CHALMERS
UNIVERSITY OF TECHNOLOGY

UNIVERSITY OF GOTHENBURG

Emergence of Agency from a Causal Perspective

Analysing how goal-directed behaviour emerges over time in systems containing learning agents

Master's thesis in Mathematical Sciences

ALVIN ÅNESTRAND

Department of Mathematical Sciences
CHALMERS UNIVERSITY OF TECHNOLOGY
UNIVERSITY OF GOTHENBURG
Gothenburg, Sweden 2024

Master's thesis 2024

Emergence of Agency from a Causal Perspective

Analysing how goal-directed behaviour emerges over time in systems containing learning agents

ALVIN ÅNESTRAND

UNIVERSITY OF GOTHENBURG

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Mathematical Sciences
Chalmers University of Technology
University of Gothenburg
Gothenburg, Sweden 2024

Emergence of Agency from a Causal Perspective
Analysing how goal-directed behaviour emerges over time in systems containing learning agents
ALVIN ÅNESTRAND

Supervisors:
Olle Häggström, Professor at the Department of Mathematical Sciences, Chalmers University of Technology

James Fox, PhD Candidate at University of Oxford and Research Director at London Initiative for Safe AI (LISA)

Tom Everitt, Staff Research Scientist at Google DeepMind

Examiner:
Torbjörn Lundh, Professor at the Department of Mathematical Sciences, Chalmers University of Technology

Master's Thesis 2024
Department of Mathematical Sciences
Chalmers University of Technology and University of Gothenburg
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Typeset in LaTeX
Gothenburg, Sweden 2024

Emergence of Agency from a Causal Perspective
Analysing how goal-directed behaviour emerges over time in systems containing learning agents
ALVIN ÅNESTRAND
Department of Mathematical Sciences
Chalmers University of Technology and University of Gothenburg

Abstract

Causal models of agents and agentic behavior allows for safety analysis of machine learning systems. Understanding how goal-directed behavior emerges from adapting to an environment is however non-trivial. This thesis addresses the gap between theoretical models and real-world implementations of machine learning systems, though a framework that formalizes the connection between system dynamics and goal-driven behavior.

This thesis introduces novel probabilistic graphical models for describing system dynamics involving learning agents, based on dynamic bayesian networks, which allows for a flexible representation of causal relationships in the training environment. To analyze goal-directed behavior that emerges from interacions between agents and the environment, the thesis also introduces *temporally abstracted models*. Such a model captures the dynamics of a system after the learning process has converged, derived from a model of the learning process. A temporally abstracted model describes potential outcomes involving equilibria between agents and the environment, and can under certain conditions be viewed as a model of goal-directed behavior in the system.

Keywords: Intelligent agent, objective, causal influence diagram, causal model, emergence, learning, equilibria, dynamical systems, goal-directed behavior

Acknowledgements

I would like to thank my supervisor Olle Häggström for his guidence. Without him I would not have had the same freedom in the choice of project, and I would likely not have been able to start the project at all. I thank him for his insightful comments and suggestions regarding potential topics. I also thank him for his patience and understanding when I made significant changes to the project plan.

I thank Torbjörn Lundh for being the examiner of this project. I appreciate his time and effort in reading and commenting on the report.

Tom Everitt and James Fox, I am very grateful for the regular meetings, and for you feedback, support, and help during the project.

The project idea was first conceived by James Fox. His thoughts and ideas about the topic made it easy to get started and gave me a clear direction to follow. His excitement about the topic was very contagious.

I thank Tom for getting me back on track when I inevitably attempted to challenge too complicated problems too early. His advice to start with the simpler cases was something that I apparently needed to hear multiple times, and I am very grateful for his patience.

I also want to thank my family for their support and understanding during this project. I did not always have the time to spend with them that I would have liked. Special thanks to my brother for acting as a sounding board for my ideas, almost always being available for interesting discussions.

Alvin Ånestrand, Gothenburg, 2024-06-17

Contents

List of Figures

List of Abbreviations

2TBN or TBN	2-slice Temporal Bayesian Network
AI	Artificial Intelligence
BN	Bayesian Network
CBN	Causal Bayesian Network
CDBN	Causal Dynamic Bayesian Network
DBN	Dynamic Bayesian Network
DBN-UE	Dynamic Bayesian Network with Update Edges
MDP	Markov Decision Process
ML	Machine Learning
nDBN	n-step Dynamic Bayesian Network
nTBN	n-slice Temporal Bayesian Network
RL	Reinforcement Learning
TAM	Temporally Abstracted Model

Notation

$\mathbf{Anc}^V$	Ancestors of V
$\mathbf{Ch}^V$	Children of V
$\mathbf{Desc}^V$	Descendants of V
$\mathbf{Pa}^V$	Parents of V

$\mathbf{V}$	Variables
$\mathbf{v}$	Values of $\mathbf{V}$
$\mathcal{G}$	Graph
$\mathcal{I}$	Intervention
$\mathcal{M}$	Model
$\mathcal{R}$	Set of Relations
$\text{dom}(V)$	Domain of V
$\text{m}\mathcal{G}$	Mechanised Graph
$\text{m}\mathcal{M}$	Mechanised Model
$\widetilde{V}$	Mechanism Variable for V
$\widetilde{V}$	Mechanism Variable of V
$\widetilde{V}'$	Mechanism Variable of V in a Prior Bayesian Network (Part of a Dynamic Bayesian Detwork)
P or Pr	Probability Distribution
r_V	Relation for V
V	Variable
v	Value of V
$V_{t:t'}$	Variable V at Time t to t'

1

Introduction

Humans are capable of long term planning and acting in order to achieve distant goals. Human values can incorporate things far away, such as people living on the other side of the planet. People altruistically donate to charities in order to support people or animals they have never met.

Current artificially created intelligent agents do not have as complex behavior. However, a more comprehensive understanding of agents and agentic behavior will be needed to create safe and robust agents as the capabilities of artificial agents increase. In a sense, understanding agency is a timeless issue, but in recent years it has nevertheless gained additional urgency from contemporary AI safety discourse, where it has been suggested that keeping AIs non-agentic may be good for safety [1]. But characterizing agency and understanding how it emerges has turned out to be not so easy. This thesis aspires to contribute to this task.

The Causal Incentives Working Group is a group of researchers that are working on a comprehensive framework for modeling agents using causal models [2]. The group is among other things interested in the emergence of agency. Emergence can be described as a phenomenon where a system exhibits properties that its individual components lack. By analysing system dynamics and training procedures for learning agents in a system — both of which are non-agentic in nature — the system can be described as containing agents with various goals after the training has converged.

This thesis aims to formalize this transition. By doing so, it seeks to advance our understanding of agency and contribute to ensure that reliability and safety of artificial agents.

1.1 Motivation

In the field of game theory, the classical models typically presuppose that agents possess complete knowledge about the game's structure, including the roles of other participants and the influence of various stochastic elements within the environment. Such assumptions do not fully capture how agents acquire this knowledge in practice. Similarly, common models for reinforcement learning such as *Markov Decision Processes* (MPDs) [3] or *Partially Observable Markov Decision Processes* (POMDPs) [4], are focused on modeling environment dynamics, rather than the learning process itself. This gap between theoretical assumptions and the dynamics of real-world

scenarios forms the central focus of this thesis. Specifically, it aims to develop a more nuanced understanding of the mechanisms through which agents gain knowledge and subsequently interact with and influence their environments.

This thesis provides a foundational framework that can be instrumental for future explorations into agent behavior and system dynamics. By examining how agents emerge within systems and interact with their surroundings, this work builds towards analysing systems through agentic concepts such as objectives, incentives, and rational decision-making. This approach is not only important for enhancing our theoretical understanding of agent-based systems but could also be vital for practical applications, particularly in designing training protocols and learning environments for artificial agents. Ensuring the safety and reliability of these agents necessitates a robust model that accounts for how agents develop and modify their behavior, and how they in turn affect their environment.

The following sections provide some further explanation of the key ideas and concepts that underpin this thesis. The nature of agents and agency for the type of agentic modeling considered in this thesis is discussed, as well as a brief overview of what is meant by the emergence of agency.

1.1.1 Agency

There are many ways to reason about and define agents and agentic behavior. In this thesis, agents are considered entities that adapt their actions to achieve something in their environment. Such agents can be referred to as goal-adaptive agents.

A particular relevant philosophical perspective for the goals of this thesis are the three levels of abstraction for understanding, explaining, and/or predicting the behavior of a system, as provided by Daniel Dennett [5]. The physical stance is the lowest level of abstraction, where the system is understood as a physical system. The design stance is the middle level, where the system is understood by the purpose of the system's design. The intentional stance is the highest level, where predictions are made based on an agent's beliefs and desires.

A single system can be described using different levels of abstraction. A human consists of atoms, but it is not practical to describe a human using the physical stance, by detailing the position and velocity of every single atom. A higher level allows for a more compressed description of a system. The intentional stance is however not a useful way to predict behaviour for some systems. For instance, a rock does not have beliefs and desires.

The goal of this thesis can be described as providing a formal framework for transitioning from the physical stance, in the form of a complete description of system dynamics, to the intentional stance, in the form of a system containing agents with goals.

1.1.2 Emergence of agency

Emergence refers to when a system exhibits properties that individual components of the system lack independently. Emergence of agency then refers to how a system can be be described as containing goal-adaptive agents at a high level of abstraction, as a result of the system progressing over time.

For instance, consider a reinforcement learning (RL) algorithm for a multi-armed bandit problem. The RL agent's action is to select one out of several options, and receives a reward based on the selection. The agent then updates its policy based on the reward. The process can be described using a graphical model representing the system dynamics over several time steps. A variable for action and reward at each time step are included.

However, we can also model the RL agent with a graph representing the decision and reward at a specific time step. This is done with a model called a *Causal Influence Diagram* (CID) [6]. In a CID, there are designated variables for representing decisions and utilities for a single agent.

The two graphical representations of the RL system are shown in the two graphs in Figure 1.1. The example and the corresponding graphs is inspired by the example in Appendix B in Kenton et al [7].

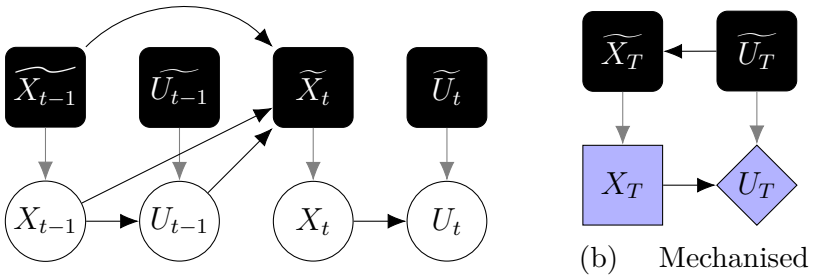
First, consider the left graph. The graph shows two time steps of the algorithm. X_t and U_t are the action and reward respectively at time step t. $\widetilde{X_t}$ is the policy at time step t, describing the probability of choosing each available option. $\widetilde{U_t}$ describes the distribution of rewards for each option at time step t, and can be called the reward mechanism. For simplicity we let $\widetilde{U_t} = \widetilde{U_{t-1}}$, the reward mechansm does not change over time. Note that the policy is updated based on the previous policy, the action taken, and the reward received.

Now, consider the graph to the right. It is the graph for a model called a *mechanised CID* [6]. The mechanised CID provides a more compressed description of the system. The graph shows a decision taken at a time step T. Note that $\widetilde{X_T}$ is causally influenced by $\widetilde{U_T}$, which shows that the policy ia dependent on the reward mechanism.

While the left graph describes the system dynamics over time, which can be compared to Dennett's physical stance, the right graph describes the system as a single agent optimizing for utility, comparable to the intentional stance. The goal of this thesis is to formalize the relationship between the two types of models. This requires a formalization of models for describing the system dynamics, as the graph to the left, and a formalization of the final model as illustrated by the graph to the right. These are the two main contributions of this thesis.

1.2 Contributions

The first contribution of this work is the development of formal models for describing learning or optimization processes, including reinforcement learning and other machine learning (ML) algorithms. The models are designed to be comprehensive



(a) System dynamics over time

(b) Mechanised CID

Figure 1.1: A graph showing system dynamics over time (a) and a mechanised CID (b). Both describe the same system, a reinforcement learning agent for a multi-armed bandit problem.

and general, accommodating a wide range of learning algorithms and systems. They are capable of representing various causal relationships in the learning environment, and the learning process itself is explicitly modeled in the form of parameters that update over time. This is achieved by modeling both system dynamics and the learning process with models based on *Dynamic Bayesian Networks* (DBNs), which are probabilistic graphical model that can represent complex causal relationships in a system [8]. Chapter 3 provides a detailed overview of these models.

The second contribution of this thesis is the formalization of modeling potential stable dynamics of the system, where the learning process has converged. This involves the development of a model a called a *Temporally Abstracted Model* (TAM). From a description of system dynamics using the models developed in the first contribution, the transition to a TAM is referred to as *temporal abstraction*. This is because the TAM does not describe the system at any specific time step, and the relationships between the variables in the TAM represent dependencies through many time steps throughout the learning process. This contribution has applications in evaluating outcomes of agent learning and interaction within a system. The formalization of the temporal abstraction and the TAM is presented in Chapter 4.

1.3 Outline of the thesis

The thesis is structured as follows:

- Chapter 2 provides an overview of the background and related work. It introduces the concepts of causal models and the emergence of agency, and discusses the existing literature on these topics.
- Chapter 3 presents the first contribution of this thesis, which is the development of formal models for describing learning processes or optimization processes.
- Chapter 4 presents the second contribution of this thesis, which is the formalization of temporal abstraction and the development of a Temporally Abstracted Model (TAM). The chapter explains how the TAM can be used to model stable dynamics in a system where the learning process has converged.

- Chapter 5 provides examples of how the models developed in this thesis can be applied to reinforcement learning algorithms. The chapter demonstrates how the models can be used to analyze the behavior of agents in different environments.
- Chapter 6 concludes the thesis with a discussion, areas for future work, and some final remarks.

2

Background

This chapter contains the relevant background theory, starting with introducing the mathematical notation used in this thesis. The following section covers *Markov chains*, a type of stochastic process that is fundamental to the thesis. Then the probabilistic graphical models that are used in the thesis are covered.

2.1 Notation

Random variables are represented with roman capital letters (e.g. V), and vectors of variables are represented using bold type (e.g. $\mathbf{V}$). Outcomes are represented with lower case letters (e.g. v), and vectors of outcomes with bold type (e.g. $\mathbf{v}$). The set of possible outcomes of a variable V is denoted $dom(V)$, and the set of possible outcomes of a set of variables $\mathbf{V}$ is denoted $dom(\mathbf{V}) = \times_{V \in \mathbf{V}} dom(V)$. For simplicity, the domain of a variable is assumed to be finite throughout this thesis. The theory and models considered in this thesis can however be extended to allow for continuous state variables.

