Cross-Domain Transfer of Constraints for Inductive Process Modeling

Will Bridewell
Computational Learning Laboratory, CSLI,
Stanford University, Stanford, CA 94305 USA
willb@csli.stanford.edu

Ljupčo Todorovski
University of Ljubljana, Faculty of Administration
Gosarjeva 5, SI-1000 Ljubljana, Slovenia
ljupco.todorovski@fu.uni-lj.si

Abstract

In this paper, we discuss a mechanism for transfer learning in the context of inductive process modeling. We begin by describing the dual role of knowledge as a source of model components and structural constraints. Next, we review the task of inductive process modeling and emphasize the effect of domain knowledge on the learning component. We then describe the performance and learning elements of the transfer task, define the form of that resulting knowledge, and introduce an evaluation methodology for the experiments. The reported results show the effect of cross-domain transfer within the larger field of ecology. We conclude by outlining future research in this area.

Introduction

Research in transfer learning (Choi *et al.* 2007; Taylor *et al.* 2007; Thrun 1996) promises mechanisms that let induction systems improve with experience even as they operate across different domains. The wide variety of learning approaches joined with the limited research and lexical ambiguity of this area demands that authors provide a clear definition of both the base induction task and the transfer task. In our case, the learning system builds models from components that meet explicit constraints. Within this context we have successfully transferred induced constraints across similar problems where the entities within an ecosystem are biologically identical but their environments differ (Bridewell & Todorovksi 2007), and we have determined that this approach can produce scientifically plausible domain knowledge (Bridewell & Todorovski 2007). We now demonstrate that the proposed methodology for transfer learning leads to knowledge that crosses subfields of ecology (i.e., domains) where both the agents and the environments differ.

The reported approach induces first-order constraints for use by an inductive process modeler (Bridewell *et al.* 2008). As we describe in the next section, these programs construct quantitative models of dynamic systems from background knowledge and time series. The particular implementation that we employ exhaustively searches a finite space in which some solutions more accurately explain the data than others. Prior work in this area has used constraints to limit search to a reduced set of plausible model structures (Todorovski *et al.* 2005), but the knowledge in those studies came from domain experts. To capture similar knowledge via an intelligent system, we view constraint induction as a supervised learning problem. From a high level, the system marks models produced by a generator as accurate or inaccurate explanations of a particular set of observations and learns rules that discriminate between those classes.

Previous applications of this approach suggest that it can substantially reduce the learning costs for the process modeler without a corresponding reduction in model accuracy (Bridewell & Todorovksi 2007). In that study, the program modeled a two species predator–prey ecosystem, analyzed the models that it considered, and learned constraints that reduced the computational requirements of an inductive process modeler by an order of magnitude. Moreover, those constraints reflected common domain knowledge, which suggested that we could use the method to discover plausible modeling assumptions and scientific theory. To test this conjecture, we induced constraints while explaining the dynamics of the Ross Sea ecosystem, which we describe later with the experiments, and identified rules that correspond to previously known or currently debated relationships in the area of aquatic ecosystems (Bridewell & Todorovski 2007).

The next section describes inductive process modeling, and the following one details our approach to constraint induction and transference. In both cases, we emphasize the general strategies as opposed to the technical details, which are published in other articles. After this, we introduce measures for evaluating the effectiveness of transfer and describe experiments that indicate the plausibility of cross-domain transfer. To conclude, we explore the implications of our findings and the future directions that they suggest.

Inductive Process Modeling

Scientists use models to both explain and predict an observed system's behavior. In the biological sciences, those models commonly refer to processes that govern system dynamics and the entities altered by those processes. Ordinary differential equations offer one way to represent these mechanisms and to develop predictive models, but they fail to make contact with the process knowledge shared among scientists. In response, we developed a language for *quantitative process models* that ties the explanatory aspects of a model (i.e., the processes) to the mathematical formulation that enables effective simulation.

