האוניברסיטה העברית בירושלים
The Hebrew University of Jerusalem

Faculty of Science
Department of Mathematics

Random-Turn and Richman Games

אקראיות ומכרזים כמנגנונים לחלוקת תורות במשחקים קומבינטורים

by
Michael Simkin

under the supervision of
Professor Gil Kalai

Submitted in partial fulfillment of the
requirements for the degree of Master of Science

October 2014
תשרי תשע"ה

לאמא ואבא, באהבה

אקראיות ומכרזים כמנגנונים לחלוקת תורות במשחקים קומבינטורים (תקציר)

24 באוקטובר 2014

משחק קומבינטורי סכום אפס בין שני שחקנים הוא משחק בו שני שחקנים, כחול ואדום, מזיזים לסירוגין אסימון על קשתותיו של גרף מכוון. חלק מקודקודי הגרף מוגדרים כקודקודים סופיים. אם במהלך המשחק האסימון מגיע לאחד מקודקודים אלה, אזי המשחק מסתיים והשחקנים מקבלים תשלום (סכום אפס) שתלוי בקודקוד.
פורמאלית:

הגדרה 0.1 משחק קומבינטורי הוא חמישיה, $G = (V, E_B, E_R, T, \nu)$ כך ש $(V, E_B \cup E_R)$ הוא גרף מכוון סופי, $T \subseteq V$ הוא קבוצת הקודקודים הסופיים, ו$\nu : T \to \mathbb{R}$ היא פונקצית התשלום לשחקן הכחול. E_B נקראת קבוצת הצלעות הכחולות, וE_R היא קבוצת הצלעות האדומות. קבוצות אלה אינן בהכרח זרות, ויש לפרש אותן כקבוצות המהלכים המותרים לכחול ואדום, בהתאמה. אנו דורשים שלכל $v \in T$ לא תהינה צלעות יוצאות, ולכל $v \in V \backslash T$ יהיה מסלול לT הן בE_B והן בE_R.

מהלך של משחק קומבינטורי המתחיל מקודקוד $s \in V$ הוא הילוך על הגרף $(V, E_B \cup E_R)$ אשר מתחיל בs, מסתיים בT או שהוא אינסופי, ואשר הצלעות בו כחולות ואדומות, לסירוגין.

משחקים קומבינטורים מציבים אתגר מעניין בפני מתמטיקאים. ראשית, המודל מכסה דוגמאות רבות של משחקים "אמיתיים", החל משח וכלה באיקס־עיגול. על כן הבנה מתמטית של משחקים אלה עלולה לתת לחוקר יתרון נגד יריביו ־ מטרה ראויה בהחלט. שנית, קיים פער עמוק בין מה שניתן להוכיח מתמטית, ובין מה שניתן לבצע ביעילות: מצד אחד, האופי הסופי של משחקים אלה, יחד עם העובדה שהם מבוססי תורות, מבטיחות קיום של אסטרטגיות טהורות אופטימליות, או אפשרות של משחק אינסופי. מצד שני, באופן כללי אין דרך למצוא אסטרטגיות אופטימליות מבלי לעבור על כל גרף המשחק. מכיוון שחוץ מאשר במקרים פשוטים במיוחד גרף המשחק עצום בגודלו, גישה זו אינה מבטיחה.

בעבודה זו אנו בוחנים שינויים אפשריים במנגנון חלוקת התורות במשחק קומבינטורי. במשחקים עם תורות אקראיים לפני כל תור מטילים מטבע, כאשר הזוכה בהטלה מבצע את המהלך הבא. במשחקי ריצ'מן (על שם המתמטיקאי דוד ריצ'מן) לפני כל תור מתבצע מכרז, כאשר המנצח במכרז משלם את סכום הצעתו לשחקן השני (אם שתי ההצעות שוות מפעילים איזשהו מנגנון שבירת שוויון. לצורך העניין, אפשר להחליט שבמקרה וההצעות שוות כחול יוכרז כמנצח במכרז), ומחליט איזה מהשחקנים יבצע את המהלך הבא. ביחס למשחקים עם תורות אקראיים, אנו סוקרים את עבודתם של יובל פרס, עודד שרם, סקוט שפילד ודוד ווילסון ב[4]. ביחס למשחקי ריצ'מן, אנו סוקרים את עבודתם של אנדרו לזרוס, דניאל לוב, ג'יימס פרופ, וולטר סטרומקויסט, ודניאל אולמן ב[3] ו[2]. בנוסף אנו עומדים על הקשר בין שני סוגי המשחקים. לבסוף אנו דנים בשאלות אלגוריתמיות הקשורות בנושא העבודה.

מירב עבודתינו עוסקת באפיון אסטרטגיות אופטימליות עבור שני סוגי המשחק. אנו מתחילים בסוג מסויים של משחק עם תורות אקראיים: משחקי בחירה.

משחקי בחירה עם תורות אקראיים

במשחק בחירה יש קבוצה סופית S. כל שחקן בתורו בוחר איבר מ-S ומוסיף אותו לאוסף שלו. לאחר שכל איברי S נבחרו, פונקציית התשלום תלויה בחלוקה שנוצרה. דרך שקולה לחשוב על משחקי בחירה היא זו: קובעים פונקציה $f : \{-1,1\}^n \to \mathbb{R}$. בתחילת המשחק ערכי המשתנים $x_1, \ldots, x_n$ אינם ידועים. בכל תור, השחקן בוחר משתנה עם ערך שטרם נקבע וקובע את ערכו (כחול חייב לקבוע שהערך יהיה 1, ואדום -1). לאחר שערכם של כל המשתנים נקבע, התשלום לכחול הוא $f(x_1, \ldots, x_n)$.

נשים לב שמשחק בחירה עם תורות אקראיים תמיד יסתיים לאחר n תורות. בנוסף, מביצוע אינדוקציה הפוכה על מספר האיברים שנבחרו ניתן לחשב הן את ערך המשחק, והן אסטרטגיות אופטימליות. עם זאת, אנו נוקטים בגישה שונה בניתוח משחקים אופטימליים. התכונה שמאפשרת לנו לעשות זאת היא הבאה:

משפט 0.2 יהי משחק בחירה עם תורות אקראיים עם פונקצית התשלום $f : \{-1,1\}^n \to \mathbb{R}$. אזי ערך המשחק הוא $\mathbb{E}_{x\sim\{-1,1\}^n}[f(x)]$, דהיינו תוחלת הפונקציה f כאשר המשתנה שלו מוגרל באופן אחיד מהקבוצה $\{-1,1\}^n$. יע"כ, כל אסטרטגיה שהיא אופטימלית עבור כחול היא אופטימלית גם עבור אדום, במובן זה שאם בשלב כלשהו במשחק בחירת המשתנה x_i היא אופטימלית עבור כחול, אזי בחירת x_i אופטימלית גם בשביל אדום.

למשפט זה כמה השלכות. ראשית, הוא אומר שבעוד שייתכן שמאמצים רבים יושקעו בניסיון למצוא אסטרטגיה אופטימלית, אם שני השחקנים אכן משחקים לפי אסטרטגיה כזו, תוצאת המשחק תיראה כמו משחק אקראי. שנית, הוא מאפשר לאפיין בחירות אופטימליות כדלהלן:

משפט 0.3 בחירת המשתנה x_i הוא אופטימלי כמהלך פתיחה במשחק אסם הוא ממקסם את הערך:

$$\hat{f}(i) = 2\left(\mathbb{E}\left[f(x)\,|x_i = 1\right] - \mathbb{E}\left[f\right]\right)$$

השימוש בנוטציה $\hat{f}(i)$ אינו מקרי: זהו מקדם פורייה המתאים למונום x_i. הקשר לאנליזת פורייה מאפשר שימוש בתוצאות מאותו תחום, ובין השאר ניתן לתת כך חסמים על התוחלת של מספר המהלכים במשחק אופטימלי.[1]

משחקים כלליים עם תורות אקראיים

יהי $G = (V, E_B, E_R, T, \nu)$ משחק קומבינטורי. האבחנה הראשונה שעלינו לשים לב אליה היא שלא כל זוג אסטרטגיות תגרום למשחק כזה להסתיים בהסתברות אחד. עם זאת, אנו מראים שקיים ערך v כך שכל אחד מהשחקנים יכול לגרום למשחק להסתיים בהסתברות 1 ולהבטיח שתוחלת התשלום אליו היא v לפחות אם השחקן הוא כחול ו-$-v$ לפחות אם השחקן הוא אדום. המפתח לתוצאה זו הוא פונקציית ריצ'מן:

[1]לפי התיאור שנתנו מספר המהלכים הוא תמיד n, שהרי אנו דורשים שהמשחק יימשך עד אשר כל המשתנים נקבעים. עם זאת, עבור פונקציות רבות מספיקה השמה חלקית של המשתנים על מנת לדעת מה יהיה הערך הסופי של הפונקציה, ובמצב זה ניתן לומר שהמשחק מסתיים לאחר שהערך של f לא תלוי בהמשך ההשמות של המשתנים.

הגדרה 0.4 יהי G משחק קומבינטורי. **פונקציית ריצ'מן** עבור G היא פונקציה $R : V \to \mathbb{R}$ המקיימת:

- לכל $v \in T$, $R(v) = \nu(v)$.
- לכל $v \in V \setminus T$, $R(v) = \frac{1}{2}\left(R^{+}(v) + R^{-}(v)\right)$.

כאשר $R^{+}(v) = \max\{R(w) \mid (v,w) \in E_B\}$, $R^{-}(v) = \min\{R(w) \mid (v,w) \in E_R\}$.

אנו מראים תוצאת קיום ויחידות עבור פונקציות ריצ'מן, ומראים שהם מאפיינים משחק אופטימלי בצורה הבאה:

משפט 0.5 לכחול (אדום) יש אסטרטגיה טהורה המבטיחה שהמשחק יסתיים בהסתברות אחד, ושתוחלת הרווח הוא לפחות $R(v)$ ($-R(v)$).

בפשטות, האסטרטגיה אומרת שאם האסימון נמצא כרגע בקודקוד v, ועל כחול לעשות מהלך, עליו להזיז את האסימון לקודקוד w כך ש $R^{+}(v) = R(w)$ (ישנו כלל נוסף החל במקרה שיש יותר מאפשרות אחת לבצע מהלך כזה, אך לא נדון בו בתקציר).

משחקי ריצ'מן

כאמור, משחקי ריצ'מן מתנהלים באופן הבא: בתחילת המשחק לכל אחד מהשחקנים יש כמות כסף, לכחול b, ולאדום r. נניח כי סכום הכסף הוא 1. לפני כל תור מתבצע מכרז על הזכות לעשות את המהלך הבא. מי שמציע את ההצעה הגבוהה משלם את סכום הצעתו ליריב, ומבצע מהלך (אם שתי ההצעות שוות הדבר נחשב כניצחון לכחול).

כפי שנרמז מהשם, פונקציות ריצ'מן הם המפתח גם לאנליזה של משחקי ריצ'מן. האנליזה כאן מתמקדת במשחקים בעלי שתי תוצאות אפשריות בלבד, ז"א משחקים שבסופם יש מנצח ומפסיד. פורמאלית, אלה משחקים עבורם $\nu(T) \subseteq \{0,1\}$. עבור משחקים אלה, האסטרטגיות האופטימליות מאופינות ע"י המשפט הבא:

משפט 0.6 יהי $G = (V, E_B, E_R, T, \nu)$ משחק קומבינטורי, ותהי R פונקצית ריצ'מן של G. נניח שהמשחק מתחיל בקודקוד $v \in V$. אם $r < R(v)$ אזי לכחול יש אסטרטגיה אופטימלית. אם $r > R(v)$ אזי לאדום יש אסטרטגיה אופטימלית.

האסטרטגיה עצמה כוללת שני מרכיבים: כלל לפיו מחליטים כמה להציע בשלבי המכרז, וכלל הקובע כיצד להזיז את האסימון במקרה של ניצחון במכרז. לגבי הזזת האסימון, הכלל יהיה כמו במשחקים עם תורות אקראיים. לגבי ההצעות במכרז, הכלל יותר מורכב ממה שניתן לפרט בתקציר, אך אם האסימון נמצא ב$v \in V$, ההצעה תהיה $R^{+}(v) - R(v) + \varepsilon$, כאשר ε הוא "מקדם ביטחון" התלוי בסדרת המהלכים האחרונים.

נשים לב שמעבר לכך שפונקציית הריצ'מן קובעת את האסטרטגיה האופטימלית הן במשחקי ריצ'מן והן במשחקים עם תורות אקראיים, ישנו קשר פשוט יותר: $R(v)$ הוא בדיוק ההסתברות של כחול לנצח במשחק עם תורות אקראיים המתחיל בv, והוא בדיוק החלק היחסי של הכסף שאדום צריך לעבור על מנת להבטיח ניצחון במשחק ריצ'מן.

המשפט האחרון מאפשר לתאר אסטרטגיות אופטימליות גם עבור משחקי ריצ'מן עם יותר משתי תוצאות אפשריות, אולם במקרה זה הניתוח אינו תלוי בפונקציית ריצ'מן שהגדרנו עבור משחקים עם תורות אקראיים.

שאלות אלגוריתמיות

יתר העבודה דנה בשאלות אלגוריתמיות. אנו דנים בבעייה של מציאת פונקצית ריצ'מן של גרף נתון, ומראים אלגוריתמים פולינומיאלים במקרים שהגרף הוא בלתי מכוון, ועבור גרפים בהם דרגת היציאה של כל הקודקודים קטנה משתיים. לאחר מכן אנו מראים אלגוריתם עבור גרפים כלליים, אשר את זמן ריצתו איננו יודעים לחסום במדויק.

בשאלת מציאת בחירה אופטימלית במשחקי בחירה עם תורות אקראיים, אנו מראים אלגוריתם הסתברותי אשר פועל בזמן $O\left(\frac{1}{\varepsilon^2}\ln\left(\frac{1}{\delta}\right)\right)$ ואשר מוצא, בהסתברות $1-\delta$, בחירה שערך המשחק אם בוחרים בה הוא במרחק ε לכל היותר מהערך של בחירה אופטימלית.

לבסוף, אנו נדרשים לשאלה הבאה: ניתן "לנצל" משחק אופטימלי על מנת לחשב פונקציה $f:\{-1,1\}^n\rightarrow\mathbb{R}$ בצורה הבאה: תהי S אסטרטגיה טהורה ואופטימלית עבור משחק הבחירה עם תורות אקראיים המתאים לf. נניח שאנו רוצים לחשב את $f(x)$. בכל שלב במשחק, אם ההשמה החלקית היא (A,B), ואם $S(A,B)=i$, אזי אם $x_i=1$ נאמר שכחול מנצח בהטלה הבאה, ואם $x_i=-1$ נאמר שאדום מנצח בה. נמשיך את המשחק עד שתוצאת המשחק ידועה. תוצאה זו תהיה שווה ל$f(x)$. נשאלת השאלה כמה שיטה זו יעילה, במונחים של מספר הביטים מx שעלינו לקרוא, ביחס לשיטות אחרות.

למעשה, השיטה שתוארה לעיל בונה עץ החלטה עבור f, והשאלה היא כיצד עץ זה משתווה לסיבוכיות הזמן של עצי החלטה דטרמיניסטים אופטימלים עבור f. שאלה נוספת היא כיצד העץ שנבנה בצורה זאת משתווה ביחס לסיבוכיות המקום של עצי החלטה עבור f. לגבי השאלה האחרונה, שוער ב[1] שבנייה זו היא אופטימלית. בניגוד לכך, אנו מראים דוגמא לפונקציה עבורה בניה זו אינה אופטימלית, לא במונחי מקום ולא במונחי זמן.

References

[1] Miao Chen. Toward optimal tree construction of monotone functions. Master's thesis, Duquesne University, August 2005.

[2] Andrew J Lazarus, Daniel E Loeb, James G Propp, Walter R Stromquist, and Daniel H Ullman. Combinatorial games under auction play. *Games and Economic Behavior*, 27(2):229–264, 1999.

[3] Andrew J Lazarus, Daniel E Loeb, James G Propp, and Daniel Ullman. Richman games. *Games of no chance*, 29:439–449, 1996.

[4] Yuval Peres, Oded Schramm, Scott Sheffield, and David B Wilson. Random-turn hex and other selection games. *American Mathematical Monthly*, 114(5):373–387, 2007.

Abstract. A two player, zero-sum, combinatorial game is one in which two players, Blue and Red, take turns moving a token along the edges of a finite directed graph. Some nodes are designated *terminal nodes.* Should the token reach one of these nodes play ends, and a zero-sum payoff associated with the node is made. We do not allow any chance element (such as the rolling of dice) in the formulation of these games.

These games are particularly enticing to mathematicians for a number of reasons. First, the model covers many examples of “real” games, from Chess to Tic-Tac-Toe. Thus, the study of these games may give one an edge against one's opponents - always a worthy goal. Second, there is a profound gap between what is mathematically provable and what is practically accomplishable: While the finitary, deterministic, nature of these games ensures the existence of deterministic optimal strategies (or else the possibility of a mutually enforced draw, with the token never reaching a terminal node), in general there is no way to discover an optimal strategy other than backwards induction over the game graph. As most games have unmanageably large graphs, this approach is of little use.

We study variations on the turn-allocation mechanism of combinatorial games. In random-turn games, each turn is allocated according to the result of the toss of a coin. In Richman games, an auction is held before each turn, with the higher bidder paying his bid to his opponent and earning the right to decide which player makes the next move. Regarding the former, we review the work of Peres, Schramm, Sheffield, and Wilson in [**18**]. Regarding the latter, we review the work of Lazarus, Loeb, Propp, Stromquist and Ullman in [**10**] and [**9**].

