Knowledge and games: theory and implementation

Witzel, S.A.

Citation for published version (APA):
Chapter 3

Strategies in interaction structures

3.1 Introduction

3.1.1 Motivation

Our aim in this chapter is to model situations similar to the example from the Introduction chapter, and examine what players may be said to know in any particular state of communication, how they can compute this knowledge, and how they can use it to eliminate strategies.

There is a substantial amount of research within game theory on the implications of assumptions concerning players’ knowledge and beliefs [16]. In particular, Tan and Werlang [144] have shown that if payoffs are commonly known and all players are rational in a formal sense and commonly believe in each other’s rationality, they will only play strategies that survive iterated elimination of strictly dominated strategies (IESDS, as explained in the Introduction chapter). Note that we do not delve into the details of possible definitions of rationality here; in our context the relevant implication is that rational players do not choose strictly dominated strategies.

Another line of research stresses the relevance of locality in strategic games. For example, in graphical games [85] the locality assumption manifests itself in payoff functions which depend only on the strategies of players’ neighbors in a graph structure over the set of players.

The framework we consider in this chapter applies a locality assumption to the information about payoffs, rather than to the payoffs themselves. Concretely, we use the framework of interaction structures from Chapter 2 and add the notion of a strategic game to it. We assume that the players’ initial information only covers their own preferences, and that they can communicate their preferences according to the interaction structure. We study the outcomes of strategy elimination that can be obtained in any given state of communication, including the situation when all communication permitted by the interaction structure has taken place. Insights
from Chapter 2 are used in order to prove that the outcomes we establish indeed correctly reflect what the players know in any particular situation. Building upon Chapter 1, we then describe two ways to implement IESDS locally in each player process in a distributed setting.

It is important to note that we do not examine strategic aspects of communication here. Firstly, that means that we do not allow players to lie; and secondly, that we do not examine why players do, or what they should, communicate. Rather, we examine what happens if they do.

To justify this focus, we can think of settings where the strategic aspects of communication itself are not relevant. One possibility is if communication is not a deliberate act, for example, if it occurs through observing behavior. If Ann keeps going to Indian restaurants, she involuntarily communicates her food preference to anyone observing her. Such communication is certainly more difficult to control, and more laborious to fake, than mere words. In a sense it is inherently credible, and research in social learning argues along similar lines [38, Ch. 3].

This also helps to explain another assumption we make, corresponding to the framework from Chapter 2: Players only communicate their own preferences, since communication about others’ preferences is either not inherently credible (if done with words) or difficult to accomplish (if conveyed through behavior). One may also assume that communicating about third parties is less common for privacy reasons.

Overall, the hyperarcs of the interaction structure can then be viewed as corresponding to groups who have occasions where they commonly observe each other, for example colleagues sharing lunch time.

In more proactive settings, in particular in the case of communicating processes, it may be more difficult to view communication as something not deliberate. Here, ignoring strategic aspects of communication can be interpreted as bounds on the players’ rationality or reasoning capabilities—they simply lack the capabilities to deal with all implications and eventualities of an inherently rich phenomenon such as communication. This holds in particular with such simple implementations as what we obtain in Section 3.5.

Generally, strategic communication is a research topic on its own, with controversial discussions (see, e.g., [132]) and many questions wide open. Crawford and Sobel [44] have considered the topic in a probabilistic setting, and Farrell and Rabin [62] have looked at related issues under the notion of cheap talk. Also within epistemic logic, formalizations of the information content of strategic communication have been suggested, e.g., by Gerbrandy [65]. But the topic is unclear, and we choose not to focus on it here.

To sum up, we make the following assumptions:

- the players are rational;
- they initially know their own preferences;
3.2 Preliminaries

- they are part of an interaction structure and can communicate their own preferences within any hyperarc they belong to;
- communication is truthful and synchronous, as in Chapter 2;
- the players have no knowledge other than what follows from these assumptions, and these are common knowledge.

These assumptions are reflected in the formalizations we make later on.

It is useful to clarify the relation between strategic games with interaction structures and pre-Bayesian games, introduced by Ashlagi et al. [10]. In these games, too, each player knows his payoff but does not know the payoffs of the other players. In our setup this private knowledge aspect of pre-Bayesian games can be trivially modelled by the empty interaction structure, or viewed as corresponding to our initial situation. Due to the different nature of these frameworks, however, the questions of interest are also different.

3.1.2 Plan of the chapter

This chapter is organized as follows. In the following Section 3.2, we review the basic definitions concerning strategic games, optimality notions and operators on restrictions of games. Next, in Section 3.3, we study the outcome of IESDS in the presence of an interaction structure. We first look at the outcome that is arrived at after all communication permitted in the given interaction structure has taken place, and then detail the outcomes obtained in any particular intermediate state of communication. The formulations we consider make no direct use of the notion of knowledge.

The connection with knowledge is made in Section 3.4, where we prove the outcomes we have obtained to be correct with respect to the epistemic framework from Chapter 2, in the sense that the outcomes capture exactly what the players can do given their partial knowledge of the game structure in any particular state. In Section 3.5, we describe two ways of implementing our procedures in a distributed setting. Finally, in Section 3.6, we suggest some future research directions.

3.2 Preliminaries

By a strategic game (in short, a game) for players \( N = \{1, \ldots, n\} \), where \( n > 1 \), we mean a tuple

\[(S_1, \ldots, S_n, \succ_1, \ldots, \succ_n),\]

where for each \( i \in N \),
- \( S_i \) is the non-empty, finite set of strategies available to player \( i \). We write \( S \) to abbreviate the set of strategy profiles: \( S = S_1 \times \cdots \times S_n \).
Introduction

As is customary in game theory, we denote the strategies of player $i$ by $s_i$, possibly with some superscripts. We also denote $i$’s strategy in a strategy profile $s \in S$ by $s_i$, and the tuple consisting of all other elements by $s_{-i}$, i.e.,

$$s_{-i} = (s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n).$$

Similarly, we use $S_{-i}$ to denote $S_1 \times \cdots \times S_{i-1} \times S_{i+1} \times \cdots \times S_n$, and for $s'_i \in S_i$ and $s_{-i} \in S_{-i}$ we write $(s'_i, s_{-i})$ to denote $(s_1, \ldots, s_{i-1}, s'_i, s_{i+1}, \ldots, s_n)$. Finally, we use $s'_i \succ_{s_{-i}} s_i$ as a notational alternative for $(s'_i, s_{-i}) \succ_i (s_i, s_{-i})$. That is, $s'_i \succ_{s_{-i}} s_i$ means that, if the other players choose the strategies given by $s_{-i}$, then player $i$ strictly prefers to choose $s'_i$ over $s_i$.

Fix now an initial strategic game $G := (S_1, \ldots, S_n, \succ_1, \ldots, \succ_n)$. We say that $(S'_1, \ldots, S'_n)$ is a restriction of $G$ if each $S'_i$ is a non-empty subset of $S_i$. We identify the restriction $(S_1, \ldots, S_n)$ with $G$.

To analyze iterated elimination of strategies from the initial game $G$, we view such procedures as operators on the set of restrictions of $G$. This set together with component-wise set inclusion forms a complete lattice.

For any restriction $G' := (S'_1, \ldots, S'_n)$ of $G$ and strategies $s_i, s'_i \in S_i$, we say that $s_i$ is strictly dominated by $s'_i$ on $S_{-i}'$ if $s'_i \succ_{s_{-i}} s_i$ for all $s_{-i}' \in S_{-i}'$. Then we introduce the following abbreviations ($\ell$ stands for “local” and $g$ stands for “global”; the terminology is from Apt [5]):

- $sd^\ell(s_i, G')$ which holds iff strategy $s_i$ of player $i$ is not strictly dominated on $S_{-i}'$ by any strategy from $S_i'$ (i.e., $\neg \exists s_i' \in S_i' \forall s_{-i}' \in S_{-i}' s'_i \succ_{s_{-i}} s_i$),

- $sd^g(s_i, G')$ which holds iff strategy $s_i$ of player $i$ is not strictly dominated on $S_{-i}'$ by any strategy from $S_i$ (i.e., $\neg \exists s_i' \in S_i \forall s_{-i}' \in S_{-i}' s'_i \succ_{s_{-i}} s_i$).

So in $sd^g$, the global version of strict dominance introduced by Chen et al. [40], it is stipulated that a strategy not be strictly dominated by a strategy from the initial game.

