A Formal Proof of Decidability of Multi-Weighted Declining Energy Games

Master's thesis by Caroline Lemke

Submission: November 30, 2024
Revision: February 24, 2025

Technische Universität Berlin
Models and Theory of Distributed Systems

Supervisor: Benjamin Bisping
First examiner: Prof. Dr.-Ing. Uwe Nestmann
Second examiner: Prof. Dr. Uli Fahrenberg

Abstract

We investigate the decidability of multi-weighted declining energy games as used by Bisping [7] to simultaneously decide all common notions of behavioural equivalence at once. These are zero-sum two-player games with reachability winning conditions played on directed graphs labelled by multi-weighted energy functions. While Bisping claims decidability and provides an algorithm to compute minimal attacker winning budgets (i.e. Pareto fronts), the claim lacks a formal proof. Using Isabelle/HOL, we provide a machine-checked proof to substantiate Bisping's claim of decidability.
Building on these insights, we abstract the necessary properties used in the proof and introduce new classes of energy games, named (generalised) Galois energy games. Simplifying and generalising Bisping's algorithm we prove the decidability of these classes, if the set of positions is finite and the energies form a well-founded bounded join-semilattice. While monotonicity is crucial in our reasoning, we are able to show decidability for energy games that are not declining. However, declining energy games result in a significantly better running time, which in some fixed-dimension cases is polynomial. Providing new decidability results, this work strengthens the theoretical foundations of multi-weighted energy games.

Zusammenfassung

In dieser Arbeit untersuchen wir die Entscheidbarkeit von multi-gewichteten absteigenden Energiespielen. Wir betrachten Nullsummenspiele mit Erreichbarkeitsgewinnbedingung, in welchen zwei Spieler*innen auf einem gerichteten Graphen spielen, der mit multi-gewichteten Energiefunktionen beschriftet ist. Solche Spiele verwendet Bisping [7], um alle gängigen Verhaltensäquivalenzen gleichzeitig zu entscheiden. Bisping behauptet ohne Beweis, dass diese Spiele entscheidbar sind und gibt einen Algorithmus zur Berechnung minimaler Gewinnbudgets des Angriffs, also von Pareto-Fronten, an. Wir stützen diese Behauptung mit einem maschinell geprüften Beweis in Isabelle/HOL.
Aufbauend auf diesen Erkenntnissen abstrahieren wir die relevanten Eigenschaften und führen neue Klassen von Energiespielen ein, die wir (verallgemeinerte) Galois-Energiespiele nennen. Durch die Vereinfachung und Verallgemeinerung Bispings Algorithmus beweisen wir die Entscheidbarkeit dieser Klassen, sofern die Menge der Positionen endlich ist und die partielle Ordnung auf Energien ein wohlfundierter beschränkter Vereinigungs-Halbverband ist. Während Monotonie für unsere Argumentation essentiell ist, können wir Entscheidbarkeit auch für nicht absteigende Energiespiele zeigen. Gleichzeitig führen absteigende Energiespiele zu einer signifikant besseren Laufzeit, die in manchen Fällen mit fester Dimension polynomiell ist. Mit neuen Resultaten zur Entscheidbarkeit stärkt diese Arbeit die theoretischen Grundlagen von multi-gewichteten Energiespielen.

Contents

Chapter 1

Introduction

Let us unpack the title in reverse order: **A formal proof of decidability of multi-weighted declining energy games.**
When studying reactive systems, many questions can be reformulated by asking, who wins some two-player **game**. One example for such a question is whether a specification of a reactive system is realisable; Abadi, Lamport and Wolper [1] understand a specification as the rules of a two-player game where the system plays against the environment and must ensure correct behaviour. Another prominent example is Stirling's bisimulation game [30] used to check whether two processes can simultaneously simulate each other. When resource constraints have to be considered, (multi-weighted) **energy** games can be utilised to represent (multiple) such quantitative goals. In this way, Chakrabarti et al. [13] study component interfaces on limited resources, such as power, and give an energy game to check their compatibility.
Energy games are a class of two-player zero-sum games with perfect information played on labelled graphs. The labels correspond to energy functions – we call them *updates*. Commonly, energies are represented as (vectors of) naturals or non-negative reals. We consider arbitrary partially ordered sets adding further assumptions where needed. The *energy level* of a play is calculated through the repeated application of updates according to the edges chosen by the players. One player – we will call them the *attacker* – is energy-bound and seeks to maintain the energy level above a zero-point, while the opposing player tries to drive it below that zero-point. In **multi-weighted** energy games, each move potentially affects multiple resources simultaneously, adding complexity to the game. We only consider the simple case where one player wins all infinite plays, allowing for an understanding of these games as multi-weighted reachability games [12].
In recent concurrency research, Bisping, Nestmann and Jansen [10, 9, 8] generalised the bisimulation game in order to simultaneously decide all common notions of behavioural equivalences. Bisping [7] developed this into a multi-weighted energy game and, by doing so, garnered our interest for a class of energy games allowing updates other than vector addition such that the energy level never increases – we will call those games *Bisping's **declining** energy games*. This thesis studies the decidability of Bisping's declining energy games and generalises these games by abstracting the properties of updates and introducing new energy games classes.

Decidability of (energy) games refers to the question of whether there exists an algorithm that can determine, in finite time, whether a player has a winning strategy for a given game configuration. In energy games this depends on the initial energy. Bisping provides an algorithm, which, he claims, decides Bisping's declining energy games without giving a full proof. This thesis adds such a proof and generalises the algorithm to decide a larger class of energy games.
With **formal proof** we refer to a machine verified proof using Isabelle/HOL [22, 28]. Isabelle is a generic interactive proof assistant supporting the formalisation of mathematical theories. Theorems formalised in Isabelle undergo automated verification, ensuring that edge cases are not overlooked. While Isabelle supports multiple logical frameworks, we use the instantiation Isabelle/HOL based on Higher Order Logic (HOL). While we did not formalise all of our results, we give a formal proof of the decidability of Bisping's declining energy games using Isabelle/HOL. By discussing key components of our formalisation, we substantiate the correspondence of the formalised results with a proof of the decidability claimed by Bisping.

Contributions

We now outline our contributions and at the same time present the structure of this thesis.
In Chapter 2 we introduce notation, discuss partial orders and summarise fundamental game-theoretic concepts.
Following that, we delve into the theory of energy games in Chapter 3. In particular, we discuss an inductive characterisation of *attacker winning budgets*, the sets of energies sufficient for the attacker to win the game from each starting position. This inductive characterisation and attacker winning budgets being upward-closed for monotonic energy games are central for our proof of decidability. For this reason, we provide a formalisation of these results in Isabelle/HOL.
In Chapter 4, we discuss Bisping's declining energy games. While studying Bisping's inversion of updates, we show that each such inversion and update form a *Galois connection* between the set of energies and the update's domain. This property can be understood as a weakened form of invertibility of updates. We then introduce a new class of energy games, which we name *Galois energy games*, where all updates satisfy this weakened form of invertibility. Further, we provide a formalisation in Isabelle/HOL of both, Bisping's declining energy games and of the fact that they form a subclass of Galois energy games.
In Chapter 5, we provide a simplification of Bisping's algorithm and use this to prove the main result of this thesis: Galois energy games are decidable if the set of positions is finite, the energies form a well-founded (bounded) join-semilattice and we further make some reasonable computability assumptions. This implies the decidability of Bisping's declining energy games with finite positions, verifying Bisping's claim. For the special case of Bisping's declining energy games we further provide a formalisation of the proof in Isabelle/HOL. We end the chapter with a complexity analysis. For some fixed-dimension cases we show the running time to be polynomial and thereby improve the upper bounds provided by Bisping as well as those stated by Brihaye and Goeminne [12].
While we are able to show undecidability for the class of monotonic energy games, our main result indicates that monotonicity is more relevant to prove

decidability than updates being declining is. We investigate this in Chapter 6. Defining a new class of energy games by some reasonable computability assumptions we obtain an even larger class of decidable energy games, which we name *generalised Galois energy games*. In the same chapter we investigate the importance of the assumptions we make about energies and winning conditions. We then suggest further possible generalisations and contextualise our results by discussing existing literature on related games and their decidability.
Finally, we give a conclusion of this thesis in Chapter 7. There we will revisit a version of the following figure summarising the classes of energy games intoduced in this thesis and their decidability.

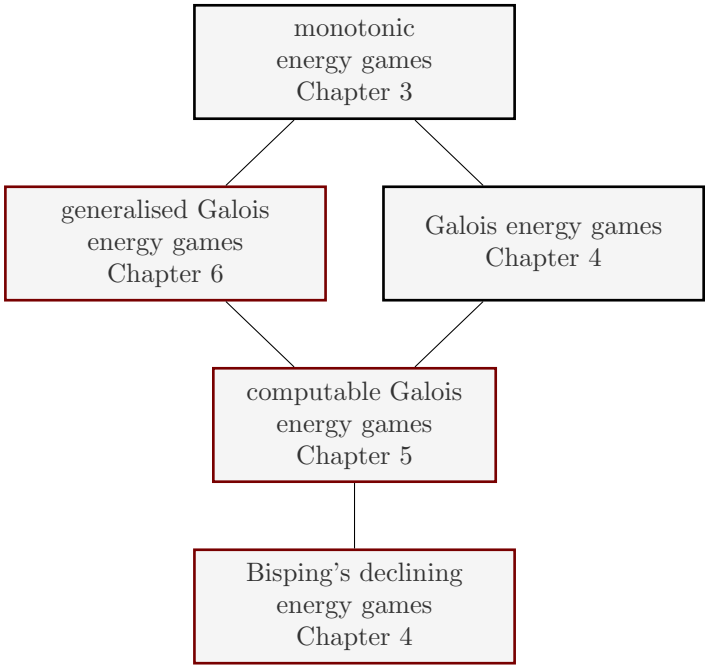

Figure 1.1: Energy game classes

In Figure 1.1 the chapters indicate the classes' first appearances.[1] The class hierarchies are visualised such that a subclass is below its superclass. Those hierarchies are established in Chapter 6 when Figure 6.1 is discussed. The classes with *Galois* in their name are new classes of energy games, which we introduced specifically to generalise Bisping's claim of decidability. Red indicates that we provide a proof of decidability for that class – if the set of positions is finite and the energies form a well-founded join-semilattice. An overview of the (un)decidability results is given in Chapter 7 when discussing Figure 7.1.

[1] Note that computable Galois energy games are discussed in Chapter 5 but are not formally introduced until Chapter 6.

Chapter 2

Preliminaries

With this chapter we lay the foundation for the concepts and terminology used throughout this thesis. We begin by introducing the basic notation in Section 2.1. Afterwards, we discuss different kinds of partial orders in Section 2.2. We conclude this chapter by stating important game-theoretic definitions in Section 2.3.

2.1 Notation

With this section we aim to provide clarity on the used symbols and terminology.

Notation 2.1 (Sets)**.** For a set A we denote the power set of A by $\mathcal{P}(A)$. We write $A \uplus B$ for the disjoint union of sets A and B and $A \backslash B$ for the relative complement of B w.r.t. A. With $|A|$ we refer to the cardinality of a set A.

Notation 2.2 (Extended naturals)**.** We write $\mathbb{N}$ for the set of natural numbers including 0. Further, we write $\mathbb{N}_\infty$ for the extended natural numbers. Those refer to the set $\mathbb{N}_\infty := \mathbb{N} \uplus \{\infty\}$ with the addition $+ : \mathbb{N}_\infty \times \mathbb{N}_\infty \to \mathbb{N}_\infty$ extending the addition on the natural numbers in the obvious way, i.e. $i + \infty = \infty + i = \infty$ for all $i \in \mathbb{N}_\infty$. Further, the usual order on natural numbers is extended by setting $i \leqslant \infty$ for all $i \in \mathbb{N}_\infty$.

Notation 2.3 (Functions)**.** Let $f : A \to B$ be a partial function between sets A and B. We denote the domain of f by $\mathrm{dom}(f)$ and write $f(a)$ and $f(A')$ for the image of an element $a \in A$ (if defined) and the image of a subset $A' \subseteq A$ respectively. If A is a Cartesian product, i.e. $A = A_0 \times ... \times A_n$ for sets $A_0, ..., A_n$ and $n \in \mathbb{N}$, we identify $f(a_0, ..., a_n) := f(a)$ for $a = (a_0, ..., a_n) \in A$ with $a_i \in A_i$ for $i \in \{0, ..., n\}$. We sometimes write $(b_a)_{a \in A} \subseteq B$ instead of $f : A \to B$ where we identify $b_a := f(a)$ for all $a \in \mathrm{dom}(f)$. If A and B are clear from the context, we might write b_a for $a \in \mathrm{dom}(f)$ without explicitly specifying f, A or B. Let C be a set and $g : B \to C$ be a partial function, then we write $g \circ f$ for the composition of g and f. Further, we write $f(\cdot)$ to denote $\lambda a.f(a)$.

Notation 2.4 (Words)**.** For any set A we write A^* for the set of finite words over A, while we denote the set of infinite words over A by A^ω. We denote the empty word with ϵ. Further, we write $|a|$ for length of a word $a \in A^* \cup A^\omega$, i.e. $|\cdot| : A^* \cup A^\omega \to \mathbb{N}_\infty$. We denote the concatenation of two words $a_1 \in A^*$

and $a_2 \in A^* \cup A^\omega$ by $a_1 a_2$ and write $A_1 A_2$ for the set of concatenations of the sets A_1 and A_2. With a^ω we refer to the word constructed from infinitely often appending $a \in A^*$ to a.

Further notation is introduced when needed.

2.2 Partial Orders

Throughout this thesis we will consider various partially ordered sets. In particular, we will examine the properties of partially ordered sets we consider as energies. For this reason we now briefly summarise fundamental concepts and define some notation along the way. For more details on (well-founded) relations we refer to Fraisse [20] and refer to Gratzer [23] for more insights into lattice theory.

Definition 2.1 (Partial order)**.** *Let P be a set and $\leqslant_P \subseteq P \times P$ be a relation on P. We identify $p \leqslant_P p'$ with $(p, p') \in \leqslant_P$ and $p \not\leqslant_P p'$ with $(p, p') \notin \leqslant_P$ for all $p, p' \in P$. Then $(P, \leqslant_P)$ is a partially ordered set if the following three conditions hold.*

- *The relation $\leqslant_P$ is reflexive, i.e. $p \leqslant_P p$ holds for all $p \in P$.*
- *The relation $\leqslant_P$ is transitive, i.e. $(p \leqslant_P p' \wedge p' \leqslant_P p'') \longrightarrow p \leqslant_P p''$ holds for all $p, p', p'' \in P$.*
- *The relation $\leqslant_P$ is antisymmetric, i.e. $(p \leqslant_P p' \wedge p' \leqslant_P p) \longrightarrow p = p'$ holds for all $p, p' \in P$.*

We call elements $p, p' \in P$ comparable, if $p \leqslant_P p' \vee p' \leqslant_P p$. Conversely, we call p and p' incomparable, if $p \not\leqslant_P p' \wedge p' \not\leqslant_P p$. A subset of pairwise incomparable elements is called an antichain. A partially ordered set is totally ordered, if any two elements are comparable. Totally ordered subsets are called chains. The strict partial order $<_P$ associated with $\leqslant_P$ is defined for all $p, p' \in P$ by $p <_P p'$ if and only if $p \leqslant_P p' \wedge p \neq p'$.

We call the inverse or opposite of a partial order its dual and observe the following.

Lemma 2.1. *Let $(P, \leqslant_P)$ be a partially ordered set. Then the dual of $\leqslant_P$, i.e. $\{(p, p') \in P \times P \mid p' \leqslant_P p\}$, is a partial order on P.*

Minimal Elements

On partially ordered sets we can define a notion of minimality.

Notation 2.5 (Min)**.** Let $(P, \leqslant_P)$ be a partially ordered set and $A \subseteq P$. We call $\operatorname{Min} A = \{m \in A \mid \forall a \in A.\ a \not\leqslant_P m\}$ the set of minimal elements in A w.r.t $\leqslant_P$. To indicate the respective order we sometimes write $\operatorname{Min}_{\leqslant_P}$ or Min_P.

Then the following observation follows directly.

Lemma 2.2. *Let $(P, \leqslant_P)$ be a partially ordered set and $A \subseteq B \subseteq P$. Then $(A \cap \operatorname{Min}_P B) \subseteq \operatorname{Min}_P A$ and thereby $\operatorname{Min}_P B \subseteq A \longrightarrow \operatorname{Min}_P B = \operatorname{Min}_P A$.*

Notation 2.6. Let $(P, \leqslant_P)$ be a partially ordered set and $A \subseteq P$. If $\operatorname{Min}_P A$ contains exactly one element, we call that element the minimum of A and denote the partial function mapping subsets of P to its minimum (if it exists) as $\min : \mathcal{P}(P) \to P$. Again we sometimes indicate the respective order by writing $\min_{\leqslant_P}$ or $\min_P$. In particular, we write $\min_{\mathbb{N}}$ when referring to the minimum w.r.t. the usual order on natural numbers.

We observe that in totally ordered sets a minimum often exists.

Lemma 2.3. *Let $(P, \leqslant_P)$ be a totally ordered set and $\varnothing \neq A \subseteq P$ be finite. Then $\operatorname{Min}_P A$ contains exactly one element.*

We now introduce a notion of maximality.

Notation 2.7 (Max)**.** Let $(P, \leqslant_P)$ be a partially ordered set and $A \subseteq P$. Analogously to Min_P and $\min_P$ we refer to the set of maximal elements of A as $\operatorname{Max} A = \{m \in A \mid \forall a \in A.\ m \nleqslant_P a\}$. Further, we call $m \in \operatorname{Max} A$ the maximum of A, if $\operatorname{Max} A$ contains exactly one element and denote the partial function mapping subsets of P to its maximum (if it exists) as $\max : \mathcal{P}(P) \to P$. We indicate the respective order as we do for the minimum.

The quasi-inverse of Min is the upwards-closure. Before introducing the latter we give a definition of quasi-inverse.

Definition 2.2 (Quasi-inverse)**.** *Let $f : A \to B$ and $g : B \to A$. Then g is a quasi-inverse of f, if $f \circ g \circ f = f$.*

Notation 2.8 (Upward-closure)**.** Let $(P, \leqslant_P)$ be a partially ordered set. For a subset $P' \subseteq P$ we denote the upward-closure of P' by $\uparrow P' := \{p \in P \mid \exists p_m \in P'.\ p_m \leqslant_P p\}$. We call P' upward-closed if $P' = \uparrow P'$. We indicate the respective order as we do for the minimum.

The following lemma justifies us previously calling the upward-closure a quasi-inverse of Min. The proof follows directly from Notation 2.5 and Notation 2.8.

Lemma 2.4. *Let $(P, \leqslant_P)$ be a partially ordered set. Then the following equalities hold for all $P' \subseteq P$.*

a) $\operatorname{Min}_P (\operatorname{Min}_P P') = \operatorname{Min}_P P'$

b) $\uparrow (\uparrow P') = \uparrow P'$

c) $\operatorname{Min}_P (\uparrow P) = \operatorname{Min}_P P'$

d) $\uparrow (\operatorname{Min}_P P') = \uparrow P'$

When considering Min_P and $\uparrow$ as functions of type $\mathcal{P}(P) \to \mathcal{P}(P)$, then $\operatorname{Min}_P \circ \uparrow \circ \operatorname{Min}_P = \operatorname{Min}_P$ and $\uparrow \circ \operatorname{Min}_P \circ \uparrow = \uparrow$.

Lattices

We now define lattices. This can be understood as abstracting the concept of minimality and maximality to the existence of lower and upper bounds respectively.

Definition 2.3 (Lattice)**.** *Let $(P, \leqslant_P)$ be a partially ordered set. We consider the following two properties.*

- *For all $p, p' \in P$ there exists a greatest lower bound $i \in P$, i.e. $i \leqslant_P p \wedge i \leqslant_P p'$ and for all $i' \in P$ with this property we have $i' \leqslant_P i$. We then call $(P, \leqslant_P)$ a meet-semilattice.*
- *For all $p, p' \in P$ there exists a smallest upper bound $s \in P$, i.e. $p \leqslant_P s \wedge p' \leqslant_P s$ and for all $s' \in P$ with this property we have $s \leqslant_P s'$. We then call $(P, \leqslant_P)$ a join-semilattice.*

A lattice is both, a meet-semilattice and a join-semilattice. We call a lattice $(P, \leqslant_P)$ bounded if there exist a minimum and maximum in P. We call a meet-semilattice (resp. join-semilattice) bounded, if a maximum (resp. minimum) exists.
Further, $(P, \leqslant_P)$ is a complete lattice if the two conditions hold for all subsets $P' \subseteq P$, i.e. if the following holds.

- *There exists a greatest lower bound $i \in P$, i.e. $\forall p \in P'.\ i \leqslant_P p$ and for all $i' \in P$ with this property we have $i' \leqslant_P i$.*
- *There exists a smallest upper bound $s \in P$, i.e. $\forall p \in P'.\ p \leqslant_P s$ and for all $s' \in P$ with this property we have $s \leqslant_P s'$.*

Notation 2.9 (infimum and supremum)**.** Let $(P, \leqslant_P)$ be a partially ordered set. We then call the partial function mapping subsets $P' \subseteq P$ to the greatest lower bound of P' in P (if it exists) the infimum and denote it with inf. Analogously we call the partial function mapping subsets $P' \subseteq P$ to the smallest upper bound of P' in P (if it exists) the supremum and denote it with sup. We indicate the respective order as we do for the minimum.

We can characterise the minimum and maximum using the empty set as follows.

Remark 2.1. Let $(P, \leqslant_P)$ be a bounded lattice. Then $\sup_P \varnothing$ is the minimum of P and $\inf_P \varnothing$ is the maximum P. If $(P, \leqslant_P)$ is a complete lattice, then $(P, \leqslant_P)$ is bounded with $\sup_P \varnothing = \inf_P P$ and $\sup_P P = \inf_P \varnothing$.

Further, we can characterise lattices as follows.

Lemma 2.5. *Let $(P, \leqslant_P)$ be a partially ordered set. Then $(P, \leqslant_P)$ is a meet-semilattice (resp. join-semilattice) if and only if all finite non-empty sets $P' \subseteq P$ have a greatest lower bound in P (resp. a smallest upper bound in P). Further, $(P, \leqslant_P)$ is a complete lattice if and only if all sets $P' \subseteq P$ have a greatest lower bound in P and a smallest upper bound in P.*

Example 2.1. The natural numbers with the usual order form a lattice, while the extented naturals form a complete lattice. Note that the natural numbers also form a bounded join-semilattice, since 0 is the minimum in $\mathbb{N}$.

We now consider orders that are not just partially ordered but not necessarily (complete) lattices, namely directed-complete partial orders.

Definition 2.4 (Directed-complete partial order)**.** *Let $(P, \leqslant_P)$ be a partially ordered set. A set $P' \subseteq P$ is directed, if all pairs of elements in P' have an upper bound in P', i.e. $\forall p, p' \in P'.\ \exists p'' \in P.\ p \leqslant_P p'' \wedge p' \leqslant_P p''$. Then $(P, \leqslant_P)$ is a directed-complete partially ordered set, if there exists a smallest upper bound of P' in P for all directed sets $P' \subseteq P$.*

The definition of directed sets directly implies the following.

Lemma 2.6. *Let $(P, \leqslant_P)$ be a partially ordered set and let $\varnothing \neq P' \subseteq P$ be directed. For a finite subset $P'' \subseteq P'$ there exists an upper bound $p \in P'$ for P'', i.e. $\forall p' \in P''.\ p' \leqslant_P p$.*

Well-founded Partial Orders

We now define well-founded orders to later use well-founded induction.

Definition 2.5 (Well-founded partial order)**.** *Let $(P, \leqslant_P)$ be a partially ordered set. Then $\leqslant_P$ is well-founded on $P' \subseteq P$ if every nonempty subset $P'' \subseteq P'$ has at least one minimal element, i.e. $\exists m \in P''.\ \forall p \in P''.\ m \neq p \longrightarrow p \not\leqslant_P m$.*

To simplify later proofs, we establish the following equivalences.

Lemma 2.7. *Let $(P, \leqslant_P)$ be a partially ordered set and $P' \subseteq P$. Then the following are equivalent.*

(i) The partial order $\leqslant_P$ is well-founded on P'.

(ii) There is no strictly decreasing infinite sequence in P'.

(iii) For every sequence $(p_i)_{i \in \mathbb{N}}$ in P' there exist $i, j \in \mathbb{N}$ such that $i < j$ and $p_i \leqslant_P p_j$.

Proof by citation. For a proof of the equivalence of (i) and (ii) we refer to section 2.4 in [20, pp. 35-36] while the equivalence of (i) and (iii) is stated in Theorem 1.2 in [24, pp. 195-196]. □

Note that (iii) in Lemma 2.7 directly implies that there are no infinite antichains in a well-founded partially ordered set. Since the minimal elements of a subset of a partially ordered set form an antichain by definition, this implies the following lemma.

Lemma 2.8. *Let $\leqslant_P$ be a well-founded partial order on P. Then $\operatorname{Min}_P A$ is finite for all $A \subseteq P$.*

Another consequence of Lemma 2.7 is well-founded induction, which we introduce as the following lemma.

Lemma 2.9. *Let $\leqslant_P$ be a well-founded partial order on P and Q be a property where Qx denotes that $x \in P$ has the property Q. Then Qx holds for all elements of P if*

$$\forall x.\, (\forall y.\ y <_P x \longrightarrow Qy) \longrightarrow Qx.$$

2.3 Games

In this section we introduce games and fundamental game-theoretic concepts such as plays, strategies and positional determinacy. For more details we refer to Kreutzer and Rabinovich [26]. Since games are played on directed graphs, we start by giving a definition of those.

Definition 2.6 (Directed graph, walk)**.** *A directed graph $\mathcal{G} = (V, E)$ consists of*

- *a set of vertices V, and*
- *a set of edges $E \subseteq V \times V$.*

Then $w = w_0 w_1 ... \in V^* \cup V^\omega$ is a walk in $\mathcal{G}$ if $(w_i, w_{i+1}) \in E$ for all $i \in \mathbb{N}$ with $i + 1 < |w|$.

Notation 2.10. Let $\mathcal{G} = (V, E)$ be a directed graph. For $v \in V$ we call vertices in the set $\{v' \mid (v, v') \in E\}$ successors of v, while $\{v' \mid (v', v) \in E\}$ is the set of predecessors of v.

Now we define games. Note that we only consider two-player zero-sum games with perfect information in which every maximal play is won by one of the two players.

Definition 2.7 (Game)**.** *A game $\mathcal{G} = (G, G_0, E, \Omega)$ consists of*

- *a set of positions G, partitioned into positions of player 0 denoted by $G_0 \subseteq G$ and positions of player 1 denoted by $G_1 := G \backslash G_0$,*
- *a set of edges $E \subseteq G \times G$ and*
- *a winning condition $\Omega \subseteq G^\omega$.*

We call $\mathcal{G}$ simple if $\Omega = \varnothing$ or $\Omega = G^\omega$.

Notation 2.11. Let $\mathcal{G} = (G, G_0, E, \Omega)$ be a game. To denote an initial position $g_0 \in G$ we write $\mathcal{G}[g_0]$.

We now define what a play is, when a play ends and who wins a play. In doing so it becomes clear, what is meant by games being played on directed graphs.

Definition 2.8 (Play)**.** *A play in G is a (finite or infinite) walk in the directed graph (G, E). A play ρ has ended if ρ is infinite or $\rho = g_0 ... g_n$ and the last position g_n has no successors. We then call g_n a deadend. An infinite play ρ is won by player 0 if $\rho \in \Omega$, while player 1 wins if $\rho \notin \Omega$. A finite ended play $\rho = g_0 ... g_n$ is won by player 0 if $g_n \in G_1$, while player 1 wins if $g_n \in G_0$.*

After defining what it means to win a play, we can now define what it means to win a game depending on a starting position. A player wins a game, if they can always enforce a winning play from the initial position.

Definition 2.9 (Strategy)**.** *Let $\mathcal{G} = (G, G_0, E, \Omega)$ be a game and $G_p \subseteq G$. A strategy is a partial mapping $s : G^* G_p \to G$ such that s is defined for all walks $\rho = g_0 ... g_n \in G^* G_p$ where g_n is not a deadend and $(g_n, s(\rho)) \in E$ holds. If $G_p = G_0$ (resp. $G_p = G_1$) we call s a strategy for player 0 (resp. player 1). A walk $\rho = g_0 g_1 ...$ is consistent with s if $g_i \in G_p$ implies $g_{i+1} = s(g_0 ... g_i)$ for all $i < |\rho|$. A strategy for player 0 (resp. player 1) is called a winning strategy w.r.t an initial position $g \in G$ if all ended plays starting in g and consistent with the strategy are won by player 0 (resp. player 1). We then say $\mathcal{G}[g]$ is won by player 0 (resp. player 1). The strategy s is called positional if $s(g_0 ... g_n) = s(g'_0 ... g'_m)$ for all $g_0 ... g_n, g'_0 ... g'_m \in G^* G_p$ with $g_n = g'_m$. Then s can be understood as a function $s : G_p \to G$.*

If there is a winner, we call a game determined. More formally we define this as follows.

Definition 2.10 (Determined game, winning region)**.** *Let $\mathcal{G} = (G, G_0, E, \Omega)$ be a game. Then $\mathcal{G}[g]$ is (positionally) determined if player 0 or player 1 has a (positional) winning strategy w.r.t $g \in G$. The winning region of player 0 is defined as $W_0 := \{g \in G \mid \mathcal{G}[g]$ is won by player $0\}$ and analogously $W_1 := \{g \in G \mid \mathcal{G}[g]$ is won by player $1\}$ is the winning region of player 1.*

We now introduce a class of games, namely parity games, where all positions are assigned a priority. Player 0 wins an infinite play in a parity game, if the smallest priority appearing infinitely often is even.

Definition 2.11 (Parity game)**.** *We call a game $\mathcal{G} = (G, G_0, E, \Omega)$ a parity game if there is a function $r : G \to \mathbb{N}$ such that the image of r is finite and $\Omega = \{g_0 g_1 ... \in G^\omega \mid \min_{\mathbb{N}}\{i \in \mathbb{N} \mid r(g_j) = i$ for infinitely many $j \in \mathbb{N}\}$ is even$\}$. For $g \in G$ we then call $r(g)$ the priority of g. If $\mathcal{G} = (G, G_0, E, \Omega)$ is a parity game with rank function r we identify $\mathcal{G}$ with (G, G_0, E, r).*

Now we are able to restate the well-known fact that parity games are positionally determined.

Theorem 1 (Positional determinacy of parity games)**.** *Let $\mathcal{G} = (G, G_0, E, r)$ be a parity game. Then $\mathcal{G}[g]$ is positionally determined for all $g \in G$. In particular $G = W_0 \uplus W_1$ holds.*

This theorem was formalised by Dittmann [17] following the proof given by Zielonka [33].

Chapter 3

Energy Games

Various definitions of energy games have emerged within research. We fix the winning conditions by focussing on reachability games over arbitrary energies. The winner of an energy games depends on the starting position and the initial energy. For each starting position we consider the set of energies that suffice to win the game – we call these sets winning budgets. We determine winners by calculating winning budgets with the algorithm given later utilising an inductive characterisation. In this chapter we introduce energy games by providing fundamental definitions of energy games, plays and winning strategies in Section 3.1. In Section 3.2 we provide an inductive characterisation of winning budgets and in Section 3.3 we discuss key parts of the formalisation of energy games and attacker winning budgets in Isabelle/HOL.[2]

3.1 Energy Games

Similarly to the games previously defined, energy games too are played on directed graphs with a partition of vertices.[3] Additionally, there is a weight function mapping edges to partial functions of energies – we call those functions updates. We start with a more general definition by not fixing a set of energies and instead considering an arbitrary partially ordered set. Further, we do not yet restrict the updates that may appear in an energy game. While some of the following definitions closely resemble those from Section 2.3, the winning conditions differ. In particular, we only consider the simple case where one player wins all infinite plays. Thereby, the considered winning conditions can be understood as reachability winning conditions.

Definition 3.1 (Energy Game)**.** *Let $(\mathcal{E}, \leqslant)$ be a partially ordered set. An energy game $\mathcal{G} = (G, G_a, \rightarrowtail)$ over the set of energies $\mathcal{E}$ consists of*

- *a set of positions G, partitioned into attacker positions $G_a \subseteq G$ and defender positions $G_d := G \backslash G_a$, and*
- *a partial function $\rightarrowtail : (G \times G) \rightharpoonup (\mathcal{E} \rightharpoonup \mathcal{E})$ mapping edges to partial functions of energies.*

[2] Formalised lemmas that are not discussed in Section 3.3 are pointed out in footnotes.

[3] Note that by requiring a partition of vertices into two (possibly empty) sets we only consider (one- and) two-player games.

Assumption 3.1. Let $(\mathcal{E}, \leqslant)$ be a partially ordered set and let $\mathcal{G} = (G, G_a, \rightarrowtail)$ be an energy game over $\mathcal{E}$.[4]

Notation 3.1. Let $g, g' \in G$. We say there is an edge between g and g' if $(g, g') \in \mathrm{dom}(\rightarrowtail)$. Then we write $g \overset{u}{\rightarrowtail} g'$ instead of $\rightarrowtail (g, g') = u$ and call u an update in $\mathcal{G}$. To denote an initial position g_0 and an initial energy e_0 we write $\mathcal{G}[g_0, e_0]$. We call an energy game n-dimensional if $\mathcal{E}$ is n-dimensional.

