Suggesting New Interactions Related to Events in a Social Network for Elderly

Holger Jost
Institute of Informatics
University of Potsdam
Potsdam, Germany
jost@cs.uni-potsdam.de

Orkunt Sabuncu
Institute of Informatics
University of Potsdam
Potsdam, Germany
orkunt@cs.uni-potsdam.de

Torsten Schaub
Institute of Informatics
University of Potsdam
Potsdam, Germany
torsten@cs.uni-potsdam.de

Improving the daily life of an elderly is one of the main objectives of Assisted Living initiative. The EasyReach project aims to achieve this by utilizing a social network for elderly and an intuitive user interface. The personal assistant in EasyReach monitors interactions of the user within the social network and suggests new interactions to keep the user socially active. We model suggesting new interactions related to events via Answer Set Programming. The assistant suggests communicating with someone from the social network related to an event that the user wants to attend or has attended.

Assisted living, Social network for elderly, Personal assistant, Answer set programming

1. INTRODUCTION

Elderly and less educated people can easily become isolated from modern social life. The EasyReach project is an innovative ICT solution to allow elderly and less educated people to participate in the benefits of IT-based social interactions. It is supported by the Ambient Assistant Living Joint Programme (2010-2013). The user interacts with the system using an inertial remote control unit commanding a set-top-box connected to a TV. The set-top-box can also connect to the Internet. A crucial component of the system is the social network of EasyReach users. In accordance with objectives of Assisted Living initiatives, the system aims to improve the daily life of a user by keeping him socially active with the utilization of the social network. To this end, there is a personal assistant software component which pro-actively suggests new interactions through the social network. The assistant monitors the activities and interactions of the user within the social network and reasons on them to suggest new interactions.

It is critical to use TV as a medium for interacting with the system since unlike with computers, elderly and less educated people generally have experience with TV. The lack of necessary technological skills also renders the use of general social networks (e.g., Facebook) impossible for target users. The specialized social network and the personal assistant in EasyReach take such constraints of elderly into consideration.

Giving information about events (e.g., birthday parties, group meetings, etc.), organizing them, and inviting people to them are frequent uses of social networks (e.g., Facebook, Linkedin, or Myspace events). Users can express their informal approval for attending an event published in the social network or express that they do not attend the event. We believe that events can also play an important role for a social network for elderly or less educated people. They might be useful in avoiding elderly people to become isolated from social life. Moreover, initial pilot studies carried out within the EasyReach project showed that end-users are willing to organize meetings and entertainment activities.

In this work, we present a use case where the personal assistant of EasyReach proactively suggests the user communicating with someone from the social network about an event that the user wants to attend. This may be beneficial to the user in several ways. The user can get more information about the event by communicating with the suggested person, invite him or her, or even ask help to assist himself for attending the event. The use case also applies after the user has attended an event. This time he can discuss outcomes of the past event or ask for some information about it or its topic.

For our use case let Bob be an EasyReach user. He checks the channel of the bingo group. He finds out that there is a bingo gathering event this weekend. He wants to attend this event and expresses his informal

approval using the user interface. The personal assistant of Bob proactively asks Bob whether he wants to contact Alice using the EasyReach system to talk about the mentioned event or not. She is chosen because she is a friend of Bob and she has already been to similar type of events in past; for instance, a backgammon gathering event.

We implemented a framework for the personal assistant to carry out such kind of uses cases. Answer Set Programming (ASP; Baral (2003)), a popular declarative problem solving approach in the field of knowledge representation and reasoning, is utilized to implement the reasoning capabilities of the personal assistant so that it can suggest new interactions related to events.

The paper is organized as follows. The next section gives a brief introduction to ASP. Section 3 describes our modeling of how the personal assistant can reason on suggesting new interactions related to events in a social network. We conclude and discuss future lines of research in Section 4.