Table 1 shows a process model for a predator–prey relation between foxes and rabbits. The three processes explain the change in concentrations of the two species through time. The *rabbit_growth* process states that a fixed environmental capacity limits the reproduction of rabbits. In comparison, the *fox_death* process refers to an unlimited exponential mortality function for the fox population. Finally, the predation process, which controls the interspecies interaction, states that the prey population decreases and the predator population increases proportionally with the size of both populations. Given an initial density for each species in an ecosystem, one can transform the process model into the underlying system of two differential equations and simulate the model to produce trajectories that reflect population dynamics over time.

The processes from Table 1 instantiate more general forms, called *generic processes*, that could apply to any ecological system. These generic processes, along with generic entities, form the background knowledge for an inductive process modeler. As an example, consider

generic process holling_1:
variables $S1\{prey\}, S2\{predator\}$;
parameters $ar[0.01, 10], ef[0.001, 0.8]$;
equations
$d[S1, t, 1] = -1 * ar * S1 * S2$;
$d[S2, t, 1] = ef * ar * S1 * S2$;

which is the general form of the predation process from the example model. Notice that we replaced the parameters with continuous ranges and the entities (i.e., foxes and rabbits) with typed identifiers. We can now describe the general task of inductive process modeling as:

- *Given:* Observations for a set of continuous variables as they change over time;
- *Given:* A set of entities that the model may include;
- *Given:* Generic processes that specify causal relations among entities;
- *Given:* Constraints that determine plausible relations among processes and entities;
- *Find:* A specific *process model* that explains the observed data and predicts unseen data accurately.

A program that carries out this task requires a performance element to produce trajectories for analysis and comparison. In our research, we convert the models to systems of equations and use a numerical solver (Cohen & Hindmarsh 1996) to generate the trajectories. The learning element can take several forms, for detailed examples see Bridewell et al. (2008) and Todorovski et al. (2005), but they all search for the model whose simulated trajectories match those from the training data. In this paper, we describe a naive approach to the learning task that highlights the role that structural constraints play when constructing models.

The generate-and-test strategy is a straightforward method for implementing an inductive process modeler. Taking this perspective, the test procedure simulates the behavior of the candidate model and reports its accuracy with respect to observed trajectories. The generator binds the entities to the generic processes to create a finite set of model components. In the absence of structural constraints, this procedure enumerates all combinations of the components, presenting each subset for evaluation.[1] This scenario leads to a search space whose size grows exponentially with the number of processes. To make this strategy feasible for complex domains, one must add structural constraints, such as which entities may participate in a process and which processes are mutually exclusive.

Table 1: A process model of a predator–prey interaction between foxes and rabbits. The notation $d[X, t, 1]$ indicates the first derivative of X with respect to time t. For lack of room, we express *rabbits.population* (i.e., the *population* property of the *rabbits* entity) as *rabbits* and likewise with fox.

model Predation;
entities $rabbits\{prey\}, foxes\{predator\}$;
process rabbit_growth;
equations
$d[rabbits, t, 1] = 1.81 * rabbits * (1 - 0.0003 * rabbits)$;
process fox_death;
equations
$d[foxes, t, 1] = -1 * 1.04 * foxes$;
process predation_holling_1;
equations
$d[rabbits, t, 1] = -1 * 0.03 * rabbits * foxes$;
$d[foxes, t, 1] = 0.30 * 0.03 * rabbits * foxes$;

To illustrate the generator and the effects of constraints, consider the foxes and rabbits from the earlier example and entertain the idea that we cannot state which species is the predator. Suppose that there are 2 generic processes for growth (e.g., exponential and logistic functions) and 2 for loss, all of which could apply to both species. Also assume that there are 5 generic processes for predation. Then, ruling out cannibalism, there are 18 model components. Limiting each component to a single appearance per model leads to $2^{18} - 1 = 262,143$ structures. However, this scenario lets rabbits eat wolves, which suggests that several candidates are implausible. A constraint that rules out this odd situation eliminates all but $2^{13} - 1 = 8,191$ of the structures. By incorporating other restrictions from population dynamics (i.e., that a model contains one process for prey growth, one for predator loss, and one for predation), we reduce the space to 20 plausible candidates.[2]

Whereas the generic processes expand the scope of the search space to capture the relevant relationships among entities, the constraints restrict it to a smaller set of plausible structures. To gather the components, one can either manually search the domain literature for equations and other representations of processes or modify techniques for feature construction (Utgoff 1986) and equation discovery (Washio & Motoda 1998). In contrast, collecting constraints requires extensive collaboration with domain experts (Atanasova *et*

[1] This description leaves out the interwoven parameter estimation routine, which takes the form of a nested gradient-descent search through a continuous, multidimensional space.

