An application of procedural knowledge space theory to the study of strategic planning in board games

Andrea Brancaccio[1,2], and Luca Stefanutti[3]

[1] School of Psychology,

University of East Anglia, Norwich, UK

[2] Institute for Applied Mathematics and Information Technologies "E. Magenes,"

National Research Council, Milan, Italy

[3] Department of Philosophy, Sociology, Education, and Applied Psychology,

University of Padua, Italy

Author Note

Corresponding author: Andrea Brancaccio, https://orcid.org/0000-0001-5919-6990
School of Psychology, University of East Anglia, Norwich, UK. e-mail:
A.Brancaccio@uea.ac.uk

Disclosures: An earlier version of this work was presented in Andrea Brancaccio's doctoral dissertation at the University of Padova.

Data Availability Statement: Data and analysis code are available at the Open Science Framework (Brancaccio & Stefanutti, 2026).

Abstract

Procedural Knowledge Space Theory (PKST) integrates Knowledge Space Theory and Problem Space Theory to model human problem-solving. This study extends PKST to competitive, two-player games and presents the first empirical application of homomorphisms to derive abstract representations of complex problem spaces. Using Tic-Tac-Toe and its numerical variant Pick-15 as test cases, the study demonstrates how PKST can model complex competitive scenarios effectively. An empirical study involving 148 participants validated these PKST extensions by analyzing their performance on Pick-15 tasks. Problem and knowledge spaces were derived at multiple levels of abstraction and tested using the Markov Solution Process Model. Results indicated that abstract representations provided a better fit for participants with prior experience in Tic-Tac-Toe, suggesting that expertise promotes the use of generalizable strategies. This work provides a rigorous proof of concept for applying homomorphisms to test theoretical predictions on problem-solving tasks. By linking abstract representations, like tactics, to problem-solving performance, the proposed methodology offers a concrete approach to evaluating cognitive theories of problem solving and decison-making. These findings highlight the potential of PKST to model broader cognitive phenomena in interactive contexts. The study emphasizes the practical utility and flexibility of PKST for formalizing and empirically testing hypotheses, extending its relevance to diverse domains of cognitive science.

Keynotes: Procedural Knowledge Space Theory, Problem-Solving, Competitive Games, Cognitive Modelling

Translational Abstract

This study builds on an existing way of modeling how people think through problems, called Procedural Knowledge Space Theory, and extends its application to describe how people make decisions when competing against someone else. To explore this, the researchers used two simple games: Tic-Tac-Toe and a numerical variant called Pick-15. They created different versions of the "problem spaces" that players navigate, ranging from detailed to more simplified representations. A total of 148 participants played Pick-15, and their performance was analyzed to determine which version of the problem space best matched their actual behavior.

The results showed that for players who already knew how to play Tic-Tac-Toe, the more abstract model provided a better explanation of their behavior. This suggests that experienced players rely on broader, more generalizable strategies that transfer across similar games. The study demonstrates that this approach can connect high-level ideas, such as tactics, to measurable behavior. In other words, it offers a practical way to test theories about how people solve problems and make decisions, even in competitive, interactive settings.

Overall, the findings indicate that the extended theory can help researchers understand a wider range of human thinking and may be useful for studying decision-making in domains beyond games.

1 Introduction

Board games offer powerful tools for examining human cognition, acting as microcosms of broader cognitive research. These games provide controlled environments to study key cognitive processes such as perception, memory, and problem-solving, with findings often extrapolated to other domains of human expertise (see e.g., Gobet, Retschitzki, & de Voogt, 2004). The structured nature of board games allows researchers to rigorously test and refine cognitive theories, providing a rich yet manageable context for exploration. From early studies on chess, exemplified by the seminal works of De Groot (1978) and Chase and Simon (1973), board game research has uncovered fundamental concepts like selective search, progressive deepening, and the critical role of pattern recognition in problem-solving.

In recent years, Procedural Knowledge Space Theory (PKST; Stefanutti, 2019) has emerged as a novel framework that bridges Knowledge Space Theory (KST; Doignon & Falmagne, 1999) and Problem Space Theory (PST; Newell & Simon, 1972). PKST combines KST ability to formally assess individual performances with PST aim to describe problem spaces in a well-defined manner. PKST combine both these tools to understand human problem-solving and cognition, while also providing a formal framework for assessing cognitive abilities (Brancaccio, de Chiusole, & Stefanutti, 2023; de Chiusole et al., 2024). This formal grounding also makes PKST particularly suitable for tasks commonly used in modern neuroscience that rely on an underlying problem space (e.g., Schuck & Niv, 2019), which are typically not embedded within a coherent theoretical framework or associated with a well-defined latent structure.

Building on these foundations, Brancaccio and Stefanutti (2024) proposed a mathematical framework for linking multiple representations of a problem space through a

pair of homomorphisms. These mappings enable the connection of alternative representations of the same problem space, allowing for empirical testing against data and comparison with each other. This innovative approach opens new avenues for studying problem-solving processes and the cognitive structures underlying them.

In this study, the theoretical concepts introduced by Brancaccio and Stefanutti (2024) are applied for the first time to human data. To show the utility of PKST and problem space homomorphisms, the classical board game Tic-Tac-Toe is used as a test case. Moreover, this study represents the first application of PKST to a setting with multiple problem solvers competing against each other to reach opposite goals.

Successful applications of the theory to single–problem-solver puzzles include the Tower of London test (Brancaccio et al., 2026; Stefanutti, de Chiusole, & Brancaccio, 2021), the Tower of Hanoi (Brancaccio & Stefanutti, 2024), the bucket-of-water puzzle (Stefanutti, 2019), and maze problems (Stefanutti & Brancaccio, 2025). Extending PKST from single–problem-solver puzzles to multi-players, competitive games required specific adaptations and generalizations of the framework. Tic-Tac-Toe offers an ideal starting point for testing, being simultaneously well-structured, sufficiently complex, and completely solved and well-studied. For these reasons, Tic-Tac-Toe and its variants are often used as model systems for testing a theory or a methodology (e.g., Newell & Simon, 1972; Van Opheusden et al., 2023; Watson, Arnold, & Tanenhaus, 2008) and as a model for cognitive development in young children (e.g., Crowley & Siegler, 1993; DeVries & Fernie, 1990).

The significance of the approach presented in this article extends beyond Tic-Tac-Toe. Board games, as controlled environments, provide a stepping stone for understanding complex, real-world problem-solving scenarios in an interactive context. In this article, problem-space homomorphisms are used to create alternative models based

on the set of rules postulated by Newell and Simon (1972) and expanded by Crowley and Siegler (1993) within a production-system framework. The positive results obtained from applying PKST and problem-space homomorphisms demonstrate the potential of this approach and support further, more complex applications, offering a rigorous methodology for exploring and testing theoretical frameworks in human problem solving.

The manuscript is organized as follows. Foundational concepts related to PKST, which are necessary for understanding the manuscript, are presented in Section 2. Section 3 introduces the problem space for two players competing against each other and discusses the application of the Minimax algorithm to PKST. Section 4 introduces the tic-tac-toe problem space and the derived goal and knowledge spaces used in the subsequent analyses. Section 5 presents a simulation study comparing these alternative representations under controlled conditions. Section 6 presents an empirical application in which the theoretical innovations are applied to data collected from a "Tic-Tac-Toe-like" game. Finally, Section 7 provides a general discussion to conclude the paper.

2 Background

This section introduces the core concepts of Procedural Knowledge Space Theory. All concepts introduced in this section are formally defined in the following subsections. PKST provides a framework for describing how individuals solve problems by linking the structure of a task to the procedures available to the problem solver (i.e., the sequences of actions they can apply to the task). It conceptualizes problem solving as the process of moving through a set of possible situations by applying sequences of actions. Within this perspective, a "problem space" represents the set of all possible

situations in a task together with the actions that connect them, while a solution corresponds to a sequence of actions leading from an initial situation to a goal.

A central distinction in PKST concerns the difference between the structure of the task and the abilities of the individual. Although the problem space specifies all possible ways a problem can be solved, individuals typically have access to only a subset of these possibilities, or also none of them. This subset, referred to as a "competence state", can be interpreted as the set of procedures that an individual is able to execute. Observable performance depends on these procedures: individuals may reach the same solution through different sequences of actions or may fail due to missing intermediate steps. To capture this, PKST introduces the notion of a "knowledge state", defined as the set of problems that an individual can solve, thereby linking latent procedural resources to observable outcomes.

A further key aspect of the framework is the role of abstraction in human problem solving. Rather than operating on fully detailed representations, individuals often rely on simplified representations that reduce complexity by ignoring irrelevant distinctions. In PKST, such simplifications are formalized through homomorphisms, which map a detailed problem space onto a more abstract one while preserving its essential structure. These mappings can be interpreted psychologically as the use of strategies or higher-level representations that group together equivalent situations. Taken together, these concepts provide a framework for generating testable hypotheses about behavior, including predictions about individual differences and the use of abstract strategies.

2.1 Procedural Knowledge Space Theory

The concept of a problem space, first introduced by Newell and Simon (1972), has been extensively utilized in understanding human problem-solving. Recently a mathematical connection to Knowledge Space Theory (Doignon & Falmagne, 1999), has been established in Stefanutti (2019). The resulting theoretical framework is called Procedural Knowledge Space Theory. This section aims to present the main ideas of PKST.

A problem space is a description that defines the various discrete states of a problem and the deterministic operations that can be performed to navigate between these states. In this context, we define a set of operations, denoted as Ω, which are the actions one can take to solve problems. A sequence of these operations can be thought of as a string, where each operation corresponds to an action taken in the problem-solving process. The set of all finite-length strings of operations is denoted by Ω^*, and ϵ represents the empty sequence in Ω^*. In the context of the problem space, the empty string ϵ corresponds to the absence of any action. In addition, any two strings of operations π, σ belonging to Ω^* can be concatenated to form another operation sequence $\pi\sigma$, which also belongs to Ω^*, reflecting the closure of Ω^* under concatenation. Thus, Ω^* is a monoid under the operation of concatenation of strings.

There exist two equivalent definitions of a problem space. The former, named "functional definition" was provided in Stefanutti (2019), whereas the latter, named "relational definition" was suggested by Brancaccio and Stefanutti (2024). The relational definition is used in this article because it provides the basis for the notion of problem space homomorphism.

The mathematical definition of a problem space is as follows. Let $\mathbf{P} = (S, \Omega, R)$

be a triple, where S is a finite set of problem states, Ω is a finite set of operations, and R is a set of triples (a, π, b) where a and b belong to S whereas π is any sequence of operations in Ω. The elements in R are interpreted as sequences of operations connecting two problem states together. Consider moreover the following five Axioms:

(PS1) *identity*: the triple (a, ϵ, a) belongs to the relation R for all problem states a in S. In other words, doing nothing does not change the inital state.

(PS2) *uniqueness*: for all problem states a and all possible sequences of operations π, if there are problem states b, c such that both $(a, \pi, b), (a, \pi, c)$ belong to R then b and c must be the same state. In other words, the final state associated with a given initial state and a given sequence of operations must be unique.

(PS3) *completeness*: for all problem states a and all possible sequences of operations π, there exists a problem state b such that (a, π, b) belongs to R.

(PS4) *transitivity*: for any three problem states a, b, c and any two sequences of operations π and σ, if both $(a, \pi, b), (b, \sigma, c)$ are in R then also $(a, \pi\sigma, c)$ must be in R. In other words, if the final state of a given path equals the initial state of another path, then the two can be joint together through that state giving rise to a longer path.

(PS5) *decomposition*: for any two problem states a, b and any two sequences of operations π and σ, if $(a, \pi\sigma, b)$ is in R then there exists a c belonging to S such that it is possible to split it into two separate subpaths (a, π, c) and (c, σ, b), both belonging to R.

Then $\mathbf{P}$ is a *problem space* if it satisfies Axioms PS1, PS2, PS3 and PS4 (Brancaccio & Stefanutti, 2024).

