Trust, Negotiations and Virtual Currencies for a Sharing Economy

Dave de Jonge and Carles Sierra

IIIA-CSIC, Bellaterra, Catalonia, Spain
{davedejonge, sierra}@iiia.csic.es

Abstract. We propose an application that allows users to request other users for help with every-day tasks. Users can pay each other for these tasks by issuing contracts in which the requester promises to return the favor in the future by performing some task for the other. Such contracts can be seen as an alternative currency, coined by the users themselves. Trust is an essential aspect of this system, as the issuer of a contract may fail to fulfill its commitments. Therefore, the application comes with a social network where users can leave comments about other users. Furthermore, our application includes a market place where users can exchange service contracts between each other, and a negotiation algorithm that can automatically trade these contracts on behalf of the user.

Keywords: Automated Negotiation, Trust, Sharing Economy, Virtual Currency, Social Networks

1 Introduction

Around the world traditional economies are converting more and more into what is known as a *sharing economy*: an economical system in which individuals share goods and services with each other, rather than buying them from official companies. Well-known examples of applications and communities based around sharing are Couch Surfing,[1] Airbnb[2] and BlaBlaCar.[3] Another important change in modern economy is the introduction of *virtual currencies*; currencies that are not regulated by governments or banks, but by informal communities and that only exist in digital form. The best known example of such currency is Bitcoin.[4]

In this paper we propose an extension of the u-Help application introduced in [2]. U-Help is a distributed community-based application that allows users to request help with every-day tasks from other community members. A user may for example request a babysitter for her child. This request is then propagated along the network so that any other user in the community may accept the request and voluntarily carry out the requested task. The original u-Help application however relies entirely on the willingness of people to help voluntarily.

---

[1] http://www.couchsurfing.org
[2] http://www.airbnb.com
[3] http://www.blablacar.com/
[4] https://bitcoin.org/

This may work in small communities where most people know each other, but it is less likely to succeed when used in larger communities with more anonymity because it would be tempting to act as a freerider. In this paper we therefore propose to extend u-Help with a virtual currency system that enables people to pay for the help they receive.

2 Alternative Currency

In our extension, when a user receives a request he or she may not only accept or reject that request, but may also ask for a favor in return. The requesting user can then pay the accepter by issuing a *service contract*, in which he promises to return the favor by helping the accepter with some task in the future.

Example: Bob asks Alice to pick up his child from school. In return Bob gives Alice a service contract in which he promises Alice to help her repair her computer, some time in the future whenever she needs it.

The users can negotiate on the terms of the contract, similar to a mechanism introduced in [3] in which agents may promise future rewards when negotiating a deal in the present. In the example above we say that Bob is the *issuer* of the contract. Such a service contract can then be used by Alice in two ways: she can *exercise* the contract, meaning that Alice requests Bob to fulfill his commitment and fix her computer, or she can *pass on* the contract to somebody else to make a payment, as if it were a sort of currency.

Example: Alice asks Charles to repair her car. As a payment Alice gives Charles the contract she earlier received from Bob. So whenever Charles' computer is broken, he can go to Bob and exercise the contract.

Note that this system is very similar to the original use of bank notes: they served as a proof that a bank owed a certain amount of gold to the holder of the bank note. In our case a service contract is a proof that another user still owes a certain favor to the holder of the contract.

We make a distinction between a *task* and a *service contract*. A task explicitly defines an activity that is going to take place at a specified time and place as a favor for a specific person. A service contract also specifies an activity but does not specify when, and does not specify who is the beneficiary. Exercising a service contract means to convert it into a task.

A service contract specifies:

- The type of activity.
- The person who is going to do the activity (the issuer of the contract).
- The duration of the activity.

A task specifies:

- The type of activity.
- The person who is going to do the activity (the issuer of the contract).
- The duration of the activity.
- The date, time and location of the activity.
- The beneficiary.

Of course it can always happen that the holder of a contract wants to exercise it on a moment inauspicious to the issuer. Therefore, the contract may include terms on how and when it can be exercised. For example: "Bob will fix your computer, but you have to request him to do so at least 2 days in advance".

