Trust mechanisms for efficiency improvement in collaborative working environments

Xingyu Zheng, Patrick Maillé, Cam Tu Phan Le, Stéphane Morucci

Trust mechanisms for efficiency improvement in collaborative working environments

Xingyu Zheng, Patrick Maillé
Institut Telecom; Telecom Bretagne
2, rue de la Châtaigneraie CS 17607
35576 Cesson Sévigné Cedex, France
Email: {xingyu.zheng},{patrick.maille}@telecom-bretagne.eu

Cam Tu Phan Le, Stéphane Morucci
Swid
80, avenue des Buttes de Coesmes
35700 Rennes, France
Email: {stephane.morucci},{plcamtu}@swid.fr

***Abstract*—We apply the notion of trust to situations of document editing, to select the successive editors of the document and avoid unnecessary readings by all collaborators after each modification. Two mechanisms using trust to improve that process are proposed, and compared to the situation without trust. Simulation results suggest that trust can improve the efficiency of the collaborative work.**

***Index Terms*—Collaborative work; trust; performance;**

I. Introduction

Many types of documents cannot be entirely produced by a single individual, and instead need to be jointly built by several collaborating participants. One can think of examples from diverse working sectors: law texts, research articles, project proposals, course textbooks, or (online) encyclopedias. In all those cases, collaborative working is needed because of the prohibitive size of the document to be produced, and/or because of the diversity of the knowledge fields and competences involved.

Several tools to facilitate collaborative work exist, such as online wikis or versioning softwares, which have been shown to help improve the team coordination and results, as well as the participant satisfaction [1].

In this paper, we propose some trust-based mechanisms aimed at optimizing the process of collaborative document building, when all participants have the same objective, that can be summarized as "writing a high quality document". Several kinds of trust usages have been proposed, mainly for grid computing environments and peer-to-peer systems. Indeed, in those cases where each node shares its resources and benefits from those offered by its peers, trust mechanisms can be used to find the best peers to exchange with, or to avoid malicious nodes [2], [3], [4]. Remark moreover that inter-company (or even inter-personal) collaborations often rely on the (most often implicit) notion of trust, that can also be modeled and somehow automated in order to make the right business decisions [5].

Nevertheless, the use of trust to improve the performance of collaborative work has not been investigated much. Our objective here is to use trust to limit the total time and effort spent to reach the common objective of the group. More precisely, we intend to use trust to avoid unnecessary readings, and efficiently select editors during the development process.

II. Using trust in a collaborative work environment

A. Definition of trust

Trust can be defined as a quantification of the confidence that an agent has in another agent to behave in an appropriate manner. In most trust-based models (see [6], [7], [8]), that quantification is summarized by a number in the interval $[0, 1]$. We denote by $\mathcal{I}$ the set of agents, and for $i, j \in \mathcal{I}$, by $t_{i,j}$ the trust value representing how much agent i trusts agent j. A trust value $t_{i,j}$ of 1 means that agent i perfectly trusts agent j (in particular, $t_{i,i}$ is fixed to 1), while $t_{i,j} = 0$ means that agent i does not trust j at all.

In this paper, we do not consider the problem of trust estimation; we assume that trust scores $(t_{i,j})_{(i,j)\in\mathcal{I}^2}$ have already been determined, and we focus on the use of those scores in collaborative work environments.

B. Agent satisfaction scores during the collaborative work

We consider a working system with a number $I = |\mathcal{I}|$ of agents who collaboratively contribute to the elaboration of a document. Those agents successively ameliorate the document in an asynchronous manner, until it reaches a satisfying quality.

In our model, each editor $k \in \mathcal{I}$ has to associate to the document an evaluation score $e_k \in [0, 1]$ after her editing the document. That score represents the quality that editor k thinks the document has. We do not consider maliciousness from participants, who are assumed to behave honestly and to truthfully declare their perceived value e_k.

