Vol. 9 (2016) *Acta Physica Polonica B Proceedings Supplement* No 1

QUANTIFYING SIMULATION PARAMETERS' EFFECTS ON NAÏVE CREATURES LEARNING TO SAFELY CROSS A HIGHWAY USING REGRESSION TREES*

Leslie Ly, Anna T. Lawniczak, Fei Yu

Department of Mathematics and Statistics University of Guelph
Guelph, Ontario N1G 2W1, Canada

(Received March 9, 2016)

A model of simulated cognitive agents (naïve creatures) learning to safely cross a cellular automaton-based highway is described. These creatures are minimal, equipped with the ability to "perceive", "reason", "judge", and "respond" in order to learn from each other by evaluating if a creature in the past was successful in crossing the highway for their current situation. A large amount of data files are generated from this simulation model under different configurations of the simulation parameters' values (such as the traffic density and the nature of these creatures in terms of fear and desire). These simulation parameters heavily influence the learning outcomes examined through the collected simulation metrics. We study how these parameters influence these metrics using regression trees.

DOI:10.5506/APhysPolBSupp.9.77

1. Introduction

In autonomous swarm robotics, modeling and simulation play an important role. The individual robots may be identified as cognitive agents [1–3] and various ideas tested in virtual reality. The developed models are usually multifactorial and understanding their dynamics and performance is often a challenging task. For example, when one requires the robotic swarm (cognitive agents) to learn how to accomplish some tasks in unknown dynamically changing environments under the constraint of having minimal computational resources. In this case, one may explore the process of learning through observation and repetition, [4] and [5], utilizing simple decision-making algorithms. In this paper, we investigate such approach, *i.e.* we investigate the performance of a simple learning algorithm based on an "observational social learning" mechanism, [4] and [5], where each cognitive

* Presented at the Summer Solstice 2015 International Conference on Discrete Models of Complex Systems, Toronto, Ontario, Canada, June 17–19, 2015.

agent learns by observing the outcomes from the actions of cognitive agents that have already attempted to carry out a task and imitating the successful ones.

We consider the model of naïve creatures learning to cross a highway introduced in [6–10]. In this model, creatures use a simple decision making formula and build their knowledge base (KB) by observing the performance of the creatures that attempted to cross the highway earlier. We investigate the effects of creatures' KB accumulation through repetition and its interaction with other model parameters on the number of "successful", "killed", and "queued" creatures when learning to cross a highway. In particular, we investigate the effects of transferring of the KB after building it through many populations of creatures within the same environment (*i.e.*, the highway with the same traffic density) as opposed to building it through only a single population of creatures within the same environment. In our analyses, we use regression trees as they work well in scenarios where the data is too complex to model using more standard methods such as ANOVA linear models. Regression tree analyses do not make any statistical model assumptions and are instead, heavily algorithmic [11]. Also, tree-based analyses handle interactions among the factors very well due to its condition based results [11]. This is an important aspect in our investigation.

The paper is organized as follows: Section 2 describes the model; Section 3 describes simulation setup and resulting data; Section 4 presents regression tree analysis of the simulation data, and Section 5 reports our conclusions and outlines future work.

2. Model of naïve creatures with fear and desire learning to cross a highway

For completeness of the paper, we review the main features of the model and refer the reader to [6–10] for details. In this work, we assume that the environment is a single-lane unidirectional highway without intersections represented by the modified Nagel–Schreckenberg cellular automaton model. See [12–15] for the details. The model consists of four steps that are applied simultaneously to all cars: acceleration, safety distance adjustment, randomization, and change of position. The cars are generated at "starting cells" randomly with car creation probability (Car Prob.) and are assigned a random speed between zero and the maximum allowed speed for cars, which is set in the configuration file.

At each time step, a creature is generated only at the crossing point (CP) set at the initialization step and is placed into the queue at this CP. Each generated creature falls with equal probability (0.25) into one of the four categories: (1) no fear nor desire; (2) only fear; (3) only desire; (4) both fear and desire. The creatures' attributes play a role in their decision making process on whether or not to cross the highway through the values of fear (aversion to risk taking) and desire (propensity to risk taking) that creature may experience. They want to cross the highway without being killed by the oncoming vehicles and have a strong instinct to survive.

Each creature is "an autonomous entity capable of interacting with its environment and other agents" capable of: (1) matching simple patterns; (2) evaluating distances in an approximate way; (3) evaluating the velocity of moving vehicles in an approximate way; (4) assigning a discrete number to an approximate class; (5) witnessing what had happened to the creatures that previously crossed the highway at this crossing point (with exclusion of the first creature); (6) evaluating what they witnessed in (5), *i.e.* if it was successful or not; (7) imitating the creatures which crossed successfully; (8) deciding not to cross and wait for better conditions or to look for a different crossing point when unsuccessful crossings outnumber the successful ones. All of these allow each crossing point (CP) to build one knowledge base (KB) during the experiment that is available to all creatures at that CP.

