Causal, Strategic, and Combined Responsibility Attribution in Situation Calculus Concurrent Game Structures

MohammadHossein Karimian[1], Shakil M. Khan[1]*, Yves Lespérance[2]

[1]Department of Computer Science, University of Regina, Regina, Saskatchewan, Canada
[2]Department of EECS, York University, Toronto, Ontario, Canada
mkm440@uregina.ca, shakil.khan@uregina.ca, lesperan@eecs.yorku.ca

Abstract

Responsibility is a central concept in accountable decision making for multiagent systems. As modern AI systems grow in complexity and autonomy, there is a growing demand for them to address issues in AI ethics, prompting researchers to formalize responsibility from diverse perspectives, including strategic responsibility. However, causal responsibility, i.e. responsibility due to actual causal contribution, has received much less attention. In this paper, we study variants of responsibility attribution from both strategic and causal perspectives within a synchronous game-theoretic logic framework that allows concurrent moves by multiple agents. Our formalization is based on Situation Calculus Synchronous Game Structures (SCSGS). We show that by combining these perspectives, one can obtain novel forms of responsibility attribution that are grounded on actual causation. While doing this, we propose an account of actual causation in SCSGS. We prove that our formalization handles the issues associated with preemption and over-determination well. We also study some key properties of responsibility and demonstrate that causal, strategic, and combined notions of responsibility are extensionally distinct.

Introduction

Responsibility is a central concept for accountable decision making in multiagent systems. As modern AI systems grow in complexity and autonomy, there is a growing demand for them to address issues in AI ethics, prompting researchers to formalize responsibility from diverse perspectives, including that of structural equation models (Chockler and Halpern 2004), STIT logic (Lorini and Schwarzentruber 2011; Lorini, Longin, and Mayor 2014; Baltag, Canavotto, and Smets 2021; Abarca and Broersen 2022), ATL (Bulling and Dastani 2013; Yazdanpanah et al. 2019), LTLf (Parker, Grandi, and Lorini 2023; De Giacomo et al. 2025), game-theory (Braham and van Hees 2012; Lorini and Mühlenbernd 2018; Baier, Funke, and Majumdar 2021), and logics of strategic and extensive games (Naumov and Tao 2021, 2023; Shi 2024). Much of this research formalizes strategic responsibility (De Giacomo et al. 2025), which involves assessing whether an agent's choice led to or "caused" a given outcome.[1] The literature distinguishes two main views on this. One, associated with Frankfurt (1969), holds that an agent is responsible only if they could have acted otherwise. The other ties responsibility to making an outcome inevitable, i.e. a "seeing-to-it" view linked to STIT logic (Horty and Belnap 1995). Based on this, in their action-based framework, (Lorini, Longin, and Mayor 2014) proposed two forms of strategic responsibility, *active responsibility* and *passive responsibility*. The former pertains to an agent ensuring that some state of affairs occurs through their actions, while the latter involves the agent's failure to prevent that effect from occurring. (Parker, Grandi, and Lorini 2023) identified two variants of passive responsibility based on whether the reasoning occurs before or after the outcome. *Passive responsibility anticipation* is a future-looking or *ex ante* notion and involves determining whether a certain choice would incur some responsibility. *Passive responsibility attribution*, on the other hand, is a retrospective or *ex post* notion, which involves assigning responsibility after the choices have been made. Strategic notions of responsibility focus on the choices that agents have and whether they promote the outcome. In multiagent game settings, it is natural to see the outcome as being determined by the entire combination of agents' moves. In this case, even doing nothing is a choice that may lead to a particular outcome (e.g., a doctor's absence might result in a patient's death).

Actual causality is the problem of identifying the causes of an observed effect from a given history of events or actions, which is also called the scenario (Halpern 2000). For instance, in a scenario where a prison guard A loads a gun and prison guard B shoots an inmate with the gun, their actions may be identified as the cause of the inmate's death. In actual causality, one focuses on what effects the actions have. Doing nothing is not an actual cause.

The notions of responsibility and actual causality are closely related. Despite this, the connection between these remains largely unexplored. To deal with this, in this paper we make a distinction between *passive causal responsibility attribution* (such as guard A's loading) and *passive strategic responsibility attribution* (such as the doctor's ab-

*Corresponding author

[1]In the literature, this is sometimes called "causal responsibility" (Vincent 2011; De Giacomo et al. 2025), which may lead to confusion.

sence). Based on this, we propose notions of responsibility grounded in strategic reasoning as well as causal contribution. Our formalization is based on the Situation Calculus Synchronous Game Structures (SCSGS) (De Giacomo, Lespérance, and Pearce 2016), a game-theoretic logic framework that allows concurrent moves by multiple agents. We show that by combining causal and strategic perspectives, one can obtain novel and stronger forms of responsibility attribution that are extensionally distinct.

While doing this, we propose an account of actual causation in SCSGS. It overcomes a major limitation of previous proposals of actual causation in action-theoretic frameworks, which were in turn proposed to deal with the expressive limitations of structural equations models-based causal models (Halpern 2016): that the scenario is a linear sequence of single-agent actions, and is thus restricted to turn-taking multiagent games. In contrast, we have a single action $tick$ whose effects depend on the combination of moves selected by the players. Each agent selects its move without knowing which move is selected by the other agents. As we will see, in domains with synchronous concurrency, besides the usual preemption problem,[2] we also face the problem of over-determination, as there may be more than one subset of the moves that are sufficient to cause the effect. In this paper, we extend previous accounts of actual causation in the situation calculus (Batusov and Soutchanski 2018; Khan and Lespérance 2021) to identify minimal subsets of moves by some of the agents that are causes of the effect, i.e., sufficient to cause it. We also identify causal chains consisting of such minimal sets of moves, and notice that there may be several of them in the scenario for a given effect.

Our contribution in this paper is thus fourfold: (i) We propose a formalization of actual causation in SCSGS. (ii) We extend previously proposed single-agent notions of strategic responsibility for coalitions of agents. (iii) Based on these, we formalize new notions of passive causal, strategic, and combined responsibility attribution. (iv) Finally, we prove some important properties of our formalization.

Preliminaries

Situation Calculus (SC). The SC is a well-known second-order language for representing and reasoning about dynamic worlds (McCarthy and Hayes 1969; Reiter 2001). In the SC, all changes are due to named actions, which are terms in the language. Situations represent a possible world history resulting from performing some actions. The constant S_0 is used to denote the initial situation where no action has been performed yet. The distinguished binary function symbol $do(a, s)$ denotes the successor situation to s resulting from performing the action a. The expression $do([a_1, \cdots, a_n], s)$ represents the situation resulting from executing actions $a_1, \cdots, a_n$, starting with situation s. As usual, a relational/functional fluent representing a property whose value may change from situation to situation takes a situation term as its last argument. There is a special predicate $Poss(a, s)$ used to state that action a is executable in situation s. Also, the special binary predicate $s \sqsubset s'$ represents that s' can be reached from situation s by executing some sequence of actions. Moreover, $s \sqsubseteq s'$ is an abbreviation of $s \sqsubset s' \vee s = s'$. Again, $s < s'$ is an abbreviation of $s \sqsubset s' \wedge Executable(s')$, where $Executable(s)$ is defined as $\forall a', s'.\ do(a', s') \sqsubseteq s \supset Poss(a', s')$, i.e. every action performed in reaching situation s was possible in the situation in which it occurred. Finally, $s \leq s'$ means $s < s' \vee s = s'$.

In the SC, a dynamic domain is specified using a basic action theory (BAT) $\mathcal{D}$ that includes the following sets of axioms: (i) (first-order or FO) initial state axioms $\mathcal{D}_{S_0}$, which indicate what was true initially; (ii) (FO) action precondition axioms $\mathcal{D}_{ap}$, characterizing $Poss(a, s)$; (iii) (FO) successor-state axioms $\mathcal{D}_{ss}$, indicating precisely when the fluents change; (iv) (FO) unique-names axioms $\mathcal{D}_{una}$ for actions, stating that different action terms represent distinct actions; and (v) (second-order or SO) domain-independent foundational axioms Σ, describing the structure of situations (Levesque, Pirri, and Reiter 1998). Although the SC is SO, Reiter (Reiter 2001) showed that for certain type of queries ϕ, $\mathcal{D} \models \phi$ iff $\mathcal{D}_{una} \cup \mathcal{D}_{S_0} \models \mathcal{R}[\phi]$, where $\mathcal{R}$ is a syntactic transformation operator called *regression* and $\mathcal{R}[\phi]$ is a SC formula that compiles dynamic aspects of the theory $\mathcal{D}$ into the query ϕ. Thus reasoning in the SC for a large class of interesting queries can be restricted to entailment checking w.r.t a FO theory (Reiter 2001).

