Effective Strategies for Enumeration Games

Martin Kummer and Matthias Ott*

Institut für Logik, Komplexität und Deduktionssysteme
Universität Karlsruhe, D-76128 Karlsruhe, Germany
Email: {kummer; m_ott}@ira.uka.de

Abstract

We study the existence of effective winning strategies in certain infinite games, so called enumeration games. Originally, these were introduced by Lachlan (1970) in his study of the lattice of recursively enumerable sets. We argue that they provide a general and interesting framework for computable games and may also be well suited for modelling reactive systems. Our results are obtained by reductions of enumeration games to regular games. For the latter effective winning strategies exist by a classical result of Büchi and Landweber. This provides more perspicuous proofs for several of Lachlan's results as well as a key for new results. It also shows a way of how strategies for regular games can be scaled up such that they apply to much more general games.

1 Introduction

Infinite games have been studied for a long time in many areas of mathematical logic. In recent years they also appeared in computer science as a framework for modelling reactive systems (see [Tho95] for a recent survey). Here the basic issue is the question of *effective determinacy*, i.e., which of the players has a computable winning strategy and how to determine such a strategy effectively from the description of the game? A central tool for answering this question is the effective determinacy result for regular games of Büchi and Landweber [BL69] which has been restated by McNaughton in a more applicable form concerning infinite games on finite graphs [McN93].

In recursion theory the game theoretic point of view is an important heuristic: Priority arguments can often be visualized as winning strategies in certain infinite games. This was first noticed by Lachlan in his influential paper [Lac70]. In this paper he also introduced the formal framework of *enumeration games* and proved an effective determinacy result for an interesting class of such games. In an enumeration game there are two players who enumerate sets of natural numbers in successive rounds. The winning condition is given by an open formula in the language of the lattice of recursively enumerable (r.e.) sets. Player I wins iff the formula is satisfied by the enumerated sets. Lachlan's determinacy result yields a decision procedure for the $\forall\exists$-formulae that are *uniformly* valid in the lattice of r.e. sets modulo finite sets.

In the present paper we give some illustrative examples which show that enumeration games may be useful for modelling aspects of reactive systems. In the main part we study to what extend enumeration games can be reduced to McNaughton's graph games. It turns

*Supported by the Deutsche Forschungsgemeinschaft Graduiertenkolleg "Beherrschbarkeit komplexer Systeme" (DFG Vo 287/5-5)

out that this can be done for interesting subclasses of the games considered by Lachlan. The reductions are of increasing complexity. In the easiest cases there is a one-to-one correspondence between the moves in the enumeration game and the graph game. In a more difficult reduction it happens that some of the moves in the graph game are never transferred to the enumeration game. A priority queue is used to guarantee that sufficiently many moves are transferred.

The original framework of enumeration games can be generalized in two directions, by changing the language of winning conditions (the "specification language"), or by changing the rules for enumeration. In the basic case there is just the predicate $Finite(U)$ stating that U is a finite set. More generally we consider other Σ_2-predicates P and show that effective determinacy still holds for all Σ_2-predicates which are complete w.r.t. extensional m-reductions. However, we also provide a natural example where the corresponding game is not effectively determined.

Finally, we present a class of enumeration games where the rules for enumeration are suitably modified and the original language of Lachlan is extended by cardinality predicates. In our version both players successively extend initial segments of the characteristic functions of their sets. Effective determinacy can again be shown by reductions to graph games and yields as a corollary that the $\forall\exists$-formulae which are uniformly valid in the boolean algebra of recursive sets are decidable.

This paper is based on [Ott95] where additional details can be found.

2 Notation and definitions

The recursion theoretic notation follows the books [Odi89, Soa87]. ω denotes the set of all natural numbers. We write X^C for the complement of the set $X \subseteq \omega$ in ω. R_1 is the set of all recursive functions. $\{\varphi_i\}_{i\in\omega}$ is a Gödel numbering of the partial recursive functions and $W_i = dom(\varphi_i)$ is the i-th recursively enumerable set. Σ_n and Π_n are the classes of the arithmetical hierarchy.

The notation for handling infinite objects follows [Tho90]. Σ^ω is the set of all ω-sequences over the alphabet Σ. ω-sequences are written in the form $\rho = \rho_0\rho_1\ldots$. We use $(\exists^\omega i)$ as an abbreviation for "there exists infinitely many i".

We consider two-person-games of infinite duration, i.e., the plays consist of ω many moves. The players are called player I and II. A *strategy* for a player is a function which yields the next move for the player, given all the previous moves. For studying *effective* strategies we assume some effective coding of the finite sequences of moves. σ and τ denote strategies for the players I and II, respectively.

In an *enumeration game* of size n each of the two players enumerates n sets [Lac70]. Player I enumerates the sets $U_1, \ldots, U_n$ and player II the sets $V_1, \ldots, V_n$. A play of an enumeration game proceeds in stages. At stage $t = 0$ all sets are empty. At stages $t = 1, 3, \ldots$ player I can enumerate an element x into a set U_i, which is denoted by $\mu_t = \langle i, x\rangle$. He is also allowed to pass. In this case we write $\mu_t = 0$. At stages $t = 2, 4, \ldots$ player II moves analogously. U_i^t and V_i^t are the sets produced after stage t for $i = 1, \ldots, n$. We stipulate (for technical reasons) that the players do not repeat any move except possibly 0, i.e., no element is enumerated twice into the same set.

The winning condition of enumeration games are *winning formulas* over the sets (set variables) $U_1, \ldots, U_n, V_1, \ldots, V_n$. Winning formulas are chosen from a *specification language*. Different specification languages lead to different classes of games. In particular, we consider specification languages in which formulas are built from the predicate *Infinite*, cardinality predicates $Card_k$ for $k = 0, 1, \ldots$, set operations $\cap$, $\cup$ and ^{C}, and the logical operations

$\wedge$, $\vee$ and $\neg$. *Infinite*(X) is *true* iff X contains infinitely many elements. $Card_k(X)$ is *true* iff X contains exactly k elements. Specification languages of this type were introduced by Lachlan [Lac70].

Player I wins a play in the enumeration game of size n with winning formula F if F is satisfied by the enumerated sets $U_1, \ldots, U_n, V_1, \ldots, V_n$. Otherwise player II wins the play. A strategy σ is a *winning strategy* for player I, if player I wins every play in which he follows σ. Winning strategies for player II are defined analogously.

A class of (enumeration) games is *effectively determined*, if in every game of the class one player has an *effective* winning strategy and if this player and his winning strategy can be *effectively* determined from a description of the game. A description of an enumeration game is a pair (n, F), where n is the size and F the winning formula of the game.

3 Some examples

The idea of modelling infinite behaviours of systems with infinite games is widespread in the literature, among others in [BL69, ALW89, Mos89, PR89, WD91, NYY92, NY92, TW94, MPS95]. Infinite games are mainly used to solve Church's problem [Chu63] of synthesizing processes (or automata) from a specification of the infinite input-output behaviour.

We give some examples for the application of enumeration games in reactive systems. It is demonstrated how aspects of the infinite behaviour of our sample systems can be expressed in the specification language of enumeration games.

