EXPLAINABILITY IS A GAME FOR PROBABILISTIC BISIMILARITY DISTANCES

ANTO NANAH JI

A THESIS SUBMITTED TO
THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

DEPARTMENT OF ELECTRICAL ENGINEERING & COMPUTER SCIENCE
YORK UNIVERSITY
TORONTO, ONTARIO

April 2025

Abstract

Software bugs cost trillions annually, requiring better bug detection tools. Testing is widely used but has limitations, especially in non-deterministic software, where code produces different outputs even with fixed inputs due to randomness and concurrency. Labelled Markov chains model randomness but suffer from state space explosion problem, where the number of states grows exponentially with system complexity. One solution is to identify behaviorally equivalent states using probabilistic bisimilarity. However, this method is not robust, small changes in probabilities can affect equivalences. To address this, probabilistic bisimilarity distances were introduced, a quantitative generalization of probabilistic bisimilarity. These distances have game-theoretic characterizations. This thesis illustrates how optimal policies, known as player's strategies, can explain distances. We formulate 1-maximal and 0-minimal policies, argue that they lead to better explanations. We present algorithms for these policies, prove an exponential lower bound for the 1-maximal algorithm, and show that symmetries simplify policies and, hence, explanations.

Declaration

I certify that this dissertation is solely my own work and, unless otherwise specified, represents my own research. ChatGPT[1] was also used in this thesis to assist with English grammar and spelling.

[1] chatgpt.com.

Acknowledgments

I would like to express my profound gratitude to my supervisor, Franck van Breugel, for his exceptional guidance, insightful feedback, and unwavering support throughout this dissertation. His expertise and dedication have been invaluable, and it has been an honor to work under his supervision. I am also deeply grateful to Professor Eric Ruppert for his constructive feedback, which significantly contributed to the refinement of my work, as well as for attending my defense. Furthermore, I would like to thank Professor Paul J. Szeptycki for his thoughtful participation in my defense and for providing valuable insights that significantly enhanced the quality of this dissertation.

TABLE OF CONTENTS

List of Tables

List of Figures

1 Introduction

The cost of software bugs is trillions of dollars every year [59], [42]. Hence, there is a pressing need for tools to detect bugs in software. The traditional way to find code bugs is performed by testing such as writing JUnit[2] test cases. However, testing has its limitations, especially in non-deterministic software, that is, code that may give rise to different executions even if all the input is fixed. Both randomness and concurrency give rise to non-determinism. We often see randomness in, for example, artificial intelligence, embedded systems, and cryptographic protocols. Consider the following Java code.

```
1  import java.util.Random;
2  public class Example
3  {
4      public static void main(String[] args)
5      {
6          Random random = new Random();
7          int n = random.nextInt(10);
8          System.out.print(1 / n);
9      }
10 }
```

The above application may result in ten executions as it randomly chooses an integer in the interval $[0, 10)$. The application will crash with an uncaught exception if the random number is zero. When we run the application once for testing, with a probability of 0.9 it prints zero or one, and with a probability of 0.1, it crashes. When we run the application twice for testing, with a probability of 0.81 it prints zero or one, and with a probability of 0.19, it crashes. When we run the application ten times for testing, with a probability of 0.35 it prints zero or one, and with a probability of 0.65, it crashes. Therefore, we can conclude that traditional testing may not be guaranteed to capture all software bugs.

Model checking is often used as an alternative to testing non-deterministic software. First, a model of the system is built. For example, if the software behaves non-deterministically due to concurrency or randomness then labelled transition systems (LTSs) are often used to model the code. This model was introduced by Keller [32].

An LTS consists of a set of states, a set of transitions, a set of labels, and a labelling function. A state of an LTS describes information about a system at a given time. For example, a state of a model of the above Java code may include the current values of all program variables together with the current value of the program counter, that is, it represents a state of the Java virtual machine (JVM). Transitions specify how the system can advance from one state to another. In the case of the above Java program, a transition typically corresponds to the execution of a sequence of byte-code instructions by the JVM and may involve the change of the value of a variable and the program counter. A state of an LTS is labelled with a label chosen from a set. Labels represent different aspects of the states. For the above Java application, a label may represent an exception thrown in the state but never caught. The same label may appear in more than one state. The labelling function assigns a label to each state in an LTS. For example, the labelling function of the model for the above Java code may assign the above mentioned exception label to the state where a division by zero occurs (more precisely, the state reached immediately after the bytecode instruction that throws the exception has been executed). The LTS modeling the above Java code can be graphically represented as follows.

[2] junit.org/junit5.

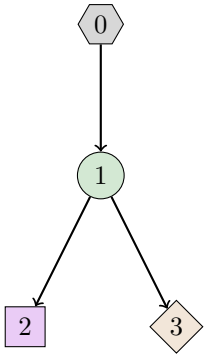


Figure 1.1: LTS modeling the above Java code

State 0 is the initial state. State 1 represents the state of the JVM when the execution has reached line 7 of the above code. State 3 represents the state of JVM when the execution of the code terminates successfully. State 2 represents the state of the JVM when the execution of the code fails to terminate due to an uncaught exception. State 0 is assigned the initial state label, state 1 is assigned the random number generation label, state 3 is assigned the successful execution label, and state 2 is assigned the division by zero exception label.

In model checking, a property to be verified is specified in addition to the model. Such a property is usually described in a logic. For example, computation tree logic (CTL) can be used to describe a property. This logic was introduced by Clarke and Emerson [8]. Many properties can be expressed in CTL. For example, we can express the property that no state in which an uncaught exception is thrown can be reached from the initial state.

Model checking helps find bugs in non-deterministic systems by exploring all possible behaviours and ensuring that the system satisfies its properties across all potential execution paths, thereby catching errors that could arise from non-deterministic choices.

LTSs focus on non-deterministic choices. However, in practice, many systems are subject to various phenomena of a stochastic nature such as uncertainty or randomness. To model random phenomena, probabilities are added to the LTS's transitions and these probabilistic systems are called labelled Markov chains (LMCs). LMCs are labelled transition systems with probability distributions for the successors of each state, that is, instead of a non-deterministic choice, the next state is chosen probabilistically. Formal definitions of notions such as LMCs will be provided in later chapters. The LMC modeling the above Java code can be graphically represented as follows. Notice that probabilities are added to the transitions of states 0 and 1, and a self-loop with a probability of one is added to states 2 and 3 to satisfy LMC properties.

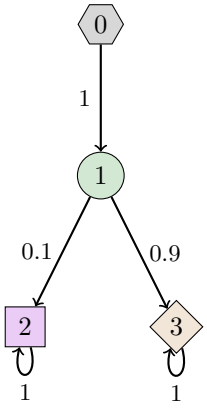


Figure 1.2: LMC modeling the above Java code

Similarly, for probabilistic models, probabilistic computational tree logic (PCTL) can be used to describe a probabilistic property. This logic was introduced by Hansson and Jonsson [28]. Numerous probabilistic properties of code with randomness can be expressed in PCTL. For example, we can express the property that a state in which an uncaught exception is thrown can be reached with a probability less than 0.1 from the initial state.

The number of states in an LTS or LMC often grows exponentially as the number of threads and random

choices increases (as in line 6 of the above Java code). This phenomenon is called the state space explosion problem. One way to deal with this is to reduce the number of states by identifying states that behave the same. A behavioural equivalence is an equivalence relation on the states of a model that captures whether state pairs behave the same.

1.1 Bisimilarity

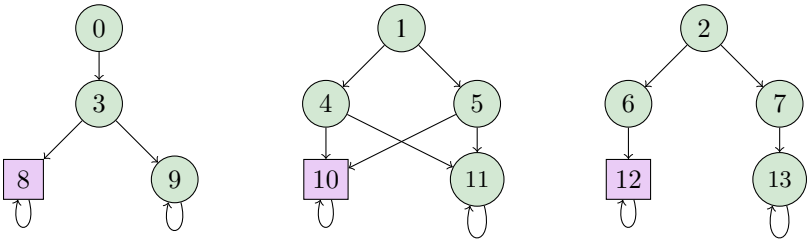


Figure 1.3: An LTS

Milner [47] and Park [48] proposed a behavioural equivalence for the states of an LTS, called *bisimilarity*. States are bisimilar if they have the same label and their transitions can be matched such that matching transitions have bisimilar target states.

As an example, consider the LTS depicted in Figure 1.3 and the following partition of its states: $T_1 = \{3, 4, 5\}$, $T_2 = \{8, 10, 12\}$, $T_3 = \{9, 11, 13, 7\}$, $T_4 = \{6\}$, $T_5 = \{2\}$, and $T_6 = \{0, 1\}$. As we will argue next, the states 0 and 1 are bisimilar. First, observe that states 8, 10, and 12 share the same label (purple) and only transition to T_2. Since their transitions can be matched, they are bisimilar. A similar argument can be used to show that states 7, 9, 11, and 13 are bisimilar. Next, states 3, 4, and 5 all share the same label (green) and transition to both T_2 and T_3. As their transitions can be matched, we can conclude that states 3, 4, and 5 are bisimilar. Finally, states 0 and 1 both share the green label and transition to only T_1. Since their transitions can be matched, we conclude that states 0 and 1 are bisimilar.

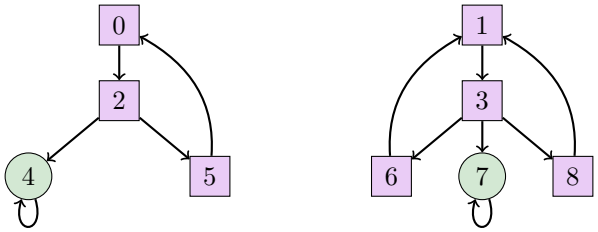


Figure 1.4: An LTS

Now consider the LTS depicted in Figure 1.4 and the following partition of its states: $T_1 = \{5, 6, 8\}$, $T_2 = \{2, 3\}$, $T_3 = \{4, 7\}$, and $T_4 = \{0, 1\}$. States 5, 6, and 8 all share the same label (purple) and each transition only to T_4. Since their transitions can be matched, they are bisimilar. Similarly, states 2 and 3 both share the label (purple) and transition to both T_1 and T_3, making them bisimilar as well. States 4 and 7 share the green label and transition only to T_3, thus making them bisimilar. Finally, states 0 and 1 have the same label (purple), with both transitioning only to T_2. Given that their transitions match, they too are bisimilar.

1.2 Probabilistic Bisimilarity

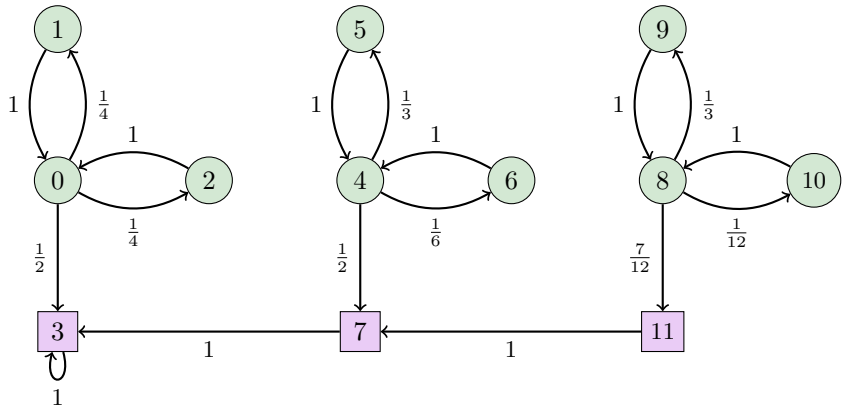


Figure 1.5: An LMC

Larsen and Skou [43] introduced the fundamental concept of *probabilistic bisimilarity*, which captures which states of an LMC are considered behaviorally equivalent. States are probabilistic bisimilar if they have the same label and transition to probabilistic bisimilarity equivalence classes with the same probabilities, that is, the sum of the probabilities of their transitions to each equivalence class of probabilistic bisimilar states is the same.

For example, consider the following partition of the states of the LMC depicted in Figure 1.5: $T_1 = \{1, 2, 5, 6\}$, $T_2 = \{3, 7, 11\}$, $T_3 = \{9, 10\}$, $T_4 = \{8\}$, and $T_5 = \{0, 4\}$. We observe that states 0 and 4 have the same label (green), and the sum of the probabilities of transitions from state 0 to each of T_1, T_2, T_3, T_4, or T_5 is equal to the sum of the probabilities of transitions from state 4. Similar conditions can be checked for the other state pairs. Hence, states 0 and 4 are probabilistic bisimilar.

Algorithms have been developed to decide probabilistic bisimilarity in LMCs by, for example, Buchholz [4], Derisavi, Hermanns, and Sanders [13], Valmari and Franceschinis [56]. It has been shown by Katoen, Kemna, Zapreev, and Jansen in [31] that first reducing the number of states of an LMC by identifying probabilistic bisimilar states and then checking a property of the reduced system is often more efficient than checking the property for the original system. Fisler and Vardi [23] observed that bisimulation minimization appears impractical for LTSs, as the resources required for minimization may be equal to or greater than those needed for model checking the unminimized LTSs.

As first observed by Giacalone, Jou, and Smolka [25], probabilistic bisimilarity is *not robust*. Miniscule changes to the probabilities in the model may result in different states being identified as behaviourally equivalent and, hence, may lead to different reduced models. These models may even satisfy different properties. Consider the following example.

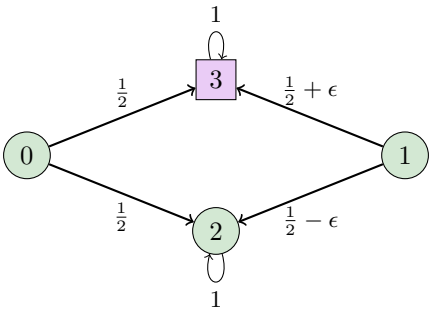


Figure 1.6: An LMC, where $\epsilon \in [0, \frac{1}{2}]$

When $\epsilon = 0$, states 0 and 1 behave exactly the same way and can be collapsed, resulting in a reduced model. However, if we change ϵ to something very close to 0 like 0.001, the states 0 and 1 will behave almost the same. This behaviour can be observed, for example, through a PCTL property, which indicates that a purple (square)

state can be reached with a probability less than or equal to 0.5 from the initial state. The reduced model satisfies this property. On the other hand, when $\epsilon > 0$, the original model does not satisfy this property.

1.3 Probabilistic Bisimilarity Distances

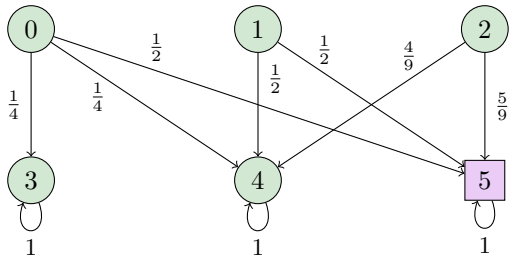


Figure 1.7: An LMC

Giacalone, Jou, and Smolka [25] suggested distances as a robust alternative to equivalences. Desharnais, Gupta, Jagadeesan, and Panangaden in [17] proposed a quantitative generalization of probabilistic bisimilarity: probabilistic bisimilarity distances (or distances for short). To each pair of states, a real number in the unit interval $[0, 1]$ is assigned that captures the behavioural similarity of the states. The smaller this number, the more alike the states behave. As shown by Desharnais et al. states are probabilistic bisimilar if and only if their distance is zero. For example, the distance between states 1 and 2 in Figure 1.7 is $\frac{1}{18} \approx 0.06$.

Over the last two decades, efficient algorithms have been developed to approximate and compute probabilistic bisimilarity distances. For instance, Chen, Van Breugel, and Worrell [6] presented a polynomial-time algorithm to compute these distances. They showed that if the transition probabilities are rational then the distances are rational and can be calculated using Khachiyan's ellipsoid method [33]. In particular, they demonstrated that the distance function can be expressed as the solution to a linear program. Tang [52] created and implemented algorithms capable of computing probabilistic bisimilarity distances for LMCs with thousands of states in just a few seconds.

Given that we can calculate the distance between two states as 0.06, it raises the question of why the distance is 0.06. In this thesis, we address the question of how to explain probabilistic bisimilarity distances.

Behavioural equivalences such as bisimilarity and probabilistic bisimilarity can be characterized in terms of logic, where states are behaviorally equivalent if and only if they satisfy the same formulas of the logic. We will explore this approach in the following section.

1.4 Logical Explanation

1.4.1 Bisimilarity

Hennessy and Milner [29] provided a logical characterization of bisimilarity by introducing Hennessy-Milner logic and proving that two states are bisimilar if and only if they satisfy the same formulas within the logic. In the case of an LTS with finitely many states, if two states are not bisimilar, there exists a formula, known as a distinguishing formula, such that one state satisfies the formula while the other does not. This formula explains the reason the two states are not bisimilar. Cleaveland [9] introduced a polynomial-time algorithm that computes a distinguishing formula for states that are not bisimilar.

For example, consider the formula that expresses that a state can transition to another state that can, in turn, transition to both a purple (square) state and a green (circle) state. Since state 0 in Figure 1.3 satisfies this formula, while state 2 does not, we can conclude that states 0 and 2 are not bisimilar.

1.4.2 Probabilistic Bisimilarity

As we mentioned earlier Larsen and Skou [43] introduced probabilistic bisimilarity to capture which states of an LMC behave the same. They also introduced a logic that characterizes probabilistic bisimilarity. Building on their

work, Desharnais, Edalat, and Panangaden [14] simplified this logic and proposed a polynomial-time algorithm to generate a formula that differentiates two states that are not probabilistically bisimilar.

For example, state 0 in Figure 1.5 transitions to a purple state in one transition with a probability of at most 0.5, whereas state 8 does not. This property can be captured in the logic, resulting in a formula that distinguishes states 0 and 8. Therefore, this formula explains that states 0 and 8 are not probabilistic bisimilar.

1.4.3 Probabilistic Bisimilarity Distances

To define probabilistic bisimilarity distances, Desharnais, Gupta, Jagadeesan, and Panangaden [17] adapted the logic of Desharnais, Edalat, and Panangaden [14] to a quantitative context. In essence, the distance between two states is defined by the formula from the logic that distinguishes them the most. Such a formula explains their probabilistic bisimilarity distance. More recently, Rady and van Breugel [50] employed a modified version of their logic to characterize probabilistic bisimilarity distances. Both Desharnais et al. and Rady et al. provided a real-valued interpretation for their respective logic where the value of a formula φ in a state, say s, denoted as $[\![\varphi]\!](s)$ is a real number in the interval $[0, 1]$.

Consider, for example, the states 1 and 2 in the LMC shown in Figure 1.7. Their distance is 0.06, which can be explained by the formula that captures the probability of reaching a purple state in one transition. The real value of this formula in state 1 is 0.5, and in state 2 it is 0.56. The difference between these values, 0.06, represents the distance between states 1 and 2.

Furthermore, Rady and van Breugel [50] explained the probabilistic bisimilarity distance for each pair of states by constructing a sequence of formulas $(\varphi_n)_n$. For a given pair of states, say s and t, they constructed a sequence of formulas $\varphi_0, \varphi_1, \varphi_2, ...$, such that the sequence $[\![\varphi_0]\!](s) - [\![\varphi_0]\!](t), [\![\varphi_1]\!](s) - [\![\varphi_1]\!](t), [\![\varphi_2]\!](s) - [\![\varphi_2]\!](t), ...$ converges to the probabilistic bisimilarity distance between s and t. This sequence $(\varphi_n)_n$ serves to explain the distance between the states s and t. This result is their explanation framework [50, Corollary 23]. The main limitation of their approach is that the explanation is generally not finitely representable.

Another way to characterize behavioural equivalences is through the use of a game, which we will explore in more detail in the following section.

1.5 Game-Based Explanation

1.5.1 Bisimilarity

As was shown by Stirling [51], bisimilarity for LTSs can be characterized by means of a two-player game. In the literature, we find different names for these players including Spoiler, Adversary, and Attacker for the first one and Duplicator, Prover, and Defender for the other player. Here, we use Spoiler and Duplicator. The game starts in a state pair (s, t). Spoiler tries to show that s and t are not bisimilar, whereas Duplicator tries to prove that they are. We assume that each state has an outgoing transition to simplify the game slightly. The game is played in rounds. If a round starts in state pair (s, t), Spoiler chooses a state $u_1 \in \{s, t\}$ and an outgoing transition of u_1 with a target, say, s'. Duplicator uses the other state $u_2 \in \{s, t\} \setminus \{u_1\}$ and chooses one of the outgoing transitions of u_2 with a target, say, t' (preferably with the same label as s' – otherwise Duplicator loses). The next round of the game continues in the state pair (s', t'). The objective of Spoiler is to reach a state pair with different labels whereas Duplicator tries to avoid ever reaching such a state pair. As has been shown by Stirling [51, Proposition 3], states s and t are bisimilar if and only if Duplicator can avoid ever reaching a state pair with different labels when the game is started in state pair (s, t) no matter how Spoiler plays. As pointed out by Fijalkow, Klin, and Panangaden [22], "this classical bisimulation game is elegant because it allows one to characterize a global property of behaviours (bisimilarity) in terms of a game whose rules only depend on local considerations."

Consider the LTS of Figure 1.3. If we start the game in state pair $(0, 1)$, then we are in $(3, 4)$ or $(3, 5)$ after one round. Assume we move to $(3, 4)$. No matter which transition Spoiler chooses, Duplicator can always pick a transition so that the game ends up in either $(8, 10)$ or $(9, 11)$. The same applies to $(3, 5)$. Since states 8 and 10 have a single outgoing transition, the game stays in $(8, 10)$ once it reaches that state pair. The same applies to $(9, 11)$. Hence, Duplicator can always avoid reaching a pair with different labels when the game is started in $(0, 1)$. The strategy of Duplicator, showing that Duplicator can always avoid state pairs with different labels, can be seen as an explanation that the states 0 and 1 are bisimilar.

Let us next consider the state pair $(0, 2)$. After one round we end up in either $(3, 6)$ or $(3, 7)$. Assume we are in $(3, 6)$. Assume that Spoiler chooses the transition from state 3 to state 9. Duplicator has the transition from state 6

to state 12 as its single choice. States 9 and 12 have different labels. Using a similar argument we can show that Duplicator cannot avoid reaching a state pair with different labels from $(3, 7)$. The strategy of Spoiler, showing that it is impossible for Duplicator to avoid a state pair with different labels, can be viewed as an explanation that the states 0 and 2 are not bisimilar.

1.5.2 Probabilistic Bisimilarity

Several characterizations of probabilistic bisimilarity for LMCs in terms of a game can be found in the literature. We will briefly review two of those next. The games are also played in rounds by two players. As in the game for bisimilarity the objective of Spoiler is to reach a state pair with different labels whereas Duplicator tries to avoid ever reaching such a state pair.

Let us first consider the game introduced by Desharnais, Laviolette, and Tracol in [16, Definition 10 with $\epsilon = 0$]. If the game is in state pair (s, t), Spoiler chooses a state $u_1 \in \{s, t\}$ and a set of states U_1. Duplicator uses the other state $u_2 \in \{s, t\} \setminus \{u_1\}$ and chooses a set of states U_2 such that the probability of reaching a state in U_1 from state u_1, that is, the sum of probabilities of the transitions from u_1 to a state in U_1, is less than or equal to the probability of reaching a state in U_2 from u_2. Such a U_2 always exists as Duplicator can pick the set of all states. Subsequently, Spoiler chooses $i \in \{1, 2\}$ and a state $s' \in U_i$. Finally, Duplicator chooses a state $t' \in U_{3-i}$ (preferably with the same label as s' – otherwise Duplicator loses). The next round of the game starts in the state pair (s', t'). As in the bisimulation game, the objective of Spoiler is to reach a state pair with different labels whereas Duplicator tries to avoid ever reaching such a state pair. Desharnais, Laviolette, and Tracol [16, Theorem 5] proved that states s and t are probabilistic bisimilar if and only if Duplicator can avoid ever reaching a state pair with different labels when the game is started in state pair (s, t) no matter how Spoiler plays.

Consider the LMC in Figure 1.5. Let us start the game in the state pair $(4, 8)$. The Spoiler plays on state 4, and their first move is $U_1 = \{5, 6\}$. With this move, the Duplicator has to choose any subset that contains the state 11 for the set U_2. Assume that the Duplicator chooses $U_2 = \{11, 9, 10\}$, which is valid since the probability of reaching from state 4 to any state in U_1 is less than or equal to the probability of reaching a state in U_2 from 8. The Spoiler then chooses state $s' = 11$, but the Duplicator cannot make a valid response and loses. This occurs because there is no state $t' \in U_1$ that has the same label as the states in U_2. Hence, the Spoiler successfully demonstrates to the Duplicator that states 4 and 8 are not bisimilar.

If the game starts in the state pair $(0, 4)$. There are many choices that both the Spoiler and the Duplicator can make, but none lead to the Spoiler winning the game. For instance, if the Spoiler plays on state 0 and chooses $U_1 = \{1, 2, 3\}$, the Duplicator will respond by playing on state 4 and choosing $U_2 = \{5, 6, 7\}$, which is valid because the probability of reaching from state 0 to any state in U_1 is less than or equal to the probability of reaching a state in U_2 from 4. Assume the Spoiler continues by playing on state 7, and the Duplicator plays on state 3. The Spoiler then selects, for example, $U_1 = \{3\}$, and the Duplicator wins immediately by playing $U_2 = \{3\}$, which is valid since the probability of reaching from state 7 to any state in U_1 is less than or equal to the probability of reaching a state in U_2 from 3. The game stays forever in state $(3, 3)$, and the Spoiler loses. Hence, the Duplicator successfully demonstrates to the Spoiler that states 0 and 4 are bisimilar.

Clerc, Fijalkow, Klin, and Panangaden [22] propose a slightly simpler game that characterizes probabilistic bisimilarity. If the game is in state-pair (s, t) then Spoiler chooses a set of states U such that the probability of transitioning from state s to a state in U is different from the probability of transitioning from state t to a state in U. If no such choice exists then Spoiler loses the game. Subsequently, Duplicator picks a state u that is in U and a state v that is not in U (preferably with the same label as u – otherwise Duplicator loses), and the game continues in the state pair (u, v). Clerc, Fijalkow, Klin, and Panangaden [22] showed that states s and t are probabilistic bisimilar if and only if a pair of states with different labels is never reached when the game is started in state pair (s, t) no matter how Spoiler plays.

Again, we first consider the state pair $(4, 8)$. Spoiler's first move is $U = \{3, 7, 11\}$ which is valid because the probability of reaching from state 4 to any state in U is different from that of state 8. With this move, the Duplicator cannot make a valid response and loses. This is because there is no state $t' \notin U$ that has the same label as the states in U. Hence, the Spoiler successfully shows to Duplicator that states 4 and 8 are not bisimilar.

Next, we consider the state pair $(0, 4)$. A possible Spoiler's first move is $U = \{1, 2, 5\}$ since the probability of reaching from state 0 to any state in U is different from that of state 4, but this move allows Duplicator to play $(1, 6)$. The Spoiler may survive longer by choosing $U = \{0\}$, which is a valid choice because the probability of reaching from state 1 to any state in U is different from that of state 6, but this also allows Duplicator to play $(0, 4)$, returning to the original position. For instance, if the Spoiler chooses $U = \{3\}$, which is valid since the probability

of reaching from state 0 to any state in U is different from that of state 4, Duplicator wins immediately by playing $(3, 7)$. At this point, Spoiler cannot make a move and loses the game. Hence, the Duplicator successfully shows Spoiler that the states 0 and 4 are bisimilar.

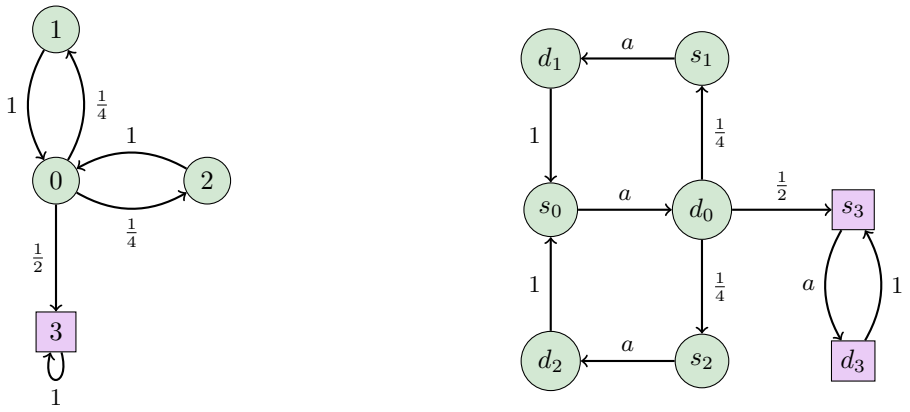


Figure 1.8: pLTS from LMC of Figure 1.5

Forejt, Jancar, Kiefer, and Worrell [24] described a game based on more general probabilistic models, such as probabilistic LTSs (pLTS). To play their game on the LMC in Figure 1.5, we first need to map it into a pLTS. This is done by duplicating all the states in the LMC and adding a transition labelled 'a' from the duplicate states to the original state, as shown in a partial view in Figure 1.8. Similarly, their game consists of two players: the Spoiler and the Duplicator. The result of the game played on the pLTS will still apply to the original LMC. Moreover, the game played on the pLTS is exactly the same as the one described by Desharnais, Laviolette, and Tracol. Ford, Beohar, König, Milius, and Schröder [40] consider an even more general setting.

1.5.3 Probabilistic Bisimilarity Distances

König and Mika-Michalski [39] generalize the game of Desharnais, Laviolette, and Tracol in two dimensions. Firstly, they consider distances which provide a generalization of equivalences [3]. Secondly, they present a general framework based on the category of sets and functions, an endofunctor on that category, and coalgebras of that endofunctor. Below, we describe the game resulting from an instantiation of their framework so that it is applicable to probabilistic bisimilarity distances for LMCs. The game starts in a triple (s, t, ϵ), where s and t are states and $\epsilon \in [0, 1)$. Spoiler tries to show that the distance between (s, t) is greater than ϵ, whereas Duplicator tries to demonstrate that the distance between (s, t) is less than or equal to ϵ. Like all the games we discussed before, the game is played in rounds. Each round consists of the following steps. Spoiler chooses a state $u_1 \in \{s, t\}$ and a fuzzy set of states U_1. Duplicator uses the other state $u_2 \in \{s, t\} \setminus \{u_2\}$ and chooses a fuzzy set of states U_2 such that the expectation of transitioning from u_1 to U_1 minus expectation of transitioning from u_2 to U_2 is at most ϵ. Such a U_2 always exists. Spoiler chooses $i \in \{1, 2\}$ and a state s'. Duplicator chooses a state t' with $U_i(s') \leq U_{3-i}(t')$. The duplicator can always choose a U_2 in the second step so that it can pick a t' in the fourth step with $U_i(s') \leq U_{3-i}(t')$. The next round starts in $(s', t', U_{3-i}(t') - U_i(s'))$. As shown by König and Mika-Michalski [39, Theorem 35 and 38], the distance between (s, t) is less than or equal to ϵ if and only if Duplicator can avoid ever reaching a state pair with different labels when the game is started in (s, t, ϵ), no matter how Spoiler plays.

Consider the LMC of Figure 1.7. The states 1 and 2 have distance $\frac{1}{18}$. Let us start the game in $(1, 2, \frac{1}{18})$. Assume that Spoiler chooses, for example, state 2 and the fuzzy set U_1 that maps state 4 to one and all other states to zero. The duplicator uses state 1 and chooses the same fuzzy set. No matter which state Spoiler chooses, Duplicator can pick the same state, and the next round starts in $(s, s, 0)$ for some state s. By considering all possible choices of Spoiler we can define a strategy of Duplicator that avoids ever reaching a state pair with different labels. This strategy explains why the distance of states 1 and 2 is at most $\frac{1}{18}$.

Next, we start the game in $(1, 2, \frac{1}{19})$. Spoiler chooses state 1 and the fuzzy set U_1 maps state 4 to one and all other states to zero. Duplicator uses state 2. To ensure that the fuzzy set U_2 satisfies the condition mentioned

[3] Desharnais et al. consider equivalence relations indexed by $\epsilon \in [0, 1]$. Above, we presented their approach for $\epsilon = 0$.

in the second step, the Duplicator has to choose a U_2 such that $U_2(5) > 0$. Spoiler chooses $i = 2$ and state 5. To satisfy the condition mentioned in the fourth step, the Duplicator has to choose state 4. Hence, the Duplicator cannot avoid reaching a state pair with different labels. This strategy of Spoiler explains why the distance of states 1 and 2 is greater than $\frac{1}{19}$.

König, Mika-Michalski, and Schröder [40] present generic algorithms for computing strategies of both Spoiler and Duplicator.

Komorida, Katsumata, Hu, Klin, Humeau, Eberhart, and Hasuo [38, Table 2] generalize the game of Fijalkow, Klin, and Panangaden to a quantitative setting using fibrations and coalgebras. Instantiating their framework to our setting amounts to the following game.

The game starts in (s, t, ϵ). Spoiler chooses a fuzzy subset of states U such that the expectation of transitioning from s to U and the expectation of transitioning from t to U differ by more than ϵ. Such a U may not always exist. If U does not exist, Spoiler loses. Duplicator chooses states s' and t' with the same label as well as $\epsilon' \in [0, 1)$ such that $U(s')$ and $U(t')$ differ by more than ϵ'. As shown in [38, Theorem 5.11], the distance of states s and t is at most ϵ if and only if Duplicator can avoid ever reaching a state pair with different labels or Spoiler gets stuck when the game is started in (s, t, ϵ), no matter how Spoiler plays.

To illustrate this game we also consider the LMC of Figure 1.7. Let us start the game in $(1, 2, \frac{1}{18})$. Since $\left(\frac{1}{2}U(4) + \frac{1}{2}U(5)\right) - \left(\frac{4}{9}U(4) + \frac{5}{9}U(5)\right) = \frac{1}{18}\left(U(5) - U(4)\right)$, we can conclude that Spoiler cannot find a U satisfies the above mentioned condition. Hence, Spoiler gets stuck. This explains why the distance of 1 and 2 is at most $\frac{1}{18}$.

Next, we start the game in $(1, 2, \frac{1}{19})$. Spoiler chooses the fuzzy subset U that maps state 5 to one and all other states to zero. This choice satisfies the above condition. The duplicator cannot avoid a state pair with different labels since for any other choice of s' and t', $U(s') = U(t')$. This strategy of Spoiler explains that the distance of 1 and 2 is greater than $\frac{1}{19}$.

1.6 Our Game

The foundation of the game studied in this thesis is an alternative characterization of the distances provided by Chen, Van Breugel, and Worrell [6, Theorem 8]. This characterization forms the basis of the algorithm developed by Bacci, Bacci, Larsen, and Mardare [1] to compute these distances. Tang and Van Breugel [53] demonstrated that their algorithm is a specific instance of Howard's policy iteration [30]. Howard's general algorithm is applicable to Markov decision processes, and Tang [52] defined the specific Markov decision processes required for Howard's policy iteration algorithm to compute the distances. A Markov decision process is a $1\frac{1}{2}$-player game: one regular player and randomness, which accounts for the remaining half. In our game, the player's role is somewhat similar to that of Duplicator. For example, assume that two states have distance zero, that is, they are probabilistic bisimilar. Then the player matches the transitions such that 1-pairs are never reached.

Similar to Stirling's bisimilarity game described in Section 1.5.1, our game involves matching transitions. However, in our setting, we match parts of transitions. For example, consider the transitions between states 1 and 2 in Figure 1.7. The transition from state 2 to state 4 can be matched by a part (with probability $\frac{4}{9}$) of the transition from state 1 to state 4. Likewise, the transition from state 1 to state 5 can be matched by a part (with probability $\frac{1}{2}$) of the transition from state 2 to state 5. The remaining parts — a portion of the transition from state 1 to state 4 and a portion of the transition from state 2 to state 5, each with probability $\frac{1}{18}$ — can also be matched. This matching is illustrated in Figure 1.9.

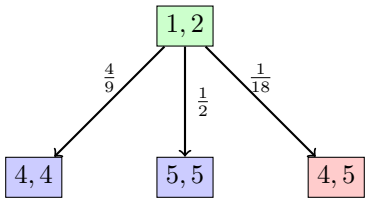


Figure 1.9: A coupling of the transitions of states 1 and 2 of the LMC of Figure 1.7

These matchings are also known as couplings, a concept introduced by Doebling [18]. The collection of couplings is referred to as the transportation polytope. While there are infinitely many ways to match parts of

transitions, as we will demonstrate, there exists a finite set of couplings from which all other couplings can be derived as convex combinations of those in the finite set (the vertices of the transportation polytope).

Let us start our game in (s, t). The player chooses a coupling of the transitions of s and t. Randomly, respecting the probabilities associated with the coupling, a matching of parts of transitions is chosen. This matching takes the game to (s', t'). If the states s' and t' are probabilistic bisimilar, in which case we call (s', t') a 0-pair (depicted in blue) as they have distance zero, or have different labels, in which case we call (s', t') a 1-pair (depicted in red), the game stops. Otherwise, the game continues in (s', t') (depicted in green). The objective of the player is to minimize the probability of reaching a 1-pair. Also, in this case, the player tries to avoid reaching 1-pairs.

Like all the other games presented above, this game is elegant, as in the quote of Fijalkow et al. in that it characterizes a global property of behaviours (the distances) through a game whose rules depend solely on local considerations (the couplings).

A strategy for the player, also known as a policy in this context, involves choosing a coupling based on local transitions of each pair of states that is neither a 0- nor a 1-pair. The policies are deterministic and memoryless. A policy is optimal if it minimizes the probability of reaching a 1-pair. The value of such a policy equals the distance between states. Thus, optimal policies capture a global property of the system. The coupling shown in Figure 1.9 is part of an optimal policy for the LMC in Figure 1.7. Notice that the probability of reaching a 1-pair is $\frac{1}{18}$, the distance of 1 and 2. As we will see through many examples, optimal policies are not unique. This thesis aims to identify the key differences between them and determine which type of optimal policy explains the distance better.

1.7 Vertex Policies

In Figure 1.10, we present an LMC and two policies. States 0 and 1 have distance one. Both policies in Figure 1.10 reach 1-pairs with probability one and, therefore, are optimal. The policy on the left is not a vertex policy, whereas the one on the right is. A policy is a vertex if and only if it matches the transitions of two states in a way that does not create a cycle in the matched target state pairs. In this example, the left policy matches state 0 with state 1, state 1 with state 2, state 2 with state 3, and state 3 with state 0, thus forming a cycle. On the other hand, the right policy does not contain a cycle; hence, it is a vertex policy. Generally, vertex policies are simpler and help us understand the distance between states more easily. In Chapter 4, Section 4.2, we will explore vertex policies further and study their properties.

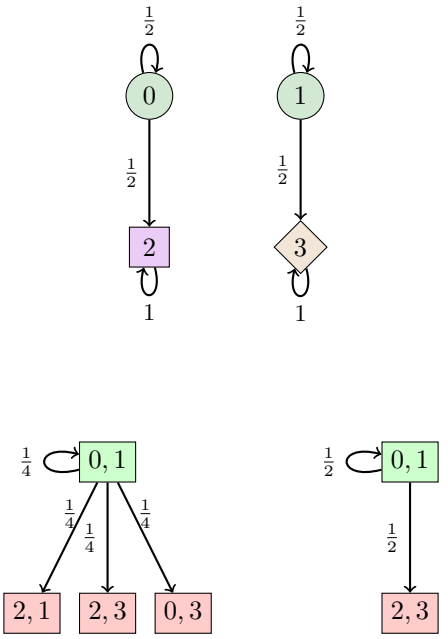


Figure 1.10: An LMC and two optimal policies

1.8 1-Maximal Policies

In Figure 1.11, we present an LMC and two policies. States 0 and 1 have distance one. Both policies in Figure 1.11 reach 1-pairs with probability one and, therefore, are optimal. The policy on the left matches the transitions of states 0 and 1 so that all target pairs are 1-pairs whereas one on the right matches the transitions so that none of the target pairs are 1-pairs.

Since policies describe the similarity between states, the left policy indicates that states 0 and 1 behave in completely different ways, while the right policy suggests that states 0 and 1 behave somewhat similarly. However, by analyzing states 0 and 1 in this LMC, we observe that they actually behave in a similar way, that is, we can match the transitions so that we cannot observe a difference after one transition. Therefore, we prefer the right policy.

An alternative justification is that the player's objective is to avoid reaching 1-pairs. Since the right policy takes longer to reach 1-pairs, we prefer it. For the policy on the right the expected length to 1-pairs is greater than for the one on the left. Hence, among the optimal policies we look for ones that maximize the expected length to 1-pairs. We call these optimal policies 1-maximal. 1-Maximal policies also determine how long the player can play the game before reaching 1-pairs.

In Chapter 6, we present an algorithm that, starting from an optimal policy computed using the algorithm by Tang and Van Breugel [53], constructs a 1-maximal optimal policy. In Section 6.1 of the same chapter, we prove an exponential lower bound for this algorithm by constructing an LMC of size $O(n)$, for which the algorithm requires $\Omega(2^n)$ iterations.

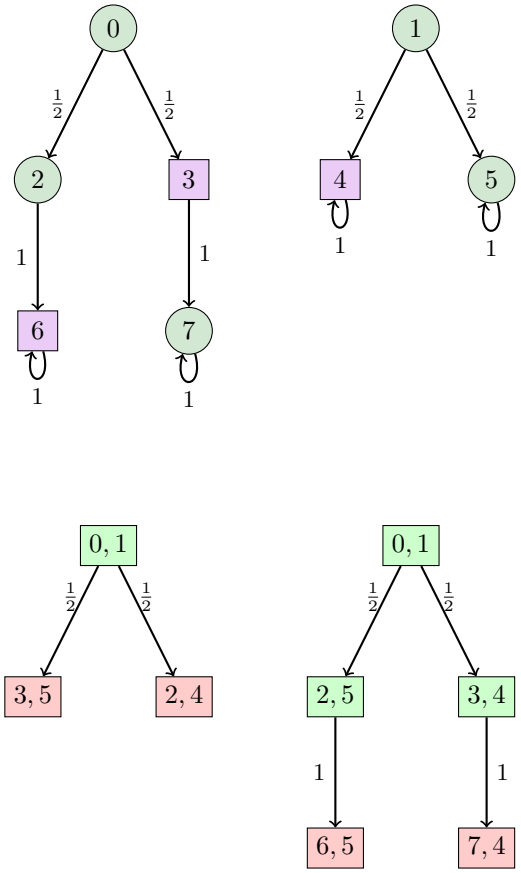


Figure 1.11: An LMC and two optimal policies

1.9 0-Minimal Policies

In Figure 1.12, we present an LMC along with two policies. The distance of states 0 and 1 is $\frac{1}{2}$. Since both policies reach a 1-pair with a probability $\frac{1}{2}$, they are optimal. The expected[4] length to a 1-pair is two for both policies, which is the maximum among all optimal policies. Therefore, both are 1-maximal. Recall that the player's objective is to avoid reaching 1-pairs. Once a policy reaches a 0-pair, it can no longer reach 1-pairs; hence, 1-pairs can be entirely avoided. As a result, in addition to avoiding 1-pairs, a secondary objective is to reach 0-pairs as soon as possible. The right policy reaches a 0-pair sooner than the left policy, as its expected length to reach 0-pairs is smaller among the 1-maximal optimal policies. We refer to such optimal policies as 0-minimal. 0-Minimal policies also determine how long the player can play the game before reaching 0-pairs. In Chapter 7, we present an algorithm that, starting from a 1-maximal optimal policy, computes a 1-maximal optimal policy that is also 0-minimal.

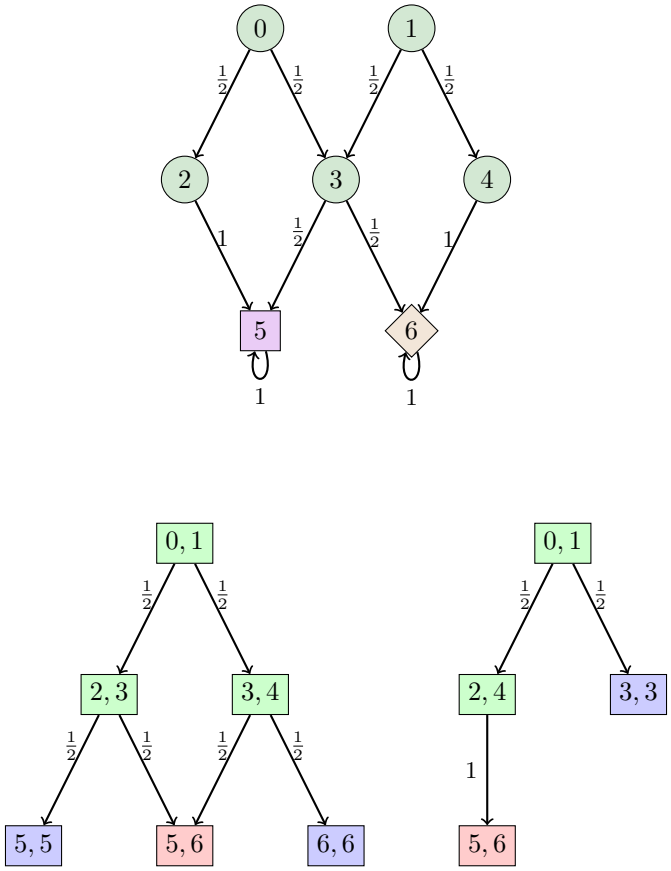


Figure 1.12: An LMC and two optimal policies

1.10 Conflict-Free Policies

Vlasman introduced the concept of conflict-free policies in [58], distinguishing between two types: label conflict-free and probabilistic bisimilar conflict-free. Furthermore, she showed that these conflicts can be removed by modifying the policy. In the following sections, we will discuss each one of them.

1.10.1 1-Conflict

In Figure 1.13, we present an LMC along with two policies. The distance of states 0 and 1 is one. Since both policies reach a 1-pair with a probability one, they are optimal. The policy on the left has a 1-conflict because

[4]More precisely, this is a conditional expectation, as we only consider paths that reach 1-pairs.

it matches two transitions whose target states have different labels (states 3 and 4), even though an alternative matching exists of two transitions with target states with the same label (as shown in the right policy). Vlasman studied this phenomenon in [58] and referred to it as label conflict. As there is a connection between label conflict free policies and 1-maximal policies, we refer to label conflict as 1-conflict. In Chapter 6, Section 6.2, we will show that a 1-maximal optimal policy has no 1-conflicts.

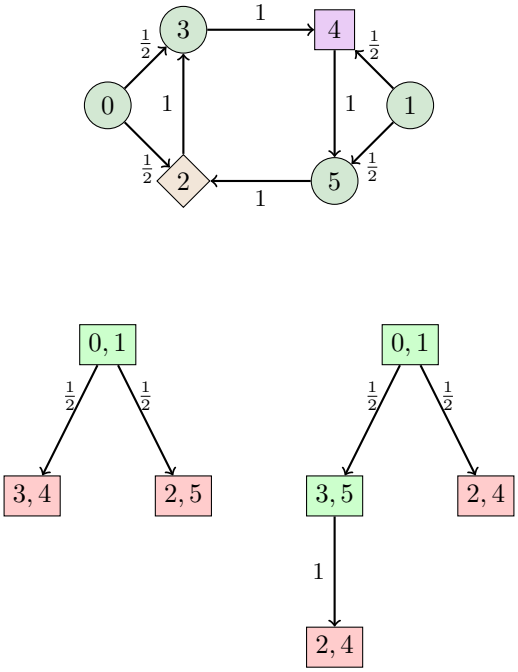


Figure 1.13: An LMC and two optimal policies

1.10.2 0-Conflict

In Figure 1.14, we present an LMC along with two policies. The distance of states 0 and 1 is $\frac{1}{2}$. Since both policies reach a 1-pair with a probability $\frac{1}{2}$, they are optimal. The policy on the left has 0-conflict because it matches two transitions whose target states are not probabilistically bisimilar (states 3 and 4), even though an alternative matching exists of two transitions with target states that are probabilistically bisimilar (as shown in the right policy). Vlasman also studied this phenomenon in [58] and referred to it as probabilistic bisimilar conflict, which we call 0-conflict. It remains an open problem whether a 0-minimal 1-maximal optimal policy is 0-conflict free.

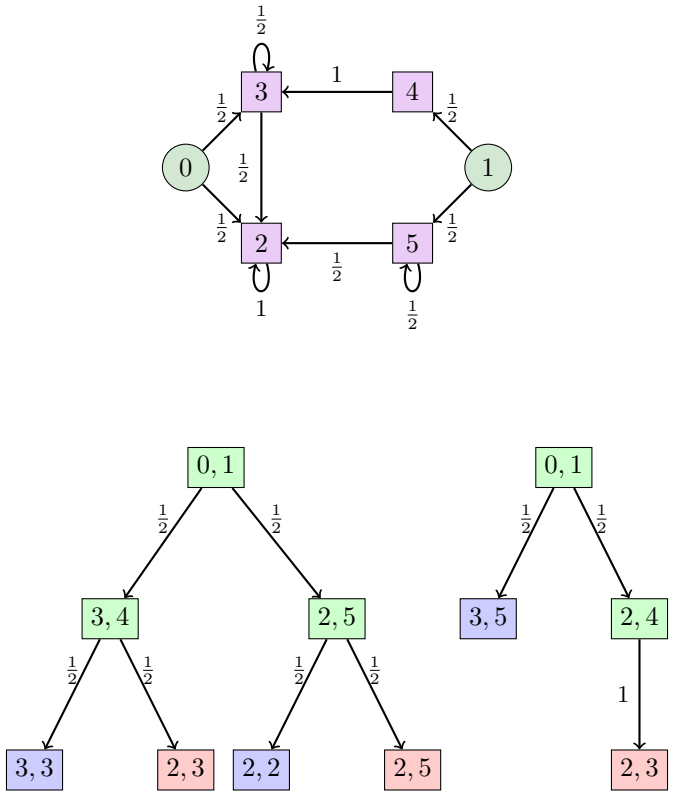


Figure 1.14: An LMC and two optimal policies

1.11 Symmetric Policies

In Figure 1.15, we present an LMC along with three policies. The distance between states 0 and 1 is one. Since all the policies reach a 1-pair with probability one, they are optimal. All the policies are vertex policies, as they match the transitions of two states in a way that does not create a cycle in the matched state pairs. Moreover, all the policies are 1-maximal and 0-minimal, as the expected lengths to 1-pairs and 0-pairs are the maximum and minimum, respectively, among all optimal policies. In the first one, the distance of 3 and 4 is explained differently from the distance of 4 and 3. Distances are symmetric. In the second policy, the explanation of the distance of 3 and 4 and that of 4 and 3 are mirror images. As a result, we consider the second policy simpler. To reflect that the one is a mirror image of the other, we introduce twisted arrows. A twisted arrow from $(0, 1)$ to $(3, 4)$ represents an arrow from $(0, 1)$ to $(4, 3)$ in the third policy. In Chapter 8 we will define symmetric policies and explore their properties.

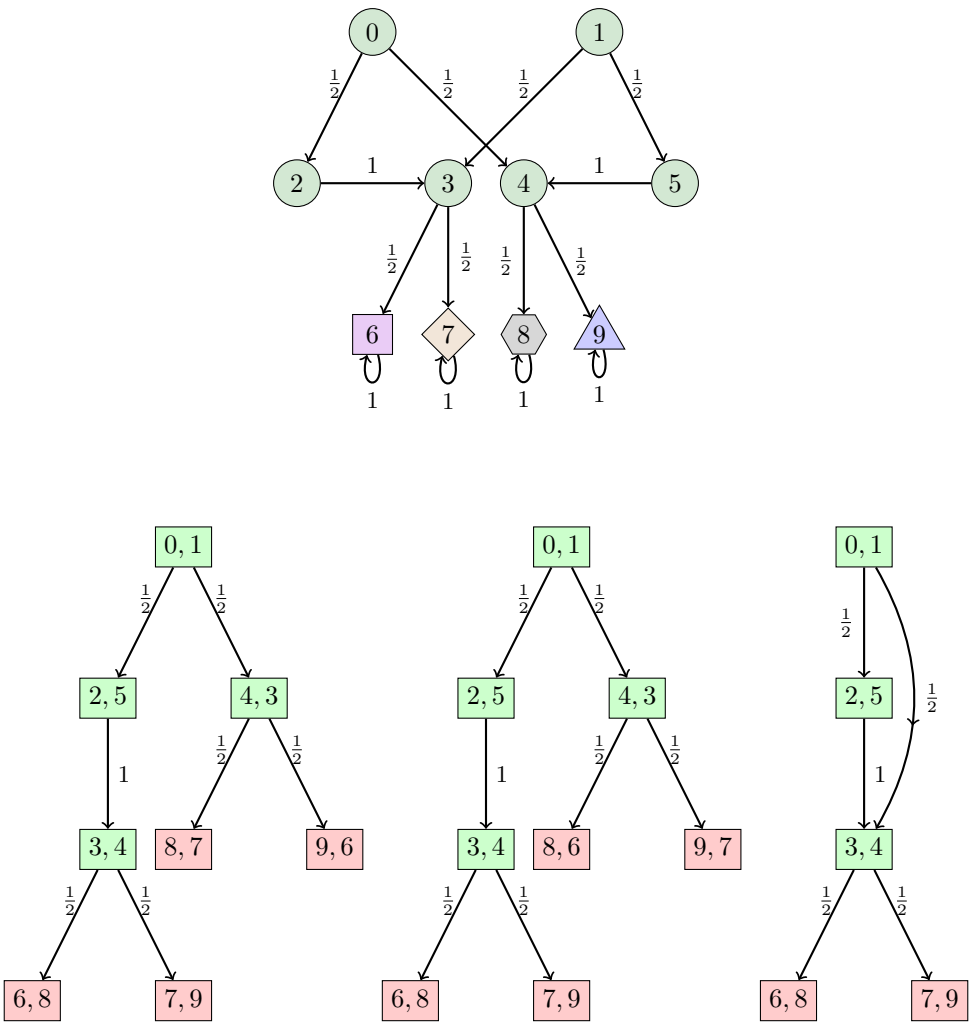


Figure 1.15: An LMC and three optimal policies

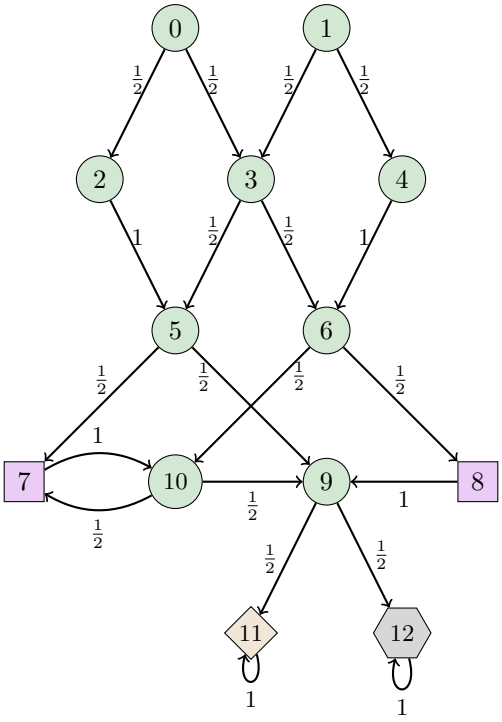


Figure 1.16: An LMC

1.12 All Policy Types

Consider the five policies for states 0 and 1, as presented in Figure 1.17, of the LMC depicted in Figure 1.16. All five policies are optimal, as each reaches a 1-pair with probability $\frac{1}{2}$, which represents the distance between states 0 and 1. Policy A is not a vertex policy, whereas policy B is. As mentioned earlier, non-vertex policies match the transitions of two states in a way that creates a cycle in the matched state pairs. In this example, policy A matches state 2 with state 3, state 3 with state 4, and state 4 with state 2, thus forming a cycle. In contrast, policy B does not contain a cycle and is therefore simpler. Recall that the player's objective is to avoid reaching 1-pairs. Policy C performs better than policy B in this regard, as the expected length to 1-pairs is greater for policy C than for policy B. Thus, policy C is preferable. The player's second objective is to reach 0-pairs as early as possible. Policy D achieves this better than policy C, as the expected length to 0-pairs is smaller for policy D than for policy C. Therefore, policy D is more desirable. In policy D, the explanation of the distance between states 10 and 9 differs from that between states 9 and 10. Recall that distances are symmetric. In policy E, the explanations for the distances between 10 and 9 and between 9 and 10 are mirror images, hence, symmetric. As a result, we consider policy E the most preferable.

