in: F. Squazzoni (Ed.), Fifth Conference of the European Social Simulation Association (ESSA), Brescia September 1–5, 2008. Brescia, Italy.

Simulating Collaborative Writing: Software Agents Produce a Wikipedia

Klaus G. Troitzsch

Universität Koblenz-Landau, Universitätsstraße 1, 56070 Koblenz, Germany
kgt@uni-koblenz.de

Abstract. This paper originates from the FP6 project "Emergence in the Loop (EMIL)" which explores the emergence of norms in artificial societies. Part of work package 3 of this project is a simulator that allows for simulation experiments in different scenarios, one of which is collaborative writing. The agents in this still prototypical implementation are able to perform certain actions, such as writing short texts, submitting them to a central collection of texts (the "encyclopaedia") or adding their texts to texts formerly prepared by other agents. At the same time they are able to comment upon others' texts, for instance checking for correct spelling, for double entries in the encyclopaedia or for plagiarisms. Findings of this kind lead to reproaching the original authors of blamable texts. Under certain conditions blamable activities are no longer performed after some time.

Keywords: Agent-Based Simulation; Norm Emergence; Immergence

Introduction

This paper originates from the FP6 project "Emergence in the Loop (EMIL)" which explores the emergence of norms in artificial societies. For this, the project defined the architecture of a normative agent [1]. A normative agent is capable of recognising norms, adopting norms under certain circumstances, planning actions and deciding whether to abide by the norm or violate it and additionally to defend it against others. The scenario currently of interest in the EMIL project is collaborative writing: Agents are expected to contribute to a fictitious Wikipedia and participate in the usual Wikipedia discussions. One of EMIL's deliverables is a simulator that can cope with the needs defined by the agent architecture and allows for running simulations in various scenarios. The current work originated from the necessity of finding out which capabilities of the agents, their environment and the simulator will have to be built into the latter.

Generally speaking (for more see the requirements analysis in [5]), the simulator will host agents with different capabilities which will act in an environment that

allows them to communicate (where communication is one of the major challenges of this project [10]) and that can be changed by the agents.

Thus the whole effort consists of three parts:

- building the simulator that can host the agents designed for different scenarios according to [5],
- designing the general architecture of the agents according to [1, 3] (these papers require that agents should consist of a norm recogniser, receiving and interpreting messages in the light of its normative frame, a norm adoption engine, generating goals from the contents of this normative frame, a decision maker, generating intentions from goals, and a normative action planner, working out the actions necessary for fulfilling the intentions), and
- describing the scenario in a way that a modeler can endow the agents with all the features that they need in the respective scenario.

Requirements of a Wikipedia scenario

In this paper we concentrate on the second and third point, i.e. developing some first ideas how scenario builders could enable agents to write articles in a Wikipedia and comment on them.[1] The Wikipedia simulation is the scenario which will be implemented in any case; other scenarios may follow, for them, perhaps other (additional) description features may be necessary.

We use the description of messages already discussed in [1] and [5], where we defined messages with the help of four attributes: the sender x, the modal M, the observer or recipient y and the contents or action transmitted α.

The modals defined in [4] comprise assertions (A), behaviours (B), requests (R), deontics (D), validations (V), and sanctions (S), i.e. messages can either convey a simple assertion, they can be just implicit as observable behaviour, they can be questions, they can be deontics (where they refer to generalised activities ("in a certain situation you must perform this kind of activity", "in a certain situation you must not perform an activity of this kind", "in a certain situation you are allowed to perform an activity of this kind, but it is up to you"). Validations refer to a particular activity and convey content such as "you should not have done this" or "this was a good action", whereas sanctions entail fines or punishments as well as rewards [2, p. 14]. Thus deontics are subdivided into prescriptions, proscriptions and permissions, all of which have very different effects on the recipient of a message. The same

[1] For a related approach see [9]. In this approach the focus is on the visibility of messages and the reputation of their authors, not on the emergence of norms that dictate what messages should look like. Instead, messages in [9] have no identifiable content, at least not in the published version, although in section 3.6 of their paper the authors mention that "a message can additionally be specified with attributes such as author, topic, keyword, relevance, acceptance, rejection, and so on." But the contents run only under "and so on". Mutual referencing between articles in their simulation is more or less random, according to three distinct referencing or communication styles. Styles like these might emerge from a more sophisticated version of our approach.

applies to validations and sanctions (the latter also include incentives, thus there are sanctions of different strength and direction ─ from a highly positive sanction, such as a prize or reward, down to a highly negative sanction, such as a condemnation to jail).