Remark 3.1. The underlying directed graph of an energy game is the graph $(G, \mathrm{dom}(\rightarrowtail))$. Energy games can be represented as labelled graphs by interpreting $g \overset{u}{\rightarrowtail} g'$ as an edge between vertices g and g' labelled by u.

Example 3.1. The following one-dimensional energy game is represented by Figure 3.1 as described in Remark 3.1. Note that attacker positions are rectangular while defender positions are round. Let $\mathcal{G}' = (G', G'_a, \rightarrowtail)$ be an energy game over $\mathbb{N}$ with positions $G' = \{g_0, g_1, g_2\}$ and attacker positions $G'_a = \{g_1\}$. Then $\rightarrowtail$ is given by $g_0 \overset{+2}{\rightarrowtail} g_1$, $g_1 \overset{-1}{\rightarrowtail} g_0$, $g_1 \overset{-1}{\rightarrowtail} g_1$ and $g_1 \overset{-2}{\rightarrowtail} g_2$, where $z \in \mathbb{Z}$ represents the function $f : \mathbb{N} \to \mathbb{N},\ e \mapsto e + z$ with $\mathrm{dom}(f) = \{e \in \mathbb{N} \mid 0 \leqslant e + z\}$.

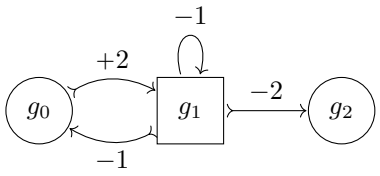

Figure 3.1: Representation of an energy game as described in Example 3.1

Analogous to Definition 2.8 we now define plays as walks.

Definition 3.2 (Play)**.** *A play is a (finite or infinite) walk $\rho = g_0 g_1 ... \in G^* \cup G^\omega$ in the underlying directed graph.*

The energy level can be understood as a tool to track resources during a play. The energy level of a play is defined by repeatedly updating the current energy according to the edges in the play. Since updates are partial functions, when calculating the energy level we might want to apply an update to something undefined. We handle this by introducing $\bot$ for undefined energies as seen in the following definition.

Definition 3.3. *Let $\bot \notin \mathcal{E}$. For any partial function $u : \mathcal{E} \to \mathcal{E}$ we define a total function $u' : \mathcal{E} \uplus \{\bot\} \to \mathcal{E} \uplus \{\bot\}$ by setting $u'(e) := \bot$ for all $e \in (\mathcal{E} \uplus \{\bot\}) \setminus \mathrm{dom}(u)$.*

Assumption 3.2. From now on we assume $\bot \notin \mathcal{E}$.

Notation 3.2. For all updates u in $\mathcal{G}$ we identify $u(e)$ being undefined for some $e \in \mathcal{E}$ and $u(e) = \bot$ in accordance with Definition 3.3.

Now we are able to define the energy level of plays depending on an initial energy.

[4]Note that we use $\leqslant$ with no index on energies. We do not index the minimum and supremum on $\mathcal{E}$ either.

Definition 3.4 (Energy level)**.** *The energy level is a partial function* $EL : (G^* \cup G^\omega) \times (\mathcal{E} \uplus \{\bot\}) \times \mathbb{N} \to \mathcal{E} \uplus \{\bot\}$ *recursively defined by* $EL(\rho, e_0, 0) := e_0$ *and* $EL(\rho, e_0, i+1) := u(EL(\rho, e_0, i))$ *with* $g_i \overset{u}{\rightarrowtail} g_{i+1}$ *and* $i + 1 < |\rho|$ *for* $\rho \in G^* \cup G^\omega$, $e_0 \in \mathcal{E}$ *and* $i \in \mathbb{N}$. *The final energy level is a partial function* $EL : (G^* \cup G^\omega) \times (\mathcal{E} \uplus \{\bot\}) \to \mathcal{E} \uplus \{\bot\}$ *with* $EL(\rho, e_0) := EL(\rho, e_0, |\rho| - 1)$ *if* $\rho \in G^* \cup G^\omega$ *is finite and* $EL(\rho, e_0) := \bot$ *if* ρ *is infinite for all* $e_0 \in \mathcal{E}$.[5]

Notation 3.3. Let $\rho \in G^* \cup G^\omega$ and $e_0 \in \mathcal{E}$. We call $EL(\rho, e_0)$ the final energy level of ρ w.r.t. e_0 and omit the specification e_0 if it is clear from the context. Further, we call with the final energy level of ρ w.r.t. e_0 undefined if $EL(\rho, e_0) = \bot$ and defined if $EL(\rho, e_0) \neq \bot$.

Remark 3.2. Definition 3.4 implies that plays can be constructed by appending plays to plays, i.e. if there is an edge between the last position of a finite play ρ_1 and the first position of a play ρ_2, then $\rho_1\rho_2$ is a play. Furthermore, $EL(\rho_1\rho_2, e) = EL(\rho_2, EL(\rho_1\rho_2, e, |\rho_1|))$ holds.

By Definition 3.4 the final energy level of infinite paths is undefined. This is justified by our definition of won plays. We only consider the simple case where all infinite plays are won by one player, namely the defender. Furthermore, the attacker is energy-bound. Thereby an infinite play and the final energy level being undefined yield the same winner.

Definition 3.5 (Won play)**.** *Let* ρ *be a play in* $\mathcal{G}$ *and* $e \in \mathcal{E}$. *If* $g_l \in G_d$ *(resp.* $g_l \in G_a$*) is the last position of a finite play and* g_l *is a deadend, we call the defender (resp. attacker) stuck. The defender wins all infinite plays. Furthermore, the defender wins* ρ *w.r.t.* e *if the* $EL(\rho, e)$ *is undefined. If a play is finite, its final energy level is defined and one player is stuck, then the other player wins.*

Definition 3.5 is visualised by a decision diagram in Figure 3.2.

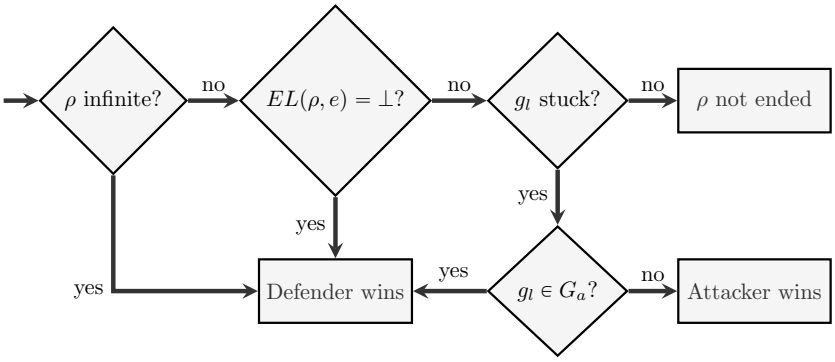

Figure 3.2: Winning conditions

Notation 3.4. Let $e \in \mathcal{E}$. We call a play maximal w.r.t e if the play is won by a player and call it play prefix w.r.t e otherwise.

[5]In other application it might make sense, to define the final energy level as a limit. For example Brihaye and Goeminne [12] do so when defining the cost function.

Remark 3.3. When considering maximal plays it is sufficient to study *minimal* maximal plays, i.e. maximal plays where no real prefix is maximal. In the following we therefore assume all plays to be a play prefix or a minimal maximal play.

As seen in Definition 2.9 we first introduce (winning) strategies before defining what it means for a player to win a game.

Definition 3.6 (Strategy and consistent play)**.** *Let $G_p \subseteq G$. A strategy $s : G^*G_p \to G$ is a partial function from play prefixes to a next position such that $s(g_0...g_n)$ is defined if $g_n \in G_p$ is not a deadend. Further, there then is an edge between g_n and $s(g_0...g_n)$. If $G_p = G_a$ (resp. $G_p = G_d$) we call s an attacker (resp. defender) strategy. A play $\rho = g_0g_1...$ is consistent with s if $g_i \in G_p$ implies $g_{i+1} = s(g_0...g_i)$ for all $i + 1 < |\rho|$.*

Since the property of being a play prefix depends on an energy, by Definition 3.6 the domain of strategies is also dependent on an energy. In the following definitions this relation will be more explicit.

Definition 3.7 (Winning strategy, won game and attacker winning budget)**.** *For an initial position g_0 and an initial energy e_0 an attacker (resp. defender) strategy s is called a winning attacker (resp. defender) strategy if all maximal plays starting in g_0 and consistent with s are won by the attacker (resp. defender) w.r.t. e_0. We then say $\mathcal{G}[g_0, e_0]$ is won by the attacker (resp. defender). The attacker winning budget of a position $g \in G$ is defined as $\mathsf{Win}_a(g) = \{e \mid \mathcal{G}[g, e]$ is won by the attacker$\}$. Analogously, the defender winning budget can be defined by $\mathsf{Win}_d(g) = \{e \mid \mathcal{G}[g, e]$ is won by the defender$\}$.*

We will later see that the winning budgets form a partition of the set of energies. Thereby, it is sufficient to study attacker winning budgets.

Example 3.2. The attacker winning budgets of $\mathcal{G}'$ from Figure 3.1 are

$$\mathsf{Win}_a(g_0) = \mathbb{N}, \quad \mathsf{Win}_a(g_1) = \mathbb{N}\backslash\{0\} \quad \text{and} \quad \mathsf{Win}_a(g_2) = \mathbb{N}.$$

That means the attacker wins all games $\mathcal{G}[g, e]$ where $(g, e) \neq (g_1, 0)$. Note that the defender wins $\mathcal{G}[g_1, 0]$.

By Definition 3.7 an attacker winning strategy wins against all defender strategies. This directly implies the following lemma.

Lemma 3.1. *No game $\mathcal{G}[g, e]$ with $g \in G$ and $e \in \mathcal{E}$ can be won by both the attacker and the defender, i.e. $\mathsf{Win}_a(g) \cap \mathsf{Win}_d(g) = \varnothing$ for all $g \in G$.*

When studying attacker winning budgets we observe that they form upward-closed sets if all updates are monotonic and their domains are upward-closed. We first give a name to this class of energy games.

Definition 3.8 (Monotonic energy game)**.** *The energy games $\mathcal{G}$ is a monotonic energy game if all updates u in $\mathcal{G}$ are monotonic and their domain is upward-closed, i.e. if $\forall e \in \operatorname{dom} u.\ \forall e' \in \mathcal{E}.\ e \leqslant e' \to e' \in \operatorname{dom}(u) \ \wedge\ u(e) \leqslant u(e')$ holds for all updates u in $\mathcal{G}$.*

We now state the upward-closedness of attacker winning budgets as the following lemma.

Lemma 3.2. *Let $\mathcal{G}$ be a monotonic energy game. Then the attacker winning budgets are upward-closed, i.e. $\uparrow \mathsf{Win}_a(g) = \mathsf{Win}_a(g)$ for all positions $g \in G$.*[6]

Proof sketch. Let $e \in \mathsf{Win}_a(g)$ and $e' \in \mathcal{E}$ with $e \leqslant e'$. Then $\mathcal{G}[g, e]$ is won by the attacker and an attacker winning strategy exists. That same strategy is an attacker winning strategy for $\mathcal{G}[g, e']$. □

Example 3.3. The energy game $\mathcal{G}'$ from Figure 3.1 is a monotonic energy game with upward-closed attacker winning budgets as seen in Example 3.2.

3.2 Inductive Characterisation of Winning Budgets

An energy is in the attacker winning budget of some positions, if there exists an attacker winning strategy. If that position is an attacker position there has to exists a winnable next position the strategy maps to. For defender positions every next position should be winnable for the attacker with the accordingly updated energy. This idea yields an inductive characterisation of attacker winning budgets. This section is dedicated to proving this characterisation stated as Theorem 3. First, we show that energy games are somewhat positionally determined.

3.2.1 Positional Determinacy

Example 3.2 illustrates that energy games are not positionally determined. In particular, there exists a position in $\mathcal{G}'$ that is neither always won by the attacker nor always won by the defender, namely g_1. The winner depends not only on the initial position but also on the initial energy. In this section we define *energy-positional* strategies, depending on the current position and current energy. It is then possible to transfer the positional determinacy of parity games as seen in Theorem 1 with slight modifications to energy games. We will show this by constructing a parity game from an energy game. For this we consider a graph connecting configurations in the following manner.
The positions in the parity game are pairs of positions of the energy game and energies. If there is an edge $g \rightarrowtail g'$ in the energy game, we add an edge between (g, e) and (g', e') for all energies updated accordingly, i.e. if $e' = u(e)$. By defining the rank to be constantly 0 and adding all positions (g, e) to player 1 where g is an attacker position or the energy is undefined, we obtain that Player 0 (resp. Player 1) winning in the parity game corresponds to the defender (resp. attacker) winning in the energy game. First we give a more formal definition of this construction.

Definition 3.9. *The parity game of $\mathcal{G}$ is $\mathcal{P}_\mathcal{G} := (G_\mathcal{G}, G_{\mathcal{G}0}, E_\mathcal{G}, r_\mathcal{G})$ where*

- $G_\mathcal{G} := \{(g, e) \mid g \in G, e \in \mathcal{E} \uplus \{\bot\}\}$,
- $G_{\mathcal{G}0} := G_\mathcal{G} \backslash G_{\mathcal{G}1}$ *with* $G_{\mathcal{G}1} := \{(g, e) \in G_\mathcal{G} \mid g \in G_a \vee e = \bot\}$,
- $E_\mathcal{G} := \{((g, e), (g', e')) \in G_\mathcal{G} \times G_\mathcal{G} \mid g \overset{u}{\rightarrowtail} g' \wedge e \neq \bot \wedge u(e) = e'\}$ *and*

[6] This lemma is formalised as `upward_closure_wb_nonpos` in Energy_Game.thy.

- $r_{\mathcal{G}} : G_{\mathcal{G}} \to \mathbb{N}, (g, e) \mapsto 0$.

Example 3.4. Consider the energy game $\mathcal{G}'$ from Example 3.1. From that we construct a parity game as seen in Definition 3.9. Figure 3.3 displays $\mathcal{G}'$ to the left and parts of the parity game of $\mathcal{G}'$ to the right. For each position in $\mathcal{G}'$ there are as many positions in $\mathcal{P}_{\mathcal{G}'}$ as there are elements in $\mathbb{N} \uplus \{\bot\}$, i.e. infinitely many. Therefore, we choose to only visualise four positions in $\mathcal{P}_{\mathcal{G}'}$ per position in $\mathcal{G}'$. Note that positions of the parity game are constructed from the position of the energy game at the same level. We visualise positions of player 1 rectangular while positions of player 0 are rounded. Since one can understand the construction as the attacker becoming player 1 and the defender becoming player 0, this is consistent with our display of $\mathcal{G}'$. Further, we colour-coded the edges, such that all edges in the parity game are constructed from the edge in the energy game in the same colour. Dotted edges start or end in a position we do not visualise.

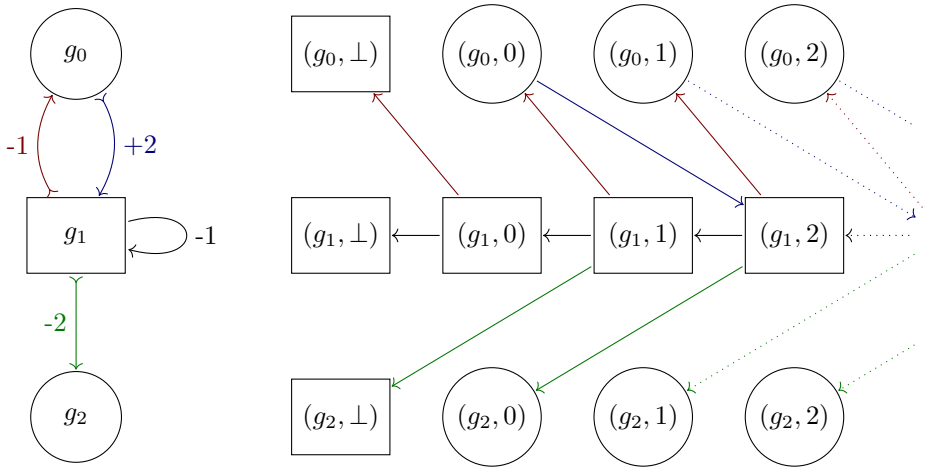

Figure 3.3: Representations of $\mathcal{G}'$ and the parity game of $\mathcal{G}'$ as described in Example 3.4

Definition 3.9 directly implies, that *stuckness* of the attacker is somewhat preserved. We state this as the following lemma.

Lemma 3.3. *A position (g, e) is a deadend in $\mathcal{P}_{\mathcal{G}}$ if and only if g is a deadend in $\mathcal{G}$ or $e = \bot$.*

Now we show that our construction yields the intended result.

Lemma 3.4. *For all energies $e \in \mathcal{E}$ and positions $g \in G$ player 0 (resp. player 1) winning $\mathcal{P}_{\mathcal{G}}$ implies that the defender (resp. attacker) wins $\mathcal{G}[g, e]$, i.e. $(g, e) \in W_0 \longrightarrow e \in \mathsf{Win}_d(g)$ and $(g, e) \in W_1 \longrightarrow e \in \mathsf{Win}_a(g)$.*

Proof. Let $e \in \mathcal{E}$ and $g \in G$. By Theorem 1 $\mathcal{P}_{\mathcal{G}}$ is positionally determined, thus there exists a positional winning strategy s for player 0 or player 1 in $\mathcal{P}_{\mathcal{G}}[(g, e)]$. Define $s_p : G^* G_p \to G$ by setting $s_p(g_0 ... g_n) := g'$ for $s\,(g_n, EL(g_0 ... g_n, e)) = (g', e')$. We show that s_p is a winning strategy in $\mathcal{G}[g, e]$ for $p = d$ or $p = a$ by case analysis.

Case "$(g, e) \in W_0$": Assume s to be a winning strategy for player 0. To show that s_d is a defender winning strategy we first show that s_d is a defender strategy. Let $\rho = g_0 ... g_n \in G^* G_d$ be a play prefix w.r.t e. By Definition 3.5 g_n is not stuck and the final energy level of ρ w.r.t e is defined, i.e. $EL(\rho, e) \in \mathcal{E}$. Hence, $(g_n, EL(\rho, e)) \in G_{\mathcal{G}0}$ is a position of player 0 and by Lemma 3.3 not stuck. Since s is a strategy for player 0, $s\,(g_n, EL(\rho, e))$ is defined and there is an edge from $(g_n, EL(\rho, e))$ to $s\,(g_n, EL(\rho, e))$ in $\mathcal{P}_\mathcal{G}$. By definition of $E_\mathcal{G}$ there also is an edge from g_n to $s_d(\rho)$ in $\mathcal{G}$. Thus s_d is a defender strategy.

Now we show that s_d is a defender winning strategy for $G[g, e]$. Let $\rho_g = g g_0 ...$ be a maximal play consistent with s_d. If ρ_g is infinite, the defender wins ρ_g. To show that the same holds in the finite case we define $\rho'_g := (g, e) g'_0 ...$ with $g'_i := (g_i, EL(\rho_g, e, i + 1)$ for all $i + 1 < |\rho_g|$. Then ρ'_g is a play in $\mathcal{P}_\mathcal{G}$ consistent with s by definition. If ρ_g is finite, the final energy level is undefined or one player is stuck since ρ_g is maximal. Then ρ'_g is finite and the last position is a deadend in $\mathcal{P}_\mathcal{G}$ by Lemma 3.3. Since we assumed s to be a winning strategy for player 0, the last position of ρ'_g then has to be in $G_{\mathcal{G}1}$. Thereby, the attacker is stuck or the final energy level of ρ_g is undefined. In any case the defender wins the play.

Case "$(g, e) \in W_1$": Assume s to be a winning strategy for player 1. The proof that s_a is a strategy is analogous to the previous case. We now show that s_a is an attacker winning strategy. Let $\rho_g = g g_0 ...$ be a maximal play consistent with s_a and define ρ'_g as before. If ρ_g is infinite, so is ρ'_g. Then ρ'_g is won by player 0 in $\mathcal{P}_\mathcal{G}$ by definition of $r_\mathcal{G}$. This is a contradiction to ρ'_g being consistent with s and s being a winning strategy for player 1. Therefore, ρ'_g is finite and so is ρ_g. Since ρ_g is maximal, the final energy level is undefined or one player is stuck. Hence the last position of ρ'_g is a deadend in $\mathcal{P}_\mathcal{G}$ by Lemma 3.3. Since player 1 wins ρ'_g, the last position of ρ'_g is in $G_{\mathcal{G}0}$. The definition of $G_{\mathcal{G}0}$ implies that the final energy level is defined and the defender is stuck at ρ_g. Thus the attacker wins the play. □

This implies, that energy games with a fixed initial position and initial energy are determined.

Corollary 3.1. *For all positions $g \in G$ and energies $e \in \mathcal{E}$ either the defender or the attacker wins $\mathcal{G}[g, e]$, i.e. $\mathcal{E} = \mathsf{Win}_a(g) \uplus \mathsf{Win}_d(g)$ for all $g \in G$.*

Proof. Let $g \in G$. For all $e \in \mathcal{E}$ Lemma 3.4 yields that $e \notin \mathsf{Win}_a(g)$ implies $(g, e) \notin W_1$. Then $(g, e) \in W_0$ by Theorem 1 and thereby $e \in \mathsf{Win}_d(g)$ by Lemma 3.4. Thus $e \in \mathsf{Win}_a(g) \cup \mathsf{Win}_d(g)$ holds for all $e \in \mathcal{E}$. Furthermore, $\mathsf{Win}_a(g) \cap \mathsf{Win}_d(g) = \varnothing$ by Lemma 3.1. □

Corollary 3.1 implies that for a fixed energy the positions can be partitioned into winning regions. This fact resembles the positional determinacy stated for parity games. To state the energy games version of positional determinacy, we first define the counterpart of positional strategies in (parity) games as seen in Definition 2.9.

Definition 3.10 (Energy-positional strategy)**.** *An energy-positional strategy $s : G_p \times \mathcal{E} \to G$ is a partial function such that $s(g, e)$ is defined if $g \in G_p$ is*

not a deadend. Further, there then is an edge between g and $s(g,e)$ for all $(g,e) \in G_p \times \mathcal{E}$. If $G_p = G_a$ (resp. $G_p = G_d$) we call s an energy-positional attacker (resp. defender) strategy. A play $\rho = g_0 g_1 ...$ is consistent with s w.r.t. e_0 if $g_i \in G_p$ implies $g_{i+1} = s(g_i, EL(\rho, e_0, i))$ for all $i + 1 < |\rho|$. For an initial position g_0 and an initial energy e_0 an energy-positional attacker (resp. defender) strategy s is called a winning energy-positional attacker (resp. defender) strategy if all maximal plays starting in g_0 with energy e_0 and consistent with s are won by the attacker (resp. defender).

Theorem 2 (Energy-positional determinacy)**.** *For all positions $g \in G$ and energies $e \in \mathcal{E}$ there exists an energy-positional winning strategy for the defender or attacker for $\mathcal{G}[g,e]$.*

Proof. Let $g \in G$ and $e \in \mathcal{E}$. Then $(g,e) \in W_0$ or $(g,e) \in W_1$ by Theorem 1. The winning strategies constructed in the proof of Lemma 3.4 only depend on the current energy and position. Therefore, constructing $s'_p : G_p \times \mathcal{E} \to G$ from a positional winning strategy s in $\mathcal{P}_\mathcal{G}$ such that $s_p(g,e) = g'$ where $(g', e') = s(g,e)$ yields an energy-positional winning strategy. □

3.2.2 Proof of Inductive Characterisation

We will prove the inductive characterisation of attacker winning budgets with well-founded induction. For this we introduce an order on pairs of positions and energies depending on an energy-positional strategy such that successors in a consistent play with the accordingly updated energy are smaller.

Definition 3.11 (Strategy induced order)**.** *Let s be an energy-positional attacker strategy. The order induced by s is defined by setting $(g', e') \leqslant_s (g,e)$ for $g, g' \in G$ and $e, e' \in \mathcal{E}$ if $g \rightarrowtail^u g'$, $e' = u(e)$ and $g \in G_a \longrightarrow s(g,e) = g'$ or if $(g', e') = (g,e)$.*

For an energy-positional attacker winning strategy this order is well-founded on the set of reachable positions in the configuration graph. We now define that set.

Definition 3.12 (Reachable position)**.** *Let s be an energy-positional strategy, $e \in \mathcal{E}$ and $g \in G$. Then the set of reachable positions $R(s,g,e)$ is the set of pairs $(g', e') \in G \times \mathcal{E}$ for which a play $\rho = g...g'$ starting in g, ending in g' and consistent with s exists such that $e' = EL(\rho, e)$.*

Lemma 3.5. *Let s be an energy-positional attacker winning strategy for $\mathcal{G}[g,e]$ with $e \in \mathcal{E}$ and $g \in G$. Then the order induced by s is well-founded on $R(s,g,e)$.*[7]

Proof sketch. For a proof by contradiction, we assume there exists a strictly decreasing sequence $((g_i, e_i))_{i \in \mathbb{N}} \subseteq R(s,g,e)$ such that $(g_{i+1}, e_{i+1}) <_s (g_i, e_i)$ holds for all $i \in \mathbb{N}$. Since $(g_0, e_0) \in R(s,g,e)$ there exists a play $\rho = g...g_0$ consistent with s such that $e_0 = EL(\rho, e)$. By Definition 3.11 $\rho' := \rho g_1 g_2 ...$ is an infinite play consistent with s. Since s is an attacker winning strategy for $\mathcal{G}[g,e]$, this is a contradiction. □

Now we can prove the inductive characterisation of attacker winning budgets.

[7] This lemma is formalised as `strategy_order_well_founded` in Energy_Game.thy.

Theorem 3 (Inductive characterisation of attacker winning budgets)**.** *The attacker winning budgets can be inductively characterised as follows:*

$$\frac{g \in G_a \qquad g \rightarrowtail^{u} g' \qquad u(e) \in \mathsf{Win}_a(g')}{e \in \mathsf{Win}_a(g)}$$

$$\frac{g \in G_d \qquad \forall g'.\ g \rightarrowtail^{u} g' \longrightarrow u(e) \in \mathsf{Win}_a(g')}{e \in \mathsf{Win}_a(g)}$$

Proof sketch. We show soundness and completeness separately, starting with soundness. Let $g \in G$ and $e \in \mathcal{E}$. If $g \in G_a$ and there exists $g' \in G$ with $g \rightarrowtail^{u} g'$ and $u(e) \in \mathsf{Win}_a(g')$, then we can show $e \in \mathsf{Win}_a(g)$ by constructing an attacker winning strategy from the existing attacker winning strategy s for $\mathcal{G}[g', u(e)]$. We construct s' by mapping g to g' such that ρ is consistent with s for all plays $g\rho$ consistent with s'. Then, s' is an attacker winning strategy for $\mathcal{G}[g, e]$ and thereby $e \in \mathsf{Win}_a(g)$. If $g \in G_d$ and $\forall g'.\ g \rightarrowtail^{u} g' \longrightarrow u(e) \in \mathsf{Win}_a(g')$ holds, then a similar construction using the at successors existing attacker winning strategies is possible, which implies $e \in \mathsf{Win}_a(g)$.
We now outline a proof of completeness. If $e \in \mathsf{Win}_a(g)$, then there is an energy-positional attacker winning stratetegy s for $\mathcal{G}[g, e]$ by Theorem 2. Since the order induced by s is well-founded on $R(s, g, e))$ by Lemma 3.5, a derivation of $e \in \mathsf{Win}_a(g)$ can be found by well-founded induction.

□

3.3 Fomalisation of Energy Games

With this section we conclude our chapter on energy games by discussing key elements of our formalisation in Isabelle/HOL of this chapters definitions and results. The complete formalisation is in Energy_Game.thy.

After Definition 3.1 we assumed $\mathcal{G} = (G, G_a, \rightarrowtail)$ to be an energy game over $\mathcal{E}$ in Assumption 3.1. Instead of formalising the definition we formalise this assumption by giving an energy game as a `locale`. All definitions and lemmas given in this section are then stated within the context `energy_game`. We formalise partial functions by utilising the datatype `option`.[8] This allows us to seamlessly switch between interpreting the output `None` as undefined or as $\bot$ as seen in Notation 3.2.[9]

```
locale energy_game =
  fixes attacker :: "'position set" and
        weight :: "'position ⇒ 'position ⇒ 'label option" and
        application :: "'label ⇒ 'energy ⇒ 'energy option"
```

Since Isabelle/HOL is typed, it is not necessary to give a set of positions. Instead of assuming $g \in G$ we will make statements about `g` of type `'position`.

[8] The datatype `option` is predefined and allows the addition of a distinct element, namely `None`, to a type `'a`. This becomes clear when looking at the definition: `datatype 'a option = None | Some 'a`.

[9] Using `option` in this way necessitates frequent use of `the`, a function with `the ∘ Some = id`.

Nonetheless, we later use the abbreviation `positions` to refer to the set of all positions. Note that we do not yet assume the energies to be partially ordered. Another notable deviation from Definition 3.1 is our formalisation of $\rightarrowtail$. Instead of directly labelling edges with partial functions of energies we introduce a type `'label`. This allows some encoding of updates which is decoded with `application`. Us writing z instead of $e \mapsto e + z$ in Example 3.1 is an example of using integers as labels to encode updates.

In Definition 3.2 we introduced plays. Those are formalised as coinductive lists of positions, i.e. as potentially infinite lists of type `'position llist`.[10] Thereby, we do not need a definition of plays and instead only define a predicate `valid_play` for needed checks whether all positions appearing consecutively in such a list are actually connected by edges. Here we only give the type declaration.

```
coinductive valid_play :: "'position llist ⇒ bool"
```

Now we give our formalisation of the energy level as defined in Definition 3.4. The purpose of the energy level is to check whether the defender has won. Since the defender automatically wins all infinite plays, we only ever apply the formalisation of the energy level to finite plays or play prefixes. Therefore, there is no need to formalise the final energy level separately.

```
fun energy_level:: "'energy ⇒ 'position llist ⇒ nat ⇒
          'energy option"
where
 "energy_level e p 0 = (if p = LNil then None else Some e)" |
 "energy_level e p (Suc i) = (if (energy_level e p i) = None ∨
       llength p ⩽ (Suc i) then None else
       apply_w (lnth p i)(lnth p (Suc i))(the (energy_level e p i)))"
```

Note that `apply_w g g'` abbreviates `application (the (weight g g'))` and `lnth p i` is the (`i`+1)-th entry of `p`.[11]

Now we formalise Definition 3.5 of won plays. Since by Theorem 2 every maximal play is either won by the defender or the attacker, a formalisation of `defender_wins_play` is sufficient.

```
definition defender_wins_play:: "'energy ⇒ 'position llist ⇒ bool"
where
  "defender_wins_play e p ≡ lfinite p ⟶
       (energy_level e p (the_enat (llength p)-1) = None ∨
       attacker_stuck p)"
```

Note that `attacker_stuck p` abbreviates the last position of `p` being an attacker position without outgoing edges and remember $EL(\rho, e) = EL(\rho, e, |\rho| - 1)$ for a finite play ρ. Since lists of type `'position llist` are potentially infinite, the function `llength` returns output of type

[10] The coinductive datatype `llist` is defined as follows.
`codatatype (lset: 'a) llist = lnull: LNil | LCons (lhd: 'a) (ltl: "'a llist")`
Those coinductive lists can be interpreted as lazy lists or `'a` streams.

[11] Note that indexing starts at 0.

`enat`, which represents the extended naturals. Then the function `the_enat` is used for typecasting.

Theorem 2 justifies only using energy-positional strategies. We formalised both, strategies that are energy-positional and those that are not. In our formalisation we add `nonpos` to definitions and lemmas to indicate that we refer to strategies that are not energy-positional. To avoid duplication we focus our discussion on energy-positional strategies, i.e. on Definition 3.10 instead of Definition 3.6. We formalise this by defining a predicate that is true if a given function is an energy-positional attacker strategy.

```
definition attacker_strategy:: "('energy ⇒ 'position ⇒
           'position option) ⇒ bool"
where
  "attacker_strategy s = (∀g e. (g ∈ attacker ∧ ¬ deadend g) ⟶
       (s e g ≠ None ∧ weight g (the (s e g)) ≠ None ))"
```

Before we can formalise attacker winning strategies, we first formalise what it means for a play to be consistent with a strategy as a predicate depending on a strategy, a play and an initial energy. For this we rephrase the condition

$$g_i \in G_a \text{ implies } g_{i+1} = s(g_i, EL(\rho, e_0, i)) \text{ for all } i < |\rho| - 1$$

for a play $\rho = g_0 g_1 ...$ to be consistent with a strategy s w.r.t. e_0 as stated in Definition 3.10. Unfolding the definition of energy games we instead give an inductive definition. In particular, a play $\rho = g_0 g_1 ...$ is consistent with a strategy s w.r.t. e_0 if

$$g_1 ... \textit{ is} \text{ consistent with } s \text{ w.r.t. } u(e_0) \text{ and } g_0 \in G_a \longrightarrow g_1 = s(g_0, e)$$

where $g_0 \rightarrowtail^u g_1$. We further define all plays of length 0 and 1 to be be consistent with any strategy w.r.t any energy.

```
coinductive play_consistent_attacker::"('energy ⇒ 'position ⇒
           'position option) ⇒ 'position llist  ⇒ 'energy ⇒ bool"
where
  "play_consistent_attacker _ LNil _" |
  "play_consistent_attacker _ (LCons v LNil) _" |
  "⟦play_consistent_attacker s Ps (the (apply_w v (lhd Ps) e));
    ¬lnull Ps; v ∈ attacker ⟶ (s e v) = Some (lhd Ps)⟧
    ⟹ play_consistent_attacker s (LCons v Ps) e"
```

While we do not assume that a list of positions has to be a valid play in order to be consistent with some strategy, we add that assumption to the lists considered in the formalisation of attacker winning strategies.

```
fun attacker_winning_strategy:: "('energy ⇒ 'position ⇒
           'position option) ⇒ 'energy ⇒ 'position ⇒ bool"
where
  "attacker_winning_strategy s e g = (attacker_strategy s ∧
       (∀p. (play_consistent_attacker s (LCons g p) e ∧
             valid_play (LCons g p)) ⟶
             ¬ defender_wins_play e (LCons g p)))"
```

Definition 3.10 defines an energy-positional attacker strategy s to be winning w.r.t initial position g_0 and initial energy e_0 if all maximal plays starting in g_0 with energy e_0 and consistent with s are won by the attacker. Since every play is either a prefix, won by the defender or won by the attacker, this is equivalent to no play starting in g_0 with energy e_0 and consistent with s being won by the defender.