The creatures attempt to cross the highway having a limited horizon of vision and perceiving only fuzzy levels of distance (*e.g.*, "close", "medium", "far") of cars within this horizon and their speeds (*e.g.*, "slow", "medium", "fast"). The ranges for these qualitative categories are set in the simulator's configuration file. The creatures may build up in the queue as a result of not crossing at each time step. If the simulation setup permits, after deciding not to cross the highway, a creature may move randomly along the highway horizontally in either direction to a new CP or it may stay at the same CP with equal probability of 1/3. The number of horizontal cells a creature may move in one-time step is 1 and the maximum distance the creature may deviate from its original CP in both directions is 5. If the creature at the top of a queue leaves the queue, the creature that was behind moves to the top of the queue. When a creature crosses the highway at a given CP, information is recorded into the knowledge base (KB) of all the creatures at this CP. The information about qualitative description of velocity (*e.g.*, such as "fast", "medium", and "slow") and of the distance (*e.g.*, such as "close", "medium", "far") is stored, respectively, in the columns and rows of the KB table.

The KB table is initialized as *tabula rasa*, *i.e.* with all its entries set to 0, allowing creatures to cross the highway regardless of the observed (distance, velocity) levels until the first successful crossing of a creature, or five consecutive unsuccessful crossings, whichever comes first. If a creature successfully crossed the highway, the perceived (distance, velocity) score in the KB table is increased by one point. If the creature was killed, it is decreased by one point.

After the initialization of the simulation, each creature at the top of the queue consults the KB table to decide if it is safe or not to cross. Its decision is based on the implemented intelligence/decision making algorithm, which for a given (distance, velocity) pair combines the "success ratio" of crossing the highway for this (distance, velocity) pair with the creatures' fear and/or desire values, as follows.

For each (distance, velocity) pair at each time step, the numerator in the success ratio is the current value from the KB table, *i.e.* it is the number of "successful crossing" minus the number of "unsuccessful crossings" for this (distance, velocity) pair up to this time. The denominator is the total number of creatures that have crossed the highway successfully regardless of the (distance, velocity) combination up to this time; *i.e.* it is the number describing the creatures' entire population success up to this time. If for some (distance, velocity) configuration at the simulation start, all creatures are killed, ratio becomes "$-5/0$". In this case, we set the success ratio to zero since "division by zero" is undefined.

A randomly generated creature will base its decision on the formula: (1) *success ratio* + *value of desire* − *value of fear*, if it has both fear and desire; (2) *success ratio* − *value of fear*, if it has only fear; (3) *success ratio* + *value of desire*, if it has only desire; (4) *success ratio*, if it has no fear and no desire. If a creature and a given (distance, velocity) combination yield from the formula a value that is less than zero, then the creature will not attempt to cross the highway under this condition and it will wait for a configuration for which the value of the formula is non-negative or it may decide to move to another crossing point.

The main simulation loop of the model consists of: (1) generation of cars at each lane of the highway using the Car Prob.; (2) generation of creatures at each CP with their attributes; (3) update of the car speeds according the Nagel–Schreckenberg model; (4) movement of the creatures from the CP queues into the highway (if the decision algorithm indicates this should occur); (5) update of locations of the cars on the highway. This includes passing other cars in the case of multi-lane highway. It also includes the logic to check if any creature has been killed; (6) advancement of the current time step. After the simulation has been completed, the results are written to output files using an output function.

3. Simulation setup and simulation data

A single run of the simulation lasts for a number of time steps set up at the initialization step and various output files are generated, *e.g.* (1) file containing the total number of creatures that have successfully crossed the highway, the total number of creatures that were killed while crossing, and the number of queued creatures at the end of the simulation; (2) file containing time-dependent data, *i.e.* at the end of each discrete time step, the number of creatures that have successfully crossed the highway up to this time, the number of creatures that were killed while crossing up to this time, and the number of queued creatures at this time; (3) file containing the state of the KB table at each time step. A large amount of data files is generated when the simulation is looped many times both at a particular configuration of the adjustable simulation parameters/factors (to create repetitions) and also at different configurations of the parameters (in order for comparison).

We consider the data generated from the simulation looped many times. The parameters that remain constant are: one-lane highway with a length of 120 cells (900 meters long), 1511 time steps, 30 repetitions, random deceleration equal to 0 (there are no erratic drivers), and a 3 by 4 KB table with an extra entry. The KB table has 3 groupings of distance and 4 groupings of speed. The creatures in this case specifically perceive: (1) "close" for a vehicle being 0 to 3 cells away, "medium" if it is 4 to 5 cells away, "far" if it is 6 to 7 cells away and "out of range" if it is 8 or more cells away, regardless of the velocity of the vehicle, and this is encoded in the extra entry; (2) "slow" when the perceived velocity of a vehicle is 0 to 3 cells per time step, "medium" when it is 4 or 5 cells per time step, "fast" when it is 6 or 7 cells per time step, and "very fast" when it is 8 to 11 cells per time step. A vehicle's maximum speed is at 11 cells per time step.

There are 6 parameters that vary through the main simulation loop. These parameters are: (1) car creation probability (Car Prob.); (2) crossing point (CP); (3) value of fear, (4) value of desire; (5) the KB transfer direction (KB Transf.), and (6) horizontal creature movement (Horiz. Cre.).

The Car Prob. determines the density of the vehicle traffic and it varies between the values: 0.1, 0.3, 0.5, 0.7, and 0.9. A vehicle is generated at the start of the highway at each time step with a given Car. Prob.