The triple **P** is called an *incomplete problem space* whenever Axiom PS3 is replaced by Axiom PS5. Indeed, it should be observed that in a problem space, the condition in PS5 follows from Axioms PS1 to PS4, therefore it needs not be included in the set of axioms. Thus, in a problem space all conditions from PS1 to PS5 hold true, whereas in an incomplete problem space Axiom PS3 does not hold. This axiom has to do with the distinction between *effective* and *ineffective* sequences of operations. It states that, given any "initial" problem state s, any arbitrary sequence of operations can be applied to that state, yielding as a result some problem state in S. Eventually, the result is s itself if the sequence of operations is ineffective at that state. In an incomplete problem space, where PS3 does not hold, it might be possible that, for some problem state s the application of some sequence π of operations is undefined. This precisely happens whenever the sequence is ineffective at that state.

Indeed, the difference between a problem space and an incomplete problem space is more technical than substantial, as it only pertains to how ineffective sequences of operations are represented. As a matter of fact, any problem space can be transformed, without loss of information, into an incomplete problem space, by just removing all of the ineffective triples (x, π, x), with $\pi \neq \epsilon$, from the set R. On the other side, any incomplete problem space can always be transformed into a complete one, by adding the pair (x, π, x) for each sequence π that is ineffective at problem state x. Because there is no loss of information in the transformation, and because incomplete problem spaces are more easily handled, the types of problem spaces considered in this article are all incomplete.

The reduction R° of a problem space (S, Ω, R) is a simplified version of the full set of relationships between states. It focuses only on the most direct connections, removing any indirect steps between states that can be reached through a sequence of

operations. In other words, R° contains only those transitions (a, π, b) where π is a single operation.

Additionally, a problem space can be graphically represented by a directed graph (digraph). The digraph of a problem space (S, Ω, R) provides a way to visualize the relationships between problem states and operations. In this graph, the collection of problem states S represents node of the graph, and the reduction R° represents the edges or connections between these states.

Within PKST, a problem is a pair (s, t), where s is the initial state of the problem, t is the goal state and there exists a sequence of operations that leads from s to t. The set Q denotes all possible problems within the problem space. A solution path for the problem (s, t) is any sequence of operations that successfully moves from the initial state s to the goal state t. Notably, multiple solution paths may exist for the same problem, as different sequences can lead to the same outcome. For instance, consider the problem of traveling from home to work. The initial state is home, and the goal state is the workplace. In the problem space, which in this case is the city where both home and workplace are located, several alternative routes (e.g., taking the highway or traveling through the city center) serve as solution paths for reaching work from home.

In the context of a problem space, some solution paths can be related to others in a way that reflects their structure. We say that one solution path (s, π, t) is a subpath of another solution path (a, σ, b) when we can express the longer path (a, σ, b) by joining the shorter path (s, π, t) with other paths. Formally, (s, π, t) is a subpath of (a, σ, b) if there are two operations α and β such that (a, α, s), (t, β, b), and $\sigma = \alpha\pi\beta$.

To illustrate this with the previous example, the route from home to the workplace can be divided into two segments: i) going from home to the bus station, and ii) traveling from the bus station to the workplace. Both segments i) and ii) are sub-

paths of the overall route connecting home to the workplace. In fact, joining these segments together reconstructs the entire path. In PKST, the problem solving ability of an individual is represented by the *competence state*, which is a set C of solution paths. Psychologically, a competence state can be interpreted as the latent set of procedures that an individual is able to execute within a given problem space. This set is such that, if (c, σ, d) is a subpath of (a, π, b), and this last belongs to C then also (c, σ, d) belongs to C. This property follows from the assumption that if a person can solve a problem through a given solution path, then they will be able to solve all the subproblems that can be encountered along that path. This assumption is consistent with classical theories of human problem solving, in which complex procedures are described as sequences of operators transforming one state into another, and successful execution of a procedure presupposes the availability of the intermediate operators composing it (Newell & Simon, 1972). Similarly, in cognitive architectures such as ACT-R (Anderson, 2014), procedural knowledge is represented through production rules, where solving a task requires the sequential application of elementary rules whose composition gives rise to more complex procedures and eventually skills (Anderson, 1982). The collection of all possible competence states forms the *competence space*. In psychological terms, the competence space represents the set of all procedurally possible individual profiles under the assumptions of the model, that is, all possible combinations of executable procedures consistent with the problem space. Another fundamental notion in PKST is that of a *knowledge state* K, which is the set of problems that can be solved using the solution paths available in that competence state. Thus, while competence states describe latent procedural resources, knowledge states describe their observable consequences at the behavioral level, namely which problems an individual can successfully solve. Two individuals may therefore share

the same knowledge state while differing in competence state if they solve the same problems through different procedures. Conversely, differences in competence states may generate differences in knowledge states when some procedures allow access to additional problems. This relationship is foundational for understanding how knowledge can be structured in the context of problem-solving. Indeed, this distinction makes it possible to separate observable performance from the underlying procedural organization that generates it, providing a formal basis for psychological hypotheses about individual differences in problem solving strategies, learning trajectories, or expertise development.

The collection of all possible knowledge states is called a *knowledge space* $\mathcal{K} \subseteq 2^Q$.

A particular type of problem space that provides a structured framework for evaluating progress and identifying errors is referred to as a "goal space" (Stefanutti et al., 2021). In a goal space, each step in the problem-solving process is classified as either "correct-so-far" or "incorrect." This classification helps determine whether a given solution path leads to a goal or to a mistake. A goal space is defined as a specific type of problem space, defined by the 5-tuple $(S_g, \Omega_g, f, g, R_g)$. It incorporates two distinct states: the failure state f, which represents an incorrect response, and the goal state g, which represents the correct solution to the problem. A goal space follows two essential properties. First, once either the failure state or the goal state is reached, any further actions become ineffective, meaning that the problem-solving process terminates at these states. Second, the goal state g must be accessible from any state in S_g except the failure state. In other words, for any starting state that is not the failure state, there must exist a possible sequence of moves that will eventually lead to the goal. Several procedures have been proposed to transform a problem space into a corresponding goal space. A straightforward approach consists of mapping all

paths that do not terminate in a goal state to a designated failure state (Stefanutti & Brancaccio, 2025).

A *problem solution process* in a goal space is a random variable whose realizations are sequences of problem states $(s_0, s_1, \ldots, s_m)$ that are taken by problem solver. It is assumed that all states encountered during this process are observable. Thus, in evaluating a problem solution, we consider the entire sequence of states as correct if the final state is the goal state $s_m = g$. On the other hand, whenever the final state is the failure state $s_m = f$ the solution process is considered incorrect.

To illustrate the theoretical explanation with a concrete case, consider the ad hoc example of a river-crossing puzzle, called Cops and Prisoner (e.g., Ishay & Lee, 2025). The setup is as follows: n cops and n prisoners are on the left side of a river and must all cross to the right side. A single boat is available, with capacity for at most two people at a time. The boat cannot cross empty and must always carry at least one passenger. The main constraint is that, on either bank, if the number of prisoners exceeds the number of cops, the prisoners escape and the attempt fails. The goal is to transfer all $2n$ individuals safely to the opposite bank.

The problem can be represented as a problem space in which each problem state is a triple (c, p, r). Here, $c \in \{0, 1, \ldots, n\}$ and $p \in \{0, 1, \ldots, n\}$ denote the number of cops and prisoners on the left bank, respectively, while $r \in \{L, R\}$ indicates the current position of the boat. For illustration, the case $n = 2$ (two cops and two prisoners) is discussed below.

The problem space for the Cops and Prisoners puzzle is defined as $\mathbf{P}_{cp} = (S_{cp}, \Omega_{cp}, R_{cp})$. The set of problem states S_{cp} consists of the 12 valid combinations of (c, p, r), where c and p range from 0 to 2, and $r \in \{L, R\}$ indicates the position of the boat. Although the full cross-product of values yields $3 \times 3 \times 2 = 18$ possible states, only 12 are ad-

missible once the puzzle's constraints are enforced. In particular, most states in which the number of prisoners exceeds the number of cops on either bank are excluded. For simplicity of notation, each state will be denoted as *cpr* without any parenthesis or commas. For example, the initial state of the puzzle, $(2, 2, L)$, where all four individuals are on the left bank, is written as $22L$, while the final state, $(0, 0, R)$, where all four individuals are on the right bank, is written as $00R$. The operation correspond to all legal boat moves: moving one cop ($\rightarrow_C$) or two cops ($\rightarrow_{CC}$) from left to right, or in the opposite direction ($\leftarrow_C$, $\leftarrow_{CC}$); moving one prisoner ($\rightarrow_P$) or two prisoners ($\rightarrow_{PP}$) from left to right, or in the opposite direction ($\leftarrow_P$, $\leftarrow_{PP}$); and moving one cop and one prisoner together ($\rightarrow_{CP}$) from left to right, or in the opposite direction ($\leftarrow_{CP}$). Thus, the set of operation is $\Omega_{cp} = \{\rightarrow_C, \leftarrow_C, \rightarrow_{CC}, \leftarrow_{CC}, \rightarrow_P, \leftarrow_P, \rightarrow_{PP}, \leftarrow_{PP}, \rightarrow_{CP}, \leftarrow_{CP}\}$.

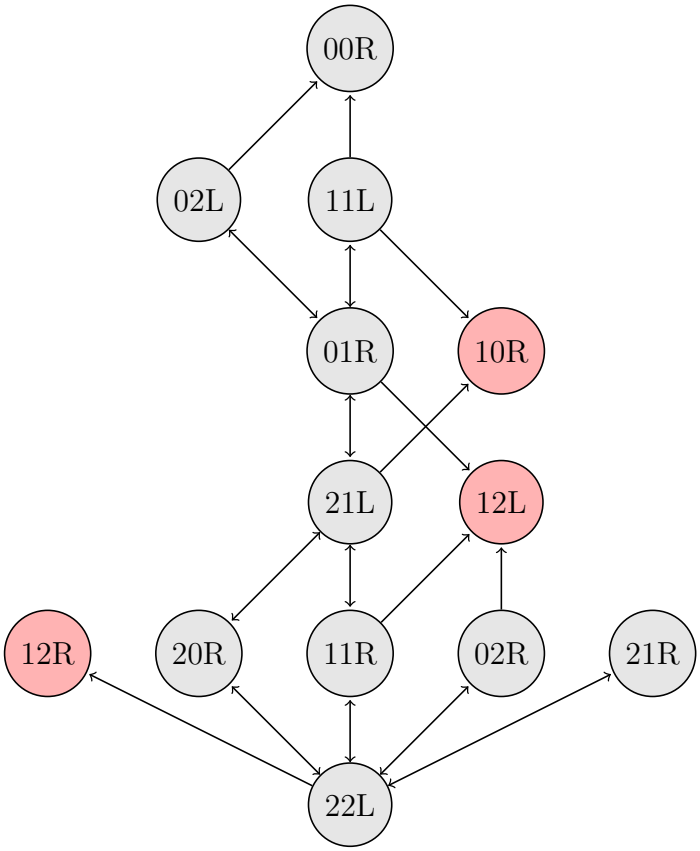


Figure 1: Digraph of the Cops and Prisoners problem space. Cyclic connections are represented using bidirectional arrows, while self-connections are omitted for clarity. Configurations in which the puzzle is lost are depicted in red.

Figure 1 displays the digraph of the problem space, with the operations omitted for readability. The ternary relation R_{cp} can be directly derived from the digraph. It should be noted, however, that the current representation includes cyclic paths, such as moving two cops from left to right and then immediately back to the left (e.g., $(22L, \rightarrow_{CC}\leftarrow_{CC}, 22L)$), but excludes ineffective moves, such as attempting to move the boat from left to right when it is already on the right side (e.g., $(21R, \rightarrow_{C}, 21R)$).

The three states highlighted in red in Figure 1 correspond to configurations in which the number of prisoners on one bank exceeds the number of cops, and the puzzle is therefore failed.