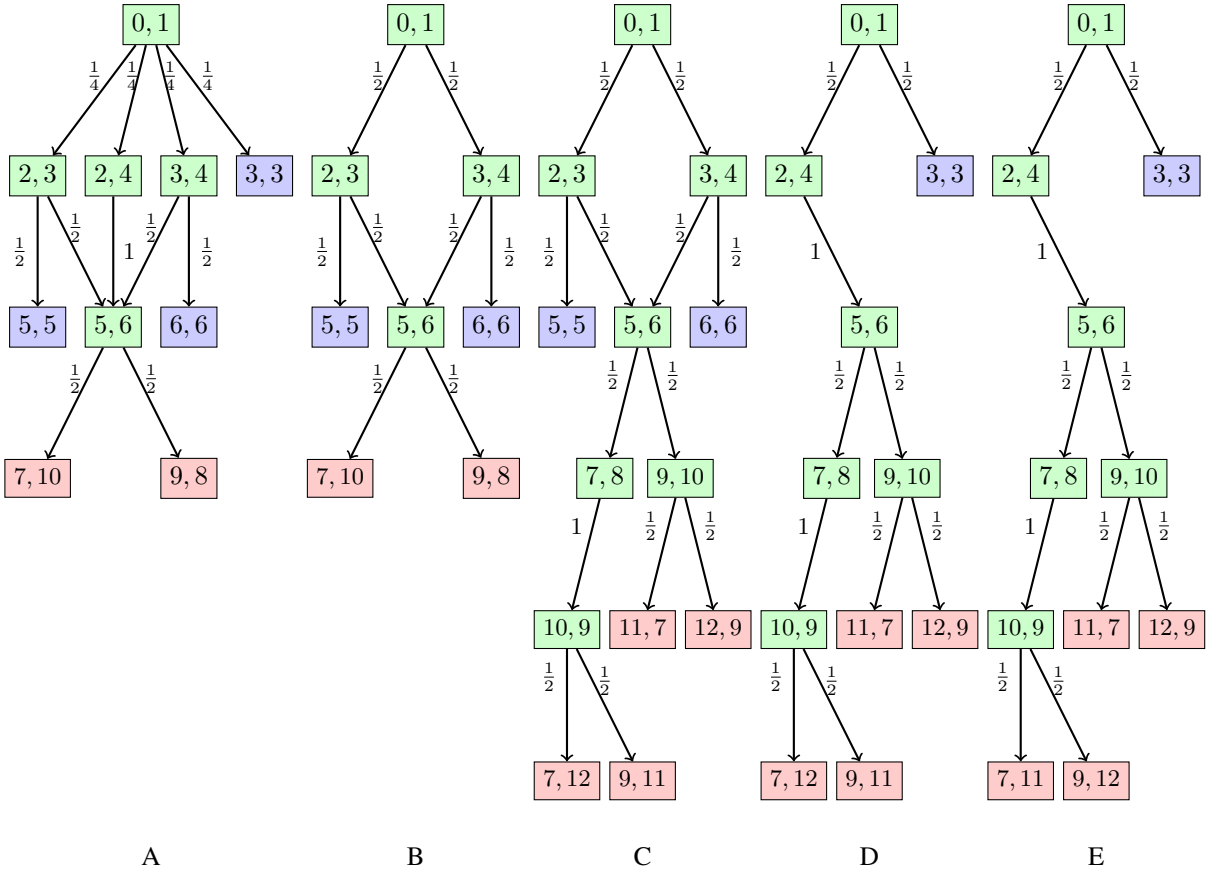


Figure 1.17: Five optimal policies of states 0 and 1 of LMC depicted in Figure 1.16: non-vertex, vertex, vertex 1-maximal, vertex 1-maximal 0-minimal, and vertex 1-maximal 0-minimal symmetric

1.13 Contributions

The main contributions of this thesis are the following.

1. We provide a unified overview of games related to probabilistic bisimilarity and probabilistic bisimilarity distances.
2. We argue that 0-minimal 1-maximal vertex optimal policies explain these distances.
3. We prove that 1-maximal vertex optimal policies are free of 1-conflicts.
4. We develop algorithms to compute 0-minimal 1-maximal vertex optimal policies and prove their correctness.
5. We prove an exponential lower bound for the algorithm that computes a 1-maximal vertex optimal policy from an optimal policy.
6. We define the notion of a policy being symmetric and check that there exist symmetric policies that are 0-minimal 1-maximal vertex optimal.
7. We implemented all algorithms in Java.

Of the above contributions, 1, 2, and 6 are joint work with van Breugel, and 3 is joint work with Vlasman, van Breugel, and Worrell.

2 Fixed Point Theorems

In this chapter, we will review the concept of fixed points and highlight several important fixed point theorems from the literature, like the Knaster-Tarski and Banach fixed point theorems. As we will see in the following chapter, we define probabilistic bisimilarity distances in terms of the fixed point of a function, and these theorems play a crucial role in explaining them. We will begin by briefly covering some important concepts from order theory and topology, as they apply to these theorems. For instance, the Knaster-Tarski fixed-point theorem relies on complete lattices, while the Banach fixed-point theorem relies on complete metric spaces. We will present each of these theorems in turn and examine their applications throughout the thesis.

2.1 Knaster-Tarski Fixed Point Theorem

In this thesis, we study distances on a finite set S of states. These distances are bounded by one and can be represented by the functions $d : S \times S \to [0, 1]$ [5]. These functions carry a natural order. For distance functions d, $e \in S \times S \to [0, 1]$, we order them by defining $d \sqsubseteq e$ if for all s, $t \in S$, $d(s, t) \leq e(s, t)$. This defines a partial order. For the definition of a partial order, we refer the reader to, for example, [11, Definition 1.2]. The real numbers $\mathbb{R}$, ordered by the standard less-than-or-equal relation $\leq$, provide another example of a partially ordered set, which is used in many of the examples throughout this chapter. As we will see in the following chapter, we define the probabilistic bisimilarity distances as a fixed point of a function.

Definition 2.1. *Let $f : X \to X$ be a function. Then $x \in X$ is a* fixed point *of f if $f(x) = x$.*

For example, if $f : \mathbb{R} \to \mathbb{R}$ is defined by $f(x) = x^2 - 3x + 4$, then 2 is a fixed point of f because $f(2) = 2$. Similarly, if $f : \mathbb{R} \to \mathbb{R}$ is defined by $f(x) = x^2 - 2$, then both 2 and -1 are fixed points of f, since $f(2) = 2$ and $f(-1) = -1$. In contrast, the function $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = x + 1$ has no fixed points at all, since $x + 1$ is always greater than x.

In our case, the space X is the set $S \times S \to [0, 1]$. The corresponding function will be $\Phi : (S \times S \to [0, 1]) \to (S \times S \to [0, 1])$, that is, given a distance function d, we have that $\Phi(d)$ is a distance function as well. Since we have an ordering on the distance functions, we can order the fixed points of these functions Φ.

Definition 2.2. *Let $(X, \sqsubseteq)$ be a partially ordered set and $f : X \to X$. Then $x \in X$ is a* pre-fixed point *of f if $f(x) \sqsubseteq x$ and a* post-fixed point *of f if $x \sqsubseteq f(x)$.*

For example, if the function $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = \frac{x}{2}$, then 2 is a pre-fixed point of f because $f(2) \leq 2$. If the function $f : \mathbb{R} \to \mathbb{R}$ is defined by $f(x) = 2x$, then 1 is a post-fixed point of f because $1 \leq f(1)$.

We are also interested in the least and greatest fixed points of these functions, if they exist.

Definition 2.3. *Let $(X, \sqsubseteq)$ be a partially ordered set, $f : X \to X$, and $x \in X$. Then x is a* least fixed point *of f if*

$$f(x) = x \text{ and}$$
$$\text{for all } y \in X, \text{ if } f(y) = y \text{ then } x \sqsubseteq y$$

and x is a greatest fixed point *of f if*

$$f(x) = x \text{ and}$$
$$\text{for all } y \in X, \text{ if } f(y) = y \text{ then } y \sqsubseteq x.$$

[5] In this thesis, we also consider functions in $S \times S \to [0, \infty]$. Results similar to the ones presented in this chapter hold for these functions as well.

For example, the function $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = x^2$ has 0 as its least fixed point and 1 as its greatest fixed point, since 0 and 1 are the only fixed points of f, and $0 \leq 1$. On the other hand, the function $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = x$ has infinitely many fixed points, but it does not have a least and a greatest fixed point.

As the above example shows, not every function has a least or a greatest fixed point. As we will show in Theorem 2.7, if function f is monotone then a least and a greatest fixed point exist.

Definition 2.4. *A function $f : X \to X$ is* monotone *if for all $x, y \in X$, we have*

$$x \sqsubseteq y \implies f(x) \sqsubseteq f(y).$$

For example, the function $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = x^3$ is a monotone function, whereas the function $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = \sin(x)$ is not a monotone function.

Definition 2.5. *Let $(X, \sqsubseteq)$ be a partially ordered set, $A \subseteq X$, and $x \in X$. Then x is the* least upper bound *of A, denoted by $\sqcup A$, if*

$$\begin{aligned} &\text{for all } a \in A, a \sqsubseteq x \text{ and} \\ &\text{for all } y \in X, \text{ if } a \sqsubseteq y \text{ for all } a \in A \text{ then } x \sqsubseteq y, \end{aligned}$$

and x is the greatest lower bound *of A, denoted by $\sqcap A$, if*

$$\begin{aligned} &\text{for all } a \in A, x \sqsubseteq a \text{ and} \\ &\text{for all } y \in X, \text{ if } y \sqsubseteq a \text{ for all } a \in A \text{ then } y \sqsubseteq x. \end{aligned}$$

For example, if the subset $A \subseteq \mathbb{R}$ is defined by $A = [0, 1)$, then $\sqcup A = 1$. On the other hand, if the subset $A = \mathbb{R}$, then A does not have a least upper bound or a greatest lower bound. Additionally, if the subset $A \subseteq \mathbb{R}$ is defined by $A = (0, 1]$, then $\sqcap A = 0$.

Given a set of distance functions $D \subseteq S \times S \to [0, 1]$, Definition 2.5 defines its least upper bound $\sqcup D$ and its greatest lower bound $\sqcap D$. This least upper bound $\sqcup D$ and the greatest lower bound $\sqcap D$ are distance functions. For all $(s, t) \in S \times S$, we have that $(\sqcup D)(s, t) = \sup_{d \in D} d(s, t)$ and $(\sqcap D)(s, t) = \inf_{d \in D} d(s, t)$.

These distance functions, with $\sqsubseteq$, form a complete lattice $\langle S \times S \to [0, 1], \sqsubseteq \rangle$, according, for example, to [15, Lemma 3.2]. Briefly, a complete lattice is a partially ordered set in which every set of elements has both a least upper bound and a greatest lower bound. For the definition of a complete lattice, we refer the reader to, for example, [11, Definition 2.1]. The functions $\bot, \top : S \times S \to [0, 1]$, defined by $\bot(s, t) = 0$ and $\top(s, t) = 1$ for all $s, t \in S$, are the least and greatest element of this complete lattice. $\langle [0, 1], \leq \rangle$ is another example of a complete lattice.

Definition 2.6. *Let $f : X \to X$. For $n \in \mathbb{N}$, the function $f^n : X \to X$ is defined by*

$$f^n(x) = \begin{cases} x & \text{if } n = 0 \\ f(f^{n-1}(x)) & \text{otherwise.} \end{cases}$$

For example, consider the function $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = x + 2$. To compute $f^3(1)$, we apply f repeatedly starting from 1:

$$\begin{aligned} f^0(1) &= 1 \\ f^1(1) &= f(1) = 1 + 2 = 3 \\ f^2(1) &= f(f(1)) = f(3) = 3 + 2 = 5 \\ f^3(1) &= f(f(f(1))) = f(5) = 5 + 2 = 7 \end{aligned}$$

Thus, $f^3(1) = 7$.

Let $\mathbb{N}$ be the set of natural numbers. Part (a) and (b) of the following theorem are due to Tarski [54] who generalized a result of Knaster [35]. This result is known as the Knaster-Tarski fixed point theorem. In the next theorem, we use $\sqcup_{n \in \mathbb{N}} f^n(\bot)$ as a shorthand for $\sqcup\{f^n(\bot) \mid n \in \mathbb{N}\}$. The same applies to $\sqcap_{n \in \mathbb{N}} f^n(\top)$.

Theorem 2.7. *Let $(X, \sqsubseteq)$ be a complete lattice with least element $\bot$ and greatest element $\top$. If $f : X \to X$ is monotone then*

(a) f has a least fixed point, which is the least pre-fixed point of f, and

(b) f has a greatest fixed point, which is the greatest post-fixed point of f.

Furthermore,

(c) if $\sqcup_{n\in\mathbb{N}} f^n(\bot)$ is a fixed point of f then it is the least fixed point of f, and

(d) if $\sqcap_{n\in\mathbb{N}} f^n(\top)$ is a fixed point of f then it is the greatest fixed point of f.

Proof.

(a) and (b) This is the Knaster-Tarski fixed point theorem (see, for example, [11, Theorem 2.35]).

(c) We know that $\sqcup_{n\in\mathbb{N}} f^n(\bot)$ exists because $(X, \sqsubseteq)$ is a complete lattice. Assume that $\sqcup_{n\in\mathbb{N}} f^n(\bot)$ and x are fixed points of f. To conclude that $\sqcup_{n\in\mathbb{N}} f^n(\bot)$ is the least fixed point of f, it suffices to show that $\sqcup_{n\in\mathbb{N}} f^n(\bot) \sqsubseteq x$. Hence, it suffices to show that $f^n(\bot) \sqsubseteq x$ for all n. We prove this by induction on n. The base case, when $n = 0$, is immediate as $\bot \sqsubseteq x$. In the inductive case, $n > 0$. Then

$$\begin{aligned} f^n(\bot) &= f(f^{n-1}(\bot)) \\ &\sqsubseteq f(x) \qquad [f \text{ is monotone and induction hypothesis}] \\ &= x \qquad [x \text{ is a fixed point of } f] \end{aligned}$$

(d) Similar to (c).

□

Here is an example that illustrates part (c) of Theorem 2.7. Let $f : [0, 5] \to [0, 5]$ be a function defined by $f(x) = \min(x + 1, 3)$. It is easy to verify that $[0, 5]$ is a complete lattice and the least element of this lattice is $\bot = 0$. We apply f iteratively starting from the least element $\bot = 0$. Then we have $f^0(\bot) = 0$, $f^1(\bot) = 1$, $f^2(\bot) = 2$, $f^3(\bot) = 3$, $f^4(\bot) = 3$, $f^5(\bot) = 3$. Thus, the least upper bound of this sequence is 3. From Theorem 2.7(c) we can conclude that 3 is the least fixed point of f.

In the following chapters, we will see that several key functions, including the probabilistic bisimilarity distances, are defined as the least fixed points. Some others are defined as the greatest fixed points. Pre-fixed and post-fixed points are used in numerous proofs, as are parts (a), (b), (c) and (d) of Theorem 2.7.

2.2 Generalization of Banach's Fixed Point Theorem

Let S be a finite set. Recall that the *infinity norm* of a function $f \in S \times S \to \mathbb{R}$, denoted by $\|f\|$, is defined by

$$\|f\| = \max_{s,t\in S} |f(s, t)|.$$

This infinity norm defines a distance on these distance functions. For distance functions d, $e \in S \times S \to [0, 1]$, the function mapping (d, e) to $\|d - e\|$ defines a complete metric. For the definition of a complete metric space, we refer the reader to [19, Definition B7.13]. These distance functions serve as an example of a complete metric space. Similarly, the set of real numbers $\mathbb{R}$ with the Euclidean distance function provides another example of a complete metric space.

Theorem 2.10, below, guarantees the existence and uniqueness of a fixed point for a function that is contractive on a nonempty complete metric space. Moreover, this result extends to so called power-contractive and nonexpansive functions as well. Let us first define these types of functions formally.

Definition 2.8. *Let $f : X \to X$. Then f is* nonexpansive *if for all $x, y \in X$,*

$$d(f(x), f(y)) \leq d(x, y)$$

and f is contractive *if there exists $c < 1$ such that for all $x, y \in X$,*

$$d(f(x), f(y)) \leq c\, d(x, y).$$

and f is power-contractive *if f^n is contractive for some $n \in \mathbb{N}$.*

The following examples of functions are nonexpansive. The function $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = \frac{x}{2}$ is contractive, whereas the function $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = x$ is not contractive. Meanwhile, the function $f : \mathbb{R} \to \mathbb{R}$ defined by $f(x) = 2x$ is expansive.

Definition 2.9. *Let $(x_n)_{n\in\mathbb{N}}$ be a sequence in X and $x \in X$. Then $(x_n)_{n\in\mathbb{N}}$* converges *to x, denoted $\lim_{n\in\mathbb{N}} x_n = x$, if*

$$\forall \epsilon > 0 : \exists N \in \mathbb{N} : \forall n \geq N : d(x_n, x) \leq \epsilon.$$

For example, consider the sequence $\left(\frac{1}{n}\right)_{n\in\mathbb{N}}$. Using the above definition, it is easy to verify that this sequence converges to 0, i.e., $\lim_{n\in\mathbb{N}} \frac{1}{n} = 0$.

Next, we present a generalization of Banach's fixed point theorem. This result dates back at least as far as the sixties. Part (a) and (b) are Banach's original fixed point theorem [3]. Part (c) can already be found in [37] and [12] contains part (d). For completeness, we provide proof of parts (c) and (d).

Theorem 2.10. *Let (X, d) be a nonempty complete metric space. Let $f : X \to X$ and $n \in \mathbb{N}$. If f^n is contractive then*

(a) f^n has a unique fixed point x and

(b) for all $y \in X$, $x = \lim_{m\in\mathbb{N}} f^{mn}(y)$.

If f is also nonexpansive then

(c) x is the unique fixed point of f and

(d) for all $y \in X$, $x = \lim_{m\in\mathbb{N}} f^m(y)$.

Proof. Let (X, d) be a nonempty complete metric space. Let $f : X \to X$ and $n \in \mathbb{N}$. Assume that f^n is contractive.

(a) and (b) This is Banach's fixed point theorem (see, for example, [19, Theorem 2.1.5]) restated for the function f^n.

(c) First, we show that x is a fixed point of f, that is, $f(x) = x$. Since x is the unique fixed point of f^n, it suffices to show that $f(x)$ is a fixed point of f^n, that is, $f^n(f(x)) = f(x)$. Since

$$\begin{aligned} d(f^n(f(x)), f(x)) &= d(f^{n+1}(x), f(x)) \\ &= d(f(f^n(x)), f(x)) \\ &\leq d(f^n(x), x) \qquad [f \text{ is nonexpansive}] \\ &= 0 \qquad [f^n(x) = x] \end{aligned}$$

we have that $d(f^n(f(x)), f(x)) = 0$ and, therefore, $f^n(f(x)) = f(x)$.

Next, we show that x is a unique fixed point of f. Assume $f(y) = y$. Since x is the unique fixed point of f^n, it suffices to show that y is a fixed point of f^n, that is, $f^n(y) = y$.

We show that for all m, $f^m(y) = y$ by induction on m. In the base case, $m = 0$, this obviously holds. In the inductive case, $m > 0$, we have that

$$\begin{aligned} f^m(y) &= f(f^{m-1}(y)) \\ &= f(y) \qquad [\text{induction hypothesis}] \\ &= y \qquad [f(y) = y] \end{aligned}$$

Hence, $f^n(y) = y$.

(d) Let $y \in X$. It suffices to show that the sequence $(f^m(y))_{m\in\mathbb{N}}$ converges to x, that is,

$$\forall \epsilon > 0 : \exists M \in \mathbb{N} : \forall m \geq M : d(f^m(y), x) \leq \epsilon. \tag{2.1}$$

Let $\epsilon > 0$. Since the sequence $(f^{mn}(y))_{m\in\mathbb{N}}$ converges to x by part (b),

$$\exists N \in \mathbb{N} : \forall m \geq N : d(f^{mn}(y), x) \leq \epsilon. \tag{2.2}$$

We take $M = Nn$. It suffices to show that $d(f^{M+k}(y), x) \leq \epsilon$ for all $k \in \mathbb{N}$. We prove this by induction on k. The base case, $k = 0$, follows immediately from (2.2). In the inductive case, when $k > 0$, we have that

$$\begin{aligned} d(f^{M+k}(y), x) &= d(f^{M+k}(y), f(x)) && \text{[part (c)]} \\ &\leq d(f^{M+k-1}(y), x) && [f \text{ is nonexpansive}] \\ &\leq \epsilon && \text{[induction hypothesis]} \end{aligned}$$

□

Theorem 2.10 is used in numerous proofs throughout this thesis to show that functions, including the function that defines probabilistic bisimilarity distances, have unique fixed points.

2.3 Fixed Point Characterization Theorem

The following result, which is based on the order and metric of the set $S \times S \to [0, 1]$, is a variation on [57, Theorem 1]. This theorem is used in several proofs throughout this thesis to characterize the least and the greatest fixed point of a function.

Theorem 2.11. *If $\Phi : (S \times S \to [0, 1]) \to (S \times S \to [0, 1])$ is monotone and nonexpansive then*

(a) $\lim_{n\in\mathbb{N}} \Phi^n(\bot)$ is the least fixed point of Φ, and

(b) $\lim_{n\in\mathbb{N}} \Phi^n(\top)$ is the greatest fixed point of Φ.

Proof. Assume that $\Phi : (S \times S \to [0, 1]) \to (S \times S \to [0, 1])$ is monotone and nonexpansive.

(a) We show that $\lim_{n\in\mathbb{N}} \Phi^n(\bot) = \sqcup_{n\in\mathbb{N}} \Phi^n(\bot)$ and $\lim_{n\in\mathbb{N}} \Phi^n(\bot)$ is a fixed point of Φ. The desired result follows from Theorem 2.7(c).

First, we observe that the sequence $(\Phi^n(\bot))_{n\in\mathbb{N}}$ is non-decreasing by showing that for all $n \in \mathbb{N}$, $\Phi^n(\bot) \sqsubseteq \Phi^{n+1}(\bot)$ by induction on n. The base case, $n = 0$, follows from the fact that $\bot$ is the least element. In the induction case, $n > 0$, we have that $\Phi^{n-1}(\bot) \sqsubseteq \Phi^n(\bot)$ by the induction hypothesis and, hence, $\Phi^n(\bot) \sqsubseteq \Phi^{n+1}(\bot)$ since Φ is monotone. For $s, t \in S$, the non-decreasing and bounded above sequence $(\Phi^n(\bot)(s, t))_{n\in\mathbb{N}}$ converges to its supremum, that is, $\lim_{n\in\mathbb{N}} \Phi^n(\bot)(s, t) = \sup_{n\in\mathbb{N}} \Phi^n(\bot)(s, t)$. Hence,

$$\begin{aligned} \left(\lim_{n\in\mathbb{N}} \Phi^n(\bot)\right)(s, t) &= \lim_{n\in\mathbb{N}} \Phi^n(\bot)(s, t) \\ &= \sup_{n\in\mathbb{N}} \Phi^n(\bot)(s, t) \\ &= \sqcup_{n\in\mathbb{N}} \Phi^n(\bot)(s, t) \\ &= \left(\sqcup_{n\in\mathbb{N}} \Phi^n(\bot)\right)(s, t). \end{aligned}$$

It remains to show that $\lim_{n\in\mathbb{N}} \Phi^n(\bot)$ is a fixed point of Φ:

$$\begin{aligned} \Phi\left(\lim_{n\in\mathbb{N}} \Phi^n(\bot)\right) &= \lim_{n\in\mathbb{N}} \Phi(\Phi^n(\bot)) && [\Phi \text{ is nonexpansive and, hence, continuous}] \\ &= \lim_{n\in\mathbb{N}} \Phi^n(\bot) \end{aligned}$$

(b) Similar to (a).

□

In this chapter, we briefly covered fixed points and studied three key theorems about fixed points. In the next chapter, we will define probabilistic bisimilarity distances as a fixed point of a function and study their properties.

3 Probabilistic Bisimilarity and Probabilistic Bisimilarity Distances

In this chapter, we will review the model of interest, labelled Markov chains (LMCs), and the notion of couplings, a concept from probability theory. Roughly speaking, couplings are the matching of transitions of an LMC. As mentioned in the introduction of this thesis, these couplings play a key role in our game, as they represent the player's strategy. We will begin by defining the set of couplings, known as the transportation polytope, and demonstrating how they can be interpreted as transportation plans. Next, we will define probabilistic bisimilarity in terms of these couplings and illustrate the definition with an example. Although there are, in general, infinitely many ways to match parts of transitions, as we will show through an example, only a finite subset of these matchings are critical for our game. We will conclude the chapter by formalizing probabilistic bisimilarity distances using couplings and studying their properties.

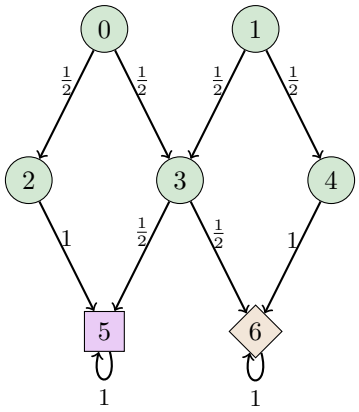


Figure 3.1: An LMC

Given a finite set X, a function $\mu : X \to [0,1]$ is a *probability distribution* on X if $\sum_{x \in X} \mu(x) = 1$. We denote the set of probability distributions on X by $\mathcal{D}(X)$. For $\mu \in \mathcal{D}(X)$ and $A \subseteq X$, we often write $\mu(A)$ for $\sum_{x \in A} \mu(x)$. Similarly, for $\omega \in \mathcal{D}(X \times X)$, $x \in X$, and $A \subseteq X$, we usually write $\omega(x, A)$ for $\sum_{a \in A} \omega(x, a)$. For $\mu \in \mathcal{D}(X)$, we define the *support* of μ by $\text{support}(\mu) = \{\, x \in X \mid \mu(x) > 0 \,\}$. We formally define LMCs as follows.

Definition 3.1. *A* labelled Markov chain *is a tuple* $\langle S, L, \tau, \ell \rangle$ *consisting of*

- *a finite set* S *of* states*,*
- *a finite set* L *of* labels*,*
- *a* transition probability function $\tau : S \to \mathcal{D}(S)$*, and*
- *a* labelling function $\ell : S \to L$*.*

We restrict our attention to LMCs with finitely many states. See Figure 3.1 for an example. In this example, $S = \{0, 1, 2, 3, 4, 5, 6\}$ and $L = \{$circle, square, diamond$\}$. The transition probability function for state 0 is defined as follows:

$$\tau(0)(0) = 0, \tau(0)(1) = 0, \tau(0)(2) = \tfrac{1}{2}, \tau(0)(3) = \tfrac{1}{2}, \tau(0)(4) = 0, \tau(0)(5) = 0, \tau(0)(6) = 0$$

It is easy to verify that $\sum_{u \in S} \tau(0)(u) = 1$. The labelling function for state 0 is $\ell(0) = \text{circle}$. Similarly, the transition probabilities and labelling functions can be defined for the remaining states.

For the remainder, we fix an LMC $\langle S, L, \tau, \ell \rangle$. Next, we introduce the notion of a *coupling* of two probability distributions and define probabilistic bisimilarity based on these couplings.

3.1 Couplings and Probabilistic Bisimilarity

We define probabilistic bisimilarity by means of the set $\Omega(\mu, \nu)$, which is the set of all couplings of the probability distributions μ and ν, known as the *transportation polytope* (see, for example, [34]).

Definition 3.2. *Let μ, $\nu \in \mathcal{D}(S)$. The set $\Omega(\mu, \nu)$ is defined by*

$$\Omega(\mu, \nu) = \{\, \omega \in \mathcal{D}(S \times S) \mid \forall s \in S : \omega(s, S) = \mu(s) \wedge \omega(S, s) = \nu(s) \,\}.$$

Next, we argue that these couplings can be viewed as transportation plans. Given probability distributions μ, $\nu \in \mathcal{D}(S)$, think of the elements of S as locations. More precisely, for each $s \in S$ there is an origin location, denoted s_o, and a destination location, s_d. At origin s_o we have a supply of $\mu(s)$ goods that we need to transport to the destinations. At the destination s_d we have a demand of $\nu(s)$ goods. Note that, since $\mu(S) = 1$ and $\nu(S) = 1$, the total supply equals the total demand. A transportation plan ω specifies how much to transport from each origin to each destination, that is, it can be viewed as a function mapping each origin-destination pair to a real number in the interval $[0, 1]$. The condition $\omega(s, S) = \mu(s)$ captures that the supply at s_o needs to be transported to the destinations. Similarly, $\omega(S, s) = \nu(s)$ captures that the demand at s_d needs to be received from all the sources.

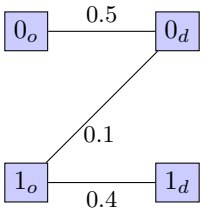
Example 3.3. *Let $S = \{0, 1\}$. The probability distributions μ and ν are given by*

$$\mu(0) = 0.5, \mu(1) = 0.5, \nu(0) = 0.6, \text{ and } \nu(1) = 0.4$$

The function $\omega : S \times S \to [0, 1]$ is defined by

$$\omega(0, 0) = 0.5, \omega(0, 1) = 0, \omega(1, 0) = 0.1, \text{ and } \omega(1, 1) = 0.4$$

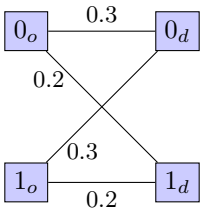
We leave it to the reader to verify that the function ω is a coupling. Such a coupling can be depicted as follows.



The function $\pi : S \times S \to [0, 1]$ is defined by

$$\pi(0, 0) = 0.3, \pi(0, 1) = 0.2, \pi(1, 0) = 0.3, \text{ and } \pi(1, 1) = 0.2$$

It is easy to verify that the function π is also a coupling. Such a coupling can be depicted as follows.



For each μ, $\nu \in \mathcal{D}(S)$, $\Omega(\mu, \nu)$ is a closed convex polytope (see, for example [45, Definition 1.1], for a definition of this notion). This set $\Omega(\mu, \nu)$ is infinite.

As we saw in Example 3.3, when we ignore the probabilities of a coupling, we obtain an undirected graph, which we refer to as *support graph*, defined as follows.

Definition 3.4. *Let $\omega \in \mathcal{D}(S \times S)$. The* support graph *of ω is (V, E) where*

- $V = S \times \{0, 1\}$ *and*
- $E = \{\, \{(s, 0), (t, 1)\} \mid \omega(s, t) > 0 \,\}$.

The next proposition shows that for each μ, $\nu \in \mathcal{D}(S)$ and $\omega \in \Omega(\mu, \nu)$, ω is a vertex in the closed convex polytope (see, for example [45, Definition 1.3], for a definition of this notion) if the support graph of ω is acyclic.

Proposition 3.5 ([34, Theorem 4])**.** *For all μ, $\nu \in \mathcal{D}(S)$ and $\omega \in \Omega(\mu, \nu)$, ω is a vertex if and only if the support graph of ω is a forest.*

For instance, the coupling ω in Example 3.3 is a vertex because its support graph does not contain cycles. In contrast, the coupling π in the same example is not a vertex, as the support graph of π contains a cycle.

We denote the *vertices* of the transportation polytope by $V(\Omega(\mu, \nu))$. Since a convex polytope is defined by a finite number of linear inequalities, and each vertex arises from a combinations of these inequalities becoming equalities, it follows that the polytope has only finitely many vertices. Therefore, this set $V(\Omega(\mu, \nu))$ is finite.

Probabilistic bisimilarity, in terms of couplings, is defined as follows.

Definition 3.6. *A relation $R \subseteq S \times S$ is a* probabilistic bisimulation *if for all $(s, t) \in R$, $\ell(s) = \ell(t)$ and there exists $\omega \in \Omega(\tau(s), \tau(t))$ with $\mathrm{support}(\omega) \subseteq R$. States s and t are* probabilistic bisimilar*, denoted $s \sim t$, if $(s, t) \in R$ for some probabilistic bisimulation R.*

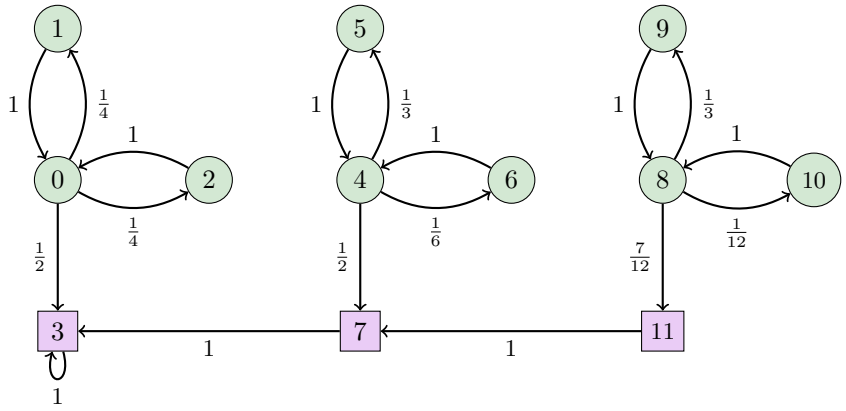


Figure 3.2: An LMC

Consider the LMC depicted in Figure 3.2. To show that the states 0 and 4 are probabilistic bisimilar in this LMC, we need to find a probabilistic bisimulation relation $\mathcal{R}$ such that $(0, 4) \in \mathcal{R}$. Consider the equivalence relation $\mathcal{R}$ with equivalent classes

$$\{1, 2, 5, 6\}, \{3, 7, 11\}, \{9, 10\}, \{8\}, \{0, 4\}$$

It can easily be verified that $\mathcal{R}$ indeed satisfies all the requirements of Definition 3.6.

Next, we define the probabilistic bisimilarity distances. Before doing so, we partition the set of state pairs into the following subsets.

Definition 3.7. *The sets S_0^2, S_1^2, and $S_?^2$ are defined by*

$$S_0^2 = \{\, (s, t) \in S \times S \mid s \sim t \,\}$$
$$S_1^2 = \{\, (s, t) \in S \times S \mid \ell(s) \neq \ell(t) \,\}$$
$$S_?^2 = (S \times S) \setminus (S_0^2 \cup S_1^2)$$

The set S_0^2 contains those state pairs that behave the same and, hence, have distance zero (see Theorem 3.11). We call these *0-pairs*. The set S_1^2 contains those state pairs that have a different label and, therefore, are fundamentally different and, hence, have distance one (see Definition 3.8). We call these *1-pairs*. The set $S_?^2$ contains

the remaining state pairs. Note that some of these state pairs may have distance one, but cannot have distance zero (see Theorem 3.11 below). For example, the state pairs of the LMC depicted in Figure 3.1 can be partitioned as follows:

$$\begin{aligned}
S^2_0 &= \{\,(0,0),(1,1),(2,2),(3,3),(4,4),(5,5),(6,6)\,\}\\
S^2_1 &= \{\,(0,5),(1,5),(2,5),(3,5),(4,5),(6,5),(0,6),(1,6),(2,6),(3,6),(4,6),\\
&\quad (5,0),(5,1),(5,2),(5,3),(5,4),(5,6),(6,0),(6,1),(6,2),(6,3),(6,4)\,\}\\
S^2_? &= \{\,(0,1),(0,2),(0,3),(0,4),(1,2),(1,3),(1,4),(2,3),(2,4),(3,4),\\
&\quad (1,0),(2,0),(3,0),(4,0),(2,1),(3,1),(4,1),(3,2),(4,2),(4,3)\,\}
\end{aligned}$$

3.2 Probabilistic Bisimilarity Distances

Let $d \in S \times S \to [0,1]$ and $\omega \in \mathcal{D}(S \times S)$. Instead of $\sum_{u,v \in S} \omega(u,v)\ d(u,v)$ we write $\omega \cdot d$ in the remainder to avoid clutter. The probabilistic bisimilarity distances are defined in terms of the following function. This function measures how to reach an i-pair from state pairs in $S^2_?$.

Definition 3.8. *For $i \in \{0,1\}$, the function $\Delta_i : (S \times S \to [0,1]) \to (S \times S \to [0,1])$ is defined by*

$$\Delta_i(d)(s,t) = \begin{cases} 0 & \text{if } (s,t) \in S^2_{1-i} \\ 1 & \text{if } (s,t) \in S^2_i \\ \sup\limits_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot d & \text{if } (s,t) \in S^2_? \text{ and } i = 0 \\ \inf\limits_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot d & \text{if } (s,t) \in S^2_? \text{ and } i = 1. \end{cases}$$

To conclude that Δ_i has both a least and a greatest fixed point, we apply the Knaster-Tarski fixed point theorem (Theorem 2.7(a) and (b)). However, before using this theorem, we first need to prove that Δ_i is a monotone function, as shown in the following proposition.

Proposition 3.9. *For $i \in \{0,1\}$, Δ_i is monotone.*

Proof. Let $i \in \{0,1\}$ and $d, e \in S \times S \to [0,1]$ with $d \sqsubseteq e$. To conclude that Δ_i is monotone, it suffices to show that $\Delta_i(d) \sqsubseteq \Delta_i(e)$, that is, $\Delta_i(d)(s,t) \leq \Delta_i(e)(s,t)$ for all $s, t \in S$. Let $s, t \in S$. We distinguish the following three cases.

- If $(s,t) \in S^2_{i-1}$ then $\Delta_i(d)(s,t) = 0 = \Delta_i(e)(s,t)$.
- If $(s,t) \in S^2_i$ then $\Delta_i(d)(s,t) = 1 = \Delta_i(e)(s,t)$.
- If $(s,t) \in S^2_?$ then

$$\begin{aligned}
\Delta_0(d)(s,t) &= \sup_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot d \\
&\leq \sup_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot e \qquad [d \sqsubseteq e] \\
&= \Delta_0(e)(s,t)
\end{aligned}$$

and

$$\begin{aligned}
\Delta_1(d)(s,t) &= \inf_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot d \\
&\leq \inf_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot e \qquad [d \sqsubseteq e] \\
&= \Delta_1(e)(s,t).
\end{aligned}$$

□

In the next proposition, we establish that Δ_i is a nonexpansive function, which we later use to prove some properties of the distances.

Proposition 3.10. *For $i \in \{0,1\}$, Δ_i is nonexpansive.*

Proof. Let $i \in \{0,1\}$ and $d, e \in S \times S \to [0,1]$. To conclude that Δ_i is nonexpansive, it suffices to show that $\|\Delta_i(d) - \Delta_i(e)\| \leq \|d - e\|$, that is $|\Delta_i(d)(s,t) - \Delta_i(e)(s,t)| \leq \|d - e\|$ for all $s, t \in S$. Let $s, t \in S$. We distinguish the following three cases.

- If $(s,t) \in S^2_{i-1}$ then $|\Delta_i(d)(s,t) - \Delta_i(e)(s,t)| = |0 - 0| = 0 \leq \|d - e\|$.
- If $(s,t) \in S^2_i$ then $|\Delta_i(d)(s,t) - \Delta_i(e)(s,t)| = |1 - 1| = 0 \leq \|d - e\|$.
- Let $(s,t) \in S^2_?$. Without loss of generality assume that

$$\sup_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot d \geq \sup_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot e.$$

Then

$$\begin{aligned} |\Delta_0(d)(s,t) - \Delta_0(e)(s,t)| &= \sup_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot d - \sup_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot e \\ &\leq \sup_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot d - \omega \cdot e \\ &= \sup_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot (d - e) \\ &\leq \|d - e\|. \end{aligned}$$

Without loss of generality assume that

$$\inf_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot d \geq \inf_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot e.$$

Then

$$\begin{aligned} |\Delta_1(d)(s,t) - \Delta_1(e)(s,t)| &= \inf_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot d - \inf_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot e \\ &\leq \inf_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot d - \omega \cdot e \\ &= \inf_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot (d - e) \\ &\leq \|d - e\|. \end{aligned}$$

□

Since Δ_i is a monotone function (Proposition 3.9) from a complete lattice to itself, we can conclude from the Knaster-Tarski fixed point theorem (Theorem 2.7(a) and (b)) that Δ_i has a least fixed point and a greatest fixed point. We denote the least fixed point of Δ_1 by δ_1 and the greatest fixed point of Δ_0 by δ_0. The least fixed point δ_1 maps each pair of states to a real number in the interval $[0,1]$: the *probabilistic bisimilarity distance* of the states. As we already mentioned, distance zero captures the probabilistic bisimilarity.

Theorem 3.11 ([17, Theorem 4.10])**.** *For all $s, t \in S$, $\delta_1(s,t) = 0$ if and only if $s \sim t$.*

Example 3.12. *We calculate the distances of the states of the LMC depicted in Figure 3.1.*

(a) $\delta_1(3,3) = 0$ *as* $(3,3) \in S^2_0$.

(b) $\delta_1(5,5) = 0$ *as* $(5,5) \in S^2_0$.

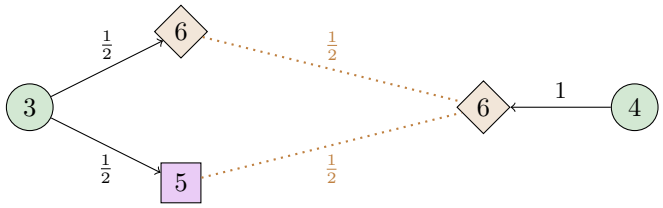
(c) $\delta_1(5,6) = 1$ *as* $(5,6) \in S^2_1$.

(d) $\delta_1(6,6) = 0$ *as* $(6,6) \in S^2_0$.

(e) The function $\omega \in \Omega(\tau(3), \tau(4))$ is defined by

$$\omega(x,y) = \begin{cases} \frac{1}{2} & \text{if } (x,y) = (6,6) \\ \frac{1}{2} & \text{if } (x,y) = (5,6) \\ 0 & \text{otherwise.} \end{cases}$$

The probability distributions $\tau(3)$ and $\tau(4)$ and the ω can be depicted as follows.



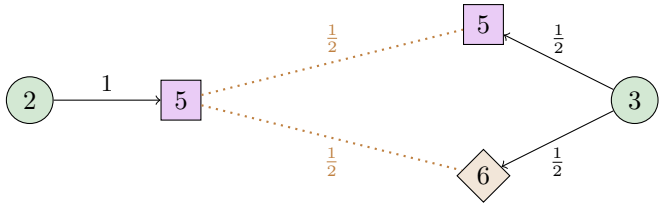
We can calculate the distance between states 3 and 4 as follows.

$$\begin{aligned} \delta_1(3,4) &= \Delta_1(\delta_1)(3,4) \\ &= \inf_{\pi \in \Omega(\tau(3),\tau(4))} \sum_{u,v \in S} \pi(u,v)\ \delta_1(u,v) \\ &= \omega(6,6)\delta_1(6,6) + \omega(5,6)\delta_1(5,6) \qquad [\Omega(\tau(3),\tau(4)) = \{\omega\}] \\ &= \frac{1}{2}\delta_1(6,6) + \frac{1}{2}\delta_1(5,6) \\ &= \frac{1}{2} \qquad [\textit{by (c) and (d)}] \end{aligned}$$

(f) The function $\pi \in \Omega(\tau(2), \tau(3))$ is defined by

$$\pi(x,y) = \begin{cases} \frac{1}{2} & \text{if } (x,y) = (5,5) \\ \frac{1}{2} & \text{if } (x,y) = (5,6) \\ 0 & \text{otherwise.} \end{cases}$$

The probability distributions $\tau(2)$ and $\tau(3)$ and the π can be depicted as follows.



We can calculate the distance between states 2 and 3 as follows.

$$\begin{aligned} \delta_1(2,3) &= \Delta_1(\delta_1)(2,3) \\ &= \inf_{\rho \in \Omega(\tau(2),\tau(3))} \sum_{u,v \in S} \rho(u,v)\ \delta_1(u,v) \\ &= \pi(5,5)\delta_1(5,5) + \pi(5,6)\delta_1(5,6) \qquad [\Omega(\tau(2),\tau(3)) = \{\pi\}] \\ &= \frac{1}{2}\delta_1(5,5) + \frac{1}{2}\delta_1(5,6) \\ &= \frac{1}{2} \qquad [\textit{by (b) and (c)}] \end{aligned}$$

(g) The function $\pi \in \Omega(\tau(2), \tau(4))$ is defined by

$$\pi(x,y) = \begin{cases} 1 & \text{if } (x,y) = (5,6) \\ 0 & \text{otherwise.} \end{cases}$$

The probability distribution $\tau(2)$ and $\tau(4)$ and the π can be depicted as follows.

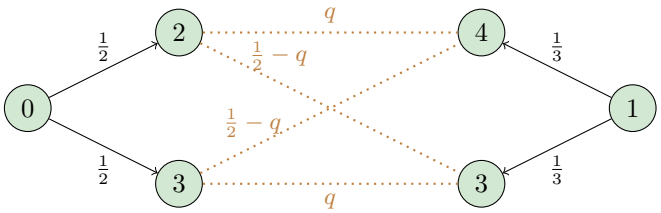
We can calculate the distance between states 2 and 4 as follows.

$$\begin{aligned}
\delta_1(2,4) &= \Delta_1(\delta_1)(2,4) \\
&= \inf_{\rho \in \Omega(\tau(2),\tau(4))} \sum_{u,v \in S} \rho(u,v)\ \delta_1(u,v) \\
&= \pi(5,6)\delta_1(5,6) \qquad [\Omega(\tau(2),\tau(4)) = \{\pi\}] \\
&= 1\delta_1(5,6) \\
&= 1 \qquad [\textit{by (c)}]
\end{aligned}$$

(h) For $q \in [0, \frac{1}{2}]$, the function $\omega_q \in \Omega(\tau(0), \tau(1))$ is defined by

$$\omega_q(x,y) = \begin{cases} q & \text{if } (x,y) = (2,4) \\ \frac{1}{2} - q & \text{if } (x,y) = (2,3) \\ \frac{1}{2} - q & \text{if } (x,y) = (3,4) \\ q & \text{if } (x,y) = (3,3) \\ 0 & \text{otherwise.} \end{cases}$$

The probability distributions $\tau(0)$ and $\tau(1)$ and the ω_q can be depicted as follows.



We can calculate the distance between states 0 and 1 as follows.

$$\begin{aligned}
\delta_1(0,1) &= \Delta_1(\delta_1)(0,1) \\
&= \inf_{\rho \in \Omega(\tau(0),\tau(1))} \sum_{u,v \in S} \rho(u,v)\ \delta_1(u,v) \\
&= \inf_{q \in [0,\frac{1}{2}]} \omega_q(2,4)\delta_1(2,4) + \omega_q(2,3)\delta_1(2,3) + \omega_q(3,4)\delta_1(3,4) + \omega_q(3,3)\delta_1(3,3) \\
&\qquad [\Omega(\tau(0),\tau(1)) = \{\omega_q \mid q \in [0,\tfrac{1}{2}]\}] \\
&= \inf_{q \in [0,\frac{1}{2}]} q\delta_1(2,4) + (\tfrac{1}{2} - q)\delta_1(2,3) + (\tfrac{1}{2} - q)\delta_1(3,4) + q\delta_1(3,3) \\
&= \inf_{q \in [0,\frac{1}{2}]} q + (\tfrac{1}{2} - q)\tfrac{1}{2} + (\tfrac{1}{2} - q)\tfrac{1}{2} \qquad [\textit{by (a), (e), (f) and (g)}] \\
&= \frac{1}{2}
\end{aligned}$$

The state pairs in S_0^2 are probabilistic bisimilar. This can be explained by means of the games mentioned in Section 1.5.2 of the introduction to this thesis. The state pairs in S_1^2 have different labels, which explains their difference in behaviour. Therefore, the distances of the state pairs in $S_?^2$ remain to be explained.

Proposition 3.13. *If $S_?^2 \neq \emptyset$ then $S_1^2 \neq \emptyset$.*

Proof. We prove the contrapositive. Assume that $S_1^2 = \emptyset$. As a consequence, all states have the same label. We leave it to the reader to verify that $S \times S$ is a probabilistic bisimulation in this case. Hence, $S_0^2 = S \times S$ and, therefore, $S_?^2 = \emptyset$. □

Therefore, for the remainder of this thesis, we assume that $S_?^2 \neq \emptyset$ and $S_1^2 \neq \emptyset$. We conclude this section with three properties of the distances that we use later in this thesis. First, we link δ_0 and δ_1.

Proposition 3.14. *For all $(s,t) \in S$, $\delta_0(s,t) + \delta_1(s,t) = 1$.*

Proof. Since Δ_i is monotone (Proposition 3.9) and nonexpansive (Proposition 3.10), we can conclude from Theorem 2.11 that $\delta_0 = \lim_{n\in\mathbb{N}} \Delta_0^n(\bot)$ and $\delta_1 = \lim_{n\in\mathbb{N}} \Delta_0^n(\top)$. Let $s, t \in S$. To conclude that $\delta_0(s,t)+\delta_1(s,t) = 1$, it suffices to show that for all $n \in \mathbb{N}$, $\Delta_0^n(\top)(s,t) + \Delta_1^n(\bot)(s,t) = 1$ by induction on n. In the base case, when $n = 0$, we have that $\top(s,t) + \bot(s,t) = 1$. In the inductive case, $n > 0$, we distinquish the following cases:

- If $(s,t) \in S_0^2$ then
$$\Delta_0^n(\top)(s,t) + \Delta_1^n(\bot)(s,t) = 1 + 0 = 1.$$
- If $(s,t) \in S_1^2$ then
$$\Delta_0^n(\top)(s,t) + \Delta_1^n(\bot)(s,t) = 0 + 1 = 1.$$
- If $(s,t) \in S_?^2$ then
$$\begin{aligned}
\Delta_0^n(\top)(s,t) &= \Delta_0(\Delta_0^{n-1}(\top))(s,t) \\
&= \inf_{\omega\in\Omega(\tau(s),\tau(t))} \omega \cdot \Delta_0^{n-1}(\top) \\
&= \inf_{\omega\in\Omega(\tau(s),\tau(t))} \omega \cdot (1 - \Delta_1^{n-1}(\bot)) \qquad \text{[induction hypothesis]} \\
&= \inf_{\omega\in\Omega(\tau(s),\tau(t))} 1 - \omega \cdot \Delta_1^{n-1}(\bot) \\
&= 1 - \sup_{\omega\in\Omega(\tau(s),\tau(t))} \omega \cdot \Delta_1^{n-1}(\bot) \\
&= 1 - \Delta_1^n(\bot)(s,t).
\end{aligned}$$
Hence, $\Delta_0^n(\top)(s,t) + \Delta_1^n(\bot)(s,t) = 1$.

□

Since δ_1 is a fixed point of Δ_1, we have that for each $(s,t) \in S_?^2$, $\delta_1 = \inf_{\rho\in\Omega(\tau(s),\tau(t))} \rho \cdot \delta_1$. In the next proposition, we show that we can restrict to vertices, that is, we can restrict ourselves to $V(\Omega(\tau(s),\tau(t)))$. Since this set of vertices is finite, the infimum becomes a minimum and, hence, we can find $\omega \in V(\Omega(\tau(s),\tau(t)))$ such that $\delta_1(s,t) = \omega \cdot \delta_1$.

Proposition 3.15. *For each $(s,t) \in S_?^2$, there exists $\omega \in V(\Omega(\tau(s),\tau(t)))$ such that $\delta_1(s,t) = \omega \cdot \delta_1$.*

Proof. Let $(s,t) \in S_?^2$. Let $X = \Omega(\tau(s),\tau(t))$. Let the function $f : X \to [0,1]$ be defined by

$$f(\omega) = \omega \cdot \delta_1.$$

First, we show that f is linear. Let $\omega, \pi \in X$ and $q \in (0,1)$. We have that

$$\begin{aligned}
f(q\,\omega + (1-q)\,\pi) &= (q\,\omega + (1-q)\,\pi) \cdot \delta_1 \\
&= q\,(\omega \cdot \delta_1) + (1-q)\,(\pi \cdot \delta_1) \\
&= q\,f(\omega) + (1-q)\,f(\pi).
\end{aligned}$$

Because a linear function on a convex polytope attains its minimum at a vertex (see, for example, [55, Theorem 2 of Chapter 1]), we can conclude that

$$\delta_1(s,t) = \Delta_1(\delta_1)(s,t) = \inf_{\pi \in \Omega(\tau(s),\tau(t))} \pi \cdot \delta_1 = \min_{\pi \in V(\Omega(\tau(s),\tau(t)))} \pi \cdot \delta_1 = \omega \cdot \delta_1$$

for some $\omega \in V(\Omega(\tau(s), \tau(t)))$. □

Lastly, the following proposition is particularly useful in demonstrating that the distance between states s and t, where $(s,t) \in S^2_?$, is one if both s and t only transition to state pairs that have distance one.