We do not want to give examples which can only be solved by our theorems. For the given examples the effective determinacy theorem of regular games would suffice (Fact 4.4). Instead, our emphasis lies in illustrating enumeration games and their connection to reactive systems.

3.1 Printer-spooler

The origin of our first example is [Gur89]. The reactive system of interest is a printer spooler (player I). The spooler has an input-queue in which the user (player II) can insert print-jobs. From time to time, depending on the environment like printer-status, network-status etc., the spooler takes the first job from the input-queue and sends it to the printer. An event in the system consists of an action a at time t. With each event we associate an identifier.

When the user (player II) inserts an job into the queue, in our model he enumerates the associated identifier into his set V_1. The event of removing and sending a print-job is interpreted as the spooler (player I) enumerates the associated identifier into the set U_1.

Gurevich's specification of the system was that of *fairness*: Every job inserted into the input-queue should eventually be sent to the printer. This requirement is fulfilled if the spooler (player I) wins the game with winning formula

$$F_1 := [\mathit{Infinite}(V_1) \rightarrow \mathit{Infinite}(U_1)].$$

Obviously, there is an effective winning strategy for player I in the game of size 1 with winning formula F_1.

We develop this example further. Instead of one printer there are now $2m$ printers. The printers $P_1, P_3, \ldots, P_{2m-1}$ are the main-printers and the printers $P_2, P_4, \ldots, P_{2m}$ are standby-printers. If printer P_{2i-1} fails, the spooler may sent jobs to P_{2i} instead of sending them to P_{2i-1}, for $i = 1, \ldots, m$. Sending jobs to Printer P_i means enumerating the associated event-identifier into the set U_i for $i = 1, \ldots, 2m$.

We stepwise specify the desired infinite behaviour of the spooler. First we require that the spooler should distribute the jobs equally among the main-printers in infinity:

$$\mathit{Infinite}(V_1) \rightarrow \bigwedge_{i=1,\ldots,m} \mathit{Infinite}(U_{2i-1}).$$

We introduce the possibility of sending jobs to the standby-printers:

$$\mathit{Infinite}(V_1) \rightarrow \bigwedge_{i=1,\ldots,m} (\mathit{Infinite}(U_{2i-1}) \vee \mathit{Infinite}(U_{2i})).$$

This formula has to be improved. When should the main-printers and when the standby-printers get infinitely many jobs? This depends on the error-quota of the main-printers. For handling error-messages we introduce sets E_i for $i = 1, \ldots, 2m$. If printer P_i sends an error-message, the associated event-identifier is enumerated into E_i. We assume that the printers send repeatedly error-messages as long as they are not ready to print. So, receiving no error message from a printer during a period Δt of time implies the printer is okay now. The sets E_i are sets of player II. Thus, player II now comprises the user and the printers, i.e., *all* agents of the environment which are relevant for the spooler-specification.

If there are only finitely many error-messages of printer P_{2i-1} then only finitely many jobs should be sent to the standby-printer P_{2i}.

Jobs can get lost. This can happen when the spooler sends a job to a printer at which an error occurred. There may be a period of time between the occurrence of an error and the arrival of the error-message at the spooler. So the spooler may send jobs to a faulty printer assuming that this printer is okay. We only want to lose finitely many jobs in this way. Therefore, we specify that if there are infinitely many error-messages from printer P_{2i-1}, then only finitely many jobs should be sent to this printer:

$$\begin{aligned} F_2 := [\mathit{Infinite}(V_1) \rightarrow \bigwedge_{i=1,\ldots,m} &[(\mathit{Finite}(E_{2i-1}) \wedge \mathit{Infinite}(U_{2i-1}) \wedge \mathit{Finite}(U_{2i})) \vee \\ &(\mathit{Infinite}(E_{2i-1}) \wedge \mathit{Finite}(U_{2i-1}) \wedge \mathit{Infinite}(U_{2i}))]] \end{aligned}$$

But in the game with winning formula F_2 player II has a winning strategy. Every time, when player I enumerates a new element in U_{2i-1}, player II extends E_{2i-1} in one of the next moves. Additionally, player II enumerates infinitely many elements in V_1. Consequently, the premise $\mathit{Infinite}(V_1)$ of F_2 is fulfilled, but the sets E_{2i-1} and U_{2i-1} either remain both finite or become both infinite for all $i = 1, \ldots, m$. Hence, the desired spooler is not realizable. The requirements on the spooler have to be reduced, e.g. by removing the atoms $\mathit{Finite}(U_{2i-1})$ in the second parts of the disjunctions. Then player I has an effective winning strategy.

Formula F_2 is a good example for a specification, where one gets additional insights by viewing the specification as a winning formula of a game.

How can one check, if a given specification is realizable or not? It follows from Theorem 4.6 that this can be done automatically for the used specification language. Furthermore, in the case of realizability an implementation can be determined effectively from the specification.

A further requirement on the spooler is that the standby-printers should only be used, if there occurs at least one error at the main printers. This can be expressed by the predicate Card_0 in an additional requirement

$$\bigwedge_{i=1,\ldots,m} (\mathit{Card}_0(E_{2i-1}) \rightarrow \mathit{Card}_0(U_{2i})).$$

The underlying extended specification language is covered by theorem 4.7.

3.2 Access-control-system

As an example for the use of set operations in specifications, we consider an access-control-system. Users can prove their rights for using a particular resource by delivering a *capability* to the access-control-system. (Of course, in such a system security is an issue and has to be achieved by cryptographic methods. But this is not relevant in our context.) With respect to the received capabilities the access-control-system grants or refuses access to the protected resources.

Because of efficiency, capabilities are valid for a period of time, i.e., a user may deliver a capability once and can use the appropriate resource as long as this capability is valid. Thus, one requirement on the access-control-system is that when the delivering of capabilities stops, the system must eventually stop granting access to the appropriate resource.

There are resources which may only be used by groups. If user i delivers a capability for the resource j at day d, the value d is enumerated into the set $V_{i,j}$. Access to the resource j is only granted (for a period of time) if two of the three users $1, 2, 3$ deliver a capability at the same day. Granting access to resource j is modeled by enumerating the associated event-identifier into the set U_j. Thus, our requirement is

$$F_3 := [\mathit{Finite}(V_{1,j} \cap V_{2,j}) \wedge \mathit{Finite}(V_{1,j} \cap V_{3,j}) \wedge \mathit{Finite}(V_{2,j} \cap V_{3,j}) \rightarrow \mathit{Finite}(U_j)].$$

Theorem 4.8 deals with the appropriate kind of specification language. Of course, there are other reasonable premises instead of the premise in formula F_3. Indeed, every formula built from $\mathit{Finite}, \cap, \wedge, \vee$ is in principle a reasonable premise.