To define the goal space, we specified $00R$ as the goal state, identified all states in which the number of prisoners exceeded the number of cops as failure states, and excluded cyclic paths. While distinguishing between goal and failure states is necessary, the exclusion of cyclic paths is arbitrary but reasonable in this application. For example, it would be implausible to regard as a valid solution a sequence in which a prisoner repeatedly moves from left to right and then back again (e.g., $(22L, \rightarrow_P \leftarrow_P, 22L)$) before eventually completing the puzzle.

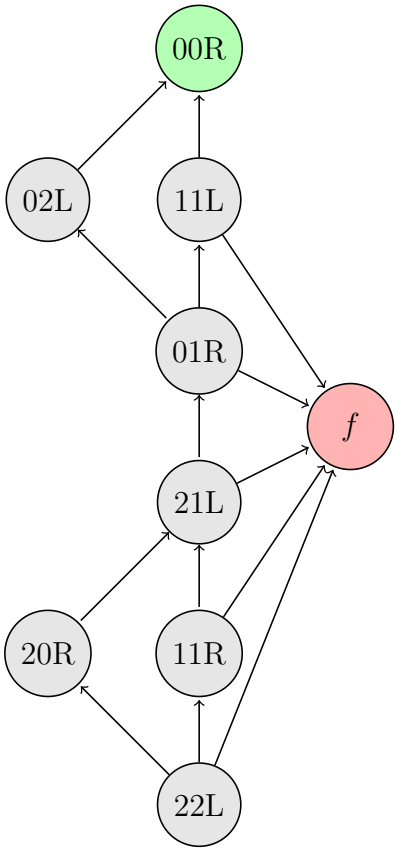


Figure 2: Digraph of the goal space derived for the Cops and Prisoners example.

The resulting goal space, shown in Figure 2, contains nine problem states, including both the goal (in green in the figure) and the failure states (in red in the figure). From this goal space, seven distinct problems can be derived, namely

$$Q_{cp} = \{(22L, 00R), (20R, 00R), (11R, 00R), (21L, 00R), (01R, 00R), (11L, 00R), (02L, 00R)\}.$$

Consider two subsets of solution paths in this goal space: $C_1 = \{(01R, \leftarrow_C \rightarrow_{CP}, 00R), (01R, \leftarrow_C, 11L), (11L, \rightarrow_{CP}, 00R), (02L, \rightarrow_{PP}, 00R)\}$ and $C_2 = \{(01R, \leftarrow_C \rightarrow_{CP}, 00R), (02L, \rightarrow_{PP}, 00R)\}$. The subset C_1 is a competence state, since the paths $(01R, \leftarrow_C, 11L)$, and $(11L, \rightarrow_{CP}, 00R)$ are all subpaths of $(01R, \leftarrow_C \rightarrow_{CP}, 00R)$, and all three be-

long to C_1. Conversely, C_2 does not constitute a competence state, since it does not include all subpaths of $(01R, \leftarrow_C \rightarrow_{CP}, 00R)$; in particular, $(11L, \rightarrow_{CP}, 00R) \notin C_2$.

The subset of problems in Q_{cp} that are solved through solution paths contained in C_1 are $\{(01R, 00R), (11L, 00R), (02L, 00R)\}$. Specifically, $(01R, 00R)$ is solved via the path $(01R, \leftarrow_C \rightarrow_{CP}, 00R)$, $(11L, 00R)$ via $(11L, \rightarrow_{CP}, 00R)$, and $(02L, 00R)$ via $(02L, \rightarrow_{PP}, 00R)$. It should be noted that although $(01R, \leftarrow_C, 11L)$ belongs to C_1, it does not solve any problem in Q_{cp}, since all problems in a goal space must terminate at the goal state. Thus, although the path belongs to the competence state, it is not represented as a problem in the knowledge state derived from it.

Applying the same rationale to all possible competence states yields the complete knowledge space for this example:

$\{\{22L, 20R, 11R, 21L, 01R, 11L, 02L\}, \{22L, 11R, 21L, 01R, 11L, 02L\},$
$\{22L, 20R, 21L, 01R, 11L, 02L\}, \{22L, 20R, 11R, 21L, 01R, 02L\}, \{22L, 20R, 11R, 21L, 01R, 11L\},$
$\{20R, 11R, 21L, 01R, 11L, 02L\}, \{22L, 11R, 21L, 01R, 02L\}, \{22L, 11R, 21L, 01R, 11L\},$
$\{22L, 20R, 21L, 01R, 11L\}, \{22L, 20R, 21L, 01R, 02L\}, \{11R, 21L, 01R, 11L, 02L\},$
$\{20R, 21L, 01R, 11L, 02L\}, \{20R, 11R, 21L, 01R, 02L\}, \{20R, 11R, 21L, 01R, 11L\},$
$\{11R, 21L, 01R, 02L\}, \{11R, 21L, 01R, 11L\}, \{20R, 21L, 01R, 11L\}, \{20R, 21L, 01R, 02L\},$
$\{21L, 01R, 11L, 02L\}, \{21L, 01R, 11L\}, \{21L, 01R, 02L\},$
$\{01R, 11L, 02L\}, \{01R, 11L\}, \{01R, 02L\}, \{11L, 02L\}, \{11L\},$
$\{02L\}, \emptyset\}$

For clarity of exposition, the problems in the knowledge state are represented using only their initial state, since the goal state is constant ($g = 00R$). For instance, state

$\{01R, 02L\}$ in the above knowledge space corresponds to state $\{(01R, 00R), (02L, 00R)\}$ when using the extensive notation.

2.2 Problem Spaces homomorphism

This section presents a summary of the theoretical results from Brancaccio and Stefanutti (2024). Problem spaces are formal structures that represent the possible solutions to a given problem or a set of problems. When dealing with complex or large problem spaces, the number of problem states, problems, and solution paths increases rapidly. As result, it is unlikely that humans mentally replicate the entire, detailed problem space as-is. For instance, in the Cops and Prisoners example presented at the end of Section 2.1, it is reasonable to assume that, when humans confront this puzzle, they may not distinguish between operations that move from left to right and those that move from right to left. In such cases, problem solvers are likely to rely on simplified mental representations that omit details unnecessary for reaching a solution. This simplification typically exploits recurring structural patterns that reduce the complexity of the problem space, such as symmetries. Consequently, problem states and operations can be abstracted into a more general strategy that leverages these inherent symmetries. For example, in the Cops and Prisoners puzzle, the operations $\rightarrow_C$ ("move one cop from left to right") and $\leftarrow_C$ ("move one cop from right to left") may be viewed as symmetric and generalized into $\leftrightarrow_C$, representing the action "move one cop from one bank to the other."

A more abstract problem space is often more plausible for human problem solvers with limited computational resources (Simon et al., 1990) (e.g., working memory, attention). However, these simplifications adhere to an internal logic that preserves the

essential relationships within the problem space, ensuring that the abstract version retains the fundamental properties of the original problem space. This alignment between the original and simplified spaces is captured by Brancaccio and Stefanutti (2024) in the concept of a "homomorphism." The paper also provides a few examples of symmetries in different problem solving contexts. A homomorphism maps a concrete, detailed problem space to a more abstract representation while preserving the relationships between states and operations. Formally, a homomorphism between two problem spaces can be defined by a pair of mappings, denoted as (ϕ, γ):

- ϕ maps each state in the concrete space to a corresponding state in the abstract space.
- γ maps each sequence of operations in the concrete space to a sequence of operations in the abstract space.

Two types of homomorphisms exist, the weak and the strong ones. The formal derivation and comparison are in Brancaccio and Stefanutti (2024), in this article only weak homomorphisms are considered. The pair (ϕ, γ) is a weak homomorphism if every path (s, π, t) in the concrete space is represented by a path $(\phi(s), \gamma(\pi), \phi(t))$ in the abstract space. Moreover, the mapping γ satisfies the condition $\gamma(\pi)\gamma(\sigma) = \gamma(\pi\sigma)$, where π and σ are sequences of operations in the concrete space.

Both mappings, ϕ and γ, are surjective but not injective in general. This means that for each problem state or sequence of operations in the abstract problem space, there exist one or more corresponding problem states or sequences of operations in the concrete space. All the elements in the concrete problem space that map to the same element in the abstract problem space form an equivalence class within the concrete problem space. A special case occurs when both ϕ and γ are bijective; in this case, the

mapping defines an isomorphism between the two problem spaces.

2.3 Markov Solution Process Model

The Markov solution process model (MSPM), developed by Stefanutti et al. (2021), views the problem solution process as a random sequence $\mathbf{S}_0, \mathbf{S}_1, \ldots \mathbf{S}_n \ldots$, where each random variable $\mathbf{S}_n$ represents a state s_n in the problem space. The MSPM assumes that the solution process is dependent on the problem solver's latent knowledge state K, and that, given this knowledge state and the presented problem (s_0, g), the process is Markovian (for a an overview see, Ethier & Kurtz, 2009). This means that in the sequence of states $s_0, s_1, \ldots s_n \ldots$, the probability of a transition to state s_n only depends on s_{n-1} and it is independent of the previous states in the sequence (memoryless property). This is expressed by the Markov property (the shortcut notation $P(s_i)$ is used in place of $P(\mathbf{S}_i = s_i)$ for convenience):

$$P(s_n|s_0, s_1, \ldots, s_{n-1}, (s_0, g), K) = P(s_n|s_{n-1}, (s_0, g), K).$$

The transition probability $P(s_n|s_{n-1}, (s_0, g), K)$ denotes the likelihood of transitioning from state s_{n-1} to s_n under knowledge state K. The probability of an entire solution process $(s_0, s_1, \ldots, s_m)$, given the knowledge state K, can then be factored as:

$$P(s_0, s_1, \ldots, s_m|(s_0, g), K) = \prod_{i=0}^{m-1} P(s_{i+1}|s_i, (s_0, g), K).$$

Considering the probability distribution over knowledge states $P_{\mathcal{K}}(K)$, the overall probability of observing a specific solution process for problem (s_0, g) is:

$$P(s_0, s_1, \ldots, s_m) = \sum_{K \in \mathcal{K}} \prod_{i=0}^{m-1} P(s_{i+1}|s_i, (s_0, g), K) P_{\mathcal{K}}(K).$$

This equation defines the general framework of the MSPM. However, the number of parameters for transition probabilities may be large.

It should be noted that a transition from state s_i to state s_{i+1} can only occur with positive probability if there is a single operation ω that transforms s_i into s_{i+1}. Therefore, $P(s_{i+1}|s_i, (s_0, g), K) > 0$ only if the transition (s_i, ω, s_{i+1}) is an edge of the graph of the goal space, implying that many transition probabilities will be zero.

Further assumptions can simplify the model. One such assumption, tested empirically by Stefanutti et al. (2021), is the MSP1 assumption. This assumption states that the transition probability from state s to state t, given the knowledge state K and problem (s_0, g), follows a simple rule:

$$P(t|s, (s_0, g), K) = \begin{cases} \beta_{st} & \text{if } (s_0, g) \in K, \\ \eta_{st} & \text{if } (s_0, g) \notin K. \end{cases}$$

Here, β_{st} and η_{st} are real parameters specific to each directed edge (s, t). The transition probability depends on both the problem solver's knowledge state and the initial state s_0.

Table 1 presents a classification of possible interpretations for all transition parameters based on the relationship between the observed transition and individual mastery of the problem (s_0, g). Columns of the table are about mastery or non-mastery of the problem (s_0, g). Rows of the table are about the transition observed in a single step of the solution, which could be either correct (s, t) with $t \neq f$ or incorrect (s, f). As the table shows, a correct transition performed by an individual who masters (s_0, g) is a *careful correct* β_{st}. On the other side, a correct transition performed by an individual who does not master the problem is a *lucky guess* η_{st}, implying that the correct transition was given by chance. Instead, an incorrect transition (s, f) carried out by an individual who masters the problem is a *careless error*. Finally,an incorrect transition carried out by an individual who does not master the problem is a *non-lucky guess*

Table 1: Classification of correct and incorrect observed transitions based on mastery or non-mastery of a given problem.