The bulk of our work deals with characterizing optimal play in each of these game types. It turns out that even if the graph of the underlying combinatorial game contains cycles, at least one player (in Richman games) or either player (in random-turn games) can force the game to end using an optimal strategy. This is shown by associating a real-valued “Richman function” to the nodes of the graph. After giving a suitable definition, we show that the Richman function of a graph exists and is unique, and maps each node to the expected value of an optimally-played random-turn game beginning from that node. We then demonstrate how the Richman function governs optimal play of both random-turn and Richman games. Thus, there is a deep connection between the two variants.

We next analyze a special instance of combinatorial games: Selection games. In a selection game each player, in turn, selects an element from some finite set S and adds it to his collection (Hex is a well-known example). Once all elements of S have been selected, the payoff function depends on the partition of S. An equivalent formulation is to set some $f : \{-1, 1\}^n \to \mathbb{R}$. At the start of the game, the variables $x_1, \ldots, x_n$ are undetermined. Each turn the player selects one of the variables and assigns it a value (1 for Blue, -1 for Red). Once all variables have been assigned, the payoff to Blue is $f(x_1, \ldots, x_n)$. We show that selection games played under the random-turn rules have the property that optimal play results in a random variable assignment. Thus the value of the game is the expectation $\mathbb{E}_{x\sim\{-1,1\}^n}[f(x)]$. From this fact it follows that a variable is an optimal selection (for either player) iff it maximizes its related degree one Fourier coefficient, $\hat{f}(i)$. The connection to Fourier analysis of Boolean functions allows us to to describe some properties of optimal random-turn games, such as expected game length.

The remainder of our work considers algorithmic questions. We discuss various ways to compute the Richman function of a graph, and show a polynomial time probabilistic algorithm for finding a near-optimal move in a selection game. We close by considering the feasibility of optimally-played random-turn selection games as a computational model for the underlying Boolean function. In particular we show that an optimal selection is not necessarily an optimal root of a decision tree for the same function, thus disproving a conjecture by Miao Chen in [**3**].

Contents

Chapter 1. Introduction 5

Chapter 2. Preliminaries 8
- 2.1. Combinatorial Games 8
- 2.2. Introduction to Boolean Analysis 11

Chapter 3. Random-Turn Games 14
- 3.1. Random-Turn Selection Games 14
- 3.2. General Random-Turn Games 21

Chapter 4. Richman Games 27
- 4.1. Win-or-Lose Richman Games 27
- 4.2. Multi-Valued Richman Games 29

Chapter 5. Algorithms and Complexity 31
- 5.1. Algorithms for Determining Optimal Moves 31
- 5.2. Optimal Play of Random-Turn Selection Games as Decision Procedures 36

Chapter 6. Future Work 41

Appendix A. Influence-Based and Optimal Decision Trees for Level-2 Recursive Ternary Majority 42

Bibliography 45

CHAPTER 1

Introduction

A two player combinatorial game is one in which two players take turns moving a token along the edges of a given directed graph. Some nodes are terminal nodes, with an associated payoff to each player. Should the token be moved to one of these nodes play will end and each player will be awarded the payoff associated with the node. Many popular games can be modeled this way: Chess, Checkers, and Go are some examples. Games where the available moves at each node depend on the outcome of some chance event (such as the role of the dice in Monopoly or backgammon) are not considered combinatorial games. The main pursuit of combinatorial game theorists is determining which of the two players (if any) has a winning strategy when play is started on a given vertex, and what that strategy is.

Perhaps the earliest result of modern game theory, Zermelo's theorem[1], implies that when the underlying graph is acyclic there exist mutually optimal strategies for the two players discoverable by backwards induction on the graph, using the minimax algorithm. Moreover this algorithm is computationally efficient as a function of the size of the graph. When the underlying graph contains directed cycles there may also be the possibility of a mutually forced draw (i.e. play the players employ strategies that ensure the token never reaches a terminal node). This too is discoverable by induction on the graph. In practice, however, this algorithm is of little use as all but the most trivial games have graphs so large that any algorithm taking the entire graph into account is infeasible.

This situation, in which a conceptually simple algorithm to determine the optimal move is known, but where the algorithm is practically useless is tantalizing. It has invited the use of non-trivial mathematical techniques to aid in the analysis both of specific combinatorial games as well as broad classes of such games. It has also resulted in the invention (or discovery, depending on the reader's philosophical bent) of games solely for the purpose of analyzing their mathematical properties. For the most part such games are uninspiring from a player's point of view, but a select few of them (most notably Hex, described as example 7) have become popular among the general game playing public.

Example 1. **Tic-Tac-Toe:** The best-known combinatorial game is probably the children's game of Tic-Tac-Toe, or noughts and crosses. The game is played on a three by three grid. Each turn, the player to move selects an empty cell and colors it his color (or marks it with his shape, usually either 'x' or 'o'). The first player to color three cells of a row, column, or diagonal with his color wins. If the grid is fully-colored without any of the above being monochromatic, the game is declared a tie. For many of us, the first game-theoretic result we discovered was

[1]For an interesting discussion of the history of Zermelo's theorem, as well as a translation of the German article where the theorem first appeared, see [**20**].

that optimal play of Tic-Tac-Toe always leads to a draw. Some of us may have even developed perfect strategies, that never lose and are able to capitalize on an opponent's mistake and turn the game into a win.

k^n Tic-Tac-Toe is played on the points $[k]^n$. Each player, in turn, selects one of these points and colors it his own color. The winner is the first to color a combinatorial line of length k his own color (a combinatorial line is a progression of elements of $[k]^n$ s.t. the difference between successive elements is non-zero and is equal to $0, 1$ or -1 in each coordinate). If the board is fully-colored with no monochromatic combinatorial lines, the game is declared a tie. In this notation, standard Tic-Tac-Toe is 3^2.

Not much is known about optimal strategies for Tic-Tac-Toe in dimensions higher than 2. This is true even for small board sizes: While 3^3 is easily shown to be a first player win, for 4^3 this is not so easy. For 5^3 it is not even known whether the game is a first player win or a draw[2]. Since any game of 5^3 must be at least 9 turns long (as the winner must claim 5 points), there is a trivial lower bound of $\frac{5^3!}{(5^3-9)!} \approx 2^{62}$ on the size of the game tree (which in actuality is far larger). This is enough to make backtracking impractical. Thus, even as simple and small a game as 5^3 Tic-Tac-Toe requires either significant computational power or non-trivial mathematics (and quite likely both) to analyze. For more on Tic-Tac-Toe, and combinatorial game theory generally, the reader is referred to Jószef Beck's *Combinatorial Games: Tic-Tac-Toe Theory* ([**1**]).

An outgrowth of the study of combinatorial games is the study of games derived from them by making small changes to the rules. One such possibility is to change the way turns are allocated. In Richman games an auction is held before each turn, with the winner paying his bid to the opponent and earning the right to decide which player makes the next move. In random-turn games a coin is flipped before each turn with the winner deciding on the next move. Both these models are interesting in their own right as natural models for real-world conflicts where the players don't necessarily alternate their actions. Furthermore, both models offer examples of graphs for which the combinatorial game seems difficult, whereas the Richman or random-turn version is computationally tractable.

Selection games are a commonly played subset of combinatorial games. Here, each player in turn selects a previously-unselected element from some fixed set, and adds it to his own collection. Once all elements of the original set have been selected the payoff to the players is a function of the partition of the original set between the two players. Hex is an example of a selection game, with the elements being the hexagonal tiles. Go and Tic-Tac-Toe are not, since in the former stones can be removed over the course of play, thereby "unselecting" previously selected elements, and in the latter the order in which elements are selected potentially influences the outcome of the game.

This work surveys the main results for Richman games and random-turn games, and demonstrates how techniques from the analysis of Boolean functions can be used to study Richman and random-turn selection games. This includes a characterization of the optimal move at each point of play, a study of algorithmic methods

[2]Interestingly, it's known that for all board sizes, the game is not a second player win, and therefore is either a first player win or a draw. This can be proved with the very elegant *strategy-stealing argument* (see [**8**]). Unfortunately, *strategy-stealing* doesn't help in constructing optimal strategies.

for finding such moves, as well as lower bounds on the expected length of optimally played random-turn games.

CHAPTER 2

Preliminaries

2.1. Combinatorial Games

This section briefly defines combinatorial games and related concepts. Although this work does not deal directly with combinatorial games, they give the basic structure to both Richman and random-turn games.

As mentioned in the introduction, a (zero-sum) two player combinatorial game is one in which two players, Blue and Red, take turns moving a token along the edges of a given directed graph. At each vertex, the set of edges along which the two players may move the token may not be the same (for example, if the game is Chess, players may only move pieces of their own color). Some nodes are terminal nodes, with an associated payoff to each player. Should the token be moved to one of these nodes play will end and each player is awarded the payoff associated with the node. Formally:

Definition 2. A two player, zero-sum, **alternating-turn combinatorial game** is a tuple $G = (V, E_B, E_R, T, \nu)$, where $(V, E_B \cup E_R)$ is a finite directed graph, $T \subseteq V$ is the set of terminal nodes, and $\nu : T \to \mathbb{R}$ is the payoff function. We assume that for each $v \in V$, there is a path from v to T in both E_B and E_R. We also assume that there are no outgoing edges from T. The edges in E_B are called blue edges, and those in E_R are red. For notational convenience, we set $E = E_B \cup E_R$.

Definition 3. Let $G = (V, E_B, E_R, T, \nu)$ be a combinatorial game. If $\nu(T) \subseteq \{-1, 1\}$ G is called a **win-or-lose** game.

Definition 4. A **play** of G starting at a node $v \in V$ is a walk on the graph (V, E), alternating between edges from E_B and E_R, that is either infinite or else reaches a terminal node $t \in T$, in which case the payoff to Blue is $\nu(t)$ and to Red is $-\nu(t)$ and the play is said to terminate. A **pure strategy** for Blue (Red) is a function deciding which edge to follow given an odd-length (even-length) proper prefix of a play of G. A player is limited to choosing edges of his own color. Given strategies for Blue and Red as well as a starting vertex, a play of G is induced, and if the play terminates so is a payoff to each player.

Remark. The distinction between blue and red edges is not always important. First, alternating-turn games can always be set up in such a way that the vertices are a disjoint union between vertices where it is always Blue's turn anytime the token reaches the vertex, and those where it is always Red's turn. In this case letting $E_B = E_R$ has no effect on the game. Second, in some contexts (as in [**10, 9**]), the games considered are only those for which $E_B = E_R$. Finally, in some cases (for example monotone selection games, defined later) it would never be an advantage for a player to take an edge of the opposing player's color, hence allowing a single edge set $E = E_B \cup E_R$ would have no effect on optimal play. In cases where there is a single edge set, we'll sometimes use the notation (V, E, T, ν).

Example 5. **Tic-Tac-Toe:** k^n Tic-Tac-Toe was described as example 1. The game graph for k^n is defined recursively as follows:

- The completely uncolored board (represented by $(\emptyset, \emptyset)$) is a node.
- For every node (A, B):
  - If A (B) contains a combinatorial line of length k, then $(A, B) \in T$ and the payoff is 1 (-1).
  - Otherwise, if $A \cup B = [k]^n$ then $(A, B) \in T$ and the payoff is 0.
  - If neither of the above hold then for every $x \in [k]^n \backslash (A \cup B)$,$(A \cup \{x\}, B)$ and $(A, B \cup \{x\})$are nodes with a blue edge from (A, B) to $(A \cup \{x\}, B)$ and a red edge from (A, B) to $(A, B \cup \{x\})$.

2.1.1. Selection Games. Selection games are a special kind of combinatorial game. These are played over a finite set S and Blue and Red take turns selecting previously unselected elements of S until the set S is partitioned into the sets B, R of elements selected by Blue and Red, respectively. Once S has been partitioned, the payoff is decided according to some function $f : 2^S \to \mathbb{R}$, $f(B)$ being Blue's payoff and $-f(B)$ being Red's. These can be thought of as combinatorial games:

- The vertices are pairs of disjoint subsets of S.
- For any vertex (A, B) and $x \in S \backslash (A \cup B)$ there is a blue edge from (A, B) to $(A \cup \{x\}, B)$ and a red one to$(A, B \cup \{x\})$.
- The terminal nodes are partitions of S.
- For any partition (A, B) of S, the payoff function is $\nu(A, B) = f(A)$.

Note that the underlying game graph for selection games is acyclic; in particular, plays of selection games always terminate.

An equivalent way of thinking about selection games is to set some function $f : \{-1, 1\}^n \to \mathbb{R}$. At the start of the game the values of the variables $x_1, \ldots, x_n$ are unknown. During each turn, a player selects some hitherto unassigned variable and assigns its value (with the limitation that Blue must set a variable's value to 1 and Red to -1). Once all the variables have been set to $x_1, \ldots, x_n$, the payoff to Blue is $f(x_1, \ldots, x_n)$. Such a game is completely specified by the function f, and every function $f : \{-1, 1\}^n \to \mathbb{R}$ specifies a selection game. In the following, we will use these two specifications of selection games interchangeably.

Definition 6. Let S the underlying set of a selection game, and let f be the payoff function. If for all $A \subseteq B \subseteq S$, $f(A) \leq f(B)$, (S, f) is called a **monotone selection game**.

For monotone selection games, we may as well let $E_B = E_R$, since it's never advantageous for a player to select an element and "donate" it to his opponent.

Example 7. **Hex:** One well known selection game is Hex, where the players alternate claiming tiles on a rhombus-shaped hexagonal grid (see figure 2.1.1). Tiles are considered adjacent if they share a side. Each player is assigned a (different) pair of opposing edges of the rhombus, and the winner is the player who, when all tiles have been claimed, has a chain of adjacent tiles linking his two sides of the board. Figure 2.1.1 shows a completed game of Hex.

Usually, play is stopped once such a chain has been created, even if the board hasn't been filled. Although our definition of selection games requires that play

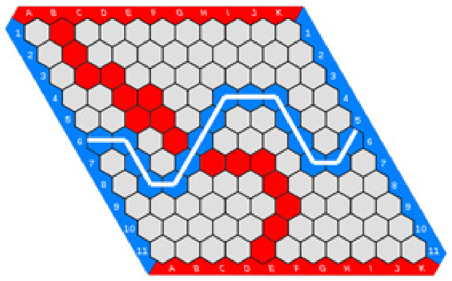

FIGURE 2.1.1. A completed game of Hex, showing a winning chain for Blue (image by Jean-Luc W (own work), licensed under CC-BY-SA-3.0-2.5-2.0-1.0 (http://creativecommons.org/licenses/by-sa/3.0), via Wikimedia Commons).

continue until all elements have been selected, Hex under the normal rules may still be considered a selection game since once a player has managed to connect his two sides his opponent is prevented from doing so, no matter how the remaining tiles are colored. This is not difficult to prove, but requires some care: if the game were played with a different definition of adjacency, for example sharing a point, it wouldn't be true. A less trivial fact is that there is *always* a winner: no matter how the board is filled, one of the players will manage to connect his sides.[1] Taken together, these facts imply that Hex's payoff function takes on only two values, which may as well be 1 (Blue wins) and -1 (Red wins). Thus Hex is a monotone, win-or-lose, selection game. Although any Boolean-valued function can be made into a selection game (and we'll see more examples later), Hex is probably the best example of a "real" game in this category.

EXAMPLE 8. **Weak Tic-Tac-Toe:** At first glance k^n Tic-Tac-Toe might seem to be a selection game: Blue and Red take turns selecting from a set until one of them has a "winning" set. However, the winner of a game of Tic-Tac-Toe is the *first* player to claim k spaces in a row. Unlike in Hex, the first player to select a "winning" set isn't necessarily the only one: If play continues until the board is full a situation may arise where *both* players have claimed such sets. This scenario is not covered under the "regular" rules of Tic-Tac-Toe.

Weak k^n Tic-Tac-Toe is the same as k^n Tic-Tac-Toe, except that Blue wins iff he manages to claim a combinatorial line of length k (equivalently, Red wins iff he prevents Blue from claiming such a line). This has the effect of transforming k^n Tic-Tac-Toe into a monotone, win-or-lose, selection game.

EXAMPLE 9. **Maker-Breaker Games:** In a Maker-Breaker game over a set S, there is a family $\mathcal{A} \subseteq 2^S$ of winning sets. Let $B \subseteq S$ be Blue's set at the end of the game. Then Blue wins if there is some $A \in \mathcal{A}$ s.t. $A \subseteq B$, and otherwise Red wins. Thus Blue, as Maker, aims to select one of the winning sets, and Red, as Breaker, aims to prevent him from doing so. All Maker-Breaker games

[1]For (two different) straightforward proofs, see **[6, 5]**. While this fact may be intuitive, it is mathematically deep. In particular, in **[5]** Gale shows that it is equivalent to the Brouwer fixed-point theorem (after suitably generalizing Hex, the equivalence is true in any dimension).

are monotone, win-or-lose, selection games. Both Hex and Weak Tic-Tac-Toe are examples of Maker-Breaker games.

The study of selection games occupies a significant portion of combinatorial game theory. In general, there is no better way to discover optimal strategies than induction on the game tree. Since the game tree is very large (having 3^n nodes - at each game position, each element may be either unselected, Blue, or Red), this is impractical. As a result for many games, including Hex and Tic-Tac-Toe (for dimensions higher than 2), not much is known, mathematically, regarding optimal play. It is therefore surprising that in the random-turn and Richman versions of these games, a great deal can be said about optimal play ([**10, 9, 18**]), including a strong characterization of the optimal strategies for these games. We present these results in the later chapters.