At the beginning of a simulation agents have a repertoire of messages (or an algorithm that enables them to formulate new messages) that they want to send to all other agents, i.e. for inclusion in the Wikipedia (where it is copied into a member of the class article), and all agents are allowed to do so, i.e. they will have an attribute that informs whether an agent is an authorised author or not (thus, at the beginning all of them will be entitled to contribute). After some time the Wikipedia, i.e. the environment, contains a list of articles which are not yet connected. These articles contain, of course, the name of the author or sender, have the modal A, as they are just assertions; their recipients are all other agents, and the contents is some string of arbitrary length, the first few elements of which have the special meaning of a keyword.

The contents string could be of any kind, for instance a simple bit string whose first eight bits serve as the entry title (such that exactly 256 entries would be possible). But it might be more helpful to have something like a text separated into words by blanks or some other punctuation. In this case, words could consist of alternating vowels and consonants forming a very primitive language which conveys no meaning, but agents could (perhaps more easily than in a simple bit string) mull over similarities and differences among Wikepedia articles. To limit the number of possible tokens (words), the character repertoire could be restricted to only three vowels and five consonants ("aloha wiki sisal").[2] Another option could be the task of sorting letters alphabetically and separating different fonts from each other, but this implies an exogenous "fitness measure" for the character strings instead of an emergent practice of article writing and citing.

Besides writing and reading articles and comments and edits, agents will also have to generate a list (each of them separately) of other agents whom they have identified and evaluated for their authority (the "board of authorities" in [4]), as their norm recogniser ─ an engine that matches new perceptions with the contents of agent's memory ─ does not only use the contents of a message but also its sender to evaluate the importance and salience of an utterance (thus an utterance could be more important when it comes from an agent with high "importance" ─ see below ─, or be more salient when it has been received several times).

Agents scan the articles for similarities and comment on them. These commenting actions can be of the following types:

- If agent x finds a match between a keyword of one of its own articles a_x and an article b_y published by someone else (y), then it includes a link to x's own article a_x in y's article b_y (which makes it necessary to add an instance variable keeping a link list to the article class (an element of a link list also contains the sender and, perhaps, a time stamp). Adding a link would qualify as modal B (behaviour).

2 As a prototype we have a small NetLogo program whose agents write articles in this primitive language, sort them by keywords, and also sort all the occurring words both alphabetically and according to word frequency. More about this rapid prototype in section 3. The output can be found in the appendix.

- Articles that have no similarity at all to any other article (i.e. articles with no or few links to other articles) could be less welcome than those that contain several keywords of other articles ─ nobody would be interested in an article in a Napoleonic Wars wiki that does not contain any of the words Borodino, Beresina, Waterloo, Napoleon, Blücher and Austerlitz. Thus articles with no links to other articles could be removed and their authors publicly or secretly blamed.
- If an agent finds a similarity between two articles (which it finds only with a certain probability while scanning the Wikipedia for articles containing words that are similar to the keywords this agent has used in its own articles), then it sends a message to the author of the two similar articles to make them aware that their articles are similar. This would again be just an assertion A.
  - The article published second might be a result of plagiarism[3]. In this case the modal might be V_m (a moral valuation).
  - The message could also contain the request to remove the plagiarism, then the modal would additionally be a deontic D.
  - If the same agent has been suspect of plagiarism several times, then the message might also have the modal S (sanction), and the fallible agent might be removed from the list of those who are authorised to contribute, at least for a while.
- If an agent x finds an article b_y that is similar but shorter than the article a_x that it is about to publish, it might merge the old article b_y with its planned article a_x (see section 3).
- If an agent z finds two articles a_x and b_y belonging to the same keyword (or to similar keywords) where the similarity between the contents of the articles is low, it will communicate this finding to both agents (an assertion A) and ask (a deontic D) both agents x and y to discuss whether they could merge or purge their articles to avoid contradictory or otherwise misleading content (although in this very artificial language it will be difficult to define what "contradictory" or "misleading" means; see below for an idea how the concept of contradiction could be implemented in this language).