To indicate the time step of a variable, a subscript is used (e.g. V_t). The domain remains the same for all time steps for a variable V, which is represented as $dom(V_t) = dom(V)$. A probability distribution over a variable V is denoted $\pi(V)$.

Given a random variable V, the parents of V are denoted $\mathbf{Pa}^V$, and the children of V are denoted $\mathbf{Ch}^V$. The parameter for the conditional probability distribution (CPD) of a variable V is denoted θ_V, and a set of parameters is denoted $\boldsymbol{\theta}$.

In Section 2.3.2, a type of variable called a *mechanism variable* is introduced. Mechanism variables are denoted with a tilde (e.g. $\widetilde{V}$), and the set of mechanism variables for a set of variables $\mathbf{V}$ is denoted $\widetilde{\mathbf{V}}$.

2.2 Markov Chains

Markov chains are a fundamental concept in probability theory and stochastic processes, representing systems that undergo transitions from one state to another in a state space. A general reference for the material in this section is [9]. A Markov chain is a stochastic process that satisfies the *Markov property*. The property states that given the state of the chain at one time point, the probability of transitioning to any future state is independent of the sequence of events before the present state.

A distribution of a variable in a Markov chain is a *stationary distribution* if the distribution remains the same after each transition. A *limiting distribution* is a distribution that the Markov chain converges to after an infinite number of transitions. Such distributions are relevant for the second contribution of the thesis, the development of a model describing the stable dynamics of a system. Such stable dynamics can be described using stationary and limiting distributions, which is why Markov chains are fundamental to the thesis.

A Marokv chain with variables at discrete time steps is called a *discrete-time Markov chain* (DTMC). Markov chains with continuous time transitions are called *continuous-time Markov chains* (CTMC), or *Markov processes*, and are not covered in this thesis. There are also *discrete-state* and *continuous-state* Markov chains, referring to Markov chains with discrete and continuous state spaces, respectively. By the finite domain assumption on random variables, this thesis considers only discrete-state Markov chains.

There are variations of Markov chains with different properties. *Time-homogeneous Markov chains* are processes where the transition probabilities are independent of time. *Higher-order Markov chains*, or Markov chains with memory, are processes where the future state depends on the past m states, also called mth-order Markov chains.

To summarize, the type of Markov chains considered in this thesis are discrete-time time-homogeneous Markov chains with discrete state space. Below follows the formal definitions of Markov chains and their properties.

Definition 2.2.1 (Discrete-time Time-homogenous Markov Chain)**.** A *discrete-time Markov chain* (DTMC) is a sequence of random variables $X_0, X_1, X_2, \ldots$ with the Markov property:

$$\Pr(X_{n+1} = x \mid X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = \Pr(X_{n+1} = x \mid X_n = x_n), \quad (2.1)$$

for all $n \in \mathbb{N}$.

A *discrete-time time-homogenous Markov chain* is a discrete-time Markov chain where the transition probabilities are independent of time:

$$\Pr(X_{n+1} = x \mid X_n = y) = \Pr(X_n = x \mid X_{n-1} = y), \quad (2.2)$$

for all $n \in \mathbb{N}$.

Henceforth, Markov chains will refer to discrete-time time-homogeneous Markov chains, since this thesis is concerned with only these types of Markov chains.

A stationary distribution of a Markov chain is a probability distribution that remains unchanged as the chain evolves over time. A limiting distribution is the distribution that the Markov chain converges to as the number of transitions tends to infinity.

Formally, a distribution π on $dom(X)$ is called a stationary distribution of a Markov chain if

$$\pi(x_n) = \sum_{x_{n-1} \in \mathrm{dom}(X)} P(x_n|x_{n-1})\pi(x_{n-1}) \text{ for all } x_n \in \mathrm{dom}(X). \quad (2.3)$$

A distribution π on $\text{dom}(X)$ is called a limiting distribution of a Markov chain if

$$\pi_j = \lim_{n\to\infty} P(X_n = j \mid X_0 = i) \tag{2.4}$$

for all $j \in \text{dom}(X)$, and we have $\sum_{j\in S} \pi_j = 1$. π_j is the probability of being in state j according to the limiting distribution.

If a Markov chain is *irreducible* and *aperiodic*, it has a unique limiting distribution which is also a stationary distribution. An irreducible Markov chain is one where it is possible to reach any state from any other state. A Markov chain is aperiodic if the greatest common divisor of the number of steps required to return to any state is 1. This property ensures that the chain can eventually reach any state at irregular intervals. Otherwise, the limiting distribution may be dependend on the initial state of the chain.

There are analoguous definitions for stationary and limiting distributions of higher-order Markov chains, which is however not covered in this thesis.

2.3 Probabilistic Graphical Models

This section covers various models used to represent probabilistic and causal relationships between variables using graphs. Such models are used both for the first and second contributions of the thesis, and can be seen as variations of the models presented in this section.

The probabilistic graphical models are best understood in the context of Pearl's hierarchy of causality [10], which is a framework that categorizes the types of questions and data analysis that can be addressed at different levels of causal inference. It consists of three levels, often referred to as the "ladder of causation." These levels are *association*, *intervention*, and *counterfactuals*. Each level allows for asking different types of *causal queries*, questions that involve reasoning about the causal relationships between variables.

The first level, association, is concerned with identifying statistical relationships between variables. At this level, we can consider queries of the type "Given that X is observed, what is the probability of observing Y?" The intervention level is concerned with the effects of interventions on a system. An example for an interventional query is "What is the effect of setting X to a particular value on the distribution of Y?"

The highest level of causality, counterfactuals, involves considering hypotheticals and reasoning about alternative realities. An example of a counterfactual query is "Given that X is observed to be a, what would the distribution of Y be if X is set to the value b?". For this thesis, the focus is on the first two levels of causality, and the theory for extending the models to the counterfactual level is left for future work.

The various models covered in this section can be understood in terms of their ability to address questions at different levels of the causal hierarchy. Models for systems not containing agents, hence referred to as non-agentic graphical models, are covered first, followed by models for systems containing agents, referred to as models of

agency. Lastly, a brief background on logical dependencies in relation to the models is provided.

2.3.1 Non-agentic graphical models

The models covered in this section are all variations of *Bayesian Networks* (BNs), which are graphical models that represent the probabilistic relationships between variables. Apart from BNs, this section also provides definitions for *Two-slice Temporal Bayesian Networks* (2TBNs) and *Dynamic Bayesian Networks* (DBNs), which describes system dynamics over time, as well as *Causal Bayesian Networks* (CBNs). For brevity, 2TBNs are sometimes simply referred to as Temporal Bayesian Networks (TBNs).

All models have graphical representations, where directed edges between nodes represent dependencies between variables. The graphs are directed acyclic graphs (DAGs), meaning that all edges are no bidirectional edges and there are no cycles in the graph.

The models differ in what level of the causal hierarchy they can address. BNs, 2TBNs and DBNs are models that can represent associations between variables, while CBNs represent causal dependencies making it possible to answer interventional queries. 2TBNs and DBNs can also be used to represent causal relationships, and are then referred to as 2-slice Causal Temporal Bayesian Networks (2CTBNs) and Causal Dynamic Bayesian Networks (CDBNs), respectively.

Definition 2.3.1 (Bayesian Network [10])**.** A Bayesian Network (BN) over a set of random variables $\mathbf{V}$ is a structure $\mathcal{M} = (\mathcal{G}, \boldsymbol{\theta})$, where $\mathcal{G}$ is a directed acyclic graph (DAG) and $\boldsymbol{\theta}$ is a set of parameters. The graph $\mathcal{G}$ has vertices $\mathbf{V}$ and edges $\mathscr{E}$ that shows dependencies between the variables, satisfying $\Pr(\mathbf{v}; \boldsymbol{\theta}) = \prod_{V \in \mathbf{V}} \Pr(v \mid \mathbf{pa}^V; \theta_V)$. A DAG that satisfies this property is said to be *Markov compatible* with the distribution Pr. Where not ambiguous the parameters can be dropped, and the distribution is written as $\Pr(v \mid \mathbf{pa}^V)$.

Definition 2.3.2 (Two-slice Temporal Bayesian Network (2TBN), [11])**.** A *two-slice temporal Bayesian network* (2TBN) is a model which specifies the transition probabilities between successive time steps for a sequence of variables $\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_T$ using a directed acyclic graph (DAG). Formally, the transition probability from time $t-1$ to t is given as:

$$P(\mathbf{Z}_t | \mathbf{Z}_{t-1}) = \prod_{Z_t \in \mathbf{Z}_t} P(Z_t | \mathbf{Pa}^{Z_t})$$

The model includes *inter-time-slice edges*, linking nodes between adjacent time steps, and *intra-time-slice edges*, connecting nodes within the same time step.

By considering the set of variables at one time step as a variable in a Markov chain, a 2TBN specifies transition probabilities for the Markov chain. The state at time t depends only on the state at time $t-1$, rendering the variables at time t conditionally independent of earlier time steps given the immediate past state.

Definition 2.3.3 (Dynamic Bayesian Network (DBN), [11])**.** A *Dynamic Bayesian Network* (DBN) is a Bayesian network extended over time, which models a sequence of variables $\mathbf{Z}_1, \mathbf{Z}_2, \ldots, \mathbf{Z}_T$. It is a structure $M = (B_1, B_{\rightarrow})$, where B_1 is a *prior Bayesian network*, a BN representing the initial state distribution $P(\mathbf{Z}_1)$, and $B_{\rightarrow}$ is a 2TBN that specifies the transition probabilities between consecutive states.

In this thesis variables that are part of the prior BN are referred to as *prior variables*, while variables at later time steps are referred to as *post variables*.

While the TBN models the transition probabilities between time steps, the DBN also includes the initial state distribution. A DBN can thereby represent an entire Markov chain, starting at time step 1.

The second level of the causal hierarchy, intervention, is addressed by Causal Bayesian Networks (CBNs). There are two types of interventions that can be modeled using CBNs. A *hard intervention* sets variables to specific values. A hard intervention on $\mathbf{Y}$ is denoted $do(\mathbf{Y} = \mathbf{y})$, meaning that $\mathbf{Y}$ is set to the value $\mathbf{y}$. The resulting joint distribution is denoted $\Pr_{\mathbf{y}}(\mathbf{V})$ or $\Pr(\mathbf{V}_{\mathbf{y}})$ [10]. A *soft intervention* changes the distribution of the variables, and are not covered in this thesis.

Definition 2.3.4 (Causal Bayesian Network, [10])**.** A *Causal Bayesian network* (CBN) is a BN $\mathcal{M} = (\mathcal{G}, \boldsymbol{\theta})$ that represents causal relationships. For a CBN, $\mathcal{G}$ is Markov compatible with the interventional distribution $\Pr_{\mathbf{y}}$ for every $\mathbf{Y} \subseteq \mathbf{V}$ and $\mathbf{y} \in dom(\mathbf{Y})$. The conditional probability distribution of a variable V given its parents $\mathbf{Pa}^V$ is given by the following equation:

$$\Pr_{\mathbf{y}}(v|\mathbf{pa}^V) = \begin{cases} 1 & \text{when } V \in \mathbf{Y} \text{ and } v \text{ is consistent with } y, \\ \Pr(v|\mathbf{pa}^V) & \text{when } V \notin \mathbf{Y} \text{ and } \mathbf{pa}^V \text{ is consistent with } y. \end{cases} \tag{2.5}$$

2TBNs and DBNs can equivalently represent causal relationships, and are then referred to as *2-slice Causal Temporal Bayesian Networks* (2CTBNs) and *Causal Dynamic Bayesian Networks* (CDBNs), respectively. These models follow the principles of CBNs, where the DAG is Markov compatible with any interventional distribution.

2.3.2 Graphical models of agency

An extension of Bayesian Networks (BNs) to model decision-making scenarios is *Influence Diagrams* (IDs), which describes systems involving a single agent. For systems that include multiple agents, the analogous model is termed *Multi-Agent Influence Diagrams* (MAIDs). Both IDs and MAIDs extend the basic BN framework by integrating decision nodes and utility nodes [12]. BNs, IDs and MAIDs can be considered as models at the first level of Pearl's causal hierarchy, association.

Advancing to the second level of Pearl's causal hierarchy, *Causal Influence Diagrams* (CIDs) is an ID for which all edges represent causal relationships. *Causal Games* (CGs) are the multi-agent equivalent [6].

Definition 2.3.5 (Multi Agent Influence Diagram (MAID), [10])**.** A *multi-agent influence diagram* (MAID) is a structure $\mathcal{M} = (\mathcal{G}, \theta)$. The graph $\mathcal{G} = (N, \mathbf{V}, \mathcal{E})$

specifies a set of agents $N = \{1, \ldots, n\}$ and a DAG $(\mathbf{V}, \mathcal{E})$ where $\mathbf{V}$ is partitioned into chance variables $\mathbf{X}$, decision variables $\mathbf{D} = \bigcup_{i \in N} \mathbf{D}^i$, and utility variables $\mathbf{U} = \bigcup_{i \in N} \mathbf{U}^i$.

The parameters $\boldsymbol{\theta} = \{\theta_{\mathbf{V}}\}_{\mathbf{V} \notin \mathbf{D}}$ specify the CPDs for all variables that are not decision variables, $\Pr(\mathbf{V}|Pa^{\mathbf{V}}; \theta_{\mathbf{V}})$. For any specification of the CPDs for decision variables induces induces a BN over $\mathbf{V}$.

Definition 2.3.6 (Causal Game (CG), [6])**.** A *causal game* (CG) is a MAID such that all edges in the graph represent causal relationships. Formally, it is a MAID $\mathcal{M} = (\mathcal{G}, \theta)$ such that for any parameterisation of the decision variable CPDs, the induced model is a CBN.

Figure 2.1 shows the relationship between the different types of graphical models presented in this thesis. The levels on the vertical axis corresponds to the steps in Pearl's causal hierarchy [10]. The graphical models can be used for correlational and interventional queries, from bottom to top. The horisontal axis corresponds to the number of agents represented in the model. From left to right, the models can represent zero, one, and multiple agents.

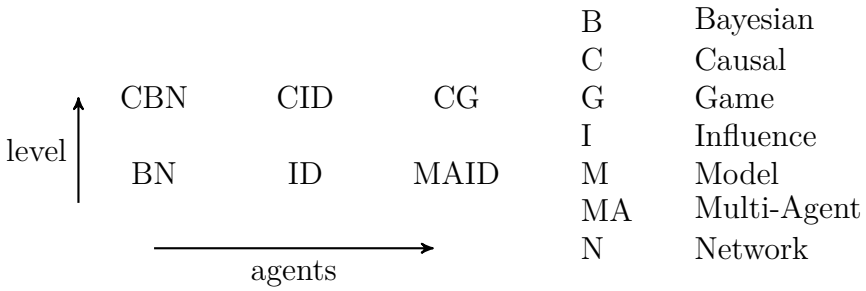


Figure 2.1: Graphical models corresponding to levels in Pearl's causal hierarchy, and zero, one, and multiple agents.