The CP determines the location at which the creature will cross the highway and it varies between the values: 40, 60, and 80 (the cell number of the highway). The distance from where the vehicles are generated is important because it will affect the nature of the vehicle traffic. For example, there will likely be more vehicles traveling at maximum speed and in a more homogeneous manner near CP 80 than at CP 40.

The value of fear and value of desire parameters both vary between the values: 0, 0.25, 0.5, 0.75, and 1. Being a part of the decision formula, these values influence the creatures' decision making process of whether or not to cross the highway.

The KB Transf. varies from: "none", "forward", and "backward". This parameter determines whether or not the KB table at the end of one run of the simulation is transferred to the beginning of the simulation at a different traffic environment (*i.e.*, with a different value of Car Prob.). When the KB Transf. is set to "none", the KB table is not transferred from an environment with one car traffic density to an environment with another car traffic density. When KB Transf. is set to "forward", the KB table is transferred from a less dense traffic environment to the one with immediately denser traffic. In this case, the simulations start in the environment with Car Prob. 0.1 and with KB table containing all entries of 0. Thus, the data with Car Prob. equal to 0.9 starts with a KB accumulated over the other four less dense traffic environments. When the KB Transf. is set to "backward", the KB table is transferred from denser traffic environment to the one with immediately less dense traffic. In this case, the simulations start in the environment with Car Prob. 0.9 with the KB table with all entries of 0 in the KB table. Thus, in the "backward" case, the KB table in the environment with Car Prob. equal to 0.1 is accumulated from the other four traffic environments with higher car density. This transfer of KB in either direction happens after building the KB by a single population of creatures within a given environment (Framework (I)). Under a different scenario (Framework (II)), the transferring of the KB in either direction happens after building the KB by several populations of creatures within a given environment. Thus, in Framework (II), the KB tables are always transferred from the current repetition to the next repetition within any particular configuration of the parameters' values. In this framework, even when KB Transf. is "none", the KB table is still transferred at the end of a simulation from one repetition to the next one within each environment with the same car density. In Framework (II) when KB is "forward" ("backward"), the KB is additionally transferred at the end of all the repetitions in the environment with lower (higher) Car Prob. to the one with an immediately higher (immediately lower) Car Prob. The distinction between Framework (I) and Framework (II) is important as the amount of learning under each Framework is different and affects the creatures' success in crossing the highway. In Framework (II), the KB tables become much more developed as there is much more transferring of the KB tables occurring.

The parameter Horiz. Cre. varies form 0 and 1 and determines whether or not the creatures can decide to move horizontally in either direction if they decide not to cross the highway. The creatures are only allowed to move horizontally if Horiz. Cre. equals 1.

Every combination of the 6 parameters/factors is considered with 30 repeats each. The "summary" files of each configuration are collected into one large dataset containing 67,500 rows. The columns include the value for each configuration parameter and also the main response variables: the number of successful creatures, the number of killed creatures, and the number of queued creatures at the end of the simulation. The dataset is created using R code.

4. Regression tree analysis

The considered model of creatures learning to cross a highway is a multifactorial model with some factors having several levels. Analysis of factor effects and their interactions is often a challenging task in multifactorial models with many independent variables. Deciding on how to model data coming from a complex simulated dynamical system in order to extract patterns of likely causes and influences may not always be easy. One may apply ANOVA to eliminate irrelevant relationships in order to improve the understanding of the system. However, due to the number of assumption under which ANOVA can be applied to produce meaningful results, this may not always be feasible. Thus, one may need to look for other methods, like regression trees, which allows us to perform the analysis regardless of the violation of the ANOVA assumptions to extract information about the system behavior, as in our study where we first applied ANOVA linear models.

In our study, the datasets are set up appropriately under the experimental design paradigm where each simulation parameter is treated as a factor. When we first applied ANOVA linear models there were difficulties, which included obtaining residuals with a non-random trend and interpreting models with interaction effects between factors. For example, just trying to examine the interaction between fear and desire requires the interpretation of $5 \times 5 = 25$ coefficients as both those factors have 5 levels. The application of regression trees works well in scenarios where the data is too complex to model using more standard methods such as ANOVA linear models. Regression tree analysis does not make any statistical model assumptions and is instead, heavily algorithmic [11]. Fortunately, tree-based analyses also handle interactions among the factors very well due to its condition based results [11]. Though the application of regression trees is best suited for predicting the results for future data/determining a decision rule, it is for precisely these reasons above, that regression tree models are applied in order to better understand the relationship among the simulation parameters and the responses: the number of successful, killed, and queued creatures.

4.1. Methodology

The methodology of trees follows the approach from [11]. Trees focus on partitioning the data space into many different regions. The estimated value of the response in a region is simply the mean of those that belong to the region. This means that all data points in the same region will have the same predicted value. The algorithm used in regression trees is as follows:

1. Consider for the data, all partitions into 2 groups that are separated by a difference in exactly 1 parameter.

2. Calculate the residual sum of squares for all partitions:
$$\mathrm{RSS}(\mathrm{Partition}) = \mathrm{RSS}(\mathrm{Part}_1) + \mathrm{RSS}(\mathrm{Part}_2)\,,$$
where $\mathrm{RSS}(\mathrm{Part}_i)$ is the residual sum of squares for group i using the mean response within each group as the predicted value.
3. Create a split according to the partition with the minimum residual sum of squares.
4. Repeat steps 1 to 3 for the two new subsets.

