# 02287 Logical Theories for uncertainty and learning: Dynamic Epistemic Friendship Model on Facebook

**Thomas L. Skafte**
s153047

**Frederik K. Drachmann**
s153430

**Abstract**

In this article we try and model the actions of commenting, liking and posting on the wall of Facebook with the language of DEFL (dynamic epistemic friendship model). DEFL allows us to model social networks by combining the functionality of a multiple worlds model with a friendship defining model. This allows for indexical statements such as 'All my friends are nice', and using PDL-transformations enables sending these forms of messages with varying degrees of privacy.

If we wish to use the DEFL model for an actual representation of a Facebook network, some of the assumptions made in DEFL are required to be removed or changed and for DEFL to still hold up some interesting changes are necessary.

## 1 Introduction

The ambition for this paper is to use the ideas and logical model Dynamic Epistemic Logic of Friendship, DEFL, from the article (Seligman, Liu, and Girard 2013) and apply it on actual operations possible within Facebook. Through exercising the ideas of the DEFL model, plausible short comings or missing topics will arise and this will lend itself well to extending the model, be it new ideas or using old ones in new ways.

As in (Seligman, Liu, and Girard 2013) we will assume that any agent will only ever speak the truth and that any information sent to other agents is immediately known to the reader. Another assumption is that friendships are symmetric and irreflexive. We will later consider the fact that a proper Facebook model would require us to drop some of these assumptions, as well as how that would be accomplished.

DEFL is a dynamic version of the initial model Epistemic Friendship Logic, EFL, from the same paper. In order to make it dynamic it is combined with Propositional Dynamic Logic, PDL, as well as the model transformation, PDL-tranformations, from General Dynamic Dynamic Logic, GDDL. All of these will briefly be touched upon in the Background section, but can be explored in more depth in their respective papers.

## 2 Background

### 2.1 EFL

The language of EFL (epistemic friendship logic) consists of standard propositions $p \in Prop$ representing indexical variables as "I am in danger", and the added nominals which corresponds to agents $n \in ANom$, which shows the indexical propostion asserting the identification such as "I am n", The language can be inductively defined as:

$$\varphi ::= p|n|\neg\varphi|(\varphi \wedge \phi)|K\varphi|F\varphi|A\varphi$$

The K is read "I know that", F is read as "all my friends" and A is read as "every agent". Models for this language is described by a Kripke model of the form $M = \langle M, A, k, f, V\rangle$. These can be explained further in (Seligman, Liu, and Girard 2013).

Nominals are used to provide a way of referencing and changing perspective within other operations. This is done by two operations, $\downarrow a$ changes the perspective to the agent nominated by $a$, and $@a$ references the agent nominated by $a$.

### 2.2 DEFL, PDL

We define a dynamic element for the model, so it can be altered by applying programs. These program consists of operations $\mathcal{D}$ and the actions models s.t. for each $\Delta \in \mathcal{D}$ and each model $M$ for $\mathcal{L}$, there is a language model $\Delta M$, and for each of the states $w$ of $M$, there exists a state $\Delta w$ for $\Delta M$. We apply this extension the language $\mathcal{L}(D)$, and from this follows the dynamic epistemic friendship logic (DEFL) by adding the elements of $\mathcal{D}$ as propositional operators defining:

$$M, w, a \models \Delta\varphi \qquad \text{iff} \quad \Delta M, \Delta w, a \models \varphi$$

We define these operations $\mathcal{D}$ by using the language of propositional dynamic logic (PDL) with the basic programs K, A, F given by:

$$\mathcal{T} \;\; \pi ::= K|F|A|\varphi?|(\pi;\pi)|(\pi \cap \pi)|\pi*$$
$$\mathcal{F} \;\; \varphi ::= p|n|\neg\varphi|(\varphi \wedge \varphi)|\langle\pi\rangle\varphi$$