Observed Transition	Mastered (s_0, g)	Not mastered (s_0, g)
Correct (s, t)	careful correct	lucky guess
Incorrect (s, f)	careless error	non-lucky guess

A monotonicity condition is applied to these parameters: if $t \neq f$ (where f represents the failure state), then η_{st} (the probability of a lucky-guess) is expected to be smaller than β_{st} (the probability of a careful correct). This is equivalent to saying that the probability of making a careless error is less than the probability of a non-lucky guess.

In summary, the model consists of the the following three types of parameters: the knowledge state probabilities $P_{\mathcal{K}}(K)$, the transition probabilities β_{st} and η_{st} for each edge (s, ω, t) of the graph of the goal space.

The parameters of the MSPM model can be estimated using maximum likelihood through the expectation-maximization (EM) algorithm. The authors have developed a MATLAB implementation of this procedure.

3 The Two Players Problem Space

In this section, the framework elaborated by Stefanutti (2019) is generalized to situations in which two players act in the same problem space in turns. A 2-*player space* problem space is defined to be a problem space (S, Ω, R) where the set S of states can be partitioned into two classes S_1 and S_2, where S_1 contains the states of the first

player, whereas S_2 contains the states of the second player. Moreover, in every single-operation path (s, ω, t), the state s belongs to S_1 if and only if the state t belongs to S_2. In the sequel, a state belonging to S_1 (respectively, S_2) is referred to as a state belonging to player 1 (respectively, player 2) if it represents a configuration of the game in which only player 1 (respectively, player 2) can act.

The directed graph of this multiplayer space turns out to be a type of *bipartite directed graph* (S_1, S_2, R°) where S_1 and S_2 are two sets of vertices and the reduction R° are the edges.

Example 1. NIM: The players, in each step, must remove one or two coins from a heap of four coins. The winner is the one who picks the last coins.

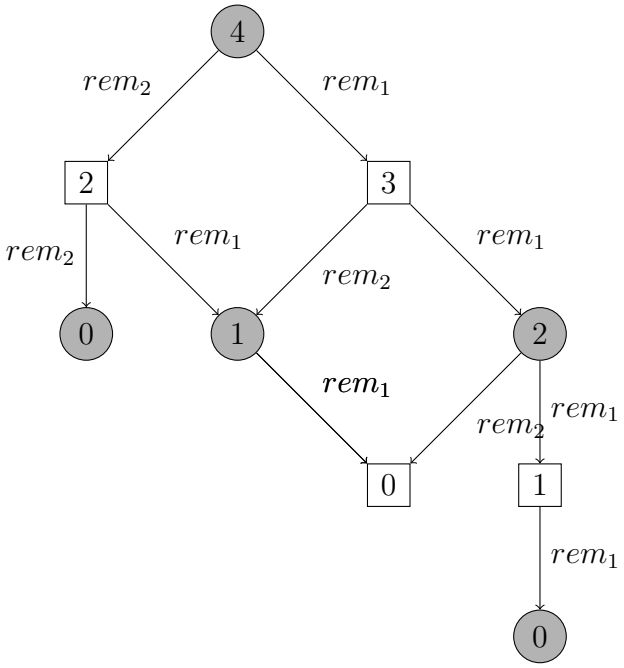


Figure 3: Bipartite digraph of the multiplayer space of the NIM example. Gray circles in the figure are the problem states of the first player, whereas the boxes are the ones of the second player. The number inside each box or circle is the number of coins in a heap in that problem state. The arcs labeled as rem_1 is the operation "remove one coin," and the arcs labeled as rem_2 is the operation "remove two coins."

Figure 3 displays the 2-player space for the NIM example. The grey circle nodes represent the problem states of the first player, and the white boxes represent the problem states of the second player. In this example, any problem state could be represented as a pair (n, k), where n is the number of coins in the heap and k is the player who plays next (represented in the graph as "grey" or "white").

In a 2-player space, it is useful to define a single-operation path as *turn*. In particular, a single operation (s, ω, t) is referred to as a turn for player 1 if the problem state s belongs to player one, otherwise it is referred to as a turn for player 2. The

concatenation of two turns is defined as a *move*. From a game perspective, a turn changes the player that can act in the game, whereas a move of the game is a pair of turns such that initial and final states belong to the same player.

3.1 The Minimax Algorithm

To define a goal space in a game, further considerations on the definition of "correct-so-far" are required. The possible outcomes in a game with two players are: the first player wins, the second player wins, or the game terminates in a draw. While at the beginning of the game, each player will try to win, as a result of the game continuation the "best" outcome for one or both players could be a draw or in extreme situations to withdraw from the game. For instance, in any game where there is a finite number of points to be collected by the two players, such as mancala, scrabble, and bao, there are game situations in which a player has collected more than half of the points early in the game. In such a situation, the only possible outcome would be a loss for the other player.

In the Nim game displayed in Example 1, initially the best possible outcomes are $(0, white)$ for player *grey* and $(0, grey)$ for *white*. Later in the game, assuming there is one coin in the heap and *grey* moves (i.e., state $(1, grey)$), the "best" (and only) outcome for both players is $(0, white)$, since at this point of the game this is the only possible outcome.

In a two-player game, the minimax algorithm is commonly used to find the optimal move in a given position. The minimax algorithm helps find the optimal move, by working backwards from the end of the game. At each step it assumes that the first player is trying to maximize the chances of winning, while on the next turn the second

player is trying to minimize the first player's chances of winning.

In applications where the problem space is known, like in the Nim Example, all possible moves in the game are explored until it reaches a terminal state (the first player wins, the second player wins, or the game ends in a draw). At this point, an evaluation function v from the set S to the set $\mathbb{R}$ of real numbers, assigns a value to the terminal state reflecting the outcome. Without loss of generality, let's say that the evaluation function gave a higher value in the case of a victory of the first player and a lower value in the case of a victory of the second player. The algorithm backtracks from the terminal state propagating the evaluation through the problem space. Specifically, the value of any problem state s_i different from a terminal state is determined based on the evaluation of the states in its neighborhood $E(s_i)$. If a state s_i belongs to the first player (i.e., the first player makes the next move), the associated value is the maximum value among those in its neighborhood, denoted as $\max v(E(s_i))$. Conversely, if s_i belongs to the second player, the associated value is the minimum value among those of all the states in its neighborhood, denoted as $\min v(E(s_i))$. Therefore, in the following, the player who selects the maximum value is referred to as the *maximizing* player, while the player who aims to minimize the maximum value for the opponent is referred to as the *minimizing* player. As a result of the minimax algorithm, we obtained a new problem space that is a projection of the original one, containing only the optimal moves. At this stage, to create a goal space, we connect all the terminal states to an abstract goal state, representing the end of the game. Similarly, all non-optimal moves are replaced with moves leading to the failure state. For example, the move from $(2, grey)$ to $(1, white)$ through operation α in Figure 3 is replaced by a move from $(2, grey)$ to f through operation α in the new goal space.

It's important to note that a problem space obtained using the minimax algorithm

by backpropagation, as described in this section, is always a goal space by construction. Indeed, (GS1) holds trivially, and (GS2) holds because all problem states, other than f and g, are connected to at least one terminal state by backpropagation, and all the terminal states are connected to the goal state. Therefore, the goal state is connected to all problem states other than the failure state.

Example 2. Resuming the Nim in Example 1 there are two possible terminal states in the game space: $(0, \text{white})$ and $(0, \text{grey})$, corresponding to wins for player white and player grey, respectively. As the first state of the game belongs to player grey, let's designate grey as the maximizing player and white as the minimizing player. Therefore, the evaluation function for the terminal states is such that $v(0, white) = 1$ and $v(0, grey) = -1$. Figure 4 illustrates the goal space derived from the application of the minimax algorithm. For each node, the value on the right represents the value $v(s)$ associated with that node.

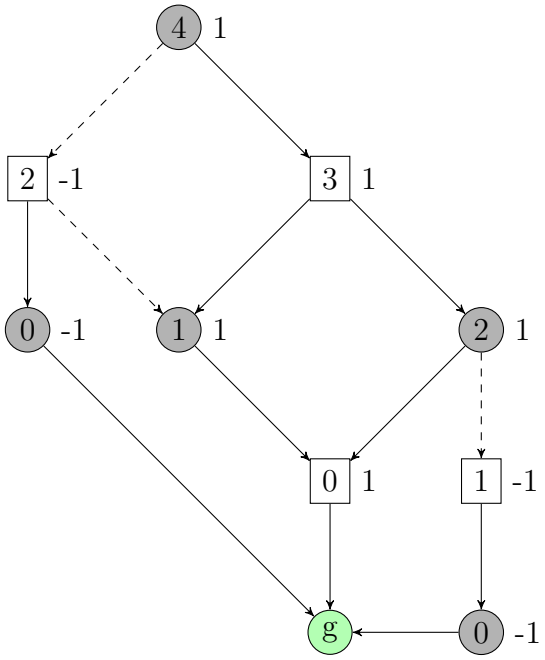


Figure 4: Goal space of the Nim game with a heap of four coins. The abstract goal states obtained as a results of the minimax procedure is depicted in green.

In the figure solid lines represent the optimal moves selected by the minimax algorithm, whereas the dashed lines represent the non-optimal ones. To exemplify how the evaluation of a non-terminal state works let us consider problem state $(2, grey)$. In it neighborhood there are two states: $(0, white)$ and $(1, white)$, with values 1 and -1 respectively. Since $grey$ is the maximizing player, the algorithm assigns the state $(2, grey)$ a value 1, which is the maximum in its neighborhood. Hence, nodes $(2, grey)$ and $(0, white)$, are connected by a solid line. Conversely, nodes $(2, grey)$ and $(1, white)$, are connected by a dashed line. Similarly, considering problem state $(2, white)$, since white is the minimizing player, the algorithm assigns it a value of -1, which is the minimum in its neighborhood. Therefore, the relevant nodes are connected by the

solid line.

Minimax algorithms are used in many fields, such as computer science, decision theory, game theory, statistics, and philosophy (Musah, Boah, & Seidu, 2020). The one presented in this work is the basic version applied for zero-sum games, in which the whole problem space is available. It is worth noticing that there exist more complex versions aimed at handling cooperative games, games of chance, and games of imperfect knowledge. However, those are not discussed in this article.

4 Tic-Tac-Toe Goal Spaces

Tic-tac-toe provides a fully specified and tractable problem space, allowing alternative abstract representations to be constructed and formally compared within an identical task environment. The methods presented in this article were applied to obtain psychologically interpretable and computationally tractable representations of the tic-tac-toe problem space. The full formal construction is reported in Appendix A. Below, we summarize the main simplifying assumptions underlying that construction, all of which preserve the goal structure of the original task.

First, problem states were treated as equivalent up to rotational and reflectional symmetries of the board (for details see Figure 6 in Appendix A).

Second, an ad hoc minimax procedure (see Equation (1) in Appendix A) was applied to derive a goal-space representation of the tic-tac-toe problem space. Only solution paths leading to a win were considered correct, whereas draw outcomes were mapped to the failure state. This restriction allows the analysis to focus on goal-directed behavior while maintaining a well-defined success criterion.

Third, individual moves were mapped onto eight tactical operations (see Table 4)

commonly described in the cognitive literature on human tic-tac-toe play (Newell & Simon, 1972; Crowley & Siegler, 1993). Problem states connected to the goal through identical sequences of tactical operations were therefore treated as equivalent.

Fourth, opponent moves that do not affect an inevitable win were collapsed into a single operation, merging states that differ only in the final unavoidable response of the opponent.

The last two abstraction steps induce two homomorphic representations of the same underlying problem space, denoted $\mathcal{P}_1$ and $\mathcal{P}_2$ (see Appendix A). Although both representations preserve optimal solution paths, they differ in structural complexity and in the dependency relations among problems, thereby yielding alternative hypotheses regarding the organization of strategic knowledge.