Synchronous Game Structures (SCSGS). Following (De Giacomo, Lespérance, and Pearce 2016), we focus on games where there are n players/agents each of whom chooses a move at every time step. All such moves are executed *synchronously* and determine the next state of the game. At each time step, the state of the game is fully observable by all agents, as are all past moves of every agent. To represent such multi-player synchronous games, we use a special class of BATs, called *situation calculus synchronous game structures (SCSGS)*, which are defined as follows.

Agents. A SCSGS $\mathcal{D}$ involves a finite set of n agents, and we use a subsort *Agents* of *Objects* which includes these finitely many agents $Ag_1, \ldots, Ag_n$, each denoted by a constant, and for which unique names $Ag_i \neq Ag_j$ for $i \neq j$ and domain closure $agent(x) \equiv x = Ag_1 \vee \cdots \vee x = Ag_n$ hold.

Moves. We also use a second subsort *Moves* of *Objects*, representing the possible moves. These come in finitely many types, represented by function symbols $M_i(\vec{x})$, which are parameterized by objects $\vec{x}$, with $Move(m) \equiv \bigvee_i \exists \vec{x}.m = M_i(\vec{x})$. Given that the parameters range over *Objects*, each agent may have an infinite number of possible moves at each time step. We have unique name and domain closure axioms (parameterized by objects) for these functions $M_i(\vec{x}) \neq M_j(\vec{y})$ for $i \neq j$, and $M_i(\vec{x}) = M_i(\vec{y}) \supset \vec{x} = \vec{y}$.

Actions. In SCSGS, there is only *one action type*, $tick(m_1, \ldots, m_n)$, which represents the execution of a joint move by all the agents at a given time step. The action $tick$ has exactly n parameters, $m_1, \ldots, m_n$, one per agent, which are of sort *Moves* and corresponds to the simultaneous choice of the move to perform by the n different agents.

Legal moves. The *legal moves* available to each agent in a

[2] Preemption happens when two competing events try to achieve the same effect, and the latter of these fails to do so, as the earlier one has already achieved the effect.

given situation are specified formally using a special predicate $LegalM$, which is defined by statements of the following form (one for each agent Ag_i and move type M_i): $LegalM(Ag_i, M_i(\vec{x}), s) \doteq \Phi_{Ag_i,M_i}(\vec{x}, s)$, i.e., agent Ag_i can legally perform move $M_i(\vec{x})$ in situation s if and only if $\Phi_{Ag_i,M_i}(\vec{x}, s)$ holds. Technically $LegalM$ is an abbreviation for $\Phi_{Ag_i,M_i}(\vec{x}, s)$, which is a uniform formula (i.e., a formula that only refers to a single situation s).

Precondition axioms. The precondition axiom for the action $tick$ is fixed and specified in terms of $LegalM$ as follows: $Poss(tick(m_1, \ldots, m_n), s) \equiv \wedge_{i=1,\ldots,n} LegalM(Ag_i, m_i, s)$. Thus the joint action by all agents $tick(m_1, \ldots, m_n)$ is executable if and only if each selected move m_i is a legal move for agent Ag_i in situation s. Since we only have one action type $tick$, this is the only precondition axiom in $\mathcal{D}_{ap}$.

Successor state axioms. We have *successor state axioms* $\mathcal{D}_{ss}$, specifying the effects and frame conditions of the joint moves $tick(m_1, \ldots, m_n)$ on the fluents. Such axioms, as usual in basic action theories, are domain specific, and characterize the actual game under consideration. Within such axioms, the agent moves, which occur as parameters of $tick$, determine how fluents change as the result of joint moves.[3]

Initial situation description. Finally, the initial state of the game is axiomatized in the *initial situation description* $\mathcal{D}_{S_0}$ as usual, in a domain specific way.

SCSGS, as defined above, are a first-order extension of the Concurrent Game Structures used with logics such as ATL* (Alur, Henzinger, and Kupferman 2002), but they incorporate an action theory to specify how agent moves change the fluents and address the frame problem.

LTL Properties. In formalizing strategic responsibility, we will use LTL temporal properties. To do this, we utilize the axiomatization of infinite paths in the SC introduced by (Khan and Lespérance 2016), which adds a new sort of paths to the language that provides a natural way to talk about "infinite future histories". Paths are infinite sequences of executable situations. Following this work, we will use the special predicates $OnPath(p, s)$ to mean that situation s is on path p, $Starts(p, s)$ to specify that the path p starts with situation s, and $Suffix(p', p, s)$ to denote that the path p' starts with s and contains the same situations as p starting from s. Based on this, (De Giacomo, Lespérance, and Mancanelli 2025) defined a special predicate $Holds(\phi, p)$ to specify that a given LTL property ϕ holds on path p. In the following, we use ϕ to range over LTL formulae.

Bottle Example. We use a variant of the well-known "bottle" example (Hall 2004), where Suzy and Billy are throwing stones at a bottle. Suzy's stones are smaller and thus she requires two throws to break the bottle while Billy's stone is large and he needs just one throw to break it. The available moves of $ag \in \{Suzy, Billy\}$ can be one of $pick_{ag}$, representing the picking of stone(s), one for $ag = Billy$ or two for $ag = Suzy$; $throw_{ag}$, i.e. throwing of a stone by ag; and a catchall $other_{ag}$ move, denoting anything other than picking and throwing. The legality of these moves is specified below. $(a).\ LegalM(pick_{ag}, s) \doteq \neg Holding(ag, s)$. $(b).LegalM(throw_{ag}, s) \doteq Holding(ag, s)$. $(c).\ LegalM(other_{ag}, s)$. Thus, e.g., throwing a stone is a legal move for agent ag in situation s is she is holding one or more stones in s. For simplicity, we assume that the $other_{ag}$ move is always possible.

There are three fluents in this domain, $Holding(ag, s)$, $Broken(s)$, and $SuzyThrown(s)$, which means that the agent ag is holding their stones in situation s, the bottle is broken in s, and $Suzy$ has already thrown once before in s, respectively. The successor-state axioms are as follows.

$$
\begin{aligned}
(d).\ & Holding(ag, do(a, s)) \equiv \\
& [ag = Suzy \wedge \exists m.\ a = tick(pick_{Suzy}, m)] \vee \\
& [ag = Billy \wedge \exists m.\ a = tick(m, pick_{Billy})] \vee \\
& [ag = Suzy \wedge Holding(ag, s) \wedge \\
& \quad \neg(\exists m.\ a = tick(throw_{Suzy}, m) \wedge SuzyThrown(s))] \vee \\
& [ag = Billy \wedge Holding(ag, s) \wedge \\
& \quad \neg\exists m.\ a = tick(m, throw_{Billy})],
\end{aligned}
$$

$$
\begin{aligned}
(e).\ & Broken(do(a, s)) \equiv [\exists m.\ a = tick(m, throw_{Billy})] \vee \\
& [\exists m.\ a = tick(throw_{Suzy}, m) \wedge SuzyThrown(s)] \vee Broken(s),
\end{aligned}
$$

$$
\begin{aligned}
(f).\ & SuzyThrown(do(a, s)) \equiv \\
& \exists m.\ a = tick(throw_{Suzy}, m) \vee SuzyThrown(s).
\end{aligned}
$$

Finally, we specify what is true initially in S_0: $(g).\ \forall ag.\ \neg Holding(ag, S_0)$. $(h).\ \neg Broken(S_0)$. We will use $\mathcal{D}_{bt}$ to refer to this axiomatization. □

Actual Causation in the SC

Based on Batusov and Soutchanski's (2018) original proposal, Khan and Lespérance (2021) (KL) recently defined cause in the SC. Both assume that the scenario is a linear sequence of actions, i.e. these do not allow concurrent actions.

