From: AAAI-02 Proceedings. Copyright © 2002, AAAI (www.aaai.org). All rights reserved.

Specific-to-General Learning for Temporal Events*

Alan Fern and **Robert Givan** and **Jeffrey Mark Siskind**
Electrical and Computer Engineering, Purdue University, West Lafayette IN 47907 USA
{afern,givan,qobi}@purdue.edu

Abstract

We study the problem of supervised learning of event classes in a simple temporal event-description language. We give lower and upper bounds and algorithms for the subsumption and generalization problems for two expressively powerful subsets of this logic, and present a positive-examples-only specific-to-general learning method based on the resulting algorithms. We also present a polynomial-time computable "syntactic" subsumption test that implies semantic subsumption without being equivalent to it. A generalization algorithm based on syntactic subsumption can be used in place of semantic generalization to improve the asymptotic complexity of the resulting learning algorithm. A companion paper shows that our methods can be applied to duplicate the performance of human-coded concepts in the substantial application domain of video event recognition.

Introduction

In many domains, interesting concepts take the form of structured temporal sequences of events. These domains include: planning, where macro-actions represent useful temporal patterns; computer security, where typical application behavior as temporal patterns of system calls must be differentiated from compromised application behavior (and likewise authorized user behavior from intrusive behavior); and event recognition in video sequences, where the structure of behaviors shown in videos can be automatically recognized.

Many proposed representation languages can be used to capture temporal structure. These include standard first-order logic, Allen's interval calculus (Allen 1983), and various temporal logics (Clarke, Emerson, & Sistla 1983; Allen & Ferguson 1994; Bacchus & Kabanza 2000). In this work, we study a simple temporal language that is a subset of many of these languages—we restrict our learner to concepts expressible with conjunction and temporal sequencing of consecutive intervals (called "Until" in linear temporal logic, or LTL). Our restricted language is a sublanguage of LTL and of the temporal event logic of Siskind (2001) as well as of standard first-order logic (somewhat painfully).

Motivation for the choice of this subset is developed throughout the paper, and includes useful learning bias, efficient computation, and expressive power. Here we construct and analyze a specific-to-general learner for this subset. This paper contains theoretical results on the algorithmic problems (concept subsumption and generalization) involved in constructing such a learner: we give algorithms along with lower and upper asymptotic complexity bounds. Along the way, we expose considerable structure in the language.

In (Fern, Siskind, & Givan 2002), we develop practical adaptations of these techniques for a substantial application (including a novel automatic conversion of relational data to propositional data), and show that our learner can replicate the performance of carefully hand-constructed definitions in recognizing events in video sequences.

Bottom-up Learning from Positive Data

The sequence-mining literature contains many general-to-specific ("levelwise") algorithms for finding frequent sequences (Agrawal & Srikant 1995; Mannila, Toivonen, & Verkamo 1995; Kam & Fu 2000; Cohen 2001; Hoppner 2001). Here we explore a specific-to-general approach.

Inductive logic programming (ILP) (Muggleton & De Raedt 1994) has explored positive-only approaches in systems that can be applied to our problem, including Golem (Muggleton & Feng 1992), Claudien (De Raedt & Dehaspe 1997), and Progol (Muggleton 1995). Our early experiments confirmed our belief that horn clauses, lacking special handling of time, give a poor inductive bias[1].

Here we present and analyze a specific-to-general positive-only learner for temporal events—our learning algorithm is only given positive training examples (where the event occurs) and is not given negative examples (where the event does not occur). The positive-only setting is of interest as it appears that humans are able to learn many event definitions given primarily or only positive examples. From a practical standpoint, a positive-only learner removes the often difficult task of collecting negative examples that are "representative" of what is not the event to be learned.

*This work was supported in part by NSF grants 9977981-IIS and 0093100-IIS, an NSF Graduate Fellowship for Fern, and the Center for Education and Research in Information Assurance and Security at Purdue University. Part of this work was performed while Siskind was at NEC Research Institute, Inc.

Copyright © 2002, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

[1] A reasonable alternative approach to ours would be to add syntactic biases (Cohen 1994; Dehaspe & De Raedt 1996) to ILP.

A typical learning domain specifies an example space (the objects we wish to classify) and a concept language (formulas that represent sets of examples that they *cover*). Generally we say a concept C_1 is more general (less specific) than C_2 iff C_2 is a subset of C_1—alternatively, a generality relation that may not be equivalent to subset may be specified, often for computational reasons. Achieving the goal of finding a concept consistent with a set of positive-only training data generally results in a trivial solution (simply return the most general concept in the language). To avoid adding negative training data, it is common to specify the learning goal as finding the least-general concept that covers all of the data[2]. With enough data and an appropriate concept language, the least-general concept often converges usefully.

We take a standard specific-to-general machine-learning approach to finding the least-general concept covering a set of positive examples. Assume we have a concept language L and an example space S. The approach relies on the computation of two quantities: the least-general covering formula (LGCF) of an example and the least-general generalization (LGG) of a set of formulas. An LGCF in L of an example in S is a formula in L that covers the example such that no other covering formula is strictly less general. Intuitively, the LGCF of an example is the "most representative" formula in L of that example. An LGG of any subset of L is a formula more general than each formula in the subset and not strictly more general than any other such formula.