Because the resulting knowledge structures remain large, each goal space was further partitioned into largely independent subspaces, yielding $\mathcal{P}_1^a$, $\mathcal{P}_1^b$, $\mathcal{P}_2^a$, and $\mathcal{P}_2^b$. Importantly, the subspaces $\mathcal{P}_1^b$ and $\mathcal{P}_2^b$ are isomorphic, implying that the corresponding knowledge spaces are structurally equivalent despite originating from different abstraction procedures. In contrast, $\mathcal{P}_1^a$ and $\mathcal{P}_2^a$ retain distinct dependency structures.

The knowledge spaces derived from these subspaces correspond to the goal and problem spaces examined in the following simulation and empirical studies (Sections 5 and 6) . This construction provides a controlled setting in which competing cognitive representations of the same underlying problem space can be formally compared within an identical behavioral task . In the simulation study, this allows differences in model performance to be attributed exclusively to representational assumptions. In the empirical study, the same representations are then evaluated against observed patterns of human performance.

5 Simulation Study

A simulation study was conducted to evaluate (a) parameter recovery and identifiability of the MSPM in the two-player game used in the empirical application presented in the next section, and (b) the ability of model selection criteria to discriminate between alternative knowledge spaces derived from abstract representations of the same underlying problem space. The simulation establishes the methodological basis for interpreting the empirical findings. A summary of the main results is reported below. Complete details of the simulation study are provided in the online supplementary materials.

5.1 Design

In the simulation study, the three goal spaces derived in Section 4 were considered: $\mathcal{P}_1^a$, $\mathcal{P}_2^a$, and $\mathcal{P}^b$, together with their corresponding knowledge spaces $\mathcal{K}_1^a$ (8,880 knowledge states), $\mathcal{K}_2^a$ (188 knowledge states), and $\mathcal{K}^b$ (46 knowledge states). The goal spaces $\mathcal{P}_1^a$ and $\mathcal{P}_2^a$ constitute homomorphic representations of the same underlying problem space, where $\mathcal{P}_1^a$ provides a more concrete description and $\mathcal{P}_2^a$ a more abstract one.

Data were generated under six conditions resulting from the crossing of three knowledge spaces ($\mathcal{K}_1^a$, $\mathcal{K}_2^a$, and $\mathcal{K}^b$) and two error levels (low vs. high). Error levels were manipulated through the transition probabilities governing the model. Specifically, careless-error parameters (β_{if}) were sampled from either a random uniform distribution in the interval $(0, .1]$ or in the interval$(0, .3]$, whereas non-lucky-guess parameters (η_{if}) were sampled from either $[.9, 1)$ (low error) or $[.7, 1)$ (high error). Remaining transition probabilities were randomly drawn and normalized to satisfy model constraints. By construction, all parameter pairs satisfied the theoretical inequality $\beta_{ij} - \eta_{ij} \geq 0$

for $j \neq f$.

For each condition, 500 datasets were generated, yielding 3,000 datasets overall. Data generation followed the standard MSPM simulation algorithm: a knowledge state was randomly sampled from the generating knowledge space, and for each item a solution path was produced by sequentially applying transition probabilities until reaching either the goal or failure state (see also (Stefanutti et al., 2021)).

5.2 Identifiability and Parameter Recovery

Local identifiability was assessed using a multi-start iterative estimation procedure. For a dataset, the model was re-estimated from 50 distinct initial values. If parameters are locally identifiable, solutions with equal likelihood should coincide (up to numerical precision). With one exception, this condition was satisfied across all knowledge spaces: parameter standard deviations across equivalent solutions were less or equal to .02. A single parameter in $\mathcal{K}_1^a$ showed a larger variability (SD = .11), indicating potential local non-identifiability in that specific case. No other issues arised.

Across conditions, most transition probabilities were recovered with negligible bias. Bias estimates were centered at or near zero in both low- and high-error levels, demonstrating that the MSPM can recover generating parameters with high accuracy under correct model specification.

A small number of parameters showed modest bias in the high-error condition. In particular, some β_{ij} parameters exhibited small negative bias, and certain $\eta_{i,g}$ parameters showed slightly increased variability. These effects were limited in magnitude and were not systematically replicated across error levels. One plausible explanation is occasional convergence to local maxima, as each simulated dataset was estimated only

once in the recovery analysis.

Systematic patterns in variability were observed. The standard deviation of bias of the β_{ij} tended to increase for transitions farther from the goal state (i.e., those associated with more difficult problems), reflecting reduced information in later-stage transitions. Conversely, the standard deviation of the bias in η_{ij} tended to be larger for transitions closer to the goal state. These patterns reflect expectatins concerning this parameters. In fact, the efficiency of the estimates of the the β_{ij} parameters decreases when observed correct transitions are scares in the data set, whereas the efficiency of the estimates of the η_{ij} parameters decreases when observed incorrect transitions are scares in the data set.

Overall, the results indicate that if the sample size is sufficiently large, under correct model specification, the MSPM provides largely unbiased parameter estimates with stable recovery properties across knowledge spaces and error levels.

5.3 Model Selection

To evaluate discriminability between alternative knowledge spaces, MSPMs based on $\mathcal{K}_1^a$ and $\mathcal{K}_2^a$ were fitted to data generated under each of these knowledge structures (conditions 1–4 in the supplementary material). Model selection performance was assessed using AIC and AICc.

A clear difference emerged between the two criteria. The AIC systematically favored the smaller model, irrespective of the true data-generating structure, leading to incorrect selection in half of the conditions. In contrast, the AICc consistently identified the correct knowledge space across all replications in all four conditions. Thus, in this context, the small-sample correction embodied in AICc appears essential for

reliable model discrimination.

6 Empirical application

In this section, an empirical study is presented that aims to test the capability of this approach to obtain and compare different models of the problem space in which human individuals play a game. A "tic-tac-toe-like" game named *pick15* (or number scrabble) has been chosen for this purpose. The study of the properties of Tic-tac-toe-like games arises in various disciplines, like for instance, mathematics (Beck, 2008), computer science (Luger, 2005), and psychology (Crowley & Siegler, 1993; Newell & Simon, 1972). Among the class of tic-tac-toe-like games, pick15 is a numerical game in which two players select numbers from one to nine without repetition from a three-by-three square grid. The first player who picks three numbers that sum to 15 wins. The numbers are placed in a grid, called a magic square, such that each row, column, or diagonal sums to fifteen, resembling the tic-tac-toe board. Pick15 was chosen for two reasons. First, the problem spaces of pick15 and tic-tac-toe are isomorphic to one another (Michon, 1967; Newell & Simon, 1972), meaning that the spatial strategies studied for tic-tac-toe map identically in pick15. Therefore, the construction of the underlying problem and goal spaces can be performed in the tic-tac-toe domain and subsequently applied to Pick 5 empirical data. Second, while tic-tac-toe is very common in Italy, the Pick15 variant is relatively unknown. We considered that using standard tic-tac-toe would have been too easy for young adults, who already have explicit experience with the game, likely resulting in a ceiling effect on our set of problems.

6.1 Materials and participants

The data were collected by using a ShinyR (Chang et al., 2024) implementation in Italian language of the 36 problems in Q_1 defined in the previous section. The problems were administered to each participant in a randomized order. After a detailed description of the rules, participants were given the following instruction:

(a) solving the problem means that the participant must play the moves for both players;

(b) the goal is that player 1 should win;

(c) each move must be optimal in respect of the player who did it (i.e., each player must try not to lose). Moreover, an example of the optimal and non-optimal move was presented to the participant.

An example of how the problem was presented to the participants is:

The player blue plays first, and player red is supposed to win.

4	9	2
3	5	7
8	1	6

For every problem, the program records each player's move until the player made an error or solved the problem correctly. Then, the participants receive feedback for each answer. In the case of the correct answer, the feedback was "You have solved the problem in an optimal way", in the case of an incorrect answer, the feedback was

"You made a mistake solving the problem". In the resolution of the problem, the participants were asked to plan ahead and take time before performing the solution. Each participant performed an initial trial consisting of two problems that were not used in the main experimental procedure and could be solved in three moves each. After they completed all the problems, participants filled a questionnaire with *i*) personal information, such as gender, age, and level of schooling, and *ii*) if they had previous knowledge about this game or a similar one. In case the response to the last question was positive, they were asked to indicate the name of such game. The experimental procedure was carried out in a laboratory of Padova University. Participants were sitting in front of a computer screen performing the procedure by using the mouse. Before starting the task, the experimenter briefly explained the procedure.

The sample consisted of 148 Italian students (74% female) recruited at the University of Padova. Within the sample, the age ranged between 20 and 29 years (M=21.4; SD=1.75). 82% of the participants had a high school diploma, 15% had a bachelor's degree, 2% had a master's degree, and the remaining a middle school diploma. 55% of the participants said to have previous knowledge about similar games, and all of them indicate tic-tac-toe as a similar game.

6.2 Statistical Methods

The three knowledge spaces $\mathcal{K}_1^a$, $\mathcal{K}_2^a$, and $\mathcal{K}^b$, derived from their respective problem spaces for the pick15, were empirically tested using the MSPM presented in Section 2.3. The Pearson χ^2 statistic was applied to assess the goodness-of-fit of the MSPM. However, the approximation to the asymptotic distribution of the χ^2 statistic may be inaccurate when dealing with large sparse matrices. To address this, a parametric

bootstrap procedure (Efron, 1979) with 1,000 replications was employed to compute the p-value for the χ^2 statistic. Furthermore, to compare the two alternative models derived from the pick15, the Akaike Information Criterion (AIC; Akaike, 1973) and the corrected Akaike Information Criterion (AICc; Hurvich & Tsai, 1989) were used as model selection criteria.

Finally, although pick15 and tic-tac-toe have isomorphic problem spaces, the underlying mechanisms for solving them differ. Tic-tac-toe strategies rely on the spatial properties of the board, while pick15 strategies are based on arithmetic properties. For example, in tic-tac-toe, player X's strategy to 'block' involves placing an X in a line where two O's are already present. In pick15, blocking requires identifying a cell containing a number that, together with two numbers already chosen by the opponent, sums to 15.

Given that the problems were presented in a magic square in this application, both the spatial and arithmetic mechanisms could be applied to solve the problems. The arithmetic strategies are supposed to be less intuitive and more cognitively demanding regarding the spatial strategy. Moreover, it is reasonable that most participants had previous experience with the tic-tac-toe and, hence, with spatial strategies. It is also reasonable that the participants who recognized strong similarity between the two games had both the spatial and the arithmetic strategies available. As a consequence, the availability of multiple solution strategies for the same problem should make it easier. Moreover, it may add symmetries to the problem space, which in turn may make equivalences between apparently different problems more explicit.

In this application, $\mathcal{K}_2^a$ is, by construction, the more abstract representation and is expected to provide a better fit to players who have access to both spatial and arithmetic strategies. In contrast, $\mathcal{K}_1^a$ being a more concrete representation, is expected

to provide a better fit to players with only one type of strategy available. To test this hypothesis, the data were divided into two groups. The first, referred to as the *tic-tac-toe group*, consists of the response patterns of players who correctly recognized tic-tac-toe as a 'similar' game. The second, referred to as the *pick15 group*, contains the response patterns of participants who failed to recognize the similarity with tic-tac-toe. The knowledge spaces presented earlier were fitted to both datasets using the MSPM. Based on this, the following hypothesis can be formulated:

(H1) The knowledge space $\mathcal{K}_2^a$ will provide a better fit for the tic-tac-toe group, while $\mathcal{K}_1^a$ will fit better with the pick15 group.

Like in the previous analysis, the goodness-of-fit of the MSPM was evaluated using the Pearson χ^2 statistic. The p-value of the χ^2 was computed through a bootstrap procedure with 1,000 replications. Model comparisons were made using the AICc. [1]

6.3 Results and Discussion

Regarding the absolute fit of the models, and considering a Type I error rate $\alpha = .10$, knowledge spaces $\mathcal{K}_2^a$ and $\mathcal{K}^b$ fit the data well, whereas $\mathcal{K}_1^a$ does not fit the data. Table 2 presents the fit results, where the Pearson χ^2 statistic and its bootstrapped p-value are displayed in the second and third columns, respectively. The model selection criteria (i.e., AIC and AICc) are shown in the last two columns. Coherently with the absolute fit of the models, between $\mathcal{K}_1^a$ and $\mathcal{K}_2^a$ the knowledge space that minimizes both the AIC and AICc is $\mathcal{K}_2^a$.

