David Helmbold
Bob Williamson (Eds.)

LNAI 2111

Computational Learning Theory

14th Annual Conference on Computational Learning Theory, COLT 2001 and 5th European Conference on Computational Learning Theory, EuroCOLT 2001
Amsterdam, The Netherlands, July 2001
Proceedings

Springer

David Helmbold Bob Williamson (Eds.)

Computational Learning Theory

14th Annual Conference
on Computational Learning Theory, COLT 2001
and 5th European Conference
on Computational Learning Theory, EuroCOLT 2001
Amsterdam, The Netherlands, July 16-19, 2001
Proceedings

Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Volume Editors

David Helmbold
University of California, Santa Cruz
School of Engineering, Department of Computer Science
Santa Cruz, CA 95064, USA
E-mail: dph@cse.ucsc.edu

Bob Williamson
Australian National University
Research School of Information Sciences and Engineering
Department of Telecommunications Engineering
Canberra 0200, Australia
E-mail: Bob.Williamson@anu.edu.au

Cataloging-in-Publication Data applied for

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

Computational learning theory : proceedings / 14th Annual Conference on Computational Learning Theory, COLT 2001 and 5th European Conference on Computational Learning Theory, EuroCOLT 2001, Amsterdam, The Netherlands, July 16 - 19, 2001. David Helmbold ; Bob Williamson (ed.). - Berlin ; Heidelberg ; New York ; Barcelona ; Hong Kong ; London ; Milan ; Paris ; Singapore ; Tokyo : Springer, 2001
(Lecture notes in computer science ; Vol. 2111 : Lecture notes in artificial intelligence)
ISBN 3-540-42343-5

CR Subject Classification (1998): I.2.6, I.2.3, F.4.1, F.1.1, F.2

ISBN 3-540-42343-5 Springer-Verlag Berlin Heidelberg New York

Springer-Verlag Berlin Heidelberg New York
a member of BertelsmannSpringer Science+Business Media GmbH

http://www.springer.de

Printed in Germany

Typesetting: Camera-ready by author
Printed on acid-free paper SPIN: 10839809 06/3142 5 4 3 2 1 0

Lecture Notes in Artificial Intelligence 2111

Subseries of Lecture Notes in Computer Science
Edited by J. G. Carbonell and J. Siekmann

Lecture Notes in Computer Science
Edited by G. Goos, J. Hartmanis and J. van Leeuwen

Springer
Berlin
Heidelberg
New York
Barcelona
Hong Kong
London
Milan
Paris
Singapore
Tokyo

Preface

This volume contains papers presented at the joint 14th Annual Conference on Computational Learning Theory and 5th European Conference on Computational Learning Theory, held at the Trippenhuis in Amsterdam, The Netherlands from July 16 to 19, 2001.

The technical program contained 40 papers selected from 69 submissions. In addition, David Stork (Ricoh California Research Center) was invited to give an invited lecture and make a written contribution to the proceedings.

The Mark Fulk Award is presented annually for the best paper co-authored by a student. This year's award was won by Olivier Bousquet for the paper "Tracking a Small Set of Modes by Mixing Past Posteriors" (co-authored with Manfred Warmuth).

We gratefully thank all of the individuals and organizations responsible for the success of the conference. We are especially grateful to the program committee: Dana Angluin (Yale), Peter Auer (Univ. of Technology, Graz), Nello Christianini (Royal Holloway), Claudio Gentile (Università di Milano), Lisa Hellerstein (Polytechnic Univ.), Jyrki Kivinen (Univ. of Helsinki), Phil Long (National Univ. of Singapore), Manfred Opper (Aston Univ.), John Shawe-Taylor (Royal Holloway), Yoram Singer (Hebrew Univ.), Bob Sloan (Univ. of Illinois at Chicago), Carl Smith (Univ. of Maryland), Alex Smola (Australian National Univ.), and Frank Stephan (Univ. of Heidelberg), for their efforts in reviewing and selecting the papers in this volume.