2. ANSWER SET PROGRAMMING

ASP emerged in the late 1990s as a new logic programming paradigm Gelfond and Lifschitz (1991). It has become a popular declarative problem solving approach in the field of knowledge representation and reasoning. This is mainly due to its appealing combination of a rich yet simple modeling language with high-performance solving capacities. Even though, syntactically, ASP programs resemble Prolog programs, they are treated by rather different computational mechanisms. Instead of relying on resolution based query evaluation, model generation is applied. The basic idea of ASP is to represent a given computational problem by a logic program whose answer sets correspond to solutions, and then to use an answer set solver for finding answer sets of the program. ASP has been used in many application areas. Among them are product configuration Soininen and Niemelä (1999), decision support for NASA shuttle controllers Nogueira et al. (2001), reasoning tools in systems biology Gebser et al. (2011b), and many more.

One typically represents the problem by a logic program with variables. These variables should be systematically replaced by variable-free terms since answer set solvers work with propositional programs. In this work we use the answer set solver from the Potassco answer set solving collection Gebser et al. (2011a). Due to space constraints we refer the reader Simons et al. (2002) for the syntax and semantics of the ASP language.

3. MODELING OF SUGGESTING NEW INTERACTIONS VIA ANSWER SET PROGRAMMING

In this section, we describe how to encode in ASP the situation regarding events published on the social network of EasyReach so that the personal assistant can suggest new interactions accordingly. The use case mentioned in Section 1 will be the running example throughout the section.

We start modeling by encoding people, a friendship graph, and events by some facts. Users of the social network are represented by instances of the predicates $person$ and $person\_name$. For instance, the following facts represent that Bob and Alice's ids are 100 and 101, respectively.

$$person(100). \quad person\_name(100, \text{“}Bob\text{”}).$$
$$person(101). \quad person\_name(101, \text{“}Alice\text{”}).$$

The friendship relation among people is represented by the predicate $friend(id_1, id_2)$. Its intuitive reading is id_2 is a friend of id_1. Instances of the $friend$ predicate model the directed friendship graph. The instances $friend(100, 101)$ and $friend(101, 100)$ represent that Bob and Alice are friends. Similarly, we encode information related to events by facts of several predicates. The following facts are related to the bingo and backgammon gathering events.

$$event(80). \quad event\_name(80, \text{“}Bingo\ gathering\text{”}).$$
$$event(81). \quad event\_name(81, \text{“}Backgammon\ Sunday\text{”}).$$

$$event\_type(80, fun). \quad event\_date(80, 20120425).$$
$$event\_type(81, fun). \quad event\_date(81, 20120320).$$

The predicates $event$, $event\_name$, $event\_type$, and $event\_date$ represent a event's id, name, type, start date respectively. The attendance to an event is represented by instances of the predicate $attends$. For instance, $attends(100, 80)$ states that Bob wants to attend the bingo gathering event. Moreover, the personal assistant adds the facts related the query. In our use case, these are the facts $user(100)$ and $about(80)$ since we are considering the personal assistant of Bob and he has recently expressed his will to attend the bingo gathering event.

There can be several reasons for the personal assistant to suggest communicating with someone concerning an event. The suggested person might be a friend of the user. He or she might have attended another event which is of the same type as the event the user wants to attend. Additionally, a combination of reasons can be valid when suggesting someone for communicating with. In this sense, reasons invoke a preference relation among people in the social network. Considering two friends of a user one of whom attended an earlier event of the same type, it is logical for the personal assistant to suggest

communicating with the one who attended the earlier event. Thus, we encode suggestions as a quantitative optimization problem where reasons have weights and the person with the maximum score is selected. A person's score is the summation of weights of the reasons satisfied. In our current prototype the following reasons and associated weights are used. Weights of reasons may change from person to person and they can be set in the user profile when one registers in EasyReach for the first time. Note that negative weights are allowed.

$$weight(friendship, 8).\ \ weight(stranger, -4).$$
$$weight(attendedSameTypeEvent, 4).$$
$$weight(attendedTogetherEarlier, 10).$$
$$weight(attendedTheEvent, 13).$$
$$weight(wantsToAttendAlso, 11).$$

