Everyone Knows that Everyone Knows

Ditmarsch, Hans van; Gattinger, Malvin; Ramezanian, Rahim

Published in:
ArXiv

Everyone Knows that Everyone Knows: Gossip Protocols for Super Experts

Hans van Ditmarsch

CNRS, LORIA, University of Lorraine, France

Malvin Gattinger

University of Groningen, Netherlands

Rahim Ramezanian

Shomara LLC, Tehran, Iran

Abstract

A gossip protocol is a procedure for sharing secrets in a network. The basic action in a gossip protocol is a telephone call wherein the calling agents exchange all the secrets they know. An agent who knows all secrets is an expert. The usual termination condition is that all agents are experts. Instead, we explore protocols wherein the termination condition is that all agents know that all agents are experts. We call such agents super experts. Additionally, we model that agents who are super experts do not make and do not answer calls. Such agents are called engaged agents. We also model that such gossip protocols are common knowledge among the agents. We investigate conditions under which protocols terminate, both in the synchronous case, where there is a global clock, and in the asynchronous case, where there is not. We show that a commonly known protocol with engaged agents may terminate faster than the same protocol without engaged agents.

1. Introduction

The gossip problem addresses how to spread secrets among a group of agents by pairwise message exchanges: telephone calls. We assume that each agent holds a single secret, and that when calling each other the agents exchange all the secrets they know. An agent may call another agent if it has that agent's telephone number. It is typically assumed that the goal of the information dissemination is that all agents know all secrets. The situation can be represented

Email addresses: hans.van-ditmarsch@loria.fr (Hans van Ditmarsch), malvin@w4eg.eu (Malvin Gattinger), rahim.ramezanian@gmail.com (Rahim Ramezanian)

by a graph or network where the nodes are the agents and where, when two nodes are linked, the agents can call each other.

There are many variations of the problem. It goes back to the early 1970s [1, 2, 3, 4, 5]. In this 'classical' setting (for an overview, see [6]) only secrets are exchanged, and the focus is on minimum execution length of protocols executed by a central scheduler. Later publications assume that the scheduling is *distributed* [7, 8]. Fairly recent developments focus on gossip protocols with *epistemic* preconditions for calls [9, 10, 11, 12, 13, 14]. For example, agents may only call another agent once, or only if they do not know the other agent's secret, etc.

In *dynamic* gossip [15, 16] the agents do not only exchange all the secrets they know but also all the telephone numbers they know. This results in network expansion: not only the secret relation but also the number relation is expanded after a call. The network is then dynamic, which explains the term. However, if the number relation is a complete digraph (the universal relation), i.e., when all agents know all telephone numbers, then the dynamic and classical gossip problem coincide. Here we will assume complete digraphs and thus not investigate dynamic gossip.

Another way to load the messages beyond merely exchanging secrets is to exchange *knowledge about secrets*. This approach is taken in [17]. Primarily, in a call the two agents may exchange all the secrets they know. But once this is done, they may also exchange the information 'everyone knows all the secrets'. This requires that the number of agents is known. And once *that* is done, they may exchange the information 'everyone knows that everyone knows all the secrets', and so on. They thus achieve higher-order shared knowledge of all secrets (all the agents know that all the agents know, etc.).

In this contribution we investigate gossip protocols with the epistemic goal that all agents know that all agents know all secrets. Clearly, this assumes that the agents know how many (other) agents there are.

- The protocol terminates when *everyone knows that everyone knows all secrets*.

However, we continue to assume that agents only exchange the same basic information as in the classical gossip problem, i.e. only secrets. So, unlike [17] we do not achieve the epistemic goal by loading the messages with epistemic features. The agents may also have knowledge of the protocol, or of the behaviour of other agents. We consider various such modifications, and will investigate how making such assumptions affect properties such as termination and execution length.

- Agents know what gossip protocol is used by all agents.
- Agents who know that everyone knows all secrets *no longer make calls*.
- Agents who know that everyone knows all secrets *no longer answer calls*.

	a	b	c	d
$\xrightarrow{ab}$	ab	ab	c	d
$\xrightarrow{cd}$	ab	ab	cd	cd
$\xrightarrow{ac}$	abcd	ab	abcd	cd
$\xrightarrow{bd}$	abcd	abcd	abcd	abcd

Table 1: Results of the call sequence $ab; cd; ac; bd$.

An agent who knows all secrets is called an *expert*, as usual. We call an agent who knows that everyone is an expert a *super expert*. So our epistemic goal is for all agents to become super experts, where we will also investigate the effect of additional assumptions such as knowledge of the protocol and that super experts no longer make and answer calls.

In the remainder of this introductory section we give examples to motivate our approach and we outline our results.

Let there be four agents a, b, c, d. Each agent holds a single secret to share. Consider the call sequence $ab; cd; ac; bd$. In a call, agents exchange all secrets they know. After the call ab, agents a and b both know two secrets, and similarly after the call cd, agents c and d both know two secrets. Therefore, after the subsequent call ac, agents a and c both know all four secrets: they are experts. Similarly, after the final call bd, b and d are experts. So, after $ab; cd; ac; bd$, all agents are experts. See Table 1.

In fact, the agents know a bit more than that. After call ac agent a is not only herself an expert but she also knows that agent c is an expert, and agent c also knows that agent a is an expert. (We typically use female pronouns to refer to a, male pronouns to refer to b, female pronouns to refer to c, and so on.) Similarly, after call bd, agent b also knows that d is an expert, and d also knows that b is an expert. Can the agents continue calling each other until they all know that they are all experts, i.e., until they all know that they all know all secrets? Yes, they can.

Let us first consider agent a. In order to get to know that everyone knows all secrets, a has to make two further calls: ab and ad. Let us suppose these calls are made, and in that order, i.e. consider the whole sequence $ab; cd; ac; bd; ab; ad$. First, note that before and after those calls the agents involved are already experts, so no factual information is exchanged. However, the agents still learn about each other that they are experts. Hence, after ab, agent a knows that b is an expert and after ad she knows that d is an expert. As she also knows this from herself, a therefore now knows that everyone is an expert. She has become a super expert.

Let us now consider agent b. In call bd he learnt that d is an expert, and in the additional call ab he learnt that a is an expert. And again he obviously knows from himself that he is an expert. Therefore, in order to get to know that everyone is an expert, b only needs to make one additional call, bc, and b then is a super expert.

We now consider agent c. Similarly, after yet another call cd, c is a super expert, which can be observed by highlighting the calls wherein c learns that another agent is an expert, as follows: $ab; cd; \mathbf{ac}; bd; ab; ad; \mathbf{bc}; \mathbf{cd}$. We caught two birds in one throw, because after that final call cd also agent d knows that all agents are experts: $ab; cd; ac; \mathbf{bd}; ab; \mathbf{ad}; bc; \mathbf{cd}$.

Therefore, all agents are super experts after the call sequence

$$ab; cd; ac; bd; ab; ad; bc; cd.$$

This contribution is about gossip protocols with the termination condition that everyone knows that everyone knows all secrets. To our knowledge this setting has not been studied in detail before. In particular it differs from [17] because we do not allow agents to exchange more information than merely their secrets.

We now motivate our modifications of the usual call rules in gossip. As a first idea, suppose any agent who is an expert no longer makes calls and no longer answers calls. We call such agents *engaged* and a call that is not answered we name a *missed call*. Given this new rule, can everyone still become an expert? Yes. For example, after the already mentioned call sequence $ab; cd; ac; bd$ all agents are experts, and all calls were answered. However, now consider the sequence $ab; ac; ad$. After this, agents a and d are experts. Agents b and c can now no longer become experts: if either were to call a or d, this would be a missed call. Note that agents do not learn any secrets from a missed call. Hence in this case b and c can never learn the secret of d: they can still call each other, and after additional call bc or cb agents b and c would both know three secrets but not all four secrets, hence they are not experts. The protocol cannot terminate.

We could additionally assume common knowledge among the agents that a missed call means that the agent not answering the call is an expert. But that does not make a big difference. After a missed call as above agents b and c would thus know that a and d are experts. But, for example, that agent b knows that a knows the secret of d, does not make b himself know the secret of d. They cannot use that knowledge to become experts themselves. With the classical gossip goal wherein all agents become experts the presence of engaged agents prevents termination even for very simple protocols. We conclude that this first idea of a condition for missed calls is not very satisfactory.

In this contribution we therefore employ the idea of missed calls in a different way. Let us now suppose that the goal of the protocol is for all agents to become super experts, and that an agent *who is a super expert* no longer makes calls and no longer answers calls. This requirement is harder to fulfil than the previous requirement that an agent *who is an expert* stops making and answering calls.

We can already satisfy the stronger termination requirement that all are super experts without such missed calls, for example, with the above sequence $ab; cd; ac; bd; ab; ad; bc; cd$. This is not entirely obvious. However, observe that after the subsequence $ab; cd; ac; bd; ab;$ *ad only* agent a knows that everyone is an expert, and in the subsequent call *bc only* agent b learns that, and *only* in

the final call cd agents c and d simultaneously learn that. No call is made to a super expert. Therefore, there are no missed calls.

However, now consider the call sequence $ab; cd; ac; bd; ab; ad; ba; ca; da$ with this missed call semantics. All final three calls are missed calls, because a already knows that everyone is an expert. What do b, c, and d respectively learn from these calls? Well, nothing whatsoever, as just like above we did not make any assumptions so far about the meaning of a missed call in this new context. Therefore, after those calls we can still make the additional calls $bc; cd$ in order to satisfy that everyone knows that everyone is an expert.

Let us now, as above, additionally assume that it is common knowledge among the agents that a missed call means that the agent not answering the call is a super expert. Now, unlike above, that makes a big difference. Given the sequence $ab; cd; ac; bd; ab; ad; ba; ca; da$, in the three final missed calls ba, ca, and da, respectively, agents b, c, d then learn from a that all agents are experts, so that after the entire sequence all agents know that all agents are experts. Again, we are done.

Before we continue, let us make three more observations. Firstly, if the three missed calls had been ordinary calls, the termination condition would not yet have been met. For example, agent d would then not know that agent c knows all secrets. Additional calls would have been needed. Secondly, although the sequence with three missed calls is one call longer than the previous sequence that also realizes the knowledge objective, in general there are terminating sequences with missed calls that are shorter than any other terminating sequence without missed calls, as we will prove later. Thirdly, as in a missed call the agent calling must already be an expert (otherwise the agent called cannot be a super expert), no factual information would have been exchanged if that call had been an ordinary call. So the presence of missed calls does not prevent agents from becoming experts in the first place, which would have wrecked our chances to reach the protocol goal.

The modelling solution for missed calls, that is novel, is similar to a modelling solution for making protocols common knowledge, presented in [18]. We incorporated both in this contribution. This also allows us to investigate how we can achieve that all agents are super experts with the constraints of some protocols known from the literature, such as the protocol CMO wherein you are only allowed once to be involved in a call (as the agent making or receiving the call) [16].

For example, consider again the sequence $ab; ac; ad$ after which agents a and d are experts. Agent a may no longer be involved in any subsequent call according to CMO. It is therefore impossible for her to get to know that everyone is an expert. So, common knowledge of a protocol comes with additional constraints. It may also come with additional advantages: in this case we can sometimes achieve common knowledge of termination under synchronous conditions, i.e., if all agents know how many calls have been made, even if they were not involved themselves in all those calls. We will report some such cases, in particular for CMO: for example, after an extension of $ab; ac; ad$ with three more calls, all agents including a are super experts. Unfortunately, if we also allow missed

calls this may no longer be the case, namely when an agent who already is a super expert *must* call another agent in order for all agents to become super experts. Such an extra complication can be overcome if agents have a notion of time, and if we allow a so-called *skip* action that merely stands for a tick of the clock. We will carefully distinguish all such modelling aspects.

