4.2 Decidability

There is an algorithm that can decide whether a formula $\varphi \in \mathcal{L}_n$ is valid. In other words, the validity problem in $\mathcal{M}_n$ is decidable. We begin by solving the model-checking problem: finding out if a formula is true in a fixed state of a certain Kripke structure.

We will restrict ourselves to finite Kripke structures. Given a finite Kripke structure $M = (S, \pi, \mathcal{K}_1, \dots, \mathcal{K}_n)$ let

$$\|M\| = |S| + \sum_{i=1}^{n} |\mathcal{K}_i|$$

denote the size of M. Moreover, for functions $f : \mathbb{N} \mapsto \mathbb{N}$ and $g : \mathbb{N} \mapsto \mathbb{N}$, the function g is on the order of f, written $O(f)$, if there is a constant c such that for almost all $k \in \mathbb{N}$

$$g(k) \leq cf(k)$$

Proposition 4.11. *It is possible to determine in time $O(|\varphi| \|M\|)$ whether $(M, s) \models \varphi$ for any $\varphi \in \mathcal{L}_n$.*

Proof sketch. Let $\varphi_1, \dots, \varphi_m = \varphi$ be the subformulas of φ listed according to length, shortest first. Then, $m \leq |\varphi|$ and if φ_i is a subformula of φ_j then $i < j$. We can prove by induction that one can label each state s in M with φ_k or $\neg\varphi_k$ depending on which one of φ_k and $\neg\varphi_k$ is true at s in time $O(k\|M\|)$. Namely, assuming that all states in M is labeled with respectively φ_j and $\neg\varphi_j$ for $j < k$, then all states can be labeled with φ_k or $\neg\varphi_k$ in time $O(\|M\|)$. Only the case of the form $K_i\varphi_j$ needs to examine the $\mathcal{K}_i$-relations. $\square$

We now turn to the validity problem.

Theorem 4.12. *If $\varphi \in \mathcal{L}_n$ is* K_n*-consistent, then φ is satisfiable in a Kripke structure with at most $2^{|\varphi|}$ states.*

Proof sketch. Let $Con(\varphi)$ be the set of maximal K_n-consistent subsets of

$$Sub^{\pm}(\varphi) = Sub(\varphi) \cup \{\neg\psi \mid \psi \in Sub(\varphi)\}$$

As in Lemma 4.4 it can be shown that every K_n-consistent subset of $Sub^{\pm}(\varphi)$ can be extended to a maximal K_n-consistent subset of $Sub^{\pm}(\varphi)$. Hence, any set in $Con(\varphi)$ contains either ψ or $\neg\psi$ for every $\psi \in Sub(\varphi)$, and

$$|Con(\varphi)| \leq 2^{|Sub(\varphi)|} \leq 2^{|\varphi|}$$

Following Lemma 4.7 we can define a structure $M = (Con(\varphi), \pi, \mathcal{K}_1, \dots, \mathcal{K}_n)$ such that for all $\psi \in Sub^{\pm}(\varphi)$, $(M, V) \models \psi$ iff $\psi \in V$. $\square$

Corollary 4.13. *The validity problem for* $\mathcal{M}_n$ *and the provability problem for* K_n *are decidable.*

Proof. The formula φ is provable in K_n if and only if $\neg\varphi$ is not K_n-consistent, and we can check this by testing if $\neg\varphi$ is K_n-consistent in any of the Kripke structures with at most $2^{|\varphi|}$ states. If $\neg\varphi$ is not satisfiable in any of the structures, then $\neg\varphi$ is not K_n-consistent by Theorem 4.12, and if $\neg\varphi$ is satisfiable then φ is not provable in K_n by Theorem 4.8. □

The validity problem is also decidable in the cases of T_n, $\mathrm{S4}_n$, $\mathrm{S5}_n$, and $\mathrm{KD45}_n$.