As this process continues, the tree is growing larger with more nodes. This process lasts until the reduction in RSS between the previous and current tree is under a small threshold. As explained in [11], a strategy is to prune a large tree (more partitions/nodes) created using a small threshold value. The process of tree pruning involves removing the nodes that least influence the minimization in residual sum of squares using k-fold validation [11]. In this paper, the analysis is performed using R with focus on the “rpart” function from the “rpart” package [16]. Multiple tree models are created using data under both frameworks on the different responses (successful, killed, and queued creatures) at different values of the complexity/tuning parameter which relates to the threshold for when to stop growing the tree. The term “tuning parameter” will be used rather than “complexity parameter” as to avoid confusion with the other terms, “Car Prob.” or “crossing point”. The pruning of the trees is considered for larger trees. In each case, all of the simulation parameters are used as factors. When viewing the tree figures, the path continues on the left side if the listed condition is true. Only the constructed trees of reasonable size (not too small nor too large) are included. Indeed, these fitted models help to illustrate the results from previous work in a more quantitative way. The following tree diagrams in this paper do not highlight the variation of the data. However, we have explored the variation of the data in previous work. If the fear value and Horiz. Cre. value are both 0, then there is very little variance in the data. Allowing for horizontal creature movement will slightly introduce more variance, while setting a non-zero value of fear will greatly increase the variance.

4.2. Data without knowledge base transfer between repeats — Framework (I)

Considering the tuning parameter of 0.01, the following trees are created and shown in Figs. 1, 2, and 3. In these three figures, the lengths of the branches are proportional to the amount of reduction of the residual sum of squares.

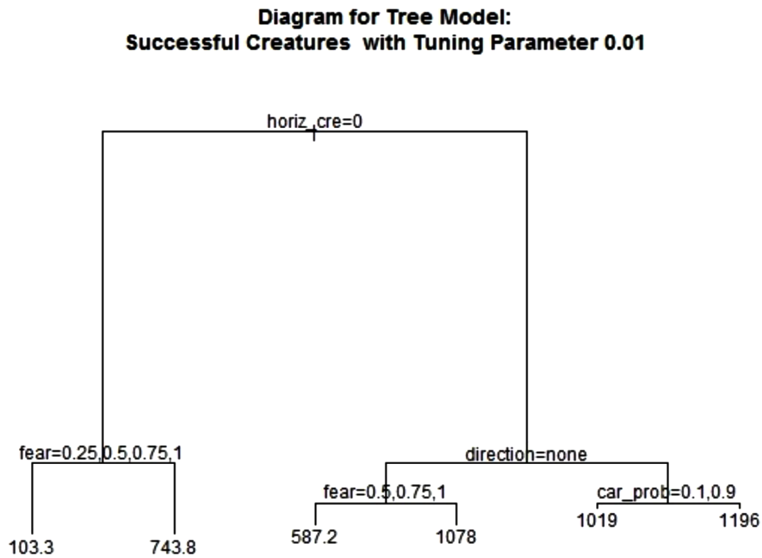

Fig. 1. Tree diagram for number of successful creatures with tuning parameter 0.01 under Framework (I). If the listed condition is true, the flow moves to the left side. Reduction in RSS is proportional to branch length.

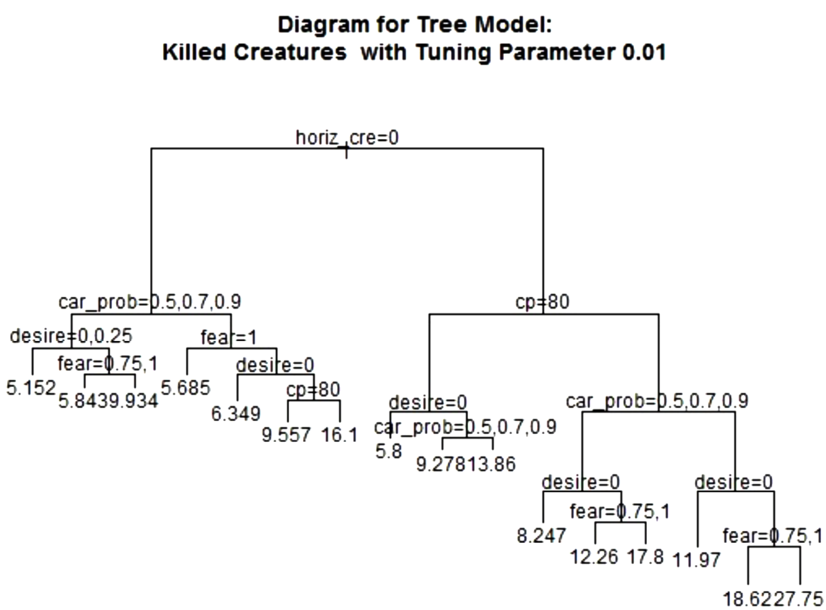

Fig. 2. Tree diagram for number of killed creatures with tuning parameter 0.01 under Framework (I). If the listed condition is true, the flow moves to the left side. Reduction in RSS is proportional to branch length.