4 Reductions to infinite graph games

In [Lac70] A. H. Lachlan stated the following fact:

Fact 4.1 (Lachlan) *The enumeration games with predicate* Infinite *and the set operations* $\cap$*,* $\cup$ *and* ^{C} *are effectively determined.*

As a consequence the uniform $\forall\exists$-theory of $\mathcal{E}^*$, the lattice of r.e. sets modulo finite sets, is decidable: A $\forall\exists$-sentence $S = (\forall V_1 \ldots \forall V_n)(\exists U_1 \ldots \exists U_n)F[V_1, \ldots, V_n, U_1, \ldots, U_n]$ with matrix F is called *uniformly* valid in $\mathcal{E}^*$ if there is an effective procedure to compute from indices $i_1, \ldots, i_n$ of the V_i's indices $j_1, \ldots, j_n$ of the U_j's such that $F[W_{i_1}, \ldots, W_{i_n}, W_{j_1}, \ldots, W_{j_n}]$ holds. The following folklore proposition connects the existence of effective winning strategies in enumeration games with uniform validity.

Proposition 4.2 *A $\forall\exists$-sentence S is uniformly valid iff player I has an effective winning strategy in the enumeration game with winning formula S.*

Proof: Let $S = (\forall V_1 \ldots \forall V_n)(\exists U_1 \ldots \exists U_n)F[V_1, \ldots, V_n, U_1, \ldots, U_n]$ be any given $\forall\exists$-sentence.

$(\Leftarrow)$: Given $i_1, \ldots, i_n$ simulate the winning strategy of player I against an opponent who enumerates $W_{i_1}, \ldots, W_{i_n}$. This defines n sets with indices say $j_1, \ldots, j_n$ (obtained from the s-m-n theorem) such that $F[W_{i_1}, \ldots, W_{i_n}, W_{j_1}, \ldots, W_{j_n}]$ holds.

$(\Rightarrow)$: We show the contraposition. Assume that player I does not have an effective winning strategy in the enumeration game specified by S. Then, by effective determinacy, player II has an effective winning strategy. Now suppose for a contradiction that the recursive function f witnesses the uniform validity of S. Using f, the recursion theorem, and the winning strategy of player II one can construct indices $i_1, \ldots, i_n$ such that $f(i_1, \ldots, i_n) = (j_1, \ldots, j_n)$ and $F[W_{i_1}, \ldots, W_{i_n}, W_{j_1}, \ldots, W_{j_n}]$ does not hold, a contradiction. ∎

Lachlan [Lac70, Section 3] gave a very brief sketch of how to prove Fact 4.1. But this sketched proof is rather difficult.

We now introduce a method for proving the effective determinacy of enumeration games. This method is suitable for a subclass of the games in Fact 4.1 (the games where the set operation ^{C} is excluded) and for many other classes of enumeration games.

4.1 Infinite graph games

The method is based on a result of R. McNaughton [McN93]. He introduced *infinite graph games*. These two person games are played on a finite graph. At any time of a play a marker is on one node of the graph. The players move this marker alternately from node to node along the edges of the graph. A play consists of ω many moves beginning with a move of player I. The winner of an (infinite) play is determined by the set of nodes, which were visited infinitely often by the marker.

The formal definition of graph games contains some restrictions, which is to some extent for technical reasons only.

Definition 4.3 An *infinite graph game* $\mathcal{G}$ is an ordered sextuple $(Q, Q_{\mathrm{I}}, Q_{\mathrm{II}}, E, q_0, \Omega)$, where (Q, E) is a finite bipartite directed graph, Q_{I}, Q_{II} are the set of nodes to which player I, II may move, respectively, q_0 is the initial node and $\Omega \subseteq 2^Q$ is the set of winning subsets of Q. We postulate $Q_{\mathrm{I}} \cup Q_{\mathrm{II}} = Q \neq \emptyset$, $Q_{\mathrm{I}} \cap Q_{\mathrm{II}} = \emptyset$ and that for each $e \in E$ there exist $p \in Q_{\mathrm{I}}$ and $q \in Q_{\mathrm{II}}$ such that either $e = (p, q)$ or $e = (q, p)$. Furthermore, for each $p \in Q$ there must be a node $q \in Q$ with $(p, q) \in E$.

A play of a graph game $\mathcal{G} = (Q, Q_{\mathrm{I}}, Q_{\mathrm{II}}, E, q_0, \Omega)$ is a sequence $\rho \in Q^\omega$ such that $\rho_0 = q_0$ and $(\rho_t, \rho_{t+1}) \in E$ for all $t \in \omega$.

$$\mathrm{In}(\rho) := \{q \in Q\colon (\exists^\omega t)[\rho_t = q]\}$$

is the set of nodes, which were visited infinitely often during the play ρ. Player I wins the play ρ if $\mathrm{In}(\rho)$ is an element of Ω, otherwise player II wins the play.

McNaughton proved the following fact:

Fact 4.4 (McNaughton) *Infinite graph games are effectively determined.*

Actually McNaughton showed a stronger result (Theorem 4.1 in [McN93]). Especially he proved that one can always construct an *LVR-strategy* for the winner. This is a strategy, which needs only finite memory capacity. The name LVR originates in the way of book-keeping the visited nodes. Fact 4.4 is not really new. It is rather a reformulation of an older result by Büchi and Landweber [BL69]. The games solved there are called *finite-state games* or *regular games*.

4.2 Reductions for games with predicate *Infinite*

In this section we prove that the enumeration games with predicate *Infinite* are effectively determined. The proof is by reductions to infinite graph games. The idea of the reductions is to associate an infinite graph game with each enumeration game. In the associated graph game we can effectively compute a winning strategy for one player by Fact 4.4. This winning strategy is translated into the enumeration game.

Suppose w.l.o.g. that player I has a winning strategy $\sigma_{\mathcal{G}}$ in the graph game. Player I simulates a play in the graph game in parallel to the play in the enumeration game. He has to translate the moves of player II from the enumeration game into the graph game. In the graph game he follows his winning strategy and retranslates the resulting moves into the enumeration game.

Assume that the size n of an enumeration game is given. We now construct the graph game with sets of nodes $Q_{\mathrm{I}}^n = \{U_0, U_1, \ldots, U_n\}$, $Q_{\mathrm{II}}^n = \{V_0, V_1, \ldots, V_n\}$, $Q_n := Q_{\mathrm{I}}^n \cup Q_{\mathrm{II}}^n$ and the edges $E_n := (Q_{\mathrm{I}}^n \times Q_{\mathrm{II}}^n) \cup (Q_{\mathrm{II}}^n \times Q_{\mathrm{I}}^n)$. A visit on a node U_i (V_i) in the graph game shall correspond to an extension of the set U_i (V_i) in the enumeration game for $i = 1, \ldots, n$. The nodes U_0 and V_0 of the graph game represent passes in the enumeration game.

At the beginning ($t = 0$) the marker is put on the node V_0 (an arbitrary node from Q_{II}^n would suffice).

In stage $t + 1 = 2s + 1$ it is player I's turn to move. He computes $q_{t+1} := \sigma_{\mathcal{G}}(q_0 \ldots q_t)$ according to his winning strategy $\sigma_{\mathcal{G}}$. If $q_{t+1} = U_0$ then player I passes in the enumeration game in stage $t + 1$, i.e., he chooses $\mu_{t+1} = 0$. Otherwise $q_{t+1} = U_i$ for an $i \in \{1, \ldots, n\}$. In this case player I enumerates a new element into the set U_i by choosing $\mu_{t+1} = \langle i, 1 + \max U_i \rangle$.