Some of the reasons are related to past event. For instance, the reason $attendedTogetherEarlier$ is satisfied when the user and the person with whom is suggested communicating attended an another event earlier. Hence, the following rule defines the $attended$ predicate by comparing the current date and the events date.

$$attended(P, E) \leftarrow attends(P, E),\ event\_date(E, SD),\ today(T),\ T > SD.$$

When the person P has expressed that he wants to attend event E in the social network (i.e., $attends(P, E)$) and the starting date of the event has passed (i.e., $T > SD$), we can reason that P has attended E by default.[1]

Now we can easily model the reasons in ASP. The intuitive reading for the $possSuggestion(P, R)$ predicate is that it is possible to suggest to the user person P for communicating about the event mentioned in the query because of reason R. The following rules encode $attendedSameTypeEvent$, $attendedTogetherEarlier$, and $wantsToAttendAlso$ reasons.

$$possSuggestion(P, attendedSameTypeEvent) \leftarrow user(U),\ about(E),\ person(P),\ P \neq U,\ event\_type(E, T),\ event\_type(E', T),\ E \neq E',\ attended(P, E').$$

$$possSuggestion(P, attendedTogetherEarlier) \leftarrow user(U),\ about(E),\ person(P),\ P \neq U,\ attended(U, E'),\ attended(P, E'),\ E \neq E'.$$

$$possSuggestion(P, wantsToAttendAlso) \leftarrow user(U),\ about(E),\ person(P),\ P \neq U,\ attends(P, E).$$

The body of the first rule checks that P satisfies the reason $attendedSameTypeEvent$. It states that there should be a different event E' which is of the same type as the event mentioned in the query ($event\_type(E, T), event\_type(E', T), E \neq E'$) and P must have attended this event ($attended(P, E')$). The other rules encode respective reasons in a similar and straightforward way.

In order to compute the score for a person we need to add the weights of reasons satisfied. The following rule does this summation by using the sum aggregate supported by the ASP language. For each person P an instance $suggest(P, S)$ is generated in the answer set giving S as his or her score. Currently we do not compute only one person with the maximum score in the encoding. This makes sense from the system point of view since the EasyReach system can sort and select a predefined number of suggestions according to their scores. Then, in case the user rejects the first suggestion and wants to see the others this selected list can be used.

$$suggest(P, S) \leftarrow\ user(U),\ person(P),\ P \neq U,\ S := \#sum[possSuggestion(P, R) : weight(R, W) = W].$$

Now, we have the whole encoding about suggesting new communication interactions about events. The main architecture of the personal assistant is responsible for running the ASP solver[2] on this encoding with additional instance data read from the social network (friendship graph, general and attendance information about events) and the bingo gathering event which he wants to attend. The resulting answer set contains literals $possSuggestion(101, friendship)$ and $possSuggestion(101, attendedSameTypeEvent)$ stating that Alice can be suggested for contacting since she is a friend of Bob and she has attended backgammon Sunday event which is of the same type as the bingo gathering event. Additionally, in the answer set $suggest(101, 12)$ is true since Alice's score is 12 (8+4).

It is easy to see that the modeling described in this section applies successfully to reasoning on suggestions regarding past events also. The personal assistant can trigger this reasoning module when a predefined amount of time passed after the starting date of the event (e.g., the next day after the event). In this case the suggestion of communicating with someone through the EasyReach network can foster social interactions in a way that the user can talk about the past event, give or ask some information about it. In our preliminary experiments, we found out intuitive cases which can be beneficial for the

[1] Note that in ASP one can neatly represent the abnormality for this kind of default reasoning by using negation as failure. So, the encoding can utilize cases like the user has not attended an event even if he has expressed his will to attend. In our current prototype we have not implemented such cases yet.