2.2. Introduction to Boolean Analysis

The main tool we use to analyze random-turn selection games is the Fourier transform for Boolean functions. This chapter introduces the definitions and results we will need. As Boolean analysis as such is not our main concern, most of the propositions are given without proofs. For a more thorough text on the subject, see [**14**]. For the most part we've followed the notations and definitions employed therein.

Boolean analysis concerns itself with the space $\mathcal{H}$ of functions $f : \{-1,1\}^n \to \mathbb{C}$ (for some fixed n). Although most of the functions we encounter will have more restricted ranges (usually $\mathbb{R}$ or even just $\{-1,1\}$), it's useful to carry out analysis in the more general setting since the above functions form an inner product space with the inner product: $\langle f,g\rangle = 2^{-n}\sum_{x\in\{-1,1\}^n} f(x)\overline{g(x)}$. Note that another way of looking at this is to consider $\{-1,1\}^n$ as a sample space with the uniform distribution. Then $\mathcal{H}$ is the set of complex random variables over this space, and $\langle f,g\rangle = \mathbb{E}_{x\sim\{-1,1\}^n}\left[f(x)\overline{g(x)}\right]$. From probability theory we know the latter is an inner product. Depending on the context, in the following we will use the inner product and expectation notations interchangeably. We will sometimes omit the subscript of the expectation, when the underlying probability space is clear from the context.

2.2.1. The Fourier Transform. Fourier analysis introduces a specific orthonormal basis for $\mathcal{H}$, and then attempts to deduce the properties of various functions from their coefficients in this basis. The natural place to start, therefore, is by defining the basis, which we do right after laying out the following notation:

Definition. Let $x \in \{-1,1\}^n$. For every $1 \le i \le n$, let x_i denote the projection of x to its ith coordinate. We will use the following compact notations to signify a change to a single coordinate of x: $x^{+i} = (x_1,\dots,x_{i-1},1,x_{i+1},\dots,x_n)$, $x^{-i} = (x_1,\dots,x_{i-1},-1,x_{i+1},\dots,x_n)$, and $x^{\oplus i} = (x_1,\dots,x_{i-1},-x_i,x_{i+1},\dots,x_n)$.

Definition 10. For every $S \subseteq [n]$, the **parity function** $\chi_S \in \mathcal{H}$ is defined as: $\chi_S(x) = (-1)^{|S\cap\{i:x_i=-1\}|} = \prod_{i\in S} x_i$. For a singleton $S = \{i\} \subseteq [n]$, we will abuse notation and write χ_i instead of $\chi_{\{i\}}$.

Proposition 11. *The set of parity functions $\{\chi_S : S \subseteq [n]\}$ forms an orthonormal basis for $\mathcal{H}$.*

The Fourier transform of a function in $\mathcal{H}$ is simply its representation as a linear combination of parity functions:

DEFINITION 12. Let $f \in \mathcal{H}$, and $S \subseteq [n]$. Then $\hat{f}(S) = \langle f, \chi_S \rangle$. The values $\left\{\hat{f}(S)\right\}_{S\subseteq[n]}$ are called the **Fourier coefficients** of f, as explained in the next proposition. For a singleton $S = \{i\}, 1 \le i \le n$, we will abuse notation and write $\hat{f}(i)$ in place of $\hat{f}(\{i\})$.

PROPOSITION 13. $f = \sum_{S\subseteq[n]} \hat{f}(S)\chi_S$

2.2.2. Monotone Functions and Influences. As our original motivation for studying Boolean functions came from selection games, it is useful to think what properties Boolean functions associated with such games might have. The foremost one is monotonicity: In many selection games, it is never harmful to add elements to one's own set. That is, if $A \subseteq B \subseteq S$, then $f(A) \le f(B)$. In particular, for win-or-lose games this property translates into the effect that once Blue has selected a winning set, continued play of the game will not affect the outcome (in fact, a win-or-lose selection game is monotone if and only if it is a Maker-Breaker game). This property is translated to Boolean functions as follows:

DEFINITION 14. For each n, define the partial order relation on $\{-1,1\}^n$ as: for $x, y \in \{-1,1\}^n$, $x \le y$ if for every i, $x_i \le y_i$. Let $f : \{-1,1\}^n \to \mathbb{R}$ is said to be **monotone** if $\forall x, y \in \{-1,1\}^n, x \le y \implies f(x) \le f(y)$.

A key notion in Boolean analysis, both generally and (as we will show) in its application to random-turn selection games, is that of a variable's influence. Roughly, this is the expected change one is able to effect in the function's value if one has the ability to change the given variable once all the other variables have been decided randomly.

DEFINITION 15. Let $f : \{-1,1\}^n \to \mathbb{R}$. The **influence** of the ith coordinate is defined as: $Inf_i[f] = \mathbb{E}\left[\left(\frac{f(x^{+i})-f(x^{-i})}{2}\right)^2\right]$.

For functions with range $\{-1,1\}$, this is related to the notion of a variable being pivotal:

DEFINITION 16. Let $f : \{-1,1\}^n \to \{-1,1\}$. For $x \in \{-1,1\}^n$, the ith coordinate is **pivotal** if $f(x) \ne f\left(x^{\oplus i}\right)$.

This is related to influence in the following way:

CLAIM 17. Let $f : \{-1,1\}^n \to \{-1,1\}$. Then for every i, $Inf_i[f] = \Pr\left[f(x) \ne f\left(x^{\oplus i}\right)\right]$. That is, the influence of the ith variable is the probability that it is pivotal.

In the special case of a monotone function $f : \{-1,1\}^n \to \{-1,1\}$, a variable's influence is related nicely to its degree-one Fourier coefficients:

PROPOSITION 18. *Let $f : \{-1,1\}^n \to \{-1,1\}$ be monotone. Then for every i, $Inf_i[f] = \hat{f}(i)$.*

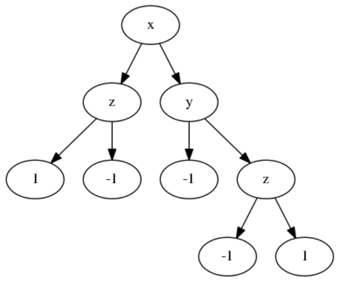

FIGURE 2.2.1. A decision tree for the function $f(x, y, z) = (\bar{x} \wedge \bar{z}) \vee (x \wedge y \wedge z)$.

2.2.3. Decision Trees for Boolean Functions. Among the questions with which we'll concern ourselves is the expected length of play of a game. For a selection game $f : \{-1, 1\}^n \to \mathbb{R}$ this is equivalent to the number of input bits read before the value of the function is known. This in turn is closely related to the notion of a decision tree for a Boolean function.

A decision tree is a binary tree that encodes an algorithm for computing a Boolean function: One starts at the tree's root, where the name of one of the variables is written. One examines the value of the variable. If it is 1, one proceeds to the right child, and if it is -1, one proceeds to the left. This is repeated, with each internal node labeled with the name of one of the variables, until a leaf is reached, where the value of the function is written. Thus, each $x \in \{-1, 1\}^n$ induces a particular walk on the tree, starting at the root and ending at a leaf. Given a distribution on $\{-1, 1\}^n$, a distribution on the random walks on the tree is induced, and the sequence of vertices visited is a Markov chain. We'll use the following notations:

DEFINITION 19. Let T be a decision tree for a Boolean function $f : \{-1, 1\}^n \to \mathbb{R}$, and let $\mathcal{D}$ be a distribution on $\{-1, 1\}^n$. Let $\ell(T(x))$ be the length (i.e. the number of edges, or equivalently the number of times a variable is examined) of the walk $x \in \{-1, 1\}^n$ induces on T. Then the expected number of variables read to compute f is $\mathbb{E}_{x \leftarrow \mathcal{D}}[\ell(T(x))]$. Let $\mathcal{T}$ be the set of all decision trees computing f. The **expected decision tree complexity** of f with distribution $\mathcal{D}$ is defined as $\Delta_{\mathcal{D}}(f) = \min_{T \in \mathcal{T}} \mathbb{E}_{x \sim \mathcal{D}}[\ell(T(x))]$. If $\mathcal{D}$ is clear from the context, we'll omit the subscripts.

REMARK. Although this definition suits our purposes, it is somewhat unusual. It's more common to define the complexity of a decision tree as the maximum, among all distributions on $\{-1, 1\}^n$, of the expected length of a walk on the tree. Note that this is equal to the height of the tree. The (deterministic) decision tree complexity of a function is then the minimal complexity of a decision tree computing it. The expected decision tree complexity is also distinct from the randomized decision tree complexity, which is the minimum, amongst all probability distributions of decision trees computing f, of the maximum among probability distributions on $\{-1, 1\}^n$ of the expected length of a walk on the tree.

EXAMPLE 20. Figure 2.2.1 shows a decision tree for the function $f(x, y, z) = (\bar{x} \wedge \bar{z}) \vee (x \wedge y \wedge z)$. When $\mathcal{D}$ is the uniform distribution on $\{-1, 1\}^n$, the expected number of variables read is $\frac{9}{4}$. This is also equal to $\Delta_{\mathcal{D}}(f)$.

CHAPTER 3

Random-Turn Games

Random-turn games are the main object of study in this work. A random-turn game is derived from an alternating-turn game by changing the rules so that instead of the two players alternating turns, each time a move is to be made a fair coin is flipped, and whichever player wins the coin toss is allowed to make the next move. Other than the change in the order of play, the rules remain the same: The players may only move the token along edges of their own color, and the game ends with the associated payoff if the token reaches a terminal node.

The random and alternating-turn versions of a game are very different. In particular, while the alternating version has no probabilistic element, probability plays a crucial role in both play and analysis of random-turn games. As will be shown, it is precisely this that makes some random-turn selection games susceptible to techniques from Boolean analysis. The highlight of this approach will be to give a characterization of the optimal move for such games, first shown in [**18**]. We'll also discuss the expected length of optimally-played random-turn selection games, and implications for the decision tree complexity of certain functions.

3.1. Random-Turn Selection Games

In the following we'll consider selection games being played on the set $[n]$, which we'll identify with the set of variables of the Boolean payoff function $f : \{-1,1\}^n \to \mathbb{R}$. We'll abuse notation somewhat: For $A \subseteq [n]$, we'll write $f(A)$ to mean $f(x_A)$ where $(x_A)_i = \begin{cases} 1 & i \in A \\ -1 & i \notin A \end{cases}$.

A pure strategy for a random-turn selection game is a function S from disjoint pairs (A,B) of subsets of $[n]$ to elements of $[n]$ s.t. $S(A,B) \in [n] \setminus (A \cup B)$. Thus if Blue is playing according to strategy S, whenever he wins the coin toss and the previously-selected elements are the sets A and B for Blue and Red respectively, Blue will select the element $S(A,B)$. Note that we consider here only "memory-free" strategies that take into account the current game configuration only, and not the historical sequence of element selections. If we were to deal with strategies in their full generality we would admit such strategies as well; however it's clear that no advantage can be gained by taking the history into account so we may as well limit our analysis to the "memory-free" strategies.

If Blue and Red are playing according to pure strategies S and T respectively, every sequence of n coin-flips induces a sequence of selections, culminating in a partition of $[n]$. We'll write $f_{S,T}(x)$ for the payoff to Blue when the game is played with strategies S and T and the results of the coin-flips are $x \in \{-1,1\}^n$. $f_{S,T}(x)$ is also called the value of the game.

3.1.1. Optimal Play. As the result of the coin flips is not known ahead of time, the outcome of a game f when the players use strategies S and T isn't a definite value (even if S and T are pure). Instead, we'll consider the expected value of the game, $\mathbb{E}_{x\sim\{-1,1\}^n}\left[f_{S,T}\left(x\right)\right]$. Just as with alternating-turn games, as finite, turn-based, games of perfect information there exist mutually optimal pure strategies for random-turn selection games. Such strategies, as well as the expected values under optimal play from any position can be calculated as follows: For any disjoint $A, B \subseteq [n]$ s.t. $A \cup B = [n]$, the game is over and its value is $f\left(A\right)$. Now, if the expected value of a game is known for all disjoint A, B s.t. $|A| + |B| = k + 1$, then for any A, B s.t. $|A| + |B| = k$, if it is Blue's turn he should choose an element $x \in [n] \setminus (A \cup B)$ that maximizes the expected value of $A \cup \{x\}, B$, and if it is Red's turn he should choose an element minimizing $A, B \cup \{x\}$. The expected value from position A, B is then the average of these two values.

The preceding paragraph gives an algorithm computing the expected value and optimal move from a given position; however it requires backtracking over the entire game graph, which is of size 3^n (since each element can be either unselected, selected by Blue, or selected by Red) hence its running time is exponential in n. It turns out that for random-turn selection games there is a fairly simple characterization of both the expected value of a game from any point of play and the optimal moves themselves. The key is the following theorem, given as theorem 2.1 in **[18]**:

Theorem 21. *The value of a random-turn selection game is the expectation of $f\left(x\right)$ when x is chosen uniformly from $\{-1,1\}^n$. Moreover, any optimal strategy for one player is also an optimal strategy for the other player.*

Proof. Let S be an optimal strategy for Blue. Should Red play by the same strategy, each element of $[n]$ has an even probability of ending up in A or in B. Thus $\mathbb{E}\left[f_{S,S}\left(x\right)\right] = \mathbb{E}\left[f\left(x\right)\right]$, so the game's expected value under optimal play is no more than $\mathbb{E}\left[f\left(x\right)\right]$. By a symmetric argument, we can show that the value of the game must also be at least $\mathbb{E}\left[f\left(x\right)\right]$, and so it is, in fact, equal to $\mathbb{E}\left[f\left(x\right)\right]$.

For the second part of the theorem, let S be an optimal strategy for Blue, and let $x_i = S\left(\emptyset, \emptyset\right)$. It's enough to show that if Red wins the initial coin-flip, selecting x_i is an optimal opening move (and the claim will follow by induction). We know that the expected value of the game is $\mathbb{E}\left[f\right]$ and that Red achieves this by playing S. Thus it remains to show only that if Red plays S Blue can't improve his position by playing a different strategy. Assume, for a contradiction, that if Red plays S Blue can improve his position by selecting x_j in the opening move. We know that the expected value of play is then $\frac{1}{2}\left(\mathbb{E}\left[f|_{x_j=1}\right] + \mathbb{E}\left[f|_{x_i=-1}\right]\right)$. Since this gives Blue an advantage over playing S,

$$\frac{1}{2}\left(\mathbb{E}\left[f|_{x_j=1}\right] + \mathbb{E}\left[f|_{x_i=-1}\right]\right) > \mathbb{E}\left[f\right] = \frac{1}{2}\left(\mathbb{E}\left[f|_{x_i=1}\right] + \mathbb{E}\left[f|_{x_i=-1}\right]\right)$$

Hence $\mathbb{E}\left[f|_{x_j=1}\right] > \mathbb{E}\left[f|_{x_i=1}\right] \implies \mathbb{E}\left[f|_{x_j=-1}\right] < \mathbb{E}\left[f|_{x_i=-1}\right]$. Now, since x_i is an optimal selection for Blue, we know that

$$\frac{1}{2}\left(\mathbb{E}\left[f|_{x_i=1}\right] + \mathbb{E}\left[f|_{x_j=-1}\right]\right) \geq \mathbb{E}\left[f\right]$$

But

$$\frac{1}{2}\left(\mathbb{E}\left[f|_{x_i=1}\right] + \mathbb{E}\left[f|_{x_j=-1}\right]\right) < \frac{1}{2}\left(\mathbb{E}\left[f|_{x_i=1}\right] + \mathbb{E}\left[f|_{x_i=-1}\right]\right) = \mathbb{E}\left[f\right]$$

which is a contradiction. So selecting x_i is optimal for both Blue and Red. $\square$

Theorem 21 has several interesting consequences. The first is that although a lot of thought and calculation might go into planning an optimal strategy for a random-turn selection game, if play indeed proceeds optimally the outcome will look no different than had both players played randomly. Second, if both Blue and Red play according to some pure optimal strategy S, then at every point in the game we know which element will be claimed next, regardless of the result of the next coin toss. The only unknown is which of the players will claim it.

Theorem 21 also tells us that under certain circumstances a shortcut can be taken in calculating the expected value of the game under optimal play. Whereas backtracking requires considering a graph of size 3^n, in order to know the value of the game it's enough to know $\mathbb{E}[f]$. It's often quite simple to calculate this, for example by taking advantage of certain symmetries among the variables. Even if the only way to calculate $\mathbb{E}[f]$ is by iterating through all possible variable assignments, there is still some gain: there are only 2^n variable assignments, whereas there are 3^n nodes on the game graph. The drawback is that opposed to with backtracking, the optimal strategy isn't calculated as part of the algorithm. Despite this, we can still give a nice characterization of the optimal move at any point in play:

THEOREM 22. *A variable $x_i, 1 \leq i \leq n$, is an optimal choice for the opening move of a selection game iff $\hat{f}(i)$ is maximal.*

PROOF. By definition,

$$\hat{f}(i) = \langle f, \chi_i \rangle = \frac{1}{2}\left(\mathbb{E}[f(x)\,|x_i = 1] - \mathbb{E}[f(x)\,|x_i = -1]\right)$$

We also know that

$$\mathbb{E}[f(x)] = \frac{1}{2}\left(\mathbb{E}[f(x)\,|x_i = 1] + \mathbb{E}[f(x)\,|x_i = -1]\right)$$

By adding the equations we get:

$$\mathbb{E}[f(x)\,|x_i = 1] = \mathbb{E}[f(x)] + \hat{f}(i)$$

Since the game played on the function $f|_{x_i=1}$ is also a selection game, its value under optimal play, according to theorem 21, is $\mathbb{E}[f|_{x_i=1}] = \mathbb{E}[f(x)\,|x_i = 1]$. From here it follows that a selection is optimal for f iff it maximizes $\mathbb{E}[f(x)] + \hat{f}(i)$, iff it maximizes $\hat{f}(i)$. □

This result was presented in a slightly weaker form, for monotone win-or-lose games, as lemma 3.1 in [**18**], where it takes on the following interpretation: since in these circumstances $\hat{f}(i) = Inf_i[f] = \Pr\left[f(x) \neq f\left(x^{\oplus i}\right)\right]$ (proposition 18), a selection is optimal iff the corresponding variable has the highest probability of being pivotal.