In stage $t + 1 = 2s + 2$ it is player II's turn to move. Player I observes the move μ_{t+1} of player II in the enumeration game. If player II passes, then the marker is put on the node V_0 of the graph. Otherwise $\mu_{t+1} = \langle i, x \rangle$ for an $i \in \{1, \ldots, n\}$. We stipulated that the players perform no repeating moves. So we can conclude $V_i^t \subset V_i^{t+1}$. In this case player I simulates the move of player II by moving the marker on the node V_i of the graph game.

Let $\rho = q_0 q_1 q_2 \ldots$ be the produced play in the graph game. It turns out that every node U_i (V_i) is in $\mathrm{In}(\rho)$ iff the set U_i (V_i) is infinite in the play of the enumeration game $(*)$.

The given construction depends only on the size n of the given enumeration game, but not on the winning formula F. We fix the size n. With each winning formula F we can associate a winning set $\Omega(F)$ by induction on the structure of F:

$$\Omega(\mathit{Infinite}(U_i)) := \{\pi \subseteq Q\colon\ U_i \in \pi\} \text{ for i=1,\ldots,n.}$$

$$\Omega(\mathit{Infinite}(V_i)) := \{\pi \subseteq Q\colon\ V_i \in \pi\} \text{ for i=1,\ldots,n.}$$

$$\Omega(F_1 \wedge F_2) := \Omega(F_1) \cap \Omega(F_2)$$

$$\Omega(\neg F_1) := 2^Q - \Omega(F_1)$$

We show that by this definition for each winning formula F of the enumeration game of size n the following Lemma holds:

Lemma 4.5 *F is valid in the enumeration game $\iff$ $In(\rho) \in \Omega(F)$.*

Proof: The proof is by induction on the structure of F. For atomic formulas the statement follows directly from $(*)$. In the induction step one only makes use of the analogy between the propositional logic operators and the set operations. ▮

Lemma 4.5 says that the plays in both games have the same winner, if player I plays according to the above translation.

We can now prove the following theorem:

Theorem 4.6 *The enumeration games with predicate* Infinite *are effectively determined by reductions to graph games.*

Proof: For a given enumeration game of size n with winning formula F we construct the graph game $\mathcal{G} = (Q_n, Q_{\mathrm{I}}^n, Q_{\mathrm{II}}^n, E_n, V_0, \Omega(F))$. By Fact 4.4 we can effectively determine the winner and an effective winning strategy from $\mathcal{G}$. If player I has an effective winning strategy $\sigma_{\mathcal{G}}$ in $\mathcal{G}$ we translate it into a strategy σ for player I in the enumeration game by the described algorithm.

If player I follows σ in the enumeration game he wins the associated play in the graph game, since $\sigma_{\mathcal{G}}$ is an winning strategy for player I. By Lemma 4.5 player I also wins the play in the enumeration game. Hence σ is a winning strategy for player I in the enumeration game. Since $\sigma_{\mathcal{G}}$ is an effective strategy and all constructions are effective, the strategy σ is also effective.

If player II has a winning strategy in $\mathcal{G}$, this strategy can be analogously translated into an winning strategy for player II in the enumeration game. ∎

One may argue that the above proof for the games with predicate *Infinite* is somewhat circumstantial. Actually there are more succinct formulations. But the given formulation is for demonstrating the method of reductions to graph games. In more difficult games the above proof scheme is also applicable and turned out to be very helpful.

A slight modification of the given proof yields the result that the games with the predicate "*is superset of* A", for an arbitrary but fixed infinite r.e. set A, are effectively determined. If A is finite this predicate is a Σ_1-predicate (see introduction of section 5).

4.3 Predicates *Infinite* and $Card_k$

At first we extend the class of enumeration games of Theorem 4.6 by allowing cardinality predicates besides the predicate *Infinite*:

Theorem 4.7 *The enumeration games with the predicates* Infinite *and* Card_k *for* $k \in \omega$ *are effectively determined by reductions to graph games.*

Proof: The proof is similar to that for games with predicate *Infinite* only. But now we additionally attach to each node of the game graph a *counter* $\gamma\colon \{U_1, \ldots, U_n, V_1, \ldots, V_n\} \rightarrow \{1, \ldots, k_1\}$. k_1 is a number such that for all atomic formulas $Card_k(W)$ occurring in the given winning formula F the value of k is less than k_1. Formally we choose as set of nodes

$$Q_{\mathrm{I}} := \{(U_i, \gamma)\colon\ i = 0, \ldots, n \text{ and } \gamma \text{ is a counter}\}$$

$$Q_{\mathrm{II}} := \{(V_i, \gamma)\colon\ i = 0, \ldots, n \text{ and } \gamma \text{ is a counter}\}.$$

The edges are defined in such a way that the cardinality of the sets U_i and V_i are counted in the appropriate γ. But we only increment the counters until the bound k_1 is reached. That is for all $i, j \in \{0, \ldots, n\}$ and all counters δ, γ we take the edge $((U_i, \delta), (V_j, \gamma))$ in the set E_n iff

$$\begin{aligned}
&(j = 0 \wedge \delta = \gamma) \vee \\
&(j \neq 0 \wedge \\
&\ \gamma(V_j) = \min\{\delta(V_j) + 1, k_1\} \wedge \\
&\ (\forall W \neq V_j)[\gamma(W) = \delta(W)]).
\end{aligned}$$

The edges $((V_i, \delta), (U_j, \gamma))$ are defined analogously. The translation of a strategy from such a graph game into the enumeration game is performed in an obvious manner. The winning sets are defined by structural induction on the formulas which at most contain atomic formulas $Card_k(W)$ with $k < k_1$. (Note that if this is true for a formula F, it is also true for all subformulas of F). We only give definitions for sets U_i, $i = 1, \ldots, n$:

$$\Omega(\mathit{Infinite}(U_i)) := \{\pi \subseteq Q_n\colon\ (\exists \gamma)[(U_i, \gamma) \in \pi]\}$$

$$\Omega(Card_k(U_i)) := \{\pi \subseteq Q_n \colon (\forall (W,\gamma) \in \pi)[\gamma(U_i) = k]\} \text{ for } k < k_1.$$

The definition for nonatomic formulas is the same as in Section 4.2. With this definition one can prove the analogous statement to Lemma 4.5. The remainder of the proof is identical with that of Theorem 4.6. ∎

4.4 Predicate *Infinite* and set operations $\cap$ and $\cup$

In this subsection we extend the specification language of Theorem 4.6 by introducing the set operations $\cap$ and $\cup$ while *Infinite* remains the only allowed predicate.