To find out what agents know, we need to consider all call sequences they consider possible. Such reasoning about call sequences is a non-trivial exercise. To automatically find and verify such protocol executions we used the model checker GoMoChe for gossip protocols available at `https://github.com/m4lvin/GoMoChe`. Assuming synchrony, this means reasoning about finite sets of call sequences, which is sufficient to verify knowledge. Assuming asynchrony, this means reasoning about infinite sets of call sequences of arbitrary finite length, which cannot be done with a model checker. However, it is often sufficient to verify ignorance, i.e., lack of knowledge, namely by producing two 'witness' call sequences with opposite properties. Such witnesses can already be found for call sequences of 'small' length, by reasoning about finite sets of call sequences, namely of a certain maximal length. We also used the model checker GoMoChe for that, to great effect.[1]

Outline. Section 2 presents a logical language and semantics for gossip protocols with the epistemic goal that all agents know that all agents know all secrets. A protocol is super-successful if all executions terminate satisfying this condition. We also recall four gossip protocols from the literature: ANY, PIG, CMO, and LNS. We obtain various results for the protocols ANY and PIG, mainly that they are (fairly) super-successful (both for the synchronous and asynchronous versions). Section 3 refines the logic in order to model common knowledge of gossip protocols. We then show that synchronous known CMO is super-successful. Section 4 further refines the semantics with the feature that super experts do not make calls and do not answer calls. We then show that, if this is known, super-successful protocol executions can be shorter. However, under these conditions CMO is no longer super-successful. Section 5 presents an even further refinement of the semantics by adding the feature of *skip* calls following terminal protocol-permitted sequences, that allow us to regain a super-successful CMO.

2. Gossip protocols for super experts

2.1. Syntax and semantics

Suppose a finite set of agents $A = \{a, b, c, \dots\}$ is given. We assume that two agents can always call each other, i.e., a complete network connects all the

[1] It should be possible in principle to have an asynchronous model checker of knowledge as well, namely using the notion of redundant call as in [14], that bounds the maximal length of a call sequence without redundant (non-informative) calls, and that therefore also makes the sets of indistinguishable call sequences (and the length of individual sequences) finite again.

agents. Let $S \subseteq A^2$ be a binary relation such that we read $S_x y$ (for $(x, y) \in S$) as "agent x knows the secret of agent y," and where S_x stands for $\{y \in A \mid S_x y\}$. For the identity relation $S = \{(x, x) \mid x \in A\}$ we write I.

The agents communicate with each other through telephone calls. During a call between two agents x and y, they exchange all the secrets that they knew before the call. So if a call takes place the binary relation S may grow.

A *call* or telephone call is a pair (x, y) of agents $x, y \in A$ for which we write xy. Agent x is the *caller* and agent y is the *callee*. Given call xy, call yx is the *dual call*. An agent x is *involved* in a call yz iff $y = x$ or $z = x$. A *call sequence* is defined by induction: the empty sequence ϵ is a call sequence. If σ is a call sequence and xy is a call, then $\sigma; xy$ is a call sequence. Let S be the secret relation between agents and σ a call sequence. The result of applying σ to S is defined recursively as:

$$S^\epsilon = S; \text{ and } S^{\sigma;xy} = S^\sigma \cup (\{(x, y), (y, x)\} \circ S^\sigma).$$

We write $|\sigma|$ to denote the length of a call sequence, $\sigma[i]$ for the ith call of the sequence, $\sigma|i$ for the first i calls of the sequence, and σ_x for the subsequence of σ that only contains calls involving x.

For a given set of agents A, a *gossip state* is a pair (S, σ), where S is a secret relation and σ a call sequence. A gossip state is *initial* if $S = I$ and $\sigma = \epsilon$. In this contribution we only consider gossip states of the form (I, σ), in which case we omit I. Hence ϵ stands for the initial state (I, ϵ), and $ab; cd$ stands for $(I, ab; cd)$, etcetera.

Definition 1 (Language)**.** *For a given finite set of agents A the language $\mathcal{L}$ of protocol conditions is given by the following BNF:*

$$\begin{array}{lcl} \varphi & := & \top \mid S_a b \mid Cab \mid \neg\varphi \mid (\varphi \wedge \varphi) \mid K_a\varphi \mid [\pi]\varphi \\ \pi & := & ?\varphi \mid ab \mid (\pi; \pi) \mid (\pi \cup \pi) \mid \pi^* \end{array}$$

where a, b range over A. We have the usual abbreviations for implication, disjunction and for dual modalities, and often omit parentheses.

The atomic formula $S_a b$ reads as 'agent a has the secret of b'. The atomic formula Cab means that agent a has called agent b (in the past). The formula $K_a\varphi$ reads 'agent a knows that φ is true'. Expression $[\pi]\varphi$ reads as 'after executing the program π, φ is true'. We also define the abbreviation $E\varphi := \bigwedge_{a \in A} K_a\varphi$ and read it as 'everyone knows φ' ($E\varphi$ is also known as *shared knowledge* or *mutual knowledge* of φ). Program iteration is defined as: $\pi^0 :=?\top$, and for $n \geq 0$, $\pi^{n+1} := \pi^n; \pi$.

Agent a is an *expert* if she knows all the secrets, formally $\bigwedge_{b \in A} S_a b$, abbreviated as Exp_a. *Everyone is an expert* is represented by the formula $Exp_A := \bigwedge_{a \in A} \bigwedge_{b \in A} S_a b$. Agent a is a *super expert* if she knows that everyone is an expert, formally $K_a Exp_A$.

Definition 2 (Protocol)**.** *A protocol* P *is a program defined by*

$$\mathsf{P} := (?\neg EExp_A; \bigcup_{a \neq b \in A} (?\mathsf{P}_{ab}; ab))^*; ?EExp_A$$

where $\mathsf{P}_{ab} \in \mathcal{L}$ *is the* protocol condition *for call* ab *of protocol* P.

The difference with the usual definition of gossip protocol as in, e.g., [18], is that goal Exp_A is replaced by goal $EExp_A$. In other words, instead of "while not everyone is an expert, select two agents to make a call" we have "while not everyone is a super expert, select two agents to make a call."

Definition 3 (Epistemic relation)**.** *Let* $a \in A$. *The* synchronous epistemic relation $\approx_a$ *is the smallest equivalence relation between call sequences such that:*

- $\epsilon \approx_a \epsilon$
- *if* $\sigma \approx_a \tau$ *and* $a \notin \{b, c, d, e\}$, *then* $\sigma; bc \approx_a \tau; de$
- *if* $\sigma \approx_a \tau$ *and* $I_b^\sigma = I_b^\tau$, *then* $\sigma; ab \approx_a \tau; ab$
- *if* $\sigma \approx_a \tau$ *and* $I_b^\sigma = I_b^\tau$, *then* $\sigma; ba \approx_a \tau; ba$

The asynchronous epistemic relation $\sim_a$ *between call sequences is defined as the relation* $\approx_a$ *except that the second clause is replaced by*

- *if* $\sigma \sim_a \tau$, $a \notin \{b, c\}$, *then* $\sigma; bc \sim_a \tau$.

Informally, the synchronous accessibility relation encodes that agents not involved in a call are still aware that a call has taken place, as considered in [9, 10]. This also implies that all agents know how many calls have taken place, i.e., there is a global clock. The asynchronous accessibility relation does not make any such assumption. Then, agents are only aware of the calls in which they are involved. Any information on other calls has to be deduced from the secrets they obtain from their calling partners.

Both epistemic relations assume that the callers not only learn what the union is of the sets of secrets they each held before the call, but also learn what set of secrets the other agent held before the call. This is known as the "inspect-then-merge" form of observation [19].

Note that for any agent a, $\approx_a \subseteq \sim_a$. This is fairly obvious, because for any call sequences σ and τ and $b, c, d, e \neq a$: $\sigma \sim_a \tau$ implies $\sigma; bc \sim_a \tau$, which implies $\sigma; bc \sim_a \tau; de$. The latter copies the clause $\sigma; bc \approx_a \tau; de$ for the synchronous case.

Definition 4 (Semantics)**.** *Let call sequence* σ *and formula* $\varphi \in \mathcal{L}$ *be given. We define* $\sigma \models \varphi$ *by induction on* φ.

$$
\begin{array}{lll}
\sigma \models \top & \textit{iff} & \textit{true} \\
\sigma \models S_a b & \textit{iff} & I^\sigma ab \\
\sigma \models Cab & \textit{iff} & ab \in \sigma \\
\sigma \models \neg\varphi & \textit{iff} & \sigma \not\models \varphi \\
\sigma \models \varphi \wedge \psi & \textit{iff} & \sigma \models \varphi \textit{ and } \sigma \models \psi \\
\sigma \models K_a\varphi & \textit{iff} & \tau \models \varphi \textit{ for all } \tau \textit{ such that } \sigma \approx_a \tau \\
\sigma \models [\pi]\varphi & \textit{iff} & \tau \models \varphi \textit{ for all } \tau \textit{ such that } \sigma [\![\pi]\!] \tau
\end{array}
$$

where

$$\begin{array}{lll}
\sigma[\![?\varphi]\!]\tau & \textit{iff} & \sigma \models \varphi \textit{ and } \tau = \sigma \\
\sigma[\![ab]\!]\tau & \textit{iff} & \tau = \sigma; ab \\
\sigma[\![\pi;\pi']\!]\tau & \textit{iff} & \textit{there is } \rho \textit{ such that } \sigma[\![\pi]\!]\rho \textit{ and } \rho[\![\pi']\!]\tau \\
\sigma[\![\pi \cup \pi']\!]\tau & \textit{iff} & \sigma[\![\pi]\!]\tau \textit{ or } \sigma[\![\pi']\!]\tau \\
\sigma[\![\pi^*]\!]\tau & \textit{iff} & \textit{there is } n \in \mathbb{N} \textit{ such that } \sigma[\![\pi^n]\!]\tau
\end{array}$$

The inductive clause for $K_a\varphi$ above is for the synchronous setting. For the asynchronous setting we replace $\sigma \approx_a \tau$ by $\sigma \sim_a \tau$ in that clause. For simplicity we do not use a separate symbol for the asynchronous semantics — it will always be clear from the context what '$\models$' stands for. A formula φ is valid*, notation $\models \varphi$, iff for all call sequences σ we have $\sigma \models \varphi$.*

We assume that all our protocols are *symmetric*, which means that for all $a \neq b \in A$ and $c \neq d \in A$, simultaneously replacing a by c and b by d in the protocol condition P_{ab} yields P_{cd}. Intuitively, a symmetric protocol gives the same instructions and does not assign any special roles to individual agents. Moreover, we only consider protocols that are *epistemic*, which means that $\mathsf{P}_{ab} \to K_a\mathsf{P}_{ab}$ is valid. This means that agents always know which calls they are allowed to make (see [18, page 170]).

If in call ab agent a or b becomes an expert, then the other agent simultaneously becomes an expert, whereas if in a call ab agent a or agent b becomes a super expert, then the other agent need not also become a super expert.

We continue with terminology on protocol termination. In some of this subsequent terminology we informally consider infinite call sequences. We denote a potentially infinite call sequence as σ_∞.

If $\sigma \models \mathsf{P}_{ab}$ we say that call ab is P-*permitted* after σ. A P-permitted call sequence is a call sequence consisting of P-permitted calls. An infinite call sequence σ_∞ is P-permitted if for any $i \in \mathbb{N}$ prefix $\sigma_\infty|i$ is P-permitted.

A P-permitted sequence σ_∞ is P-*fair* iff either σ_∞ is finite or for all $x \neq y \in A$, if for all i there is $j > i$ such that xy is P-permitted in $\sigma_\infty|j$ then for all i there is $j > i$ such that $\sigma_\infty[j] = xy$. Intuitively, fairness means that eventually all calls are made arbitrarily often as long as they are permitted.

A call sequence σ is *super-successful* if after σ all the agents are super experts. A sequence σ is P-*maximal* (or P-*terminal*, or *terminating*) iff it is P-permitted and for any $x, y \in A$, $\sigma; xy$ is not P-permitted. An infinite call sequence σ_∞ is P-maximal iff any prefix $\sigma_\infty|i$ is P-permitted. A protocol P is *super-successful* iff all P-maximal sequences are super-successful (and thus finite). A protocol P is *fairly super-successful* iff all fair P-maximal sequences are super-successful. The notion of fairness is needed because already very simply protocols allow infinite call sequences.

Finally, a call sequence σ is *successful* iff after σ all the agents are experts. Also analogously to the previous terminology involving super-successful, we define *successful protocol* and *fairly successful protocol*.

2.2. Gossip protocols ANY, CMO, PIG and LNS

Four gossip protocols feature in this contribution. The protocol conditions are for any $a, b \in A$ with $a \neq b$.

- ANY with protocol condition $\mathsf{ANY}_{ab} := \top$;
- CMO with protocol condition $\mathsf{CMO}_{ab} := \neg Cab \wedge \neg Cba$;
- PIG with protocol condition $\mathsf{PIG}_{ab} := \hat{K}_a \bigvee_{c \in A}((S_a c \wedge \neg S_b c) \vee (\neg S_a c \wedge S_b c))$;
- LNS with protocol condition $\mathsf{LNS}_{ab} := \neg S_a b$.