[2] A plausible model contains one of two growth processes for rabbits, one of two loss processes for foxes, and one of five for predation. This gives $2 * 2 * 5 = 20$ total models.

Table 2: The domain-general relationships that can appear in a constraint. Each predicate also has an explicit negation.

includes_process(model_id, generic_process)
includes_process_type(model_id, process_type)
includes_entity_instance(model_id, entity_instance)
includes_entity(model_id, generic_entity)
includes_process_entity_instance(model_id, generic_process, entity_instance)
includes_process_entity(model_id, generic_process, generic_entity)
includes_process_type_entity_instance(model_id, process_type, entity_instance)
includes_process_type_entity(model_id, process_type, generic_entity)

al. 2006) due to their implicit nature. The next section introduces a method for automated constraint acquisition.

Learning Modeling Constraints

We treat the task of inducing modeling constraints as a supervised learning problem. The examples are descriptions of the process model structures that state which processes and entities appear and how they relate. An example's label relates to its predictive accuracy: positive examples are accurate models that closely match observed trajectories, whereas negative examples are inaccurate models. Applying a supervised learning algorithm to this data set yields rules that relate a model's structural properties to its quantitative accuracy. For instance, using this approach we can learn rules that define an accurate predation model as one with exactly three processes: one for growth, one for loss, and one for the interaction. A system can compile these rules into modeling constraints that can substantially reduce the search space explored by an inductive process modeler.

In earlier papers (Bridewell & Todorovksi 2007; Bridewell & Todorovski 2007), we introduced an approach to constraint induction that employs tools from inductive logic programming (Lavrač & Džeroski 1994). This research rests on two central ideas: (1) one can learn rules that classify structures as accurate or inaccurate for a particular problem and (2) these rules transfer across problems and possibly across domains. In this section, we describe the performance and learning elements of the transfer task with an emphasis on the generality of the approach.

To cast constraint induction into the supervised learning mold, we define a set of relational features, listed in Table 2, that may appear in the constraints and that rely on predicates which describe a quantitative process model. These features refer to the generic processes and entities that a model instantiates, the particular entities that appear, and the type of a generic process, which defines a higher-level grouping of the components. We also prepare data sets, which contain logical descriptions of structures considered during model induction. For instance, the predicate *process_instance(3, 14, exponential_growth)* declares that Process 14 in Model 3 instantiates the generic process *exponential_growth*. This will make the ground fact *includes_process(3, exponential_growth)* true.

In addition to describing the models with these features, the system assigns a target class to each model based on its quantitative fit. To this end, the program selects performance thresholds for the coefficient of determination (r^2), which falls in the [0,1] interval, and labels models that exceed the threshold as *accurate* and those below it as *inaccurate*. Since several thresholds are possible, we developed a procedure that selects an appropriate one based on the training data. The first step sorts the models by their accuracy and divides them into 10 bins. The next step identifies the point of maximal performance change between consecutive models within each bin. If multiple points are close to each other in terms of r^2, the procedure eliminates all but one. The final threshold is the one that, based on the training set, minimizes the number of accurate models (i.e., those that we want to reserve as candidates) and maximizes the recall rate of those with the best performance.

With a feature set and a threshold in hand, one can apply a supervised learning program to induce the constraints. Given the relational nature of the structural descriptors, we use Aleph, an inductive logic programming system, (Srinivasan 2000) for this step. The resulting rule set can directly constrain the candidate structures considered by an inductive process modeler. Within the context of this study, structures that the rules would classify as inaccurate are removed from consideration.