Theorem 4.8 *The enumeration games with predicate* Infinite *and the set operations* $\cap$ *and* $\cup$ *are effectively determined by reductions to graph games.*

Proof: Because of the distributivity and associativity laws each term built from sets with the operations $\cap$ and $\cup$ can be represented in the form

$$\bigcup_{i \in M} \bigcap_{j \in M_i} A_j.$$

But for arbitrary sets $B_1, \ldots, B_m$ we have

$$Infinite\Big(\bigcup_{i=1}^{m} B_i\Big) \iff \bigvee_{i=1}^{m} Infinite(B_i). \tag{1}$$

Hence we can restrict ourselves to the games with atoms

$$Infinite\Big(\bigcap_{W \in M} W\Big)$$

for nonempty sets $M \subseteq \{U_1, \ldots, U_n, V_1, \ldots, V_n\}$.

The idea of the reduction is to introduce graph nodes for each subset $M \subseteq \{U_1, \ldots, U_n, V_1, \ldots, V_n\}$. A node with $M = \emptyset$ represents a pass in the enumeration game. A visit on a graph node marked with subset $M \neq \emptyset$ shall correspond to an extension of the set

$$S(M) := \bigcap_{W \in M} W - \bigcup_{W \notin M} W.$$

But there is only hope that player I (II) can extend this set, when at least one U_i (V_i) is a member of M. This leads one to the following definitions:

$$Q_{\mathrm{I}}^n := \{(\mathrm{I}, M) \colon M \subseteq \{U_1, \ldots, U_n, V_1, \ldots, V_n\} \text{ and } (M = \emptyset \text{ or } M \cap \{U_1, \ldots, U_n\} \neq \emptyset)\}$$

$$Q_{\mathrm{II}}^n := \{(\mathrm{II}, M) \colon M \subseteq \{U_1, \ldots, U_n, V_1, \ldots, V_n\} \text{ and } (M = \emptyset \text{ or } M \cap \{V_1, \ldots, V_n\} \neq \emptyset)\}.$$

Again, we connect the two sets completely:

$$E_n := (Q_{\mathrm{I}}^n \times Q_{\mathrm{II}}^n) \cup (Q_{\mathrm{II}}^n \times Q_{\mathrm{I}}^n).$$

We consider the translation of a strategy for player I from the graph game into the enumeration game. If player II in stage $t+1$ enumerates the new element x into V_i^{t+1}, we determine the set

$$M := \{U_i \colon x \in U_i^t\} \cup \{V_i \colon x \in V_i^t\}$$

and move the marker to the node $(\mathrm{II}, M \cup \{V_i\})$.

The other direction is a little bit more complicated. Consider the graph move $q_{t+1} = (\mathrm{I}, M)$ of player I. Now player I wants to enumerate a new element into the set $S(M)$. This is easy if $M = \{U_i\}$. If $|M| > 1$ then he has to find an U_i such that the set $S(M - \{U_i\})$ contains at least one element. But it is possible that all of these sets are empty.

The solution is to put all graph moves into a *buffer* and to translate them later when the moves are *executable*. The buffer is organized as a (horizontal) queue, that is new moves are inserted from the right and executable moves are searched from the left. This secures that all moves which are executable infinitely often will eventually be translated. For technical reasons we must allow player I to perform moves in the enumeration game as long as there are executable ones in the buffer *without* any move of player II in-between. I.e., player I is allowed to perform finitely many moves at each stage. Otherwise player II could hinder player I making $U_1 \cap U_2$ infinite by answering every move of x into U_1 or U_2 with enumerating x also into V_1 in the subsequent move. This generalization does not affect the question of effective determinacy.

Let us now construct the winning sets of the graph game such that all plays in both games have the same winner. A set $\bigcap_{W \in M} W$ becomes infinite iff there is an $N \supseteq M$ such that the set $S(N)$ is extended infinitely often during the play. One can show that this is the case iff either

- (I, N) or (II, N) is visited infinitely often in the graph game and $|N| = 1$ or
- (I, N) is visited infinitely often and there is $U_i \in N$ such that $N - \{U_i\} \neq \emptyset$ and $S(N - \{U_i\})$ is infinite or
- (II, N) is visited infinitely often and there is $V_i \in N$ such that $N - \{V_i\} \neq \emptyset$ and $S(N - \{V_i\})$ is infinite.

Let $D_{\mathrm{I}} := \{U_1, \ldots, U_n\}$ and $D_{\mathrm{II}} := \{V_1, \ldots, V_n\}$. We first define a relation $\mathit{Consistent}(q, \pi)$ for $\pi \subseteq Q_n$ and $q = (\Lambda, N) \in \pi$ as the smallest relation with the following properties:

- $|N| = 1 \implies \mathit{Consistent}(q, \pi)$
- $(\exists \Lambda' \in \{\mathrm{I}, \mathrm{II}\})(\exists W \in D_\Lambda)[\mathit{Consistent}((\Lambda', N - \{W\}), \pi)] \implies \mathit{Consistent}(q, \pi)$.

For each $M \subseteq \{U_1, \ldots, U_n, V_1, \ldots, V_n\}$, $M \neq \emptyset$ we define

$$\Omega(\mathit{Infinite}(\bigcap_{W \in M} W)) := \{\pi \subseteq Q_n \colon (\exists (\Lambda, N) \in \pi)[M \subseteq N \wedge \mathit{Consistent}((\Lambda, N), \pi)]\}$$

The remainder of the proof follows the outline of the previous determinacy proofs. ∎

The method of reductions to graph games fails if one extends the specification language of Theorem 4.8 by the set operation ^{C}. This is because set expressions with complement behave non-monotonic. Moreover, the arithmetical hierarchy indicates that the problem with complement is more difficult. The index set $\{i \colon W_i^C \text{ is finite}\}$ is Σ_3-complete while $\{i \colon W_i \text{ is finite}\}$ is only a Σ_2-complete set.

5 Specifications by Σ_2-predicates

We call a predicate P on the recursively enumerable sets a Σ_n-predicate if the index set $M := \{i \colon P(W_i)\}$ is in the class Σ_n of the arithmetical hierarchy. Because in the language of winning formulas negation is allowed, the following considerations cover also the case $M \in \Pi_n$.

By use of the Rice/Shapiro-Theorem Σ_1-predicates can be extended unambiguously to the domain 2^ω. It is easy to show that the enumeration games with such an extended Σ_1-predicate are effectively determined. This can be proved directly by reduction to a finite game.

We now consider games for Σ_2-predicates. First we show for a special kind of such predicates that the corresponding games are effectively determined. Then we give an example for a game with a Σ_2-predicate such that none of the players has an effective winning strategy.

5.1 Σ_2-complete predicates with extensional m-reductions

In general it is not clear how Σ_n-predicates for $n > 1$ should be extended to the domain 2^ω. So we restrict the rules of enumeration games by requiring that both players have to play according to *effective* strategies. Hence all sets enumerated during a play are always recursively enumerable. This restriction is only valid for this subsection.