Theorem 4.14. *If* $\varphi \in \mathcal{L}_n$ *is* T_n-, $\mathrm{S4}_n$-, $\mathrm{S5}_n$-, *or* $\mathrm{KD45}_n$-*consistent, then* φ *is satisfiable in a Kripke structure in respectively* $\mathcal{M}_n^r$, $\mathcal{M}_n^{rt}$, $\mathcal{M}_n^{rst}$, *or* $\mathcal{M}_n^{elt}$, *with at most* $2^{|\varphi|}$ *states.*

Proof. Analogous to Theorem 4.12. □

Corollary 4.15. *The validity problem for* $\mathcal{M}_n^r$, $\mathcal{M}_n^{rt}$, $\mathcal{M}_n^{rst}$, *and* $\mathcal{M}_n^{elt}$ *and the provability problem for* T_n, $\mathrm{S4}_n$, $\mathrm{S5}_n$, *and* $\mathrm{KD45}_n$ *are decidable.*

4.3 Adding common and distributed knowledge

The operator C_G is defined as an infinite conjunction. One might therefore expect that it can not be characterized with a finite number of axiom schemas. This would be wrong as the fixed-point axiom, in fact, encodes the infinite conjunction.

Let K_n^C, T_n^C, $\mathrm{S4}_n^C$, $\mathrm{S5}_n^C$, and $\mathrm{KD45}_n^C$ consist of all the axioms of respectively K_n, T_n, $\mathrm{S4}_n$, $\mathrm{S5}_n$, and $\mathrm{KD45}_n$ together with the schemas

C1 $E_G\varphi \Leftrightarrow \bigwedge_{i\in G} K_i\varphi$

C2 $C_G\varphi \Rightarrow E_G(\varphi \wedge C_G\varphi)$

RC1 From $\varphi \Rightarrow E_G(\varphi \wedge \psi)$ infer $\varphi \Rightarrow C_G\psi$ (Induction rule)

Theorem 4.16. *For the language* $\mathcal{L}_n^C$

1. K_n^C *is a sound and complete axiomatization of* $\mathcal{M}_n$
2. T_n^C *is a sound and complete axiomatization of* $\mathcal{M}_n^r$
3. $\mathrm{S4}_n^C$ *is a sound and complete axiomatization of* $\mathcal{M}_n^{rt}$
4. $\mathrm{S5}_n^C$ *is a sound and complete axiomatization of* $\mathcal{M}_n^{rst}$

5. $\mathrm{KD45}_n^C$ *is a sound and complete axiomatization of* $\mathcal{M}_n^{elt}$

Proof sketch. Theorems 3.7–3.9 proves soundness. The proof of completeness is similar to that of Theorem 4.8. $\square$

The axioms of distributed knowledge are

D1 $D_{\{i\}}\varphi \Leftrightarrow K_i\varphi$

D2 $D_G\varphi \Rightarrow D_{G'}\varphi$ if $G \subseteq G'$

as well as axioms analogous to those of the knowledge operator K_i. For instance,

$$D_G\varphi \wedge D_G(\varphi \wedge \psi) \Rightarrow D_G\psi$$

corresponds to A2 and holds in $\mathcal{M}_n$, whereas

$$D_G\varphi \Rightarrow \varphi$$

which corresponds to A3 only holds in $\mathcal{M}_n^r$.

Let K_n^D, T_n^D, $\mathrm{S4}_n^D$, $\mathrm{S5}_n^D$, and $\mathrm{KD45}_n^D$ consist of all the axioms of respectively K_n, T_n, $\mathrm{S4}_n$, $\mathrm{S5}_n$, and $\mathrm{KD45}_n$ together with D1, D2, and the distributed axioms corresponding to the relevant knowledge axioms.

Theorem 4.17. *For the language* $\mathcal{L}_n^D$

1. K_n^D *is a sound and complete axiomatization of* $\mathcal{M}_n$
2. T_n^D *is a sound and complete axiomatization of* $\mathcal{M}_n^r$
3. $\mathrm{S4}_n^D$ *is a sound and complete axiomatization of* $\mathcal{M}_n^{rt}$
4. $\mathrm{S5}_n^D$ *is a sound and complete axiomatization of* $\mathcal{M}_n^{rst}$
5. $\mathrm{KD45}_n^D$ *is a sound and complete axiomatization of* $\mathcal{M}_n^{elt}$

By combining the axioms of common and distributed knowledge we get sound and complete axiomatizations for the language $\mathcal{L}_n^{CD}$.