Given the existence and uniqueness (up to set equivalence) of the LGCF and LGG (which is non-trivial to show for some concept languages) the specific-to-general approach proceeds by: 1) Use the LGCF to transform each positive training instance into a formula of L, and 2) Return the LGG of the resulting formulas. The returned formula represents the least-general concept in L that covers all the positive training examples. This learning approach has been pursued for a variety of concept languages including, clausal first-order logic (Plotkin 1971), definite clauses (Muggleton & Feng 1992), and description logic (Cohen & Hirsh 1994). It is important to choose an appropriate concept language as a bias for this learning approach or the concept returned will be (or resemble) the disjunction of the training data.

In this work, our concept language is the AMA temporal event logic presented below and the example space is the set of all models of that logic. Intuitively, a training example depicts a model where a target event occurs. (The models can be thought of as movies.) We will consider two notions of generality for AMA concepts and, under both notions, study the properties and computation of the LGCF and LGG.

AMA Syntax and Semantics

We study a subset of an interval-based logic called event logic developed by Siskind (2001) for event recognition in video sequences. This logic is "interval-based" in explicitly representing each of the possible interval relationships given originally by Allen (1983) in his calculus of interval relations (e.g., "overlaps", "meets", "during"). Event logic allows the definition of static properties of intervals directly and dynamic properties by hierarchically relating subintervals using the Allen interval relations.

Here we restrict our attention to a subset of event logic we call AMA, defined below. We believe that our choice of event logic rather than first-order logic, as well as our restriction to AMA, provide a useful learning bias by ruling out a large number of 'practically useless' concepts while maintaining substantial expressive power. The practical utility of this bias is shown in our companion paper (Fern, Siskind, & Givan 2002). Our choice can also be seen as a restriction of LTL to conjunction and "Until", with similar motivations.

It is natural to describe temporal events by specifying a sequence of properties that must hold consecutively; e.g., "a hand picking up a block" might become "the block is not supported by the hand and then the block is supported by the hand." We represent such sequences with *MA timelines*[3], which are sequences of conjunctive state restrictions. Intuitively, an MA timeline represents the events that temporally match the sequence of consecutive conjunctions. An AMA formula is then the conjunction of a number of MA timelines, representing events that can be simultaneously viewed as satisfying each conjoined timeline. Formally,

$$\begin{array}{rcll} \textit{state} & ::= & \mathtt{true} \mid \textit{prop} \mid \textit{prop} \wedge \textit{state} & \\ \textit{MA} & ::= & (\textit{state}) \mid (\textit{state}); \textit{MA} & \textit{// may omit parens} \\ \textit{AMA} & ::= & \textit{MA} \mid \textit{MA} \wedge \textit{AMA} & \end{array}$$

where *prop* is a primitive proposition. We often treat states as proposition sets (with $\mathtt{true}$ the empty set), MA formulas as state sets[4], and AMA formulas as MA timeline sets.

A temporal model $\mathcal{M} = \langle \mathcal{M}, \mathcal{I} \rangle$ over the set of propositions PROP is a pair of a mapping M from the natural numbers (representing time) to truth assignments to PROP, and a closed natural number interval I. The natural numbers in the domain of M represent time discretely, but there is no prescribed unit of continuous time alloted to each number. Instead, each number represents an arbitrarily long period of continuous time during which nothing changed. Similarly, states in MA timelines represent arbitrarily long periods of time during which the conjunctive state restrictions hold.[5]

- A state s is satisfied by model $\langle M, I \rangle$ iff $M[x]$ assigns P true for every $x \in I$ and $P \in s$.
- An MA timeline $s_1; s_2; \cdots; s_n$ is satisfied by a model $\langle M, [t, t'] \rangle$ iff $\exists t'' \in [t, t']$ s. t. $\langle M, [t, t''] \rangle$ satisfies s_1, and either $\langle M, [t'', t'] \rangle$ or $\langle M, [t''+1, t'] \rangle$ satisfies $s_2; \cdots; s_n$.
- An AMA formula $\Phi_1 \wedge \Phi_2 \wedge \ldots \wedge \Phi_n$ is satisfied by $\mathcal{M}$ iff each Φ_i is satisfied by $\mathcal{M}$.

[2]In some cases, there can be more than one such least-general concept. The set of all such concepts is called the "specific boundary of the version space" (Mitchell 1982).

[3]MA stands for "Meets/And", an MA timeline being the "Meet" of a sequence of conjunctively restricted intervals.

[4]Timelines may contain duplicate states, and the duplication can be significant. For this reason, when treating timelines as sets of states, we formally intend sets of *state-index pairs*. We do not indicate this explicitly to avoid encumbering our notation, but this must be remembered whenever handling duplicate states.

[5]We note that Siskind (2001) gives a continuous-time semantics for event logic for which our results below also hold.

The condition defining satisfaction for MA timelines may appear unintuitive, as there are two ways that $s_2; \cdots; s_n$ can be satisfied. Recall that we are using the natural numbers to represent arbitrary static continuous time intervals. The transition between consecutive states s_i and s_{i+1} can occur either within a static interval (i.e., one of constant truth assignment), that happens to satisfy both states, or exactly at the boundary of two static time intervals. In the above definition, these cases correspond to $s_2; \cdots; s_n$ being satisfied during the time intervals $[t'', t']$ and $[t''+1, t']$, respectively.