$$
\begin{aligned}
[\![p]\!]^M =& V(p),\ \text{ for } p \in \text{Prop}\\
[\![n]\!]^M =& W \times g(n), \text{for } n \in \text{ANom}\\
[\![(\varphi \wedge \psi)]\!]^M =& [\![\varphi]\!]^M \cap [\![\psi]\!]^M\\
[\![\neg\varphi]\!]^M =& W \setminus [\![\varphi]\!]^M\\
[\![\langle\pi\rangle\varphi]\!]^M =& \{w \in W | w[\![\neg\pi]\!]^M v \text{ and } v \in [\![\neg\varphi]\!]^M\\
& \text{ for some } v \in W\}\\
[\![K]\!]^M =& \{\langle(w,a),(v,a)\rangle | k_a(w,v)\}\\
[\![F]\!]^M =& \{\langle(w,a),(w,b)\rangle | f_w(a,b)\}\\
[\![A]\!]^M =& \{\langle(w,a),(w,b)\rangle | a,b \in A, w \in W\}\\
[\![\varphi?]\!]^M =& \{\langle(w,a),(w,b)\rangle | w \in [\![\varphi]\!]^M\}\\
[\![\pi_1;\pi_2]\!]^M =& \{\langle w,v\rangle | w[\![\neg\pi]\!]^M s \text{ and } s[\![\neg\pi]\!]^M v \text{ for some } s \in W\}\\
[\![\pi_1 \cup \pi_2]\!]^M =& [\![\pi_1]\!]^M \cup [\![\pi_2]\!]^M\\
[\![\pi*]\!]^M =& \{\langle w,v\rangle | w = v \text{ or } w_i[\![\neg\pi]\!]^M w_{i+1}\\
& \text{for some } n \geq 0, w_0, ..., w_n, \in W,\\
& w_0 = w \text{ and } w_n = v\}
\end{aligned}
$$

### 2.3 PDL transformation

For representing models after applying some actions we use the simplest of GDDL operators called PDL-transformations. These programs consisting of operations transforms the model by redefining the basic program. For example the operator $[K := \pi]$ acts on a model M to produce a new model $[K := \pi]M$, s.t.:

$$[\![K]\!]^{[K:=\pi]M} = [\![\pi]\!]^M$$

When there are no changes to a state $[K := \pi]w = w$, so we can define the semantics as follows:

$$M, w, a \models [K := \pi]\varphi \quad \text{iff} \quad [K := \pi]M, w, a \models \varphi$$

## 3 Facebook network

Consider the example where we have four people, Alice (a), Bob (b), Charlie (c) and Dave (d). They are each friends with two of the others, such that it forms a square. Alice now considers whether Bob and Dave are friends. This can be modelled as in Figure 1, with the two worlds $u, v$.

We will be considering the Facebook Model to be a *named agents model*, meaning that for all agents $a \in A$ there is a nominal $n \in ANom$ such that $g(n) = a$.

### 3.1 Private messages

We shortly introduce the notion of messages as in the article (Seligman, Liu, and Girard 2013). First we define a cut, this will remove all the uncertainties about $\varphi$.

$$\text{cut}_K = (\varphi; K; \varphi) \cup (\neg\varphi; K; \neg\varphi)$$

In essence this shows the agent whether $\varphi$ holds for him. This can be shown with the following expression.

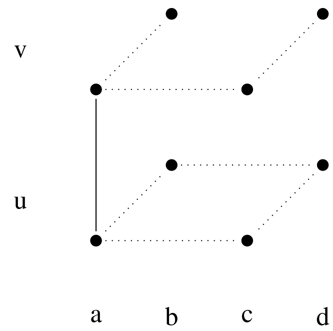



Figure 1: Uncertainty about friendship

$$[K := \text{cut}_K]A(K\varphi \vee \neg\varphi)$$

Which means all agents now knows either $\neg\varphi$ or $\varphi$. Now the actually send the messages we have to extend this cut further. We want to send a message (in this case anonymously) to an agent $n$.

$$\text{send}_\theta(\psi) = [K := (n?; \text{cut}_K(\psi)) \cup (\neg n?; K)]$$

Here the agent $n$ is introduced to the truth or falsity of $\psi$ and it preserves all $k_a$ relations for agents not satisfying $n$.