We call each relation of the form $sd^\ell$ or $sd^g$ an optimality notion. We say then that the optimality notion $\phi$ used by player $i$ is monotonic if for all restrictions $G''$ and $G'$ and strategies $s_i$,

$$G'' \subseteq G' \text{ and } \phi(s_i, G'') \text{ implies } \phi(s_i, G').$$

\footnote{We are interested in deterministic behavior, guided by deterministic knowledge, rather than the study of equilibria or randomizing players. For this reason, we do not consider mixed strategies here.}
3.3. Iterated strategy elimination

As noted in [32, 5], $sd^g$ is monotonic, while $sd^l$ is not (though their respective outcomes are equivalent in finite games, as discussed in the proof of Theorem 3.3.2).

Given an operator $T$ on a finite lattice $(D, \subseteq)$ and $k \geq 0$, by $T^k$ we denote the $k$-fold iteration of $T$, where $T^0 = D$ (so the iterations start “at the top”) and put $T^\infty := \bigcap_{k \geq 0} T^k$. We call $T$ monotonic if for all $D', D'' \subseteq D$,

$$D' \subseteq D'' \text{ implies } T(D') \subseteq T(D'').$$

Finally, an interaction structure $H$, as in Chapter 2, is a hypergraph on $N$, i.e., a set of non-empty subsets of $A \subseteq N$, called hyperarcs.

3.3 Iterated strategy elimination

In this section we define procedures for iterated elimination of strictly dominated strategies.

Let us fix a strategic game $G = (S_1, \ldots, S_n, \succ_1, \ldots, \succ_n)$ for players $N$, an interaction structure $H \subseteq 2^N \setminus \{\emptyset\}$, and an optimality notion $\phi$. In Section 3.3.1, we look at the outcome that the players can obtain after all communication permitted by $H$ has taken place, that is, when within each hyperarc of $H$ all of its members’ preferences have been communicated. In Section 3.3.2, we then look at the outcomes obtained in any particular intermediate state of communication.

The formulations we give here make no direct use of a formal notion of knowledge. The connection with a formal epistemic model is made in Section 3.4.

All the iterations of the considered operators start at $(S_1, \ldots, S_n)$.

3.3.1 Completed communication

Let us assume that within each hyperarc $A \in H$, all its members have shared all information about their preferences. We leave the exact definition of communication to Section 3.3.2 and the epistemic formalization to Section 3.4, and focus on an operational description for now.

For each group of players $G \in N$, let $S_G$ denote the set of those restrictions of $G$ which only restrict the strategy sets of players from $G$. That is,

$$S_G := \{(S'_1, \ldots, S'_n) \mid S'_i \subseteq S_i \text{ for } i \in G \text{ and } S'_i = S_i \text{ for } i \notin G\}.$$  

Now we introduce an elimination operator $T_G$ on each such set $S_G$, defined as follows. For each $G' = (S'_1, \ldots, S'_n) \in S_G$, 

$$T_G(G') := (S''_1, \ldots, S''_n),$$

where for all $i \in N$,

$$S''_i := \begin{cases} \{s_i \in S'_i \mid \phi(s_i, G')\} & \text{if } i \in G \\ S'_i & \text{otherwise.} \end{cases}$$
We call $T_G^\infty$ the **outcome of iterated elimination (of non-$\phi$-optimal strategies)** on $G$. We then define the restriction $G(H)$ of $G$ as

$$G(H) := (G(H)_1, \ldots, G(H)_n),$$

where for all $i \in \mathbb{N}$,

$$G(H)_i := T_{\{i\}}(\bigcap_{A : i \in A \in H} T_A^\infty)_i.$$ 

That is, the $i$th component of $G(H)$ is the $i$th component of the result of applying $T_{\{i\}}$ to the intersection of $T_A^\infty$ for all $A \in H$ containing $i$. We call $G(H)$ the **outcome of iterated elimination (of non-$\phi$-optimal strategies) with respect to** $H$. Note that $G(H)$ implicitly depends on $\phi$.

Let us “walk through” this definition to understand it better. Given a player $i$ and a hyperarc $A \in H$ such that $i \in A$, $T_A^\infty$ is the outcome of iterated elimination on $A$, starting at $(S_1, \ldots, S_n)$. The strategies of players from outside of $A$ are not affected by this process. This elimination process is performed for each hyperarc that $i$ is a member of. By intersecting the outcomes, i.e., by considering the restriction $\bigcap_{A : i \in A \in H} T_A^\infty$, one arrives at a restriction in which all such “group-wise” iterated eliminations have taken place. However, in this restriction some of the strategies of player $i$ may be non-$\phi$-optimal. They are eliminated using one application of the $T_{\{i\}}$ operator. We illustrate this process, and in particular this last step, in the following.

### 3.3.1. Example

Consider local strict dominance, $sd^\ell$, in the following three-player game $G$ where the payoffs of players 1 and 2 and those of players 1 and 3 respectively depend on each other’s actions, but the payoffs of player 2 and 3 are independent:

<table>
<thead>
<tr>
<th></th>
<th>Pl. 1 (U)</th>
<th>Pl. 2, 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pl. 1</td>
<td>(U)</td>
<td>(L, l)</td>
</tr>
<tr>
<td><em>D</em></td>
<td>1, 1, 1</td>
<td>0, 1, 0</td>
</tr>
<tr>
<td></td>
<td>0, 1, 1</td>
<td>1, 1, 0</td>
</tr>
</tbody>
</table>

That is, for example, if player 1 chooses strategy $U$, player 2 chooses $L$, and player 3 chooses $r$, then the payoff for player 1 is 0, player 2 gets 1, and player 3 gets 0. Now assume an interaction structure which reflects these payoff dependencies, i.e., a hypergraph $H = \{\{1, 2\}, \{1, 3\}\}$ (but note that this is just one particular example, in general the interaction structure need not reflect the payoff dependencies). We obtain $T_{\{1, 2\}}^\infty = (\{U, D\}, \{L\}, \{l, r\})$ and $T_{\{1, 3\}}^\infty = (\{U, D\}, \{L, R\}, \{l\})$. The restriction defined by these two outcomes is $(\{U, D\}, \{L\}, \{l\})$, and in the final step player 1 can now combine the results from his two independent interactions and eliminate his strategy $D$ by one application of $T_{\{1\}}$. The outcome of the whole process is thus $G(H) = (\{U\}, \{L\}, \{l\})$. 
In this example, the outcome with respect to the given interaction structure coincides with the outcome of customary IESDS on the fully specified game matrix. We should emphasize that this is not the case in general, and the purpose of this example is simply to illustrate how the operators work. Example 3.3.3 later on shows in a different setting how the interaction structure can influence the outcome.

Note that when \( H \) consists of the single hyperarc \( N \) that contains all the players, then for each player \( i \), \( \bigcap_{A:i \in A \in H} T^\infty_A \) reduces to \( T^\infty_N \), and this is closed under application of each operator \( T_{\{i\}} \). So then, indeed, \( G(H) = T^\infty_N \), that is, \( G(H) \) in this special case coincides with the customary outcome of iterated elimination of non-\( \phi \)-optimal strategies.

In general, this customary outcome is included in the outcome w.r.t. any hypergraph \( H \). This result is established in Theorem 3.3.2, and Example 3.3.3 shows a case where the inclusion is proper.

**3.3.2. Theorem.** For \( \phi \in \{sd^k, sd^\ell\} \) and for all hypergraphs \( H \), we have \( T^\infty_N \subseteq G(H) \).

**Proof.** First, consider \( \phi = sd^\ell \), and \( G \subseteq G' \subseteq N \). By definition, this implies that for all restrictions \( G' \) we have \( T_{G'}(G') \subseteq T_G(G') \). Since \( \phi \) is monotonic, so is the operator \( T_C \) for all \( C \subseteq N \). Hence by a straightforward induction \( T^\infty_G \subseteq T^\infty_{G'} \) for all \( G \subseteq N \), and consequently, for all players \( i \),

\[
T^\infty_N \subseteq \bigcap_{A:i \in A \in H} T^\infty_A. \tag{3.1}
\]

Hence, for all \( i \in N \),

\[
T^\infty_N = T_{\{i\}}(T^\infty_N) \subseteq T_{\{i\}}(\bigcap_{A:i \in A \in H} T^\infty_A),
\]

where the inclusion holds by the monotonicity of \( T_{\{i\}} \). Consequently \( T^\infty_N \subseteq G(H) \).