When $\mathcal{M}$ satisfies Φ we say $\mathcal{M}$ is a model of Φ. We say AMA Ψ_1 *subsumes* AMA Ψ_2 iff every model of Ψ_2 is a model of Ψ_1, written $\Psi_2 \leq \Psi_1$, and Ψ_1 *properly subsumes* Ψ_2 when, in addition, $\Psi_1 \not\leq \Psi_2$. We may also say Ψ_1 is *more general (or less specific) than* Ψ_2 or that Ψ_1 *covers* Ψ_2. Siskind (2001) provides a method to determine whether a given model satisfies a given AMA formula. We now give two illustrative examples.

Example 1. *(Stretchability) The MA timelines* $S_1; S_2; S_3$, $S_1; S_2; S_2; S_2; S_3$, *and* $S_1; S_1; S_2; S_3; S_3$ *are all equivalent. In general, MA timelines have the property that duplicating any state results in an equivalent formula. Given a model* $\langle M, I\rangle$*, we view each* $M[x]$ *as a continuous time-interval that can be divided into an arbitrary number of subintervals. So, if state* S *is satisfied by* $\langle M, [x, x]\rangle$*, then so is the sequence* $S; S; \cdots; S$.

Example 2. *(Infinite Descending Chains) Given propositions* A *and* B*, the MA timeline* $\Phi = (A \wedge B)$ *is subsumed by each of* $A; B$, $A; B; A; B$, $A; B; A; B; A; B$, $\ldots$. *This is clear from a continuous-time perspective, as an interval where* A *and* B *are true can always be broken into subintervals where both* A *and* B *hold—any AMA formula over only* A *and* B *will subsume* Φ*. This example illustrates that there are infinite descending chains of AMA formulas that each properly subsume a given formula.*

Motivation for AMA MA timelines are a very natural way to capture "stretchable" sequences of state constraints. But why consider the conjunction of such sequences, i.e., AMA? We have several reasons for this language enrichment. First of all, we show below that the AMA LGG is unique; this is not true for MA. Second, and more informally, we argue that parallel conjunctive constraints can be important to learning efficiency. In particular, the space of MA formulas of length k grows in size exponentially with k, making it difficult to induce long MA formulas. However, finding several shorter MA timelines that each characterize *part* of a long sequence of changes is exponentially easier. (At least, the space to search is exponentially smaller.) The AMA conjunction of these timelines places these shorter constraints simultaneously and often captures a great deal of the concept structure. For this reason, we analyze AMA as well as MA and, in our empirical companion paper, we bound the length k of the timelines considered.

Our language, analysis, and learning methods here are described for a propositional setting. However, in a companion paper (Fern, Siskind, & Givan 2002) we show how to adapt these methods to a substantial empirical domain (video event recognition) that requires relational concepts. There, we convert relational training data to propositional training data using an automatically extracted object correspondence between examples and then universally generalizing the resulting learned concepts. This approach is somewhat distinctive (compare (Lavrac, Dzeroski, & Grobelnik 1991; Roth & Yih 2001)). The empirical domain presented there also requires extending our methods here to allow states to assert proposition negations and to control the exponential growth of concept size with a restricted-hypothesis-space bias to small concepts (by bounding MA timeline length).

AMA formulas can be translated to first-order clauses, but it is not straightforward to then use existing clausal generalization techniques for learning. In particular, to capture the AMA semantics in clauses, it appears necessary to define subsumption and generalization relative to a background theory that restricts us to a "continuous-time" first-order–model space. In general, least-general generalizations relative to background theories need not exist (Plotkin 1971), so clausal generalization does not simply subsume our results.

Basic Concepts and Properties of AMA

We use the following conventions: "propositions" and "theorems" are the key results of our work, with theorems being those results of the most difficulty, and "lemmas" are technical results needed for the later proofs of propositions or theorems. We number the results in one sequence. Complete proofs are available in the full paper.

Least-General Covering Formula. A logic can discriminate two models if it contains a formula that satisfies one but not the other. It turns out AMA formulas can discriminate two models exactly when *internal positive* event logic formulas can do so. Internal positive formulas are those that define event occurrence only in terms of positive (non-negated) properties within the defining interval (i.e., satisfaction by $\langle M, I\rangle$ depends only on the proposition truth values given by M inside the interval I). This fact indicates that our restriction to AMA formulas retains substantial expressiveness and leads to the following result that serves as the least-general covering formula (LGCF) component of our learning procedure. The *MA-projection* of a model $\mathcal{M} = \langle M, [i, j]\rangle$ is an MA timeline $s_0; s_1; \cdots; s_{j-i}$ where state s_k gives the true propositions in $M(i+k)$ for $0 \leq k \leq j-i$.

Proposition 1. *The MA-projection of a model is its LGCF for internal positive event logic (and hence for AMA), up to semantic equivalence.*

Proposition 1 tells us that the LGCF of a model exists, is unique, and is an MA timeline. Given this property, when a formula Ψ covers all the MA timelines covered by another formula Ψ', we have $\Psi' \leq \Psi$. Proposition 1 also tells us that we can compute the LGCF of a model by constructing the MA-projection of that model—it is straightforward to do this in time polynomial in the size of the model.

Subsumption and Generalization for States. A state S_1 subsumes S_2 iff S_1 is a subset of S_2, viewing states as sets of propositions. From this we derive that the intersection of

states is the least-general subsumer of those states and that the union of states is likewise the most general subsumee.

Interdigitations. Given a set of MA timelines, we need to consider the different ways in which a model could simultaneously satisfy the timelines in the set. At the start of such a model, the initial state from each timeline must be satisfied. At some point, one or more of the timelines can transition so that the second state in those timelines must be satisfied in place of the initial state, while the initial state of the other timelines remains satisfied. After a sequence of such transitions in subsets of the timelines, the final state of each timeline holds. Each way of choosing the transition sequence constitutes a different "interdigitation" of the timelines.