Proposition 3.16. *For all $(s,t) \in S^2_?$, if $\delta_1(u,v) = 1$ for all $u \in \text{support}(\tau(s))$ and $v \in \text{support}(\tau(t))$, then $\delta_1(s,t) = 1$.*

Proof. Let $(s,t) \in S^2_?$ and $\omega \in \Omega(\tau(s), \tau(t))$. First, we show that $\text{support}(\omega) \subseteq \text{support}(\tau(s)) \times \text{support}(\tau(t))$. For all $u, v \in S$,

$$\begin{aligned}
(u,v) \in \text{support}(\omega) &\Leftrightarrow \omega(u,v) > 0 \\
&\Rightarrow \omega(u,S) > 0 \text{ and } \omega(S,v) > 0 \\
&\Leftrightarrow \tau(s)(u) > 0 \text{ and } \tau(t)(v) > 0 \\
&\Leftrightarrow u \in \text{support}(\tau(s)) \text{ and } v \in \text{support}(\tau(t)) \\
&\Leftrightarrow (u,v) \in \text{support}(\tau(s)) \times \text{support}(\tau(t)).
\end{aligned}$$

Assume that $\delta_1(u,v) = 1$ for all $(u,v) \in \text{support}(\tau(s)) \times \text{support}(\tau(t))$. Hence, $\delta_1(u,v) = 1$ for all $(u,v) \in \text{support}(\omega)$. Next, we show that $\omega \cdot \delta_1 = 1$.

$$\begin{aligned}
\omega \cdot \delta_1 &= \sum_{u,v \in S} \omega(u,v)\delta_1(u,v) \\
&= \sum_{(u,v) \in \text{support}(\omega)} \omega(u,v)\delta_1(u,v) \\
&= \sum_{(u,v) \in \text{support}(\omega)} \omega(u,v) \qquad [\delta_1(u,v) = 1 \text{ for all } (u,v) \in \text{support}(\omega)] \\
&= \sum_{(u,v) \in S} \omega(u,v) \\
&= 1 \qquad [\omega \in \Omega(\tau(s), \tau(t))]
\end{aligned}$$

Finally, we have that

$$\begin{aligned}
\delta_1(s,t) &= \Delta_1(\delta_1)(s,t) \\
&= \inf_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot \delta_1 \\
&= 1.
\end{aligned}$$

□

In this chapter, we introduced the concept of couplings and how they are used to define probabilistic bisimilarity and probabilistic bisimilarity distances. In the following chapter, we will explore policies. Recall that, in our game, a policy represents the strategy chosen by the player. We will define policies formally, examine their properties, and explore their close relationship with couplings.

4 Policies

The foundation of the game studied in this thesis is based on matching parts of transitions. These matchings are also known as couplings, a concept we studied in the previous chapter. Recall that our game starts in a state pair (s, t). The player chooses a coupling of the transitions of s and t. Randomly, respecting the probabilities associated with the coupling, a matching of parts of transitions is chosen. This matching takes the game to (s', t'). If the states s' and t' are probabilistic bisimilar, we call (s', t') a 0-pair (depicted in blue) because they have distance zero. If the states have different labels, we call (s', t') a 1-pair (depicted in red). In either case, the game stops. If neither of these conditions holds the game continues in state pair (s', t') (depicted in green). The objective of the player is to minimize the probability of reaching a 1-pair. Also, in this case, the player tries to avoid reaching 1-pairs.

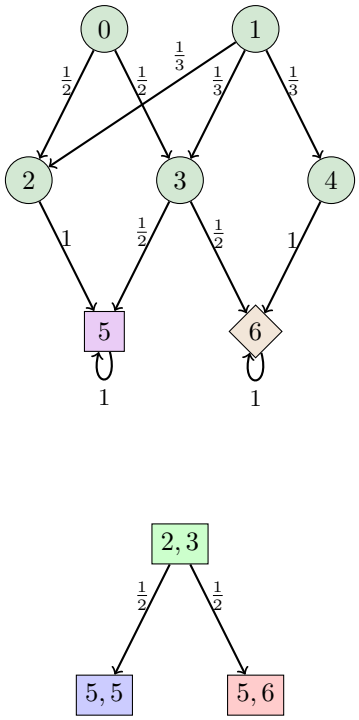


Figure 4.1: An LMC and a matching of the transitions of states 2 and 3

For instance, consider the transitions of states 2 and 3 in Figure 4.1. A portion of the transition from state 2 to state 5 (with probability $\frac{1}{2}$) can be matched with the transition from state 3 to state 5. The remaining portion of the transition from state 2 to state 5 is matched with the transition from state 3 to state 6, also with probability $\frac{1}{2}$. This matching is illustrated in Figure 4.1. Notice that the state pair $(5, 5)$ is depicted in blue as $(5, 5) \in S_0^2$ and the state pair $(5, 6)$ is depicted in red as $(5, 6) \in S_1^2$.

As we mentioned in the introduction of this thesis, our game is elegant, as it captures a global property of behaviors (the distances) through rules that depend solely on local considerations (the couplings). A strategy for the player, also known as a policy in this context, involves choosing a coupling for each pair of states that is neither a 0- nor a 1-pair. These policies are memoryless, meaning that the choice of coupling is made independently of prior history. Moreover, they are deterministic, meaning that for any pair of states, exactly one coupling is chosen

from the set of couplings.

In this chapter, we will define policies and examine some properties. We will start by introducing policies and explaining their role in characterizing probabilistic bisimilarity distances. The chapter will conclude by exploring various types of policies, including optimal and vertex policies.

4.1 Policies and Distances

As mentioned earlier, a player's strategy, referred to as a policy, involves selecting a coupling for each pair of states that is neither a 0-pair nor a 1-pair. We formally define them as follows.

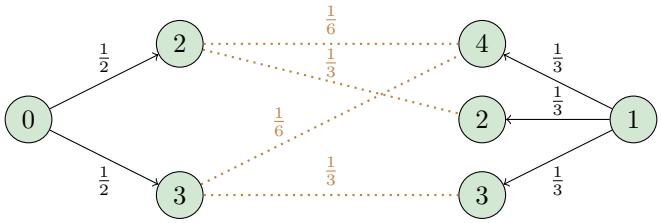
Definition 4.1. *The set $\mathcal{P}$ of* policies *is defined by*

$$\mathcal{P} = \{\, P \in S_?^2 \to \mathcal{D}(S \times S) \mid \forall (s,t) \in S_?^2 : P(s,t) \in \Omega(\tau(s), \tau(t)) \,\}.$$

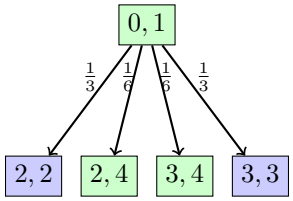
In the next example, we present a visual representation of these policies.

Example 4.2. *In this example, we graphically represent (parts of) three policies Q, R, and Z for the LMC depicted in Figure 4.1. The transitions of the states 0 and 1 can be represented as follows.*

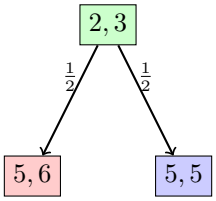
We begin with the policy Q. The matching of (the target states) of these transitions in policy Q is shown next.



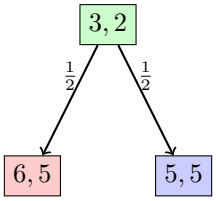
As seen above, $\frac{1}{3}$ of the transitions from state 0 to state 2 is matched with $\frac{1}{2}$ of the transition from state 1 to state 4. In this representation, state pairs $(s,t) \in S_0^2$ are depicted in blue, $(s,t) \in S_1^2$ in red, and $(s,t) \in S_?^2$ in green. The following graph illustrates the coupling $Q(0,1)$.



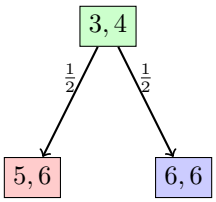
The following graph presents the coupling $Q(2,3)$.



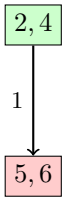
The following graph presents the coupling $Q(3,2)$.



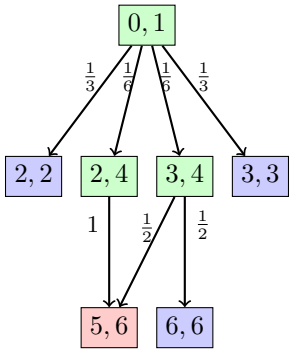
The following graph presents the coupling $Q(3,4)$.



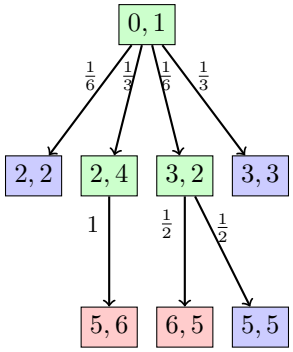
The following graph presents the coupling $Q(2,4)$.



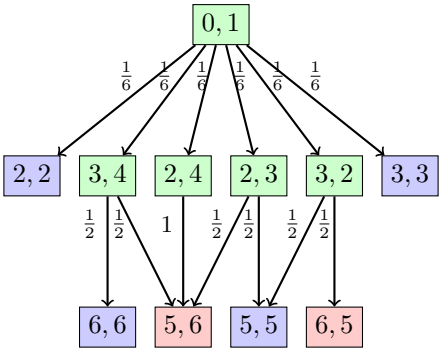
Putting it all together, the policy Q *is depicted as follows.*



By following similar steps as outlined above, we can obtain the policy, which we will call R*, shown below,*



and the policy Z *depicted as follows.*



Given a policy, we can define the probability of reaching a 0- or 1-pair from any state pair as follows. This definition serves as a foundation for establishing a fixed-point characterization of probabilistic bisimilarity distances.

Definition 4.3. *Let $P \in \mathcal{P}$ and $i \in \{0,1\}$. The function $\Delta_{iP} : (S \times S \to [0,1]) \to (S \times S \to [0,1])$ is defined by*

$$\Delta_{iP}(d)(s,t) = \begin{cases} 1 & \text{if } (s,t) \in S_i^2 \\ 0 & \text{if } (s,t) \in S_{1-i}^2 \\ P(s,t) \cdot d & \text{if } (s,t) \in S_?^2. \end{cases}$$

To conclude that Δ_{iP} has a least fixed point, we apply the Knaster-Tarski fixed point theorem (Theorem 2.7(a)). However, before using this theorem, we first need to prove that Δ_{iP} is a monotone function, as shown in the following proposition. This result is a generalization of results that can be found in the literature (see, for example, [52, Proposition 6.1.3]).

Proposition 4.4. *For all $P \in \mathcal{P}$ and $i \in \{0,1\}$, Δ_{iP} is monotone.*

Proof. Let $P \in \mathcal{P}$, $i \in \{0,1\}$, and $d, e \in S \times S \to [0,1]$. Assume that $d \sqsubseteq e$. Let $s, t \in S$. We distinguish the following cases.

- If $(s,t) \in S_i^2$ then
$$\Delta_{iP}(d)(s,t) = 1 = \Delta_{iP}(e)(s,t).$$
- If $(s,t) \in S_{1-i}^2$ then
$$\Delta_{iP}(d)(s,t) = 0 = \Delta_{iP}(e)(s,t).$$
- Otherwise, $(s,t) \in S_?^2$. Then
$$\begin{aligned} \Delta_{iP}(d)(s,t) &= P(s,t) \cdot d \\ &\leq P(s,t) \cdot e \qquad [d \sqsubseteq e] \\ &= \Delta_{iP}(e)(s,t). \end{aligned}$$

Hence, $\Delta_{iP}(d) \sqsubseteq \Delta_{iP}(e)$. □

Since Δ_{iP} is a monotone function (Proposition 4.4) from a complete lattice to itself, we can conclude from the Knaster-Tarski fixed point theorem (Theorem 2.7(a)) that Δ_{iP} has a least fixed point. We denote the least fixed point of Δ_{iP} by δ_{iP}.

Example 4.5. *We calculate δ_{1P} and δ_{0P} of the state pairs of the policies Q and R depicted in Example 4.2 as follows.*

(a) $\delta_{1Q}(3,3) = \delta_{1R}(3,3) = 0$ *as* $(3,3) \in S_0^2$.

(b) $\delta_{1Q}(5,5) = \delta_{1R}(5,5) = 0$ *as* $(5,5) \in S_0^2$.

(c) $\delta_{1Q}(5,6) = 1$ *as* $(5,6) \in S_1^2$.

(d) $\delta_{1Q}(6,6) = 0$ *as* $(6,6) \in S_0^2$.

(e) $\delta_{1Q}(6,5) = \delta_{1R}(6,5) = 1$ *as* $(6,5) \in S_1^2$.

(f) $\delta_{1Q}(2,2) = \delta_{1R}(2,2) = 0$ *as* $(2,2) \in S_0^2$.

(g) $\delta_{0Q}(5,6) = 0$ *as* $(5,6) \in S_1^2$.

(h) $\delta_{0Q}(5,5) = \delta_{0R}(5,5) = 1$ *as* $(5,5) \in S_0^2$.

(i) $\delta_{0Q}(6,5) = \delta_{0R}(6,5) = 0$ *as* $(5,6) \in S_1^2$.

(j)

$$\begin{aligned}
\delta_{1Q}(2,4) &= \Delta_{1Q}(\delta_{1Q})(2,4) \\
&= Q(2,4) \cdot \delta_{1Q} \\
&= \sum_{(u,v) \in S \times S} Q(2,4)(u,v)\delta_{1Q}(u,v) \\
&= Q(2,4)(5,6)\delta_{1Q}(5,6) \\
&= 1\delta_{1Q}(5,6) \qquad [\textit{by (c)}] \\
&= 1 \cdot 1 \\
&= 1
\end{aligned}$$

(k)

$$\begin{aligned}
\delta_{0Q}(2,4) &= \Delta_{0Q}(\delta_{0Q})(2,4) \\
&= Q(2,4) \cdot \delta_{0Q} \\
&= \sum_{(u,v) \in S \times S} Q(2,4)(u,v)\delta_{0Q}(u,v) \\
&= Q(2,4)(5,6)\delta_{0Q}(5,6) \\
&= 1\delta_{0Q}(5,6) \qquad [\textit{by (g)}] \\
&= 1 \cdot 0 \\
&= 0
\end{aligned}$$

(l)

$$\begin{aligned}
\delta_{1Q}(3,4) &= \Delta_{1Q}(\delta_{1Q})(3,4) \\
&= Q(3,4) \cdot \delta_{1Q} \\
&= \sum_{(u,v) \in S \times S} Q(3,4)(u,v)\delta_{1Q}(u,v) \\
&= Q(3,4)(6,6)\delta_{1Q}(6,6) + Q(3,4)(5,6)\delta_{1Q}(5,6) \\
&= \frac{1}{2}\delta_{1Q}(6,6) + \frac{1}{2}\delta_{1Q}(5,6) \\
&= \frac{1}{2} \cdot 0 + \frac{1}{2} \cdot 1 \qquad [\textit{by (c) and (d)}] \\
&= \frac{1}{2}
\end{aligned}$$

(m)

$$\begin{aligned}
\delta_{1Q}(0,1) &= \Delta_{1Q}(\delta_{1Q})(0,1) \\
&= Q(0,1) \cdot \delta_{1Q} \\
&= \sum_{(u,v)\in S\times S} Q(0,1)(u,v) \cdot \delta_{1Q}(u,v) \\
&= Q(0,1)(2,2)\delta_{1Q}(2,2) + Q(0,1)(2,4)\delta_{1Q}(2,4) + Q(0,1)(3,3)\delta_{1Q}(3,3) \\
&+ Q(0,1)(3,4)\delta_{1Q}(3,4) \\
&= \frac{1}{3}\delta_{1Q}(2,2) + \frac{1}{6}\delta_{1Q}(2,4) + \frac{1}{3}\delta_{1Q}(3,3) + \frac{1}{6}\delta_{1Q}(3,4) \\
&= \frac{1}{3}\cdot 0 + \frac{1}{6}\cdot 1 + \frac{1}{3}\cdot 0 + \frac{1}{6}\cdot\frac{1}{2} \qquad [\textit{by (f), (j), (a), and (l)}] \\
&= \frac{3}{12}
\end{aligned}$$

(n)

$$\begin{aligned}
\delta_{1R}(3,2) &= \Delta_{1R}(\delta_{1R})(3,2) \\
&= R(3,2) \cdot \delta_{1R} \\
&= \sum_{(u,v)\in S\times S} R(3,2)(u,v)\delta_{1R}(u,v) \\
&= R(3,2)(5,5)\delta_{1R}(5,5) + R(3,2)(6,5)\delta_{1R}(6,5) \\
&= \frac{1}{2}\delta_{1R}(5,5) + \frac{1}{2}\delta_{1R}(6,5) \\
&= \frac{1}{2}\cdot 0 + \frac{1}{2}\cdot 1 \qquad [\textit{by (b) and (e)}] \\
&= \frac{1}{2}
\end{aligned}$$

(o)

$$\begin{aligned}
\delta_{0R}(3,2) &= \Delta_{0R}(\delta_{0R})(3,2) \\
&= R(3,2) \cdot \delta_{0R} \\
&= \sum_{(u,v)\in S\times S} R(3,2)(u,v)\delta_{0R}(u,v) \\
&= R(3,2)(5,5)\delta_{0R}(5,5) + R(3,2)(6,5)\delta_{0R}(6,5) \\
&= \frac{1}{2}\delta_{0R}(5,5) + \frac{1}{2}\delta_{0R}(6,5) \\
&= \frac{1}{2}\cdot 1 + \frac{1}{2}\cdot 0 \qquad [\textit{by (h) and (i)}] \\
&= \frac{1}{2}
\end{aligned}$$

(p)

$$\begin{aligned}
\delta_{1R}(2,4) &= \Delta_{1R}(\delta_{1R})(2,4) \\
&= R(2,4) \cdot \delta_{1R} \\
&= \sum_{(u,v)\in S\times S} R(2,4)(u,v)\delta_{1R}(u,v) \\
&= R(2,4)(5,6)\delta_{1R}(5,6) \\
&= 1\delta_{1R}(5,6) \qquad [\textit{by (c)}] \\
&= 1\cdot 1 \\
&= 1
\end{aligned}$$

(q)

$$\begin{aligned}
\delta_{1R}(0,1) &= \Delta_{1R}(\delta_{1R})(0,1)\\
&= R(0,1)\cdot\delta_{1R}\\
&= \sum_{(u,v)\in S\times S} R(0,1)(u,v)\cdot\delta_{1R}(u,v)\\
&= R(0,1)(2,2)\delta_{1R}(2,2) + R(0,1)(2,4)\delta_{1R}(2,4) + R(0,1)(3,3)\delta_{1R}(3,3)\\
&+ R(0,1)(3,2)\delta_{1R}(3,2)\\
&= \frac{1}{6}\delta_{1R}(2,2) + \frac{1}{3}\delta_{1R}(2,4) + \frac{1}{3}\delta_{1R}(3,3) + \frac{1}{6}\delta_{1R}(3,2)\\
&= \frac{1}{6}\cdot 0 + \frac{1}{3}\cdot 1 + \frac{1}{3}\cdot 0 + \frac{1}{6}\cdot\frac{1}{2} \qquad \textit{[by (f), (p), (a), and (n)]}\\
&= \frac{5}{12}
\end{aligned}$$

The probabilistic bisimilarity distances can be characterized in terms of a policy that minimizes the probability of reaching a 1-pair.

Theorem 4.6 ([6, Theorem 8])**.** $\delta_1 = \min_{P\in\mathcal{P}} \delta_{1P}$.

To conclude that Δ_{iP} has a unique fixed point, we apply the Banach's fixed point theorem (Theorem 2.10(c)). However, before using this theorem, we first need to prove that Δ_{iP} is a nonexpansive and power-contractive function, as shown in the following propositions. We denote P restricted to $S^2_?$ by $P_?$, that is, $P_? \in S^2_? \to (S^2_? \to [0,1])$.

Proposition 4.7. *For all $P \in \mathcal{P}$, $i \in \{0,1\}$, $d, e \in S\times S \to [0,1]$, $s,t \in S$, and $n \in \mathbb{N}$,*

(a) if $(s,t) \in S^2_0 \cup S^2_1$ then $\Delta^{n+1}_{iP}(d)(s,t) = \Delta^{n+1}_{iP}(e)(s,t)$, and

(b) if $(s,t) \in S^2_?$ then $|\Delta^{n+1}_{iP}(d)(s,t) - \Delta^{n+1}_{iP}(e)(s,t)| \le P^n_?(s,t)(S^2_?)\,\|d-e\|$.

Proof. Let $P \in \mathcal{P}$, $i \in \{0,1\}$, and $d, e \in S\times S \to [0,1]$ and $s, t \in S$.

(a) We distinguish two cases.

- If $(s,t) \in S^2_i$ then
$$\Delta^{n+1}_{iP}(d)(s,t) = 1 = \Delta^{n+1}_{iP}(e)(s,t).$$
- If $(s,t) \in S^2_{1-i}$ then
$$\Delta^{n+1}_{iP}(d)(s,t) = 0 = \Delta^{n+1}_{iP}(e)(s,t).$$

(b) Let $(s,t) \in S^2_?$. Without loss of generality, assume that $\Delta^{n+1}_{iP}(d)(s,t) \ge \Delta^{n+1}_{iP}(e)(s,t)$. We prove this case by induction on n. In the base case, when $n = 0$, we have that

$$\begin{aligned}
\Delta_{iP}(d)(s,t) - \Delta_{iP}(e)(s,t) &= P(s,t)\cdot d - P(s,t)\cdot e\\
&= P(s,t)\cdot(d-e)\\
&= \sum_{u,v\in S} P(s,t)(u,v)\,(d(u,v) - e(u,v))\\
&\le \sum_{u,v\in S} P(s,t)(u,v)\,\|d-e\|\\
&\le \|d-e\|\\
&= P^0_?(s,t)(S^2_?)\,\|d-e\|.
\end{aligned}$$

In the inductive case, when $n > 0$, we have that

$$\begin{aligned}
&\Delta_{iP}^{n+1}(d)(s,t) - \Delta_{iP}^{n+1}(e)(s,t) \\
&= P(s,t) \cdot \Delta_{iP}^{n}(d) - P(s,t) \cdot \Delta_{iP}^{n}(e) \\
&= P(s,t) \cdot (\Delta_{iP}^{n}(d) - \Delta_{iP}^{n}(e)) \\
&= \sum_{u,v \in S} P(s,t)(u,v) \left(\Delta_{iP}^{n}(d)(u,v) - \Delta_{iP}^{n}(e)(u,v)\right) \\
&= \sum_{(u,v) \in S_?^2} P_?(s,t)(u,v) \left(\Delta_{iP}^{n}(d)(u,v) - \Delta_{iP}^{n}(e)(u,v)\right) \qquad \text{[part (a)]} \\
&\leq \sum_{(u,v) \in S_?^2} P_?(s,t)(u,v) P_?^{n-1}(u,v)(S_?^2) \, \|d - e\| \qquad \text{[induction hypothesis]} \\
&= P_?^{n}(s,t)(S_?^2) \, \|d - e\|.
\end{aligned}$$

□

Proposition 4.8. *For all $P \in \mathcal{P}$, $i \in \{0,1\}$, Δ_{iP} is nonexpansive.*

Proof. Let $P \in \mathcal{P}$, $i \in \{0,1\}$, d, $e \in S \times S \to [0,1]$, and s, $t \in S$. It suffices to show that $|\Delta_{iP}(d)(s,t) - \Delta_{iP}(e)(s,t)| \leq \|d - e\|$. Without loss of generality, assume that $\Delta_{iP}(d)(s,t) \geq \Delta_{iP}(e)(s,t)$. We distinguish two cases.

- If $(s,t) \in S_0^2 \cup S_1^2$ then from Proposition 4.7(a) we can conclude that

$$\Delta_{iP}(d)(s,t) - \Delta_{iP}(e)(s,t) = 0 \leq \|d - e\|.$$

- If $(s,t) \in S_?^2$ then

$$\begin{aligned}
\Delta_{iP}(d)(s,t) - \Delta_{iP}(e)(s,t) &= P(s,t) \cdot d - P(s,t) \cdot e \\
&= \left(\sum_{u,v \in S} P(s,t)(u,v) \, d(u,v) \right) - \left(\sum_{u,v \in S} P(s,t)(u,v) \, e(u,v) \right) \\
&= \sum_{u,v \in S} P(s,t)(u,v) \left(d(u,v) - e(u,v)\right) \\
&\leq \sum_{u,v \in S} P(s,t)(u,v) \, \|d - e\| \\
&\leq \|d - e\|.
\end{aligned}$$

□

The next proposition shows that, given an initial state pair (s,t), the total probability of either remaining in $S_?^2$ for n steps, or reaching to a 0-pair or a 1-pair after $n+1$ steps, is at most one.

Proposition 4.9. *For all $P \in \mathcal{P}$, $n \in \mathbb{N}$ and $(s,t) \in S_?^2$,*

$$P_?^{n}(s,t)(S_?^2) + \Delta_{0P}^{n+1}(\bot)(s,t) + \Delta_{1P}^{n+1}(\bot)(s,t) \leq 1.$$

Proof. Let $P \in \mathcal{P}$ and $(s,t) \in S_?^2$. We prove this proposition by induction on n. In the base case, $n = 0$, we have that

$$P_?^{0}(s,t)(S_?^2) + \Delta_{0P}(\bot)(s,t) + \Delta_{1P}(\bot)(s,t) = 1 + P(s,t) \cdot \bot + P(s,t) \cdot \bot = 1 + 0 + 0 = 1.$$

In the inductive case, $n > 0$, we have that

$$\begin{aligned}
&P^n_?(s,t)(S^2_?) + \Delta^{n+1}_{0P}(\bot)(s,t) + \Delta^{n+1}_{1P}(\bot)(s,t) \\
&\quad= \left(\sum_{(u,v)\in S^2_?} P_?(s,t)(u,v)\, P^{n-1}_?(u,v)(S^2_?)\right) + \left(\sum_{(u,v)\in S^2} P(s,t)(u,v)\, \Delta^n_{0P}(\bot)(u,v)\right) \\
&\qquad+ \left(\sum_{(u,v)\in S^2} P(s,t)(u,v)\, \Delta^n_{1P}(\bot)(u,v)\right) \\
&\quad= \sum_{(u,v)\in S^2_?} P(s,t)(u,v)\, (P^{n-1}_?(u,v)(S^2_?) + \Delta^n_{0P}(\bot)(u,v) + \Delta^n_{1P}(\bot)(u,v)) \\
&\qquad+ \sum_{(u,v)\in S^2_0\cup S^2_1} P(s,t)(u,v)\, (\Delta^n_{0P}(\bot)(u,v) + \Delta^n_{1P}(\bot)(u,v)) \\
&\quad\leq \sum_{(u,v)\in S^2_?} P(s,t)(u,v) + \sum_{(u,v)\in S^2_0\cup S^2_1} P(s,t)(u,v) \qquad \text{[induction hypothesis]} \\
&\quad= 1.
\end{aligned}$$

□

Proposition 4.10. *For all $P \in \mathcal{P}$ and $i \in \{0,1\}$, there exists $n \in \mathbb{N}$ such that Δ^n_{iP} is contractive.*

Proof. Let $P \in \mathcal{P}$ and $i \in \{0,1\}$. Since Δ_{1P} is monotone (Proposition 4.4) and nonexpansive (Proposition 4.8), we can conclude from Theorem 2.11(a) that

$$\delta_{1P} = \lim_{n\in\mathbb{N}} \Delta^n_{1P}(\bot). \tag{4.1}$$

Let $(s,t) \in S^2_?$. By Theorem 3.11, $\delta_1(s,t) > 0$. According to Theorem 4.6, $\delta_{1P}(s,t) > 0$. Hence, from (4.1) we can conclude that there exists $n_{st} \in \mathbb{N}$ such that for all $n \geq n_{st}$ we have that $\Delta^{n+1}_{1P}(\bot)(s,t) > 0$. By Proposition 4.9, $P^n_?(s,t)(S^2_?) < 1$.

Let $n = (\max_{(s,t)\in S^2_?} n_{st}) + 1$ and $c = \max_{(s,t)\in S^2_?} P^n_?(s,t)(S^2_?)$. Note that $c < 1$. We conclude this proof by showing that Δ^n_{iP} is c-contractive. Let d, $e \in S \times S \to [0,1]$ and s, $t \in S$. It suffices to show that $|\Delta^n_{iP}(d)(s,t) - \Delta^n_{iP}(e)(s,t)| \leq c\,\|d - e\|$. We distinguish the following cases.

- If $(s,t) \in S^2_0 \cup S^2_1$ then

$$\begin{aligned}
|\Delta^n_{iP}(d)(s,t) - \Delta^n_{iP}(e)(s,t)| &= 0 \qquad \text{[Proposition 4.7(a)]} \\
&\leq c\,\|d - e\|.
\end{aligned}$$

- Let $(s,t) \in S^2_?$. Then $n \geq n_{st} + 1$ and

$$\begin{aligned}
|\Delta^n_{iP}(d)(s,t) - \Delta^n_{iP}(e)(s,t)| &\leq P^{n-1}_?(s,t)(S^2_?)\,\|d - e\| \qquad \text{[Proposition 4.7(b)]} \\
&\leq c\,\|d - e\| \qquad [n - 1 \geq n_{st}]
\end{aligned}$$

□

The following corollary shows that Δ_{iP} has a unique fixed point.

Corollary 4.11. *For all $P \in \mathcal{P}$ and $i \in \{0,1\}$, δ_{iP} is the unique fixed point of Δ_{iP}.*

Proof. Let $P \in \mathcal{P}$ and $i \in \{0,1\}$. Since Δ^n_{iP} is contractive for some $n \in \mathbb{N}$ (Proposition 4.10) and Δ_{iP} is nonexpansive (Proposition 4.8), we can conclude from Theorem 2.10(c) that Δ_{iP} has a unique fixed point. Because δ_{iP} is a fixed point of Δ_{iP} by definition, we can conclude the desired result. □

We conclude this section by linking δ_{0P} and δ_{1P}. This result is similar to the Proposition 3.14 from the previous chapter. This result will be used later in the thesis.

Proposition 4.12. *For all $P \in \mathcal{P}$ and $s, t \in S$, $\delta_{0P}(s,t) + \delta_{1P}(s,t) = 1$.*

Proof. Let $P \in \mathcal{P}$ and $s, t \in S$. We first prove for all $n \in \mathbb{N}$,

$$\Delta^n_{0P}(\top)(s,t) + \Delta^n_{1P}(\bot)(s,t) = 1,$$

by induction on n. In the base case, $n = 0$, we have that

$$\Delta^0_{0P}(\top)(s,t) + \Delta^0_{1P}(\bot)(s,t) = \top(s,t) + \bot(s,t) = 1 + 0 = 1.$$

In the inductive case, we have that $n > 0$. We distinguish the following cases.

- If $(s,t) \in S^2_0$ then
$$\Delta^n_{0P}(\top)(s,t) + \Delta^n_{1P}(\bot)(s,t) = 1 + 0 = 1.$$
- If $(s,t) \in S^2_1$ then
$$\Delta^n_{0P}(\top)(s,t) + \Delta^n_{1P}(\bot)(s,t) = 0 + 1 = 1.$$
- Otherwise, $(s,t) \in S^2_?$. Then
$$\begin{aligned}\Delta^n_{0P}(\top)(s,t) + \Delta^n_{1P}(\bot)(s,t) &= P(s,t) \cdot \Delta^{n-1}_{0P}(\top) + P(s,t) \cdot \Delta^{n-1}_{1P}(\bot) \\ &= P(s,t) \cdot (\Delta^{n-1}_{0P}(\top) + \Delta^{n-1}_{1P}(\bot)) \\ &= P(s,t) \cdot 1 \qquad \text{[induction hypothesis]} \\ &= 1.\end{aligned}$$

Since Δ_{0P} is monotone (Proposition 4.4) and nonexpansive (Proposition 4.8), we can conclude from Theorem 2.11 that $\delta_{0P} = \lim_{n\in\mathbb{N}} \Delta^n_{0P}(\top)$ and $\delta_{1P} = \lim_{n\in\mathbb{N}} \Delta^n_{1P}(\bot)$. Therefore, for all $s, t \in S$ we have that

$$\begin{aligned}\delta_{0P}(s,t) + \delta_{1P}(s,t) &= \left(\lim_{n\in\mathbb{N}} \Delta^n_{0P}(\top)(s,t)\right) + \left(\lim_{n\in\mathbb{N}} \Delta^n_{1P}(\bot)(s,t)\right) \\ &= \lim_{n\in\mathbb{N}} (\Delta^n_{0P}(\top)(s,t) + \Delta^n_{1P}(\bot)(s,t)) \\ &= \lim_{n\in\mathbb{N}} 1 \\ &= 1.\end{aligned}$$

□

4.2 Vertex Policies

For states s and t, the transportation polytope $\Omega(\tau(s), \tau(t))$ is generally infinite. As a result, our game, if formulated in terms of all couplings, is infinite in general. However, as we will see below, we can restrict our attention to the vertices of the transportation polytope $\Omega(\tau(s), \tau(t))$, making our game as well as the set of policies finite.

Definition 4.13. *The set $\mathcal{V}$ of* vertex policies *is defined by*

$$\mathcal{V} = \{\, P \in S^2_? \to \mathcal{D}(S \times S) \mid \forall (s,t) \in S^2_? : P(s,t) \in V(\Omega(\tau(s), \tau(t))) \,\}.$$

Proposition 4.14. *The set $\mathcal{V}$ is nonempty and finite.*

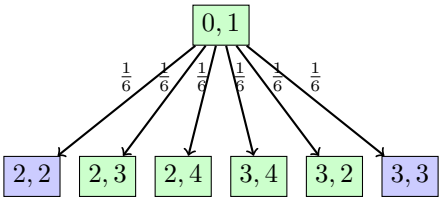
Proof. It suffices to show that for all $(s,t) \in S^2_?$, the set $V(\Omega(\tau(s), \tau(t)))$ is nonempty and finite. Let $(s,t) \in S^2_?$. Since a convex polytope is defined by a finite number of linear inequalities, and each vertex arises from a combinations of these inequalities becoming equalities, it follows that the polytope has only finitely many vertices. Therefore, the set $V(\Omega(\tau(s), \tau(t)))$ is finite. By means of the North-West corner method [10, Chapter 23], one can construct an element of $V(\Omega(\tau(s), \tau(t)))$. □

Even if we restrict ourselves to vertex policies, we can still characterize the distances in terms of reachability probabilities.

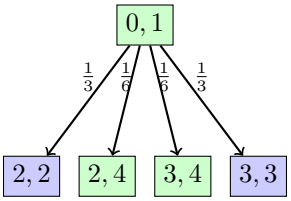
Theorem 4.15 ([52, Proposition 6.1.6]). *There exists $P \in \mathcal{V}$ such that $\delta_{1P} \sqsubseteq \delta_1$.*

Theorem 4.16 ([52, Theorem 6.1.7]). $\delta_1 = \min_{P \in \mathcal{V}} \delta_{1P}$.

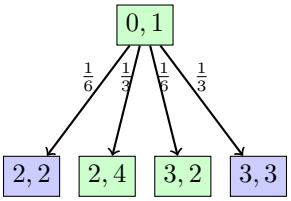
Example 4.17. *Let us focus on the state pair $(0,1)$ and recall the policies Q, R, and Z at this state pair, as shown in Example 4.2. It is easy to verify that $Q(0,1)$, $R(0,1)$, and $Z(0,1)$ belong to the set $\Omega(\tau(0),\tau(1))$. The graphical representation of $Z(0,1)$ is shown as follows.*



Notice that $Z(0,1)$ matches state 2 with state 4, state 4 with state 3, and state 3 with state 2, thus forming a cycle. Hence, by Proposition 3.5, $Z(0,1)$ is not a vertex. The graphical representation of $Q(0,1)$ is shown as follows.



The graphical representation of $R(0,1)$ is shown as follows.



From the above representations, we can conclude that $Q(0,1)$ and $R(0,1)$ belong to the set $V(\Omega(\tau(0),\tau(1)))$.

The next proposition shows that if P is an optimal but not a vertex policy, then a vertex optimal policy V can always be constructed from P by replacing all state pairs that have non-vertex couplings with their corresponding vertex couplings.

Proposition 4.18. *For all $P \in P_{opt} \setminus V_{opt}$ there exists $V \in V_{opt}$ such that for all $(s,t) \in S_?^2$, $\text{support}(V(s,t)) \subseteq \text{support}(P(s,t))$ and $\text{support}(V(u,v)) \subset \text{support}(P(u,v))$ for some $(u,v) \in S_?^2$.*

Proof. Assume that $P \in P_{opt} \setminus V_{opt}$. Then there exists at least one $(s,t) \in S_?^2$ such that $P(s,t) \notin V(\Omega(\tau(s),\tau(t)))$. Since $P \in P_{opt}$, $\delta_1(s,t) = P(s,t) \cdot \delta_1$ by Proposition 4.21 and, hence, $P(s,t) \in \Omega_{\text{opt}}(\tau(s),\tau(t))$. From Proposition 4.24(b) we can conclude that $P(s,t) \notin V(\Omega_{\text{opt}}(\tau(s),\tau(t)))$. Therefore, $P(s,t)$ is the convex cobmination of a set V_{st} of couplings in $V(\Omega_{opyt}(\tau(s),\tau(t)))$, that is, $P(s,t) = \sum_{\omega \in V_{st}} q_\omega\, \omega$, where $q_\omega > 0$ and $\sum_{\omega \in V_{st}} q_\omega = 1$. Therefore,

$$\text{support}(P(s,t)) = \text{support}\left(\sum_{\omega \in V_{st}} q_\omega\, \omega\right) = \bigcup_{\omega \in V_{st}} \text{support}(\omega).$$

As a result, $\text{support}(\omega) \subseteq \text{support}(P(s,t))$ for all $\omega \in V_{st}$. Because $P(s,t)$ is not a vertex of $\Omega(\tau(s),\tau(t))$, its support graph has a cycle by Proposition 3.5. Since all $\omega \in V_{st}$ are vertices of $\Omega(\tau(s),\tau(t))$, their support graphs are acyclic. Hence, $\text{support}(\omega) \subset \text{support}(P(s,t))$ for all $\omega \in V_{st}$. We choose an arbitrary ω_{st} from V_{st}. We construct V as follows:

$$V(s,t) = \begin{cases} P(s,t) & \text{if } P(s,t) \in V(\Omega(\tau(s),\tau(t))) \\ \omega_{st} & \text{otherwise} \end{cases}$$

It remains to show that $V \in V_{opt}$. By Proposition 4.21, it suffices to show that for all $(s,t) \in S_?^2$, we have that $\delta_1(s,t) = V(s,t) \cdot \delta_1$. We distinguish the following two cases.

- If $P(s,t) \in V(\Omega(\tau(s),\tau(t)))$ then $\delta_1(s,t) = P(s,t)\cdot\delta_1 = V(s,t)\cdot\delta_1$ since $P \in Popt$.
- Otherwise,$\delta_1(s,t) = \omega_{st}\cdot\delta_1 = V(s,t)\cdot\delta_1$ since $\omega_{st} \in V(\Omega_{\mathrm{opt}}(\tau(s),\tau(t))$.

□

4.3 Optimal Policies

A policy is called optimal if it captures the distances, that is, the distance between s and t equals the probability of reaching 1-pairs from a state pair (s,t) using policy P.

Definition 4.19. *A policy $P \in \mathcal{P}$ is* optimal *if $\delta_{1P} = \delta_1$.*

Example 4.20. *Let us focus on the state pair $(0,1)$ and recall the policies Q and R at this state pair, as shown in Example 4.2. It is easy to check that $\delta_1(0,1) = \frac{3}{12}$ in the LMC depicted in Figure 4.1, and from Example 4.5, we know that $\delta_{1Q}(0,1) = \frac{3}{12}$. We leave it to the reader to verify that remainging state pairs (u,v) under policy Q satisfies $\delta_{1P}(u,v) = \delta_1(u,v)$. Hence, by Definition 4.19, we can conclude that Q is an optimal policy. In contrast, from Example 4.5, we know that $\delta_{1R}(0,1) = \frac{5}{12}$. Since $\delta_{1R}(0,1) \neq \delta_1(0,1)$, by Definition 4.19, we can conclude that R is not an optimal policy.*

The following proposition is an alternative characterization of optimal policies which turns out to be very useful in several of our proofs.

Proposition 4.21. *For all $P \in \mathcal{P}$, P is optimal if and only if $\delta_1(s,t) = P(s,t)\cdot\delta_1$ for all $(s,t) \in S^2_?$.*

Proof. Let $P \in \mathcal{P}$. We prove two implications. Assume that P is optimal. Let $(s,t) \in S^2_?$. Then

$$\begin{aligned}\delta_1(s,t) &= \delta_{1P}(s,t) && [P \text{ is optimal}]\\ &= \Delta_{1P}(\delta_{1P})(s,t)\\ &= P(s,t)\cdot\delta_{1P}\\ &= P(s,t)\cdot\delta_1 && [P \text{ is optimal}]\end{aligned}$$

To prove the other implication, assume that $\delta_1(s,t) = P(s,t)\cdot\delta_1$ for all $(s,t) \in S^2_?$. According to Theorem 4.6, $\delta_1 \sqsubseteq \delta_{1P}$. Therefore, it remains to show that $\delta_{1P} \sqsubseteq \delta_1$. Since δ_{1P} is the least fixed point of Δ_{1P}, it suffices to show that δ_1 is a fixed point of Δ_{1P}. We distinguish the following cases.

- If $(s,t) \in S^2_0$ then

$$\begin{aligned}\Delta_{1P}(\delta_1)(s,t) &= 0\\ &= \delta_1(s,t) && [\text{Theorem 3.11}]\end{aligned}$$

- If $(s,t) \in S^2_1$ then

$$\begin{aligned}\Delta_{1P}(\delta_1)(s,t) &= 1\\ &= \Delta(\delta_1)(s,t)\\ &= \delta_1(s,t).\end{aligned}$$

- If $(s,t) \in S^2_?$ then

$$\begin{aligned}\Delta_{1P}(\delta_1)(s,t) &= P(s,t)\cdot\delta_1\\ &= \delta_1(s,t) && [\text{by assumption}]\end{aligned}$$

□

According to Theorem 4.6 and 4.16, optimal and optimal vertex policies exist. We denote the set of optimal policies by $\mathcal{P}_{\mathrm{opt}}$ and the set of optimal vertex policies by $\mathcal{V}_{\mathrm{opt}}$.

Proposition 4.22. *The set $\mathcal{V}_{\mathrm{opt}}$ is nonempty and finite.*

Proof. According to Proposition 4.14, the set $\mathcal{V}$ is nonempty and finite. It remains to show that there exists a $P \in \mathcal{V}$ that is optimal. By Proposition 4.21, it suffices to prove that for all $(s,t) \in S^2_?$ there exists $P(s,t) \in V(\Omega(\tau(s),\tau(t)))$ such that $\delta_1(s,t) = P(s,t) \cdot \delta_1$, which immediately follows from Proposition 3.15. □

The next proposition states that for a closed convex polytope $X \subseteq \mathbb{R}^n$, the set of points where a linear function f takes its minimum or maximum value is itself a closed convex polytope. Moreover, the vertices of this set are exactly those vertices of X at which f attains its minimum or maximum.

Proposition 4.23. *Let $X \subseteq \mathbb{R}^n$ be a closed convex polytope and $f : X \to \mathbb{R}$ a linear function.*

(a) If $m = \inf_{x \in X} f(x)$ and $M = \{\, x \in X \mid f(x) = m \,\}$ then M is a closed convex polytope and $V(M) = V(X) \cap M$.

(b) If $m = \sup_{x \in X} f(x)$ and $M = \{\, x \in X \mid f(x) = m \,\}$ then M is a closed convex polytope and $V(M) = V(X) \cap M$.

Proof. Let $X \subseteq \mathbb{R}^n$ be a closed convex polytope and $f : X \to \mathbb{R}$ a linear function. We only prove (a) as (b) can be proved similarly. Assume that $m = \inf_{x \in X} f(x)$. Since a linear function on a convex polytope attains its minimum (at a vertex), we can conclude that $m = \min_{x \in X} f(x)$. Let $M = \{\, x \in X \mid f(x) = m \,\}$.

Next, we show that the set M is a closed convex polytope. Since X is a closed convex polytope, it is the intersection of finitely many closed half spaces. Hence, M is the intersection of those closed half spaces and the closed half spaces defined by the linear inequalities $f(x) \leq m$ and $f(x) \geq m$ and, therefore, a closed convex polytope.

To conclude that $V(M) = V(X) \cap M$, we prove two inclusions. Assume that $x \in V(X) \cap M$. Obviously, $x \in M$. It remains to show that x is a vertex of M. Since $x \in V(X)$, we have that x is the intersection of finitely many half spaces defining the convex polytope X. Each such half space can be defined by a linear inequality. Furthermore, $x \in M$ if and only if $f(x) \leq m$ and $f(x) \geq m$. Hence, x is the intersection of the half spaces defining X as well as the two defined by the inequalities $f(x) \leq m$ and $f(x) \geq m$. Therefore, $x \in V(M)$.

Assume that $x \in V(M)$. Then $x \in M$. It remains to show that x is a vertex of X. Towards a contradiction, assume that x is not a vertex of X. Then $x = q\,y + (1-q)\,z$ for some y, $z \in V(X)$ and $q \in (0,1)$. Since

$$\begin{aligned} m &= f(x) && [x \in M] \\ &= f(q\,y + (1-q)\,z) \\ &= q\,f(y) + (1-q)\,f(z) && [f \text{ is linear}] \\ &\geq q\,m + (1-q)\,m && [m = \min_{x \in X} f(x)] \\ &= m \end{aligned}$$

we can conclude that $f(y) = m$ and $f(z) = m$ and, therefore, y, $z \in V(X) \cap M$. From the first inclusion, that we proved above, we can deduce that y, $z \in V(M)$. But that contradicts $x \in V(M)$. □

We conclude this section by proving that the set of optimal policies forms a closed convex polytope. This result will be used later in the thesis.

To avoid clutter, we use $\Omega_{\mathrm{opt}}(\tau(s),\tau(t))$ to denote the set $\{\, \omega \in \Omega(\tau(s),\tau(t)) \mid \delta_1(s,t) = \omega \cdot \delta_1 \,\}$ and $V(\Omega_{\mathrm{opt}}(\tau(s),\tau(t)))$ to denote the set $\{\, \omega \in V(\Omega(\tau(s),\tau(t))) \mid \delta_1(s,t) = \omega \cdot \delta_1 \,\}$.

Proposition 4.24. *For all $(s,t) \in S^2_?$,*

(a) the set $\Omega_{\mathrm{opt}}(\tau(s),\tau(t))$ is a closed convex polytope, and

(b) for all $\omega \in \Omega(\tau(s),\tau(t))$, $\omega \in V(\Omega(\tau(s),\tau(t)))$ and $\delta_1(s,t) = \omega \cdot \delta_1$ if and only if $\omega \in V(\Omega_{\mathrm{opt}}(\tau(s),\tau(t)))$.

Proof. Let $(s,t) \in S^2_?$. As we have already shown in the proof of Proposition 3.15, the function mapping $\omega \in \Omega(\tau(s),\tau(t))$ to $\omega \cdot \delta_1$ is linear. We have that

$$\inf_{\omega \in \Omega(\tau(s),\tau(t))} \omega \cdot \delta_1 = \Delta_1(\delta_1)(s,t) = \delta_1(s,t).$$

Let $M = \{\, \omega \in \Omega(\tau(s),\tau(t)) \mid \omega \cdot \delta_1 = \delta_1(s,t) \,\}$. Since $\Omega_{\mathrm{opt}}(\tau(s),\tau(t)) = \Omega(\tau(s),\tau(t)) \cap M$, we can conclude from Proposition 4.23 that $\Omega_{\mathrm{opt}}(\tau(s),\tau(t))$ is a closed convex polytope and $V(\Omega_{\mathrm{opt}}(\tau(s),\tau(t))) = V(\Omega(\tau(s),\tau(t))) \cap M$. □

In this chapter, we introduced the concept of policies and their role in defining probabilistic bisimilarity distances. Tang and Van Breugel [53] present an algorithm for computing an optimal vertex policy for a given LMC. As mentioned in the introduction, while such a policy can be seen as an explanation of the distances, we argued that 0-minimal and 1-maximal optimal vertex policies are desirable. Both 0-minimal and 1-maximal policies are defined in terms of the expected lengths of paths to 0-pairs and 1-pairs. In the next chapter, we will define these expected lengths and explore their properties.

5 Expected Length

The foundation of the game studied in this thesis relies on selecting a coupling for each pair of states that is neither a 0-pair nor a 1-pair. To achieve this, a player follows a strategy, also known as a policy, a concept discussed in the previous chapter. Recall that if the states s and t are probabilistic bisimilar, we call (s, t) a 0-pair (depicted in blue) because they have distance zero. If the states have different labels, we call (s, t) a 1-pair (depicted in red). If neither condition holds, the state pair (s, t) is depicted in green. Given an optimal policy P and a pair of states (s, t), we are interested in the expected length of paths from (s, t) to i-pairs when using P. We restrict to only those paths that reach an i-pair, that is, it is a conditional expectation, as we will show through an example. In this chapter, we will formulate the expected length to an i-pair, and in the next two chapters, we will develop algorithms to maximize or minimize these expected lengths.

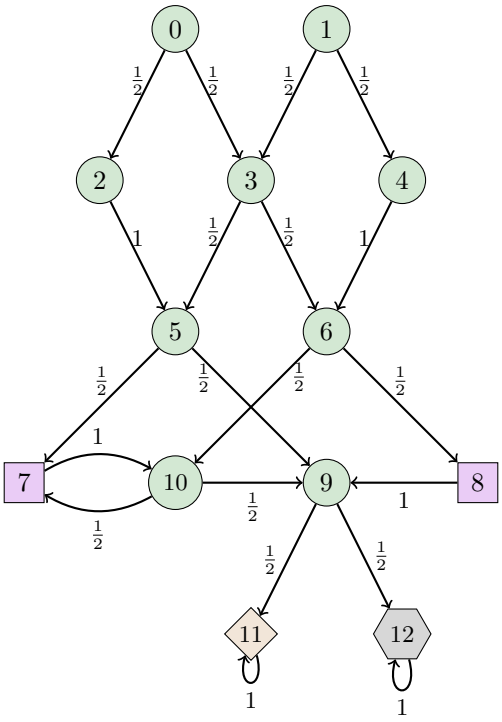


Figure 5.1: An LMC

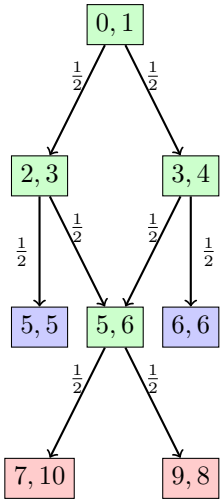


Figure 5.2: Optimal vertex policy of states 0 and 1 of LMC depicted in Figure 5.1

Let $i \in \{0, 1\}$. Given an optimal policy P and a pair of states (s, t), we are interested in the expected length of paths from (s, t) to i-pairs when using P. We restrict ourselves to paths that reach an i-pair, so this is a conditional expectation. For example, consider the policy depicted in Figure 5.2. Let us focus on the 1-pairs: $(7, 10)$ and $(9, 8)$. To reach the state pair $(7, 10)$ starting from the state pair $(0, 1)$, there are 2 paths, each with a probability of $\frac{1}{8}$, and each path has a length of 3. Similarly, to reach the state pair $(9, 8)$ from the state pair $(0, 1)$, there are also 2 paths, each with a probability of $\frac{1}{8}$, and each path has a length of 3. The probability of reaching a 1-pair from state pairs $(0, 1)$ is $\frac{1}{2}$. Hence, the conditional expectation is $\frac{\frac{1}{8}\cdot 3+\frac{1}{8}\cdot 3+\frac{1}{8}\cdot 3+\frac{1}{8}\cdot 3}{\frac{1}{2}} = 3$.

As we have already seen, $\delta_{iP}(s, t)$ captures the probability of all paths that reach an i-pair from (s, t). However, because we restrict ourselves to optimal policies (even though we state many properties more generally) we have that $\delta_{iP}(s, t) = \delta_i(s, t)$. This represents the denominator of the conditional expectation. Since P is optimal, the denominator is independent of P. Therefore, we omit it. Hence, in the following we only compute the numerator.

With abuse of language, we call the numerator the expected length. In this chapter, we will learn how to calculate these values.

Definition 5.1. *For $i \in \{0, 1\}$, the set D_i is defined by*

$$D_i = \{\, (s, t) \in S \times S \mid \delta_i(s, t) \neq 0 \,\}.$$

Note that $S_i^2 \subseteq D_i \subseteq S_i^2 \cup S_?^2$. The expected lengths are defined in terms of the following function.

Definition 5.2. *Let $P \in \mathcal{P}$ and $i \in \{0, 1\}$. The function $\bar{\Lambda}_{iP} : (D_i \to [0, \infty]) \to (D_i \to [0, \infty])$ is defined by*

$$\bar{\Lambda}_{iP}(l)(s, t) = \begin{cases} 0 & \text{if } (s, t) \in S_i^2 \\ \displaystyle\sum_{(u,v)\in D_i} P(s, t)(u, v)\,(\delta_i(u, v) + l(u, v)) & \text{otherwise.} \end{cases}$$

To conclude that $\bar{\Lambda}_{iP}$ has a least fixed point, we apply the Knaster-Tarski fixed point theorem (Theorem 2.7(a)). However, before using this theorem, we first need to prove that $\bar{\Lambda}_{iP}$ is a monotone function, as shown in the following proposition.

Proposition 5.3. *For all $P \in \mathcal{P}$ and $i \in \{0, 1\}$, $\bar{\Lambda}_{iP}$ is monotone.*

Proof. Let $P \in \mathcal{P}$, $i \in \{0, 1\}$, and $k, l \in D_i \to [0, \infty]$. Assume $k \sqsubseteq l$. Let $(s, t) \in D_i$. It suffices to show that $\bar{\Lambda}_{iP}(k)(s, t) \leq \bar{\Lambda}_{iP}(l)(s, t)$. We distinguish the following cases.