Other types of comments are conceivable. These few examples should suffice to discuss whether this could be a promising concept for the simulation of Wikipedia communication and co-operation. Measures for similarity can easily be found for bit strings (even in case they are of different length, in this case the comparison function must return two values, the point in the longer bit string where the substring starts that is most similar to the shorter string, and the degree of the similarity[4]).

---

3 The plagiarism idea was originally developed for the simple bit string, but in a more complex language (such as the one presented in section 3) plagiarism could be modelled as well (note that plagiarism in the real world does not originate from random effects). Perhaps one could design the model in a way that writing a new article is considerably more costly than copying from an old article.

4 The minimum Hamming distance between the shorter bit string of length ℓ_r and all the substrings of length ℓ_r of the longer substring is calculated and returned as the degree of similarity, while the number of the bit in the longer string where the best matching substring starts is returned as the starting point.

Another mode of checking for similarity and novelty at the same time is to compress each text separately and jointly and to compare the lengths of the compressed versions ─ if the

The language described above might not be sufficient for expressing comments; depending on how detailed the modal of a message is described in the M part of the message, it might be sufficient to just mention the identifier of the article in the contents part of the message, letting the recipient know that the reproach refers to behaviour with respect to this article. Further extensions of the toy version of the Wikipedia simulation will make clear what else is needed.

Another question is what might emerge from communication like this. Obviously, assertions have no direct consequences for the agents' behaviour. Deontics and validations will be processed by the norm recogniser and the norm adoption engine and be converted into a goal which is then processed by the decision maker. It is questionable whether the normative action planner is necessary at all in the Wikipedia scenario, as the action to be performed will consist of just "pressing a submit button" for the next contribution to either Wikipedia or the discussion forum. In other scenarios, the normative action planner might be necessary.[5]

We are currently developing a small number of toy scenarios (among them a simple Wikipedia simulation, see section 3, and a traffic scenario, see [8]) in order to find out what else is needed. The Hume scenario suggested by Rainer Hegselmann [7] and the multiple context scenario [4] is also considered for an implementation.

The Wikipedia prototype

In the NetLogo Wikipedia simulation[6], agents can perform different activities (see Figure 1):

- write an article (A1) and either submit it (A2) or add it to an existing article referring to the same keyword (A3),
- plagiarise, i.e. copying an existing article and publishing it for a new keyword (A4),
- search the current state of the encyclopaedia for double entries, for words that do not obey the vowel harmony or for plagiarisms (A5), and reproach the respective author or authors (A6),
- count articles that contain a word about which they wrote an article (A7),
- do nothing.

Which of these activities they select depends on the profits they individually generated when performing these activities in the past (a simple form of

length is not increased in the compression result of the joint version, then nothing new is added. This kind of algorithm is appropriate for the case of the simple language. The prototype uses a slightly simpler version: it just counts the different words that occur in the two texts separately and the different words that occur in the concatenated texts, and if the sum of the two former counts equals the latter count than the two texts have no word in common.

[5] Think of a smoker–non-smoker scenario where the decision is "smoke" which can result in a number of different action plans (leave the restaurant and smoke outside, smoke within the restaurant with or without the consent of the other guests, etc.).

[6] The NetLogo program can be downloaded from `http://www.uni-koblenz.de/~kgt/Pub/Wikipedia.nlogo`.

reinforcement learning). These profits can be selected with the help of NetLogo sliders.

An article is a string consisting of the characters "aei bklsw." (including the blank and the full stop separating words and sentences, respectively) which is introduced by a word, followed by a colon, as the keyword, and ending in a reference to both author and time, enclosed in square brackets. In longer simulations, it might be necessary to reduce the number of possible words, either by restricting the word length (increasing the probability that the next character is a blank ─ in the current version the maximum word length is five letters) or by forbidding certain co-occurrences of letters (for example by vowel harmony as in Hungarian and several other languages, such that "a" and "e" would not co-occur in a word). This, however, is currently done in an emergent manner, where commenting authors take offence at words violating the vowel harmony. Word length and/or vowel harmony can be conceived of as correlates of style (which is often an object of discussion in the real-world Wikipedia [6]). Other accidental features of the words of this language can also be interpreted for the process of commenting simulated articles, e.g. the words "wasal" and "lawas" could be defined as opposites to each other.

