Modelling and Analysing Social Networks through Formal Methods and Heuristic Searches

Antonella Santone[1] and Gigliola Vaglini[2]

[1]*Dipartimento di Ingegneria, University of Sannio, Benevento, Italy*

[2]*Dipartimento di Ingegneria della Informazione, University of Pisa, Pisa, Italy*

Keywords: Social Networks, Formal Methods, Heuristic Searches.

Abstract: The paper presents a process algebraic approach to formal specification and verification of social networks. They are described using the Calculus of Communicating Systems and we reason and verify such formal systems by using directed model checking, which uses AI-inspired heuristic search strategies in order to improve model checking techniques.

1 INTRODUCTION

Recently, great interest arose for analysis methods of complex social networks, from communication networks, to friendship networks, to professional and organisational networks. A social network is a set of people (or organizations or other social entities) connected by a set of social relationships, such as friendship, co-working or information exchange. Social network analysis focuses on the analysis of patterns of relationships among people, organizations, states and such social entities. Social network analysis provides both a visual and a mathematical analysis of human relationships. The Web can also be considered as a social network. Social networks are formed between Web pages by hyperlinking to other Web pages.

Heuristic search (Pearl, 1984) is one of the classical techniques in Artificial Intelligence and has been applied to a wide range of problem-solving tasks including puzzles, two player games, and path finding problems. A key assumption of heuristic search is that we can assign a utility or cost to each state. This cost guides the search suggesting the next state to expand; in this way the most promising paths are considered first.

Model checking (Clarke et al., 2001) is a method to formally and automatically verify the correctness of finite-state concurrent and distributed systems. As model checking can be seen as a search in a state space, heuristics can be exploited to explore state spaces and verify properties. This approach is known as *directed model checking* (Edelkamp et al., 2001; Santone, 2003).

In this paper an application of directed model checking to social networks is presented. First social networks are described using the Calculus of Communicating Systems (CCS) of Milner (Milner, 1989). Then, we consider some interesting properties of social network and we show how these properties can be expressed in a temporal logic and verified using either model checking or heuristic search. As an example of heuristic search, we analyse the *social distance*: "how far (in terms of social distance) an actor is from others". In fact, the connections of an actor's social neighbours can be very important, even if the actor is not directly connected to them. An admissible heuristic function is defined which is syntactically defined, i.e., based on the CCS specification only, and can be automatically computed. With an admissible heuristic function, the A* algorithm is guaranteed to find the minimal social distance. This is a preliminary work to formalise a social network in a simple way. Our interest is in showing how formal methods can be applied in this field. As future work we want to enrich this model to capture more aspects.

2 THE PROPOSED METHOD

The term *network* has different meanings in different disciplines. In the social sciences, a network is usually defined as a set of actors (or agents, or nodes, or vertices) that may have relationships (or links, or edges, or ties) with one another see Fig. 1, which is a running example of this paper.

Santone A. and Vaglini G..
Modelling and Analysing Social Networks through Formal Methods and Heuristic Searches.
DOI: 10.5220/0004068903360339
In *Proceedings of the 7th International Conference on Software Paradigm Trends* (ICSOFT-2012), pages 336-339
ISBN: 978-989-8565-19-8

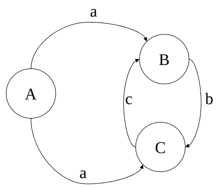

Figure 1: A simple example of network.

The two most common ways of representing social network data are by drawing the network and by using matrices. This section presents a representation of social networks through CCS and interesting properties are formalised by using the selective mu-calculus logic (Barbuti et al., 1999). The purpose is using formal verification environments also to analyse social networks. In this preliminary work social networks are formalised in a simple way, as discussed below, as future work the model can be enriched to capture more aspects. Each node N, with k outgoing arcs, is represented as the following CCS process:

$$N \stackrel{\text{def}}{=} \sum_{i=1}^{k} n.M_i$$

where M_i, for $i \in [1..k]$, are the immediate successors of N. Note that all arcs of a node N are labelled with the corresponding lower-case letter n.

For example, the CCS process representing the network in Fig. 1 is: $A \stackrel{\text{def}}{=} a.B + a.C \quad B \stackrel{\text{def}}{=} b.C \quad C \stackrel{\text{def}}{=} c.B$.

Clearly, many networks can be composed using the CCS parallel operator "|".

2.1 Basic Properties of Networks

In this section we first recall some important properties of networks and we formalise them using selective mu-calculus logic.

Size of a network. The size of a network can be determined in terms of the number of nodes of the network (as stated in (Hanneman and Riddle, 2005)) or, alternatively, as the number of edges in the network. In the network of our running example (shown in Fig. 1) the number of nodes is 3, while the number of edges is 4. Using any verification environment it is sufficient to exploit specific commands and we obtain the number of both states and edges.