- If $(s, t) \in S_i^2$ then
$$\bar{\Lambda}_{iP}(k)(s, t) = 0 = \bar{\Lambda}_{iP}(l)(s, t).$$

- Otherwise,

$$\begin{aligned}\bar{\Lambda}_{iP}(k)(s,t) &= \sum_{(u,v)\in D_i} P(s,t)(u,v)\,(\delta_i(u,v)+k(u,v))\\ &\leq \sum_{(u,v)\in D_i} P(s,t)(u,v)\,(\delta_i(u,v)+l(u,v)) \qquad [k \sqsubseteq l]\\ &= \bar{\Lambda}_{iP}(l)(s,t).\end{aligned}$$

Hence, $\bar{\Lambda}_{iP}(k) \sqsubseteq \bar{\Lambda}_{iP}(l)$. □

We denote the least fixed point of $\bar{\Lambda}_{iP}$ by λ_{iP}.

Proposition 5.4. *For all $P \in \mathcal{P}$ and $i \in \{0,1\}$, if $l \in D_i \to [0,\infty)$ then $\bar{\Lambda}_{iP}(l) \in D_i \to [0,\infty)$.*

Proof. Follows immediately from the definition of $\bar{\Lambda}_{iP}$. □

We denote the restriction of $\bar{\Lambda}_{iP}$ to $D_i \to [0,\infty)$ by Λ_{iP}. By Proposition 5.4, we have that Λ_{iP} is a function from $D_i \to [0,\infty)$ to itself. Below, we will often implicitly use that $\bar{\Lambda}_{iP}(l) = \Lambda_{iP}(l)$ if $l \in D_i \to [0,\infty)$. For example, we may use Proposition 5.3 to conclude that $\Lambda_{iP}(k) \sqsubseteq \Lambda_{iP}(l)$ for $k, l \in D_i \to [0,\infty)$ with $k \sqsubseteq l$, and mention that Λ_{iP} is monotone.

To conclude that Λ_{iP} has a unique fixed point, we apply the Banach's fixed point theorem (Theorem 2.10(c)). However, before using this theorem, we first need to prove that Λ_{iP} is a nonexpansive and power-contractive function, as shown in the following propositions.

Proposition 5.5. *For all $P \in \mathcal{P}$, $i \in \{0,1\}$, $k, l \in D_i \to [0,\infty)$, $(s,t) \in D_i$, and $n \in \mathbb{N}$,*

(a) if $(s,t) \in S_i^2$ then $\Lambda_{iP}^{n+1}(k)(s,t) = \Lambda_{iP}^{n+1}(l)(s,t)$,

(b) if $(s,t) \in D_i \setminus S_i^2$ then $|\Lambda_{iP}^{n+1}(k)(s,t) - \Lambda_{iP}^{n+1}(l)(s,t)| \leq P_?^n(s,t)(S_?^2)\,\|k-l\|$.

Proof. Let $P \in \mathcal{P}$ and $i \in \{0,1\}$. Assume that $k, l \in D_i \to [0,\infty)$ and $(s,t) \in D_i$. We distinguish the following cases.

(a) If $(s,t) \in S_i^2$ then for all $n \in \mathbb{N}$,

$$\Lambda_{iP}^{n+1}(k)(s,t) = 0 = \Lambda_{iP}^{n+1}(l)(s,t).$$

(b) Let $(s,t) \in D_i \setminus S_i^2 \subseteq S_?^2$. Without loss of generality, assume that $\Lambda_{iP}^{n+1}(k)(s,t) \geq \Lambda_{iP}^{n+1}(l)(s,t)$. We prove this case by induction on n. In the base case, when $n = 0$, we have that

$$\begin{aligned}&\Lambda_{iP}(k)(s,t) - \Lambda_{iP}(l)(s,t)\\ &= \left(\sum_{(u,v)\in D_i} P(s,t)(u,v)\,(\delta_i(u,v)+k(u,v))\right) - \left(\sum_{(u,v)\in D_i} P(s,t)(u,v)\,(\delta_i(u,v)+l(u,v))\right)\\ &= \sum_{(u,v)\in D_i} P(s,t)(u,v)\,(k(u,v)-l(u,v))\\ &\leq \sum_{(u,v)\in D_i} P(s,t)(u,v)\,\|k-l\|\\ &\leq \|k-l\|\\ &= P_?^0(s,t)(S_?^2)\,\|k-l\|.\end{aligned}$$

In the inductive case, when $n > 0$, we have that

$$\begin{aligned}
&\Lambda_{iP}^{n+1}(k)(s,t) - \Lambda_{iP}^{n+1}(l)(s,t) \\
=&\left(\sum_{(u,v)\in D_i} P(s,t)(u,v)\left(\delta_i(u,v) + \Lambda_{iP}^{n}(k)(u,v)\right)\right) \\
&-\left(\sum_{(u,v)\in D_i} P(s,t)(u,v)\left(\delta_i(u,v) + \Lambda_{iP}^{n}(l)(u,v)\right)\right) \\
=&\sum_{(u,v)\in D_i} P(s,t)(u,v)\left(\Lambda_{iP}^{n}(k)(u,v) - \Lambda_{iP}^{n}(l)(u,v)\right) \\
=&\sum_{(u,v)\in D_i\setminus S_i^2} P_?(s,t)(u,v)\left(\Lambda_{iP}^{n}(k)(u,v) - \Lambda_{iP}^{n}(l)(u,v)\right) \qquad \text{[part (a)]} \\
\leq&\sum_{(u,v)\in S_?^2} P_?(s,t)(u,v)\left(\Lambda_{iP}^{n}(k)(u,v) - \Lambda_{iP}^{n}(l)(u,v)\right) \qquad [D_i \setminus S_i^2 \subseteq S_?^2] \\
\leq&\sum_{(u,v)\in S_?^2} P_?(s,t)(u,v) P_?^{n-1}(u,v)(S_?^2)\,\|k-l\| \qquad \text{[induction hypothesis]} \\
=&P_?^n(s,t)(S_?^2)\,\|k-l\|.
\end{aligned}$$

□

Proposition 5.6. *For all $P \in \mathcal{P}_{\mathrm{opt}}$ and $i \in \{0,1\}$, there exists $n \in \mathbb{N}$ such that Λ_{iP}^n is contractive.*

Proof. Let $P \in \mathcal{P}_{\mathrm{opt}}$ and $i \in \{0,1\}$. Since Δ_{iP} is monotone (see Proposition 4.4) and nonexpansive (Proposition 4.8), we can conclude from Theorem 2.11(a) that

$$\delta_{iP} = \lim_{n\in\mathbb{N}} \Delta_{iP}^n(\bot). \tag{5.1}$$

Let $(s,t) \in D_i$. Hence, $\delta_i(s,t) \neq 0$. Since P is optimal, for $(s,t) \in D_1$ we have that $\delta_{1P}(s,t) \neq 0$. Since P is optimal, for $(s,t) \in D_0$ we have that $\delta_{1P}(s,t) \neq 1$. From Proposition 4.12, we can conclude that for $(s,t) \in D_0$ we have that $\delta_{0P}(s,t) \neq 0$. Therefore, $\delta_{iP}(s,t) \neq 0$. Hence, from (5.1) we can conclude that exists $n_{st} \in \mathbb{N}$ such that for all $n \geq n_{st}$ we have that $\Delta_{iP}^{n+1}(\bot)(s,t) > 0$. By Proposition 4.9, $P_?^n(s,t)(S_?^2) < 1$.

Let $n = (\max_{(s,t)\in D_i} n_{st}) + 1$ and $c = \max_{(s,t)\in D_i} P_?^n(s,t)(S_?^2)$. Note that $c < 1$. We conclude this proof by showing that Λ_{iP}^n is c-contractive. Let $k,\, l \in S \times S \to [0,\infty)$ and $(s,t) \in D_i$. It suffices to show that $|\Lambda_{iP}^n(k)(s,t) - \Lambda_{iP}^n(l)(s,t)| \leq c\,\|k-l\|$. We distinguish the following cases.

- If $(s,t) \in S_i^2$ then

$$\begin{aligned}
|\Lambda_{iP}^n(k)(s,t) - \Lambda_{iP}^n(l)(s,t)| &= 0 \qquad \text{[Proposition 5.5(a)]} \\
&\leq c\,\|k-l\|.
\end{aligned}$$

- Let $(s,t) \in D_i \setminus S_i^2$. Then $n \geq n_{st} + 1$ and

$$\begin{aligned}
|\Lambda_{iP}^n(k)(s,t) - \Lambda_{iP}^n(l)(s,t)| &\leq P_?^{n-1}(s,t)(S_?^2)\,\|k-l\| \qquad \text{[Proposition 5.5(b)]} \\
&\leq c\,\|k-l\| \qquad [n-1 \geq n_{st}]
\end{aligned}$$

□

From the above propositions, we can conclude that Λ_{iP} is a power-contraction. Next, we show that Λ_{iP} is nonexpansive.

Proposition 5.7. *For all $P \in \mathcal{P}$ and $i \in \{0,1\}$, Λ_{iP} is nonexpansive.*

Proof. Let $P \in \mathcal{P}$, $i \in \{0,1\}$, $k,\, l \in S \times S \to [0,\infty)$, and $s,\, t \in S$. It suffices to show that $|\Lambda_{iP}(k)(s,t) - \Lambda_{iP}(l)(s,t)| \leq \|k-l\|$. Without loss of generality, assume that $\Lambda_{iP}(k)(s,t) \geq \Lambda_{iP}(l)(s,t)$. We distinguish two cases.

- If $(s,t) \in S_i^2$ then from Proposition 5.5(a) we can conclude that

$$\Lambda_{iP}(k)(s,t) - \Lambda_{iP}(l)(s,t) = 0 \leq \|k-l\|.$$

- If $(s,t) \in D_i \setminus S_i^2$ then

$$\begin{aligned}
&\Lambda_{iP}(k)(s,t) - \Lambda_{iP}(l)(s,t) \\
&= P(s,t) \cdot (\delta_i + k) - P(s,t) \cdot (\delta_i + l) \\
&= \left(\sum_{(u,v) \in D_i} P(s,t)(u,v) \left(\delta_i(u,v) + k(u,v)\right) \right) - \left(\sum_{(u,v) \in D_i} P(s,t)(u,v) \left(\delta_i(u,v) + l(u,v)\right) \right) \\
&= \sum_{(u,v) \in D_i} P(s,t)(u,v) \left(k(u,v) - l(u,v)\right) \\
&\leq \sum_{(u,v) \in D_i} P(s,t)(u,v) \, \|k-l\| \\
&\leq \|k-l\|.
\end{aligned}$$

□

Since Λ_{iP} is a nonexpansive and a power-contraction, we can conclude the following proposition.

Proposition 5.8. *For all $P \in \mathcal{P}_{\text{opt}}$ and $i \in \{0,1\}$, λ_{iP} is the unique fixed point of Λ_{iP}.*

Proof. Let $P \in \mathcal{P}_{\text{opt}}$ and $i \in \{0,1\}$. Since the set S is finite and, therefore, the set D_i is finite, each function $l \in D_i \to [0,\infty)$ is bounded. The set of bounded functions $D_i \to [0,\infty)$ endowed with the infinity norm forms a complete metric space. Hence, Λ_{iP} is a function from the nonempty complete metric space $D_i \to [0,\infty)$ to itself.

Because Λ_{iP}^n is contractive for some $n \in \mathbb{N}$ by Proposition 5.6 and Λ_{iP} is nonexpansive by Proposition 5.7, we can conclude from Theorem 2.10(c) that Λ_{iP} has a unique fixed point, say l. Hence, l is a fixed point of $\bar{\Lambda}_{iP}$. Since λ_{iP} is the least fixed point of $\bar{\Lambda}_{iP}$, we have that $\lambda_{iP} \sqsubseteq l$. Since l is bounded, λ_{iP} is bounded as well. Therefore, λ_{iP} is a fixed point of Λ_{iP}, its unique fixed point. □

Corollary 5.9. *For all $P \in \mathcal{P}_{\text{opt}}$ and $i \in \{0,1\}$, $\lambda_{iP} \in D_i \to [0,\infty)$.*

Proof. Immediate consequence of Proposition 5.8. □

In the next example, we will apply Definition 5.2 to calculate the expected length to 1-pairs and 0-pairs for the state pair $(0,1)$ of the policy depicted in Figure 5.2.

Example 5.10. *Consider the state pairs depicted in the LMC in Figure 5.1. The reader can easily verify the following.*

$$\begin{aligned}
\delta_1(7,10) &= 1 & \qquad \delta_0(6,6) &= 1 \\
\delta_1(9,8) &= 1 & \qquad \delta_0(5,5) &= 1 \\
\delta_1(5,6) &= 1 & \qquad \delta_0(5,6) &= 0 \\
\delta_1(2,3) &= \frac{1}{2} & \qquad \delta_0(2,3) &= \frac{1}{2} \\
\delta_1(3,4) &= \frac{1}{2} & \qquad \delta_0(3,4) &= \frac{1}{2}
\end{aligned}$$

Consider the policy depicted in Figure 5.2. The expected length to a 1-pair for the state pair $(0,1)$ is calculated

as follows.

$$
\begin{aligned}
\lambda_{1P}(7,10) &= 0 \textit{ as } (7,10) \in S_1^2 \\
\lambda_{1P}(9,8) &= 0 \textit{ as } (9,8) \in S_1^2 \\
\lambda_{1P}(5,6) &= \Lambda_{1P}(\lambda_{1P})(5,6) \\
&= \sum_{(u,v)\in D_1} P(5,6)(u,v)\left(\delta_1(u,v) + \lambda_{1P}(u,v)\right) \\
&= P(5,6)(7,10)\left(\delta_1(7,10) + \lambda_{1P}(7,10)\right) + P(5,6)(9,8)\left(\delta_1(9,8) + \lambda_{1P}(9,8)\right) \\
&= \frac{1}{2}(1+0) + \frac{1}{2}(1+0) \\
&= 1 \\
\lambda_{1P}(2,3) &= \Lambda_{1P}(\lambda_{1P})(2,3) \\
&= \sum_{(u,v)\in D_1} P(2,3)(u,v)\left(\delta_1(u,v) + \lambda_{1P}(u,v)\right) \\
&= P(2,3)(5,6)\left(\delta_1(5,6) + \lambda_{1P}(5,6)\right) \\
&= \frac{1}{2}(1+1) \\
&= 1 \\
\lambda_{1P}(3,4) &= \Lambda_{1P}(\lambda_{1P})(3,4) \\
&= \sum_{(u,v)\in D_1} P(3,4)(u,v)\left(\delta_1(u,v) + \lambda_{1P}(u,v)\right) \\
&= P(3,4)(5,6)\left(\delta_1(5,6) + \lambda_{1P}(5,6)\right) \\
&= \frac{1}{2}(1+1) \\
&= 1 \\
\lambda_{1P}(0,1) &= \Lambda_{1P}(\lambda_{1P})(0,1) \\
&= \sum_{(u,v)\in D_1} P(0,1)(u,v)\left(\delta_1(u,v) + \lambda_{1P}(u,v)\right) \\
&= P(0,1)(2,3)\left(\delta_1(2,3) + \lambda_{1P}(2,3)\right) + P(0,1)(3,4)\left(\delta_1(3,4) + \lambda_{1P}(3,4)\right) \\
&= \frac{1}{2}\left(\frac{1}{2}+1\right) + \frac{1}{2}\left(\frac{1}{2}+1\right) \\
&= \frac{3}{2}
\end{aligned}
$$

The expected length to a 0-pair for the state pair $(0,1)$ *is calculated as follows.*

$$\begin{aligned}
\lambda_{0P}(5,5) &= 0 \text{ as } (5,5) \in S_0^2 \\
\lambda_{0P}(6,6) &= 0 \text{ as } (6,6) \in S_0^2 \\
\lambda_{0P}(2,3) &= \Lambda_{0P}(\lambda_{0P})(2,3) \\
&= \sum_{(u,v)\in D_0} P(2,3)(u,v)\left(\delta_0(u,v) + \lambda_{0P}(u,v)\right) \\
&= P(2,3)(5,5)\left(\delta_0(5,5) + \lambda_{0P}(5,5)\right) \\
&= \frac{1}{2}(1+0) \\
&= \frac{1}{2} \\
\lambda_{0P}(3,4) &= \Lambda_{0P}(\lambda_{0P})(3,4) \\
&= \sum_{(u,v)\in D_0} P(3,4)(u,v)\left(\delta_0(u,v) + \lambda_{0P}(u,v)\right) \\
&= P(3,4)(6,6)\left(\delta_0(6,6) + \lambda_{0P}(6,6)\right) \\
&= \frac{1}{2}(1+0) \\
&= \frac{1}{2} \\
\lambda_{0P}(0,1) &= \Lambda_{0P}(\lambda_{0P})(0,1) \\
&= \sum_{(u,v)\in D_0} P(0,1)(u,v)\left(\delta_0(u,v) + \lambda_{0P}(u,v)\right) \\
&= P(0,1)(2,3)\left(\delta_0(2,3) + \lambda_{0P}(2,3)\right) + P(0,1)(3,4)\left(\delta_0(3,4) + \lambda_{0P}(3,4)\right) \\
&= \frac{1}{2}\left(\frac{1}{2} + \frac{1}{2}\right) + \frac{1}{2}\left(\frac{1}{2} + \frac{1}{2}\right) \\
&= 1
\end{aligned}$$

In this chapter, we defined the expected length to an i-pair starting from a state pair (s,t)(s,t) under a given policy. One might think that computing these expected lengths is straightforward. However, some policies—such as those depicted in Figure 1.10—may contain loops, which makes their computation slightly more intricate. Recall that the player's objective in the game studied in this thesis is to avoid reaching 1-pairs, meaning that, among all the optimal policies, the expected length to reach a 1-pair should be maximized. In the following chapter, we will develop an algorithm that, starting from an optimal policy, computes a 1-maximal optimal policy. Furthermore, we will prove an exponential lower bound for this algorithm by constructing an LMC of size $O(n)$, for which the algorithm requires $\Omega(2^n)$ iterations.

6 1-Maximal Policies

The objective of the player in the game studied in this thesis is to avoid reaching 1-pairs. This means that, among all optimal policies, the expected length to reach a 1-pair should be maximized. We refer to an optimal policy that maximizes the expected length to a 1-pair as a 1-maximal policy. In this chapter, we will formulate the 1-maximal policies and illustrate with an example that these policies are not unique. Moreover, we will present an algorithm that, starting from an optimal policy computed using the Tang and Van Breugel [53] algorithm, computes a 1-maximal optimal policy. We will also prove its correctness and termination. Furthermore, we will prove an exponential lower bound for this algorithm by constructing an LMC of size $O(n)$, for which the algorithm requires $\Omega(2^n)$ iterations. Finally, recalling the concept of 1-conflict from the introduction, we will conclude this chapter by proving that a 1-maximal optimal policy does not have any 1-conflicts.

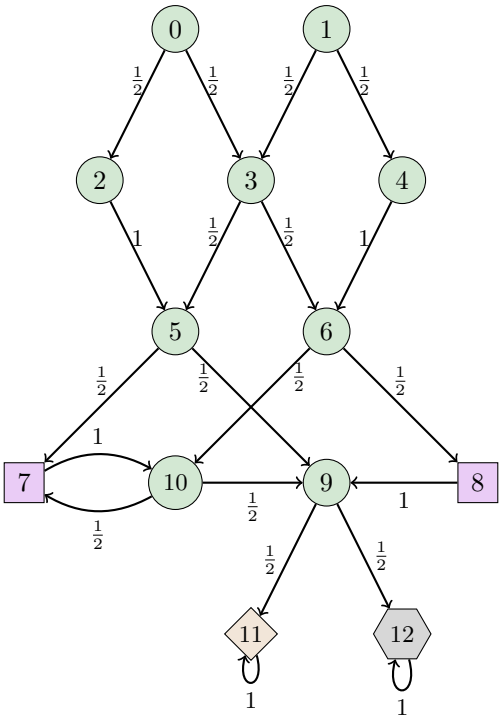


Figure 6.1: An LMC

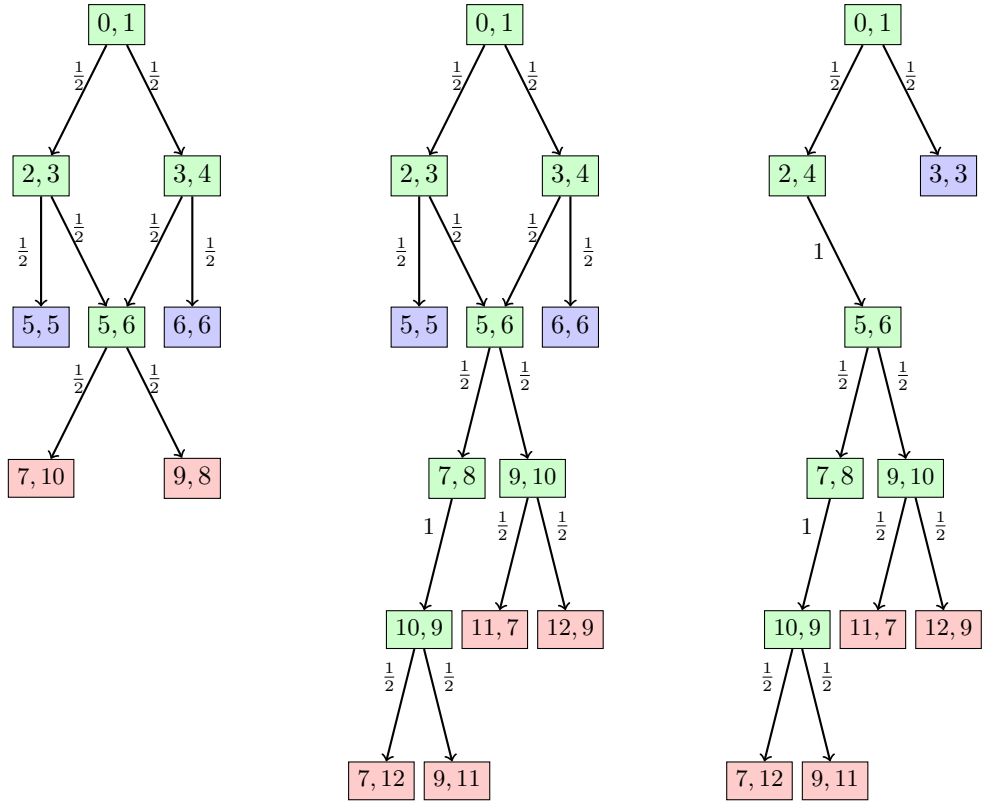


Figure 6.2: Three optimal policies for states 0 and 1 in the LMC depicted in Figure 6.1: one vertex policy, and two vertex 1-maximal policies

For example, consider the policies depicted in Figure 6.2. All of the policies are optimal, as each reaches a 1-pair with a probability of $\frac{1}{2}$, which represents the distance between states 0 and 1. The expected length to a 1-pair in the left policy is 1.5, while it is 2.5 in both the middle and right policies. As illustrated in Figure 6.2, optimal vertex policies that maximize the expected length to 1-pairs are not unique. This maximal expected length is defined as follows.

Definition 6.1. *The function* $\lambda_1 : D_1 \to [0, \infty]$ *is defined by*

$$\lambda_1(s,t) = \sup_{P \in \mathcal{P}_{\text{opt}}} \lambda_{1P}(s,t).$$

An optimal policy that realizes that the maximal expected length is called 1-maximal.

Definition 6.2. *A policy* $P \in \mathcal{P}_{\text{opt}}$ *is* 1-maximal *if* $\lambda_{1P} = \lambda_1$.

We denote the set of optimal 1-maximal policies by $\mathcal{P}_{\text{opt}}^{\max}$ and the set of optimal 1-maximal vertex policies by $\mathcal{V}_{\text{opt}}^{\max}$. To avoid clutter, instead of $\sum_{(u,v)\in D_1} \omega(u,v)\,(\delta_1(u,v) + l(u,v))$ we write $\omega \cdot (\delta_1 + l)$. Notice that the summation is over elements in D_1. In the next proposition, we provide an alternative characterization of 1-maximal that we use in several of our proofs.

Proposition 6.3. *For all* $P \in \mathcal{P}_{\text{opt}}$, P *is 1-maximal if and only if* $\lambda_1(s,t) = P(s,t) \cdot (\delta_1 + \lambda_1)$ *for all* $(s,t) \in D_1 \setminus S_1^2$.

Proof. Let $P \in \mathcal{P}_{\text{opt}}$. We prove two implications. Assume that P is 1-maximal. Let $(s,t) \in D_1 \setminus S_1^2$. Then

$$\begin{aligned}
\lambda_1(s,t) &= \lambda_{1P}(s,t) && [P \text{ is 1-maximal}] \\
&= \Lambda_{1P}(\lambda_{1P})(s,t) \\
&= P(s,t) \cdot (\delta_1 + \lambda_{1P}) \\
&= P(s,t) \cdot (\delta_1 + \lambda_1) && [P \text{ is 1-maximal}]
\end{aligned}$$

To prove the other implication, assume that $\lambda_1(s,t) = P(s,t)\cdot(\delta_1+\lambda_1)$ for all $(s,t)\in D_1\setminus S_1^2$. Since λ_{1P} is the unique fixed point of Λ_{1P} (Proposition 5.8), it suffices to show that λ_1 is a fixed point of Λ_{1P}. We distinguish the following cases.

- If $(s,t)\in S_1^2$ then

$$\begin{aligned}\Lambda_{1P}(\lambda_1)(s,t) &= 0\\ &= \sup_{Q\in\mathcal{P}_{\mathrm{opt}}} \Lambda_{1Q}(\lambda_{1Q})(s,t)\\ &= \sup_{Q\in\mathcal{P}_{\mathrm{opt}}} \lambda_{1Q}(s,t)\\ &= \lambda_1(s,t).\end{aligned}$$

- If $(s,t)\in D_1\setminus S_1^2$ then

$$\begin{aligned}\Lambda_{1P}(\lambda_1)(s,t) &= P(s,t)\cdot(\delta_1+\lambda_1)\\ &= \lambda_1(s,t) \qquad \text{[by assumption]}\end{aligned}$$

□

The next proposition proves that Λ_{1P} attains its maximum at a vertex.

Proposition 6.4. *For all $l\in D_1\to[0,\infty)$ and $(s,t)\in D_1$,*

$$\sup_{\omega\in\Omega_{\mathrm{opt}}(\tau(s),\tau(t))} \omega\cdot(\delta_1+l) = \max_{\omega\in V(\Omega_{\mathrm{opt}}(\tau(s),\tau(t)))} \omega\cdot(\delta_1+l).$$

Proof. There are two differences between the left hand side and the right hand side: $\sup$ versus $\max$ and $\Omega_{\mathrm{opt}}(\tau(s),\tau(t))$ versus $V(\Omega_{\mathrm{opt}}(\tau(s),\tau(t)))$.

Let $l\in D_1\to[0,\infty)$ and $(s,t)\in D_1$. Let X be the closed convex polytope and $f: X\to[0,1]$ the linear function of the proof of Proposition 3.15. Let $m=\inf_{x\in X} f(x)$. Then

$$m = \inf_{\omega\in\Omega(\tau(s),\tau(t))} \omega\cdot\delta_1 = \Delta(\delta_1)(s,t) = \delta_1(s,t).$$

Let $M=\{\,x\in X\mid f(x)=m\,\}$. By Proposition 4.23(a), M is a closed convex polytope and $V(M)=V(X)\cap M$.

Let $l\in D_1\to[0,\infty)$. Let the function $g: M\to[0,\infty)$ be defined by

$$g(\omega)=\omega\cdot(\delta_1+l).$$

Next, we show that g is linear. Let $\omega,\pi\in M$ and $q\in(0,1)$. We have that

$$\begin{aligned}g(q\,\omega+(1-q)\,\pi) &= (q\,\omega+(1-q)\,\pi)\cdot(\delta_1+l)\\ &= (q\,\omega\cdot(\delta_1+l)) + ((1-q)\pi\cdot(\delta_1+l))\\ &= q\,g(\omega)+(1-q)\,g(\pi).\end{aligned}$$

Because a linear function on a convex polytope attains its maximum at a vertex, we can conclude that the linear function g attains its maximum at $V(M)=V(X)\cap M$, which is the desired result. □

If λ_{1P} is a strict post-fixed point of Λ_{1Q} for some policies P and Q, then the following proposition shows that λ_{1P} is less than the fixed point of Λ_{1Q}.

Proposition 6.5. *For all $P,Q\in\mathcal{P}$, if $\lambda_{1P}\sqsubset\Lambda_{1Q}(\lambda_{1P})$ then $\lambda_{1P}\sqsubset\lambda_{1Q}$.*

Proof. Let $P,Q\in\mathcal{P}$. We split the proof in two parts. First, we show that if $\lambda_{1P}\sqsubseteq\Lambda_{1Q}(\lambda_{1P})$ then $\lambda_{1P}\sqsubseteq\lambda_{1Q}$. Assume that $\lambda_{1P}\sqsubseteq\Lambda_{1Q}(\lambda_{1P})$. We first show that for all $n\in\mathbb{N}$, $\lambda_{1P}\sqsubseteq\Lambda_{1Q}^n(\lambda_{1P})$ by induction on n. The base case, $n=0$, is trivial. In the inductive case, $n>0$, we have that

$$\begin{aligned}\lambda_{1P} &\sqsubseteq \Lambda_{1Q}(\lambda_{1P}) \qquad \text{[assumption]}\\ &\sqsubseteq \Lambda_{1Q}(\Lambda_{1Q}^{n-1}(\lambda_{1P})) \qquad \text{[induction hypothesis and } \Lambda_{1Q} \text{ is monotone (Proposition 5.3)]}\\ &= \Lambda_{1Q}^n(\lambda_{1P}).\end{aligned}$$

From the above and Theorem 2.10(d) we can conclude that

$$\lambda_{1P} \sqsubseteq \lim_{n\in\mathbb{N}} \Lambda_{1Q}^n(\lambda_{1P}) = \lambda_{1Q}.$$

It remains to show that $\lambda_{1P} \neq \Lambda_{1Q}(\lambda_{1P})$ implies $\lambda_{1P} \neq \lambda_{1Q}$. We prove the contrapositive, that is, $\lambda_{1P} = \lambda_{1Q}$ implies $\lambda_{1P} = \Lambda_{1Q}(\lambda_{1P})$. That is, λ_{1Q} is a fixed point of Λ_{1Q}, which holds by definition. □

The following proposition proves that for every optimal policy P, there exists an optimal vertex policy Q such that λ_{1P} is less than equal to λ_{1Q}.

Proposition 6.6. *For all $P \in \mathcal{P}_{\mathrm{opt}}$ there exists $Q \in \mathcal{V}_{\mathrm{opt}}$ such that $\lambda_{1P} \sqsubseteq \lambda_{1Q}$.*

Proof. Let $P \in \mathcal{P}_{\mathrm{opt}}$. Let $(s,t) \in S_?^2$ and, by Proposition 6.4, define

$$Q(s,t) \in \operatorname*{arg\,max}_{\omega\in V(\Omega_{\mathrm{opt}}(\tau(s),\tau(t)))} \omega \cdot (\delta_1 + \lambda_{1P}).$$

By Proposition 4.21, $Q \in \mathcal{V}_{\mathrm{opt}}$. Since Λ_{1Q}^m is contractive for some $m \in \mathbb{N}$ (Proposition 5.5) and Λ_{1Q} is non-expansive (Proposition 5.7) and λ_{1Q} is the unique fixed point of Λ_{1Q}, we can conclude from Theorem 2.10(d) that $\lambda_{1Q} = \lim_{n\in\mathbb{N}} \Lambda_{1Q}^n(\lambda_{1P})$. Hence, to conclude that $\lambda_{1P} \sqsubseteq \lambda_{1Q}$, it suffice to show that for all $n \in \mathbb{N}$, $\lambda_{1P} \sqsubseteq \Lambda_{1Q}^n(\lambda_{1P})$. We prove this by induction on n. The base case, $n = 0$, is trivial. Let $n > 0$ and $(s,t) \in D_1$. We distinguish the following cases.

- If $(s,t) \in S_1^2$ then
$$\lambda_{1P}(s,t) = \Lambda_{1P}(\lambda_{1P})(s,t) = 0 \leq \Lambda_{1Q}^n(\lambda_{1P})(s,t).$$
- If $(s,t) \in D_1 \setminus S_1^2$ then

$$\begin{aligned}
\lambda_{1P}(s,t) &= \Lambda_{1P}(\lambda_{1P})(s,t) \\
&= P(s,t) \cdot (\delta_1 + \lambda_{1P}) \\
&\leq Q(s,t) \cdot (\delta_1 + \lambda_{1P}) \\
&= \Lambda_{1Q}(\lambda_{1P})(s,t) \\
&\leq \Lambda_{1Q}(\Lambda_{1Q}^{n-1}(\lambda_{1P}))(s,t) \qquad [\Lambda_{1Q} \text{ is monotone (Proposition 5.3) and induction hypothesis}] \\
&= \Lambda_{1Q}^n(\lambda_{1P})(s,t).
\end{aligned}$$

□

Corollary 6.7. *For all $(s,t) \in D_1$, $\lambda_1(s,t) = \max_{P\in\mathcal{V}_{\mathrm{opt}}} \lambda_{1P}(s,t)$.*

Proof. Immediate consequence of Proposition 4.22 and 6.6. □

Corollary 6.8. $\lambda_1 \in D_1 \to [0,\infty)$.

Proof. Immediate consequence of Corollary 5.9 and 6.7. □

Below we present a policy iteration algorithm that computes a 1-maximal optimal vertex policy from an optimal vertex policy. The following function serves as the foundation for the algorithm.

Definition 6.9. *The function $\Lambda_1 : (D_1 \to [0,\infty)) \to (D_1 \to [0,\infty))$ is defined by*

$$\Lambda_1(l)(s,t) = \begin{cases} 0 & \text{if } (s,t) \in S_1^2 \\ \sup\limits_{\omega\in\Omega_{\mathrm{opt}}(\tau(s),\tau(t))} \omega \cdot (\delta_1 + l) & \text{otherwise.} \end{cases}$$

In the next proposition, we establish that Λ_1 is a monotone function, which is one of the key ingredients to prove that it has a unique fixed point.

Proposition 6.10. *For all $k, l \in D_1 \to [0,\infty]$, if $k \sqsubseteq l$ then $\Lambda_1(k) \sqsubseteq \Lambda_1(l)$.*

Proof. Let $k, l \in D_1 \to [0,\infty]$. Assume $k \sqsubseteq l$. Let $(s,t) \in D_1$. It suffices to show that $\Lambda_1(k)(s,t) \leq \Lambda_1(l)(s,t)$. We distinguish the following cases.

- If $(s,t) \in S_1^2$ then
$$\Lambda_1(k)(s,t) = 0 = \Lambda_1(l)(s,t).$$
- Otherwise,
$$\begin{aligned}\Lambda_1(k)(s,t) &= \sup_{\omega \in \Omega_{\text{opt}}(\tau(s),\tau(t))} \sum_{(u,v)\in D_1} \omega(u,v)\,(\delta_1(u,v) + k(u,v)) \\ &\leq \sup_{\omega \in \Omega_{\text{opt}}(\tau(s),\tau(t))} \sum_{(u,v)\in D_1} \omega(u,v)\,(\delta_1(u,v) + l(u,v)) \qquad [k \sqsubseteq l] \\ &= \Lambda_1(l)(s,t).\end{aligned}$$

Hence, $\Lambda_1(k) \sqsubseteq \Lambda_1(l)$. □

In the next proposition, we establish that Λ_1 is a nonexpansive function, which we later use to prove some properties of symmetric policies.

Corollary 6.11. *Λ_1 is nonexpansive.*

Proof. Let $k, l \in S \times S \to [0,\infty)$, and $(s,t) \in D_1$. It suffices to show that $|\Lambda_1(k)(s,t) - \Lambda_1(l)(s,t)| \leq \|k - l\|$. Without loss of generality assume $\Lambda_1(k)(s,t) \geq \Lambda_1(l)(s,t)$. We distinguish two cases.

- If $(s,t) \in S_1^2$ then we can conclude that
$$\Lambda_1(k)(s,t) - \Lambda_1(l)(s,t) = 0 \leq \|k - l\|.$$
- If $(s,t) \in D_1 \setminus S_1^2$ then we have that
$$\begin{aligned}&\Lambda_1(k)(s,t) - \Lambda_1(l)(s,t) \\ &= \sup_{\omega \in \Omega_{\text{opt}}(\tau(s),\tau(t))} \omega \cdot (\delta_1 + k) - \sup_{\pi \in \Omega_{\text{opt}}(\tau(s),\tau(t))} \pi \cdot (\delta_1 + l) \\ &= \max_{\omega \in \Omega_{\text{opt}}(\tau(s),\tau(t))} \omega \cdot (\delta_1 + k) - \max_{\pi \in \Omega_{\text{opt}}(\tau(s),\tau(t))} \pi \cdot (\delta_1 + l) \qquad \text{[Proposition 6.4]} \\ &= \max_{\omega \in \Omega_{\text{opt}}(\tau(s),\tau(t))} \min_{\pi \in \Omega_{\text{opt}}(\tau(s),\tau(t))} \omega \cdot (\delta_1 + k) - \pi \cdot (\delta_1 + l) \\ &\leq \max_{\omega \in \Omega_{\text{opt}}(\tau(s),\tau(t))} \omega \cdot (\delta_1 + k) - \omega \cdot (\delta_1 + l) \\ &= \max_{\omega \in \Omega_{\text{opt}}(\tau(s),\tau(t))} \omega \cdot (k - l) \\ &= \max_{\omega \in \Omega_{\text{opt}}(\tau(s),\tau(t))} \sum_{(u,v)\in D_1} \omega(u,v)\,(k(u,v) - l(u,v)) \\ &\leq \max_{\omega \in \Omega_{\text{opt}}(\tau(s),\tau(t))} \sum_{(u,v)\in D_1} \omega(u,v)\,\|k - l\| \\ &= \|k - l\|.\end{aligned}$$

□

The next proposition shows the relation between Λ_{1P} and Λ_1.

Proposition 6.12. *For all $l \in D_1 \to [0,\infty)$,*

(a) for all $P \in \mathcal{P}_{\text{opt}}$, $\Lambda_{1P}(l) \sqsubseteq \Lambda_1(l)$ and

(b) there exists $P \in \mathcal{V}_{\text{opt}}$ such that $\Lambda_1(l) = \Lambda_{1P}(l)$.

Proof. Let $l \in D_1 \to [0,\infty)$.

(a) Let $P \in \mathcal{P}_{\text{opt}}$ and $(s,t) \in D_1 \setminus S_1^2$. We first show that $P(s,t) \in \Omega_{\text{opt}}(\tau(s),\tau(t))$. Since $P \in \mathcal{P}_{\text{opt}}$ it follows from Proposition 4.21 that $\delta_1(s,t) = P(s,t) \cdot \delta_1$. Therefore, can conclude that

$$P(s,t) \in \Omega_{\text{opt}}(\tau(s),\tau(t)) \tag{6.1}$$

Let $(s,t) \in D_1$. We distinguish two cases.

- If $(s,t) \in S_1^2$ then
$$\Lambda_{1P}(l)(s,t) = 0 = \Lambda_1(l)(s,t).$$
- If $(s,t) \in D_1 \setminus S_1^2$ then
$$\begin{aligned}\Lambda_{1P}(l)(s,t) &= P(s,t) \cdot (\delta_1 + l) \\ &\leq \sup_{\omega \in \Omega_{\text{opt}}(\tau(s),\tau(t))} \omega \cdot (\delta_1 + l) \qquad [(6.1)] \\ &= \Lambda_1(l)(s,t).\end{aligned}$$

(b) Let $(s,t) \in S_?^2$ and, by Proposition 6.4, define

$$P(s,t) \in \operatorname*{arg\,max}_{\omega \in V(\Omega_{\text{opt}}(\tau(s),\tau(t)))} \omega \cdot (\delta_1 + l).$$

By Proposition 4.21, $P \in \mathcal{V}_{\text{opt}}$. Let $(s,t) \in D_1$. We distinguish two cases.

- If $(s,t) \in S_1^2$ then
$$\Lambda_1(l)(s,t) = 0 = \Lambda_{1P}(l)(s,t).$$
- If $(s,t) \in D_1 \setminus S_1^2$ then
$$\begin{aligned}\Lambda_1(l)(s,t) &= \sup_{\omega \in \Omega_{\text{opt}}(\tau(s),\tau(t))} \omega \cdot (\delta_1 + l) \\ &= \max_{\omega \in V(\Omega_{\text{opt}}(\tau(s),\tau(t)))} \omega \cdot (\delta_1 + l) \qquad [\text{Proposition 6.4}] \\ &= P(s,t) \cdot (\delta_1 + l) \\ &= \Lambda_{1P}(l)(s,t).\end{aligned}$$

□

The next proposition shows that λ_1 is a post-fixed point of Λ_1.

Proposition 6.13. $\lambda_1 \sqsubseteq \Lambda_1(\lambda_1)$.

Proof. Since for all $P \in \mathcal{P}_{\text{opt}}$,

$$\begin{aligned}\lambda_{1P} &= \Lambda_{1P}(\lambda_{1P}) \\ &\sqsubseteq \Lambda_1(\lambda_{1P}) \qquad [\text{Proposition 6.12(a)}] \\ &\sqsubseteq \Lambda_1(\lambda_1) \qquad [\lambda_{1P} \sqsubseteq \lambda_1 \text{ and } \Lambda_1 \text{ is monotone (Proposition 6.10)}]\end{aligned}$$

we can conclude that $\lambda_1 \sqsubseteq \Lambda_1(\lambda_1)$. □

The next proposition shows the link between Λ_1 and λ_1.

Proposition 6.14. *For all $l \in D_1 \to [0,\infty)$,*

(a) if $\Lambda_1(l) \sqsubseteq l$ then $\lambda_1 \sqsubseteq l$ and

(b) if $l \sqsubseteq \Lambda_1(l)$ then $l \sqsubseteq \lambda_1$.

Proof. Let $l \in D_1 \to [0,\infty)$.

(a) Assume that $\Lambda_1(l) \sqsubseteq l$. Let $P \in \mathcal{P}_{\mathrm{opt}}$. We show that for all $n \in \mathbb{N}$, $\Lambda_{1P}^n(\bot) \sqsubseteq l$ by induction on n. The base case, $n = 0$, follows from $\bot \sqsubseteq l$. In the inductive case, when $n > 0$, we have that

$$\begin{aligned}\Lambda_{1P}^n(\bot) &= \Lambda_{1P}(\Lambda_{1P}^{n-1}(\bot)) \\ &\sqsubseteq \Lambda_{1P}(l) \qquad \text{[induction hypothesis and } \Lambda_{1P} \text{ is monotone (Proposition 5.3)]} \\ &\sqsubseteq \Lambda_1(l) \qquad \text{[Proposition 6.12(a)]} \\ &\sqsubseteq l \qquad \text{[by assumption]}\end{aligned}$$

From Theorem 2.10(d) we can conclude that $\lambda_{1P} \sqsubseteq l$. Hence, $\lambda_1 \sqsubseteq l$.

(b) Assume that $l \sqsubseteq \Lambda_1(l)$. According to Proposition 6.12(b), $\Lambda_1(l) = \Lambda_{1P}(l)$ for some $P \in \mathcal{V}_{\mathrm{opt}}$. Next, we will show that $l \sqsubseteq \Lambda_{1P}^n(l)$ for all $n \in \mathbb{N}$ by induction on n. The base case, $n = 0$, is trivial. In the inductive case, when $n > 0$, we have that

$$\begin{aligned}l &\sqsubseteq \Lambda_1(l) \qquad \text{[by assumption]} \\ &= \Lambda_{1P}(l) \qquad \text{[Proposition 6.12(b)]} \\ &\sqsubseteq \Lambda_{1P}(\Lambda_{1P}^{n-1}(l)) \qquad \text{[induction hypothesis and } \Lambda_{1P} \text{ is monotone (Proposition 5.3)]} \\ &= \Lambda_{1P}^n(l).\end{aligned}$$

Since Λ_{1P} is nonexpansive (Proposition 5.7) and Λ_{P1}^m is contractive for some $m \in \mathbb{N}$ (Proposition 5.5), we can conclude from Theorem 2.10(d) that

$$l \sqsubseteq \lim_{n \in \mathbb{N}} \Lambda_{1P}^n(l) = \lambda_{1P} \sqsubseteq \lambda_1.$$

□

The key ingredient of the correctness proof of our algorithm is the following result.

Theorem 6.15. *λ_1 is the unique fixed point of Λ_1.*

Proof. First, we show that λ_1 is a fixed point of Λ_1. According to Proposition 6.13, we have that $\lambda_1 \sqsubseteq \Lambda_1(\lambda_1)$. Since Λ_1 is monotone (Proposition 6.10), we can conclude that $\Lambda_1(\lambda_1) \sqsubseteq \Lambda_1(\Lambda_1(\lambda_1))$. From Proposition 6.14(b) we can deduce that $\Lambda_1(\lambda_1) \sqsubseteq \lambda_1$. Combined with Proposition 6.13 we get that $\Lambda_1(\lambda_1) = \lambda_1$, that is, λ_1 is a fixed point of Λ_1.

Next, we show λ_1 is the unique fixed point of Λ_1. Assume that l is a fixed point of Λ_1, that is, $\Lambda_1(l) = l$. Then $\Lambda_1(l) \sqsubseteq l$ and $l \sqsubseteq \Lambda_1(l)$. By Proposition 6.14(a) and (b), we have that $\lambda_1 \sqsubseteq l$ and $l \sqsubseteq \lambda_1$. Hence, $l = \lambda_1$. Therefore, λ_1 is the unique fixed point of Λ_1. □

Our algorithm starts from an optimal vertex policy P, which can be obtained by the policy iteration algorithm of [53]. It computes λ_{1P} for that policy P (line 1), which can be done by means of standard algorithms (see, for example, [2, Section 10.1.1]). As long as there is a state pair in $D_1 \setminus S_1^2$ that is not locally 1-maximal with respect to the current policy (line 2), the policy at (s,t) is improved to a locally 1-maximal choice (lines 3 and 4). After this change to the policy P, the algorithm recomputes λ_{1P} (line 5). The loop maintains the invariant that P is an optimal vertex policy as π in line 3 is a vertex coupling and satisfies $\delta_1(s,t) = \pi \cdot \delta_1$ (see Proposition 4.21). At termination, we have that λ_{1P} is a fixed point of Λ_1 and, by Theorem 6.15, equals λ_1 and, therefore, is 1-maximal.

Algorithm 1 1-maximal optimal vertex policy

Input: optimal vertex policy P

1: compute λ_{1P}
2: **while** $\exists (s,t) \in D_1 \setminus S_1^2 : \Lambda_1(\lambda_{1P})(s,t) > \lambda_{1P}(s,t)$ **do**
3: $\quad \pi \leftarrow \underset{\omega \in V(\Omega(\tau(s),\tau(t))) \wedge \delta_1(s,t) = \omega \cdot \delta_1}{\arg\max} \omega \cdot (\delta_1 + \lambda_{1P})$
4: $\quad P(s,t) \leftarrow \pi$
5: $\quad$ compute λ_{1P}
6: **end while**

The following proposition proves that the above algorithm terminates.

Proposition 6.16. *The algorithm terminates.*

Proof. Towards a contradiction, assume that the algorithm does not terminate. Let us denote the value of the variable P in the i-th iteration by P_i. Then P_{i+1} is defined by

$$P_{i+1}(x,y) = \begin{cases} \pi & \text{if } (x,y) = (s,t) \\ P_i(x,y) & \text{otherwise,} \end{cases}$$

where

$$\pi \in \operatorname*{arg\,max}_{\omega \in V(\Omega_{\mathrm{opt}}(\tau(s),\tau(t)))} \omega \cdot (\delta_1 + \lambda_{1P_i}).$$

By Proposition 4.21, $P_{i+1} \in \mathcal{V}_{\mathrm{opt}}$. Next, we prove that $\lambda_{1P_i} \sqsubset \Lambda_{1P_{i+1}}(\lambda_{1P_i})$. Let $(x,y) \in D_1$. We distinguish three cases.

- If $(x,y) \in S_1^2$ then
$$\lambda_{1P_i}(x,y) = \Lambda_{1P_i}(\lambda_{1P_i})(x,y) = 0 = \Lambda_{1P_{i+1}}(\lambda_{1P_i})(x,y).$$
- If $(x,y) = (s,t)$ then
$$\begin{aligned} \lambda_{1P_i}(s,t) &< \Lambda_1(\lambda_{1P_i})(s,t) && \text{[condition of the loop]} \\ &= \sup_{\omega \in \Omega_{\mathrm{opt}}(\tau(s),\tau(t))} \omega \cdot (\delta_1 + \lambda_{1P_i}) \\ &= \max_{\omega \in V(\Omega_{\mathrm{opt}}(\tau(s),\tau(t)))} \omega \cdot (\delta_1 + \lambda_{1P_i}) && \text{[Proposition 6.4]} \\ &= \pi \cdot (\delta_1 + \lambda_{1P_i}) \\ &= P_{i+1}(s,t) \cdot (\delta_1 + \lambda_{1P_i}) \\ &= \Lambda_{1P_{i+1}}(\lambda_{1P_i})(s,t). \end{aligned}$$
- Otherwise,
$$\begin{aligned} \lambda_{1P_i}(x,y) &= \Lambda_{1P_i}(\lambda_{1P_i})(x,y) \\ &= P_i(x,y) \cdot (\delta_1 + \lambda_{1P_i}) \\ &= P_{i+1}(x,y) \cdot (\delta_1 + \lambda_{1P_i}) \\ &= \Lambda_{1P_{i+1}}(\lambda_{1P_i})(x,y). \end{aligned}$$

From Proposition 6.5 we can conclude that $\lambda_{1P_i} \sqsubset \lambda_{1P_{i+1}}$.

Since the set $\mathcal{V}_{\mathrm{opt}}$ is finite and the algorithm does not terminate, $P_i = P_j$ for some $i < j$. This contradicts that $\lambda_{1P_i} \sqsubset \lambda_{1P_j}$. □

From the above corollary, we know that our algorithm terminates. The following proposition shows that, upon termination, it computes the 1-maximal vertex optimal policy.

Proposition 6.17. *At termination,* $\lambda_{1P} = \lambda_1$.

Proof. At termination, we have that $\Lambda_1(\lambda_{1P})(s,t) \leq \lambda_{1P}(s,t)$ for all $(s,t) \in D_1$. Since for all $(s,t) \in S_1^2$,

$$\Lambda_1(\lambda_{1P})(s,t) = 0 = \Lambda_{1P}(\lambda_{1P})(s,t) = \lambda_{1P}(s,t),$$

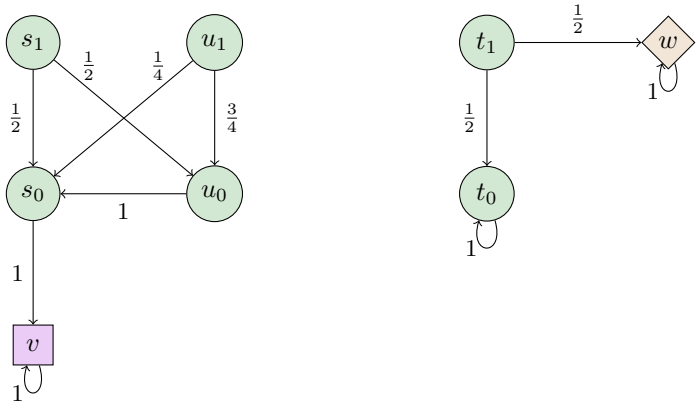
we have that $\Lambda_1(\lambda_{1P}) \sqsubseteq \lambda_{1P}$. By Proposition 6.12(a), $\lambda_{1P} = \Lambda_{1P}(\lambda_{1P}) \sqsubseteq \Lambda_1(\lambda_{1P})$. Hence, $\Lambda_1(\lambda_{1P}) = \lambda_{1P}$, that is, λ_{1P} is a fixed point of Λ_1. Since λ_1 is the unique fixed point of Λ_1 (Theorem 6.15), we can deduce that $\lambda_{1P} = \lambda_1$. □

In the next section, we prove an exponential lower bound for the above algorithm.

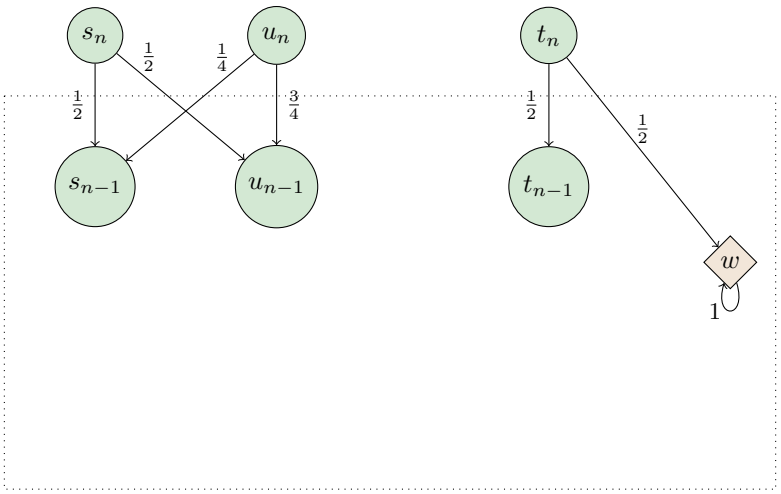
6.1 Exponential Lower Bound

In this section, we will prove an exponential lower bound for Algorithm 1 described in the previous section, which computes a 1-maximal (optimal vertex) policy. In particular, for each $n \in \mathbb{N}$, we construct an LMC of size $\mathcal{O}(n)$. We will consider policies for which the matching of the target states is fixed for most state pairs in the constructed LMC, except for the pairs (s_j, t_j) where $1 \leq j \leq n$. These state pairs have two possible matchings for the target states, from which the algorithm can choose. The algorithm follows a sequence of bit strings to determine which matchings of the j-th index state pair to switch. This sequence is similar to the Gray code sequence found in the literature, as we will see later. One of its key properties is that consecutive strings differ by a single bit. This bit, say k, indicates that the algorithm switches the policy at (s_k, t_k). Since there are two possible matching options for these types of state pairs, the algorithm selects the other matching. Each choice of matching for the target states of the pairs (s_j, t_j) results in a distinct policy. Therefore, for an LMC of size n, there are 2^n possible bit strings and 2^n distinct policies. As we will show below, the algorithm must explore all these policies to find a 1-maximal policy, implying that the algorithm requires $\Omega(2^n)$ iterations to identify a 1-maximal policy.