Writing an article starts with constructing a keyword out of the letters "aei bklsw" (including blank but not the full stop) where the probability of selecting a particular letter as the first letter in the word is equal (with the exception of the blank) whereas the probability of selecting the next letter depends on the previous letter, according to a stochastic matrix which is currently constant (but could as well change over time, according to the practice developing in the community). The blank character is selected with a certain probability, ending the construction of the word. The first word of an article is marked by a following colon as a keyword (and for some trivial technical reason it is preceded by the character ">"). The following words are constructed the same, and after each word a full stop is inserted with a certain (low) probability, such that the chain of words is separated into something like sentences. At the end of a sentence the article ends with another (low) probability, and the author adds its name (NetLogo's `who` number) and the time when it is published (NetLogo's `ticks`). When an agent has finished a word it might decide to discard it as it violates the vowel harmony (in the beginning this is not forbidden although other agents might take offence at such words).

If an agent has decided to submit its article it first has to find out whether an article referring to the same keyword already exists, if so it has to decide whether it wants to insert its text into the existing article or whether the article is just going to be published (in the beginning this is not forbidden although other agents might take offence at double entries).

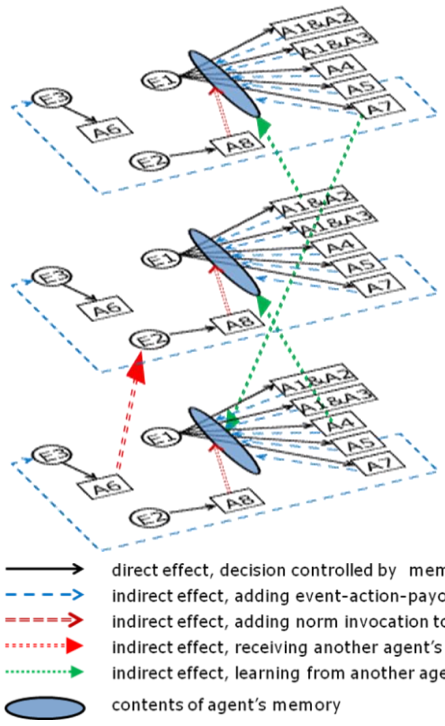
If an agent has decided to add to an existing article it first has to find out whether such an article exists, and if not publishes its text as an ordinary article.

Commenting on articles is currently implemented as a scan of all articles, searching for entries which refer to the same keyword and for entries referring to a keyword that violates vowel harmony. The scanning agent generates a list of all those keywords and identifies all the authors that wrote a second or third article referring to the same keywords as blamable (a mild deontic of the proscription type) and deletes all younger articles (a sanction in terms of [1], as the "authority" or "importance" of an author is measured in terms of the number of entries published by this author, i.e.

the number of articles published by this author is decremented ─ currently this has no consequences in the implemented model, but it could have, e.g. in a way that a sanction or validation coming from an important author is more severe, as explained above). Authors having published articles referring to keywords violating vowel harmony are also blamed, and such a keyword is replaced with a similar word obeying the vowel harmony (by exchanging an `a' with an `e' or the other way round; this replacement also affects the article list, as the entries referring to the two words have to be merged).

A third activity of agents is the search for similarities between two randomly selected articles. If an agent detects a co-occurrence of more than a certain percentage of words between two articles, it identifies the younger of the two articles as a potential plagiarism and blames its author who loses the profit generated before from intentional plagiarism. If the similarity is just by chance, the same hard punishment occurs.

More formally, perceptions, events, actions and the relations among them can be described as in Figure 1.



E1 -> [A1 & (A2|A3)] | A4 | A5 | A7
E2 -> A8
E3 -> A6

Perceptions and events

- E1: an action is profitable for this agent
- E2: this agent was blamed for an offending action
- E3: another agent's action was offending

Actions

- A1: draft an article
- A2: submit it
- A3: add it to an existing article
- A4: plagiarise
- A5: search for offending articles
- A6: reproach
- A7: count citations
- A8: add norm invocation to memory

Figure 1: Relations between events, perceptions, actions and the effects of actions

When an agent finds itself in a situation where it could contribute to the encyclopaedia it consults its memory to find how profitable the different actions available in this situation might be. The memory does not only contain information

about earlier payoffs, but also information about norm invocations from the side of other agents who might have taken offence at earlier actions of this agent (as these ─ E3→A6 ─ will have resulted in a reproach which in turn was received ─ E2→A8 ─ and stored in the agent's memory as an indirect consequence of one of its earlier actions). Beside learning from own experience about payoffs and from reproaches, agents can in principle also learn from observing other agents' actions and the resulting payoffs.