Social distance. The connections of an actor's social neighbours can be very important, even if the actor is not directly connected to them. In other words, sometimes being a friend of a friend may be quite consequential. To capture this aspect of how individuals are embedded in networks, one approach is to examine how far (in terms of social distance) an actor is from others. The distance between two actors is the minimum number of edges that takes to go from one to another.

Reachability. Reachability between nodes is established by the existence of a path between the nodes. In simpler words, an actor is *reachable* by another if there exists a set of connections by which we can go from the source to the target actor, regardless of how many others fall between them. In general, the property: "node Y is reachable from node X" can be expressed with the following logic formula:

$$\langle x \rangle_{\emptyset} \langle y \rangle_{\emptyset} \, \mathtt{tt} \tag{1}$$

On the other hand, the property: "node Y is reachable from node X, without crossing nodes in the set S" can be expressed with the following logic formula:

$$\langle x \rangle_{\emptyset} \langle y \rangle_{S} \, \mathtt{tt} \tag{2}$$

Recall the network shown in Fig. 1:

- "C is reachable from A". The logic formula is: $\psi_1 = \langle a \rangle_{\emptyset} \langle c \rangle_{\emptyset} \mathtt{tt}$. It holds that $A \models \psi_1$ (instantiation of (1)).
- "it is possible that C is reachable from A not crossing node B. The logic formula is: $\psi_2 = \langle a \rangle_{\emptyset} \langle c \rangle_{\{b\}} \mathtt{tt}$. It holds that $A \models \psi_2$ (instantiation of (2)).

Cycle. A cycle is a walk where the beginning and end point of the walk are the same actor. In general, the property: "a cycle on actor X" can be expressed with the following logic formula:

$$\nu Z. \langle x \rangle_{\emptyset} Z \tag{3}$$

We formally check, on the CCS process A, the following property, which is an instantiation of the previous formula (3):

- "there exists a cycle on actor B": $\varphi_1 = \nu Z. \langle b \rangle_{\emptyset} Z$. It holds that $A \models \varphi_1$.

3 HEURISTIC BASED METHOD FOR CALCULATING SOCIAL DISTANCE

The problem of finding the minimal distance between two actors can be seen as a search problem: therefore, heuristic search (Pearl, 1984) taken from Artificial Intelligence can be used. We assume the reader be familiar with heuristic search algorithms, like A* and Greedy. A* returns a minimal-cost solution path whether the heuristic estimate function $\widehat{h}$ satisfies the so-called *admissibility* condition, i.e., $\widehat{h}$ is optimistic. Now, we define an admissible heuristic function able to find the minimum social distance from p to the node N. The heuristic we present is based on counting the number of actions that a process can perform before reaching a node N. Remember that in the CCS formalisation a node N has outgoing arcs labelled with n. We suppose that each edge has cost equal to 1. We formally define the function $\widehat{h_n}(p)$.

Definition 1 (**$\widehat{h_n}(p)$**). *Let p be a CCS process, $\mathcal{S}, \mathcal{S}'$ sets of visible actions, n the visible action of the node N to reach, and $\mathcal{C}$ a set of pairs $\{\langle x, \mathcal{S}' \rangle\}$, where x is a constant occurring in p. First, we define the auxiliary function $\widehat{h_n}(p, \mathcal{S}, \mathcal{C})$, inductively on p, as follows. Then, $\widehat{h_n}(p) = \widehat{h_n}(p, \emptyset, \emptyset)$.*

$$\textbf{R1}.\ \widehat{h_n}(nil, \mathcal{S}, \mathcal{C}) = \infty.$$

For $p = nil$ the function $\widehat{h_n}$ returns ∞ as this is not the node N.

$$\textbf{R2}.\ \textbf{if } \alpha \in \mathcal{S} \cup \{n\} \textbf{ then } \widehat{h_n}(\alpha.p, \mathcal{S}, \mathcal{C}) = 0$$
$$\textbf{else } \widehat{h_n}(\alpha.p, \mathcal{S}, \mathcal{C}) = 1 + \widehat{h_n}(p, \mathcal{S}, \mathcal{C}).$$

When applied to $\alpha.p$ the function returns 0 if α is either a restricted action (i.e., $\alpha \in \mathcal{S}$) or the desired action n, otherwise we recursively apply the function to find, if any, the node N. Roughly speaking, if α is restricted by $\mathcal{S}$ then $\alpha.p$ could not be able to move; thus, we optimistically return 0.