The statement of theorem 22 refers to optimal *opening* moves; however, since assigning some values of variables as part of gameplay leaves us with a selection game, theorem 22 characterizes the optimal move at any stage of gameplay. The only caveat is that the Fourier coefficients must be calculated w.r.t. the variables already assigned. We'll use the following notation: Let $J \subseteq [n]$. x_J will denote an assignment of variables in J, which is a partial assignment of variables in $[n]$. Then $f|_{x_J} : \{-1,1\}^{n-|J|} \to \mathbb{R}$ will denote the function f given the partial assignment.

3.1.2. Expected Length of Optimal Play. For some random-turn selection games, the payoffs may be known before all the variables have been assigned. This is often the case in monotone, win-or-lose selection games. For example in Hex, once one of the players has a chain of tiles connecting his two sides, he is the winner no matter how play unfolds from that point. If we decide that play ends once the outcome is known, it's interesting to consider the expected length of an optimally played random-turn selection game. Assuming both players play according to the same optimal strategy, a random-turn selection game can be thought of as carrying out the computation of the value of the underlying function according to some decision tree, where at each node the variable being queried maximizes $\widehat{f|_{x_J}}(i)$ (x_J being the assignment of variables up to that point). Accordingly, we can use bounds on the expected decision tree complexity of f to bound the expected length of play. To this end we present the following proposition, which is a strengthening of inequality 2 in [**18**], where the result is stated only for monotone, win-or-lose functions:

Proposition 23. *Let S be an optimal strategy for a non-constant game f. Let $\mathbb{E}[\#turns]$ be the expected number of turns when both Blue and Red play according to S. Then*

$$\mathbb{E}[\#turns] \geq \frac{\left(\sum_{i=1}^{n} \hat{f}(i)\right)^2}{Var[f]}$$

Proof. We'll use the following notation: For $c \in \mathbb{R}$, the function $f_c : \{-1,1\}^n \to \mathbb{R}$ is defined as $f_c(x) = f(x) + c$.

It's clear that a strategy is optimal for f_c iff it is optimal for f, and that for any x, the number of steps taken to evaluate $f_c(x)$ is the same as the number of steps taken to evaluate $f(x)$. Further, due to the orthogonality of the Fourier basis for any $1 \leq i \leq n$, $\widehat{f_c}(i) = \widehat{f}(i)$. We'll prove the following family of inequalities:

$$\forall c \in \mathbb{R}, \mathbb{E}[\#turns] \geq \frac{\left(\sum_{i=1}^{n} \hat{f}(i)\right)^2}{\mathbb{E}\left[(f+c)^2\right]}$$

Let $1_i(x) = \begin{cases} 1 & x_i \textit{ examined on input } x \\ 0 & \textit{otherwise} \end{cases}$. Then $\mathbb{E}[\#turns] = \mathbb{E}\left[\sum_{i=1}^{n} 1_i(x)\right]$.
Now, by definition:

$$\sum_{i=1}^{n} \hat{f}(i) = \sum_{i=1}^{n} \widehat{f_c}(i) = \sum_{i=1}^{n} \mathbb{E}[f_c(x)\, x_i] = \mathbb{E}\left[f_c(x) \sum_{i=1}^{n} x_i\right]$$

For any x, if x_i isn't examined, it means that $f_c(x^{+i}) = f_c(x^{-i})$. Hence the contributions of x^{+i}, x^{-i} to $f_c(x)\, x_i$ cancel each other out. So:

$$\sum_{i=1}^{n} \hat{f}(i) = \mathbb{E}\left[f_c(x) \sum_{i=1}^{n} 1_i x_i\right] \leq \sqrt{\mathbb{E}\left[f_c(x)^2\right] \mathbb{E}\left[\left(\sum_{i=1}^{n} 1_i x_i\right)^2\right]}$$

Where the inequality follows from Cauchy-Schwarz. For any $j \neq i$, $\mathbb{E}[1_i 1_j x_i x_j] = 0$, since conditioned on x_i being examined before x_j and $x_i = 1$, x_j is 1 or -1 with equal probabilities. Similar distributions hold for other combinations of the order of examination of x_i and x_j and the value of the first query, so the expected value is 0. This implies that $\mathbb{E}\left[\left(\sum_{i=1}^{n} 1_i x_i\right)^2\right] = \mathbb{E}\left[\sum_{i=1}^{n} 1_i^2\right] = \mathbb{E}\left[\#turns\right]$. Hence (after squaring the inequality):

$$\mathbb{E}\left[\#turns\right] \geq \frac{\left(\sum_{i=1}^{n} \hat{f}\left(i\right)\right)^2}{\mathbb{E}\left[f_c^2\right]} = \frac{\left(\sum_{i=1}^{n} \hat{f}\left(i\right)\right)^2}{\mathbb{E}\left[\left(f+c\right)^2\right]}$$

Setting $c = -\hat{f}\left(\emptyset\right)$ gives the desired inequality. Furthermore, since $\mathbb{E}\left[\left(f+c\right)^2\right]$ is minimal when $c = -\hat{f}\left(\emptyset\right)$, this is the tightest bound obtainable from the above family of inequalities. □

Remark. Inequality 2 in [**18**] gives this inequality in the case that f is a win-or-lose monotone function, and with $c = 0$ (where c is as in the proof). This gives the following:

$$\mathbb{E}\left[\#turns\right] \geq \left(\sum_{i=1}^{n} I_i\left[f\right]\right)^2$$

which uses the fact that $\hat{f}\left(i\right) = Inf_i\left[f\right]$ when f is a monotone win-or-lose function. This is based on the same inequality appearing in [**17**], with the Fourier coefficients replacing the influences, as the monotonicity and Boolean range assumptions aren't made. The proof is almost identical to that of the more general formulation we've presented; the only difference is the consideration of different values for c and finding the value that gives the sharpest bound. The more general inequality seems to have appeared for the first time in [**16**].

Another bound on the expected length of optimal play is shown in [**18**]. The following inequality, for a monotone function $f : \{-1,1\}^n \to \{-1,1\}$ is given in [**15**] as theorem 3.1:

$$Var\left[f\right] \leq \sum_{i=1}^{n} \Pr\left[x_i \text{ } examined\right] Inf_i\left[f\right]$$

In the setting of random-turn games, this implies:

Proposition 24. *Let $f : \{-1,1\}^n \to \mathbb{R}$ be a random-turn monotone win-or-lose selection game. Then, when played optimally:*

$$\mathbb{E}\left[\#turns\right] \geq \frac{Var\left[f\right]}{\max_i Inf_i\left[f\right]}$$

We'll now demonstrate how these propositions can be used to analyze optimal play of several specific random-turn games. In [**18**], the original motivation for defining random-turn games was Hex. In particular, attempts were made to analyze the geometry of optimal play as well as its expected length. While some insights were gained, the tools available weren't enough to provide proofs. In contrast, several other games were introduced for which the above tools enabled proving properties of optimal random-turn play. We'll define And-Or functions, and state the results regarding optimal play. One of these results shows that proposition 24 is tight for And-Or trees played with coins with a specific bias (though we haven't shown this, the results above carry over to the case of biased coins). We then introduce the Tribes functions, which show that proposition 24 is tight for

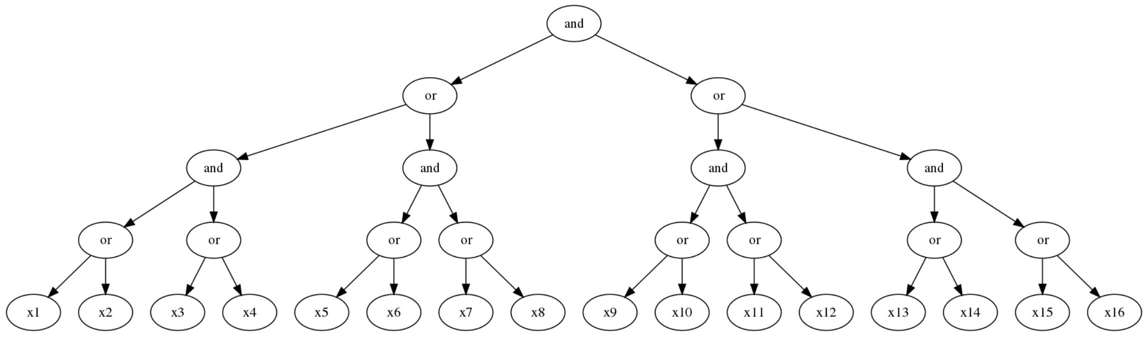

FIGURE 3.1.1. A level-2 And-Or tree.

balanced coins as well. We won't go into the proofs for And-Or trees; they are fairly straightforward, and are similar to the analogous proofs for Tribes functions, which we do provide.

EXAMPLE 25. **And-Or Trees:** The And-Or functions, $f_n : \{-1,1\}^{4^n} \to \{-1,1\}$ are defined recursively as follows:

$$f_0(x) = x$$

For $n \geq 1$, and $(x_1, \ldots, x_{4^{n+1}}) \in \{-1,1\}^{4^{n+1}}$ let $\bar{x_i} = \left(x_{(i-1)\cdot 4^n+1}, \ldots, x_{i\cdot 4^n}\right)$ for $i = 1,2,3,4$. Then f_{n+1} is defined as:

$$f_{n+1}(x_1, \ldots, x_{4^{n+1}}) = (f_n(\overline{x_1}) \vee f_n(\overline{x_2})) \wedge (f_n(\overline{x_3}) \vee f_n(\overline{x_4}))$$

f_n can also be thought of as a complete binary tree of height n, where the leaves are distinct variables and the internal nodes are labeled either "and" or "or" according to the parity of their level (see figure 3.1.1).

[18], theorem 5.1 describes optimal play of And-Or functions as follows:

THEOREM 26. *Consider an optimally played game of level-h And-Or with coin-toss probability p. If a move is played in some subtree T rooted at some vertex v, then the succeeding moves are all played in T until the label of v is determined. Moreover, the labels of the level $h-1$ vertices are determined in an order that is an optimally played game on the tree truncated at level $h-1$.*

Armed with this knowledge, explicit recursion formulas for the length of optimal play as well as the probability of a Blue win on the level-h tree can be given. These formulas depend on the coin bias, p. Setting $p = \frac{3-\sqrt{5}}{2}$ gives, for the expected length of play, precisely $\left(\frac{1+\sqrt{5}}{2}\right)^{2h}$. If we allow f_n to take on the values $\sqrt{\frac{1-p}{p}}$ and $-\sqrt{\frac{p}{1-p}}$ instead of 1 and -1 respectively, proposition 24 is tight.

EXAMPLE 27. **Tribes:** The tribes functions, first constructed in **[2]**, are the standard examples of nearly-balanced Boolean functions (i.e. the function is 1 with probability close to $\frac{1}{2}$) with influences as small as possible. The construction is as follows: Let $k, m \in \mathbb{N}$. Then $T_k^m : \{-1,1\}^{km} \to \{-1,1\}$ is defined as:

$$T_k^m(x_{1,1}, x_{1,2}, \ldots, x_{1,k}, x_{2,1} \ldots, x_{2,k}, \ldots, x_{m,1}, \ldots, x_{m,k}) = \bigvee_{i=1}^{m} \bigwedge_{j=1}^{k} x_{i,j}$$

Informally, the variables are split into m tribes of k members each. The function is 1 iff at least one of the tribes unanimously votes 1, otherwise it is -1. We'll give a complete analysis of optimal play of random-turn tribes, but we'll first generalize the function somewhat: Let $n_1, \ldots n_k \in \mathbb{N}$, and $N = \sum_{i=1}^{k} n_i$. The generalized tribes function, $T_{n_1,\ldots,n_k} : \{-1,1\}^N \to \{-1,1\}$ is defined as $T_{n_1,\ldots,n_k}(x_{1,1}, \ldots x_{1,n_1}, \ldots x_{n,1}, \ldots, x_{n,n_k}) = \bigvee_{i=1}^{k} \bigwedge_{j=1}^{n_i} x_{i,j}$. In other words, similar to the tribes function, the generalized tribes function is 1 iff one the tribes unanimously votes 1. The only difference is that in the generalized tribes function the tribes are allowed to be of different sizes. Note that $T_k^m = T_{n_1,\ldots,n_m}$, if $n_i = k$ for all i. The following hold:

- $\Pr[T_{n_1,\ldots,n_k} = -1] = \prod_{i=1}^{k} (1 - 2^{-n_i})$. For the special case of T_k^m, $\Pr[T_k^m = -1] = \left(1 - 2^{-k}\right)^m$.
- $T_{n_1,\ldots,n_k}$ is a monotone function, hence proposition 18 applies and $\widehat{T_{n_1,\ldots,n_k}}(i) = Inf_i[T_{n_1,\ldots,n_k}] = \Pr[x_i \textit{ is pivotal}]$ for any i. A variable is pivotal iff the other members of its tribe all vote 1 and none of the other tribes votes 1 unanimously. Hence $\widehat{T_{n_1,\ldots,n_k}}(x_{i,j}) = 2^{1-n_i} \prod_{\ell \neq i} (1 - 2^{-n_\ell})$ (note that variables in the same tribe have the same influence). In the case of T_k^m, $\widehat{T_k^m}(i) = 2^{1-k}\left(1 - 2^{-k}\right)^{m-1}$ for all i.
- Partially assigning variables to a generalized tribes function leaves us with a generalized tribes function, or a constant function, as follows: Let $T_{n_1,\ldots,n_k}$ be a generalized tribes function. WLG let's assume we're assigning a value to x_{k,n_k}. If we assign it -1, then the kth tribe has not voted 1 unanimously and so $T_{n_1,\ldots,n_k}|_{x_{k,n_k}=-1} = T_{n_1,\ldots,n_{k-1}}$ (if $k = 1$ we're left with the constant function -1). If we assign $x_{k,n_k} = 1$, then if $n_k = 1$, the kth tribe has voted 1 unanimously so the function is the constant function 1. Otherwise, $n_k > 1$ so we're left with $T_{n_1,\ldots,n_k-1}$.

Using these observations, we'll prove the following claim:

Claim 28. Let $T_{n_1,\ldots,n_k}$ be a generalized tribes function. A variable selection is optimal for the random-turn selection game played on $T_{n_1,\ldots,n_k}$ iff it is a member of the smallest tribe.

Proof. Based on theorem 22, it's enough to prove that the influence of a variable is larger the smaller the tribe to which it belongs. Assume $n_i < n_j$. The influence of variables in the ith tribe is $2^{1-n_i} \prod_{\ell \neq i} (1 - 2^{-n_i})$, and that of a variable in the jth tribe is $2^{1-n_j} \prod_{\ell \neq j} (1 - 2^{-n_i})$. We need to prove that $2^{1-n_i} \prod_{\ell \neq i} (1 - 2^{-n_i}) > 2^{1-n_j} \prod_{\ell \neq j} (1 - 2^{-n_i})$. This is true iff $2^{1-n_i} (1 - 2^{-n_j}) > 2^{1-n_j} (1 - 2^{-n_i})$, which holds since $n_i < n_j$. □

Since, as already observed, assigning a variable in a generalized tribes function leaves us with a generalized tribes function, optimal play of random-turn tribes consists of iteratively selecting a member of the undetermined tribe with the fewest undetermined variables (where by "determined tribe" we mean either all variables set to 1 or at least one variable set to -1). We'll now calculate the expected length of optimally-played random-turn tribes:

Proposition 29. *The expected length of optimally-played random-turn T_k^m is* $2\left(2^k - 1\right)\left(1 - \left(1 - 2^{-k}\right)^m\right)$.