Alternatively viewed, each model simultaneously satisfying the timelines induces a *co-occurrence relation* on tuples of timeline states, one from each timeline, identifying which tuples co-occur at some point in the model. We represent this concept formally as a set of tuples of co-occurring states that can be ordered by the sequence of transitions. Intuitively, the tuples in an interdigitation represent the maximal time intervals over which no MA timeline has a transition, giving the co-occurring states for each such time interval.

A relation R on $X_1 \times \cdots \times X_n$ is *simultaneously consistent* with orderings $\leq_1, \ldots, \leq_n$, if, whenever $R(x_1, \ldots, x_n)$ and $R(x'_1, \ldots, x'_n)$, either $x_i \leq_i x'_i$, for all i, or $x'_i \leq_i x_i$, for all i. We say R is *piecewise total* if the projection of R onto each component is total (i.e., every state in any X_i appears in R).

Definition 1. *An interdigitation I of a set of MA timelines $\{\Phi_1, \ldots, \Phi_n\}$ is a* co-occurrence *relation over $\Phi_1 \times \cdots \times \Phi_n$ (viewing timelines as sets of states) that is piecewise total, and simultaneously consistent with the state orderings of the Φ_i. We say that two states $s \in \Phi_i$ and $s' \in \Phi_j$ for $i \neq j$ co-occur in I iff some tuple of I contains both s and s'. We sometimes refer to I as a sequence of tuples, meaning the sequence lexicographically ordered by the Φ_i state orderings.*

There are exponentially many interdigitations of even two MA timelines (relative to the timeline lengths). Figure 1 shows an interdigitation of two MA timelines.

We first use interdigitations to syntactically characterize subsumption between MA timelines. An interdigitation I of two MA timelines Φ_1 and Φ_2 is a *witness* to $\Phi_1 \leq \Phi_2$ if, for every pair of co-occuring states $s_1 \in \Phi_1$ and $s_2 \in \Phi_2$, we have $s_1 \leq s_2$. Below we establish the equivalence between witnessing interdigitations and MA subsumption.

Proposition 2. *For MA timelines Φ_1 and Φ_2, $\Phi_1 \leq \Phi_2$ iff there is an interdigitation that witnesses $\Phi_1 \leq \Phi_2$.*

IS(·) and IG(·). Interdigitations are useful in analyzing both conjunctions and disjunctions of MA timelines. When conjoining timelines, all states that co-occur in an interdigitation must simultaneously hold at some point, so that viewed as sets, the union of the co-occuring states must hold. A sequence of such unions that must hold to force the conjunction of timelines to hold (via some interdigitation) is called an "interdigitation specialization" of the timelines. Dually, an "interdigitation generalization" involving intersections of states upper bounds the disjunction of timelines.

Suppose s_1, s_2, s_3, t_1, t_2, and t_3 are each sets of propositions (i.e., states). Consider the timelines $S = s_1; s_2; s_3$ and $T = t_1; t_2; t_3$. The relation

$$\{\ \langle s_1, t_1 \rangle, \langle s_2, t_1 \rangle, \langle s_3, t_2 \rangle, \langle s_3, t_3 \rangle\ \}$$

is an interdigitation of S and T in which states s_1 and s_2 co-occur with t_1, and s_3 co-occurs with t_2 and t_3. The corresponding IG and IS members are

$$\begin{array}{l} s_1 \cap t_1;\ s_2 \cap t_1;\ s_3 \cap t_2;\ s_3 \cap t_3\ \in\ \mathrm{IG}(\{S, T\}) \\ s_1 \cup t_1;\ s_2 \cup t_1;\ s_3 \cup t_2;\ s_3 \cup t_3\ \in\ \mathrm{IS}(\{S, T\}). \end{array}$$

If $t_1 \subseteq s_1, t_1 \subseteq s_2, t_2 \subseteq s_3$, and $t_3 \subseteq s_3$, then the interdigitation *witnesses* $S \leq T$.

Figure 1: An interdigitation with IG and IS members.

Definition 2. *An interdigitation generalization (specialization) of a set Σ of MA timelines is an MA timeline $s_1; \ldots; s_m$, such that, for some interdigitation I of Σ with m tuples, s_j is the intersection (respectively, union) of the components of the j'th tuple of the sequence I. The set of interdigitation generalizations (respectively, specializations) of Σ is called* $\mathrm{IG}(\Sigma)$ *(respectively,* $\mathrm{IS}(\Sigma)$*).*

Each timeline in $\mathrm{IG}(\Sigma)$ (dually, $\mathrm{IS}(\Sigma)$) subsumes (is subsumed by) each timeline in Σ. For our complexity analyses, we note that the number of states in any member of $\mathrm{IG}(C)$ or $\mathrm{IS}(C)$ is lower-bounded by the number of states in any of the MA timelines in C and is upper-bounded by the total number of states in all the MA timelines in C. The number of interdigitations of C, and thus of members of $\mathrm{IG}(C)$ or $\mathrm{IS}(C)$, is exponential in that same total number of states.

We now give a useful lemma and a proposition concerning the relationships between conjunctions and disjunctions of MA concepts (the former being AMA concepts). For convenience here, we use disjunction on MA concepts, producing formulas outside of AMA with the obvious interpretation.