[1] The data and code used in this study are available at `https://osf.io/6uk7w/?view_only=342e9bf775e0437798abcc2c82ab5379`

Table 2: Knowledge spaces in the first column, the Pearson χ^2 statistic and its bootstrapped p-values in the second and third columns, respectively, Akaike Information Criterion in the third column, and corrected Akaike Information Criterion in the fourth column.

$\mathcal{K}$	χ^2	p-value	AIC	AICc
$\mathcal{K}_1^a$	1.73×10^{15}	0.05	25,295	7,098
$\mathcal{K}_2^a$	2.70×10^{14}	0.21	8,014	6,862
$\mathcal{K}^b$	9.87×10^{3}	0.80	2,267	2,367

These results show that, for the entire sample, the more abstract representation provides a better fit to the data. This outcome was expected, given that the participants were university students, who are likely to possess the cognitive ability to apply more abstract strategies.

Regarding the comparison between the two groups, the data from the pick15 group fitted all knowledge spaces. The bootstrapped p-values are 0.15 for $\mathcal{K}_1^a$ with a χ^2 statistic of 9.77×10^{13} and 0.12 for $\mathcal{K}_2^a$ with a χ^2 statistic of 1.83×10^{14}. Similarly, the bootstrapped p-values for the tic-tac-toe group are 0.20 for $\mathcal{K}_1^a$ with a χ^2 statistic of 1.62×10^{14} and 0.40 for $\mathcal{K}_2^a$ with a χ^2 statistic of 1.40×10^{14}.

Table 3: The table displays the knowledge space in the first column and the corrected Akaike Information Criterion for the tic-tac-toe group and the pick15 group in the second and third columns, respectively.

$\mathcal{K}$	tic-tac-toe	pick15
$\mathcal{K}_1^a$	3,583	**3,308**
$\mathcal{K}_2^a$	**3,562**	3,314

779 The results of the model comparison are displayed in Table 3. Consistent with
780 hypothesis (H1), the larger knowledge structure $\mathcal{K}_1^a$ minimizes the AICc for the pick15
781 group, while the smaller structure $\mathcal{K}_2^a$ minimizes it for the tic-tac-toe group.

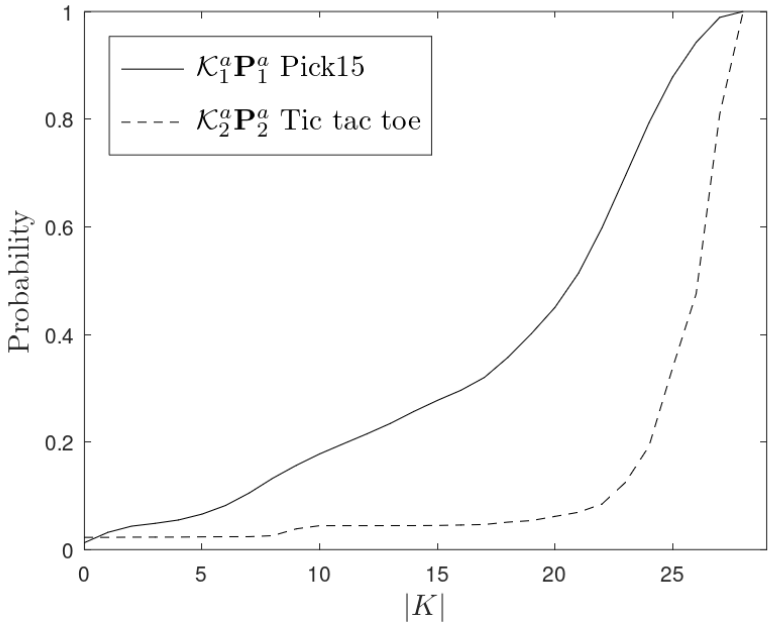


Figure 5: The figure illustrates the two cumulative probability distributions based on the knowledge state cardinalities $|K|$. The dashed line represents the cumulative probability distribution of the pick15 group estimates for $\mathcal{K}_a^1$, while the solid line represents the cumulative probability distribution of the tic-tac-toe group estimates for $\mathcal{K}_a^2$.

Figure 5 displays the cumulative distribution of knowledge state probabilities as a function of the cardinality of the knowledge states. For both groups, the estimated probability distributions from the models that best fit the data were used (i.e., $\mathcal{K}_1^a$ for the pick15 group and $\mathcal{K}_2^a$ for the tic-tac-toe group). The solid line represents the cumulative distribution for the pick15 group, and it dominates the dashed line which represents the cumulative distribution for the tic-tac-toe group. These results indicate that the knowledge states of participants in the tic-tac-toe group tend to have a higher cardinality, while those in the pick15 group tend to be of lower cardinality.

Overall, the comparisons between groups suggest that being aware of previous experience with isomorphic games improved performance, as indicated by the cumulative curves. Moreover, the data seems to be consistent with the more abstract model, and this is in line with our hypothesis that experienced players used a more abstract representation of the game than beginners.

7 General Discussion

The present study introduces the proof of concepts of several theoretical extensions. Specifically, it expands the Procedural Knowledge Space Theory framework to include competitive games, presenting the concept of a 2-players problem space. In the 2-players problem space, players' operations in the game are categorized into moves and turns: moves represent the individual actions performed by a single player, while turns represent the exchange of moves between two players. To define the optimal move within a two-player problem space, we introduced a method for incorporating the classical minimax algorithm into the PKST framework to identify optimal paths in such spaces. Compared to the previous application of a goal space with the Tower of London (Stefanutti et al., 2021), using the minimax algorithm to create the goal space provides a type of optimization that aligns with the competitive nature of board games.

These theoretical extensions are applied in an empirical investigation of the 2-player problem space of the game Tic-Tac-Toe. In this context, an isomorphic variant of Tic-Tac-Toe, called Pick-15, was administered to a sample of 148 participants.

This study also represents the first empirical application of the theoretical concepts of problem space homomorphism introduced in Brancaccio and Stefanutti (2024).

Specifically, we utilized tactical operations described in the literature (Crowley & Siegler, 1993; Newell & Simon, 1972) to define a homomorphism among three problem spaces, resulting in equivalence classes among operations and problem states. The use of homomorphism in this case offers several practical advantages. First, it simplifies the model by reducing the number of parameters, thereby decreasing its complexity. Second, it preserves the structural properties of the original problem space maintaining a link between spaces through the homomorphism itself. Lastly, homomorphic problem spaces incorporate distinct theoretical assumptions, enabling their comparison and empirical testing.

The empirical study successfully validated the problem space of Pick-15, providing initial evidence supporting the applicability of the extended PKST framework to competitive two-player games. A post-hoc analysis suggests that data from the players who self-reported previous experience with similar games, such as Tic-Tac-Toe, fitted better with the more abstract representation of the problem space and demonstrated superior overall performance. This result indicates that the framework could potentially model experience and, eventually, expertise, as practice appear to shape different problem space representations.

Few limitations are present in this study. On the theoretical side, the application of the Minimax algorithm to derive the Tic-Tac-Toe problem space is not yet fully integrated into the broader PKST framework. Nonetheless, Stefanutti and Brancaccio (2025) recently extended the framework by introducing the concept of an optimizing path space. Further theoretical work exploring the relationship between optimizing path spaces and the goal space derived in this study could embed this latter into the general theoretical framework.

In the empirical study, prior experience with similar games was not controlled for in

advance and might serve as a proxy for individual characteristics beyond practice, such as competitiveness, visuospatial abilities, or intelligence. To better evaluate the role of practice and expertise, future studies should account for and control these potential confounding variables. In addition, each participant was asked to play both their own moves and those of their opponent. While this approach is standard for professional or amateur players studying game situations or solving problems (e.g., tsumego in the game of Go or puzzles in chess), it likely reduced ecological validity, particularly for occasional players. A more naturalistic design would not require participants to play their opponent's move, for example, by introducing a computerized opponent.

This limitation could be addressed by developing an adaptive assessment procedure based on problem spaces. Similar to existing adaptive procedures in Knowledge Space Theory, such a system could efficiently and accurately identify a player's knowledge state. By leveraging information within the problem space, the adaptive system could analyze the entire solution path taken by the problem solver and adjust its gameplay as the opponent, tailoring moves to the player's ability. This approach would result in a more efficient and ecological assessment process.

This study provides a robust proof of concept for applying homomorphisms to derive well-defined representations of abstract and human-like task structures. The proposed methodology suggests that this approach can be employed to test various assumptions about expertise, and limitations about human performance. Theories of expertise often posit that experts accumulate knowledge through chunking mechanisms and develop strategies for generalization that might result in abstract problem spaces. For instance, in template theory (Gobet & Simon, 1996), experts develop abstract structures such as templates. Similarly, this methodology could be applied to decision-making theories. For example, prospect theory (Kahneman & Tversky, 1979) emphasizes the role of

heuristics in guiding decisions.

The approach outlined in this manuscript can be integrated into these frameworks to generate specific predictions about the types of abstractions, whether templates, heuristics, or others, and their effects on the problem space. While in current applications the goal space has been obtained by assuming optimal performance through minimax, the same methodology can be adapted to incorporate suboptimal paths. For example, one may restrict the problem space to paths compatible with a particular heuristic or impose constraints that reflect known limits in human cognition, consistent with behavioral-economic formulations of bounded rationality (Simon, 1955, 1956; Simon et al., 1990). These predictions can then be empirically tested and compared using statistical indices, offering a concrete means of evaluating theoretical models in specific problem-solving contexts. Such integration would further validate the utility of the PKST framework while deepening our understanding of the tested phenomena and their broader implications.

References

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In B. N. Petrov & F. Csaki (Eds.), *Second international symposium on information theory* (pp. 267–281). Academiai Kiado.

Anderson, J. R. (1982). Acquisition of cognitive skill. *Psychological review*, *89*(4), 369. doi: https://doi.org/10.1037/0033-295X.89.4.369

Anderson, J. R. (2014). *Rules of the mind*. Psychology Press. doi: https://doi.org/10.4324/9781315806938

Beck, J. (2008). *Combinatorial games: tic-tac-toe theory* (Vol. 114). Cambridge university press Cambridge.

Brancaccio, A., de Chiusole, D., Epifania, O. M., Anselmi, P., Spinoso, M., Mazzoni, N., ... et al. (2026). Two markov solution process models for the assessment of planning in problem solving. *Psychometrika*, *91*(1), 360–390. doi: 10.1017/psy.2025.10042

Brancaccio, A., de Chiusole, D., & Stefanutti, L. (2023). Algorithms for the adaptive assessment of procedural knowledge and skills. *Behavior Research Methods*, *55*(7), 3929–3951. doi: https://doi.org/10.3758/s13428-022-01998-y

Brancaccio, A., & Stefanutti, L. (2024). Homomorphisms between problem spaces. *Journal of Mathematical Psychology*, *123*, 102888. doi: https://doi.org/10.1016/j.jmp.2024.102888

Brancaccio, A., & Stefanutti, L. (2026). *Analysis code for "an application of procedural knowledge space theory to the study of strategic planning in board games"*. Open Science Framework. Retrieved from `https://osf.io/6uk7w/?view_only=342e9bf775e0437798abcc2c82ab5379`

Chang, W., Cheng, J., Allaire, J., Sievert, C., Schloerke, B., Xie, Y., ... Borges, B. (2024). shiny: Web application framework for r [Computer software manual]. Retrieved from `https://CRAN.R-project.org/package=shiny` (R package version 1.10.0)

Chase, W. G., & Simon, H. A. (1973). Perception in chess. *Cognitive psychology*, *4*(1), 55–81. doi: https://doi.org/10.1016/0010-0285(73)90004-2

Crowley, K., & Siegler, R. S. (1993). Flexible strategy use in young children's tic-tac-toe. *Cognitive Science*, *17*(4), 531-561. doi: https://doi.org/10.1207/s15516709cog1704_3

de Chiusole, D., Spinoso, M., Anselmi, P., Bacherini, A., Balboni, G., Mazzoni, N., ... others (2024). Psycassist: A web-based artificial intelligence system designed for adaptive neuropsychological assessment and training. *Brain Sciences*, *14*(2), 122. doi: 10.3390/brainsci14020122

De Groot, A. (1978). *Thought and choice in chess* (Vol. 4). Walter de Gruyter.