$$\textbf{R3}.\ \widehat{h_n}(p+q, \mathcal{S}, \mathcal{C}) = \min(\widehat{h_n}(p, \mathcal{S}, \mathcal{C}), \widehat{h_n}(q, \mathcal{S}, \mathcal{C})).$$

$$\textbf{R4}.\ \widehat{h_n}(p|q, \mathcal{S}, \mathcal{C}) = \min(\widehat{h_n}(p, \mathcal{S}, \mathcal{C}), \widehat{h_n}(q, \mathcal{S}, \mathcal{C})).$$

When either the choice or the parallel composition of two processes is encountered the minimum number of actions between the two components is returned.

$$\textbf{R5}.\ \widehat{h_n}(p \backslash L, \mathcal{S}, \mathcal{C}) = \widehat{h_n}(p, \mathcal{S} \cup L \cup \overline{L}, \mathcal{C}).$$

The function is initially applied to a process with $\mathcal{S} = \emptyset$ which is modified when the function is applied to $p \backslash L$ adding the actions in $L \cup \overline{L}$, to $\mathcal{S}$.

$$\textbf{R6}.\ \widehat{h_n}(p[f], \mathcal{S}, \mathcal{C}) = \widehat{h_n}(p, f^{-1}(\mathcal{S} \cup \{n\}), \mathcal{C}).$$

When considering a relabelled process we must take as set of actions the set $f^{-1}(\rho) = \{\alpha \mid f(\alpha) \in \rho\}$, since now the interesting actions are also those relabelled by f into actions in $\mathcal{S}$ or in n.

$$\textbf{R7}.\ \textbf{if } x \stackrel{\text{def}}{=} p_x,\ \langle x, \mathcal{S} \rangle \in \mathcal{C} \textbf{ then } \widehat{h_n}(x, \mathcal{S}, \mathcal{C}) = \infty\ .$$
$$\textbf{else } \widehat{h_n}(x, \mathcal{S}, \mathcal{C}) = \widehat{h_n}(x, \mathcal{S}, \mathcal{C} \cup \{\langle x, \mathcal{S} \rangle\})$$

We expand the body of each constant x only once since each constant already expanded is stored in $\mathcal{C}$. Initially, $\mathcal{C}$ is equal to the empty set. We return ∞ when we encounter a constant already expanded.

The following theorem ensure the admissibility of the heuristic function.

Theorem 1. *Let p be a CCS process and s be a state. It holds that: $\widehat{h_n}(s) \leq h^*(s)$, where $h^*(s)$ is the actual cost of a preferred path from s to a goal node.*

4 AN EXAMPLE

Let us apply our approach to a social network to obtain the minimal social distance between two actors. The node expansion terminates when a goal node is reached. The CCS definition of the social network p is the parallel composition of two sub-nets A and X with synchronisation on the action c.

$$\begin{array}{lll}
A \stackrel{\text{def}}{=} a.B & B \stackrel{\text{def}}{=} b.F + b.C & C \stackrel{\text{def}}{=} c.D + c.H + c.E \\
H \stackrel{\text{def}}{=} h.I & F \stackrel{\text{def}}{=} f.M & M \stackrel{\text{def}}{=} m.N \\
N \stackrel{\text{def}}{=} n.Q & Q \stackrel{\text{def}}{=} q.E & D \stackrel{\text{def}}{=} d.C \\
I \stackrel{\text{def}}{=} i.E & E \stackrel{\text{def}}{=} e.I & X \stackrel{\text{def}}{=} x.Z \\
Z \stackrel{\text{def}}{=} z.E & & p \stackrel{\text{def}}{=} (A|X[\overline{c}/z]) \backslash \{c\}
\end{array}$$

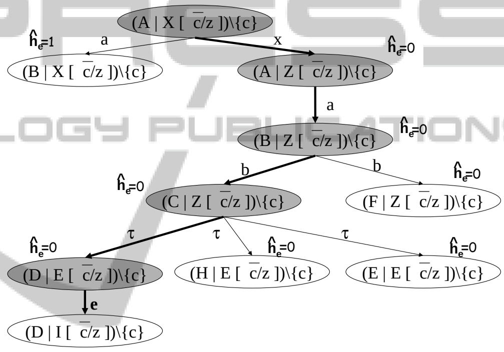

Figure 2: A simple example with Greedy and $\widehat{h_e}$.

Suppose that we want to obtain the minimal social distance between A and E. We apply the Greedy strategy. The application of A* is immediate: it is sufficient to use the evaluation function $f(s) = g(s) + \widehat{h}(s)$. In Fig. 2 the $\widehat{h_e}$-value of each node is reported and the shaded nodes represent expanded states. It holds that actor E can be reached with the minimal path with length equal to 5 (i.e. $x - a - b - \tau - e$). There exists another path reaching actors E with length equal to 7. We have obtained this result generating only 10 states while the standard transition system of p has 24 states and 39 transitions.