KL introduced the notion of *dynamic formulae*. An effect φ in their framework is a situation-suppressed dynamic formula.[4] Given an effect φ, the actual causes are defined relative to a scenario s. When s is ground, the tuple $\langle \varphi, s \rangle$ is called a *causal setting*. Also, it is assumed that $\mathcal{D} \models Executable(s) \wedge \neg\varphi[S_0] \wedge \varphi[s]$. Here $\varphi[s]$ denotes the formula obtained from φ by restoring the appropriate situation argument into all fluents in φ (see Def. 2).

KL required that each situation be associated with a time-stamp, which can then be used to uniquely identify an action occurrence. A time-stamp is an integer for their theory. KL assumed that the initial situation starts at time-stamp 0 and each action increments the time-stamp by one. Thus, their action theory includes the following axioms: $timeStamp(S_0) = 0;\ \forall a, s, ts.\ timeStamp(do(a, s)) = ts$

[3] In many cases, moves don't interfere with each other and the effects are just the union of those of each move. One can also exploit previous work on axiomatizing parallel actions to generate successor state axioms (Reiter 2001; Pinto 1998).

[4] While KL also study epistemic causation, we restrict our discussion to objective causality only. Also, in the following, we will use the terms situation, scenario, and history interchangeably.

$\equiv timeStamp(s) = ts-1$. With this, causes in their framework is a non-empty set of action-time-stamp pairs.

The notion of *dynamic formulae* is defined as follows:

Definition 1 *Let $\vec{x}$, θ_a, and $\vec{y}$ respectively range over object terms, action terms, and object and action variables. The class of* dynamic formulae *φ is defined inductively using the following grammar: $\varphi ::= P(\vec{x}) \mid Poss(\theta_a) \mid After(\theta_a, \varphi) \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \exists\vec{y}.\ \varphi$.*

We will use φ for DF. $\varphi[\cdot]$ is defined as follows:

Definition 2

$$\varphi[s] \doteq \begin{cases} P(\vec{x}, s) & \text{if } \varphi \text{ is } P(\vec{x}) \\ Poss(\theta_a, s) & \text{if } \varphi \text{ is } Poss(\theta_a) \\ \varphi'[do(\theta_a, s)] & \text{if } \varphi \text{ is } After(\theta_a, \varphi') \\ \neg(\varphi'[s]) & \text{if } \varphi \text{ is } (\neg\varphi') \\ \varphi_1[s] \wedge \varphi_2[s] & \text{if } \varphi \text{ is } (\varphi_1 \wedge \varphi_2) \\ \exists\vec{y}.\ (\varphi'[s]) & \text{if } \varphi \text{ is } (\exists\vec{y}.\ \varphi') \end{cases}$$

Recently, (Karimian, Khan, and Lespérance 2025a) proposed a variant of KL's definition of cause that captures the causal chain while identifying causes. Here we briefly discuss this. The idea behind how causes are identified is as follows. Given an effect φ and scenario s, if some action of the action sequence in s triggers φ to change its truth value from false to true relative to $\mathcal{D}$, and if there are no actions in s after it that change the value of φ back to false, then this action is a *primary* or *direct* actual cause of achieving φ in s, denoted using $CausesDirectly(a, ts, \varphi, s)$.

Definition 3 (Primary Cause (KL 2021))

$$CausesDirectly(a, ts, \varphi, s) \doteq \exists s_a.\ timeStamp(s_a) = ts \wedge$$
$$S_0 < do(a, s_a) \leq s \wedge \neg\varphi[s_a] \wedge \forall s'.(do(a, s_a) \leq s' \leq s \supset \varphi[s']).$$

That is, a executed at time-stamp ts is the *primary cause* of effect φ in situation s iff a was executed in a situation with time-stamp ts in scenario s, a caused φ to change its truth value to true, and no subsequent actions on the way to s falsified φ.

Now, note that a (primary) cause a might have been non-executable initially. Also, a might have only brought about the effect conditionally and this context condition might have been false initially. Thus earlier actions in the trace that contributed to the preconditions and the context conditions of a cause must be considered as causes as well. $CausesByChain(a, ts, cc, \varphi, s)$, which means that action a at timestamp ts is a cause of an effect φ in scenario s through causal chain cc, inductively captures all such primary and indirect causes and specifies the causal chain.[5]

Definition 4 (Actual Cause Through Causal Chain)

$$\begin{aligned} &CausesByChain(a, ts, cc, \varphi, s) \doteq \\ &\forall P.[\forall a, ts, s, cc, \varphi.(CausesDirectly(a, ts, \varphi, s) \\ &\qquad \supset P(a, ts, ((a, ts)), \varphi, s)) \\ &\quad \wedge \forall a, ts, cc', s, \varphi.(\exists a', ts', s'.(CausesDirectly(a', ts', \varphi, s) \\ &\qquad \wedge timeStamp(s') = ts' \wedge s' < s \\ &\qquad \wedge P(a, ts, cc', [Poss(a') \wedge After(a', \varphi)], s') \\ &\qquad \wedge cc = Append(cc', (a', ts')) \\ &\qquad \supset P(a, ts, cc, \varphi, s)) \\ &\quad] \supset P(a, ts, cc, \varphi, s). \end{aligned}$$

Thus, $CausesByChain$ is defined to be the least relation P such that if a executed at time-stamp ts directly causes φ in scenario s then (a, ts, cc, φ, s) is in P, where $cc = ((a, ts))$; and if a' executed at ts' is a direct cause of φ in s, the time-stamp of s' is ts', $s' < s$, and $(a, ts, cc', [Poss(a') \wedge After(a', \varphi)], s')$ is in P (i.e. a executed at ts is a direct or indirect cause of $[Poss(a') \wedge After(a', \varphi)]$ in s' through causal chain cc'), then (a, ts, cc, φ, s) is in P, where $cc = Append(cc', (a', ts'))$. Here the effect $[Poss(a') \wedge After(a', \varphi)]$ requires a' to be executable and φ to hold after a'. Also, $Append$ is defined as follows: $Append(((a_1, ts_1), \ldots, (a_n, ts_n)), (a, ts)) \doteq ((a_1, ts_1), \ldots, (a_n, ts_n), (a, ts))$.

Tick Actions as Causes in the SCSGS. The above formalization of actual causation was formulated for domains specified by BATs in the situation calculus. However, it can be used directly for SCSGS domains, as long as one focuses on identifying the $tick$ actions in the scenario that caused the effect, and causal chains consisting of $tick$ actions. This is not surprising as SCSGS are special kinds of BATs. We illustrate this in the example below.

Example (cont'd). Consider the scenario $\sigma_1 = do([tick(pick_{Suzy}, other_{Billy}), tick(throw_{Suzy}, pick_{Billy}), tick(other_{Suzy}, other_{Billy}), tick(throw_{Suzy}, throw_{Billy})], S_0)$. We want to find the actual causes of the effect $\varphi_1 = Broken(s)$. We can show that:[6] $\mathcal{D}_{bt} \models CausesByChain(tick(pick_{Suzy}, other_{Billy}), 0, cc, \varphi_1, \sigma_1)$, where $cc = ((tick(pick_{Suzy}, other_{Billy}), 0), (tick(throw_{Suzy}, pick_{Billy}), 1), (tick(throw_{Suzy}, throw_{Billy}), 3))$. Explaining backward in cc, the last $tick$ action executed at time-stamp 3 is included in the causal chain as (either of the moves in) it directly caused the breaking of the bottle. The second $tick$ action executed at 1 is also included because it is a (secondary/indirect) cause as it brought about the preconditions of the last $tick$ action (by making Billy's throw legal), besides bringing about the context condition (that $SuzyThrown$) under which Suzy's second throw can break the bottle. Finally, the first $tick$ action is also a cause as it made the second $tick$ action executable. □

[5] In this, we need to quantify over situation-suppressed DF. Thus we must encode such formulae as terms and formalize their relationship to the associated SC formulae. This is tedious but can be done essentially along the lines of (De Giacomo, Lespérance, and Levesque 2000). We assume that we have such an encoding and use formulae as terms directly.