Lemma 3. *Given an MA formula Φ that subsumes each member of a set Σ of MA formulas, some $\Phi' \in \mathrm{IG}(\Sigma)$ is subsumed by Φ. Dually, when Φ is subsumed by each member of Σ, some $\Phi' \in \mathrm{IS}(\Sigma)$ subsumes Φ. In each case, the length of Φ' can be bounded by the size of Σ.*

Proof: (Sketch) Construct a witnessing interdigitation for the subsumption of each member of Σ by Φ. Combine these interdigitations I_Σ to form an interdigitation I of $\Sigma \cup \{\Phi\}$ such that any state s in Φ co-occurs with a state s' only if s and s' co-occur in some interdigitation in I_Σ. "Project" I to an interdigitation of Σ and form the corresponding member Φ' of $\mathrm{IG}(\Sigma)$. Careful analysis shows $\Phi' \leq \Phi$ with the desired size bound. The dual is argued similarly. □

Proposition 4. *The following hold:*

1. *(and-to-or) The conjunction of a set Σ of MA timelines is equal to the disjunction of the timelines in* $\mathrm{IS}(\Sigma)$.
2. *(or-to-and) The disjunction of a set Σ of MA timelines is subsumed by the conjunction of the timelines in* $\mathrm{IG}(\Sigma)$.

Proof: $(\bigvee \mathrm{IS}(\Sigma)) \leq (\bigwedge \Sigma)$ and $(\bigvee \Sigma) \leq (\bigwedge \mathrm{IG}(\Sigma))$ are

straightforward. $(\bigwedge \Sigma) \leq (\bigvee \mathrm{IS}(\Sigma))$ follows from Lemma 3 by considering any timeline covered by $(\bigwedge \Sigma)$. $\square$

Using "and-to-or", we can now reduce AMA subsumption to MA subsumption, with an exponential size increase.

Proposition 5. *For AMA Ψ_1 and Ψ_2, $(\Psi_1 \leq \Psi_2)$ iff for all $\Phi_1 \in \mathrm{IS}(\Psi_1)$ and $\Phi_2 \in \Psi_2, \Phi_1 \leq \Phi_2$*

Subsumption and Generalization

We give algorithms and complexity bounds for the construction of least-general generalization (LGG) formulas based on an analysis of subsumption. We give a polynomial-time algorithm for deciding subsumption between MA formulas. We show that subsumption for AMA formulas is coNP-complete. We give existence, uniqueness, lower/upper bounds, and an algorithm for the LGG on AMA formulas. Finally, we give a syntactic notion of subsumption and an algorithm that computes the corresponding syntactic LGG that is exponentially faster than our semantic LGG algorithm.

Subsumption. Our methods rely on a novel algorithm for deciding the subsumption question $\Phi_1 \leq \Phi_2$ between MA formulas Φ_1 and Φ_2 in polynomial-time. Merely searching for a witnessing interdigitation of Φ_1 and Φ_2 provides an obvious decision procedure for the subsumption question—however, there are exponentially many such interdigitations. We reduce this problem to the polynomial-time operation of finding a path in a graph on pairs of states in $\Phi_1 \times \Phi_2$.

Theorem 6. *Given MA timelines Φ_1 and Φ_2, we can check in polynomial time whether $\Phi_1 \leq \Phi_2$.*

Proof: (Sketch) Write Φ_1 as $s_1, \ldots, s_m$ and Φ_2 as $t_1, \ldots, t_n$. Consider a directed graph with vertices V the set $\{v_{i,j} \mid 1 \leq i \leq m, 1 \leq j \leq n\}$. Let the (directed) edges E be the set of all $\langle v_{i,j}, v_{i',j'} \rangle$ such that $s_i \leq t_j$, $s_{i'} \leq t_{j'}$, and both $i \leq i' \leq i+1$ and $j \leq j' \leq j+1$. One can show that $\Phi_1 \leq \Phi_2$ iff there is a path in $\langle V, E \rangle$ from $v_{1,1}$ to $v_{m,n}$. Paths here correspond to witnessing interdigitations. $\square$

A polynomial-time MA-subsumption tester can be built by constructing the graph described in this proof and employing any polynomial-time path-finding method. Given this polynomial-time algorithm for MA subsumption, Proposition 5 immediately suggests an exponential-time algorithm for deciding AMA subsumption—by computing MA subsumption between the exponentially many IS timelines of one formula and the timelines of the other formula. The following theorem tells us that, unless $P = NP$, we cannot do any better than this in the worst case.

Theorem 7. *Deciding AMA subsumption is coNP-complete.*

Proof: (sketch) AMA-subsumption of Ψ_1 by Ψ_2 is in coNP because there are polynomially checkable certificates to non-subsumption. In particular, there is a member Φ_1 of $\mathrm{IS}(\Psi_1)$ that is not subsumed by some member of Ψ_2, which can be checked using MA-subsumption in polynomial time.

We reduce the problem of deciding the satisfiability of a 3-SAT formula $S = C_1 \wedge \cdots \wedge C_m$ to the problem of recognizing non-subsumption between AMA formulas. Here, each C_i is $(l_{i,1} \vee l_{i,2} \vee l_{i,3})$ and each $l_{i,j}$ either a proposition P chosen from $P_1, \ldots, P_n$ or its negation $\neg P$. The idea of the reduction is to view members of $\mathrm{IS}(\Psi_1)$ as representing truth assignments. We exploit the fact that all interdigitation specializations of $X; Y$ and $Y; X$ will be subsumed by either X or Y—this yields a binary choice that can represent a proposition truth value, except that there will be an interdigitation that "sets" the proposition to both true and false.