Definition 6.18. *The LMC C_0 is defined as*



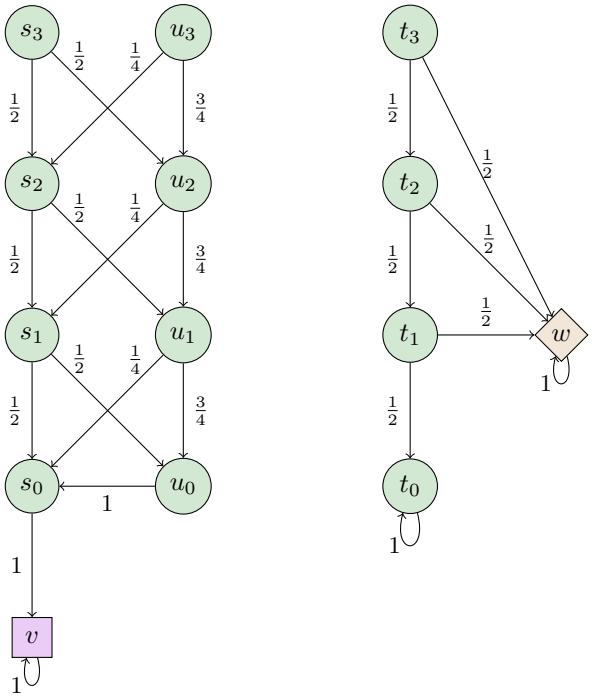
If $n > 0$ then the LMC C_n is defined as



The dashed square represents the LMC C_{n-1}.

Note that C_n has $3n + 8$ states and $6n + 11$ transitions and, hence, is of size $\mathcal{O}(n)$.

Example 6.19. *The LMC C_3 is depicted as follows.*



The next proposition calculates the distances of the state pairs in C_n.

Proposition 6.20.

(a) $\delta_1(t_0, v) = 1$

(b) For all $n \in \mathbb{N}$, $\delta_1(s_n, w) = 1$

(c) For all $n \in \mathbb{N}$, $\delta_1(u_n, w) = 1$

(d) For all $n \in \mathbb{N}$, $\delta_1(s_n, t_n) = 1$ *and* $\delta_1(u_n, t_n) = 1$

Proof.

(a) $\delta_1(t_0, v) = \Delta(\delta_1)(t_0, v) = 1$ as $(t_0, v) \in S_1^2$.

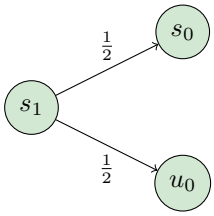
(b) For each $n \in \mathbb{N}$, $\delta_1(s_n, w) = \Delta(\delta_1)(s_n, w) = 1$ as $(s_n, w) \in S_1^2$.

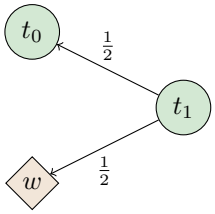
(c) For each $n \in \mathbb{N}$, $\delta_1(u_n, w) = \Delta(\delta_1)(u_n, w) = 1$ as $(u_n, w) \in S_1^2$.

(d) We prove this case by induction on n. In the base case, when $n = 0$, we can conclude from (a) and Proposition 3.16 that $\delta_1(s_0, t_0) = 1$.

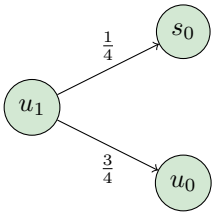
From the above and Proposition 3.16 we can conclude that $\delta_1(u_0, t_0) = 1$.

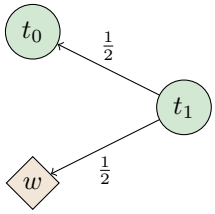
Next, we consider the case $n = 1$. Again using Proposition 3.16 we can conclude from the above that $\delta_1(s_1, t_1) = 1$.



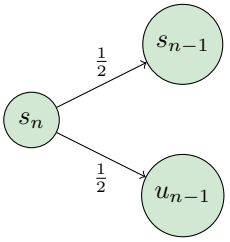


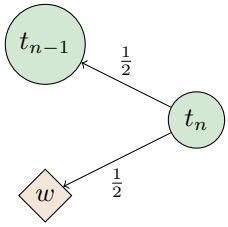
and $\delta_1(u_1, t_1) = 1$.

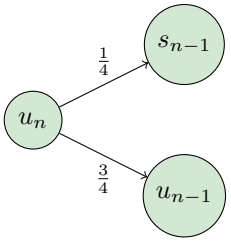


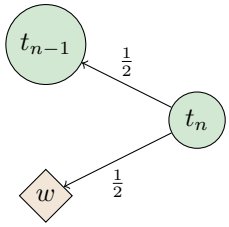


In the inductive case, $n > 1$.









Since $\delta_1(s_{n-1}, w) = 1$ and $\delta_1(u_{n-1}, w) = 1$, by (b) and (c), and $\delta_1(s_{n-1}, t_{n-1}) = 1$ and $\delta_1(u_{n-1}, t_{n-1}) = 1$ by the induction hypothesis, we can conclude that $\delta_1(s_n, t_n) = 1$ and $\delta_1(u_n, t_n) = 1$ by Proposition 3.16.

□

The following definition defines the coupling between the state pairs in C_n.

Definition 6.21. *The function $\rho_0 : S \times S \to [0, 1]$ is defined by*

$$\rho_0(x, y) = \begin{cases} 1 & \text{if } (x, y) = (t_0, v) \\ 0 & \text{otherwise.} \end{cases}$$

The function $\pi_0 : S \times S \to [0, 1]$ is defined by

$$\pi_0(x, y) = \begin{cases} 1 & \text{if } (x, y) = (s_0, t_0) \\ 0 & \text{otherwise.} \end{cases}$$

For all $n \geq 1$, the function $\pi_n : S \times S \to [0, 1]$ is defined by

$$\pi_n(x, y) = \begin{cases} \frac{1}{4} & \text{if } (x, y) = (s_{n-1}, t_{n-1}) \\ \frac{1}{4} & \text{if } (x, y) = (u_{n-1}, t_{n-1}) \\ \frac{1}{2} & \text{if } (x, y) = (u_{n-1}, w) \\ 0 & \text{otherwise.} \end{cases}$$

For all $n \geq 1$, the function $\omega_{0n} : S \times S \to [0,1]$ is defined by

$$\omega_{0n}(x,y) = \begin{cases} \frac{1}{2} & \text{if } (x,y) = (s_{n-1}, t_{n-1}) \\ \frac{1}{2} & \text{if } (x,y) = (u_{n-1}, w) \\ 0 & \text{otherwise.} \end{cases}$$

For all $n \geq 1$, the function $\omega_{1n} : S \times S \to [0,1]$ is defined by

$$\omega_{1n}(x,y) = \begin{cases} \frac{1}{2} & \text{if } (x,y) = (s_{n-1}, w) \\ \frac{1}{2} & \text{if } (x,y) = (u_{n-1}, t_{n-1}) \\ 0 & \text{otherwise.} \end{cases}$$

The next proposition shows that for every state pair $(s,t) \in S_?^2$ there exist a vertex optimal coupling.

To conclude the existence of vertex coupling, we apply Proposition 3.15. However, before using this proposition, we must first establish that the functions introduced above are indeed vertices of transportation polytopes.

Proposition 6.22.

(a) $\rho_0 \in V(\Omega(\tau(s_0), \tau(t_0)))$

(b) For all $n \geq 0$, $\pi_n \in V(\Omega(\tau(u_n), \tau(t_n)))$

(c) For all $n \geq 1$, $\omega_{0n} \in V(\Omega(\tau(s_n), \tau(t_n)))$

(d) For all $n \geq 1$, $\omega_{1n} \in V(\Omega(\tau(s_n), \tau(t_n)))$

Proof.

(a) To conclude that $\rho_0 \in \Omega(\tau(s_0), \tau(t_0))$ it suffices to show that for all $x, y \in S$,

$$\rho_0(x, S) = \tau(s_0)(x) \text{ and } \rho_0(S, y) = \tau(t_0)(y).$$

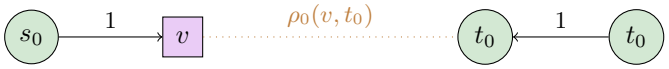
We distinguish the following cases.

- If $x = v$ then $\rho_0(v, S) = 1 = \tau(s_0)(v)$.
- Otherwise, $\rho_0(x, S) = 0 = \tau(s_0)(x)$.

Hence, $\rho_0(x, S) = \tau(s_0)(x)$.

- If $y = t_0$ then $\rho_0(S, t_0) = 1 = \tau(t_0)(t_0)$.
- Otherwise, $\rho_0(S, y) = 0 = \tau(t_0)(y)$.

Hence, $\rho_0(S, y) = \tau(t_0)(y)$. Therefore, $\rho_0 \in \Omega(\tau(s_0), \tau(t_0))$. The coupling ρ_0 can be presented by the following graph between states s_0 and t_0.



Since the support graph of ρ_0 is acyclic, by Proposition 3.5, we have that $\rho_0 \in V(\Omega(\tau(s_0), \tau(t_0)))$.

(b) To conclude that $\pi_0 \in \Omega(\tau(u_0), \tau(t_0))$ it suffices to show that for all $x, y \in S$,

$$\pi_0(x, S) = \tau(u_0)(x) \text{ and } \pi_0(S, y) = \tau(t_0)(y).$$

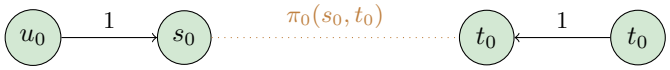
We distinguish the following cases.

- If $x = s_0$ then $\pi_0(s_0, S) = 1 = \tau(u_0)(s_0)$.
- Otherwise, $\pi_0(x, S) = 0 = \tau(u_0)(x)$.

Hence, $\pi_0(x, S) = \tau(u_0)(x)$.

- If $y = t_0$ then $\pi_0(S, t_0) = 1 = \tau(t_0)(t_0)$.
- Otherwise, $\pi_0(S, y) = 0 = \tau(t_0)(y)$.

Hence, $\pi_0(S, y) = \tau(t_0)(y)$. Therefore, $\pi_0 \in \Omega(\tau(u_0), \tau(t_0))$. The coupling π_0 can be presented by the following graph between states u_0 and t_0.



Since the support graph of π_0 is acyclic, by Proposition 3.5, we have that $\pi_0 \in V(\Omega(\tau(u_0), \tau(t_0)))$.

Let $n \geq 1$. To conclude that $\pi_n \in \Omega(\tau(u_n), \tau(t_n))$ it suffices to show that for all $x, y \in S$,

$$\pi_n(x, S) = \tau(u_n)(x) \text{ and } \pi_n(S, y) = \tau(t_n)(y).$$

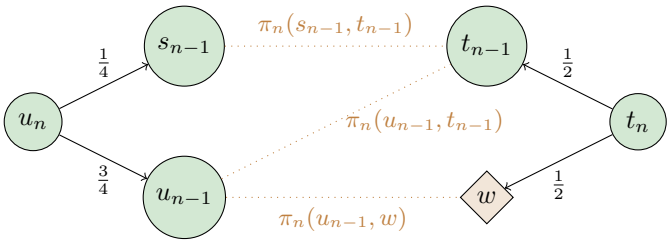
We distinguish the following cases.

- If $x = s_{n-1}$ then $\pi_n(s_{n-1}, S) = \frac{1}{4} = \tau(u_n)(s_{n-1})$.
- If $x = u_{n-1}$ then $\pi_n(u_{n-1}, S) = \frac{3}{4} = \tau(u_n)(u_{n-1})$.
- Otherwise, $\pi_n(x, S) = 0 = \tau(u_n)(x)$.

Hence, $\pi_n(x, S) = \tau(u_n)(x)$.

- If $y = t_{n-1}$ then $\pi_n(S, t_{n-1}) = \frac{1}{2} = \tau(t_n)(t_{n-1})$.
- If $y = w$ then $\pi_n(S, w) = \frac{1}{2} = \tau(t_n)(w)$.
- Otherwise, $\pi_n(S, y) = 0 = \tau(t_n)(y)$.

Hence, $\pi_n(S, y) = \tau(t_n)(y)$. Therefore, $\pi_n \in \Omega(\tau(u_n), \tau(t_n))$. The coupling π_n can be presented by the following graph between states u_n and t_n.



Since the support graph of π_n is acyclic, by Proposition 3.5, we have that $\pi_n \in V(\Omega(\tau(u_n), \tau(t_n)))$.

(c) Let $n \geq 1$, To conclude that $\omega_{0n} \in \Omega(\tau(s_n), \tau(t_n))$ it suffices to show that for all $x, y \in S$,

$$\omega_{0n}(x, S) = \tau(s_n)(x) \text{ and } \omega_{0n}(S, y) = \tau(t_n)(y).$$

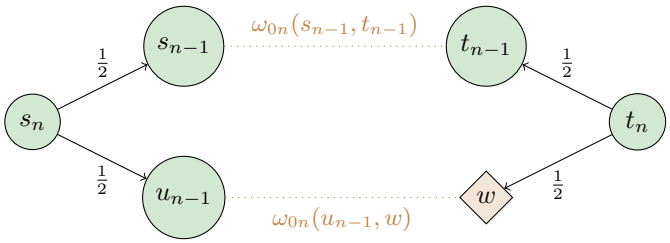
We distinguish the following cases.

- If $x = s_{n-1}$ then $\omega_{0n}(s_{n-1}, S) = \frac{1}{2} = \tau(s_n)(s_{n-1})$.
- If $x = u_{n-1}$ then $\omega_{0n}(u_{n-1}, S) = \frac{1}{2} = \tau(s_n)(u_{n-1})$.
- Otherwise, $\omega_{0n}(x, S) = 0 = \tau(s_n)(x)$.

Hence, $\omega_{0n}(x, S) = \tau(s_n)(x)$.

- If $y = t_{n-1}$ then $\omega_{0n}(S, t_{n-1}) = \frac{1}{2} = \tau(t_n)(t_{n-1})$.
- If $y = w$ then $\omega_{0n}(S, w) = \frac{1}{2} = \tau(t_n)(w)$.
- Otherwise, $\omega_{0n}(S, y) = 0 = \tau(t_n)(y)$.

Hence, $\omega_{0n}(S, y) = \tau(t_n)(y)$. Therefore, $\omega_{0n} \in \Omega(\tau(s_n), \tau(t_n))$. The coupling ω_{0n} can be presented by the following graph between states s_n and t_n.



Since the support graph of ω_{0n} is acyclic, by Proposition 3.5, we have that $\omega_{0n} \in V(\Omega(\tau(s_n), \tau(t_n)))$.

(d) Let $n \geq 1$. To conclude that $\omega_{1n} \in \Omega(\tau(s_n), \tau(t_n))$ it suffices to show that for all $x, y \in S$,

$$\omega_{1n}(x, S) = \tau(s_n)(x) \text{ and } \omega_{1n}(S, y) = \tau(t_n)(y).$$

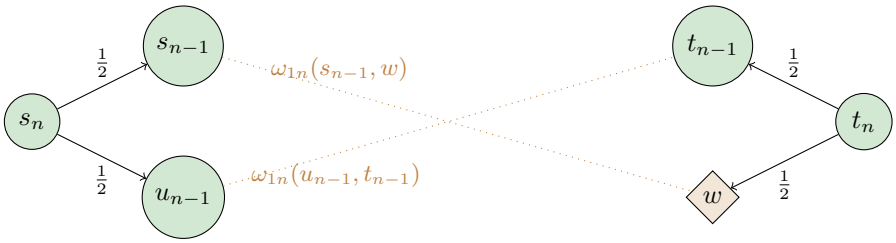
We distinguish the following cases.

- If $x = s_{n-1}$ then $\omega_{1n}(s_{n-1}, S) = \frac{1}{2} = \tau(s_n)(s_{n-1})$.
- If $x = u_{n-1}$ then $\omega_{1n}(u_{n-1}, S) = \frac{1}{2} = \tau(s_n)(u_{n-1})$.
- Otherwise, $\omega_{1n}(x, S) = 0 = \tau(s_n)(x)$.

Hence, $\omega_{1n}(x, S) = \tau(s_n)(x)$.

- If $y = t_{n-1}$ then $\omega_{1n}(S, t_{n-1}) = \frac{1}{2} = \tau(t_n)(t_{n-1})$.
- If $y = w$ then $\omega_{1n}(S, w) = \frac{1}{2} = \tau(t_n)(w)$.
- Otherwise, $\omega_{1n}(S, y) = 0 = \tau(t_n)(y)$.

Hence, $\omega_{1n}(S, y) = \tau(t_n)(y)$. Therefore, $\omega_{1n} \in \Omega(\tau(s_n), \tau(t_n))$. The coupling ω_{1n} can be presented by the following graph between states s_n and t_n.



Since the support graph of ω_{1n} is acyclic, by Proposition 3.5, we have that $\omega_{1n} \in V(\Omega(\tau(s_n), \tau(t_n)))$.

□

Since the above proposition shows that the functions defined in Definition 6.1 are vertex couplings, it follows that for each $(s, t) \in S_?^2$, there exists $\omega \in V(\Omega(\tau(s), \tau(t)))$. The next proposition shows that these functions satisfy the following $\delta_1(s, t) = \omega \cdot \delta_1$.

Proposition 6.23.

(a) $\rho_0 \cdot \delta_1 = \delta_1(s_0, t_0)$

(b) For all $n \geq 0$, $\pi_n \cdot \delta_1 = \delta_1(u_n, t_n)$

(c) For all $n \geq 1$, $\omega_{0n} \cdot \delta_1 = \delta_1(s_n, t_n)$

(d) For all $n \geq 1$, $\omega_{1n} \cdot \delta_1 = \delta_1(s_n, t_n)$

Proof.

(a) The coupling ρ_0 can be presented by the following graph between s_0 and t_0.

We have that

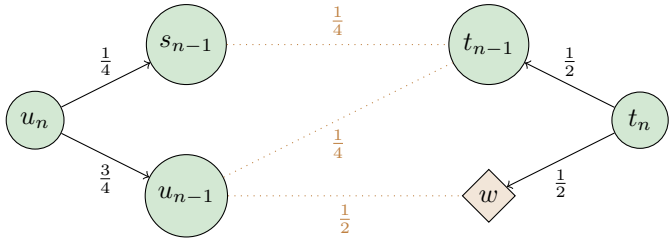
$$\begin{aligned}\rho_0 \cdot \delta_1 &= \delta_1(t_0, v) \\ &= 1 \qquad \text{[Proposition 6.20(a)]} \\ &= \delta_1(s_0, t_0) \qquad \text{[Proposition 6.20(d)]}\end{aligned}$$

(b) The coupling π_0 can be presented by the following graph between states u_0 and t_0.

We have that

$$\begin{aligned}\pi_0 \cdot \delta_1 &= \delta_1(s_0, t_0) \\ &= 1 \qquad \text{[Proposition 6.20(d)]} \\ &= \delta_1(u_0, t_0) \qquad \text{[Proposition 6.20(d)]}\end{aligned}$$

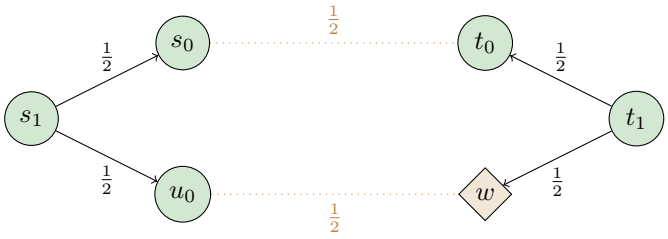
Let $n \geq 1$. The coupling π_n can be presented by the following graph between states u_n and t_n.



We have that

$$\begin{aligned}\pi_n \cdot \delta_1 &= \frac{1}{4}\delta_1(s_{n-1}, t_{n-1}) + \frac{1}{4}\delta_1(u_{n-1}, t_{n-1}) + \frac{1}{2}\delta_1(u_{n-1}, w) \\ &= \frac{1}{4} + \frac{1}{4} + \frac{1}{2} \qquad \text{[Proposition 6.20(c) and (j)]} \\ &= 1 \\ &= \delta_1(u_n, t_n) \qquad \text{[Proposition 6.20((d)]}\end{aligned}$$

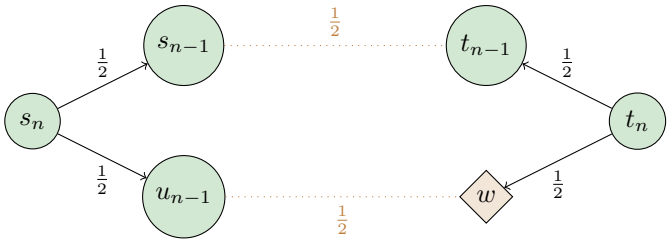
(c) The coupling ω_{01} can be presented by the following graph between states s_1 and t_1.



We have that

$$\begin{aligned}\omega_{01} \cdot \delta_1 &= \frac{1}{2}\delta_1(s_0, t_0) + \frac{1}{2}\delta_1(u_0, w) \\ &= \frac{1}{2} + \frac{1}{2} \qquad \text{[Proposition 6.20(c) and (d)]} \\ &= 1 \\ &= \delta_1(s_1, t_1) \qquad \text{[Proposition 6.20(d)]}\end{aligned}$$

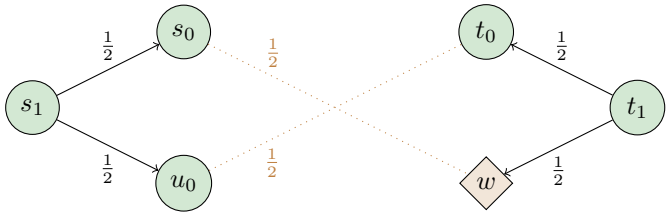
Let $n > 1$. The coupling ω_{0n} can be presented by the following graph between states s_n and t_n.



We have that

$$\begin{aligned}\omega_{0n} \cdot \delta_1 &= \frac{1}{2}\delta_1(s_{n-1}, t_{n-1}) + \frac{1}{2}\delta_1(u_{n-1}, w) \\ &= \frac{1}{2} + \frac{1}{2} \qquad \text{[Proposition 6.20(c) and (j)]} \\ &= 1 \\ &= \delta_1(s_n, t_n) \qquad \text{[Proposition 6.20(d)]}\end{aligned}$$

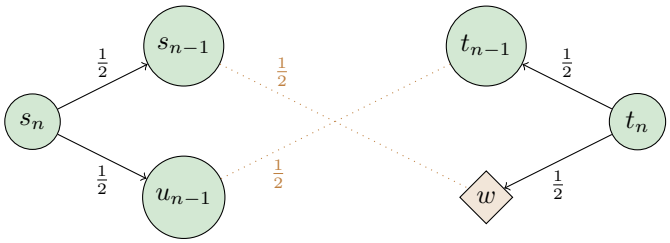
(d) The coupling ω_{11} can be presented by the following graph between states s_1 and t_1.



We have that

$$\begin{aligned}\omega_{11} \cdot \delta_1 &= \frac{1}{2}\delta_1(s_0, w) + \frac{1}{2}\delta_1(u_0, t_0) \\ &= \frac{1}{2} + \frac{1}{2} \qquad \text{[Proposition 6.20(b) and (e)]} \\ &= 1 \\ &= \delta_1(s_1, t_1) \qquad \text{[Proposition 6.20(d)]}\end{aligned}$$

Let $n > 1$. The coupling ω_{1n} can be presented by the following graph between states s_n and t_n.



We have that

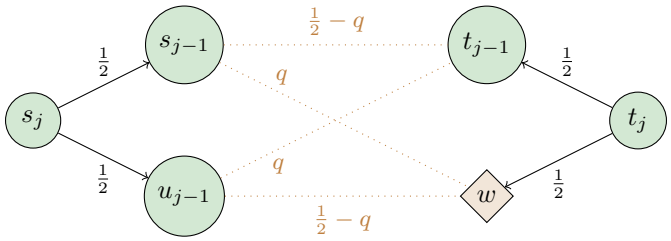
$$\begin{aligned}\omega_{1n} \cdot \delta_1 &= \frac{1}{2}\delta_1(s_{n-1}, w) + \frac{1}{2}\delta_1(u_{n-1}, t_{n-1}) \\ &= \frac{1}{2} + \frac{1}{2} \qquad \text{[Proposition 6.20(b) and (j)]} \\ &= 1 \\ &= \delta_1(s_n, t_n) \qquad \text{[Proposition 6.20(d)]}\end{aligned}$$

□

The next proposition shows that ω_{0j} and ω_{1j} are the only vertex couplings of $\tau(s_j)$ and $\tau(t_j)$.

Proposition 6.24. *For all* $1 \leq j \leq n$, $V(\Omega(\tau(s_j), \tau(t_j)) = \{\omega_{0j}, \omega_{1j}\}$.

Proof. Let $1 \leq j \leq n$. Each coupling $\omega \in V(\Omega(\tau(s_j), \tau(t_j)))$ is of the following form for some $q \in [0, \frac{1}{2}]$.



According to Proposition 3.5, $\omega \in V(\Omega(\tau(s_j), \tau(t_j)))$ if and only if the support graph of ω is acyclic, that is, q is either 0 or $\frac{1}{2}$. If $q = 0$ then $\omega = \omega_{0j}$ and if $q = \frac{1}{2}$ then $\omega = \omega_{1j}$. □

For each $n \geq 1$, we introduce a sequence $(\theta_i^n)_i$ of length 2^n where each element θ_i^n is a bit string of length n. The 1-maximal algorithm does not start from an arbitrary policy; instead, it begins with a specific initial policy determined by the first bit string, which indicates the choice of target states for the state pairs (s_j, t_j) for $1 \leq j \leq n$. In subsequent iterations, rather than selecting an arbitrary state pair that is not locally optimal, the algorithm switches the matchings of the target states for a specific pair. As we will see later, only one bit differs between any two consecutive bit strings. Suppose the current and next strings differ at index j; in this case, the algorithm selects the pair (s_j, t_j) to switch its matching. Recall that there are two possible matchings for each state pair (s_j, t_j); the algorithm then finds the other matching for (s_j, t_j) and updates it accordingly. These bit strings tell the algorithm which target state matchings of the (s_j, t_j) state pairs to switch.

Definition 6.25. *The sequence* $(\theta_i^1)_i$ *is defined by*

$$\theta_i^1 = \begin{cases} 0 & \text{if } i = 0 \\ 1 & \text{if } i = 1 \end{cases}$$

Let $n > 1$. *The sequence* $(\theta_i^n)_i$ *is defined by*

$$\theta_i^n = \begin{cases} 0\theta^{n-1}_{\frac{i}{2}} & \text{if } i \bmod 4 = 0 \\ 1\theta^{n-1}_{\frac{i-1}{2}} & \text{if } i \bmod 4 = 1 \\ 1\theta^{n-1}_{\frac{i}{2}} & \text{if } i \bmod 4 = 2 \\ 0\theta^{n-1}_{\frac{i-1}{2}} & \text{if } i \bmod 4 = 3 \end{cases}$$

We refer to the above sequence as the reverse ordered Gray code, as it rearranges the standard Gray code sequence in reverse order (see, for example, [27] for a definition of Gray code). One of the key properties of Gray code is its single-bit change property, which guarantees that only one bit differs between any two consecutive strings. Another key property is that the code cycles through all 2^n bit strings of length n. The next example shows the above sequence when $n = 3$.

Example 6.26. *The sequence* $(\theta_i^3)_i$ *is defined as follows.*

$$(\theta_i^3)_i = \begin{cases} 000 \\ 100 \\ 110 \\ 010 \\ 011 \\ 111 \\ 101 \\ 001 \end{cases}$$

These sequences have specific properties. The following propositions prove some of these properties of $(\theta_i^n)_i$, which are used later in this section. First, we show that every bit string in this sequence is unique.

Proposition 6.27. *For all $n \geq 1$ and $0 \leq i, j < 2^n$, if $i \neq j$ then $\theta_i^n \neq \theta_j^n$.*

Proof. We prove this proposition by induction on n.

- In the base case, when $n = 1$, we consider two cases
  - If $i = 0$ and $j = 1$ then
    $$\theta_i^1 = 0 \neq 1 = \theta_j^1$$
  - If $i = 1$ and $j = 0$ then
    $$\theta_i^1 = 1 \neq 0 = \theta_j^1$$

  Hence, $\theta_i^1 \neq \theta_j^1$.
- In the inductive case, $n > 1$. By Definition 6.25, we have that
  $$\theta_i^n = \begin{cases} 0\theta_{\frac{i}{2}}^{n-1} & \text{if } i \bmod 4 = 0 \\ 1\theta_{\frac{i-1}{2}}^{n-1} & \text{if } i \bmod 4 = 1 \\ 1\theta_{\frac{i}{2}}^{n-1} & \text{if } i \bmod 4 = 2 \\ 0\theta_{\frac{i-1}{2}}^{n-1} & \text{if } i \bmod 4 = 3 \end{cases}$$

  and
  $$\theta_j^n = \begin{cases} 0\theta_{\frac{j}{2}}^{n-1} & \text{if } j \bmod 4 = 0 \\ 1\theta_{\frac{j-1}{2}}^{n-1} & \text{if } j \bmod 4 = 1 \\ 1\theta_{\frac{j}{2}}^{n-1} & \text{if } j \bmod 4 = 2 \\ 0\theta_{\frac{j-1}{2}}^{n-1} & \text{if } j \bmod 4 = 3 \end{cases}$$

  We consider three cases.
  - If $\theta_i^n[n] = \theta_j^n[n] = 0$ then we distinguish four cases.
    - Assume that $\theta_i^n = 0\theta_{\frac{i}{2}}^{n-1}$ and $i \bmod 4 = 0$ and $\theta_j^n = 0\theta_{\frac{j}{2}}^{n-1}$ and $j \bmod 4 = 0$. Since $i \neq j$, we have $\frac{i}{2} \neq \frac{j}{2}$. By the induction hypothesis, $\theta_{\frac{i}{2}}^{n-1} \neq \theta_{\frac{j}{2}}^{n-1}$ and hence we have $\theta_i^n \neq \theta_j^n$.
    - Assume that $\theta_i^n = 0\theta_{\frac{i}{2}}^{n-1}$ and $i \bmod 4 = 0$ and $\theta_j^n = 0\theta_{\frac{j-1}{2}}^{n-1}$ and $j \bmod 4 = 3$. Since $i \neq j$, we show that $\frac{i}{2} \neq \frac{j-1}{2}$. Towards a contradiction, assume that $\frac{i}{2} = \frac{j-1}{2}$. Then $j = i + 1$. Since $i \bmod 4 = 0$, we can conclude that $j \bmod 4 = 1$ which contradicts $j \bmod 4 = 3$. Therefore, $\frac{i}{2} \neq \frac{j-1}{2}$. By the induction hypothesis, $\theta_{\frac{i}{2}}^{n-1} \neq \theta_{\frac{j-1}{2}}^{n-1}$ and therefore we have $\theta_i^n \neq \theta_j^n$.
    - Assume that $\theta_i^n = 0\theta_{\frac{i-1}{2}}^{n-1}$ and $i \bmod 4 = 3$ and $\theta_j^n = 0\theta_{\frac{j}{2}}^{n-1}$ and $j \bmod 4 = 0$. This case is similar to the second case..
    - Assume that $\theta_i^n = 0\theta_{\frac{i-1}{2}}^{n-1}$ and $i \bmod 4 = 3$ and $\theta_j^n = 0\theta_{\frac{j-1}{2}}^{n-1}$ and $j \bmod 4 = 3$. This case is similar to the first case.
  - The case where $\theta_i^n[n] = \theta_j^n[n] = 1$ is similar to the first case.
  - If $\theta_i^n[n] \neq \theta_j^n[n]$ then $\theta_i^n \neq \theta_j^n$.

  Hence, $\theta_i^n \neq \theta_j^n$.

□

Next, we show that two consecutive bit strings differ by a single bit.

Proposition 6.28. *For all $n \geq 1$ and $0 \leq i < 2^n - 1$, θ_i^n and θ_{i+1}^n differ by a single bit.*

Proof. We prove this proposition by induction on n.

- In the base case, when $n = 1$, we have that

$$\theta_0^1 = 0$$
$$\theta_1^1 = 1$$

  Hence, θ_0^1 and θ_1^1 differ by a single bit.

- In the inductive case, $n > 1$, we consider four cases.
  - If $i \bmod 4 = 0$ then

$$\theta_i^n = 0\theta_{\frac{i}{2}}^{n-1}$$
$$\theta_{i+1}^n = 1\theta_{\frac{(i+1)-1}{2}}^{n-1} = 1\theta_{\frac{i}{2}}^{n-1}$$

    Hence, θ_i^n and θ_{i+1}^n differ by a single bit.
  - If $i \bmod 4 = 1$ then

$$\theta_i^n = 1\theta_{\frac{i-1}{2}}^{n-1}$$
$$\theta_{i+1}^n = 1\theta_{\frac{i+1}{2}}^{n-1}$$

    By the induction hypothesis, $\theta_{\frac{i-1}{2}}^{n-1}$ and $\theta_{\frac{i+1}{2}}^{n-1}$ differ by a single bit. Hence, θ_i^n and θ_{i+1}^n differ by a single bit.
  - If $i \bmod 4 = 2$ then

$$\theta_i^n = 1\theta_{\frac{i}{2}}^{n-1}$$
$$\theta_{i+1}^n = 0\theta_{\frac{(i+1)-1}{2}}^{n-1} = 0\theta_{\frac{i}{2}}^{n-1}$$

    Hence, θ_i^n and θ_{i+1}^n differ by a single bit.
  - If $i \bmod 4 = 3$ then

$$\theta_i^n = 0\theta_{\frac{i-1}{2}}^{n-1}$$
$$\theta_{i+1}^n = 0\theta_{\frac{i}{2}}^{n-1}$$

    By the induction hypothesis, $\theta_{\frac{i-1}{2}}^{n-1}$ and $\theta_{\frac{i}{2}}^{n-1}$ differ by a single bit. Hence, θ_i^n and θ_{i+1}^n differ by a single bit.

□

The next proposition proves when the parity of a bit string is even.

Proposition 6.29. *For all $n \geq 1$ and $0 \leq i < 2^n$, the parity of θ_i^n is even iff $i \bmod 2 = 0$.*

Proof. We prove this proposition by induction on n.

- In the base case, when $n = 1$, we have that

$$\theta_0^1 = 0$$
$$\theta_1^1 = 1$$

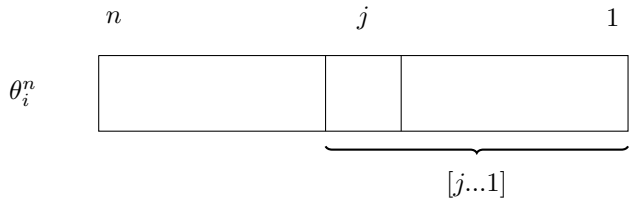
  Hence, the parity of θ_i^1 is even iff $i \bmod 2 = 0$.

- In the inductive case, $n > 1$, we consider four cases.
  - If $i \bmod 4 = 0$ then $\theta_i^n = 0\theta_{\frac{i}{2}}^{n-1}$. By the induction hypothesis, $\theta_{\frac{i}{2}}^{n-1}$ has even parity as $\frac{i}{2} \bmod 2 = 0$. Hence, the parity of θ_i^n is even in this case.

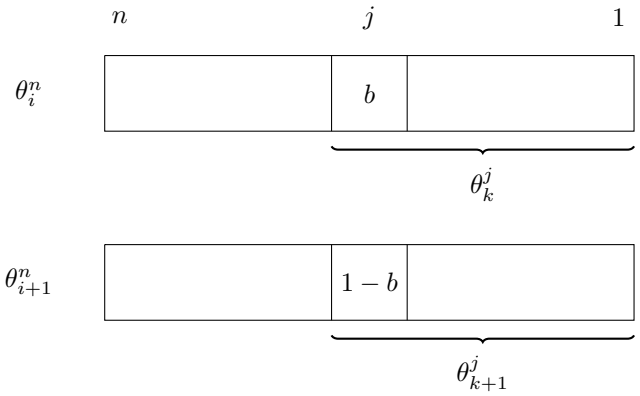
– If $i \bmod 4 = 1$ then $\theta_i^n = 1\theta_{\frac{i-1}{2}}^{n-1}$. By the induction hypothesis, $\theta_{\frac{i-1}{2}}^{n-1}$ has even parity as $\frac{i-1}{2} \bmod 2 = 0$. Hence, the parity of θ_i^n is odd in this case.
– If $i \bmod 4 = 2$ then $\theta_i^n = 1\theta_{\frac{i}{2}}^{n-1}$. By the induction hypothesis, $\theta_{\frac{i}{2}}^{n-1}$ has odd parity as $\frac{i}{2} \bmod 2 = 1$. Hence, the parity of θ_i^n is even in this case.
– If $i \bmod 4 = 3$ then $\theta_i^n = 0\theta_{\frac{i-1}{2}}^{n-1}$. By the induction hypothesis, $\theta_{\frac{i-1}{2}}^{n-1}$ has odd parity as $\frac{i-1}{2} \bmod 2 = 1$. Hence, the parity of θ_i^n is odd in this case.

□

For the remainder, $\theta_i^n[j]$ refers to the j-th bit from right in the string θ_i^n, and $\theta_i^n[j...1]$ refers to the bits from index j down to index 1 in the bit string θ_i^n.



The following proposition highlights a structural pattern in consecutive bit strings. Suppose two consecutive bit strings differ at index j. Then, consider the substring consisting of bits from position j down to position 1 in each of the two strings. In the previous bit string, this substring is identical to the k-th bit string in the sequence of all bit strings of length j, for some $0 \leq k < 2^j - 1$. In the next bit string, the same substring corresponds to the $(k+1)$-th bit string in that sequence. This relationship is illustrated below, where $b \in \{0, 1\}$.



Proposition 6.30. *For all $n \geq 1$ and $0 \leq i < 2^n - 1$ and $1 \leq j \leq n$, if $\theta_i^n[j] \neq \theta_{i+1}^n[j]$ then $\theta_i^n[j..1] = \theta_k^j$ and $\theta_{i+1}^n[j..1] = \theta_{k+1}^j$ for some $0 \leq k < 2^j - 1$.*

Proof. Let $n \geq 1$ and $0 \leq i < 2^n - 1$ and $1 \leq j \leq n$. We prove this proposition by induction on $n - j$.

- In the base case, when $j = n$, we have that

$$\theta_i^n[n..1] = \theta_i^n$$
$$\theta_{i+1}^n[n..1] = \theta_{i+1}^n$$

  In this case $k = i$.

- In the inductive case, $1 \leq j < n$. By Definition 6.25, we have that

$$\theta_i^n = \begin{cases} 0\theta_{\frac{i}{2}}^{n-1} & \text{if } i \bmod 4 = 0 \\ 1\theta_{\frac{i-1}{2}}^{n-1} & \text{if } i \bmod 4 = 1 \\ 1\theta_{\frac{i}{2}}^{n-1} & \text{if } i \bmod 4 = 2 \\ 0\theta_{\frac{i-1}{2}}^{n-1} & \text{if } i \bmod 4 = 3 \end{cases}$$

and

$$\theta_{i+1}^n = \begin{cases} 1\theta_{\frac{i}{2}}^{n-1} & \text{if } i \bmod 4 = 0 \\ 1\theta_{\frac{i+1}{2}}^{n-1} & \text{if } i \bmod 4 = 1 \\ 0\theta_{\frac{i}{2}}^{n-1} & \text{if } i \bmod 4 = 2 \\ 0\theta_{\frac{i+1}{2}}^{n-1} & \text{if } i \bmod 4 = 3 \end{cases}$$

Since $\theta_i^n[j] \neq \theta_{i+1}^n[j]$ and $1 \leq j < n$, we are left with the following cases.

$$\theta_i^n = \begin{cases} 1\theta_{\frac{i-1}{2}}^{n-1} & \text{if } i \bmod 4 = 1 \\ 0\theta_{\frac{i-1}{2}}^{n-1} & \text{if } i \bmod 4 = 3 \end{cases}$$

and

$$\theta_{i+1}^n = \begin{cases} 1\theta_{\frac{i+1}{2}}^{n-1} & \text{if } i \bmod 4 = 1 \\ 0\theta_{\frac{i+1}{2}}^{n-1} & \text{if } i \bmod 4 = 3 \end{cases}$$

We consider the case that $i \bmod 4 = 3$. Since $\theta_i^n[j] \neq \theta_{i+1}^n[j]$ we have that $\theta_{\frac{i-1}{2}}^{n-1}[j] \neq \theta_{\frac{i+1}{2}}^{n-1}[j]$. Therefore, by the induction hypothesis, $\theta_{\frac{i-1}{2}}^{n-1}[j...1] = \theta_l^j$ and $\theta_{\frac{i+1}{2}}^{n-1}[j...1] = \theta_{l+1}^j$ for some $0 \leq l < 2^j - 1$. Hence,

$$\begin{aligned} \theta_i^n[j...1] &= (0\theta_{\frac{i-1}{2}}^{n-1})[j...1] \\ &= \theta_{\frac{i-1}{2}}^{n-1}[j...1] \qquad [j < n] \\ &= \theta_l^j \end{aligned}$$

and

$$\begin{aligned} \theta_{i+1}^n[j...1] &= (0\theta_{\frac{i+1}{2}}^{n-1})[j...1] \\ &= \theta_{\frac{i+1}{2}}^{n-1}[j...1] \qquad [j < n] \\ &= \theta_{l+1}^j \end{aligned}$$

The other case, where $i \bmod 4 = 1$ and $\theta_i^n[j...1] = (1\theta_{\frac{i-1}{2}}^{n-1})[j...1]$ and $\theta_{i+1}^n[j...1] = (1\theta_{\frac{i+1}{2}}^{n-1})[j...1]$, can be proved similarly.

□

The following proposition proves that if two consecutive bit strings differ at index j, then the prior one has even parity from index 1 to j, while the next one has odd parity from index 1 to j.

Proposition 6.31. *For all $n \geq 1$ and $0 \leq i < 2^n - 1$ and $1 \leq j \leq n$, if $\theta_i^n[j] \neq \theta_{i+1}^n[j]$ then $\theta_i^n[j..1]$ has even parity and $\theta_{i+1}^n[j..1]$ has odd parity.*

Proof. Let $n \geq 1$ and $0 \leq i < 2^n - 1$ and $1 \leq j \leq n$. Assume that $\theta_i^n[j] \neq \theta_{i+1}^n[j]$. By Proposition 6.30, for some $0 \leq k < 2^j - 1$ we have that

$$\begin{aligned} \theta_i^n[j...1] &= \theta_k^j \\ \theta_{i+1}^n[j...1] &= \theta_{k+1}^j \end{aligned}$$

By Definition 6.25, we have that

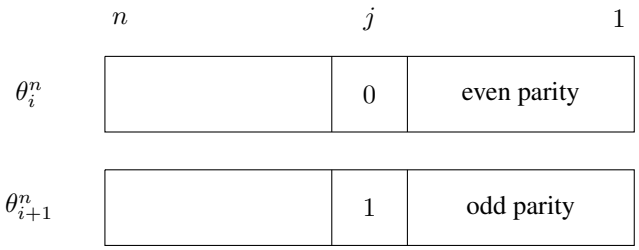
$$\theta_k^j = \begin{cases} 0\theta_{\frac{k}{2}}^{j-1} & \text{if } k \bmod 4 = 0 \\ 1\theta_{\frac{k-1}{2}}^{j-1} & \text{if } k \bmod 4 = 1 \\ 1\theta_{\frac{k}{2}}^{j-1} & \text{if } k \bmod 4 = 2 \\ 0\theta_{\frac{k-1}{2}}^{j-1} & \text{if } k \bmod 4 = 3 \end{cases}$$

and

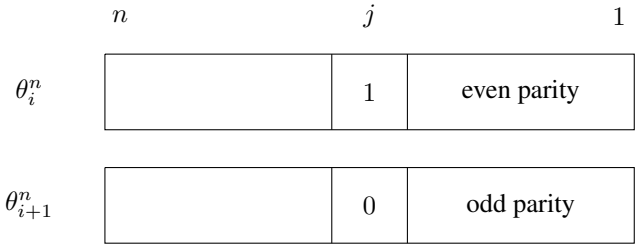
$$\theta^j_{k+1} = \begin{cases} 1\theta^{j-1}_{\frac{k}{2}} & \text{if } k \bmod 4 = 0 \\ 1\theta^{j-1}_{\frac{k+1}{2}} & \text{if } k \bmod 4 = 1 \\ 0\theta^{j-1}_{\frac{k}{2}} & \text{if } k \bmod 4 = 2 \\ 0\theta^{j-1}_{\frac{k+1}{2}} & \text{if } k \bmod 4 = 3 \end{cases}$$

We distinguish two cases.

- Assume that $\theta^n_i[j] = \theta^j_k[j] = 0$ and $\theta^n_{i+1}[j] = \theta^j_{k+1}[j] = 1$. Then we can conclude that $\theta^j_k = 0\theta^{j-1}_{\frac{k}{2}}$, $\theta^j_{k+1} = 1\theta^{j-1}_{\frac{k}{2}}$, and $k \bmod 4 = 0$. Because $k \bmod 4 = 0$, we have that $\frac{k}{2} \bmod 2 = 0$ and, by Proposition 6.29, we know that $\theta^{j-1}_{\frac{k}{2}}$ has even parity. Therefore, θ^j_k has even parity and θ^j_{k+1} has odd parity. Hence, $\theta^n_i[j..1]$ has even parity and $\theta^n_{i+1}[j..1]$ has odd parity as shown below.



- Assume that $\theta^n_i[j] = \theta^j_k[j] = 1$ and $\theta^n_{i+1}[j] = \theta^j_{k+1}[j] = 0$. Then we can conclude that $\theta^j_k = 1\theta^{j-1}_{\frac{k}{2}}$, $\theta^j_{k+1} = 0\theta^{j-1}_{\frac{k}{2}}$, and $k \bmod 4 = 2$. Because $k \bmod 4 = 2$, we have that $\frac{k}{2} \bmod 2 = 1$ and, by Proposition 6.29, we know that $\theta^{j-1}_{\frac{k}{2}}$ has odd parity. Therefore, θ^j_k has even parity and θ^j_{k+1} has odd parity. Hence, $\theta^n_i[j..1]$ has even parity and $\theta^n_{i+1}[j..1]$ has odd parity as shown below.



□

Finally, the next proposition proves that the rightmost bit of a bit string is equal to 0 if and only if the index of the bit string lies within the first half of the entire sequence.

Proposition 6.32. *For all $n \geq 1$ and $0 \leq i < 2^n$, $\theta^n_i[1] = 0$ if and only if $i < 2^{n-1}$.*

Proof. We prove this proposition by induction on n.

- In the base case, when $n = 1$, we have that

$$\theta^1_0[1] = 0$$
$$\theta^1_1[1] = 1$$

Hence, $\theta^1_i[1] = 0$ iff $i < 1$.

- In the inductive case, $n > 1$, by Definition 6.25, we have that

$$\theta_i^n = \begin{cases} 0\theta_{\frac{i}{2}}^{n-1} & \text{if } i \bmod 4 = 0 \\ 1\theta_{\frac{i-1}{2}}^{n-1} & \text{if } i \bmod 4 = 1 \\ 1\theta_{\frac{i}{2}}^{n-1} & \text{if } i \bmod 4 = 2 \\ 0\theta_{\frac{i-1}{2}}^{n-1} & \text{if } i \bmod 4 = 3 \end{cases}$$

, which implies that

$$\theta_i^n[1] = \begin{cases} \theta_{\frac{i}{2}}^{n-1}[1] & \text{if } i \bmod 4 = 0 \\ \theta_{\frac{i-1}{2}}^{n-1}[1] & \text{if } i \bmod 4 = 1 \\ \theta_{\frac{i}{2}}^{n-1}[1] & \text{if } i \bmod 4 = 2 \\ \theta_{\frac{i-1}{2}}^{n-1}[1] & \text{if } i \bmod 4 = 3 \end{cases}$$

We consider two cases.

  - Assume that i is even. Then $\theta_i^n[1] = \theta_{\frac{i}{2}}^{n-1}[1] = 0$ iff $\frac{i}{2} < 2^{n-2}$ by induction. That is, $i < 2^{n-1}$.
  - Assume that i is odd. Then $\theta_i^n[1] = \theta_{\frac{i-1}{2}}^{n-1}[1] = 0$ iff $\frac{i-1}{2} < 2^{n-2}$ by induction. That is, $i - 1 < 2^{n-1}$. Since i is odd, this is equivalent to $i < 2^{n-1}$.

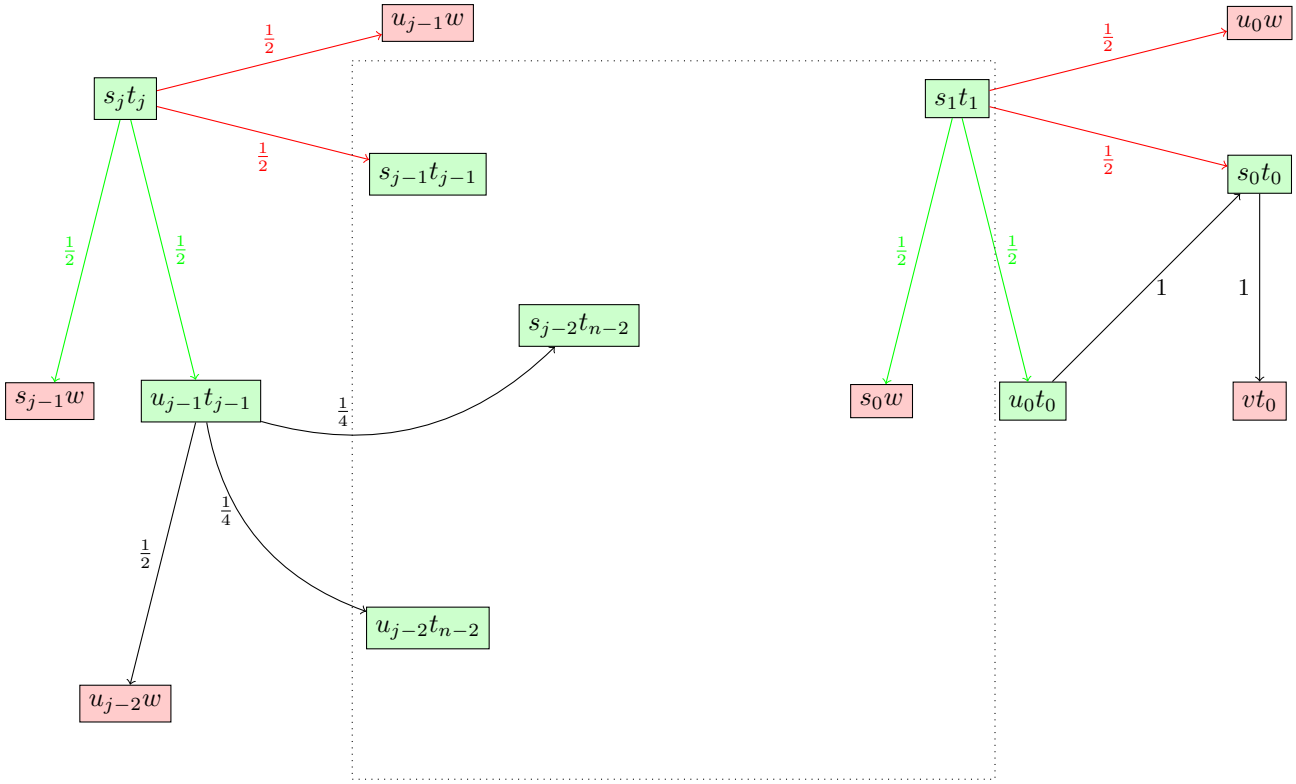
□

For $n \geq 1$ and $0 \leq i < 2^n$, each bit string θ_i^n defines a policy P_i^n. According to Proposition 3.15, for each $(x, y) \in S_?^2$ there exists $\omega_{xy} \in V(\Omega(\tau(x), \tau(y)))$ such that $\delta_1(x, y) = \omega_{xy} \cdot \delta_1$. For the remainder of this section, we fix such a ω_{xy} for each $(x, y) \in S_?^2$.

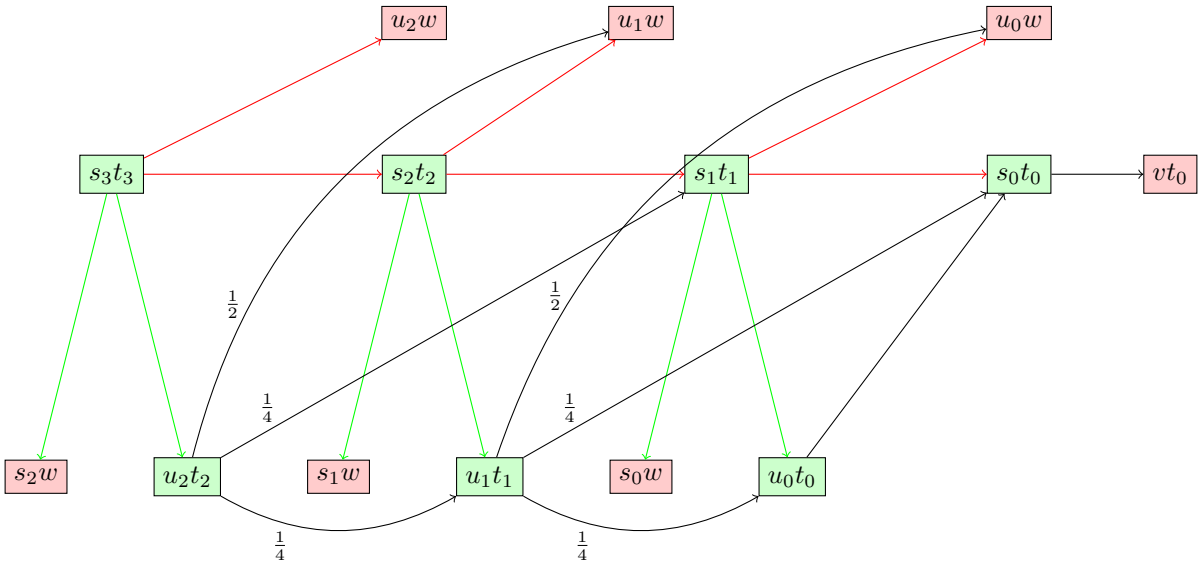
Definition 6.33. *The vertex policy P_i^n is defined by*

- $P_i^n(u_0, t_0) = \pi_0$
- $P_i^n(s_0, t_0) = \rho_0$
- *For* $1 \leq j \leq n$,
  - $P_i^n(s_j, t_j) = \begin{cases} \omega_{0j} & \textit{if } \theta_i^n[j] = 0 \\ \omega_{1j} & \textit{if } \theta_i^n[j] = 1 \end{cases}$
  - $P_i^n(u_j, t_j) = \pi_j$
- *For all other* $(x, y) \in S_?^2$, $P_i^n(x, y) = \omega_{xy}$.