DeVries, R., & Fernie, D. (1990). Stages in children's play of tic tac toe. *Journal of Research in Childhood Education*, *4*(2), 98–111.

Doignon, J.-P., & Falmagne, J.-C. (1999). *Knowledge Spaces*. Berlin, Heidelberg, and New York: Springer-Verlag.

Efron, B. (1979). Bootstrap methods: another look at the jackknife. *Annals of Statistics*, *7*, 1–26. doi: 10.1214/aos/1176344552

Ethier, S. N., & Kurtz, T. G. (2009). *Markov processes: characterization and convergence*. John Wiley & Sons.

Gobet, F., Retschitzki, J., & de Voogt, A. (2004). *Moves in mind: The psychology of board games*. Psychology Press.

Gobet, F., & Simon, H. A. (1996). Templates in chess memory: A mechanism for

recalling several boards. *Cognitive psychology*, *31*(1), 1–40.

Hurvich, C. M., & Tsai, C. (1989, 06). Regression and time series model selection in small samples. *Biometrika*, *76*(2), 297-307. doi: 10.1093/biomet/76.2.297

Ishay, A., & Lee, J. (2025). Llm+ al: Bridging large language models and action languages for complex reasoning about actions. In *Proceedings of the aaai conference on artificial intelligence* (Vol. 39, pp. 24212–24220).

Kahneman, D., & Tversky, A. (1979). Prospect theory: An analysis of decision under risk. *Econometrica*, *47*(2), 363–391. doi: 10.1142/9789814417358_0006

Luger, G. F. (2005). *Artificial intelligence: structures and strategies for complex problem solving*. Pearson education.

Michon, J. A. (1967). The game of jam: an isomorph of tic-tac-toe. *The American Journal of Psychology*, *80*(1), 137–140. doi: doi.org/10.2307/1420555

Musah, I., Boah, D. K., & Seidu, B. (2020). A comprehensive review of solution methods and techniques for solving games in game theory. *Journal of Game Theory*, *9*(2), 25–31. doi: doi:10.5923/j.jgt.20200902.01

Newell, A., & Simon, H. A. (1972). *Human problem solving* (Vol. 104) (No. 9). Prentice-hall Englewood Cliffs, NJ.

Schuck, N. W., & Niv, Y. (2019). Sequential replay of nonspatial task states in the human hippocampus. *Science*, *364*(6447), eaaw5181.

Simon, H. A. (1955). A behavioral model of rational choice. *The quarterly journal of economics*, 99–118.

Simon, H. A. (1956). Rational choice and the structure of the environment. *Psychological review*, *63*(2), 129.

Simon, H. A., et al. (1990). Invariants of human behavior. *Annual review of psychology*, *41*(1), 1–20.

Stefanutti, L. (2019). On the assessment of procedural knowledge: From problem spaces to knowledge spaces. *British Journal of Mathematical and Statistical Psychology*, *72*(2), 185–218. doi: https://doi.org/10.1111/bmsp.12139

Stefanutti, L., & Brancaccio, A. (2025). The assessment of global optimization skills in procedural knowledge space theory. *Journal of Mathematical Psychology*, *125*, 102907. doi: https://doi.org/10.1016/j.jmp.2025.102907

Stefanutti, L., de Chiusole, D., & Brancaccio, A. (2021). Markov solution processes: Modeling human problem solving with procedural knowledge space theory. *Journal of Mathematical Psychology*, *103*, 102552. doi: https://doi.org/10.1016/j.jmp.2021.102552

Van Opheusden, B., Kuperwajs, I., Galbiati, G., Bnaya, Z., Li, Y., & Ma, W. J. (2023). Expertise increases planning depth in human gameplay. *Nature*, *618*(7967), 1000–1005. doi: https://doi.org/10.1038/s41586-023-06124-2

Watson, D. G., Arnold, J. E., & Tanenhaus, M. K. (2008). Tic tac toe: Effects of predictability and importance on acoustic prominence in language production. *Cognition*, *106*(3), 1548–1557.

8 Appendix A: The Tic-Tac-Toe goal space

Tic-tac-toe was chosen as the application domain in this study for two main reasons. First, there is extensive literature on the game's strategies and problem space (Beck, 2008; Crowley & Siegler, 1993; Newell & Simon, 1972), which provides a well-established theoretical foundation. Second, tic-tac-toe is a simple game that allows us to test the proposed approach while minimizing the confounding factors that would be introduced by more complex domains of application.

The tic-tac-toe problem space $\mathbf{P}_{ttt} = (S_{ttt}, \Omega_{ttt}, R_{ttt})$ includes all possible board configurations, where symmetric configurations obtained through rotations or reflections are treated as equivalent problem states. For example, the leftmost and central configurations shown in Figure 6 illustrate a rotation: one configuration can be obtained from the other by a 180 degree rotation. The rightmost configuration illustrates a reflection of the left most configuration along the vertical axis. Considering equivalence

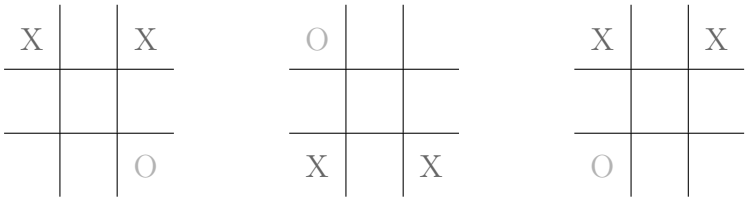


Figure 6: Examples of equivalent problem states up to rotation and reflection.

under rotations and reflections, the total number of distinct problem states is 765.

Of those problem states, 138 are terminal ones: 91 in which the first player (player 1) wins, 44 in which the second player (player 2) wins, and 3 draw configurations. Considering player 1 as the maximizing player and player 2 as the minimizing one, at each terminal state is assigned a value using the following evaluation function:

$$v_{ttt}(s) = \begin{cases} 10 - p(s) & \text{if player 1 won} \\ 0 & \text{if is a draw} \\ p(s) - 10 & \text{if player 2 won} \end{cases} \quad (1)$$

Here, $p(s)$ represents the number of cells containing an X or an O in the game configuration represented by problem state s. The parameters $p(s)$, may take any integer value from 5 to 9, and are aimed at penalizing long games. The larger the number of moves, the higher the value of this parameter. As a consequence, the absolute value

of $v_{ttt}(s)$ monotonically decreases with the number of moves. The constant 10 ensures that the evaluation is always a positive number for player 1 and a negative number for player 2. Of course this choice of numbers is not the only possible one. Others are possible that respects all the given constraints. The value of $v_{ttt}(s)$ for any non-terminal state is obtained through the application of the backtracking mechanism discussed in the previous section.

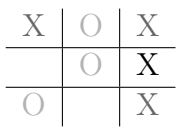
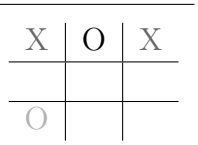
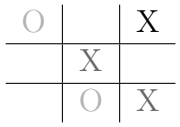
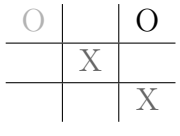
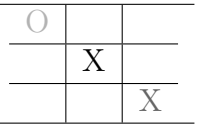
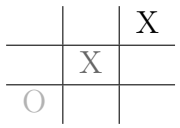
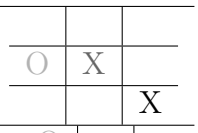
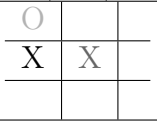
In this application, any path whose terminal configuration corresponds to a draw is mapped to the failure state in the goal space, and all problem states traversed exclusively by such paths are likewise removed or reclassified as failures. This means that only the paths leading to a win are considered correct , and paths leading to a draw are considered incorrect. This assumption is arbitrary for this application and other situation can suggest difference assumption a discussed in Section 3. This resulted in a goal space containing 614 problem states. Such a goal space is too large to derive a knowledge space using the methods presented in Stefanutti (2019).

In response to these issues, other less obvious and more abstract symmetries have been considered to further simplify the problem space. Abstract representations of $\mathbf{P}_{ttt}$ have been produced in various ways, and for each of them, the algorithm proposed by Brancaccio and Stefanutti (2024) was used to test that all of them are homomorphic to the concrete one. These abstract representations are described in the rest of the section. As for the first abstract representation, Newell and Simon (1972) proposed a set of *tactical operations*, that were later refined and applied by Crowley and Siegler (1993) in their research.

Table 4 presents the eight different tactical operations applied in this study. The first column lists the name of each operation, while the second column provides a brief, informal definition. The definitions are given in terms of the X symbol, but they hold for the O symbol. The third column describes an example, which is graphically represented in the fourth column. In each diagram in the fourth column, the last move played is shown in black, while the others appear in shades of grey. A formal description of the operations can be found in the supplementary material.

These definitions allow for ambiguous situations, where two tactical operations could both be considered valid candidates for the same move. For instance, the move of X in the winning diagram of Table 4 satisfies the criteria for both the winning and the empty-side tactics. Newell and Simon (1972), in their implementation, proposed a strict weak ordering between operations to resolve such ambiguities by selecting the most suitable option when two or more tactical operations can be applied to the same move. Specifically, in this work the strict weak order depicted in Figure 7 is applied.

Table 4: Tactical operations lists, with informal definitions (column 2), examples (column 3), and diagrams (column 4). The last move in each diagram is black, earlier moves are grey

Name	**Definition**	**Example**	**Diagram**
win	It occurs when three Xs are on the same line (i.e., row, column, or diagonal).	In the diagram, X playing on the right side complete the line and win the game.	
block	It occurs when symbol occupies a line that contains two of the opponent ones.	In the diagram, O playing on the top side prevent X from winning by playing in that square.	
fork	It occurs when the symbol occupies an intersection between two lines that contains two of her symbols and none of the opponent's ones.	In the diagram, X already has a symbol on the central square and the bottom right corners. By playing on the top right corner, X will win the game in the next move by playing the right side or the bottom left corner.	
block fork	It occurs when the symbol occupies an intersection between two lines that contains two of the opponents' symbols and none of the player.	In the diagram, O playing on the top right corner prevents X from making a fork by playing in that square.	
center	It occurs when a symbol occupies the central square	In the diagram, X playing on the central square.	
opposite corner	It occurs when a symbol occupies a corner square, and in the same diagonal, there is an opponent's symbol in the other corner.	In the diagram, X playing on the opposite corner to O.	
empty corner	It occurs when a symbol occupies a corner square.	In the diagram, X playing on the right bottom corner square.	
empty side	It occurs when a symbol occupies a side square.	In the diagram, X playing on the left side square just below O.	