We now prove the same claim for \( \phi = sd^k \). We need to distinguish the \( T_C \) operator for \( \phi = sd^k \) and \( \phi = sd^\ell \) in the latter case \( T_{C,g} \) and in the former case \( T_{C,\ell} \). The reason that we use the latter operators is that they are monotonic and closely related to the former operators. Namely, as noted in [4], \( T^\infty_{N,\ell} = T^\infty_{N,g} \), and the proof carries over for \( N \) replaced by an arbitrary \( C \subseteq N \). Now fix an arbitrary \( i \in N \), then

\[
\bigcap_{A:i \in A \in H} T^\infty_{A,g} = \bigcap_{A:i \in A \in H} T^\infty_{A,\ell},
\]

and by (3.1) for \( \phi = sd^\ell \), \( T^\infty_{N,g} \subseteq \bigcap_{A:i \in A \in H} T^\infty_{A,\ell} \), so

\[
T^\infty_{N,\ell} = T^\infty_{N,g} \subseteq \bigcap_{A:i \in A \in H} T^\infty_{A,\ell}. \tag{3.2}
\]

Further, we have \( T^\infty_{N,\ell} = T^\infty_{N,g} \) and \( T^\infty_{N,g} = T_{\{i\},g}(T^\infty_{N,g}) \), so \( T^\infty_{N,\ell} = T_{\{i\},g}(T^\infty_{N,\ell}) \). Hence, by (3.2) and monotonicity of \( T_{\{i\},g} \),

\[
T^\infty_{N,\ell} = T_{\{i\},g}(T^\infty_{N,\ell}) \subseteq T_{\{i\},g}(\bigcap_{A:i \in A \in H} T^\infty_{A,\ell}).
\]
Chapter 3. Strategies in interaction structures

Also, for all \( i \in N \) and all restrictions \( \mathcal{G}' \) we have, by definition,

\[
T_{\{i\},\mathcal{G}'}(\mathcal{G}') \subseteq T_{\{i\},\ell}(\mathcal{G}'),
\]

so by the last inclusion

\[
T_{N,\ell}^{\infty} \subseteq T_{\{i\},\ell}(\bigcap_{A: i \in A \in H} T_{A,\ell}^{\infty}).
\]

Consequently, \( T_{N,\ell}^{\infty} \subseteq \mathcal{G}(H) \), as desired. \( \square \)

The inclusion proved in this result cannot be reversed, even when each pair of players shares a hyperarc. The following Example 3.3.3 also proves the intuition that such a graph structure can be less informative than a proper (non-degenerate) hypergraph structure.

3.3.3. EXAMPLE. Consider the following strategic game with three players. The payoffs of player 1 and 2 here only depend on each other’s choices, and the payoffs of player 3 only depend on the choices of player 2 and 3:

<table>
<thead>
<tr>
<th></th>
<th>Pl. 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pl. 1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>L</td>
</tr>
<tr>
<td>U</td>
<td>0,1</td>
</tr>
<tr>
<td>D</td>
<td>1,0</td>
</tr>
</tbody>
</table>

Payoff of players 1 and 2

<table>
<thead>
<tr>
<th></th>
<th>Pl. 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pl. 3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>L</td>
</tr>
<tr>
<td>A</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>1</td>
</tr>
</tbody>
</table>

Payoff of player 3

If we assume the hypergraph \( H \) that consists of the single hyperarc \( \{1, 2, 3\} \), then the outcome of iterated elimination of non-\( \phi \)-optimal strategies w.r.t. \( H \) is the customary outcome which equals \( (\{D\}, \{R\}, \{A\}) \). Indeed, player 1 can eliminate his strictly dominated strategy \( U \), then player 2 can eliminate \( L \), and subsequently player 3 can eliminate \( B \).

In contrast, if the hypergraph consists of all pairs of players, so \( H = \{\{1, 2\}, \{2, 3\}, \{1, 3\}\} \), then the outcome of iterated elimination of non-\( \phi \)-optimal strategies w.r.t. \( H \) equals \( (\{D\}, \{R\}, \{A, B\}) \).

Informally, the reason for this difference is that in the latter case, player 3 can eliminate \( B \) only using the fact that player 2 eliminated \( L \), but this information is available only to players 1 and 2.

To familiarize ourselves further with our definitions, we establish the following intuitive monotonicity result. Say that \( H' \) extends \( H \) just if for each \( A \in H \) there is \( A' \in H' \) such that \( A \subseteq A' \).

3.3.4. PROPOSITION. If \( H' \) extends \( H \) and \( T \) is monotonic, then \( \mathcal{G}(H') \subseteq \mathcal{G}(H) \).
3.3. Iterated strategy elimination

Proof. We prove the following stronger proposition: If \( H' \) extends \( H \) and for each \( G \in \{ \{ i \} \mid i \in N \} \cup H \cup H', \) \( T_G \) is monotonic, then \( G(H') \subseteq G(H) \).

First note that for all restrictions \( G' \):

\[
G \subseteq G' \implies T_{G'}(G') \subseteq T_G(G'). \tag{3.3}
\]

To see this, suppose that \( G \subseteq G' \). Then for all \( i \in N \), either

- \( i \in G \), in which case \( i \in G' \), so \( T_G(G')_i = T_{G'}(G')_i \), or
- \( i \notin G \), in which case \( T_G(G')_i = G'_i \supseteq T_{G'}(G')_i \).

In each case, \( T_{G'}(G')_i \subseteq T_G(G')_i \).

From (3.3) and the monotonicity of \( T_G \) and \( T_{G'} \), it follows that:

\[
G \subseteq G' \implies T_G^\infty \subseteq T_{G'}^\infty. \tag{3.4}
\]

Now we show that if \( H' \) extends \( H \), then \( \bigcap_{A' \in A' \in H'} T_{A'}^\infty \subseteq \bigcap_{A \in A \in H} T_A^\infty \). Fix \( i \in N \) and take some \( s_i \notin (\bigcap_{A \in A \in H} T_A^\infty)_i \). Then there is \( A \in H \) such that \( i \in A \) and \( s_i \notin (T_A^\infty)_i \). Then since \( H' \) extends \( H \), there is \( A' \in H' \) such that \( A \subseteq A' \), so by (3.4), \( s_i \notin (T_{A'}^\infty)_i \supseteq \bigcap_{A' \in A' \in H'} T_{A'}^\infty \).

So, since each \( T_{(i)} \) is monotonic,

\[
G(H')_i = T_{(i)} \left( \bigcap_{A' \in A' \in H'} T_{A'}^\infty \right)_i \subseteq T_{(i)} \left( \bigcap_{A \in A \in H} T_A^\infty \right)_i = G(H)_i. \]

3.3.2 Intermediate states

The setting considered in Section 3.3.1 corresponds to a state where within each hyperarc, all its members have shared all information about their preferences. Given the game \( G \) and the hypergraph \( H \), the outcome \( G(H) \) defined there thus reflects, assuming the process described in Section 3.1.1, what strategies players who initially know only their own preferences can eliminate if they communicate all they can communicate in \( H \). We now define formally what communication we assume possible, and then look at intermediate states, where only certain preferences have been communicated.

Any player \( i \) can communicate his preferences to any \( A \in H \) with \( i \in A \). We take a message by \( i \) to consist of a preference statement \( s'_i \succ_{s_i} s_i \) for \( s_i, s'_i \in S_i \) and \( s_{-i} \in S_{-i} \). We denote such a message by \( (i, A, s'_i \succ_{s_i} s_i) \), and require that \( i \in A \) and that it is truthful with respect to the given game \( G \), that is, that indeed \( s'_i \succ_{s_i} s_i \) in \( G \). Note that the fact that \( i \) is the sender is, strictly speaking, never used. Thus, in accordance with the interpretation of communication described in Section 3.1.1, we may drop the sender and simply say “the players in \( A \) commonly observe that \( s'_i \succ_{s_i} s_i \)”.