Special thanks go to our conference co-chairs, Peter Grünwald and Paul Vitányi, as well as Marja Hegt. Together they handled the conference publicity and all the local arrangements to ensure a successful conference. We would also like to thank ACM SIGACT for the software used in the program committee deliberations and Stephen Kwek for maintaining the COLT web site.

Finally, we would like to thank The National Research Institute for Mathematics and Computer Science in the Netherlands (CWI), The Amsterdam Historical Museum, and The Netherlands Organization for Scientific Research (NWO) for their sponsorship of the conference.

May 2001

David Helmbold
Bob Williamson
Program Co-chairs
COLT/EuroCOLT 2001

Table of Contents

How Many Queries Are Needed to Learn One Bit of Information?*

Hans Ulrich Simon[1]

Fakultät für Mathematik, Ruhr-Universität Bochum, D-44780 Bochum, Germany
simon@lmi.ruhr-uni-bochum.de

Abstract. In this paper we study the question how many queries are needed to "halve a given version space". In other words: how many queries are needed to extract from the learning environment the one bit of information that rules out fifty percent of the concepts which are still candidates for the unknown target concept. We relate this problem to the classical exact learning problem. For instance, we show that lower bounds on the number of queries needed to halve a version space also apply to randomized learners (whereas the classical adversary arguments do not readily apply). Furthermore, we introduce two new combinatorial parameters, the halving dimension and the strong halving dimension, which determine the halving complexity (modulo a small constant factor) for two popular models of query learning: learning by a minimum adequate teacher (equivalence queries combined with membership queries) and learning by counterexamples (equivalence queries alone). These parameters are finally used to characterize the additional power provided by membership queries (compared to the power of equivalence queries alone). All investigations are purely information-theoretic and ignore computational issues.

1 Introduction

The exact learning model was introduced by Angluin in [1]. In this model, a learner A tries to identify an unknown target concept C_* (of the form $C_* : X \rightarrow \{0,1\}$ for a finite set X) by means of queries that must be honestly answered by an oracle. Although the oracle must not lie, it may select its answers in a worstcase fashion such as to slow down the learning process as much as possible. In the (worstcase) analysis of A, we assume that the oracle is indeed an adversary of A that makes full use of this freedom. (In the sequel, we sometimes say "adversary" instead of "oracle" for this reason.) Furthermore, A must be able to identify any target concept selected from a (known) concept class $\mathcal{C}$. Again, A is subjected to a worstcase analysis, i.e., we count the number of queries needed to identify the hardest concept from $\mathcal{C}$ (that is the concept that forces A to invest a maximal number of queries).

Among the most popular query types are the following ones:

* This work has been supported in part by the ESPRIT Working Group in Neural and Computational Learning II, NeuroCOLT2, No. 27150. The author was also supported by the Deutsche Forschungsgemeinschaft Grant SI 498/4-1.

Equivalence Queries A selects a hypothesis H from its hypothesis class $\mathcal{H}$. (Typically, $\mathcal{H} = \mathcal{C}$ or $\mathcal{H}$ is a superset of $\mathcal{C}$.) If $H = C_*$, the oracle answers "YES" (signifying that A suceeded to identify the target concept). Otherwise, the oracle returns a counterexample, i.e., an $x \in X$ such that $H(x) \neq C_*(x)$.

Membership Queries A selects an $x \in X$ and receives label $C_*(x)$ from the oracle.