Finally, we formalise attacker winning budgets as first introduced in Definition 3.7. Again we deviate by defining attacker winning budgets with energy-positional strategies as justified by Theorem 2 and formalise attacker winning budgets as a predicate.

```
fun winning_budget:: "'energy ⇒ 'position ⇒ bool" where
 "winning_budget e g = (∃s. attacker_winning_strategy s e g)"
```

To formalise Theorem 3, we first formalise the inductive definition of attacker winning budgets.

```
inductive winning_budget_ind:: "'energy ⇒ 'position ⇒ bool"
where
 "winning_budget_ind e g" if "g ∈ attacker ∧
       (∃g'. weight g g' ≠ None ∧ apply_w g g' e ≠ None ∧
        winning_budget_ind (the (apply_w g g' e)) g')" |
 "winning_budget_ind e g" if "g ∉ attacker ∧
       (∀g'. weight g g' ≠ None ⟶ (apply_w g g' e ≠ None ∧
        winning_budget_ind (the (apply_w g g' e)) g'))"
```

Note that this definition yields the least fixed point satisfying the characterisation given in Theorem 3. Since `the` is a total function `the None` is defined and when applying `the` it is necessary to check that `apply_w g g' e ≠ None` holds.

We do not formalise Theorem 2, which implies that there exists an attacker winning strategy if and only if there exists an energy-positional attacker winning strategy. Since we used this fact in the proof of Theorem 3, we add the following assumption where needed.

Assumption 3.3. Let $e \in E$ and $g \in G$ be a position in $\mathcal{G}$. Then there exists an attacker winning strategy for $\mathcal{G}[g, e]$ if and only if there exists an energy-positional attacker winning for $\mathcal{G}[g, e]$.

Using strategies that are not energy-positional we formalise soundness and using energy-positional strategies we show completeness. Then, we can conclude Theorem 3 as `inductive_winning_budget` by Assumption 3.3.

```
lemma winning_budget_implies_ind:
  assumes "winning_budget e g"
  shows "winning_budget_ind e g"

lemma winning_budget_ind_implies_nonpos:
  assumes "winning_budget_ind e g"
  shows "nonpos_winning_budget e g"

lemma inductive_winning_budget:
  assumes "nonpos_winning_budget = winning_budget"
  shows "winning_budget = winning_budget_ind"
```

Note that in these lemmas `g` is automatically assumed to be a position and `e` an energy as a result of typing.

The upward-closure of winning budgets in monotonic energy games stated in Lemma 3.2 will be used heavyly in our proof of decidability. For this reason we formalise this fact. Even though this is not a result of Theorem 3, we show a version of this lemma that uses the inductive characterisation of attacker winning budgets by using `winning_budget_ind`.

```
lemma upward_closure_wb_ind:
  assumes "⋀g g' e e'. weight g g' ≠ None ⟹ apply_w g g' e ≠ None
       ⟹ leq e e' ⟹ apply_w g g' e' ≠ None ∧
       leq (the (apply_w g g' e)) (the (apply_w g g' e'))"
       and "leq e e'" and "winning_budget_ind e g"
  shows "winning_budget_ind e' g"
```

Here `leq` is some predicate for pairs of energies representing a relation. Note that we did not need to assume this relation to be a partial order.

Chapter 4

Bisping's Declining Energy Games

Bisping, Nestmann, and Jansen [10, 9] generalised the bisimulation game to find Hennessy-Milner logic (HML) formulae distinguishing finite-state processes. Those formalae are elements of some HML-sublanguage from van Glabbeek's linear-time-branching-time spectrum[21] and thus their existence is a statement about behavioural equivalences. The HML-sublanguages from the linear-time–branching-time spectrum can be characterised by depth properties, which can be represented by six-dimensional vectors of extended natural numbers. Understanding these vectors as energies Bisping[7] developed a multi-weighted energy game deciding all common notions of (strong) behavioural equivalences at once, the *spectroscopy game*. This game motivated this thesis and illustrates the benefit of decidability.
In his work Bisping [7] introduced a class of energy games, he calls *declining energy games*, which includes the spectroscopy game. We will call this class of energy games *Bisping's declining energy games*. These games over vectors of naturals differ from other energy games by using minima in updates. Thereby, classic algorithms to decide energy games are not applicable. Bisping provides a modified algorithm, which he claims decides this class of energy games. In Chapter 5 we will discuss this algorithm and prove Bisping's claim of decidability for a larger class of energy games.
In this chapter we introduce Bisping's declining energy games in Section 4.1. There we also introduce Bisping's inversion of updates, a weakened form of invertibility. Abstracting this property of updates using Galois connections we define a new class of energy games, namely *Galois energy games*, in Section 4.2. With Section 4.3 we conclude this chapter by discussing our formalisation of Bisping's declining energy games.

4.1 Bisping's Declining Energy Games

Bisping's declining energy games are defined over a fixed set of energies with a set of possibly occurring updates. We start by introducing Bisping's energies, i.e. n-dimensional vectors with entries in the extended natural numbers ordered component-wise. Afterwards, we define Bisping's updates and with those we

define Bisping's declining energy games.

Bisping's Energies

Notation 4.1 (Bisping's energies)**.** Bisping's energies refers to the partially ordered set $(\mathbb{N}_\infty^n, \leqslant)$ of n-dimensional vectors of extended naturals for some fixed dimension $n \in \mathbb{N}$ with the component-wise order, i.e. $(e_0, ..., e_{n-1}) \leqslant (e'_0, ..., e'_{n-1})$ if and only if $\forall i \in \{0, ..., n-1\}$. $e_i \leqslant e'_i$ for all $(e_0, ..., e_{n-1}), (e'_0, ..., e'_{n-1}) \in \mathbb{N}_\infty$.[12]

Note that we do not claim, that Bisping invented n-dimensional vectors of extended naturals. Our naming is merely intended to reflect our reason to include this partially ordered set, namely to define the energy games introduced by Bisping [7].

Lemma 4.1. *Bisping's energies are well-founded for any dimension $n \in \mathbb{N}$.*

Proof sketch. We show (iii) of Lemma 2.7 by induction over the dimension n with a trivial base case. Assume the statement holds for n. For a fixed sequence in $\mathbb{N}_\infty^{n+1}$ there exists a subsequence such that the last entries are (not necessarily strictly) increasing. When considering only the first n entries of every vector in that subsequence, the induction hypothesis can be applied yielding the desired result. □

From $\mathbb{N}_\infty$ being a complete lattice we can directly conclude that Bisping's energies form a complete lattice.

Lemma 4.2. *Bisping's energies with the component-wise infimum and supremum form a complete lattice for any dimension $n \in \mathbb{N}$ and the minimum of $\mathbb{N}_\infty^n$ is the zero-vector.*

Notation 4.2. With $\sup : \mathcal{P}(\mathbb{N}_\infty^n) \to \mathbb{N}_\infty^n$ we refer to the component-wise supremum.

Bisping's Updates

Now we define Bisping's updates. They allow three ways of updating an entry of an energy vector.

- Not changing the entry.
- Subtracting one from the entry.
- Replacing the entry with the minimum of some of the entries, including the replaced entry.

Bisping's updates can be represented as n-dimensional vectors where each component contains the information how to update the corresponding component of energy vectors.

[12] This is a reformulation of Definition 4 as stated by Bisping [7].

Definition 4.1 (Bisping's updates)**.** *The set of Bisping's updates* $\mathcal{U}_n$ *is represented as the subset of* $(\{0,-1\} \uplus \mathcal{P}(\{0,...,n-1\}))^n$ *where* $u_i = D \subseteq \{0,...,n-1\}$ *implies* $i \in D$ *for all* $(u_0,...,u_{n-1}) \in \mathcal{U}_n$*. We then interpret* $u = (u_0,...,u_{n-1}) \in \mathcal{U}_n$ *as a partial function* $u : \mathbb{N}_\infty^n \to \mathbb{N}_\infty^n$ *in the following manner* $u(e_0,...,e_{n-1}) := (e'_0,...,e'_{n-1})$ *with*

$$e'_i := \begin{cases} e_i, & \text{if } u_i = 0 \\ e_i - 1, & \text{if } u_i = -1 \wedge e_i > 0 \\ \min_{\mathbb{N}}\{e_k \mid k \in D\}, & \text{if } u_i = D \subseteq \{0,...,n-1\} \end{cases} \quad \text{for } (e_0,...,e_{n-1}) \in \mathbb{N}_\infty^n.$$ [13]

This definition directly implies that Bisping's updates are declining, justifying the name choice.[14]

Notation 4.3. We call the energy game $\mathcal{G}$ declining, if all updates u in $\mathcal{G}$ are declining, i.e. $\forall e \in \mathrm{dom}(u).\ u(e) \leqslant e$.

Bisping's Declining Energy Games

Now we are able to properly define Bisping's declining energy games.

Definition 4.2 (Bisping's declining energy games)**.** *The class of Bisping's declining (n-dimensional) energy games contains all energy games* $(G_B, G_{Ba}, \rightarrowtail)$ *over* $\mathbb{N}_\infty^n$ *where all updates are elements of* $\mathcal{U}_n$ *for a fixed dimension* $n \in \mathbb{N}$*.*

By Definition 4.1 Bisping's updates are monotonic with an upward-closed domain.[15] This directly implies the following lemma.

Lemma 4.3. *Bisping's declining energy games are monotonic energy games.*

4.1.1 Bisping's Inversion

Bisping's algorithm relies on a weakened form of invertibility of updates. Inverting an update $u \in \{0,-1\}^n$ is trivial, while taking the minimum is not injective and thereby not invertible. Thus, in general Bisping's updates are not invertible. However, Bisping gives a calculation[16] such that

$$\mathrm{inv}_B(u)(e') = \min\{e \mid e' \leqslant u(e)\} \tag{4.1}$$

holds for all $u \in \mathcal{U}_n$ and $e' \in \mathbb{N}_\infty^n$. We later argue how this can be understood as a *weakened form of invertibility*. First, we introduce Bisping's inversion.

Example 4.1. Let $n = 2$ and $u := (-1, \{0,1\}) \in \mathcal{U}_n$. Then $(a,b) \leqslant u(x,y) = (x-1, \min_{\mathbb{N}}\{x,y\})$ implies $a \leqslant x-1$, i.e. $a+1 \leqslant x$, and $b \leqslant \min_{\mathbb{N}}\{x,y\}$, i.e. $b \leqslant x \wedge b \leqslant y$, for all $(a,b),(x,y) \in \mathbb{N}_\infty^2$. Therefore, $(a,b) \leqslant u(x,y)$ implies $\max_{\mathbb{N}}\{a+1,b\} \leqslant x$ and $b \leqslant y$. In particular, we have

$$\begin{aligned} \mathrm{inv}_B(u)(a,b) &\overset{(4.1)}{=} \min\{(x,y) \mid (a,b) \leqslant u(x,y)\} \\ &\overset{\text{Def. } u}{=} \min\{(x,y) \mid (a,b) \leqslant (x-1, \min_{\mathbb{N}}\{x,y\})\} \\ &= (\max_{\mathbb{N}}\{a+1,b\}, b). \end{aligned}$$

[13] This is a reformulation of Definition 6 by Bisping [7].
[14] This statement is formalised as `updates_declining` in Update.thy.
[15] This is formalised as `updates_monotonic` and `upd_domain_upward_closed` in Update.thy.
[16] We refer to Definition 13 by Bisping [7].

Abstracting Example 4.1 we can define Bisping's inversion for $u = (u_0, ..., u_{n-1}) \in \mathcal{U}_n$ and $e = (e_0, ..., e_{n-1}) \in \mathbb{N}_\infty^n$ such that the i-th component of $\mathrm{inv}_B(u)(e)$ is the maximum of e_i (plus one if applicable) and each entry e_j where $i \in u_j \subseteq \{0, ..., n-1\}$.

Definition 4.3 (Bisping's inversion)**.** *Let $n \in \mathbb{N}$. For $u = (u_0, ..., u_{n-1}) \in \mathcal{U}_n$ we define $\mathrm{inv}_B(u) : \mathbb{N}_\infty^n \to \mathbb{N}_\infty^n$ by setting*

$$\mathrm{inv}_B(u)(e)_i := \max_{\mathbb{N}} (\{e_i - u_i \mid u_i \in \{0, -1\}\} \cup \{e_j \mid i \in u_j \subseteq \{0, ..., n-1\}\})$$

for all $e = (e_0, ..., e_{n-1}) \in \mathbb{N}_\infty^n$ and $i \in \{0, ..., n-1\}$.

This definition directly implies the following.

Lemma 4.4. *Let $n \in \mathbb{N}$ and $u \in \mathcal{U}_n$. Then $\mathrm{inv}_B(u)$ is computable.*

Remark 4.1. Bisping describes the calculation in a different manner. He defines

$$\mathsf{upd}^{-1}(u, e) := \sup_{\mathbb{N}} (\{e - one(u)\} \cup \{m(j) \mid u_j \subseteq \mathcal{P}(\{0, ..., n-1\})\})$$

for $n \in \mathbb{N}$, $u = (u_0, ..., u_{n-1}) \in \mathcal{U}_n$ and $e = (e_0, ..., e_{n-1}) \in \mathbb{N}_\infty^n$ with

$$one(u)_i := \begin{cases} -1, & \text{if } u_i = -1 \\ 0, & \text{else} \end{cases} \quad \text{and} \quad m(j)_i := \begin{cases} e_j, & \text{if } i \in u_j \subseteq \{0, ..., n-1\} \\ 0, & \text{else} \end{cases}$$

for all $i \in \{0, ..., n-1\}$.

Both definitions yield the same result.

Proof. Let $e = (e_0, ..., e_{n-1}) \in \mathbb{N}_\infty^n$ and $i \in \{0, ..., n-1\}$. Then we have

$$e_i - one(u)_i = \sup_{\mathbb{N}}(\{e_i\} \cup \{e_i - u_i \mid u_i \in \{0, -1\}\}) \tag{4.2}$$

by definition of *one*. Further, we have

$$\begin{aligned} &\{m(j)_i \mid u_j \subseteq \{0, ..., n-1\}\} \\ &= \{0 \mid i \notin u_j \subseteq \{0, ..., n-1\}\} \cup \{e_j \mid i \in u_j \subseteq \{0, ..., n-1\}\} \end{aligned} \tag{4.3}$$

by definition of m. Since $e_i \in \mathbb{N}_\infty$ we also have

$$0 \leqslant e_i \leqslant \mathrm{inv}_B(u)(e)_i \tag{4.4}$$

by Definition 4.3. Now we can assert the following.

$$\begin{aligned} \mathsf{upd}^{-1}(u, e)_i &= \sup_{\mathbb{N}}(\{e_i - one(u)_i\} \cup \{m(j)_i \mid u_j \subseteq \mathcal{P}(\{0, ..., n-1\})\}) \\ &\overset{(4.2)}{=} \sup_{\mathbb{N}}(\{e_i\} \cup \{e_i - u_i \mid u_i \in \{0, -1\}\} \\ &\qquad \cup \{m(j)_i \mid u_j \subseteq \mathcal{P}(\{0, ..., n-1\})\}) \\ &\overset{(4.3)}{=} \sup_{\mathbb{N}}(\{e_i\} \cup \{e_i - u_i \mid u_i \in \{0, -1\}\} \\ &\qquad \cup \{0 \mid i \notin u_j \subseteq \{0, ..., n-1\}\} \cup \{e_j \mid i \in u_j \subseteq \{0, ..., n-1\}\} \\ &\overset{(4.4)}{=} \sup_{\mathbb{N}}(\{e_i - u_i \mid u_i \in \{0, -1\}\} \cup \{e_j \mid i \in u_j \subseteq \{0, ..., n-1\}\}) \\ &\overset{\text{Def. } 4.3}{=} \mathrm{inv}_B(u)(e)_i. \end{aligned}$$

□

Remark 4.2. When formalising Definition 4.3 we will take the maximum of a potentially larger set, where a zero might have been added, to calculate inv_B. For $n \in \mathbb{N}$, $u = (u_0, ..., u_{n-1}) \in \mathcal{U}_n$, $e = (e_0, ..., e_{n-1}) \in \mathbb{N}_\infty^n$ and $i \in \{0, ..., n-1\}$ we formalise $\text{inv}_B(u)(e)_i$ as

$$\begin{aligned}\max_{\mathbb{N}}(&\{e_i \mid \exists j \neq i.\ u_j \in \{0, -1\}\} \cup \{e_i - u_i \mid u_i \in \{0, -1\}\} \\ &\cup \{0 \mid i \notin u_j \subseteq \{0, ..., n-1\}\} \cup \{e_j \mid i \in u_j \subseteq \{0, ..., n-1\}\}).\end{aligned}$$

Equality follows from simple calculations, mostly given in the preceding proof.

4.2 Galois Energy Games

In the previous section we showed that Bisping's updates are monotonic and $\min\{e \mid e' \leqslant u(e)\}$ exists for all $u \in \mathcal{U}_n$ and $e' \in \mathbb{N}_\infty^n$. These properties define a Galois connection. In this section we introduce Galois connections and use them to define the class of energy games to which Bisping's algorithm can be applied. We name this class *Galois energy games*. Figure 4.1 visualises the class hierarchy, where a subclass is below its superclass.

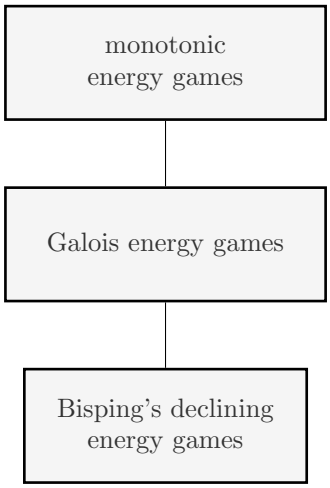

Figure 4.1: Class diagram with Galois energy games

We now introduce Galois connections, after which Galois energy games are named. Galois connections are partial functions between partially ordered sets that can be understood as a weakened form of invertibility.

Definition 4.4 (Galois connection)**.** *Let $\mathcal{P} = (P, \leqslant_P)$ and $\mathcal{Q} = (Q, \leqslant_Q)$ be partially ordered sets and let $f : P \to Q$ and $g : Q \to P$ be functions. Then, $(\mathcal{P}, f, g, \mathcal{Q})$ is a Galois connection if $f(p) \leqslant_Q q \longleftrightarrow p \leqslant_P g(q)$ holds for all $p \in P$ and $q \in Q$. We then say that f and g form a Galois connection between P and Q.*[17]

We now illustrate what we mean by *weakened form of invertibility.*

[17]This definition is sometimes called a monotone Galois connection and thereby differentiated from an antitone Galois connection as noted by Smith [29].

Example 4.2. 1. The embedding of integers into reals and the floor function form a Galois connection between the integers and the real numbers with their usual ordering.

2. For a function between sets A and B the image and reverse image form a Galois connection between the power sets of A and B ordered by inclusion.

Understanding Galois connections as *weakened form of invertibility* is further justified by the following observation.

Remark 4.3. Let $(\mathcal{P}, f, g, \mathcal{Q})$ be a Galois connection. Then, f and g are mutual quasi-inverses, i.e. $f \circ g \circ f = f$ and $g \circ f \circ g = g$.

By restating Proposition 4.1 as stated by Erné et al. [18] we observe some characteristic properties of Galois connections.

Lemma 4.5. *Let $\mathcal{P} = (P, \leqslant_P)$ and $\mathcal{Q} = (Q, \leqslant_Q)$ be partially ordered sets and (f, g) a pair of functions with $f : P \to Q$ and $g : Q \to P$. Then the following are equivalent:*

a) $(\mathcal{P}, f, g, \mathcal{Q})$ is a Galois connection.

b) f and g are monotonic, $g \circ f$ is increasing, and $f \circ g$ is decreasing.

c) f is monotonic and for each $q \in Q$ the largest element p with $f(p) \leqslant_Q q$ is $g(q)$, i.e. $g(q) = \max_P\{p \mid f(p) \leqslant_Q q\}$.

d) g is monotonic and for each $p \in P$ the smallest element q with $p \leqslant_P g(q)$ is $f(p)$, i.e $f(p) = \min_Q\{q \mid p \leqslant_P g(q)\}$.

The following is a direct consequence of Lemma 4.5.

Remark 4.4. Let $(P, \leqslant_P)$ and $(Q, \leqslant_Q)$ be partially ordered sets. Note that a monotone function $f : P \to Q$ uniquely defines a Galois connection between P and Q if it exists. The same holds for a monotonic function $g : Q \to P$.

We now use Galois connections to define a class of energy games whose updates satisfy this weakened form of invertibility.

Definition 4.5 (Galois energy game)**.** *A Galois energy game $\mathcal{G}_\mathcal{G} = (\mathcal{G}, \mathrm{inv})$ consists of an energy game $\mathcal{G}$ over $\mathcal{E}$ and a partial function $\mathrm{inv} : (\mathcal{E} \to \mathcal{E}) \to (\mathcal{E} \to \mathcal{E})$ such that all updates in $\mathcal{G}$ have upward-closed domains and $(\mathcal{E}, \mathrm{inv}(u), u, \mathrm{dom}(u))$ is a Galois connection for all updates u in $\mathcal{G}$. We then call $\mathcal{G}$ a Galois energy game.*

Notation 4.4. Let $\mathcal{G}_\mathcal{G} = (\mathcal{G}, \mathrm{inv})$ be a Galois energy game and u an update in $\mathcal{G}$. We then write u^{-1} instead of $\mathrm{inv}(u)$.[18]

Remark 4.5. Let $\mathcal{G}_\mathcal{G} = (\mathcal{G}, \mathrm{inv})$ with an energy game $\mathcal{G}$ over $\mathcal{E}$ and a partial function $\mathrm{inv} : (\mathcal{E} \to \mathcal{E}) \to (\mathcal{E} \to \mathcal{E})$. Note that by Remark 4.4 an update u in $\mathcal{G}$ uniquely defines a Galois connections between $\mathcal{E}$ and $\mathrm{dom}(u)$, if such a Galois connection exists. Therefore, if $\mathcal{G}_\mathcal{G}$ is a Galois energy game, then $\mathcal{G}$ uniquely defines inv.[19]

[18] As discussed in Example 4.2 and Remark 4.3 this notation should be taken with a grain of salt since we do not actually assume u and u^{-1} to be mutually inverse.

[19] This justifies us calling both $\mathcal{G}_\mathcal{G} = (\mathcal{G}, \mathrm{inv})$ and $\mathcal{G}$ a Galois energy game.

Lemma 4.5 and Definition 4.5 directly imply the following.

Lemma 4.6. *An energy game $\mathcal{G}$ is a Galois energy game if and only if $\mathcal{G}$ is a monotonic energy game and $\min\{e \in \mathbb{N} \mid e' \leqslant u(e)\}$ exists for all updates u in $\mathcal{G}$ and energies $e' \in \mathcal{E}$.*

Example 4.3. We now consider the energy game $\mathcal{G}'$ from Figure 3.1. In Example 3.3 we noted that this is a monotonic energy game. Further, $\min\{e \in \mathbb{N} \mid e' \leqslant u(e)\}$ exists for all $e' \in \mathcal{E}$. Thus by Lemma 4.6 $\mathcal{G}'$ is a Galois energy game. In particular, inv can be defined by $\mathrm{inv}(z)(e) = \begin{cases} e - z, & \text{if } z \leqslant e \\ 0, & \text{else} \end{cases}$ for all $e \in \mathbb{N}$ where again $e \mapsto e + z$ is abbreviated by z for $z \in \mathbb{Z}$.

We now show that Bisping's declining energy games are Galois energy games.

Lemma 4.7. *Bisping's declining energy games are Galois energy games.*

Proof sketch. Let $n \in \mathbb{N}$ and let $\mathcal{G}$ be a Bisping's declining n-dimensional energy game. Then $\mathcal{G}$ is a monotonic energy game by Lemma 4.3. Further, $\mathrm{inv}_B(u)(e') = \min\{e \mid e' \leqslant u(e)\}$ for all $u \in \mathcal{U}_n$ and $e' \in \mathbb{N}_\infty^n$ follows from Definition 4.3 and a case analysis of possible updates. Therefore, $\mathcal{G}$ is a Galois energy games by Lemma 4.6. □

With Example 4.3 we have seen a Galois energy game that is not a Bisping's declining energy game. That example can be interpreted as an energy game over Bisping's energies of dimension one since $N \subseteq N_\infty$ holds and the updates can be extended accordingly. We now show for any dimension that Galois energy games over Bisping's energies form a larger class of energy games. Similarly to Definition 4.1 we define a representation of a set of updates. In particular, this set represents all updates where each entry of an energy vector is updated in one of the following manners:

- An integer is added to the entry.
- The entry is multiplied by a natural.
- The entry is replaced with a minimum of arbitrary entries.

Definition 4.6. *Let $n \in \mathbb{N}$ and $\mathcal{U}'_n := (+\mathbb{Z} \cup \cdot\mathbb{N} \cup \mathcal{P}(\{0, ..., n-1\}))^n$. We then interpret $u = (u_0, ..., u_{n-1}) \in \mathcal{U}'_n$ as a partial function $u : \mathbb{N}_\infty^n \to \mathbb{N}_\infty^n$ by setting*

$$u(e_0, ..., e_{n-1}) := (e'_0, ..., e'_{n-1}) \text{ with } e'_i := \begin{cases} e_i + z, & \text{if } u_i = +z \wedge -z \leqslant e_i \\ e_i \cdot m, & \text{if } u_i = \cdot m \\ \min_{k \in D} e_k, & \text{if } u_i = D \subseteq \{0, ..., n-1\} \end{cases}$$

for all $(e_0, ..., e_{n-1}) \in \mathbb{N}_\infty^n$.

Note that the updates represented by an element of $\mathcal{U}_n$ form a true subset of the updates represented with $\mathcal{U}'_n$.

Lemma 4.8. *Let $\mathcal{G} = (G, G_a, \rightarrowtail)$ be an n-dimensional energy game over Bisping's energies with updates in $\mathcal{U}'_n$. Then $\mathcal{G}$ is a Galois energy game.*

Proof sketch. By Definition 4.6 all updates in $\mathcal{U}'_n$ are monotonic and their domain is upward-closed. Using Lemma 4.6 we only show that

$\text{inv}(u)(e') := \min\{e \mid e' \leqslant u(e)\}$ exists for all $u \in \mathcal{U}'_n$ and $e' \in \mathcal{E}$ by giving a calculation[20] similar to Bisping's inversion.
Let u be an update in $\mathcal{G}$, i.e. $u = (u_0, ..., u_{n-1}) \in \mathcal{U}'_n$. For $e' = (e'_0, ..., e'_{n-1}) \in \mathbb{N}^n_\infty$ we set

$$\begin{aligned}\text{inv}(u)(e')_i := \max(&\{e'_i - z \mid u_i = +z \in +\mathbb{Z} \wedge z \leqslant e'_i\} \\ &\cup \{\lceil \tfrac{e'_i}{m} \rceil \mid u_i = \cdot m \in \cdot\mathbb{N}\} \\ &\cup \{e'_j \mid i \in u_j \subseteq \{0, ..., n-1\}\} \\ &\cup \{0 \mid i \notin u_j \subseteq \{0, ..., n-1\} \vee (u_i = +z \in +\mathbb{Z} \wedge e'_i < z)\})\end{aligned}$$

for $i \in \{0, ..., n-1\}$. In doing so, the i-th component of $\text{inv}_B(u)(e')$ can be understood as the maximum of

- $e'_i - z$, if this is not negative and the update adds z in the i-th component, i.e. if $u_i = +z \in +\mathbb{Z}$ and $z \leqslant e'_i$
- $\lceil \frac{e'_i}{m} \rceil$, if the update multiplies the i-th component with m, i.e. if $u_i = \cdot m \in \cdot\mathbb{N}$
- e'_j, if the i-th component is used when taking the minimum resulting in the j-th component, i.e. if $i \in u_j \subseteq \{0, ..., n-1\}$ for some $j \in \{0, ..., n-1\}$
- 0 else. □

4.3 Formalisation of Bisping's Declining Energy Games

In this section we discuss key elements of our formalisation of this chapters definitions. This section is structured analogously to the files of our Isabelle/HOL theory. The dependencies of those files are visualised by Figure 4.2, where the theories above are imported by the ones below.[21]

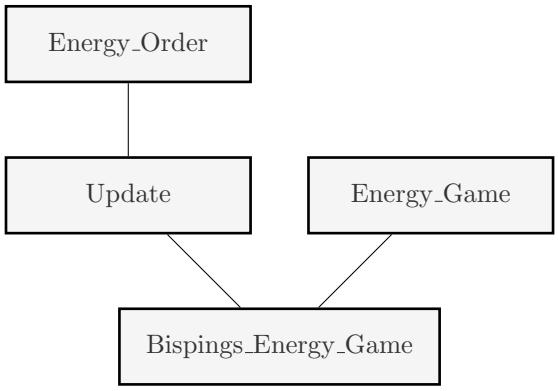

Figure 4.2: Extract from session graph

[20] Note that this calculation is computable.
[21] Note that Figure 4.2 only includes some of our theories and does not visualise other imports.

We already discussed our formalisation of energy games in Section 3.3. In this section we first discuss our formalisation of energies and refer to Energy_Order.thy for the complete formalisation. Afterwards we focus on Bisping's (inverse) updates as formalised in Updates.thy and conclude this section with a discussion of our formalisation of Bisping's declining energy games in Bispings_Energy_Game.thy.

4.3.1 Formalisation of Bisping's Energies

In Notation 4.1 Bisping's energies are introduced as $\mathbb{N}_\infty^n$ for a fixed dimension $n \in \mathbb{N}$ with the component-wise order. We formalise energies as lists and fix a dimension later. For this we introduce the type synonym `energy`.

```
type_synonym energy = "enat list"
```

Note that the set $\mathbb{N}_\infty^n$ then is formalised as the set `{e::energy. length e = n}`, where `n` will later be the fixed dimension. This formalisation choice necessitates length checks. One example of such a check is in our formalisation of the component-wise order, in which we define only vectors of the same length to potentially be comparable.

```
definition energy_leq:: "energy ⇒ energy ⇒ bool" (infix "e≤" 80)
where
  "energy_leq e e' = ((length e = length e') ∧
       (∀i < length e. (e ! i) ≤ (e' ! i)))"
```

The infix declaration allows us to write `e e≤ e'` instead of `energy_leq e e'`. Note that `length` returns the length of a list and `e ! i` returns the `i`-th entry of a list `e`.

After proving that this is a partial order we formalise that Bisping's energies are well-founded as stated in Lemma 4.1 by utilising an existing formalisation of well-quasi-orders.[22] In particular we use `wqo_on` to state that for a fixed dimension n the component-wise order is well-founded on $\mathbb{N}_\infty^n$.

```
lemma energy_leq_wqo:
  shows "wqo_on energy_leq {e::energy. length e = n}"
```

We then define the set of minimal elements w.r.t. the component-wise order. Further, we formalise the component-wise supremum taking the dimension as part of the input to compensate the not yet fixed dimension.

```
definition energy_Min:: "energy set ⇒ energy set"
where
  "energy_Min A = {e∈A . ∀e'∈A. e≠e' ⟶ ¬ (e' e≤ e)}"

definition energy_sup :: "nat ⇒ energy set ⇒ energy" where
"energy_Sup n A = map (λi. Sup {(e!i) | e. e ∈ A}) [0..<n]"
```

[22] Well-quasi-orders are reflexive and transitive binary relations fulfilling (iii) from Lemma 2.7.

We utilise the on extended natural numbers predefined `Sup` to map `i` to the supremum of all `i`-th entries of elements of `A` for each `i`$\in \{0, ..., n-1\}$.