An intermediate state now is given by the set \( M \) of messages which have been communicated.
We now adjust the definition of an optimality notion to account for intermediate states. An **intermediate optimality notion** $\phi_{G,M}$ (derived from an optimality notion $\phi$) uses only information shared among the group $G$ in the intermediate state given by $M$. That is, with singleton $G = \{i\}$ only $i$’s preferences are used, and with larger $G$ only preferences contained in messages to $G$ are used. For example, $sd_{G,M}^G(s_i, G')$ holds if

\[-\exists s'_i \in S_i \forall s_{-i} \in S'_{-i} s'_i \succ_{s_{-i}} s_i \quad \text{if} \ G = \{i\}\]

\[-\exists s'_i \in S_i \forall s_{-i} \in S'_{-i} M \models G \models s'_i \succ_{s_{-i}} s_i \quad \text{otherwise},\]

where by $M \models G \models s'_i \succ_{s_{-i}} s_i$ we mean that $s'_i \succ_{s_{-i}} s_i$ is entailed by the messages contained in $M$ which $G$ received. Specifically, $M \models G \models s'_i \succ_{s_{-i}} s_i$ iff there are $(\cdot, G^k, s'_i \succ_{s_{-i}} s_{k+1}^t) \in M$ for $k \in \{1, \ldots, \ell - 1\}$ such that $G^k \supseteq G$, $s_i^1 = s'_i$ and $s'_i = s_i$.

We can then define a generalization of the $T_G$ operator as follows:

$$T_{G,M}(G') := (S_1', \ldots, S_n'),$$

where $G' = (S_1', \ldots, S_n')$ and for all $i \in N$

$$S_i' := \{s_i \in S_i \mid \phi_{G,M}(s_i, G')\}.$$

Note that, as before, $S_i'$ remains unchanged if $i \notin G$, since then $\phi_{G,M}(s_i, G')$ always holds. Indeed, for it to be false, there would have to be some message $(i, G, \cdot) \in M$, which would imply $i \in G$.

Similarly, we now define the **outcome of iterated elimination (of non-$\phi$-optimal strategies)** with respect to $H, M$ to be the restriction $G(H, M)$, where for $i \in N$

$$G(H, M)_i := T_{\{i\},M} \left( \bigcap_{A \ni i \in \overline{A}} T_{\overline{A},M} \right)_i.$$

Here, $\overline{H}$ denotes the closure of $H$ under intersection. That is, $H \subseteq \overline{H}$ and if $A, A' \in \overline{H}$ then also $A \cap A' \in \overline{H}$. This is necessary because certain information may be entailed by messages sent to different hyperarcs. For example, with $(j, A, s''_{s-j} \succ_{s_{-j}} s'_j), (j, A', s'_j \succ_{s_{-j}} s_j) \in M$, the combined information that $s''_{s_{-j}} \succ_{s_{-j}} s_j$ is available to $A \cap A'$. Notice that this formulation leaves room for optimizations. For example, one start by looking at $M$ and only consider groups to which there actually exist messages, but we do not focus on this issue here.

It is easy to see that in the case where the players have communicated all there is to communicate, i.e., for

$$M_{H}^{\text{all}} := \{(i, A, s'_i \succ_{s_{-i}} s_i) \mid i \in N, A \in H, s_i, s'_i \in S_i \text{ with } s'_i \succ_{s_{-i}} s_i \text{ in } G\},$$

the intermediate outcome coincides with the previously defined outcome:

$$G(H, M_{H}^{\text{all}}) = G(H).$$
This corresponds to the intuition that $G(H)$ captures the elimination process when all possible communication has taken place. In particular, all entailed information has also been communicated in $M^\text{all}_H$, which is why we did not need to consider $\overline{H}$ in Section 3.3.1.

Again, we “walk through” the definition of $G(H, M)$. First, a local elimination process is run on each hyperarc of $\overline{H}$, using only information which has been communicated there (which now no longer covers all members’ preferences, but only the ones according to the intermediate state $M$). Then, in the final step, each player combines his insights from all hyperarcs of which he is a member, and he eliminates any strategies that he thereby learns not to be optimal.

3.3.5. Example. Consider again the game $G$ from Example 3.3.1, and the initial state where $M = \emptyset$.

We have $T^\infty_{A,M} = G$ for all $A \in \overline{H}$, that is, without communication no strategy can “commonly” be eliminated. However, players 2 and 3 can “privately” eliminate one of their strategies each, since each knows his own preferences. This fact and the effect that this elimination cannot be iterated upon by other players are captured in the final step performed by these respective players. The results of the final steps are thus

$$
T_{\{1\},M}(\bigcap_{A:1 \in A} T^\infty_{A,M}) = (\{U, D\}, \{L\}, \{l, r\}),
$$

$$
T_{\{2\},M}(\bigcap_{A:2 \in A} T^\infty_{A,M}) = (\{U, D\}, \{L\}, \{l, r\}),
$$

$$
T_{\{3\},M}(\bigcap_{A:3 \in A} T^\infty_{A,M}) = (\{U, D\}, \{L\}, \{l\}),
$$

so the overall outcome is

$$
G(H, M) = (\{U, D\}, \{L\}, \{l\}).
$$

Consider now the intermediate state

$$
M' = \{(2, \{1, 2\}, L \succ_{s-2} R) \mid s_{-2} \in S_{-2}\},
$$

that is, a state where player 2 has shared with player 1 the information that for any joint strategy of players 1 and 3, he prefers his strategy $L$ over $R$. Then only the result of player 1 changes:

$$
T_{\{1\},M'}(\bigcap_{A:1 \in A} T^\infty_{A,M'}) = (\{U, D\}, \{L\}, \{l, r\}),
$$

while the other results and the overall outcome remain the same. If additionally player 3 communicates all his information in the hyperarc he shares with player 1, that is, if the intermediate state is

$$
M'' = M' \cup \{(3, \{1, 3\}, l \succ_{s-3} r) \mid s_{-3} \in S_{-3}\},
$$


then player 1 can combine all the received information and obtain
\[ T_{\{1\},M'}(\bigcap_{A:1 \in A \in \Pi} T_{A,M}^\infty) = (\{U\}, \{L\}, \{I\}). \]
This is also the overall outcome \( G(H, M'') \), which coincides with the outcome \( G(H, M''|H) \) where all information has been communicated.

Let us now illustrate the importance of using entailment in intermediate optimality notions and \( \Pi \) (rather than \( H \)) in the definition of \( G(H, M) \).

3.3.6. Example. We look at a game involving four players, but we are only interested in the preferences of two of them. The other two players serve merely to create different hyperarcs. The strategies and payoffs of player 1 and 2 are as follows:

<table>
<thead>
<tr>
<th></th>
<th>L</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>3.0</td>
<td>1.1</td>
</tr>
<tr>
<td>B</td>
<td>2.0</td>
<td>1.1</td>
</tr>
<tr>
<td>C</td>
<td>1.1</td>
<td>0.0</td>
</tr>
<tr>
<td>D</td>
<td>0.0</td>
<td>5.1</td>
</tr>
</tbody>
</table>

For player 3 and 4 we assume a “dummy” strategy each, denoted \( X \) and \( Y \).

Consider the hypergraph
\[ H = \{\{1, 2, 3\}, \{1, 2, 4\}\} \]
and the intermediate state
\[ M = \{(1, \{1, 2, 3\}, A \succ_{LXY} B), (1, \{1, 2, 4\}, B \succ_{LXY} C), (1, \{1, 2, 3\}, A \succ_{RXY} C)\}. \]
The fact that player 1, independently of what the remaining players do, strictly prefers \( A \) over \( C \) is not explicit in these pieces of information, but it is entailed by them, since \( A \succ_{LXY} B \) and \( B \succ_{LXY} C \) imply \( A \succ_{LXY} C \). However, this combination of information is only available to \( \{1, 2, 3\} \cap \{1, 2, 4\} \).

Player 2 can make use of this fact that \( C \) is dominated, and eliminate his own strategy \( L \). If we now look at a state where player 2 has communicated his relevant preferences,
\[ M' = M \cup \{(2, \{1, 2, 3\}, R \succ_{aXY} L) | a \in \{A, B, D\}\}, \]
we notice that player 1, in turn, can eliminate \( A \) and \( B \), but only building upon the initial combination of information available to \( \{1, 2, 3\} \cap \{1, 2, 4\} \). There is no single hyperarc in the original hypergraph which has all the required information available. It thus becomes clear that we need to take into account iterated elimination on intersections of hyperarcs.
3.4 Epistemic foundation

In this section, we provide an epistemic foundation for our framework. The aim is to prove that the definition of the outcome \( G(H, M) \) correctly captures what strategies the players can eliminate using all they “know”, in a formal sense.