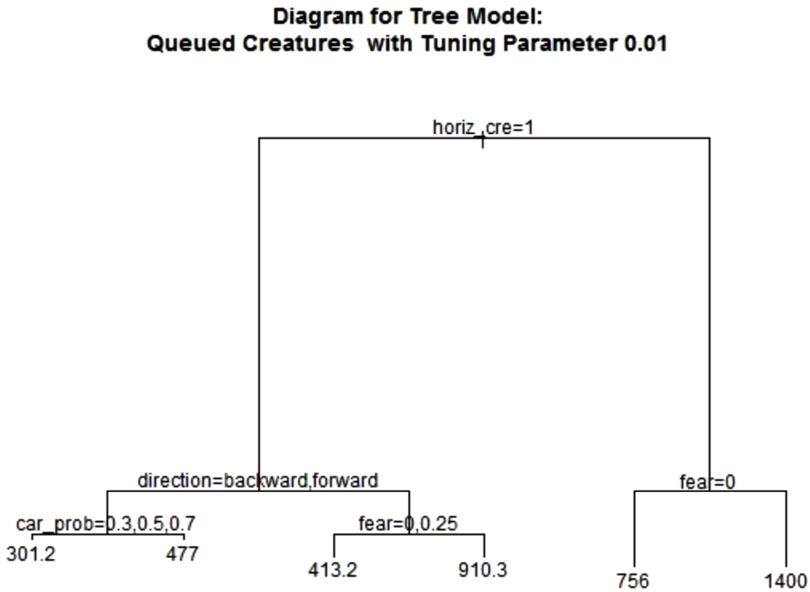

Fig. 3. Tree diagram for number of queued creatures with tuning parameter 0.01 under Framework (I). If the listed condition is true, the flow moves to the left side. Reduction in RSS is proportional to branch length.

From the tree for successful creatures (Fig. 1), the main reduction in residual sum of squares is through dividing the data by whether or not the creatures can move horizontally along the highway. When the creatures are not able to move horizontally, a simulation with a non-zero value for fear is estimated to have 103.3 successful creatures, while a simulation with fear set to 0 is estimated to have about 743.8 successful creatures. If the creatures can move horizontally, the main factors influencing the number of successful creatures are KB Transf. and Car Prob. when the knowledge base is indeed transferred from one traffic environment to the next one. When the knowledge base does not transfer to the next traffic environment (direction = "none"), then the estimated number of successful creatures is 1,078 for fear value of 0 and 0.25 and it is only 587.2 for fear value 0.5, 0.75 and 1. Thus, higher fear values have a significantly depressing effect on number of successful creatures.

There is an interesting result in the tree of Fig. 1 when KB transfer direction is different from "none", *i.e.* when KB Transf. is "forward" or "backward". For Car Prob. either 0.1 or 0.9, the estimated number of successful creatures is 1,019, which is actually lower than the estimated number 1,196 of successful creatures for other values of Car Prob. This is because the tree considers both "forward" and "backward" transferring of KB tables in the same category and it combines the data coming from two extreme cases: the data coming from the environments with the lowest amount of knowledge (data with Car Prob. 0.1 & KB Transf. "forward" and data with Car Prob. 0.9 & KB Transf. "backward") with the data coming from the environments with highest amount of knowledge (data with Car Prob. 0.9 & KB Transf. "forward" and data with Car Prob. 0.1 & KB Transf. "backward"). The scenarios when Car Prob. is either 0.3, or 0.5, or 0.7 have a higher mean number of successful creatures, because in these scenarios regardless what is the direction of KB transfer, "forward" or "backward", the creatures always start with some preexisting knowledge, *i.e.* they do not start *tabula rasa*. This result shows that: (1) the knowledge that the creatures acquire helps them to succeed; (2) starting *tabula rasa* has detrimental effect on creatures' success which cannot be compensated easily even by the data coming from the environments with the highest amount of knowledge. Furthermore, the creatures' ability to move horizontally (Horiz. Cre. equal to 1) contributes to the higher estimated numbers of successful creatures. This is consistent with results reported in our previous work.

The tree for the number of killed creatures (Fig. 2) is larger than the one for successful creatures. Given the same tuning parameter of 0.01, we observe that more factors may affect the number of killed creatures than the number of successful creatures. It is estimated that more creatures are killed under the scenario where creatures can move horizontally. In this scenario, the smallest estimated number of killed creatures is 5.8 and their largest estimated number is 27.75. If they do not move horizontally, their smallest estimated number is 5.152 and their largest estimated number is 16.1. Taking the time to interpret this tree supports the idea that higher values of desire promote a higher number of killed creatures and higher values of fear promote a lower number of killed creatures. For example, when creatures are not allowed to move horizontally, for values of desire less or equal to 0.25, the highest estimated number of killed creatures is 6.349, while for values of desire greater than 0.25, this number is 16.1. In the case when the creatures are allowed to move horizontally, for desire value 0, the highest estimated number of killed creatures is 11.97, while if desire value is greater than 0, this number is 27.75. Looking at the fear values, in the case when creatures are not allowed to move horizontally, if these values are 0.75 and 1, then the highest estimated number of killed creatures is 5.843, while if these values are less than 0.75, then the highest estimated number of killed creatures is 16.1. This tree analysis also shows that selection of crossing point is influential. Setting the crossing point further away from the start of highway will lower the estimated number of killed creatures in many cases.