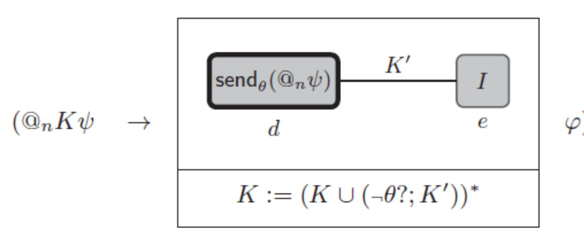
We now want to also send messages about a sender $n$ to some agent $n$, which will be having the following syntax $[n \triangleleft \psi! : m]\varphi$, and is described by:

$$(@_n K\psi \rightarrow [\text{send}_n(@_n\psi)]\varphi)$$

The message could be something as "I am unemployed". With this we make the assumption that the message is known by the sender. A message send to everyone would look as follows $[n \triangleleft \psi! : \top]\varphi$. We also want to introduce a way to send a message that says something about the receiver. We therefore define $[n : \psi! \triangleright m]\varphi$ that says agent $n$ announces $\psi$ about $\theta$ and $\varphi$ what holds afterwards for the agents:

$$(@_n A(n \rightarrow \psi) \rightarrow [\text{send}_n(\psi)]\varphi)$$

These messages are only semi-private since all agents can observe that the messages are sent but do not know of the content in the messages. To introduce a private announcement we have to use the PDL-transformation explained in (Seligman, Liu, and Girard 2013). It looks as follows



We call this formula $\llbracket n \triangleleft \psi! : m \rrbracket$. These terms are very loosely explained, and can be further explored in (Seligman, Liu, and Girard 2013).

With these definitions we can start by defining the standard private message between friends sent with Messenger. We try and show the case: "Bob (b) tells Charlie (c) that I am happy ($\varphi$)."

$$\llbracket b : \varphi! \triangleright c \rrbracket\ @_c K @_b \varphi$$

Then there is also the possibility that Bob sends this message individually in private to several of his friends, for example "Bob tells Charlie that I am happy, and then he tells Eve and then he tells Kurt". This can be modelled the following way.

$$\llbracket b : \varphi! \triangleright c \rrbracket\ @_c K @_b \varphi \wedge \llbracket b : \varphi! \triangleright e \rrbracket\ @_e K @_b \varphi \\ \wedge \llbracket b : \varphi! \triangleright k \rrbracket\ @_k K @_b \varphi$$

When Bob tells his friends individually it means that none of his friends knows, that the others knows, that Bob is happy. This is different from a group message since none of the friends are aware of the other message Bob is sending.

### 3.2 Group Messages

In Facebook there exists exclusive groups, that not everyone are a part of. If we consider Facebook as a set of agents, $F$, then a group $\Theta$ is a subset of $F$, $\Theta \subseteq F$. Now let $\theta$ be the proposition that the agent is in group $\Theta$. Using the same model as earlier we can model that Alice is considering whether or not Dave is in group $\Theta$, that she is also in.

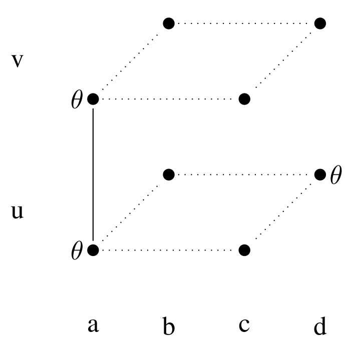



Figure 2: Uncertainty about propositions

We will be using the proposition of whether or not the agent is in the group, $\theta$, to represent the group $\Theta$.

$$A(\theta) = \Theta$$

This notion of groups can be used to represent any set of agents that have a proposition in common. For example the set of agents that are following a *Page* or even more fleeting sets such as the set of friends that an agent has, which would be:

$$\text{Proposition noting the friends of } a = \langle F \rangle a$$

With regards to sending messages, we can now use the notation of groups to send to more than one agent at a time. When an agent wants to send a message to every agent in $\Theta$ the proposition defining $\Theta$ is used.

$$\llbracket n \triangleleft \psi! : \theta \rrbracket A(\theta \to K @_n \psi)$$

Sending the message $\phi$ about yourself, to all your friends would be:

$$\llbracket n \triangleleft \phi! : \langle F \rangle n \rrbracket A(\langle F \rangle n \to K @_n \psi)$$

**Multiple Private Messages vs Group Messages** There is a distinct difference between sending a private message to multiple agents individually and sending a shared group message to the same set receivers. When sending individually the receivers have no knowledge of whom else the sender has sent the information to, if any. In contrast when sending a group message, every member of the group knows that any other member of the group knows, ad infinitum. This is what is known as common knowledge, but it is only within the group that was sent to. Thus it results in common knowledge for a subgroup of agents within the model, but not for the entire set of agents in the model.