We proceed as follows. First, in Section 3.4.1, we briefly introduce an epistemic model formalizing the players’ knowledge. We draw upon results from Chapter 2. In Section 3.4.2, we give a general epistemic formulation of strict dominance and argue that it correctly captures the notion. Section 3.4.2 contains the main result of our epistemic analysis, namely that the outcome \( G(H, M) \) indeed yields the outcome stipulated by the epistemic formulation.

We focus on the global version of strict dominance, \( sd^g \), mainly because the presentation is then more concise. However, our results about its outcomes carry over to the local version due to the equivalence result mentioned in the proof of Theorem 3.3.2.

3.4.1 Epistemic language and states

Again, we assume a fixed game \( G \) with strategies \( S_i \) for each player \( i \), and a hypergraph \( H \) representing the interaction structure. Analogously to Chapter 2, we use a propositional epistemic language with a set \( At \) of atoms which is divided into disjoint subsets \( At_i \), one for each player \( i \), where
\[
At_i = \{ s'_i \succ s_{-i}, s_i \mid s'_i, s_i \in S_i, s_{-i} \in S_{-i} \}.
\]
The set \( At \) describes all possible relative preferences between pairs of strategies. We consider the usual connectives \( \land \) and \( \lor \) (but not the negation \( \neg \)), and a common knowledge operator \( C_G \) for any group \( G \subseteq N \) of players. As in Chapter 2, we write \( K_i \) for \( C_i \). By \( L^+ \) we denote the set of formulas built from the atoms in \( At \) using these two connectives and knowledge operators.

A valuation \( V \) is a subset of \( At \), consisting of those atoms that are assumed true. A concrete game \( G \) induces exactly one valuation which correctly represents it, but since we need to model that players may not have full knowledge of the game and may consider other preference orders possible, we need to allow other options. However, since the setup is commonly known, players will only consider valuations possible which indeed may reflect the preference ordering of some game. Therefore, we require valuations \( V \) to be such that for each \( i \) and each \( s_{-i} \in S_{-i} \), the restriction \( V \cap \{ s \succ s_{-i} \} \) represents a strict partial order. For example, \( \{ s \succ a \ t \} \) is a valuation (given a game with appropriate strategy sets), while \( \{ s \succ a \ t, t \succ a \ u \} \) and \( \{ s \succ a \ t, t \succ a \ s \} \) are not. Note that such valuations satisfy the properties (v1) and (v2) from Chapter 2, so the framework discussed there can be applied.

Recall from Section 3.3.2 that a message from player \( i \) to \( A \in H \) has the form \( (i, A, s'_i \succ s_{-i}, s_i) \), where \( i \in A, s_i, s'_i \in S_i \), and \( s_{-i} \in S_{-i} \). Truthfulness now
depends on the particular valuation under consideration. Concretely, a message \((\cdot, \cdot, p)\) is **truthful** with respect to a valuation \(V\) if, indeed, \(p \in V\).

A **state**, or **possible world**, is a pair \((V, M)\), where \(V\) is a valuation and \(M\) is a set of truthful (with respect to \(V\)) messages.

Our setting is an instance of the framework defined in Chapter 2, and the formal **semantics** is as defined there (Section 2.2). We repeat here only the intuition that \(C_G \varphi\) means that \(\varphi\) is **common knowledge** among \(G\), that is, everybody in \(G\) knows \(\varphi\), everybody knows that everybody knows \(\varphi\), etc. In particular, \(K_i \varphi\) means that player \(i\) knows \(\varphi\). Our assumptions that player \(i\) from the beginning knows the true facts in \(At_i\), and that the basic assumptions from Section 3.1.1 are commonly known among the players, are reflected in the setup we have described.

### 3.4.2 Correctness result

We here use our insights from Chapter 2 in order to prove that the **T** operator defined in Section 3.3 is correct with respect to an epistemic formulation of our setting.

We start by giving an epistemic formula describing the global version of iterated elimination of strictly dominated strategies. Note that, in contrast to the formulation in Section 3.2, this formula states when a strategy is known to be strictly dominated.

We define, for \(i \in N\) and \(s_i \in S_i\),

\[
\begin{align*}
\text{dom}^1(s_i) & := K_i \bigvee_{s'_i \in S_i, s_{-i} \in S_{-i}} s'_i \succ_{s_{-i}} s_i, \\
\text{dom}^{\ell+1}(s_i) & := K_i \bigvee_{s'_i \in S_i, s_{-i} \in S_{-i}} (s'_i \succ_{s_{-i}} s_i \vee \bigvee_{j \in N \setminus \{i\}} \text{dom}^\ell(s_j)).
\end{align*}
\]

That is, in the base case, player \(i\) knows that \(s_i\) is strictly dominated if \(i\) knows that there is an alternative strategy \(s'_i\) which, for all joint strategies of the other players, is strictly preferred. Furthermore, after iteration \(\ell + 1\), \(i\) knows that \(s_i\) is strictly dominated if \(i\) knows that there is an alternative strategy \(s'_i\) such that, for all joint strategies \(s_{-i}\) of the other players, either \(s'_i\) is strictly preferred or some strategy \(s_j\) in \(s_{-i}\) is already known by \(j\) to be strictly dominated after iteration \(\ell\).

We restrict attention to formulas \(\text{dom}^\ell(s_i)\) with \(\ell \in \{1, \ldots, \hat{\ell}\}\), where \(\hat{\ell} = \sum_{i \in N} |S_i|\). By the semantics of the considered formulas, there is some \(\ell\) within this range such that for all \(\ell' \geq \ell\), \(\text{dom}^{\ell'}\) is equivalent to \(\text{dom}^\ell\). To reflect the fact that this can be seen as the outcome of the iteration, we denote \(\text{dom}^\ell\) by \(\text{dom}^\infty\).

As a first connection with the **T** operator defined in Section 3.3, we have the following epistemic counterpart of Proposition 3.3.4. This is due to the fact that, intuitively, if we look at states where all communication allowed by a given
hypergraph has taken place, then knowledge (of positive formulas) can only grow as that hypergraph grows.

3.4.1. **Proposition.** If \( H' \) extends \( H \), then for any \( i \in N \) and \( s_i \in S_i \),

\[
(V, M_H^{\text{all}}) \models \text{dom}^\infty(s_i) \text{ implies } (V, M_{H'}^{\text{all}}) \models \text{dom}^\infty(s_i),
\]

where \( M^{\text{all}} \) is as defined in Section 3.3.2.

*Proof.* Follows from Lemma 2.3.3 and the fact that \( \text{dom}^\infty(s_i) \in \mathcal{L}^+ \).

We now proceed to the main result of this section. We prove that the non-epistemic formulation of iterated elimination of non-\( sd^g \)-optimal strategies, as given in Section 3.3, is correct with respect to the epistemic formulation of strict dominance.

3.4.2. **Theorem.** For any strategic game \( \mathcal{G} \), hypergraph \( H \), set of messages \( M \) (truthful with respect to \( \mathcal{G} \)), and \( i \in N \),

\[
\mathcal{G}(H, M)_i = \{ s_i \in S_i \mid (V, M) \not\models \text{dom}^\infty(s_i) \},
\]

where \( V \) is the valuation induced by \( \mathcal{G} \).

In order to prove this result, we need some preparatory steps.

3.4.3. **Lemma.** For any \( \ell \geq 1 \), \( i \in N \), \( s_i \in S_i \), and state \((V, M)\),

\[
(V, M) \models \text{dom}^{\ell+1}(s_i)
\]

iff \((V, M) \models \bigvee_{s'_i \in S_i} \left( (K_i s'_i \succ s_{-i} s_i) \lor \bigwedge_{A_i \in A \subseteq N \setminus \{i\}} \bigwedge_{j \in A \setminus \{i\}} C_A \text{dom}^{\ell}(s_j) \right)\).