The acronym ANY stands for *make ANY call* and is the standard (uninformed) protocol in the gossip literature [7] (not necessarily with the epistemic interpretation in our work). In any infinite *fair* ANY-sequence any call occurs arbitrarily often.

The acronym PIG stands for *Possible Information Growth*. Intuitively, the call ab is permitted if a considers it possible that: a will learn a secret c that b knows but not a, or that: b will learn a secret c that a knows but not b. It has been investigated in [10, 15]. Both ANY and PIG permit infinite call sequences.

The acronym CMO stands for *Call Me Once*. You are allowed to call an agent if you have not yet been involved in a call with that agent. This protocol was introduced in [16] and is reminiscent of [20]. As any two out of n agents are only allowed to have a call once, the maximum number of calls in CMO is $\binom{n}{2} = \frac{n(n-1)}{2}$.

The acronym LNS stands for *Learn New Secrets*. A call ab is LNS-permitted iff agent a does not know the secret of agent b [10, 15, 16]. This protocol is traditionally known as NOHO, for *No One Hears Own* [6]. Both CMO and LNS only permit finite call sequences.

If we identify a protocol P with its extension (the set of P-permitted call sequences), we note that $\mathsf{LNS} \subset \mathsf{CMO} \subset \mathsf{ANY}$ and that $\mathsf{PIG} \subset \mathsf{ANY}$. For the expert goal we additionally have $\mathsf{CMO} \subset \mathsf{PIG}$ [15, Prop. 53]. We will see later (Corollary 37) that this no longer holds for the super expert goal.

Already with a merely strengthened epistemic goal and without the more involved semantics in subsequent sections we can obtain novel results for gossip protocols, on which we will now report: ANY and PIG are fairly super-successful. CMO and LNS are not super-successful, and for those protocols we only have interesting results with more involved semantics. We will therefore only report on results for these protocols later.

2.3. Results for the protocol ANY

The first result is fairly obvious, but proved for good measure.

Proposition 5. ANY *is fairly super-successful.*

Proof. As long as $EExp_A$ does not hold, any call xy is ANY-permitted. The argument is therefore as usual for fair executions.

Let σ_∞ be a (possibly infinite) fair maximal ANY-permitted sequence. Towards a contradiction suppose we do not have $EExp_A$ after any finite prefix of σ_∞. Consider the following two cases.

- The sequence σ_∞ is finite. Let x be an agent who is not a super expert after σ_∞. Then there must be an agent $y \neq x$ such that x is uncertain whether y is an expert. The call xy is ANY-permitted after σ_∞. This contradicts the maximality of σ_∞.
- The sequence σ_∞ is infinite. Then there is a finite prefix $\tau \sqsubset \sigma_\infty$ such that for all sequences $\tau \sqsubseteq \rho \sqsubset \sigma_\infty$ no further secrets are learned after τ, i.e. $I^\tau = I^\rho$. Consider the following two cases.
  - $I^\tau \neq A^2$. Then there are $x, y \in A$ such that $y \in A \setminus I^\tau_x$. So the call xy is ANY-permitted after τ but it is not executed in σ_∞. This contradicts the fairness assumption.
  - $I^\tau = A^2$. Then there are $x, y \in A$ such that after every prefix of σ_∞, agent x does not know that y is expert. This means for any sequence ρ with $\tau \sqsubseteq \rho \sqsubset \sigma_\infty$ there is a sequence π such that ρ is indistinguishable from π for agent x (either $\rho \sim_x \pi$ or $\rho \approx_x \pi$) and $A^2 = I^\rho_x = I^\pi_x \neq I^\pi_y$. As call xy is ANY-permitted after both (indistinguishable sequences) ρ and π but is never executed, again this contradicts the fairness assumption. □

Example 6. *Let $A = \{a, b, c\}$, and let the protocol be asynchronous* ANY*. We show that after call sequence $ab; ac; ab; cb$ it holds that $EExp_A$.*

- *After the prefix $ab; ac$, agents a and c are experts.*
- *After the prefix $ab; ac; ab$, agents a and b are super experts.*

  Agent a already knew that c is an expert and in call ab also learns that b now is an expert. Therefore, she is a super expert: $ab; ac; ab \models K_a Exp_A$.

  In the third call, ab, agent b learns that a is an expert. Because in the first call ab agent a did not know the secret of c yet, but now gives it to b, agent b can infer that the call ac must have taken place between the two ab calls. As in that call ac agent c became an expert, agent b also knows that agent c is an expert. Therefore also agent b is a super expert.
- *Now consider the entire sequence $ab; ac; ab; cb$. In final call cb, agent c becomes a super expert. After the second call, ac, agent a is an expert, hence c knows this. After the last call cb agent b is an expert, hence c also knows this. Therefore agent c knows that all agents are experts.*

Example 7. *Let now $A = \{a, b, c, d\}$, let the protocol be aynchronous* ANY*. A super-successfully terminating sequence $ab; cd; ac; bd; ab; ad; bc; cd$ consisting of eight calls was already given in the introductory Section 1.*

It is easy to see that for n agents after $2n - 3$ calls an agent can be a super expert, both in the synchronous and in the asynchronous case.

Example 8. *Let again $A = \{a, b, c, d\}$. We show that after the five call sequence $ab; cd; ac; bd; ba$ agent b is a super expert.*

After prefix $ab; cd; ac; bd$ agent b is an expert. Agent b does not know what the second and third calls were, but he knows that no call between a and d took place. However, he is uncertain whether agent a is an expert. For example, an alternative sequence considered possible by b is $ab; cd; cd; bd$. This uncertainty is resolved in the fifth call.

Now consider the sequence $ab; cd; ac; bd; ab$. This reveals to b that a must have been involved in the second or third call of the sequence. As in the fourth call bd agent b learns that d has been involved in a call but did not yet know the secret of a, b learns that this cannot have been the second call. As a is already an expert in the call ab, this reveals that the third call must have been between a and c. Agent b now only consider possible the sequence $ab; cd; ac; bd; ab$ (where the calls not involving him could also have been in the other direction). Therefore, agent b knows that all agents are experts.

There is however a far more straightforward way to become a super expert.

Example 9. *Let there be n agents. Let an agent call other agents in succession. (These are $n - 1$ calls.) Let that agent call all other agents again in succession except the last one. (These are $n - 2$ calls.) Then this agent is now a super expert. (Altogether, these are $(n - 1) + (n - 2) = 2n - 3$ calls.) An example for 4 agents is the 5 call sequence $ab; ac; ad; ab; ac$. The call ad is not needed for the second time, as in call ad agent d already became an expert.*

Conjecture 10. *For n agents, the minimum number of calls for an agent to become a super expert is $2n - 3$.*

The basis for this conjecture is that merely one less call, $2n - 4$, is the minimum number of calls for all agents to become experts [2]. Given that, a natural question to ask is whether, independently from minima, an agent can become an expert and a super expert in the same call, which seems unlikely. But in fact this is possible, at least for synchronous ANY. We do not know if it is possible for asynchronous ANY.

Example 11. *Consider 4 agents, synchronous* ANY *and $ab; ac; cd; ab; bc; ab$. In the final call, agent a becomes an expert and a super expert. See Table 2. This sequence was found after an exhaustive search with the model checker GoMoChe.*

Proposition 12. *Synchronous* ANY *permits shorter super-successful sequences than asynchronous* ANY*.*

Proof. We have not proved that for any $n \geq 3$ a shorter super-successful sequence exists. However, for a given 'small' number of agents it is straightforward to come up with such a shorter execution sequence by model checking.

First, consider 3 agents a, b, c and recall the minimal super-successful asynchronous call sequence $ab; ac; ab; cb$ of Example 6. The prefix $ab; ac; ab$ is already synchronously super-successful. Agent c is not involved in the third call, and

	a		b		c		d		initial state
$\xrightarrow{ab}$	ab		ab		c		d		
$\xrightarrow{ac}$	abc		ab		abc		d		
$\xrightarrow{cd}$	abc		ab		abcd	CD	abcd	CD	
$\xrightarrow{ab}$	abc	CD	abc		abcd	CD	abcd	CD	
$\xrightarrow{bc}$	abc	CD	abcd	BCD	abcd	BCD	abcd	CD	
$\xrightarrow{ab}$	abcd	ABCD	abcd	ABCD	abcd	BCD	abcd	CD	a is expert and super expert

Table 2: Results of $ab; ac; cd; ab; bc; ab$. Each column describes what an agent knows: a lower case y in the column of x means $S_x y$; an upper case Y means $K_x Exp_y$. Therefore, "abcd" denotes an expert and "ABCD" denotes a super expert.

this is common knowledge to all agents. In fact, all three agents only consider this sequence $ab; ac; ab$ possible.

Let there now be 4 agents a, b, c, d, as in the introductory Section 1 where we discussed an 8 call super-successful sequence $ab; cd; ac; bd; ab; ad; bc; cd$. We can reach $EExp_A$ in only seven calls, namely with sequence:

$$ab; cd; ac; ad; bc; ba; bd$$

The reasoning was validated by the model checker GoMoChe, and what agents learn in these calls is shown in Table 3. Let us sketch the justification of these results.

After prefix $ab; cd; ac; ad$ we have three experts a, c and d. In the fifth call bc agent b becomes an expert (similarly to Example 8), and as usual b and c learn about each other that they are experts. In addition, and somewhat surprisingly, c also learns in that call that d is an expert. This is due to synchrony and can be checked as follows: c knows that between the third call ac and the fifth call bc there must have been a call which must have between between a and d or between a and b. But in the fifth call bc agent b only knows the secrets of a and b, hence this fourth call did not involve b. Therefore, it must have involved d, which implies that d is an expert. (See Table 3).

Note that agent c only became a super expert in call bc because of synchrony, and that c is not involved in calls after that, and therefore asynchronously considers it possible that bc was the last call. Therefore, this seven-call sequence is not super-successful asynchronously.

Of course, there could be other call sequences of at most 7 calls that are asynchronously super-successful. This has been ruled out by exhaustive search in the model checker GoMoChe. □

2.4. Results for the protocol PIG

The PIG protocol has infinite executions for four or more agents [15]. Sequence $ab; ab; ab; \ldots$ is asynchronous PIG-permitted. Call ab is indistinguishable

	a	b	c	d	initial state
$\stackrel{ab}{\rightarrow}$	ab	ab	c	d	
$\stackrel{cd}{\rightarrow}$	ab	ab	cd	cd	
$\stackrel{ac}{\rightarrow}$	abcd A C	ab	abcd A C	cd	
$\stackrel{ad}{\rightarrow}$	abcd A CD	ab	abcd A C	abcd A CD	
$\stackrel{bc}{\rightarrow}$	abcd A CD	abcd BC	abcd ABCD	abcd A CD	$K_c Exp_A$
$\stackrel{ba}{\rightarrow}$	abcd ABCD	abcd ABC	abcd ABCD	abcd A CD	
$\stackrel{bd}{\rightarrow}$	abcd ABCD	abcd ABCD	abcd ABCD	abcd ABCD	$EExp_A$

Table 3: Results of $ab; cd; ac; ad; bc; ba; bd$.

for agent a from call sequence $ab; bc$, after which agent b has learnt something new. Thus, after first call ab, the same call ab is again PIG-permitted. Similarly, $ab; ab \sim_a ab; ab; bc$, thus ab is again PIG-permitted after $ab; ab$, and so on. Somewhat similarly, under synchronous conditions, the sequence $ab; cd; ab; cd; ab; cd; \ldots$ is PIG-permitted, as after any even number of calls agent a considers it possible that agent b was involved in the previous call and would thus have learnt a new secret in that call. Therefore, each odd call can again be call ab. Termination results for the PIG protocol are therefore restricted to fair call sequences. These results are not as obvious as for ANY, given the protocol condition $\mathsf{PIG}_{ab} := \hat{K}_a \bigvee_{c \in A}((S_a c \wedge \neg S_b c) \vee (\neg S_a c \wedge S_b c))$.

Lemma 13. $\bigvee_{a,b \in A} \mathsf{PIG}_{ab} \leftrightarrow \neg EExp_A$ *is valid.*

Proof. Assume $\bigvee_{a,b \in A} \mathsf{PIG}_{ab}$. If an agent a considers it possible that there is a secret that is not known by another agent b or by herself, then she considers it possible that that other agent or herself is not an expert: $\neg K_a \neg\neg Exp_b \vee \neg K_a \neg\neg Exp_a$. Either way, she then does not know that all agents are experts, $\neg K_a Exp_A$, and therefore $\neg EExp_A$. The other direction is similar. □

Lemma 13 might seem to suggest that ANY and PIG have the same extension. But this is false. Not all ANY permitted call sequences are PIG permitted (and this does not depend on whether the goal is for all to become expert or for all to become super expert). Let in call sequence $\tau; ab$ agents a and b become expert in that final call ab, then ab is not PIG permitted in any extension of $\tau; ab$, whereas ab remains ANY permitted. However, for a certain strengthening of the semantics to be presented in Section 4, this difference in extension disappears (Proposition 35).