Let Q be the set of propositions

$$\{\mathrm{True}_k \mid 1 \leq k \leq n\} \cup \{\mathrm{False}_k \mid 1 \leq k \leq n\},$$

and let Ψ_1 be the conjunction of the timelines

$$\bigcup_{i=1}^{n} \{(Q; \mathrm{True}_i; \mathrm{False}_i; Q), (Q; \mathrm{False}_i; \mathrm{True}_i; Q)\}.$$

Each member of $\mathrm{IS}(\Psi_1)$ will be subsumed by either True_i or False_i for each i, and thus "represent" at least one truth assignment. Let Ψ_2 be the formula $s_1; \ldots; s_m$, where

$$\begin{aligned} s_i \quad = \quad & \{\mathrm{True}_j \mid l_{i,k} = P_j \text{ for some } k\} \cup \\ & \{\mathrm{False}_j \mid l_{i,k} = \neg P_j \text{ for some } k\}. \end{aligned}$$

Each s_i can be thought of as asserting "not C_i". It can now be shown that there exists a certificate to non-subsumption of Ψ_1 by Ψ_2, i.e., a member of $\mathrm{IS}(\Psi_1)$ not subsumed by Ψ_2, if and only if there exists a satisfying assignment for S. $\square$

We later define a weaker polynomial-time–computable subsumption notion for use in our learning algorithms.

Least-General Generalization. The existence of an AMA LGG is nontrivial as there are infinite chains of increasingly specific formulas that generalize given formulas: e.g., each member of the chain $P; Q$, $P; Q; P; Q$, $P; Q; P; Q; P; Q; P; Q, \ldots$ covers $P \wedge Q$ and $(P \wedge Q); Q$.

Theorem 8. *There is an LGG for any finite set Σ of AMA formulas that is subsumed by every generalization of Σ.*

Proof: Let Γ be the set $\bigcup_{\Psi' \in \Sigma} \mathrm{IS}(\Psi')$. Let Ψ be the conjunction of the finitely many MA timelines that generalize Γ while having size no larger than Γ. Each timeline in Ψ generalizes Γ and thus Σ (by Proposition 4), so Ψ must generalize Σ. Now, consider an arbitrary generalization Ψ' of Σ. Proposition 5 implies that Ψ' generalizes each member of Γ. Lemma 3 then implies that each timeline of Ψ' subsumes a timeline Φ, no longer than the size of Γ, that also subsumes the timelines of Γ. Then Φ must be a timeline of Ψ, by our choice of Ψ, so every timeline of Ψ' subsumes a timeline of Ψ. Then Ψ' subsumes Ψ, and Ψ is the desired LGG. $\square$

Strengthening "or-to-and" we can compute an AMA LGG.

Theorem 9. *For a set Σ of MA formulas, the conjunction Ψ of all MA timelines in $\mathrm{IG}(\Sigma)$ is an AMA LGG of Σ.*

Proof: That Ψ subsumes the members of Σ is straightforward. To show Ψ is "least", consider Ψ' subsuming the members of Σ. Lemma 3 implies that each timeline of Ψ' subsumes a member of $\mathrm{IG}(\Sigma)$. This implies $\Psi \leq \Psi'$. $\square$

Combining this result with Proposition 4, we get:

Theorem 10. $\mathrm{IG}(\bigcup_{\Psi\in\Sigma}\mathrm{IS}(\Psi))$ *is an AMA LGG of the set* Σ *of AMA formulas.*

Theorem 10 leads to an algorithm that is doubly exponential in the input size because both $\mathrm{IS}(\cdot)$ and $\mathrm{IG}(\cdot)$ produce exponential size increases. We believe we cannot do better:

Theorem 11. *The smallest LGG of two MA formulas can be exponentially large.*

Proof: (Sketch) Consider the formulas $\Phi_1 = s_{1,*}; s_{2,*}; \ldots ; s_{n,*}$ and $\Phi_2 = s_{*,1}; s_{*,2}; \ldots ; s_{*,n}$, where $s_{i,*} = P_{i,1} \wedge \cdots \wedge P_{i,n}$ and $s_{*,j} = P_{1,j} \wedge \cdots \wedge P_{n,j}$. For each member $\varphi = x_1; \ldots ; x_{2n-1}$ of the exponentially many members of $\mathrm{IG}(\{\Phi_1, \Phi_2\})$, define $\overline{\varphi}$ to be the timeline $P - x_2; \ldots ; P - x_{2n-2}$, where P is all the propositions. It is possible to show that the AMA LGG of Φ_1 and Φ_2, e.g., the conjunction of $\mathrm{IG}(\{\Phi_1, \Phi_2\})$, must contain a separate conjunct excluding each of the exponentially many $\overline{\varphi}$. □

Conjecture 12. *The smallest LGG of two AMA formulas can be doubly-exponentially large.*

Even when there is a small LGG, it is expensive to compute:

Theorem 13. *Determining whether a formula* Ψ *is an AMA LGG for two given AMA formulas* Ψ_1 *and* Ψ_2 *is co-NP-hard, and is in co-NEXP, in the size of all three formulas together.*

Proof: (Sketch) Hardness by reduction from AMA subsumption. Upper bound by the existence of exponentially-long certificates for "No" answers: members X of $\mathrm{IS}(\Psi)$ and Y of $\mathrm{IG}(\mathrm{IS}(\Psi_1) \cup \mathrm{IS}(\Psi_2))$ such that $X \not\leq Y$. □

Syntactic Subsumption. We now introduce a tractable generality notion, syntactic subsumption, and discuss the corresponding LGG problem. Using syntactic forms of subsumption for efficiency is familiar in ILP (Muggleton & De Raedt 1994). Unlike AMA semantic subsumption, syntactic subsumption requires checking only polynomially many MA subsumptions, each in polynomial time (via theorem 6).