*Proof.* We have

\[
(V, M) \models \text{dom}^{\ell+1}(s_i)
\]

iff (by definition)

\[
(V, M) \models K_i \bigwedge_{s'_i \in S_i} \bigwedge_{s_{-i} \in S_{-i}} \left( (K_i s'_i \succ s_{-i} s_i) \lor \bigwedge_{j \in N \setminus \{i\}} \text{dom}^{\ell}(s_j) \right)
\]

iff (by Lemma 2.3.5)

\[
(V, M) \models \bigvee_{s'_i \in S_i} \bigwedge_{s_{-i} \in S_{-i}} \left( K_i s'_i \succ s_{-i} s_i \lor \bigwedge_{j \in N \setminus \{i\}} K_i \text{dom}^{\ell}(s_j) \right)
\]

iff (by Lemma 2.3.8 and Theorem 2.3.9 since \( \text{dom}^{\ell}(s_j) = K_j(\cdots) \)).

\[\square\]
3.4.4. **Lemma.** For any $\ell \geq 1$, $i \in A \subseteq \overline{H}$, $s_i \in S_i$, and state $(V, M)$,

$$s_i \notin T_{A,M}^\ell(S_1, \ldots, S_n) \text{ iff } (V, M) \models C_A \text{dom}^\ell(s_i).$$

**Proof.** By induction on $\ell$. The base case follows straightforwardly from the definitions. Now assume the claim holds for $\ell$. Then, focusing on the interesting case where $A \neq \{i\}$, we have the following chain of equivalences:

$$s_i \notin T_{A,M}^{\ell+1}(S_1, \ldots, S_n) \text{ iff } (\text{by definition}) s_i \notin T_{A,M}^\ell(S_1, \ldots, S_n) \text{ or } \neg sd_{A,M}^\ell(s_i, T_{A,M}^\ell(S_1, \ldots, S_n))$$

$$\text{iff } (\text{by monotonicity of } sd^\ell) \neg sd_{A,M}^\ell(s_i, T_{A,M}^\ell(S_1, \ldots, S_n))$$

$$\text{iff } (\text{by definition}) \exists s' \in S_i \forall s_{-i} \in T_{A,M}^\ell(S_1, \ldots, S_n) - i M \models s'_{i} \succ_{s_{-i}} s_i$$

$$\text{iff } \exists s' \in S_i \forall s_{-i} \in S_{-i} M \models s'_{i} \succ_{s_{-i}} s_i \text{ or } s_{-i} \notin T_{A,M}^\ell(S_1, \ldots, S_n) - i$$

$$\text{iff } \exists s' \in S_i \forall s_{-i} \in S_{-i} M \models s'_{i} \succ_{s_{-i}} s_i \text{ or } \exists j \in A \setminus \{i\} s_j \notin T_{A,M}^\ell(S_1, \ldots, S_n)$$

$$\text{iff } (\text{by induction hypothesis}) \exists s' \in S_i \forall s_{-i} \in S_{-i} M \models s'_{i} \succ_{s_{-i}} s_i \text{ or } \exists j \in A \setminus \{i\} (V, M) \models C_A \text{dom}^\ell(s_j)$$

$$\text{iff } (\text{by Lemma 2.3.8}) \exists s' \in S_i \forall s_{-i} \in S_{-i} (V, M) \models C_A s'_{i} \succ_{s_{-i}} s_i \text{ or } \exists j \in A \setminus \{i\} (V, M) \models C_A \text{dom}^\ell(s_j)$$

$$\text{iff } (V, M) \models \bigvee_{s' \in S_i \forall s_{-i} \in S_{-i}} (C_A s'_{i} \succ_{s_{-i}} s_i \lor \bigvee_{j \in A \setminus \{i\}} C_A \text{dom}^\ell(s_j))$$

$$\text{iff } (V, M) \models C_A \bigvee_{s' \in S_i \forall s_{-i} \in S_{-i}} (s'_{i} \succ_{s_{-i}} s_i \lor \bigvee_{j \in A \setminus \{i\}} \text{dom}^\ell(s_j))$$

$$\text{iff } (V, M) \models C_A \text{dom}^{\ell+1}(s_i).$$

We are now ready to prove the main result. 
3.5. Distributed implementation

Proof of Theorem 3.4.2. We have:

\[ s_i \not\in \mathcal{G}(H, M) \]

iff (by definition)

\[ s_i \not\in T_{(i),M}(\bigcap_{A_i \in A} T^\infty_{A_i,M}) \]

iff \( s_i \in T_{(i),M}(\bigcap_{A_i \in A} T^\infty_{A_i,M}(S_1, \ldots, S_n)) \)

iff \( \exists s'_i \in S_i \forall s_{-i} \in \bigcap_{A_i \in A} T^\infty_{A_i,M}(S_1, \ldots, S_n) - i \ s'_i \succ s_{-i} \ s_i \)

iff \( \exists s'_i \in S_i \forall s_{-i} \in S_{-i} s'_i \succ s_{-i} \ s_i \) or

\[ s_{-i} \not\in \bigcap_{A_i \in A} T^\infty_{A_i,M}(S_1, \ldots, S_n) - i \]

iff \( \exists s'_i \in S_i \forall s_{-i} \in S_{-i} s'_i \succ s_{-i} \ s_i \) or

\[ \exists A : i \in A \in T^\infty_{A_i,M}(S_1, \ldots, S_n) - i \]

iff \( \exists s'_i \in S_i \forall s_{-i} \in S_{-i} s'_i \succ s_{-i} \ s_i \) or

\[ \exists A : i \in A \in T^\infty_{A_i,M}(S_1, \ldots, S_n) \]

iff (by Lemma 3.4.4)

\[ \exists s'_i \in S_i \forall s_{-i} \in S_{-i} s'_i \succ s_{-i} \ s_i \] or

\[ (V, M) \models \bigvee_{A_i \in A} \bigvee_{j \in A \setminus \{i\}} C_A \text{dom}^\infty(s_j) \]

iff (since \( s'_i \succ s_{-i} \ s_i \in A_i \))

\[ \exists s'_i \in S_i \forall s_{-i} \in S_{-i} (V, M) \models K_i s'_i \succ s_{-i} \ s_i \] or

\[ (V, M) \models \bigvee_{A_i \in A} \bigvee_{j \in A \setminus \{i\}} C_A \text{dom}^\infty(s_j) \]

iff \( (V, M) \models \bigvee_{s'_i \in S_i} \bigvee_{s_{-i} \in S_{-i}} ((K_i s'_i \succ s_{-i} s_i) \ \vee \bigvee_{A_i \in A} \bigvee_{j \in A \setminus \{i\}} C_A \text{dom}^\infty(s_j)) \)

iff (by Lemma 3.4.3)

\[ (V, M) \models \text{dom}^\infty(s_i). \]

\[ \square \]

3.5 Distributed implementation

An epistemic model such as the one we have built and used in Chapter 2 and Section 3.4 allows us to reason about the players’ knowledge. However, such a model always takes the perspective of an outside observer, the modeler (in this case us), and tells us what the players can be said to know, assuming they are perfect reasoners. It thus ascribes knowledge to the players, without really telling us (or them, for that matter) how exactly they might arrive at that knowledge. Similarly, the \( T \) operator from Section 3.3 is formulated in a centralized way. How does the situation look from the players’ point of view, and what reasoning mechanisms might they use?
Obviously, it cannot be the case that each player simply maintains a copy of our central model, or something equivalent to it, because the central model contains information about the whole system and all players, which is not available to an individual player. Of course, there may be situations where the players would have access to such a central model; for example, a virtual world might provide an interface to the control programs of its simulated inhabitants through which they can query a centralized model maintained by the world, and thus find out what they (can be said to) know. We explore this setting in Chapter 4, but in a truly distributed system this is not possible.

The aim of this section is to “localize” both the $T$ operator and the epistemic model to obtain algorithms which can be executed by any player $i$ himself. We straightforwardly see that they correspond to “$i$’s part” of the centralized versions, and are in that sense correct for the elimination outcome of $i$’s strategies, and for $i$’s knowledge. So, in Section 3.5.1, we directly implement an iterated elimination process localized to player $i$, and we easily see that it coincides with the centralized version of the $T$ operator on $i$’s strategies. Correctness is thus implied by Section 3.4, in the sense that the implementation really eliminates all strategies a player can eliminate, given his theoretically ascribed knowledge. In Section 3.5.2, we follow the approach we have described in [155, 157] and called explicit knowledge programming, which involves encapsulating epistemic information and algorithms in a knowledge module local to any player $i$. This module processes the events that the player observes and enables him to transparently evaluate a certain class epistemic formulas talking about $i$’s knowledge, which includes the $\text{dom}^i(s_i)$ formulas. Again using the results from Section 3.4, we see that the knowledge module is correct with respect to the given class of formulas. Exploiting the restrictions of our setting, the knowledge module is computationally simple, as opposed to, for example, a general, full-blown epistemic logic theorem prover. In this sense, our approach is in accordance with Hayes [77]: such a knowledge module is a system which “[has] a logical inference structure—[is] making deductively valid inferences—without being a classical uniform theorem-prover which just ‘grinds clauses together’.” See Section 3.6 for some more discussion on this approach.