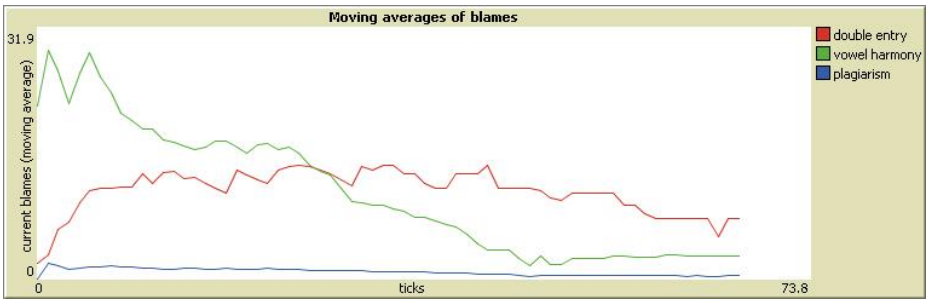


Figure 2: Moving averages of issued blames for the three offences (over 25 time steps, after approximately 70 time steps)

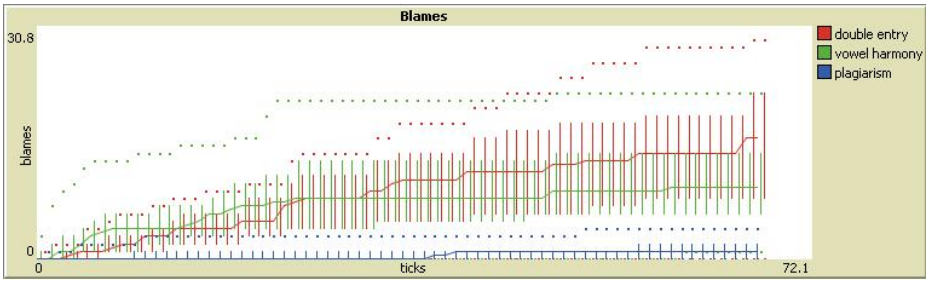


Figure 3: Distribution of individually received blames. Vertical bars are quartile differences, the upper dots are maxima, the dots below are minima, and the curves in the middle are the medians of the distribution of received blames

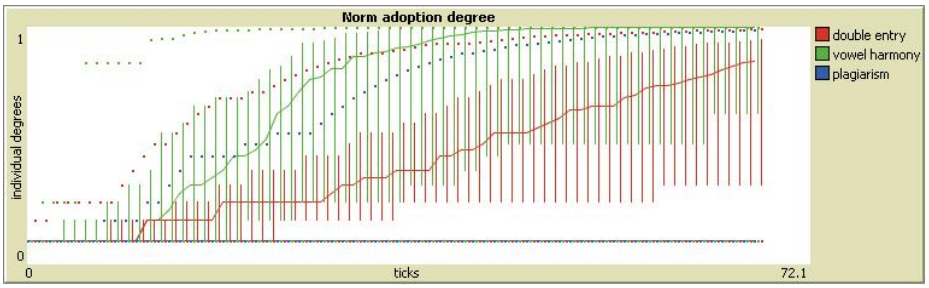


Figure 4: Distribution of the individual norm adoption degrees. Vertical bars, dots and curves have the same meaning as in Figure 3

Figures 2, 3 and 4 show three preliminary results of a long simulation run (70 ticks with 50 agents), in which the agents perform their activities depending on previous profits. The probabilities for each activity are proportional to the sum of all previously received profits (including the negative profit for detected plagiarisms). When one agent takes offence at the outcome of another agent it issues a blame. The individually received blames are shown in Figure 3, where one can see that in the beginning most blames were received for the violation of vowel harmony ─ bad style. Figure 2 shows how many blames were issued on an average during the past 25 ticks. Here one can see that vowel harmony violations occurred and were detected rather often in the early phase of the simulation, but soon decreased in number, whereas double entries occurred only after some time and plagiarisms even later; again it should be stressed that most suspected plagiarisms were unintentional as only three actually occurred on purpose ─ it must be mentioned that agents are initialised with a certain chance of abstaining from plagiarism once they had decided to commit plagiarism (to be controlled by a slider).