The evaluation score is meant to be used by all the other collaborators, who combine it with their trust in the last editor of the document to compute a personal *satisfaction score* for the document. Formally, if $\mathrm{Id}_{\mathrm{last}}$ is the identity of the last document editor, each agent $i \in \mathcal{I}$ automatically computes (without even opening the document) the satisfaction score

$$s_i := \min\left[(t_{i,\mathrm{Id}_{\mathrm{last}}} + \alpha \mathrm{n}_{\mathrm{edit}}(i)) \times e_{\mathrm{Id}_{\mathrm{last}}} \ , \ 1\right], \tag{1}$$

where

- $\mathrm{n}_{\mathrm{edit}}(i)$ is the number of editions by agent i in the process,
- α is the bias in the satisfaction calculation, that corresponds to each extra edition of the document.

The rationale behind that expression is as follows: participants take the declared score $e_{\text{Id}_{\text{last}}}$ into account, but are less optimistic in following the opinion of Id_{last} when they do not trust that agent. In that sense, the multiplicative combination rule can be interpreted as providing a lower bound of the quality of the document: the document can be of high quality, but before agent i verifies that through a careful reading, she remains cautious and lowers the declared evaluation score, at a greater scale if she does not trust the last editor. Moreover, we can reasonably assume that the more an agent has gone through the document, the more confident she is in its quality. As a result, we consider that the satisfaction value of an agent increases with the number of times she has edited the document. We upper bound the result by 1, so that satisfaction scores are always in the interval $[0, 1]$.

III. Management of collaborative working through trust

In this section, we introduce three different ways of managing the collaborative working environment. By management, we mean here the process of deciding which agent will be the next to edit the document.

A. Collaborative work model

We consider that the "edition token" circulates among agents, i.e., they successively edit the document according to a given management scheme. At each document modification, the last editor Id_{last} attaches to the document her evaluation score $e_{\text{Id}_{\text{last}}}$, and all collaborators compute their corresponding satisfaction value according to (1).

We assume that each agent $i \in \mathcal{I}$ has a threshold $b_i \in [0, 1)$ that she considers to be the minimal satisfaction level for the document to be acceptable. Following the idea of collaborative work as a way to reach a consensus among participating agents, we consider that the document editing process stops when all agents are satisfied with the quality of the document, i.e., when $s_i \geq b_i$, $\forall i \in \mathcal{I}$. Until that condition is satisfied, we assume that users go on successively editing the document.

From (1), the development process stops with probability 1 if the management scheme gives each unsatisfied agent a non-negative probability of editing the document.

We now describe the three process management algorithms that are studied in this paper, assuming that time is slotted.

B. A naïve scheme: pure round-robin document edition

The first mechanism that we consider does not actually use trust values to manage the document development process. Only satisfaction scores are used to decide when to stop the process, but the order in which participants edit the document is given as fixed. More precisely, we assume in that case that the "edition token" circulates in a round-robin fashion: if agents are numbered $\{1, 2, ..., I\}$ (up to a permutation), then the identity of the editor just follows a simple rotation pattern $1, 2, ..., I, 1, 2, ...$, until all participants are satisfied. If the editor who is supposed to edit the document is not available, then the token is immediately given to the next one and no time slot is lost.

C. Using trust to choose the next editor: round-robin among unsatisfied agents

We suggest here to use the satisfaction scores computed during the document development process, to improve the efficiency of that process. To that end, we consider some strategies that have originally been proposed for peer selection in peer-to-peer networks, and we apply them here to select the next editor. In this subsection, we adapt the strategy consisting in selecting, as the peer to download files from, one peer among all sufficiently trusted peers.

In our collaborative work context, this can be translated into the following decision process: the next editor of the document is still chosen according to a round-robin scheme, but only among agents i for which the current value of the satisfaction score is below their quality threshold b_i. In other words, we simply follow the scheme of the previous subsection, but skipping participants who are already satisfied with the current quality of the document. If the next unsatisfied agent is unavailable, then the token goes to the next unsatisfied agent in the predefined order. If all unsatisfied agents are unavailable, then the time slot is lost (nobody edits the document).