We formalised the most relevant part ofLemma 4.2, i.e. that the component-wise order on $\mathbb{N}_\infty^n$ forms a bounded join-semilattice. Specifically, we formalised that the component-wise supremum is a the smallest upper bound and that the supremum of the empty set is the zero vector.

```
lemma energy_sup_is_sup:
  shows "⋀a. a ∈ A ⟹ length a = n ⟹ a e≤ (energy_sup n A)" and
        "⋀s. (⋀a. a∈ A ⟹a e≤ s) ⟹ length s = n ⟹
        (energy_sup n A) e≤ s"

lemma empty_Sup_is_zero:
  assumes "i < n"
  shows "(energy_sup n {}) ! i = 0"
```

4.3.2 Formalisation of Bisping's Updates

In Definition 4.1 we started by defining the set $\mathcal{U}_n$ to represent Bisping's updates. We formalise the set $\{0, -1\} \uplus \mathcal{P}(\{0, ..., n-1\})$ as a datatype where 0 is represented by `zero`, -1 is represented by `minus_one` and a set $D \in \mathcal{P}(\{0, ..., n-1\})$ is represented by `min_set D`. Then we again use lists to represent vectors and fix a dimension later.

```
datatype update_component = zero | minus_one | min_set "nat set"
type_synonym update = "update_component list"
```

For an update component u_k to be represented as a set D and formalised as `min_set D` the conditions $k \in D$ and $D \subseteq \{0, ..., n-1\}$ are required for a fixed dimension n. Because we do not yet fix a dimension, we require $D \subseteq \{0, ..., (|u|-1)\}$ instead since later the length of updates will coincide with the dimension. We now introduce a predicate to check whether these conditions are fulfilled by all components of an update.

```
abbreviation "valid_update u ≡ (∀i D. u ! i = min_set D ⟶
       i ∈ D ∧ D ⊆ {x. x < length u})"
```

To formalise the remainder of Definition 4.1 we now consider the application of updates. We first introduce `apply_component` where `apply_component i uc v` is the application of an update component `uc` to the `i`-th entry of an energy vector `v`. Again we formalise a partial function by using the datatype `option`. If the application of an update yields `None` in any component, the output of `apply_update` will be `None`, this is formalised using `those`. Note that the predefined function `those` converts terms of type `'a option list` to type `'a list option` such that any `None` entry in a list results in the output being `None`.

```
fun apply_component::"nat ⇒ update_component ⇒ energy ⇒
           enat option"
where
  "apply_component i zero e = Some (e ! i)" |
  "apply_component i minus_one e = (if ((e ! i) > 0)
       then Some ((e ! i) - 1) else None)" |
  "apply_component i (min_set A) e = Some (min_list (nths e A))"

fun apply_update:: "update ⇒ energy ⇒ energy option"
where
  "apply_update u e = (if (length u = length e)
       then (those (map (λi. apply_component i (u ! i) e)
       [0..<length e])) else  None)"
```

Note that `nths e A` returns the list created from the list `e` only containing entries indexed by an element in `A`, while `min_list` returns the minimal element of a non-empty list. From now on we abbreviate `the (apply_update u e)` with `upd u e`.

Now we formalise Bisping's inversion. As stated in Remark 4.2 we slightly deviate from Definition 4.3. Similarly to the application of updates we first introduce `apply_inv_component` where `apply_inv_component i u e` is the `i`-th component of $\mathrm{inv}_B(\mathtt{u})(\mathtt{e})$.

```
fun apply_inv_component::"nat ⇒ update ⇒ energy ⇒ enat"
where
  "apply_inv_component i u e = Max (set (map (λ(j,up).
       (case up of zero ⇒ e ! i |
       minus_one ⇒ (if i=j then (e ! i)+1 else e ! i) |
       min_set A ⇒ (if i∈A then (e ! j) else 0)))
       (List.enumerate 0 u)))"

fun apply_inv_update:: "update ⇒ energy ⇒ energy option"
where
  "apply_inv_update u e = (if (length u = length e)
       then Some (map (λi. apply_inv_component i u e) [0..<length e])
       else None)"
```

Note that `List.enumerate 0` maps a list of type `'a list` to a list of type `(nat × 'a) list` by enumerating the entries starting with `0`. From now on we abbreviate `the (apply_inv_update u e)` by `inv_upd u e`

With the formalisation of Bisping's inversion we are now able to formalise Lemma 4.7. Since we did not formalise the class of Galois energy games, we do this by showing the properties used to define Galois energy games in Definition 4.5.

```
lemma upd_domain_upward_closed
  assumes "apply_update u e ≠ None" and "e e≤ e'"
  shows "apply_update u e' ≠ None"

lemma galois_connection:
  assumes "apply_update u e' ≠ None" and "length e = length e'" and
          "valid_update u"
  shows "(inv_upd u e) e≤ e' = e e≤ (upd u e')"
```

Note that the assumption `apply_update u e ≠ None` implies the length of `u` and `e'` being equal. Further, note that we did not use the preexisting formalisation of Galois connections by Kappelmann [25] or by Alasdair Armstrong and Simon Foster [5]. While compatibility could be achieved, this is not needed for the formalisation of our results. Note that there is no preexisting formalisation of Lemma 4.5. Instead of adding this fact, we formalised the relevant properties of updates and their "inverse" separately.[23]

4.3.3 Formalisation of Bisping's Declining Energy Games

Now we are able to formalise Bisping's declining energy games. Similar to the formalisation of energy games in Section 3.3 we do this as a `locale`. All following definitions and lemmas are then stated within the context `bispings_energy_game`.

```
locale bispings_energy_game =
       energy_game attacker weight apply_update
  for attacker ::  "'position set" and
      weight :: "'position ⇒ 'position ⇒ update option"
+
  fixes dimension :: "nat"
  assumes valid_updates: "∀p. ∀p'. ((weight p p' ≠ None )
       ⟶ ((length (the (weight p p')) = dimension) ∧
       valid_update (the (weight p p'))))"
```

Note that we fixed a dimension and called it `dimension`. In this text we identify the dimension with n. Further, with `valid_updates` we assume all updates to be of length `dimension` and to be a valid update, i.e. to be an element of $\mathcal{U}_n$.

Note that $\mathsf{Win}_a(g_d) = \mathcal{E}$ holds for a defender position $g_d \in G_d$, that is a deadend. Since we did not introduce a type for energies in $\mathbb{N}_\infty^n$, our formalisation of attacker winning budgets implies that all lists of type `enat` regardless of their length are in the previously formalised attacker winning budgets of such positions. This can be understood as us previously formalising $\mathsf{Win}_a(g)$ as $\{e \mid \mathcal{G}[g, e] \text{ is won by the attacker}\}$ instead of $\{e \in \mathbb{N}_\infty^n \mid \mathcal{G}[g, e] \text{ is won by the attacker}\}$. While we assume these sets to be equal, we have to make $e \in \mathbb{N}_\infty^n$ an explicit assumption. We do this with length checks and reformalising attacker winning budgets through their inductive characterisation as `winning_budget_len`.

[23] For an arbitrary dimension $n \in \mathbb{N}$ we showed monotonocity of $\mathrm{inv}_B(u)$ for any update $u \in \mathcal{U}_n$ as `inverse_monotonic` and the equality $\mathrm{inv}_B(u)(e') = \min\{e \mid e' \leqslant u(e)\}$ for all $e' \in \mathbb{N}_\infty^n$ as `apply_inv_is_min`. Further, we showed that $u \circ \mathrm{inv}_B(u)$ is increasing as `leq_up_inv` and that $\mathrm{inv}_B(u) \circ u$ is decreasing as `inv_up_leq` for all $u \in \mathcal{U}_n$.

```
inductive winning_budget_len::"energy ⇒ 'position ⇒ bool"
where
 "winning_budget_len e g" if "length e = dimension ∧
       g ∉ attacker ∧ (∀g'. (weight g g' ≠ None) ⟶
       ((apply_update (the (weight g g')) e)≠ None ∧
       (winning_budget_len (upd (the (weight g g')) e)) g'))" |
 "winning_budget_len e g" if "length e = dimension ∧
       g ∈ attacker ∧ (∃g'. (weight g g' ≠ None) ∧
       (apply_update (the (weight g g')) e)≠ None ∧
       (winning_budget_len (upd (the (weight g g')) e) g'))"
```

To check compatibility with previous definitions we state the following.

```
lemma winning_bugget_len_is_wb:
  assumes "nonpos_winning_budget = winning_budget"
  shows "winning_budget_len e g = (winning_budget e g ∧
       length e = dimension)"
```

Note that we explicitly add Assumption 3.3 allowing us to use both, energy-positional strategies and those that are not, in our proof.

Chapter 5

Decidability

For a fixed position g the *fixed initial credit problem* asks the question whether some energy e is sufficient for the attacker to win the game starting in g, i.e. whether $e \in \mathsf{Win}_a(g)$. The *unknown initial credit problem* corresponds to asking for a fixed initial position g whether there exists an energy sufficient for the attacker to win the game starting in g, i.e. whether $\mathsf{Win}_a(g) \neq \varnothing$. Subsuming both these problem, we ask for the attacker winning budget of each position.

Notation 5.1. We call $\mathcal{G}$ decidable, if there exists an algorithm to compute all attacker winning budgets in a finite amount of time.

In this chapter we will discuss a simplified version of the algorithm provided by Bisping to decide Bisping's declining energy games. The core idea of the algorithm calculating attacker winning budgets is that of a shortest path algorithm – starting with deadends and moving backwards through the graph using the inductive characterisation of attacker winning budgets.
For a play to be won by the attacker, it has to end in a defender position without outgoing edges. At such a position g_d we have $\mathcal{E} = \mathsf{Win}_a(g_d)$, which is easy to calculate. Then, for an attacker position g with $g \rightarrowtail^u g_d$ every energy sufficient to move from g to g_d is in the attacker winning budget of g, i.e. $\mathrm{dom}(u) \subseteq \mathsf{Win}_a(g)$. So far we only described an application of Theorem 3, particularly of the following rule.

$$\frac{g \in G_a \qquad g \rightarrowtail^u g' \qquad u(e) \in \mathsf{Win}_a(g')}{e \in \mathsf{Win}_a(g)}$$

To describe the energies when moving backwards through the graph, invertibility would be useful. Assuming the updates in $\mathcal{G}$ to be invertible Theorem 3 would imply the following rule.

$$\frac{g \in G_a \qquad g \rightarrowtail^u g' \qquad e' \in \mathsf{Win}_a(g')}{u^{-1}(e') \in \mathsf{Win}_a(g)}$$

In this chapter we show that the weakened form of invertibility of updates, which exists in Galois energy games, is sufficient to derive (minimal) attacker winning budgets by inversion. This allows us to apply Bisping's algorithm to a larger class of energy games. To substantiate and generalise Bisping's claim of decidability we prove the algorithm's correctness before arguing for termination.

Thereby we show the main result of this thesis, which can be stated as the following theorem.

Theorem 5 (Decidability of Galois energy games)**.** *Let $\mathcal{G}_{\mathcal{G}} = (\mathcal{G}, \mathrm{inv})$ be a Galois energy game over $\mathcal{E}$ with a finite set of positions G. Further, let* $\mathrm{inv}\, u$ *be computable for all updates u in $\mathcal{G}$ and let $(\mathcal{E}, \leqslant)$ be a well-founded bounded join-semilattice such that the set of minimal energies as well as the suprema of finite sets of energies are computable. Then $\mathcal{G}$ is decidable.*

Since attacker winning budgets of monotonic energy games are upward-closed by Lemma 3.2, when given the set of energies that are minimal in some attacker winning budget, we can infer for all energies whether they are in that winning budget. Consequently, it is sufficient to provide an algorithm calculating minimal attacker winning budgets, which can be understood as a Pareto front. This idea is discussed in more detail in Section 5.1, where we introduce an order such that Kleene's fixed point theorem is applicable. In Section 5.2 we then present and discuss Algorithm 1 before proving Theorem 5. Note that we do not formalise Theorem 5 in its general form in Isabelle/HOL. Instead we only formalise the result for Bisping's energy games and discuss our formalisation in Section 5.3. For this reason we give more extensive proofs compared to previous chapters. In Section 5.3 we also briefly discuss Bisping's original algorithm and our simplification. We end this chapter with a complexity analysis in 5.4.

5.1 Kleene's Fixed Point Theorem

The algorithm introduced in Section 5.2 is a fixed point algorithm. To prove its correctness we will use Kleene's fixed point theorem. In this section we prepare the proof by restating that theorem. First, we define a set of candidates for minimal attacker winning budgets on which the algorithm given later operates and an order on this set such that Kleene's fixed point theorem applies.

5.1.1 Minimal Attacker Winning Budgets as Pareto Fronts

When given the set of energies that are minimal in some attacker winning budget, we can infer for all energies whether they are in that winning budget by Lemma 3.2. In other words, minimal attacker winning budgets form a Pareto front. In this section minimal winning budgets and possible Pareto fronts are discussed. First, we introduce minimal attacker winning winning budgets.

Definition 5.1 (Minimal attacker winning budget)**.** *For $g \in G$ we call $\mathsf{Win}_a^{\min}(g) := \operatorname{Min} \mathsf{Win}_a(g)$ the minimal attacker winning budget of g.*

Example 5.1. The minimal attacker winning budgets of $\mathcal{G}'$ from Figure 3.1 are

$$\mathsf{Win}_a^{\min}(g_0) = \{0\}, \quad \mathsf{Win}_a^{\min}(g_1) = \{1\} \quad \text{and} \quad \mathsf{Win}_a^{\min}(g_2) = \{0\}.$$

To ensure that such minimal attacker winning budgets always exist, we assume the energies to be well-founded.

Assumption 5.1. Let $(\mathcal{E}, \leqslant)$ be well-founded.

Remark 5.1. By Lemma 3.2 the upward-closure of a minimal attacker winning budget in a monotonic energy game is the attacker winning budget, i.e. $\mathsf{Win}_a(g) = \uparrow \mathsf{Win}_a^{\min}(g)$ for all positions $g \in G$.

In multi-objective optimisation multiple objectives need to be satisfied simultaneously, and trade-offs between them are often necessary. A solution to such an optimisation problem, where no objective can be improved without worsening at least one other objective, is called Pareto optimal. The set of all Pareto optimal solutions is the Pareto front. For a fixed $g \in G$ any $e \in \mathsf{Win}_a^{\min}(g)$ is a Pareto solution of the energy game $\mathcal{G}$ with initial position g, i.e. $\mathsf{Win}_a^{\min}(g)$ is the Pareto front of the energy game $\mathcal{G}$ with initial position g. Further, $\mathsf{Win}_a^{\min}$ can be understood as a mapping $\mathsf{Win}_a^{\min} : G \to \mathcal{P}(\mathcal{E})$ from positions to the respective Pareto fronts. In later proofs we will consider such mappings and compare them to $\mathsf{Win}_a^{\min}$. For this reason we define the set of possible Pareto fronts.[24]

Definition 5.2 (Possible Pareto front)**.** *For a set of energies $E \subseteq \mathcal{E}$ we denote the sets of incomparable elements in E with $\mathcal{P}_{\mathsf{pareto}}(E)$, i.e. $\mathcal{P}_{\mathsf{pareto}}(E) = \{A \in \mathcal{P}(E) \mid A \text{ is an antichain}\}$. A function $F : G \to \mathcal{P}(\mathcal{E})$ is a possible Pareto front of the energy game $\mathcal{G}$ if all positions are mapped to antichains, i.e. $\forall g \in G.\ F(g) \in \mathcal{P}_{\mathsf{pareto}}(\mathcal{E})$. We denote the set of possible Pareto fronts of $\mathcal{G}$ as $\mathsf{PARETO}_{\mathcal{G}}$.*

Remark 5.2. By definition mapping positions to minimal attacker winning budgets is a possible Pareto front, i.e. $\mathsf{Win}_a^{\min} \in \mathsf{PARETO}_{\mathcal{G}}$.

We now introduce and discuss a partial order on possible Pareto fronts. Note that by Definition 5.2 for a possible Pareto front F we do not enforce $F(g)$ to be a set of attacker winning budgets for any position g. Despite this we give an intuition for the order on $\mathsf{PARETO}_{\mathcal{G}}$ defined in Definition 5.3 by considering the image $F(g)$ as a possible solution space of $\mathcal{G}$ with initial position g: We want $F \prec F'$ to hold, if for all positions F covers a smaller solution space than F'.

Definition 5.3 (Order on possible Pareto fronts)**.** *The order on possible Pareto fronts is defined for all $F, F' \in \mathsf{PARETO}_{\mathcal{G}}$ by $F \preceq F'$ if $F(g) \subseteq \uparrow F'(g)$ for all $g \in G$.*

We now justify previously calling this order a *partial order*.

Lemma 5.1. *The order on possible Pareto fronts is a partial order on $\mathsf{PARETO}_{\mathcal{G}}$.*

Proof. We show reflexivity, transitivity and antisymmetry in that order. For this let $F, F', F'' \in \mathsf{PARETO}_{\mathcal{G}}$.

1. For all $g \in G$ we have $F(g) \subseteq \uparrow F(g)$, hence $F \preceq F$. This implies that the order on possible Pareto fronts is reflexive.
2. If $F \preceq F'$ and $F' \preceq F''$, then $F(g) \subseteq \uparrow F'(g) \subseteq \uparrow (\uparrow F''(g)) = \uparrow F''(g)$ for all $g \in G$ and thereby $F \preceq F''$. This implies that the order on possible Pareto fronts is transitive.

[24] Note that we actually define the set of mappings from positions to possible Pareto fronts. For the sake of readability we allow this slight inconsistency in our naming.

3. We now show that the the order on possible Pareto fronts is antisymmetric. For this we assume $F \preceq F'$ and $F' \preceq F$. We show $F = F'$ by showing $F(g) = F'(g)$ for all $g \in G$. Let $g \in G$ and $e \in F(g)$. Since $F \preceq F'$ we have $F(g) \subseteq \uparrow F'(g)$ and there exists $e' \in F'(g)$ with $e' \leqslant e$. Further, there exists $e'' \in F(g)$ with $e'' \leqslant e'$ since $F' \preceq F$ implies $F'(g) \subseteq \uparrow F(g)$. Since all distinct elements of $F(g)$ are incomparable, from $e'' \leqslant e' \leqslant e$ follows $e'' = e$ and thereby $e = e' \in F'(g)$. Since $e \in F(g)$ was arbitrary, we have $F(g) \subseteq F'(g)$. Reversing the roles of F and F' we can conclude $F'(g) \subseteq F(g)$ and thus $F(g) = F'(g)$. □

With this order we can also define an infimum and a supremum.

Definition 5.4 (infimum and supremum on possible Pareto fronts)**.** *We define* $\inf_\preceq : \mathcal{P}(\mathsf{PARETO}_\mathcal{G}) \to \mathsf{PARETO}_\mathcal{G}$ *and* $\sup_\preceq : \mathcal{P}(\mathsf{PARETO}_\mathcal{G}) \to \mathsf{PARETO}_\mathcal{G}$ *by setting*

$$(\inf_\preceq P)(g) := \operatorname{Min} \bigcap_{F \in P} \uparrow F(g)$$
$$(\sup_\preceq P)(g) := \operatorname{Min} \bigcup_{F \in P} \uparrow F(g)$$

for all $g \in G$ *and* $P \subseteq \mathsf{PARETO}_\mathcal{G}$.[25]

The next lemma justifies our name choice.

Lemma 5.2. *The partially ordered set* $(\mathsf{PARETO}_\mathcal{G}, \preceq)$ *is a complete lattice with infimum* $\inf_\preceq$ *and supremum* $\sup_\preceq$.

Proof. Let $P \subseteq \mathsf{PARETO}_\mathcal{G}$. We start by showing $\inf_\preceq P \preceq F' \preceq \sup_\preceq P$ for all $F' \in P$. For all $F' \in P$ and $g \in G$ we have

$$(\inf_\preceq P)(g) \subseteq \uparrow (\inf_\preceq P)(g) = \uparrow \bigcap_{F \in P} \uparrow F(g) \subseteq \uparrow\uparrow F'(g) = \uparrow F'(g) \quad \text{and}$$
$$F'(g) \subseteq \uparrow F'(g) \subseteq \bigcup_{F \in P} \uparrow F(g) = \uparrow \sup_\preceq P.$$

We now show that $\inf_\preceq P$ and $\sup_\preceq P$ are maximal and minimal respectively with this property. Assume $F_i \preceq F$ for all $F \in P$. For all $g \in G$ we then have $F_i(g) \subseteq \uparrow F(g)$ for all $F \in P$ and thereby $F_i(g) \subseteq \bigcap_{F \in P} \uparrow F(g) \subseteq (\inf_\preceq P)(g)$. Thus $F_i \preceq \inf_\preceq P$. Now assume $F \preceq F_s$ for all $F \in P$. For all $g \in G$ we then have $F(g) \subseteq \uparrow F_s(g)$ and thereby $\uparrow F(g) \subseteq \uparrow F_s(g)$ for all $F \in P$. This implies $(\sup_\preceq P)(g) \subseteq \bigcup_{F \in P} \uparrow F(g) \subseteq \uparrow F_s(g)$ for all $g \in G$ and therefore $\sup_\preceq P \preceq F_s$. □

Lemma 5.2 and Definition 2.4 directly imply the following.

Corollary 5.1. *The order on possible Pareto fronts is a directed-complete partial order with minimal element* $\mathbf{0} : G \to \mathcal{P}(\mathcal{E}),\ g \mapsto \varnothing$.

[25] The upward-closure in the definition of the supremum could be omitted without changing the definition. We choose to write $\uparrow$ to illustrate symmetry. Further, note that by the use of Min in Definition 5.4, $\sup_\preceq P,\ \inf_\preceq P \in \mathsf{PARETO}_\mathcal{G}$ holds for all $P \subseteq \mathsf{PARETO}_\mathcal{G}$.

5.1.2 Kleene's Fixed Point Theorem

Before we can restate Kleene's fixed point theorem we need the following definition.

Definition 5.5 (Scott-continuous)**.** *Let $f : P \to Q$ be a function between partially ordered sets $(P, \leqslant_P)$ and $(Q, \leqslant_Q)$. Then f is Scott-continuous, if it preserves directed suprema, i.e. for all directed sets $P' \subseteq P$ with a supremum the set $\{f(p) \mid p \in P'\}$ has a supremum such that $f(\sup_P P') = \sup_Q \{f(p) \mid p \in P'\}$ holds.*

Since the supremum of a set of two comparable elements is its maximum, Definition 5.5 directly implies the following.

Lemma 5.3. *A Scott-continuous function is monotonic.*

Notation 5.2. Let $f : P \to P$ be a total function and $i \in \mathbb{N}$. Then, f^i is inductively defined by $f^0 := id_P$ being the identity on P and $f^{j+1} := f \circ f^j$ for all $j \in \mathbb{N}$.

Now we restate Kleene's fixed point theorem [32].

Theorem 4 (Kleene's fixed point theorem)**.** *Let $(P, \leqslant_P)$ be a directed-complete partial order with least element $p \in P$ and a Scott-continuous function $f : P \to P$. Then the set of fixed points of f in P forms a complete lattice and $\sup_P \{f^i(p) \mid i \in \mathbb{N}\}$ is the least fixed point of f in P.*[26]

Kleene's fixed point theorem combined with Corollary 5.1 directly imply that Kleene's fixed point theorem applies when considering functions operating on $\mathsf{PARETO}_{\mathcal{G}}$.

Corollary 5.2. *The least fixed point of a Scott-continuous function $f : \mathsf{PARETO}_{\mathcal{G}} \to \mathsf{PARETO}_{\mathcal{G}}$ is $\sup_{\preceq} \{f^i(\mathbf{0}) \mid i \in \mathbb{N}\}$.*

5.2 The Algorithm

In this section we discuss the algorithm, which we use to prove Theorem 5. Algorithm 1 is a least fixed point algorithm calculating minimal attacker winning budgets. We prove this claim in Section 5.2.1 using Kleene's fixed point theorem. In Section 5.2.2 we then argue that the algorithm terminates and thereby prove decidability.
The core idea of the algorithm is to move backwards through the graph using the structure of (minimal) attacker winning budgets in Galois energy games. We now state Algorithm 1, the simplified version of Bisping's algorithm.

[26] Note that this is not the most general version of Kleene's fixed point theorem as there exist multiple generalisations, in particular this theorem also holds for ω-continuous functions on ω-complete partially ordered sets as shown by Mashburn [27] and formalised by Yamada and Dubut [32].

```
1  def compute_winning_budgets(𝒢 = (G, G_a, ↦)):
2      new_a_win := [g ↦ {} | g ∈ G]
3      do
4          a_win := new_a_win
5          for g ∈ G :
6              if g ∈ G_a :
7                  new_a_win[g] := Min{u⁻¹(e') | g ↣ᵘ g' ∧ e' ∈ a_win[g']}
8              else:
9                  new_a_win[g] := Min{sup{u⁻¹(e_g') | g ↣ᵘ g'} |
                     ∀g'. g ↣ᵘ g' ⟶ e_g' ∈ a_win[g']}
10     while new_a_win ≠ a_win
11     return a_win
```

Algorithm 1: Algorithm for computing minimal attacker winning budgets of Galois energy game $\mathcal{G}$.

We first discuss the ideas behind Lines 5 to 9. To calculate the attacker winning budgets of attacker positions we use the derivation rule mentioned in this chapter's introduction.

$$\frac{g \in G_a \qquad g \rightarrowtail^{u} g' \qquad e' \in \mathsf{Win}_a(g')}{u^{-1}(e') \in \mathsf{Win}_a(g)}$$

In Algorithm 1 this corresponds to Line 7. Similarly, a derivation rule for defender positions is utilised in Line 9. For an energy to be in the attacker winning budget of a defender position, the energy has to be sufficient to win when moving to any successor. For upward-closed attacker winning budgets that is $e \in \mathsf{Win}_a(g_d)$ for $g_d \in G_d$ if $u^{-1}(e_{g'}) \leqslant e$ for some $e_{g'} \in \mathsf{Win}_a(g')$ for each g' with $g_d \rightarrowtail^{u} g'$. If the supremum exists, this yields the following.

$$\frac{g \in G_d \qquad \forall g'.\ g \rightarrowtail^{u} g' \longrightarrow e_{g'} \in \mathsf{Win}_a(g')}{\left(\sup\{u^{-1}(e_{g'}) \mid g \rightarrowtail^{u} g'\}\right) \in \mathsf{Win}_a(g)}$$

By Line 2 we begin with $\mathbf{0}$ and then apply Lines 3 to 10 until we reach a fixed point, which is returned.
Note that Theorem 5 only applies to Galois energy games. Therefore, we apply Algorithm 1 to Galois energy games such that u^{-1} exists for all updates u in $\mathcal{G}$. Further, we make the following assumption.

Assumption 5.2. Let the set of positions G be finite and let $(\mathcal{E}, \leqslant)$ be a bounded join-semilattice.

If the set of positions if finite, we can argue that the sets of energies considered in Line 9 are finite. Then, there always exists a supremum in $\mathcal{E}$ by Assumption 5.2.

Example 5.2. We consider $\mathcal{G}'$ from Figure 3.1, which is a Galois energy game as discussed in Example 4.3. Note that $\mathbb{N}$ with the usual order is a well-founded bounded join-semilattice. In Table 5.1 we give new_a_win as it is updated when applying the algorithm to $\mathcal{G}'$. In this example a fixed point is reached after four iterations of the while loop and the algorithm terminates after five iterations.

Iteration count	$\mathsf{new\_a\_win}[g_0]$	$\mathsf{new\_a\_win}[g_1]$	$\mathsf{new\_a\_win}[g_2]$
0	$\varnothing$	$\varnothing$	$\varnothing$
1	$\varnothing$	$\varnothing$	$\{0\}$
2	$\varnothing$	$\{2\}$	$\{0\}$
3	$\{0\}$	$\{2\}$	$\{0\}$
4	$\{0\}$	$\{1\}$	$\{0\}$
5	$\{0\}$	$\{1\}$	$\{0\}$

Table 5.1: Application of Algorithm 1 to $\mathcal{G}'$

5.2.1 Proof of Correctness

Algorithm 1 is a least fixed point algorithm. We prove correctness by defining a Scott-continuous function $\mathsf{Iteration} : \mathsf{PARETO}_{\mathcal{G}} \to \mathsf{PARETO}_{\mathcal{G}}$ such that its application corresponds to one iteration of the while-loop in Algorithm 1. By Corollary 5.2 the algorithm then calculates the least fixed point of this function. We then show that the minimal attacker winning budgets are the least fixed point.

Definition 5.6 (Iteration)**.** *Define* $\mathsf{Iteration} : \mathsf{PARETO}_{\mathcal{G}} \to \mathsf{PARETO}_{\mathcal{G}}$ *by setting*

$$\mathsf{Iteration}(F)(g_a) := \mathrm{Min}\{u^{-1}(e') \mid g_a \rightarrowtail^{u} g' \wedge e' \in F(g')\} \quad \textit{and}$$
$$\mathsf{Iteration}(F)(g_d) := \mathrm{Min}\{\sup\{u^{-1}(e_{g'}) \mid g_d \rightarrowtail^{u} g'\} \mid \forall g'.\ g_d \rightarrowtail^{u} g' \longrightarrow e_{g'} \in F(g')\}$$

for all attacker positions $g_a \in G_a$*, defender positions* $g_d \in G_d$ *and possible Pareto fronts* $F \in \mathsf{PARETO}_{\mathcal{G}}$*.*

Note that by the use of Min in the definition of $\mathsf{Iteration}$ this actually is a functor of possible Pareto fronts. The attacker case corresponds to Line 7 in Algorithm 1 while the defender case corresponds to Line 9. We now prepare the proof of Scott-continuity of $\mathsf{Iteration}$ by showing that it is monotonic.[27]

Lemma 5.4. *Let* $\mathcal{G}$ *be a Galois energy game. Then* $\mathsf{Iteration}$ *is monotonic.*

Proof. Let $F, F' \in \mathsf{PARETO}_{\mathcal{G}}$ with $F \preceq F'$. We now show $\mathsf{Iteration}(F) \preceq \mathsf{Iteration}(F')$, i.e. $\mathsf{Iteration}(F)(g) \subseteq \uparrow \mathsf{Iteration}(F')(g)$ for all $g \in G$, by case distinction.

Case "$g_a \in G_a$": Let $g_a \in G_a$ and $e \in \mathsf{Iteration}(F)(g_a)$. By Definition 5.6 there exists $g' \in G$ and $e' \in F(g')$ such that $g_a \rightarrowtail^{u} g'$ and $e = u^{-1}(e')$. Since $F \preceq F'$, there exists $e'' \in F'(g')$ with $e'' \leqslant e'$. Hence $u^{-1}(e'') \in \uparrow \mathsf{Iteration}(F')(g_a)$ by Definition 5.6. Since $\mathcal{G}_{\mathcal{G}}$ is a Galois energy game, u^{-1} is monotonic and therefore $u^{-1}(e'') \leqslant u^{-1}(e') = e$. Thus $e \in \uparrow \mathsf{Iteration}(F')(g_a)$. Since e was arbitrary, this implies $\mathsf{Iteration}(F)(g_a) \subseteq \uparrow \mathsf{Iteration}(F')(g_a)$.

Case "$g_d \in G_d$": Let $g_d \in G_d$ and $e \in \mathsf{Iteration}(F)(g_d)$. By Definition 5.6 there exists $e_{g'} \in F(g')$ for each $g' \in G$ with $g \rightarrowtail^{u} g'$ such that $e = \sup\{u^{-1}(e_{g'}) \mid g \rightarrowtail^{u} g'\}$. Since $F \preceq F'$, there exists $e'_{g'} \in F'(g')$ with $e'_{g'} \leqslant e_{g'}$ for each $g' \in G$ with $g \rightarrowtail^{u} g'$. Hence

[27] While Scott-continuity implies monotonicity by Lemma 5.3, we choose to show both properties separately for more manageable proofs.

$\sup\{u^{-1}(e'_{g'}) \mid g \rightarrowtail^{u} g'\} \in \uparrow \mathsf{Iteration}(F')(g_a)$ by Definition 5.6. Since u^{-1} is monotonic, we have $u^{-1}(e'_{g'}) \leqslant u^{-1}(e_{g'})$ for all $g' \in G$ with $g \rightarrowtail^{u} g'$ and thereby $\sup\{u^{-1}(e'_{g'}) \mid g \rightarrowtail^{u} g'\} \leqslant \sup\{u^{-1}(e_{g'}) \mid g \rightarrowtail^{u} g'\} = e$. Thus $e \in \uparrow \mathsf{Iteration}(F')(g_d)$. This implies $\mathsf{Iteration}(F)(g_d) \subseteq \uparrow \mathsf{Iteration}(F')(g_d)$ since e was arbitrary. □

Lemma 5.4 and Corollary 5.1 directly imply the following.

Corollary 5.3. *Let $\mathcal{G}$ be a Galois energy game. Then* $\mathsf{Iteration}^i(\mathbf{0}) \preceq \mathsf{Iteration}^j(\mathbf{0})$ *for all $i, j \in N$ with $i \leqslant j$.*

Using Lemma 5.4 we now show, that Iteration is Scott-continuous.

Lemma 5.5. *Let $\mathcal{G}$ be a Galois energy game. Then* Iteration *is Scott-continuous.*

Proof. Let $P \subseteq \mathsf{PARETO}_{\mathcal{G}}$ be a directed set. We now show $\mathsf{Iteration}(\sup_{\preceq} P) = \sup_{\preceq}\{\mathsf{Iteration}(F) \mid F \in P\}$. For all $F \in P$ we have $\mathsf{Iteration}(F) \preceq \mathsf{Iteration}(\sup_{\preceq} P)$ since Iteration is monotonic and $F \preceq \sup_{\preceq} P$. This implies $\sup_{\preceq}\{\mathsf{Iteration}(F) \mid F \in P\} \preceq \mathsf{Iteration}(\sup_{\preceq} P)$.
We now show $\mathsf{Iteration}(\sup_{\preceq} P) \preceq \sup_{\preceq}\{\mathsf{Iteration}(F) \mid F \in P\}$. For this we show $\mathsf{Iteration}(\sup_{\preceq} P)(g) \subseteq \uparrow (\sup_{\preceq}\{\mathsf{Iteration}(F) \mid F \in P\})(g)$ for all $g \in G$ by case distinction. Note that we use the property of Galois energy games that u^{-1} is monotonic for all updates u in $\mathcal{G}$.