Figure 4 shows the process of norm emergence. The first blame an agent receives with respect to one of the possible offences (duplicated entries, vowel harmony violations, plagiarisms) puts it to its individual normative frame and makes it deliberate before it gets into danger to commit the same offence again: the probability of offending again is 90 per cent in the first deliberation, in other words: the norm adoption degree is 10 per cent after the first reproach, and later on it increases to $1 - (1 - \nu_0) \exp(-\varphi n)$ where ν_0 is the initial norm adoption degree and φ is a flexibility parameter (both are 0.1 in this simulation run) and n is the number of deliberations performed with respect to this activity.

As soon as at least one half of all agents have received at least one blame and have performed their first deliberations with respect to this kind of offence, the offence is copied to the public normative board (in the current simulation, this happened at times 13, 19 and 34 for vowel harmony, double entry and plagiarism, respectively).

It may remain an open question whether this norm emerged from the individual behaviour rules: these say that agents may blame other agents for a certain behaviour and that agents may consider such a blame and change their behaviour. Perhaps this is still a regularity, but the appearance of a norm on the normative board (although initiated by NetLogo's observer, which ─ in a way ─ counts the ballots in a referendum) could be considered as the emergence of an explicit norm. Moreover, one could very easily add the possibility that the first agent observing that blames became suddenly rarer could decree the norm (of course, this necessitates a formalisation of "suddenly" and "rarer").

Implementing a representation base and norm recogniser is not trivial in NetLogo, but the current implementation endows every agent with a normative frame, a directory with currently only up to three entries, each consisting of the name of the respective rule, the number of individually received blames for violating this rule and the number of instances when it abode by the respective rule. Thus the norm recogniser is a simple comparison between the received message and the names of the rules already stored in the individual "normative frame" (as it is called in [1]). Thus if an agent receives a blame containing the hint at double entry or at vowel harmony or at plagiarism it increases the respective number of received blames. After the first blame it takes into account both the possibility of abiding by the pre-norm or violating

it, and whenever the decision is in favour of the norm, the respective norm adoption degree (or: the salience of the normative belief) is increased.