PROOF. Let $E_{m,k}$ be the expected length of optimally-played T_k^m. Then $E_{1,k} = \sum_{i=1}^{k} i2^{-i} + k2^{-k}$ (the first term is the contribution to the expected length when the tribe doesn't unanimously vote 1, and the second term the contribution when it does). Thus $E_{1,k} = 2\left(1 - 2^{-k}\right)$. Now, consider an optimally played game of T_k^{m+1}. Based on theorem 28, we can describe play as follows: First, an optimal game of T_k^m is played on the first m tribes. If the result is 1, then this is the result of the game on T_k^{m+1} as well, so play ends. If the result is -1, then a game of T_k^1 is played on the final tribe of T_k^{m+1}. Hence we get the recursion relation:

$$E_{m+1,k} = E_{m,k} + \Pr\left[T_k^m = -1\right] E_{1,k} = E_{m,k} + 2\left(1 - 2^{-k}\right)^{m+1}$$

To which the solution is:

$$E_{m,k} = 2\left(2^k - 1\right)\left(1 - \left(1 - 2^{-k}\right)^m\right)$$

□

We can use the foregoing analysis to show that proposition 24 is tight: As with any ± 1-valued random variable

$$Var\left[T_k^m\right] = 4\Pr\left[T_k^m = 1\right]\left(1 - \Pr\left[T_k^m = 1\right]\right) = 4\left(1 - 2^{-k}\right)^m\left(1 - \left(1 - 2^{-k}\right)^m\right)$$

Further, all influences are the same, and are equal to $2^{1-k}\left(1 - 2^{-k}\right)^{m-1} = \max_{i,j} Inf_{i,j}\left[f\right]$. Hence:

$$\frac{Var\left[f\right]}{\max_{i,j} Inf_{i,j}\left[f\right]} = \frac{4\left(1 - 2^{-k}\right)^m\left(1 - \left(1 - 2^{-k}\right)^m\right)}{2^{1-k}\left(1 - 2^{-k}\right)^{m-1}} = 2\left(2^k - 1\right)\left(1 - \left(1 - 2^{-k}\right)^m\right)$$

The last term is exactly $E_{m,k}$, so proposition 24 is tight. Further, since by setting $m = \left\lfloor 2^k \ln 2 \right\rfloor$, $\Pr\left[T_k^m = 1\right]$ can be made arbitrarily close to $\frac{1}{2}$, the bound is tight for essentially unbiased functions.

3.2. General Random-Turn Games

In this section we consider random-turn variants of general combinatorial games (as defined in definition 2). We characterize optimal strategies, and discuss the connection between general random-turn games and random-turn selection games.

In the following, let $G = (V, E_B, E_R, T, \nu)$ be a combinatorial game. If the underlying graph contains cycles then not all pairs of strategies result in the game terminating. However, we'll show that optimal strategies exist that result in the game terminating with probability one. For a pair of strategies S, T that terminate with probability one, we'll write $\nu_{S,T}(v)$ for the expected payoff to Blue when the game starts at $v \in V$ and the strategies employed by Blue and Red are S and T respectively.

3.2.1. Richman Functions. The key notion when analyzing general random-turn games is the so-called Richman function, introduced in [**10**] for the purpose of analyzing Richman games (discussed in chapter 4). Simply put, a Richman function for a game G is a function from the set of nodes to the expected value of the game under optimal play should play begin at the given node. Thus whenever Blue wins the coin toss he should move the token to a vertex that maximizes the Richman function, whereas Red should try to minimize it. In the following, we'll formalize and prove these assertions.

We'll need the following notation: Let (V, E_B, E_R) be a (finite) directed graph, with the edges colored either blue or red, or both. For any $f : V \to \mathbb{R}$, for any $v \in V$ with outgoing edges of both colors, $f^+(v) = \max\{f(w) : (v,w) \in E_B\}$ and $f^-(v) = \min\{f(w) : (v,w) \in E_R\}$.

Definition 30. Let $G = (V, E_B, E_R, T, \nu)$ be a game. A Richman function for G is a function $R : V \to \mathbb{R}$ s.t:

- $\forall v \in T,\ R(v) = \nu(v)$
- $\forall v \in V \setminus T,\ R(v) = \frac{1}{2}\left(R^+(v) + R^-(v)\right)$

Recalling definition 2, in our setting, where V is finite and there is a colored path from every node to a member of T, Richman functions exist and are unique. For directed acyclic graphs the proof is straightforward (induction on the maximal distance of a node to T). However for cyclic graphs the proof is more involved. The proofs given here are similar to those given in **[10]** for win-or-lose games, with the required adaptations for multi-valued games.

Proposition 31. *Let $G = (V, E_B, E_R, T, \nu)$ be a game. There exists a Richman function $R : V \to \mathbb{R}$.*

Proof. We'll use the following notation: Let $r_{min} = \min\{\nu(t) : t \in T\}, r_{max} = \max\{\nu(t) : t \in T\}$.

We'll show two existence proofs. The first is an application of the Brouwer fixed-point theorem: Define the function $f : [r_{min}, r_{max}]^V \to [r_{min}, r_{max}]^V$ which maps functions $\varphi : V \to [r_{min}, r_{max}]$ to:

$$f(\varphi)(v) = \begin{cases} \nu(v) & v \in T \\ \frac{1}{2}\left(\varphi^+(v) + \varphi^-(v)\right) & v \notin T \end{cases}$$

Then f is a continuous mapping of a compact, convex set into itself and so it has a fixed point $R : V \to [r_{min}, r_{max}]$. $f(R) = R$, which by definition means that R is a Richman function.

The second proof is more involved, but is constructive and will be of use later when we consider the game-theoretic interpretation of the Richman function.

We'll inductively define a sequence of functions $R_n : V \to \mathbb{R}$ by:

$$R_0(v) = \begin{cases} \nu(v) & v \in T \\ r_{min} & v \notin T \end{cases}$$

And for every $n \in \mathbb{N}$:

$$R_{n+1}(v) = \begin{cases} \nu(v) & v \in T \\ \frac{1}{2}\left(R_n^+(v) + R_n^-(v)\right) & v \notin T \end{cases}$$

(This is the same as letting $R_{n+1} = f(R_n) = f^{n+1}(R_0)$).

For every v, the sequence $\{R_n(v)\}_{n\in\mathbb{N}}$ is bounded above by r_{max}. We'll show by induction that it is increasing: First, clearly $R_1 \geq R_0$. Now, for $n \geq 1$, assume that $R_n \geq R_{n-1}$. Let $v \in V$. If $v \in T$ then $R_{n+1}(v) = \nu(v) = R_n(v)$ so $R_{n+1}(v) \geq R_n(v)$ as desired. Otherwise, $v \in V \setminus T$, in which case $R_{n+1}(v) = \frac{1}{2}\left(R_n^+(v) + R_n^-(v)\right)$. Now, since $R_n \geq R_{n-1}$, we know that $R_n^+(v) \geq R_{n-1}^+(v), R_n^-(v) \geq R_{n-1}^-(v)$. Hence $\frac{1}{2}\left(R_n^+(v) + R_n^-(v)\right) \geq \frac{1}{2}\left(R_{n-1}^+(v) + R_{n-1}^-(v)\right) = R_n(v)$. So $R_{n+1}(v) \geq R_n(v)$, and the sequence is increasing.

As the functions $\{R_n\}_{n\in\mathbb{N}}$ are bounded above and increasing, they converge to a limit $R : V \to \mathbb{R}$, for which the conditions defining Richman functions hold. $\square$

In order to prove the uniqueness of Richman functions (and for the discussion that follows) we'll need the following definition and lemma:

Definition 32. Let $G = (V, E_B, E_R, T, \nu)$ be a game with Richman function R, and let $v \in V \setminus T$. An edge of steepest ascent (descent) from v is an edge $(v, w) \in E_B$ $((v, w) \in E_R)$ s.t. $R(w) = R^+(v)$ $(R(w) = R^-(w))$.

Lemma 33. *Let G_v^+ (G_v^-) be the transitive closure of v under the steepest ascent (descent) relation. That is, $w \in G_v^+$ $(w \in G_v^-)$ iff there exists some path $v = v_0, v_1, \ldots, v_n = w$ s.t. for all $0 \leq i < n, (v_i, v_{i+1})$ is an edge of steepest ascent (descent). Then $G_v^+ \cap T \neq \emptyset$ $(G_v^- \cap T \neq \emptyset)$.*

In other words, for every $v \in V$ there exists a path of steepest ascent (descent) from v to T.

Proof. We'll prove the lemma for the steepest ascent relation. The proof for the steepest descent relation is similar.

Let $w \in G_v^+$ be s.t. $R(w)$ is maximal. If $w \in T$ we're done. Otherwise w has successors in G and so also in G_v^+. Let u be a successor of w along an edge of steepest ascent. Then $R^+(w) = R(u) \leq R(w)$, since $R(w)$ is maximal in G_v^+. It's always true that $R(w) \leq R^+(w)$, so in fact $R(w) = R^+(w) = R(u)$. This implies that $R(w) = R^-(w)$, so all of w's successors in G have the same value, and so they are all in G_v^+. By induction it can be shown that all descendants of w (in G) are in G_v^+ and they all have the same value, $R(w)$. In particular, w has a descendant $u \in T$, so $u \in G_v^+$ and $G_v^+ \cap T \neq \emptyset$ as desired. $\square$

Proposition 34. *Let $G = (V, E_B, E_R, T, \nu)$ be a game. The Richman function $R : V \to \mathbb{R}$ is unique.*

Proof. Let $f_1, f_2 : V \to \mathbb{R}$ be Richman functions. Let $v \in V$ be s.t. $f_1(v) - f_2(v) = M$ is maximal. Let u_1, u_2, w_1, w_2 be appropriately colored successors of v s.t. $f_i^+(v) = f_i(u_i), f_i^-(v) = f_i(w_i)$. The following inequalities hold:

$$f_1(u_1) - f_2(u_2) \leq f_1(u_1) - f_2(u_1) \leq M$$

$$f_1(w_1) - f_2(w_2) \leq f_1(w_2) - f_2(w_2) \leq M$$

By adding the two inequalities we get on the left side $f_1(u_1) + f_1(w_1) - (f_2(u_2) + f_2(w_2)) = 2(f_1(v) - f_2(v)) = 2M$. This implies that all the inequalities are actually equalities, so $f_1(u_1) - f_2(u_1) = M$. Since (v, u_1) is an edge of steepest ascent for f_1, we can thus proceed to show that for any $w \in G_v^+$ (with the ascents measured w.r.t. f_1), $f_1(w) - f_2(w) = M$. Since by lemma 33 there is some $t \in G_v^+ \cap T$, it follows that $M = f_1(t) - f_2(t)$. But by the definition of Richman functions $f_1(t) = f_2(t) = \nu(t)$, so $M = 0$, hence $f_1(v) - f_2(v) = 0$. Since v maximizes $f_1 - f_2$, we conclude that $f_1 \leq f_2$. By switching the roles of f_1 and f_2 and repeating the argument we obtain the reverse inequality, and conclude that $f_1 = f_2$. $\square$

Richman functions characterize optimal strategies for random-turn games in the sense that whenever Blue wins the coin toss he should move the token along a vertex of steepest ascent, and Red should move the token along a vertex of steepest descent. This is embodied in the following theorem:

THEOREM 35. *Let $G = (V, E_B, E_R, T, \nu)$ be a random-turn game with Richman function R. Then the following strategy is optimal for Blue (Red), and ensures that the game ends with probability one: Whenever the token is at a non-terminal vertex, if Blue (Red) wins the coin toss, he should move the token along an edge of steepest ascent (descent) that is part of a minimal-length path of steepest ascent (descent) to T. Moreover, the value of the game when started at $v \in V$ is $R(v)$.*

PROOF. We'll first note that the strategy is well-defined because by lemma 33 there is a path of steepest ascent (descent) from any vertex $v \in V \setminus T$ to T, and hence there is such a shortest such path. Let $S : V \setminus T \to V$ be the strategy.

It's enough to prove that if Blue plays according to the given strategy, then from any vertex $v \in V$ reached during game play the game will end with probability one and Blue's expected payoff is at least $R(v)$. The same proof with Red in place of Blue will prove that if Red uses the above strategy his expected payoff is no more than $R(v)$. Thus $R(v)$ is the value of the game, and the strategies are optimal.

We'll first prove that the game ends with probability one: Let $N = \max_{v \in V} d_{G_v^+}(v, T)$ (where $d_{G_v^+}(v, T)$ is the length of the shortest path of steepest ascent from v to T). If at any point of play Blue wins N consecutive coin tosses, his play will result in the game terminating, as moving the token along a shortest path of steepest ascent from any vertex (as Blue will do when playing S) will result in the token reaching T in at most N turns. Since as play progresses Blue will almost surely win N consecutive coin tosses, play will terminate with probability 1.

In order to prove that the expected payoff for Blue when playing according to the above strategy from any $v \in V$ is at least $R(v)$, we'll use the notion of a *truncated game*: The n-truncated game for G is the random-turn game played on the underlying graph of G, the only difference being that if T is not reached within n turns, the game ends and the payoff to Blue is r_{min}. Any strategy for G induces a strategy for the truncated game in the obvious way, and it's clear that if the strategy for G ensures the game ends with probability 1 then the expected payoff for the truncated game is less than the expected payoff of the full game. We'll define a sequence of functions $f_n : V \to \mathbb{R}$ s.t. $f_n(v)$ is the expected payoff of the n-truncated game starting at v if Blue plays S and Red plays optimally, as follows:

$$f_0(v) = \begin{cases} \nu(v) & v \in T \\ r_{min} & v \notin T \end{cases}$$

and for all $n \geq 0$:

$$f_{n+1}(v) = \begin{cases} \nu(v) & v \in T \\ \frac{1}{2}\left(f_n(S(v)) + f_n^-(v)\right) & v \notin T \end{cases}$$

Since after a single move has been made in the $n+1$-truncated game the remaining play constitutes an n-truncated game, f_n is indeed the expected payoff for Blue playing S and Red playing optimally. f_n is bounded and increasing and therefore converges to a function f satisfying:

$$f(v) = \begin{cases} \nu(v) & v \in T \\ \frac{1}{2}\left(f(S(v)) + f^-(v)\right) & v \notin T \end{cases}$$

Clearly R satisfies these conditions, so if we can show that there is only one such function, we may conclude that $f = R$. We won't go into the details, but a technique similar to that used in the proof of proposition 34 works here as well.

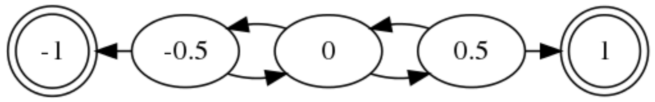

FIGURE 3.2.1. A game of Tug of War with $n = 3$. The Richman values are written inside the vertices. The terminal vertices are marked with a double circle.

We've shown that $\forall v \in V, R(v) = \lim_{n\to\infty} f_n(v)$. Now assume that in the full random-turn game, Blue is playing S and Red is playing some strategy T. Then for every n $\nu_{S,T}(v) \geq f_n(v)$, and so $\nu_{S,T}(v) \geq \lim_{n\to\infty} f_n(v) = R(v)$, as desired. $\square$

REMARK. In [**10**], an alternate approach is provided to proving uniqueness of the Richman function for win-or-lose games: First, the analog of theorem35 is established for *any* Richman function (similarly to theorem 38). Assuming, for a contradiction, that two different Richman functions exist for a given game, this would mean that there are two contradicting values to the game. Hence the Richman function is unique.

It is not clear to us whether this approach can be adapted to work in the case of multi-valued games as well. In order to prove the analog of theorem 35 without using the uniqueness of Richman functions, the authors consider Richman games rather than random-turn games. Richman games do not generalize as nicely to the multi-valued case (see section 4.2), and we do not know whether the reliance on Richman games can be replaced with random-turn games.

COROLLARY 36. *If G is a win-or-lose game ($\nu(T) = \{-1, 1\}$), then the probability that Blue wins an optimally-played game of random-turn G starting at $v \in V$ is $\frac{R(v)+1}{2}$.*

3.2.2. Optimal Play as a Random Walk.

Theorem 35 describes a class of optimal strategies for random-turn combinatorial games. If S and T are such strategies for the two players, we can define the optimal-play graph (V, E_{opt}), where E_{opt} is the set of all edges that may be taken during optimal play, i.e. for every $v \in V \setminus T$, $(v, S(v)), (v, T(v)) \in E_{opt}$. $G_{opt} = (V, E_{opt}, T, \nu)$ is a combinatorial game, and S and T are optimal strategies for its random-turn variant. If we start an optimally-played game of G at $v \in V$, the game is simply a random walk on G_{opt} until T is reached. This observation enables us to describe some optimally-played games.

EXAMPLE 37. **Tug of War:** Tug of War is played on a graph with $n+2$ vertices, arranged in a path (see figure 3.2.1). If the vertex set is $\{v_0, v_1, \ldots, v_n, v_{n+1}\}$ then $T = \{v_0, v_{n+1}\}$, $\nu(v_0) = -1, \nu(v_{n+1}) = 1$, and for every $1 \leq i \leq n$,$(v_i, v_{i-1}), (v_i, v_{i+1}) \in E = E_B = E_R$. The Richman values $R(v_i)$ form an arithmetic progression: $R(v_i) = \frac{2i}{n+1} - 1$. Blue's optimal strategy is to move the token to the right whenever possible, and Red's to move it to the left. Since this is a win-or-lose game, the probability that Blue wins when play starts at v_i is $\frac{R(v_i)+1}{2} = \frac{i}{n+1}$. An optimal play of Tug of War looks like a random walk on the path of vertices.

3.2.3. Richman Functions for Selection Games.

It's interesting to consider Richman functions for selection games played on a function f, and the relationship of the Richman function to f itself. Here the nodes of the graph are the partial assignments of variables, with outgoing edges from each partial assignment x_J (which hasn't determined f) to partial assignments that extend x_J by one variable (a blue edge for an assignment of 1, and a Red edge for an assignment of -1). Since the Richman value of a node is the expected value of the game under optimal play starting from the given node, and we know that for selection games this is equal to the expected value of the function when the undetermined variables are assigned uniformly and independently, we have, for any partial assignment x_J, $R(x_J) = \widehat{f|_{x_J}}(\emptyset)$. For any $i \in [n] \setminus J$, this implies that $\widehat{f|_{x_J}}(i) = \frac{1}{2}\left(R\left(x_J^{+i}\right) - R\left(x_J^{-i}\right)\right)$.