The diagram below shows the policies defined above, where if a state pair has two matchings for the transitions of their target states, one is depicted with red arrows and the other with green arrows. If there is only one matching, it is depicted with black arrows. The dashed square in the diagram represents the remaining matching for the states (s_j, t_j) where $1 < j < 2^{n-1}$.

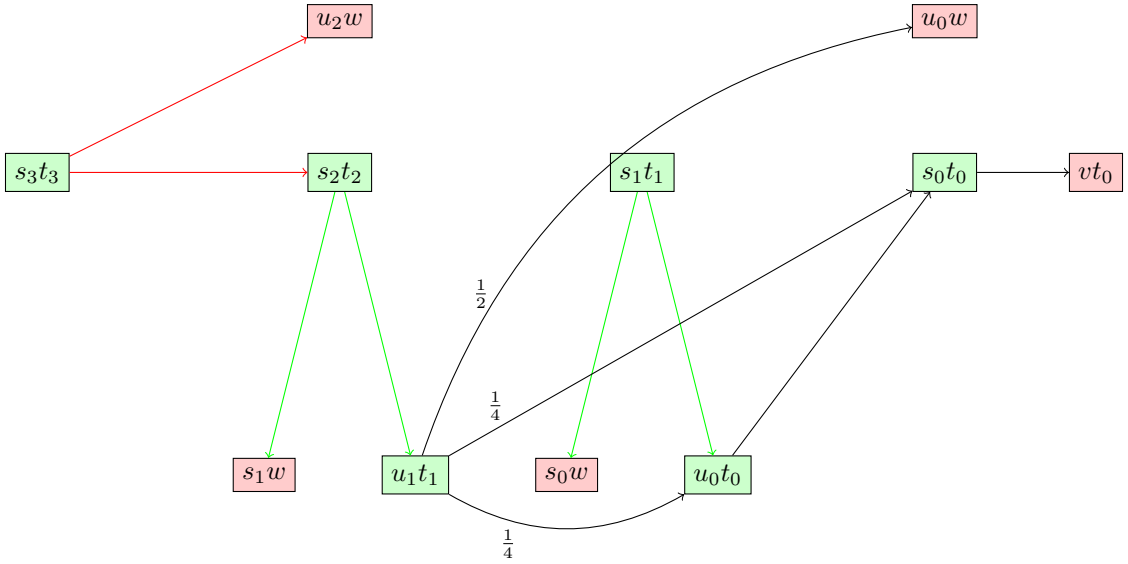


The next diagram shows all the matchings of the policy when $n = 3$. The edges of matchings are colored as mentioned earlier. In this diagram, the actual probabilities are denoted only if the probabilities of the outgoing transitions of a state pair are not all the same, as is the case for the state pair (u_1, t_1).



The next example shows the diagram of a policy P_4^3.

Example 6.34. *As we saw in the Example 6.26,* $\theta_4^3 = 011$. *Its corresponding policy is* P_4^3, *which is depicted below.*



The probabilities and edge colorings follow the conventions introduced earlier.

The next proposition proves that the policies defined above are optimal.

Proposition 6.35. *For all $n \geq 1$ and $0 \leq i < 2^n$, P_i^n is optimal.*

Proof. Let $n \geq 1$ and $0 \leq i < 2^n$ and $(x, y) \in S_?^2$. We distinguish the following cases.

- If $(x, y) = (u_0, t_0)$ then

$$\begin{aligned} P_i^n(u_0, t_0) \cdot \delta_1 &= \pi_0 \cdot \delta_1 \\ &= \delta_1(u_0, t_0). \qquad \text{[Proposition 6.23(b)]} \end{aligned}$$

- If $(x, y) = (s_0, t_0)$ then

$$\begin{aligned} P_i^n(s_0, t_0) \cdot \delta_1 &= \rho_0 \cdot \delta_1 \\ &= \delta_1(s_0, t_0). \qquad \text{[Proposition 6.23(a)]} \end{aligned}$$

- If $(x, y) = (u_j, t_j)$ for some $1 \leq j \leq n$ then

$$\begin{aligned} P_i^n(u_j, t_j) \cdot \delta_1 &= \pi_j \cdot \delta_1 \\ &= \delta_1(u_j, t_j). \qquad \text{[Proposition 6.23(b)]} \end{aligned}$$

- Assume $(x, y) = (s_j, t_j)$ for some $1 \leq j \leq n$. We distinguish two cases.
  - If $\theta_i^n[j] = 0$ then

$$\begin{aligned} P_i^n(s_j, t_j) \cdot \delta_1 &= \omega_{0j} \cdot \delta_1 \\ &= \delta_1(s_j, t_j) \qquad \text{[Proposition 6.23(c)]} \end{aligned}$$

  - If $\theta_i^n[j] = 1$ then

$$\begin{aligned} P_i^n(s_j, t_j) \cdot \delta_1 &= \omega_{1j} \cdot \delta_1 \\ &= \delta_1(s_j, t_j). \qquad \text{[Proposition 6.23(d)]} \end{aligned}$$

- For all other $(x, y) \in S_?^2$,

$$\begin{aligned} P_i^n(x, y) \cdot \delta_1 &= \omega_{xy} \cdot \delta_1 \\ &= \delta_1(x, y) \end{aligned}$$

Hence, by Proposition 4.21, P_i^n is an optimal policy. □

The next proposition links the policies and the bit strings.

Proposition 6.36. *For all $n \geq 1$ and $0 \leq i < 2^n - 1$ and $1 \leq j \leq n$, if $\theta_i^n[j] = 0$ and $\theta_{i+1}^n[j] = 1$ then*

$$P_{i+1}^n(s,t) = \begin{cases} \omega_{1j} & \text{if } (s,t) = (s_j, t_j) \\ P_i^n(s,t) & \text{otherwise} \end{cases}$$

Proof. Let $n \geq 1$ and $0 \leq i < 2^n - 1$ and $1 \leq j \leq n$. We assume that $\theta_i^n[j] = 0$ and $\theta_{i+1}^n[j] = 1$. By Definition 6.33, $P_{i+1}^n(s_j, t_j) = \omega_{1j}$. It remains to show that for all $(x,y) \neq (s_j, t_j)$, $P_{i+1}^n(x,y) = P_i^n(x,y)$. Let $(x,y) \neq (s_j, t_j)$. We distinguish the following cases.

- If $(x,y) = (u_0, t_0)$ then
$$P_{i+1}^n(u_0, t_0) = \pi_0 = P_i^n(u_0, t_0).$$
- If $(x,y) = (s_0, t_0)$ then
$$P_{i+1}^n(s_0, t_0) = \rho_0 = P_i^n(s_0, t_0).$$
- If $(x,y) = (s_k, t_k)$ for some $1 \leq k \leq n$ and $k \neq j$ then we consider two cases.
  - If $\theta_{i+1}^n[k] = 0$ then $\theta_i^n[k] = 0$ because $\theta_{i+1}^n[j] \neq \theta_i^n[j]$ and $k \neq j$ and θ_{i+1}^n and θ_i^n differ at a single bit (Proposition 6.28). Therefore,
  $$P_{i+1}^n(s_k, t_k) = \omega_{0k} = P_i^n(s_k, t_k).$$
  - The case, where $\theta_{i+1}^n[k] = 1$, can be proved similarly.
- If $(x,y) = (u_k, t_k)$ for some $1 \leq k \leq n$ then
$$P_{i+1}^n(u_k, t_k) = \pi_k = P_i^n(u_k, t_k).$$
- Otherwise
$$P_{i+1}^n(x,y) = \omega_{xy} = P_i^n(x,y).$$

□

The next proposition defines the expected lengths to reach a 1-pair based on the policies P_i^n.

Proposition 6.37. *For all $n \geq 1$ and $0 \leq i < 2^n$,*

$$\lambda_{1P_i^n}(s_0, t_0) = 1$$
$$\lambda_{1P_i^n}(u_0, t_0) = 2$$

and for all $j > 0$,

$$\lambda_{1P_i^n}(s_j, t_j) = \begin{cases} 1 + \frac{1}{2}\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) & \text{if } \theta_i^n[j] = 0 \\ 1 + \frac{1}{2}\lambda_{1P_i^n}(u_{j-1}, t_{j-1}) & \text{if } \theta_i^n[j] = 1 \end{cases}$$

and

$$\lambda_{1P_i^n}(u_j, t_j) = 1 + \frac{1}{4}\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) + \frac{1}{4}\lambda_{1P_i^n}(u_{j-1}, t_{j-1})$$

Proof. Let $n \geq 1$ and $0 \leq i < 2^n$. We have that

$$\begin{aligned} \lambda_{1P_i^n}(s_0, t_0) &= \Lambda_{1P_i^n}(\lambda_{1P_i^n})(s_0, t_0) \\ &= P_i^n(s_0, t_0) \cdot (\delta_1 + \lambda_{1P_i^n}) \\ &= \rho_0 \cdot (\delta_1 + \lambda_{1P_i^n}) \\ &= \delta_1(t_0, v) + \lambda_{1P_i^n}(t_0, v) \\ &= 1 + \lambda_{1P_i^n}(t_0, v) \qquad \text{[Proposition 6.20(a)]} \\ &= 1 \qquad [\lambda_{1P_i^n}(t_0, v) = \Lambda_{1P_i^n}(\lambda_{1P_i^n}) = 0 \text{ as } (t_0.v) \in S_1^2] \end{aligned}$$

and

$$\begin{aligned}
\lambda_{1P_i^n}(u_0,t_0) &= \Lambda_{1P_i^n}(\lambda_{1P_i^n})(u_0,t_0)\\
&= P_i^n(u_0,t_0)\cdot(\delta_1+\lambda_{1P_i^n})\\
&= \pi_0\cdot(\delta_1+\lambda_{1P_i^n})\\
&= \delta_1(s_0,t_0)+\lambda_{1P_i^n}(s_0,t_0)\\
&= 1+\lambda_{1P_i^n}(s_0,t_0) \qquad \text{[Proposition 6.20(d)]}\\
&= 2 \qquad [\lambda_{1P_i^n}(s_0,t_0)=1 \text{ as shown above}]
\end{aligned}$$

Let $j>0$. We consider two cases.

- If $\theta_i^n[j]=0$ then

$$\begin{aligned}
&\lambda_{1P_i^n}(s_j,t_j)\\
&= \Lambda_{1P_i^n}(\lambda_{1P_i^n})(s_j,t_j)\\
&= P_i^n(s_j,t_j)\cdot(\delta_1+\lambda_{1P_i^n})\\
&= \omega_{0j}\cdot(\delta_1+\lambda_{1P_i^n})\\
&= \frac{1}{2}\left(\delta_1(u_{j-1},w)+\lambda_{1P_i^n}(u_{j-1},w)\right)+\frac{1}{2}\left(\delta_1(s_{j-1},t_{j-1})+\lambda_{1P_i^n}(s_{j-1},t_{j-1})\right)\\
&= \frac{1}{2}\left(1+\lambda_{1P_i^n}(u_{j-1},w)\right)+\frac{1}{2}\left(1+\lambda_{1P_i^n}(s_{j-1},t_{j-1})\right) \qquad \text{[Proposition 6.20(c) and (d)]}\\
&= \frac{1}{2}+\frac{1}{2}\left(1+\lambda_{1P_i^n}(s_{j-1},t_{j-1})\right)\\
&\quad [\lambda_{1P_i^n}(u_{j-1},w)=\Lambda_{1P_i^n}(\lambda_{1P_i^n})(u_{j-1},w)=0 \text{ as } (u_{j-1},w)\in S_1^2]\\
&= 1+\frac{1}{2}\lambda_{1P_i^n}(s_{j-1},t_{j-1})
\end{aligned}$$

- If $\theta_i^n[j]=1$ then

$$\begin{aligned}
&\lambda_{1P_i^n}(s_j,t_j)\\
&= \Lambda_{1P_i^n}(\lambda_{1P_i^n})(s_j,t_j)\\
&= P_i^n(s_j,t_j)\cdot(\delta_1+\lambda_{1P_i^n})\\
&= \omega_{1j}\cdot(\delta_1+\lambda_{1P_i^n})\\
&= \frac{1}{2}\left(\delta_1(s_{j-1},w)+\lambda_{1P_i^n}(s_{j-1},w)\right)+\frac{1}{2}\left(\delta_1(u_{j-1},t_{j-1})+\lambda_{1P_i^n}(u_{j-1},t_{j-1})\right)\\
&= \frac{1}{2}\left(1+\lambda_{1P_i^n}(s_{j-1},w)\right)+\frac{1}{2}\left(1+\lambda_{1P_i^n}(u_{j-1},t_{j-1})\right) \qquad \text{[Proposition 6.20(c) and (d)]}\\
&= \frac{1}{2}+\frac{1}{2}\left(1+\lambda_{1P_i^n}(u_{j-1},t_{j-1})\right)\\
&\quad [\lambda_{1P_i^n}(s_{j-1},w)=\Lambda_{1P_i^n}(\lambda_{1P_i^n})(s_{j-1},w)=0 \text{ as } (s_{j-1},w)\in S_1^2]\\
&= 1+\frac{1}{2}\lambda_{1P_i^n}(u_{j-1},t_{j-1})
\end{aligned}$$

and

$$
\begin{aligned}
&\lambda_{1P_i^n}(u_j, t_j) \\
&= \Lambda_{1P_i^n}(\lambda_{1P_i^n})(u_j, t_j) \\
&= P_i^n(u_j, t_j) \cdot (\delta_1 + \lambda_{1P_i^n}) \\
&= \pi_j \cdot (\delta_1 + \lambda_{1P_i^n}) \\
&= \frac{1}{2}\left(\delta_1(u_{j-1}, w) + \lambda_{1P_i^n}(u_{j-1}, w)\right) + \frac{1}{4}\left(\delta_1(u_{j-1}, t_{j-1}) + \lambda_{1P_i^n}(u_{j-1}, t_{j-1})\right) + \\
&\quad \frac{1}{4}\left(\delta_1(s_{j-1}, t_{j-1}) + \lambda_{1P_i^n}(s_{j-1}, t_{j-1})\right) \\
&= \frac{1}{2}\left(1 + \lambda_{1P_i^n}(u_{j-1}, w)\right) + \frac{1}{4}\left(1 + \lambda_{1P_i^n}(u_{j-1}, t_{j-1})\right) + \frac{1}{4}\left(1 + \lambda_{1P_i^n}(s_{j-1}, t_{j-1})\right) \\
&\qquad \text{[Proposition 6.20(c) and (d)]} \\
&= \frac{1}{2} + \frac{1}{4}\left(1 + \lambda_{1P_i^n}(u_{j-1}, t_{j-1})\right) + \frac{1}{4}\left(1 + \lambda_{1P_i^n}(s_{j-1}, t_{j-1})\right) \\
&\qquad [\lambda_{1P_i^n}(u_{j-1}, w) = \Lambda_{1P_i^n}(\lambda_{1P_i^n})(u_{j-1}, w) = 0 \text{ as } (u_{j-1}, w) \in S_1^2] \\
&= 1 + \frac{1}{4}\lambda_{1P_i^n}(u_{j-1}, t_{j-1}) + \frac{1}{4}\lambda_{1P_i^n}(s_{j-1}, t_{j-1})
\end{aligned}
$$

□

Example 6.38. *Since each choice of matching of the target states of state pairs (s_j, t_j) gives rise to a new policy, there are 2^n policies, $\{P_1^n, ..., P_{2^n}^n\}$, where P_i^n is defined based on the bit string θ_i^n, for $1 \leq i \leq n$.*

Consider the state pairs (s_3, t_3), (s_2, t_2), and (s_1, t_1) depicted in the Example 6.19. For each policy of that LMC, the table below shows the expected length to reach a 1-pair, along with the corresponding bit string. As observed in the table, when the bit value at index j in the bit string changes, the corresponding value of $\Lambda_{1P_i^3}(s_j, t_j)$ also changes.

i	θ_i^3	$\Lambda_{1P_i^3}(s_3, t_3)$	$\Lambda_{1P_i^3}(s_2, t_2)$	$\Lambda_{1P_i^3}(s_1, t_1)$
0	000	1.78	1.75	1.5
1	100	1.9	1.75	1.5
2	110	1.9	1.875	1.5
3	010	1.93	1.875	1.5
4	011	1.93	1.875	2
5	111	1.96	1.875	2
6	101	1.96	2	2
7	001	2	2	2

The expected lengths for the state pairs (s_3, t_3), (s_2, t_2), and (s_1, t_1), as shown in the table above, can be easily verified by the reader. Furthermore, the table shows that it takes 8 iterations to find a 1-maximal optimal vertex policy.

Consider a bit string θ_i^n with corresponding policy P_i^n. The next proposition proves that the difference in the expected lengths to a 1-pair for this policy, starting from (s_j, t_j) and (u_j, t_j), is $\frac{1}{4^j}$.

Proposition 6.39. *For all $n \geq 1$ and $0 \leq i < 2^n$ and $1 \leq j \leq n$, if $\theta_i^n[j...1]$ has odd parity then*

$$\lambda_{1P_i^n}(s_j, t_j) - \lambda_{1P_i^n}(u_j, t_j) = \frac{1}{4^j}$$

otherwise,

$$\lambda_{1P_i^n}(u_j, t_j) - \lambda_{1P_i^n}(s_j, t_j) = \frac{1}{4^j}$$

Proof. Let $n \geq 1$, $0 \leq i < 2^n$ and $1 \leq j \leq n$. We distinguish two cases.

- Assume $n = 1$. Then $j = 1$ and $i \in \{0, 1\}$. We consider two cases.

– If $i = 0$ then $\theta_0^1[1] = 0$ has even parity and since

$$\begin{aligned}\lambda_{1P_0^n}(s_1, t_1) &= 1 + \frac{1}{2}\lambda_{1P_0^n}(s_0, t_0) \qquad \text{[Proposition 6.37]} \\ &= 1 + \frac{1}{2} \qquad \text{[Proposition 6.37]} \\ &= \frac{3}{2}\end{aligned}$$

and

$$\begin{aligned}\lambda_{1P_0^n}(u_1, t_1) &= 1 + \frac{1}{4}\lambda_{1P_0^n}(s_0, t_0) + \frac{1}{4}\lambda_{1P_0^n}(u_0, t_0) \qquad \text{[Proposition 6.37]} \\ &= 1 + \frac{1}{4} + \frac{1}{4}\lambda_{1P_0^n}(u_0, t_0) \qquad \text{[Proposition 6.37]} \\ &= 1 + \frac{1}{4} + \frac{1}{2} \qquad \text{[Proposition 6.37]} \\ &= \frac{7}{4}\end{aligned}$$

we can conclude that

$$\begin{aligned}\lambda_{1P_0^n}(u_1, t_1) - \lambda_{1P_0^n}(s_1, t_1) &= \frac{7}{4} - \frac{3}{2} \\ &= \frac{1}{4}\end{aligned}$$

– If $i = 1$ then $\theta_1^1[1] = 1$ has odd parity and since

$$\begin{aligned}\lambda_{1P_1^n}(s_1, t_1) &= 1 + \frac{1}{2}\lambda_{1P_1^n}(u_0, t_0) \qquad \text{[Proposition 6.37]} \\ &= 1 + 1 \qquad \text{[Proposition 6.37]} \\ &= 2\end{aligned}$$

and

$$\begin{aligned}\lambda_{1P_1^n}(u_1, t_1) &= 1 + \frac{1}{4}\lambda_{1P_1^n}(s_0, t_0) + \frac{1}{4}\lambda_{1P_1^n}(u_0, t_0) \qquad \text{[Proposition 6.37]} \\ &= 1 + \frac{1}{4} + \frac{1}{4}\lambda_{1P_0^n}(u_0, t_0) \qquad \text{[Proposition 6.37]} \\ &= 1 + \frac{1}{4} + \frac{1}{2} \qquad \text{[Proposition 6.37]} \\ &= \frac{7}{4}\end{aligned}$$

we can conclude that

$$\begin{aligned}\lambda_{1P_1^n}(s_1, t_1) - \lambda_{1P_1^n}(u_1, t_1) &= 2 - \frac{7}{4} \\ &= \frac{1}{4}\end{aligned}$$

- Assume $n > 1$. We prove this case by induction on j. The base case, $j = 1$, is similar to the $n = 1$ case. In the inductive case, $j > 1$. We consider four cases.

– If $\theta_i^n[j] = 0$ and $\theta_i^n[j...1]$ has even parity then $\theta_i^n[(j-1)...1]$ has also even parity. Furthermore,

$$\begin{aligned}
&\lambda_{1P_i^n}(u_j, t_j) - \lambda_{1P_i^n}(s_j, t_j) \\
&= \left(1 + \frac{1}{4}\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) + \frac{1}{4}\lambda_{1P_i^n}(u_{j-1}, t_{j-1})\right) - \left(1 + \frac{1}{2}\lambda_{1P_i^n}(s_{j-1}, t_{j-1})\right) \\
&\quad \text{[Proposition 6.37]} \\
&= \frac{1}{4}\lambda_{1P_i^n}(u_{j-1}, t_{j-1}) - \frac{1}{4}\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) \\
&= \frac{1}{4}\left(\lambda_{1P_i^n}(u_{j-1}, t_{j-1}) - \lambda_{1P_i^n}(s_{j-1}, t_{j-1})\right) \\
&= \frac{1}{4}\left(\frac{1}{4^{j-1}}\right) \qquad \text{[induction hypothesis]} \\
&= \frac{1}{4^j}
\end{aligned}$$

– If $\theta_i^n[j] = 0$ and $\theta_i^n[j...1]$ has odd parity then $\theta_i^n[(j-1)...1]$ has also odd parity. Furthermore,

$$\begin{aligned}
&\lambda_{1P_i^n}(s_j, t_j) - \lambda_{1P_i^n}(u_j, t_j) \\
&= \left(1 + \frac{1}{2}\lambda_{1P_i^n}(s_{j-1}, t_{j-1})\right) - \left(1 + \frac{1}{4}\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) + \frac{1}{4}\lambda_{1P_i^n}(u_{j-1}, t_{j-1})\right) \\
&\quad \text{[Proposition 6.37]} \\
&= \frac{1}{4}\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) - \frac{1}{4}\lambda_{1P_i^n}(u_{j-1}, t_{j-1}) \\
&= \frac{1}{4}\left(\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) - \lambda_{1P_i^n}(u_{j-1}, t_{j-1})\right) \\
&= \frac{1}{4}\left(\frac{1}{4^{j-1}}\right) \qquad \text{[induction hypothesis]} \\
&= \frac{1}{4^j}
\end{aligned}$$

– If $\theta_i^n[j] = 1$ and $\theta_i^n[j...1]$ has odd parity then $\theta_i^n[(j-1)...1]$ has even parity. Furthermore,

$$\begin{aligned}
&\lambda_{1P_i^n}(u_j, t_j) - \lambda_{1P_i^n}(s_j, t_j) \\
&= \left(1 + \frac{1}{4}\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) + \frac{1}{4}\lambda_{1P_i^n}(u_{j-1}, t_{j-1})\right) - \left(1 + \frac{1}{2}\lambda_{1P_i^n}(u_{j-1}, t_{j-1})\right) \\
&\quad \text{[Proposition 6.37]} \\
&= \frac{1}{4}\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) - \frac{1}{4}\lambda_{1P_i^n}(u_{j-1}, t_{j-1}) \\
&= \frac{1}{4}\left(\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) - \lambda_{1P_i^n}(u_{j-1}, t_{j-1})\right) \\
&= \frac{1}{4}\left(\frac{1}{4^{j-1}}\right) \qquad \text{[induction hypothesis]} \\
&= \frac{1}{4^j}
\end{aligned}$$

– If $\theta_i^n[j] = 1$ and $\theta_i^n[j...1]$ has even parity then $\theta_i^n[(j-1)...1]$ has odd parity. Furthermore,

$$\begin{aligned}
&\lambda_{1P_i^n}(s_j, t_j) - \lambda_{1P_i^n}(u_j, t_j) \\
&= \left(1 + \frac{1}{2}\lambda_{1P_i^n}(u_{j-1}, t_{j-1})\right) - \left(1 + \frac{1}{4}\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) + \frac{1}{4}\lambda_{1P_i^n}(u_{j-1}, t_{j-1})\right) \\
&\qquad [\text{ Proposition 6.37}] \\
&= \frac{1}{4}\lambda_{1P_i^n}(u_{j-1}, t_{j-1}) - \frac{1}{4}\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) \\
&= \frac{1}{4}\left(\lambda_{1P_i^n}(u_{j-1}, t_{j-1}) - \lambda_{1P_i^n}(s_{j-1}, t_{j-1})\right) \\
&= \frac{1}{4}\left(\frac{1}{4^{j-1}}\right) \qquad [\text{induction hypothesis}] \\
&= \frac{1}{4^j}
\end{aligned}$$

□

The next proposition shows that if two consecutive bit strings are equal from index 1 to j, the prior bit string's policy has the same expected length to reach a 1-pair as the next bit string's policy, both starting from (s_j, t_j) or (u_j, t_j).

Proposition 6.40. *For all $n \geq 1$ and $0 \leq i < 2^n - 1$ and $1 \leq j \leq n$, if $\theta_i^n[j...1] = \theta_{i+1}^n[j...1]$ then $\lambda_{1P_i^n}(s_j, t_j) = \lambda_{P_{i+1}^n}(s_j, t_j)$ and $\lambda_{1P_i^n}(u_j, t_j) = \lambda_{P_{i+1}^n}(u_j, t_j)$.*

Proof. We prove this proposition by induction on j.

- In the base case, $j = 0$, we have that

$$\begin{aligned}
\lambda_{1P_i^n}(s_0, t_0) = 1 = \lambda_{1P_{i+1}^n}(s_0, t_0) \qquad &[\text{Proposition 6.37}] \\
\lambda_{1P_i^n}(u_0, t_0) = 2 = \lambda_{1P_{i+1}^n}(u_0, t_0) \qquad &[\text{Proposition 6.37}]
\end{aligned}$$

- In the inductive case, $j > 0$. We consider two cases.
  - If $\theta_i^n[j] = \theta_{i+1}^n[j] = 0$ then

$$\begin{aligned}
\lambda_{1P_i^n}(s_j, t_j) &= 1 + \frac{1}{2}\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) \qquad [\text{Proposition 6.37}] \\
&= 1 + \frac{1}{2}\lambda_{1P_{i+1}^n}(s_{j-1}, t_{j-1}) \qquad [\text{induction hypothesis}] \\
&= \lambda_{1P_{i+1}^n}(s_j, t_j) \qquad [\text{Proposition 6.37}]
\end{aligned}$$

  - If $\theta_i^n[j] = \theta_{i+1}^n[j] = 1$ then

$$\begin{aligned}
\lambda_{1P_i^n}(s_j, t_j) &= 1 + \frac{1}{2}\lambda_{1P_i^n}(u_{j-1}, t_{j-1}) \qquad [\text{Proposition 6.37}] \\
&= 1 + \frac{1}{2}\lambda_{1P_{i+1}^n}(u_{j-1}, t_{j-1}) \qquad [\text{induction hypothesis}] \\
&= \lambda_{1P_{i+1}^n}(s_j, t_j) \qquad [\text{Proposition 6.37}]
\end{aligned}$$

  and

$$\begin{aligned}
\lambda_{1P_i^n}(u_j, t_j) &= 1 + \frac{1}{4}\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) + \frac{1}{4}\lambda_{1P_i^n}(u_{j-1}, t_{j-1}) \qquad [\text{Proposition 6.37}] \\
&= 1 + \frac{1}{4}\lambda_{1P_{i+1}^n}(s_{j-1}, t_{j-1}) + \frac{1}{4}\lambda_{1P_{i+1}^n}(u_{j-1}, t_{j-1}) \qquad [\text{induction hypothesis}] \\
&= \lambda_{1P_{i+1}^n}(u_j, t_j) \qquad [\text{Proposition 6.37}]
\end{aligned}$$

□

The next proposition shows that if two consecutive bit strings differ at index j, the prior bit string's policy has a shorter expected length to reach a 1-pair than the next bit string's policy, with both starting from (s_j, t_j). In essence, it shows that (s_j, t_j) is locally non-optimal.

Proposition 6.41. *For all $n \geq 1$ and $0 \leq i < 2^n - 1$ and $1 \leq j \leq n$, if $\theta_i^n[j] \neq \theta_{i+1}^n[j]$ then $\lambda_{1P_i^n}(s_j, t_j) < \lambda_{P_{i+1}^n}(s_j, t_j)$.*

Proof. Let $n \geq 1$ and $0 \leq i < 2^n$ and $1 \leq j \leq n$. Assume that $\theta_i^n[j] \neq \theta_{i+1}^n[j]$. We consider two cases.

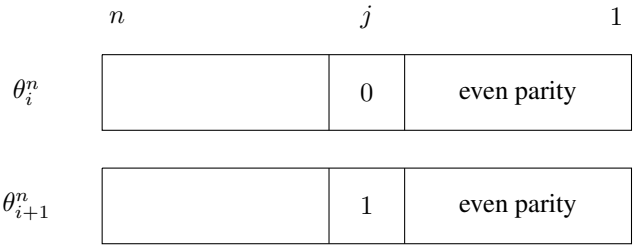
- If $j = 1$ then, by Proposition 6.32, we have that $\theta_i^n[1] = 0$ and $\theta_{i+1}^n[1] = 1$. Hence,

$$\begin{aligned}
\lambda_{1P_i^n}(s_1, t_1) &= 1 + \frac{1}{2}\lambda_{1P_i^n}(s_0, t_0) && \text{[Proposition 6.37]} \\
&= 1 + \frac{1}{2} && \text{[Proposition 6.37]} \\
&= \frac{3}{2} \\
&< 2 \\
&= 1 + 1 && \text{[Proposition 6.37]} \\
&= 1 + \frac{1}{2}\lambda_{1P_{i+1}^n}(u_0, t_0) && \text{[Proposition 6.37]} \\
&= \lambda_{1P_{i+1}^n}(s_1, t_1)
\end{aligned}$$

  Therefore, $\lambda_{1P_i^n}(s_1, t_1) < \lambda_{1P_{i+1}^n}(s_1, t_1)$.

- Let $1 < j \leq n$. Since θ_i^n and θ_{i+1}^n differ at index j, by Proposition 6.28, we can conclude that $\theta_i^n[(j-1)...1] = \theta_{i+1}^n[(j-1)...1]$. We consider two cases.

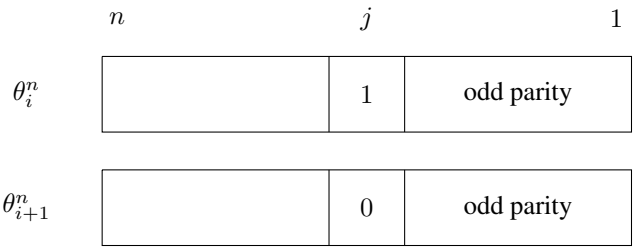
  - Assume that $\theta_i^n[j] = 0$ and $\theta_{i+1}^n[j] = 1$. By Proposition 6.31(b), $\theta_i^n[j..1]$ has even parity and $\theta_{i+1}^n[j..1]$ has odd parity. Hence, $\theta_i^n[(j-1)...1]$ and $\theta_{i+1}^n[(j-1)...1]$ have even parity.



    Furthermore,

$$\begin{aligned}
\lambda_{1P_i^n}(s_j, t_j) &= 1 + \frac{1}{2}\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) && \text{[Proposition 6.37]} \\
&= 1 + \frac{1}{2}\lambda_{1P_{i+1}^n}(s_{j-1}, t_{j-1}) && \text{[Proposition 6.40]} \\
&< 1 + \frac{1}{2}\lambda_{1P_{i+1}^n}(u_{j-1}, t_{j-1}) && \text{[Proposition 6.39]} \\
&= \lambda_{1P_{i+1}^n}(s_j, t_j) && \text{[Proposition 6.37]}
\end{aligned}$$

  - Assume that $\theta_i^n[j] = 1$ and $\theta_{i+1}^n[j] = 0$. By Proposition 6.31(a), $\theta_i^n[j..1]$ has even parity and $\theta_{i+1}^n[j..1]$ has odd parity. Hence, $\theta_i^n[(j-1)...1]$ and $\theta_{i+1}^n[(j-1)...1]$ have odd parity.



Furthermore,

$$\begin{aligned}
\lambda_{1P_i^n}(s_j, t_j) &= 1 + \frac{1}{2}\lambda_{1P_i^n}(u_{j-1}, t_{j-1}) && \text{[Proposition 6.37]} \\
&< 1 + \frac{1}{2}\lambda_{1P_i^n}(s_{j-1}, t_{j-1}) && \text{[Proposition 6.39]} \\
&= 1 + \frac{1}{2}\lambda_{1P_{i+1}^n}(s_{j-1}, t_{j-1}) && \text{[Proposition 6.40]} \\
&= \lambda_{1P_{i+1}^n}(s_j, t_j) && \text{[Proposition 6.37]}
\end{aligned}$$

□

Algorithm 1 1-maximal optimal vertex policy

1: $i \leftarrow 0$
2: $P \leftarrow$ an optimal vertex policy
3: compute λ_{1P}
4: **while** $\exists (s,t) \in S_?^2 : \Lambda_1(\lambda_{1P})(s,t) > \lambda_{1P}(s,t)$ **do**
5: $\quad \pi \leftarrow \underset{\omega \in V(\Omega(\tau(s),\tau(t))) \wedge \delta_1(s,t) = \omega \cdot \delta_1}{\arg\max} \omega \cdot (\delta_1 + \lambda_{1P})$
6: $\quad P(s,t) \leftarrow \pi$
7: $\quad$ compute λ_{1P}
8: $\quad i \leftarrow i + 1$
9: **end while**

Next, we show that there exists an execution of the above policy iteration algorithm which takes $\Omega(2^n)$ iterations to find a 1-maximal optimal vertex policy.

Theorem 6.42. *For each $n \geq 0$, there exists an execution of the algorithm for C_n such that for all $0 \leq i < 2^n$ at the start of the $(i+1)$-th iteration of the loop, $P = P_i^n$.*

Proof. Consider the LMC C_n defined in Definition 6.18. We prove this theorem by induction on i.

- In the base case, $i = 0$. In line 2, we can choose P_0^n and, by Proposition 6.35, we have that the vertex policy P_0^n is optimal. Hence, there exists an execution of the algorithm for C_n such that at the start of the first iteration of the loop, $P = P_0^n$.
- In the inductive case, $i > 0$, by the induction hypothesis, there exists an execution of the algorithm for C_n such that at the start of the i-th iteration of the loop, $P = P_{i-1}^n$. We extend the execution by first choosing (s_j, t_j) such that $\theta_{i-1}^n[j] \neq \theta_i^n[j]$ for (s,t) in line 4. By Proposition 6.28, we know that there always exists such a j. To show that this choice causes the execution to enter the loop, we need to prove that $\Lambda_1(\lambda_{1P_{i-1}^n})(s_j, t_j) > \lambda_{1P_{i-1}^n}(s_j, t_j)$. We distinguish two cases.
  - Assume that $\theta_{i-1}^n[j] = 0$. Then $\theta_i^n[j] = 1$ and, hence, $P_{i-1}^n(s_j, t_j) = \omega_{0j}$ and $P_i^n(s_j, t_j) = \omega_{1j}$. Since $\theta_{i-1}^n[j] \neq \theta_i^n[j]$, we can conclude from Proposition 6.28 that

$$\theta_{i-1}^n[(j-1)...1] = \theta_i^n[(j-1)...1]. \tag{6.2}$$

Furthermore,

$$
\begin{aligned}
&\omega_{0j} \cdot (\delta_1 + \lambda_{1P^n_{i-1}}) \\
&= P^n_{i-1}(s_j, t_j) \cdot (\delta_1 + \lambda_{1P^n_{i-1}}) && (6.3)\\
&= \Lambda_{1P^n_{i-1}}(\lambda_{1P^n_{i-1}})(s_j, t_j) \\
&= \lambda_{1P^n_{i-1}}(s_j, t_j) \\
&< \lambda_{1P^n_i}(s_j, t_j) \qquad [\theta^n_{i-1}[j] \neq \theta^n_i[j] \text{ and Proposition 6.41}] \\
&= \Lambda_{1P^n_i}(\lambda_{1P^n_i})(s_j, t_j) \\
&= P^n_i(s_j, t_j) \cdot (\delta_1 + \lambda_{1P^n_i}) \\
&= \omega_{1j} \cdot (\delta_1 + \lambda_{1P^n_i}) \\
&= \sum_{(u,v)\in D_1} \omega_{1j}(u, v)(\delta_1(u, v) + \lambda_{1P^n_i}(u, v)) \\
&= \omega_{1j}(s_{j-1}, w)(\delta_1(s_{j-1}, w) + \lambda_{1P^n_i}(s_{j-1}, w)) + && (6.4)\\
&\quad \omega_{1j}(u_{j-1}, t_{j-1})(\delta_1(u_{j-1}, t_{j-1}) + \lambda_{1P^n_i}(u_{j-1}, t_{j-1})) \\
&= \omega_{1j}(s_{j-1}, w)(1 + \lambda_{1P^n_i}(s_{j-1}, w)) + \omega_{1j}(u_{j-1}, t_{j-1})(1 + \lambda_{1P^n_i}(u_{j-1}, t_{j-1})) \\
&\quad \text{[Proposition 6.20(b) and (d)]} \\
&= 1 + \frac{1}{2}\lambda_{1P^n_i}(u_{j-1}, t_{j-1}) \\
&\quad [\lambda_{1P^n_i}(s_{j-1}, w) = \Lambda_{1P^n_i}(\lambda_{1P^n_i})(s_{j-1}, w) = 0 \text{ as } (s_{j-1}, w) \in S^2_1] \\
&= 1 + \frac{1}{2}\lambda_{1P^n_{i-1}}(u_{j-1}, t_{j-1}) \qquad [\text{Proposition 6.40 and (6.2)}] \\
&= \omega_{1j}(s_{j-1}, w)(1 + \lambda_{1P^n_{i-1}}(s_{j-1}, w)) + \omega_{1j}(u_{j-1}, t_{j-1})(1 + \lambda_{1P^n_{i-1}}(u_{j-1}, t_{j-1})) \\
&\quad [\lambda_{1P^n_{i-1}}(s_{j-1}, w) = \Lambda_{1P^n_{i-1}}(\lambda_{1P^n_{i-1}})(s_{j-1}, w) = 0 \text{ as } (s_{j-1}, w) \in S^2_1] \\
&= \omega_{1j}(s_{j-1}, w)(\delta_1(s_{j-1}, w) + \lambda_{1P^n_{i-1}}(s_{j-1}, w)) + \\
&\quad \omega_{1j}(u_{j-1}, t_{j-1})(\delta_1(u_{j-1}, t_{j-1}) + \lambda_{1P^n_{i-1}}(u_{j-1}, t_{j-1})) \\
&\quad \text{[Proposition 6.20(b) and (d)]} && (6.5)\\
&= \sum_{(u,v)\in D_1} \omega_{1j}(u, v)(\delta_1 + \lambda_{1P^n_{i-1}}(u, v)) \\
&= \omega_{1j} \cdot (\delta_1 + \lambda_{1P^n_{i-1}}).
\end{aligned}
$$

In addition,

$$
\begin{aligned}
&\Lambda_1(\lambda_{1P^n_{i-1}})(s_j, t_j) \\
&= \max_{\omega\in V(\Omega(\tau(s_j),\tau(t_j)))\wedge\delta_1(s_j,t_j)=\omega\cdot\delta_1} \omega \cdot (\delta_1 + \lambda_{1P^n_{i-1}}) && (6.6)\\
&= \max_{\omega\in\{\omega_{0j},\omega_{1j}\}} \omega \cdot (\delta_1 + \lambda_{1P^n_{i-1}}) \qquad [\text{Proposition 6.23(c) and (d) and 6.24}] && (6.7)\\
&= \max\{\omega_{0j} \cdot (\delta_1 + \lambda_{1P^n_{i-1}}), \omega_{1j} \cdot (\delta_1 + \lambda_{1P^n_{i-1}})\} \\
&= \omega_{1j} \cdot (\delta_1 + \lambda_{1P^n_{i-1}}) \qquad [(6.4)]
\end{aligned}
$$

Since

$$
\begin{aligned}
\Lambda_1(\lambda_{1P^n_{i-1}})(s_j, t_j) &= \omega_{1j} \cdot (\delta_1 + \lambda_{1P^n_{i-1}}) \qquad [(6.7)] \\
&> \omega_{0j} \cdot (\delta_1 + \lambda_{1P^n_{i-1}}) \qquad [(6.4)] \\
&= P^n_{i-1}(s_j, t_j) \cdot (\delta_1 + \lambda_{1P^n_{i-1}}) \\
&= \Lambda_{1P^n_{i-1}}(\lambda_{1P^n_{i-1}})(s_j, t_j) \\
&= \lambda_{1P^n_{i-1}}(s_j, t_j)
\end{aligned}
$$

the condition in line 4 is satisfied. Because, by Proposition 6.23(c) and (d) and 6.24,

$$\operatorname*{arg\,max}_{\omega \in V(\Omega(\tau(s),\tau(t))) \wedge \delta_1(s,t)=\omega\cdot\delta_1} \omega \cdot (\delta_1 + \lambda_{1P_{i-1}^n}) = \operatorname*{arg\,max}_{\omega \in \{\omega_{0j},\omega_{1j}\}} \omega \cdot (\delta_1 + \lambda_{1P_{i-1}^n})$$

and, by (6.4),

$$\operatorname*{arg\,max}_{\omega \in \{\omega_{0j},\omega_{1j}\}} \omega \cdot (\delta_1 + \lambda_{1P_{i-1}^n}) = \{\omega_{1j}\},$$

we assign ω_{1j} to π in line 5. Line 6 amounts to $P_{i-1}^n(s_j, t_j) \leftarrow \omega_{1j}$. Hence, by Proposition 6.36, we obtain P_i^n. Therefore, at the end of the loop we have $P = P_i^n$.

– The other case, where $\theta_{i-1}^n[j] = 1$, can be proved similarly.

□

Corollary 6.43. *For each $n \in \mathbb{N}$, there exists an LMC of size $\mathcal{O}(n)$ such that Algorithm 1 takes $\Omega(2^n)$ iterations.*

6.2 1-Conflict free

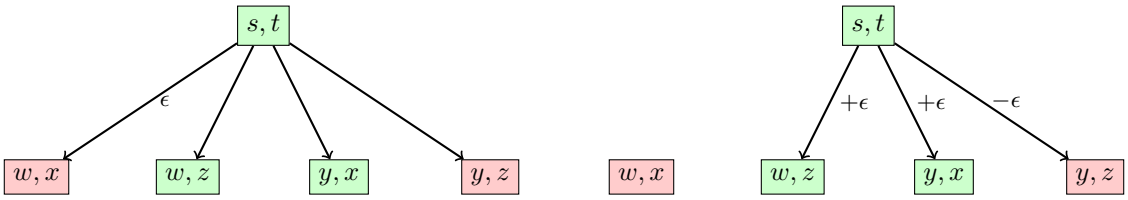
As we mentioned earlier, Vlasman introduced the concept of label conflict free policies in [58]. We refer to them as 1-conflict free policies. Roughly speaking, a policy has a 1-conflict if it matches two transitions whose target states have different labels, even though an alternative matching exists where the target states have the same label. Vlasman also showed that these conflicts can be removed by modifying the policy. Next, we will show that a 1-maximal optimal policy has no 1-conflicts.

In [58, Definition 4.1.1], Vlasman introduces the following notion.

Definition 6.44.

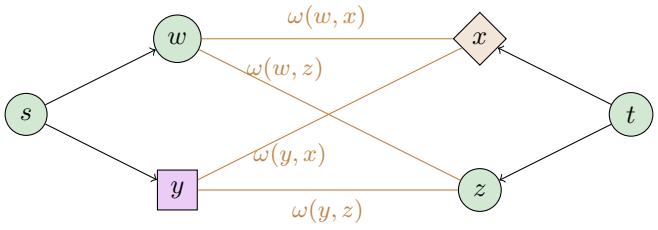
- *Let $(w, x), (y, z) \in S_1^2$. Then (w, x) and (y, z) are in* 1-conflict *if $(w, z) \notin S_1^2$ or $(x, y) \notin S_1^2$.*
- *Let $\omega \in \mathcal{D}(S \times S)$. Then ω is* 1-conflict free *if for all $(w, x), (y, z) \in \operatorname{support}(\omega) \cap S_1^2$, (w, x) and (y, z) are not in 1-conflict.*
- *Let $P \in \mathcal{P}$. Then P is* 1-conflict free *if for all $(s, t) \in S_?^2$, $P(s, t)$ is 1-conflict free.*

As shown by Vlasman [58, Theorem 4.1.1], these conflicts can be removed by modifying the policy, as illustrated next. Consider the left coupling $P(s, t)$ with $\epsilon = P(s, t)(w, x)$, where (w, x) and (y, z) are in 1-conflict. The right coupling, on the other hand, represents the modification of $P(s, t)$ that eliminates the 1-conflict.



The next proposition explains how to modify the matchings of the target states for a state pair (s, t) that has a 1-conflict, in order to make the state pair 1-conflict free. It also establishes properties related to the distances of these target states for the state pair (s, t).

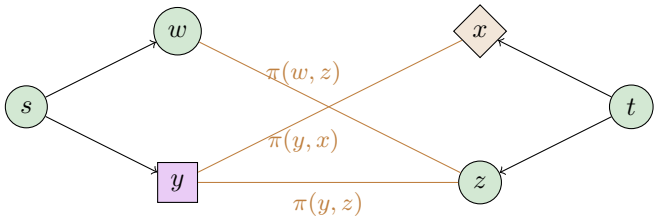
Proposition 6.45. *Let $P \in \mathcal{P}_{\text{opt}}$, $(s, t) \in S_?^2$, and $\omega = P(s, t)$. Assume that w, x, y, $z \in S$ with $w \neq y$ and $x \neq z$.*



(a) If $\omega(y,z) \geq \omega(w,x)$ *then for* π *defined by*

$$\pi(u,v) = \begin{cases} 0 & \text{if } (u,v) = (w,x) \\ \omega(w,z) + \omega(w,x) & \text{if } (u,v) = (w,z) \\ \omega(y,x) + \omega(w,x) & \text{if } (u,v) = (y,x) \\ \omega(y,z) - \omega(w,x) & \text{if } (v,v) = (y,z) \\ \omega(u,v) & \text{otherwise} \end{cases}$$

we have that $\pi \in \Omega(\tau(s), \tau(t))$.



(b) If furthermore $\delta_1(w,z) + \delta_1(y,x) \leq \delta_1(w,x) + \delta_1(y,z)$ *then for* Q *defined by*

$$Q(u,v) = \begin{cases} \pi & \text{if } (u,v) = (s,t) \\ P(u,v) & \text{otherwise} \end{cases}$$

we have that $Q \in \mathcal{P}_{\text{opt}}$ *and*

(c) if $\omega(w,x) > 0$ *then* $\delta_1(w,z) + \delta_1(y,x) = \delta_1(w,x) + \delta_1(y,z)$.

Proof. Let $P \in \mathcal{P}_{\text{opt}}$, $(s,t) \in S_?^2$, and $\omega = P(s,t)$. Assume that w, x, y, $z \in S$ with $w \neq y$ and $x \neq z$.

(a) Assume that $\omega(y,z) \geq \omega(w,x)$. To conclude that $\pi \in \Omega(\tau(s), \tau(t))$, we first prove that for all $u \in S$, $\pi(u,S) = \tau(s)(u)$. Let $u \in S$. We distinguish three cases.

– If $u = w$ then

$$\begin{aligned} \pi(w,S) &= \pi(w,x) + \pi(w,z) + \sum_{v \in S \setminus \{x,z\}} \pi(w,v) \\ &= 0 + (\omega(w,z) + \omega(w,x)) + \sum_{v \in S \setminus \{x,z\}} \omega(w,v) \\ &= \sum_{v \in S} \omega(w,v) \\ &= \omega(w,S) \\ &= \tau(s)(w) \qquad [\omega \in \Omega(\tau(s), \tau(t))] \end{aligned}$$

– If $u = y$ then

$$\begin{aligned} \pi(y,S) &= \pi(y,x) + \pi(y,z) + \sum_{v \in S \setminus \{x,z\}} \pi(y,v) \\ &= (\omega(y,x) + \omega(w,x)) + (\omega(y,z) - \omega(w,x)) + \sum_{v \in S \setminus \{x,z\}} \omega(y,v) \\ &= \omega(y,x) + \omega(y,z) + \sum_{v \in S \setminus \{x,z\}} \omega(y,v) \\ &= \sum_{v \in S} \omega(y,v) \\ &= \omega(y,S) \\ &= \tau(s)(y) \qquad [\omega \in \Omega(\tau(s), \tau(t))] \end{aligned}$$

– If $u \neq w$ and $u \neq y$ then

$$\begin{aligned}\pi(u,S) &= \sum_{v\in S} \pi(u,v)\\ &= \sum_{v\in S} \omega(u,v)\\ &= \omega(u,S)\\ &= \tau(s)(u) \qquad [\omega \in \Omega(\tau(s),\tau(t))]\end{aligned}$$

Second,we prove that for all $v \in S$, $\pi(S,v) = \tau(t)(v)$. Let $v \in S$. We distinguish three cases.

– If $v = x$ then

$$\begin{aligned}\pi(S,x) &= \pi(w,x) + \pi(y,x) + \sum_{u\in S\setminus\{w,y\}} \pi(u,x)\\ &= 0 + (\omega(y,x) + \omega(w,x)) + \sum_{u\in S\setminus\{w,y\}} \omega(u,x)\\ &= \sum_{u\in S} \omega(u,x)\\ &= \omega(S,x)\\ &= \tau(t)(x) \qquad [\omega \in \Omega(\tau(s),\tau(t))]\end{aligned}$$

– If $v = z$ then

$$\begin{aligned}\pi(S,z) &= \pi(w,z) + \pi(y,z) + \sum_{u\in S\setminus\{w,y\}} \pi(u,z)\\ &= (\omega(w,z) + \omega(w,x)) + (\omega(y,z) - \omega(w,x)) + \sum_{u\in S\setminus\{w,y\}} \omega(u,z)\\ &= \omega(w,z) + \omega(y,z) + \sum_{u\in S\setminus\{w,y\}} \omega(u,z)\\ &= \sum_{u\in S} \omega(u,z)\\ &= \omega(S,z)\\ &= \tau(t)(z) \qquad [\omega \in \Omega(\tau(s),\tau(t))]\end{aligned}$$

– If $v \neq x$ and $v \neq z$ then

$$\begin{aligned}\pi(S,v) &= \sum_{u\in S} \pi(u,v)\\ &= \sum_{u\in S} \omega(u,v)\\ &= \omega(S,v)\\ &= \tau(t)(v) \qquad [\omega \in \Omega(\tau(s),\tau(t))]\end{aligned}$$

(b) From (a) we can conclude that $Q \in \mathcal{P}$. Assume that

$$\delta_1(w,z) + \delta_1(y,x) \leq \delta_1(w,x) + \delta_1(y,z) \tag{6.8}$$

Since P is optimal, we have that $\delta_{1P} = \delta_1$. To prove that Q is an optimal policy, by Theorem 4.6 it suffices to show that $\delta_{1Q} \sqsubseteq \delta_{1P}$. Since δ_{1Q} is the least fixed point of Δ_{1Q}, and by Theorem 2.7(a), we can conclude that δ_{1Q} is also the least pre-fixed point of Δ_{1Q}. Therefore, to prove $\delta_{1Q} \sqsubseteq \delta_{1P}$, it suffices to show that δ_{1P} is a pre-fixed point of Δ_{1Q}, that is, $\Delta_{1Q}(\delta_{1P}) \sqsubseteq \delta_{1P}$. To prove that for all $u, v \in S$, $\Delta_{1Q}(\delta_{1P})(u,v) \leq \delta_{1P}(u,v)$, we consider the following cases.

– If $(u, v) \in S_0^2$ then

$$\Delta_{1Q}(\delta_{1P})(u, v) = 0 = \Delta_{1P}(\delta_{1P})(u, v) = \delta_{1P}(u, v).$$

– If $(u, v) \in S_1^2$ then

$$\Delta_{1Q}(\delta_{1P})(u, v) = 1 = \Delta_{1P}(\delta_{1P})(u, v) = \delta_{1P}(u, v).$$

– Let $(u, v) \in S_?^2$. We distinguish two cases.

∗ If $(u, v) \neq (s, t)$ then

$$\begin{aligned}\Delta_{1Q}(\delta_{1P})(u, v) &= Q(u, v) \cdot \delta_{1P} \\ &= P(u, v) \cdot \delta_{1P} \\ &= \Delta_{1P}(\delta_{1P})(u, v) \\ &= \delta_{1P}(u, v).\end{aligned}$$

∗ Assume $(u, v) = (s, t)$. We observe that π behaves differently from ω for (w, x), (w, z), (y, x), and (y, z). Since

$$\begin{aligned}&(\omega - \pi) \cdot \delta_{1P} \\ &= (\omega(w, x) - \pi(w, x))\, \delta_{1P}(w, x) + (\omega(w, z) - \pi(w, z))\, \delta_{1P}(w, z) \\ &+ (\omega(y, x) - \pi(y, x))\, \delta_{1P}(y, x) + (\omega(y, z) - \pi(y, z))\, \delta_{1P}(y, z) \\ &= (\omega(w, x) - 0)\, \delta_{1P}(w, x) + (\omega(w, z) - (\omega(w, z) + \omega(w, x)))\, \delta_{1P}(w, z) \\ &+ (\omega(y, x) - (\omega(y, x) + \omega(w, x)))\, \delta_{1P}(y, x) \\ &+ (\omega(y, z) - (\omega(y, z) - \omega(w, x)))\, \delta_{1P}(y, z) \\ &= \omega(w, x)\, \delta_{1P}(w, x) - \omega(w, x)\, \delta_{1P}(w, z) - \omega(w, x)\, \delta_{1P}(y, x) \\ &+ \omega(w, x)\, \delta_{1P}(y, z) \\ &= \omega(w, x)\, (\delta_{1P}(w, x) - \delta_{1P}(w, z) - \delta_{1P}(y, x) + \delta_{1P}(y, z)) \\ &= \omega(w, x)\, (\delta_1(w, x) - \delta_1(w, z) - \delta_1(y, x) + \delta_1(y, z)) \\ &\qquad [P \text{ is optimal and, hence, } \delta_{1P} = \delta_1] \\ &\geq 0 \qquad [(6.8)]\end{aligned}$$

we have that

$$\begin{aligned}\Delta_{1Q}(\delta_{1P})(s, t) &= Q(s, t) \cdot \delta_{1P} \\ &= \pi \cdot \delta_{1P} \\ &\leq \omega \cdot \delta_{1P} \\ &= P(s, t) \cdot \delta_{1P} \\ &= \Delta_{1P}(\delta_{1P})(s, t) \\ &= \delta_{1P}(s, t).\end{aligned}$$

Hence, Q is an optimal policy.