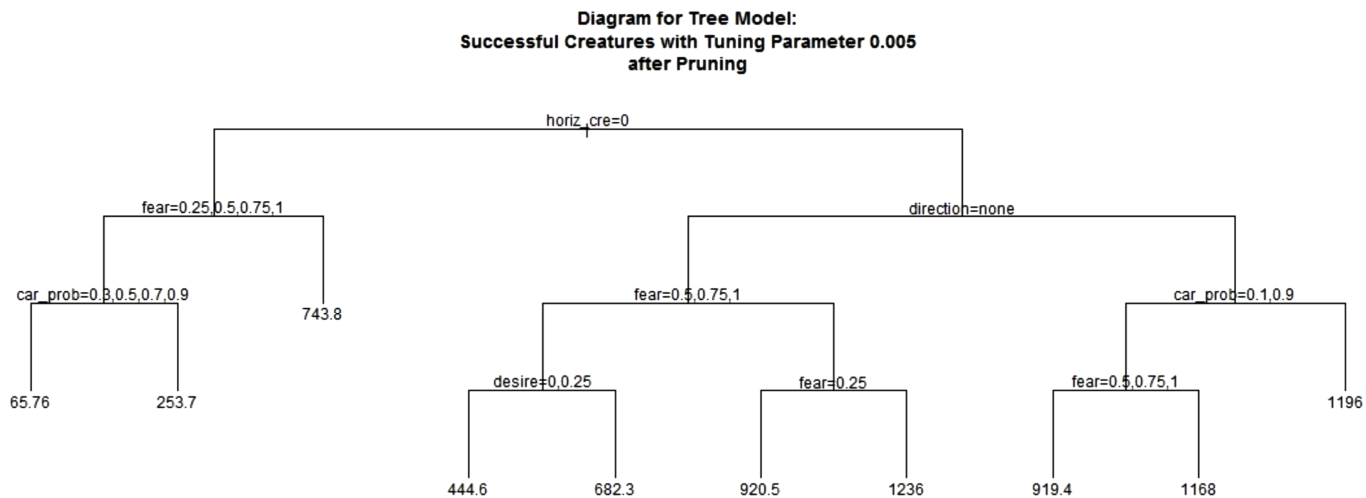

Fig. 4. Tree diagram for number of successful creatures with tuning parameter 0.005 under Framework (I). The technique of pruning the tree is applied. If the listed condition is true, the flow moves to the left side. The length of the branches is of uniform length.

For example, when creatures are not allowed to move horizontally, the estimated number of killed creatures is 9.557 at CP 80, while at either CP 40 or CP 60, combined, it is 16.1. When creatures are allowed to move horizontally, then the highest number of killed creatures at CP 80 is 13.86, while at either CP 40 or CP 60, combined, it is 27.75.

As expected, the tree for the number of queued creatures (Fig. 3) is opposite to, or a mirror of the tree for successful creatures. This is because: (1) the number of successful, killed and queued creatures at the end of the simulation adds up to the total number of generated creatures, fixed at 1511; (2) the numbers of killed creatures are over all small in comparison with successful ones, *i.e.* the highest estimated number of killed creatures is 27.75, while the lowest estimated number of successful creatures is 103.3. Notice that the starting condition on the Fig. 1 is Horiz. Cre. 0, while in Fig. 3 it is Horiz. Cre. 1. In Fig. 3, we notice that if creatures are not allowed to move horizontally and fear is greater than zero, then the estimated number of queued creature is 1,400, which is very high given that the total number of creatures is 1511. However, if they are allowed to move horizontally, the highest estimated number of queued creatures is 910.3, which is for fear values greater than 0.25. Thus, it seems that if creatures are not allowed to move horizontally, fearful creatures at the beginning of simulations may prevent all other creatures from attempting to cross the highway. Figure 4 is the tree for number of successful creatures using tuning parameter 0.005. The technique of pruning the tree is applied, as the smaller tuning parameter allows for larger trees. The length of each branch is now a constant length to make the display of the tree readable. This tree is more in-depth than the previous tree for successful creatures in Fig. 1. It has additional partitions for the value of fear, Car Prob. and also the value of desire. Without horizontal creature movement, when there is a non-zero value of fear, more creatures are able to cross the highway successfully with Car Prob. set to 0.1, *i.e.* with the estimated number of 253.7 *vs.* 65.78. When creatures can move horizontally, the additional partitions further support the idea that higher values of fear decrease the number of successful creatures, while higher values of desire increase this value.

4.3. Data with knowledge base transfer between repeats — Framework (II)

An additional factor to consider under this framework is the repetition number. Considering the tuning parameter of 0.01, the following trees are created and shown as Fig. 5 and Fig. 6.