At each time of the learning process, the so-called version space $\mathcal{V}$ contains all concepts from $\mathcal{C}$ that do not contradict to the answers that have been received so far. Clearly, the learner succeeded to identify C_* as soon as $\mathcal{V} = \{C_*\}$. It is well-known in the learning community that the task of identifying an unknown but fixed target concept from $\mathcal{C}$ is equivalent to the task of playing against another adversary who need not to commit itself to a target concept in the beginning. The answers of this adversary are considered as honest as long as they do not lead to an empty version space. The learner still tries to shrink the version space to a singleton and thereby to issue as few queries as possible. We will refer to this task as the "contraction task" (or the "contraction game"). At first glance, the contraction task seems to give more power to the adversary. However, if we assume that A is deterministic, both tasks require the same number of queries: since A is deterministic, one can "predict" which concept C_* will form the final (singleton) version space in the hardest scenario of the contraction task. Now, it does not hurt the adversary, to commit itself to C_* as the target concept in the beginning.

Since randomized learning complexity will be an issue in this paper, we briefly illustrate that the initial commitment to a target concept is relevant when we allow randomized learners:

Example 1. Consider the model of learning by means of equivalence queries. Let the concept and hypothesis class coincide with the set of all functions from X to $\{0, 1\}$, where $X = \{1, \ldots, d\}$. Clearly, the contraction task forces each deterministic (or randomized) algorithm to issue d equivalence queries because each (non-redundant) query halves the version space. As for deterministic algorithms, the same remark is valid for the learning task. However the following randomized learning algorithm needs in the average only $d/2$ queries:
Pick a first hypothesis $H_0 : X \to \{0, 1\}$ uniformly at random. Given that the current hypothesis is H and that counterexample $x \in X$ is received, let the next hypothesis H' coincide with H on $X \setminus \{x\}$ and set $H'(x) \triangleq 1 - H(x)$.
The number of queries needed to identify an arbitrary (but fixed) target concept C_* equals the number of instances on which C_* and H_0 disagree. This is $d/2$ in the average.

This example demonstrates that the typical adversary arguments, that are used in the literature for proving lower bounds on the number of queries, do not readily apply to randomized learners.[1]

[1] To the best of our knowledge, almost all papers devoted to query learning assume deterministic learning algorithms. A notable exception is the paper [5] of Maass that

The main issue in this paper is the number of queries needed to halve (as opposed to contract) an initial version space $\mathcal{V}_0$. There are several reasons for this kind of research:

- Contraction of the version space by iterated halving is considered as very efficient. Iterated halving is an important building stone of well known strategies such as the "Majority Vote Strategy", for instance. The binary search paradigm is based on halving. Halving may therefore be considered as an interesting problem in its own right.
- Halving the version space yields exactly one bit of information. In this sense, we explore the hardness to extract one bit of information from the learning environment. This sounds like an elementary and natural problem.
- Although the contraction task is not meaningful for randomized learners, we will be able to show that the halving task is meaningful. This makes adversary arguments applicable to randomized learning algorithms.
- We can characterize the halving complexity for two popular query types (equivalence and membership queries) by tight combinatorial bounds (leaving almost no gap).[2] These bounds can be used to characterize the additional power provided by membership queries (compared to the power of equivalence queries alone).

The paper is organized as follows. In Section 2, we present the basic definitions and notations. In Section 3, we view the tasks of learning, contraction and halving as a game between the learner (contraction algorithm, halving algorithm, respectively) and an adversary. In Section 4, we investigate the relation between halving and learning complexity (including randomized learning complexity). In Section 5, we present the combinatorial (lower and upper) bounds on the halving complexity. In Section 6, these bounds are used to characterize the additional power provided by membership queries (compared to the power of equivalence queries alone).

2 Basic Definitions and Notations

Let X be a finite set and $\mathcal{C}, \mathcal{H}$ be two families of functions from X to $\{0,1\}$. In the sequel, we refer to X as the *instance space*, to $\mathcal{C}$ as the *concept class*, and to $\mathcal{H}$ as the *hypothesis class*. It is assumed that $\mathcal{C} \subseteq \mathcal{H}$. A labeled instance $(x, b) \in X \times \{0,1\}$ is called a *sample-point*. A *sample* is a collection of sample-points. For convenience, we represent each sample S as a partially defined binary

demonstrates the significance of supporting examples in the on-line learning model, when the learner is randomized and the learning environment is oblivious.