### 3.3 Posts, Likes and Comments

The action of liking a post on Facebook can be different depending on what settings the individual user have when it comes to the "apparent" privacy of their content. The privacy depends both on the "liker" (sender) of the like and the receiver of the like. The most simple case is when both users have the most private settings, which means the post being liked can only be seen by friends, and the action of liking is not shared to the liker's (sender's) friends.

$$\text{Like}: \ [n : l(post)! \triangleright \langle F \rangle m] @_m F K @_n l(post)$$

The same principle can be applied to posts and comments, the only change is the information passed to the friends can be more complex.

$$\text{Post}: \ [n : \varphi! \triangleright \langle F \rangle m] @_m F K @_n \varphi \\ \text{Comment}: \ [n : \varphi! \triangleright \langle F \rangle m] @_m F K @_n \varphi$$

We can loosen up the privacy on Facebook, and talk about public posts. These posts are available for all users at any given time. The special feature about public posts is, when a user comments/likes the post, the post will show up on their friends wall, even their friends may not following/friends with the person that originally made the post. This can cause the information to cascade through Facebook, if friends of friends of friends, and so forth, keep commenting and liking. We will discuss the effects of the cascading in the discussion. The cascade can be described as a chain of posts to friends, of friends, of friends etc. By the end of the chain possibly everybody on Facebook knows, if the whole social network on Facebook is fully connected.

It also possible to ask questions in posts/comments and a model for this is defined in (Seligman, Liu, and Girard 2013), so to complete this task is trivial and we will not expand upon it.

# 4 DEFL Extensions

## 4.1 Notification

We remove the assumption that when a users posts/likes/comments it does not become knowledge for the agent instantly but instead, the user gets a notification, but then has to go and click on the notification to get the information from the sender. An notification can be written almost as before in group message the only difference is the message holds no information apart from there is something the agent can check. We assume that agents in the system considers receiving notifications from all agents. $\psi$ is the predicate of having received the notification.

$$[\![ n \triangleleft \psi! : \langle F \rangle n ]\!] @_n FK\psi$$

In the previous model receivers of the information knew that all other friends of n had read the post. In this setting the friends only know that friends of n has received a notification, but they will never know if the friends read the message or not. When a one the friends reads the message.

The issue is that messages defined in this setting are actually a cut, meaning it does actually "add" extra information to the model, but rather removes uncertainty depending on the message, therefore notifications cannot be modeled as message, but needs to be a direct change to the world model an adding a new predicates.

It is possible to show situation if the model holds the information about this notification and message initially. "Bob(b) posts a message (m) to his two friends Jack (j) and Sofie (s), but they get a notification but do not read the message, but they do not know if their counter part red the message."

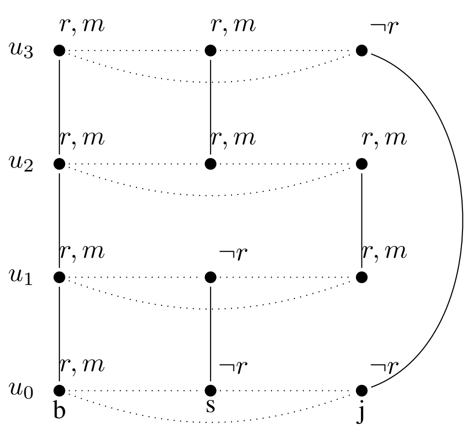



Figure 3: Caption

These are the possible states. b is unsure of any combination of Sofie and Jack having red the message, but he and everybody else knows, since he sent the message he must know it. Sofie is only in doubt about Jack, and vice versa. This models the idea of a notification and the uncertainty it creates for the agents. If Sofie then told her friends that she has not red the message $[\![ s \triangleleft \neg r : \langle F \rangle j ]\!] @_s FK @_s \neg r$. The model would change the following way:

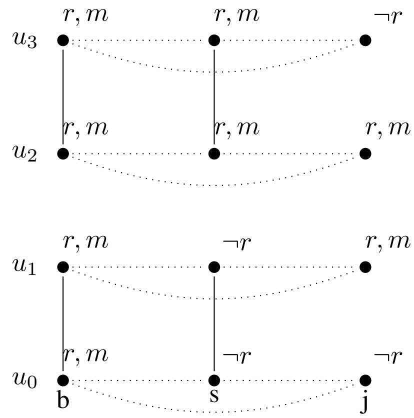



Figure 4: Caption

Only Sofie and Bob are in doubt of two worlds because the do know the state of Jack, but Jack is completely sure about the state of the world.