Proposition 14. PIG *is fairly super-successful.*

Proof. The proof of this proposition is the same that of Proposition 5, because as long as $EExp_A$ does not hold, any call xy is not only ANY-permitted but also PIG-permitted. This follows from Lemma 13. We therefore omit proof details. □

Example 15. *The call sequence $\sigma = ab; cd; ac; bd; ab; ad; cb; cd$ from introductory Section 1 is also* PIG*-permitted. We can adapt σ to get a successful* ANY*-permitted sequence that is not* PIG*-permitted: in σ, repeat penultimate call cb before final call cd, i.e., with the additional call in bold, $ab; cd; ac; bd; ab; ad; cb; \mathbf{cb}; cd$.*

3. Common knowledge of gossip protocols

3.1. Syntax and semantics — known protocols

We now enrich the framework by modelling common knowledge of protocols. This requires that we replace the knowledge modality by a knowledge modality depending on a given protocol, and that we replace the epistemic relations by more restricted relations incorporating common knowledge of the protocols (it is a restriction as this reduces the uncertainty about call sequences). The resulting semantic framework is more complex, because these definitions require mutual recursion both in the syntax and in the semantics. In the syntax, because what an agent knows now depends on a given protocol, whereas the protocol is defined with respect to a protocol condition, that could be a knowledge formula, that needs to be evaluated in the semantics. Similarly, in the semantics, the epistemic relation (that interprets a knowledge modality) depends on a given protocol, and thus on the interpretation of the protocol conditions: formulas, so we are back in the syntax. We adapt the framework presented in [18] to our needs.

Definition 16 (Language and Protocol — known protocols)**.** *In the BNF of the language $\mathcal{L}$ we replace the inductive clause $K_a\varphi$ by an inductive clause $K_a^{\mathsf{P}}\varphi$. For $\bigwedge_{a\in A} K_a^{\mathsf{P}}\varphi$ we write $E^{\mathsf{P}}\varphi$. Then, a protocol P is a now a program defined by*

$$\mathsf{P} := (?\neg E^{\mathsf{P}}Exp_A; \bigcup_{a\neq b\in A} (?\mathsf{P}_{ab}; ab))^*; ?E^{\mathsf{P}}Exp_A$$

Formula $K_a^{\mathsf{P}}\varphi$ means that agent a knows φ given (common knowledge between all agents of) protocol P. So, $E^{\mathsf{P}}Exp_A$ means that everyone is a super expert given protocol P. We call $K_a^{\mathsf{P}}\varphi$ *protocol dependent knowledge* (of φ).

We now define $\approx_a^{\mathsf{P}}$ and $\sim_a^{\mathsf{P}}$, simultaneously with the satisfaction relation $\models$. The only change for the known protocol version with respect to the prior Definition 4 of $\models$, is that we replace K_a by K_a^{P} everywhere and $\approx_a$ by $\approx_a^{\mathsf{P}}$ everywhere (and similarly for $\sim_a$). Only the knowledge clause of the semantics is therefore given.

Definition 17 (Epistemic relations and semantics — known protocols)**.**
Let $a \in A$. The synchronous accessibility relation $\approx_a^{\mathsf{P}}$ between call sequences is the smallest symmetric and transitive relation such that:

- $\epsilon \approx_a^{\mathsf{P}} \epsilon$,
- *if $\sigma \approx_a^{\mathsf{P}} \tau$, $a \notin \{b, c, d, e\}$, $\sigma \models \mathsf{P}_{bc}$ and $\tau \models \mathsf{P}_{de}$ then $\sigma; bc \approx_a^{\mathsf{P}} \tau; de$*
- *if $\sigma \approx_a^{\mathsf{P}} \tau$, $I_b^\sigma = I_b^\tau$, $\sigma \models \mathsf{P}_{ab}$ and $\tau \models \mathsf{P}_{ab}$, then $\sigma; ab \approx_a^{\mathsf{P}} \tau; ab$*

- *if* $\sigma \approx_a^{\mathsf{P}} \tau$, $I_b^\sigma = I_b^\tau$, $\sigma \models \mathsf{P}_{ba}$ *and* $\tau \models \mathsf{P}_{ba}$, *then* $\sigma; ba \approx_a^{\mathsf{P}} \tau; ba$

The asynchronous accessibility relation $\sim_a^{\mathsf{P}}$ *between call sequences is the same as the relation* $\approx_a^{\mathsf{P}}$ *except that the second clause is replaced by*

- *if* $\sigma \sim_a^{\mathsf{P}} \tau$, $a \notin \{b, c\}$, *and* $\sigma \models \mathsf{P}_{bc}$, *then* $\sigma; bc \sim_a^{\mathsf{P}} \tau$

Finally, in the inductive definition of $\models$ *we replace the clause for* $K_a\varphi$ *by:*

$$\sigma \models K_a^{\mathsf{P}}\varphi \quad \textit{iff} \quad \tau \models \varphi \textit{ for all } \tau \textit{ such that } \sigma \approx_a^{\mathsf{P}} \tau$$

On the set of P-permitted call sequences the relations $\approx_a^{\mathsf{P}}$ and $\sim_a^{\mathsf{P}}$ are equivalence relations, but not on the set of all call sequences: see below and see also [18].

For $K_a^{\mathsf{ANY}}\varphi$ we write $K_a\varphi$, for $\approx_a^{\mathsf{ANY}}$ we write $\approx_a$ and for $\sim_a^{\mathsf{ANY}}$ we write $\sim_a$. This is not ambiguous, because if for all $a, b \in A$, $\mathsf{P}_{ab} = \top$, we regain the syntax and semantics of the previous Section 2.

In Definition 16 of the version of the language and the protocols assuming commonly known protocols, formula $K_a^{\mathsf{P}}\varphi$ contains as parameter a protocol P, and vice versa a protocol P contains protocol conditions P_{ab} that are formulas. This is well-defined, once we see $K_a^{\mathsf{P}}\varphi$ as $K_a(X, \varphi)$ where X is the list of formulas P_{xy} for $x \neq y \in A$, in other words, as a modality with not a single argument φ, but with $\binom{|A|}{2} + 1$ arguments.[2] For formal precision, in the Appendix (page 32) we give the well-founded preorder demonstrating that the semantics is well-defined. As also discussed at length in [18], this excludes self-referential protocols.

Protocol P with the syntax and semantics for common knowledge of protocols is referred to as *known* P.

We list some elementary properties of the semantics below, but refer to [18] for further discussion and proofs. Here, $a, b \in A$, protocols P, P', and $\varphi \in \mathcal{L}$ are all arbitrary.

- $\models K_a^{\mathsf{P}}\varphi \to K_a^{\mathsf{P}}K_a^{\mathsf{P}}\varphi$, and $\models \neg K_a^{\mathsf{P}}\varphi \to K_a^{\mathsf{P}}\neg K_a^{\mathsf{P}}\varphi$. Intuitively, K_a^{P} has two of the standard properties of knowledge, namely positive and negative introspection.
- $\not\models K_a^{\mathsf{P}}\varphi \to \varphi$. Whenever σ is not P-permitted, then $\sigma \models K_a^{\mathsf{P}}\bot$. In other words, if you are in violation of the protocol, anything goes. However, whenever σ is P-permitted, then $\sigma \models K_a^{\mathsf{P}}\varphi \to \varphi$.
- $\models \mathsf{P}_{ab} \to \mathsf{P}'_{ab}$ implies $\models K_a^{\mathsf{P}'}\varphi \to K_a^{\mathsf{P}}\varphi$; as $K_a^{\mathsf{ANY}}\varphi = K_a\varphi$, for all $a, b \in A$, $\mathsf{ANY}_{ab} = \top$ and $\psi \to \top$ is valid for all ψ, a corollary is that $\models K_a\varphi \to K_a^{\mathsf{P}}\varphi$.
- $\models S_a b \leftrightarrow K_a^{\mathsf{P}}S_a b$ and $\models \neg S_a b \leftrightarrow K_a^{\mathsf{P}}\neg S_a b$. Whether a knows the secret of b can be determined from the call sequence and independently from the protocol.

[2] So, despite its notation, we should *not* see $K_a^{\mathsf{P}}\varphi$ as constructed from P and φ. One should rather see P as shorthand for a function $\mathsf{P} : A \times A \to \mathcal{L}$, where image $\mathsf{P}(A \times A)$ is the list X.

3.2. Results for the protocol CMO and minor other results

Common knowledge of the protocol ANY does not make any difference, as the previous syntax and semantics is the special case for $\mathsf{P}_{xy} = \top$ for all agents $x \neq y \in A$. Minor results for PIG, LNS will be discussed in relation to results for CMO, that we will therefore present first. We recall that PIG is slightly more restrictive than ANY.

For the protocol CMO, whether the agents know that CMO is executed makes a big difference. It is the difference between being super-successful or not.

Proposition 18. *Synchronous (not commonly known)* CMO *is not super-successful.*

Proof. There are counterexamples whenever $|A| \geq 4$.

Given $A = \{a_1, a_2, \ldots, a_n\}$, let ρ be a maximal CMO-permitted sequence between agents $\{a_1, a_2, \ldots, a_{n-1}\}$. From [16] it follows that after ρ all agents $a_1, a_2, \ldots, a_{n-1}$ know all their secrets. So they are all experts except that none knows the secret of a_n. Now define the call sequence σ by having agent a_n call all other agents after ρ:

$$\sigma := \rho; a_n a_1; a_n a_2; \ldots; a_n a_{n-1}$$

We note that σ is again a maximal CMO sequence, as $\binom{n-1}{2} + (n-1) = \binom{n}{2}$. After σ, all agents are experts, and agent a_n is the only super expert. Let $i, j < n$ and $i \neq j$. Now consider the following call sequence τ where a_n only calls a_j (many times) and a_i (once, at the same moment as in σ):

$$\tau := \rho; \overbrace{a_n a_j; a_n a_j; \ldots a_n a_j}^{i-1 \text{ times}}; a_n a_i; \overbrace{a_n a_j; a_n a_j \ldots a_n a_j}^{n-i-1 \text{ times}}$$

We then have that $\sigma \approx_{a_i} \tau$ and that $\tau \not\models \mathit{Exp}_A$. Therefore, $\sigma \models \neg K_{a_i} \mathit{Exp}_A$. As σ is maximal and not super-successful, CMO is not super-successful. □

Proposition 19. *Asynchronous* CMO *is not super-successful.*

Proof. There are counterexamples whenever $|A| \geq 4$.

Consider again the call sequence ρ and σ from the proof of Theorem 18. The sequence $\rho; a_n a_i$ is CMO-permitted, and $\sigma \sim_{a_i} \rho; a_n a_i$. After $\rho; a_n a_i$, only agents a_n and a_i are experts but none of the remaining agents. Therefore, $\sigma \not\models K_{a_i} \mathit{Exp}_A$, so $\sigma \not\models E\mathit{Exp}_A$. As σ is maximal and not super-successful, CMO is not super-successful.

It does not matter whether CMO is known, as we also have $\sigma \sim_{a_i}^{\mathsf{CMO}} \rho; a_n a_i$. □

Example 20. *Consider the semantics without protocol knowledge. Let* $A = \{a, b, c, d\}$ *and consider the sequence* $\sigma := ab; ac; bc; ad; db; dc$. *This sequence is* CMO*-permitted,* CMO*-maximal, and satisfies* Exp_A.

Observe that $\sigma \approx_b ab; ac; bc; ad; db; ad$, *where in the call sequence on the right side we replaced the final call dc in* σ *by ad. This sequence is not* CMO*-permitted, as call ad occurs twice. After* $ab; ac; bc; ad; db; ad$, *agent c does not*

know the secret of d, therefore $ab; ac; bc; ad; db; ad \not\models Exp_A$. *From that and* $\sigma \approx_b ab; ac; bc; ad; db; ad$ *then follows that* $\sigma \not\models K_b Exp_A$, *and therefore* $\sigma \not\models EExp_A$, *so that* σ *is not super-successful.*

Example 21. *Consider again call sequence* σ *from the previous Example 20. Now assume asynchrony. Consider the prefix* $ab; ac; bc; ad$ *of* σ. *Note that* $\sigma \sim_a ab; ac; bc; ad$, *as a is not involved in the final two calls. Observe that after* $ab; ac; bc; ad$ *agents b and c do not know the secret of d (*$ab; ac; bc; ad \models \neg S_b d \wedge \neg S_c d$*), so that* $ab; ac; bc; ad \not\models Exp_A$. *From that and* $ab; ac; bc; ad; db; dc \sim_a ab; ac; bc; ad$ *it follows that* $\sigma \not\models K_a Exp_A$, *which implies* $\sigma \not\models EExp_A$, *so that again* σ *is not super-successful.*

We only used CMO*-permitted call sequences in the argument. It therefore also demonstrates that known* CMO *is not super-successful (as reported in Proposition 19).*

We will now show that known CMO is super-successful.