5 Multi-agent systems

When analyzing a system we divide it into two parts, the agents and the environment. For example, the agents might be processes that communicate with each other, and the environment is the media through which the communication takes place. At a particular point in time, every agent is in a certain state that records, for instance, what messages the agent has sent and received. Similarly, the environment can be in a state which describes what messages have been lost or corrupted. Thus, the environment is just another agent with a special role.

Definition 5.1. Let L_e be the set of possible states for the environment and let L_i be the set of possible local states for the agent i, $i = 1, \ldots, n$. Then,

$$\mathcal{G} = L_e \times L_1 \times \cdots \times L_n$$

is the set of global states, and a run over $\mathcal{G}$ is a function from the time domain, the natural numbers, to $\mathcal{G}$. Moreover, a point is a pair (r, m) consisting of a run r and a time m. If

$$r(m) = (s_e, s_1, \ldots, s_n)$$

is the global state at point (r, m), then

$$r_e(m) = s_e$$

and

$$r_i(m) = s_i$$

That is, $r_i(m)$ is the local state of the agent i at point (r, m). A round takes place between two time points (r, m) and $(r, m+1)$, and we view an agent as performing an action during a round. A system $\mathcal{R}$ over $\mathcal{G}$ is a nonempty set of runs over $\mathcal{G}$, where each run depicts a possible behavior of the system. Finally, a point (r, m) is a point in a system $\mathcal{R}$ if $r \in \mathcal{R}$.

Example 5.2 (The bit-transmission problem). Take two processes, the sender S and the receiver R, that communicate by sending a bit, zero or one, over a communication link that is lossy. And assume that the link in any particular round either transports the bit from one end to the other or loses it. Let S send its bit in every round until it receives a message from R acknowledging that it has received the bit and call this message from R the *ack* message. Moreover, let R begin to repeatedly send the *ack* as soon as it has received a message from S.

Formalizing, let $L_S = \{0, 1, (0, ack), (1, ack)\}$ be the local states of S, where the local state is $k \in \{0, 1\}$ if S is sending k but have not received an ack, and where the local state is (k, ack) if S has received an ack. Similarly, let $L_R = \{\lambda, 0, 1\}$ be the local states of R, where λ indicates that R has not received any bits yet, and where 0 or 1 indicates that R has received a 0 or a 1. We let the environment record what has taken place in the system. Use $(sendbit, \lambda)$, $(\lambda, sendack)$, and $(sendbit, sendack)$ to express that respectively S is sending and R does nothing, S does nothing and R is sending, and both are sending. The set of states of the environment is then sequences of these three pairs, where the mth member of the sequences describes the actions of S and R in round m.

The runs of the system must satisfy certain constraints. The first global state in any run must be of the form $(\langle\rangle, k, \lambda)$, where $\langle\rangle$ is the empty sequence denoting that nothing has happened and where k is the initial bit of S. In addition, for consecutive states $r(m) = (s_e, s_S, s_R)$ and $r(m{+}1) = (s'_e, s'_S, s'_R)$:

- If $s_R = \lambda$, then $s'_S = s_S$, $s'_e = s_e \cdot (sendbit, \lambda)$, where $\cdot$ stands for concatenation, and either $s'_R = \lambda$ or $s'_R = s_S$.
- If $s_S = s_R = k$, then $s'_R = s_R$, $s'_e = s_e \cdot (sendbit, sendack)$, and either $s'_S = k$ or $s'_S = (k, ack)$.
- If $s_S = (k, ack)$, then $s'_S = s_S$, $s'_R = s_R$, and $s'_e = s_e \cdot (\lambda, sendack)$.

Call this system $\mathcal{R}^{bt}$.

Assume that we have a set of primitive propositions Φ. Then, an interpreted system $\mathcal{I}$ is a pair $(\mathcal{R}, \pi)$, where $\mathcal{R}$ is a system over a set $\mathcal{G}$ of global states and π is an interpretation for Φ over the states in $\mathcal{G}$. Thus, for every $p \in \Phi$ and $s \in \mathcal{G}$, $\pi(s)(p) \in \{true, false\}$. We extend π to the points of $\mathcal{R}$ by defining

$$\pi(r, m) = \pi\big(r(m)\big)$$

and refer to the points and states of $\mathcal{R}$ as the points and states of $\mathcal{I}$.