(c) Assume that $\omega(w, x) > 0$. Because

$$\begin{aligned}&\omega(w, x)\, (\delta_1(w, x) - \delta_1(w, z) - \delta_1(y, x) + \delta_1(y, z)) \\ &= (\omega - \pi) \cdot \delta_1 \qquad [\text{see above}] \\ &= \omega \cdot \delta_{1P} - \pi \cdot \delta_{1Q} \qquad [P \text{ and } Q \text{ are optimal}] \\ &= P(s, t) \cdot \delta_{1P} - Q(s, t) \cdot \delta_{1Q} \\ &= \Delta_{1P}(\delta_{1P})(s, t) - \Delta_{1Q}(\delta_{1Q})(s, t) \\ &= \delta_{1P}(s, t) - \delta_{1Q}(s, t) \\ &= \delta_1(s, t) - \delta_1(s, t) \qquad [P \text{ and } Q \text{ are optimal}] \\ &= 0\end{aligned}$$

we can conclude that $\delta_1(w, z) + \delta_1(y, x) = \delta_1(w, x) + \delta_1(y, z)$.

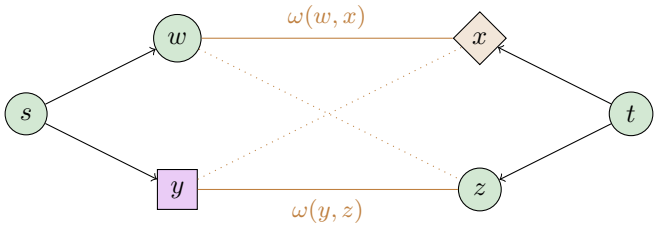
□

The following proposition shows that a 1-maximal optimal policy has no 1-conflicts.

Proposition 6.46. *For all $P \in \mathcal{P}_{\text{opt}}$, if P is 1-maximal then it is 1-conflict free.*

Proof. Let $P \in \mathcal{P}_{\text{opt}}$. We prove the contrapositive, that is, if P has a 1-conflict then it is not 1-maximal. This is shown by constructing $Q \in \mathcal{P}_{\text{opt}}$ with $\lambda_{1P} \sqsubset \lambda_{1Q}$ and, hence, P is not 1-maximal.

Assume that P has a 1-conflict. Hence, there exists $(s,t) \in S_?^2$ such that $P(s,t)$ has a 1-conflict. This implies that there exists $(w,x),(y,z) \in \text{support}(P(s,t)) \cap S_1^2$ such that (w,x) and (y,z) are in 1-conflict. Therefore, either $(w,z) \notin S_1^2$ or $(x,y) \notin S_1^2$. Without loss of generality, assume $(w,z) \notin S_1^2$. Since $\ell(w) \neq \ell(x)$ and $\ell(y) \neq \ell(z)$ and $\ell(w) = \ell(z)$, we can conclude that $\ell(w) \neq \ell(y)$ and $\ell(x) \neq \ell(z)$ and, hence, $w \neq y$ and $x \neq z$. We will need this later in the proof to apply Proposition 6.45. Let $\omega = P(s,t)$.



Without loss of generality, assume $\omega(y,z) \geq \omega(w,x)$. Since $(w,x) \in \text{support}(\omega)$, we have that $\omega(w,x) > 0$ and, therefore, we can conclude from Proposition 6.45(c) that $\delta_1(w,z) + \delta_1(y,x) = \delta_1(w,x) + \delta_1(y,z)$. Let Q be the optimal policy defined as in Proposition 6.45. It remains to show that $\lambda_{1P} \sqsubset \lambda_{1Q}$. Therefore, we prove that $\lambda_{1P} \sqsubseteq \lambda_{1Q}$ and $\lambda_{1P} \neq \lambda_{1Q}$. Since λ_{1P} is the least pre-fixed point of Λ_{1P}, it suffices to show that λ_{1Q} is a pre-fixed point of Λ_{1P}, that is, $\Lambda_{1P}(\lambda_{1Q}) \sqsubseteq \lambda_{1Q}$. Therefore, it suffices to show that for all $(u,v) \in D_1$, $\Lambda_{1P}(\lambda_{1Q})(u,v) \leq \lambda_{1Q}(u,v)$. We distinguish two cases.

- If $(u,v) \in S_1^2$ then
$$\Lambda_{1P}(\lambda_{1Q})(u,v) = 0 = \Lambda_{1Q}(\lambda_{1Q})(u,v) = \lambda_{1Q}(u,v).$$
- Otherwise, we have that $(u,v) \in D_1 \setminus S_1^2$. We distinguish two cases.
  - If $(u,v) \neq (s,t)$ then
$$\begin{aligned}\Lambda_{1P}(\lambda_{1Q})(u,v) &= P(u,v) \cdot (\delta_1 + \lambda_{1Q}) \\ &= Q(u,v) \cdot (\delta_1 + \lambda_{1Q}) \\ &= \Lambda_{1Q}(\lambda_{1Q})(u,v) \\ &= \lambda_{1Q}(u,v).\end{aligned}$$
  - Assume that $(u,v) = (s,t)$. Since, by Proposition 6.45(c), $\delta_1(w,z) + \delta_1(y,x) = \delta_1(w,x) + \delta_1(y,z)$ and $(w,x), (y,z) \in S_1^2$ and, therefore, $\delta_1(w,x) = 1$ and $\delta_1(y,z) = 1$, we can conclude that $\delta_1(w,z) = 1$ and $\delta_1(y,x) = 1$.

Note that π behaves differently from ω for $X = \{(w,z),(y,x),(y,z),(w,x)\}$. We have that

$$\begin{aligned}
&\pi(w,z)\,(\delta_1(w,z)+\lambda_{1Q}(w,z))+\pi(y,x)\,(\delta_1(y,x)+\lambda_{1Q}(y,x))\\
&+\pi(y,z)\,(\delta_1(y,z)+\lambda_{1Q}(y,z))+\pi(w,x)\,(\delta_1(w,x)+\lambda_{1Q}(w,x))\\
=&\pi(w,z)\,(\delta_1(w,z)+\lambda_{1Q}(w,z))+\pi(y,x)\,(\delta_1(y,x)+\lambda_{1Q}(y,x))+\pi(y,z)+\pi(w,x)\\
&\qquad[(w,x),(y,z)\in S_1^2]\\
=&\pi(w,z)\,(1+\lambda_{1Q}(w,z))+\pi(y,x)\,(1+\lambda_{1Q}(y,x))+\pi(y,z)+\pi(w,x)\\
&\qquad[\delta_1(w,z)=1 \text{ and } \delta_1(y,x)=1]\\
=&(\omega(w,z)+\omega(w,x))\,(1+\lambda_{1Q}(w,z))+(\omega(y,x)+\omega(w,x))\,(1+\lambda_{1Q}(y,x))+\\
&(\omega(y,z)-\omega(w,x))+0\\
=&\omega(w,z)\,(1+\lambda_{1Q}(w,z))+\omega(y,x)\,(1+\lambda_{1Q}(y,x))+\omega(y,z)+\\
&\omega(w,x)\,(1+\lambda_{1Q}(w,z)+1+\lambda_{1Q}(y,x)-1)\\
>&\omega(w,z)\,(1+\lambda_{1Q}(w,z))+\omega(y,x)\,(1+\lambda_{1Q}(y,x))+\omega(y,z)+\omega(w,x)\\
&\qquad[\lambda_{1Q}(w,z)>0 \text{ since } (w,z)\notin S_1^2]\\
=&\omega(w,z)\,(\delta_1(w,z)+\lambda_{1Q}(w,z))+\omega(y,x)\,(\delta_1(y,x)+\lambda_{1Q}(y,x))+\omega(y,z)+\omega(w,x)\\
&\qquad[\delta_1(w,z)=1 \text{ and } \delta_1(y,x)=1]\\
=&\omega(w,z)\,(\delta_1(w,z)+\lambda_{1Q}(w,z))+\omega(y,x)\,(\delta_1(y,x)+\lambda_{1Q}(y,x))\\
+&\omega(y,z)\,(\delta_1(y,z)+\lambda_{1Q}(y,z))+\omega(w,x)\,(\delta_1(w,x)+\lambda_{1Q}(w,x))\\
&\qquad[(y,z),(w,x)\in S_1^2]
\end{aligned}$$

From the above we can deduce that

$$\sum_{(u,v)\in X}\omega(u,v)\,(\delta_1(u,v)+\lambda_{1Q}(u,v))<\sum_{(u,v)\in X}\pi(u,v)\,(\delta_1(u,v)+\lambda_{1Q}(u,v)) \tag{6.9}$$

Moreover, we have that

$$\begin{aligned}
&\Lambda_{1P}(\lambda_{1Q})(s,t)\\
&=P(s,t)\cdot(\delta_1+\lambda_{1Q})\\
&=\sum_{(u,v)\in X}P(s,t)(u,v)\,(\delta_1(u,v)+\lambda_{1Q}(u,v))\\
&+\sum_{(u,v)\in D_1\setminus X}P(s,t)(u,v)\,(\delta_1(u,v)+\lambda_{1Q}(u,v))\\
&=\sum_{(u,v)\in X}\omega(u,v)\,(\delta_1(u,v)+\lambda_{1Q}(u,v))+\sum_{(u,v)\in D_1\setminus X}\omega(u,v)\,(\delta_1(u,v)+\lambda_{1Q}(u,v))\\
&<\sum_{(u,v)\in X}\pi(u,v)\,(\delta_1(u,v)+\lambda_{1Q}(u,v))+\sum_{(u,v)\in D_1\setminus X}\pi(u,v)\,(\delta_1(u,v)+\lambda_{1Q}(u,v))\\
&\qquad[(6.8)]\\
&=\sum_{(u,v)\in X}Q(s,t)(u,v)\,(\delta_1(u,v)+\lambda_{1Q}(u,v))\\
&+\sum_{(u,v)\in D_1\setminus X}Q(s,t)(u,v)\,(\delta_1(u,v)+\lambda_{1Q}(u,v))\\
&=Q(s,t)\cdot(\delta_1+\lambda_{1Q})\\
&=\Lambda_{1Q}(\lambda_{1Q})(s,t)\\
&=\lambda_{1Q}(s,t).
\end{aligned} \tag{6.10}$$

Since

$$\begin{aligned}\lambda_{1P}(s,t) &= \Lambda_{1P}(\lambda_{1P})(s,t) \\ &\leq \Lambda_{1P}(\lambda_{1Q})(s,t) \qquad [\lambda_{1P} \sqsubseteq \lambda_{1Q} \text{ and } \Lambda_{1P} \text{ is monotone (Proposition 5.3)}] \\ &< \lambda_{1Q}(s,t) \qquad [(6.9)]\end{aligned}$$

we can conclude that $\lambda_{1P} \neq \lambda_{1Q}$. □

As we have shown in the proof of Proposition 6.45, we can remove 1-conflicts. As we will illustrate in the next example, this specific way of removing 1-conflicts may not maintain that a policy is a vertex policy.

Example 6.47. *Consider the following LMC.*

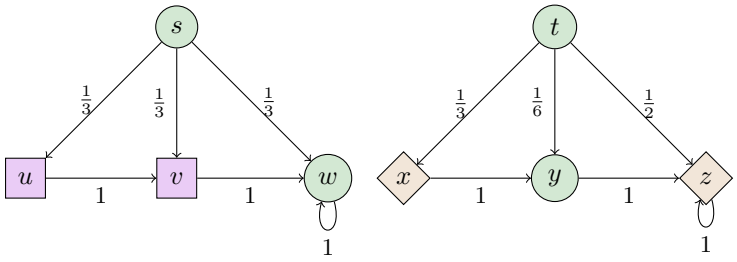
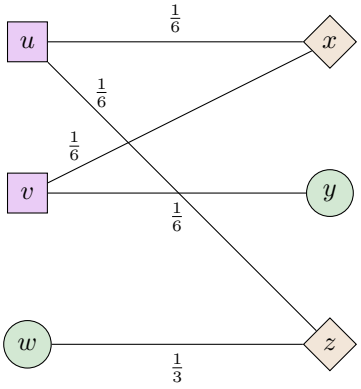
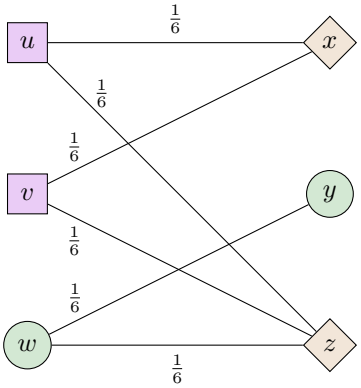


Figure 6.3: An LMC

We leave it to the reader to verify that all states are at a distance one of each other. Consider a vertex policy P with $P(s,t)$ depicted as follows.



Note that (v,y) and (w,z) are in 1-conflict. Then $Q(s,t)$ as constructed in the proof of Proposition 6.46 is represented as follows.



Since the support graph of $Q(s,t)$ contains a cycle, by Proposition 3.5 we can conclude that $Q(s,t) \notin V(\Omega(\tau(s),\tau(t)))$ and, hence, Q is not a vertex policy.

In this chapter, we developed an algorithm (Algorithm 1) that, starting from an optimal vertex policy, computes a 1-maximal optimal vertex policy. Additionally, we proved an exponential lower bound for this algorithm. Finally, we proved that a 1-maximal optimal policy avoids any 1-conflicts. In the following chapter, we will develop an algorithm that, starting from a 1-maximal optimal vertex policy, computes a 0-minimal 1-maximal optimal vertex policy.

7 0-Minimal Policies

As we mentioned earlier, among all the optimal 1-maximal policies, we are also interested in expected lengths that reach 0-pairs. We refer to an optimal 1-maximal policy that minimizes the expected length to a 0-pair as a 0-minimal policy. In this chapter, we will demonstrate that a policy that minimizes the expected length to 0-pairs is simpler and more desirable. We will begin by formulating the 0-minimal policies and presenting an algorithm that, starting from a 1-maximal optimal vertex policy computed using the algorithm from the previous chapter, computes a 0-minimal 1-maximal optimal vertex policy. Additionally, we will prove the correctness and termination of this algorithm.

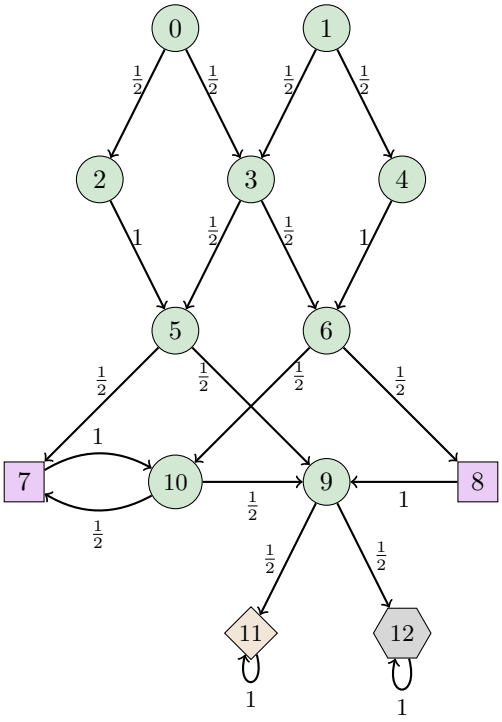


Figure 7.1: An LMC

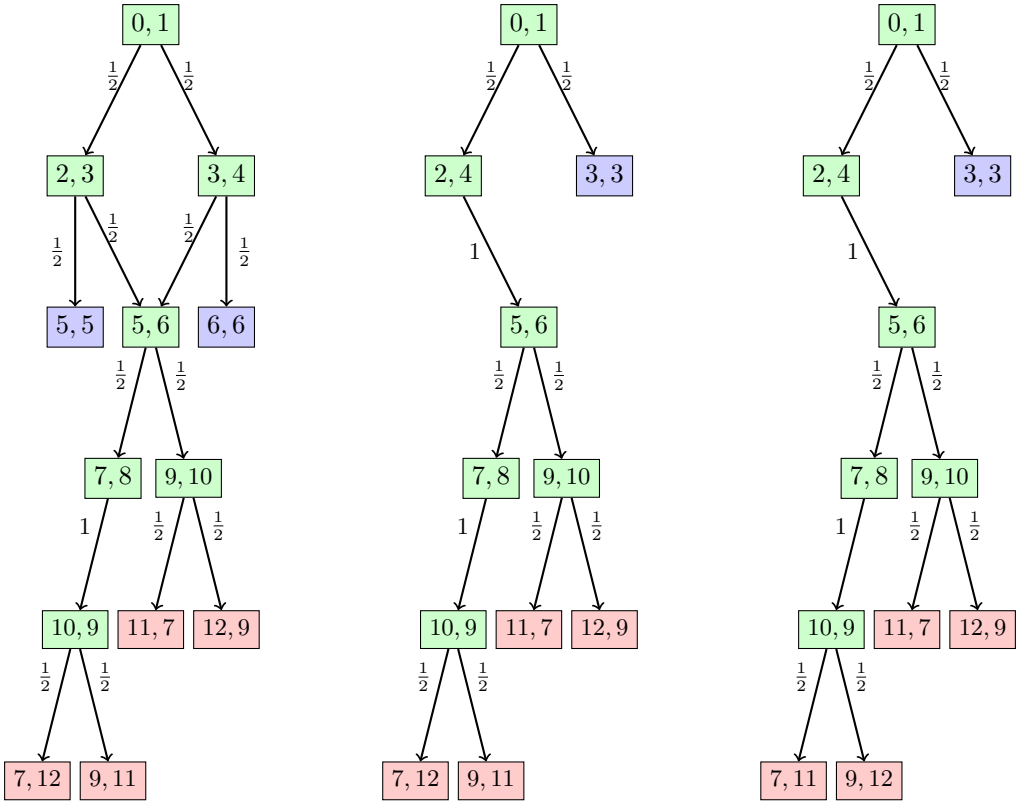


Figure 7.2: Three optimal policies for states 0 and 1 in the LMC depicted in Figure 7.1: one vertex 1-maximal policy, and two vertex 1-maximal 0-minimal policies

For example, consider the policies depicted in Figure 7.2. All of the policies are optimal, as each reaches 1-pairs with a probability of $\frac{1}{2}$, which represents the distance between states 0 and 1. They are also 1-maximal as the expected length to 1-pairs is 2.5. However, The expected length to 0-pairs in the left policy is 1 and 0.5 in the right policy. As illustrated in Figure 7.2, an optimal 1-maximal policy that minimizes the expected length of paths to 0-pairs is simpler and, hence, more desirable. It also shows that optimal 1-maximal 0-minimal policies are not unique. This minimal expected length is defined as follows.

Definition 7.1. *The function* $\lambda_0 : D_0 \to [0, \infty)$ *is defined by*

$$\lambda_0(s,t) = \inf_{P \in \mathcal{P}_{\mathrm{opt}}^{\max}} \lambda_{0P}(s,t).$$

A 1-maximal optimal policy that matches that minimal expected length is called 0-minimal.

Definition 7.2. *A policy* $P \in \mathcal{P}_{\mathrm{opt}}^{\max}$ *is* 0-minimal *if* $\lambda_{0P} = \lambda_0$.

To avoid clutter, we denote the set of optimal 1-maximal couplings $\{\, \omega \in \Omega(\tau(s), \tau(t)) \mid \delta_1(s,t) = \omega \cdot \delta_1 \wedge \lambda_1(s,t) = \omega \cdot (\delta_1 + \lambda_1) \,\}$ by $\Omega_{\mathrm{opt}}^{\max}(\tau(s), \tau(t))$ and the set of optimal 1-maximal vertex couplings $\{\, \omega \in V(\Omega(\tau(s), \tau(t))) \mid \delta_1(s,t) = \omega \cdot \delta_1 \wedge \lambda_1(s,t) = \omega \cdot (\delta_1 + \lambda_1) \}$ by $V(\Omega_{\mathrm{opt}}^{\max}(\tau(s), \tau(t)))$.

In the next proposition, we provide an alternative characterization of 0-minimal that we use in several of our proofs.

Proposition 7.3. *For all* $P \in \mathcal{P}_{\mathrm{opt}}^{\max}$*,* P *is 0-minimal if and only if* $\lambda_0(s,t) = P(s,t) \cdot (\delta_0 + \lambda_0)$ *for all* $(s,t) \in D_0 \setminus S_0^2$.

Proof. Let $P \in \mathcal{P}_{\mathrm{opt}}^{\max}$. We prove two implications. Assume that P is 0-minimal. Let $(s,t) \in D_0 \setminus S_0^2$. Then

$$\begin{aligned}
\lambda_0(s,t) &= \lambda_{0P}(s,t) && [P \text{ is 0-minimal}] \\
&= \Lambda_{0P}(\lambda_{0P})(s,t) \\
&= P(s,t) \cdot (\delta_0 + \lambda_{0P}) \\
&= P(s,t) \cdot (\delta_0 + \lambda_0) && [P \text{ is 0-minimal}]
\end{aligned}$$

To prove the other implication, assume that $\lambda_0(s,t) = P(s,t) \cdot (\delta_0 + \lambda_0)$ for all $(s,t) \in D_0 \setminus S_0^2$. Since λ_{0P} is the unique fixed point of Λ_{0P} (Proposition 5.8), it suffices to show that λ_0 is a fixed point of Λ_{0P}. We distinguish the following cases.

- If $(s,t) \in S_0^2$ then

$$\begin{aligned}\Lambda_{0P}(\lambda_0)(s,t) &= 0 \\ &= \inf_{Q \in \mathcal{P}_{\mathrm{opt}}^{\max}} \Lambda_{0Q}(\lambda_{0Q})(s,t) \\ &= \inf_{Q \in \mathcal{P}_{\mathrm{opt}}^{\max}} \lambda_{0Q}(s,t) \\ &= \lambda_0(s,t)\end{aligned}$$

- If $(s,t) \in D_0 \setminus S_0^2$ then

$$\begin{aligned}\Lambda_{0P}(\lambda_0)(s,t) &= P(s,t) \cdot (\delta_0 + \lambda_0) \\ &= \lambda_0(s,t) \qquad \text{[by assumption]}\end{aligned}$$

□

The next proposition proves that Λ_{0P}, as defined in Chapter 5, attains its minimum at a vertex.

Proposition 7.4. *For all $l \in D_0 \to [0,\infty)$ and $(s,t) \in D_0$,*

$$\inf_{\omega \in \Omega_{\mathrm{opt}}^{\max}(\tau(s),\tau(t))} \omega \cdot (\delta_0 + l) = \min_{\omega \in V(\Omega_{\mathrm{opt}}^{\max}(\tau(s),\tau(t)))} \omega \cdot (\delta_0 + l).$$

Proof. Consider the closed convex polytope M and the linear function $g : M \to [0,\infty)$ introduced in the proof of Proposition 6.4. Let $n = \sup_{\omega \in M} g(\omega)$. Then

$$n = \sup_{\omega \in \Omega_{\mathrm{opt}}(\tau(s),\tau(t))} \omega \cdot (\delta_1 + \lambda_1) = \bar{\Lambda}_1(\lambda_1)(s,t) = \lambda_1(s,t). \tag{7.1}$$

Let $N = \{\, \omega \in M \mid g(\omega) = n \,\}$. By Proposition 4.23(b), N is a closed convex polytope and

$$\begin{aligned}V(N) &= V(M) \cap N \qquad \text{[Proposition 4.23(b)]} \\ &= V(X) \cap M \cap N \qquad \text{[Proposition 6.4]}\end{aligned}$$

Let $l \in D_0 \to [0,\infty)$. Let the function $h : N \to [0,\infty)$ be defined by

$$h(\omega) = \omega \cdot (\delta_0 + l).$$

Next, we show that h is linear. Let $\omega, \pi \in N$ and $q \in (0,1)$. We have that

$$\begin{aligned}h(q\,\omega + (1-q)\,\pi) &= (q\,\omega + (1-q)\,\pi) \cdot (\delta_0 + l) \\ &= (q\,\omega \cdot (\delta_0 + l)) + ((1-q)\pi \cdot (\delta_0 + l)) \\ &= q\,h(\omega) + (1-q)\,h(\pi).\end{aligned}$$

Because a linear function on a convex polytope attains its minimum at a vertex, we can conclude that the linear function h attains its minimum at some $\omega \in V(N) = V(X) \cap M \cap N$. Since $\omega \in V(N)$ if and only if $\omega \in V(\Omega_{\mathrm{opt}}^{\max}(\tau(s),\tau(t)))$, the desired result follows. □

If λ_{0P} is a strict pre-fixed point of Λ_{0Q} for some policies P and Q, then the next proposition shows that λ_{0Q} is less than λ_{0P}.

Proposition 7.5. *For all $P, Q \in \mathcal{P}$, if $\Lambda_{0Q}(\lambda_{0P}) \sqsubset \lambda_{0P}$ then $\lambda_{0Q} \sqsubset \lambda_{0P}$.*

Proof. Let $P, Q \in \mathcal{P}$. We split the proof in two parts. First, we show that if $\Lambda_{0Q}(\lambda_{0P}) \sqsubseteq \lambda_{0P}$ then $\lambda_{0Q} \sqsubseteq \lambda_{0P}$. Assume that $\Lambda_{0Q}(\lambda_{0P}) \sqsubseteq \lambda_{0P}$. We first show that for all $n \in \mathbb{N}$, $\Lambda^n_{0Q}(\lambda_{0P}) \sqsubseteq \lambda_{0P}$ by induction on n. The base case, $n = 0$, is trivial. In the inductive case, $n > 0$, we have that

$$\begin{aligned}\Lambda^n_{0Q}(\lambda_{0P}) &= \Lambda_{0Q}(\Lambda^{n-1}_{0Q}(\lambda_{0P})) \\ &\sqsubseteq \Lambda_{0Q}(\lambda_{0P}) \qquad \text{[induction hypothesis and } \Lambda_{0Q} \text{ is monotone (Proposition 5.3)]} \\ &\sqsubseteq \lambda_{0P}. \qquad \text{[by assumption]}\end{aligned}$$

Because Λ^n_{P0} is contractive (Proposition 5.6) and Λ_{0P} is nonexpansive (Proposition 5.7), we can conclude from the above and Theorem 2.10(d) that

$$\lambda_{0Q} = \lim_{n\in\mathbb{N}} \Lambda^n_{0Q}(\lambda_{0P}) \sqsubseteq \lambda_{0P}.$$

It remains to show that $\Lambda_{0Q}(\lambda_{0P}) \neq \lambda_{0P}$ implies $\lambda_{0Q} \neq \lambda_{0P}$. We prove the contrapositive, that is, $\lambda_{0Q} = \lambda_{0P}$ implies $\lambda_{0P} = \Lambda_{0Q}(\lambda_{0P})$. That is, λ_{0Q} is a fixed point of Λ_{0Q}, which holds by definition. □

The following proposition proves that for every 1-maximal optimal policy P, there exists a 1-maximal optimal vertex policy Q such that λ_{0Q} is less than or equal to λ_{0P}.

Proposition 7.6. *For all $P \in \mathcal{P}^{\max}_{\text{opt}}$ there exists $Q \in \mathcal{V}^{\max}_{\text{opt}}$ such that $\lambda_{0Q} \sqsubseteq \lambda_{0P}$.*

Proof. Let $P \in \mathcal{P}^{\max}_{\text{opt}}$. Let $(s,t) \in S^2_?$ and, by Proposition 7.4, define

$$Q(s,t) \in \operatorname*{arg\,min}_{\omega \in V(\Omega^{\max}_{\text{opt}}(\tau(s),\tau(t)))} \omega \cdot (\delta_0 + \lambda_{0P}).$$

Since P is optimal, by Corollary 5.9, we have that $\lambda_{0P} \in D_0 \to [0,\infty)$. By Proposition 4.21, $Q \in \mathcal{V}_{\text{opt}}$. By the definition of $Q(s,t)$ we have that $\lambda_1(s,t) = Q(s,t)\cdot(\delta_1+\lambda_1)$ and, hence, by Proposition 6.3 we can conclude that $Q \in \mathcal{V}^{\max}_{\text{opt}}$. Next, we will show that $\Lambda_{0Q}(\lambda_{0P}) \sqsubseteq \lambda_{0P}$, that is, λ_{0P} is a pre-fixed point of Λ_{0Q}. Let $(s,t) \in D_0$. We distinguish the following cases.

- If $(s,t) \in S^2_0$ then
$$\Lambda_{0Q}(\lambda_{0P})(s,t) = 0 = \Lambda_{0P}(\lambda_{0P})(s,t) = \lambda_{0P}(s,t).$$
- If $(s,t) \in D_0 \setminus S^2_0$ then
$$\begin{aligned}\Lambda_{0Q}(\lambda_{0P})(s,t) &= Q(s,t) \cdot (\delta_0 + \lambda_{0P}) \\ &\leq P(s,t) \cdot (\delta_0 + \lambda_{0P}) \\ &= \Lambda_{0P}(\lambda_{0P})(s,t) \\ &= \lambda_{0P}(s,t).\end{aligned}$$

Since Λ_{0Q} is monotone (Proposition 5.3), Λ_{0Q} has a least fixed point, which is the least pre-fixed point, according to Theorem 2.7(a). By definition, λ_{0Q} is the least fixed point and, therefore, the least pre-fixed point of Λ_{0Q}. Since we have shown that λ_{0P} is a pre-fixed point of Λ_{0Q}, we can conclude that $\lambda_{0Q} \sqsubseteq \lambda_{0P}$. □

Corollary 7.7. *For all $(s,t) \in D_0$, $\lambda_0(s,t) = \min_{P\in\mathcal{V}^{\max}_{\text{opt}}} \lambda_{0P}(s,t)$.*

Proof. Let $(s,t) \in D_0$. By Definition 7.1 we have that $\lambda_0(s,t) = \inf_{P\in\mathcal{P}^{\max}_{\text{opt}}} \lambda_{0P}(s,t)$. We prove the following two inequalities.

- Since $\mathcal{V}^{\max}_{\text{opt}} \subseteq \mathcal{P}^{\max}_{\text{opt}}$, we can conclude that $\inf_{P\in\mathcal{P}^{\max}_{\text{opt}}} \lambda_{0P}(s,t) \leq \min_{Q\in\mathcal{V}^{\max}_{\text{opt}}} \lambda_{0P}(s,t)$.
- Towards a contradiction, assume that $\inf_{P\in\mathcal{P}^{\max}_{\text{opt}}} \lambda_{0P}(s,t) < \min_{Q\in\mathcal{V}^{\max}_{\text{opt}}} \lambda_{0Q}(s,t)$. Then, there exists a $P \in \mathcal{P}^{\max}_{\text{opt}}$ such that $\lambda_{0P}(s,t) < \min_{Q\in\mathcal{V}^{\max}_{\text{opt}}} \lambda_{0Q}(s,t)$. Therefore, $\lambda_{0P}(s,t) < \lambda_{0Q}(s,t)$ for all $Q \in \mathcal{V}^{\max}_{\text{opt}}$. This contradicts Proposition 7.6.

Hence, $\lambda_0(s,t) = \inf_{P\in\mathcal{P}^{\max}_{\text{opt}}} \lambda_{0P}(s,t) = \min_{P\in\mathcal{V}^{\max}_{\text{opt}}} \lambda_{0P}(s,t)$. □

Below we present a policy iteration algorithm that computes a 0-minimal 1-maximal optimal vertex policy from a 1-maximal optimal vertex policy. The following function serves as the foundation for this algorithm.

Definition 7.8. *The function* $\Lambda_0 : (D_0 \to [0,\infty)) \to (D_0 \to [0,\infty))$ *is defined by*

$$\Lambda_0(l)(s,t) = \begin{cases} 0 & \text{if } (s,t) \in S_0^2 \\ \inf\limits_{\omega \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \omega \cdot (\delta_0 + l) & \text{otherwise.} \end{cases}$$

In the next proposition, we establish that Λ_0 is a monotone function, which allows us to prove that it has a unique fixed point.

Proposition 7.9. *For all* $k, l \in D_0 \to [0,\infty]$*, if* $k \sqsubseteq l$ *then* $\Lambda_0(k) \sqsubseteq \Lambda_0(l)$.

Proof. Let $k, l \in D_0 \to [0,\infty]$. Assume $k \sqsubseteq l$. Let $(s,t) \in D_0$. It suffices to show that $\Lambda_0(k)(s,t) \leq \Lambda_0(l)(s,t)$. We distinguish the following cases.

- If $(s,t) \in S_0^2$ then
$$\Lambda_0(k)(s,t) = 0 = \Lambda_0(l)(s,t).$$
- Otherwise,
$$\begin{aligned} \Lambda_0(k)(s,t) &= \inf_{\omega \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \omega \cdot (\delta_0 + k) \\ &\leq \inf_{\omega \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \omega \cdot (\delta_0 + l) \qquad [k \sqsubseteq l] \\ &= \Lambda_0(l)(s,t). \end{aligned}$$

Hence, $\Lambda_0(k) \sqsubseteq \Lambda_0(l)$. □

The following proposition proves that if you have two couplings ω and π that are 1-maximal and optimal, then minimizing over ω first and maximizing over π gives the same result as maximizing over π and then minimizing over ω. In other words, the order of minimizing and maximizing does not matter when optimizing the expression $\omega \cdot (\delta_0 + k) - \pi \cdot (\delta_0 + l)$ for all $k, l \in D_0 \to [0,\infty)$. This result is based on the minimax theorem from the literature [20, Theorem 1(ii)].

Proposition 7.10. *For all* $k, l \in D_0 \to [0,\infty)$ *and* $(s,t) \in S_?^2$*,*

$$\begin{aligned} &\min_{\omega \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \max_{\pi \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \omega \cdot (\delta_0 + k) - \pi \cdot (\delta_0 + l) \\ &= \\ &\max_{\pi \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \min_{\omega \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \omega \cdot (\delta_0 + k) - \pi \cdot (\delta_0 + l). \end{aligned}$$

Proof. Let $k, l \in D_0 \to [0,\infty)$ and $(s,t) \in S_?^2$. To prove the equality, we apply a version of the minimax theorem. To be able to apply that theorem, we need to show that

1. the set $\Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))$ is convex,
2. the set $\Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))$ is compact, and
3. the function mapping $\omega, \pi \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))$ to $\omega \cdot (\delta_0 + k) - \pi \cdot (\delta_0 + l)$ is bilinear.

It remains to prove the above three properties.

1. This can already be found in the proof of Proposition 7.4.
2. As shown in the proof of Proposition 7.4, the set $\Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))$ is closed. Since S is a finite set, we can conclude that $S \times S \to [0,1]$ is compact. Because a closed subset of a compact set is compact, we can conclude that $\Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))$ is compact.

3. Let ω, π, $\rho \in \Omega_{\text{opt}}^{\max}(\tau(s), \tau(t))$ and $q \in (0, 1)$. Because

$$\begin{aligned}
&(q\,\omega + (1-q)\,\rho) \cdot (\delta_0 + k) - \pi \cdot (\delta_0 + l) \\
=&q\,\omega \cdot (\delta_0 + k) + (1-q)\,\rho \cdot (\delta_0 + k) - \pi \cdot (\delta_0 + l) \\
=&q\ (\omega \cdot (\delta_0 + k) - \pi \cdot (\delta_0 + l)) + (1-q)\ (\rho \cdot (\delta_0 + k) - \pi \cdot (\delta_0 + l))
\end{aligned}$$

and

$$\begin{aligned}
&\omega \cdot (\delta_0 + k) - (q\,\pi + (1-q)\,\rho) \cdot (\delta_0 + l) \\
=&\omega \cdot (\delta_0 + k) - (q\,\pi \cdot (\delta_0 + l) + (1-q)\,\rho \cdot (\delta_0 + l)) \\
=&q\ (\omega \cdot (\delta_0 + k) - \pi \cdot (\delta_0 + l)) + (1-q)\ (\omega \cdot (\delta_0 + k) - \rho \cdot (\delta_0 + l))
\end{aligned}$$

we can conclude that the function is bilinear.

□

In the next proposition, we establish that Λ_0 is a nonexpansive function, which we later use to prove some properties of symmetric policies.

Proposition 7.11. *Λ_0 is nonexpansive.*

Proof. Let $k, l \in D_0 \to [0, \infty)$, and $(s, t) \in D_0$. It suffices to show that $|\Lambda_0(k)(s,t) - \Lambda_0(l)(s,t)| \leq \|k - l\|$. Without loss of generality, assume $\Lambda_0(k)(s,t) \geq \Lambda_0(l)(s,t)$. We distinguish two cases.

- If $(s,t) \in S_0^2$ then we can conclude that

$$\Lambda_0(k)(s,t) - \Lambda_0(l)(s,t) = 0 \leq \|k - l\|.$$

- If $(s,t) \in D_0 \setminus S_0^2$ then we have that

$$\begin{aligned}
&\Lambda_0(k)(s,t) - \Lambda_0(l)(s,t) \\
&= \inf_{\omega \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \omega \cdot (\delta_0 + k) - \inf_{\pi \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \pi \cdot (\delta_0 + l) \\
&= \min_{\omega \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \omega \cdot (\delta_0 + k) - \min_{\pi \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \pi \cdot (\delta_0 + l) && \text{[Proposition 7.4]} \\
&= \min_{\omega \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \max_{\pi \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \omega \cdot (\delta_0 + k) - \pi \cdot (\delta_0 + l) \\
&= \max_{\pi \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \min_{\omega \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \omega \cdot (\delta_0 + k) - \pi \cdot (\delta_0 + l) && \text{[Proposition 7.10]} \\
&\leq \max_{\pi \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \pi \cdot (\delta_0 + k) - \pi \cdot (\delta_0 + l) \\
&= \max_{\pi \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \pi \cdot (k - l) \\
&= \max_{\pi \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \sum_{(u,v) \in D_0} \pi(u,v)\,(k(u,v) - l(u,v)) \\
&\leq \max_{\pi \in \Omega_{\text{opt}}^{\max}(\tau(s),\tau(t))} \sum_{(u,v) \in D_0} \pi(u,v)\,\|k - l\| \\
&= \|k - l\|.
\end{aligned}$$

□

The next proposition shows the relation between Λ_{0P} and Λ_0.

Proposition 7.12. *For all $l \in D_0 \to [0, \infty)$,*

(a) for all $P \in \mathcal{P}_{\text{opt}}^{\max}$, $\Lambda_0(l) \sqsubseteq \Lambda_{0P}(l)$ and

(b) there exists $P \in \mathcal{V}_{\text{opt}}^{\max}$ such that $\Lambda_0(l) = \Lambda_{0P}(l)$.

Proof. Let $l \in D_0 \to [0, \infty)$.

(a) Let $P \in \mathcal{P}^{\max}_{\text{opt}}$ and $(s,t) \in D_0 \setminus S^2_0$. We first show that $P(s,t) \in \Omega^{\max}_{\text{opt}}(\tau(s), \tau(t))$. Since $P \in \mathcal{P}^{\max}_{\text{opt}}$ it follows from Proposition 4.21 that $\delta_1(s,t) = P(s,t) \cdot \delta_1$. Furthermore, by Proposition 6.3, we have that $\lambda_1(s,t) = P(s,t) \cdot (\delta_1 + \lambda_1)$. Therefore, can conclude that

$$P(s,t) \in \Omega^{\max}_{\text{opt}}(\tau(s), \tau(t)) \tag{7.2}$$

Let $(s,t) \in D_0$. We distinguish two cases.

- If $(s,t) \in S^2_0$ then
$$\Lambda_0(l)(s,t) = 0 = \Lambda_{0P}(l)(s,t).$$
- If $(s,t) \in D_0 \setminus S^2_0$ then

$$\begin{aligned}\Lambda_0(l)(s,t) &= \inf_{\omega \in \Omega^{\max}_{\text{opt}}(\tau(s),\tau(t))} \omega \cdot (\delta_0 + l) \\ &\leq P(s,t) \cdot (\delta_0 + l) \qquad [(7.2)] \\ &= \Lambda_{0P}(l)(s,t).\end{aligned}$$

(b) Let $(s,t) \in S^2_?$ and, by Proposition 7.4, define

$$P(s,t) \in \operatorname*{arg\,min}_{\omega \in V(\Omega^{\max}_{\text{opt}}(\tau(s),\tau(t)))} \omega \cdot (\delta_0 + l).$$

By Proposition 4.21 and 6.3, $P \in \mathcal{V}^{\max}_{\text{opt}}$. Let $(s,t) \in D_0$. We distinguish two cases.

- If $(s,t) \in S^2_0$ then
$$\Lambda_0(l)(s,t) = 0 = \Lambda_{0P}(l)(s,t).$$
- If $(s,t) \in D_0 \setminus S^2_0$ then

$$\begin{aligned}\Lambda_0(l)(s,t) &= \min_{\omega \in V(\Omega^{\max}_{\text{opt}}(\tau(s),\tau(t)))} \omega \cdot (\delta_0 + l) \qquad [\text{Proposition 7.4}] \\ &= P(s,t) \cdot (\delta_0 + l) \\ &= \Lambda_{0P}(l)(s,t).\end{aligned}$$

□

The next proposition shows that λ_0 is a pre-fixed point of Λ_0.

Proposition 7.13. $\Lambda_0(\lambda_0) \sqsubseteq \lambda_0$.

Proof. Since for all $P \in \mathcal{P}^{\max}_{\text{opt}}$,

$$\begin{aligned}\Lambda_0(\lambda_0) &\sqsubseteq \Lambda_0(\lambda_{0P}) \qquad [\lambda_0 \sqsubseteq \lambda_{0P} \text{ and } \Lambda_0 \text{ is monotone (Proposition 7.9)}] \\ &\sqsubseteq \Lambda_{0P}(\lambda_{0P}) \qquad [\text{Proposition 6.12(a)}] \\ &= \lambda_{0P}.\end{aligned}$$

we can conclude that $\Lambda_0(\lambda_0) \sqsubseteq \lambda_0$. □

The next proposition shows the link between Λ_0 and λ_0.

Proposition 7.14. *For all $l \in D_0 \to [0, \infty)$,*

(a) if $l \sqsubseteq \Lambda_0(l)$ then $l \sqsubseteq \lambda_0$ and

(b) if $\Lambda_0(l) \sqsubseteq l$ then $\lambda_0 \sqsubseteq l$.

Proof. Let $l \in D_0 \to [0, \infty)$.

(a) Assume that $l \sqsubseteq \Lambda_0(l)$. Let $P \in \mathcal{P}_{\text{opt}}^{\max}$. We show that for all $n \in \mathbb{N}$, $l \sqsubseteq \Lambda_{0P}^n(\Lambda_0(l))$ by induction on n. The base case, $n = 0$, follows from the assumption $l \sqsubseteq \Lambda_0(l)$. In the inductive case, when $n > 0$, we have that

$$\begin{aligned} l &\sqsubseteq \Lambda_0(l) && \text{[by assumption]} \\ &\sqsubseteq \Lambda_{0P}(l) && \text{[Proposition 6.12(a)]} \\ &\sqsubseteq \Lambda_{0P}(\Lambda_{0P}^{n-1}(\Lambda_0(l))) && \text{[induction hypothesis and } \Lambda_{0P} \text{ is monotone (Proposition 5.3)]} \\ &= \Lambda_{0P}^n(\Lambda_0(l)) \end{aligned}$$

Because Λ_{P0}^n is contractive (Proposition 5.6) and Λ_{0P} is nonexpansive (Proposition 5.7), we can conclude from the above and Theorem 2.10(d) that $l \sqsubseteq \lambda_{0P}$. Hence, $l \sqsubseteq \lambda_0$.

(b) Assume that $\Lambda_0(l) \sqsubseteq l$. According to Proposition 6.12(b), $\Lambda_0(l) = \Lambda_{0P}(l)$ for some $P \in \mathcal{V}_{\text{opt}}^{\max}$. Next, we will show that $\Lambda_{0P}^n(l) \sqsubseteq l$ for all $n \in \mathbb{N}$ by induction on n. The base case, $n = 0$, is trivial. In the inductive case, when $n > 0$, we have that

$$\begin{aligned} \Lambda_{0P}^n(l) &= \Lambda_{0P}(\Lambda_{0P}^{n-1}(l)) \\ &\sqsubseteq \Lambda_{0P}(l) && \text{[induction hypothesis and } \Lambda_{0P} \text{ is monotone (Proposition 5.3)]} \\ &= \Lambda_0(l) && \text{[Proposition 6.12(b)]} \\ &\sqsubseteq l && \text{[by assumption]} \end{aligned}$$

From Proposition 5.7 and Proposition 5.5 and Theorem 2.10(d) we can conclude that

$$\lambda_0 \sqsubseteq \lambda_{0P} = \lim_{n \in \mathbb{N}} \Lambda_{0P}^n(l) \sqsubseteq l.$$

□

The key ingredient of the correctness proof of our algorithm is the following result.

Theorem 7.15. *λ_0 is the unique fixed point of Λ_0.*

Proof. First, we show that λ_0 is a fixed point of Λ_0. According to Proposition 7.13, we have that $\Lambda_0(\lambda_0) \sqsubseteq \lambda_0$. Since Λ_0 is monotone (Proposition 7.9), we can conclude that $\Lambda_0(\Lambda_0(\lambda_0)) \sqsubseteq \Lambda_0(\lambda_0)$. From Proposition 7.14(b) we can deduce that $\lambda_0 \sqsubseteq \Lambda_0(\lambda_0)$. Combined with Proposition 7.13 we get that $\Lambda_0(\lambda_0) = \lambda_0$, that is, λ_0 is a fixed point of Λ_0.

Next, we show λ_0 is the unique fixed point of Λ_0. Assume that l is a fixed point of Λ_0, that is, $\Lambda_0(l) = l$. Then $l \sqsubseteq \Lambda_0(l)$ and $\Lambda_0(l) \sqsubseteq l$. By Proposition 7.14(a) and (b), we have that $l \sqsubseteq \lambda_0$ and $\lambda_0 \sqsubseteq l$. Hence, $l = \lambda_0$. Therefore, λ_0 is the unique fixed point of Λ_0. □

Our algorithm that computes a 0-minimal 1-maximal vertex policy from a 1-maximal vertex policy (Algorithm 2) is very similar in structure to Algorithm 1. Instead of focusing on 1-pairs, we concentrate on 0-pairs. Furthermore, we maintain as a loop invariant that P is not only an optimal vertex policy but also that it is 1-maximal.

Algorithm 2 0-minimal 1-maximal optimal vertex policy

Input: 1-maximal optimal vertex policy P

compute λ_{0P}
while $\exists (s,t) \in D_0 \setminus S_0^2 : \Lambda_0(\lambda_{0P})(s,t) < \lambda_{0P}(s,t)$ **do**
  $\pi \leftarrow \underset{\omega \in V(\Omega(\tau(s),\tau(t))) \wedge \delta_1(s,t) = \omega \cdot \delta_1 \wedge \lambda_1(s,t) = \omega \cdot (\delta_1 + \lambda_1)}{\arg\min} \omega \cdot (\delta_0 + \lambda_{0P})$
  $P(s,t) \leftarrow \pi$
  compute λ_{0P}
end while

The following proposition proves that the above algorithm terminates.

Proposition 7.16. *The algorithm terminates.*

Proof. Towards a contradiction, assume that the algorithm does not terminate. Let us denote the value of the variable P in the i-th iteration by P_i. Then P_{i+1} is defined by

$$P_{i+1}(x,y) = \begin{cases} \pi & \text{if } (x,y) = (s,t) \\ P_i(x,y) & \text{otherwise,} \end{cases}$$

where

$$\pi \in \underset{\omega \in V(\Omega^{\max}_{\mathrm{opt}}(\tau(s),\tau(t)))}{\arg\min} \omega \cdot (\delta_0 + \lambda_{0P_i}).$$

By Proposition 4.21 and 6.3, $P_{i+1} \in \mathcal{V}^{\max}_{\mathrm{opt}}$. Next, we prove that $\Lambda_{0P_{i+1}}(\lambda_{0P_i}) \sqsubset \lambda_{0P_i}$. Let $(x,y) \in D_0$. We distinguish three cases.

- If $(x,y) \in S_0^2$ then
$$\Lambda_{0P_{i+1}}(\lambda_{0P_i})(x,y) = 0 = \Lambda_{0P_i}(\lambda_{0P_i})(x,y) = \lambda_{0P_i}(x,y).$$
- If $(x,y) = (s,t)$ then
$$\begin{aligned} \Lambda_{0P_{i+1}}(\lambda_{0P_i})(s,t) &= P_{i+1}(s,t) \cdot (\delta_0 + \lambda_{0P_i}) \\ &= \pi \cdot (\delta_0 + \lambda_{0P_i}) \\ &= \min_{\omega \in V(\Omega^{\max}_{\mathrm{opt}}(\tau(s),\tau(t)))} \omega \cdot (\delta_0 + \lambda_{0P_i}) \\ &= \inf_{\omega \in \Omega^{\max}_{\mathrm{opt}}(\tau(s),\tau(t))} \omega \cdot (\delta_0 + \lambda_{0P_i}) \qquad \text{[Proposition 7.4]} \\ &= \Lambda_0(\lambda_{0P_i})(s,t) \\ &< \lambda_{0P_i}(s,t). \qquad \text{[condition of the loop]} \end{aligned}$$
- Otherwise,
$$\begin{aligned} \Lambda_{0P_{i+1}}(\lambda_{0P_i})(x,y) &= P_{i+1}(x,y) \cdot (\delta_0 + \lambda_{0P_i}) \\ &= P_i(x,y) \cdot (\delta_0 + \lambda_{0P_i}) \\ &= \Lambda_{0P_i}(\lambda_{0P_i})(x,y) \\ &= \lambda_{0P_i}(x,y). \end{aligned}$$

From Proposition 6.5 we can conclude that $\lambda_{0P_{i+1}} \sqsubset \lambda_{0P_i}$. Since the set $\mathcal{V}^{\max}_{\mathrm{opt}}$ is finite and the algorithm does not terminate, $P_i = P_j$ for some $i < j$. This contradicts that $\lambda_{0P_j} \sqsubset \lambda_{0P_i}$. □

From the above corollary, we know that our algorithm terminates. The following proposition shows that, upon termination, it computes a 0-minimal 1-maximal vertex optimal policy.

Proposition 7.17. *At termination,* $\lambda_{0P} = \lambda_0$.

Proof. At termination, we have that $\Lambda_0(\lambda_{0P})(s,t) \geq \lambda_{0P}(s,t)$ for all $(s,t) \in D_0 \setminus S_0^2$. Since for all $(s,t) \in S_0^2$,

$$\Lambda_0(\lambda_{0P})(s,t) = 0 = \Lambda_{0P}(\lambda_{0P})(s,t) = \lambda_{0P}(s,t),$$

we have that $\Lambda_0(\lambda_{0P}) \sqsupseteq \lambda_{0P}$. By Proposition 6.12(a), $\Lambda_0(\lambda_{0P}) \sqsubseteq \Lambda_{0P}(\lambda_{0P}) = \lambda_{0P}$. Hence, $\Lambda_0(\lambda_{0P}) = \lambda_{0P}$, that is, λ_{0P} is a fixed point of Λ_0. Since λ_0 is the unique fixed point of Λ_0 (Theorem 6.15), we can deduce that $\lambda_{0P} = \lambda_0$. □

In this chapter, we developed an algorithm (Algorithm 2) that, starting from a 1-maximal optimal vertex policy, computes a 0-minimal 1-maximal optimal vertex policy. Recall the notion of a symmetric policy introduced in the introduction of this thesis. In the following chapter, we will explore symmetric policies and study their properties.

8 Symmetric Policies

Since probabilistic bisimilarity distances are symmetric—that is, the distance between s and t is equal to the distance between t and s for all states s and t—this raises the question of whether the distance between s and t can be explained in the same way as the distance between t and s. In this chapter, we will address this question by defining symmetric policies and exploring their properties. We call a policy symmetric if the policy at state pair s and t is the mirror image of the policy at state pair t and s. To define symmetric policies, we fix a total order $\prec$ on the set of states in an LMC, which enables us to transform any policy into a symmetric one. To reflect that one policy is the mirror image of another, we introduce twisted arrows in the graphical representations of policies. As we will see in the graphical representation of a symmetric policy, we only need to represent the state pairs (s, t) with $s \preceq t$, where an arrow from (s, t) to (u, v) with $u \succ v$ is depicted as a twisted arrow from (s, t) to (v, u). Consequently, once we have computed a 0-maximal 1-minimal optimal vertex policy, we can convert it into a symmetric 0-maximal 1-minimal optimal vertex policy. Symmetric policies, in general, are simpler and, hence, preferable.