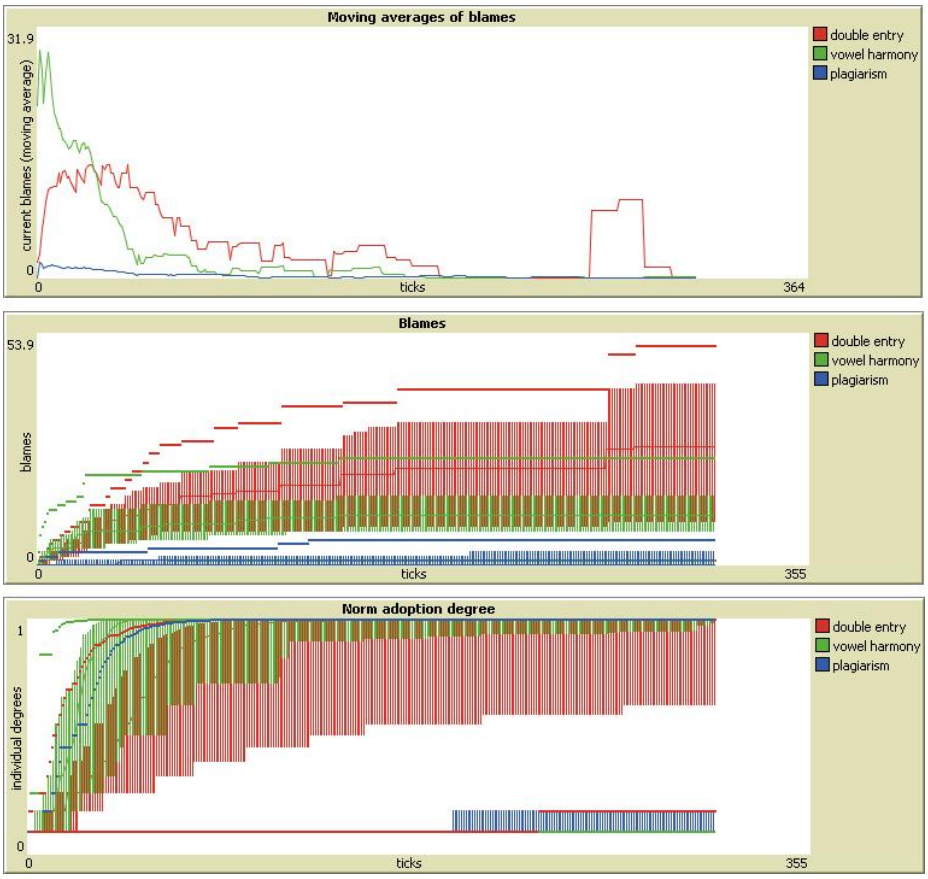


Figure 5: The same simulation run as in Figures 2 through 4, but after approximately 320 simulation runs.

The implementation of this prototype clearly suggests that for every type of message ("key") there must be a receptor in the linguistic repertoire of the authors ("lock") that responds to this message; other messages cannot be understood. Thus much of the agents' complexity lies in their linguistic repertoire. The normative frame of the individual agents is currently capable of receiving messages of any content (as the list of entries can easily be added to), but currently no agent is in a position to blame other agents except for the three implemented cases, and agents cannot even react behaviourally upon the blame for plagiarism, as they have no method for withdrawing a blamable article; not plagiarising is just a consequence of the low probability of executing the respective decision and the low profit generated from plagiarising (as the profit for an undetected plagiarism has to be reimbursed after detection).

Other types of acting, commenting and discussing can be ─ and must be ! ─ implemented, too, e.g. replacing old articles with one's own which in turn can be blamed by the author of the replaced article. These extensions can be programmed by adding to the lists of activities and of offences and by adding procedures describing the related behaviour of the agents. Due to the structure of NetLogo and its language, a more extended version of the current simulation would not be easy to understand.

Conclusion

So far, this is work in progress. But thinking about the scenario discussed in this paper leads to the following conclusions with respect to the three tasks defined in the introduction:

- building the simulator that can host the agents designed for different scenarios according to [5],
- designing the general architecture of the agents according to [1], and
- describing the scenario in a way that a modeler can endow the agents with all the features that they need in the respective scenario.

The general architecture of an agent ─ as superficially described in the introduction ─ must include the following features:

The norm recogniser: This component of an agent receives messages or makes observations of different types and can interpret them. At the start of a simulation, or for an agent entering the simulation, these messages or observations will only be of the type **(1)** "x is the case at time t_x". Even then the virgin agent will compare incoming messages to the public norm board (if there is any) and try to find out whether any of the existing entries in the norm board matches the message in question. If there is such an entry it might be of the form **(2)** "if x is the case then y is likely to happen soon after" or **(3)** "x should be avoided" or **(4)** "if x is the case y should be avoided". But perhaps at the beginning no such entries exist, but at least entries of type **(3)** should pre-exist as we will see in the next paragraph.

After several messages have been stored in the fact base of the agent, it will have to draw some conclusions from the facts, e.g. **(5)** "more often than expected by pure chance, y happened soon after x" or **(6)** "if (y should be avoided) and (if x is the case then y is likely to happen soon) then x should be avoided, too". **(5)** is a private version of **(2)** which will be stored in the private normative frame and could be communicated to another agent (as a deontic or valuation) or directly published as a "new normative belief" and stored in the norm board. **(6)** is also a new normative belief. Publishing a new normative belief will usually be postponed until the same normative belief has been uttered several times. In order that a normative belief of type **(6)** can be generated, it is necessary that at the very beginning at least one deontic such as **(3)** exists. There must be at least one goal that an agent tries to achieve, otherwise it would never be proactive. It might be possible that different agents have different goals, i.e. the only deontic with which they are born might be different between agents, and it might be interesting what happens in such a world. In a traffic scenario, for instance, some agents might have the entry **(3a)** "a collision with a pedestrian should be avoided", for others it might be **(3b)** "slowing

down should be avoided" (the latter derived from the observation that slowing down increases the risk that the car behind bumps into one's own, thus generating another kind of collision), but in this scenario all agents might have fallen victim to collision and died before any norm could emerge, but if there are enough agents (or if agents do not die due to collisions) there might be a majority of the collision avoiders, or the collision avoiders reach the threshold when the normative belief is entered into the norm board earlier than the others.