3 Trust

The notion of trust is essential in this system. After all, Charles and Bob may not even know each other, so Charles does not know whether Bob is really handy enough with computers to solve his problem. If Charles does not trust Bob he will not accept any contract issued by Bob as a payment. Therefore, we propose a reputation system that helps users to decide whether to accept service contracts issued by unknown people. This reputation system consists of a database that stores for every user:

- His or her skills.
- Reviews of this user written by other users and their satisfaction levels for previous tasks executed by this user.
- A social graph, showing whether two users have common friends.
- His or her number of "outstanding" contracts (contracts that he or she has issued but that have not been exercised yet).

Every user has a profile where this information can be found. Other users can leave comments on this profile which cannot be removed. Of course, one should always consider the possibility that the issuer of a contract will not fulfill his commitment. In that case one can leave a negative comment on the issuer's profile, which will reflect badly on his reputation. This will strongly decrease his chances of receiving help from others in the future as they will consider contracts issued by him less valuable. Therefore, users have an incentive not to issue more contracts than they are actually able to fulfill. Again, this resembles the way that real monetary systems work: people accept payments in dollars because they trust that the Federal Reserve will not print dollars excessively and cause hyperinflation. In our proposed system every single user essentially acts as a central bank for his or her own currency.

4 Negotiations

When Alice requests Bob to execute some task for her, they need to agree on how Alice is going to repay this favor. These negotiations can become difficult if, for example, Alice does not have anything to offer in return that Bob is interested in. To solve this problem we propose an online *market place* where users can exchange service contracts. This market place has several advantages:

1. It enlarges the set of possible deals that can be made between users.
2. It enables users to exchange service contracts they do not need for contracts they do need.
3. It enables users to "buy back" contracts issued by themselves.

An example of the first case would be when Alice finds a contract on the market place that Bob is interested in. She can then agree with Bob to "buy" that contract in exchange for a contract issued by herself, so that she can give that contract to Bob, and Bob can help Alice with her computer. The second case applies when Bob has accepted a contract issued by Alice, but is not interested in exercising it. He can then trade that contract on the market place for another one that he is interested in. The third case applies if Alice has issued many contracts which have not yet been exercised, while she herself owns many contracts issued by others. If she is afraid she will not be able to fulfill all the contracts she issued she can take them off the market by trading them for the contracts she owns.

If there are thousands of users looking for contracts or offering contracts in the market place at the same time it may be very difficult to find the right deal. Therefore, we propose to incorporate an automated negotiating agent into the application based on the algorithm introduced in [1]. The user can then tell his agent the kinds of contracts he is looking for, the kinds of contracts he is willing to issue, and the contracts he currently owns (issued by others). The agent will then contact the market place to negotiate a deal with the agents of other users. The agent may propose deals to the other agents and may accept deals proposed by them. It determines the value of any such deal based on the terms of the contracts involved in it as well as on the reputations of the issuers of those contracts. Once several agents agree on a deal the users themselves will also be asked if they agree with that deal. If not, they can still reject it.

5 Acknowledgments

References

1. Dave de Jonge and Carles Sierra. NB3: a multilateral negotiation algorithm for large, non-linear agreement spaces with limited time. *Autonomous Agents and Multi-Agent Systems*, 29(5):896–942, 2015.
2. Andrew Koster, Jordi Madrenas, Nardine Osman, Marco Schorlemmer, Jordi Sabater-Mir, Carles Sierra, Dave de Jonge, Angela Fabregues, Josep Puyol-Gruart, and Pere García. u-Help: supporting helpful communities with information technology. In *Proceedings of the First International Conference on Agreement Technologies (AT 2012)*, volume 918, pages 378–392, Dubrovnik, Croatia, 15/10/2012 2012.
3. Sarvapali D. Ramchurn, Carles Sierra, Luís Godo, and Nicholas R. Jennings. Negotiating using rewards. *Artificial Intelligence*, 171:805–837, 2007.