[6] Note that since the definition of $CausesByChain$ inductively constructs the causal chain, considering some of the causes, e.g., the primary cause, will only give us a suffix of the complete causal chain cc; for simplicity, we thus only show the most indirect cause below, which captures the complete chain cc.

While the above formalization provides some insight on what $tick$ actions are causes and can be used to identify the completely irrelevant $tick$ actions, e.g. the one at time-stamp 2, observe that some irrelevant moves might still be included in the discovered causes, such as $other_{Billy}$ at time-stamp 0 in our example. In other words, our formalization of this does not specify what moves within the identified $tick$ actions are contributing to the effect. To deal with this, we next propose a formalization of agent moves as causes.

Agent Moves as Causes in the SCSGS

We now go a step further by pinpointing the moves that actually contributed to the effect within the $tick$ actions that are identified as causes. Note that, since unlike actions, agent moves within each $tick$ action are concurrently performed, it is possible that more than one alternative chain of subsets of moves in the scenario are each by itself sufficient to bring about the effect. For instance, in our example, either $((pick_{Suzy}, 0), (throw_{Suzy}, 1), (throw_{Suzy}, 3))$ or $((pick_{Billy}, 1), (throw_{Billy}, 3))$ would have been sufficient to break the bottle. Just as with causal chains in the SC, we will identify these refined causal chains in two steps. In the first step, we identify the minimal set of moves in each action that is a direct cause of the effect in some refined chain (e.g., $throw_{Billy}$ in the last $tick$ of the second refined causal chain above). We call these sets of direct causes *minimal moves primary causes* since they are minimal sets of moves that are causes. However, we must consider that there might be more than one minimal moves primary cause in one single action. For example in the last tick of our example, we can consider either only Suzy's throwing or Billy's throwing to be a minimal moves primary cause. In the second step, using refined causes, we define the refined chains mentioned above, which we call *minimal moves causal chains*.

In keeping with the formalization of dynamic domains in the SC, we will consider actions (but not moves) as (refined) causes.[7] Thus our formalization of this does not omit the irrelevant moves in each time-stamp altogether, but rather replaces them with the special move $wait$ within the $tick$ action to remove their effects. $wait$ has no effects (the domain modeler must ensure this), and is always legal: $\forall ag, s.\ \ LegalM(ag, wait, s)$.[8] Thus, for instance, in our example, one such refined action that is a cause is $tick(pick_{Suzy}, wait)$. We collect all of these for all causal chains in our new definition of causes.

We now define minimal moves primary causes:

Definition 5 (Minimal Moves Primary Cause)

$$MinMovCausesDir(a, ts, (a', ts), \varphi, s) \doteq$$
$$\exists s_a.\ timeStamp(s_a) = ts \wedge (S_0 < do(a, s_a) \leq s) \wedge \neg\varphi[s_a] \wedge$$
$$\forall s'.(do(a, s_a) \leq s' \leq s \supset \varphi[s']) \wedge MinSuffSubset(a', a, s_a, \varphi, s).$$

[7]Note that the action theory specifies how the situation changes when actions, i.e., joint moves, are performed. This allows interfering or synergic effects to be specified.

[8]In many settings, an agent might be forced to make a non-$wait$ move, e.g. in chess. But our simplifying assumption that the agent can always $wait$ is only used to extract the contributions of her moves for the purpose of causation. We could add constraints to rule out such moves in real play.

The above definition is exactly as the definition of primary causes (Def. 3), but has an additional parameter (a', ts) that returns a tuple consisting of a minimal subset of moves a' of the primary cause a that, when executed in s_a (i.e., the situation where the primary cause was executed in the original scenario), is sufficient to cause the effect φ in s (formalized using $MinSuffSubset$), and the time stamp ts of a. $MinSuffSubset(a', a, s', \varphi, s)$ means that the $tick$ action a' consists of a sufficient subset of moves of a in s' to achieve φ up to s, and it is minimal.

Definition 6

$$MinSuffSubset(a', a, s', \varphi, s) \doteq SuffSubset(a', a, s', \varphi, s)$$
$$\wedge \neg\exists a^*.\ a^* \neq a' \wedge SuffSubset(a^*, a', s', \varphi, s).$$

a' is a sufficient subset of moves of a in s' to achieve φ up to s, i.e. $SuffSubset(a', a, s', \varphi, s)$, iff a' is a subset of moves of a, and the execution of this subset a' in s' is sufficient to cause φ in s, i.e. φ becomes true after a' is executed in s' and it remains true after the subsequent actions between $do(a, s')$ and s are executed starting in $do(a', s')$.

Definition 7

$$SuffSubset(a', a, s', \varphi, s) \doteq$$
$$SubsetMovs(a', a) \wedge \exists s''.\ timeStamp(s'') = timeStamp(s)$$
$$\wedge s' < s'' \wedge \forall a_1, s_1, a_1', s_1'.[$$
$$(do(a, s') < do(a_1, s_1) \leq s \wedge$$
$$do(a', s') < do(a_1', s_1') \leq s'' \wedge$$
$$timeStamp(s_1') = timeStamp(s_1)) \supset (a_1 = a_1')]$$
$$\wedge (\forall s_1'.(s' < s_1' \leq s'') \supset \varphi[s_1']).$$

Here $SubsetMovs(a', a)$, meaning that the $tick$ action a' is exactly as a, but with some of the moves possibly replaced with the $wait$ move, is defined as follows:

Definition 8 (a' Consists of a Subset of a)

$$SubsetMovs(a', a) \doteq$$
$$\exists m_1', \cdots, m_n', m_1, \cdots, m_n.\ a' = tick(m_1', \cdots, m_n')$$
$$\wedge a = tick(m_1, \cdots, m_n)$$
$$\wedge (\forall j.\ 1 \leq j \leq n \supset (m_j' = m_j \vee m_j' = wait)).$$

Using minimal moves primary causes, we next formalize minimal moves causal chains, a variant of $CausesByChain$ that inductively constructs the minimal moves chain instead.

Definition 9 (Min. Moves Cause by Causal Chain)

$$MinMovesCausesByChain(a, ts, cc, \varphi, s) \doteq$$
$$\forall P.[\forall a, ts, cc, a', s, \varphi.[MinMovCausesDir(a, ts, (a', ts), \varphi, s)$$
$$\supset\ P(a, ts, ((a', ts)), \varphi, s)]$$
$$\wedge \forall a, ts, cc', s, \varphi.[\exists a'', a', ts', s'.($$
$$MinMovCausesDir(a', ts', (a'', ts'), \varphi, s)$$
$$\wedge\ timeStamp(s') = ts' \wedge s' < s$$
$$\wedge\ P(a, ts, cc', [Poss(a'') \wedge After(a'', \varphi)], s')$$
$$\wedge\ cc = Append(cc', (a'', ts'))) \supset\ P(a, ts, cc, \varphi, s)]$$
$$] \supset P(a, ts, cc, \varphi, s).$$

Thus $MinMovesCausesByChain$ is the smallest set such that if a $tick$ action a executed at time-stamp ts directly caused the effect φ in scenario s with the minimal moves primary cause (a', ts), then (a, ts, cc, φ, s) is

in that set, where $cc = ((a', ts))$; and if a' executed at ts' is a direct cause of φ in s with minimal moves primary cause (a'', ts'), the time-stamp of s' is ts', $s' < s$, and $(a, ts, cc', [Poss(a'') \wedge After(a'', \varphi)], s')$ is in P (i.e. a executed at ts is a direct or indirect cause of $[Poss(a'') \wedge After(a'', \varphi)]$ in s' through minimal moves causal chain cc'), then (a, ts, cc, φ, s) is in P, where $cc = Append(cc', (a'', ts'))$. This thus incrementally constructs the refined causal chains using $MinMovCausesDir$.

Minimal moves causal chains, as defined above, can be incomplete and might only include part of the chain. Thus, we also define a notion of complete chains.