[2] The derivation of these bounds is based on ideas and results from [4, 2], where bounds on the number of queries needed for learning (i.e, for contracting the version space) are presented. These bounds, however, leave a gap. It seems that the most accurate combinatorial bounds are found on the level of the halving task. (See also [3] for a survey on papers presenting upper and lower bounds on query complexity.)

function over domain X. More formally, S is of the form $S : X \rightarrow \{0, 1, ?\}$, where $S(x) =?$ indicates that S is undefined on instance x. The set

$$\text{supp}(S) \triangleq \{x \in X : S(x) \neq ?\} \tag{1}$$

is called the ***support*** of S. Note that a concept or hypothesis can be viewed as a sample with full support. The ***size*** of S is the number of instances in its support. S' is called ***subsample*** of S, denoted as $S' \sqsubseteq S$, if $\text{supp}(S') \subseteq \text{supp}(S)$ and $S'(x) = S(x)$ for each instance $x \in \text{supp}(S')$. We say that sample S and concept C are ***consistent*** if $S \sqsubseteq C$. We say that S ***has a consistent explanation in*** $\mathcal{C}$ if there exists a concept $C \in \mathcal{C}$ such that S and C are consistent. The terminology for hypotheses is analogous.

In the exact learning model, a *learner (learning algorithm)* A has to identify an unknown *target concept* $C_* \in \mathcal{C}$ by means of queries. The query learning process can be informally described as follows. Each query must be honestly answered by an oracle. Learning proceeds in rounds. In each round, A issues the next query and obtains an honest answer from the oracle. The current ***version space*** $\mathcal{V}$ is the set of concepts from $\mathcal{C}$ that do not contradict to the answers received so far. Initially, $\mathcal{V} = \mathcal{C}$. From round to round, $\mathcal{V}$ shrinks. However, at least the target concept C_* always belongs to $\mathcal{V}$ because the answers given by the oracle are honest. The learning process stops when $\mathcal{V} = \{C_*\}$.

For the sake of simplicity, we formalize this general framework only for the following popular models of exact learning:

Equivalence Query Learning (EQ-Learning) Each allowed query can be identified with a hypothesis $H \in \mathcal{H}$. If $H = C_*$, the only honest answer is "YES" (signifying that the target concept has been exactly identified by A). Otherwise, an honest answer is a *counterexample* to H, i.e., an instance x such that $H(x) \neq C_*(x)$.

Membership Query Learning (MQ-Learning) Each allowed query can be identified with an instance $x \in X$. The only honest answer is $C_*(x)$.

EQ-MQ-Learning The learner may issue both types of queries.

Let $\mathcal{V}$ be the current version space. If A issues a membership query with instance x and receives the binary label b, then the subsequent version space is given by

$$\mathcal{V}[x, b] \triangleq \{C \in \mathcal{V} : C(x) = b\}. \tag{2}$$

Similarly, if A issues an equivalence query with hypothesis H and receives the counterexample x, then the subsequent version space is given by

$$\mathcal{V}[H, x] \triangleq \{C \in \mathcal{V} : C(x) \neq H(x)\}. \tag{3}$$

Clearly, answer "YES" to an equivalence query immediately leads to the final version space $\{C_*\}$.

In general, $\mathcal{V}[Q, R]$ denotes the version space resulting from the current version space $\mathcal{V}$ when A issues query Q and receives answer R. We denote by $\mathcal{Q}$

the set of queries from which Q must be selected. Given $\mathcal{C}, \mathcal{H}$, a collection $\mathcal{Q}$ of allowed queries, and a deterministic learner A, we define $\mathrm{DLC}_A^{\mathcal{Q}}(\mathcal{C}, \mathcal{H})$ as the following unique number q:

- There exists a target concept $C_* \in \mathcal{C}$ and a sequence of honest answers to the queries selected by A such that the learning process does not stop before round q.
- For each target concept $C_* \in \mathcal{C}$ and each sequence of honest answers to the queries selected by A, the learning process stops after round q or earlier.