Definition 3. *AMA* Ψ_1 *is syntactically subsumed by AMA* Ψ_2 *(written* $\Psi_1 \leq_{syn} \Psi_2$*) iff for each MA timeline* $\Phi_2 \in \Psi_2$*, there is an MA timeline* $\Phi_1 \in \Psi_1$ *such that* $\Phi_1 \leq \Phi_2$*.*

Proposition 14. *AMA syntactic subsumption can be decided in polynomial time.*

Syntactic subsumption trivially implies semantic subsumption—however, the converse does not hold in general. Consider the AMA formulas $(A; B) \wedge (B; A)$, and $A; B; A$ where A and B are primitive propositions. We have $(A; B) \wedge (B; A) \leq A; B; A$; however, we have neither $A; B \leq A; B; A$ nor $B; A \leq A; B; A$, so that $A; B; A$ does not syntactically subsume $(A; B) \wedge (B; A)$. Syntactic subsumption fails to recognize constraints that are only derived from the interaction of timelines within a formula.

Syntactic Least-General Generalization. The *syntactic AMA LGG* is the syntactically least-general AMA formula that syntactically subsumes the input AMA formulas[6]. Based on the hardness gap between syntactic and semantic AMA subsumption, one might conjecture that a similar gap exists between the syntactic and semantic LGG problems. Proving such a gap exists requires closing the gap between the lower and upper bounds on AMA LGG shown in Theorem 10 in favor of the upper bound, as suggested by Conjecture 12. While we cannot yet show a hardness gap between semantic and syntactic LGG, we do give a syntactic LGG algorithm that is exponentially more efficient than the best semantic LGG algorithm we have found (that of Theorem 10).

Theorem 15. *There is a syntactic LGG for any AMA formula set* Σ *that is syntactically subsumed by all syntactic generalizations of* Σ*.*

Proof: Let Ψ be the conjunction of all the MA timelines that syntactically generalize Σ, but with size no larger than Σ. Complete the proof using Ψ as in Theorem 8. □

Semantic and syntactic LGG are different, though clearly the syntactic LGG must subsume the semantic LGG. For example, $(A; B) \wedge (B; A)$, and $A; B; A$ have a semantic LGG of $A; B; A$, as discussed above; but their syntactic LGG is $(A; B; \mathbf{true}) \wedge (\mathbf{true}; B; A)$, which subsumes $A; B; A$ but is not subsumed by $A; B; A$. Even so, on MA formulas:

Proposition 16. *Any syntactic AMA LGG for an MA formula set* Σ *is also a semantic LGG for* Σ*.*

Proof: We first argue the initial claim $(\Phi \leq \Psi)$ iff $(\Phi \leq_{\mathrm{syn}} \Psi)$ for AMA Ψ and MA Φ. The reverse direction is immediate, and for the forward direction, by the definition of $\leq_{\mathrm{syn}}$, each conjunct of Ψ must subsume "some timeline" in Φ, and there is only one timeline in Φ. Now to prove the theorem, suppose a syntactic LGG Ψ of Σ is not a semantic LGG of Σ. Conjoin Ψ with any semantic LGG Ψ' of Σ—the result can be shown, using our initial claim, to be a syntactic subsumer of the members of Σ that is properly syntactically subsumed by Ψ, contradicting our assumption. □

With Theorem 11, an immediate consequence is that we cannot hope for a polynomial-time syntactic LGG algorithm.

Theorem 17. *The smallest syntactic LGG of two MA formulas can be exponentially large.*

Unlike the semantic LGG case, for the syntactic LGG we have an algorithm whose time complexity matches this lower-bound. Theorem 10, when each Ψ is MA, provides a method for computing the semantic LGG for a set of MA timelines in exponential time using IG (because $\mathrm{IS}(\Psi) = \Psi$ when Ψ is MA). Given a set of AMA formulas, the syntactic LGG algorithm uses this method to compute the polynomially-many semantic LGGs of sets of timelines, one chosen from each input formula, and conjoins all the results.

Theorem 18. *The formula* $\bigwedge_{\Phi_i\in\Psi_i}\mathrm{IG}(\{\Phi_1, \ldots, \Phi_n\})$ *is a syntactic LGG of the AMA formulas* $\Psi_1, \ldots, \Psi_n$*.*

Proof: Let Ψ be $\bigwedge_{\Phi_i\in\Psi_i}\mathrm{IG}(\{\Phi_1, \ldots, \Phi_n\})$. Each timeline Φ of Ψ must subsume each Ψ_i because Φ is an out-

[6] Again, "least" means that no formula properly syntactically subsumed by the syntactic LGG can subsume the input formulas.