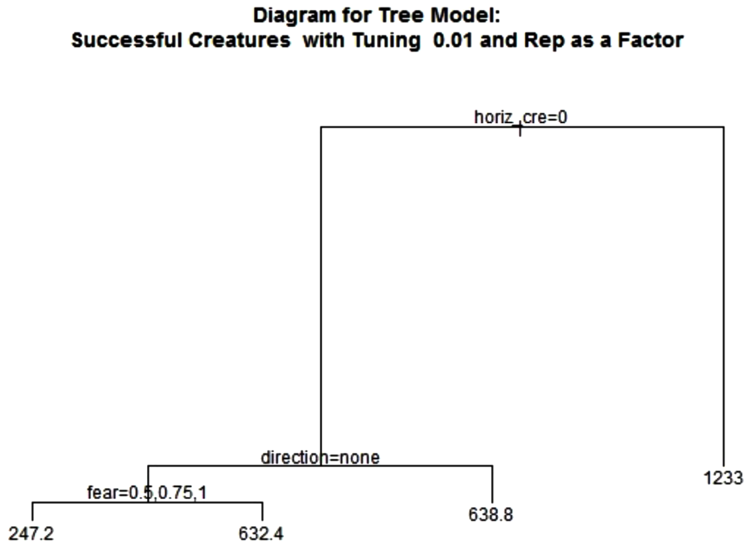

Fig. 5. Tree diagram for number of successful creatures with tuning parameter 0.01 under Framework (II). If the listed condition is true, the flow moves to the left side. Reduction in RSS is proportional to branch length. The repetition number is also considered as a factor.

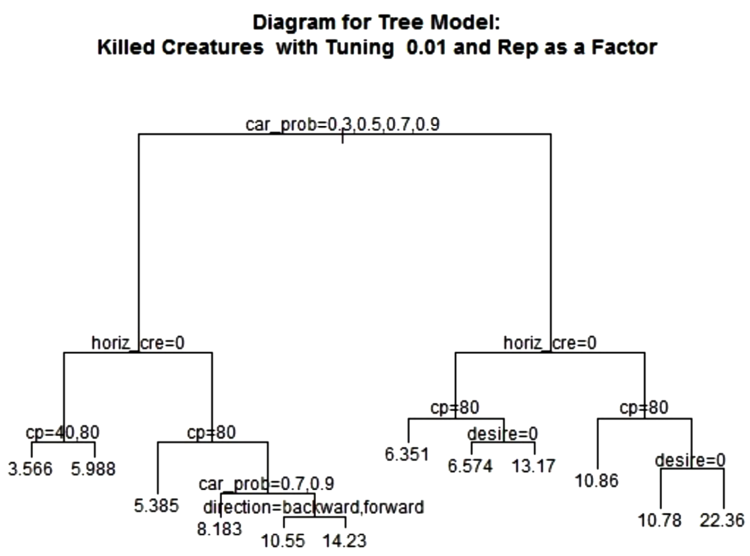

Fig. 6. Tree diagram for number of killed creatures with tuning parameter 0.01 under Framework (II). If the listed condition is true, the flow moves to the left side. Reduction in RSS is proportional to branch length. The repetition number is also considered as a factor.

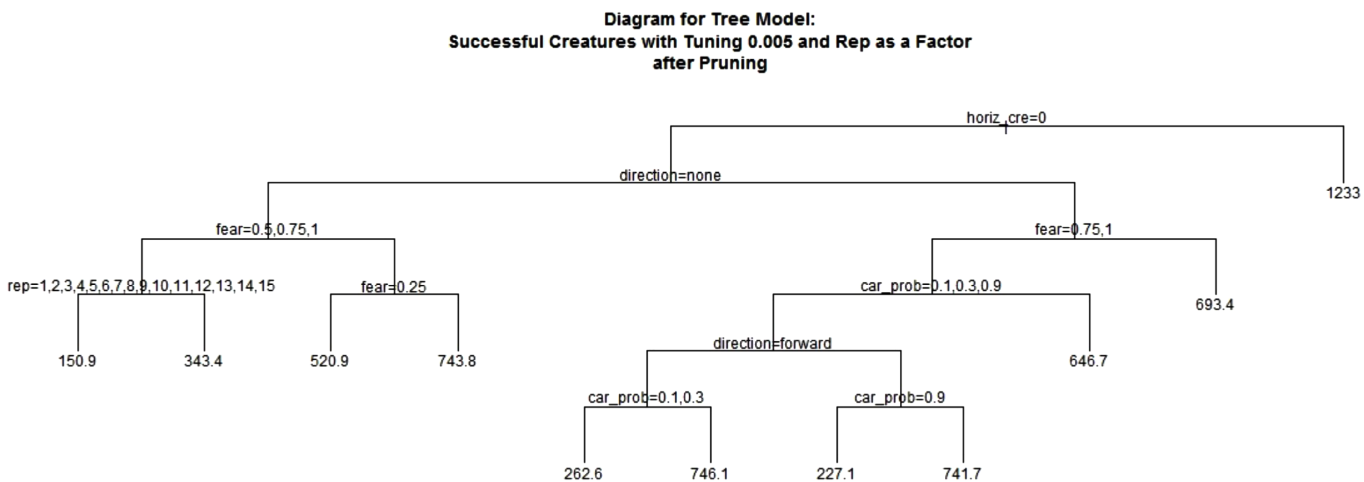

Fig. 7. Tree diagram for number of successful creatures with tuning parameter 0.005 under Framework (II). The technique of pruning the tree is applied. If the listed condition is true, the flow moves to the left side. The length of the branches is of uniform length. The repetition number is also considered as a factor and is used in the decision rule of this particular tree.