[2] This can be achieved by calling a subroutine triggered on the insertion of the fact that the user wants to attend an event to the social network. In our use case, it corresponds to the fact Bob expresses his approval for attending the bingo gathering event.

elderly to be socially active and to avoid becoming isolated from life. For instance, consider that Priscilla has attended a meeting of her local church published in EasyReach. Her personal assistant can suggest her communicating with Alice, who is not a friend of Priscilla, about this meeting. Although Alice is not a friend, she was chosen because she has also attended the same meeting (the $attendedTheEvent$ reason). This suggestion may foster new social interactions. In another case, Jeff has attended a classical music concert. Among all of his friends, Bob is suggested by the personal assistant since Jeff and Bob attended another event earlier ($friendship$ and $attendedTogetherEarlier$ reasons). This suggestion might motivate Jeff in telling how the concert was to his friend Bob.

4. DISCUSSION

In this work, a modeling of suggesting new interactions in a social network for elderly is presented. This modeling is achieved via an ASP program. With the addition of the instance data read from the social network, the personal assistant can compute an answer set that contains people with scores to suggest communicating with related to an event. This event can be in future which the user wants to attend or in past which he attended earlier.

We experimented some use cases using small friendship graphs and hand-generated events which simulate the social network. EasyReach is an ongoing project and its user interaction and social network components are currently being developed. Additionally, real world pilots and evaluations with people in several senior residences have been planned. These pilots will help us to improve the personal assistant in suggesting new interactions. To this end, the real world usage data may point out what kinds of suggestions lead to an improvement in the social life of an elderly.

The related work Gulbrendsen et al. (2012) also forms a specialized social network for elderly. There is a recommender module that suggests cultural and social events to attend and people to attend with. The events, however, are not handled explicitly within the social network, which makes it hard to utilize reasons like a user has already attended same kind of event or both users have attended a past event. Additionally, suggesting a user communicate with a person about a past event is not possible since events only in future are featured.

Currently, a small set of keywords is used to specify types of events. An interesting future line of research is to incorporate more general and complex taxonomies or ontologies about events into reasoning. For instance, Gulbrendsen et al. (2012) uses a taxonomy to model event types. The subsumption relation among keywords by a taxonomy of event types can be helpful for the reasoner to find implicitly relevant events. ASP is a viable option for representing and reasoning on taxonomies or ontologies.

Although our modeling has been developed by taking social networks for elderly into account, it can be applied to general social networks featuring events. The huge amount of data generated in the social network, however, can be problematic.In order to compute suggestions efficiently, we should consider conditions and methods to partition the friendship graph and to focus on a subset of related events that leads to feasible suggestions.

5. REFERENCES

Baral, C. (2003). *Knowledge Representation, Reasoning and Declarative Problem Solving*. Cambridge University Press.

Gebser, M., Kaminski, R., Kaufmann, B., Ostrowski, M., Schaub, T., and Schneider, M. (2011a). Potassco: The Potsdam answer set solving collection. *AI Communications*, 24(2):105–124.

Gebser, M., Schaub, T., Thiele, S., and Veber, P. (2011b). Detecting inconsistencies in large biological networks with answer set programming. *Theory and Practice of Logic Programming*, 11(2-3):323–360.

Gelfond, M. and Lifschitz, V. (1991). Classical negation in logic programs and disjunctive databases. *New Generation Computing*, 9:365–385.

Gulbrendsen, S., Fikkan, E., Grunt, E., Mehl, K., Shamsolketabi, S., Singh, J., Vrucinic, M., Mathisen, B., and Kofod-Petersen, A. (2012). Social network for elderly. In *Proceedings of the Twelfth International Conference on Innovative Internet Community Systems (I2CS'12)*.

Nogueira, M., Balduccini, M., Gelfond, M., Watson, R., and Barry, M. (2001). An A-prolog decision support system for the space shuttle. In *Proceedings of the Third International Symposium on Practical Aspects of Declarative Languages (PADL'01)*, pages 169–183.

Simons, P., Niemelä, I., and Soininen, T. (2002). Extending and implementing the stable model semantics. *Artificial Intelligence*, 138(1-2):181–234.

Soininen, T. and Niemelä, I. (1999). Developing a declarative rule language for applications in product configuration. In *Proceedings of the First International Workshop on Practical Aspects of Declarative Languages (PADL'99)*, pages 305–319.