A Σ_2-predicate P is called Σ_2-complete if the index set $M := \{i : P(W_i)\}$ is Σ_2-complete. The predicate *Finite* is an example of a Σ_2-complete predicate, because $\mathrm{Fin} := \{i\colon W_i \text{ is finite}\}$ is a Σ_2-complete index set. So for every Σ_2-complete set M there are recursive functions $f, g \in R_1$ (so-called m-reductions) such that for all i:

$$(i \in \mathrm{Fin} \iff f(i) \in M) \text{ and } (i \in M \iff g(i) \in \mathrm{Fin}).$$

A recursive function $f \in R_1$ is *extensional* if for all i, j:

$$W_i = W_j \implies W_{f(i)} = W_{f(j)}.$$

We consider extensional m-reductions because for these reductions the Theorem of Myhill/Shepherdson is applicable:

Theorem 5.1 (Myhill/Shepherdson) *If $f \in R_1$ is extensional, then there exists an enumeration operator $\Phi\colon 2^\omega \to 2^\omega$ with $\Phi(W_i) = W_{f(i)}$ for all $i \in \omega$.*

The definition of an enumeration operator can be found e.g. in the book [Rog67], as well as Theorem 5.1 and Lemma 5.2. We don't state this definition because we only need one property of enumeration operators here, the continuity:

Lemma 5.2 *Every enumeration operator $\Phi\colon 2^\omega \to 2^\omega$ is continuous. I.e., for each increasing sequence $(A_s)_{s\in\omega}$ of subsets of ω:*

$$\Phi(\bigcup_{s\in\omega} A_s) = \bigcup_{s\in\omega} \Phi(A_s).$$

We are now ready for proving the following result:

Theorem 5.3 *Let P be a predicate such that $M := \{i\colon P(W_i)\}$ is Σ_2-complete and there are extensional m-reductions between M and Fin. Then the enumeration games with predicate P in which both players follow effective strategies are effectively determined.*

Proof: We will reduce the games with predicate P to the games with predicate *Finite*, which are effectively determined by Theorem 4.6. $U_1, \ldots, U_n, V_1, \ldots, V_n$ denote the sets which are enumerated in the games with predicate P, and $\tilde{U}_1, \ldots, \tilde{U}_n, \tilde{V}_1, \ldots, \tilde{V}_n$ denote the sets of the games with predicate *Finite*. Let F be the winning formula of a game with predicate P. Then $\tilde{F}$ is the winning formula of the game with predicate *Finite* built by replacing all occurrences of $P(U_i)$, $P(V_i)$ with $Finite(\tilde{U}_i)$, $Finite(\tilde{V}_i)$, respectively.

Assume w.l.o.g. that player I has an effective winning strategy in the game with winning formula $\tilde{F}$. We translate this winning strategy into the game with winning formula F.

Let Φ_f and Φ_g denote the corresponding enumeration operators from Theorem 5.1:

$$(\forall j)[\Phi_f(W_j) = W_{f(j)}] \text{ and } (\forall j)[\Phi_g(W_j) = W_{g(j)}].$$

Because f and g are m-reductions we have:

$$(\forall j \in \omega)[Finite(W_j) \iff P(\Phi_f(W_j))] \tag{2}$$

$$(\forall j \in \omega)[P(W_j) \iff Finite(\Phi_g(W_j))] \tag{3}$$

In the translation we use the operators Φ_f and Φ_g to transform the enumerated sets between the two games:

1 **Translation from $\tilde{F}$ to F**
For all $i \in \{1,\ldots,n\}, t \in \omega$ compute an index $\tilde{h}(i,t)$ with $\tilde{U}_i^t = W_{\tilde{h}(i,t)}$ and enumerate the set $W_{f(\tilde{h}(i,t))}$ into U_i (by *dovetailing*).

2 **Translation from F to $\tilde{F}$**
For all $i \in \{1,\ldots,n\}, t \in \omega$ compute an index $h(i,t)$ with $V_i^t = W_{h(i,t)}$ and enumerate the set $W_{g(h(i,t))}$ into $\tilde{V}_i$ (by *dovetailing*).

The indices $h(i,t)$ and $\tilde{h}(i,t)$ can be computed because the sets $\tilde{U}_i^t$ and V_i^t are finite. By Lemma 5.2 we get for all $i = 1,\ldots,n$:

$$U_i = \bigcup_{t\in\omega} W_{f(\tilde{h}(i,t))} = \bigcup_{t\in\omega} \Phi_f(\tilde{U}_i^t) = \Phi_f(\bigcup_{t\in\omega} \tilde{U}_i^t) = \Phi_f(\tilde{U}_i)$$

$$\tilde{V}_i = \bigcup_{t\in\omega} W_{g(h(i,t))} = \bigcup_{t\in\omega} \Phi_g(V_i^t) = \Phi_g(\bigcup_{t\in\omega} V_i^t) = \Phi_g(V_i).$$

Because both players follow effective strategies (by hypothesis) the sets $\tilde{U}_i$ and V_i are all recursively enumerable. Hence the sets U_i and $\tilde{V}_i$ are recursively enumerable, too. From (2) and (3) it follows that the formula F is fulfilled iff the formula $\tilde{F}$ is fulfilled. Since player I follows an effective winning strategy in the game with winning formula $\tilde{F}$, he also wins the game with winning formula F. Thus we have constructed an effective winning strategy for player I in the game with winning formula $\tilde{F}$. ∎

5.2 A game which is not effectively determined

It can be shown that Theorem 5.3 does not hold if the hypothesis 'extensional' is omitted [Ott95]. However, the counterexample looks somewhat contrived.

We now present a more natural example of a specification which is not effectively determined (however, in this case the Σ_2-predicate is not m-complete).

It is well-known that for every r.e. set A the index set $\{i : W_i \not\subseteq A\}$ belongs to Σ_2.

Theorem 5.4 *There is a recursively enumerable set A such that the enumeration game with winning formula*

$$F := [U_1 \subseteq A \leftrightarrow (V_1 \subseteq A \vee V_2 \subseteq A)]$$

in which both players follow effective strategies is not effectively determined.

Proof sketch: For every A, player I has a winning strategy recursive in A: As long as V_1^t and V_2^t are subsets of A, player I does nothing. If for the first time $V_1^t \not\subseteq A \ \wedge \ V_2^t \not\subseteq A$, then player I chooses an $x \in V_1^t - A$ and puts x into U_1^{t+1}.

Now it is easy to see that for every strategy τ of player II there is a recursive strategy of player I which wins against τ.

The r.e. sets A for which player I has a recursive winning strategy can be characterized as follows.

Claim: Player I has a recursive winning strategy in the enumeration game with winning formula

$$F := [U_1 \subseteq A \leftrightarrow (V_1 \subseteq A \vee V_2 \subseteq A)]$$

iff there is a recursive function f such that for all x, y:

$$W_{f(x,y)} \subseteq A \iff (x \in A \ \vee \ y \in A).$$

Using a finite injury priority argument one can construct an r.e. set A which does not satisfy the condition of the claim. Thus, for the corresponding game neither player I nor player II has a recursive winning strategy. ∎

6 Enumeration games on recursive sets

We now consider enumeration games in which the players enumerate characteristic functions instead of sets. Consequently, if both players follow effective strategies the enumerated games are recursive. Therefore we call them *games on recursive sets*.