In Campennì's [4] norm recogniser, only "deontic commands and evaluations" are accessible to the norm recogniser whereas all the other messages are filtered out. But this does not seem necessary as it might be more useful that the norm recogniser also deals with assertions and behaviours (such as **(1)**) in order to be able to generate normative beliefs such as **(7)** "x should be avoided as it often leads to the unwelcome y" ─ which can easily be classified as a norm innovation.

The norm adoption engine: This component of an agent is responsible for generating goals from the contents of the normative frame. Returning to the example above, two normative goals might be produced, namely **(8)** "avoid the collision with the pedestrian in front" and **(9)** "avoid the collision with the car behind your own".

The decision maker: This component generates the normative intention from the goal or goals provided by the norm adoption engine. In different circumstances this decision can produce very different intentions, and, of course, several intentions at the same time. In the traffic example above the intention of sidestepping could be produced (driving on with constant velocity but evading the pedestrian, avoiding both the collision with the pedestrian and the collision from behind, but these details will have to be worked out by the normative action planner). On the other hand, another agent might produce the intention of warning all three other agents (pedestrian, first car, second car) by means of a police siren in order to stop all three agents (this would result in a norm defence action).

The normative action planner: This component has to work out the concrete actions that have to be taken in order to fulfill the intention produced by the decision maker (see the example in the previous paragraph).

The examples will have shown that an agent will have a rule interpreter as one main component that enables it to perform the logical operations described in the previous paragraphs. The prototypes, once accomplished, will list all the details of the rule interpreter. For now it may suffice that EMIL-S will be able to host agents that can cope with facts such as **(1)** and **(3)** and logical formulas such as **(2)** and **(4)**. But these facts and formulas still contain variables such as x and y above. And it will be the task of scenario builders to prepare a configuration (perhaps as an XML file formalising the contents of a graphical representation such as in Figure 1) for each scenario that lists the range of these variables and feeds the simulator with a few entries in the initial norm board.

References

1. Giulia Andrighetto, Marco Campenni, Rosaria Conte, and Marco Paolucci. On the immergence of norms: a normative agent architecture. In *Proceedings of AAAI Sym- posium, Social and Organizational Aspects of Intelligence*, Washington DC, 2007.
2. Giulia Andrighetto, Rosaria Conte, and Paolo Turrini. Emergence in the loop: Simulating the two way dynamics of norm innovation. In Guido Boella, Leendert van der Torre, and Harko Verhagen, editors, *Dagstuhl Seminar Proceedings 07122, Normative Multi-agent Systems, Vol. I*, 2007.
3. Giulia Andrighetto, Marco Campenni, Federico Ceccone, and Rosaria Conte. Conformity inMultiple Contexts: Imitation vs. Norm Recognition. *Paper accepted for the World Congress of Social Simulation, Fairfax VA*, July 2008,
4. Marco Campennì. The norm recogniser at work. *Presentation at AAAI'2007*, Washington.
5. EMIL. Emergence in the loop: simulating the two way dynamics of norm innovation, deliverable 3.1─ requirement analysis: Requirements that EMIL-S must meet, 2007.
6. Chris Goldspink. Normative self-regulation in the emergence of global network institutions: The Case of Wikipedia. *Presentation at the Australia and New Zealand Systems Conference 2007: Systemic development: local solutions in a global environment (ANZSYS-07).* 2007.
7. Rainer Hegselmann. Modelling Hume ─ a draft for HUME1:0. Draft as of February 18, 2008, 2008.
8. Ulf Lotzmann and Michael Möhring. A TRASS-based agent model for traffic simulation. accepted for 22nd European Conference on Modelling and Simulation ECMS 2008, 2008.
9. Thomas Malsch, Christoph Schlieder, Peter Kiefer, Maren Lübcke, Rasco Perschke, Marco Schmitt, and Klaus Stein. Communication between process and structure: Modelling and simulating message reference networks with COM/TE. *Journal of Artificial Societies and Social Simulation*, 10(1), 2007. http://jasss.soc.surrey. ac.uk/10/1/9.html.
10. Klaus G. Troitzsch. Multi-agent systems and simulation: a survey from an application perspective. In Adelinde Uhrmacher and Danny Weyns, editors, *Agents, Simulation and Applications*, pages 3.1{3.23. Taylor and Francis, London, 2008. to appear.