Definition 10 (Complete Min. Moves Causal Chain)

$$CompleteMinMovesCausalChain(cc, \varphi, s) \doteq$$
$$\exists a, ts.\ MinMovesCausesByChain(a, ts, cc, \varphi, s) \wedge$$
$$\forall cc', ts', a'.cc' \neq cc \wedge MinMovesCausesByChain(a', ts', cc', \varphi, s)$$
$$\supset (\exists a^*, ts^*.\ (a^*, ts^*) \in cc \wedge (a^*, ts^*) \notin cc').$$

Thus cc is a complete minimal moves causal chain given effect φ and scenario s iff cc is a minimal moves causal chain for some $tick$ action a and timestamp ts, and cc includes a minimal moves cause (a^*, ts^*) that no other distinct minimal moves causal chains cc' of φ and s includes. As shown below, there can be more than one complete minimal moves causal chains.

Example (cont'd). We can show the following result.

Proposition 1 (Complete Causal Chains in σ_1)

$$\mathcal{D}_{bt} \models CompleteMinMovesCausalChain(cc_1, \varphi_1, \sigma_1) \wedge$$
$$CompleteMinMovesCausalChain(cc_2, \varphi_1, \sigma_1),$$

where $cc_1 = ((tick(pick_{Suzy}, wait),0),(tick(throw_{Suzy}, wait),1),$ $(tick(throw_{Suzy}, wait), 3))$, *and* $cc_2 = ((tick(wait, pick_{Billy}),1),$ $(tick(wait, throw_{Billy}),3))$.

Thus, in our example, we have two distinct complete causal chains, one that stems from the refined primary cause involving Suzy's second throw move and Billy's $wait$ move at time-stamp 3, and another originating from Billy's throw and Suzy's $wait$ move, again at time-stamp 3. □

Properties of Actual Causation in SCSGS

Preemption. Preemption occurs when there are more than one competing contributors (actions) to an effect, but they happen one after another/consecutively. In such cases, only the first of these should be identified as the actual cause. The (effects of the) latter actions are said to be preempted by the actual cause. We illustrate this below.

Example 2. Consider the new scenario $\sigma_2 = do([tick(pick_{Suzy}, pick_{Billy}), tick(throw_{Suzy}, other_{Billy}), tick(throw_{Suzy}, other_{Billy}), tick(other_{Suzy}, throw_{Billy})], S_0)$. In this, we can show the following result.

Proposition 2

$$\mathcal{D}_{bt} \models \neg\exists cc.\ CausesByChain(tick(other_{Suzy}, throw_{Billy}), 3,$$
$$cc, \varphi_1, \sigma_2) \wedge \neg\exists cc, m.\ MinMovesCausesByChain(tick(m,$$
$$throw_{Billy}), 3, cc, \varphi_1, \sigma_2).$$

Thus as expected, Billy's throw is not considered as part of any (refined) causal chain. □

Over-determination. Over-determination happens when the effect is contributed by some events, but a smaller subset of these would have been sufficient for the effect to hold. For example, in a voting scenario, where 6 out of 10 votes are required for a candidate to win, if 7 voters voted for candidate A, saying that all of these 7 votes are the cause of candidate A's winning the ballot would be over-determination as any 6 of these would suffice. An acceptable solution to this is to only identify any 6-vote subsets as causes (Halpern 2016).

Example (cont'd). In our first bottle example, there are only two refined causal chains; the first one only consists of Billy's moves, and the second only consists of Suzy's. The definition $MinMovesCausesByChain$ ensures that only these two chains exist, since if we were to consider the chain that includes the time-stamp 3 throw moves of both Suzy and Billy, it will not be a minimal. Formally:

Proposition 3

$$\mathcal{D}_{bt} \models \neg\exists a, ts, cc.\ MinMovesCausesByChain(a, ts,$$
$$Append(cc, (tick(throw_{Suzy}, throw_{Billy}), 3)), \varphi_1, \sigma_1).$$

Thus, $tick(throw_{Suzy}, throw_{Billy})$ at time 3 cannot be a part of any refined causal chain for any action and timestamp. □

In general, since refined causal chains are constructed to be minimal, if we have two refined chains in the same time-stamp, it cannot be the case that one of them is a refined version of the other (i.e., has a subset of moves of the other).

Theorem 1 (No Over-Determination)

$$\mathcal{D} \models \forall a, a_1, a_2, ts, cc, \varphi, s.$$
$$(MinMovesCausesByChain(a, ts, Cons((a_1, ts), cc), \varphi, s)$$
$$\wedge MinMovesCausesByChain(a, ts, Cons((a_2, ts), cc), \varphi, s))$$
$$\supset (a_1 = a_2 \vee (\neg SubsetMovs(a_1, a_2) \wedge \neg SubsetMovs(a_2, a_1))).$$

Here $Cons((a, ts), cc)$ denotes the sequence where (a, ts) is added to the front of cc.

Proof Sketch: By induction on the length of the minimal moves causal chain using properties of minimal sufficient subsets of joint moves. □

Persistence. Finally, we study the conditions under which (refined) causal chains persist when the scenario changes.

Theorem 2 (Persistence of the Causal Chain)

$$\mathcal{D} \models \forall s, s', cc, ts, a, \varphi.\ CausesByChain(a, ts, cc, \varphi, s)$$
$$\wedge s \leq s' \wedge \forall s^*, a^*.(s \leq do(a^*, s^*) \leq s' \supset \varphi[s^*])$$
$$\supset CausesByChain(a, ts, cc, \varphi, s').$$

Proof Sketch: By induction on the length of the causal chain. □

That is, if a $tick$ action a executed in ts is the cause of an effect φ in scenario s through causal chain cc, then a in ts remains the cause of φ in all subsequent situations/scenarios s' if φ does not change after it was achieved in s. This is because since the situation where φ was achieved does not change in the extended scenario, neither does the causal chain.

A similar result can be shown for refined causal chains.

Theorem 3 (Persistence of Refined Causal Chains)

$$\mathcal{D} \models \forall s,s',cc,ts,a,\varphi.\ MinMovesCausesByChain(a, ts, cc, \varphi, s) \wedge s \leq s' \wedge \forall s^*, a^*.(s \leq do(a^*, s^*) \leq s' \supset \varphi[s^*]) \supset MinMovesCausesByChain(a, ts, cc, \varphi, s').$$

Proof Sketch: By induction on the length of the minimal moves causal chain using properties of minimal sufficient subsets of joint moves. □

Causal, Strategic, Combined Responsibility

Before we can give our formalization of responsibility in SCSGS, we need to define some notions to support strategic reasoning in SCSGS. We define an agent strategy f_{ag} as a function from situations to agent ag's move in that situation, i.e. $f_{ag}(s) = m_{ag}(\vec{x})$. For a coalition of agents C, a joint strategy $\vec{f}_C = \cup_{ag \in C} f_{ag}$ and $\vec{f}_{C,ag}$ is $f_{ag} \in \vec{f}_C$.

(De Giacomo, Lespérance, and Mancanelli 2025) defined a notion of an agent being able to force an LTL temporal property ϕ by following a strategy f when operating in a nondeterministic domain, where the environment determines the outcome of the agent's actions. Here, we adapt this for multiagent settings modelled as SCSGS:

Definition 11 (Multi-Agent CanForceBy)

$$CanForceBy(C, \phi, \vec{f}_C, s) \doteq \forall p.Out(p, C, \vec{f}_C, s) \supset Holds(\phi, p),$$

where, $Out(p, C, \vec{f}_C, s) \doteq Starts(p, s) \wedge \forall a, s'.OnPath(p, s') \wedge OnPath(p, do(a, s')) \supset \forall ag.(ag \in C \supset AgentMove(a, ag) = \vec{f}_{C,ag}(s'))$,

and, $AgentMove(tick(m_1, ..., m_i, ..., m_N), i) = m_i$.

That is, coalition C can force temporal property ϕ using strategy $\vec{f}_C$ starting in situation s against any possible moves by agents outside C iff ϕ holds over all paths that can result from C following $\vec{f}_C$ starting in s. Here, $Out(p, C, \vec{f}_C, s)$ means that path p is a possible outcome trace when C executes strategy $\vec{f}_C$ starting from situation s.

We also define a variant of the above that identifies the minimal set of agents that can force an effect.

Definition 12 (Minimal Multi-Agent CanForceBy)

$$MinCanForceBy(C, \phi, \vec{f}_C, s) \doteq CanForceBy(C, \phi, \vec{f}_C, s) \wedge \neg\exists C', \vec{f}_{C'}.\ C' \subset C \wedge CanForceBy(C', \phi, \vec{f}_{C'}, s).$$

Note that, for a given ϕ and s, $MinCanForceBy$ does not necessarily hold for a unique C.