The models in Figure 2.1 can also be extended by *mechanisation*, to represent the CPDs for the random variables graphically. In mechanised models, the variables are divided into *object-level variables* and *mechanism variables*, that are also referred to as *mechanisms*. Each object-level variable has a mechanism variable that represent the CPD of the object-level variable. The mechanism for a decision node determines the policy of the agent, and is called a *decision rule*.

Definition 2.3.7 (Mechanised Multi-Agent Influence Diagram (MAID) [6])**.** A mechanised MAID is a MAID with mechanism variables to represent the CPDs of the object-level variables, as well as a set of *rationality relations*. Rationality relations describe how decision rules are determined for the mechanised MAID. Generally, several different values can be rational (consistent with the rationality relations) for a given decision rule. The rationality relation for a decision rule determines a set of possible values, given the values of the parents of the decision rule.

Given a MAID $\mathcal{M} = (\mathcal{G}, \boldsymbol{\theta})$ and a set of rationality relations $\mathcal{R}$, a mechanised MAID is a structure $\mathrm{m}\mathcal{M} = (\mathrm{m}\mathcal{G}, \boldsymbol{\theta}, \mathcal{R})$. The mechanised graph $\mathrm{m}\mathcal{G} = (N, \mathbf{V} \cup \widetilde{\mathbf{V}}, \mathrm{m}\mathcal{E})$ is a directed (possibly cyclic) graph over $\mathbf{V}$ and $\widetilde{\mathbf{V}}$. $\mathbf{V}$ are the object-level variables,

present in the original MAID, and $\widetilde{\mathbf{V}}$ are the mechanism variables, denoted with a tilde. The graph has the edges $\mathrm{m}\mathcal{E} := \mathcal{E} \cup \{(\widetilde{V}, V)\}_{V \in \mathbf{V}} \cup \mathcal{E}'$ where $\mathcal{E}' \subseteq \bigcup_{D \in \mathbf{D}}((\widetilde{\mathbf{V}} \setminus \widetilde{D}) \times \widetilde{D})$. $\mathbf{D}$ is the set of decision variables.

$\mathcal{R} = \{r_D\}_{D \in \mathbf{D}}$ is a set of rationality relations, where each $r_D \subseteq \mathrm{dom}(\mathbf{Pa}^{\widetilde{D}}) \times \mathrm{dom}(\widetilde{D})$ is a serial relation.

For instance, an agent i might play a *best response* with respect to the other agents' decision rules. Then, for each decision variable $V \in \mathbf{D}^i$, the rationality relation r_V for V is defined as

$$\widetilde{v} \in r_V^{\mathrm{BR}}(\mathbf{pa}_{\widetilde{V}}) \iff \widetilde{v} \in \arg\max_{\widetilde{w} \in \mathrm{dom}(\widetilde{V})} \sum_{U \in U^i} \mathbb{E}_{(\widetilde{w}, \widetilde{-v})}[U]. \tag{2.6}$$

Here, $\widetilde{-v}$ is the vector of all decision rules except for the decision V.

A goal-adaptive agent has the capacity to adapt its actions to achieve specific goals. This can be represented with a feedback loop in a causal model of agency. A simple example is shown in Figure 2.2. The variable D represents a decision, and U represents an utility. The black nodes represent mechanism variables. The arrows going from D to U and from $\tilde{U}$ to $\tilde{D}$ represent a causal relationship. The decision is adaptive as the policy $\tilde{D}$ is updated based on the mechanism for the utility $\tilde{U}$.

To understand how the mechanism edge can represent agency, consider the reasoning of an agent. You might brush your teeth twice a day to avoid decay. You know that if you don't brush your teeth, you will get cavities. This means that you know something about the causal mechanism for how teeth work, teeth are less likely to decay if you brush them. This means that the mechanism for teeth decay influence your behavior. You choose to brush your teeth frequently, which in turn influences how nice your teeth are. In the figure, D represents the choice of brushing, U represents teeth health, $\tilde{D}$ represents the likelihood of brushing, and $\tilde{U}$ represents the CPD for the teeth health given that you brush your teeth.

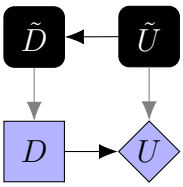


Figure 2.2: Adaptive feedback loop for an agent.

When mechanised models containing agents are used for asking causal queries, it is possible to consider queries before or after the policies of the agents have been set. The former is referred to as *pre-policy queries* and the latter as *post-policy queries* [6]. Equivalently, there are pre-policy and post-policy interventions. For pre-policy interventions, agents can respond and change their policy based on the intervention, while for post-policy interventions agents cannot respond to the intervention. A pre-policy intervention is an intervention on mechanism variables, while a post-policy intervention is an intervention on object-level variables.

In cases where two or more variables are sampled according to the same CPD, it can be represented as a single mechanism variable with several children. This approach has previously been used to model imperfect recall [13]. Each edge from the shared mechanism node to each object-level variable represents an independent draw from the same distribution. An example graph where the mechanism variable $\tilde{D}$ is shared between two decision nodes D_0 and D_1 is shown in Figure 2.3.

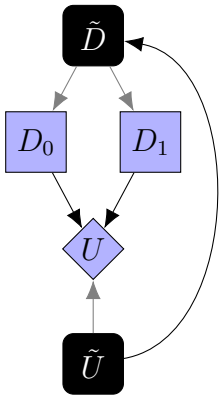


Figure 2.3: A CID containing a mechanism node with two object-level children.

2.3.3 Logical causality

There is a type of dependency between variables that is not causal, but logical. For instance, if one variable represents a proposition A and another represents $\neg A$ (not A), the two variables cannot take the same value. If one variable is true, the other must be false. This is a logical connection between the two variables, and they are said to be *logically connected*, or *logically dependent*. An intervention that sets both variables to the same value would be logically inconsistent.

Such logical dependencies can arise when reasoning about agents. In Soares and Fallenstein [14], it is explained that such non-causal logical constraints arise when an agent's action is logically related to a part of the environment without a causal connection, such as when other agents posess information about the first agent's policy.

For instance, consider an agent playing a one-shot Prisoner's Dilemma with an identical version of itself. Each agent knows that the other is identical. Since the two agents are identical, there is a logical dependence between their actions, as considered in Soares and Fallenstein [14] and MacDermott, Everitt, and Belardinelli [15]. Neither agent directly causes the other agent's action, instead there is simply a requirement that the two agents' actions are the same.

In this thesis, we will model such dependencies using the same framework as for causal dependencies to represent *logical causality*, similar to some previous work in the decision theory literature [14], [16], [17].

3

Mechanised Temporal and Dynamic Bayesian Networks

To represent learning and optimization, this thesis introduces several new models, based on TBNs and DBNs. The chapter begins by introducing mechanised versions of BNs, TBNs, and DBNs, similar to the extension of MAIDs called mechanised MAIDs (Definition 2.3.7). These mechanised models can represent the CPDs of variables graphically. They also build towards introducing mechanism variables that have incoming edges called *update edges*, as covered in the last section of this chapter. Update edges allow for mechanism variables to change dependent on other parts of a system, which is used to model learning and optimization processes.

The definition of TBNs and DBNs can be extended to account for dependencies over n time steps. As such extensions are not central to the thesis, this is however not included in the chapter.

3.1 Mechanisation of BNs, TBNs and DBNs

Mechanised models contain object-level variables and mechanism variables, as introduced in the definition of mechanised MAIDs in the Background chapter (Definition 2.3.7). In this section mechanised versions of BNs, TBNs and DBNs are introduced. The section begins with an example, which is followed by the theoretical framework for mechanisation.

3.1.1 Example: a mechanised DBN

A mechanised DBN can be represented graphically as a DBN with an additional set of edges from the mechanism variables to the object-level variables. Consider Algorithm 1, an algorithm that can be modeled using a DBN with three variables A, B and C. The first three lines describe the initial conditions, while the following lines describe how the variables are updated at each time step.

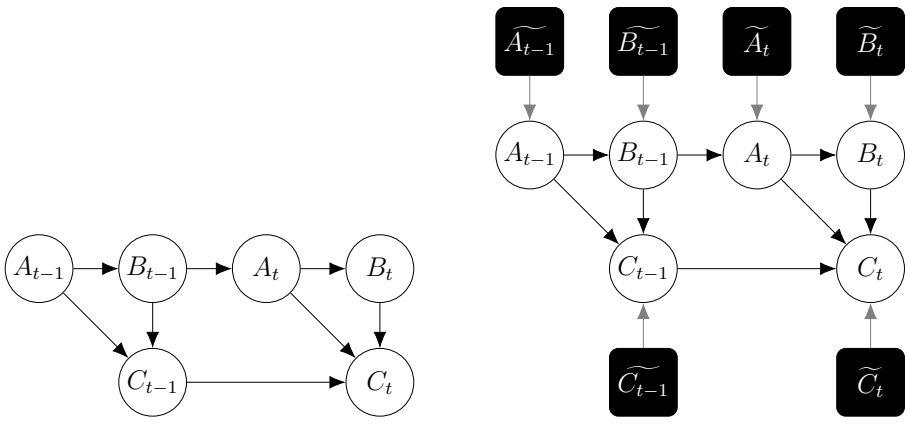
To mechanise the DBN, mechanism nodes are introduced. The variables A, B and C for each time step are then referred to as object-level variables, and the corresponding mechanism variables are denoted $\widetilde{A}$, $\widetilde{B}$ and $\widetilde{C}$. The mechanism variables represent the parameters for the CPDs of the corresponding object-level variables. We have that $\widetilde{A_t} = \widetilde{A_{t-1}} = p$, $\widetilde{B_t} = \widetilde{B_{t-1}} = (q^0, q^1)$, and $\widetilde{C_t} = \widetilde{C_{t-1}} = r$. Figure 3.1 shows the

Algorithm 1 Example: a mechanised DBN.

1: Set A_1 to 0 or 1 with equal probability.
2: Set B_1 to 0 or 1 with equal probability.
3: Set C_1 to 0 or 1 with equal probability.
4: **for** $t = 2, 3, \ldots$ **do**
5: Set A_t to B_{t-1} with probability p, and set to $1 - B_{t-1}$ with probability $1 - p$.
6: **if** $A_t = 0$ **then**
7: Set B_t to 1 with probability q^0 and to 0 with probability $1 - q^0$.
8: Set C_t to 0 with probability 0.5, to B_t with probability r and to $1 - B_t$ with probability $0.5 - r$.
9: **else if** $A_t = 1$ **then**
10: Set B_t to 1 with probability q^1 and to 0 with probability $1 - q^1$.
11: Set C_t to C_{t-1} with probability 0.5, to B_t with probability $0.5 - r$, and to $1 - B_t$ with probability r.
12: **end if**
13: **end for**

graph for the DBN to the left, together with the corresponding mechanised DBN to the right.

Note that the 2TBN that is used to specify the DBN describes transition probabilities that are the same for each time step. This can be represented by using a single mechanism variable with several object-level children, one for each time step, as in Figure 3.2. Mechanism variables usually have a single child, but mechanisms with several children have however previously been used in Fox, MacDermott, Hammond, *et al.* [13].



(a) DBN (b) Mechanised DBN

Figure 3.1: DBN for Algorithm 1 (a), and the corresponding mechanised DBN (b). A DBN is commonly represented graphically at an arbitrary time step, without including the prior BN, which is the case for both (a) and (b). For (b), mechanism nodes are added for each object-level variable.

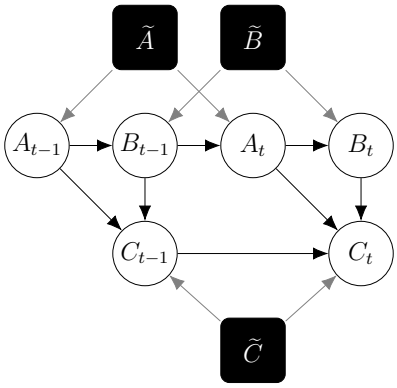


Figure 3.2: Mechanised DBN for the example in Section 3.1.1, with mechanism nodes with object-level children at several time steps.

3.1.2 Theoretical framework for mechanisation

For the modeling in this thesis, three mechanised models are introduced. These are mechanised BNs, mechanised TBNs, and mechanised DBNs.

Given a model $M = (G, (\theta))$, that is either a BN, TBN or DBN, a new parent θ_V is introduced for each variable V in the model. The new parent $\widetilde{V}$ is a mechanism variable, representing the parameters of the CPD of V given its parents in the original model. The new model $mM = (mG, (\theta))$ is then defined by the graph mG where the variables are partitioned into object-level variables and mechanism variables. The object-level variables are the same as in the original model, while the mechanism variables are the new parents.

Formally, the mechanised graph is defined as $\mathrm{m}\mathcal{G} = (\mathbf{V} \cup \widetilde{\mathbf{V}}, \mathrm{m}\mathcal{E})$, and is a directed graph over $\mathbf{V}$ and $\widetilde{\mathbf{V}}$. It has the edges $\mathrm{m}\mathcal{E} := \mathcal{E} \cup \{(\widetilde{V}, V)\}_{V \in \mathbf{V}}$, where $\mathcal{E}$ is the set of edges in the original graph.

Given a BN B, there is a corresponding mechanised BN mB, and given a TBN $B_{\to}$, there is a corresponding mechanised TBN $mB_{\to}$. Given a DBN $M = (B_1, B_{\to})$, the corresponding mechanised DBN is denoted $mM = (mB_1, mB_{\to})$. The mechanised variables introduced for the prior variables in a DBN (variables in the prior BN) are referred to as *prior mechanisms*, while mechanism variables at later time steps are referred to as *post mechanisms*.

While a mechanism node is added for a variable at each time step, the mechanism node can be merged into a single mechanism node with several children, one for each time step, as in Figure 3.2. However, the definition requires that mechanism variables only have object-level children.

3.2 Update edges in mechanised models

Previously in the literature it has been assumed that mechanism variables are not influenced by object-level variables, as in Hammond, Fox, Everitt, *et al.* [6] and Kenton, Kumar, Farquhar, *et al.* [7], which is also the case for the mechanised models

as defined in the previous chapter. This section discusses how and why to relax this constraint.

In a mechanised MAID or CG, the decision rules are described as mechanisms. Consider a decision D in a MAID. In the MAID, it is only represented once, but intuitively the agent taking the decision will learn to adapt its behavior by playing the same game multiple times, and over time learn what decisions are best. Since D is represented as an object-level variable, and the corresponding decision rule is a mechanism, it seems natural that if the learning process is represented with a mechanised model, D should be represented as an object-level variable while $\widetilde{D}$ should be represented as a mechanism variable. In the learning process, $\widetilde{D}$ would then change over time, based on the agent's experience. However, this would require that $\widetilde{D}$ can have ingoing edges, which is not allowed in the current definition of mechanised BNs, TBNs and DBNs.

In practice, it might not be clear whether to consider a variable to be an object-level variable or a mechanism variable. Specifying a variable as a mechanism variable with ingoing update edges instead of an object-level variable is a way of indicating that the mechanism variable should be considered as a potential decision rule for a learning agent.