The method has been successfully applied to a real case study where several functionalities of social networks have been modelled such as: registration, add as friend, news feed.

5 RELATED WORK AND CONCLUSIONS

The paper propose several contributions: (i) an algebraic description of social networks through CCS; (ii) the definition of several properties of social networks (minimum distance, reachability, cycles) using a temporal logic, so that they can be verified trough model checking; (iii) an admissible heuristic function to verify social distance easy to compute and automatically calculated. A consequence of point (i) is that all the efficient techniques developed in CCS verification environments can be used, as for example compositional analysis (Clarke et al., 1989; Santone, 2002). Moreover, the result of point (ii) is that we offer a query language based on temporal logic instead of just a set of fixed properties; thus the approach allows the specification and verification of any interesting property (also not built-in in traditional tools).

At present, the most popular tool for Network analysis is UCINET[1] based on mathematical operations on matrices. For example, to prove the property "X is reachable from Y not crossing node Z", UCINET must found all paths from Y to X and check that Z does not occur in any path. In the presented approach the property is defined in the temporal logic and then automatically verified in the model checking environment.

To evaluate the effectiveness of the proposed method, we have to consider both complexity and scalability. From the complexity point of view, it is easy to show that the complexity of the calculation of $\widehat{h}(n)$ is linear in the length of the CCS specification. The heuristic function is simple to calculate since it is syntactically defined, i.e., based on the CCS syntax only; moreover, there is no need for user intervention or manual efforts to compute it. From the scalability point of view, in the work (Gradara et al., 2005) an experimental study has been carried out to prove the better scalability of the heuristic-based method with respect to other techniques generally used (BFS, DFS, etc.). Even if in that work a different property (deadlock-freeness in concurrent systems) has been considered, the results can be useful for a valid comparison. In some case studies a reduction of the state space of 55% has been reached and a consequently reduction in time with respect to BFS, for instance. Moreover, while it is well-known that most current formal methods are successfully applicable to small-scale systems, but do not scale up well, in this paper, directed model checking proposed in (Santone, 2003) has been proved able to allow formal methods to scale up.

Recently, the authors in (He et al., 2007) also propose a process algebraic approach to modeling and verifying the collective behaviors in social networks using MWB (Victor and Moller, 1994) with no consideration of the scalability. In (Jamali and Abolhassani, 2006) a state of the art survey of the works done on social network analysis, ranging from pure mathematical analyses in graphs to analyzing the social networks in Semantic Web, is given. The main goal is to provide a road map for researchers working on different aspects of Social Network Analysis.

[1] http://www.analytictech.com/ucinet/

REFERENCES

Barbuti, R., Francesco, N. D., Santone, A., and Vaglini, G. (1999). Selective mu-calculus and formula-based equivalence of transition systems. *J. Comput. Syst. Sci.*, 59(3):537–556.

Clarke, E. M., Grumberg, O., and Peled, D. (2001). *Model checking*. MIT Press.

Clarke, E. M., Long, D. E., and McMillan, K. L. (1989). Compositional model checking. In *LICS*, pages 353–362.

Edelkamp, S., Lluch-Lafuente, A., and Leue, S. (2001). Directed explicit model checking with hsf-spin. In *SPIN*, pages 57–79.

Gradara, S., Santone, A., and Villani, M. L. (2005). Using heuristic search for finding deadlocks in concurrent systems. *Inf. Comput.*, 202(2):191–226.

Hanneman, R. and Riddle, M. (2005). Introduction to social network methods. *Riverside, CA: University of California, Riverside (published in digital form at http://www.faculty.ucr.edu/ hanneman/nettext)*.

He, Z., Yuan, L., and Zeng, G. (2007). A process algebraic approach to modeling collective behaviors in social networks. In *Proceedings of the Third International Conference on Semantics, Knowledge and Grid (SKG 2007)*.

Jamali, M. and Abolhassani, H. (2006). Different aspects of social network analysis. In *Web Intelligence*, pages 66–72.

Milner, R. (1989). *Communication and concurrency*. PHI Series in computer science. Prentice Hall.

Pearl, J. (1984). *Heuristics - intelligent search strategies for computer problem solving*. Addison-Wesley series in artificial intelligence. Addison-Wesley.

Santone, A. (2002). Automatic verification of concurrent systems using a formula-based compositional approach. *Acta Inf.*, 38(8):531–564.

Santone, A. (2003). Heuristic search + local model checking in selective mu-calculus. *IEEE Trans. Software Eng.*, 29(6):510–523.

Victor, B. and Moller, F. (1994). The mobility workbench - a tool for the pi-calculus. In *CAV*, pages 428–440.