	Subsumption		Semantic AMA LGG			Synt. AMA LGG		
Inputs	Sem	Syn	Low	Up	Size	Low	Up	Size
MA	P	P	P	coNP	EXP	P	coNP	EXP
AMA	coNP	P	coNP	NEXP	2-EXP?	P	coNP	EXP

Table 1: Complexity Results Summary. The LGG complexities are relative to *input plus output* size. The size column reports the largest possible output size. The "?" denotes a conjecture.

put of IG on a set containing a timeline of Ψ_i. Now consider Ψ' syntactically subsuming every Ψ_i. We show that $\Psi \leq_{\text{syn}} \Psi'$ to conclude. Each timeline Φ' in Ψ' subsumes a timeline $T_i \in \Psi_i$, for each i, by our assumption that $\Psi_i \leq_{\text{syn}} \Psi'$. But then by Lemma 3, Φ' must subsume a member of $\text{IG}(\{T_1, \ldots, T_n\})$—and that member is a timeline of Ψ—so each timeline Φ' of Ψ' subsumes a timeline of Ψ. We conclude $\Psi \leq_{\text{syn}} \Psi'$, as desired. $\square$

This theorem yields an algorithm that computes a syntactic AMA LGG in exponential time. The method does an exponential amount of work even if there is a small syntactic LGG (typically because many timelines can be pruned from the output because they subsume what remains). It is still an open question as to whether there is an output efficient algorithm for computing the syntactic AMA LGG—this problem is in coNP and we conjecture that it is coNP-complete. One route to settling this question is to determine the output complexity of semantic LGG for MA input formulas. We believe this problem to be coNP-complete, but have not proven this; if this problem is in P, there is an output-efficient method for computing syntactic AMA LGG based on Theorem 18.

Conclusion

Table 1 summarizes the upper and lower bounds we have shown. In each case, we have provided a theorem suggesting an algorithm matching the upper bound shown. The table also shows the size that the various LGG results could possibly take relative to the input size. The key results in this table are the polynomial-time MA subsumption and AMA syntactic subsumption, the coNP lower bound for AMA subsumption, the exponential size of LGGs in the worst case, and the apparently lower complexity of syntactic AMA LGG versus semantic LGG. We described how to build a learner based on these results and, in our companion work, demonstrate the utility of this learner in a substantial application.

References

Agrawal, R., and Srikant, R. 1995. Mining sequential patterns. In *Proc. 11th Int. Conf. Data Engineering*, 3–14.

Allen, J. F., and Ferguson, G. 1994. Actions and events in interval temporal logic. *Journal of Logic and Computation* 4(5).

Allen, J. F. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM* 26(11):832–843.

Bacchus, F., and Kabanza, F. 2000. Using temporal logics to express search control knowledge for planning. *Artificial Intelligence* 16:123–191.

Clarke, E. M.; Emerson, E. A.; and Sistla, A. P. 1983. Automatic verification of finite state concurrent systems using temporal logic specifications: A practical approach. In *Symposium on Principles of Programming Languages*, 117–126.

Cohen, W., and Hirsh, H. 1994. Learnability of the classic description logic: Theoretical and experimental results. In *4th International Knowledge Representation and Reasoning*, 121–133.

Cohen, W. 1994. Grammatically biased learning: Learning logic programs using an explicit antecedent description lanugage. *Artificial Intelligence* 68:303–366.

Cohen, P. 2001. Fluent learning: Elucidation the structure of episodes. In *Symposium on Intelligent Data Analysis*.

De Raedt, L., and Dehaspe, L. 1997. Clausal discovery. *Machine Learning* 26:99–146.

Dehaspe, L., and De Raedt, L. 1996. Dlab: A declarative language bias formalism. In *International Syposium on Methodologies for Intelligent Systems*, 613–622.

Fern, A.; Siskind, J. M.; and Givan, R. 2002. Learning temporal, relational, force-dynamic event definitions from video. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence*.

Hoppner, F. 2001. Discovery of temporal patterns—learning rules about the qualitative behaviour of time series. In *5th European Principles and Practice of Knowledge Discovery in Databases*.

Kam, P., and Fu, A. 2000. Discovering temporal patterns for interval-based events. In *International Conference on Data Warehousing and Knowledge discovery*.

Lavrac, N.; Dzeroski, S.; and Grobelnik, M. 1991. Learning nonrecursive definitions of relations with LINUS. In *Proceedings of the Fifth European Working Session on Learning*, 265–288.

Mannila, H.; Toivonen, H.; and Verkamo, A. I. 1995. Discovery of frequent episodes in sequences. In *International Conference on Data Mining and Knowledge Discovery*.

Mitchell, T. 1982. Generalization as search. *Artificial Intelligence* 18(2):517–542.

Muggleton, S., and De Raedt, L. 1994. Inductive logic programming: Theory and methods. *Journal of Logic Programming* 19/20:629–679.

Muggleton, S., and Feng, C. 1992. Efficient induction of logic programs. In Muggleton, S., ed., *Inductive Logic Programming*. Academic Press. 281–298.

Muggleton, S. 1995. Inverting entailment and Progol. In *Machine Intelligence*, volume 14. Oxford University Press. 133–188.

Plotkin, G. D. 1971. *Automatic Methods of Inductive Inference*. Ph.D. Dissertation, Edinburgh University.

Roth, D., and Yih, W. 2001. Relational learning via propositional algorithms: An information extraction case study. In *International Joint Conference on Artificial Intelligence*.

Siskind, J. M. 2001. Grounding the lexical semantics of verbs in visual perception using force dynamics and event logic. *Journal of Artificial Intelligence Research* 15:31–90.