In other words, $\mathrm{DLC}_A^{\mathcal{Q}}(\mathcal{C}, \mathcal{H})$ is the smallest number q of queries such that A is guaranteed to identify any target concept from $\mathcal{C}$ with hypotheses from $\mathcal{H}$ using q queries from $\mathcal{Q}$. The *deterministic learning complexity* associated with $\mathcal{C}, \mathcal{H}, \mathcal{Q}$ is given by

$$\mathrm{DLC}^{\mathcal{Q}}(\mathcal{C}, \mathcal{H}) \triangleq \min_{A} \mathrm{DLC}_A^{\mathcal{Q}}(\mathcal{C}, \mathcal{H}), \tag{4}$$

where A varies over all deterministic learners.

3 Games Related to Learning

Since we measure the number of queries needed by the learner in a worstcase fashion, we can model the learning process as a game between two players: the learner A and its adversary ADV. We use the notation ADV_A to indicate the strongest possible adversary of A. We begin with a rather straightforward interpretation of exact learning as a game.

3.1 The Learning Game

$\mathcal{C}, \mathcal{H}$ and $\mathcal{Q}$ are fixed and known to both players. The game proceeds as follows. In a first move (invisible to A), ADV picks the target concept C_* from C. Afterwards, both players proceed in rounds. In each round, first player A makes its move by selecting a query from $\mathcal{Q}$. Then, ADV makes its move by selecting an honest answer. The game is over when the current version space does not contain any concept different from C_*. The goal of A is to finish the game as soon as possible, whereas the goal of ADV is to continue playing as long as possible. A is evaluated against the strongest adversary ADV_A that forces A to make a maximum number of moves (or the maximum expected number of moves in the case of a randomized learner).

It should be evident that the number of rounds in the learning game between a deterministic learner A and ADV_A coincides with the quantity $\mathrm{DLC}_A^{\mathcal{Q}}(\mathcal{C}, \mathcal{H})$ that was defined in the previous section. Thus, $\mathrm{DLC}^{\mathcal{Q}}(\mathcal{C}, \mathcal{H})$ coincides with the number of rounds in the learning game between the best deterministic learner and its adversary.

We define $\mathrm{RLC}_A^{\mathcal{Q}}(\mathcal{C}, \mathcal{H})$ as the expected number of rounds in the learning game between the (potentially) randomized learner A and its strongest adversary

ADV_A.[3] The *randomized learning complexity* associated with $\mathcal{C}, \mathcal{H}, \mathcal{Q}$ is given by

$$\mathrm{RLC}^{\mathcal{Q}}(\mathcal{C}, \mathcal{H}) \triangleq \min_{A} \mathrm{RLC}_A^{\mathcal{Q}}(\mathcal{C}, \mathcal{H}), \tag{5}$$

where A varies over all (potentially) randomized learners.

3.2 The Contraction Game

It is well known that, in the case of deterministic learners A, the learning game can be replaced by a conceptually simpler game, differing from the learning game as follows:

- The first move of ADV is omitted, i.e., ADV makes no commitment about the target concept in the beginning.
- Each (syntactically correct)[4] answer that does not lead to an empty version space is honest.
- The game is over when the version space is a singleton.

Again, the goal of player A is to finish the game as soon as possible, whereas the goal of the adversary is to finish as late as possible. A is evaluated against its strongest adversary ADV_A. We will refer to this new game as the contraction game and to A as a contraction algorithm.

The following lemmas recall some well known facts (in a slightly more general setting).

Lemma 1. *As for the contraction game, there exist two deterministic optimal players A_* and ADV_*, i.e., the following holds:*

1. *Let A be any (potentially randomized) contraction algorithm. Then, ADV_* forces A to make at least as many moves as A_*.*
2. *Let ADV be any (potentially randomized) oracle. Then A_* needs no more moves against ADV than against ADV_*.*

The proof uses a standard argument which is given here for sake of completeness.