CHAPTER 4

Richman Games

Random-turn games are formed by taking some combinatorial game and changing the way turns are allocated. Richman games, named for their inventor David Richman and first presented in [**10**], are another way of doing so. As with random-turn games, there is some underlying combinatorial game. At the start of the game, Blue and Red are allocated some amount of money (which may not be the same for both of them). At each turn, the two players bid for the right to decide which of them will take the turn. The higher bidder pays the amount of his bid to his opponent, and the player designated to make the next move decides how the token should be moved. If the bids are equal some tie-breaking mechanism is used to decide who is considered the auction winner (the player awarded the win must still pay his bid to his opponent). For simplicity's sake we'll assume for the time being that all ties are awarded to Blue, although we'll consider other mechanisms as well. The game ends when the token is moved to a terminal node, with the associated payoffs going to each of the players. At this point the balance of each player's bank account, used for "buying" turns, is discarded (i.e. this money has no value once the game is over).

In this chapter we will present some results relating to Richman games, mainly from [**10**] and [**9**], and discuss a surprising relationship between Richman games and Random-turn games.

Remark. In [**10**] and [**9**] it is assumed that $E_B = E_R$. In keeping with this work's basic model, we do not require this. However, colored edges introduce the possibility that neither player desires the next turn (consider the game where the first person to move loses). Optimal play in this case (which dictates that both players bid zero) depends heavily on the tie-breaking mechanism. In order to avoid this complication, we allow the winner of the auction to force his opponent to take the next turn, following the suggestion of [**4**]. Thus, it is never a disadvantage to win the auction. This is in contrast to Richman games as presented in the aforementioned papers, which require the winner of the auction to take the next turn. With this modification to the rules, the results carry over to the colored-edge case.

4.1. Win-or-Lose Richman Games

In [**10**] and [**9**] the discussion is limited to win-or-lose games. The following summarizes the results therein.

In the following, let $G = (V, E_B, E_R, T, \nu)$ be a win-or-lose game. By this we mean that ν is limited to two values, which WLG are assumed to be $\{0, 1\}$ (in this we break from our usual custom of allowing Boolean variables to take on the values $\{-1,1\}$. The reason for this will become apparent when we discuss winning strategies). We also assume WLG that the total money supply is 1.

As might be expected from the name, the Richman function is key in Richman games. Let $R : V \to \mathbb{R}$ be the Richman function for G, and let $\{R_n\}_{n\in\mathbb{N}}$ be the functions from the proof of proposition 31. In this setting the Richman value of a vertex $v \in V$ represents the *critical ratio* of the money that Red must have in order to force a win from v, as formalized in the following theorem (theorem 2.2 and corollary 2.5 in [**10**]):

THEOREM 38. *For any vertex $v \in V$ Blue has a winning strategy for the Richman game played from v if Red's share of the money is strictly less than $R(v)$. Furthermore, if Red's share is strictly less than $R_n(v)$ Blue has a strategy that will guarantee a win within n turns (and the strategy in this case is made explicit in the proof).*

Similarly, Red has a winning strategy if his share of the money strictly exceeds $R(v)$, and he can force a win within n moves if his share of the money strictly exceeds $2R(v) - R_n(v)$.

PROOF. We'll prove the claim only for Blue. The claim for Red is deduced by reversing the roles of Blue and Red, and considering the game $G' = (V, E_R, E_B, T, 1-\nu)$, for which it can be verified that $R' = 1 - R$ is the unique Richman function and the functions $R'_n = 1 - 2R + R_n$ take the place of the functions R_n.

We'll prove the second part of the claim by induction on n. If $n = 0$, then Red's share is strictly less than $R_0(v)$ only if $v \in T$ and $\nu(v) = 1$. In this case Blue has already won, and the claim holds.

Now assume the claim is true for n, and that with the token placed at v Red has strictly less than $R_{n+1}(v)$. If $v \in T$ this means $\nu(v) = 1$ and Blue has already won, so the claim holds. Otherwise Let $R < R_{n+1}(v)$ be Red's share. Blue should bid $|R_n^+(v) - R_{n+1}(v)|$ (the assumption that $R < R_{n+1}(v)$ guarantees that this is a valid bid). If Blue wins the auction and $R_n^+(v) \geq R_n^-(v)$ he should move the token to a blue successor $u \in V$ of v s.t. $R_n(u) = R_n^+(v)$. If $R_n^+(v) < R_n^-(v)$ he should force Red to make the next turn. Either way Red's balance will have increased by $|R_n^+(v) - R_{n+1}(v)|$, and the token will be on a vertex u s.t. $R_n(u) \geq R_{n+1} + |R_n^+(v) - R_{n+1}(v)| > R + |R_n^+(v) - R_{n+1}(v)|$. Hence, by the induction hypothesis, Blue can win within n turns. If Red wins the auction, Red's balance decreases by at least $|R_n^+(v) - R_{n+1}(v)|$. If Red forces Blue to move, he should move the token to a blue successor $u \in V$ s.t. $R_n(u) = R_n^+(v)$. Then, similar to the previous case, regardless of which player moves the token it will be at a vertex u s.t. $R_n(u)$ is greater than Red's share of the money, and the induction hypothesis applies. This completes the proof of the second part of the theorem.

Now, assume Red begins the game with a share strictly less than $R(v)$. $R(v) = \lim_{n\to\infty} R_n(v)$ hence there is some n s.t. Red's share is strictly less than $R_n(v)$, so the second part of the theorem applies and Blue has a winning strategy. $\square$

REMARK 39. Note that theorem 38 holds regardless of the tie-breaking method used when the bids are equal. This mechanism becomes relevant only in the critical case, where Red's share of the money is exactly $R(v)$, and $0 < R(v) < 1$. In this case theorem 38 dictates that bidding more than $R^+(v) - R(v)$ will result in one's opponent gaining a winning position, thus optimal play dictates that both sides bid $R^{+}(v) - R(v)$. The analysis of optimal play thus depends on the tie breaking mechanism employed:

- If all ties are awarded to Blue (as in [**7**]), Blue has a winning strategy for all $v \in V$ s.t. $R(v) > 0$ (that is, iff there is a path from v to some $t \in T$ with $\nu(t) = 1$).
- If ties are decided by the toss of a coin (as in [**10**]), the game reduces to the random-turn game. If the players play according to theorem 35 then the expected value of the game is $R(v)$.
- If ties are decided by alternating turns between Blue and Red (as in [**9**]), the game reduces to the alternating-turn game.
- With more exotic tie-breaking methods (such as arbitrary mappings from the set of vertices to the winner in case of a tie bid), the game may not have a value (see example in [**7**]).

The connection between random-turn and Richman games is especially strong in the case of win-or-lose games. In this case the Richman function represents both the critical ratio of the budget that Red must possess to force a win in the Richman game (theorem 38), and the probability of a Blue win in the random-turn game (because the payoffs are limited to $\{0,1\}$, the value of the game is equal to the probability of Blue winning under optimal play. Thus by theorem 35 the Richman function is the probability of a Blue win).

4.2. Multi-Valued Richman Games

In this section we consider Richman games with more than two outcomes. Let $G = (V, E_B, E_B, T, \nu)$, and let $\nu(T) = \{r_{min} = r_1 < \ldots < r_n = r_{max}\}$. In this case the random-turn game is governed by the Richman function of section 3.2.1. However, unlike with win-or-lose Richman games, multi-valued Richman games are analyzed differently: Using the technique of theorem 38, Blue finds the largest i s.t. he can ensure the payoff to be at least r_i. He then plays according to a strategy that ensures this. This takes the following formal form:

For each $k = 1, \ldots n$, we'll define the game $G_k = (V, E_B, E_R, T, \nu_k)$ where $\nu_k(v) = \begin{cases} 1 & \nu(v) \geq r_k \\ 0 & otherwise \end{cases}$. Let $\{R_k\}_{k=1}^{n}$ be the matching Richman functions. Clearly $1 \equiv R_1 \geq R_2 \geq \ldots \geq R_n$. Assume, as before, that the total money supply is 1, and assume that R is Red's share of the money. Assume the game starts at $v \in V$, and let k be the largest integer s.t. $R_k(v) \geq R$. If $R_k(v) > R$, then by the discussion in the previous section the value of the game is r_k. If $R_k(v) = R$ then the expected outcome of the game played under the tie-breaking procedure must be weighed against the fact that there may be some $l < k$ s.t. $R_l(v) > R_k(v) = R$ in which case Blue can force the outcome to be at least r_l.

The connection of this analysis to the Richman function of section 3.2.1 isn't as strong as in the win-or-lose case. It appears the Richman function of section 3.2.1 doesn't contain all the information necessary to decide on optimal play of G as a Richman game. The fundamental difference between Richman games and random-turn games is that as deterministic games, optimal play of Richman games depends only on the topology of the graph w.r.t. the terminal nodes, but does not depend on the actual values of the terminal nodes. In random-turn games optimal play takes into account the value of the payoffs as well. For win-or-lose games the two coincide but for multi-valued games optimal play may be quite different.

CHAPTER 5

Algorithms and Complexity

The preceding chapters gave various characterizations and descriptions of optimal play of random-turn and Richman games. If one desires to actually play a game well, however, these characterizations are not enough. One must have an efficient way to compute optimal strategies. This chapter considers algorithms achieving this as well as their complexity.

In section 5.2 we consider a separate algorithmic question, that of whether playing a random-turn game is an efficient way to compute a Boolean function.

5.1. Algorithms for Determining Optimal Moves

5.1.1. Selection Games. Theorem 22 tells us that finding the optimal move in a random-turn selection game is equivalent to finding the variable that maximizes its degree-one Fourier coefficient. Thus, it's sufficient to know the values of the degree-one coefficients. This fact can be leveraged into various algorithms for choosing optimal moves. In general, direct calculation of the values $\hat{f}(i)$ for an n variable selection game requires evaluating f on all 2^n possible inputs. Thus the naive approach of direct calculation is computationally quite infeasible. For some games it's possible to determine the variable with maximal influence in time polynomial in n. For instance, this is the case for generalized tribes functions: Claim 28 shows that a variable is an optimal selection iff it's a member of an undetermined tribe with the smallest number of undetermined variables. This property can be tested in time that is polynomial (in fact, linear) in n. While the ability to play specific games optimally is nice, the ability to play *any* selection game well is nicer. We'll show that for all win-or-lose selection games a simple probabilistic algorithm exists to find a near-optimal variable in polynomial time (as shown in [**18**]).

5.1.1.1. *Probabilistic Algorithm for Finding Near-Optimal Selection.* As observed in [**18**], a simple and efficient way to evaluate the degree-one Fourier coefficients for win-or-lose games is to randomly sample elements of $\{-1,1\}^n$. This is because, by definition, $\hat{f}(i) = \mathbb{E}_{x\sim\{-1,1\}^n}[f(x)\,x_i]$. $f(x)\,x_i$ can be regarded as the result of a Bernoulli trial, so we can use a Hoeffding bound in order to relate the number of independent samples taken to the probability of deviation from the expectation. Specifically, if we let S^i_m denote the average of m independent samplings of $f(x)\,x_i$, then $Pr\left[\left|S^i_m - \hat{f}(i)\right| > \varepsilon\right] \leq 2\exp\left(-\frac{1}{2}\varepsilon^2 m\right)$. Thus, $\frac{2}{\varepsilon^2}\ln\left(\frac{2}{\delta}\right)$ independent samplings for each variable are sufficient to ensure that except with probability δ, the i s.t. S^i_m is maximal is within ε of being optimal. Thus a near-optimal selection can be found with $n\frac{2}{\varepsilon^2}\ln\left(\frac{2}{\delta}\right)$ steps. Replacing ε and δ with $\frac{\varepsilon}{n}$, we get that after $\frac{2n^3}{\varepsilon^2}\ln\left(\frac{2n}{\varepsilon}\right)$ samplings, except with probability $\frac{\varepsilon}{n}$ the selection is within $\frac{\varepsilon}{n}$ of being optimal. Against an optimal opponent, playing this strategy wins with probability at least $p-\varepsilon$, p being the probability of a win under random play.

We generalize this to the multi-valued case: Let $M = \max_{x \in \{-1,1\}^n} |f(x)|$. Then, again using the Hoeffding bound, $Pr\left[\left|S_m^i - \hat{f}(i)\right| > \varepsilon\right] \le 2\exp\left(-\frac{m\varepsilon^2}{M}\right)$. Thus, $\frac{M}{\varepsilon^2}\ln\left(\frac{2}{\delta}\right)$ samplings for each variable are sufficient to ensure that except with probability δ, the i s.t. S_m^i is maximal is within ε of being optimal. Thus a near-optimal selection can be found with $n\frac{M}{\varepsilon^2}\ln\left(\frac{2}{\delta}\right)$ steps. Replacing ε and δ with $\frac{\varepsilon}{n}$, we get that after $\frac{Mn^3}{\varepsilon^2}\ln\left(\frac{2n}{\varepsilon}\right)$ samplings, except with probability $\frac{\varepsilon}{n}$ the selection is within $\frac{\varepsilon}{n}$ of being optimal. Against an optimal opponent, playing this strategy has expected outcome of at least $\mathbb{E}[f] - \varepsilon$.

5.1.2. Evaluating Richman Functions. We don't know, in general, the difficulty of finding the Richman function R of a game $G = (V, E_B, E_R, T, \nu)$. In this section we'll introduce and discuss several algorithms for this problem that were first described in [**9**]. In [**9**] the algorithms were presented for win-or-lose games. We generalize them to multi-valued games. We'll begin with cases for which efficient algorithms are known, and then introduce a general algorithm, whose exact complexity we don't know.

5.1.2.1. *Directed Acyclic Graphs.* If the graph $(V, E_B \cup E_R)$ is a DAG, the values R can be calculated by backwards induction on the graph: The leaves of the graph are exactly the members of T, so their Richman values are known. At each step find a vertex $v \in V$ s.t. all its successors' Richman values are known. Set $R(v) = \frac{1}{2}\left(R^+(v) + R^-(v)\right)$.

The running time of this algorithm is polynomial in the size of G, so it may be considered efficient. However, in many cases G itself is very large, and so in practice the algorithm may not be helpful. For example, every selection game can be represented as a DAG, yet there may be as many as 3^n nodes. Most interesting functions have a much more compact representation than an exponentially long list, so this algorithm may not be of much use.

5.1.2.2. *Graphs with Small Outdegree and Uncolored Edges.* Another special case is when every vertex has outdegree two or less, and $E_B = E_R$. In this case, for every $v \in V \setminus T$, let v_1, v_2 be the (possibly non-distinct) successors of v. Then, as we will show in claim 41, R is the unique solution of the linear equation

$$\forall v \in V, R(v) = \begin{cases} \frac{1}{2}\left(R(v_1) + R(v_2)\right) & v \notin T \\ \nu(v) & v \in T \end{cases}$$

Thus R may be found by solving the linear system, which may be done in time polynomial in $|V|$.

Note, however, that unlike most of our results, which generalize to the biased-coin case, this approach only works if the probability of each player winning the next turn is $\frac{1}{2}$.

5.1.2.3. *Undirected Graphs.* One last case for which an efficient algorithm to compute R is known is that of undirected graphs, that is where $E = E_B = E_R$, and for every $v, w \in V \setminus T$, $(v,w) \in E \iff (w,v) \in E$. This algorithm is defined, and its correctness and runtime proved, in theorem 12 in [**9**]. Although it is stated there for win-or-lose games only, the construction and proof carry over to the general case without change. We'll present here only the algorithm, and refer the interested reader to the original paper for the proofs.

We'll need the concept of a partial Richman function (PRF): A PRF R' for G is a subgraph $(V', E') \subseteq (V, E)$ s.t. $T \subseteq V$, E' is a set of undirected edges, and

R' is a Richman function for (V', E', T, ν). Our algorithm will build a sequence of PRFs with increasing domain s.t. once a value is assigned to a vertex it does not change in successive iterations. The final PRF in the sequence will be augmented to a Richman function for G. The algorithm is as follows:

(1) Set $(V', E') = (T, \emptyset)$, and $R' = \nu$. (This constitutes a PRF).
(2) If possible, find a path $v_0, v_1, \ldots, v_n$ in G with the following qualities:
   - $v_0, v_n \in V'$
   - $v_1, \ldots, v_{n-1} \notin V'$
   - $v_0 \neq v_n$
   - Among all paths satisfying the three preceding conditions, this path has the highest average slope, i.e. $\frac{R'(v_n) - R'(v_0)}{n}$ is maximal.
(3) If such a path is found, set $V' = V' \cup \{v_1, \ldots v_{n-1}\}$, $E' = E' \cup \{v_0 v_1, v_1 v_2, \ldots, v_{n-1} v_n\}$, and for every $v_i, 1 \leq i \leq n-1$ set $R'(v_i) = \frac{R'(v_n) - R'(v_0)}{n} i + R'(v_0)$. Continue from step 2.
(4) If no such path is found this means (see proof in **[9]**) that every vertex in $v \in V \setminus V'$ is connected by a path not in E' to a single vertex $w \in V'$. Set $R'(v) = R'(w)$.

R' constructed as above is a Richman function for G.

5.1.2.4. *Linear Programming Solution for General Games.* In the following, for every $v \in V \setminus T$, let v^+ be a blue successor of v s.t. $R(v^+) = R^+(v)$, and let v^- be a red successor s.t. $R(v^-) = R^-(v)$.