When an object-level variable influences a mechanism variable, we will represent this with an *update edge* from the object-level variable to the mechanism variable. A node with an outgoing update edge is called an *update node*, corresponding to an *update variable*. TBNs and DBNs containing update edges are referred to as *Temporal Bayesian Networks with Update Edges* (TBN-UEs) and *Dynamic Bayesian Networks with Update Edges* (DBN-UEs), respectively.

3.2.1 Example: a DBN with update edges

Consider a modification of the mechanised DBN example (Section 3.1.1). The mechanism variable $\widetilde{B_t}$ is influenced by B_{t-1}, C_{t-1} and A_{T-1}, as well as the value of the mechanism variable at the previous time step, $\widetilde{B_{t-1}}$, as described in Algorithm 2. The resulting graph of the modified system is shown in Figure 3.3.

$\widetilde{B_t} = (q_t^0, q_t^1)$, with the initialization $q_1^0 = q_1^1 == 0.5$. $\alpha = 0.05$ is the step size for the update. If this update sets q_t^i, where $i \in \{0, 1\}$, to a value that is not allowed, $q_t^i \notin (0, 1)$, then q_t^i is set to the closest value in the interval $(0, 1)$, after which q_t^i is no longer updated.

This system describes an optimization process or RL algorithm, where $\widetilde{B}$ adapts to maximize the expected value of C, given the value of A. This can be seen by how the update to $\widetilde{B}$ functions. If $C_{t-1} = 1$, the update is designed to make the probability for the observed value of B_{t-1} higher at the next time step, and if $C_{t-1} = 0$ the update decreases the probability for the observed B_{t-1}.

Let $C_{t-1} = 1$. If $B_{t-1} = 1$, the update is $\alpha(2C_{t-1} - 1)(2B_{t-1} - 1) = 0.05 \cdot 1 \cdot (2 \cdot 1 - 1) = 0.05$, which increases the probability of $B_t = 1$. If $B_{t-1} = 0$, the update is $\alpha(2C_{t-1} - 1)(2B_{t-1} - 1) = 0.05 \cdot 1 \cdot (2 \cdot 0 - 1) = -0.05$, which instead increases the

Algorithm 2 Example: a mechanised DBN-UE.

Set A_1 to 0 or 1 with equal probability.
Set B_1 to 0 or 1 with equal probability.
Set C_1 to 0 or 1 with equal probability.
for $t = 2, 3, \ldots$ **do**
    Set A_t to B_{t-1} with probability p, and set to $1 - B_{t-1}$ with probability $1 - p$.
    if $A_t = 0$ **then**
        Set B_t to 1 with probability q^0 and to 0 with probability $1 - q^0$.
        Set C_t to 0 with probability 0.5, to B_t with probability r and to $1 - B_t$ with probability $0.5 - r$.
        if $q_t^0 \notin \{0, 1\}$ **then**
            $q_{t+1}^0 = q_t^0 + \alpha(2C_t - 1)(2B_t - 1)$
            if $q_{t+1}^0 \notin [0, 1]$ **then**
                Set q_{t+1}^0 to the closest value in $[0, 1]$.
            end if
        else
            $q_{t+1}^0 = q_t^0$
        end if
    else if $A_t = 1$ **then**
        Set B_t to 1 with probability q^1 and to 0 with probability $1 - q^1$.
        Set C_t to C_{t-1} with probability 0.5, to B_t with probability $0.5 - r$, and to $1 - B_t$ with probability r.
        if $q_t^1 \notin \{0, 1\}$ **then**
            $q_{t+1}^1 = q_t^1 + \alpha(2C_t - 1)(2B_t - 1)$
            if $q_{t+1}^1 \notin [0, 1]$ **then**
                Set q_{t+1}^1 to the closest value in $[0, 1]$.
            end if
        else
            $q_{t+1}^1 = q_t^1$
        end if
    end if
end for

probability of $B_t = 0$. The variable C can thereby be interpreted as a reward signal or utility. The calculations are analogous for $C_{t-1} = 0$.

Note that the system in Figure 3.3 does not correspond to a normal DBN, as the one shown in Figure 3.1. While the systems have a similar structure for the object-level variables, there are additional dependencies in the DBN-UE that are not present in the DBN. For example, B_{t-1} and C_{t-1} influence B at every following time step through the update edges, a dependency that would not be present in a graph whithout mechanism variables and update edges. In fact, a requirement for the formal definition of models with update edges is that update edges can only be introduced when necessary. If an update edge only influences a mechanism variable at a single time step, this dependency should instead be modeled as an influence on the object-level child.

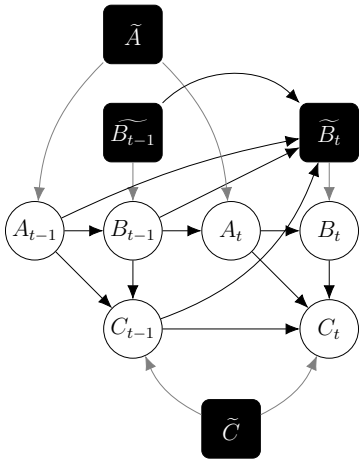


Figure 3.3: DBN-UE for the example in Section 3.2.1.

3.2.2 Theoretical framework for update edges

The main requirement for update edges is that they can only be introduced when it is necessary to do so. This is because update edges are supposed to model an optimization process, where a mechanism variable is adapted at each step to achieve some objective. If a mechanism variable is only influenced by variables in the current and last time steps, the causal dependency can be represented by a path through object-level variables, and the introduction of update edges is unnecessary. This means that an update edge is not allowed when it only affects a mechansm variable at a single time step.

Definition 3.2.1 (Temporal Bayesian Network with Update Edges (TBN-UE))**.** An *Temporal Bayesian Network with Update Edges* (TBN-UE) is a mechanised TBN which contains mechanism variables with ingoing edges, called update edges. A TBN-UE is always a mechanised model.

Mechanism variables that do not have incoming update edges are required to remain constant over time, and can therefore be represented by a single mechanism variable with several object-level children, one for each time step.

An extra requirement is that the update edges are allowed only when there is a direct dependence of the mechanism variable on the update variable at every future time-step. This is to ensure that update edges are not introduced to model dependencies that could accurately be represented by influence on object-level variables.

Definition 3.2.2 (Dynamic Bayesian Network with Update Edges (DBN-UE))**.** A *Dynamic Bayesian Network with Update Edges* (DBN-UE) is a mechanised DBN for which the mechanised TBN is a TBN-UE. The prior network in a DBN-UE is a mechanised BN, which can but does not necessarily, contain update edges.

4

Temporal Abstraction

This chapter covers the second contribution of this thesis. To analyse stable dynamics of a system, examine learned behavior, as well as reason about equilibria, Temporally Abstracted Models (TAMs) are introduced, which can be derived from TBN-UEs and DBN-UEs. The transition to a TAM is referred to as *temporal abstraction*.

A TAM is a mechanised model, that can be considered a variant of a mechanised DBN, describing a system after the learning process for each learning agent has converged. The reason that the model is considered a 'temporal abstraction' is that it does not describe the system at a specific state with a specific outcome for the process. Instead it describes potential outcomes from the learning process, called *stationary outcomes*, corresponding to possible stable dynamics that the system can converge to. Additionally, if a TAM is derived from a causal model (CTBN or CDBN), it can be used for interventional queries.

The TAM, describing the system after convergence, has mechanism variables for which possible values are determined by a set of *stationarity relations*, similar to the rationality relations for mechanised MAIDs (Definition 2.3.7).

This chapter begins with analysing the example in Section 3.2.1, to introduce the concept of a TAM. The chapter then continues with the formal framework for temporal abstraction, and concludes with a section covering the relation between a TAM and the graphical models of agency introduced in the Background chapter (Section 2.3.2).

4.1 Example: a Temporally Abstracted Model

To illustrate the temporal abstraction, this section shows how to reason about the stable dynamics of the example considered in Section 3.2.1. This illustration is devided into four parts, starting with analysing possible stationary distributions for the system, given values of the mechanism variables. The illustration continues with introducing a latent confounder that is required for representing the system using a TAM. Then the dependencies between mechanism variables are described and graphical representations are provided.

4.1.1 Analysing outcomes

For a TAM, that is supposed to model dynamics after the optimization process has converged, we consider stationary distributions for the system after each mechanism variable with ingoing update edges has converged. This is possible since a DBN can be interpreted as describing a Markov chain.

For the example in Section 3.2.1, we can examine the resulting system dynamics, given different combinations of mechanism variables. If $A_t = 0$, C_t is set to B_t with probability r and to $1 - B_t$ with probability $0.5 - r$. If $r > 0.25$, this means that $B_t = 1$ is optimal, while for $r < 0.25$, $B_t = 0$ is optimal. Analogously, if $A_t = 1$, $B_t = 0$ is optimal for $r > 0.25$, and $B_t = 1$ is optimal for $r < 0.25$. For $r = 0.25$, B does not affect the value of C.

This allows for analysing what the system can converge to, given different values for the mechanism variables. First consider the case $r > 0.25$. Then the optimal value for B_t is to set it to $1 - A_t$, which we can assume that the algorithm will learn (there is a low probability of converging to a suboptimal policy, which is ignored for simplicity). This allows for calculating the stationary distribution for A. Let $P(A = 1)$ and $P(A = 0) = 1 - P(A = 1)$ represent a stationary distribution for A. Since B is set to $1 - A$, the chance of A_t changing value between time steps is p, and the chance of A_t staying the same is $1 - p$. This gives the following equation for the stationary distribution of A:

$$P(A = 1) = pP(A = 0) + (1 - p)P(A = 1) \Rightarrow \tag{4.1}$$
$$pP(A = 1) = pP(A = 0) \tag{4.2}$$

For $p \neq 0$, this means that $P(A = 1) = 1/2$ and $P(A = 0) = 1/2$. For $p = 0$, any distribution is stationary, as long as the total probability is 1. In this case, A is constant. For the following analysis, $p \neq 0$ is assumed.

Now consider the stationary distribution for C. It can be represented as a conditional distribution given A, $P(C = 1|A)$. If $A = 1$ at one time step, A was 0 at the previous time step with probability p, and 1 with probability $1 - p$. B can be assumed to take the optimal value, which results in C being set to 1 with probability r for both $A = 1$ and $A = 0$. This gives the following equations:

$$P(C = 1|\ A = 1) = 0.5(pP(C = 1|\ A = 0) + (1 - p)P(C = 1|\ A = 1)) + r \tag{4.3}$$
$$P(C = 1|\ A = 0) = r \tag{4.4}$$

Setting $P(C = 1|\ A = 0) = r$ in the first equation gives:

$$P(C = 1|\ A = 1) = 0.5(pr + (1 - p)P(C = 1|\ A = 1)) + r \Rightarrow \quad (4.5)$$
$$P(C = 1|\ A = 1)(1 - 0.5(1 - p)) = 0.5pr + r \Rightarrow \quad (4.6)$$
$$P(C = 1|\ A = 1) = \frac{r(0.5p + 1)}{0.5 + 0.5p} = \frac{r(0.5p + 0.5)}{0.5p + 0.5} + \frac{0.5r}{0.5p + 0.5} = r + \frac{r}{1 + p} \quad (4.7)$$

Note that $P(C = 1|\ do(A = 0)) = P(C = 1|A = 0) = r$, since for $A = 0$ the previous value of C has no effect on the current value. However, since setting A to 1 at one time step does not affect it's value at previous time steps, $P(C = 1|\ do(A = 1))$ is not equal to $P(C = 1|\ A = 1)$. Instead, it is given by the following:

$$P(C = 1|\ do(A = 1)) = \quad (4.8)$$
$$= 0.5(P(A = 1)P(C = 1|\ A = 1) + P(A = 0)P(C = 1|\ A = 0)) + r = \quad (4.9)$$
$$= 0.5(0.5P(C = 1|\ A = 1) + 0.5P(C = 1|\ A = 0)) + r = \quad (4.10)$$
$$= 0.25(r + \frac{r}{1 + p}) + 0.25r + r = 1.5r + \frac{r}{4(1 + p)} \quad (4.11)$$

Now consider the case $r < 0.25$. The optimal value for B is then equal to A, which is learned by the algorithm (again, the small probability of converging to a subuptimal policy is ignored). If $A = 1$ at one time step, A was 1 at the previous time step with probability p, and 0 with probability $1 - p$. The stationary distribution for A is given by:

$$P(A = 1) = p\Pr(A = 1) + (1 - p)P(A = 0) \Rightarrow \quad (4.12)$$
$$(1 - p)P(A = 1) = (1 - p)P(A = 0) \quad (4.13)$$

As for $r > 0.25$, this means that $P(A = 1) = 1/2$ and $P(A = 0) = 1/2$ if $p \neq 1$. If $p = 1$, all distributions are stationary as long as the total probability is 1, and A is constant. The conditional distribution for C is given by:

$$P(C = 1|\ A = 1) = 0.5(pP(C = 1|\ A = 1) + (1 - p)P(C = 1|\ A = 0)) + 0.5 - r \quad (4.14)$$
$$P(C = 1|\ A = 0) = 0.5 - r \quad (4.15)$$

Setting $P(C = 1|\ A = 0) = 0.5 - r$ in the first equation gives:

$$P(C = 1|\ A = 1) = 0.5(pP(C = 1|\ A = 1) + (1 - p)(0.5 - r)) + 0.5 - r \Rightarrow \quad (4.16)$$
$$P(C = 1|\ A = 1)(1 - 0.5p) = 0.5(1 - p)(0.5 - r) + 0.5 - r \Rightarrow \quad (4.17)$$
$$P(C = 1|\ A = 1) = \frac{(0.5 - r)(1 - 0.5p + 0.5)}{1 - 0.5p} = 0.5 - r + \frac{0.5 - r}{2 - p} \quad (4.18)$$

As for the case $r > 0.25$, $P(C = 1|\ do(A = 0)) = P(C = 1|\ A = 0) = r$, while $P(C = 1|\ do(A = 1)) \neq P(C = 1|\ A = 1)$. The interventional distribution is however not provided here.

There are three different special cases that stand out. First, if $r = 0.25$, there is no optimal value for B. The algorithm will however converge at some point to a mechanism $\widetilde{B}$. For the cases where $r > 0.25$ and $p = 0$, or $r < 0.25$ and $p = 1$, A is constant. For these cases, if one of the parameters q^0 or q^1 converges, the other will never be updated, since from that point A only takes one value and there is no opportunity for learning a behavior that is optimal for the other value of A.

4.1.2 Introducing a latent confounder

As mentioned in the previous section, an interventional distribution for C given an intervention on A is not necessarily equal to the observational distribution when considering the stationary distribution of the system. To account for this, a latent confounder E is introduced. E is a variable that influences both A and C, but is not directly observed.

For simplicity, consider values for the parameters p and r that do not correspond to any of the special cases mentioned in the previous section. This means that the stationary distribution for A is given by A taking the values 0 and 1 with equal probability 0.5. Let A be dependent on E as $A = E$ with probability 1. The stationary distribution for C can be expressed as following:

- If $E = 0$: If no intervention is made on A, this implies $A = 0$, which means that C is set to 0 with probability 0.5, to $B = 1$ with probability r, and to $1 - B = 0$ with probability $0.5 - r$.