Proof. Consider the decision tree T that models the moves of both players. Each node of T is of type either $\mathcal{Q}$ or $\mathcal{R}$ (signifying which player makes the next move). Each node of type $\mathcal{Q}$ is marked by a version space (reflecting the actual configuration of the contraction game), and each node of type $\mathcal{R}$ is marked by a version space and a question (again reflecting the actual configuration of the game including the last question of A). The structure of T can be inductively described as follows:

[3] Here, ADV_A knows the program of A, but A determines its next move by means of secret random bits. Thus, ADV_A knows the probability distribution of the future moves, but cannot exactly predict them. This corresponds to what is called "weakly oblivious environment" in [5].

[4] E.g., the answer to an equivalence (or membership) query must be an instance from X (or a binary label, respectively).

- Its root is of type $\mathcal{Q}$ and marked $\mathcal{C}$ (the initial version space in the contraction game).
- A node of type $\mathcal{Q}$ that is marked by a singleton (version space of size 1) is a leaf (signifying that the game is over).
- Each inner node v of type $\mathcal{Q}$ that is marked $\mathcal{V}$ has k children $v[Q_1], \ldots, v[Q_k]$, where $Q_1, \ldots, Q_k$ denote the non-redundant questions that A is allowed to issue at this stage. Node $v[Q_i]$ is of type $\mathcal{R}$ and marked $(\mathcal{V}, Q_i)$.
- Each inner node w of type $\mathcal{R}$ that is marked $(\mathcal{V}, Q)$ has l children $w[R_1], \ldots,$ $w[R_l]$, where $R_1, \ldots, R_l$ denote the honest answers of ADV to question Q at this stage. Node $w[R_j]$ is of type $\mathcal{Q}$ and marked $\mathcal{V}[Q, R_j]$ (the version space resulting from $\mathcal{V}$ when A issues query Q and receives answer R_j).

It is easy to describe deterministic optimal strategies for both players in a bottom-up fashion. At each node of T, the optimal decisions for A and ADV result from the following rules:

- Each leaf is labeled 0 (signifying that no more moves of A are needed to finish the game).
- If a node w of type $\mathcal{R}$ that is labeled $(\mathcal{V}, Q)$ has children $w[R_1], \ldots, w[R_l]$ labeled $(n_1, \ldots, n_l)$, respectively, and $n_j = \max\{n_1, \ldots, n_l\}$, then w is labeled n_j. Furthermore, ADV should answer R_j to question Q, given that $\mathcal{V}$ is the current version space.
- If a node v of type $\mathcal{Q}$ that is labeled $\mathcal{V}$ has children $v[Q_1], \ldots, v[Q_k]$ labeled $(m_1, \ldots, m_k)$, respectively, and $m_i = \min\{m_1, \ldots, m_k\}$, then v is labeled $1 + m_i$. Furthermore, A should ask question Q_i, given that $\mathcal{V}$ is the current version space.

Note that these rules can be made deterministic by resolving the ties in an arbitrary deterministic fashion. It is easy to prove for each node v of T (by induction on the height of v) that the following holds:
If v is marked $\mathcal{V}$, then the rules specify two deterministic optimal players (in the sense of Lemma 1) for the partial contraction game that starts with initial version space $\mathcal{V}$. The bottom-up label associated with v specifies the number of rounds in this partial game when both player follow the rules.
The extrapolation of this claim to the root node of T yields Lemma 1.

Lemma 1 implies that A_* is the best contraction algorithm among all (possibly randomized) algorithms. (Remember that each algorithm A is evaluated against its strongest adversary ADV_A.) It implies also that ADV_* is the strongest adversary of A_*.