Reconsider the example of predator–prey interaction from the previous section. In this scenario, the approach for constraint induction can learn the following two rules, which describe accurate models, among others.

```
accurate model(M) :-
  includes_process_type_entity(M, loss, predator),
  includes_process_entity(M, exp_growth, prey),
  includes_process(M, hassell_varley_2).
accurate model(M) :-
  includes_process_type_entity(M, loss, predator),
  includes_process_entity(M, exp_growth, prey),
  includes_process(M, holling_type_3).
```

Both statements characterize the structure of accurate predator–prey models. In particular, they state that the model must comprise three processes: predator loss of any available kind, some form of prey growth, and one of two specific interaction processes. These rules have the appearance of constraints, and an inductive process modeler can use them to reduce the search space to those model structures that obey them.

Evaluating the Transfer of Constraints

The main hypothesis of this paper is not only that the constraints learned using the proposed approach are useful for solving future modeling problems in the same domain, but also that they will successfully transfer to other problems and domains as well. Before testing this hypothesis, we must determine how to measure the transferred knowledge's effectiveness in terms of model accuracy and search time.

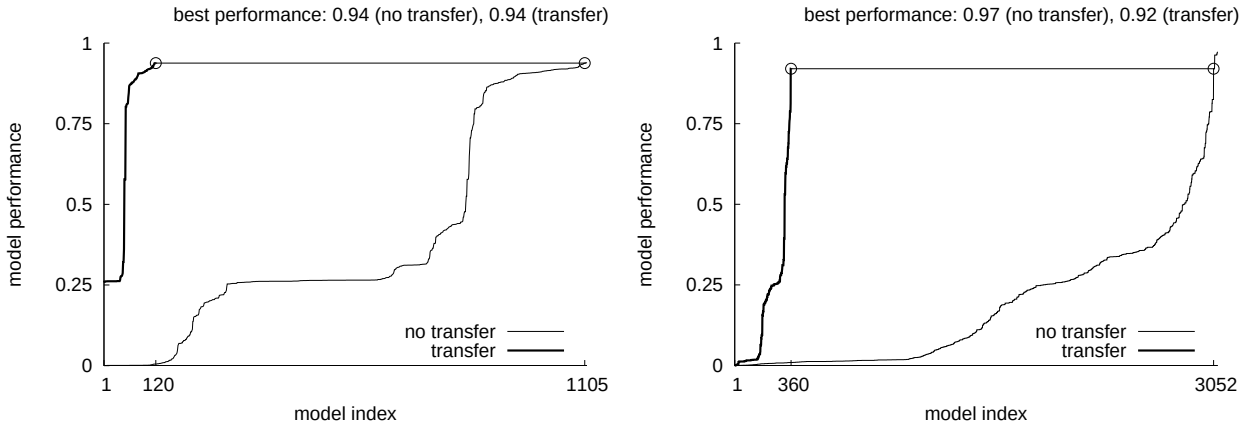

Figure 1: Comparing the learning curves of HIPM when applied to the task of modeling plankton change in the Ross Sea in 1997–1998 (left-hand side) and the Bled Lake in 2002 (right-hand side). Each graph shows the comparison of the learning curve of HIPM applied without constraints ("no transfer" curve) with the learning curve of HIPM using the constraints induced for the 1996–1997 Ross Sea data ("transfer" curve). The circles denote the best model in the constrained search space.

In previous work (Bridewell & Todorovksi 2007), we evaluated the learned constraints using measures of recall and search-space reduction. However, we found that the recall rate failed to match the aspects of the problem that we hoped to capture. Specifically, we are more concerned with reduced accuracy in the constrained problem. In our experience, the difference in r^2 scores between model i and model $(i-1)$ is often negligible, and measures of recall fail to capture this information. For example, consider a search space of 100 models. If the top 20 have equivalent accuracy within some epsilon, but the top 10 violate a set of constraints, then a measure of recall for the top 10% of models would be misleadingly low. Moreover, given the uncertainty associated with one's data, even the thirtieth best model may be satisfactory. With this in mind, we are less concerned with the number of accurate models ruled out by the constraints than with the trade off between accuracy and run time. To this end, we developed two measures: one that records the loss in accuracy between the transfer and "no-transfer" cases and a second that illustrates the reduction in search.