We now define knowledge in interpreted systems. To each interpreted system $\mathcal{I} = (\mathcal{R}, \pi)$ associate a Kripke structure $\mathcal{M}_{\mathcal{I}} = (S, \pi, \mathcal{K}_1, \ldots, \mathcal{K}_n)$ such that S is the points in $\mathcal{I}$. Furthermore, if $s = (s_e, s_1, \ldots, s_n)$ and $s' = (s'_e, s'_1, \ldots, s'_n)$ are global states in $\mathcal{R}$, then say that s and s' are indistinguishable to agent i, and write $s \sim_i s'$, if $s_i = s'_i$. Similarly, for two points (r, m) and (r', m'), write $(r, m) \sim_i (r', m')$ if $r(m) \sim_i r'(m')$. Now define $\mathcal{K}_i$ as the equivalence relation $\sim_i$. For formulas $\varphi \in \mathcal{L}_n^{CD}$ let

$$(\mathcal{I}, r, m) \models \varphi \qquad \text{iff} \qquad (\mathcal{M}_{\mathcal{I}}, s) \models \varphi$$

for $s = (r, m)$. For example,

$$(\mathcal{I}, r, m) \models K_i\varphi \qquad \text{iff} \qquad (\mathcal{I}, r', m') \models \varphi$$

for all (r', m') such that $(r, m) \sim_i (r', m')$. Finally, if $(\mathcal{I}, r, m) \models \varphi$ for all points (r, m) in $\mathcal{I}$, then φ is valid in $\mathcal{I}$ and we write $\mathcal{I} \models \varphi$, and if $\mathcal{I} \models \varphi$ for all $\mathcal{I}$ in a set $\mathcal{C}$, then φ is valid in $\mathcal{C}$ and we write $\mathcal{C} \models \varphi$.

Example 5.3. We describe the bit-transmission problem as an interpreted system. Let the primitive propositions be

- $bit = 0$ to describe that the initial bit of S is 0,
- $bit = 1$ to describe that the initial bit of S is 1,
- *recbit* to describe that R has received the bit from S,
- *recack* to describe that S has received the acknowledgment from R,
- *sendbit* to describe that S has sent a bit to R, and
- *sendack* to describe that R has sent an acknowledgment to S.

The interpreted system is then $\mathcal{I}^{bt} = (\mathcal{R}^{bt}, \pi^{bt})$, where π^{bt} is defined such that

- $(\mathcal{I}^{bt}, r, m) \models bit = k$ when $r_S(m)$ is either k or (k, ack) for $k = 0, 1$,
- $(\mathcal{I}^{bt}, r, m) \models recbit$ if $r_R(m) \neq \lambda$,
- $(\mathcal{I}^{bt}, r, m) \models recack$ if $r_S(m) = (k, ack)$ for $k = 0, 1$,
- $(\mathcal{I}^{bt}, r, m) \models sendbit$ if the last pair in $r_e(m)$ is either $(sendbit, \lambda)$ or $(sendbit, sendack)$, and
- $(\mathcal{I}^{bt}, r, m) \models sendack$ if the last pair in $r_e(m)$ is either $(\lambda, sendack)$ or $(sendbit, sendack)$.

For example, after R has received a bit from S, then R knows the value of the bit. Namely, if (r, m) is point such that $r_R(m) = k$ for $k = 0, 1$, then for all points $(r', m') \sim_R (r, m)$ also $r'_R(m') = k$. Hence, at these points $r'_S(m') = k$ or $r'_S(m') = (k, ack)$ and consequently $(\mathcal{I}^{bt}, r', m') \models bit = k$. Thus, $(\mathcal{I}^{bt}, r, m) \models K_R(bit = k)$.