The uniqueness of the Richman function (proposition 34), together with the fact that R is defined by means of a system of linear equations and inequalities (definition 30), suggest finding Richman functions by means of solving the linear program given by the equations and inequalities. The problem with this approach is that the equations themselves, at the outset, may not be known: In general we can't know which children of a vertex $v \in V \setminus T$ maximize and minimize R. In some cases it isn't difficult to ascertain this. Notably, if the outdegree of all vertices is less than or equal to two, v^+ and v^- are simply the two (not necessarily distinct) successors of v. In this case there is a unique solution to the linear equations (see section 5.1.2.2).

In the general case, we must first discover the identity of $v^+$and v^-. One simple, if inefficient, way to do this is to guess: For each $v \in V \setminus T$, we can arbitrarily choose successors of each color and call them v^+ and v^-. Then attempt to solve the linear program. If there is a solution, it is the unique Richman function. By iterating through all possible choices, eventually a correct one will be found and the algorithm will have discovered the Richman function.

A different approach is to use the functions R_n (from the proof of proposition 31), as follows: Iteratively calculate R_n, beginning with R_0. For each vertex $v \in V \setminus T$ let v_n^+, v_n^- be appropriately colored successors of v that achieve $R_n^+(v), R_n^-(v)$, respectively. Attempt to solve the linear program:

$$R(v) = \begin{cases} \frac{1}{2}\left(R(v_n^+) + R(v_n^-)\right) & v \in V \setminus T \\ R(v) & v \in T \end{cases}$$

With the constraints:

$$\forall w : (v, w) \in E_B, R(w) \leq R\left(v_n^+\right)$$

$$\forall w : (v, w) \in E_R, R(w) \geq R\left(v_n^-\right)$$

If there is a solution, output it and halt. By uniqueness of the Richman function it's equal to R. If no solution is found, repeat the process.

We'll first note that this algorithm does always find R: Since R_n converges to R, at some point the ordering of the values $\{R_n(v)\}_{v\in V}$ will coincide with the (weak) ordering of the values $\{R(v)\}_{v\in V}$, at which point R will be found as the unique solution to the linear program.

The efficiency of this algorithm depends on how quickly the values $\{R_n(v)\}_{v\in V}$ achieve their final ordering. This depends on the speed of convergence of R_n to R as well as the minimal difference between the values $\{R(v)\}_{v\in V}$. In the general case where there may be more than two possible payoffs, some of which may be irrational, we do not know of a good bound for the differences between Richman values. For the case where the payoffs are rational, we'll bound these in the following claims:

Claim 40. Let $N = |V| - |T|$. Then for every $n \in \mathbb{N}, v \in V$, $|R(v) - R_n(v)| \leq (r_{max} - r_{min})\left(1-2^{-N}\right)^{\left\lfloor \frac{n}{N}\right\rfloor}$.

Proof. Recall S (defined in theorem 38), which is the optimal strategy for Blue. S is a function from $V\backslash T$ to V. In order to ease notation, in the following we'll extend S's domain to all of V by setting, for each $v \in T$, $S(v) = v$. Further, for any $f : V \to \mathbb{R}$, recall that $f^+(v), f^-(v)$ were defined only for $v \in V \setminus T$. We'll extend this notation to all of V as well, by setting, for any $v \in T$, $f^+(v) = f^-(v) = f(v)$.

Note that R_n is the expected value of the n-truncated game when both players play optimally. f_n (from the proof of theorem 35) is the value of the n-truncated game when Blue is playing S and Red is playing optimally against S. Hence $R(v) \geq R_n(v) \geq f_n(v)$. We'll show by induction that $R(v) - f_n(v)$ satisfies the above inequality, and so it holds for $R(v) - R_n(v)$ as well:

First, for any $n \leq N$, $R(v) - f_n(v) \leq r_{max} - r_{min}$, so the bound holds. Now let $n > N$. Note that if Blue wins N coin tosses in a row, the game surely ends, i.e. $S^N(v) \in T$ for any $v \in V$. Hence, no more than N coin tosses will occur without either the game ending or Red winning one of the tosses. Using the law of complete probability, we can rewrite the expected outcome of the game:

$$R(v) = \sum_{m=1}^{N} 2^{-m} R^-\left(S^{m-1}(v)\right) + 2^{-N} R\left(S^N(v)\right)$$

$$f_n(v) = \sum_{m=1}^{N} 2^{-m} f^-_{n-m+1}\left(S^{m-1}(v)\right) + 2^{-N} f_{n-N}\left(S^N(v)\right)$$

Since $S^N(v) \in T$, we can rewrite the last equality as:

$$f_n(v) = \sum_{m=1}^{N} 2^{-m} f^-_{n-m+1}\left(S^{m-1}(v)\right) + 2^{-N} R\left(S^N(v)\right)$$

By using the induction hypothesis, we can write:

$$f_n(v) \geq \sum_{m=1}^{N} 2^{-m}\left(R^-\left(S^{m-1}(v)\right) - (r_{max} - r_{min})\left(1-2^{-N}\right)^{\left\lfloor \frac{n-m}{N}\right\rfloor}\right) + 2^{-N} R\left(S^N(v)\right)$$

Hence:

$$f_n(v) \geq R(v) - (r_{max} - r_{min})\left(1-2^{-N}\right)^{\left\lfloor \frac{n-N}{N}\right\rfloor}\left(1-2^{-N}\right) = R(v) - (r_{max} - r_{min})\left(1-2^{-N}\right)^{\left\lfloor \frac{n}{N}\right\rfloor}$$

which gives the desired inequality. □

The next step is to bound the differences between the values $\{R(v)\}_{v \in V}$. To do this we'll need the following notation and two claims:

Recall the graph of optimal play $G_{opt} = (V, E_{opt})$, where $(v, w) \in E_{opt}$ iff $w = S(v)$ or $w = T(v)$ (with S and T being the optimal strategies for Blue and Red respectively, described in theorem 35). Let A be G_{opt}'s adjacency matrix.

Claim 41. R is the unique solution of the linear system:

$$\forall v \in V, R(v) = \begin{cases} \nu(v) & v \in T \\ \frac{1}{2}\left(R\left(v^{+}\right) + R\left(v^{-}\right)\right) & v \notin T \end{cases}$$

Note that proposition 34 shows that R is the unique Richman function. However, this does not preclude the possibility that the linear system has multiple solutions, only one of which satisfies the definition of a Richman function (in particular, perhaps $R(v^{+}) \neq R^{+}(v)$ for some solution to the linear system).

The fact that the linear system has a unique solution was used implicitly in **[9]**, but no proof was provided.

Proof. Since with optimal play G ends with probability one, G_{opt} has the property that every random walk (with the uniform distribution on the (at most two) outgoing edges of every node) almost surely reaches T. For every $n \in \mathbb{N}, v \in V$, the sum of the entries in the row corresponding to v in A^n is the number of walks of length n starting at v that do not reach T in $n-1$ steps or less. Let n be large enough such that with non-zero probability, a random walk on G_{opt} starting at any vertex reaches T in $n-1$ steps or less. Then the ℓ^1 norm of each row of A^n is strictly less than 2^n. Since the spectrum of a matrix is bounded by the maximal ℓ^1 norm of one of its rows, this implies that 2^n isn't an eigenvalue of A^n, and thus 2 isn't an eigenvalue of A.

Now, note that the system of linear equations can be written as $R = \frac{1}{2}AR + \nu$, which is equivalent to $(2I - A)R = 2\nu$. Since 2 isn't an eigenvalue of A, $2I - A$ is invertible and the system has a unique solution. □

Claim 42. If $\nu(T) \subseteq \mathbb{Z}$ then for every $u, v \in V$, if $R(u) \neq R(v)$ then $|R(u) - R(v)| \geq \left(2^{|T|}6^{\frac{N}{2}}\right)^{-1}$.

Proof. First note that any vertex with exactly one successor may be eliminated from the equation, as its value must equal its successor. Thus (once these vertices have been eliminated, repeating the process if necessary) , every row in A is either all 0s (if the row corresponds to a terminal node) or else all 0s except for two 1s.

Since R is the unique solution to the system $(2I - A)R = \nu$, the solution can be found by Cramer's rule. Hence:

$$|R(v)| = \frac{|\det(M_v)|}{|\det(2I - A)|}$$

Where M_v is $2I - A$ with ν having replaced v's column. The absolute value of the determinant of a matrix is bounded above by the product of the norms of its rows. In this case, $2I - A$ has $|T|$ rows of norm 2, and the remaining rows have norm $6^{\frac{1}{2}}$. Thus $|\det(2I - A)| \leq 2^{|T|}6^{\frac{N}{2}}$. M_v is an integer valued matrix. Hence $\det(M_v)$

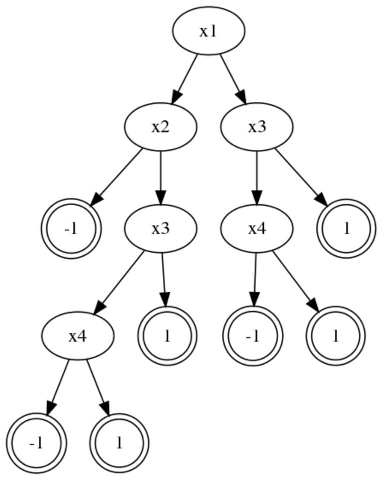

FIGURE 5.2.1. An influence-based construction of a decision tree for a level-2 And-Or function.

is a whole number. Therefore $R(v) - R(u)$ is a multiple of $|\det(2I - A)|^{-1}$, so if $R(u) \neq R(v)$, $|R(v) - R(u)| \geq \left(2^{|T|} 6^{\frac{N}{2}}\right)^{-1}$. □

COROLLARY 43. *If $\nu(T) \subseteq \mathbb{Q}$, let Q be the least positive integer s.t. $Q\nu(T) \subseteq \mathbb{Z}$. Then for every $u, v \in V$, if $R(u) \neq R(v)$, $|R(u) - R(v)| \geq \left(Q2^{|T|} 6^{\frac{N}{2}}\right)^{-1}$.*

PROOF. The previous claim holds for the vector QR, and the bound follows. □

Putting the above together, we can finally give a bound on the algorithm's runtime. If $n = O\left(2^N N\left(|V| + \log\left(\left(r_{max} - r_{min}\right) Q\right)\right)\right)$, the order of the values $\{R_n(v)\}_{v \in V}$ will coincide with the order of the values $\{R(v)\}_{v \in V}$. For the special case of win-or-lose games, we get the bound $O\left(2^{|V|} |V|^2\right)$.

Unfortunately, this bound is exponential. However in **[9]** the authors report that for win-or-lose games the algorithm does very well in practice. Note that the problem of finding a Richman function for a win-or-lose game is in NP. Thus it would be difficult to show that the bound of $O\left(2^{|V|} |V|^2\right)$ is tight. On the other hand we have not shown that the problem is NP-hard, and indeed it may be possible to find a polynomial-time algorithm.

5.2. Optimal Play of Random-Turn Selection Games as Decision Procedures

As laid out in section 3.1.1, for every selection game $f : \{-1, 1\}^n \to \mathbb{R}$ there exists a pure strategy S that is optimal for both players. S induces a decision tree for calculating f: At each stage, query $S(x_J)$ where x_J is the partial assignment of the variables queried so far. Repeat this until enough variables have been queried to determine the value of f. We'll call this the **influence-based construction** of a decision tree for f. Figure 5.2.1 illustrates this for a level 2 And-Or tree (see example 25 for the definition).

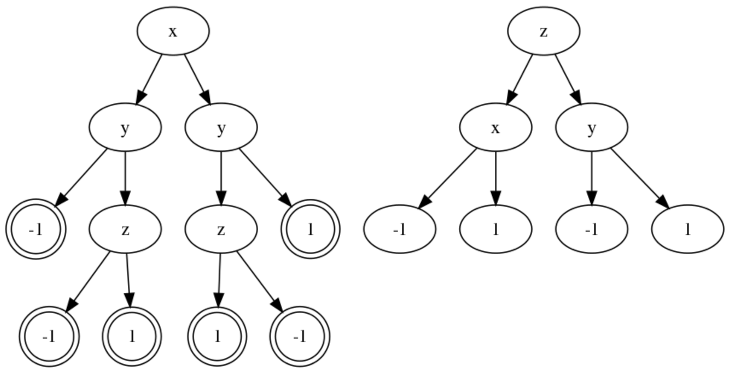

FIGURE 5.2.2. Two decision trees for the function $f(x,y,z) = \begin{cases} x & z=-1 \\ y & z=1 \end{cases}$. $\hat{f}(z) = 0$, $\hat{f}(x) = \hat{f}(y) = \frac{1}{2}$. At left is the influence-based construction of a decision tree for f with expected complexity $\frac{5}{2}$. At right is a more efficient decision tree, with expected complexity 2.

Let T be an influence-based construction of a decision tree for f. The question arises of of how the expected complexity of T compares with the expected decision tree complexity $\Delta(f)$ (see definition 19). Propositions 23 and 24 are, in effect, lower bounds on $\Delta(f)$. Example 27 demonstrates that for generalized tribes functions, T achieves $\Delta(f)$, and is therefore the most efficient decision tree (in terms of expected running time) calculating f.

A far-reaching conjecture would be that for any Boolean f, T is the most efficient decision tree computing f. This is certainly false for non-monotone functions: Consider the function $f(x,y,z) = \begin{cases} x & z=-1 \\ y & z=1 \end{cases}$. Figure 5.2.2 illustrates how the conjecture fails for f. This leaves open the possibility that for monotone functions, the decision tree induced by optimal play indeed achieves $\Delta(f)$. This is related to the question of whether the influence-based construction is the smallest (in terms of the number of leaves) decision tree for a monotone function. The latter was conjectured for win-or-lose functions in [**3**], where it was also reported, based on computer experiments, that this is indeed the case for all monotone functions of six variables or less. In [**18**], the question of whether the influence-based construction achieves the expected decision tree complexity for Hex and Recursive Ternary Majority (example 44) was left open.

Of these questions, we do not know the answer to the one regarding Hex. However it turns out that for level-2 Recursive Ternary Majority there is a more efficient decision tree than the influence-based construction, both in terms of expected number of input bits read as well as the number of leaves. This disproves the conjecture in [**3**], and answers the more general conjecture regarding the relation between influence-based constructions of decision trees for monotone functions and their expected decision tree complexity.

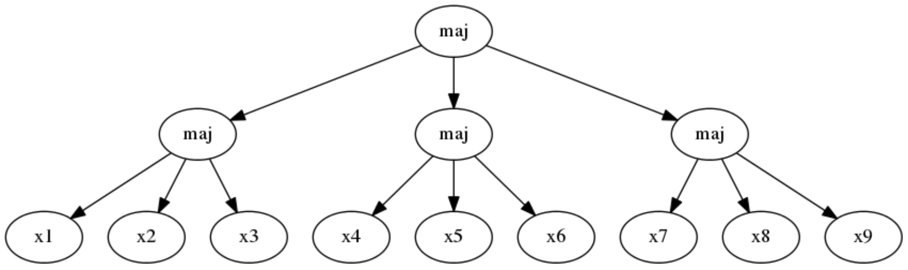

FIGURE 5.2.3. A level-2 Recursive Ternary Majority tree.

EXAMPLE 44. **Recursive Ternary Majority:** Let $maj : \{-1,1\}^3 \to \{-1,1\}$ be the function returning the value of the majority of its input bits. The level-n Recursive Ternary Majority function, $f_n : \{-1,1\}^{3^n} \to \{-1,1\}$, is defined recursively as follows:

$$f_0(x) = x$$

$$f_{n+1}(x) = maj\left(f_n(x_1,\ldots,x_{3^n}), f(x_{3^n+1},\ldots,x_{2\cdot 3^n}), f(x_{2\cdot 3^n+1},\ldots,x_{3^{n+1}})\right)$$

f_{n+1} can be thought of as a ternary tree, with the root labeled by the majority function, and each of the three children the root of an f_n tree. f_0 is a leaf labeled by a single variable (see figure 5.2.3).

These functions were given in [**19**] (where they were attributed to Ravi Boppana) as examples of functions for which the deterministic decision tree complexity is provably less than the randomized decision tree complexity. The exact randomized complexity is an open problem: the best known bounds are $O(2.64946^n)$ ([**12**]) and $\Omega(2.55^n)$ ([**11**]) for the level-n Recursive Ternary Majority function.

We'll use the bounds of propositions 23 and 24 to lower-bound $\Delta(f_n)$: The symmetries between input variables allow us to conclude that all variables have the same influence. We also observe that x_1 is pivotal in f_{n+1} iff it's pivotal in $f_n(x_1,\ldots,x_{3^n})$ and $f_n(x_{3^n+1},\ldots,x_{2\cdot 3^n}) \neq f_n(x_{2\cdot 3^n+1},\ldots,x_{3^{n+1}})$ (i.e. iff it's pivotal in the leftmost f_n child of the root and the other two children evaluate to different values). These events are independent, and the second occurs with probability $\frac{1}{2}$. As this is a monotone win-or-lose function proposition 18 applies, and $\widehat{f_n}(1)$ is equal to the probability of x_1 being pivotal. This yields the following recursion formula: $\widehat{f_{n+1}}(1) = \frac{1}{2}\widehat{f_n}(1)$. Since $\widehat{f_0}(1) = 1$, $\widehat{f_n}(1) = 2^{-n}$. The function is balanced, hence $Var[f_n] = 1$. There are 3^n variables, hence the lower bound on the expected number of input bits read from proposition 23 is $\left(\sum_{i=1}^{3^n} \widehat{f_n}(i)\right)^2 = \left(\frac{3}{2}\right)^{2n} = \left(\frac{9}{4}\right)^n$ and the bound from proposition 24 is 2^n.