- If $E = 1$: If no intervention is made on A, this implies $A = 1$, which means that C is set to the previous value of C with probability 0.5, to $B = 0$ with probability $0.5 - r$, and to $1 - B = 1$ with probability r.

Setting C to the previous value corresponds to setting C to 1 with probability $2\frac{r}{1+p}$ for $r > 0.25$, and with probability $2\frac{0.5-r}{2-p}$ for $r < 0.25$, in order to be constistent with the stationary distribution for C given $A = 1$. Adding the latent confounder E allows for letting the causal dependence between A and C be correct, even when reasoning about the stationary distribution after convergence. An intervention can be made on A, which is then no longer dependent on E. For instance, if $E = 0$, the intervention $\text{do}(A = 1)$ should be interpreted as setting A to 1 while examining the distribution for C as if A was 0. The interventional distribution for C given E can be derived as following:

$$P(C = 1|\ E = 0, do(A = 1)) = \tag{4.19}$$
$$= 0.5(pP(C = 1|\ A = 1) + (1 - p)P(C = 1|\ A = 0)) + r = \tag{4.20}$$
$$= 0.5(p(r + \frac{r}{1 + p}) + (1 - p)r) + r = \tag{4.21}$$
$$= 1.5r + \frac{pr}{2(1 + p)} \tag{4.22}$$

The interventional distribution should be interpreted as setting C to its previous value with probability 0.5, which is 1 with probability $r + \frac{pr}{1+p}$, and set otherwise set it to $1 - B = 1$ with probability r.

Note that while in this analysis, the cases $r > 0.25$ or $r < 0.25$ have been considered when deriving the distributions. However, the calculations can be done for any combination of q^0 and q^1.

4.1.3 Mechanism dependencies

The stationary distribution for the system can be represented by a graphical model, as shown in Figure 4.1. It shows a specific outcome, corresponding to a converged value for $\widetilde{B}$. After $\widetilde{B}$ has converged, the stationary distribution is represented using a mechanised DBN, where the first time step is an arbitrary time step T that is large enough that $\widetilde{B}$ has converged. At this time step the latent variable E is introduced, which influences the object-level variables A_T and C_T, as described in the previous section. The joint distribution of A_T, B_T and C_T for the mechanised DBN is a stationary distribution for the Markov chain.

Note that $\widetilde{B}$ will converge to different values depending on the value of r. Such dependencies are considered in this section. A TAM describes every possible outcome for the system, where the DBN in Figure 4.1 describes the graph for a specific outcome.

A TAM has a similar structure to a mechanised DBN, with mechanism variables and object-level variables. As for a DBN, variables at the first time step are called prior variables, and variables at later time steps are called post variables, with mechanism variables being prior mechanisms and post mechanisms respectively. For a TAM there can however be edges between mechanism variables that are not present in a mechanised DBN.

The TAM for this example is shown in Figure 4.2, in subfigure (b). $\widetilde{A}$ and $\widetilde{C}$ are the same as in the orginal DBN-UE, shown in the same figure as subfigure (a). These post mechanisms affects possible stationary distributions for the system. The prior mechanisms $\widetilde{A_T}'$, $\widetilde{C_T}'$, $\widetilde{B_T}'$ and $\widetilde{E}'$ are determined by the post mechanisms, and any stationary distribution for the system is represented by the prior mechanisms.

Assume that p and r have values that do not correspond to any of the special cases. Then for $r > 0$, $\widetilde{B} = (q^0, q^1) = (1, 0)$, and for $r < 0$, $\widetilde{B} = (q^0, q^1) = (0, 1)$. The prior mechanisms for A_T and C_T should, in combination with E, represent

the stationary distribution for the system. As described in the previous section, A is set to E with probability 1, which can be represented with a prior mechanism $\widetilde{A_T}' = P(A_T = E) = 1$. The dependence of C on A and E is given by the r and by the expected value of a previous state of C given the value of E. Let the parameterisation be $\widetilde{C_T}' = (r, r')$, where $r' = 2\frac{r}{1+p}$ for $q^1 = 0$ and $q^0 = 1$, and $r' = 2\frac{0.5-r}{2-p}$ for $q^0 = 0$ and $q^1 = 1$. The value of r' is not calculated here for other combinations of q^0 and q^1, but r' is in general dependent on p, q^0 and q^1 (or equivalently, $\widetilde{C_T}'$ is dependent on $\widetilde{A_T}'$ and $\widetilde{B}'$).

Now, consider the special cases. For $r > 0.25$ and $p = 0$, or $r < 0.25$ and $p = 1$, A is either constantly 0 or constantly 1. Note that the value of A is is mutually dependent on the value for $\widetilde{B}$. If q^0 or q^1 has converged, that determines the value of A, but A determines which of the parameters q^0 or q^1 converges first. Neither comes before the other. Note that while any distribution over A is stationary for these cases, only two are associated with converged values for either q^0 or q^1. If one has converged, the stationary distribution for A is either $P(A = 1) = 0$ or $P(A = 1) = 1$, which should be represented by $P(E = 1) = 0$ or $P(E = 1) = 1$, since A is set to E with probability 1. This means that $\widetilde{E}' = P(E = 1)$ is either 1 or 0.

For $r = 0.25$, it is imossible to determine what $\widetilde{B}$ converges to, but if $p = 0$ or $p = 1$, it is not required that both q^0 and q^1 converge, since A could become constant.

Since A is simply always set to E, the value of p, r or the converged value for $\widetilde{B}$ has no effect on the prior mechanism $\widetilde{A_T}'$. Instead, the stationary distribution for A is represented by $\widetilde{E}'$. There is no reason to allow $\widetilde{A_T}'$ to have different values, and so there are no ingoing or outgoing edges from $\widetilde{A_T}'$. Note that this was a modeling choice to make the TAM simpler. Other parameterisations are possible, for which there might be ingoing and outgoing edges from $\widetilde{A_T}'$. All possible ways to construct the TAM necessarily needs to include the latent confounder E.

Note that for two of the special cases, $\widetilde{B}$ is determined simultaneously with $\widetilde{E}'$, and there is a dependence in both directions. Intuitively, this can be understood as the agent affecting the environment, and the environment affecting the agent. Graphically, this is represented by directed edges in both directions between $\widetilde{B}$ and $\widetilde{E}'$.

$\widetilde{E}'$ is determined together with other prior mechanisms and $\widetilde{B}$. There are therefore directed edges in both directions between $\widetilde{E}'$ and the other prior mechanisms, except for any prior mechanism for which the object-level child is not directly dependent on E, as $\widetilde{B_T}'$ in this example.

Further explanation for dependencies between mechanism variables are given in Section 4.2.2.2.

4.1.4 Graphical representation

The graph for the specific outcome analyzed in this example is shown in Figure 4.1. It shows that the outcome is a DBN, where the joint distribution over the object-level

variables in the prior BN, except for the variable E, is a stationary distribution for the Markov chain.

Figure 4.2 shows TAM including dependencies between the mechanism variables. Note that for the TAM, as well as the specific outcome in Figure 4.1, the first time step is T. This is an arbitrary time step that is large enough that $\widetilde{B}$ has converged.

For A and C, the prior mechanisms are separated from the post mechanisms, while $\widetilde{B}$ is the mechanism parent for B at both time steps. This is because at time step T, both A and C have different sets of parents compared to future time steps (both have the parent E), while B has the same parents and therefore has the same distribution at all time steps. Not separating the prior mechanism from the post mechanisms for a variable is possible when the object-level variable has the same set of object-level parents at time step E. Any variable with E as a parent will not have a corresponding mechanism that can be merged with other mechanism variables. If more time steps were included in the graph, the post mechanism variables for A and C would have been the mechanisms for all time steps after T.

A simpler graphical representation can be constructed by simply not representing all prior mechanisms. This makes the model easier to represent and reason about. This is shown in Figure 4.3. The mechanism parent of a prior object-level variable is replaced with the mechanism parent of the corresponding post object-level variable, which is then shared over all time steps. While being technically incorrect, the graph still shows some of the more important features of the model, as the mutual dependence between E and $\widetilde{B}$ and that $\widetilde{C}$ influences $\widetilde{B}$.

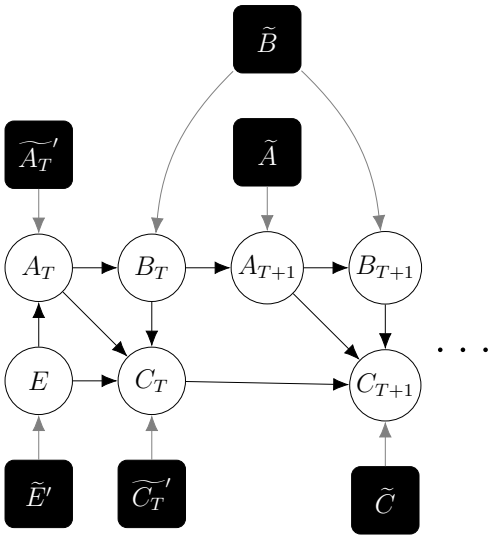
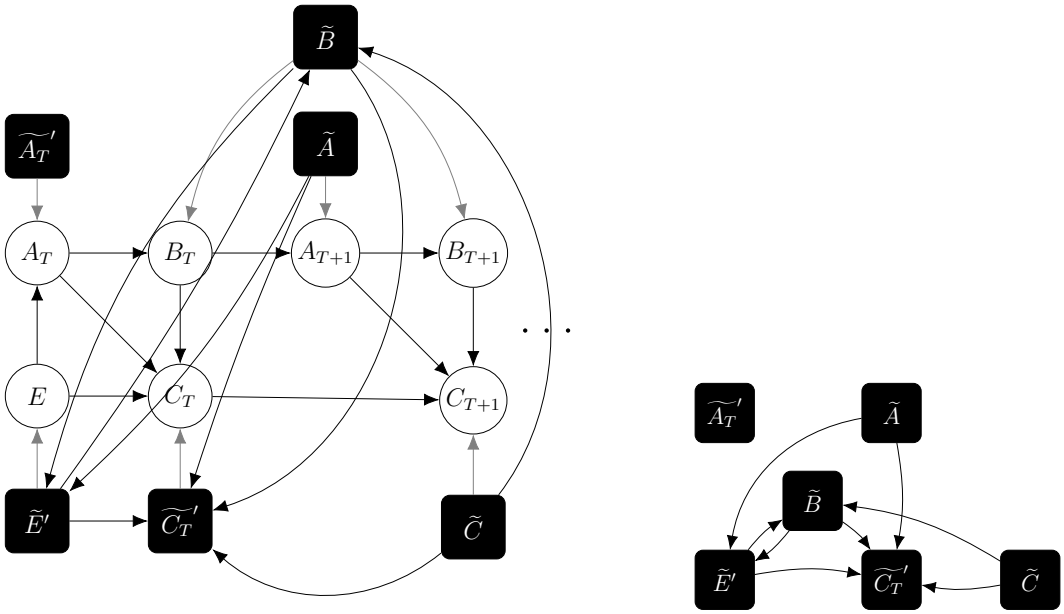


Figure 4.1: Graph for a specific outcome of the TAM for the example in Section 4.1.



(a) TAM, merged mechanism for B (b) Mechanism subgraph

Figure 4.2: TAM after merging the mechanism variables (a), and the mechanism subgraph for the same model (b).

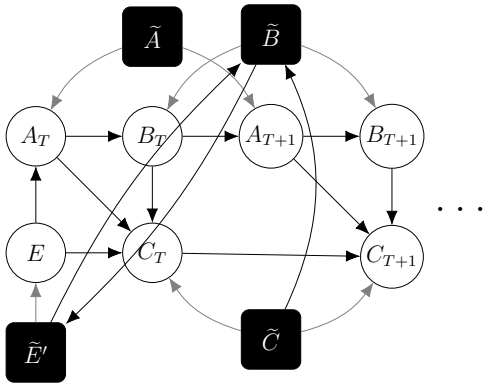


Figure 4.3: TAM for the example without prior represented, except for $\widetilde{E}$.

4.2 Theoretical framework for temporal abstraction

The theoretical framework is devided into three parts. First, the temporally abstracted model is defined. It is a mechanised model for which some mechanism variables are determined by stationarity relations, which are defined and explained in the second part. Lastly, the framework is completed by defining how to answer queries for a TAM, including conditional and interventional queries.

Note that while the first part covers the structure for TAMs, the second part is needed to justify the structure. An extensive explanation of the dependencies between mechanism variables are provided in Section 4.2.2.2.

4.2.1 Temporally Abstracted Model

The TAM has a similar structure to a mechanised DBN. The most notable difference is that mechanism variables in a TAM can be dependent on each other in a way that is not allowed for a mechanised DBN. Some mechanism variables are left unspecified, and possible values are determined by a set of stationarity relations, analoguous to how decision rules for mechanised MAIDs are determined by rationality relations. Any specification of the mechanism variables that satisfies the stationarity relations induces a mechanised DBN.

A TAM is derived from a TBN-UE or DBN-UE, the original model of a system. It models the system after mechanism variables with ingoing update variables have converged. Given the original model, mechanism variables with update edges can often converge to several different values. A TAM represents all possible values that the mechanisms can converge to. A system can also have several stationary distributions associated with the converged mechanism variables.

Consider the example in the previous section. $\widetilde{B}$ corresponds to the converged value of the mechanism with update edges in the original DBN-UE. Converged mechanism variables are hence called *dynamic mechanisms*, and are necessarily post mechanism variables. Note that $\widetilde{B}$ could be considered the mechanism for B_T as well, and so it is a dynamic mechanism even though it is essentially both a prior and post mechanism variable, as Figure 4.2 shows.

In Figure 4.2, we see that both prior mechanisms and dynamic mechanisms have ingoing edges from other mechanisms. A mechanism's parents determines possible values for the mechanism through what is called a stationarity relation. The stationarity relations are derived from the original model, and the formal definition is given in Section 4.2.2.

Definition 4.2.1 (Temporally Abstracted Model (TAM))**.** Given a TBN-UE or DBN-UE $\mathrm{m}\mathcal{M}_o$, where the subscript o indicates that the model is the original model that the TAM is derived from, a TAM is a mathematical structure $\mathrm{m}\mathcal{M} = (\mathrm{m}G, \boldsymbol{\theta}, \mathcal{R})$. The mechanised graph $\mathrm{m}\mathcal{G} = (N, \mathbf{V} \cup \widetilde{\mathbf{V}}, \mathrm{m}\mathcal{E})$ is a directed graph over $\mathbf{V}$ and $\widetilde{\mathbf{V}}$ with edges $\mathrm{m}\mathcal{E} := \mathcal{E} \cup \{(\widetilde{V}, V)\}_{V \in \mathbf{V}} \cup \mathcal{E}'$, and $\mathcal{E}' \subseteq (\widetilde{\mathbf{V}} \times (\widetilde{\mathbf{W}}' \cup \widetilde{\mathbf{D}}))$. The set $\widetilde{\mathbf{V}}$ is the set of all mechanism variables, $\widetilde{\mathbf{W}}'$ is the set of all prior mechanism variables, and $\widetilde{\mathbf{D}}$ is the set of dynamic mechanisms. $\boldsymbol{\theta}$ are the parameters for the CPDs corresponding to post mechanism variables except dynamic mechanisms, which are determined exactly in the original model as mechanism variables without ingoing update edges.