5.1 Time

In order to be able to express statements like "the receiver eventually knows the senders initial bit" we have to add temporal operators to the language. We focus on the four operators $\Box$ for always, $\Diamond$ for eventually, $\bigcirc$ for next, and $\mathcal{U}$ for until. The semantics is defined as follows,

$$
\begin{array}{lll}
(\mathcal{I}, r, m) \models \Box\varphi & \text{iff} & (\mathcal{I}, r, m') \models \varphi \text{ for all } m' \geq m \\
(\mathcal{I}, r, m) \models \Diamond\varphi & \text{iff} & (\mathcal{I}, r, m') \models \varphi \text{ for some } m' \geq m \\
(\mathcal{I}, r, m) \models \bigcirc\varphi & \text{iff} & (\mathcal{I}, r, m+1) \models \varphi \\
(\mathcal{I}, r, m) \models \varphi \mathcal{U} \psi & \text{iff} & (\mathcal{I}, r, m') \models \psi \text{ for some } m' \geq m \text{ and} \\
& & (\mathcal{I}, r, m'') \models \varphi \text{ for all } m'' \text{ with } m \leq m'' < m'
\end{array}
$$

For example, "the receiver eventually knows the senders initial bit" can be encoded as

$$\Diamond\big(K_R(bit = 0) \vee K_R(bit = 1)\big)$$

5.2 Synchronous systems

A system is synchronous if the agents always know what round they are in, or equivalently if time is common knowledge. Since the knowledge of the agents is determined by their local state, the time or round must be encoded in the state.

Formally, a system $\mathcal{R}$ is synchronous if for all agents i and points (r, m) and (r', m') in $\mathcal{R}$,

$$(r, m) \sim_i (r', m') \quad \text{implies} \quad m = m'$$

Moreover, an interpreted system $\mathcal{I} = (\mathcal{R}, \pi)$ is synchronous if $\mathcal{R}$ is.

For example, the muddy children problem is best described by a synchronous system since the children answer at the same time.

5.3 Perfect recall

We might expect that as an agent communicates with other agents it learns new things and as a consequence its knowledge grows over time. However, nothing in an interpreted systems stops an agent from forgetting what it has once known. If the agent never forgets, then we say that the agent has perfect recall. As all of the agents knowledge is encoded in its local state, the local state must therefore reflect everything that has happened to the agent, i.e., it must contain all of its history.

Let the local state sequence of agent i at point (r, m) be the sequence of local states the agent has gone through from time 0 to m in run r without consecutive repetitions. For instance, if agent i has gone through the sequence of local states $\langle s_i, s_i, s_i', s_i, s_i \rangle$ from time 0 to 4, then the local state sequence for i at point $(r, 4)$ is $\langle s_i, s_i', s_i \rangle$ assuming that $s_i \neq s_i'$.

Now, an agent has perfect recall in a system $\mathcal{R}$ if at all points (r, m) and (r', m') in $\mathcal{R}$, $(r, m) \sim_i (r', m')$ implies that the agent has the same local state sequence at both (r, m) and (r', m').

Perfect recall does not guarantee that if an agent knows something at a specific time, then it knows the same thing later, i.e., $\mathcal{I} \models K_i\varphi \Rightarrow \Box K_i\varphi$ does not hold. For example, in a synchronous system the agents know at time 0 that the time is 0, but at time 1 they no longer know this since it is no longer true.

5.4 Message-passing systems

When analyzing distributed systems one is often interested in the communication that takes place between the agents. In message-passing systems this is made explicit and here we call the agents processes.

Fix a set Σ_i of initial states for process i, a set INT_i of internal actions for i, and a set MSG of messages. Then, a history for i is a sequence of states whose first element is in Σ_i and whose later elements are nonempty sets of members of the form $send(\mu, j, i)$, $receive(\mu, j, i)$, and $int(a, i)$, where $\mu \in \mathrm{MSG}$ and $a \in \mathrm{INT}_i$. Intuitively, $send(\mu, j, i)$ stands for the action "μ is sent to j by i," $receive(\mu, j, i)$ stands for "μ is received from j by i," and $int(a, i)$ stands for "internal action a is performed by i." We associate a history for the process i to every point (r, m) such that the history reflects the actions the process has taken.

We speak of $send(\mu, j, i)$, $receive(\mu, j, i)$, and $int(a, i)$ as events, and say that an event is in a local state $r_i(m)$ if it appears in the history of the process i in (r, m). More precisely, if $\langle E_1, \ldots, E_k \rangle$ is the history of i in (r, m) and if e is an event, then we write $e \in r_i(m)$ if there is a $j \in \{1, \ldots, k\}$ such that $e \in E_j$. Furthermore, we say that an event e occurs in round $m + 1$ of run r if $e \in r_i(m + 1)$ but $e \notin r_i(m)$ for some i.