Common to both approaches is that the events a player observes need to be kept track of. So we assume that the program of any player $i$ stores and can at any time access the initial strategy sets $(S_1, \ldots, S_n)$ of the given game $G$, the given interaction structure $H$, $i$’s own preferences $\succ_i$ induced by $G$, as well as the messages $M_i$ he has observed. For the sake of clarity, we use $C(M_i)$ to denote the transitive closure of the messages that have been observed by $i$. That is, $M_i \subseteq C(M_i)$, and if $(j, A, s''_j \succ_{s-j} s'_j), (j, A', s'_j \succ_{s-j} s_j) \in C(M_i)$, then also $(j, A \cap A', s''_j \succ_{s-j} s_j) \in C(M_i)$.

Note that the algorithms we describe are symmetric in the sense of Chapter 1, and thus JCSP lends itself as implementation platform.
3.5. Distributed implementation

Algorithm 1: Computing $\mathcal{G}(H, M)$, using a $T$ operator implementation

// compute $\bigcap_{A : i \in A \in \pi} T_{A,M}^\infty$
1 foreach $A$ with $i \in A \in \overline{H}$ do
2 $(S'_{1,A}, \ldots, S'_{n,A}) := (S_1, \ldots, S_n)$;
3 repeat
4 $changed := false$;
5 $(S''_{1,A}, \ldots, S''_{n,A}) := (S'_{1,A}, \ldots, S'_{n,A})$;
6 foreach $j \in A$ and $s_j \in S'_{j,A}$ do
7 if $\exists s_j' \in S'_{j,A} \forall s_{-j} \in S''_{-j,A} (j, A, s_j' \succ s_{-j}, s_j) \in C(M_i)$ then
8 $S''_{j,A} := S''_{j,A} \setminus \{s_j\}$;
9  $changed := true$;
10 end
11 $(S''_{1,A}, \ldots, S''_{n,A}) := (S''_{1,A}, \ldots, S''_{n,A})$;
12 until not changed;
13 end
14 $(S'_1, \ldots, S'_n) := \bigcap_{A : i \in A \in \pi} (S'_{1,A}, \ldots, S'_{n,A})$;

// compute $T_{(i),M}$
15 $S''_i := S'_i$;
16 foreach $s_i \in S'_i$ do
17 if $\exists s_i' \in S'_i \forall s_{-i} \in S'_{-i} s_i' \succ s_{-i}, s_i$ then
18 $S''_i := S''_i \setminus \{s_i\}$;
19 end
20 end
21 return $S''_i$;

3.5.1 $T$ operator approach

Algorithm 1 describes an implementation of the $T$ operator as defined in Section 3.3.2. This implementation can straightforwardly be seen to execute directly the definition of the $T$ operator; the only noteworthy change occurs in line 7, where we have $M_i$ instead of $M$. This, however, does not make any difference, since with $i \in A$ we have $(j, A, s_j' \succ s_{-j}, s_j) \in M_i$ if and only if $(j, A, s_j' \succ s_{-j}, s_j) \in M$. Intuitively speaking, the evaluation of $T_{A,M}^\infty$ only uses information shared by all members of $A$, and thus locally available to each member.

The implementation is certainly not the most efficient one, for example, one might first look at $M_i$ in order to see which hyperarcs of $\overline{H}$ even need to be considered. But for our purposes this suffices, and we now focus on the knowledge module approach.
3.5.2 Knowledge module approach

In the alternative, modular approach, the player program explicitly uses the epistemic formulation defined in Section 3.4.2, evaluating epistemic formulas of the form $\text{dom}^i$ in order to test which strategies are known to be dominated. Since we are here, as explained in Section 3.1.1, not examining exactly why certain communications are performed, we do not discuss the whole player program in detail. It can be thought of as a main loop consisting of communication statements and, whenever communication has taken place, tests involving $\text{dom}^i$ formulas in order to determine what strategies can currently be eliminated. Whenever the player program encounters such an epistemic formula, it calls a function to evaluate it.

This evaluation function is what we focus on here. It is provided by the player’s knowledge module, which also keeps track of the relevant information (i.e., the observed messages). We here provide an evaluation function for such a knowledge module, using the results from Chapter 2. It correctly evaluates a more general class of formulas than only the $\text{dom}^i$: For any $\varphi \in \mathcal{L}^+$, at any intermediate state $M$, it can efficiently compute whether $(V, M) \models K_i \varphi$, where $V$ is as induced by $\mathcal{G}$.

Note that, even though $(V, M)$ refers to all players, the knowledge module is only allowed to use the information available to $i$, that is, $\succ_i$ and $M_i$. For this reason, even though the formulation can be thought of as a model checking problem, in general it is closer to a validity check: $i$ has to check whether the formula in question, $K_i \varphi$, holds in all models compatible with his information; or, if we represent his information as formulas $\psi_1, \ldots, \psi_t$, whether $\models \neg(\psi_1 \land \ldots \land \psi_t) \lor K_i \varphi$.

While the exact relationship between model checking and testing for validity depends on the concrete formalism, in general, testing for validity is intractable [74]. With the specific restrictions of our scenario, however, our case indeed turns out to correspond to a rather simple instance of model checking, since all models compatible with $i$’s information are equivalent with respect to $i$’s knowledge, so only one of them needs to be checked.

The straightforward implementation is described in Algorithm 2. To test whether a player $i$ knows a formula $\varphi \in \mathcal{L}^+$, he needs to execute $\text{eval}(i, \varphi)$. Recall that, for a word $w = i_1 \ldots i_t \in N^*$, we use $K_w$ to abbreviate $K_{i_1} \ldots K_{i_t}$ and $\text{Set}(w)$ to mean $\{i_1, \ldots, i_t\}$.

The evaluation is done in a recursive way, directly reflecting the semantics as defined in Chapter 2. It analyzes the formula at hand and evaluates its components, collecting the $K$ operators it sees on the way until an atom is reached, over which the chain of collected $K$ operators is then evaluated. This procedure is possible due to the fact that $K$ distributes over all connectives we use, as established in Chapter 2. Some comments on particular lines of the algorithm follow:

**Line 3** reflects that $i$ knows his own preferences, and is in accordance with the semantics of our model.
Algorithm 2: Knowledge evaluation function $\text{eval}(w, \varphi)$ of player $i$

Input: $w \in N^*$, $\varphi \in \mathcal{L}^+$
Output: true if $(V, M) \vDash K_w \varphi$; false otherwise

1

\begin{algorithmic}[1]
\State switch $\varphi$ do
\State \hspace{1em} case $p \in \text{At}$ if \(\text{Set}(w) \subseteq \{i\}\) and $p \in \text{At}$, then return true iff $p \in \succ_i$;
\State \hspace{1em} else if $\langle \cdot, A, p \rangle \in C(M_i)$ with some $A \supseteq \text{Set}(w)$ then return true;
\State \hspace{1em} else return false;
\State \hspace{1em} end
\State \hspace{1em} case $\varphi_1 \land \varphi_2$ return $\text{eval}(w, \varphi_1)$ and $\text{eval}(w, \varphi_2)$;
\State \hspace{1em} case $\varphi_1 \lor \varphi_2$ return $\text{eval}(w, \varphi_1)$ or $\text{eval}(w, \varphi_2)$;
\State \hspace{1em} case $K_j \varphi'$ with $j \in N$
\State \hspace{2em} if $\text{Set}(w) \cup \{j\} = \{i\}$ or there is $A \in \overline{H}$ with $\text{Set}(w) \cup \{j\} \subseteq A$ then
\State \hspace{3em} return $\text{eval}(w \circ i, \varphi')$;
\State \hspace{1em} else return false;
\State \hspace{1em} end
\State end
\end{algorithmic}

Lines 4 and 5 reflect Lemma 2.3.8, with $M$ equivalently replaced by $M_i$ in line 5 since $i \in \text{Set}(w) \subseteq A$; intuitively, the respective message has been sent to $A$ if and only if $i$ has observed it, since $i \in A$.

Line 11 is correct because of Lemma 2.3.5.