For both measures, we appeal to learning curves based upon a worst-case modeling scenario. That is, we assume that the model generator produces structures in order from those with the worst eventual r^2 scores to those with the best. The first measure that we report compares the accuracy of the best models in each case by assessing the percentage of the total possible performance that is lost under the transfer condition. We calculate this number as $100 * (1 - \frac{t_{best}}{nt_{best}})$, where t_{best} is the highest r^2 value in the transfer condition and nt_{best} is the highest one in the no-transfer one.

The second measure that we report compares the index of the best model from the transfer scenario with its index in the no-transfer case. The benefit from transfer is calculated according to the formula $\frac{(nt_i - t_i)}{nt_n}$, where t_i is the index of the model with highest r^2 in the transfer condition (which is equivalent to the total number of models in that case), nt_i is the index of that model in the no-transfer condition, and nt_n is the total number of models in the no-transfer case. As an example, suppose that the complete space contains 500 models and the constrained space has 100. Also, let the best model in both cases be the same. In this scenario, the transfer benefit is 0.8, which means that the best model appeared 80% faster with transferred constraints than without.

The next section uses both measures to describe initial results of our approach on a cross-domain transfer task within the general area of ecological modeling.

Experimental Results

To evaluate the utility of constraint-based transfer learning for inductive process modeling, we investigated its effects on the task of explaining ecosystem dynamics. Models of population dynamics identify the mechanisms that drive observed changes in species concentrations within an ecosystem. These mechanisms may include processes of nutrient uptake for plants, grazing for herbivores, and predation for carnivores as possible examples. For this study, we focused our attention on the general domain of aquatic ecosystems, which includes both freshwater and marine (i.e., saltwater) environments. Although these ecosystem types share more in common with each other than with terrestrial systems, they differ substantially in their species composition and in the presence and the dynamics of the relevant environmental factors. Whereas our earlier work investigated the limited, near-transfer condition where the same species were placed in different controlled environments (Bridewell & Todorovksi 2007), this research expands to cases where both the agents and the environment differ.

The two ecosystems that we selected differ not only along the freshwater–saltwater dimension but also in geographical location. The marine data are from the 1996–1997 and 1997–1998 summer seasons of the Ross Sea (Arrigo *et al.* 2003), which lies in the Southern Ocean off the coast of Antarctica. In this scenario, we had observations of five

variables, phytoplankton concentration, nitrate concentration, ice concentration, available light, and water temperature. To these we added unmeasured variables for the zooplankton and iron concentrations—two properties thought to affect the phytoplankton bloom. In comparison, the freshwater data came from the Bled Lake (Atanasova *et al.* 2006) in Slovenia during the months of 2002 where the lake was not covered in ice. Observations from this site covered seven variables: the amount of available light, the water temperature, and the concentrations of phytoplankton, zooplankton, nitrate, phosphate, and silica. Importantly, these domains differed in the particular plankton species, the set of operative nutrients, and the expressed environmental factors (e.g., there is no night during the summer in the Ross Sea).

For the experimental evaluation, we used HIPM (Todorovski *et al.* 2005) to exhaustively explore the space of models that could explain the Ross Sea 1996–1997 and Bled Lake 2002 data sets and our constraint learning approach to induce transferable knowledge. In each case we employed the same generic process library for aquatic ecosystems, but the instantiated entities differed between the Ross Sea and Bled Lake. The effect of this is apparent in the total number of candidate solutions for each domain: 1108 for the Ross Sea and 3120 for Bled Lake. After both runs of HIPM, we selected a threshold for constraint induction where the knowledge minimized the size of the search space and maximized the recall rate for the top 50 models in the training set. We applied the constraints induced at that threshold to the other domains, which were test sets for the transfer task.