Theorem 26 describes the recursive nature of optimal play of And-Or trees. If optimal play of Recursive Majority followed a pattern similar to optimal play on And-Or trees, in the sense that once a variable had been played in a subtree optimal play would remain in the subtree until it was determined, the expected number of turns in optimal play would be $\left(\frac{5}{2}\right)^n$: First the two leftmost trees would be evaluated, and if they disagree (an event that occurs with probability $\frac{1}{2}$) the third tree would be evaluated as well (this algorithm also demonstrates that $\left(\frac{5}{2}\right)^n$

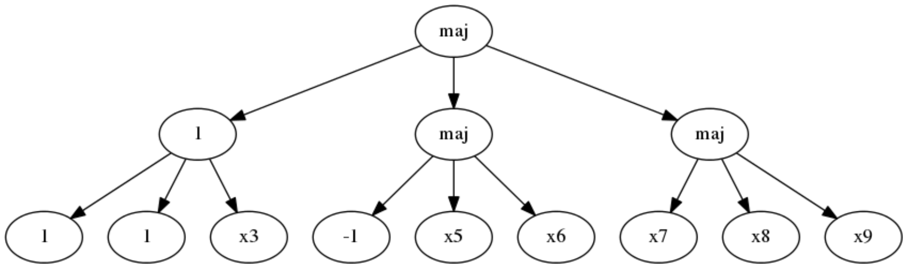

FIGURE 5.2.4. A partially played game of level-2 recursive ternary majority, with both players playing optimally. Blue won the first two coin tosses, and selected x_1 and x_2. This determined the left-most subtree. Red won the third toss and selected x_4. Let x_J be this partial assignment. The influences are now: $\widehat{f|_{x_J}}(5) = \widehat{f|_{x_J}}(6) = \frac{1}{4}$ and $\widehat{f|_{x_J}}(7) = \widehat{f|_{x_J}}(8) = \widehat{f|_{x_J}}(9) = \frac{3}{8}$. Hence the optimal next selection is one of x_7, x_8, x_9, none of which are in the second, partially determined, subtree. Thus optimal play doesn't necessarily follow a recursive pattern similar to And-Or trees.

is an upper bound on the expected decision complexity of f_n). However, this is not the case, as figure 5.2.4 illustrates.

Regarding the efficiency of the influence based construction, it's clear that the expected deterministic complexity of $f_1 = maj$ is 2.5, which is also the complexity of the influence-based construction. We used a computer to construct both the influence based construction and the optimal decision tree for f_2 (these are given in appendix A). The influence-based construction reads $\frac{396}{64}$ input bits on average whereas the optimal construction reads only $\frac{393}{64}$. Furthermore, the influence-based construction has 120 leaves in the decision tree, whereas the optimal construction (in terms of expected bits read) has only 116. This shows that the influence-based construction of a decision tree need not be optimal, either in expected number of bits read or in the size of the tree.

We can't say much about the expected complexity of the influence-based construction for f_n. We note, however, that the fact that the optimal construction for f_2 is expected to read $\frac{393}{64}$ bits on average gives an upper bound of $O\left(\left(\frac{393}{64}\right)^{\frac{n}{2}}\right)$ ($\sqrt{\frac{393}{64}} \approx 2.47 < \frac{5}{2}$ so this is an improvement on the "simple" recursion described above), with the following algorithm: If n is even recursively evaluate the subtrees of depth two in the order dictated by the optimal tree for f_2. The expected number of subtrees evaluated is $\frac{393}{64}$, and each one requires reading $O\left(\left(\frac{393}{64}\right)^{\frac{n-2}{2}}\right)$ bits on average. Thus the expected number of bits read is $O\left(\left(\frac{393}{64}\right)^{\frac{n}{2}}\right)$. If n is odd the bound may be achieved by evaluating all three subtrees (noting that the depth one subtrees are of even height), which achieves the same asymptotic bound.

To summarize, we've shown a lower bound of 2.25^n and an upper bound of $\approx 2.47^n$ on the expected number of queries required to evaluate f_n, n even, with the uniform distribution on inputs. For n odd, the upper bound should be replaced with $\approx 3\cdot 2.47^{n-1}$. It would be interesting to know what the expected deterministic complexity of f_n is exactly.

CHAPTER 6

Future Work

The foregoing work opens several doors for consideration in the future. One avenue is to consider other variations on the turn-allocation mechanism. For example one may consider the Poorman variant of Richman games, where the highest bidder pays his bid to the bank. Many such variants, including Poorman games, have already been discussed in [**9**]. For many of these the analysis is quite similar to Richman games, although some variants (for instance, allowing the players to place negative bids) may result in drastically different game play. It would be interesting to know if new variants exist, that are either interesting in their own right as games or else whose analysis proves interesting.

Another possibility is to relax the definition of a combinatorial game: In particular, we did not allow probability to play a part in the definition of a combinatorial game. We may generalize the definition of a combinatorial game and associate with each non-terminal vertex a probability distribution on the subsets of outgoing edges. Thus, once a player has won the right to move, the moves available to him are chosen according to the probability distribution on the current vertex. In this way, we may analyze (for example) random-turn Backgammon.

We may also consider different notions of winning. For example, in [**13**] the author considers infinite turn-based games played on Büchi automata (which are, in particular, finite graphs). Blue wins iff the infinite play is accepted by the automaton. It would be interesting to analyze random-turn and Richman variations on such games.

Can we say more about the expected length of random-turn selection games? This can be asked both about specific games such as Hex and Recursive Ternary Majority, and as a general question: We've shown that the influence based construction of a decision tree is not necessarily optimal, either in expected number of queries or in space. What more can be said about such constructions?

Finally, and perhaps most interestingly, there are the algorithmic questions associated with optimal play: We have not been able to show an efficient algorithm for finding the Richman function of a general graph. On the other hand, this problem is in NP. Is it in fact NP-complete?

APPENDIX A

Influence-Based and Optimal Decision Trees for Level-2 Recursive Ternary Majority

This appendix supplements example 44 and demonstrates that the influence-based decision tree for level-2 recursive ternary majority is less efficient, both in terms of space and in terms of expected number of bits read, than the optimal decision tree. To this end, we display two diagrams: figure A.0.1 shows the influence-based decision tree, while figure A.0.2 shows a decision tree that is more efficient both in both of the above senses (it is, in fact, optimal in terms of expected number of bits read, but we will not prove this).

The level-2 recursive ternary majority function has nine input bits, resulting in a fairly large tree. To reduce the size of the diagrams, we take advantage of the symmetries between Blue and Red, as well as those between the input bits, and assume WLG that Red has won the first coin toss and selected x_1. Thus, the diagrams actually show decision trees for the function $f(x_2, \ldots, x_n) = f_2(-1, x_2, \ldots, x_9)$. In effect, we are showing non-optimality of the influence based construction for a function of eight variables. Internal nodes in the decision tree are marked with a single circle, with the variable to be queried written inside. Leaves are marked with a double circle, with the identity of the winner (1 for Blue, -1 for Red) written inside.

As may be verified directly, the expected number of input bits read in the influence-based construction is $\frac{396}{64}$, whereas in the second construction it is $\frac{393}{64}$, showing non-optimality in terms of expected number of bits read. With regard to number of leaves, the influence-based tree has 120 leaves, whereas the second tree has 116 leaves.

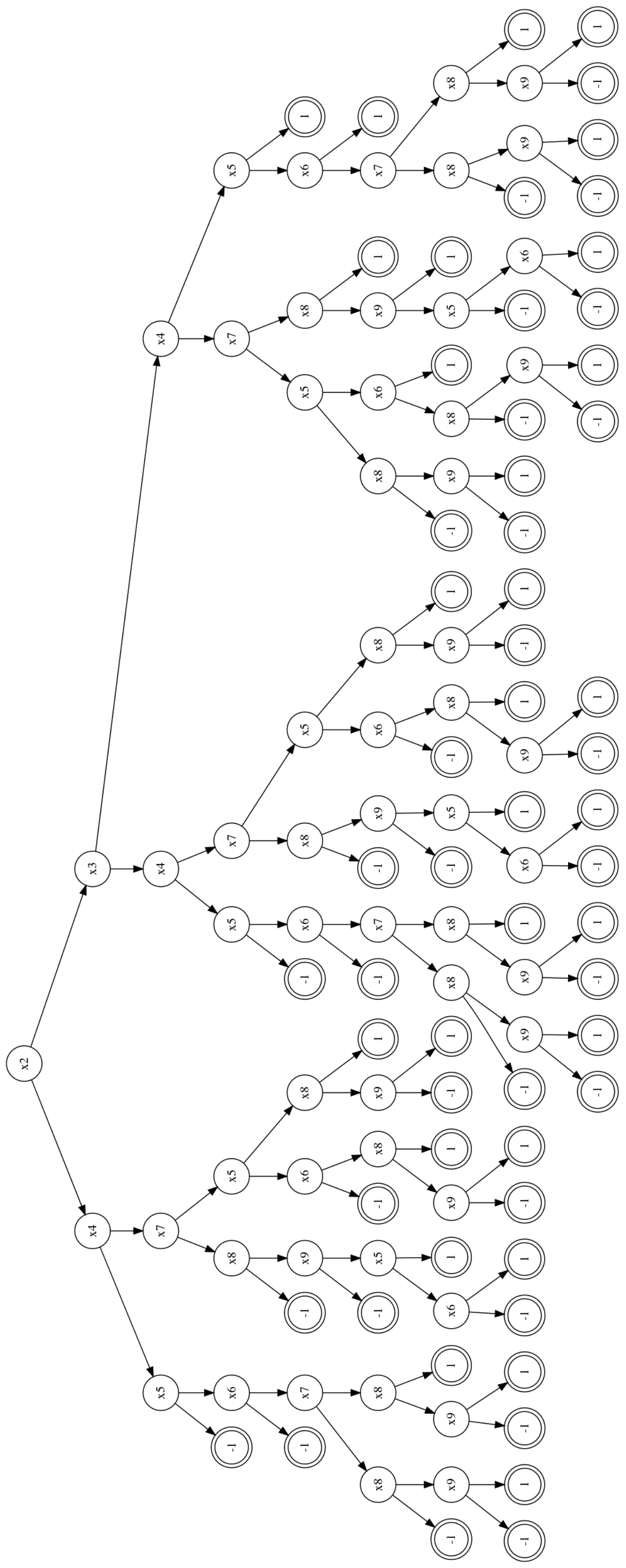

FIGURE A.0.1. Influence based construction of a decision tree for the function $f\left(x_2, \ldots, x_n\right) = f_2\left(-1, x_2, \ldots, x_9\right)$. The expected number of bits read is $\frac{332}{64}$, and the number of leaves is 60.

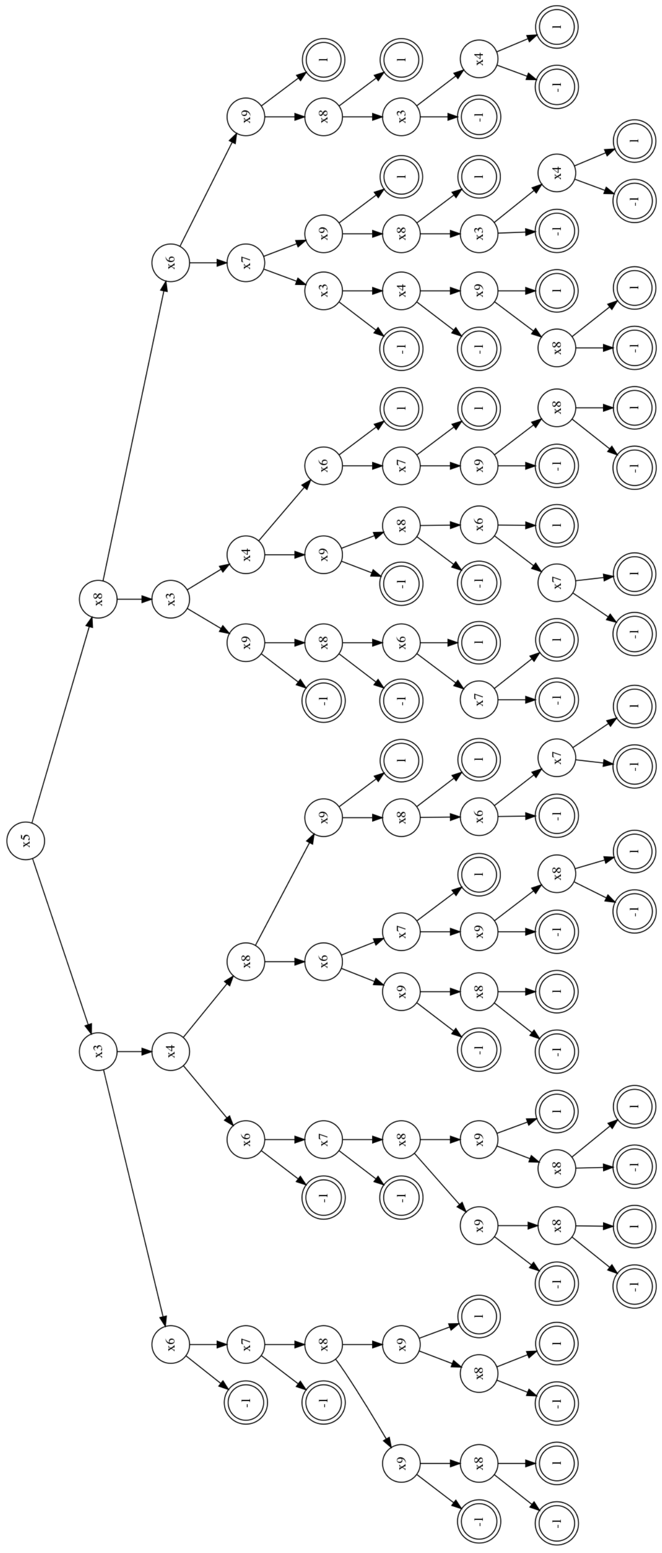

FIGURE A.0.2. A decision tree for $f(x_2, \ldots, x_n) = f_2(-1, x_2, \ldots, x_9)$. The expected number of bits read is $\frac{329}{64}$, and the number of vertices is leaves is 58.

Bibliography

[1] József Beck. *Combinatorial games: tic-tac-toe theory*. Number 114. Cambridge University Press, 2008.

[2] Michael Ben-Or and Nathan Linial. Collective coin flipping. *Randomness and Computation*, 5:91–115, 1990.

[3] Miao Chen. Toward optimal tree construction of monotone functions. Master's thesis, Duquesne University, August 2005.

[4] Mike Develin and Sam Payne. Discrete bidding games. *the electronic journal of combinatorics*, 17(1):R85, 2010.

[5] David Gale. The game of hex and the brouwer fixed-point theorem. *American Mathematical Monthly*, pages 818–827, 1979.

[6] Ryan B Hayward and Jack van Rijswijck. Hex and combinatorics. *Discrete Mathematics*, 306(19):2515–2528, 2006.

[7] Gil Kalai, Reshef Meir, and Moshe Tennenholtz. General-sum bidding games. *arXiv preprint arXiv:1311.0913*, 2013.

[8] Michael Krivelevich. Positional games. *arXiv preprint arXiv:1404.2731*, 2014.

[9] Andrew J. Lazarus, Daniel E. Loeb, James G. Propp, Walter R. Stromquist, and Daniel H. Ullman. Combinatorial games under auction play. *Games and Economic Behavior*, 27(2):229–264, 1999.

[10] Andrew J. Lazarus, Daniel E. Loeb, James G. Propp, and Daniel H. Ullman. Richman games. *Games OF No Chance*, 1994.

[11] Nikos Leonardos. An improved lower bound for the randomized decision tree complexity of recursive majority. In *Automata, Languages, and Programming*, pages 696–708. Springer, 2013.

[12] Frédéric Magniez, Ashwin Nayak, Miklos Santha, and David Xiao. Improved bounds for the randomized decision tree complexity of recursive majority. In *Automata, Languages and Programming*, pages 317–329. Springer, 2011.

[13] Robert McNaughton. Infinite games played on finite graphs. *Annals of Pure and Applied Logic*, 65(2):149–184, 1993.

[14] Ryan O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

[15] Ryan O'Donnell, Michael Saks, Oded Schramm, and Rocco A Servedio. Every decision tree has an influential variable. In *Foundations of Computer Science, 2005. FOCS 2005. 46th Annual IEEE Symposium on*, pages 31–39. IEEE, 2005.

[16] Ryan O'Donnell and Rocco A Servedio. Learning monotone decision trees in polynomial time. *SIAM Journal on Computing*, 37(3):827–844, 2007.

[17] Ryan O'Donnell and Rocco Anthony Servedio. On decision trees, influences, and learning monotone decision trees. 2004.

[18] Yuval Peres, Oded Schramm, Scott Sheffield, and David B Wilson. Random-turn hex and other selection games. *American Mathematical Monthly*, 114(5):373–387, 2007.

[19] Michael Saks and Avi Wigderson. Probabilistic boolean decision trees and the complexity of evaluating game trees. In *Foundations of Computer Science, 1986., 27th Annual Symposium on*, pages 29–38. IEEE, 1986.

[20] Ulrich Schwalbe and Paul Walker. Zermelo and the early history of game theory. *Games and economic behavior*, 34(1):123–137, 2001.