In his moves player II defines the values of the characteristic functions $\chi_{V_1}(x), \ldots, \chi_{V_n}(x)$ successively for $x = 0, 1, 2, \ldots$, i.e., in each move he chooses an element from the alphabet $\{0,1\}^n$. Player I is allowed to define the values of $\chi_{U_1}(x), \ldots, \chi_{U_n}(x)$ for an x such that $\chi_{V_1}(x), \ldots, \chi_{V_n}(x)$ are already defined, i.e., he extends the corresponding move $b \in \{0,1\}^n$ from player II with a vector $a \in \{0,1\}^n$ and produces the word ba. Player I is also allowed to pass in his moves. But he must extend all moves from player II exactly once during a play. In this case we call the play *complete*. However, for each move of player II he can wait arbitrary long until he extends it. By convention player I loses all incomplete plays. We let player II do the first move.

In the above definition the options of player I and player II are asymmetric. However, this is just what is needed to decide the uniformly valid $\forall\exists$-sentences in $\mathcal{B}$, the boolean algebra of recursive sets (see Corollary 6.2). A $\forall\exists$-sentence $S = (\forall V_1 \ldots \forall V_n)(\exists U_1 \ldots \exists U_n)F[V_1, \ldots, V_n, U_1, \ldots, U_n]$ with matrix F is called *uniformly* valid in $\mathcal{B}$ if there is an effective procedure to compute from indices $i_1, \ldots, i_n$ of characteristic functions of V_i's indices $j_1, \ldots, j_n$ of characteristic functions of the U_j's such that $F[M_{i_1}, \ldots, M_{i_n}, M_{j_1}, \ldots, M_{j_n}]$ holds. Here M_k denotes the recursive set with characteristic function φ_k.

For games on recursive sets we can allow the entire specification language of Lachlan still preserving effective determinacy:

Theorem 6.1 *The enumeration games on recursive sets with the predicates* Infinite *and* Card_k *for* $k \in \omega$ *and the set operations* $\cap$, $\cup$ *and* ^{C} *are effectively determined by reductions to graph games.*

Proof: If player I extends a move $b = b_1, \ldots, b_n \in \{0,1\}^n$ of player II with $a = a_1, \ldots, a_n \in \{0,1\}^n$ this means that the corresponding $x \in \omega$ is enumerated into the set

$$S(ba) := \bigcap_{a_i=1} U_i \cap \bigcap_{b_i=1} V_i \cap \bigcap_{a_i=0} U_i^C \cap \bigcap_{b_i=0} V_i^C.$$

For $ab \neq a'b'$ the sets $S(ab)$ and $S(a'b')$ never have an element in common. Each set expression built from the sets $U_1, \ldots, U_n, V_1, \ldots, V_n$ and the operations $\cap$, $\cup$ and ^{C} can be represented as a disjoint union of sets $S(c)$ where $c \in \{0,1\}^{2n}$. Because of (1) and

$$Card_k(\bigcup_{c \in C} S(c)) \iff \sum_{c \in C} |S(c)| = k \iff \bigvee_{\substack{\{k_c \in \omega:\ c \in C\} \\ \sum_{c \in C} k_c = k}} \bigwedge_{c \in C} Card_{k_c}(S(c))$$

for $C \subseteq \{0,1\}^{2n}$, we can restrict the specification language by admitting only set expressions $S(c)$ for $c \in \{0,1\}^{2n}$.

With every play of the game an interpretation of the winning formulas is associated:

$Infinite(S(c))$ is *true* $\iff$ c is produced infinitely often,

$Card_k(S(c))$ is *true* $\iff$ c is produced exactly k times.

Player I wins a play in the game with winning formula F iff the play is complete and F is true under this interpretation.

We now describe how these games can be reduced to a special kind of games, in which the rules are more restrictive. We call the original games the *target games* and the second games *restricted games*. The reductions of restricted games to graph games are straightforward.

Restricted Games: We fix a winning formula F in the target game. Let k_1 be a number such that $k < k_1$ for all atoms $Card_k(S(c))$ occurring in F.

A central idea of the reduction is that if player II plays $m := 2^n k_1$-times the same move $b \in \{0,1\}^n$, then at least one word ba will be produced at least k_1-times. So there is no more chance for atoms $Card_k(S(ba))$ occurring in F to become *true*.

The rules of the restricted games are as follows. At stage $t = 1$ player II plays a single move b_1^1 and m-times a move b_2^1 which we indicate by writing $(b_2^1)^m$. At stage $t = 2$ player I extends b_1^1 to $b_1^1 a_1^2$, and k_1 moves from $(b_2^1)^m$ with the same element a_2^2. At stage $t = 3$ player II again plays a single move b_1^3 and a *block* $(b_2^3)^m$. At stage $t = 4$ player I accordingly extends b_1^3 and k_1 moves from $(b_2^3)^m$ with a_1^4 and a_2^4, respectively. But additionally he extends all $m - k_1$ remaining moves from the block $(b_2^1)^m$. These $m - k_1$ extensions are called the *update* in state t. From now on the following stages proceed like the stages 3 and 4.

Reducing target games to restricted games: We have to show that if a player has an effective winning strategy in the restricted game with formula F, this player also has an effective winning strategy in the target game. For player I this is easy. He only has to reorder the moves of player II in the target game into the form $b_1^1(b_2^1)^m b_1^2(b_2^2)^m \ldots$. This is always possible, because the alphabet $\{0,1\}^n$ is finite and player I can pass in the target game as long as there are not enough moves of player II to build the next block. His strategy in the restricted games then tells him, how to extend all moves to win the game.

Assume now that player II has an effective winning strategy in the restricted game. There occurs the following problem. Consider the moves $b_1^t(b_2^t)^m$ from player II at a stage $t \geq 3$ in the restricted game. Player II can translate these moves by playing correspondingly one time b_1^t and m-times b_2^t in the target game. Now he has to translate the next moves from player I from the target game into the restricted game. For this he needs the extension of b_1^t, k_1 equal extensions of the played b_2^t's, and all extensions of the moves b_2^{t-2}. But by passing, player I can wait with these extensions as long as he wants to, while player II has to perform a proper move in each step. And player II gets the next suggestion, how to move, from his strategy in the restricted game not earlier than he has simulated the next moves of player I.

Player II solves this problem by playing additional moves b_2^t, until player I has done all extensions needed for the next translation step. The intuition is that player I already can produce at least k_1-times the word $b_2^t a$ for each $a \in \{0,1\}^n$, if he wants to. In other words, player I gets no further advantage.

In particular, at each stage $t \geq 3$ player II waits, until all of his b_2^{t-2}-moves from the stage $t-2$ have been extended. These may be more than m extensions. For the update in stage t he has to select exactly $m - k_1$ of those extensions which are not among the k_1 extensions a_2^{t-2} of stage $t-2$. He selects all extensions which occur less than k_1-times. From each of the others (excluding the a_2^{t-2}-extensions) he selects k_1 occurrences. He fills up this selection with arbitrary additional extensions of the b_2^{t-2}-moves such that at all exactly $m - k_1$ extensions are selected. Player II then translates the selected extensions into the restricted game.

At each stage player II plays at most finitely many additional moves in the target game. So he only plays a move infinitely often iff the appropriate move also occurs infinitely often in the restricted game. By the translation an extension occurs infinitely often in the target game iff it occurs infinitely often in the restricted game. For the extensions which occur less than k_1-times there is a one-to-one correspondence between the two games. Hence in both plays exactly the same atomic formulas are valid. Therefore the translated strategy is an effective winning strategy for player II in the target game.