Lemma 2. *$DLC^{\mathcal{Q}}(\mathcal{C}, \mathcal{H})$ coincides with the number of rounds, say q_*, in the contraction game between A_* and ADV_*.*

The proof of this lemma (given here for sake of completeness) is well known in the learning community and is, in fact, the justification of the popular adversary arguments within the derivation of lower bounds on the number of queries needed by deterministic learners.

Proof. The contraction game coincides with the learning game, except for the commitment that the adversary has to make in the first step of the learning game: the selection of a target concept $C_* \in \mathcal{C}$. Thus, $\mathrm{DLC}^{\mathcal{Q}}(\mathcal{C},\mathcal{H}) \le q^*$. It suffices therefore to show that for each deterministic learner A there exists an adversary ADV_A that forces at least q_* moves of A.

To this end, let A be an arbitrary, but fixed, deterministic learner. Let ADV_* be the optimal deterministic adversary in the contraction game that was described in the proof of Lemma 1. Let A play against ADV_* in the contraction game.[5] Let $q \ge q_*$ be the number of queries needed by A to finish the contraction game against player ADV_*, and let C_* be the unique concept in the final (singleton) version space. Now we may use an adversary ADV_A in the learning game that selects C_* as a target concept in the beginning and then simulates ADV_*. Since A is deterministic, this will lead to the same sequence of moves as in the contraction game. Thus, ADV can force $q \ge q^*$ moves of A.

Note that a lower bound argument can deal with a sub-optimal (but, may be, easier to analyze) adversary ADV (instead of ADV_*). Symmetrically, an upper bound argument may use a sub-optimal (but, may be, easier to analyze) contraction algorithm A (instead of A_*).

We briefly remind the reader to Example 1. If $\mathcal{C}$ contains all functions from $\{1,\dots,d\}$ to $\{0,1\}$, then Example 1 shows that

$$\mathrm{DLC}^{EQ}(\mathcal{C},\mathcal{C}) = d \text{ and } \mathrm{RLC}^{EQ}(\mathcal{C},\mathcal{C}) \le d/2.$$

In the light of Lemmas 1 and 2, this demonstrates that the contraction game does not model the learning game when randomized learners are allowed.

3.3 The Halving Game

The halving game is defined like the contraction game except that it may start with an arbitrary initial version space $\mathcal{V}_0 \subseteq \mathcal{C}$ (known to both players), and it is over as soon as the current version space $\mathcal{V}$ contains at most half of the concepts of $\mathcal{V}_0$. Player A (called halving algorithm in this context) tries to halve $\mathcal{V}_0$ as fast as possible. Player ADV_A is its strongest adversary.

Like in the contraction game, there exist two optimal deterministic players: A_* (representing the optimal halving algorithm) and ADV_* (wich is also the strongest adversary for A_*). (Compare with Lemma 1.) Let $\mathrm{HC}^{\mathcal{Q}}(\mathcal{V}_0,\mathcal{H})$ be defined as the number of rounds in the halving game between A_* and ADV_*. In other words, $\mathrm{HC}^{\mathcal{Q}}(\mathcal{V}_0,\mathcal{H})$ is the smallest number of queries that suffices to halve the initial version space $\mathcal{V}_0$ when all queries are answered in a worstcase fashion. This parameter has the disadvantage of being not monotonic: a subset of $\mathcal{V}_0$ might be harder to halve than $\mathcal{V}_0$ itself. In order to force monotonicity, we define the *halving complexity* associated with $\mathcal{C},\mathcal{H},\mathcal{Q}$ as follows:

$$\mathrm{HC}^{\mathcal{Q}}_*(\mathcal{C},\mathcal{H}) = \max\{\mathrm{HC}^{\mathcal{Q}}(\mathcal{V},\mathcal{H}) : \mathcal{V} \subseteq \mathcal{C}\} \tag{6}$$

[5] This looks like a dirty trick because A is an algorithm that expects to play the learning game. We will however argue later that A cannot distinguish the communication with ADV_* from the communication with an adversary ADV in the learning game.