Finally, we define a predicate stating that situation s is consistent with C following strategy $\vec{f}_C$ starting from s':

Definition 13 (Strategy $\vec{f}_C$ is Consistent with Sit. s)

$$ConsStrategySit(C, s, \vec{f}_C, s') \doteq \forall a^*, s^*, ag.\ s' < do(a^*, s^*) \leq s \wedge ag \in C \supset \vec{f}_{C,ag}(s^*) = AgentMove(a^*, ag).$$

Strategic Responsibility. We are now ready to define various notions of strategic responsibility. For this, we closely follow the definitions for the single agent case presented in (Parker, Grandi, and Lorini 2023; De Giacomo et al. 2025), but extend these for a coalition C and SCSGS. In what follows, we use path formulae in the context of $CanForceBy$ and dynamic formulae in that of causal chains.

We start with active responsibility. According to (Parker, Grandi, and Lorini 2023), an agent is actively responsible for an outcome under some strategy if (i) the strategy she selected forces that outcome, and (ii) it was possible for her to select an alternative strategy that would have not forced that outcome for at least some environment response. We extend this idea for a coalition of agents C with a strategy $\vec{f}_C$.

Definition 14 (Active Responsibility By)

$$ActiveRespBy(C, \vec{f}_C, \varphi, s) \doteq MinCanForceBy(C, \Diamond\varphi, \vec{f}_C, s) \wedge \exists \vec{g}_C, \vec{g'}_{\overline{C}}.\ CanForceBy(C \cup \overline{C}, \Box\neg\varphi, (\vec{g}_C, \vec{g'}_{\overline{C}}), s),$$

where $(\vec{g}_C, \vec{g'}_{\overline{C}})$ *represents the strategy for each agent* i *such that for any situation* s, $(\vec{g}_C, \vec{g'}_{\overline{C}})(s) = g_i(s)$ *if* $i \in C$, *and* $(\vec{g}_C, \vec{g'}_{\overline{C}})(s) = g'_i(s)$, *otherwise.*

Thus, a coalition C is actively responsible for φ using strategy $\vec{f}_C$ starting in situation s iff it can force φ to eventually hold using $\vec{f}_C$ starting in s, and there is another strategy $\vec{g}_C$ for C that could have been used to always avoid φ starting in s at least for some strategy $\vec{g'}_{\overline{C}}$ of the rest of the agents $\overline{C}$.

Next, we define passive responsibility anticipation. According to (De Giacomo et al. 2025), an agent anticipates (weak) passive responsibility for an outcome under some strategy F if (i) there exists an environment strategy G such that F and G together brings about the outcome, and (ii) there exists an agent strategy F' such that F' along with the same environment strategy (i.e. G) would not bring about φ. Again, we extend this idea for a coalition of agents C:

Definition 15 (Passive Responsibility Anticipation By)

$$PassiveRespAntBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s) \doteq CanForceBy(C \cup \overline{C}, \Diamond\varphi, (\vec{f}_C, \vec{f}_{\overline{C}}), s) \wedge \exists \vec{g}_C.CanForceBy(C \cup \overline{C}, \Box\neg\varphi, (\vec{g}_C, \vec{f}_{\overline{C}}), s),$$

where $(\vec{f}_C, \vec{f}_{\overline{C}})$ *and* $(\vec{g}_C, \vec{f}_{\overline{C}})$ *is as defined above in Def. 14.*

Thus, C passively anticipates responsibility for φ using strategies $\vec{f}_C$ and $\vec{f}_{\overline{C}}$ starting in s iff C can force φ to eventually hold using $\vec{f}_C$ starting in s if the other agents followed $\vec{f}_{\overline{C}}$, and C also has a strategy $\vec{g}_C$ to ensure that φ always remains false starting in s if the other agents followed $\vec{f}_{\overline{C}}$.

Finally, we define a retrospective notion of passive responsibility. According to (De Giacomo et al. 2025), an agent has (weak) passive responsibility for some outcome under strategy F and history H such that the outcome holds in H, if (i) there exists an environment strategy G such that H is consistent with F and G, and (ii) there exists another agent strategy F' such that when executed along with G, it would have brought about the opposite outcome. We now generalize this for a coalition C:

Coalition	CausalResp	PassiveResp	Comb
$\{kllr1, kllr2\}$	✓	✓	✓
$\{waiter\}$	✓	×	×
$\{guard\}$	×	✓	×

Table 1: Responsibilities in the attempted murder scenario.

Definition 16 (Passive Responsibility Attribution By)

$$\begin{aligned}&PassiveRespAttribBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s) \doteq \\ &\quad \varphi[s] \wedge ConsStrategySit(C, s, \vec{f}_C, S_0) \wedge \\ &\quad ConsStrategySit(\overline{C}, s, \vec{f}_{\overline{C}}, S_0) \wedge \exists s'.s' < s \wedge \\ &\quad PassiveRespAntBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s').\end{aligned}$$

Thus, coalition C has passive responsibility for outcome φ in situation/history s by strategies $\vec{f}_C$ and $\vec{f}_{\overline{C}}$ iff φ is observed in s, s is consistent with both $\vec{f}_C$ and $\vec{f}_{\overline{C}}$ starting in the initial situation S_0, and C could have anticipated passive responsibility for φ by $\vec{f}_C$ and $\vec{f}_{\overline{C}}$ in an earlier situation s' in the history of s.[9]

Attempted Murder Example. Consider a domain $\mathcal{D}_{AM}$, in which there are four agents, two killers *kllr1* and *kllr2*, a *waiter*, and a *guard*. The killers each can *poison* a drink with 50% lethal strength, and can *serve* it to a victim. The waiter can also *serve* the drink. The guard can *intervene* if the drink is poisoned, before it is served (or at the same time it is being served). All agents can also instead choose to play *other* moves. The outcome is $MurderAttempted(s)$, which becomes true if the drink is 100% poisoned and then served without the guard intervening.

Now, consider the following individual strategies f_{ag} of agent $ag \in \{kllr1, kllr2, waiter, guard\}$. Agent *kllr1* poisons in the first tick and plays the 'other' move in the second and third. Agent *kllr2* poisons in the first tick, plays other in the second, and serves the drink in the third if it is not served yet, otherwise plays other in the third tick as well. Agent *guard* simply plays other in all three ticks. Finally, agent *waiter* serves in the second tick and chooses to play other in the first tick and in the third tick. In the following, we will combine these individual strategies f_{ag} to obtain strategies $\vec{f}_C$ for various coalitions C or their complements $\overline{C}$. Thus, e.g., if $C_1 = \{kllr1, kllr2\}$, we will use $\vec{f}_{C_1}$ to denote the strategy of this coalition where the individual strategies of each member ag of the coalition (in this case, *kllr1* and *kllr2*) are as specified above by f_{ag}; and $\vec{f}_{\overline{C_1}}$ to denote the strategy of $\overline{C_1} = \{waiter, guard\}$, where, again, the individual strategies of *waiter* and *guard* are as specified above by f_{ag}.

In the actual scenario, both killers poison the drink in the first tick and the waiter serves it in the second. At all remaining ticks, the agents choose the other move, so the guard never intervenes. We can show that the coalition $C_1 = \{kllr1, kllr2\}$ is passively responsible for the attempted murder by strategies $\vec{f}_{C_1}$ and $\vec{f}_{\overline{C_1}}$. The waiter's move is a cause, but if she does not serve, *kllr2* can still deliver on the third tick, so the waiter cannot prevent the outcome. Thus she is not passively responsible by the strategies $\vec{f}_{C_2}$ and $\vec{f}_{\overline{C_2}}$, where $C_2 = \{waiter\}$. The guard, however, could have blocked delivery and thus prevented the outcome. We can show that by strategies $\vec{f}_{C_3}$ and $\vec{f}_{\overline{C_3}}$, the coalition $C_3 = \{guard\}$ is also passively responsible. The strategic responsibilities for this scenario and strategies are shown in Column 3 of Table 1. See (Karimian, Khan, and Lespérance 2025b) for a formalization.