Definition 5.4. Given a set of initial states Σ_i, a set of internal actions INT_i, and a set of messages MSG, a message-passing system $\mathcal{R}$ is a system such that for every point (r, m) in $\mathcal{R}$

MP1 $r_i(m)$ is a history for i,

MP2 for every event $receive(\mu, j, i) \in r_i(m)$, there exists a corresponding event $send(\mu, i, j) \in r_j(m)$, and

MP3 $r_i(0)$ is a sequence of length one and $r_i(m+1)$ is either equal to $r_i(m)$ or the result of appending a set of events to $r_i(m)$.

By making the local state of each process be its history, we force the processes to have perfect recall.

A reliable message-passing system is a system in which communication is guaranteed, i.e., a system in which every message sent is eventually received. Hence, a message-passing system is reliable if it fulfills

MP4 for all processes i and j, and all points (r, m), if $send(\mu, j, i) \in r_i(m)$ then there is a $m' \geq m$ such that $receive(\mu, i, j) \in r_j(m')$.

5.5 Asynchronous message-passing systems

Asynchronous message-passing systems are message-passing systems that are not synchronous.

Call a set H of histories prefix-closed if whenever $h \in H$ then every prefix of h is also in H. Let $H_1, \ldots, H_n$ be prefixed-closed sets of histories for the processes $1, \ldots, n$ respectively, and define $\mathcal{R}(H_1, \ldots, H_n)$ as the set of all runs satisfying MP1, MP2, and MP3 such that the local states of each process i is in H_i. Then, an asynchronous message-passing systems is a system of the form $\mathcal{R}(H_1, \ldots, H_n)$ for some $H_1, \ldots, H_n$. As expected an asynchronous reliable message-passing system is an a.m.p. system whose runs also satisfy MP4.

We now show that the only information a process has about the ordering of events is what follows from its own history and the fact that a message has to be sent before it can be received.

For events e and e', and for a run r, write $e \xrightarrow{r} e'$ if

- $e = send(\mu, j, i)$ corresponds to $e' = receive(\mu, i, j)$ for some i, j, and μ,
- $e, e' \in r_i(m)$ for some i and m, and either $e = e'$ or $e \in r_i(m')$ but $e' \notin r_i(m')$ for some $m' < m$, or
- for some event e'', $e \xrightarrow{r} e''$ and $e'' \xrightarrow{r} e'$.

That is, $e \xrightarrow{r} e'$ stands for the fact that e might have caused e' in run r.

For an event e and a point (r, m), write $e \in r(m)$ if $e \in r_i(m)$ for some i. Assume that for every pair of events e and e' there is a primitive proposition $e \leq e' \in \Phi$. We say that the interpretation of $e \leq e'$ in the interpreted a.m.p. system $\mathcal{I} = (\mathcal{R}, \pi)$ is standard if

$$\pi(r, m)(e \leq e') = true$$

if and only if

$$e, e' \in r(m) \text{ and for all } m',\ e' \in r(m') \text{ implies } e \in r(m')$$

That is $e \leq e'$ is true in (r, m) if and only if e and e' both occur by round m of run r and e occurs no later than e' in r. Hence, the primitive propositions $e \leq e'$ precisely capture the order in which events happen. It turns out that if every process is only allowed to perform one action per round, then a process can only know $e \leq e'$ if $e \xrightarrow{r} e'$.

Proposition 5.5. *Let G be the group of all processes and let $\mathcal{R}$ be an a.m.p. system such that each process performs at most one action per round. Assume that the interpretation of $e \leq e'$ is standard. Then,*

$$(\mathcal{I}, r, m) \models D_G(e \leq e') \qquad \textit{iff} \qquad e \xrightarrow{r} e' \textit{ and } e, e' \in r(m)$$

Proof. Let r be a run in an a.m.p. system $\mathcal{R}$, let e be an event and let k be a positive integer. Define r' as follows for all i. Let m_i be the earliest time for which there is an event $e' \in r_i(m_i)$ such that $e \xrightarrow{r} e'$ or $e = e'$. Define

$$r'_i(m) = \begin{cases} r_i(m) & \text{if } m < m_i \\ r_i(m_i - 1) & \text{if } m_i \leq m < m_i + k \\ r_i(m - k) & \text{if } m_i + k \leq m \end{cases}$$

Otherwise, if there is no event e' and time m_i such that $e \xrightarrow{r} e'$ and $e' \in r_i(m_i)$, then let $r'_i(m) = r_i(m)$ for all m. We show that r' is a run in $\mathcal{R}$.