Case "$g_a \in G_a$": Let $g_a \in G_a$ and $e \in \mathsf{Iteration}(\sup_{\preceq} P)(g_a)$. Then there exists g' such that $g_a \rightarrowtail^{u} g'$ and $e' \in (\sup_{\preceq} P)(g')$ such that $e = u^{-1}(e')$ by Definition 5.6. From $e' \in (\sup_{\preceq} P)(g') = \mathrm{Min} \bigcup_{F \in P} \uparrow F(g')$ follows that there exists $F \in P$ such that $e' \in \uparrow F(g')$ and thereby $e'' \in F(g')$ with $e'' \leqslant e'$ exists. Since u^{-1} is monotonic, we have $u^{-1}(e'') \leqslant u^{-1}(e') = e$. By Definition 5.6 we have $u^{-1}(e'') \in \uparrow \mathsf{Iteration}(F)(g_a)$ and therefore $e \in \uparrow \mathsf{Iteration}(F)(g_a) \subseteq \uparrow (\sup_{\preceq}\{\mathsf{Iteration}(F) \mid F \in P\})(g_a)$. This implies $\mathsf{Iteration}(\sup_{\preceq} P)(g_a) \subseteq \uparrow (\sup_{\preceq}\{\mathsf{Iteration}(F) \mid F \in P\})(g_a)$ since e was arbitrary.

Case "$g_d \in G_d$": Let $g_d \in G_d$ and $e \in \mathsf{Iteration}(\sup P)(g_d)$. Then for each g' with $g_d \rightarrowtail^{u} g'$ there exists $e_{g'} \in (\sup_{\preceq} P)(g')$ such that $e = \sup\{u^{-1}(e_{g'}) \mid g_d \rightarrowtail^{u} g'\}$ by Definition 5.6. Since $e_{g'} \in (\sup_{\preceq} P)(g') = \mathrm{Min} \bigcup_{F \in P} \uparrow F(g')$, there exists $F_{g'} \in P$ with $e_{g'} \in \uparrow F_{g'}(g')$ for each g' with $g_d \rightarrowtail^{u} g'$. Since G is finite, so is the set of successors of g_d and likewise $\{F_{g'} \mid g_d \rightarrowtail^{u} g'\}$. By Lemma 2.6 there exists $F' \in P$ such that $F_{g'} \preceq F'$ for all g' with $g_d \rightarrowtail^{u} g'$ since P is directed. Then there exists $e'_{g'} \in F'(g')$ with $e'_{g'} \leqslant e_{g'}$ for each g' with $g_d \rightarrowtail^{u} g'$. For all g' with $g_d \rightarrowtail^{u} g'$ we then have $u^{-1}(e'_{g'}) \leqslant u^{-1}(e_{g'})$ since u^{-1} is monotonic. Hence $\sup\{u^{-1}(e'_{g'}) \mid g_d \rightarrowtail^{u} g'\} \leqslant \sup\{u^{-1}(e_{g'}) \mid g_d \rightarrowtail^{u} g'\} = e$ and $\sup\{u^{-1}(e'_{g'}) \mid g_d \rightarrowtail^{u} g'\} \in \uparrow \mathsf{Iteration}(F')(g_d)$ by Definition 5.6. Thus $e \in \uparrow \mathsf{Iteration}(F')(g_d) \subseteq \uparrow (\sup_{\preceq}\{\mathsf{Iteration}(F) \mid F \in P\})(g_d)$ and we conclude $\mathsf{Iteration}(\sup_{\preceq} P)(g_d) \subseteq \uparrow (\sup_{\preceq}\{\mathsf{Iteration}(F) \mid F \in P\})(g_d)$. □

Lemma 5.5 and Corollary 5.2 directly imply the following.

Corollary 5.4. *Let $\mathcal{G}$ be a Galois energy game. Then* $\sup_{\preceq}\{\mathsf{Iteration}^i(\mathbf{0}) \mid i \in \mathbb{N}\}$ *is the least fixed point of* Iteration *in* $\mathsf{PARETO}_{\mathcal{G}}$.

Now we show that $\mathsf{Win}_a^{\min}$ is the least fixed point of $\mathsf{Iteration}$ and thereby prove the correctness of Algorithm 1 as the following theorem.

Lemma 5.6 (Correctness)**.** *Let $\mathcal{G}$ be a Galois energy game. Then $\mathsf{Win}_a^{\min}$ is a the least fixed point of $\mathsf{Iteration}$ in $\mathsf{PARTEO}_{\mathcal{G}}$.*

Proof. First note that by Lemma 4.5 u and u^{-1} are monotonic, $u \circ u^{-1}$ is increasing and $u^{-1} \circ u$ is decreasing for all updates u in $\mathcal{G}$ since $\mathcal{G}$ is a Galois energy games.
We start by showing that $\mathsf{Win}_a^{\min}$ is a fixed point of $\mathsf{Iteration}$. We do this by defining $S : G \to \mathcal{P}(\mathcal{E})$ such that the following two claims hold.

$$\mathsf{Win}_a^{\min} = \operatorname{Min} \circ S \tag{1}$$

$$\mathsf{Iteration}(\operatorname{Min} \circ S) = \operatorname{Min} \circ S \tag{2}$$

We define S inductively through the derivation rules mentioned in this sections introduction, i.e.

$$\frac{g \in G_a \qquad g \rightarrowtail^{u} g' \qquad e' \in S(g')}{u^{-1}(e') \in S(g)} \qquad \frac{g \in G_d \qquad \forall g'.\ g \rightarrowtail^{u} g' \longrightarrow e_{g'} \in S(g')}{\big(\sup\{u^{-1}(e_{g'}) \mid g \rightarrowtail^{u} g'\}\big) \in S(g)}$$

Then $\operatorname{Min} \circ S \in \mathsf{PARETO}_{\mathcal{G}}$ by Definition 5.2.

(1) Note that $\mathsf{Win}_a(g) = \uparrow S(g)$ directly implies $\mathsf{Win}_a^{\min}(g) = \operatorname{Min} S(g)$ for all $g \in G$. We therefore show $\mathsf{Win}_a(g) = \uparrow S(g)$ for all $g \in G$ by mutual inclusion.

"$\subseteq$" Let $g \in G$ and $e \in \mathsf{Win}_a(g)$ with energy-positional attacker winning strategy s for $\mathcal{G}[g, e]$. Then the order induced by s is well-founded on $R(s, g, e)$ by Lemma 3.5. We utilise well-founded induction to show that $e' \in \uparrow S(g')$ holds for all $(g', e') \in R(s, g, e)$. Let $(g', e') \in R(s, g, e)$. Assume $e'' \in \uparrow S(g'')$ holds for all $(g'', e'') \in R(s, g, e)$ with $(g'', e'') <_s (g', e')$. We now show $e' \in \uparrow S(g')$ by case distinction.

Case "$g' \in G_a$": Assume $g' \in G_a$. Since s is an attacker winning strategy, we then have $u(e') \in \mathsf{Win}_a(s(g', e'))$ with $g' \rightarrowtail^{u} s(g', e')$. Hence $(s(g', e'), u(e')) \in R(s, g, e)$ and $(s(g', e'), u(e')) \leqslant_s (g', e')$ by Definition 3.12 and Definition 3.11. If $(s(g', e'), u(e')) = (g', e')$ was true, then a play ending in a cycle would be consistent with s. Since infinite plays are won by the defender and s is an attacker winning strategy, we have $(s(g', e'), u(e')) <_s (g', e')$. By induction hypothesis $u(e') \in \uparrow S(s(g', e'))$ holds and there exists an e'' such that $e'' \leqslant u(e')$ and $e'' \in S(s(g', e'))$. By definition of S the latter implies $u^{-1}(e'') \in S(g')$ since g' is an attacker position. Further, $e'' \leqslant u(e')$ implies $u^{-1}(e'') \leqslant u^{-1}(u(e'))$ since u^{-1} is monotonic. Because $u^{-1} \circ u$ is decreasing, we have $u^{-1}(e'') \leqslant u^{-1}(u(e')) \leqslant e'$ and thus $e' \in \uparrow S(g')$.

Case "$g' \in G_d$": Assume $g' \in G_d$. By Definition 3.12 and Definition 3.11 we then have $(g'', u(e')) \in R(s, g, e)$ and $(g'', u(e')) \leqslant_s (g', e')$ for all g'' with $g' \rightarrowtail^{u} g''$. If

$(g'', u(e')) = (g', e')$ was true for any g'' with $g' \rightarrowtail^u g''$, then s would be consistent with a play ending an a cycle. Since infinite plays are won by the defender and s is an attacker winning strategy, we have $(g'', u(e')) <_s (g', e')$ for all g'' with $g' \rightarrowtail^u g''$. By induction hypothesis for each g'' with $g' \rightarrowtail^u g''$ there exists an energy $e'_{g''}$ such that $e'_{g''} \leqslant u(e')$ and $u^{-1}(e'_{g''}) \in S(g'')$. Since u^{-1} is monotonic and $u^{-1} \circ u$ is decreasing, we have $u^{-1}(e'_{g''}) \leqslant u^{-1}(u(e')) \leqslant e'$ for all g'' with $g' \rightarrowtail^u g''$. This implies $\left(\sup\{u^{-1}(e'_{g''}) \mid g' \rightarrowtail^u g''\}\right) \leqslant e'$. Further, we have $\left(\sup\{u^{-1}(e'_{g''}) \mid g' \rightarrowtail^u g''\}\right) \in S(g')$ by definition of S since g' is a defender position. Thus, we have $e' \in \uparrow S(g')$.

Since $(g, e) \in R(s, g, e)$ by Definition 3.12, we conclude $e \in \uparrow S(g)$.

"$\supseteq$" Let $g \in G$. We now show $\mathsf{Win}_a(g) \supseteq \uparrow S(g)$. Since $\mathsf{Win}_a(g)$ is upward-closed by Lemma 3.2 it suffices to show $S(g) \subseteq \mathsf{Win}_a(g)$. We do this by induction over the structure of S and show two claims. First we show that for all attacker positions g_a we have

$$\{u^{-1}(e') \mid g_a \rightarrowtail^u g' \wedge e' \in \mathsf{Win}_a(g')\} \subseteq \mathsf{Win}_a(g_a). \tag{a}$$

Then we show for all defender positions g_d that

$$S'(g_d) \subseteq \mathsf{Win}_a(g_d) \quad \text{where} \tag{d}$$
$$S'(g_d) := \{\sup\{u^{-1}(e_{g'}) \mid g_d \rightarrowtail^u g'\} \mid \forall g'.\ g_d \rightarrowtail^u g' \longrightarrow e_{g'} \in \mathsf{Win}_a(g')\}.$$

(a) Let $g_a \in G_a$ and $g' \in G$ with $g_a \rightarrowtail^u g'$ and $e' \in \mathsf{Win}_a(g')$. Since $u \circ u^{-1}$ is increasing, this implies $u(u^{-1}(e')) \in \mathsf{Win}_a(g')$ by Lemma 3.2. Then, $u^{-1}(e') \in \mathsf{Win}_a(g_a)$ follows from Theorem 3 and g_a being an attacker position.

(d) Let $g_d \in G_d$ and $e_{g'} \in \mathsf{Win}_a(g')$ for all g' with $g \rightarrowtail^u g'$. We then have $u(u^{-1}(e_{g'}) \in \mathsf{Win}_a(g')$ for all g' with $g \rightarrowtail^u g'$ by Lemma 3.2 since $u \circ u^{-1}$ is increasing. Since u is monotonic this implies $u\left(\sup\{u^{-1}(e_{g'}) \mid g \rightarrowtail^u g'\}\right) \in \mathsf{Win}_a(g')$ for all g' with $g \rightarrowtail^u g'$ again by Lemma 3.2. By Theorem 3 we thereby have $\left(\sup\{u^{-1}(e_{g'}) \mid g \rightarrowtail^u g'\}\right) \in \mathsf{Win}_a(g_d)$ since g_d is a defender position.

(2) We now show that $\mathrm{Min} \circ S$ is a fixed point of $\mathsf{Iteration}$. For the sake of readability we define $S_{\min}$ such that $\mathsf{Iteration}(\mathrm{Min} \circ S) = \mathrm{Min} \circ S_{\min}$ by setting

$$S_{\min}(g_a) := \{u^{-1}(e') \mid g_a \rightarrowtail^u g' \wedge e' \in \mathrm{Min}\, S(g')\} \quad \text{and}$$
$$S_{\min}(g_d) := \{\sup\{u^{-1}(e_{g'}) \mid g_d \rightarrowtail^u g'\} \mid \forall g'.g_d \rightarrowtail^u g' \longrightarrow e_{g'} \in \mathrm{Min}\, S(g')\}$$

for attacker positions $g_a \in G_a$ and defender positions $g_d \in G_d$. We now show $\mathrm{Min} \circ S_{\min} = \mathrm{Min} \circ S$. Note that $S_{\min}(g) \subseteq S(g)$ by definition for all $g \in G$. By Lemma 2.2 it suffices to show $\mathrm{Min}\, S(g) \subseteq S_{\min}(g)$ for all $g \in G$. We show this by case distinction.

Case "$g_a \in G_a$": Let $g_a \in G_a$. Assume $e \in \mathrm{Min}\, S(g_a)$. By definition of S there exist g' and e' such that $g_a \rightarrowtail^u g'$, $e' \in S(g')$ and $e = u^{-1}(e')$.

Let $e^m \in \operatorname{Min} S(g')$ such that $e^m \leqslant e'$. Since u^{-1} is monotonic, we have $u^{-1}(e^m) \leqslant u^{-1}(e') = e$. Further, $u^{-1}(e^m) \in S_{\min}(g_a) \subseteq S(g_a)$ holds by definition of $S_{\min}$. Since e is minimal in $S(g_a)$, this implies $u^{-1}(e^m) = e$ and thereby $e \in S_{\min}(g_a)$.

Case "$g_d \in G_d$": Let $g_d \in G_d$. Assume $e \in \operatorname{Min} S(g_d)$. By definition of S there exists $e_{g'} \in S(g')$ for each $g_d \rightarrowtail^u g'$ such that $e = \sup\{u^{-1}(e_{g'}) \mid g_d \rightarrowtail^u g'\}$. Let $e^m_{g'} \in \operatorname{Min} S(g')$ with $e^m_{g'} \leqslant e_{g'}$ for each g' with $g_d \rightarrowtail^u g'$. By monotonicity of u^{-1} we then have $u^{-1}(e^m_{g'}) \leqslant u^{-1}(e_{g'})$ for all g' with $g_d \rightarrowtail^u g'$, which implies $\sup\{u^{-1}(e^m_{g'}) \mid g_d \rightarrowtail^u g'\} \leqslant \sup\{u^{-1}(e_{g'}) \mid g_d \rightarrowtail^u g'\} = e$. By definition $\sup\{u^{-1}(e^m_{g'}) \mid g_d \rightarrowtail^u g'\} \in S_{\min}(g_d) \subseteq S(g_d)$ since g_d is a defender position. Since e is minimal in $S(g_d)$, this implies $\sup\{u^{-1}(e^m_{g'}) \mid g_d \rightarrowtail^u g'\} = e$ and thus $e \in S_{\min}(g_d)$.

What remains to show, is that $\mathsf{Win}_a^{\min}$ is the *least* fixed point of $\mathsf{Iteration}$ in $\mathsf{PARTEO}_{\mathcal{G}}$. Let $F \in \mathsf{PARTEO}_{\mathcal{G}}$ be the least fixed point of $\mathsf{Iteration}$ in $\mathsf{PARTEO}_{\mathcal{G}}$, which exists by Corollary 5.4. We now show $F = \mathsf{Win}_a^{\min}$ using antisymmetry of the order on possible Pareto fronts. Since $\mathsf{Win}_a^{\min}$ is a fixed point we have $F \preceq \mathsf{Win}_a^{\min}$ by Kleene's fixed point theorem. We now show $\mathsf{Win}_a^{\min} \preceq F$, i.e. $\mathsf{Win}_a^{\min}(g) \subseteq \uparrow F(g)$ for all $g \in G$. Note that for all $g \in G$ we have $\mathsf{Win}_a^{\min}(g) = \operatorname{Min} S(g) \subseteq S(g)$ by (1). It therefore suffices to show $S(g) \subseteq \uparrow F(g)$ for all $g \in G$. We show this by induction over the structure of S in two steps. First we show

$$e' \in S(g') \subseteq \uparrow F(g') \longrightarrow u^{-1}(e') \in \uparrow F(g_a) \tag{a}$$

for attacker positions $g_a \in G_a$ with $g_a \rightarrowtail^u g'$. Then we show

$$\begin{aligned} &\big(\forall g'.\ g_d \rightarrowtail^u g' \longrightarrow e_{g'} \in S(g') \subseteq \uparrow F(g')\big) \\ &\quad\longrightarrow \big(\sup\{u^{-1}(e_{g'}) | g_d \rightarrowtail^u g'\}\big) \in \uparrow F(g_d) \end{aligned} \tag{d}$$

for defender positions $g_d \in G_d$.

(a) Assume $g_a \in G_a$ and $g' \in G$ with $g_a \rightarrowtail^u g'$ and $e' \in S(g') \subseteq \uparrow F(g')$. Then, there exists $e'' \in F(g')$ such that $e'' \leqslant e'$ and therefore $u^{-1}(e'') \leqslant u^{-1}(e')$ by monotonicity of u^{-1}. By Definition 5.6 we then have $u^{-1}(e'') \in \uparrow \mathsf{Iteration}(F)(g_a)$. This implies $u^{-1}(e'') \in \uparrow F(g_a)$ since F is a fixed point of $\mathsf{Iteration}$. Thus, we have $u^{-1}(e') \in \uparrow F(g_a)$.

(d) Assume $g_d \in G_d$ with $e_{g'} \in S(g') \subseteq \uparrow F(g')$ for each g' with $g_d \rightarrowtail^u g'$. Hence, for each g' with $g_d \rightarrowtail^u g'$ there exists $e'_{g'} \in F(g_d)$ such that $e'_{g'} \leqslant e_{g'}$ and therefore $u^{-1}(e'_{g'}) \leqslant u^{-1}(e_{g'})$ by monotonicity of u^{-1}. Then, we have $\sup\{u^{-1}(e'_{g'}) \mid g_d \rightarrowtail^u g\} \leqslant \sup\{u^{-1}(e_{g'}) \mid g_d \rightarrowtail^u g\}$. Further, we have $\sup\{u^{-1}(e'_{g'}) \mid g_d \rightarrowtail^u g\} \in \uparrow \mathsf{Iteration}(F)(g_d)$ by Definition 5.6. This implies $\sup\{u^{-1}(e'_{g'}) \mid g_d \rightarrowtail^u g\} \in \uparrow F(g_d)$ since F is a fixed point of $\mathsf{Iteration}$ and thereby $\sup\{u^{-1}(e'_g) \mid g_d \rightarrowtail^u g\} \in \uparrow F(g_d)$. □

5.2.2 Proof of Termination

To prove Theorem 5 we now argue that Algorithm 1 terminates. For this we need to make some reasonable computability assumptions.

Assumption 5.3. For finite sets $A \subseteq \mathcal{E}$ let $\sup A$ and $\operatorname{Min} A$ be computable.

With this assumption we observe that each iteration of the while-loop in Algorithm 1 is computable.

Lemma 5.7. *Let $\mathcal{G}$ be a Galois energy game and let u^{-1} be computable for all updates u in $\mathcal{G}$. Then* $\mathsf{Iteration}^i(\mathbf{0})$ *is computable for all $i \in \mathbb{N}$.*

Proof. We show that $\mathsf{Iteration}^i(\mathbf{0})$ can be calculated with a finite amount of simple operations by induction over $i \in \mathbb{N}$. Note that $\mathsf{Iteration}^0(\mathbf{0}) = (\mathbf{0})$ is computable. We now show that $\mathsf{Iteration}^i(\mathbf{0})$ being computable implies $\mathsf{Iteration}^{i+1}(\mathbf{0})$ being computable for all $i \in \mathbb{N}$. Assume $\mathsf{Iteration}^i(\mathbf{0})$ to be computable for all positions for a fixed $i \in \mathbb{N}$. We now consider $\mathsf{Iteration}^{i+1}(\mathbf{0})(g)$ for all $g \in G$ by case distinction.

Case "$g_a \in G_a$": Assume $g_a \in G_a$. Then, we have

$$\mathsf{Iteration}\left(\mathsf{Iteration}^i(\mathbf{0})\right)(g_a) = \operatorname{Min}\{u^{-1}(e') \mid g_a \rightarrowtail^u g' \wedge e' \in \mathsf{Iteration}^i(\mathbf{0})(g')\}$$

by Definition 5.6. Since the set of positions is finite by Assumption 5.2, so is the set of successors of g_a. Further, $\mathsf{Iteration}^i(\mathbf{0})(g')$ is finite for all $g' \in G$ by Assumption 5.1 and Lemma 2.8. Thereby, calculating $\mathsf{Iteration}\left(\mathsf{Iteration}^i(\mathbf{0})\right)(g_a)$ entails calculating $u^{-1}(e')$ for finitely many e' and subsequently applying the minimum. This results in a computable set since u^{-1} is computable for all updates u in $\mathcal{G}$ and by Assumption 5.3.

Case "$g_d \in G_d$": Assume $g_d \in G_d$. Then, we have

$$\begin{aligned}&\mathsf{Iteration}\left(\mathsf{Iteration}^i(\mathbf{0})\right)(g_d)\\ &= \operatorname{Min}\{\sup\{u^{-1}(e_{g'}) \mid g_d \rightarrowtail^u g'\} \mid \forall g'.g_d \rightarrowtail^u g' \longrightarrow e_{g'} \in \mathsf{Iteration}^i(\mathbf{0})(g')\}\end{aligned}$$

by Definition 5.6. Since $\mathsf{Iteration}^i(\mathbf{0})(g')$ is finite for all $g' \in G$ by Assumption 5.1 and Lemma 2.8 and the set of positions is finite by Assumption 5.2, there are only finitely many functions $f : G \to \mathcal{E}$ with $f(g') \in \mathsf{Iteration}^i(\mathbf{0})(g')$ for all $g' \in G$ with $g_d \rightarrowtail^u g'$ that need to be considered. In particular we consider no more than $(\max_{\mathbb{N}}\{|\mathsf{Iteration}^i(\mathbf{0})(g')| \mid g_d \rightarrowtail^u g'\})^{|G|}$ different mappings from successors g' of g_d to elements of $\mathsf{Iteration}^i(\mathbf{0})(g')$. Therefore, calculating $\mathsf{Iteration}\left(\mathsf{Iteration}^i(\mathbf{0})\right)(g_a)$ entails calculating $\sup\{u^{-1}(f(g')) \mid g_d \rightarrowtail^u g'\}$ for finitely many $f : G \to \mathcal{E}$ and subsequently applying the minimum. Since the set of positions is finite and u^{-1} is computable for all updates u in $\mathcal{G}$, such suprema can be calculated in a finite amount of steps by Assumption 5.3. Thereby, $\mathsf{Iteration}^{i+1}(\mathbf{0})(g_d)$ is computable. □

We now show that only finitely many iterations are necessary with the following lemma.

Lemma 5.8. *Let $\mathcal{G}$ be a Galois energy game. Then, there exists $i \in \mathbb{N}$ such that* $\mathsf{Win}_a^{\min} = \mathsf{Iteration}^i(\mathbf{0})$.

Proof. We show this using antisymmetry of the order on possible Pareto fronts. First, note that $\mathsf{Win}_a^{\min} = \sup_{\preceq}\{\mathsf{Iteration}^i(\mathbf{0}) \mid i \in \mathbb{N}\}$ by Lemma 5.6 and Corollary 5.4, which implies $\mathsf{Iteration}^i(\mathbf{0}) \preceq \sup_{\preceq}\{\mathsf{Iteration}^i(\mathbf{0}) \mid i \in \mathbb{N}\} = \mathsf{Win}_a^{\min}$ for

all $i \in \mathbb{N}$ by Lemma 5.2. Thereby we only need to show that there exists $i \in \mathbb{N}$ such that $\mathsf{Win}_a^{\min} \preceq \mathsf{Iteration}^i(\mathbf{0})$. We start by showing the following claims for all positions $g \in G$ and $j, j' \in \mathbb{N}$ with $j \leqslant j'$.

$$\forall e \in \mathsf{Win}_a^{\min}(g).\ \exists i_g^e \in \mathbb{N}.\ e \in \mathsf{Iteration}^{i_g^e}(\mathbf{0})(g) \tag{5.1}$$

$$\forall e \in \mathsf{Win}_a^{\min}(g).\ e \in \mathsf{Iteration}^{j}(\mathbf{0})(g) \longrightarrow e \in \mathsf{Iteration}^{j'}(\mathbf{0})(g) \tag{5.2}$$

$$\mathsf{Win}_a^{\min}(g) = \mathsf{Iteration}^{j}(\mathbf{0})(g) \longrightarrow \mathsf{Win}_a^{\min}(g) = \mathsf{Iteration}^{j'}(\mathbf{0})(g) \tag{5.3}$$

(5.1) Let $g \in G$ and $e \in \mathsf{Win}_a^{\min}(g)$. By Definition 5.4 $e \in \mathsf{Win}_a^{\min}(g) = \left(\sup_{\preceq}\{\mathsf{Iteration}^i(\mathbf{0}) \mid i \in \mathbb{N}\}\right)(g)$ implies that there exists $i_g^e \in \mathbb{N}$ such that $e \in \uparrow \mathsf{Iteration}^{i_g^e}(\mathbf{0})(g)$. Hence there exists $e' \in \mathsf{Iteration}^{i_g^e}(\mathbf{0})(g)$ with $e' \leqslant e$. Since $\mathsf{Iteration}^{i_g^e}(\mathbf{0}) \preceq \mathsf{Win}_a^{\min}$, there exists $e'' \in \mathsf{Win}_a^{\min}(g)$ with $e'' \leqslant e'$. Then, $e'' \leqslant e' \leqslant e$ implies $e'' = e$ by minimality of e in $\mathsf{Win}_a(g)$ and thus $e = e' \in \mathsf{Iteration}^{i_g^e}(\mathbf{0})(g)$.

(5.2) Let $g \in G$ and $j, j' \in \mathbb{N}$ with $j \leqslant j'$. By Corollary 5.3 we have $\mathsf{Iteration}^j(\mathbf{0}) \preceq \mathsf{Iteration}^{j'}(\mathbf{0}) \preceq \mathsf{Win}_a^{\min}$. Then for $e \in \mathsf{Win}_a^{\min}(g) \cap \mathsf{Iteration}^j(\mathbf{0})(g)$ there exists $e' \in \mathsf{Iteration}^{j'}(\mathbf{0})(g)$ with $e' \leqslant e$ and $e'' \in \mathsf{Win}_a^{\min}(g)$ with $e'' \leqslant e'$. Then $e'' \leqslant e' \leqslant e$ implies $e'' = e$ by minimality of e in $\mathsf{Win}_a(g)$ and thus $e = e' \in \mathsf{Iteration}^{j'}(\mathbf{0})(g)$.

(5.3) Let $g \in G$ and $j, j' \in \mathbb{N}$ with $j \leqslant j'$ and assume $\mathsf{Win}_a^{\min}(g) = \mathsf{Iteration}^j(\mathbf{0})(g)$. By Corollary 5.3 we have $\mathsf{Iteration}^j(\mathbf{0}) \preceq \mathsf{Iteration}^{j'}(\mathbf{0}) \preceq \mathsf{Win}_a^{\min}$. Then $\mathsf{Win}_a^{\min}(g) = \mathsf{Iteration}^j(\mathbf{0})(g)$ implies $\uparrow \mathsf{Win}_a^{\min}(g) \subseteq \uparrow \mathsf{Iteration}^{j'}(\mathbf{0})(g) \subseteq \uparrow \mathsf{Win}_a^{\min}(g)$ and $\mathsf{Iteration}^{j'}(\mathbf{0})(g) = \mathsf{Win}_a^{\min}(g)$ follows.

For each position $g \in G$ and each energy $e\mathsf{Win}_a^{\min}(g)$ there exists $i_g^e \in \mathbb{N}$ such that $e \in \mathsf{Iteration}^{i_g^e}(\mathbf{0})(g)$ by (5.1). Since $\mathsf{Win}_a^{\min}(g)$ is finite for all $g \in G$ Lemma 2.8, we can define $i_g \in \mathbb{N}$ with $i_g := \max_{\mathbb{N}}\{i_g^e \mid e \in \mathsf{Win}_a^{\min}(g)\}$ for each position $g \in G$. Then, (5.2) implies $\mathsf{Win}_a^{\min}(g) \subseteq \mathsf{Iteration}^{i_g}(\mathbf{0})(g)$ and thereby $\mathsf{Win}_a^{\min}(g) = \mathsf{Iteration}^{i_g}(\mathbf{0})(g)$ since $\mathsf{Iteration}^{i_g}(\mathbf{0}) \preceq \mathsf{Win}_a^{\min}$. Since G is finite, we can define $i \in \mathbb{N}$ by $i := \max_{\mathbb{N}}\{i_g \mid g \in G\}$ and have $\mathsf{Win}_a^{\min}(g) = \mathsf{Iteration}^i(g)$ for all $g \in G$ by (5.3), i.e. $\mathsf{Win}_a^{\min} = \mathsf{Iteration}^i$. □

Now we can piece together the proof of Theorem 5. Let us first restate the theorem using Assumption 5.1 , Assumption 5.2 and Assumption 5.3.

Theorem 5 (Decidability of Galois energy games)**.** *Let $\mathcal{G}$ be a Galois energy game and let u^{-1} be computable for all updates u in $\mathcal{G}$. Then, $\mathcal{G}$ is decidable.*

Proof. By Lemma 5.7 each iteration of the while-loop in Algorithm 1 can be calculated in finitely many steps. By Lemma 5.8 only finitely many iterations are necessary to compute the least fixed point of $\mathsf{Iteration}$. Thus Algorithm 1 terminates and returns the minimal attacker winning budgets of $\mathcal{G}$ by Lemma 5.6. □

In summary we need the set of energies to be well-founded for minimal elements to exist. The energy game needs to be a Galois energy game with a finite set of positions and the set of energies to form a bounded join-semilattice to even

define Algorithm 1. The set of positions being finite is also used to prove Scott-continuity of Iteration. Thereby, we are able to apply Kleene's fixed point theorem and give a proof of correctness. Further, the set of positions being finite is necessary for the algorithm to terminate. Then, the order on energies being well-founded and computable in the sense of Assumption 5.3 and u^{-1} being computable for all updates u in $\mathcal{G}$ ensure termination and thus decidability.

5.3 Decidability of Bisping's Declining Energy Games

We now investigate the implications of Theorem 5 for Bisping's declining energy games. By Lemma 4.1 and Lemma 4.2 Bisping's energies form a well-founded complete lattice. Thereby, Theorem 5 directly implies the following.

Corollary 5.5. *Let* $(\mathcal{G}, \mathrm{inv})$ *be a Galois energy game over Bisping's energies with a finite set of positions. If* $\mathrm{inv}(u)$ *is computable for all updates* u *in* $\mathcal{G}$*, then* $\mathcal{G}$ *is decidable.*

With Lemma 4.7 we conclude the following theorem.

Theorem 6 (Decidability of Bisping's declining energy games)**.** *Bisping's declining energy games are decidable if their set of positions is finite.*

This result is formalised in Isabelle/HOL. The formalisation is discussed in Section 5.3.1. Afterwards, we state Bisping's algorithm [7] in our notation and compare it to our simplified version in Section 5.3.2.

5.3.1 Formalisation of Decidability of Bisping's Declining Energy Games

We now discuss key elements of our formalisation of Theorem 6. The complete formalisation is in Decidability.thy. While we do not formalise the more general version of Theorem 6, i.e. Theorem 5, we do follow the proof of Theorem 5 as outlined in the previous sections. For this we utilise a preexisting formalisation of Kleene's fixed point theorem by Yamada and Dubut [32].[28]
In Section 4.3 we discussed our formalisation of Bisping's declining energy games in Isabelle/HOL. In particular, we formalised that Bisping's declining energy games are Galois energy games and that updates fulfil the properties stated in b) and d) of Lemma 4.5. By importing the theory Bispings_Energy_Game.thy we build upon this. Again, we state our lemmas within a context specified by a `locale`. In particular, we assume the energy game to be a Bisping's declining energy game, Assumption 3.3 to hold and the set of positions to be finite. The latter combined with our formalisation of the supremum on Bisping's energies allows us to use Assumption 5.2 as well as Assumption 5.1 .

[28] Note that we developed a more self-contained theory before striving towards compatibility with the formalisation by Yamada and Dubut [32]. For this reasons there sometimes exist two formalised versions of a statement. With one exception we only discuss our later formalisation.

```
locale bispings_energy_game_assms =
       bispings_energy_game attacker weight dimension
  for attacker ::  "'position set" and
      weight :: "'position ⇒ 'position ⇒ update option" and
      dimension :: "nat"
+
assumes nonpos_eq_pos: "nonpos_winning_budget = winning_budget" and
        finite_positions: "finite positions"
```

We now discuss our formalisation corresponding to Section 5.1 for the case of Bisping's declining energy games. First, we introduce the following abbreviations to switch between the definition of winning budgets as sets as seen in Definition 3.7 and a definition as predicates as seen in our formalisation with `winning_budget` and `winning_budget_len`.

```
abbreviation "a_win g ≡ {e. winning_budget_len e g}"
abbreviation "a_win_min g ≡ energy_Min (a_win g)"
```

Now we formalise the set $\mathsf{PARETO}_{\mathcal{G}}$ from Definition 5.2 for Bisping's declining energy games.

```
definition possible_pareto:: "('position ⇒ energy set) set"
where
  "possible_pareto ≡ {F. ∀g. F(g) ⊆ {e. length e = dimension} ∧
       (∀e e'. (e ∈ F(g) ∧ e' ∈ F(g) ∧ e ≠ e') ⟶
       (¬ e e≤ e' ∧ ¬ e' e≤ e))}"
```

Note that we explicitly state that all energies a possible Pareto front maps to are of fixed dimension.