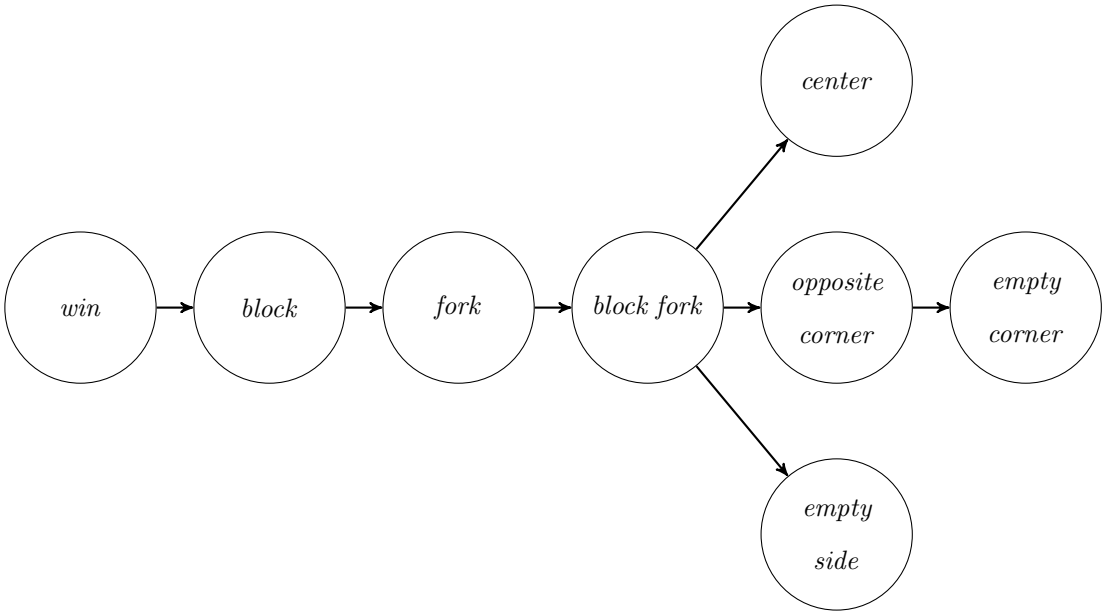


Figure 7: Graphical representation of the strict weak order among the tactical operations.

Figure 7 can be interpreted as follows: Given two tactics, ω_1 and ω_2, the operation ω_1 is selected over ω_2 if there is a directed path connecting ω_1 to ω_2. For example, the move depicted in the fork diagram in Table 3 will be labeled as a fork rather than an opposite corner because fork is connected to opposite corner by a directed path, specifically *fork* $\rightarrow$ *block fork* $\rightarrow$ *opposite corner*. Additionally, the operations opposite corner, center, and empty side are considered incomparable with one another, as they refer to different squares on the grid.

Therefore, to simplify the original goal space $\mathbf{P}_{ttt}$, all the operations are mapped to one of the eight tactical operations presented above using the described procedure. In this specific application, the function γ of the homomorphism is simply the function that maps every single basic operations to a tactical operation according to the rules

that are described above.

Next, all problem states connected to the goal state by exactly the same sequences of tactical operations are considered equivalent. Formally, two problem states $s_1, s_2 \in S_{ttt}$ are considered equivalent if, for each $\pi \in \Omega^*_{ttt}$, $(s_1, \pi, g) \in R_{ttt}$ if and only if $(s_2, \pi, g) \in R_{ttt}$. This provides a homomorphism between the concrete problem space of the tic tac toe and a more abstract one based on tactics.

The resulting goal space $\mathbf{P}_1 = (S_1, f, g, \Omega_1, R_1)$ contains 47 problem states. A set Q_1 of 36 problems requiring at least three moves was derived from $\mathbf{P}_1$. The procedure described in Stefanutti (2019) was used to derive a knowledge space $\mathcal{K}_1$ from the goal space $\mathbf{P}_1$. The derived knowledge space contains 56,421 knowledge states. A large knowledge space like $\mathcal{K}_1$ is difficult to validate empirically, as it requires an even larger sample size. To address this, $\mathbf{P}_1$ was split into two subspaces that minimized the number of shared paths. This approach was successfully used in Stefanutti et al. (2021), allowing us to test all the relationships between problems assumed in the problem space while at the same time limiting the combinatorial explosion of the corresponding knowledge spaces. The resulting subspaces $\mathbf{P}^a_1$ and $\mathbf{P}^b_1$ are displayed in Figure 8. As it can be seen, they only share the final move, leading to the goal state (from 45 to g in Figure 8).

In the Figure 8 subspace $\mathbf{P}^a_1$ is depicted in black and subspace $\mathbf{P}^b_1$ in blue. In the figure, the goal state is labeled g, and the 45 problem states are arbitrarily labeled from 1 to 45; the failure state f has been omitted to simplify the figure. It should be noted that all problem states except g are connected to f. Subspace $\mathbf{P}^a_1$ contains 38 problem states (i.e., 36 plus the goal and the failure state), while $\mathbf{P}^b_1$ contains 12 problem states (i.e., 10 plus the goal and the failure state). The knowledge spaces $\mathcal{K}^a_1$ and $\mathcal{K}^b_1$, derived from these subspaces, contain 8,880 and 46 knowledge states, respectively.

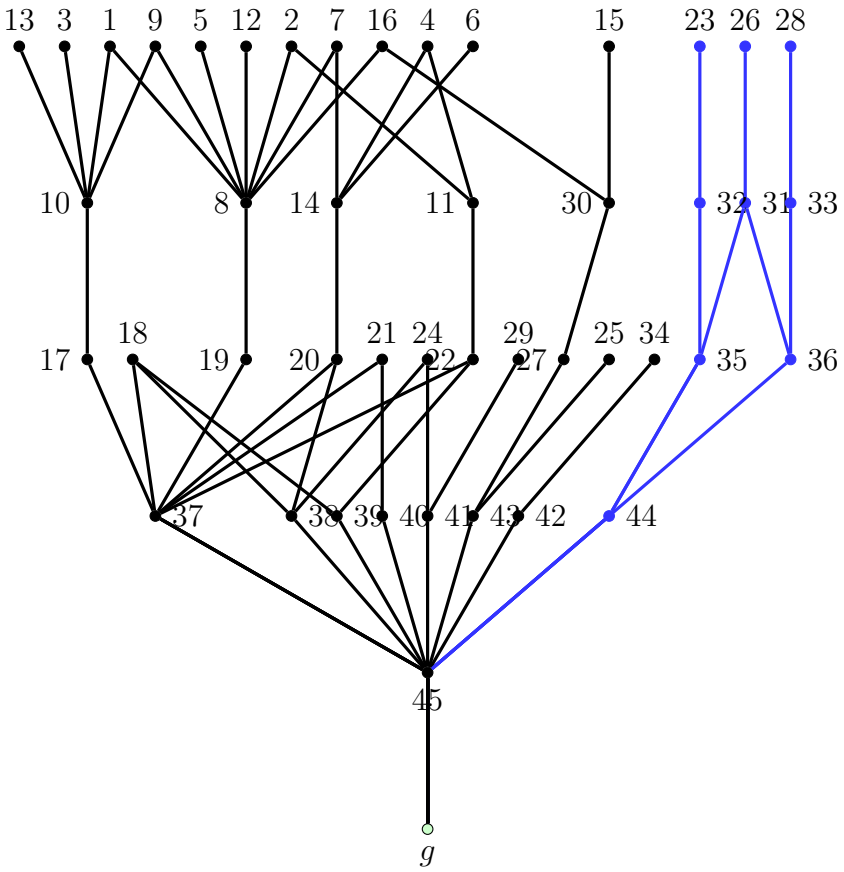


Figure 8: The two goal space $\mathbf{P}_a^1$ in black and $\mathbf{P}_b^1$ in blue.

Further considerations of the obtained problems led to the observation that any two problems differing only in the last turn are indeed equivalent. This is because, regardless of the opponent's move, a win cannot be avoided. Thus, all the last moves of the opponent are mapped into a single operation, and similarly, all the last opponent states are mapped into a single state, resulting in a new homomorphism (ϕ_2, γ_2). The application of these mappings produce a new goal space $\mathbf{P}_2 = (S_2, f, g, \Omega_2, R_2)$.

This last contains 21 problem states, and among all the problems that can be obtained for $\mathbf{P}_2$, the set Q_2 of all and only those that require three or more moves

was considered. To facilitate comparison between the goal spaces, the same partition applied to $\mathbf{P}_1$ was performed on $\mathbf{P}_2$, resulting in two distinct goal subspaces, $\mathbf{P}_2^a$ and $\mathbf{P}_2^b$. It is important to highlight that these subspaces share several paths. The resulting goal space $\mathbf{P}_2$ and the two subspaces are illustrated in Figure 9.

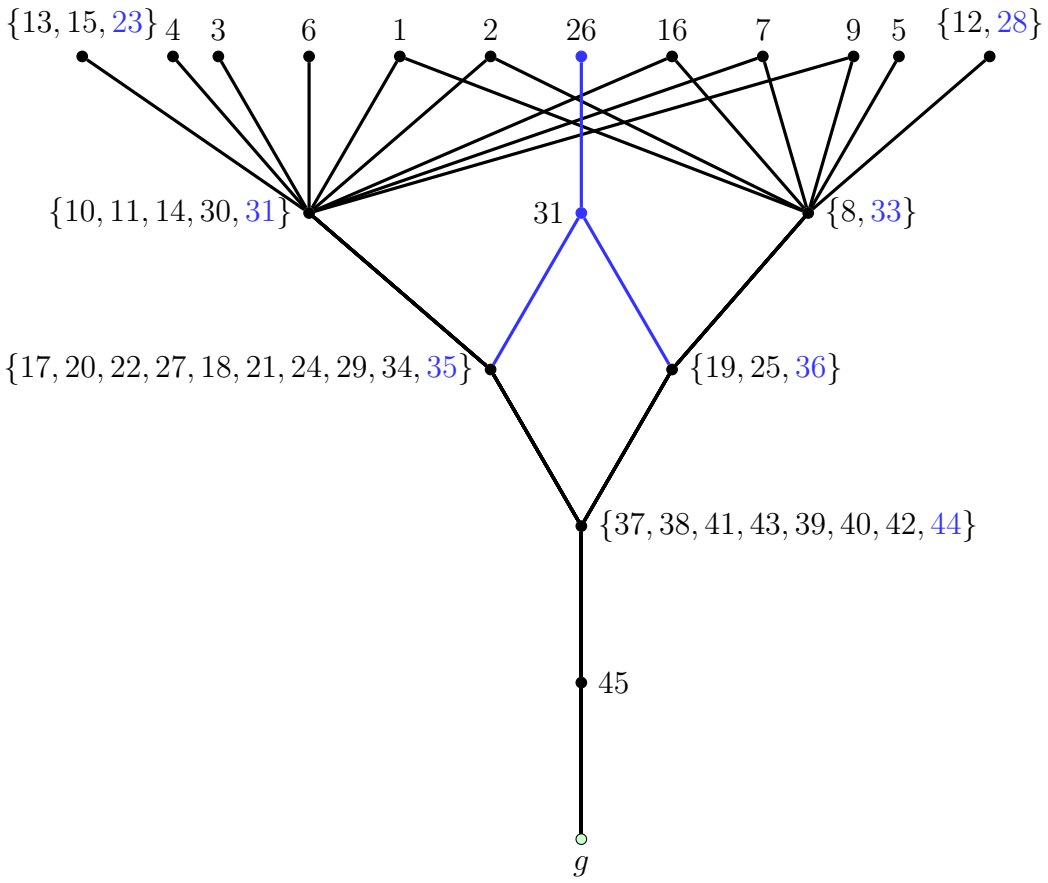


Figure 9: The two goal spaces $\mathbf{P}_a^2$ in black and $\mathbf{P}_b^2$ in blue. In the curly brackets are the problem states in $\mathbf{P}_1$ that are equivalent at that node in the graph.

In Figure 9 subspace $\mathbf{P}_2^a$ is depicted in black and subspace $\mathbf{P}_2^b$ in blue, the goal state is representaed in green. In the figure, each node is labeled with the equivalence classes of problem states in $\mathbf{P}_1$ induced by the homomorphism ϕ_2. This illustrates how several

problem states become equivalent under the new homomorphism. For example, the problem states 12 and 28, together with the corresponding paths leading to g in $\mathbf{P}_1$, are considered equivalent in $\mathbf{P}_2$. Subspace $\mathbf{P}_2^a$ contains 17 problem states (i.e., 15 plus the goal and the failure state), and the knowledge space $\mathcal{K}_2^a$, derived from it, has 188 knowledge states. The problem space $\mathbf{P}_2^b$ is isomorphic to $\mathbf{P}_1^b$, so it will be referred to as $\mathbf{P}^b$. The knowledge space derived from $\mathbf{P}^b$ is $\mathcal{K}^b$, which is isomorphic to $\mathcal{K}_1^b$ (in the sense of an order isomorphism with respect to set inclusion).