Theorem 22. *Synchronous known* CMO *is super-successful.*

Proof. The extension of CMO consists of finite call sequences of length at most $\binom{n}{2}$. Consider a maximal CMO call sequence σ. If $|\sigma| < \binom{n}{2}$, then it satisfies $E^{\mathsf{CMO}} Exp_A$ (otherwise it would not be maximal, as there are still CMO-permitted calls) so it is super-successful. Otherwise $|\sigma| = \binom{n}{2}$. We now use that CMO is successful, i.e., for goal Exp_A [16]. As there are no call sequences of length greater than $\binom{n}{2}$, and as CMO is successful, all sequences of length $\binom{n}{2}$ satisfy Exp_A. As the setting is synchronous, given σ, all agents only consider call sequences of that length. Therefore, regardless of the epistemic relations, they only consider call sequences satisfying Exp_A. Therefore $E^{\mathsf{CMO}} Exp_A$: σ is super-successful. □

Example 23. *This example features synchronous known* CMO. *The results in this example have been validated with the model checker GoMoChe. They are displayed in Tables 4 and 5, and in Figure 1.*

Given four agents a, b, c, d, *we always reach* $E^{\mathsf{CMO}} Exp_A$ *in five calls when the first two calls have no overlap, as in* $ab; cd$. *The only* CMO*-permitted call that has then not yet been made is ad.*

Given synchrony it is not always obvious how agents not involved in a call learn that agents become super experts in that call. We will therefore justify in detail how this may come to pass for some agents.

For example, in third call bd agent c learns that d becomes a super expert. This is because in the second call cd, agent c learns that the first call was ab, and as c is not involved in the third call, this must be one of ab, ad, bd *(or the dual call). As c knows that ab has already taken place, the third call must therefore have been between a and d or between b and d. This always involves d, and d then always becomes an expert. Therefore, c knows that d is an expert.*

Similarly, in the fifth call bc, agent d becomes a super expert (and in particular learns that a is an expert), because d knows that the two remaining

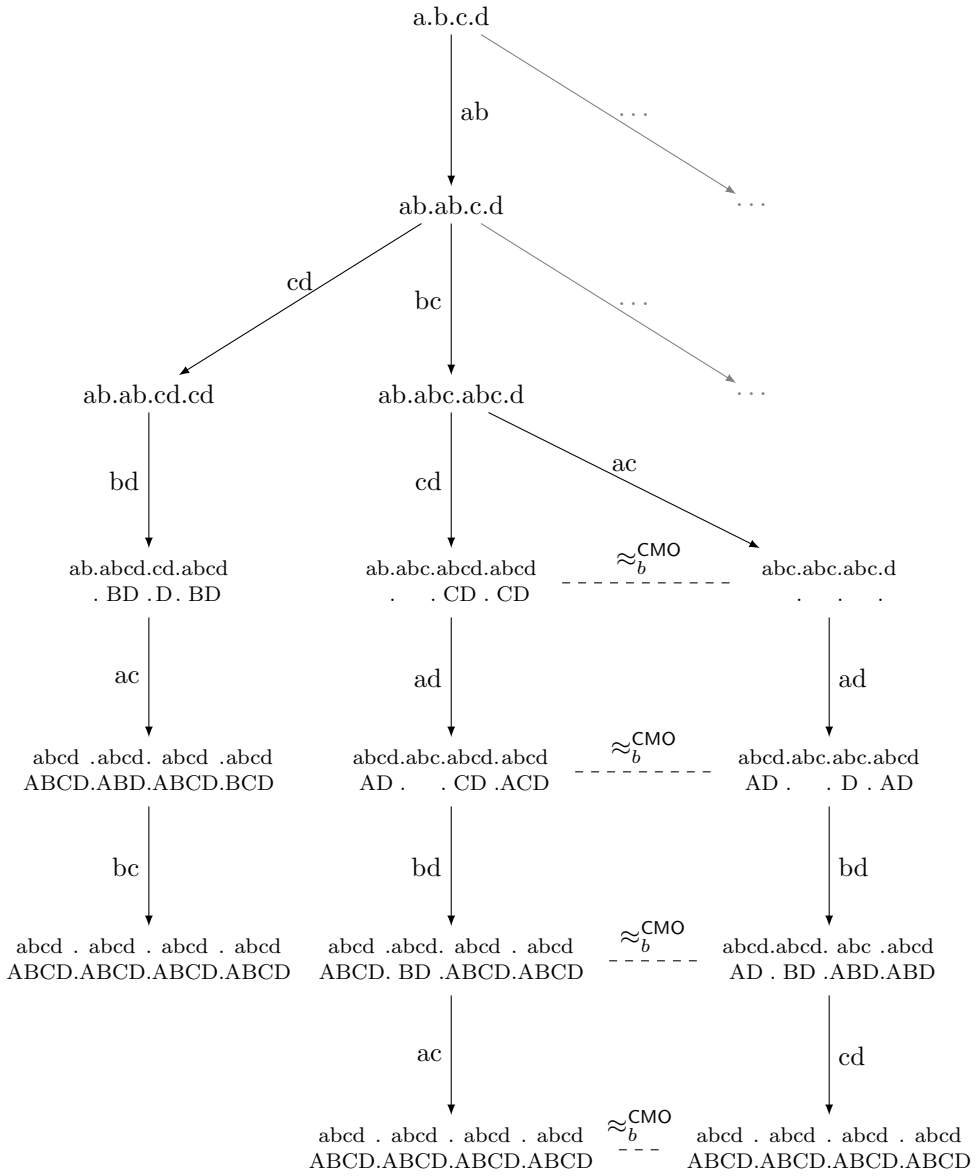

Figure 1: A **partial** view of the CMO execution tree for four agents. If the first two calls are disjoint, termination is (always) after five calls. Otherwise, it is (always) after six calls. Two other branches are suggested at depths 0 and 1 of the tree, but most other branches are not depicted. In particular, after $ab; bc$ call bd (or db) can be made, so that the same agent, b, occurs in the first three calls. Such a sequence therefore also terminates after six calls.

CMO-*permitted calls were* bc *and* ad*. As* d *was not involved,* d *knows that the call was* bc*.*

	a	b	c	d
$\xrightarrow{ab}$	ab	ab	c	d
$\xrightarrow{cd}$	ab	ab	cd	cd
$\xrightarrow{bd}$	ab	abcd B D	cd D	abcd B D
$\xrightarrow{ac}$	abcd ABCD	abcd AB D	abcd ABCD	abcd BCD
$\xrightarrow{bc}$	abcd ABCD	abcd ABCD	abcd ABCD	abcd ABCD

Table 4: The results of $ab; cd; bd; ac; bc$.

However, if we start with overlapping calls $ab; bc$*, then* $E^{\mathsf{CMO}}Exp_A$ *is only reached after* 6 *calls. For example, consider the sequence* $ab; bc; cd; ad; bd; ca$*. After this sequence everyone is a super expert. We show the results of this sequence in Table 5.*

	a	b	c	d
$\xrightarrow{ab}$	ab	ab	c	d
$\xrightarrow{bc}$	ab	abc	abc	d
$\xrightarrow{cd}$	ab	abc	abcd CD	abcd CD
$\xrightarrow{ad}$	abcd A D	abc	abcd CD	abcd A CD
$\xrightarrow{bd}$	abcd ABCD	abcd B D	abcd ABCD	abcd ABCD
$\xrightarrow{ac}$	abcd ABCD	abcd ABCD	abcd ABCD	abcd ABCD

Table 5: The results of $ab; bc; cd; ad; bd; ac$.

After the five calls $ab; bc; cd; ad; bd$ *agent* b *considers* $ab; bc; ac; ad; bd$ *possible, after which* c *is not an expert. But* b *has already been in a call with each other agent, and hence* b *is no longer* CMO*-permitted to make calls. However, call* ac *has not yet been made. Although agent* a *is a super expert, call* ac *is* CMO*-permitted, after which the protocol terminates super-successfully.*

There is more to observe from the CMO-permitted final call ac in the sequence $ab; bc; cd; ad; bd; \mathbf{ac}$ in Example 23. Final call ac is not LNS-permitted, as agent a is an expert (and ca is also not LNS-permitted). The sequence $ab; bc; cd; ad; bd$ without that final ac is LNS-maximal and not super-successful. This is because $ab; bc; cd; ad; bd \approx_b^{\mathsf{LNS}} ab; bc; ac; ad; bd$. Therefore, agent b considers $ab; bc; ac; ad; bd$ possible after which c is not an expert.

Corollary 24. *Synchronous known* LNS *is not super-successful.*

4. Agents not answering or making calls

4.1. Syntax and semantics — engaged agents

In the first place we now model that agents who are super experts do not *make* calls. We do this by changing the definition of gossip protocol and the epistemic relation. The condition that needs to be satisfied for an agent to be permitted to call is now that the agent is not a super expert.

In the second place we also model that agents who are super experts do not *answer* calls. We do that by changing the definition of the epistemic relation. A call sequence cannot be extended with a call made by an agent who already is a super expert.

Agents who neither make nor answer calls are called *engaged agents* (as in 'engaged in other activities' for the former and as in 'the line is engaged' for the latter). A call that is not answered is a *missed call*.

Definition 25 (Protocol — engaged agents)**.** *A protocol* P *is a program defined by*

$$\mathsf{P} := (\bigcup_{a\neq b\in A} (?(\neg K_a^{\mathsf{P}} Exp_A \wedge \mathsf{P}_{ab}); ab))^*; ?E^{\mathsf{P}} Exp_A$$

where for all $a \neq b \in A$, $\mathsf{P}_{ab} \in \mathcal{L}$ *is the* protocol condition *for call* ab *of protocol* P.

This protocol definition is different from the previous Definitions 2 and 16 but also different from the usual definition (e.g., [18]):

$$(?\neg Exp_A; \bigcup_{a\neq b\in A} (?\mathsf{P}_{ab}; ab))^*; ?Exp_A$$

As our termination condition is stronger, we already replaced "while not everyone is an expert" by "while not everyone is a super expert" and the protocol becomes Definition 25:

$$(?\neg E^{\mathsf{P}} Exp_A; \bigcup_{a\neq b\in A} (?\mathsf{P}_{ab}; ab))^*; ?E^{\mathsf{P}} Exp_A$$

Then, as we do not want super experts to make calls, we strengthen the protocol condition by adding $\neg K_a^P Exp_A$ to it:

$$(?\neg E^{\mathsf{P}} Exp_A; \bigcup_{a\neq b\in A} (?(\neg K_a^{\mathsf{P}} Exp_A \wedge \mathsf{P}_{ab}); ab))^*; ?E^{\mathsf{P}} Exp_A$$

Finally, as $\bigwedge_{a\in A} K_a^{\mathsf{P}} Exp_A$ is $E^{\mathsf{P}} Exp_A$, it is not hard to see that the same call sequences are allowed when we remove the first test on $\neg E^{\mathsf{P}} Exp_A$, which leads to the above Definition 16.

We continue with the changed epistemic relations. The definition of the semantic relation $\models$ remains the same.