At each time step $T, T+1, T+2\ldots$ the TAM has the same object-level variables as $\mathrm{m}\mathcal{M}_o$. At each time step $T+1, T+2, T+3\ldots$ the TAM has the same mechanism variables (post mechanism variables) as $\mathrm{m}\mathcal{M}_o$, except for dynamic mechanisms. At time step T, the TAM can also have two extra variables, E and its mechanism $\widetilde{E}'$. E has no object-level parents.

$\mathcal{R} = \{r_V\}_{V \in \widetilde{\mathbf{W}}' \cup \widetilde{\mathbf{D}}}$ is a set of stationarity relations, given by the original model $\mathrm{m}\mathcal{M}_o$. It can be written as $\mathcal{R} = \mathcal{R}(\mathrm{m}\mathcal{M}_o)$. Each $r_V \subseteq \mathrm{dom}(\mathbf{Pa}^{\widetilde{V}}) \times \mathrm{dom}(\widetilde{V})$ is a serial relation, specifying possible values for the mechanism $\widetilde{V}$. If an outcome for the set of

mechanisms with ingoing edges in the TAM, $\widetilde{\mathbf{W}}' \cup \widetilde{\mathbf{D}}$, satisfies the set of stationarity relations $\mathcal{R}$, we say that the outcome $\tilde{\mathbf{w}}' \cup \tilde{\mathbf{d}}$ is a *stationary outcome* of the TAM.

We can consider variations of the TAM. For instance, the dependence between $\widetilde{E}'$ and other prior mechanism variables can be bidirectional, since the mechanism variables are determined together. However, we could let $\widetilde{E}'$ be determined first, which would then specify possible values for the other mechanism variables. The requirement for the TAM is that it represents possible stationary outcomes, but this requirement does not specify exactly what mechanism dependencies are allowed.

4.2.2 Stationarity relations

The possible values for the mechanism variables in a TAM are determined by a set of stationarity relations. As in the example in the previous section, the TAM should describe the system for which the learning has converged, and any outcome for the prior mechanism variables should induce a stationary distribution for the system with respect to the transition probabilities given by the original model. Possible values for each mechanism with ingoing edges from other mechanisms in the TAM are determined by its parents. To represent this, a relation is introduced for each prior mechanism and dynamic mechanism, specifying possible values given its parents.

Definition 4.2.2 (Stationarity relation)**.** Each prior mechanism and dynamic mechanism in a TAM has a corresponding stationarity relation. These stationarity relations have three requirements. The first is that any set of mechanism variables allowed by the stationarity relations must induce a stationary distribution $p(\mathbf{Z_T})$ over the prior object-level variables in the TAM, with the exception of E. The stationary distribution is with respect to the transition probabilities given by the original model.

The second requirement is that any outcome for a dynamic mechanism in the TAM that satisfies the stationarity relations, specifies a value that don't change at future time steps. Intuitively, this describes dynamics for the system where update edges no longer affect the mechanisms with ingoing update edges.

The third requirement is that any dependencies between object-level variables are the same for each time step, except for dependencies on the variable E. E is introduced to make this possible, as it represents dependencies between prior object-level variables that would otherwise be missing from the TAM (if there is no such dependency, E is not introduced). E can have an outgoing edge to any prior object-level variable in the TAM which would have had a parent at the time step $T-1$, had variables at time step $T-1$ been included in the TAM. It's mechanism, $\widetilde{E}'$, is logically dependent on the other prior mechanism variables, since they are specified together to represent the stationary distribution p. Graphically, this is represented by edges between $\widetilde{E}'$ and other prior mechanisms.

4.2.2.1 Explanation for the requirements

The TAM is supposed to describe goal-adaptive agents, agents that adapt to their environment to achieve a goal. Dynamic mechanisms in a TAM should correspond

to decision rules, as long as the original TBN-UE or DBN-UE describes a system with one or more learning agents (which is not necessarily the case).

For a mechanism vairable with ingoing update edges that has not converged, the agent has not yet fully adapted. It would change between time steps, which would make it difficult to determine that the mechanism correspond to a learning agent. If the mechanism changes a lot between time steps, it seems more appropriate to model it as an object-level variable. Unless the mechanism is constant or changes only slightly, it is hard to determine that the mechanism is changing to achieve some goal.

Instead of setting an arbitrary limit on how much the mechanism can vary while still being considered a decision rule, it is required that the mechanism variable remains constant. While this is a strong requirement, this approach still provides qualitative insights for systems where the mechanism changes only slightly between time steps.

Consider a person who frequently changes their mind significantly. It would be difficult to model this person as an agent since they do not act consistently and cannot be described as optimizing for the same goal over time. The person is effectually not optimizing for anything, and it would be more appropriate to model the person as a part of the environment (though this might feel unfair to the person in question).

Note that an agent necessarily needs to adapt the behavior until the behavior converges, and should start adapting the behavior again if the environment changes. In summary, an entity is less agentic if the behavior changes too much between time steps, but also is not agentic if the behavior does not adapt at all. The speed of behavior change is crucial for something to be considered an agent.

For a mechanism variable that does not converge, it can be modeled as an object-level variable with a corresponding mechanism variable that remains constant across all time steps. This alternative model for the same system meets the assumption of converging mechanism variables, allowing the system to be abstracted to a TAM. Specifying a mechanism with update edges as a mechanism variable in a TBN-UE can be interpreted as indicating that the TAM should consider possible values for the variable to converge to. Dynamic mechanisms have ingoing edges from any other mechanism variable which affects what values the dynamic mechanism can take.

The first requirement in the definition of stationarity relations is that the distribution p is a stationary distribution under the transition probabilities of the original model. If there are some stable dynamics that the agent learns to optimize for, it can naturally be described using a stationary distribution. For instance, consider a variable that cycles between three values. The optimal policy for an agent that does not directly observe the value, but for which the value is still relevant to the agent's goals, the agent's optimal policy will be to act as if the value is drawn from the uniform distribution over the three values, which is also the stationary distribution. Equivalently, the dynamic mechanisms in a TAM can be predicted to optimize for some goals, if they correspond to learning agents, with respect to a stationary distribution over the prior object-level variables.

Note that it would be possible to relaxe the stationary distribution requirements, and only have the second and third requirements. This would allow for a more general model with interesting dynamics, but would also be harder to interpret. Such a relaxation is left for future work.

4.2.2.2 Explanation for each type of mechanism dependence

To understand temporal abstraction and the stationarity relations, let us analyse each type of dependence between mechanism variables one by one. The example in the previous section is used as a reference. The right graph in Figure 4.2 contains each type of dependence. The types of dependencies can be divided into the following categories:

1. From post mechanisms to dynamic mechanisms
2. From post mechanisms except dynamic mechanisms to prior mechanisms
3. Between prior mechanisms
4. Between prior mechanisms and dynamic mechanisms
5. Between dynamic mechanisms

First consider the edge $\widetilde{C}$ to $\widetilde{B}$, which is an edge from a post mechanism to the only dynamic mechanism. The post mechanisms are present at each time step in the original model. If there is a path from a mechanism variables without ingoing update edges, to a mechanism variable with ingoing update edges, this can (but do not necessarily) result in an edge from the corresponding post mechanism to the dynamic mechanism in the TAM.

The edges from $\widetilde{C}$ to $\widetilde{C_T}'$, from $\widetilde{A}$ to $\widetilde{E}'$, and from $\widetilde{A}$ to $\widetilde{C_T}'$ are from post mechanisms to prior mechanisms. The post mechanisms naturally affect possible stationary distributions for their corresponding object-level variables, which is the reason for this dependence.

The third category are for edges between prior mechanisms, which in Figure 4.2 are the edges between $\widetilde{E}'$, and $\widetilde{C_T}'$. $\widetilde{E}'$ is determined together with prior mechanisms which has object-level children with E as a parent, since E is introduced for modeling dependencies between the variables that are not otherwised represented by the edges between the object-level variables. If there is an object-level path (not through update edges) from an object-level variable V to another object-level variable W, potentially over several time steps, this generally induces a dependency from $\widetilde{V_T}'$ to $\widetilde{W_T}'$. The stationary distribution of W is dependent on the stationary distribution of V. The direction of the dependency between the object-level variables also determines the direction of the dependency between the prior mechanism variables. In Figure 4.2, there is an edge from $\widetilde{E}'$ to $\widetilde{C_T}'$ due to this dependency, since $\widetilde{E}'$ can be interpreted as representing the stationary distribution of A. Since C does not affect A except through $\widetilde{B}$, there is no edge from $\widetilde{C_T}'$ to $\widetilde{E}'$.

Dynamic mechanisms and prior mechanisms are also determined together. Dynamic

mechanisms are generally post mechanisms, since at step T the dynamic mechanism might not fully determine the CPD, if the object-level child at time step T is dependent on E. If it would have been necessary to have E as a parent to B_T, the mechanism for B_T would need to be separated as a separate mechanism from $\widetilde{B}$, similar to $\widetilde{A_T}'$ and $\widetilde{C_T}'$.

The last category is edges between dynamic mechanisms. These dependencies are determined equivalently to dependencies between prior mechanisms, or the dependencies between dynamic mechanisms and prior mechanisms.

Note that the graphical dependencies in the original model might not fully determine dependencies in the TAM. Consider for instance a mechanism with ingoing update edges that, while being influenced by update variables, actually can only converge to a single value, independently of the learning environment. This would be a case where the dynamic mechanism does not have ingoing edges in the TAM, and could not correspond to anything goal-adaptive since it is not dependent on the environment.

4.2.2.3 Optional additional requirements

We can consider variations of the stationarity relations with additional requirements. With the extra requirement that p is a limiting distribution, the stationarity relations have the *limiting property*. If a TAM is derived from a DBN-UE, the stationary distributions need to be consistent with the prior as well. For each set of values for the variables at a time step that has a positive probability according to the stationary distribution, there must also have a positive probability for the set of values at some time step given the prior BN. The stationarity relations then have the *prior compatibility property*, for a specific prior mechanised BN.

4.2.2.4 Example: a set of stationarity relations

Consider the example in Section 4.1. For simplicity, it is assumed that the training algorithm converges to an oprimal value for $\widetilde{B}$. The stationarity relations for the example in the previous section are as follows:

$$\widetilde{b} \in r_B(\mathbf{pa}_{\widetilde{B}}) \Leftrightarrow \widetilde{b} \in \operatorname{argmax}_{\widetilde{b}} \mathbb{E}[C_T \mid \widetilde{b}, \widetilde{a}, \widetilde{a_T}', \widetilde{c_T}', \widetilde{e}'] \tag{4.23}$$

$$\widetilde{a_T}' \in r_{A_T}(\mathbf{pa}_{\widetilde{A'_T}}) \Leftrightarrow \Pr(A_T = a \mid \widetilde{a_T}', \widetilde{a}, \widetilde{c_T}', \widetilde{e}') = \tag{4.24}$$

$$= \Pr(A_{T+1} = a \mid \widetilde{a_T}', \widetilde{a}, \widetilde{c_T}', \widetilde{e}') \text{ for all } a \in \mathrm{dom}(A) \tag{4.25}$$

$$\widetilde{c_T}' \in r_{C_T}(\mathbf{pa}_{\widetilde{C'_T}}) \Leftrightarrow \Pr(C_T = c \mid do(A_T = a), \widetilde{c_T}', \widetilde{c}, \widetilde{a_T}', \widetilde{e}') = \tag{4.26}$$

$$= \Pr(C_{T+1} = c \mid do(A_{T+1} = a), \widetilde{c_T}', \widetilde{c}, \widetilde{a_T}', \widetilde{e}') \quad \forall c \in \mathrm{dom}(C), a \in \mathrm{dom}(A) \tag{4.27}$$

$$\widetilde{e}' \in r_{E'}(\mathbf{pa}_{\widetilde{E}'}) \Leftrightarrow \Pr(A_T = a, C_T = c \mid \widetilde{e}', \widetilde{a_T}', \widetilde{c_T}', \widetilde{b}) = \tag{4.28}$$

$$= \Pr(A_{T+1} = a, C_{T+1} = c \mid \widetilde{e}', \widetilde{a_T}', \widetilde{c_T}', \widetilde{b}) \quad \forall a \in \mathrm{dom}(A), c \in \mathrm{dom}(C) \tag{4.29}$$

Note that the stationarity relation for $\widetilde{E}'$ specifies that any outcome $\widetilde{e}'$ should induce

a stationary distribution for both A and C. The stationarity relation for $\widetilde{B}$ specifies that $\widetilde{B}$ should be an optimal value for the mechanism at time step T, maximizing the expected value of C_T. For $\widetilde{C_T}'$, the stationarity relation specifies that the interventional distribution of C_T given A_T should be the same as the interventional distribution of C_{T+1} given A_{T+1}. This ensures the stationarity relation satisfies the third requirement in Definition 4.2.2, that the dependencies between object-level variables are the same for each time step.

When deriving a TAM for the exmple, while the resulting mechanism variables follow the stationarity relations above, additional modeling choices were made. For instance, $\widetilde{A_T}'$ is not allowed to take different values, since the stationary distribution for A could be accurately modeled using $\widetilde{E}'$. This means that the stationarity relation for $\widetilde{A_T}'$ is replaced with this relation:

$$\widetilde{a_T}' \in r_{A_T}(\mathbf{pa}_{\widetilde{A'_T}}) \Leftrightarrow \widetilde{a_T}' = 1 \tag{4.30}$$

Of course, no relation is needed for $\widetilde{A_T}'$, since it is simply set to 1.

4.2.3 Queries in a TAM

As there are pre-policy and post-policy queries in mechanised models of games, there are *pre-stationary* and *post-stationary* queries in a TAM, corresponding to queries before and after mechanism variables for the TAM have been specified.

Definition 4.2.3 (Answer to a conditional query)**.** Given a TAM $\mathrm{m}\mathcal{M}$ with the set of stationarity relations $\mathcal{R}$, the answer to a conditional query of the probability of $\mathbf{x}$ given observation $\mathbf{z}$ is given by the set $\Pr(\mathbf{x} \mid \mathbf{z}) := \{\Pr^{\widetilde{\mathbf{W}}}(\mathbf{x} \mid \mathbf{z})\}_{\widetilde{\mathbf{W}} \in \mathcal{R}(\mathrm{m}\mathcal{M}|\mathbf{z})}$, where $\widetilde{\mathbf{W}} = \widetilde{\mathbf{X}} \cup \widetilde{\mathbf{D}}$ is the set of mechanisms that are determined by stationarity relations. $\mathbf{X}$ is the set of prior mechanisms (at time step T), and $\mathbf{D}$ is the set of dynamic mechanisms. $\mathcal{R}(\mathrm{m}\mathcal{M}|\mathbf{z}) := \{\widetilde{\mathbf{W}} \in \mathcal{R}(\mathrm{m}\mathcal{M}) : \Pr^{\widetilde{\mathbf{W}}}(\mathbf{z}) > 0\}$ is the set of *conditional stationary outcomes*. In general, $\mathbf{Z} \subseteq \mathbf{V} \cup \widetilde{\mathbf{V}}$ can include mechanism variables.