We first report on the within domain transfer between the two data sets from the Ross Sea. The results, presented in Figure 1, show that the constrained space consisted of 120 candidate structures reduced from the 1108 possible, which yields a transfer benefit of 0.89. Moreover, the best model in the transfer case was identical to the one found without transfer ($r^2 = 0.94$), so there was no performance cost.

Shifting to the cross-domain tasks, our preliminary results suggest that we can expect knowledge to transfer from marine to freshwater domains and vice versa. Applying constraints learned while modeling the Ross Sea to the Bled Lake condition, shown in Figure 1, yields results quite similar to the within-domain case. The best model in the transfer condition is the 43rd in the no-transfer one, which gives a transfer benefit of 0.88, and an 8.5 fold reduction in the total size of the search space. Although the constraints filter out several of the best models, there is only a 5.2% performance loss with absolute r^2 scores dropping from 0.97 to 0.92. When we apply constraints from Bled Lake to the Ross Sea, we get a transfer benefit of 0.98 as illustrated in Figure 2. Although the "best" model is once again filtered out, the difference between it and the selected one (number 37 in the no-transfer case) is negligable. These results suggest the insensitivity of this approach to the direction of transfer.

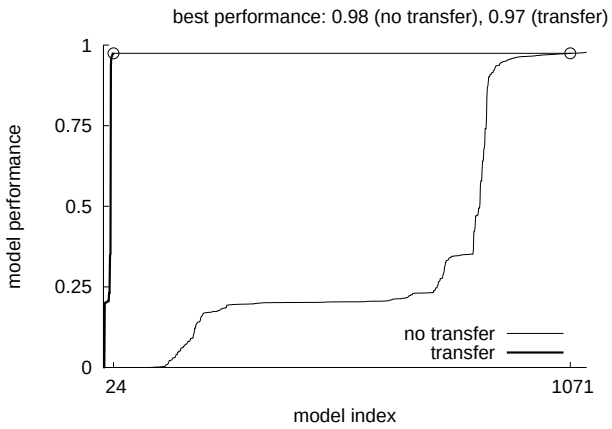

Figure 2: Transferring the constraints from the lake domain, induced on the Bled Lake data set, to the marine ecosystems domain (the task of modeling plankton change in the Ross Sea in 1996–1997).

Discussion and Closing Remarks

We emphasize again that the results reported in this paper are preliminary, but their strength bolsters our earlier findings on within-domain transfer (Bridewell & Todorovksi 2007). These combined experiments give us reason to believe that our approach is a powerful technique for learning transferable knowledge usable by an inductive process modeler. Our next immediate step involves expanding the analysis to be both broader, by introducing multiple years of Bled Lake data, and deeper, by identifying the characteristics of the constraints that transfer well and by determining the effects of learning parameters (e.g., the threshold for model classification) on the sets of constraints.

In addition to the above, we are considering other near term avenues of research. For instance, when multiple traces of a system's dynamics exist, such as the two years from the Ross Sea, we may be able to induce more general knowledge by combining these traces. One possibility involves combining the model structures considered in each case into a single set for constraint induction. Another plausible approach would have the system retain the intersection of the induced constraints from each modeling task. Other strategies are also available and experiments along these lines can ultimately inform us about how to incrementally update constraints after each problem. A longer term agenda would include problems that require transfer across more distant domains and extension of the entire approach to other areas of artificial intelligence.

Although other researchers have investigated rule learning as a means to transfer knowledge (e.g., Taylor & Stone 2007), there are few examples of constraint induction. Bessiere et al. (2006) present a system that learns constraint networks using a version space approach, but these networks are propositional and the emphasis is on automated programming as opposed to constraint transfer. Kemp (2007) uses hierarchical Bayesian models to perform constraint induction, but he focuses on psychological plausibility more-so than the development of explicit constraints that carry over from one task to the next.