Reducing restricted games to graph games: The nodes of the graph games are composed of three types of information. Of course, we need the letters b_1, b_2 and a_1, a_2 used in the current moves, and for each $a \in \{0,1\}^n$ the number of extensions in the current update step. The nodes in Q_{II}^n contain the b_i, and the nodes in Q_{I}^n the a_i and the update information.

In order to define the edges and the winning sets we also need some book-keeping of the letters used in preceeding moves. To the nodes of player II we add a component b_3 which stores the component b_2 of the preceeding node of player II. The nodes of player I are equipped with components b_1, b_2, b_3 holding the values of the corresponding components of the preceeding node of player II.

At last, we need a counter in each node analogously to the proof of Theorem 4.7. The counter holds the number of occurences of each word $ba \in \{0,1\}^{2n}$. We only count until the boundary k_1 is reached.

Now it is straightforward to define the edges and the winning sets. For the definition of $\Omega(\mathit{Infinite}(S(c)))$ one has to notice that there are three possibilities of building a word $c \in \{0,1\}^{2n}$: as a single extension $b_1 a_1$, as a block extension $b_2 a_2$ or as an extension of b_3 in an update step. Because there is a one-to-one correspondence between the moves in the two games, it is easy to translate strategies from the graph game into the restricted game. ∎

An interesting consequence of the given proof is that if player I has a winning strategy, he actually has a winning strategy which extends every move of player II after a constant amount of time. This is because he only has to await a constant number of player II-moves, until he can build the next input $b_1^t(b_2^t)^m$ for the restricted game.

Corollary 6.2 *The $\forall\exists$-sentences which are uniformly valid in the boolean algebra of recursive sets are decidable.*

Proof sketch: Given an $\forall\exists$-sentence S with matrix F we consider the enumeration game on recursive sets with winning formula F. The sets which are universally quantified in S belong to player II, the sets which are existentially quantified belong to player I. Similar as in Proposition 4.2 one can show that S is uniformly valid in the boolean algebra of recursive sets iff player I has an effective winning strategy. The latter is decidable by Theorem 6.1. ∎

7 Conclusion

There are many more interesting specification languages which remain to be considered. An open problem of Lachlan [Lac70] is whether the enumeration games with predicates $Card_0$ and the set operations $\cap, \cup$ and ^{C} are effectively determined. An enumeration game on partial functions was studied in [Ott95] motivated by a question from inductive inference. Here the effective determinacy result yields a decision procedure for parallel learning [KS94].

In this paper we have presented the notion of enumeration game and clarified the connection with graph games by giving several nontrivial reductions. Enumeration games may be a suitable framework for modelling reactive systems. From the standpoint of computability they offer a rich source for studying effective strategies. In contrast, if recursive games are approached by effectivizing the definition of Borel games, then already in the basic case of recursive winning conditions there may be only non-arithmetical winning strategies [Bla72].

Acknowledgement: We would like to thank Susanne Kaufmann for helpful discussions.

References

[ALW89] Martin Abadi, Leslie Lamport, Pierre Wolper. Realizable and unrealizable specifications of reactive systems. In *Proc. of 16th Int'l Colloquium on Automata, Languages, and Programming*, Lect. Notes in Comput. Sci., Vol. 372, pages 1–17. Springer-Verlag, 1989.

[BL69] J. Richard Büchi, Lawrence H. Landweber. Solving sequential conditions by finite-state strategies. *Trans. Ameri. Math. Soc.*, 138:295–311, 1969.

[Bla72] Andreas Blass. Complexity of winning strategies. *Discrete Mathematics*, 3:295–300, 1972.

[Chu63] Alonzo Church. Logic, arithmetic and automata. In *Proceedings of the International Congress of Mathematicians, August 1962*, pages 23–35, Stockholm, 1963.

[Gur89] Yuri Gurevich. The logic in computer science column: Infinite games. *Bulletin of the European Association for Theoretical Computer Science*, 38:93–100, 1989.

[KS94] Martin Kummer, Frank Stephan. Inclusion problems in parallel learning and games. *Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory, COLT 94*, pages 287–298, ACM Press, 1994.

[Lac70] Alistair H. Lachlan. On some games which are relevant to the theory of recursively enumerable sets. *Ann. of Math.*, 91(2):291–310, 1970.

[McN93] Robert McNaughton. Infinite games played on finite graphs. *Annals of Pure and Applied Logic*, 65:149–184, 1993.

[Mos89] Yiannis N. Moschovakis. A game-theoretic modeling of concurrency. In *Proceedings, Fourth Annual Symposium on Logic in Computer Science*, pages 154–163. IEEE Computer Society Press, 1989.

[MPS95] Oded Maler, Amir Pnueli, Joseph Sifakis. On the synthesis of discrete controllers for timed systems. In *STACS 95*, Lect. Notes in Comput. Sci., Vol. 900, pages 229–242. Springer-Verlag, 1995.

[NY92] Anil Nerode, Alexander Yakhnis. Modelling hybrid systems as games. Technical Report 92-36, Mathematical Sciences Institute, Cornell University, October 1992.

[NYY92] Anil Nerode, Alexander Yakhnis, Vladimir Yakhnis. Concurrent programs as strategies in games. In *Logic from Computer Science: Proceedings of a Workshop held November 13-17, 1989.* Springer-Verlag, 1992.

[Odi89] Piergiorgio Odifreddi. *Classical recursion theory.* North-Holland, Amsterdam, 1989.

[Ott95] Matthias Ott. Strategien in Aufzählungsspielen. Diplomarbeit, Institut für Logik, Komplexität und Deduktionssysteme, Universität Karlsruhe, February 1995.

[PR89] Amir Pnueli, Roni Rosner. On the synthesis of a reactive module. In *Conference Record of the Sixteenth Annual ACM Symposium on Principles of Programming Languages*, pages 179–190, Austin, Texas, 1989.

[Rog67] Hartley Rogers. *Theory of recursive functions and effective computability.* McGraw-Hill, New York, 1967.

[Soa87] Robert I. Soare. *Recursively enumerable sets and degrees.* Perspectives in Mathematical Logic. Springer-Verlag, Berlin, 1987.

[Tho90] Wolfgang Thomas. Automata on infinite objects. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 133–191. Elsevier Science Publishers B. V., 1990.

[Tho95] Wolfgang Thomas. On the synthesis of strategies in infinite games. In *STACS 95*, Lect. Notes in Comput. Sci., Vol. 900, pages 1–13. Springer-Verlag, 1995.

[TW94] J. G. Thistle, W. M. Wonham. Supervision of infinite behaviour of discrete-event systems. *SIAM Journal on Control and Optimization*, 32(4):1098–1113, 1994.

[WD91] Howard Wong-Toi, David L. Dill, Synthesizing processes and schedulers from temporal specifications. In *Computer-Aided Verification '90*, DIMACS Series in Discrete Mathematics and Theoretical Computer Science Volume 3, pages 177–186. American Mathematical Society, 1991.