The tree for the number of queued creatures is excluded as the same mirrored nature occurs similar to the results under Framework (I). The lengths of the branches are proportional to the amount of reduction of the residual sum of squares. Contrasting with the equivalent tree model under Framework (I), *i.e.* with Fig. 1, Fig. 5 illustrates the dominance of the Horiz. Cre. parameter for the number of successful creatures due to the lack of additional partitioning when Horiz. Cre. equals 1. In Fig. 5, the corresponding estimated number of successful creatures is 1233 and it is higher than all the estimated numbers in Fig. 1. The observed higher numbers in Fig. 5 are a consequence of the additional KB transferring/learning occurring under Framework (II) not present under Framework (I). This phenomenon and the described dominant effect of the Horiz. Cre. parameter value 1 was also observed in our previous work. In the tree for the killed creatures (Fig. 6), similar interpretations from the tree for killed creatures for the data under Framework (I) can be made. The initial split however, is on Car Prob. rather than Horiz. Cre.

Figure 7 is the tree for number of successful creatures using tuning parameter 0.005. It is an equivalent of Fig. 4. As before, the larger tree emerges when the smaller value of the tuning parameter is considered and we make the display of the tree more readable by setting the length of each branch to a constant length. This tree is more in-depth than the previous tree for successful creatures in Fig. 4. It has additional partitions for the value of fear, Car Prob. and KB Transf. Despite creating a larger tree, there are still no additional partitions for when Horiz. Cre. equals 1 (at least after being pruned), as in Fig. 5. This tree model also considers the repetition number to be influential to the number of successful creatures in situations with Horiz. Cre. equal to 0, with KB Transf. equal to "none", and with one of the higher fear values (0.5, 0.75, or 1). Under these scenarios, if the repetition value is in the higher half of the values (greater than 15), the number of estimated successful creatures is greatly increased: comparing 343.4 to 150.9. This is consistent with the scheme of Framework (II) and confirms that the higher value repetitions will have a more developed knowledge base leading to a larger number of successful creatures.

5. Conclusions and future work

The presented results show that a more developed KB or allowing the creatures to move horizontally along the highway improves the number of creatures that successfully cross. Higher values of fear will decrease the number of killed creatures at the cost of also decreasing the number of successful creatures. The value of desire is less influential but has an opposite effect when compared to the value of fear, *i.e.* it increases the numbers of successful and killed creatures. The performed analysis quantifies and confirms our qualitative observations presented in previous works. We plan to continue our analyses by exploring the effects of other model parameters and different decision making formulas on creatures success rate of crossing the highway.

The authors acknowledge useful discussions with Bruno Di Stefano, Hao Wu and Shengkun Xie. A.T.L. acknowledges partial financial support from the NSERC of Canada.

REFERENCES

[1] S. Russell, P. Norvig, *Artificial Intelligence, A Modern Approach*, Pearson Education Limited, 2014.

[2] D.L. Poole, A.K. MacKworth, *Artificial Intelligence, Foundations of Computational Agents*, Cambridge University Press, Cambridge, US, 2010.

[3] J. Ferber, *Multi-Agent Systems. An Introduction to Distributed Artificial Intelligence*, Addison Wesley, London, UK, 1999.

[4] Ch.L. Nehavin, K. Dautenhahn, *Imitation and Social Learning in Robots, Humans and Animals*, Cambridge University Press, Cambridge, UK, 2007.

[5] W. Hoppitt, K.N. Laland, *Social Learning, An Introduction to Mechanisms, Methods, and Models*, Princeton University Press, Princeton 2013.

[6] B.N. Di Stefano, A.T. Lawniczak, *Acta Phys. Pol. B Proc. Suppl.* **5**, 21 (2012).

[7] A.T. Lawniczak, J.B. Ernst, B.N. Di Stefano, *Lect. Notes Comput. Phys.* **7495**, 425 (2012).

[8] A.T. Lawniczak, J.B. Ernst, B.N. Di Stefano, *Proceedia Comput. Sci.* **18**, 2611 (2013).

[9] A.T. Lawniczak, B.N. Di Stefano, J.B. Ernst, *Lect. Notes Comput. Phys.* **8751**, 688 (2014).

[10] A.T. Lawniczak, B.N. Di Stefano, J.B. Ernst, IEEE Symposium on Intelligent Agents (IA) 2014, Orlando, FL, 09–12 Dec. 2014, p. 30, DOI:10.1109/IA.2014.7009455.

[11] J.J. Faraway, *Extending the Linear Model with R: Generalized Linear, Mixed Effects and Nonparametric Regression Models*, Boca Raton, FL: Chapman & Hall/CRC, 2006.

[12] K. Nagel, M. Schreckenberg, *J. Physique I* **2**, 2221 (1992).

[13] A.T. Lawniczak, B.N. Di Stefano, Proc. of Automata-2008, *Theory and Applications of Cellular Automata*, Luniver, Press, UK, 2008, p. 527.

[14] A.T. Lawniczak, B.N. Di Stefano, *Lect. Notes Comput. Phys.* **6350**, 600 (2010).

[15] A.T. Lawniczak, B.N. Di Stefano, *Acta Phys. Pol. B, Proc. Suppl.* **3**, 479 (2010).

[16] T. Therneau, B. Atkinson, B. Ripley, rpart: Recursive Partitioning and Regression Trees, retrieved from `http://CRAN.R-project.org/package=rpart`, 2014.