D. Having the least satisfied agent improve the document

We now consider the strategy from peer-to-peer networks, where peers choose to download their requested files from the most trusted host [2]. We adapt that policy to the context of collaborative work, by giving the "edition token" to an available collaborative agent k for whom the satisfaction score is the furthest below her threshold, i.e., the next editor is the (an) agent $k \in \arg\min_{i \in \mathcal{I}, i \text{ available}}(s_i - b_i)$. As for the previous scheme, the time slot is lost when all unsatisfied agents are unavailable.

Intuitively, that third scheme should be more efficient than the one defined in Subsection III-C, since we choose to specifically target the minimal *satisfaction minus threshold* value, and the process ending precisely depends on that value (it stops when $\min_{i \in \mathcal{I}} s_i - b_i \geq 0$).

On the other hand, the potential extra efficiency of that scheme will have a price in terms of applicability, since implementing it in a fully decentralized way becomes difficult. Therefore, only if the efficiency gain of the development process is significative should we be inclined to prefer that scheme over the simpler round-robin among unsatisfied clients.

IV. Experiments

This section presents some simulation results aimed at evaluating the performance of the three management methods of Subsections III-B, III-C, and III-D.

A. Simulation model

Trust values $(t_{i,j})_{i,j \in \mathcal{I}}$ are chosen randomly and independently, according to a uniform distribution on the interval $[0.5, 1]$. The parameter α equals 0.1 in our simulations, and the quality satisfaction threshold b_i of each participant i is fixed to 0.9. To model users being available or not over time, we consider that at each time slot, each participant is available

with a given probability, independently of all other events. We fix that availability probability to 0.5 in our simulations.

We introduce some asymmetry among participants by associating to each one $i \in \mathcal{I}$ a *performance level* $q_i \in [0,1]$, that can represent the "writing quality" of participant i, and that is chosen randomly for each participant $i \in \mathcal{I}$, according to a uniform law on $[0.5, 1]$. We assume that each participant improves the quality of the document -at least from her point of view- when editing it. More precisely, in our simulations the evaluation score e_k set by editor k is computed according to the formula

$$e_k := \; s_{k,\text{prev}} + (1 - s_{k,\text{prev}}) \times q_k, \tag{2}$$

where $s_{k,\text{prev}}$ is the satisfaction value that k had just before editing the document.

The efficiency of the document development process is measured by two performance metrics: the *total duration* of the document development process (i.e., the number of time slots spent until a satisfying document quality is reached), and the *total effort* spent by the collaborating community during the whole process. That last value is estimated by considering the quality parameter q_k of the previous subsection, that we consider to also represent the effort spent by editor k each time she edits the document. The total effort is then the sum over all edition slots of that value q_k.

B. Performance evaluation

Figure 1 shows the ratio of our metrics of interest (i.e., the number of iterations and the total effort in the development process) for our two trust-based schemes, over the corresponding metrics for the round-robin mechanism.

When focusing on the time metric, it unexpectedly appears that the naïve round-robin mechanism is the one that performs fastest when the number of collaborators is below 10. This is due to the fact that collaborators are sometimes unavailable, and time slots are lost. Such situations are less frequent when the number of collaborators becomes large, and trust-based mechanisms outperform the round-robin scheme by targeting only unsatisfied agents. The gain in terms of time is for example of 20% for collaborative groups of $I = 50$ participants. In terms of total effort, our two trust-based management schemes always yield an improvement with respect to the naïve round-robin mechanism: the total effort is reduced by about 20% for $I = 50$ collaborators, but the improvement is of less than 10% if there are less than 10 collaborators.