MP1. The run r' satisfies MP1 since r does.

MP2. Let $receive(\mu, j, i) \in r'_i(m)$. If $m < m_i$ or no m_i exists, then $r'_i(m) = r_i(m)$ and $receive(\mu, j, i) \in r_i(m)$. Therefore, there exists an event $send(\mu, i, j) \in r_j(m)$. If $m \geq m_j$, then there is an event $e' \in r_j(m)$ such that $e \xrightarrow{r} e'$ or $e = e'$. Thus, $e \xrightarrow{r} send(\mu, i, j)$, as either $e' = send(\mu, i, j)$ or e' precedes $send(\mu, i, j)$. Hence, $e \xrightarrow{r} receive(\mu, j, i)$. But this means that $m \geq m_i$ as $receive(\mu, j, i) \in r_i(m)$. This contradiction implies $m < m_j$. Thus, $r'_j(m) = r_j(m)$ and $send(\mu, i, j)$ is in $r'_j(m)$.

If $m_i \leq m < m_i + k$, then $r'_i(m) = r_i(m_i - 1)$. Hence, $receive(\mu, j, i) \in r_i(m_i - 1)$ and $send(\mu, i, j) \in r_j(m_i - 1)$. But then, $m_j > m_i - 1$ by the argument above, and $send(\mu, i, j) \in r'_j(m_i - 1)$. Therefore, $send(\mu, i, j) \in r'_j(m)$ as $m_i - 1 < m$.

If $m_i + k \leq m$, then $r'_i(m) = r_i(m - k)$. Hence, $receive(\mu, j, i) \in r_i(m - k)$ and $send(\mu, i, j) \in r_j(m - k)$. Thus, $send(\mu, j, i)$ is in $r'_j(m - k)$ if $m - k < m_j$ and in $r'_j(m)$ if $m_j \leq m - k < m_j + k$ or if $m_j + k \leq m - k$. The event $send(\mu, i, j)$ is therefore in $r'_j(m)$ and MP2 is satisfied.

MP3. As $m_i > 0$ if it exists, $r'_i(0) = r_i(0)$. Hence, $r'_i(0)$ is a sequence of length one. Noting that if $r'_i(m) = r_i(m')$ and if $r'_i(m+1) = r_i(m'')$, then $m' = m''$ or $m' + 1 = m''$ we see that MP3 is satisfied.

Now, assume that e and e' have both occurred by round m of r and that $e \xrightarrow{r} e'$. Then, by the definition of $\xrightarrow{r}$, $\pi(r,m)(e \leq e') = \mathit{true}$ as $\mathcal{I}$ is standard. Furthermore, if $(r,m) \sim_i (r',m')$ for all i, then $r(m) = r'(m')$. Hence, $(\mathcal{I}, r, m) \models D_G(e \leq e')$ by the definition of distributed knowledge.

If e and e' have not both occurred by round m, then $(\mathcal{I}, r, m) \not\models e \leq e'$. Assume that $e \xrightarrow{r} e'$ does not hold and that $e, e' \in r(m)$. Examine the point (r, m). If $(\mathcal{I}, r, m) \not\models e \leq e'$ we are done. Otherwise, $(\mathcal{I}, r, m) \models e \leq e'$ and e occurs no later than e', but not in the history of the same process. Using the construction above find a run r' such that e' precedes e. More precisely, let m_i be the round in which e occurs for process i and let m_j be the round in which e' occurs for process j. By choosing $k > m_j - m_i$ we can let e occur in round $m_j + 1$ or later. Hence, $r(m) \sim_i r'(m')$ for all i and some m', and e' precedes e in r'. Thus, $(\mathcal{I}, r', m') \not\models e \leq e'$ and $(\mathcal{I}, r, m) \not\models D_G(e \leq e')$. $\square$