particular solution in this situation would be 0.5.

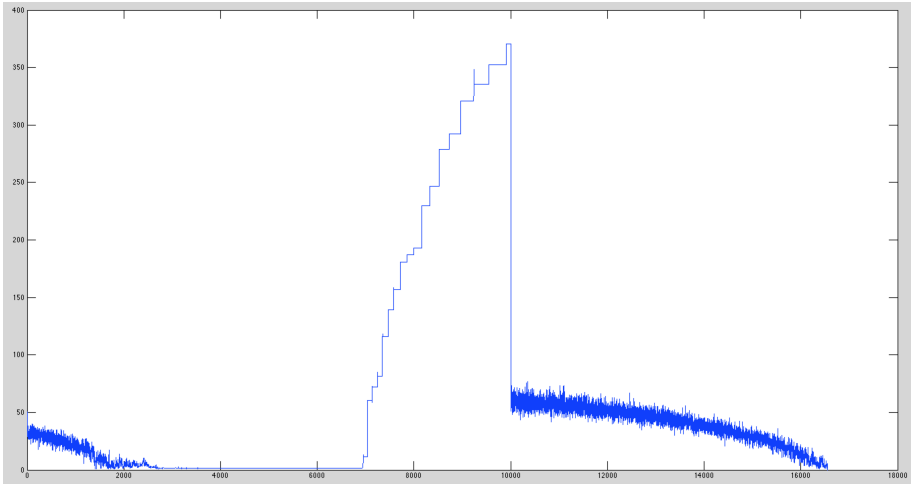
What we need is a different algorithm that will prune the tree of posible states and possible equilibria. This algorithm will take the current one and preform three simple modifications:

- The player making his move we be chosen from all players at random every iteration
- The player's possible actions will not include actions that are from initial set squares at the beginning of the game. i.e., all game states that are, by definition, not solutions will be eliminated from the possible future action profile of any player.
- The value of the temperature will vary over time, and will reset after sufficient iterations

We can show that each of these maintains a potential game. We will start from the bottom. As long as we keep our value of temperature "sufficiently large", then we will reach the same conclusion that we do for regular log-linear learning, that is that the set of potential maximizers are the only stochastically stable states. This will force us out of a bad equilibrium if we are stuck there too long. Next, if the player does not get to play any action he want, i.e., any action that started out on the board to begin with, than the tree is trimmed of solutions we know dont exist. Since the solution is not included in that set, but is the stationary distrobution, than the rules of a potential game still apply. If the player making the move has been chosed uniformly, than we can trust that we are still moving towards a stationary distrobution, since each player is still updating asynchronously [1]. It matters not who updates in any given iteration, as long as someone does, and only one person does. The solution to the first puzzle we looked at can be seen followed by the potential function after finding it.

7		3		1	4	6		
			7		9		8	
		8	3					1
5		9					4	
				2		9	1	6
2			9		8			7
	8	7				4	5	
9	1			3		2		
3								9

7	9	3	8	1	4	6	2	5
1	2	6	7	5	9	3	8	4
4	5	8	3	6	2	7	9	1
5	3	9	1	7	6	8	4	2
8	7	4	5	2	3	9	1	6
2	6	1	9	4	8	5	3	7
6	8	7	2	9	1	4	5	3
9	1	5	4	3	7	2	6	8
3	4	2	6	8	5	1	7	9



Note that this is a very different graph. In the previous graphs, we stay at around the same value, but never get close to 0, in this graph, we decrease steadily as the temperature drops. Once we reach a Nash equilibrium, our temperature goes back to a chaotic value, and we try again. This method finds a solution much faster and with much higher reliability. This is because we dont get stuck at a location with high resistance to the solution, we instead move to a new state of the game that has a good chance of converging.

Finding an equilibrium in a real life engineering setting is not always possible with log-linear learning. Sometimes we are forced to a nonoptimal equilibrium. Constantly resetting the temperature allows us to find a solution reliably fast. Many of the puzzles ran did not converge under regular log-linear learning in millions of iteration. We know that it has to at some point, the math has proven that, but it is far more efficient to change the temperature as a function of time.

References

[1] C. Alos-Ferrer and N. Netzer, The logit-response dynamics. *Ganes and Economic Behavior*, 68:413-427 2010.

[2] L. Blume, The statistical mechanics of strategic interaction. *Games and Economic Behavior*, pp. 387–424, 1993.

[3] A. Cournot, *Recherches sur les principes mathématiques de la théorie des richesses*, Hachette, 1838.

[4] A. Herzberg and M. Murty, Sudoku Squares and Chromatic Polynomials. *Notices of the AMS* June/July, 2007

[5] A. Neyman, Correlated Equilibrium and Potential Games. *International Journal of Game Theory*, pp. 223–227, 1997.

[6] J. Marden and J. Shamma, Revisiting Log-Linear Learning: Asynchrony, Completeness and Payoff-Based Implementation *Games and Economic Behavior*, Volume 75, Issue 4, pp. 788–808, 2012.

[7] D. Monderer and L. Shapley, Fictitious Play Property for Games with Identical Interests. *Journal of Economic Theory*, Volume 68, pp. 258–265, 1996

[8] D. Monderer and L. Shapley, Potential Games. *Games and Economic Behavior*, Article No. 0044 pp. 124–143, 1994.

[9] H.P. Young. The Evolution of Conventions. *Econometrica*, pp. 57–84, 1993.