Definition 26 (Epistemic relation — engaged agents)**.**
Let $a \in A$. The synchronous accessibility relation $\approx_a^{\mathsf{P}}$ between call sequences is the smallest symmetric and transitive relation such that:

- $\epsilon \approx_a^{\mathsf{P}} \epsilon$,
- *if $\sigma \approx_a^{\mathsf{P}} \tau$, $a \notin \{b,c,d,e\}$, $\sigma \models \neg K_b^{\mathsf{P}} Exp_A \wedge \mathsf{P}_{bc}$ and $\tau \models \neg K_d^{\mathsf{P}} Exp_A \wedge \mathsf{P}_{de}$ then $\sigma; bc \approx_a^{\mathsf{P}} \tau; de$*
- *if $\sigma \approx_a^{\mathsf{P}} \tau$, $I_b^\sigma = I_b^\tau$, $\sigma \models \neg K_a^{\mathsf{P}} Exp_A \wedge \mathsf{P}_{ab}$, $\tau \models \neg K_a^{\mathsf{P}} Exp_A \wedge \mathsf{P}_{ab}$, and ($\sigma \models K_b^{\mathsf{P}} Exp_A$ iff $\tau \models K_b^{\mathsf{P}} Exp_A$), then $\sigma; ab \approx_a^{\mathsf{P}} \tau; ab$*
- *if $\sigma \approx_a^{\mathsf{P}} \tau$, $I_b^\sigma = I_b^\tau$, $\sigma \models \neg K_b^{\mathsf{P}} Exp_A \wedge \mathsf{P}_{ba}$, $\tau \models \neg K_b^{\mathsf{P}} Exp_A \wedge \mathsf{P}_{ba}$, and ($\sigma \models K_a^{\mathsf{P}} Exp_A$ iff $\tau \models K_a^{\mathsf{P}} Exp_A$), then $\sigma; ba \approx_a^{\mathsf{P}} \tau; ba$*

The asynchronous accessibility relation $\sim_a^{\mathsf{P}}$ between gossip states is as the relation $\approx_a^{\mathsf{P}}$ except that the second clause is replaced by

- *if $\sigma \sim_a^{\mathsf{P}} \tau$, $a \notin \{b,c\}$, and $\sigma \models \neg K_b^{\mathsf{P}} Exp_A \wedge \mathsf{P}_{bc}$, then $\sigma; bc \sim_a^{\mathsf{P}} \tau$*

In the first place, the above definitions incorporate that agents no longer make calls once they are super experts. This is the part $\neg K_a^{\mathsf{P}} Exp_A$ in the definition of protocol, and the parts $\neg K_a^{\mathsf{P}} Exp_A$ and $\neg K_b^{\mathsf{P}} Exp_A$ in respectively the third and fourth item of Definition 26 of the epistemic relation.

In the second place, the extra conditions "$\sigma \models K_b^{\mathsf{P}} Exp_A$ iff $\tau \models K_b^{\mathsf{P}} Exp_A$" and "$\sigma \models K_a^{\mathsf{P}} Exp_A$ iff $\tau \models K_a^{\mathsf{P}} Exp_A$" in the third and fourth items of the definition of the epistemic relation model that agents b and a, respectively, no longer answer calls once they are super experts. For example, in the third item it has the effect that after a missed call ab, any state τ after which ab is not a missed call (b answers the call) is no longer considered possible by agent a. In other words, we then have that $\sigma; ab \not\approx_a \tau; ab$.

The properties of protocol-dependent knowledge K_a^{P} listed in the previous section also hold for the semantics extended with the feature of engaged agents. In particular, on the set of all call sequences that are P-permitted and such that super experts do not make calls, the relations $\approx_a^{\mathsf{P}}$ and $\sim_a^{\mathsf{P}}$ are equivalence relations.

A special feature of the semantics with engaged calls is that calling a super expert will also make the callee a super expert:

Lemma 27. *In the semantics with engaged calls, $\models K_b^{\mathsf{P}} Exp_A \to [ab] K_a^{\mathsf{P}} Exp_A$.*

Proof. We give the proof for the asynchronous epistemic relation. The proof is similar for the synchronous relation. Let $\sigma \models K_b^{\mathsf{P}} Exp_A$ and assume $\sigma \models \neg K_a^{\mathsf{P}} Exp_A \wedge \mathsf{P}_{ab}$. Let τ' be such that $\sigma; ab \sim_a \tau'$. Given the definition of the epistemic relation, $\tau' = \tau; ab$, and from $\sigma; ab \sim_a^{\mathsf{P}} \tau; ab$ we also obtain $\sigma \sim_a^{\mathsf{P}} \tau$. As $\sigma \models K_b^{\mathsf{P}} Exp_A$ and $\sigma \sim_a^{\mathsf{P}} \tau$, from the definition of the epistemic relation we obtain $\tau \models K_b^{\mathsf{P}} Exp_A$, and thus also $\tau; ab \models K_b^{\mathsf{P}} Exp_A$. As knowledge is correct after P-permitted sequences (Section 3), also $\tau; ab \models Exp_A$. And as τ was arbitrary such that $\sigma; ab \sim_a^{\mathsf{P}} \tau; ab$, we obtain $\sigma; ab \models K_a^{\mathsf{P}} Exp_A$ and thus $\sigma \models [ab] K_a^{\mathsf{P}} Exp_A$ as desired. □

The dual effect of this semantics for missed calls is, that when after σ agent b answers a call from a, any state τ wherein agent b would have been a super expert is no longer considered possible by a. In particular, even when a learns that b already knew all secrets before the call ab, she learns that b was not yet a super expert after σ. Of course, b may have become a super expert in the call ab.

4.2. Results for the protocols ANY, PIG and CMO

Consider asynchrony. We will show that with n agents, super-succesful termination is reached in $n-2+\binom{n}{2}$ calls, which is of $\mathcal{O}(n^2)$ complexity, whereas with engaged agents super-successful termination is reached in $3n-4$ calls, which is of $\mathcal{O}(n)$ complexity. We conjecture that these $n-2+\binom{n}{2}$ and $3n-4$ are minimal. These conjectures are for asynchronous ANY. We recall that synchronous protocols typically take fewer calls until super success than their asynchronous versions (Section 2.3), whereas asynchronous protocols other than ANY typically take more calls until super success.

Proposition 28. *Given $n \geq 4$ agents, super-successful asynchronous* ANY *termination without engaged agents can be achieved in $n-2+\binom{n}{2}$ calls.*

Proof. Consider n agents, select 4 agents a, b, c, d among these n and 1 agent a among these 4. First, let a call all the agents except b, c, d. These are $(n-4)$ calls. Then, let a, b, c, d execute the sequence $ab; cd; ac; bd$. These are 4 calls. Note that in the final two calls ac and bd these agents become experts. Apart from ac and bd, we now let all remaining pairs of agents also call each other. There are $\binom{n}{2}$ pairs of agents (and these include ac and bd). Altogether these are $(n-4)+4-2+\binom{n}{2} = n-2+\binom{n}{2}$ calls. When after a call both agents are experts, they know this from one another. Therefore, after the $\binom{n}{2}$ calls, all agents know that all agents are experts: $EExp_A$ holds. □

Proposition 29. *Given n agents, super-successful asynchronous* ANY *termination with engaged agents can be achieved in $3n-4$ calls.*

Proof. Select an agent a among the n agents. First, agent a calls all other agents. These are $n-1$ calls. Then, agent a calls all agents again in the same order, except the last one that was called in the first round. These are $n-2$ calls. Finally, all other agents call a. These are $n-1$ calls. Altogether these are $3n-4$ calls. The final $n-1$ calls are all missed calls. After a missed call the calling agent is also a super expert (Lemma 27). All agents are then super experts: $EExp_A$ holds. □

We conjecture that these bounds are hard.

Conjecture 30. *Given n agents, super-successful asynchronous* ANY *termination without engaged agents requires at least $n-2+\binom{n}{2}$ calls.*

Conjecture 31. *Given n agents, super-successful asynchronous* ANY *termination with engaged agents requires at least $3n-4$ calls.*

Given that $n-2+\binom{n}{2}$ is $\mathcal{O}(n^2)$ and that $3n-4$ is $\mathcal{O}(n)$, we also conjecture that these complexity bounds are hard.

Towards proving the minimality of $n-2+\binom{n}{2}$, observe that in the proof of Proposition 28 the first call in which two agents become experts is call $n-1$. This is the minimum, as $n-1$ links are need to connect n points in a graph. So no agents are experts in the first $n-2$ calls. Also observe that in all subsequent $\binom{n}{2}$ calls, agents x and y become expert when calling each other or learn from each other that they already were experts when calling each other. This suggests that the only way in which an agent asynchronously can get to know that another agent is an expert (before or after the call) is by calling that agent. Not surprisingly, for synchrony we did not expect this (see Section 2.3 for multiple counterexamples). But, maybe somewhat surprisingly, also for asynchrony this is false, as the next example demonstrates. This does not disprove the conjecture, but unfortunately it rules out an easy proof.

Example 32. *Consider $\sigma = ac; ad; ac; bc; ac$. After the sequence $ac; ad; ac$ these three agents share their secrets. In call ad agent a learns that d has not been involved in a call with b and in the second call ac agent a learns that c has not been involved in a call with b after the first call ac. Therefore a knows that whomever b makes his first call with, he will become expert. In the third call ac of σ agent a learns that c knows the secret of b, so there should have been a call between b and c or between b and d. (If between b and d, that call could could have taken place between call ad and the second call ac, but not if between b and c.) Either way, b then would be an expert.* So a knows that b is an expert. *However, there has been no prior call between a and b wherein they both became or already were experts.*

On the other hand, this is not an efficient way to make a know that b is an expert.

First, let us show that we cannot extend σ with two more calls to be super-successful, from which follows that at least three more calls are needed, which is the conjectured minimum of $(4-2)+\binom{4}{2}=8$ calls:

After σ, nobody is a super expert, because d is not even an expert. Now at most the two calling agents can become a super expert in a call. So the only way for a two call extension of σ to be super-successful is that the next two calls are disjoint. Therefore, only one of these calls involves agent d. Because of asynchrony, the order of these disjoint calls does not matter, so it suffices to consider a single extra call involving d. In that call agent d should then become an expert and a super expert at the same time. This call can be ad, bd, or cd (or possibly the dual of any of these). It is easy to see that extending the five-call sequence with ad, bd, or cd makes d an expert but not a super expert.

$ac; ad; ac; bc; ac; \mathbf{ad}$	$\sim_d$	$ac; ad; ab; \mathbf{ad}$	*on the right, c is not an expert*
$ac; ad; ac; bc; ac; \mathbf{bd}$	$\sim_d$	$ac; ad; ab; \mathbf{bd}$	*on the right, c is not an expert*
$ac; ad; ac; bc; ac; \mathbf{cd}$	$\sim_d$	$ac; ad; bc; \mathbf{cd}$	*on the right, a is not an expert*

Therefore, no extension of less than eight calls is super-successful.

In fact, the model checker GoMoChe not only confirms that no super-successful seven-call sequence exists, but even establishes that no super-successful eight-call sequence exists. So, this prefix $\sigma = ac; ad; ac; bc; ac$ is not an efficient start in order to get super-successful termination.

We continue with some results for asynchronous ANY demonstrating how the feature of engaged agents affects termination.

Example 33. *Consider again Example 6 for three agents a, b, c and super-successful call sequence $ab; ac; ab; cb$. With engaged agents, final call cb is a missed call. The sequence remains super-successful (but we need that final call).*

Example 34. *Given are six agents a, b, c, d, e, f. We first assume asynchronous* ANY *without engaged agents. We enact the procedure also used in the proof of Proposition 28. A standard solution to obtain Exp_A is $ae; af; ab; cd; ac; bd; ae; af$. It consists of eight calls. After any of the final four calls $ac; bd; ae; af$, the involved agents are experts. The agents can continue to verify that all other agents are experts in subsequent calls. Altogether this requires each pair of agents to make a call after which they both are (or remain) experts. For 6 agents we need $8 + 15 - 4 = 19$ calls. (This is also the conjectured minimum.) An example executing with all calls in lexicographic order is as follows.*

$$ae; af; ab; cd; ac; bd; ae; af; ab; ad; bc; be; bf; cd; ce; cf; df; ed; ef$$

With engaged agents, a simpler sequence with 15 instead of 19 calls is already super-successful:

$$ae; af; ab; cd; ac; bd; ae; af; ab; ad; ba; ca; da; ea; fa$$

In this sequence first a becomes a super expert, in call ad. Then all other agents call agent a. These are the final five calls $ba; ca; da; ea; fa$, These are therefore all missed calls in which b to f also become super experts.

However, this is not the conjectured minimum of $3n-4 = 3 \cdot 6-4 = 14$. This is because agent a only becomes a super expert in the tenth call, and not in the ninth, the known minimum. If so, extending the sequence from such a ninth call with missed calls results in 14 calls instead. The method also used in Example 8 constructs a 14-call sequence that is super-successful. All calls involve a. First, a calls everyone else, then a calls everyone else except the last agent f, finally everyone else calls a, all of which are missed calls. We obtain:

$$ab; ac; ad; ae; af; \quad ab; ac; ad; ae; \quad ba; ca; da; ea; fa$$

We continue with a minor result involving PIG.