Causal Responsibility Attribution. While the above notions of strategic responsibility account for the choices made by the coalition and their consequences, they do not capture the causal contributions to the outcome. For example, in the above scenario, both $\{kllr1, kllr2\}$ and $\{guard\}$ are considered passively responsible, yet this attribution overlooks their causal contribution to the attempted murder. To address this, we now define causal responsibility. Since actual causality is a retrospective notion, this is also defined relative to a history s, and is thus an ex post notion.

Definition 17 (Causal Responsibility Attribution)

$$\begin{aligned}&CausRespAttrib(C, \varphi, s) \doteq \\ &\exists cc.CompleteMinMovesCausalChain(cc, \varphi, s) \wedge \\ &\forall ag.\ ag \in C \supset \exists a, ts.(a, ts) \in cc \wedge AgentMove(a, ag) \neq wait.\end{aligned}$$

Thus, coalition C is causally responsible for effect φ in situation s iff all agents inside C contributed to the effect using a non-$wait$ move in at least one (and the same) complete minimal-moves causal chain.

Using this, we define a combined notion of responsibility.

Definition 18 (Passive Combined Resp. Attribution)

$$\begin{aligned}&PassiveCombRespAttribBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s) \doteq \\ &\quad CausRespAttrib(C, \varphi, s) \wedge ConsStrategySit(C, s, \vec{f}_C, S_0) \wedge \\ &\quad ConsStrategySit(\overline{C}, s, \vec{f}_{\overline{C}}, S_0) \wedge \\ &\quad PassiveRespAttribBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s).\end{aligned}$$

Thus, C has combined responsibility for φ by $\vec{f}_C$ and $\vec{f}_{\overline{C}}$ in s iff it is causally responsible for φ in s, s is consistent with both $\vec{f}_C$ and $\vec{f}_{\overline{C}}$ starting in the initial situation S_0, and C is passively responsible for φ by $\vec{f}_C$ and $\vec{f}_{\overline{C}}$ in s. Note that, there are cases where C has passive strategic responsibility for φ, but it is not causally responsible for it.

Returning to our example, we can show that $\{kllr1, kllr2\}$ and $\{waiter\}$ are causally responsible, while $\{guard\}$ is not, allowing us to distinguish between coalitions that merely enable outcomes and those that bring them about. $\{waiter\}$'s causal involvement however lacks strategic intent given the strategies of the others agents, which is consistent with her innocent bystander role. See Table 1.

[9]Note that unlike in the case for active responsibility, our definitions of passive responsibility do not ensure that the responsible group is minimal. With some effort, passive responsibility can also be minimized. In fact it is the case that if $PassiveRespAntBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s)$ and C' is a superset of C, then $PassiveRespAntBy(C', \vec{f}'_C, \vec{f}_{\overline{C'}}, \varphi, s)$ (where the individual strategies are the same, i.e. $(\vec{f}_C, \vec{f}_{\overline{C}}) = (\vec{f}_{C'}, \vec{f}_{\overline{C'}})$). Thus with our definition of passive responsibility anticipation, it is indeed useful to focus on the minimal coalitions that have passive responsibility.

Properties of Responsibility

• **Temporal Consistency:** A direct consequence of Def. 15 and 16 is that if a coalition is causally responsible in anticipation, then—if the anticipated structure unfolds—it is also responsible in attribution.

Corollary 1 (From Anticipation to Attribution)

$$\begin{aligned}&\mathcal{D} \models \forall C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s.\ (PassiveRespAntBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s)\\ &\wedge \exists s^*.\ s^* > s \wedge \varphi[s^*] \wedge ConsStrategySit(C, s^*, \vec{f}_C, s)\\ &\wedge ConsStrategySit(\overline{C}, s^*, \vec{f}_{\overline{C}}, s))\\ &\supset PassiveRespAttribBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s).\end{aligned}$$

This also (trivially) holds in the other direction.

Corollary 2 (From Attribution to Anticipation)

$$\begin{aligned}&\mathcal{D} \models \forall C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s.\ PassiveRespAttribBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s)\\ &\supset \exists s'.\ s' < s \wedge PassiveRespAntBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s').\end{aligned}$$

• **Non-Redundancy of Causal Responsibility:**

Theorem 4 (Causal vs. Strategic Responsibility)

$$\begin{aligned}\mathcal{D} \not\models \forall C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s.\ &PassiveRespAttribBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s)\\ &\supset CausRespAttrib(C, \varphi, s),\\ \mathcal{D} \not\models \forall C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s.\ &CausRespAttrib(C, \varphi, s)\\ &\supset PassiveRespAttribBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s).\end{aligned}$$

Proof Sketch: By counterexample; see attempted murder example above. □

• **Responsibility Persistence:** If a coalition C has passive responsibility for φ by $\vec{f}_C$ and $\vec{f}_{\overline{C}}$ in s, then it will retain this responsibility in all subsequent situations s' if both C and $\overline{C}$ keep following these strategies from s to s', and if φ holds in s'.

Theorem 5 (Persistence of Passive Responsibility)

$$\begin{aligned}\mathcal{D} \models \forall C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s, s'.[&PassiveRespAttribBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s)\\ &\wedge s < s' \wedge ConsStrategySit(C, s', \vec{f}_C, s)\\ &\wedge ConsStrategySit(C, s', \vec{f}_{\overline{C}}, s) \wedge \varphi[s']]\\ &\supset PassiveRespAttribBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s').\end{aligned}$$

Proof Sketch: Follows from the antecedent and Definitions 15 and 16. □

Moreover, if in addition φ remains true in all situations in between s and s', then the above persistence result can be extended for combined responsibility as well.

Corollary 3 (Persistence of Combined Responsibility)

$$\begin{aligned}\mathcal{D} \models \forall C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s, s'.\\ [&PassiveCombRespAttribBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s)\\ &\wedge s < s' \wedge ConsStrategySit(C, s', \vec{f}_C, s)\\ &\wedge ConsStrategySit(C, s', \vec{f}_{\overline{C}}, s)\\ &\wedge \forall a^*, s^*.\ (s \leq do(a^*, s^*) \leq s' \supset \varphi[s^*])]\\ \supset &PassiveCombRespAttribBy(C, \vec{f}_C, \vec{f}_{\overline{C}}, \varphi, s').\end{aligned}$$

Proof Sketch: Follows from Definitions 10, 17, and 18, and Theorems 3 and 5. □

Conclusion

We proposed an account of causation as well as causal, strategic, and combined responsibility attribution in a synchronous game-theoretic multiagent logic framework. Our proposal builds on Batusov and Soutchanski's (2018) original formulation of achievement causality in the situation calculus. The relationship between that framework and Halpern and Pearl's intervention-based counterfactual causality (2016) was also formally examined in that work. Closely related to our work is the preliminary study in (Karimian, Khan, and Lespérance 2025a), which formalizes actual cause in the SCSGS; here we refine on this by defining complete causal chains and using these to formalize responsibility. To our knowledge, the only other work linking responsibility to actual causation is (Chockler and Halpern 2004), where degrees of responsibility are defined in terms of the number of changes required to avoid the outcome.

Our proposal is nevertheless limited in many ways. We only dealt with achievement causation and considered objective responsibility exclusively. While we handled the responsibility of coalitions, we did not consider how responsibility/blame should be ultimately distributed between the members of the coalition. There are many philosophical puzzles, such as the bystander effect and the circle-of-blame, that need to be settled before such attribution can be formalized. In the future, it would be interesting to study maintenance causation in SCSGS. Also, responsibility attribution should account for the knowledge of the agent, which requires the integration of epistemic logic with the current proposal. To rule out accidental effects, one must integrate conative logic and notions of goals and intentions with responsibility. This would allow one to distinguish responsibility incurred due to intentional actions and accidental ones. Further, considering obligations and deontic logic might shed some light on the bystander effect, e.g., by stipulating that due to her obligations, the daycare worker should be held more strongly responsible than all other bystanders when it comes to the muddy child. Finally, in the future, it would be interesting to look into the practical aspects of this research.

Acknowledgements

We would like to thank the anonymous reviewers for helping us improve this paper. This work is partially supported by the National Science and Engineering Research Council of Canada, by the University of Regina, and by York University.