In order for this to work together with the $\text{send}_\theta(\psi)$ PDL-transformation, it must be altered slightly.

$$\text{send}_\theta(\psi) = [K := (n?; r?*; \text{cut}_K(\psi)) \cup (\neg n?; K)]$$

In this new version we consider the possibility of whether or not the agent has read the message, $r?$, before updating the model using the $\text{cut}_K$ operation. In order to keep the possibility that the agent reads the message later the $*$ is required to keep checking endlessly.

## 4.2 Beliefs

The formal structure for applying this belief system to DEFL is described in (P. Girard and Liu 2012), except for the part about friendship that is specific to this system. We therefore try and give some intuition about how this belief system could be used in DEFL.

We could model the belief for each individual agent as normally. If we take the coin flipping example where two agents have slightly different opinions on the coins bias.

| Hypothesis | Mass | Belief | Plausibility |
|---|---|---|---|
| Heads | 0.2 | 0.2 | 0.5 |
| Tails | 0.5 | 0.5 | 0.8 |
| Either | 0.3 | 1 | 1 |

Table 1: Agent 1

| Hypothesis | Mass | Belief | Plausibility |
|---|---|---|---|
| Heads | 0.5 | 0.5 | 0.5 |
| Tails | 0.5 | 0.5 | 0.5 |
| Either | 0 | 1 | 1 |

Table 2: Agent 2

These two agents are then friends in our DEFL model. The interesting part then comes when for example agent 2 tries to tell agent that the coin is actually a fair coin, because if we assume that friends completely trust each other then agent 1 would just change his beliefs. The assumption works in theory, but is not plausible in reality. Some people trust others more, therefore it would make sense to add a factor to the friendship relation indicating how much they actually trust each other. We keep the assumption of symmetry, meaning friends will have the same mutual trust. Another part is people might trust each other more or less depending on the subject they are discussing, but we will disregard this fact. Lets assume the trust between agent 1 and agent 2 is 0.8, where 1 is full trust and 0 is no trust, so they are actually not friends. This means the information that agent 1 receives from agent 2 will be taken with a grain of salt. For agent 1 this means he wont change his information about tails, since they agree, but on heads he increases the probability slightly depending on the trust. This a very rough idea of how to try and implement beliefs into the DEFL system.

This ties into the notion of degrees of friendship that is not currently modelled in the system. A way of modelling this could be adding weight to the friend edges indicating the strength of the friendship. The weight could be set in regards to how much they trust each other, which then ties into the belief. By introducing the notion of different level of friends, the ability to target specific levels of friends would make sense. If we look at our original model $M = \langle W, A, k, f, V \rangle$ the family $f$ of friendship relations are changed, so they are also have a weight between their edges called e $f_w(a, b, e)$. We then change how the interpret $\mathcal{L}$ for the friendship.

$$
\begin{array}{rll}
M, w, a \models F\varphi & \text{iff} & M, x, a \models \text{ for every } b \in A \\
 & & \text{such that } \langle a, b \rangle \in f_w(a) \\
M, w, a \models F(x)\varphi & \text{iff} & M, x, a \models \text{ for every} \\
 & & b \in A \text{ such that } \langle a, b \rangle \in f_w(a, e) \\
 & & \text{and } x < e \text{ where } f_w(a, e) \\
 & & \text{and } 0 < x \leq 1
\end{array}
$$

Also to introduce the same notation into the program language:

$$[\![F(x)]\!]^M = \{\langle (w, a, e), (w, b, e), x < e \rangle | f_w(a, b, e)\}$$

This would mean the program only gets edges of friends that have a friendship relation with weight over x. The friendship value would be between $0 < w \leq 1$, therefore setting x to 0 would mean all friendship relations. There would be no friendship relations with weight 0, since that would mean the people are not friends. Therefore the previous friendship is equal to doing a $F(0)\varphi$. To show only the friends with a threshold lower then some x would be to do the negation $\neg F(x)$. With this addition to the language it is possible to contact friends at specific levels.

## 5 Discussion

Without the notion of belief the DEFL model can only say an agent knows something or does not know it. With the addition of belief we get to show with what certainty the agents believe the different possibilities, which allows for more accurate modeling. Since any information that an agent obtain is from another agent we also introduced a value to indicate how trustworthy an agent finds other agents. This trust value is tied to the friendship relation, so if an agent acquires information from someone who is not their friend, then there is no value of which the trustworthiness can be assessed. Arguably this might be a design flaw and reason enough to split the two definitions apart. Another solution could be to have agents not trust anyone who is not their friend, but more than likely this does not model the world correctly.