Proposition 35. *Protocols* ANY *and* PIG *have the same extension in the engaged agents semantics.*

Proof. This follows directly from Lemma 13 that $\bigvee_{a,b \in A} \mathsf{PIG}_{ab} \leftrightarrow \neg EExp_A$ is valid. Any call ab can only be executed if a is not a superexpert, i.e., if she considers it possible that some agent does not know some secret. □

So far, all the news involving engaged agents seems good: speedier termination. We close with a bit of bad news. When engaged agents withdraw from the conversation this can impede dissemination of information, and even prevent that execution terminate super-successfully. We recall Theorem 22 that synchronous known CMO is super-successful. Unfortunately, with engaged agents it is no longer super-successful.

Theorem 36. *Synchronous known* CMO *with engaged agents is not super-successful.*

Proof. The proof is by counterexamle. Consider again Example 23 and Table 4. Consider (prefix) sequence $ab; bc; cd; ad; bd$. After this sequence everyone but b is a super expert.

Agent b considers $ab; bc; ac; ad; bd$ possible (see again Figure 1) after which c is not an expert. But b has already been in a call with each other agent, and hence b is no longer permitted to make calls. On the other hand, agents a and c have not been in a call yet, so ac and ca are CMO-permitted, but they are both super experts (see Table 4) and will therefore not make a call. The protocol terminates unsuccessfully. □

If only agent b had the assurance that after the possible though not actual sequence $ab; bc; ac; ad; bd$ the final call cd would be made ... Although we assume synchrony, nothing is known about the interval between calls, so b does not have such assurance. Therefore, b cannot become a super expert.

In the next section we will show that by another extension of the semantics modelling 'clock ticks' explicitly (in *skip* programs) we can still make CMO super-successful.

For now, however, let us harvest one more result from Example 23. The final call ac of sequence $ab; bc; cd; ad; bd; \mathbf{ac}$ of Example 23 is CMO-permitted (without engaged agents), because a has not yet been involved in a call with c. So even though a is a super expert, she will make that call. But the call ac is not PIG-permitted, as agent a is a super expert (Lemma 13). Therefore, although for the expert goal it was known that CMO $\subset$ PIG [15, Prop. 53] (the extension of CMO is contained in the extension of PIG), this no longer holds for the super expert goal, with known protocols and engaged agents.

Corollary 37. *With synchronous known protocols and engaged agents:* CMO $\not\subseteq$ PIG.

5. Adding skip calls

5.1. Syntax and semantics — skip

In this section we investigate how adding a *skip* program to the language and semantics makes a difference in the termination of gossip protocols. We assume all prior enrichments of the semantics: known protocols and engaged agents. We will later see that our *skip* is different from the PDL-*skip* program defined as the test program $?\top$ [21]. It rather is the *skip* featuring in some other

publications on epistemic gossip [9, 10], that should be seen as an explicit tick of the clock, during which no call is made. Given that it means absence of a call, such a *skip* program should not be named a *skip* **call**. However, as we wish to continue to name call sequences to which *skip* programs have been added 'call sequences', we stick to the term *skip* call.

We first change the program part of the BNF of the logical language to also take into account *skip* calls. The relevant part of Definition 1 was

$$\pi \quad := \quad ?\varphi \mid ab \mid (\pi;\pi) \mid (\pi \cup \pi) \mid \pi^*$$

and the new definition is:

Definition 38 (Programs — skip)**.**

$$\pi \quad := \quad ?\varphi \mid \mathit{skip} \mid ab \mid (\pi;\pi) \mid (\pi \cup \pi) \mid \pi^*$$

where a, b range over A.

To allow *skip* calls, we change the crucial Definition 2 of protocol. Let us recall the original definition:

$$\mathsf{P} := \Big(\bigcup_{a \neq b \in A} (?(\neg K_a^{\mathsf{P}} \mathit{Exp}_A \wedge \mathsf{P}_{ab}); ab) \Big)^*; ?E^{\mathsf{P}} \mathit{Exp}_A$$

The new definition is as follows.

Definition 39 (Protocol — skip)**.**

$$\begin{array}{rcl} \mathsf{P} & := & \big(\bigcup_{a \neq b \in A} (?(\neg K_a^{\mathsf{P}} \mathit{Exp}_A \wedge \mathsf{P}_{ab}); ab)\big)^*; \\ & & ?\neg \bigvee_{a \neq b \in A} (\neg K_a^{\mathsf{P}} \mathit{Exp}_A \wedge \mathsf{P}_{ab}); \\ & & \big(\bigcup_{a \neq b \in A} (?(\neg K_a^{\mathsf{P}} \mathit{Exp}_A \wedge \neg \mathsf{P}_{ab}); \mathit{skip})\big)^*; \\ & & ?E^{\mathsf{P}} \mathit{Exp}_A \end{array}$$

where for all $a \neq b \in A$, $\mathsf{P}_{ab} \in \mathcal{L}$ is the protocol condition for call ab of protocol P.

Formula $\neg \bigvee_{a \neq b \in A} (\neg K_a^{\mathsf{P}} \mathit{Exp}_A \wedge \mathsf{P}_{ab})$ is the stop condition for the first arbitrary iteration. It is equivalent to the more intuitive $\bigwedge_{a \neq b \in A} (\mathsf{P}_{ab} \to K_a^{\mathsf{P}} \mathit{Exp}_A)$, which we will use further below. Given its position in the program, we could replace the second arbitrary iteration $\big(\bigcup_{a \neq b \in A} (?(\neg K_a^{\mathsf{P}} \mathit{Exp}_A \wedge \neg \mathsf{P}_{ab}); \mathit{skip})\big)^*$ by the shorter $\big(\bigcup_{a \in A} (?\neg K_a^{\mathsf{P}} \mathit{Exp}_A; \mathit{skip})\big)^*$ without changing the meaning of the protocol: the stop condition in the middle enforces that any agent satisfying $\neg K_a^{\mathsf{P}} \mathit{Exp}_A$ also satisfies $\neg \mathsf{P}_{ab}$. We left the condition $\neg \mathsf{P}_{ab}$ in place for intuitive clarity.

The second arbitrary iteration only fires if anyone satisfying the protocol condition is already a super expert, but when there still are agents who are not super experts (so that the protocol has not terminated super-successfully) but who do not satisfy the protocol condition.

We continue with the epistemic relations. Just as for the engaged agents semantics, the semantic relation $\models$ remains unchanged (Definition 17), we merely need to define the interpretation of program *skip*.

Definition 40 (Epistemic relations and semantics of programs — skip)**.** *Let $a \in A$. The synchronous accessibility relation $\approx_a^{\mathsf{P}}$ between call sequences is the smallest symmetric and transitive relation satisfying all the clauses of Definition 3 plus the following two inductive clauses involving skip.*

- *if $\sigma \approx_a^{\mathsf{P}} \tau$, $a \notin \{b, c\}$, $\sigma \models \bigwedge_{c \neq d \in A}(\mathsf{P}_{cd} \to K_c^{\mathsf{P}} Exp_A)$ and $\tau \models \neg K_b^{\mathsf{P}} Exp_A \wedge \mathsf{P}_{bc}$, then $\sigma; skip \approx_a^{\mathsf{P}} \tau; bc$*
- *if $\sigma \approx_a^{\mathsf{P}} \tau$, $a \notin \{b, c\}$, $\sigma \models \bigwedge_{c \neq d \in A}(\mathsf{P}_{cd} \to K_c^{\mathsf{P}} Exp_A)$ and $\tau \models \bigwedge_{c \neq d \in A}(\mathsf{P}_{cd} \to K_c^{\mathsf{P}} Exp_A)$, then $\sigma; skip \approx_a^{\mathsf{P}} \tau; skip$*

The asynchronous epistemic relation $\approx_a^{\mathsf{P}}$ is defined similarly, by adding the single clause:

- *if $\sigma \sim_a^{\mathsf{P}} \tau$ and $\sigma \models \bigwedge_{c \neq d \in A}(\mathsf{P}_{cd} \to K_c^{\mathsf{P}} Exp_A)$ then $\sigma; skip \sim_a^{\mathsf{P}} \tau$*

To the semantics of programs (Definition 4) we add the interpretation of skip:

$$\sigma[\![skip]\!]\tau \quad \textit{iff} \quad \tau = \sigma; skip$$

where $I^{\sigma;skip} := I^{\sigma}$.

Note that *skip* calls can only occur at the postfix of a permitted call sequence. In other words, all call sequences σ that are executions of protocols according to the *skip* semantics have shape $\sigma_1; \sigma_2$ where σ_1 only contains calls ab for some $a, b \in A$, whereas σ_2 only contains *skip* calls. This also holds for infinite call sequences, i.e., an infinite call sequence may consist of calls ab only, or of a finite prefix of such calls followed by an infinite postfix of *skip* calls.

Recalling the semantics of programs (Definition 4) we see that the PDL-*skip* defined as $?\top$ is defined as

$$\sigma[\![?\top]\!]\tau \quad \text{iff} \quad \tau = \sigma.$$

Note that this does not extend the call sequence, unlike our 'clock tick' *skip*.

Skip calls do not have factual consequences (changes of the value of atomic propositions): atoms $S_a b$ do not change because $I^{\sigma;skip} := I^{\sigma}$, and atoms Cab do not change as *skip* is not a call. However, they may have other informative consequences.

In the asynchronous semantics, *skip* calls do not have informative consequences.They go, so to speak, unnoticed. This is expressed by the following proposition.

Proposition 41. *Assume asynchrony. Let call sequence σ be given such that $\sigma \models \bigwedge_{c \neq d \in A}(\mathsf{P}_{cd} \to K_c^{\mathsf{P}} Exp_A)$. Then $\sigma \models K_a^{\mathsf{P}}\varphi \leftrightarrow [skip]K_a^{\mathsf{P}}\varphi$.*

Proof. First note that for any φ and σ: $\sigma \models [skip]\varphi$, iff $\tau \models \varphi$ for all τ such that $\sigma[\![skip]\!]\tau$, iff $\sigma; skip \models \varphi$.

Let now $\varphi \in \mathcal{L}$ and call sequence σ such that $\sigma \models \bigwedge_{c \neq d \in A}(\mathsf{P}_{cd} \to K_c^{\mathsf{P}} Exp_A)$ be given. Then: $\sigma \models K_a^{\mathsf{P}}\varphi$, iff $\tau \models \varphi$ for all $\tau \sim_a^{\mathsf{P}} \sigma$, iff (*) $\tau \models \varphi$ for all $\tau \sim_a^{\mathsf{P}} \sigma; skip$, iff $\sigma; skip \models K_a^{\mathsf{P}}\varphi$, iff $\sigma \models [skip]K_a^{\mathsf{P}}\varphi$. Therefore $\sigma \models K_a^{\mathsf{P}}\varphi \leftrightarrow [skip]K_a^{\mathsf{P}}\varphi$.

In (*) we use that if $\sigma \models \bigwedge_{c \neq d \in A}(\mathsf{P}_{cd} \to K_c^{\mathsf{P}} Exp_A)$, then from Definition 40 it follows that $\tau \sim_a^{\mathsf{P}} \sigma$ iff $\tau \sim_a^{\mathsf{P}} \sigma; skip$. □

If *skip* calls can take place any time we even have $\models \varphi \leftrightarrow [skip]\varphi$, as suggested by Wiebe van der Hoek in the context of [9, 10]. However, for our semantics only permitting *skip* when all agents are super experts this is false. For example, given a super-successful sequence σ for a protocol P, we have that $\sigma \models [skip]K_a^{\mathsf{P}}\bot$, as *skip* is not permitted after termination. On the other hand, evidently, $\sigma \not\models K_a^{\mathsf{P}}\bot$. So, $\sigma \not\models K_a^{\mathsf{P}}\bot \leftrightarrow [skip]K_a^{\mathsf{P}}\bot$.

In the synchronous semantics, *skip* calls may have informative consequences, as we will now see. Because the agents become aware of time, this may result in knowledge gain.

5.2. Results for the protocol CMO

Theorem 42. *Synchronous known* CMO *with engaged agents and skip is super-successful.*

Proof. Let σ be a maximal CMO-permitted sequence. Since CMO is successful, after executing σ all agents are experts: Exp_A holds. If $E^{\mathsf{CMO}}Exp_A$ now also holds, we are done. If $E^{\mathsf{CMO}}Exp_A$ does not hold, then, since σ is maximal, any agent who has not yet been involved in a call with some other agent, is already a super expert: $\bigwedge_{b\neq c\in A}(\mathsf{CMO}_{bc} \to K_b^{\mathsf{CMO}}Exp_A)$. Also, since σ is maximal but not super-successful, there is an agent a who is not a super expert but who has been involved in a call with all other agents $\neg K_a^{\mathsf{CMO}}Exp_A \wedge \bigwedge_{b\in A}\neg\mathsf{CMO}_{ab}$.