As seen in Definition 5.3 and Definition 5.4 we then formalise the order on possible Pareto fronts as well as the supremum. Note that we do not need to formalise the infimum for our proof.

```
definition pareto_order:: "('position ⇒ energy set) ⇒
           ('position ⇒ energy set) ⇒ bool"  (infix "≤" 80)
where
  "pareto_order F F' ≡ (∀g e. e ∈ F(g) ⟶
       (∃e'. e' ∈ F'(g) ∧ e' e≤ e))"

definition pareto_sup:: "('position ⇒ energy set) set
           ⇒ ('position ⇒ energy set)"
where
  "pareto_sup P g = energy_Min {e. ∃F. F ∈ P ∧ e ∈ F g}"
```

We then formalise Corollary 5.1 in three steps. First, we formalise Lemma 5.1, i.e. that $(\mathsf{PARETO}_{\mathcal{G}}, \leq)$ is a partial order. Secondly, we formalise $(\mathsf{PARETO}_{\mathcal{G}}, \leq)$ being directed-complete.[29] For this we utilise the preexisting formalisation of directed-completeness by Yamada and Dubut [32]. Lastly, we formalise that $\mathbf{0}$ indeed is the minimal element of $\mathsf{PARETO}_{\mathcal{G}}$.

[29] Note that `pareto_directed_complete` is only a statement about the existence of least upper bounds in directed sets. We formalise that $\sup_{\leq} : \mathcal{P}(\mathsf{PARETO}_{\mathcal{G}}) \to \mathsf{PARETO}_{\mathcal{G}}$ from Definition 5.4 maps to a least upper bound of any set as `pareto_sup_is_sup`. Thereby, we implicitly show that the order on possible Pareto fronts forms a (complete) join-semilattice. We use this for most of our proofs, in particular to prove `pareto_directed_complete`.

```
lemma pareto_partial_order:
  shows "reflp_on possible_pareto (≤)" and
        "transp_on possible_pareto (≤)" and
        "antisymp_on possible_pareto (≤)"

lemma pareto_directed_complete:
  shows "directed_complete possible_pareto (≤)"

lemma pareto_minimal_element:
  shows "(λg. {}) ≤ F"
```

Note that `reflp_on` is a formalisation of reflexivity, i.e. `reflp_on P (p≤)` is true if the relation `p≤` is reflexive on the set `P`. Similarly, `transp_on` and `antisymp_on` are formalisations of transitivity and antisymmetry respectively.

We now start formalising Iteration as seen in Definition 5.6 and thereby begin formalising Section 5.2.1 for the case of Bisping's declining energy games.

```
definition iteration:: "('position ⇒ energy set) ⇒
           ('position ⇒ energy set)"
where
  "iteration F g ≡ (if g ∈ attacker then
       energy_Min {inv_upd (the (weight g g')) e' | e' g'.
       length e' = dimension ∧ weight g g' ≠ None ∧ e' ∈ F g'}
       else energy_Min {energy_Sup dimension
       {inv_upd (the (weight g g')) (e_index g') | g'.
       weight g g' ≠ None} | e_index. ∀g'. weight g g' ≠ None ⟶
       length (e_index g')= dimension ∧ e_index g' ∈ F g'})"
```

With this formalisation we have defined a mapping of total functions from positions to sets of vectors of extended natural numbers, not just of possible Pareto fronts. By adding length checks we ensure that we at least map to functions from positions to sets of energies. Further, note that in the defender case we switched from implicitly using a function $G \to \mathcal{E}$ by indexing the energies to explicitly using `e_index`.

We then formalise Lemma 5.5, i.e. that Iteration is Scott-continuous. To prove this we first prove monotonicity but omit a discussion of that lemma because of Lemma 5.3.

```
lemma iteration_scott_continuous:
  assumes "finite positions"
  shows "scott_continuous possible_pareto (≤) possible_pareto (≤)
        iteration"
```

Note that `scott_continuous P (p≤) Q (q≤) f` is true, if the function `f` mapping element of `P` to `Q` is Scott-continuous w.r.t. the relations `p≤` on `P` and `q≤` on `Q`.

Finally we formalise Lemma 5.6, i.e. that the algorihm indeed calculates minimal attacker winning budgets for Bisping's declining energy games. We do this by formalising the equality $\mathsf{Win}_a^{\min} = \sup_{\preceq}\{\mathsf{Iteration}^i(\mathbf{0}) \mid i \in \mathbb{N}\}$ as `winamin_is_lfp_sup`. By Corollary 5.4 this yields the intended result, that

$\mathsf{Win}_a^{\min}$ is the least fixed point of $\mathsf{Iteration}$ in $\mathsf{PARETO}_{\mathcal{G}}$. The latter is formalised as `winamin_is_lfp`.

```
lemma a_win_min_is_lfp_sup:
  assumes "finite positions"
  shows "pareto_sup {(iteration ^^ i) (λg. {}) |. i} = a_win_min"

lemma a_win_min_is_lfp:
  assumes "finite positions"
  shows "extreme {s ∈ possible_pareto. (iteration s) = s} (≥)
       a_win_min"
```

Note that `iteration ^^ i` is a formalisation of $\mathsf{Iteration}^i$ for $i \in \mathbb{N}$ as defined in Notation 5.2. Further, note that `extreme P (p≤) p` is true, if `p` $\in$ `P` is an upper bound of `P` w.r.t. a relation `p`$\preceq$ on `P`. With $\succeq$ we abbreviate the dual of the order on possible Pareto fronts $\preceq$.

We do not formalise every argument for termination we gave in Section 5.2.2. Instead we only formalise Lemma 5.8, i.e. that a fixed point is reached after finitely many iterations.

```
lemma finite_iterations:
  assumes "finite positions"
  shows "∃i. a_win_min = (iteration ^^ i) (λg. {})"
```

It is simple to see that $\mathrm{inv}_B(u)$ is computable for all $u \in \mathcal{U}_n$ as a direct consequence of Definition 4.3. Similarly, we do not see a need to formalise Lemma 5.7 for Bisping's declining energy games, i.e. that $\mathsf{Iteration}^i(\mathbf{0})$ is computable for any finite $i \in \mathbb{N}$. We formalised the key arguments used in the proof of Theorem 5 for the case of Bisping's declining energy games, specifically Lemma 5.8 and Lemma 5.6. Thereby, we provided a formal proof of Theorem 6 using Isabelle/HOL.

5.3.2 Bisping's Algorithm

In this section we first give Bisping's algorithm[30] in our notation and afterwards compare it to Algorithm 1.

[30] We refer to Algorithm 1 in [7].

```
1  def compute_winning_budgets(𝒢 = (G, G_a, ↦)):
2      a_win := [g ↦ {} | g ∈ G]
3      todo := {g ∈ G_d | g is a deadend}
4      while todo ≠ ∅ :
5          g := some todo
6          todo := todo\{g}
7          if g ∈ G_a :
8              new_a_win := Min(a_win[g] ∪ {u^{-1}(e') | g ↣^u g' ∧ e' ∈
                a_win[g']})
9          else:
10             defender_post := {g' | g ↣^u g'}
11             options := {(g', u^1(e')) | g ↣^u g' ∧ e' ∈ a_win[g']}
12             if defender_post ⊆ dom(options) :
13                 new_a_win := Min{sup{u^{-1}(e_{g'}) | g' ∈ defender_post} |
                    ∀g' ∈ defender_post. (g', e_{g'}) ∈ dom(options)}
14             else:
15                 new_a_win := ∅
16         if new_a_win ≠ a_win[g] :
17             a_win[g] := new_a_win
18             todo := todo ∪ {g_p | g_p ↣^u g}
19     Win_a^min := a_win
20     return Win_a^min
```

Algorithm 2: Bisping's algorithm [7] with adapted notation.

We now discuss the alterations performed to yield Algorithm 1 and show that both algorithms yield the same output.

1. One noticeable change is our use of a do-while-loop where Bisping used a different while-loop. This change shortened the algorithm by not introducing todo and omitting the conditional in Line 16 of Algorithm 2. Thereby, we calculate the new possible Pareto fronts of all positions simultaneously, while Algorithm 2 does this one position at a time. Note that the latter only calculates $\mathsf{Iteration}(\mathsf{a\_win})(g)$ if a_win was updated for a successor of a position $g \in G$. Since $\mathbf{0} \preceq \mathsf{a\_win} \preceq \mathsf{Win}_a^{\min}$ holds after each iteration of Lines 4 to 18 and the last the calculated a_win is a fixed point of Iteration, both algorithms output is $\mathsf{Win}_a^{\min}$.

2. Another noticeable change is that we left out the definition of defender_post and options and thereby omitted the conditional in Line 12 of Algorithm 2. Note that $\mathsf{defender\_post} \nsubseteq \mathsf{options}$ implies that the set calculated in Line 13 of Algorithm 2 is empty. Since $\operatorname{Min} \varnothing = \varnothing$, this case distinction is not necessary. However, when executing an implementation of these algorithms this conditional might save computing time.

3. The last difference that is not just a change of notation is in Line 8 of Algorithm 2, where $\operatorname{Min}(\mathsf{a\_win}[\mathsf{g}] \cup \{u^{-1}(e') \mid \mathsf{g} \rightarrowtail^u g' \wedge e' \in \mathsf{a\_win}[g']\})$ is calculated. Since $\mathsf{a\_win}[\mathsf{g}] \subseteq \uparrow \{u^{-1}(e') \mid \mathsf{g} \rightarrowtail^u g' \wedge e' \in \mathsf{a\_win}[g']\}$, we omitted $\mathsf{a\_win}[\mathsf{g}]$ in Line 7 of Algorithm 1.

To conclude this comparison we note that Algorithm 1 is easier to comprehend

and therefore its proof of correctness is simpler. Since both algorithms yield the same results, we proved the correctness of both versions.

5.4 Complexity

We now analyse the time and space complexity of Algorithm 1 and compare it to that of Algorithm 2. Since we gave an extensive proof of termination, we only outline some of the proofs in this subsection and do not formalise these results in Isabelle/HOL.

We start our complexity analysis by understanding the assignments of $\mathsf{new\_a\_win}$ during the application of Algorithm 1. For an example we refer to Table 5.1 in Example 5.2. Note that the first assignment of $\mathsf{new\_a\_win}$ in Line 2 is $\mathbf{0} = \mathsf{Iteration}^0(\mathbf{0})$. In the i-the iteration of the while-loop $\mathsf{Iteration}^i(\mathbf{0})$ is assigned to $\mathsf{new\_a\_win}$ for $i \in \mathbb{N}$.[31]

When applying Algorithm 1 to some Galois energy game $\mathcal{G}$, the first iteration of the while-loop assigns the set containing only the minimum of $\mathcal{E}$ to defender positions that are deadends. All other positions get assigned the empty set. In other words, $\mathsf{Iteration}^1(\mathbf{0})(g) \neq \varnothing$, if and only if there exists a play of length one starting in position g and ending in position where the defender is stuck.

If all successors of a defender position g_d were previously assigned a non-empty set, then a non-empty set is assigned to g_d in the next iteration of the while-loop. Similarly, if there exists a successor of an attacker position g_a that was previously assigned a non-empty set, then some non-empty set is assigned to g_a in the next iteration of the while-loop. After the first iteration, a position g is assigned a non-empty set for the first time if and only if the attacker can enforce a move from g to some successor that previously was assigned a non-empty set. In other words, $\mathsf{Iteration}^i(\mathbf{0})(g) \neq \varnothing$, if and only if the attacker can enforce a play of length $\leqslant i$ starting in position g and ending in a defender position that is a deadend.

This implies, that after $|G|$ iterations the information, which positions are winnable for the attacker has travelled from the defender positions that are deadends to all other winnable positions. We state this as the following lemma.

Lemma 5.9. *Let $\mathcal{G}$ be a Galois energy game and let u^{-1} be computable for all updates u in $\mathcal{G}$. Then $\mathsf{Win}_a(g) \neq \varnothing \longleftrightarrow \mathsf{Iteration}^{|G|}(\mathbf{0})(g) \neq \varnothing$ for all positions $g \in G$.*

For declining energy games an even stronger result holds: A fixed point is calculated within $|G|$ iterations of the while-loop. We state this as the following lemma.

Lemma 5.10. *Let $\mathcal{G}$ be a declining Galois energy game. Further, let u^{-1} be computable for all updates u in $\mathcal{G}$. Then $\mathsf{Win}_a(g) = \uparrow \mathsf{Iteration}^{|G|}(\mathbf{0})(g)$ for all positions $g \in G$.*

To prove this we construct a "minimal" attacker winning strategy.[32] This is inspired by similar constructions discussed by Brihaye and Goeminne [12] in

[31] Note $\mathsf{Iteration}^i(\mathbf{0})$ is defined for $i \in \mathbb{N}$ even if i is larger than the number of iterations before termination.

[32] In particular, we construct a strategy requiring memory of size $|G_a| \cdot w$ where w is an upper bound of the cardinality of minimal attacker winning budgets.

the proof of Proposition 2 and by Chatterjee et al. [14] as Lemma 11. To avoid formally introducing strategy trees, we only discuss the proof idea.

Proof sketch. The inclusion $\uparrow \mathsf{Iteration}^{|G|}(\mathbf{0})(g) \subseteq \mathsf{Win}_a(g)$ holds for all $g \in G$ by Lemma 5.6. We now outline a proof of $\mathsf{Win}_a(g) \subseteq \uparrow \mathsf{Iteration}^{|G|}(\mathbf{0})(g)$. For this we argue that in declining energy games an attacker wins at some position if and only if they can enforce a winning play with length $\leqslant |G|$ by constructing a "minimal" attacker winning strategy.
Let $g \in G$ and $e \in \mathsf{Win}_a(g)$, then there exists $e' \in \mathsf{Win}_a^{\min}(g)$ with $e' \leqslant e$ and an energy-positional attacker winning strategy s for $\mathcal{G}[g, e']$. If g is an attacker position, then $u(e') \in \mathsf{Win}_a(s(g, e'))$ holds for the update u with $g \rightarrowtail^{u} s(g, e')$ by Theorem 3. Since u is monotonic and attacker winning budgets are upward-closed, this implies $u(e) \in \mathsf{Win}_a(s(g, e'))$. By imitating the strategy s an attacker winning strategy s' for $\mathcal{G}[g, e]$ can be constructed such that all plays starting in g and consistent with s' are consistent with s. Following this idea an attacker winning strategy can be constructed, that maps all $(g_a, e_a) \in G_a \times \mathcal{E}$ with $e_a \in \mathsf{Win}_a(g_a)$ the same way some attacker winning strategy for $\mathcal{G}[g_a, e'_a]$ with $e'_a \in \mathsf{Win}_a^{\min}(g_a)$ and $e'_a \leqslant e_a$ would.
Since $\mathcal{G}$ is declining, all plays consistent with such a strategy are acyclic – otherwise the defender could enforce an infinite play or there would exist a smaller sufficient energy $e''_a < e'_a$. Thus, the length of all plays consistent with such a strategy is bounded by the number of positions. □

Lemma 5.10 directly implies the following.

Corollary 5.6. *Let $\mathcal{G}$ be a declining Galois energy game and let u^{-1} be computable for all updates u in $\mathcal{G}$. Then Algorithm 1 with input $\mathcal{G}$ terminates after at most $|G| + 1$ iterations of the while-loop.*

We again consider the more general case, i.e. that $\mathcal{G}$ might not be a declining energy game. Once a non-empty set was assigned as $\mathsf{new\_a\_win}[g]$ to position g in iteration i of the while-loop in Algorithm 1, it only improves, i.e. $\mathsf{Iteration}^i(\mathbf{0})(g) \subseteq \uparrow \mathsf{Iteration}^j(\mathbf{0})(g)$ for all $i, j \in \mathbb{N}$ with $i \leqslant j$. We use this property to calculate an upper bound of the size of the calculated sets $\mathsf{new\_a\_win}[g]$. For this we define $wu_{\mathcal{G}}$ as an upper bound of the worst energy that might be calculated during the first $|G|$ iterations of the while-loop in Algorithm 1. This is then an upper bound for all energies calculated and potentially added to any $\mathsf{new\_a\_win}[g]$ during the run of Algorithm 1.

Definition 5.7. *Let $\mathcal{G}$ be a Galois energy game and let m be the minimum in $\mathcal{E}$. Then $wu_{\mathcal{G}}$ is the least upper bound of the worst combination of at most $|G| - 1$ "inverse" updates applied to m, i.e. $wu_{\mathcal{G}} := \sup\{u_1^{-1} \circ ... \circ u_i^{-1}(m) \mid i \in \{0, ..., |G| - 1\} \ \wedge \ u_1, ..., u_i$ are updates in $\mathcal{G}\}$ where $u_1 \circ ... \circ u_i(m) := m$ for $i = 0$.*[33]

This definition directly implies the following lemma.

Lemma 5.11. *Let $\mathcal{G}$ be a Galois energy game and let u^{-1} be computable for all updates u in $\mathcal{G}$. Then $wu_{\mathcal{G}}$ is an upper bound for all energies added to $\mathsf{new\_a\_win}$ during some iteration, i.e. $\forall i \in \mathbb{N}.\ \forall g \in G.\ \forall e \in \mathsf{Iteration}^i(\mathbf{0})(g).\ e \leqslant wu_{\mathcal{G}}$.*

[33]Note that the set of updates in $\mathcal{G}$ is finite if the set of positions is. By Assumption 5.2 such a supremum always exists.

This allows us to define $\mathsf{pareto}(wu_{\mathcal{G}})$, a set depending on $wu_{\mathcal{G}}$ which contains all $\mathsf{new\_a\_win}[g]$ that might be calculated during the run of Algorithm 1.

Definition 5.8. *For $e \in \mathcal{E}$ we set*

$$\mathsf{pareto}(e) := \{A \in \mathcal{P}(\mathcal{E}) \mid A \text{ is an antichain and } \forall a \in A.\ a \leqslant e\}.$$

Then w_e is the maximal cardinality of elements in $\mathsf{pareto}(e)$[34] and h_e is the maximal length of sequences $A_0, A_1...$ in $\mathsf{pareto}(e)$ with $A_j \subsetneq \uparrow A_{j+1}$[35], i.e.

$$w_e := \sup_{\mathbb{N}}\{|A| \mid A \in \mathsf{pareto}(e)\} \qquad \text{and}$$
$$h_e := \sup_{\mathbb{N}}\{i \in \mathbb{N} \mid \exists A_0, ..., A_{i-1} \in \mathsf{pareto}(e).\ \forall j \in \{0, ..., i-2\}.\ A_j \subsetneq \uparrow A_{j+1}\}.$$

To state our complexity result we first introduce some more notation.

Notation 5.3. We identify the branching degree of an energy game with the branching degree of its underlying graph, i.e. the maximal number of successors any position has.

Theorem 5 applies only if some reasonable computability assumptions are made. We now give names to upper bounds of computing times.[36]

Notation 5.4. Let $\mathcal{G}$ be a Galois energy game. With $wi_{\mathcal{G}}$ we denote an upper bound for all updates u in $\mathcal{G}$ and all $e \in \mathcal{E}$ of the computing time needed to calculate $u^{-1}(e)$. Further, $s_{\mathcal{E}}$ is an upper bound for the computing time needed to calculate the supremum of two energies and $c_{\mathcal{E}}$ is an upper bound for the computing time needed to compare two elements.

Now we are able to over-approximate the running time of Algorithm 1 as follows.

Theorem 7 (Complexity)**.** *Let $\mathcal{G}$ be a Galois energy game with branching degree o. Further, let u^{-1} be computable for all updates u in $\mathcal{G}$. Then, Algorithm 1 on input $\mathcal{G}$ terminates in a time in*

$$\mathcal{O}\left(h_{wu_{\mathcal{G}}} \cdot |G|^2 \cdot o \cdot w_{wu_{\mathcal{G}}}^2 \cdot (wi_{\mathcal{G}} + s_{\mathcal{E}} + w_{wu_{\mathcal{G}}} \cdot c_{\mathcal{E}})\right).$$

If $\mathcal{G}$ is declining, then the running time is in

$$\mathcal{O}\left(|G|^2 \cdot o \cdot w_{wu_{\mathcal{G}}}^2 \cdot (wi_{\mathcal{G}} + s_{\mathcal{E}} + w_{wu_{\mathcal{G}}} \cdot c_{\mathcal{E}})\right).$$

In both cases the output is calculated using the space of at most $|G| \cdot w_{wu_{\mathcal{G}}}$ energies.

Proof. We start our running time analysis by over-approximating the number of iterations of the while-loop needed for Algorithm 1 to terminate on input $\mathcal{G}$. By Lemma 5.9 every position with a non-empty attacker winning budget is assigned some non-empty set after $|G|$ iterations of the while-loop. Since the assigned sets are in $\mathsf{pareto}(wu_{\mathcal{G}})$, those assigned sets cannot be truly updated more than

[34]Note that the cardinality of a maximum antichain in a partially ordered set is called the width of that partially ordered set. For this reason we choose the letter w in this definition.

[35]Note that the maximal cardinality of a chain in a partially ordered set is called the height of that partially ordered set. When defining a partial order accordingly, the sequences used to define $h_{wu_{\mathcal{G}}}$ can thereby be considered chains, which explains our choice of h in this definition.

[36]In some instances there might not exists such a finite upper bound. However, the calculations performed during the run of Algorithm 1 are bounded if the algorithm terminates.

$h_{wu_\mathcal{G}}$ times by Definition 5.8. Since one more iteration might be needed to obtain new_a_win = a_win, the number of iterations before termination does not exceed $|G| + |G| \cdot h_{wu_\mathcal{G}} + 1 \in \mathcal{O}(|G| \cdot h_{wu_\mathcal{G}})$. By Corollary 5.6 this number can be improved for declining energy games. Then, a number of iterations of the while-loop in $\mathcal{O}(|G|)$ is sufficient.
During one iteration, new_a_win[g] is calculated for all positions $g \in G$, i.e. for $|G|$ positions. We now analyse the computing time of those calculations by case distinction.

Case "g $\in G_a$": To calculate new_a_win[g] for an attacker positions g a set with no more elements than the branching degree times the maximal cardinality of previously calculated a_win[g'] is calculated and the minimum is applied. The cardinality of that set can be over-approximated by $o \cdot w_{wu_\mathcal{G}}$. The elements are energies of the form $u^{-1}(e)$ and can be calculated in $wi_\mathcal{G}$ time each. Thereby, the set can be calculated in $\mathcal{O}\left(o \cdot w_{wu_\mathcal{G}} \cdot wi_\mathcal{G}\right)$ time. Since the number of minimal elements of that set is bounded by $w_{wu_\mathcal{G}}$, each element has to be compared to at most $w_{wu_\mathcal{G}}$ other elements to apply the minimum, which can be done in $c_\mathcal{E}$ time per comparison, This adds up to a computing time in $\mathcal{O}\left(o \cdot w_{wu_\mathcal{G}} \cdot (wi_\mathcal{G} + w_{wu_\mathcal{G}} \cdot c_\mathcal{E})\right)$ to update new_a_win[g] once for an attacker position g.

Case "g $\in G_d$": To calculate new_a_win[g] for a defender position g the following procedure can be applied:[37]

```
1 def compute_new_a_win(G, a_win, g):
2 |   new[g] := Min E
3 |   for g' with g ⇀^u g':
4 |   |   new[g] := Min{sup{e', u^-1(e_g')} | e' ∈ new[g]
  |   |     ∧ e_g' ∈ a_win[g']}
5 |   return new[g]
```

Note that $\operatorname{Min}\mathcal{E}$ is the set containing only the minimum in $\mathcal{E}$, i.e. $\{\sup\varnothing\}$. Some energy e is in the returned new[g] if and only if for each g' with g $\rightharpoonup^u$ g' there exists some $e_{g'} \in$ a_win[g'] such that $e = \sup\left(\{u^{-1}(e_{g'}) \mid \text{g} \rightharpoonup^u g'\} \cup \operatorname{Min}\mathcal{E}\right) = \sup\{u^{-1}(e_{g'}) \mid \text{g} \rightharpoonup^u g'\}$ and e is minimal with this property. Thereby, compute_new_a_win yields the intended result seen in Line 9 in Algorithm 1.

We now consider the complexity of compute_new_a_win. The size of new[g] is bounded by $w_{wu_\mathcal{G}}$ at all times. The same applies to a_win[g'] for all $g' \in G$. Thereby, the cardinality of the set to which the minimum is applied in Line 4 is bounded by $w^2_{wu_\mathcal{G}}$. Each element of that set is calculated by taking the supremum of an energy of the form $u^{-1}(e_{g'})$ and some previously calculated energy. This implies that this set can be computed in a time in $\mathcal{O}\left(w^2_{wu_\mathcal{G}} \cdot (wi_\mathcal{G} + s_\mathcal{E})\right)$. Applying the minimum entails no more than $w^2_{wu_\mathcal{G}} \cdot w_{wu_\mathcal{G}}$ comparisons, each executed in a time bounded by $c_\mathcal{E}$. With this we can conclude a running time of compute_new_a_win in $\mathcal{O}\left(1 + o \cdot w^2_{wu_\mathcal{G}} \cdot (wi_\mathcal{G} + s_\mathcal{E} + w_{wu_\mathcal{G}} \cdot c_\mathcal{E})\right)$. Since the algorithm for a graph with branching degree equal to zero is trivial, we

[37] Note that this procedure is inspired by Brihaye and Goeminne [12].

may assume $1 \leqslant o$ and thereby a running time of compute_new_a_win in $\mathcal{O}\left(o \cdot w_{wu_\mathcal{G}}^2 \cdot (wi_\mathcal{G} + s_\mathcal{E} + w_{wu_\mathcal{G}} \cdot c_\mathcal{E})\right)$.

Note that the computing time for attacker positions is dominated by that of defender positions. Therefore, calculating new_a_win[g] for any position $\mathsf{g} \in G$ from previously calculated a_win[g'] is possible in $\mathcal{O}\left(o \cdot w_{wu_\mathcal{G}}^2 \cdot (wi_\mathcal{G} + s_\mathcal{E} + w_{wu_\mathcal{G}} \cdot c_\mathcal{E})\right)$ time. In total each iteration can be calculated in $\mathcal{O}\left(|G| \cdot o \cdot w_{wu_\mathcal{G}}^2 \cdot (wi_\mathcal{G} + s_\mathcal{E} + w_{wu_\mathcal{G}} \cdot c_\mathcal{E})\right)$. Note that in big O notation the assignment in Line 4 in Algorithm 1 is not relevant to the computing time of one iteration. The same holds true for checks performed in Line 10. Similarly, the computing time of Line 2 is not relevant in big O notation. In total this adds (or rather factors) up to a running time of the algorithm in $\mathcal{O}\left(h_{wu_\mathcal{G}} \cdot |G|^2 \cdot o \cdot w_{wu_\mathcal{G}}^2 \cdot (wi_\mathcal{G} + s_\mathcal{E} + w_{wu_\mathcal{G}} \cdot c_\mathcal{E})\right)$. In the case of declining energy games we may omit $h_{wu_\mathcal{G}}$.
Since each calculated new_a_win[g] contains at most $w_{wu_\mathcal{G}}$ elements, the space needed to compute the output is the space of $|G| \cdot w_{wu_\mathcal{G}}$ energies. □

We now consider the running time when applying Algorithm 1 to Bisping's declining energy games.

Lemma 5.12. *Let $\mathcal{G}$ be an n-dimensional Bisping's energy game for some $n \in \mathbb{N}$. When applied to $\mathcal{G}$, Algorithm 1 has a running time in $\mathcal{O}\left(o \cdot |G|^{2 \cdot n} \cdot \left(n^2 + |G|^{n-1} \cdot n\right)\right)$ using $\mathcal{O}\left(n \cdot |G|^n\right)$ space for the output.*

Proof. We prove this by using the complexity results for declining energy games stated in Theorem 7. Substituting suitable over-approximations of $w_{wu_\mathcal{G}}$, $wi_\mathcal{G}$, $c_\mathcal{E}$ and $s_\mathcal{E}$ results in the stated complexity.
Consider how much each component of an energy might decrease when an update u in $\mathcal{G}$ is applied. By Definition 4.1 each component strictly decreases only if one is subtracted or the component is replaced with some minimum. Thereby, the application of any combination of up to $|G| - 1$ updates $u_1, ..., u_i \in \mathcal{U}_n$ to $(|G|, ..., |G|) \in \mathbb{N}_\infty^n$ yields a result larger than the zero-vector, i.e. $(0, ..., 0) \leqslant u_i \circ ... \circ u_1(|G|, ..., |G|)$. This implies $u_1^{-1} \circ ... \circ u_i^{-1}(0, ..., 0) \leqslant u_1^{-1} \circ ... \circ u_i^{-1} \circ u_i \circ ... \circ u_1(|G|, ..., |G|) \leqslant (|G|, ..., |G|)$, since u^{-1} is monotonic and $u^{-1} \circ u$ is decreasing for all updates $u \in \mathcal{U}_n$. From Definition 5.7 then follows $wu_\mathcal{G} \leqslant (|G|, ..., |G|) \in \mathbb{N}_\infty^n$.
Note that the set $A := \{e \in \mathbb{N}_\infty^n \mid e \leqslant (|G|, ..., |G|)\}$ has $(|G| + 1)^n$ elements and $\mathsf{pareto}(wu_\mathcal{G}) \subseteq \mathcal{P}(A)$ holds. Let $B \subseteq A$ such that $(|G| + 1)^{n-1} + 1 \leqslant |B|$. Then there exist distinct elements $b, b' \in B$ only differing in one entry. Then $b < b'$ or $b' < b$ holds. Thereby $w_{wu_\mathcal{G}}$, the cardinality of sets in $\mathsf{pareto}(wu_\mathcal{G})$, is bounded by $(|G| + 1)^{n-1} \in \mathcal{O}\left(|G|^{n-1}\right)$.
Further, note that an energy of the form $u^{-1}(e)$ can be calculated component-wise and for each component at most n values have to be considered as the potential maximum. This implies $wi_\mathcal{G} \in \mathcal{O}(n^2)$. Since the supremum in $\mathbb{N}_\infty^n$ is calculated component-wise too, we have $s_\mathcal{E} \in \mathcal{O}(n)$ and thereby $wi_\mathcal{G} + s_\mathcal{E} \in \mathcal{O}(n^2)$. Similarly, comparing two energies is computed by component-wise comparisons in $\mathcal{O}\left(n\right)$ time.
Assuming the space needed for one natural number to be in $\mathcal{O}\left(1\right)$, each energy needs space in $\mathcal{O}(n)$. □

We now compare this to the complexity results stated as Lemma 6 by Bisping [7] by first restating the latter.

Lemma 5.13. *Let $n \in \mathbb{N}$ and let $\mathcal{G} = (G, G_a, \rightarrowtail)$ be a Bisping's energy game over $\mathbb{N}_\infty^n$ with a finite set of positions G. Further, let o be the branching degree of $\mathcal{G}$. Then Algorithm 2 terminates on input $\mathcal{G}$ in $\mathcal{O}\left(|\rightarrowtail| \cdot |G|^n \cdot (o + |G|^{(n-1)o})\right)$ time using $\mathcal{O}(|G|^n)$ space for the output.*

Remark 5.3. Since both, Algorithm 1 and Algorithm 2, calculate the same result, the same space complexity is to be expected. However, when considering the dimension n Theorem 7 and the approximations seen in Corollary 5.12 imply a bound of the needed space in $\mathcal{O}(n \cdot |G|^n)$.
Since the running time stated in Corollary 5.12 is polynomial for a fixed dimension, it is a significant improvement compared to Lemma 5.13. Note that this is not a true comparison of actual running times of Algorithm 1 and Algorithm 2. Instead, we only compared over-approximations of worst running times. We used finer approximations resulting in an unfair comparison. In Line 13 of Algorithm 2 a procedure like compute_new_a_win could be applied, resulting in a running time similar to the one stated in Corollary 5.12.

Note that the actual computing time heavily depends on the implementation. Using parallelisation Vogel [31] was able to improve the actual running times by factor 10 compared to a implementation of Algorithm 2 by Bisping.

Chapter 6

Generalisations

In this chapter we discuss and contextualise our main result, Theorem 5. To this end we study the class of energy games it applies to, i.e. where the following condition is fulfilled:

1. The energy game is a Galois energy game such that u^{-1} is computable for all updates u.

This can be understood as assumptions about updates. Note that we formulated further conditions through Assumption 5.1 , Assumption 5.2 and Assumption 5.3. To make those explicit, add the following conditions.

2. The energies form a well-founded bounded join-semilattice such that the supremum and the minimal elements of a finite set are computable.
3. The set of positions is finite.

Through Definition 3.5 we fixed winning conditions for all energy games considered in this thesis. In a different setting, other winning conditions might be necessary, for example parity winning conditions [4]. Therefore, Definition 3.5 can be understood as our fourth condition needed for Theorem 5 to be applicable. For this reason we investigate the following four axes of generalisation:

1. Properties of updates.
2. Properties of energies.
3. Properties of the underlying graph.
4. Winning conditions.

We split these into two sections. In Section 6.1 we focus on the properties of updates. First, we discuss that properties of Galois energy games are not sufficient and state undecidability of that class. Secondly, we argue that monotonicity is the essential factor in our reasoning. In particular, we are able to generalise Theorem 5 to a larger class of monotonic energy games. In Section 6.2 we start by discussing the necessary properties of energies and the game graph. Afterwards, we discuss the implications and limitations of our results in regards to other winning conditions. There we consider related games with varying winning conditions and suggest further generalisations.

6.1 Generalising Updates

We start this section by discussing updates that prevent the application of Theorem 5 in Section 6.1.1. In particular we are able to state undecidability for the class of Galois energy games. Afterwards, we consider monotonic energy games that are not Galois energy games. In Section 6.1.2 we then discuss a generalisation of Algorithm 1 and study a larger class of decidable energy games, namely *generalised Galois energy games*. In Section 6.1.3 we then provide an overview of all classes of energy games introduced in this thesis.