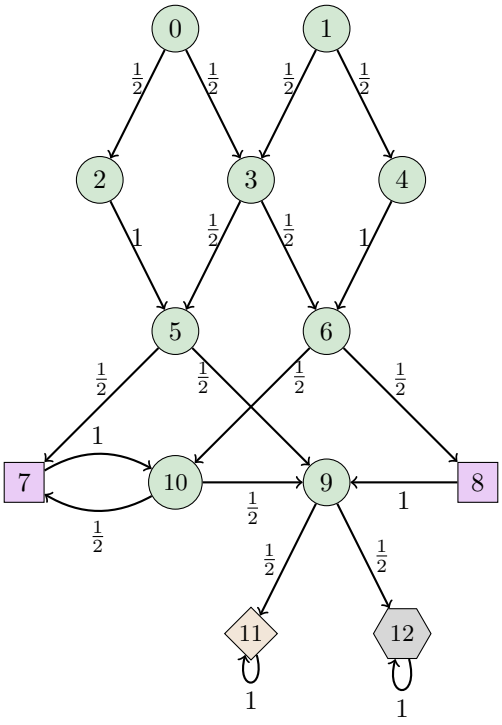


Figure 8.1: An LMC

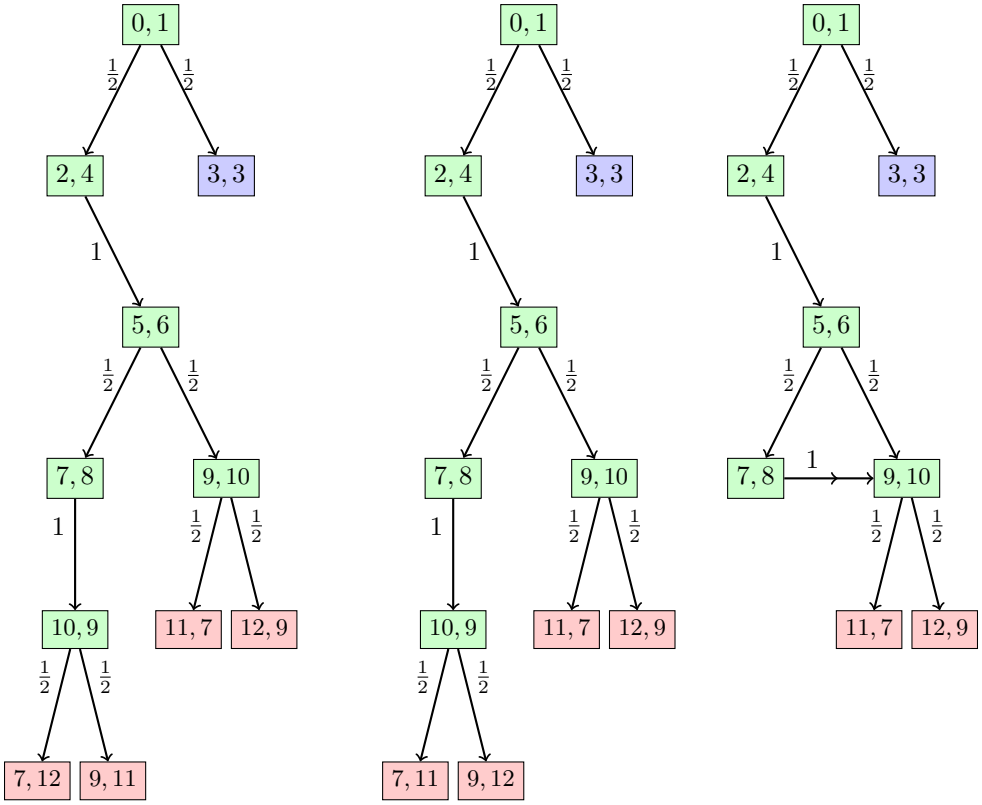


Figure 8.2: Two optimal vertex 1-maximal 0-minimal policies for states 0 and 1 in the LMC depicted in Figure 8.1: non-symmetric and symmetric

For example, consider the policies depicted in Figure 8.2. All of the policies are optimal, as each reaches 1-pairs with a probability of $\frac{1}{2}$, which represents the distance between states 0 and 1. They are also 1-maximal, as the expected length to 1-pairs is 2.5, and 0-minimal, since the expected length to 0-pairs is 0.5. The policy in the middle is the symmetric version of the one on the right. The explanation of the distance of 10 and 9 and that of 9 and 10 are mirror images in the middle policy. To reflect that the one is a mirror image of the other, we introduce twisted arrows. A twisted arrow from $(7, 8)$ to $(9, 10)$ represents an arrow from $(7, 8)$ to $(10, 9)$ in the left policy.

We call a policy P symmetric if $P(s, t)$ and $P(t, s)$ are mirror images.

Definition 8.1. *Let $P \in \mathcal{P}$. P is* symmetric *if for all $(s, t), (u, v) \in S^2_?$,*

$$P(s, t)(u, v) = P(t, s)(v, u).$$

For the remainder of this section, we fix a total order $\prec$ on S. This allows us to turn a policy into a symmetric one as follows.

Definition 8.2. *Let $P \in \mathcal{P}$. We define $P_\prec$ by*

$$P_\prec(s, t)(u, v) = \begin{cases} P(s, t)(u, v) & \text{if } s \prec t \\ P(t, s)(v, u) & \text{if } s \succ t \end{cases}$$

This construction preserves all the properties of policies in which we are interested. The next proposition shows that the symmetric version of a policy is also a policy.

Proposition 8.3. *For all $P \in \mathcal{P}$, $P_\prec \in \mathcal{P}$.*

Proof. Let $P \in \mathcal{P}$ and $(s, t) \in S^2_?$. It remains to show that $P_\prec(s, t) \in \Omega(\tau(s), \tau(t))$. We consider two cases.

- If $s \prec t$, then $P_\prec(s, t) = P(s, t) \in \Omega(\tau(s), \tau(t))$.

- If $s \succ t$, then we should show that $P_\prec(s,t) \in \Omega(\tau(s), \tau(t))$. Since, for all $u \in S$,

$$\begin{aligned} P_\prec(s,t)(u,S) &= \sum_{v \in S} P_\prec(s,t)(u,v) \\ &= \sum_{v \in S} P(t,s)(v,u) \\ &= P(t,s)(S,u) \\ &= \tau(s)(u) \qquad [P(s,t) \in \Omega(\tau(s), \tau(t))] \end{aligned}$$

and

$$\begin{aligned} P_\prec(s,t)(S,u) &= \sum_{v \in S} P_\prec(s,t)(v,u) \\ &= \sum_{v \in S} P(t,s)(u,v) \\ &= P(t,s)(u,S) \\ &= \tau(t)(u) \qquad [P(s,t) \in \Omega(\tau(s), \tau(t))] \end{aligned}$$

we can conclude that $P_\prec(s,t) \in \Omega(\tau(s), \tau(t))$. □

The following proposition proves that using the above definition, we can turn a policy into a symmetric one.

Proposition 8.4. *For all $P \in \mathcal{P}$, $P_\prec$ is symmetric.*

Proof. Let $P \in \mathcal{P}$ and $(s,t), (u,v) \in S_?^2$. We distinguish two cases.

- If $s \prec t$, then $P_\prec(s,t)(u,v) = P(s,t)(u,v) = P_\prec(t,s)(v,u)$.
- If $s \succ t$, then $P_\prec(s,t)(u,v) = P(t,s)(v,u) = P_\prec(t,s)(v,u)$.

□

The next proposition demonstrates that if a policy is optimal, its symmetric version is also optimal.

Proposition 8.5. *For all $P \in \mathcal{P}_{\mathrm{opt}}$, $P_\prec \in \mathcal{P}_{\mathrm{opt}}$.*

Proof. Let $P \in \mathcal{P}_{\mathrm{opt}}$ and $(s,t) \in S_?^2$. We distinguish two cases.

- If $s \prec t$ then

$$\begin{aligned} P_\prec(s,t) \cdot \delta_1 &= P(s,t) \cdot \delta_1 \\ &= \delta_1(s,t) \qquad [P \text{ is optimal and Proposition 4.21}] \end{aligned}$$

- If $s \succ t$ then

$$\begin{aligned} P_\prec(s,t) \cdot \delta_1 &= \sum_{u,v \in S} P_\prec(s,t)(u,v)\, \delta_1(u,v) \\ &= \sum_{u,v \in S} P(t,s)(v,u)\, \delta_1(u,v) \\ &= \sum_{u,v \in S} P_\prec(t,s)(v,u)\, \delta_1(v,u) \qquad [\delta_1 \text{ is symmetric}] \\ &= P(t,s) \cdot \delta_1 \\ &= \delta_1(t,s) \qquad [P \text{ is optimal and Proposition 4.21}] \\ &= \delta_1(s,t) \qquad [\delta_1 \text{ is symmetric}] \end{aligned}$$

Hence, by Proposition 4.21, $P_\prec$ is optimal. □

The next proposition shows that the symmetric version of a vertex policy is a vertex policy.

Proposition 8.6. *For all $P \in \mathcal{V}$, $P_\prec \in \mathcal{V}$.*

Proof. Let $P \in \mathcal{V}$ and $(s,t) \in S^2_?$. We need to show that $P_\prec(s,t)$ is a vertex. We distinguish the following two cases.

- If $s \prec t$ then $P_\prec(s,t) = P(t,s)$, which, by assumption, is a vertex.
- Assume that $s \succ t$. There is an edge between $(v,0)$ and $(u,1)$ in the support graph of $P(t,s)$ if and only if there is an edge between $(u,0)$ and $(v,1)$ in the support graph of $P_\prec(s,t)$. Hence, the support graphs are isomorphic. Since $P(s,t)$ is a vertex, its support graph is a forest by Proposition 3.5. Since the support graphs are isomorphic, the support graph of $P_\prec(s,t)$ is a forest as well. Therefore, from Proposition 3.5 we can conclude that $P_\prec(s,t)$ is a vertex.

□

Corollary 8.7. *For all $P \in \mathcal{V}_{\mathrm{opt}}$, $P_\prec \in \mathcal{V}_{\mathrm{opt}}$.*

Proof. Immediate consequence of Proposition 8.5 and 8.6. □

The following proposition proves that if there are two couplings, π and $\omega \in \Omega_{\mathrm{opt}}(\tau(t),\tau(s))$, where π is the symmetric version of ω, then for all symmetric $l \in D_1 \to [0,\infty)$, the set of values $\omega \cdot (\delta_1 + l)$ is equal to the set of values $\pi \cdot (\delta_1 + l)$. This result is used in the next proposition.

Proposition 8.8. *For all $l \in D_1 \to [0,\infty)$, if l is symmetric then*

$$\{\, \omega \cdot (\delta_1 + l) \mid \omega \in \Omega_{\mathrm{opt}}(\tau(s),\tau(t)) \,\} = \{\, \pi \cdot (\delta_1 + l) \mid \pi \in \Omega_{\mathrm{opt}}(\tau(t),\tau(s)) \,\}$$

Proof. Let $l \in D_1 \to [0,\infty)$. Assume that l is symmetric. Let $\omega \in \Omega_{\mathrm{opt}}(\tau(s),\tau(t))$. Then $\delta_1(s,t) = \omega \cdot \delta_1$. We define $\pi(u,v) = \omega(v,u)$ for all $(u,v) \in S^2_?$. To conclude that $\pi \in \Omega(\tau(t),\tau(s))$ it suffices to show that for all x, $y \in S$,

$$\pi(x,S) = \tau(t)(x) \text{ and } \pi(S,y) = \tau(s)(y).$$

For all $x \in S$,

$$\begin{aligned}
\pi(x,S) &= \sum_{v \in S} \pi(x,v) \\
&= \sum_{v \in S} \omega(v,x) \\
&= \omega(S,x) \\
&= \tau(t)(x) \qquad [\omega \in \Omega(\tau(s),\tau(t))]
\end{aligned}$$

and for all $y \in S$,

$$\begin{aligned}
\pi(S,y) &= \sum_{v \in S} \pi(v,y) \\
&= \sum_{v \in S} \omega(y,v) \\
&= \omega(y,S) \\
&= \tau(s)(y) \qquad [\omega \in \Omega(\tau(s),\tau(t))]
\end{aligned}$$

Hence, $\pi \in \Omega(\tau(t), \tau(s))$. Next, we show that for all $(s,t) \in S_?^2$ we have $\delta_1(t,s) = \pi \cdot \delta_1$.

$$\begin{aligned}
\pi \cdot \delta_1 &= \sum_{(u,v) \in S_?^2} \pi(u,v)\delta_1(u,v) \\
&= \sum_{(u,v) \in S_?^2} \omega(v,u)\delta_1(u,v) \\
&= \sum_{(u,v) \in S_?^2} \omega(v,u)\delta_1(v,u) && [\delta_1 \text{ is symmetric}] \\
&= \omega \cdot \delta_1 \\
&= \delta_1(s,t) && [\delta_1(s,t) = \omega \cdot \delta_1, \text{ by assumption}] \\
&= \delta_1(t,s) && [\delta_1 \text{ is symmetric}]
\end{aligned}$$

Hence, $\pi \in \Omega_{\mathrm{opt}}(\tau(t), \tau(s))$. Finally, we show that $\pi \cdot (\delta_1 + l) = \omega \cdot (\delta_1 + l)$.

$$\begin{aligned}
\pi \cdot (\delta_1 + l) &= \sum_{(u,v) \in D_1} \pi(u,v)(\delta_1(u,v) + l(u,v)) \\
&= \sum_{(u,v) \in D_1} \omega(v,u)(\delta_1(u,v) + l(u,v)) \\
&= \sum_{(u,v) \in D_1} \omega(v,u)(\delta_1(v,u) + l(u,v)) && [\delta_1 \text{ is symmetric}] \\
&= \sum_{(u,v) \in D_1} \omega(v,u)(\delta_1(v,u) + l(v,u)) && [l \text{ is symmetric, by assumption}] \\
&= \omega \cdot (\delta_1 + l)
\end{aligned}$$

Therefore, $\{\omega \cdot (\delta_1 + l) \mid \omega \in \Omega_{\mathrm{opt}}(\tau(s), \tau(t))\} \subseteq \{\pi \cdot (\delta_1 + l) \mid \pi \in \Omega_{\mathrm{opt}}(\tau(t), \tau(s))\}$. Similarly, it can be shown that $\{\pi \cdot (\delta_1 + l) \mid \pi \in \Omega_{\mathrm{opt}}(\tau(t), \tau(s))\} \subseteq \{\omega \cdot (\delta_1 + l) \mid \omega \in \Omega_{\mathrm{opt}}(\tau(s), \tau(t))\}$. □

The next proposition proves that the 1-maximal expected length function is symmetric.

Proposition 8.9. *λ_1 is symmetric.*

Proof. First, we show that for all $n \in \mathbb{N}$, $\Lambda_1^n(\bot)$ is symmetric by induction on n. In the base case, $n = 0$, we need to show that $\bot$ is symmetric which is obviously true. In the inductive case, assume that $n > 0$. By induction, $\Lambda_1^{n-1}(\bot)$ is symmetric. We have that

$$\begin{aligned}
\Lambda_1^n(\bot)(s,t) &= \Lambda_1(\Lambda_1^{n-1}(\bot))(s,t) \\
&= \sup_{\omega \in \Omega_{\mathrm{opt}}(\tau(s), \tau(t))} \omega \cdot (\delta_1 + \Lambda_1^{n-1}(\bot)) \\
&= \sup_{\pi \in \Omega_{\mathrm{opt}}(\tau(t), \tau(s))} \pi \cdot (\delta_1 + \Lambda_1^{n-1}(\bot)) && [\text{induction hypothesis and Proposition 8.8}] \\
&= \Lambda_1(\Lambda_1^{n-1}(\bot))(t,s) \\
&= \Lambda_1^n(\bot)(t,s)
\end{aligned}$$

As a consequence of the above, $\lim_{n \in \mathbb{N}} \Lambda_1^n(\bot)$ is symmetric. Hence, we can conclude from the fact that Λ_1 is monotone (Proposition 6.10) and nonexpansive (Corollary 6.11) and Theorem 2.11(a) that λ_1 is symmetric. □

The next proposition shows that we can turn a 1-maximal optimal policy into a 1-maximal optimal symmetric policy.

Proposition 8.10. *For all $P \in \mathcal{P}_{\mathrm{opt}}^{\max}$, $P_\prec$ is 1-maximal.*

Proof. Let $P \in \mathcal{P}_{\mathrm{opt}}^{\max}$. According to Proposition 6.3, it is suffices to show that for all $(s,t) \in D_1$

$$\lambda_1(s,t) = P_\prec(s,t) \cdot (\delta_1 + \lambda_1).$$

We distinguish two cases.

- If $s \prec t$ then

$$\begin{aligned} P_{\prec}(s,t) \cdot (\delta_1 + \lambda_1) &= P(s,t) \cdot (\delta_1 + \lambda_1) \\ &= \lambda_1(s,t) \qquad [P \text{ is 1-maximal and Proposition 6.3}] \end{aligned}$$

- If $s \succ t$ then

$$\begin{aligned} & P_{\prec}(s,t) \cdot (\delta_1 + \lambda_1) \\ &= \sum_{u,v \in D_1} P_{\prec}(s,t)(u,v)\,(\delta_1(u,v) + \lambda_1(u,v)) \\ &= \sum_{u,v \in D_1} P(t,s)(v,u)\,(\delta_1(u,v) + \lambda_1(u,v)) \\ &= \sum_{u,v \in D_1} P(t,s)(v,u)\,(\delta_1(v,u) + \lambda_1(u,v)) \qquad [\delta_1 \text{ is symmetric}] \\ &= \sum_{u,v \in D_1} P(t,s)(v,u)\,(\delta_1(v,u) + \lambda_1(v,u)) \qquad [\lambda_1 \text{ is symmetric (Proposition 8.9)}] \\ &= P(t,s) \cdot (\delta_1 + \lambda_1) \\ &= \lambda_1(t,s) \qquad [P \text{ is 1-maximal and Proposition 6.3}] \\ &= \lambda_1(s,t) \\ &\quad [\lambda_1 \text{ is symmetric (Proposition 8.9)}] \end{aligned}$$

□

The following proposition proves that if there are two couplings, ω and $\pi \in \Omega^{\max}_{\text{opt}}(\tau(t), \tau(s))$, where π is the symmetric version of ω, then for all symmetric $l \in D_0 \to [0,\infty)$, the set of values $\omega \cdot (\delta_0 + l)$ is equal to the set of values $\pi \cdot (\delta_0 + l)$. This result is used in the next proposition.

Proposition 8.11. *For all $l \in D_0 \to [0,\infty)$, if l is symmetric then*

$$\{\, \omega \cdot (\delta_0 + l) \mid \omega \in \Omega^{\max}_{\text{opt}}(\tau(s), \tau(t)) \,\} = \{\, \pi \cdot (\delta_0 + l) \mid \pi \in \Omega^{\max}_{\text{opt}}(\tau(t), \tau(s)) \,\}.$$

Proof. Let $l \in D_0 \to [0,\infty)$. Assume that l is symmetric. Let $\omega \in \Omega^{\max}_{\text{opt}}(\tau(s), \tau(t))$. Then $\lambda_1(s,t) = \omega \cdot (\delta_1 + \lambda_1)$. We define $\pi(u,v)$ as in the proof of Proposition 8.8. To conclude that $\pi \in \Omega^{\max}_{\text{opt}}(\tau(t), \tau(s))$, it remains to show that for all $(s,t) \in D_1$ we have $\lambda_1(t,s) = \pi \cdot (\delta_1 + \lambda_1)$.

$$\begin{aligned} \pi \cdot (\delta_1 + \lambda_1) &= \sum_{(u,v) \in D_1} \pi(u,v)(\delta_1(u,v) + \lambda_1(u,v)) \\ &= \sum_{(u,v) \in D_1} \omega(v,u)(\delta_1(u,v) + \lambda_1(u,v)) \\ &= \sum_{(u,v) \in D_1} \omega(v,u)(\delta_1(v,u) + \lambda_1(u,v)) \qquad [\delta_1 \text{ is symmetric}] \\ &= \sum_{(u,v) \in D_1} \omega(v,u)(\delta_1(v,u) + \lambda_1(v,u)) \qquad [\lambda_1 \text{ is symmetric (Proposition 8.9)}] \\ &= \omega \cdot (\delta_1 + \lambda_1) \\ &= \lambda_1(s,t) \qquad [\text{by assumption}] \\ &= \lambda_1(t,s) \qquad [\lambda_1 \text{ is symmetric (Proposition 8.9)}] \end{aligned}$$

Finally, we show that $\pi \cdot (\delta_0 + l) = \omega \cdot (\delta_0 + l)$.

$$\begin{aligned}
\pi \cdot (\delta_0 + l) &= \sum_{(u,v) \in D_0} \pi(u,v)(\delta_0(u,v) + l(u,v)) \\
&= \sum_{(u,v) \in D_0} \omega(v,u)(\delta_0(u,v) + l(u,v)) \\
&= \sum_{(u,v) \in D_0} \omega(v,u)(\delta_0(v,u) + l(u,v)) && [\delta_0 \text{ is symmetric}] \\
&= \sum_{(u,v) \in D_0} \omega(v,u)(\delta_0(v,u) + l(v,u)) && [l \text{ is symmetric, by assumption}] \\
&= \omega \cdot (\delta_0 + l)
\end{aligned}$$

Therefore, $\{\, \omega \cdot (\delta_0 + l) \mid \omega \in \Omega^{\max}_{\text{opt}}(\tau(s), \tau(t)) \,\} \subseteq \{\, \pi \cdot (\delta_0 + l) \mid \pi \in \Omega^{\max}_{\text{opt}}(\tau(t), \tau(s)) \,\}$. Similarly, it can be shown that $\{\, \pi \cdot (\delta_0 + l) \mid \pi \in \Omega^{\max}_{\text{opt}}(\tau(t), \tau(s)) \,\} \subseteq \{\, \omega \cdot (\delta_0 + l) \mid \omega \in \Omega^{\max}_{\text{opt}}(\tau(s), \tau(t)) \,\}$. □

The next proposition proves that the 0-minimal expected length function is symmetric.

Proposition 8.12. *λ_0 is symmetric.*

Proof. First, we show that for all $n \in \mathbb{N}$, $\Lambda_0^n(\bot)$ is symmetric by induction on n. In the base case, $n = 0$, we need to show that $\top$ is symmetric which is obviously true. In the inductive case, assume that $n > 0$. By induction, $\Lambda_0^{n-1}(\bot)$ is symmetric. We have that

$$\begin{aligned}
\Lambda_0^n(\bot)(s,t) &= \Lambda_0(\Lambda_0^{n-1}(\bot))(s,t) \\
&= \inf_{\omega \in \Omega^{\max}_{\text{opt}}(\tau(s), \tau(t))} \omega \cdot (\delta_0 + \Lambda_0^{n-1}(\bot)) \\
&= \inf_{\pi \in \Omega^{\max}_{\text{opt}}(\tau(t), \tau(s))} \pi \cdot (\delta_0 + \Lambda_0^{n-1}(\bot)) && [\text{induction hypothesis and Proposition 8.11}] \\
&= \Lambda_0(\Lambda_0^{n-1}(\bot))(t,s) \\
&= \Lambda_0^n(\bot)(t,s)
\end{aligned}$$

As a consequence of the above, $\lim_{n \in \mathbb{N}} \Lambda_0^n(\bot)$ is symmetric. Hence, we can conclude from the fact that Λ_0 is monotone (Proposition 6.10) and nonexpansive (Proposition 7.11) and Theorem 2.11(a) that λ_0 is symmetric. □

The next proposition shows that we can turn a 0-minimal 1-maximal optimal policy into a 0-minimal 1-maximal optimal symmetric policy.

Proposition 8.13. *For all $P \in \mathcal{P}^{\max}_{\text{opt}}$, if P is 0-minimal then $P_{\prec}$ is 0-minimal.*

Proof. Let $P \in \mathcal{P}^{\max}_{\text{opt}}$. According to Proposition 7.3, it is suffices to show that for all $(s,t) \in D_0$

$$\lambda_0(s,t) = P_{\prec}(s,t) \cdot (\delta_0 + \lambda_0).$$

We distinguish two cases.

- If $s \prec t$ then

$$\begin{aligned}
P_{\prec}(s,t) \cdot (\delta_0 + \lambda_0) &= P(s,t) \cdot (\delta_0 + \lambda_0) \\
&= \lambda_0(s,t) && [P \text{ is 0-maximal Proposition 7.3}]
\end{aligned}$$

- If $s \succ t$ then

$$\begin{aligned}
P_{\prec}(s,t) \cdot (\delta_0 + \lambda_0) &= \sum_{u,v \in D_0} P_{\prec}(s,t)(u,v)\,(\delta_0(u,v) + \lambda_0(u,v)) \\
&= \sum_{u,v \in D_0} P(t,s)(v,u)\,(\delta_0(u,v) + \lambda_0(u,v)) \\
&= \sum_{u,v \in D_0} P(t,s)(v,u)\,(\delta_0(v,u) + \lambda_0(u,v)) \qquad [\delta_0 \text{ is symmetric}] \\
&= \sum_{u,v \in D_0} P(t,s)(v,u)\,(\delta_0(v,u) + \lambda_0(v,u)) \\
&\qquad [\lambda_0 \text{ is symmetric (Proposition 8.12)}] \\
&= P(t,s) \cdot (\delta_0 + \lambda_0) \\
&= \lambda_0(t,s) \qquad [P \text{ is 0-maximal Proposition 7.3}] \\
&= \lambda_0(s,t) \qquad [\lambda_0 \text{ is symmetric (Proposition 8.12)}]
\end{aligned}$$

□

In this chapter, we explored symmetric policies and showed that we can convert 0-minimal 1-maximal optimal vertex policies into 0-minimal 1-maximal optimal vertex symmetric policies. As we have seen, symmetric policies are simpler and help us explain the probabilistic bisimilarity distance more easily, making them preferable. In the next chapter, we will see the Java implementation of the 1-maximal Algorithm 1 and the 0-minimal Algorithm 2 that we discussed in the earlier chapters.

9 Java Implementation

In this chapter, we will provide an overview of the Java implementation[6] of the Algorithms 1 and 2 described in this thesis. First, we start by discussing how we represent an LMC in our implementation.

9.1 LMC Implementation

In our implementation, the class *LabelledMarkovChain* represents an LMC. It takes two input files: a label file (.lab) and a transition file (.tra). For example, consider the following label file, where each state is assigned a label from the set of labels $\{a, b, c\}$,

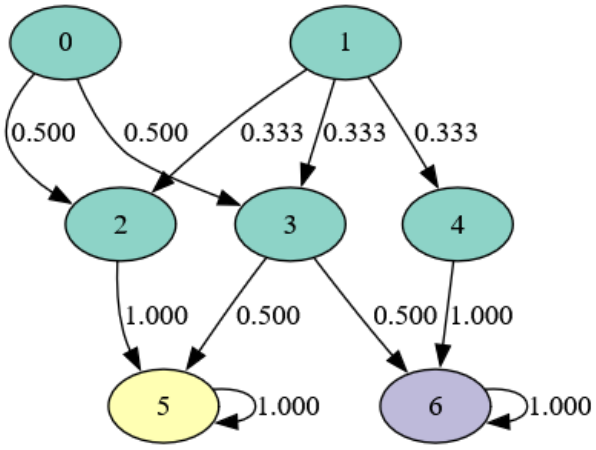
```
0: "a"
1: "a"
2: "a"
3: "a"
4: "a"
5: "b"
6: "c"
```

and the transition file which follows this format: the first row contains two numbers: the first indicates the number of states, and the second indicates the number of transitions. Starting from the second row, each line has three numbers: the source state, the target state, and the transition probability between them. Notice that the sum of the transition probabilities for each state is always equal to one, as required in LMCs.

```
7 11
0 2 0.5
0 3 0.5
1 3 0.3333333
1 4 0.3333333
1 2 0.3333333
2 5 1
3 5 0.5
3 6 0.5
6 6 1
5 5 1
4 6 1
```

In our code, we added a method that generates a graphical representation of the input LMC. For example, the graphical representation of the input LMC above is shown as follows.

[6]The code is available at github.com/antoNanahJi/Explainability.



Next, we will provide an overview of the Java implementation of the 1-maximal algorithm.

9.2 1-Maximal Algorithm

Recall the 1-maximal Algorithm 1 presented in Chapter 6.

Algorithm 1 1-maximal optimal vertex policy

Input: optimal vertex policy P

1: compute λ_{1P}
2: **while** $\exists (s,t) \in D_1 \setminus S_1^2 : \Lambda_1(\lambda_{1P})(s,t) > \lambda_{1P}(s,t)$ **do**
3: $\quad \pi \leftarrow \underset{\omega \in V(\Omega(\tau(s),\tau(t))) \wedge \delta_1(s,t)=\omega\cdot\delta_1}{\arg\max} \omega \cdot (\delta_1 + \lambda_{1P})$
4: $\quad P(s,t) \leftarrow \pi$
5: $\quad$ compute λ_{1P}
6: **end while**

The above algorithm begins with an optimal vertex policy P. To compute an optimal vertex policy, we used a Java implementation[7] of the algorithm presented in [52, Section 6.2]. Then it computes λ_{1P} for this policy (line 1). In our implementation, λ_{1P} is computed using the power method discussed in [2, Theorem 10.15].

As long as there exists a state pair in $D_1 \setminus S_1^2$ that is not locally 1-maximal with respect to the current policy (line 2), the policy at (s,t) is updated to a locally 1-maximal choice (lines 3 and 4). This step is implemented using Java's linear programming *SimplexSolver*[8] class. The objective function for this solver is $\omega \cdot (\delta_1 + \lambda_{1P})$, subject to the linear constraints $\omega \in \Omega(\tau(s), \tau(t))$ and $\delta_1(s,t) = \omega \cdot \delta_1$. The solver returns a new coupling for the state pair (s,t). This new coupling is a vertex of the transportation polytope.

After updating the policy P, the algorithm recomputes λ_{1P} (line 5) using the same power method. The loop maintains the invariant that P is an optimal vertex policy as π in line 3 is a vertex and satisfies $\delta_1(s,t) = \pi \cdot \delta_1$ (see Proposition 4.21).

At termination, we have that λ_{1P} is a fixed point of Λ_1 and, by Theorem 6.15, equals λ_1 and, therefore, is 1-maximal.

For example, let us focus on state pairs $(0,4)$ of the LMC depicted in Figure 9.1. For this state pair, if the algorithm starts with the optimal vertex policy shown in Figure 9.2, which is obtained using the policy iteration algorithm from [53], it then produces the 1-maximal optimal vertex policy illustrated in Figure 9.3.

[7] The code is available at bitbucket.org/discoveri/probabilistic-bisimilarity-distances.

[8] commons.apache.org/proper/commons-math/javadocs/api-3.6.1/org/apache/commons/math3/optim/linear/SimplexSolver.html.

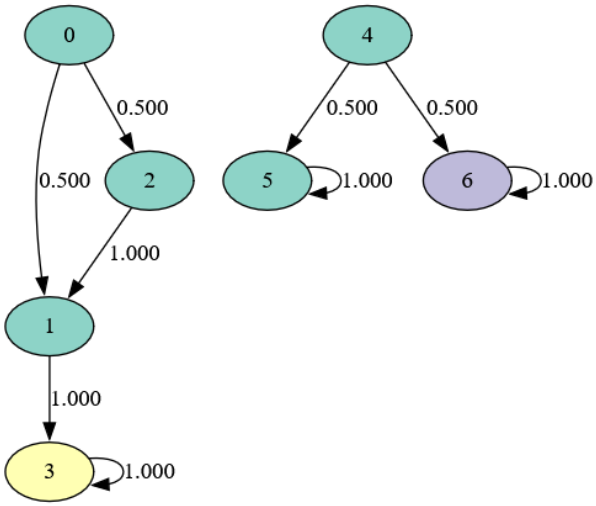


Figure 9.1: An LMC

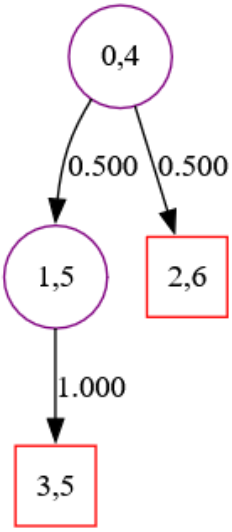


Figure 9.2: A vertex optimal policy for LMC in Figure 9.1

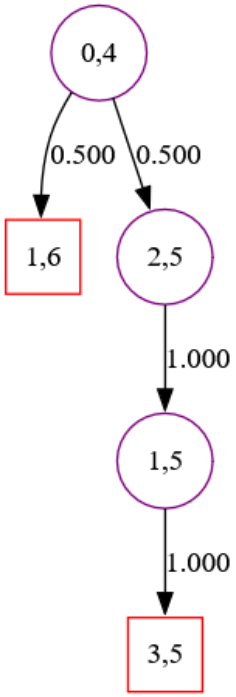


Figure 9.3: A 1-maximal vertex optimal policy for LMC in Figure 9.1

Next, we will provide an overview of the Java implementation of the 0-minimal algorithm.

9.3 0-Minimal Algorithm

Recall the 0-minimal Algorithm 2 presented in Chapter 7.

Algorithm 2 0-minimal 1-maximal optimal vertex policy

Input: 1-maximal optimal vertex policy P

1: compute λ_{0P}
2: **while** $\exists (s,t) \in D_0 \setminus S_0^2 : \Lambda_0(\lambda_{0P})(s,t) < \lambda_{0P}(s,t)$ **do**
3: $\quad \pi \leftarrow \underset{\omega \in V(\Omega(\tau(s),\tau(t))) \wedge \delta_1(s,t)=\omega\cdot\delta_1 \wedge \lambda_1(s,t)=\omega\cdot(\delta_1+\lambda_1)}{\arg\min} \omega \cdot (\delta_0 + \lambda_{0P})$
4: $\quad P(s,t) \leftarrow \pi$
5: $\quad$ compute λ_{0P}
6: **end while**

Our algorithm that computes a 0-minimal 1-maximal vertex policy from a 1-maximal vertex policy (Algorithm 2) is very similar in structure to Algorithm 1. Instead of focusing on 1-pairs, we concentrate on 0-pairs. Furthermore, we maintain as a loop invariant that P is not only an optimal vertex policy but also that it is 1-maximal.

The above algorithm begins with a 1-maximal optimal vertex policy P, which can be obtained using the 1-maximal Algorithm 1. Then it computes λ_{0P} for this policy (line 1). In our implementation, λ_{0P} is also computed using the power method discussed in [2, Theorem 10.15].

As long as there exists a state pair in $D_0 \setminus S_0^2$ that is not locally 0-minimal with respect to the current policy (line 2), the policy at (s,t) is updated to a locally 0-minimal choice (lines 3 and 4). This step is also implemented using Java's linear programming *SimplexSolver* class. The objective function for this solver is $\omega \cdot (\delta_0 + \lambda_{0P})$, subject to the linear constraints $\omega \in \Omega(\tau(s), \tau(t))$ and $\delta_1(s,t) = \omega \cdot \delta_1$ and $\lambda_1(s,t) = \omega \cdot (\delta_1 + \lambda_1)$. The solver returns a new coupling, which is a vertex of the transportation polytope, for the state pair (s,t).

After updating the policy P, the algorithm recomputes λ_{0P} (line 5) using the same power method. The loop maintains the invariant that P is not only an optimal vertex policy but also that it is 1-maximal.

At termination, we have that λ_{0P} is a fixed point of Λ_0 and, by Theorem 7.15, equals λ_0 and, therefore, is 0-minimal.

For example, let us focus on state pairs $(0, 1)$ of the LMC depicted in Figure 9.4. For this state pair, if the algorithm starts with the 1-maximal optimal vertex policy shown in Figure 9.5, which is obtained using the 1-maximal Algorithm 1, it then produces the 0-minimal 1-maximal optimal vertex policy illustrated in Figure 9.6.

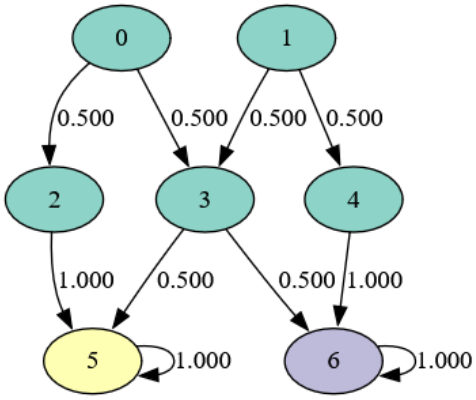


Figure 9.4: An LMC

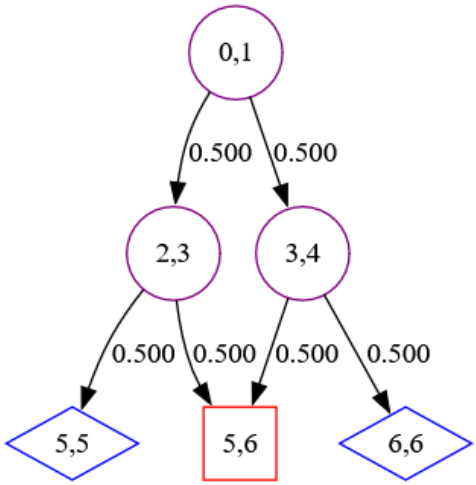


Figure 9.5: A 1-maximal vertex optimal policy for LMC in Figure 9.4

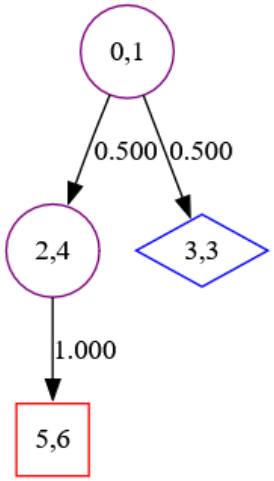


Figure 9.6: A 0-minimal 1-maximal vertex optimal policy for LMC in Figure 9.4

Next, we present the experimental results for the algorithms described in this thesis.

9.4 Experimental Results

In our experimental evaluation we used the two examples of [52, Section 9.3], namely an example of two dies due to Knuth and Yao [36] and randomized quick sort, as well as the Miller-Rabin primality test [46, 49].

The experiments were run on an Intel machine with an i7-8700T CPU and 15 GB of RAM. For each of the three examples, we first computed an optimal vertex policy, then a 1-maximal optimal vertex policy, and finally a 0-minimal 1-maximal optimal vertex policy. We timed 55 trails for each of the three stages of the computation and report the average (and standard deviation) in milliseconds in Table 9.1. Since a Java virtual machine needs to perform just-in-time compilation and optimization, we discarded the first eight trails. Garbage collection was triggered in between trials to minimize its impact on our measurements. As can be seen, finding 0-minimal and 1-maximal policies can be done significantly faster than finding optimal policies.

	optimal	1-maximal	0-minimal
Miller-Rabin	32,149 (149)	479 (2)	1,069 (5)
Two dies	3,142 (1)	2,271 (1)	1,569 (9)
Quicksort	218,232 (8,971)	38,528 (1,139)	46,416 (1,165)

Table 9.1: Average (and standard deviation) in milliseconds of time to compute optimal, 1-maximal, and 0-minimal policies

10 Conclusion

We have shown that explainability is a game in the context of probabilistic bisimilarity distances of LMCs. More precisely, symmetric 0-minimal 1-maximal vertex optimal policies for the $1\frac{1}{2}$-player games presented in this thesis explain the probabilistic bisimilarity distances.

The literature review in the introduction of this thesis provides a comprehensive and well-organized overview of game-theoretic and logical characterizations of probabilistic bisimilarity distances. In the introduction, we presented our game, which is based on matching parts of transitions. These matchings, also known as couplings, were introduced in Chapter 3, and in Chapter 4, we defined the player's strategy, also referred to as a policy, which involves selecting a coupling. The player's objective is to avoid reaching 1-pairs, meaning that the expected length of the game before reaching a 1-pair should be maximized. Additionally, we are interested in the expected length to reach a 0-pair.

In Chapter 5, we formulated the expected length to an i-pair, and in Chapter 6, we presented Algorithm 1, which maximizes the expected length to reach a 1-pair. We also proved an exponential lower bound for this algorithm by constructing an LMC of size $O(n)$, for which the algorithm requires $\Omega(2^n)$ iterations. In this chapter, we further demonstrated that a 1-maximal optimal policy does not have any 1-conflicts, a concept introduced by Vlasman in [58].

In Chapter 7, we presented Algorithm 2, which minimizes the expected length to reach a 0-pair. Proving an exponential lower bound for this algorithm remains a topic for future research. Moreover, Vlasman [58] introduced the notion of probabilistic bisimilar conflict, which we refer to as 0-conflict. Let us now recall the definition.

Definition 10.1.

- *Let* $(w, x), (y, z) \in S_?^2$. *Then* (w, x) *and* (y, z) *are in* 0-conflict *if* $w \sim z$ *or* $x \sim y$.
- *Let* $\omega \in \mathcal{D}(S \times S)$. *Then* ω *is* 0-conflict free *if for all* (w, x), $(y, z) \in \text{support}(\omega) \cap D_0$, (w, x) *and* (y, z) *are not in 0-conflict.*
- *Let* $P \in \mathcal{P}$. *Then* P *is* 0-conflict free *if for all* $(s, t) \in S_?^2$, $P(s, t)$ *is 0-conflict free.*

Another avenue for further research could involve exploring whether a 0-minimal 1-maximal vertex optimal policy is 0-conflict free.

In Chapter 8, we explored symmetric policies and showed that 0-minimal 1-maximal vertex optimal policies can be converted into symmetric 0-minimal 1-maximal vertex optimal policies. Modifying the policy iteration algorithm from [53], along with Algorithms 1 and 2, to maintain that the policy is symmetric as a loop invariant is a potential direction for future research. This could allow us to focus on only half of the state pairs currently considered in the algorithms, thereby reducing computation time.

In Chapter 9, we gave an overview of the Java implementation of the Algorithms 1 and 2 described in this thesis. However, due to the nature of floating-point arithmetic, some rounding errors occurred during computation. A deeper investigation into these errors and their mitigation is left for future research.

Apart from the area of probabilistic model checking, probabilistic bisimilarity distances also play a role in other fields including control theory [26], fault-tolerance [5], privacy [7], quantum computing [21], reinforcement learning [44], and systems-biology [41]. As a consequence, we anticipate that our results are widely applicable.

Bibliography

[1] Giorgio Bacci, Giovanni Bacci, Kim G Larsen, and Radu Mardare. On-the-fly exact computation of bisimilarity distances. In Nir Piterman and Scott Smolka, editors, *Proceedings of the 19th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, volume 7795 of *Lecture Notes in Computer Science*, pages 1–15, Rome, Italy, Mar 2013. Springer-Verlag.

[2] Christel Baier and Joost-Pieter Katoen. *Principles of Model Checking*. The MIT Press, Cambridge, MA, USA, Apr 2008.

[3] Stefan Banach. Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales. *Fundamenta Mathematicae*, 3(1):133–181, 1922.

[4] Peter Buchholz. Exact performance equivalence: An equivalence relation for stochastic automata. *Theoretical Computer Science*, 215(1):263–287, May 1999.

[5] Pablo F Castro, Pedro R D'Argenio, Ramiro Demasi, and Luciano Putruele. Measuring masking fault-tolerance. In *Proceedings of the 25th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, Lecture Notes in Computer Science, pages 375–392. Springer-Verlag, 2019.

[6] Di Chen, Franck van Breugel, and James Worrell. On the complexity of computing probabilistic bisimilarity. In Lars Birkedal, editor, *Proceedings of the 15th International Conference on Foundations of Software Science and Computational Structures, volume 7213 of Lecture Notes in Computer Science*, page 437–451, Tallinn, Estonia, Mar/Apr 2012. Springer-Verlag.

[7] Dmitry Chistikov, Andrzej S Murawski, and David Purser. Bisimilarity distances for approximate differential privacy. In Shaon Lahiri and Connie Wang, editors, *Proceedings of the 16th International Symposium on Automated Technology for Verification and Analysis.*, volume 11138 of *Lecture Notes in Computer Science*, pages 194–210, Los Angeles, CA, USA, Oct 2018. Springer-Verlag.

[8] Edmund M. Clarke and E. Allen Emerson. Design and synthesis of synchronization skeletons using branching time temporal logic. In Dexter Kozen, editor, *Logics of Programs*, pages 52–71, Berlin, Heidelberg, 1982. Springer Berlin Heidelberg.

[9] Rance Cleaveland. On automatically distinguishing inequivalent processes. In Edmund Clarke and Robert Kurshan, editors, *Proceedings of a Discrete Mathematics and Theoretical Computer Science Workshop on Computer Aided Verification*, volume 3 of *Discrete Mathematics and Theoretical Computer Science Series in Discrete Mathematics and Theoretical Computer Science*, pages 463–476, New Brunswick, NJ, USA, Jun 1990. DIMACS/AMS.

[10] George Dantzig. Application of the simplex method to a transportation problem. *Activity analysis of production and allocation*, pages 359–373, 1951.

[11] Brian Davey and Hilary Priestley. *Introduction to lattices and order*. Cambridge University Press, Cambridge, United Kingdom, 2002.

[12] Eric Denardo. Contraction mappings in the theory underlying dynamic programming. *SIAM Review*, 9(2):165–177, Apr 1967.

[13] Salem Derisavi, Holger Hermanns, and William H Sanders. Optimal state-space lumping in Markov chains. *Information Processing Letters*, 87(6):309–315, Jun 2003.

[14] J. Desharnais, A. Edalat, and P. Panangaden. A logical characterization of bisimulation for labelled Markov processes. In *Proceedings. Thirteenth Annual IEEE Symposium on Logic in Computer Science (Cat. No.98CB36226)*, pages 478–487, Indianapolis, IN, USA, 1998. IEEE.

[15] Josee Desharnais, Radha Jagadeesan, Vineet Gupta, and Prakash Panangaden. The metric analogue of weak bisimulation for probabilistic processes. In *Proceedings of the 17th Annual IEEE Symposium on Logic in Computer Science*, pages 413–422, Openhagen, Denmark, Jul 2002.

[16] Josée Desharnais, François Laviolette, and Mathieu Tracol. Approximate analysis of probabilistic processes: Logic, simulation and games. In *Proceedings of the 5th International Conference on the Quantitative Evaluation of System*, pages 264–273, Saint-Malo, France, Sep 2008. IEEE.

[17] Josée Desharnais, Vineet Gupta, Radha Jagadeesan, and Prakash Panangaden. Metrics for labelled Markov processes. *Theoretical Computer Science*, 318(3):323–354, Aug 2004.

[18] Wolfgang Doeblin. Exposé de la théorie des chaınes simples constantes de Markova un nombre fini d'états. *Revue mathématique de l'Union Interbalkanique*, 2(77-105):78–80, 1938.

[19] Ryszard Engelking. *General Topology*. Berlin, Germany, 1989.

[20] Ky Fan. Minimax theorems. *Proceedings of the National Academy of Sciences*, 39(1):42–47, Jan 1953.

[21] Yuan Feng, Runyao Duan, and Mingsheng Ying. Bisimulation for quantum processes. In *Proceedings of the 38th Annual ACM SIGPLAN-SIGACT Symposium on Principles of programming languages*, volume 46, pages 523–534, Austin, TX, USA, Jan 2011. ACM.

[22] Nathanaël Fijalkow, Bartek Klin, and Prakash Panangaden. Expressiveness of probabilistic modal logics, revisited. In Ioannis Chatzigiannakis, Piotr Indyk, Fabian Kuhn, and Anca Muscholl, editors, *Proceedings of 44th International Colloquium on Automata, Languages and Programming*, volume 80 of *Leibniz International Proceedings in Informatics*, pages 105:1–105:12, Warrsaw, Poland, Jul 2017. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

[23] Kathi Fisler and Moshe Y. Vardi. Bisimulation and model checking. In *Advanced Research Working Conference on Correct Hardware Design and Verification Methods*, pages 338–342, Berlin, Heidelberg, Sep 1999. Springer.

[24] Vojtěch Forejt, Petr Jančar, Stefan Kiefer, and James Worrell. Game characterization of probabilistic bisimilarity, and applications to pushdown automata. *Logical Methods in Computer Science*, 14, Nov 2018.

[25] Alessandro Giacalone, Chi-Chang Jou, and Scott Smolka. Algebraic reasoning for probabilistic concurrent systems. In *Proceedings of the IFIP WG 2.22.3 Working Conference on Programming Concepts and Methods*, pages 443–458, Sea of Gallilee, Israel, Apr 1990. North-Holland.

[26] Antoine Girard and George J. Pappas. Approximate bisimulation: A bridge between computer science and control theory. *European Journal of Control*, 17(5):568–578, Sep 2011.

[27] Frank Gray. Pulse code communication. *US Patent Number 2632058*, Mar 1953.

[28] Hans Hansson and Bengt Jonsson. A logic for reasoning about time and reliability. *Formal Aspects of Computing*, 6.5:512–535, Sep 1994.

[29] Matthew Hennessy and Robin Milner. On observing nondeterminism and concurrency. In Jaco de Bakker and Jan van Leeuwen, editors, *Proceedings of the 7th Colloquium on Automata, Languages and Programming*, volume 85 of *Lecture Notes in Computer Science*, pages 299–309, Noordwijkerhout, The Netherlands, Jul 1980. Springer-Verlag.

[30] Ronald A Howard. Dynamic programming and Markov processes. *The MIT Press*, 2:39–47, 1960.

[31] Joost-Pieter Katoen, Tim Kemna, Ivan Zapreev, and David Jansen. Bisimulation minimisation mostly speeds up probabilistic model checking. In Orna Grumberg and Michael Huth, editors, *Proceedings of the 13th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, volume 4424 of *Lecture Notes in Computer Science*, pages 87–101, Braga, Portugal, Mar 2007. Springer-Verlag.

[32] Robert M. Keller. Formal verification of parallel programs. *Commun. ACM*, 19(7):371–384, Jul 1976.

[33] L.G. Khachiyan. Polynomial algorithms in linear programming. *USSR Computational Mathematics and Mathematical Physics*, 20(1):53–72, 1980.

[34] Victor Klee and Christoph Witzgall. Facets and vertices of transportation polytopes. In George Dantzig and Arthur Veinott, editors, *Proceedings of the 5th Summer Seminar on the Mathematics of the Decision Sciences*, volume 11 of *Lectures in Applied Mathematics*, pages 257–282, Stanford, CA, USA, Jun/Jul 1967. AMS.

[35] Bronislaw Knaster. Un théorème sur les fonctions d'ensembles. *Annales Polonici Mathematici*, 6:133–134, 1928.

[36] Donald Knuth and Andrew Yao. The complexity of nonuniform random number generation. In Joseph Traub, editor, *Proceedings of a Symposium on New Directions and Recent Result*, page 375–428, Pittsburgh, PA, USA, Apr 1976. Academic Press.

[37] Ignace Kolodner. Fixed points. *The American Mathematical Monthly*, 71(9):906, Oct 1964.

[38] Yuichi Komorida, Shin-ya Katsumata, Nick Hu, Bartek Klin, Samuel Humeau, Clovis Eberhart, and Ichiro Hasuo. Codensity games for bisimilarity. In *Proceedings of the 34th Annual ACM/IEEE Symposium on Logic in Computer Science*, number 2, pages 1–13, Vancouver, Canada, Jun 2019. IEEE.

[39] Barbara König and Christina Mika-Michalski. (Metric) bisimulation games and real-valued modal logics for coalgebras. In Sven Schewe and Lijun Zhang, editors, *Proceedings of the 29th International Conference on Concurrency Theory*, volume 118 of *Leibniz International Proceedings in Informatics*, pages 37:1–37:17, Beijing, China, Sep 2018. Schloss-Dagstuhl-Leibniz Zentrum für Informatik.

[40] Barbara König, Christina Mika-Michalski, and Lutz Schröder. Explaining non-bisimilarity in a coalgebraic approach: Games and distinguishing formulas. In Daniela Petrisan and Jurriaan Rot, editors, *Proceedings of the 5th IFIP WG 1.3 International Workshop on Coalgebraic Methods in Computer Science*, volume 12094 of *Lecture Notes in Computer Science*, pages 133–154, Dublin, Ireland, Apr 2020. Springer-Verlag.

[41] Ilya Korsunsky, Kathleen McGovern, Tom LaGatta, Loes Olde Loohuis, Terri Grosso-Applewhite, Nancy Griffeth, and Bud Mishra. Systems biology of cancer: a challenging expedition for clinical and quantitative biologists. *Frontiers in Bioengineering and Biotechnology*, 2:27, Aug 2014.

[42] Herb Krasner. The cost of poor software quality in the US: A 2022 report. *Product Consortium Information Software QualityTM (CISQTM)*, 2, Dec 2022.

[43] Kim Larsen and Arne Skou. Bisimulation through probabilistic testing. In *Proceedings of the 16th Annual ACM Symposium on Principles of Programming Languages*, pages 344–352, Austin, TX, USA, Jan 1989. ACM.

[44] Qiyuan Liu, Qi Zhou, Rui Yang, and Jie Wang. Robust representation learning by clustering with bisimulation metrics for visual reinforcement learning with distractions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2023.

[45] Peter McMullen and Geoffrey Colin Shephard. *Convex Polytopes*. Lecture note series, 3. Cambridge University Press, Jul 1971.

[46] Gary Miller. Riemann's hypothesis and tests for primality. In Wliam Rounds, Nancy Martin, Jack Carlyle, and Michael Harrison, editors, *Proceedings of the 7th annual ACM Symposium on Theory of computing*, pages 234–239, Albuquerque, NM, USA, May 1975.

[47] Robin Milner. *A Calculus of Communicating Systems*, volume 92 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, Germany, 1980.

[48] David Park. Concurrency and automata on infinite sequences. In Peter Deussen, editor, *Proceedings of the 5th GI-Conference on Theoretical Computer Science*, volume 104 of *Lecture Notes in Computer Science*, page 167–183, Karlsruhe, Germany, Mar 1981. Springer-Verlag.

[49] Michael Rabin. Probabilistic algorithm for testing primality. *Journal of number theory*, 12(1):128–138, Feb 1980.

[50] Amgad Rady and Franck van Breugel. Explainability of probabilistic bisimilarity distances for labelled Markov chains. In *Proceedings of the 26th International Conference on Foundations of Software Science and Computation Structures*, volume 13992 of *Lecture Notes in Computer Science*, pages 285–307, Paris, France, Apr 2023. Springer-Verlag.

[51] Colin Stirling. Modal and temporal logics for processes. In Faron Moller and Graham Birtwistle, editors, *Proceedings of 8th Banff Higher Order Workshop*, volume 1043 of *Lecture Notes in Computer Science*, pages 149–237, Banff, Canada, Aug/Sep 1995. Springer-Verla.

[52] Qiyi Tang. *Computing Probabilistic Bisimilarity Distances*. PhD thesis, York University, Toronto, Canada, Aug 2018.

[53] Qiyi Tang and Franck van Breugel. Computing Probabilistic Bisimilarity Distances via Policy Iteration. In Josée Desharnais and Radha Jagadeesan, editors, *27th International Conference on Concurrency Theory (CONCUR 2016)*, volume 59 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 22:1–22:15, Dagstuhl, Germany, Aug 2016. Schloss Dagstuhl – Leibniz-Zentrum für Informatik.

[54] Alfred Tarski. A lattice-theoretic fixed point theorem and its applications. *Pacific Journal of Mathematics*, 5(2):285–309, Jun 1955.

[55] Kathleen Trustrum. Linear programming. London, UK, 1971. London, Routledge and Kegan, Paul.

[56] Antti Valmari and Giuliana Franceschinis. Simple $O(m \log n)$ time Markov chain lumping. In *Proceedings of the 16th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, TACAS'10, page 38–52, Berlin, Heidelberg, Mar 2010. Springer-Verlag.

[57] Franck van Breugel. On behavioural pseudometrics and closure ordinals. *Information Processing Letters*, 112(19):715–718, Oct 2012.

[58] Emily Vlasman. On logical and game characterizations of probabilistic bisimilarity distances for labelled Markov chains. Master's thesis, University of Oxford, Oxford, UK, Sep 2023.

[59] W. Eric Wong, Xuelin Li, and Philip A. Laplante. Be more familiar with our enemies and pave the way forward: A review of the roles bugs played in software failures. *Journal of Systems and Software*, 133:68–94, 2017.