Because a is not a super expert, there is a call sequence τ such that $\sigma \approx_a \tau$ and $\tau \not\models Exp_A$, i.e., there are $b, c \in A$ such that $\tau \not\models S_bc$. Therefore $\tau \not\models Cbc$ and $\tau \not\models Ccb$, so that $\tau \models \mathsf{CMO}_{bc}$. Protocol dependent knowledge is truthful after the CMO-permitted sequence τ, therefore, from $\tau \not\models Exp_A$ it also follows that $\tau \not\models K_b^{\mathsf{P}}Exp_A$.

From all this it therefore follows that $\sigma; skip \approx_a \tau; bc$. If we now have that $\sigma; skip \models E^{\mathsf{CMO}}Exp_A$, we are done. Otherwise, we repeat the procedure until the maximum number $\binom{n}{2}$ of CMO-permitted calls has been reached. After that, $E^{\mathsf{CMO}}Exp_A$ is a property of that horizon. □

If in the above proof the horizon of $\binom{n}{2}$ calls has been reached, it is even *common knowledge* that all agents are experts, and thus it is common knowledge that they are super experts (common knowledge is an infinitary epistemic notion proposed in, for example, [22, 23, 24]). However, if termination is earlier, we are uncertain if such common knowledge is then reached. We conjecture that it is.

Example 43. *We recall Figure 1, Example 23, and Theorem 36. Synchronous known* CMO *is super-successful, however with engaged agents it is not.*

Reconsider $\sigma = ab; bc; cd; da; bd$ *and* $\tau = ab; bc; ac; ad; bd$*, and recall that* $\sigma \approx_b \tau$*. After* σ *all agents are experts. Agent* b *does not know that, because* b *considers* τ *possible. Call* ac *is not* CMO*-permitted after* σ*, because* a *is a super expert. After* τ *agent* c *does not know the secret of* d *and so* cd *is* CMO*-permitted. We now have that* $\sigma; skip \approx_b \tau; cd$ *and* $\sigma; skip \models E^{\mathsf{CMO}}Exp_A$*.*

Again, as in Example 23, common knowledge that all agents are experts is now obtained; but as observed above, it is unclear whether this is always the case.

6. Conclusion and further research

We explored gossip protocols wherein the termination condition is that all agents are *super experts*: all agents know that all agents know all secrets. For such protocols with super expert epistemic goals we also modelled *engaged agents*: agents who are super experts do not make and do not answer calls. For our results it matters whether *gossip protocols are common knowledge* among the agents.

We investigated conditions under which such gossip protocols terminate, both in the synchronous case, where there is a global clock, and in the asynchronous case, where there is not. We show that with engaged agents, and where the meaning of not answering calls is common knowledge among the agents, protocols can terminate faster than without engaged agents. We proved that the protocol CMO wherein agents may only be involved once in a call with another agent, is super-successful (always terminates for the super expert goal) in the presence of a global clock.

Our results appear to generalize to protocols with common knowledge termination conditions, which we wish to investigate in future research. It may also be of interest to investigate gossip protocols with very different epistemic calling conditions.

Acknowledgements. Hans van Ditmarsch is also affiliated to IMSc, Chennai, India, as a research associate. This work is loosely based on a contribution with the same main title (Everyone Knows that Everyone Knows) and one additional author (Rasoul Ramezanian) for a forthcoming volume honouring Mohammad Ardeshir at his retirement.

References

[1] B. Baker, R. Shostak, Gossips and telephones, Discrete Mathematics 2 (3) (1972) 191–193. doi:10.1016/0012-365X(72)90001-5.

[2] R. Tijdeman, On a telephone problem, Nieuw Archief voor Wiskunde 3(19) (1971) 188–192.

[3] W. Knödel, New gossips and telephones, Discrete Mathematics 13 (1975) 95.

[4] D. Boyd, J. Steele, Random exchanges of information, Journal of Applied Probability 16 (1979) 657–661. doi:10.2307/3213094.

[5] D. West, A class of solutions to the gossip problem, part I, Discrete Mathematics 39 (3) (1982) 307–326.

[6] S. Hedetniemi, S. Hedetniemi, A. Liestman, A survey of gossiping and broadcasting in communication networks, Networks 18 (1988) 319–349. doi:10.1002/net.3230180406.

[7] A.-M. Kermarrec, M. van Steen, Gossiping in distributed systems, SIGOPS Oper. Syst. Rev. 41 (5) (2007) 2–7. doi:10.1145/1317379.1317381.

[8] P. Eugster, R. Guerraoui, A. Kermarrec, L. Massoulié, Epidemic information dissemination in distributed systems, IEEE Computer 37 (5) (2004) 60–67. doi:10.1109/MC.2004.1297243.

[9] K. Apt, D. Grossi, W. van der Hoek, Epistemic protocols for distributed gossiping, in: Proceedings of 15th TARK, 2015, pp. 51–66. doi:10.4204/EPTCS.215.5.

[10] M. Attamah, H. van Ditmarsch, D. Grossi, W. van der Hoek, Knowledge and gossip, in: Proc. of 21st ECAI, IOS Press, 2014, pp. 21–26. doi:10.3233/978-1-61499-419-0-21.

[11] M. Attamah, H. van Ditmarsch, D. Grossi, W. van der Hoek, The pleasure of gossip, in: C. Başkent, L. Moss, R. Ramanujam (Eds.), Rohit Parikh on Logic, Language and Society, Springer, 2017, pp. 145–163.

[12] K. Apt, D. Wojtczak, Verification of distributed epistemic gossip protocols, J. Artif. Intell. Res. 62 (2018) 101–132. doi:10.1613/jair.1.11204.

[13] M. Cooper, A. Herzig, F. Maffre, F. Maris, P. Régnier, The epistemic gossip problem, Discret. Math. 342 (3) (2019) 654–663. doi:10.1016/j.disc.2018.10.041.

[14] H. van Ditmarsch, W. van der Hoek, L. Kuijer, The logic of gossiping, Artificial Intelligence 286 (2020) 103306. doi:10.1016/j.artint.2020.103306.

[15] H. van Ditmarsch, J. van Eijck, P. Pardo, R. Ramezanian, F. Schwarzentruber, Epistemic protocols for dynamic gossip, J. Applied Logic 20 (2017) 1–31. doi:10.1016/j.jal.2016.12.001.

[16] H. van Ditmarsch, J. van Eijck, P. Pardo, R. Ramezanian, F. Schwarzentruber, Dynamic gossip, Bulletin of the Iranian Mathematical Society 45(3) (2019) 701–728. doi:10.1007/s41980-018-0160-4.
URL https://arxiv.org/abs/1511.00867

[17] A. Herzig, F. Maffre, How to share knowledge by gossiping, AI Commun. 30 (1) (2017) 1–17. doi:10.3233/AIC-170723.

[18] H. van Ditmarsch, M. Gattinger, L. Kuijer, P. Pardo, Strengthening gossip protocols using protocol-dependent knowledge, FLAP 6 (1) (2019) 157–203.
URL https://arxiv.org/abs/1907.12321

[19] K. Apt, D. Grossi, W. van der Hoek, When are two gossips the same?, in: G. Barthe, G. Sutcliffe, M. Veanes (Eds.), Proc. of 22nd LPAR, Vol. 57 of EPiC Series in Computing, 2018, pp. 36–55. doi:10.29007/ww65.

[20] B. Doerr, T. Friedrich, T. Sauerwald, Quasirandom rumor spreading, ACM Trans. Algorithms 11 (2) (2014) 1–35. doi:10.1145/2650185.

[21] D. Harel, D. Kozen, J. Tiuryn, Dynamic Logic, MIT Press, Cambridge MA, 2000, foundations of Computing Series.

[22] D. Lewis, Convention, a Philosophical Study, Harvard University Press, 1969.

[23] R. Aumann, Agreeing to disagree, Annals of Statistics 4(6) (1976) 1236–1239.
URL https://www.jstor.org/stable/2958591

[24] J. Halpern, Y. Moses, Knowledge and common knowledge in a distributed environment, Journal of the ACM 37(3) (1990) 549–587. doi:10.1145/79147.79161.

Appendix

We show that protocol dependent knowledge $K_a^{\mathsf{P}}\varphi$ is well-defined. Define a relation $<$ as follows. For any call sequences σ, τ, formulas φ, ψ and agents a, b, c:

1. $(\sigma, \varphi) < (\tau, \psi)$ if φ is a subformula of ψ
2. $(\sigma, \mathsf{P}_{ab}) < (\tau, K_c^{\mathsf{P}}\varphi)$ where $a \neq b$
3. $(\sigma, \top) < (\tau, \varphi)$ where φ is not an atom
4. $(\sigma, S_a b) < (\tau, \varphi)$ where φ is not an atom
5. $(\sigma, Cab) < (\tau, \varphi)$ where φ is not an atom and $a \neq b$

The relation $<$ is a well-founded partial order, with pairs (*any call sequence*, *any atom*) at the bottom. Recalling that $K_a^{\mathsf{P}}\varphi$ can be interpreted as $K_a(X, \varphi)$ where $X = \{P_{bc} \mid b \neq c \in A\}$, clause 2. that $(\sigma, \mathsf{P}_{ab}) < (\tau, K_c^{\mathsf{P}}\varphi)$ is already subsumed by clause 1., as P_{ab} is then a subformula of $K_c^{\mathsf{P}}\varphi$.

We now show that the satisfaction relation $\models$ is well-defined using that relation $<$ is well-founded. We do this for the engaged agents semantics, without that it is even simpler. The proof is by structural induction. All clauses are trivial except knowledge.

In order to determine $\sigma \models K_a^{\mathsf{P}}\varphi$, we need to determine for all τ such that $\tau \sim_a^{\mathsf{P}} \sigma$ (where τ may be σ) that $\tau \models \varphi$, as well as (for the engaged agents semantics) $\tau \models K_b^{\mathsf{P}} \mathit{Exp}_A$ or $\tau \models \neg K_b^{\mathsf{P}} \mathit{Exp}_A$ for agents b possibly different from a.

- Concerning $\tau \models \varphi$, from clause 1. we obtain $(\tau, \varphi) < (\sigma, K_a^{\mathsf{P}}\varphi)$.
- Concerning $\tau \models K_b^{\mathsf{P}} \mathit{Exp}_A$, this can be determined by checking that $\rho \models \mathit{Exp}_A$ for any $\rho \sim_b^{\mathsf{P}} \tau$. Determining $\rho \sim_b^{\mathsf{P}} \tau$ introduces another obligation that will be honoured below. Now $\rho \models \mathit{Exp}_A$ means that $\rho \models S_c d$ for any $c, d \in A$ (not necessarily different from a or b). We then obtain from clause 4. that $(\rho, S_c d) < (\sigma, K_a^{\mathsf{P}}\varphi)$. The case $\tau \models \neg K_b^{\mathsf{P}} \mathit{Exp}_A$ is treated similarly, first using that $(\tau, K_b^{\mathsf{P}} \mathit{Exp}_A) < (\tau, \neg K_b^{\mathsf{P}} \mathit{Exp}_A)$, by clause 1.

- Concerning $\tau \sim_a^{\mathsf{P}} \sigma$, this requires to establish $\tau' \models \mathsf{P}_{cd}$ for $c, d \in A$ (where c or d may be a) and prefixes τ' of τ. We now use clause 2. that $(\tau', \mathsf{P}_{cd}) < (\sigma, K_a^{\mathsf{P}}\varphi)$.

  Similarly, concerning the novel obligation $\rho \sim_b^{\mathsf{P}} \tau$ we need to establish $\rho' \models \mathsf{P}_{cd}$ for prefixes ρ' of ρ. Again, we use clause 2. to get $(\rho', \mathsf{P}_{cd}) < (\sigma, K_a^{\mathsf{P}}\varphi)$.

Note that it plays no role whether τ or ρ are $\sim_a$ or $\sim_b$ related to σ or even by some chain of such indistinguishability links.

Further note that τ and ρ may in length largely exceed σ (and even may have σ as a prefix themselves) given asynchrony. But this does not matter, the length of sequences does not play a role in the order (it is of some importance to observe this).

A particular case of clause 1. is when $\psi = [\tau]\varphi$, such that for any program τ that is a call sequence, $(\sigma; \tau, \varphi) < (\sigma, [\tau]\varphi)$.