References

Abarca, A. I. R.; and Broersen, J. M. 2022. A Stit Logic of Responsibility. In Faliszewski, P.; Mascardi, V.; Pelachaud, C.; and Taylor, M. E., eds., *21st International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2022, Auckland, New Zealand, May 9-13, 2022*, 1717–1719. International Foundation for Autonomous Agents and Multiagent Systems (IFAAMAS).

Alur, R.; Henzinger, T. A.; and Kupferman, O. 2002. Alternating-time temporal logic. *J. ACM*, 49(5): 672–713.

Baier, C.; Funke, F.; and Majumdar, R. 2021. A Game-Theoretic Account of Responsibility Allocation. In Zhou, Z., ed., *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, 1773–1779. ijcai.org.

Baltag, A.; Canavotto, I.; and Smets, S. 2021. Causal Agency and Responsibility: A Refinement of STIT Logic. In Giordani, A.; and Malinowski, J., eds., *Logic in High Definition: Trends in Logical Semantics*, 149–176.

Batusov, V.; and Soutchanski, M. 2018. Situation Calculus Semantics for Actual Causality. In McIlraith, S. A.; and Weinberger, K. Q., eds., *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, 1744–1752. AAAI Press.

Braham, M.; and van Hees, M. 2012. An Anatomy of Moral Responsibility. *Mind*, 121(483): 601–634.

Bulling, N.; and Dastani, M. 2013. Coalitional Responsibility in Strategic Settings. In *Proceedings of the 14th International Workshop on Computational Logic in Multi-Agent Systems (CLIMA)*, volume 8143 of *Lecture Notes in Computer Science*, 172–189. Springer.

Chockler, H.; and Halpern, J. Y. 2004. Responsibility and Blame: A Structural-Model Approach. *Journal of Artificial Intelligence Research*, 22: 93–115.

De Giacomo, G.; Lespérance, Y.; and Levesque, H. J. 2000. ConGolog, A Concurrent Programming Language based on the Situation Calculus. *Artificial Intelligence*, 121(1-2): 109–169.

De Giacomo, G.; Lespérance, Y.; and Mancanelli, M. 2025. Situation Calculus Temporally Lifted Abstractions for Generalized Planning. In Walsh, T.; Shah, J.; and Kolter, Z., eds., *Proceedings of the 39th Annual AAAI Conference on Artificial Intelligence (AAAI-25), February 25 - March 4, 2025, Philadelphia, Pennsylvania, USA*, 14848–14857. AAAI Press.

De Giacomo, G.; Lespérance, Y.; and Pearce, A. R. 2016. Situation Calculus Game Structures and GDL. In *ECAI*, 408–416.

De Giacomo, G.; Lorini, E.; Parker, T.; and Parretti, G. 2025. Responsibility Anticipation and Attribution in LTLf. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2025, Montreal, Canada, 16-22 August 2025*. ijcai.org.

Frankfurt, H. G. 1969. Alternate possibilities and moral responsibility. *The Journal of Philosophy*, 66(23): 829–839.

Hall, N. 2004. Two Concepts of Causation. In Collins, J.; Hall, N.; and Paul, L. A., eds., *Causation and Counterfactuals*, 225–276. MIT Press.

Halpern, J. Y. 2000. Axiomatizing Causal Reasoning. *Journal of Artificial Intelligence Research*, 12: 317–337.

Halpern, J. Y. 2016. *Actual Causality*. MIT Press. ISBN 978-0-262-03502-6.

Horty, J. F.; and Belnap, N. 1995. The deliberative STIT: A study of action, omission, ability, and obligation. *J. Philos. Log.*, 24(6): 583–644.

Karimian, M.; Khan, S. M.; and Lespérance, Y. 2025a. On the Semantics of Actual Causality in Situation Calculus Concurrent Game Structures. In *38th Canadian Conference on Artificial Intelligence, Canadian AI 2025, Calgary, AB, Canada, May 26-29, 2025, Proceedings*. Canadian Artificial Intelligence Association.

Karimian, M.; Khan, S. M.; and Lespérance, Y. 2025b. Causal, Strategic, and Combined Responsibility Attribution in Situation Calculus Concurrent Game Structures - Extended Version. arXiv:(to appear).

Khan, S. M.; and Lespérance, Y. 2016. Infinite Paths in the Situation Calculus: Axiomatization and Properties. In Baral, C.; Delgrande, J. P.; and Wolter, F., eds., *Principles of Knowledge Representation and Reasoning: Proceedings of the Fifteenth International Conference, KR 2016, Cape Town, South Africa, April 25-29, 2016*, 565–568. AAAI Press.

Khan, S. M.; and Lespérance, Y. 2021. Knowing Why - On the Dynamics of Knowledge about Actual Causes in the Situation Calculus. In Dignum, F.; Lomuscio, A.; Endriss, U.; and Nowé, A., eds., *AAMAS '21: 20th International Conference on Autonomous Agents and Multiagent Systems, Virtual Event, United Kingdom, May 3-7, 2021*, 701–709. ACM.

Levesque, H. J.; Pirri, F.; and Reiter, R. 1998. Foundations for the Situation Calculus. *Electronic Transactions on Artificial Intelligence (ETAI)*, 2: 159–178.

Lorini, E.; Longin, D.; and Mayor, E. 2014. A logical analysis of responsibility attribution: emotions, individuals and collectives. *J. Log. Comput.*, 24(6): 1313–1339.

Lorini, E.; and Mühlenbernd, R. 2018. The Long-Term Benefits of Following Fairness Norms under Dynamics of Learning and Evolution. *Fundam. Informaticae*, 158(1-3): 121–148.

Lorini, E.; and Schwarzentruber, F. 2011. A logic for reasoning about counterfactual emotions. *Artif. Intell.*, 175(3-4): 814–847.

McCarthy, J.; and Hayes, P. J. 1969. Some Philosophical Problems from the Standpoint of Artificial Intelligence. *Machine Intelligence*, 4: 463–502.

Naumov, P.; and Tao, J. 2021. Two Forms of Responsibility in Strategic Games. In Zhou, Z., ed., *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, 1989–1995. ijcai.org.

Naumov, P.; and Tao, J. 2023. Counterfactual and seeing-to-it responsibilities in strategic games. *Ann. Pure Appl. Log.*, 174(10): 103353.

Parker, T.; Grandi, U.; and Lorini, E. 2023. Anticipating Responsibility in Multiagent Planning. In Gal, K.; Nowé, A.; Nalepa, G. J.; Fairstein, R.; and Radulescu, R., eds., *ECAI 2023 - 26th European Conference on Artificial Intelligence, September 30 - October 4, 2023, Kraków, Poland - Including 12th Conference on Prestigious Applications of Intelligent Systems (PAIS 2023)*, volume 372 of *Frontiers in Artificial Intelligence and Applications*, 1859–1866. IOS Press.

Pinto, J. 1998. Concurrent Actions and Interacting Effects. In *KR*, 292–303.

Reiter, R. 2001. *Knowledge in Action. Logical Foundations for Specifying and Implementing Dynamical Systems*. Cambridge, MA, USA: MIT Press. ISBN 9780262182188.

Shi, Q. 2024. Responsibility in Extensive Form Games. In Wooldridge, M. J.; Dy, J. G.; and Natarajan, S., eds., *Thirty-Eighth AAAI Conference on Artificial Intelligence, AAAI 2024, Thirty-Sixth Conference on Innovative Applications of Artificial Intelligence, IAAI 2024, Fourteenth Symposium on Educational Advances in Artificial Intelligence, EAAI 2014, February 20-27, 2024, Vancouver, Canada*, 19920–19928. AAAI Press.

Vincent, N. A. 2011. A Structured Taxonomy of Responsibility Concepts. In Vincent, N. A.; van de Poel, I.; and van den Hoven, J., eds., *Moral Responsibility: Beyond Free Will and Determinism*, 15–35. Dordrecht: Springer Netherlands.

Yazdanpanah, V.; Dastani, M.; Jamroga, W.; Alechina, N.; and Logan, B. 2019. Strategic Responsibility Under Imperfect Information. In Elkind, E.; Veloso, M.; Agmon, N.; and Taylor, M. E., eds., *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019*, 592–600. International Foundation for Autonomous Agents and Multiagent Systems.