6.1.1 Properties of Updates

We now consider the properties of updates required for the application of Theorem 5. For this we first state a version of the theorem that is equivalent by Remark 4.4.

Corollary 6.1. *Let $\mathcal{G}$ be a monotonic energy game. If $\min\{e \mid e' \leqslant u(e)\}$ exists and is computable for all updates u in $\mathcal{G}$ and for all $e' \in \mathcal{E}$, then $\mathcal{G}$ is decidable.*

For an energy game over $\mathcal{E}$ to be decidable by Theorem 5 $\min\{e \mid e' \leqslant u(e)\}$ has to exist and be computable for all updates u and energies $e' \in \mathcal{E}$. In this subsection we discuss updates that prevent the application of Theorem 5. We first give an example of a Galois energy game where $\mathrm{inv}(u)$ is not computable for some update u. We construct such an energy game utilising the halting problem.

Example 6.1. We consider $\{0,1\}^*$ with the length-based order $\leqslant_{\text{len}}$, i.e. $x \leqslant_{\text{len}} y$ if $|x| < |y|$ or $|x| = |y| \wedge x \leqslant_{\text{lex}} y$ for all $x, y \in \{0,1\}^*$where $\leqslant_{\text{lex}}$ refers to the lexicographical order. This order is well-founded on $\{0,1\}^*$ and a bounded join-semilattice with minimum ϵ.
Let $\mathcal{G} = (\{g_0, g_1\}, \varnothing, \rightarrowtail)$ be an energy game over $\{0,1\}^*$ with the lexicographic order $\leqslant_{\text{len}}$. Further, let $\rightarrowtail$ be given by $g_0 \overset{f}{\rightarrowtail} g_1$ with $f : \{0,1\}^* \to \{0,1\}^*$. We assume there to be some encoding of Turing machines and use that to define $TM : \{0,1\}^* \to \{0,1\}^*$ by setting $TM(x) = \max_{\text{len}}\{m \mid m \leqslant_{\text{len}} x \wedge m \text{ encodes a Turing machine}\}$ for all $x \in \{0,1\}^*$ where such a maximum exists. Let f append 0 to $x \in \{0,1\}^*$, if $x \in \mathrm{dom}(TM)$ and the by $TM(x)$ encoded Turing machine halts on input $TM(x)$. Otherwise, let f append 1.
Note that f is a total function and thereby $\mathrm{dom}(f) = \{0,1\}^*$ is upward-closed. Further, we show that f is monotonic. Let $x, y \in \{0,1\}^*$ with $x \leqslant_{\text{len}} y$. Then $f(x) = xa$ and $f(y) = yb$ for some $a, b \in \{0,1\}$. Note that $yb = f(y) <_{\text{len}} f(x) = xa$ implies $x = y$ and $b < a$ since $x \leqslant_{\text{len}} y$. Since $x = y$ implies $a = b$, this would be a contradiction and we can conclude $f(x) \leqslant_{\text{len}} f(y)$. We thereby have shown that $\mathcal{G}$ is a monotonic energy game. Since $(\{0,1\}^*, \leqslant_{\text{len}})$ is well-founded and the set $\{e \mid e' \leqslant_{\text{len}} f(e)\}$ is non-empty for all $e' \in \{0,1\}^*$, the minimum $\min_{\text{len}}\{e \mid e' \leqslant_{\text{len}} f(e)\}$ always exists. Thereby, $\mathcal{G}$ is a Galois energy game.
For any Turing machine with encoding $m \in \{0,1\}^*$, we have $TM(m) = m$. Then $f(m) = m0$ if the Turing machine halts on input m and $f(m) = m1$ otherwise. Thereby $m = \min_{\text{len}}\{e \mid m1 \leqslant_{\text{len}} f(e)\}$ if and only if the Turing machine does not halt on input m. The halting problem implies that $\min_{\text{len}}\{e \mid e' \leqslant_{\text{len}} f(e)\}$ is not computable for all $e' \in \{0,1\}^*$.

We now use this example to construct an undecidable Galois energy game.

Theorem 8 (Undecidability)**.** *The class of Galois energy games over well-founded bounded join-semilattices with finite sets of positions is not decidable.*

Proof. We construct energy games depending on a Turing machine such that the fixed initial credit problem is undecidable. In this construction we add a position to the Galois energy game considered in Example 6.1.
For $m \in \{0,1\}^*$ let $\mathcal{G}_m = (\{g_0, g_1, g_2\}, \varnothing, \rightarrowtail)$ be an energy game over $(\{0,1\}^*, \leqslant_{\text{len}})$. Further, let $\rightarrowtail$ be given by $g_0 \overset{f}{\rightarrowtail} g_1$ and $g_1 \overset{f_m}{\rightarrowtail} g_2$ with f as defined in Example 6.1 and f_m the identity function on $\text{dom}(f_m) := \uparrow_{\text{len}} \{m1\}$. The domain of f_m is upward closed by definition, f_m is monotonic and $\min\{e \mid e' \leqslant_{\text{len}} f_m(e)\}$ exists for all $e' \in \{0,1\}^*$. This combined with the findings of Example 6.1 implies that $\mathcal{G}_m$ is a Galois energy game.
For any Turing machine with encoding $m \in \{0,1\}^*$, we then have $m \in \mathsf{Win}_a(g_0)$ in $\mathcal{G}_m$ if and only if that Turing machine does not halt on input m. The haling problem implies undecidability. □

We now discuss energy games where the existence instead of the computability of minima is the issue. In general an energy game over Bisping's energies is not a Galois energy game, if there exists an update, which updates entries of an energy vector in one of the following manners:

1. Replacing it with the maximum of some other entries.
2. Adding another entry to it.
3. Replacing it with a constant.

We substantiate this claim with the following example.

Example 6.2. We consider three possible updates u that might appear in a 2-dimensional monotonic energy game over Bisping's energies. We show that these updates do not appear in a Galois energy game. Using Lemma 4.6 we show that $\min\{e \mid e' \leqslant u(e)\}$ does not exist for some $e' \in \mathbb{N}_\infty^2$. Note that all three updates are monotonic with $\text{dom}(u) = \mathbb{N}_\infty^2$ being upward-closed.

1. Consider $u : \mathbb{N}_\infty^2 \to \mathbb{N}_\infty^2,\ (e_0, e_1) \mapsto (\max_{\mathbb{N}}\{e_0, e_1\}, \max_{\mathbb{N}}\{e_0, e_1\})$. Then $(0,1), (1,0) \in \text{Min}\{(e_0, e_1) \mid (1,1) \leqslant u(e_0, e_1) = (\max_{\mathbb{N}}\{e_0, e_1\}, \max_{\mathbb{N}}\{e_0, e_1\})\}$. Therefore, $\min\{e \mid (1,1) \leqslant u(e)\}$ does not exist.
2. Consider $u : \mathbb{N}_\infty^2 \to \mathbb{N}_\infty^2,\ (e_0, e_1) \mapsto (e_0 + e_1, e_0 + e_1)$. Then $(0,1), (1,0) \in \text{Min}\{(e_0, e_1) \mid (1,1) \leqslant u(e_0, e_1) = (e_0 + e_1, e_0 + e_1)\}$. Therefore, $\min\{e \mid (1,1) \leqslant u(e)\}$ does not exist.
3. Consider $u : \mathbb{N}_\infty^2 \to \mathbb{N}_\infty^2,\ (e_0, e_1) \mapsto (0,0)$. Then $\text{Min}\{(e_0, e_1) \mid (1,1) \leqslant u(e_0, e_1) = (0,0)\} = \varnothing$. Therefore, $\min\{e \mid (1,1) \leqslant u(e)\}$ does not exist.

Note that these examples of updates can be scaled to higher dimensions by defining updates that exhibit the same behaviour in the first two components while not changing other entries and by adding zero-entries to the energy vectors.

6.1.2 Generalised Algorithm

Example 6.2 illustrates the limitations of Galois energy games and thus of our main result, Theorem 5. In this section we further investigate decidability of monotonic energy games. While we are unable to show that all monotonic energy games are decidable, we are able to generalise our results to a class containing energy games with the updates mentioned in Example 6.2. This class will be introduced as *generalised Galois energy games*. We first discuss our generalised algorithm, Algorithm 3. To prove that this algorithm indeed calculates minimal attacker winning budgets in a finite amount of time we follow the proof as outlined in Section 5.2. While producing similar results, we aim to not duplicate proofs and therefore choose to shorten our arguments.
The core idea of Algorithm 3 is to solely utilise the inductive characterisation of attacker winning budgets, i.e. Theorem 3, and attacker winning budgets being upward-closed as stated in Lemma 3.2. More precisely we do not change the structure of Algorithm 3. The only alterations appear in Line 7 and Line 9.

```
1  def compute_winning_budgets(𝒢 = (G, G_a, ↦)):
2      new_a_win := [g ↦ {} | g ∈ G]
3      do
4          a_win := new_a_win
5          for g ∈ G :
6              if g ∈ G_a :
7                  new_a_win[g] := Min{e | g ↣^u g' ∧ u(e) ∈ ↑ a_win[g']}
8              else:
9                  new_a_win[g] := Min{sup{e_g' | g ↣^u g'} |
                     ∀g'. g ↣^u g' ⟶ u(e_g') ∈ ↑ a_win[g']}
10     while new_a_win ≠ a_win
11     return a_win
```

Algorithm 3: Generalised algorithm for computing minimal attacker winning budgets of monotonic energy game $\mathcal{G}$.

Similar to Definition 5.6 we start our proof of correctness by defining $\mathsf{Iteration}'$, which corresponds to one application of the while-loop of Algorithm 3.

Definition 6.1. *Define* $\mathsf{Iteration}' : \mathsf{PARETO}_{\mathcal{G}} \to \mathsf{PARETO}_{\mathcal{G}}$ *by setting*

$$\mathsf{Iteration}'(F)(g_a) := \mathrm{Min}\{e \mid g_a \overset{u}{\rightarrowtail} g' \wedge u(e) \in \uparrow F(g')\}$$
$$\mathsf{Iteration}'(F)(g_d) := \mathrm{Min}\{\sup\{e_{g'} \mid g_d \overset{u}{\rightarrowtail} g'\} \mid \forall g'.\ g_d \overset{u}{\rightarrowtail} g' \longrightarrow u(e_{g'}) \in \uparrow F(g')\}$$

for all attacker positions $g_a \in G_a$, defender positions $g_d \in G_d$ and possible Pareto fronts $F \in \mathsf{PARETO}_{\mathcal{G}}$.[38]

Similar to Lemma 5.5, we now show that $\mathsf{Iteration}'$ is Scott-continuous for monotonic energy games.

Lemma 6.1. *Let $\mathcal{G}$ be a monotonic energy game. Then* $\mathsf{Iteration}'$ *is Scott-continuous.*

[38] Note that, by the use of Min in the definition of $\mathsf{Iteration}'$, this actually defines a functor of possible Pareto fronts.

Proof. Let $P \subseteq \mathsf{PARETO}_{\mathcal{G}}$ be a directed set. Note that $F \preceq F'$ for $F, F' \in \mathsf{PARETO}_{\mathcal{G}}$ implies $\uparrow F(g) \subseteq \uparrow F'(g)$ for all $g \in G$ and thereby $\mathsf{Iteration}'(F) \preceq \mathsf{Iteration}'(F')$ by Definition 6.1. Therefore, $\mathsf{Iteration}'$ is monotonic, which implies $\sup_{\preceq}\{\mathsf{Iteration}'(F) \mid F \in P\} \preceq \mathsf{Iteration}'(\sup_{\preceq} P)$.
What remains to show is $\mathsf{Iteration}'(\sup_{\preceq} P) \preceq \sup_{\preceq}\{\mathsf{Iteration}'(F) \mid F \in P\}$. We show $\mathsf{Iteration}'(\sup_{\preceq} P)(g) \subseteq \uparrow (\sup_{\preceq}\{\mathsf{Iteration}'(F) \mid F \in P\})(g)$ for all $g \in G$ by case distinction.

Case "$g_a \in G_a$": Let $g_a \in G_a$ and $e \in \mathsf{Iteration}'(\sup_{\preceq} P)(g_a)$. Then there exists $g' \in G$ with $g_a \overset{u}{\rightarrowtail} g'$ and $u(e) \in \uparrow (\sup_{\preceq} P)(g')$. Further, there exists $F \in P$ such that $u(e) \in \uparrow F(g')$. This implies $e \in \uparrow \mathsf{Iteration}'(F)(g_a)$ and thereby $e \in \uparrow (\sup_{\preceq}\{\mathsf{Iteration}'(F) \mid F \in P\})(g_a)$.

Case "$g_d \in G_d$": Let $g_d \in G_d$ and $e \in \mathsf{Iteration}'(\sup_{\preceq} P)(g_d)$. For each $g' \in G$ with $g_d \overset{u}{\rightarrowtail} g'$ there exists $e_{g'}$ such that $u(e_{g'}) \in \uparrow (\sup_{\preceq} P)(g')$ and $e = \sup\{e_{g'} \mid g_d \overset{u}{\rightarrowtail} g'\}$. Further, for each $g' \in G$ with $g_d \overset{u}{\rightarrowtail} g'$ there exists $F_{g'} \in P$ with $u(e_{g'}) \in \uparrow F_{g'}(g')$. Since the set of positions is finite and P is directed, there exists $F \in P$ with $F_{g'} \preceq F$ for all $g' \in G$ with $g_d \overset{u}{\rightarrowtail} g'$. This implies $e \in \uparrow \mathsf{Iteration}'(F)(g_d)$ and thus $e \in \uparrow (\sup_{\preceq}\{\mathsf{Iteration}'(F) \mid F \in P\})(g_d)$. □

Lemma 6.1 implies that Kleene's fixed point theorem can be applied. With this we can show the following lemma.

Lemma 6.2. *Let $\mathcal{G}$ be a monotonic energy game. Then $\mathsf{Win}_a^{\min}$ is a least fixed point of $\mathsf{Iteration}'$ in $\mathsf{PARTEO}_{\mathcal{G}}$.*

Proof. This proofs' structure closely resembles the proof of Lemma 5.6. We inductively define the set S' as follows.

$$\frac{g \in G_a \qquad g \overset{u}{\rightarrowtail} g' \qquad u(e) \in \uparrow S'(g')}{e \in S'(g)}$$

$$\frac{g \in G_d \qquad \forall g'.\ g \overset{u}{\rightarrowtail} g' \longrightarrow u(e_{g'}) \in \uparrow S'(g')}{\left(\sup\{e_{g'} \mid g \overset{u}{\rightarrowtail} g'\}\right) \in S'(g)}$$

By this definition we have $\mathsf{Iteration}'(S') = \mathrm{Min} \circ S'$. Since $\uparrow \mathrm{Min}\, S'(g) = \uparrow S'(g)$ for all $g \in G$, we have $\mathsf{Iteration}'(\mathrm{Min} \circ S') = \mathrm{Min} \circ S'$, i.e. $\mathrm{Min} \circ S'$ is a fixed point of $\mathsf{Iteration}'$. We now show $\mathsf{Win}_a^{\min} = \mathrm{Min} \circ S'$ by showing $\mathsf{Win}_a(g) = \uparrow S'(g)$ for all $g \in G$ using mutual inclusion.

"$\subseteq$" We show $\mathsf{Win}_a(g) \subseteq \uparrow S'(g)$ for all $g \in G$ by induction using the inductive characterisation of Win_a in Theorem 3. Let $g \in G$ and assume $e \in \mathsf{Win}_a(g)$. If $g \in G_a$, we further assume that there exists $g' \in G$ with $g \overset{u}{\rightarrowtail} g'$ and $u(e) \in \mathsf{Win}_a(g') \cap \uparrow S'(g')$. Then $e \in S'(g)$ by definition. If $g \in G_d$, we assume $u(e) \in \mathsf{Win}_a(g') \cap \uparrow S'(g')$ for all $g' \in G$ with $g \overset{u}{\rightarrowtail} g'$. Then let $e_{g'} := e$ for all $g' \in G$ with $g \overset{u}{\rightarrowtail} g'$. If there exists at least one successor of g, then $e = \sup\{e_{g'} \mid g \overset{u}{\rightarrowtail} g'\} \in S'(g)$ by definition of S'. If there is no successor of g, then $\sup \varnothing \in S'(g)$ is the minimal element of $\mathcal{E}$ and thus $e \in \uparrow S'(g)$. In all cases $e \in \mathsf{Win}_a(g)$ implies $e \in \uparrow S'(g)$.

"$\supseteq$" By Lemma 3.2 it suffices to show $S'(g) \subseteq \mathsf{Win}_a(g)$ for all $g \in G$. We show this by induction using the structure of S'. Let $g \in G$ and assume $e \in S'(g)$. If $g \in G_a$, we further assume that there exists $g' \in G$ with $g \overset{u}{\rightarrowtail} g'$ and $e' \leqslant u(e)$ for some $e' \in \mathsf{Win}_a(g') \cap S'(g')$. By Lemma 3.2 this implies $u(e) \in \mathsf{Win}_a(g')$ and thereby $e \in \mathsf{Win}_a(g)$ by Theorem 3. If $g \in G_d$, we assume that there exists $e_{g'}$ for each $g' \in G$ with $g \overset{u}{\rightarrowtail} g'$ such that $e = \sup\{e_{g'} \mid g \overset{u}{\rightarrowtail} g'\}$ and further $e'_{g'} \leqslant u(e_{g'})$ for some $e'_{g'} \in \mathsf{Win}_a(g') \cap S'(g')$. Since $\mathcal{G}$ is monotonic, we then have $e'_{g'} \leqslant u(e_{g'}) \leqslant u(e)$ and $u(e) \in \mathsf{Win}_a(g')$ for all $g' \in G$ with $g \overset{u}{\rightarrowtail} g'$ by Lemma 3.2. Then Theorem 3 implies $e \in \mathsf{Win}_a(g)$. In all cases $e \in \mathsf{Win}_a(g)$ follows from $e \in S'(g)$.

By Lemma 6.1 and Kleene's fixed point theorem there exists a least fixed point $F \in \mathsf{PARETO}_{\mathcal{G}}$ of $\mathsf{Iteration}'$ and $F \preceq \mathsf{Win}_a^{\min}$ holds, since $\mathsf{Win}_a^{\min}$ is a fixed point of $\mathsf{Iteration}'$. Analogously to our proof of $\mathsf{Win}_a(g) \supseteq \uparrow S'(g)$ for all $g \in G$, a simple induction over the structure of S' shows $S'(g) \subseteq \uparrow F(g)$ for all $g \in G$ by utilising that F is a fixed point. Therefore, we have $\mathsf{Win}_a^{\min}(g) = \mathrm{Min} \circ S'(g) \subseteq \uparrow F(g)$ for all $g \in G$ and thereby $\mathsf{Win}_a^{\min} \preceq F$. Thus $\mathsf{Win}_a^{\min} = F$ is the least fixed point of $\mathsf{Iteration}'$. □

We now argue for termination. Note that the same reasoning used in the proof of Lemma 5.8 yields the following lemma. We omit duplicating that proof.

Lemma 6.3. *Let $\mathcal{G}$ be a monotonic energy game. Then there exists $i \in \mathbb{N}$ such that $\mathsf{Win}_a^{\min} = \mathsf{Iteration}'^i(\mathbf{0})$.*

What remains to show is the counterpart to Lemma 5.7, which we state as the following lemma. Note that we again assume some computability.

Lemma 6.4. *Let $\mathcal{G}$ be a monotonic energy game. If $\mathrm{Min}\{e \mid e' \leqslant u(e)\}$ is computable for all $e' \in \mathcal{E}$ and all updates u in $\mathcal{G}$, then $\mathsf{Iteration}'^i(\mathbf{0})$ is computable for all $i \in \mathbb{N}$.*

Proof. We show that $\mathsf{Iteration}'^i(\mathbf{0})$ can be calculated with a finite number of simple operations by induction over $i \in \mathbb{N}$. Since $\mathsf{Iteration}'^0(\mathbf{0}) = (\mathbf{0})$ we now assume $\mathsf{Iteration}'^i(\mathbf{0})$ to be computable for a fixed $i \in \mathbb{N}$ and show computability of $\mathsf{Iteration}'^{i+1}(\mathbf{0})(g)$ for all $g \in G$ by case distinction.

Case "$g_a \in G_a$": Assume $g_a \in G_a$. For more readability we introduce two abbreviations. First, we introduce N to abbreviate a version of the neighbourhood of g_a, more precisely let $N := \{(g', u) \mid g_a \overset{u}{\rightarrowtail} g'\}$. Note that $\mathrm{Min}\,\mathsf{Iteration}'^i(\mathbf{0})(g') = \mathsf{Iteration}'^i(\mathbf{0})(g')$ for all $g' \in G$ with $g_a \overset{u}{\rightarrowtail} g'$ by definition of $\mathsf{Iteration}'$. Let $\mathsf{It}(g') := \mathsf{Iteration}'^i(\mathbf{0})(g')$ for all $g' \in G$ with

$g_a \rightarrowtail^u g'$. Then the following equations hold.

$$
\begin{aligned}
&\mathsf{Iteration}'^{i+1}(\mathbf{0})(g_a)\\
&= \mathrm{Min}\{e \mid g_a \rightarrowtail^u g' \wedge u(e) \in \uparrow \mathsf{Iteration}'^{i}(\mathbf{0})(g')\}\\
&= \mathrm{Min}\{e \mid \exists (g', u) \in N.\ u(e) \in \uparrow \mathsf{It}(g')\}\\
&= \mathrm{Min} \bigcup_{(g',u)\in N} \{e \mid u(e) \in \uparrow \mathsf{It}(g')\}\\
&= \mathrm{Min} \bigcup_{(g',u)\in N} \{e \mid \exists e' \in \mathsf{It}(g').\ e' \leqslant u(e)\}\\
&= \mathrm{Min} \bigcup_{(g',u)\in N} \bigcup_{e'\in \mathsf{It}(g')} \{e \mid e' \leqslant u(e)\}\\
&= \mathrm{Min} \bigcup_{(g',u)\in N} \bigcup_{e'\in \mathsf{It}(g')} \mathrm{Min}\{e \mid e' \leqslant u(e)\}
\end{aligned}
$$

Since the set of positions is finite, so is N. Further, N is computable and we assumed $\mathsf{It}(g')$ to be computable for all $g' \in G$. Note that $\mathsf{It}(g')$ is finite for all $g' \in G$ by Assumption 5.1 and Lemma 2.8. Calculating $\mathsf{Iteration}'^{i+1}(\mathbf{0})(g_a)$ therefore entails calculating $\mathrm{Min}\{e \mid e' \leqslant u(e)\}$ for finitely many e' and subsequently applying the minimum. Since $\mathrm{Min}\{e \mid e' \leqslant u(e)\}$ is computable for all $e' \in \mathcal{E}$, this yields the intended result.

Case "$g_d \in G_d$": Assume $g_d \in G_d$. For more readability we again introduce some abbreviations. We reuse N to abbreviate a version of the neighbourhood of g_d, this time let $N := \{g' \mid g_d \rightarrowtail^u g'\}$. Again we set $\mathsf{It}(g') := \mathrm{Min}\,\mathsf{Iteration}'^{i}(\mathbf{0})(g') = \mathsf{Iteration}'^{i}(\mathbf{0})(g')$ for all $g' \in G$ with $g_d \rightarrowtail^u g'$. To make the implicitly used function to index energies explicit we introduce sets of possible index functions. Let $\mathrm{Idx}' = \{f' : N \to \mathcal{E} \mid \forall g' \in N.\ f'(g') \in \mathsf{It}(g')\}$. Further, let $\mathrm{Idx}(f') := \{f : N \to \mathcal{E} \mid \forall g' \in N.\ f'(g') \leqslant u(f(g'))\}$ for all $f' : N \to \mathcal{E}$ and $\mathrm{Idx}_{\min}(f') := \{f : N \to \mathcal{E} \mid \forall g' \in N.\ f(g') \in \mathrm{Min}\{e \mid f'(g') \leqslant u(e)\}\}$ for all $f' : N \to \mathcal{E}$. Then the following equations hold.

$$
\begin{aligned}
&\mathsf{Iteration}'^{i+1}(\mathbf{0})(g_d)\\
&= \mathrm{Min}\{\sup\{e_{g'} \mid g_d \rightarrowtail^u g'\} \mid \forall g'.\ g_d \rightarrowtail^u g' \longrightarrow u(e_{g'}) \in \uparrow \mathsf{Iteration}'^{i}(\mathbf{0})(g')\}\\
&= \mathrm{Min}\{\sup\{e_{g'} \mid g' \in N\} \mid \forall g' \in N.\ u(e_{g'}) \in \uparrow \mathsf{It}(g')\}\\
&= \mathrm{Min}\{\sup\{e_{g'} \mid g' \in N\} \mid \forall g' \in N.\ \exists e'_{g'} \in \mathsf{It}(g').\ e'_{g'} \leqslant u(e_{g'})\}\\
&= \mathrm{Min}\{\sup\{e_{g'} \mid g' \in N\} \mid \exists f' \in \mathrm{Idx}'.\ \forall g' \in N.\ f'(g') \leqslant u(e_{g'})\}\\
&= \mathrm{Min}\{\sup\{f(g') \mid g' \in N\} \mid \exists f' \in \mathrm{Idx}'.\ f \in \mathrm{Idx}(f')\}\\
&= \mathrm{Min} \bigcup_{f'\in \mathrm{Idx}'} \{\sup\{f(g') \mid g' \in N\} \mid f \in \mathrm{Idx}(f')\}\\
&= \mathrm{Min} \bigcup_{f'\in \mathrm{Idx}'} \bigcup_{f\in \mathrm{Idx}(f')} \{\sup\{f(g') \mid g' \in N\}\}\\
&= \mathrm{Min} \bigcup_{f'\in \mathrm{Idx}'} \bigcup_{f\in \mathrm{Idx}_{\min}(f')} \{\sup\{f(g') \mid g' \in N\}\}
\end{aligned}
$$

Since the set of positions is finite, so is N and N is computable. Further, we assumed $\mathsf{lt}(g')$ to be computable for all $g' \in G$. Note that $\mathsf{lt}(g')$ is finite for all $g' \in G$ by Assumption 5.1 and Lemma 2.8. Further, $|\mathrm{Idx}'| \leqslant |N|^{\max_{\mathbb{N}}\{|\mathsf{lt}(g')| \mid g' \in N\}}$ is finite and computable. Note that $\mathrm{Min}\{e \mid f'(g') \leqslant u(e)\}$ is computable for a fixed $f' : N \to \mathcal{E}$ and $g' \in N$. Calculating $\mathsf{Iteration}'^{i+1}(\mathbf{0})(g_d)$ therefore entails calculating $\sup\{f(g') \mid g' \in N\}$ for finitely many f and subsequently applying the minimum. This yields the intended result. □

A direct consequence of Lemma 6.2, Lemma 6.3 and Lemma 6.4 is the following theorem.

Theorem 9 (Decidability)**.** *Let $\mathcal{G}$ be a monotonic energy game. If $\mathrm{Min}\{e \mid e' \leqslant u(e)\}$ is computable for all updates u in $\mathcal{G}$ and all $e' \in \mathcal{E}$, then $\mathcal{G}$ is decidable.*

We now briefly compare Theorem 9 to our previous decidability result, Theorem 5. Notably, Theorem 9 and Corollary 6.1, which is equivalent to Theorem 5, only differ in two details. Firstly, we once use min and once Min, secondly we explicitly assume the minimum to exist only in Corollary 6.1. While computability implies existence, sets of the form $\mathrm{Min}\{e \mid e' \leqslant u(e)\}$ always exist by Assumption 5.1. We now define a new class of energy games such that Theorem 9 is applicable to that class.

Definition 6.2 (Generalised Galois energy game)**.** *A generalised Galois energy game is a monotonic energy game $\mathcal{G}$ over $\mathcal{E}$ where $\mathrm{Min}\{e \mid e' \leqslant u(e)\}$ is computable for all updates u in $\mathcal{G}$ and all $e' \in \mathcal{E}$.*

Note that we generalised the subclass of Galois energy games to which Theorem 5 is applicable rather than generalising all Galois energy games. In particular, we generalised the following class.

Definition 6.3 (Computable Galois energy game)**.** *A computable Galois energy game is a Galois energy game $(\mathcal{G}, \mathrm{inv})$ such that $\mathrm{inv}(u)$ is computable for all updates u in $\mathcal{G}$.*

Remark 6.1. Let $\mathcal{G}$ be a monotonic energy games. Then $\mathcal{G}$ is a computable Galois energy games if and only if $\min\{e \mid e' \leqslant u(e)\}$ exists and is computable for all updates u in $\mathcal{G}$ and all $e' \in \mathcal{E}$. If $m := \min\{e \mid e' \leqslant u(e)\}$ exists for some update u in $\mathcal{G}$ and $e' \in \mathcal{E}$, we have $\mathrm{Min}\{e \mid e' \leqslant u(e)\} = \{m\}$. Thus, all computable Galois energy games are generalised Galois energy games.

Now we combine Theorem 9 with Lemma 4.1 and Lemma 4.2 to directly conclude the following.

Corollary 6.2. *Let $\mathcal{G}$ be a generalised Galois energy game over Bisping's energies. Then $\mathcal{G}$ is decidable.*

Using Corollary 6.2 we now show that we indeed generalised Theorem 5. In particular, energy games with updates described in Example 6.2 are decidable.

Example 6.3. Consider the 2-dimensional energy game $\mathcal{G} = (\{g_0, g_1\}, \varnothing, \rightarrowtail)$ over Bisping's energies where $\rightarrowtail$ is given by $g_0 \overset{u}{\rightarrowtail} g_1$. If u is one of the three examples introduced in Example 6.2, then $\mathcal{G}$ is not a Galois energy game. However, $\mathrm{Min}\{e \mid e' \leqslant u(e)\}$ is computable for all $e' \in \mathbb{N}_\infty^2$ and thereby $\mathcal{G}$ is a generalised Galois energy game. We show this by providing calculations.

1. Note that for all $(e'_0, e'_1) \in \mathbb{N}^2_\infty$ the following holds.

$$\begin{aligned}&\mathrm{Min}\{(e_0, e_1) \mid (e'_0, e'_1) \leqslant (\max_{\mathbb{N}}\{e_0, e_1\}, \max_{\mathbb{N}}\{e_0, e_1\})\}\\ &= \{(\max_{\mathbb{N}}\{e'_0, e'_1\}, 0), (0, \max_{\mathbb{N}}\{e'_0, e'_1\})\}\end{aligned}$$

2. Note that for all $(e'_0, e'_1) \in \mathbb{N}^2_\infty$ the following holds.

$$\begin{aligned}&\mathrm{Min}\{(e_0, e_1) \mid (e'_0, e'_1) \leqslant (e_0 + e_1, e_0 + e_1)\}\\ &= \begin{cases}\{(e_0, e_1) \mid e_0 + e_1 = \max_{\mathbb{N}}\{e'_0, e'_1\}\}, & \text{if } e'_0 \neq \infty \neq e'_1\\ \{(0, \infty), (\infty, 0)\}, & \text{if } e'_0 = \infty \ \vee \ e'_1 = \infty.\end{cases}\end{aligned}$$

3. Note that for all $(e'_0, e'_1) \in \mathbb{N}^2_\infty$ the following holds.

$$\mathrm{Min}\{(e_0, e_1) \mid (e'_0, e'_1) \leqslant (0, 0)\} = \begin{cases}\{(0, 0)\} & \text{, if } (e'_0, e'_1) = (0, 0)\\ \varnothing & \text{, else.}\end{cases}$$

The close resemblance of the definition of generalised Galois energy games and computable Galois energy games as discussed in Remark 4.5 suggest that we can understand generalised Galois energy games using Galois connections. We now explicitly state the Galois connections hidden in Definition 6.2.

Remark 6.2. For updates u in $\mathcal{G}$ we write $u^{\min}$ for the composition of Min and the image of u, i.e. $u^{\min} : \mathcal{P}(\mathrm{dom}(u)) \to \mathcal{P}(\mathcal{E}), A \mapsto \mathrm{Min}\{u(a) \mid a \in A\}$. We now construct a partial order $\succeq$ on sets of incomparable energies such $\mathcal{G}$ is a generalised Galois energy game if and only if the following conditions hold for all updates u in $\mathcal{G}$:

1. $\mathrm{dom}(u)$ is upward-closed.
2. $\big((\mathcal{P}_{\mathsf{pareto}}(\mathcal{E}), \succeq), \mathrm{inv}(u^{\min}), u^{\min}, (\mathcal{P}_{\mathsf{pareto}}(\mathrm{dom}(u)), \succeq)\big)$ is a Galois connection.
3. $\mathrm{inv}(u^{\min})(\{e'\})$ is computable for all $e' \in \mathcal{E}$.

We want to relate sets of energies A and B if all elements in B have a lower bound in A. Such a relation is similar to the dual of the order on possible Pareto fronts. For this reason, we use $\succeq$. For $A, B \in \mathcal{P}_{\mathsf{pareto}}(\mathcal{E})$ we set $A \succeq B$ if and only if $B \subseteq \uparrow A$. Then, the arguments provided in the proof of Lemma 5.1 and Lemma 5.2 can be easily adjusted to show that $(\mathcal{P}_{\mathsf{pareto}}(\mathcal{E}), \succeq)$ forms a complete lattice with the infimum $\inf_{\succeq} : \mathcal{P}(\mathcal{P}_{\mathsf{pareto}}(\mathcal{E})) \to \mathcal{P}_{\mathsf{pareto}}(\mathcal{E}),\ E \mapsto \mathrm{Min} \bigcup\{A \mid A \in E\}$.[39]
This order preserves monotonicity of updates in $\mathcal{G}$, i.e. an update u in $\mathcal{G}$ is monotonic w.r.t. $\leqslant$ if and only if $u^{\min}$ is monotonic w.r.t. $\succeq$. Further, $\min_{\succeq}\{A \mid B \succeq u^{\min}(A)\}$ exists for all updates u in $\mathcal{G}$ and all antichains $B \subseteq \mathcal{E}$.[40]

[39] Note that due to the similarities of $\succeq$ and the dual of the order on possible Pareto fronts, the roles of supremum and infimum reverse.