Given a proper model of Facebook, one could pinpoint sources of influence, meaning people that when they utter themselves on Facebook the information spreads quickly to a large portion of the network. These people are extremely valuable when it comes to what the general public believes, and they have the power to use information. The power of this compared to a regular social graph, is the ability to monitor how information moves. Normally in a social graph a agent (nodes) out degree would determine how much impact they have, but this is not necessarily true if people do not listen to what this agent says.

The effect of informational cascade (Baltag et al. 2013) on sites such as Facebook can happen via posts, likes and comments. The property arises, when people are commenting and liking on posts. When a post is liked it is shared among the friends of the one that liked it. On of his friends could then like, and it would be shared again and so forth. This could potentially spread across the whole of Facebook and give rise to a version of Informational Cascade. The more people that believe the fact that was stated in the post, the more likely other people are to believe in it regardless of it being true or false. This is due to it becoming common knowledge among a larger group of people. It is hard to say when the limit caps out in comparison to the Urn Example but it has the same properties and can cause the same kind of misinformation. An example of this could be Fake News, where people wrongly come to believe something because of popular opinion and people of influence saying it is true.

## 6 Conclusion

We have used DEFL in the Facebook domain to try and model the interactions under some assumptions, to fit Facebook into the DEFL model. The interactions we model are posting, liking and commenting. What we discovered were how similar the actions are and how easy they are to model in DEFL. After having shown these we tried to expand on the DEFL model by notifications, beliefs and hierarchy of friendship. We modified the DEFL model so the friendship connection had a weight indicating the friendship quality. In the end we discussed of the issue of the model, and how the model could maybe can show how something "fake" can become information that people believe to be true.

## 7 Future Works

**Asymmetric Friendships** The assumption that all friendships are symmetric turns out to model Facebook quite well since this is how friendships statuses work on the platform, however an asymmetric friendship model could work, but some very large changes would be required.

Most notably the way an agent $a$ sends a message to all its friends is by using the proposition that they have $a$ as a friend.

$$[\![n \triangleleft \psi! : \theta]\!] A(\theta \rightarrow K@_n\psi)$$

This style of using propositions would no longer be available as agent $a$ might have agent $b$ as a friend, but $b$ might not have $a$ as a friend. Thus if $a$ uses the same method for broadcasting to its friends, then $b$ would not receive the message.

**Abstracting the notion of friendship** The way that DEFL models friendship is that there is a direct connection between two agents. By abstracting this notion we can also use DEFL on internet connections, saying that when two addresses connect to each other a friendship relation is created. This allows us to model how the internet interacts, and since each agent is a computer we can still use the notion of knowledge and uncertainty from a possible worlds model. There are possible many other ways to abstract friendships into useful interpretations.

## 8 Related works

The article (Hansen and Christoff 2015) takes a very different approach to the idea of modeling social networks. Their idea utilizes the normal structure of a network, but then allows information to diffuse through the network. Instead of being in doubt about different worlds, all agents in the network have a vector symbolizing their knowledge. The vector may for example contain information about if the person is infected or not.

This way of representing the world it simpler than ”almost” creating infinitely many different world models as in (Seligman, Liu, and Girard 2013) to try and show all the knowledge that an agent has (or might gain). The agents knowledge is ”just” defined by a vector filled with information. Of course for both cases it would probably not be feasible to show all knowledge about all agents and connection in a social network, but modeling smaller parts of interactions would be interesting as we have tried to do.

## References


[Baltag et al. 2013] Baltag, A.; Christoff, Z.; Hansen, J.; and Smets, S. 2013. Logical models of informational cascades. 405–432.

[Hansen and Christoff 2015] Hansen, J., and Christoff, Z. 2015. A logic for diffusion in social networks. *Journal of Applied Logic* 13(1):48–77.

[P. Girard and Liu 2012] P. Girard, J. S., and Liu, F. 2012. General dynamic dynamic logic.

[Seligman, Liu, and Girard 2013] Seligman, J.; Liu, F.; and Girard, P. 2013. Facebook and the epistemic logic of friendship. *CoRR* abs/1310.6440.