Those results suggest that with few collaborators, introducing trust-based management will not improve the speed of the development process, and the improvement in terms of effort will be quite small. Therefore, in those cases it might not be interesting to use trust. On the other hand, when the number of collaborators becomes large, then applying one of our trust-based schemes yields a non-negligible improvement, both in terms of time and of effort spent. For both performance measures, the difference between the two trust-based schemes is always at the advantage of the one targeting the least satisfied agent, but remains small when compared to the improvement brought by trust. This suggests that the round-robin among unsatisfied agents should be preferred due to the implementation and privacy drawbacks of the scheme involving the identification of least-satisfied agents.

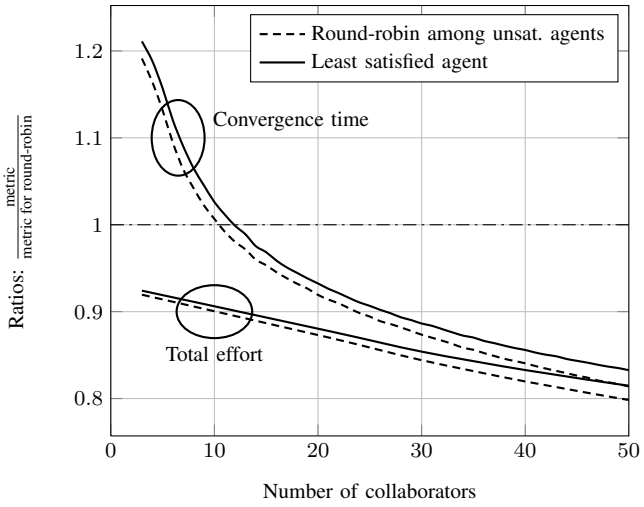

Figure 1. Time and Effort gain due to trust-based management.

V. Future work

The main direction we would like to investigate consists in modelling objective misalignments among participants, and designing a trust-based scheme robust to agent selfishness. To ensure that property, the system should be studied as a noncooperative game [9], and the corresponding agent equilibrium strategies should lead to a globally efficient outcome.

Acknowledgments

This work has been partially funded by the French *Agence Nationale pour la Recherche* through the FLUOR project.

References

[1] P. B. Lowry, J. F. Nunamaker Jr, A. Curtis, and M. R. Lowry, "The impact of process structure on novice, virtual collaborative writing teams," *IEEE Transactions on Professional Communication*, vol. 48, no. 4, p. 341, Dec 2005.

[2] X. Ding, W. Yu, and Y. Pan, "A dynamic trust management scheme to mitigate malware proliferation in P2P networks," in *Proc. of IEEE ICC*, Beijing, PR China, 2008.

[3] Y. Wang and J. Vassileva, "A review on trust and reputation for Web service selection," in *Proc. of 27th International Conference on Distributed Computing Systems Workshops*, Toronto, Canada, Jun 2007.

[4] Q. Zhang, Y. Zhuo, and Z. Gong, "A trust inspection model based on society behavior similarity rule in dynamic networks," in *Proc. of International Conference on Computer Science and Software Engineering*, Wuhan, PR China, Dec 2008, pp. 970–973.

[5] S. Ruohomaa, "Trust management for inter-enterprise collaborations," *Web proceedings of the I-ESA*, vol. 7, 2007.

[6] J. Golbeck and J. Hendler, "Inferring binary trust relationships in web-based social networks," *ACM Transactions on Internet Technology*, vol. 6, no. 4, pp. 497–529, 2006.

[7] S. Marti, "Trust and reputation in peer-to-peer networks," Ph.D. dissertation, Stanford University, May 2005.

[8] G. Suryanarayana and R. N. Taylor, "A survey of trust management and resource discovery technologies in peer-to-peer applications," University of California, Irvine, Tech. Rep. UCI-ISR-04-6, Jul 2004.

[9] M. J. Osborne and A. Rubinstein, *A Course in Game Theory*. MIT Press, 1994.