If the original model describes causal dependencies the resulting TAM will also describe causal dependencies, and can be called a *causal TAM* (CTAM). For a CTAM we can, additionally to making predictions through conditional queries, make predictions through interventional queries, corresponding to the second level of the causal hierarchy.

Consider a mechanism variable that represents a prior distribution for one of the object-level variables. It is related to the stationary outcome of the mechanism for the object-level variable at future time steps. The dependence between the variables is a logical dependence. The prior mechanism variable is not directly caused by the mechanism variable at future time steps, but the mechanism variable at future time steps determines what values the prior mechanism are logically possible. It would be impossible to make an intervention on the prior mechanism variable that sets it to a value that is not allowed by the corresponding stationarity relation.

This type of logical dependence holds for all prior mechanisms and dynamic mechanisms. The stationarity relations specify logically valid values for each mechanism variable.

Definition 4.2.4 (Answer to an interventional query)**.** Given a TAM $\mathrm{m}\mathcal{M} = (\mathrm{m}G, \boldsymbol{\theta}, \mathcal{R})$, the answer to an interventional query on the probability of $\mathbf{x}$ given intervention $\mathcal{I}$ on variables $\mathbf{Y}$ is given by the set $\Pr^{\mathcal{R}}(\mathbf{x}|\mathcal{I}) := \{\Pr^{\widetilde{\mathbf{W}}}(\mathbf{x}|\mathcal{I})\}_{\widetilde{\mathbf{W}} \in \mathcal{R}(\mathrm{m}\mathcal{M}_{\mathcal{I}})}$ where $\mathcal{R}(\mathrm{m}\mathcal{M}_{\mathcal{I}})$ is the stationarity relations induced by the intervention $\mathcal{I}$.

An intervention on a prior mechanism variable in a TAM would specify the stationary distribution for the corresponding object-level variable. However, values for prior mechanisms that are logically possible are determined by the corresponding stationarity relation. An intervention on a prior mechanism can only specify a value that is allowed by the stationarity relation.

An intervention on post mechanisms corresponds to setting the corresponding mechanism variables to specific values in the original mechanised TBN-UE $\mathrm{m}B_{\rightarrow,o}$, at all time steps. $\mathrm{m}B_{\rightarrow,o}$ could be the entire original model for the TAM, or part of a DBN-UE. This is the case for dynamic mechanisms as well, which then do no longer have ingoing update edges in the modified TBN-UE $\mathrm{m}B_{\rightarrow,o,\mathcal{I}}$. An intervention on a dynamic mechanism can set it to a value that would not otherwise be allowed by the stationarity relation.

An intervention restricted to mechanism variables in the TAM is a *pre-stationary intervention*, since it determines possible stationary outcomes for mechanism variables that are not intervened on. An intervention restricted to object-level variables is a *post-stationary intervention*, since it does not change possible stationary outcomes, i.e., $\mathcal{R}(\mathrm{m}\mathcal{M}_I) = \mathcal{R}(\mathrm{m}\mathcal{M})$ when $\mathbf{Y} \subseteq \mathbf{V}$, where $\mathbf{V}$ is the set of all object-level variables.

4.3 Relation to models of agency

A TAM is well-specified as a mathematical construct that can handle causal queries for a system, abstracted over time, but how it relates to the models for systems containing agents, as covered in Section 2.3.2, is not clear. In this section, we will discuss how a TAM corresponds such models.

First, the TAM for the example presented earlier (Section 4.1) will be discussed. Then it is considered how a TAM can be limited to a finite number of time steps under the assumption that any eventual agent optimizes for values of variables up to a certain time step. Lastly, it is discussed how decision and utility nodes can be identified in a TAM.

4.3.1 Example: TAM as a mechanised CID

For the TAM in the example presented earlier (Section 4.1), we know that C could be interpreted as a reward, while B could be interpreted as actions. The training procedure can be interpreted as $\widetilde{B}$ updating to select B_t to optimize for C_t, at the

same time step, and so we can limit the TAM to only show time step T. For the resulting model, B_T is represented as a decision variable and C_T is a utility variable. The TAM can then be represented as a mechanised CID, as the graph to the left in Figure 4.4. While a mechanised CID or causal game generally only allows the decision rule to be determined by a relation, a TAM can be represented as a mechanised CID with the relaxation that other mechanisms can also be determined by relations. Since $\widetilde{C}$ is not explicitly present in the left subgraph, it can instead be considered a part of $\widetilde{C_T}' = (r, r')$ to show that $\widetilde{B}$ adapts to optimize for C_T through the edge from $\widetilde{C_T}'$ to $\widetilde{B}$.

Note that prior mechanism variables should not be considered as decision rules, as they are determined by the converged values of the mechanism variables represented by post mechanism variables. In the example, we could consider $\widetilde{B}$ as a decision rule because the mechanism for B was merged for all time steps, meaning that we no longer consider the mechanism for B_T to be a prior mechanism.

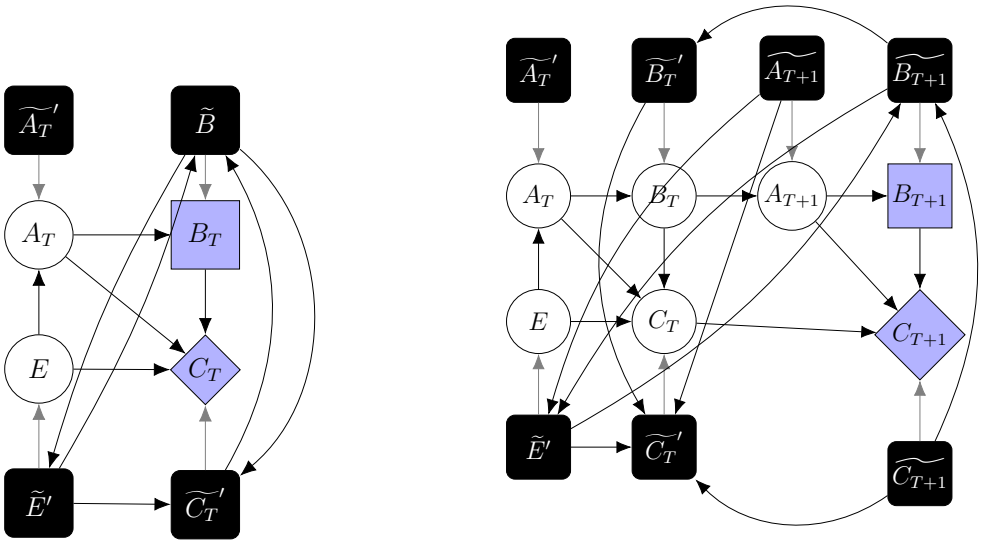
Alternatively, we could consider including two time steps, and separating the prior mechanism variable for B. This would retain the information that C is dependent on its previous value. It would however make the model more complex. The corresponding graph is shown to the right in Figure 4.4.

When two time steps are included, it might seem more appropriate to merge the mechanism for B, and consider both B_T and B_{T+1} as decision nodes. The reason this is not done is that B is chosen to optimize for C at the current time step, which implies that B_T and B_{T+1} correspond to different utility nodes. A creative way to represent this would be to consider B_T and B_{T+1} as decisions from two different agents that shares the same decision rule. This does however not seem to be a natural way to represent the system, since it essentially only contains a single agent. In Figure 4.4, the system is instead modeled as $\widetilde{B_{T+1}}'$ being the decision rule, and $\widetilde{B_T}'$ being affected by the decision rule, since it necessarily takes the exact same value. There is an edge from $\widetilde{B_T}'$ to $\widetilde{C_T}'$ since $\widetilde{B_T}'$ represents the stationary distribution for B, and the stationary distribution for C is dependent on the stationary distribution for B.

4.3.2 Myopic agents

Under the assumption that an agent is optimizing for values up to a certain time step $T + \tau$, we can limit the model to only show variables up to time step $T + \tau$, which would then still represent anything we might want to reason about regarding the agent. Such a model will be referred to as a *finite TAM*. A TAM can be interpreted as a mechanised DBN with an unspecified prior. By limiting the number of time steps, the resulting model is a mechanised BN and a set of stationarity relations, with some mechanism variables left unspecified.

An agent is however not necessarily myopic, but might learn to discount future outcomes, in which case a large number of time steps would be enough for an accurate model.



(a) Single time step

(b) Two time steps

Figure 4.4: TAM for the example in Section 4.1 presented as a variant of a mechanised CID, including one time step (a) or two time steps (b).

4.3.3 Specifying prior mechanism variables

The stationarity relations allows for a set of values for the mechanism variables in a TAM. Unlike the common definition of CIDs and CGs, where only the CPDs for decision variables are left unspecified, both mechanism variables in the prior, and dynamic mechanisms are unspecified.

However, a TAM can be converted to a model that more closely correspond to a mechanised CID or CG by intervening on prior mechanism variables to set them to some specific values allowed by the stationarity relations. This would leave only dynamic mechanisms unspecified. The domains of the prior mechanisms would be determined by the stationarity relations, specifying the possible values they can take.

4.3.4 Other considerations for identifying agency

Sometimes it is fairly simple to identify decision and utility nodes, as for the example in Section 4.1 where it was possible to infer that C is a reward and B a decision from analysing how $\widetilde{B}$ is updated. It is however not always that simple. This section covers some considerations for interpreting a TAM as a model of agency.

4.3.4.1 Not all TAMs represent agents

A TBN-UE that specifies how a mechanism adapts over time does not necessarily correspond to a learning or optimization process. The mechanism might simply converge to a value that is not designed to optimize for anything in the environment. In such cases, there will be a corresponding TAM, but it would not be appropriate to consider the dynamic mechanism's object-level child as a decision variable.

4.3.4.2 Utility variables

A potential agent might not learn to maximize the expected value of some variable, but learn to minimize its value or achieve a specific value. In such cases, a utility variable might be constructed as a function of the variable the agent cares about. Note that it does not make sense to make an intervention on the mechanism for the utility variable, since it is simply chosen based on another variable that was originally in the system.

Secondly, the process might optimize for a relationship between variables, rather than a value for a specific variable. In such cases, the utility variable might be constructed as a function of several variables. Here it also does not make sense to make an intervention on the mechanism for the utility variable, since it is simply chosen based on other variables that were originally in the system.

Thirdly, the mechanisms with ingoing update edges might converge without actually optimizing for anything in the environment. In this case it is simply not possible to identify any utility variable, and the variable for which the mechanism has ingoing update edges cannot be considered a decision variable.

4.3.4.3 Mechanism variable or object-level variable

It is not always clear what should be considered to be a mechanism variable. Consider for example a system containing two learning agents, there are two mechanism variables that have ingoing update edges. We might get some possible stationary for this system, for which both agents have learned about the system dynamics and the learning has converged.

However, now consider letting the mechanism variable for one of the learning agents be an object-level variable. We might find that there are stationary outcomes where the learning converges for the remaining agent, but not for the other. Now only one of the learning agents is actually agentic, while the other is better modeled as part of the environment. The same initial system state can result in different number of agents. To consider all possible agentic outcomes, we need to consider all possible ways to divide the system into object-level and mechanism variables.

5

Examples

In this chapter, examples for applying the concepts introduced in the previous chapters are presented for two different reinforcement learning algorithms. First, there is a revisit of the multi-armed bandit problem, which was introduced in the introduction (Section 1.1.2). Then, to present a multi-agent example, the actor-critic algorithm is described.

For these examples, the focus is on the causal structure of the systems, since formal derivation of convergence properties is not the focus of this thesis.

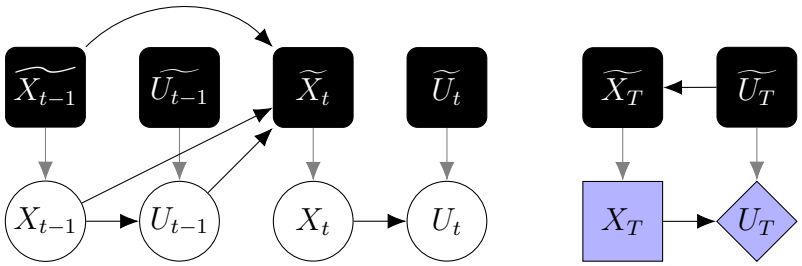
5.1 Multi-armed Bandit

The multi-armed bandit problem is a simple reinforcement learning problem where an agent must choose between different actions, each with an unknown reward distribution. The agent must learn which action is the best to maximize its reward. The problem is called the multi-armed bandit problem because it is analogous to a gambler choosing which slot machine to play.

We return to the example of a multi-arm bandit problem that was presented in the introduction (Section 1.1.2). Figure 1.1 is presented here again in Figure 5.1 for convenience. From Chapter 3, we now know that the graph to the left describes a TBN-UE or DBN-UE, depending on whether a prior BN is provided. From Chapter 4, we know that the right graph describes a TAM. If the update to the policy at each time step is designed to converge when sufficient evidence has been gathered for which option is optimal, the stationary distribution of the system will be a policy that maximizes the reward. The agent's decision rule can be designed to contain a parameter of how certain it is that the optimal option has been identified, that would decrease any time the agent receives a reward that contradicts its expectation. This would ensure that the only stable state of the system is the optimal policy.

5.2 Actor-critic

Actor-critic is a type of reinforcement learning algorithm that uses two separate models: an actor model and a critic model. The actor model is responsible for selecting actions, while the critic evaluates the actions taken by the actor.



(a) DBN-UE (b) TAM

Figure 5.1: DBN-UE for the multi-armed bandit problem (left) and the corresponding TAM (right).

The one-step actor-critic algorithm, as described in [18], can be represented with a TAM, as shown in Figure 5.2. The actor selects an action based on the current state, and the critic evaluates the action taken by the actor. The critic provides feedback to the actor, which is used to update the actor's policy. The actor-critic algorithm is an example of a goal-adaptive system, where the goal is to maximize the reward.