[40] Note that the infimum always exists and $u^{\min}\left(\inf_{\succeq}\{A \in \mathcal{P}(\mathrm{dom}(u)) \mid B \succeq u^{\min}(A)\}\right) \subseteq \uparrow B$ holds for all $B \subseteq \mathcal{E}$.

Note that the following equalities hold for all updates u in $\mathcal{G}$ and all $e' \in \mathcal{E}$.

$$\begin{aligned}
\mathrm{Min}\{e \mid e' \leqslant u(e)\} &= \mathrm{Min}\{e \mid u(e) \in \uparrow \{e'\}\} \\
&= \mathrm{Min} \bigcup \{A \in \mathcal{P}(\mathrm{dom}(u)) \mid \forall a \in A.\ u(a) \in \uparrow \{e'\}\} \\
&= \mathrm{Min} \bigcup \{A \in \mathcal{P}_{\mathsf{pareto}}(\mathrm{dom}(u)) \mid \forall a \in A.\ u(a) \in \uparrow \{e'\}\} \\
&= \mathrm{Min} \bigcup \{A \mid u^{\min}(A) \subseteq \uparrow \{e'\}\} \\
&= \mathrm{Min} \bigcup \{A \mid \{e'\} \geq u^{\min}(A)\} \\
&= \inf_{\geq} \{A \mid \{e'\} \geq u^{\min}(A)\} \\
&= \min_{\geq} \{A \mid \{e'\} \geq u^{\min}(A)\}
\end{aligned}$$

We set $\mathrm{inv}(u^{\min}) : \mathcal{P}_{\mathsf{pareto}}(\mathcal{E}) \to \mathcal{P}_{\mathsf{pareto}}(\mathrm{dom}(U)),\ B \mapsto \min_{\geq}\{A \mid B \geq u^{\min}(A)\}$ for updates u in $\mathcal{G}$. Then, $\mathrm{inv}(u^{\min})(\{e'\}) = \mathrm{Min}\{e \mid e' \leqslant u(e)\}$. Further, $\mathrm{inv}(u^{\min})$ and $u^{\min}$ form a Galois connection between $(\mathcal{P}_{\mathsf{pareto}}(\mathcal{E}), \geq)$ and $(\mathcal{P}_{\mathsf{pareto}}(\mathrm{dom}(u)), \geq)$, if and only if u is monotonic by Lemma 4.5. This implies the stated equivalence.

With this remark we consider our name choice *generalised Galois energy games* justified.

6.1.3 Energy Game Classes

We now give an overview of the classes of energy games we defined throughout this thesis. These classes only differ in the possibly occurring updates. All classes are subclasses of monotonic energy games as defined in Definition 3.8, i.e. energy games where all updates are monotonic with an upward-closed domain. When defining (generalised) Galois energy games we further assume some minima to exist (and be computable). A summary of the defining properties of updates is given in Table 6.1. Note that we only specify properties additional to the games being monotonic.

	For all updates u and energies e'	
	... exists.	**... is computable.**
Monotonic energy games	$\mathrm{Min}\{e \mid e' \leqslant u(e)\}$	
Generalised Galois energy games	$\mathrm{Min}\{e \mid e' \leqslant u(e)\}$	$\mathrm{Min}\{e \mid e' \leqslant u(e)\}$
Galois energy games	$\min\{e \mid e' \leqslant u(e)\}$	
Computable Galois energy games	$\min\{e \mid e' \leqslant u(e)\}$	$\min\{e \mid e' \leqslant u(e)\}$

Table 6.1: Definiting properties of energy game classes

The set $\mathrm{Min}\{e \mid e' \leqslant u(e)\}$ exists for all updates u and all energies e'. Thereby, this property holds but is not defining as indicated by the light gray in Table 6.1. The class hierarchies are visualised in Figure 6.1, where a subclass is below its superclass.

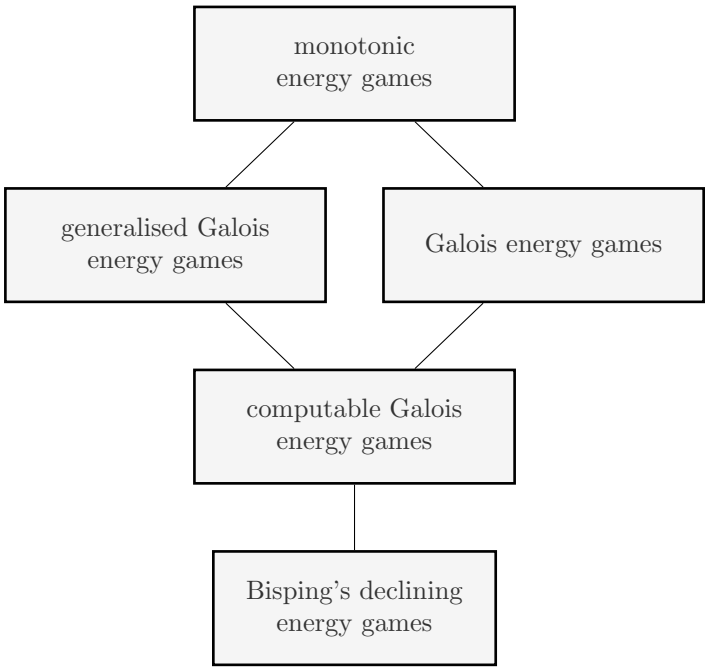

Figure 6.1: Class diagram

By Lemma 4.8 and Lemma 4.4 Bisping's declining energy games are a true subclass of computable Galois energy games. With Remark 6.1 the remaining subclass-superclass-relations visualised in Figure 6.1 are directly implied by the defining properties summarised in Table 6.1.
We now recapitulate previous observations implying that Galois energy games and generalised Galois energy games do not form a subclass-superclass-relation. By Example 6.1 there exists a Galois energy game such that $\min\{e \mid e' \leqslant u(e)\}$ exists for all energies e' and all updates u but is not always computable. This energy game is not a generalised Galois energy game since $\mathrm{Min}\{e \mid e' \leqslant u(e)\} = \{\min\{e \mid e' \leqslant u(e)\}\}$ is not always computable. By Example 6.3 there exist generalised Galois energy games that are not Galois energy games.
Further, note that these examples imply that monotonic energy games are a true superclass of (generalised) Galois energy games. In particular, all five classes in Figure 6.1 are distinct, since Example 6.1 (resp. Example 6.3) implies that computable Galois energy games are a true subclass of (generalised) Galois energy games.

6.2 Related Games

In this section we discuss related definitions of (energy) games within research. We start by discussing the assumption we made about energies and about the game graph to achieve decidability. Afterwards, we focus on related games defined over vectors of (extended) naturals with updates in $(+\mathbb{Z})^n$.[41] There are

[41] For this notation we refer to Definition 4.6.

various different winning conditions for energy games. We compare winning conditions to those fixed in Definition 3.5. First, we consider multi-weighted reachability games and abstractions of those. Afterwards, we discuss the symmetric case to Definition 3.5 where the attacker wins all infinite plays if the energy level does not become undefined. Finally, we end this section by mentioning mean-payoff and parity games.

6.2.1 Assumptions about Energies and the Game Graph

Theorem 5 and Theorem 9 only apply to energy games over well-founded bounded join-semilattices such that the supremum and the minimal elements of a finite set are computable with finite sets of positions by Assumption 5.1, Assumption 5.2 and Assumption 5.3. In this section we study these properties and their relevance for our decidability results. Since some reasonable computability assumptions are necessary for decidability, we do not further discuss Assumption 5.3 and focus on the other properties instead.

Bounded Join-Semilattices

We now consider part of Assumption 5.2, specifically the energies forming a bounded join-semilattice. In both, Algorithm 1 and Algorithm 3, we calculate $\mathsf{new\_a\_win}[g]$ for defender positions g by calculating suprema of the form $\sup\{e_{g'} \mid g \rightarrowtail^{u} g'\}$. If g is not a deadend, then the finite branching degree ensures the existence of such suprema in join-semilattices. This assumption is indispensable to be able to apply our algorithms.
The energies forming a bounded join-semilattice further ensures the existence of a supremum of the empty set, which is calculated if g is a deadend. Specifically, $\mathsf{new\_a\_win}[g]$ is then assigned the set only containing the minimum in $\mathcal{E}$. While join-semilattices being well-founded does not imply the existence of a minimum[42], it does imply the existence of a finite set of minimal elements. When assigning all minimal elements to defender positions that are deadends, our line of argumentation used in the proofs of Theorem 5 and Theorem 9 remains applicable. Thus, we can prove decidability over well-founded unbounded join-semilattices.

Well-founded Energies

Bisping's energies as well as $\mathbb{N}^n$ for some $n \in \mathbb{N}$ with the component-wise order are commonly used to represent energies [7, 12]. Note that both are well-founded bounded join-semilattices. Other frequently used sets of energies are vectors of non-negative (extended) reals with the component-wise order [19]. While these energies are bounded join-semilattices, they are not well-founded. Thereby, our decidability results, Theorem 5 and Theorem 9, do not directly apply to energy games over these energies.
However, in declining monotonic energy games over any partially ordered set of energies avoiding cycles can only positively impact the energy sufficient for the attacker to win. We can use arguments similar to those seen in Lemma 5.9 and Lemma 5.10 and apply them to declining Galois energy games

[42]For example Bisping's energies without the zero-vector form a well-founded join-semilattice that is unbounded for dimensions $\geqslant 2$.

over energies that are not well-founded. Then, we can conclude that the set $\{u_1^{-1} \circ ... \circ u_i^{-1}(m) \mid i \in \{0, ..., |G| - 1\} \ \wedge \ u_1, ..., u_i$ are updates in $\mathcal{G}\}$ as seen in Definition 5.7 with m being the minimum in $\mathcal{E}$ contains all candidates of minimal energies in some attacker winning budget. In a finite game graph the number of updates is finite and thereby so is this set. This ensures that minimal attacker winning budgets exist and are finite without Assumption 5.1. Thereby, we obtain termination of Algorithm 1 and can conclude decidability for computable Galois energy games over energies that are not well-founded.
When applying this reasoning to generealised Galois energy games, we have to assume the sets $\text{Min}\{e \mid e' \leqslant u(e)\}$ to be finite (and computable) for all updates u in $\mathcal{G}$ and all $e' \in \mathcal{E}$. Then, we can conclude decidability without Assumption 5.1, if the game is declining.

Properties of the Game Graph

We now consider the properties of the game graph needed for our decidability results. In Assumption 5.2 we assume the set of positions to be finite. This is essential for our proof of decidability. Finite positions imply a finite branching degree, which ensures the existence of the suprema calculated in Line 9 of Algorithm 1 and Algorithm 3. Further, the finite branching degree was used in the proofs of correctness, while the finite set of positions ensured termination and thus decidability. It is standard to assume a finite game graph in decidability considerations [12, 15, 19]. However, we want to note that infinite games do arise in some applications and there exist some positive decidability results [11]. In particular, when studying behavioural equivalences of processes on infinite labelled transition systems, Bisping [7] considers energy games with infinite branching degree.
Other common assumptions about the game graph are that there are no deadends [15] and that the two players alternate turns [2]. Assuming no deadends implies by Definition 3.5 that the attacker cannot win the game. Therefore, this assumption negatively impacts the significance of our decidability results. However, when discussing related games, we consider other winning conditions where this assumption can be made. For example for parity games as introduced in Definition 2.11 one can always construct a parity game with no deadends won by a player if and only if they previously won [17]. Assuming the players' turns to be alternating yields an inter-reducible class of energy games. For example one can replace an edge $g_a \rightarrowtail^u g'_a$ between two attacker positions by adding a new defender position g_d such that $g_a \rightarrowtail^u g_d \rightarrowtail^{id} g'_a$ where id is the identity function on $\mathcal{E}$.

6.2.2 Reachablility Games

We now introduce multi-weighted reachability games as defined by Brihaye and Goeminne [12]. We argue that those games w.r.t. the component-wise order form a subclass of Galois energy games and compare time complexity. Afterwards, we briefly discuss vector addition systems with states and monotonic games as defined by Abdulla et al. [2].
We start by introducing the games considered by Brihaye and Goeminne [12] and by translating their notion of multi-weighted reachability games to energy games as defined in Chapter 3. Their games are also played on finite directed

labelled graphs. However, they consider different winning conditions where the attacker[43] wins a play if that play reaches some position in a fixed target set of positions F. In particular, they consider games over $\mathbb{N}_\infty^n$ where all updates add some vector in $\mathbb{N}^n$. They then ask for the minimal energies sufficient for the attacker to enforce a play ending in the target set. These games can be modelled by energy games as previously defined in the following manner.
Let (V, E) be the underlying directed graph of a multi-weighted reachability game as defined by Brihaye and Goeminne. Then there exist subsets $V_a \subseteq V$ representing attacker positions and $F \subseteq V$ representing the targets of the attacker. We now construct an energy game such that the targets correspond to positions where the defender is stuck. For this we consider the graph (V', E') with vertex set $V' := V \uplus F_a$ where F_a represents copies of $V_a \cap F$ and a set of edges E' obtained from E by removing all outgoing edges of positions in $F \backslash V_a$ and adding edges connecting positions in $V_a \cap F$ with their copies in F_a. We label the new edges with the identity function and label old edges with an update subtracting the energy that was previously added. Thereby, we obtain an energy game $\mathcal{R} := (V', V_a, \rightarrowtail)$ such that $\mathcal{R}[g, e]$ is won by the attacker if and only if the attacker can enforce a play with energy level $\leqslant e$ ending in a target position in the original game when starting at position g.
Brihaye and Goeminne provide an algorithm that can be interpreted as calculating minimal attacker winning budgets for such games. At the same time their algorithm calculates an energy-positional attacker winning strategy. We could modify our algorithms accordingly resulting in the output of an attacker winning strategy with a memory proportional to $|G_a| \cdot w_{wu_\mathcal{G}}$. Such a modification is possible for all energy games that our decidability results apply to and does not negatively impact the running time or the space needed for the output in big O notation.
We now compare running times. Brihaye and Goeminne consider two orders on $\mathbb{N}_\infty^n$, the lexicographical order and the component-wise order. We focus on the latter and thereby only consider energy games over Bisping's n-dimensional energies. For an n-dimensional energy game $\mathcal{R}$ that was constructed from such a multi-weighted reachability game as described above let $w \in \mathbb{N}$ be the largest number subtracted in any component by the updates in $\mathcal{R}$. Then their algorithm has a complexity in $\mathcal{O}\left(n^4 \cdot w^{4 \cdot n} \cdot |V|^{4 \cdot n + 3}\right)$.[44] Using approximations similar to those seen in the proof of Lemma 4.4 we have $wu_\mathcal{R} \leqslant (w \cdot |V'|, ..., w \cdot |V'|)$ and thereby $w_{wu_\mathcal{R}} \in \mathcal{O}\left(w^{n-1} \cdot |V'|^{n-1}\right)$. Over-approximating the branching degree by $|V'| \in \mathcal{O}(|V|)$ Theorem 7 implies a running time in $\mathcal{O}\left(n^2 \cdot w^{3(n-1)} \cdot |V|^{3 \cdot n}\right)$ since $\mathcal{R}$ is declining. While this is a slight improvement compared to the bound stated by Brihaye and Goeminne, we again note that we provide finer approximations instead of improving actual running times.

Abstracting Reachability Games

We now consider abstractions of these reachability games. A related concept, where reachability conditions can be defined, is that of vector addition systems with states (VASS) [11]. VASS games are defined over directed graphs labelled with updates in $(+\mathbb{Z})^n$. In VASS games energy levels cannot become negative in any component. Thereby, VASS games are similar to energy games over $\mathbb{N}^n$

[43] Note that Brihaye and Goeminne call the attacker *Player 1*.
[44] This result is stated as Theorem 4 by Brihaye and Goeminne [12].

with an important difference: Transitions resulting in the energy level becoming undefined and the defender winning in an energy game are disabled in VASS games. While VASS games are undecidable, if both players are able to update the energy, asymmetric VASS games, i.e. where this ability is restricted for one player, are decidable [16]. Note that VASS games with reachability winning conditions[45] abstract multi-weighted reachability games as defined by Brihaye and Goeminne.
VASS games can be further abstracted by characterising them through configurations, i.e. pairs of positions and energies, and the transitions between them[2, 3].[46] In this way different versions of games can be unified as illustrated by Abdulla et al. [2] considering monotonic games and proving undecidability of the safety problem. The safety problem asks whether a configuration is winning for a player trying to avoid a set of final configuration while the other player tries to enforce such a configuration. This can be understood as reachability winning conditions. In particular, the final configurations in games with safety winning conditions closely resemble the target sets in multi-weighted reachability games but with the symmetric case where the attacker wins all infinite plays. Abdulla et al. show decidability for a subclass of monotonic games, namely downward-closed games. Interestingly, they consider well quasi-orderings on configurations which can be induced by a well-founded partial order on energies. Investigating the relation of those downward-closed monotonic games and monotonic energy games over well-founded energies could be promising.

6.2.3 Other Winning Conditions

We now briefly discuss other winning conditions and how they relate to the energy games studied in this thesis.

The Symmetric Case

We now consider generalised energy games as defined by Chatterjee et al. [15]. They consider games over $\mathbb{N}^n$ with the component-wise order played on finite directed graphs labelled with updates in $(+\mathbb{Z})^n$. Those games can be considered energy games as defined in Chapter 3 with slightly different winning conditions. In particular, they assume the attacker[47] to win all infinite plays.
When changing the winner of infinite plays, we are unable to directly apply our results. This starts in Section 3.2.1 where the given construction of a parity game no longer serves to prove energy-positional determinacy. However, a similar construction to that in Definition 3.9 assigning an uneven priority to all positions yields the same result, i.e. energy games with the changed winning condition being ernergy-positionally determined. This can be used to prove an inductive characterisation of defender winning budgets similar to Theorem 3.
Using symmetry most of our arguments can be adjusted to the changed winning conditions. Since multiple researchers use the discussed symmetric winning conditions, formally proving this claim might be worthwhile [3, 15].

[45] Note that Courtois and Schmitz [16] differentiate between reachability and state reachability problems. Here we refer to the latter.

[46] The same idea was used in the construction of a parity game from an energy game as introduced in Definition 3.9.

[47] Note that Chatterjee et al. call the attacker *Player 1*.

Mean-Payoff Games

Chatterjee et al. [15] also consider multi-dimensional games with another winning condition: Mean-Payoff Games. There, when given a vector of integers v, a play is won by the attacker if the mean of all energy levels of prefixes of that play is at least v. Since infinite memory may be necessary to win such a game[48], it is unlikely that our reasoning can be transferred to instances with those winning conditions.

Parity Energy Games

We introduced parity games in Definition 2.11. Parity winning conditions can also be defined on energy games. For example Abdulla et. al [4] define games on integer vectors, i.e. games played on directed graphs labelled with updates in $(+\mathbb{Z})^n$, with parity winning conditions and note that the unknown initial credit problem for such games is coNP-complete. Bernet et al. [6] first proposed transforming parity conditions to safety conditions. Then, Chatterjee et al. [14] used this idea by transforming the parity condition into additional dimensions. Using an upper bound for the elements of winning budgets they argue decidability by giving an incremental algorithm to compute minimal winning budgets. The core idea resembles that of Algorithm 3 and expanding this research could yield valuable insights into parity Galois energy games.

[48] This is stated as Lemma 7 Chatterjee et al. [15].

Chapter 7

Conclusion

Our goal with this thesis was to understand decidability of multi-weighted (declining) energy games by studying Bisping's declining energy games and providing a formal proof using Isabelle/HOL. We were able to identify crucial properties of energy games in order to decide the games using Bisping's algorithm. We proved decidability for a larger class of energy games and substantiated the decidability claimed by Bisping with a machine-checked proof. We now summarise our approach and main results as well as their implications and limitations.
With our Isabelle/HOL theory Energy_Game.thy we provided a formalisation of energy games with reachability winning conditions as defined in Definition 3.5 and used by Bisping [7] as well as implicitly used by Brihaye and Goeminne [12]. By its generality this could serve as a foundation for further formalisations of multi-weighted energy games with reachability winning conditions and their properties. Combining this with Energy_Order.thy we laid the basis for formalisations of energy games over $\mathbb{N}_\infty^n$ (and thereby also for energy games over $\mathbb{N}^n$) with the component-wise order in Isabelle/HOL. However, when building upon our formalisation one might want to add a proof of Theorem 2 which implies Assumption 3.3. Such a proof could be achieved using the proof structure seen in Section 3.2.1 and utilising the existing formalisation of positional determinancy of parity games by Dittmann [17].
We used these theories to formalise Bisping's updates and then Bisping's declining energy games in Isabelle/HOL. By simplifying Bisping's algorithm we were able to give a proof of decidability while simultaneously proving correctness and termination of the original algorithm. After showing that Bisping's declining energy games are (computable) Galois energy games we were able to formalise our proof of the decidability stated in Theorem 5 for the special case of Bisping's declining energy games in Isabelle/HOL. Specifically, we showed that (a simplification of) Bisping's algorithm indeed calculates minimal attacker winning budgets which can be understood as Pareto fronts of the game and thereby deciding both the fixed and the unknown initial credit problem.
By introducing Galois energy games we abstracted the properties needed for Bisping's algorithm to be applicable. In particular, all updates have be monotonic, have an upward-closed domain and satisfy a weakened form of invertibility characterised by Galois connections. Studying this class of energy games we showed undecidability of Galois energy games and thereby of the superclass of monotonic energy games. Under some reasonable computability assumptions

we observed decidability of computable Galois energy games.
Commonly, energy games are played on graphs labelled by elements in $\mathbb{Z}^n$ representing vector addition and thereby invertible updates. Such energy games are computable Galois energy games when considering reachability winning conditions. In Bisping's declining energy games minima of components may also appear in updates. While such updates are not invertible, there always exists a unique minimum $\min\{e \mid e' \leqslant u(e)\}$ for all updates u and energies e'. With computable Galois energy games we considered monotonic energy games where $\min\{e \mid e' \leqslant u(e)\}$ is computable for all updates u and energies e'.
Abstracting this computability assumption we were able to further generalise our results. If the set $\{e \mid e' \leqslant u(e)\}$ does not have a unique minimum, the set $\mathrm{Min}\{e \mid e' \leqslant u(e)\}$ exists regardless. With generalised Galois energy games we considered monotonic energy games where those sets are computable for all updates u and energies e'. By generalising Bisping's algorihm we showed decidability of this class. In particular, we allow updates taking maxima of components in decidable energy games.
We now summarise the introduced classes of energy games with Figure 7.1 where a subclass is below its superclass.

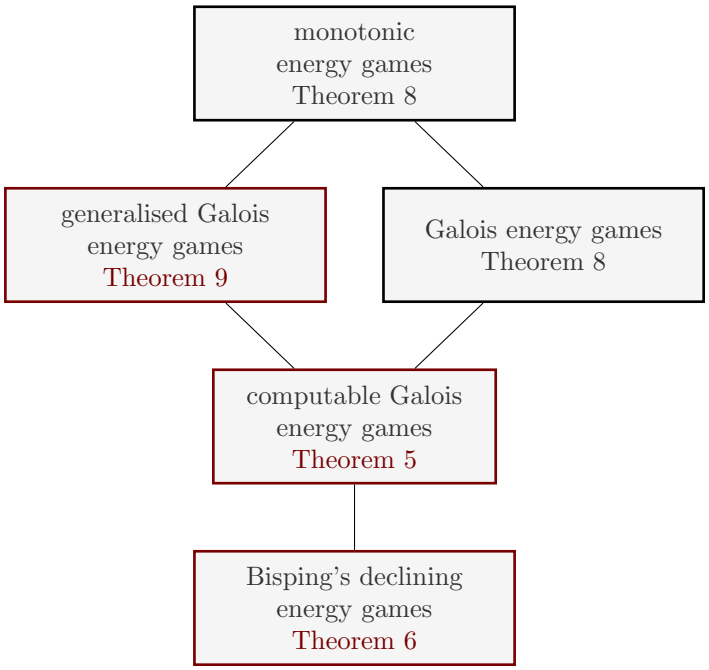

Figure 7.1: Class diagram with decidability results

In Figure 7.1 red indicates decidability for energy games in the respective class over well-founded (bounded) join-semilattices with finite sets of positions and reasonable computability assumptions. While we only formalised Theorem 6 in Isabelle/HOL, we gave full proofs for all referenced theorems. When considering the properties of the energies necessary for our results to hold, we outlined that bounded join-semilattices suffice if the energy game is declining. The decidability claimed by Bisping [7] is stated as Theorem 6 and is a consequence of

Theorem 5. The latter further implies the decidability results discussed by Brihaye and Goeminne [12]. Both theorems are consequences of Theorem 9. With Theorem 8 we showed undecidability of Galois energy games without any computability assumptions. This also implies undecidability of monotonic energy games.

We analysed the complexity of the simplified version of Bisping's algorithm and expressed it w.r.t. the size of the graph $|G|$ and the worst combination of up to $|G| - 1$ updates. For declining energy games we argued that the running time is polynomial w.r.t. the size of the graph and an upper bound on the cardinality of potential attacker winning budgets. For Bisping's declining energy games with a fixed dimension this results in a running time polynomial in the size of the graph. Using finer approximations we were able to provide better running times compared to those stated by Bisping [7] or by Brihaye and Goeminne [12].

While we were able to find relaxed conditions for updates such that decidability remains, we only considered a limited set of winning conditions. By transferring our arguments to the symmetric case, where the attacker wins all infinite plays, our results imply the decidability stated for generalised energy games by Chatterjee et al. [15]. By focusing on well-founded join-semilattices as energies we further abstracted games on integer vectors introduced by Abdulla et al. [4]. The question whether our line of argumentation can be adapted for parity or other winning conditions requires additional studies.

Bibliography

[1] M. Abadi, L. Lamport, and P. Wolper. "Realizable and unrealizable specifications of reactive systems". In: *Automata, Languages and Programming*. Vol. 372. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1989, pp. 1–17. DOI: 10.1007/BFb0035748.

[2] P. A. Abdulla, A. Bouajjani, and J. D'orso. "Monotonic and downward closed games". In: *Journal of Logic and Computation* 18.1 (2008), pp. 153–169. DOI: 10.1093/logcom/exm062.

[3] P. A. Abdulla et al. "Infinite-state energy games". In: *Computer Science Logic (CSL) and Logic in Computer Science (LICS)*. Association for Computing Machinery, 2014, pp. 1–10. DOI: 10.1145/2603088.2603100.

[4] P. A. Abdulla et al. "Solving parity games on integer vectors". In: *Concurrency Theory (CONCUR)*. Vol. 8052. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2013, pp. 106–120. DOI: 10.1007/978-3-642-40184-8_9.

[5] A. Armstrong and S. Foster. *Galois connections*. URL: https://isabelle.in.tum.de/library/HOL/HOL-Algebra/Galois_Connection.html (visited on 2024/10/25).

[6] J. Bernet, D. Janin, and I. Walukiewicz. "Permissive strategies: From parity games to safety games". In: *RAIRO – Theoretical Informatics and Applications* 36.3 (2002), pp. 261–275. DOI: 10.1051/ita:2002013.

[7] B. Bisping. "Process equivalence problems as energy games". In: *Computer Aided Verification (CAV)*. Vol. 13964. Lecture Notes in Computer Science. Springer Nature Switzerland, 2023, pp. 85–106. DOI: 10.1007/978-3-031-37706-8_5.

[8] B. Bisping and D. N. Jansen. "One energy game for the spectrum between branching bisimilarity and weak trace semantics". In: *Expressiveness in Concurrency (EXPRESS) and Structural Operational Semantics (SOS)*. Vol. 412. Electronic Proceedings in Theoretical Computer Science. Open Publishing Association, 2024, pp. 71–88. DOI: 10.4204/EPTCS.412.6.

[9] B. Bisping, D. N. Jansen, and U. Nestmann. "Deciding all behavioral equivalences at once: A game for linear-time-branching-time spectroscopy". In: *Logical Methods in Computer Science* 18.3 (2022). DOI: 10.46298/lmcs-18(3:19)2022.

[10] B. Bisping and U. Nestmann. “A game for linear-time–branching-time spectroscopy”. In: *Tools and Algorithms for the Construction and Analysis of Systems (TACAS)*. Vol. 12651. Lecture Notes in Computer Science. Springer International Publishing, 2021, pp. 3–19. DOI: 10.1007/978-3-030-72016-2_1.

[11] T. Brázdil, P. Jančar, and A. Kučera. “Reachability games on extended vector addition systems with states”. In: *Automata, Languages and Programming (ICALP)*. Vol. 6199. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2010, pp. 478–489. DOI: 10.1007/978-3-642-14162-1_40.

[12] T. Brihaye and A. Goeminne. “Multi-weighted reachability games”. In: *Reachability Problems*. Vol. 14235. Lecture Notes in Computer Science. Springer Nature Switzerland, 2023, pp. 85–97. DOI: 10.1007/978-3-031-45286-4_7.

[13] A. Chakrabarti et al. “Resource interfaces”. In: *Embedded Software*. Vol. 2855. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2003, pp. 117–133. DOI: 10.1007/978-3-540-45212-6_9.

[14] K. Chatterjee, M. Randour, and J.-F. Raskin. “Strategy synthesis for multi-dimensional quantitative objectives”. In: *Concurrency Theory (CONCUR)*. Vol. 7454. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2012, pp. 115–131. DOI: 10.1007/978-3-642-32940-1_10.

[15] K. Chatterjee et al. “Generalized mean-payoff and energy games”. In: *Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*. Vol. 8. Leibniz International Proceedings in Informatics (LIPIcs). Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2010, pp. 505–516. DOI: 10.4230/LIPIcs.FSTTCS.2010.505.

[16] J.-B. Courtois and S. Schmitz. “Alternating vector addition systems with states”. In: *Mathematical Foundations of Computer Science (MFCS)*. Vol. 8634. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2014, pp. 220–231. DOI: 10.1007/978-3-662-44522-8_19.

[17] C. Dittmann. “Positional Determinacy of Parity Games”. In: *Archive of Formal Proofs* (2015). https://isa-afp.org/entries/Parity_Game.html, Formal proof development. ISSN: 2150-914x.

[18] M. Erné et al. “A Primer on Galois Connections”. In: *Annals of the New York Academy of Sciences* 704.1 (1993), pp. 103–125. DOI: 10.1111/j.1749-6632.1993.tb52513.x.

[19] Zoltán Ésik et al. “An algebraic approach to energy problems II – the algebra of energy functions”. In: *Acta Cybernetica* 23.1 (2017), pp. 229–268. DOI: 10.14232/actacyb.23.1.2017.14.

[20] R. Fraisse and P. Clote. *Theory of Relations*. Elsevier, 1986.

[21] R. J. van Glabbeek. “The linear time - branching time spectrum”. In: *Theories of Concurrency: Unification and Extension (CONCUR)*. Ed. by J. C. M. Baeten and J. W. Klop. Vol. 458. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 1990, pp. 278–297. DOI: 10.1007/BFb0039066.

[22] T. Göthel. "Mechanical verification of parameterized real-time systems". PhD thesis. Technische Universität Berlin, 2012. DOI: 10.14279/depositonce-3248.

[23] G. Gratzer. *Lattice theory: Foundation*. Springer, 2011. DOI: 10.1007/978-3-0348-0018-1.

[24] D. H. J. de Jongh and R. Parikh. "Well-partial orderings and hierarchies". In: *Indagationes Mathematicae*. Vol. 80. 3. Elsevier, 1977, pp. 195–207. DOI: 10.1016/1385-7258(77)90067-1.

[25] K. Kappelmann. "Transport via partial galois connections and Equivalences". In: *Archive of Formal Proofs* (2023). https://isa-afp.org/entries/Transport.html, Formal proof development. ISSN: 2150-914x.

[26] S. Kreutzer and R. Rabinovich. "Logic, Games, Automata". Lecture notes. Technische Universität Berlin. 2020.

[27] J. D. Mashburn. "The least fixed point property for omega chain continuous functions". In: 9.2 (1981), pp. 231–244.

[28] T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL – A Proof Assistant for Higher-Order Logic*. Vol. 2283. Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2002. DOI: 10.1007/3-540-45949-9.

[29] P. Smith. *The Galois connection between syntax and semantics*. University of Cambridge. 2010.

[30] C. Stirling. "Bisimulation, modal logic and model checking games". In: *Logic Journal of IGPL* 7.1 (1999), pp. 103–124. DOI: 10.1093/jigpal/7.1.103.

[31] G. Vogel. *Accelerating Process Equivalence Energy Games using WebGPU*. Bachelor's Thesis. Available at https://github.com/Gobbel2000/thesis-gpuequiv/releases/latest. 2024.

[32] A. Yamada and J. Dubut. "Complete non-orders and fixed points". In: *Archive of Formal Proofs* (2019). https://isa-afp.org/entries/Complete_Non_Orders.html, Formal proof development. ISSN: 2150-914x.

[33] W. Zielonka. "Infinite games on finitely coloured graphs with applications to automata on infinite trees". In: *Theoretical Computer Science* 200.1-2 (1998), pp. 135–183. DOI: 10.1016/S0304-3975(98)00009-7.