Line 12 corresponds to Theorem 2.3.10 and allows the evaluation to be cut off if the set of collected $K$ operators is not included in any $A \in \overline{H}$.

In particular, this last point has the effect that the exponential blowup caused by the recursive part of the $\text{dom}^t$ formulas depends on the hypergraph, rather than on the set of players. This makes evaluation especially efficient for hypergraphs with high locality.

While iterated elimination of strictly dominated strategies using the fully specified payoff matrix is a simple procedure [86], in a general setting of incomplete information, where arbitrary constellations of knowledge may occur, a player would have to maintain individual models for each other player, including each other player’s models of each other players, and so on. With the restrictions in our scenario, these nested models collapse to one common model for each hyperarc, which allows us to apply the simpler procedures we described. The complexity of our algorithms therefore depends mainly on the hypergraph. In that sense, our framework simplifies the computation of knowledge in a similar way as the graphical games [85] mentioned in Section 3.1 simplify the computation of equilibria.
Chapter 3. Strategies in interaction structures

Figure 3.1: Protocol of program run for the game from Example 3.3.3. Messages are abbreviated, e.g., $L >_UR$ actually represents two messages: one containing $L >_{UA} R$ and one $L >_{UB} R$. When displaying a player’s current picture of the game, we leave the matrix entries blank since the numerical payoffs are never communicated, only the relative preferences.
3.6 Conclusions

3.5.1. Example. Consider again the game from Example 3.3.3 with the hypergraph $H = \{ \{1, 2\}, \{2, 3\}, \{1, 3\} \}$. The game is depicted in Figure 3.1 together with a protocol of a program run in which players communicate their preferences and perform strategy elimination according to their respective current knowledge.

The outcome that the players arrive at after all communication allowed by $H$ has taken place is, as expected, the same as in Example 3.3.3, for the reasons discussed there. While player 2 may deduce that player 3 would be able to eliminate $B$ if player 3 knew that player 2 eliminated $L$, from the communicated information alone player 3 cannot deduce that.

Note that in this particular run, in line 8, player 1 computes player 2’s knowledge before player 2 actually computes it in line 10. In that sense, player 1 for a certain amount of time ascribes knowledge to player 2 which player 2, strictly speaking, does not have. We come back to this issue at the end of Section 3.6.1.

3.6 Conclusions

In this chapter, we looked at strategic games in the presence of interaction structures. We assumed that initially the players know only their own preferences, and that they can truthfully communicate information about their own preferences within their parts of the interaction structure. We defined operators to perform iterated elimination of strictly dominated strategies in any given state of communication, along with an epistemic model, based on the framework from Chapter 2, showing that the outcome of these operators is in a certain sense correct. We also discussed distributed implementations of the resulting procedures, connecting back to Chapter 1.

3.6.1 Related work

A few more remarks may be in place about how our approach of explicit knowledge programming relates to related notions from the literature. The basic idea of the closely related topics of explicit knowledge and algorithmic knowledge [111, 59, 121, 75] is to restrict the “classic”, logical notion of knowledge we have considered so far, in order to reflect what an agent could be said to be able to compute, rather than to logically know. One approach is to limit the accessibility relations in certain ways, reflecting the assumption that considering, or accessing, other possible worlds is computationally costly. However, these formalisms still take a modeler’s point of view in order to reason about what such an agent can do, rather than describing exactly how the agent does it. The ascribing is, in a sense, taken to a higher level: Instead of knowledge itself, methods for computing knowledge are ascribed to the agents, and the resulting central model is then used, again, by the modeler in order to reason about what the agents can be said to be able to.
Chapter 3. Strategies in interaction structures

compute. Naturally, the resulting logics may be more sophisticated and harder than the basic logic of knowledge.

Our approach, while also involving restrictions, differs in where these restrictions are placed. We put them on the situations we consider, on the initial knowledge and the ways in which additional knowledge can be created (e.g., communication), as well as on the class of epistemic statements we consider. This is done in such a way that the agents, within these limits, are able to compute the *unrestricted* logical notion of knowledge.

Another related approach is that of *knowledge-based programs* by Fagin et al. [61]. It resembles our approach in that epistemic statements are allowed to appear literally in the code of such programs. However, these knowledge-based programs are used exclusively for specification and verification of so-called *standard programs*, which do not contain epistemic statements. These resulting executable programs behave as required by their knowledge-based specification, but they are not assumed to actually “compute [the] knowledge in any way”. One may thus say that they use their knowledge *implicitly*, and in a sense our \( T \) operator implementation in Section 3.5.1 corresponds to such a standard program.

Our approach in Section 3.5.2 differs from the knowledge-based programs of Fagin et al. [61] in that we allow epistemic statements in the actual executable programs, thus giving them *explicit access* to their knowledge, through concrete algorithms with which they can compute what they may be said to know by a modeler. Reasons for this approach include the following ones. Firstly, we believe that the abstraction level that epistemic statements provide are useful not only for the specifier of a program, but also for the actual programmer, who may not even be known at the time of specification, especially with today’s extensible and open platforms. Secondly, programs containing epistemic statements can be easier to maintain than programs that behave equivalently but are formulated on a lower level, and the corresponding knowledge module (such as the one presented in Section 3.5.2) can be updated and verified separately. See Chapter 4 for an illustrating case study in the context of computer games.

It is important to note that, as we have seen in the context of Example 3.5.1, this different viewpoint also makes our notion of knowledge somewhat different from that of Fagin et al. [61]. Their notion of knowledge is defined in terms of possible runs of the whole distributed system. Being aware of the system as a whole and the exact programs of all its processes, the modeler who uses this notion to reason *about* processes would not ascribe knowledge to one process concerning knowledge of another process before ascribing that knowledge to that other process. We, on the other hand, take the subjective viewpoint of an intelligent agent and try to simulate epistemic reasoning *of* such an agent. In contrast to an external modeler, the agent may not be aware of the complete system and the exact internal workings of all other agents. An agent’s reasoning about others thus has to be based on assumptions. One reasonable assumption seems to be that other intelligent agents do check and use their knowledge at some point, at the latest
when it becomes relevant—when they need to act on it, which in our case is when they finally pick their strategy. In a reasonably homogeneous society of agents, if an agent can compute another agent’s knowledge, then that other agent can certainly also do it himself; in a society with greatly varying reasoning skills, the agents would need to model each other’s reasoning capabilities more explicitly. In either case, the distinction whether other agents really possess certain knowledge as soon as they would in principle be able to deduce it, or whether they only possess it at the time when it becomes manifest in their actions, is interesting from a philosophical point of view but impossible to determine for an agent and irrelevant for choosing his own actions.

So knowing that some other agent knows something, strictly speaking, for us rather means concluding that that other agent will be able to figure it out whenever it is relevant. Arguably, this also reflects the best that we as humans can do with an intangible concept such as others’ knowledge.

Note that this kind of temporary inconsistency across the agents’ knowledge states is of a different nature than that occurring with the eager protocol discussed in Chapter 1, Section 1.5. There, the inconsistency occurs on the level of ascribed knowledge, and it may happen that agents act on inconsistent knowledge, which is why we chose not to adopt that protocol. Here, in contrast, an agent correctly ascribes “potential” knowledge to another agent, which that other agent merely has not computed yet. Under the described assumption that the discrepancy between ascribed and computed knowledge only lasts between two external actions by the respective agent (i.e., the assumption that agents do compute their knowledge by the time it becomes relevant for acting), this inconsistency never manifests itself.

### 3.6.2 Possible extensions

While the restrictions we have imposed may be so severe that our scenario seems almost trivial, it has still served to explain our approach, and can be used as a starting point for further considerations. In a bottom-up way, our analysis could be extended in a number of ways:

- Allowing players to send information about the preferences of other players that they learned through interaction. We have started to extend the epistemic framework from Chapter 2 into that direction, see [9].

- Allowing other forms of messages, for example, messages containing information that a strategy has been eliminated, or epistemic statements.

- Considering strategic aspects of communication, even if truthfulness is required (should a certain piece of information be sent or not?)

- Considering formation or evolution of interaction structures, given strategic advantages of certain interaction structures over others.
Ultimately, a setting of strategic communication and proactive behavior of querying and telling (possibly false) information may be envisioned, involving agents that plan under consideration of epistemic states, actions and goals. But, as mentioned in Section 3.1.1, for such an open-ended framework much foundational work remains to be done.