Algorithm 3 One-step Actor—Critic (episodic), [18]

Input: a differentiable policy parameterization $\pi(a|s,\theta)$
Input: a differentiable state-value function parameterization $\hat{v}(s,w)$
Parameters: step sizes $\alpha^\theta > 0, \alpha^w > 0$
Initialize policy parameter $\theta \in \mathbb{R}^{d'}$ and state-value weights $w \in \mathbb{R}^d$ (e.g., to 0)
for each episode **do**
  Initialize S (first state of episode)
  $I \leftarrow 1$
  while S is not terminal **do**
    $A \sim \pi(\cdot|S,\theta)$
    Take action A, observe S', R
    $\delta \leftarrow R + \gamma\hat{v}(S',w) - \hat{v}(S,w)$ (if S' is terminal, then $\hat{v}(S',w) \coloneqq 0$)
    $w \leftarrow w + \alpha^w \delta \nabla_w \hat{v}(S,w)$
    $\theta \leftarrow \theta + \alpha^\theta I \delta \nabla_\theta \ln \pi(A|S,\theta)$
    $I \leftarrow \gamma I$
    $S \leftarrow S'$
  end while
end for

Consider a slight modification of the algorithm, where it runs indefinetely instead of looping over several episodes. Note that in practice the algorithm continues to update with some small step size, but for our purposes we can consider the algorithm to have converged when the policy and value function have not changed significantly for a long time.

When representing the one-step actor-critic algorithm as a TAM, we do not need to consider every variable at time step T. Instead, we can consider only the state variable S, which is the only variable that causally influences the states in the next

time step (except for update variables in the original DBN-UE). In Figure 5.2, the graph for the resulting TAM is shown.

The state S_T is given as input to the actor that takes action A_{T+1}, which causally influences the next state S_{T+1} and the reward R_{T+1}. The critic evaluates both states S_T and S_{T+1}, and returns estimated utilities C^1_{T+1} and C^2_{T+1} respectively. The action is evaluated by δ_{T+1}, which can be interpreted as the difference between the estimated value of the state S_T and the actual outcome from the state S_T and the action A_{T+1}.

Since every mechanism variable is causally upstream of δ in the training process, which is an update variable for both the policy and the value function, there are ingoing edges from each mechanism variable to both the policy and the value function, $\widetilde{A}$ and $\widetilde{C}$. $\widetilde{S_T}'$ is some stationary distribution for the state associated with converged a policy function, since only the policy function affects the state directly. While only two time steps are shown in the graph, the algorithm can be run indefinetely, as any stationary outcome of a TAM specifies a DBN.

Note that A_{T+1} is chosen to maximize δ_{T+1}, while C^1_{T+1} is chosen to minimize its absolute value. This is inferred from the training algorithm. If δ is positive, the value function is updated to increase the value of the state, which decreases δ for the next time the state is visited. Conversely, if δ is negative, the value function is updated to decrease the value of the state, which increases δ for the next time the state is visited.

To symbolize this, A_{T+1} and C_{T+1} are decision nodes colored blue and red, respectively. The utility node δ_{T+1} is marked with both colors, since it is both maximized and minimized by the decisions. Normally an utility node is maximized, but here we take the liberty to mark it as an utility node for the critic as well, even though the critic attempts to minimize its absolute value.

Interestingly, we get something goal-adaptive if we abstract to only the action, state and reward, but the influence of the reward mechanism on the decision rule comes from training. This would be an interesting interaction between emergence through training and emergence through abstraction.

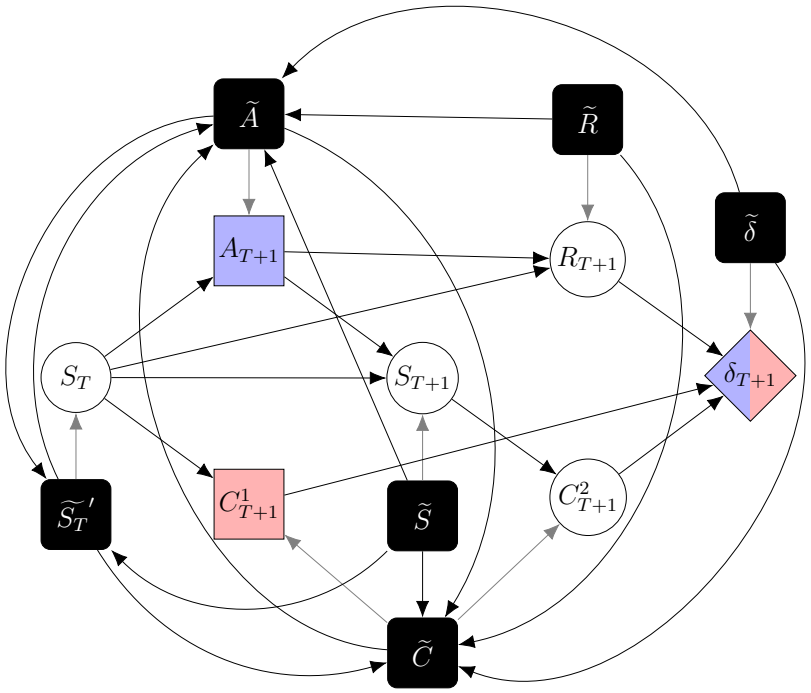


Figure 5.2: Resulting TAM for the one-step actor-critic algorithm.

6

Conclusion

This thesis has developed a formal framework for describing learning or optimization processes, and for describing stable dynamics of systems containing learning agents. However, there is still much work to be done in order to fully understand the implications of the framework. This chapter starts with a discussion of the framework and its implications. There are then a number of suggestions for future work. The chapter ends with some final words, reflecting on the work done and the potential for further development.

6.1 Discussion

The discussion is divided into five subsections. The first subsection discusses the assumptions made in the framework, and what qualitative insights can be gained even if the assumptions are not completely true. There is then a discussion about the nature of agents and how it relates to this thesis. The third subsection discusses the nature of mechanism variables, and the final subsection discusses the logical nature of TAMs.

6.1.1 Relaxing assumptions and qualitative insights

We can gain a better qualitative understanding of systems, even if these assumptions are not all completely true. Consider the case where the update edges continue to have an effect on mechanism variables, but the influence is small. The resulting system could reach a distributon that is almost stationary, and the abstraction would still be useful.

This thesis introduces formal mathematical definitions for modeling emergence of agency, but the theory can also provide qualitative insights. In practise, agents and the environment changes slowly enough that describing the system with a TAM is useful. Even if the stationary distributions are not known or computationally infeasible to calculate, the causal structure of the system can be useful.

Additionally, even if the exact nature of the learning process is not known, the theory can still provide insights. Given the assumption that the system reaches a stationary distribution, this limits the possible policies that agents might converge to. Assumptions can also be made regarding game-theoretic properties of the system, such that the policies converge to game-theoretic equilibria.

6.1.2 Understanding agency

Agents in general have to learn about their environment. This means that some parts of the environment are required to follow static or slowly changing patterns, which the agents can adapt to.

The stationarity relation requires that each mechanism with ingoing update edges converges, and the prior mechanism variables in the TAM specify a stationary distribution. How then, should we reason about systems that never converge? Humans learn to adapt to each other and the world, assumingly because the environment changes slow enough that meaningful predictions can be made, but any equillibrium is never reached. However, a TAM might still provide qualitative insights about the system if the system changes slowly enough that a distribution can be considered approximately stationary.

You could also consider a mechanism variable that converges even if the rest of the system does not have a stationary distribution. This could be an interesting extension of the theory, though it could prove hard to show that the mechanism variable converges. The resulting model could describe equilibria reached by potential agents, while not specifying all dynamics for the rest of the system.

Consider your own behaviour and thoughts. First, your memory is modeled as an object-level node, which affects any decision you make. In such a model, do your policy change? Or does your policy already incorporate how to use the memory for decisions? In this model, the policy might actually change very slowly, since your policy uses the memory approximately the same way in each time step (though your behavior might also change due to outside factors like time of day). However, you accumulate more knowledge over time, implying that there is no stationary distribution for your memory.

6.1.3 Mechanism variables

It is not always clear what should be considered an object-level variable and what should be considered a mechanism variable. An option would be to simply consider variables that are constanst or that can converge to specific values to be mechanism variables in a DBN-UE. This ensures that mechanism variables are the variables that don't change in a TAM.

Specifying some of the variables as mechanism variables seem to make it easier to reason about though. Specifying mechanism variables can be interpreted as specifying which variables we are interested in convergence of, that potentially can represent learning agents.

6.1.4 TAM as a logical object

A TAM can be interpreted as describing stable system dynamics. It is worth noting that the TAM does not describe the system at any specific time step, it does not describe mechanism variables exactly but has a set of possible values according to the stationarity relation, and the relationships between the mechanism variables

represent dependencies through many time steps. This makes TAM, while useful as a concept for making predictions and modeling learning, a model that is logical by nature. It is induced by system dynamics that describes reality, but does not itself represent anything that can be found in the physical universe. By extension, since TAMs were designed to represent how agents adapts, this means that models such as CIDs and CGs might also be considered logical by nature.

6.2 Future work

There are several areas of further development. Here they are divided into four categories. Further examination of stationarity relations are discussed first, followed by possible extensions to the framework. The third category is about the relation to models of agency, and the final category is research and experiments that could be done.

6.2.1 Examine stationarity relations

The TAM requires that there are stationary distributions for the system, where mechanism variables with ingoing update edges converge. An important area of future work would be to examine necessary and sufficient conditions for when this is fulfilled.

The mechanisms with ingoing update edges are required to converge, but this thesis does not provide a method for determining if they do. To handle this the framework could be related convergence proofs.

6.2.2 Possible extentions

The simplest extension is to consider introducing a vector of variables $\mathbf{E}$ instead of a single variable E, where each variable in the vector represents separate and unrelated dependencies for the prior object-level variables in the TAM. Such unrelated dependencies can technically be represented by a single variable, but introducing more would allow for a more intuitive representation that graphically shows that there are several different dependencies.

As mentioned in the discussion, it might be interesting to consider convergence of mechanisms in cases where a stationary distribution is not required for the rest of the system, as discussed in Section 6.1.2. This would require a relaxation of the stationarity relation.

Another possible extension would be to associate probabilities to the stationary outcomes, which would be possible given that the original model is a DBN-UE which provides a distribution for the initial variables through a prior BN.

6.2.3 Relation to models of agency

Further work could also be done to relate the framework to the models of agency. While some considerations are provided in this thesis, the relation can be further explored and formalized.

A promising approach would be to introduce a formalization for adding utility variables in the TAM when such variables are not explicitly present in the system, but can be inferred from how the agents behave.

6.2.4 Research and experiment

Game-theory is commonly used to make predictions of how agents will behave in a system. This thesis, which examines how a training methodology affects will result in various equilibria, adds a new perspective.

If it is observed that a system is stable while containing learning agents, the TAM suggests that the system can be described by a DBN, where the prior distribution is given by a stationary distribution for the stable system. Given an assumption of what each agent's utility is, this provides a CG. Then we could use game-theoretic analysis to predict how agents will behave in the system, for example by examining Nash equilibria. The TAM suggests the extra requirement that any Nash equilibria must be compatible with the stationary distribution.

This type of analysis shows that the TAM can be used to understand how to make game-theoretic predictions about systems containing learning agents. Applying this to real-world systems could be an interesting area of future work. The framework presented in this thesis can be used for examining what CIDs or CGs that results from various training methods. This has applications in evaluating how suitable a given training method is for creating safe agents.

Further research on how to apply the framework could be done in two main areas:

1. Designing a training methodology that robustly results in safe agents.
2. Predicting properties for an agent given a specific training methodology.

6.3 Final words

This thesis should be seen as an initial exploration of a novel approach to understanding emergence of agency. This thesis is an early exploration of a novel approach to understanding emergence of agency. My initial ambition was to better understand properties of agents given the training procedure and learning environment, for the purpose of making them safe.

However, there was not enough time to conduct an in-depth analysis of potential applications. Developing the formal framework required significant effort, and its implications are still not fully understood. Safety-related applications remain a promising direction for future research.

Bibliography

[1] K. E. Drexler, "Reframing superintelligence: Comprehensive ai services as general intelligence," *Technical Report #2019-1*, 2019. [Online]. Available: https://www.fhi.ox.ac.uk/wp-content/uploads/Reframing_Superintelligence_FHI-TR-2019-1.1-1.pdf.

[2] *Causal incentives working group*. [Online]. Available: https://causalincentives.com.

[3] W. Uther, "Markov decision processes," *Encyclopedia of Machine Learning*, 2011. DOI: 10.1007/978-0-387-30164-8_512.

[4] P. Poupart, "Partially observable markov decision processes," *Encyclopedia of Machine Learning*, 2011. DOI: 10.1007/978-0-387-30164-8_629.

[5] D. C. Dennett, "Intentional systems," *The Journal of Philosophy*, vol. 68, no. 4, 1971.

[6] L. Hammond, J. Fox, T. Everitt, R. Carey, A. Abate, and M. Wooldridge, "Reasoning about causality in games," *Artificial Intelligence*, vol. 320, p. 103 919, Jul. 2023, ISSN: 0004-3702. DOI: 10.1016/j.artint.2023.103919. [Online]. Available: http://dx.doi.org/10.1016/j.artint.2023.103919.

[7] Z. Kenton, R. Kumar, S. Farquhar, J. Richens, M. MacDermott, and T. Everitt, "Discovering agents," *Artificial Intelligence*, vol. 322, 2023. [Online]. Available: https://doi.org/10.1016/j.artint.2023.103963.

[8] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009, ISBN: 9780262013192.

[9] P. A. Gagniuc, *Markov Chains: From Theory to Implementation and Experimentation*. Wiley, 2017. DOI: 10.1002/9781119387596.

[10] J. Pearl, *Causality*. Cambridge University Press, 2009. DOI: 10.1017/CBO9780511803161.

[11] K. P. Murphy, *Dynamic Bayesian Networks: Representation, Inference and Learning*. UC Berkeley, 2002. [Online]. Available: https://www.cs.ubc.ca/~murphyk/Thesis/thesis.pdf.

[12] R. A. Howard and J. E. Matheson, "Influence diagrams," *Decision Analysis*, vol. 2, no. 3, 2005. [Online]. Available: https://doi.org/10.1287/deca.1050.0020.

[13] J. Fox, M. MacDermott, L. Hammond, P. Harrenstein, A. Abate, and M. Wooldridge, "On imperfect recall in multi-agent influence diagrams," *Electronic Proceedings in Theoretical Computer Science*, vol. 379, 201–220, Jul. 2023, ISSN: 2075-2180. DOI: 10.4204/eptcs.379.17. [Online]. Available: http://dx.doi.org/10.4204/EPTCS.379.17.

[14] N. Soares and B. Fallenstein, *Toward idealized decision theory*, 2015. arXiv: 1507.01986 [cs.AI].

[15] M. MacDermott, T. Everitt, and F. Belardinelli, *Characterising decision theories with mechanised causal graphs*, 2023. arXiv: 2307.10987 [cs.AI].

[16] E. Yudkowsky and N. Soares, *Functional decision theory: A new theory of instrumental rationality*, 2018. arXiv: 1710.05060 [cs.AI].

[17] B. A. Levinstein and N. Soares, "Cheating death in damascus," *The Journal of Philosophy*, vol. 117, no. 5, pp. 237–266, 2020. [Online]. Available: https://doi.org/10.5840/jphil2020117516.

[18] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 2018. [Online]. Available: http://incompleteideas.net/book/the-book-2nd.html.