In this paper, we highlighted an approach that lets an intelligent system transfer knowledge across learning tasks and improve its performance with experience. We evaluated the approach according to a new measure that indicates the ef-

fective reduction of the search space and showed how to visualize this reduction in a manner analogous to a learning curve. Experiments in applying this method in the context of inductive process modeling showed a dramatic reduction in the number of candidates considered with very little decrease in overall model fitness. Finally, we note that the form of the learned knowledge fits well with the generate-and-test strategy and as a result may generalize readily to other artificial intelligence systems.

Acknowledgments

This research was supported by Grant No. IIS-0326059 from the National Science Foundation. We thank Pat Langley, Stuart Borrett, and Tolga Könik for discussions that influenced the ideas in this paper. We also thank Kevin Arrigo and Gert van Dijken for the use of their data from the Ross Sea and their expertise in this ecosystem.

References

Arrigo, K.; Worthen, D.; and Robinson, D. H. 2003. A coupled ocean-ecosystem model of the Ross Sea. Part 2: Iron regulation of phytoplankton taxonomic variability and primary production. *Journal of Geophysical Research* 108(C7):3231.

Atanasova, N.; Todorovski, L.; Džeroski, S.; and Kompare, B. 2006. Constructing a library of domain knowledge for automated modelling of aquatic ecosystems. *Ecological Modelling* 194:14–36.

Bessiere, C.; Coletta, R.; Koriche, F.; and O'Sullivan, B. 2006. Acquiring constraint networks using a SAT-based version space algorithm. In *Proceedings of the Twenty-First National Conference on Artificial Intelligence*, 23–24. Boston, MA: AAAI Press.

Bridewell, W., and Todorovksi, L. 2007. Learning declarative bias. In *Proceedings of the Seventeenth Annual International Conference on Inductive Logic Programming*, 63–77. Corvallis, OR: Springer.

Bridewell, W., and Todorovski, L. 2007. Extracting constraints for process modeling. In *Proceedings of the Fourth International Conference on Knowledge Capture*, 87–94. Whistler, BC: ACM Press.

Bridewell, W.; Langley, P.; Todorovski, L.; and Džeroski, S. 2008. Inductive process modeling. *Machine Learning* 71:1–32.

Choi, D.; Könik, T.; Nejati, N.; Park, C.; and Langley, P. 2007. Structural transfer of cognitive skills. In *Proceedings of the Eighth International Conference on Cognitive Modeling*, 115–120. Ann Arbor, MI: Taylor & Francis Group.

Cohen, S., and Hindmarsh, A. 1996. CVODE, a stiff/nonstiff ODE solver in C. *Computers in Physics*, 10:138–143.

Jost, C., and Ellner, S. P. 2000. Testing for predator dependence in predator–prey dynamics: A non-parametric approach. *Proceedings of the Royal Society of London Series B-Biological Sciences* 267:1611–1620.

Kemp, C. 2007. *The acquisition of inductive constraints.* Ph.D. thesis, Massachusetts Institute of Technology, Cambridge, MA.

Lavrač, N., and Džeroski, S. 1994. *Inductive Logic Programming: Techniques and Applications*. New York, NY: Ellis Horwood.

Srinivasan, A. 2000. *The Aleph Manual*. Computing Laboratory, Oxford University.

Taylor, M. E.; Stone, P.; and Liu, Y. 2007. Transfer learning via inter-task mappings for temporal difference learning. *Journal of Machine Learning Research*, 8:2125–2167.

Taylor, M. E., and Stone, P. 2007. Cross-domain transfer for reinforcement learning. In *Proceedings of the Twenty-Fourth International Conference on Machine Learning*, 879–886. Corvallis, OR: Omnipress.

Thrun, S. 1996. *Explanation-based neural network learning: A lifelong learning approach*. Boston, MA: Kluwer Academic Publishers.

Todorovski, L.; Bridewell, W.; Shiran, O.; and Langley, P. 2005. Inducing hierarchical process models in dynamic domains. In *Proceedings of the Twentieth National Conference on Artificial Intelligence*, 892–897. Pittsburgh, PA: AAAI Press.

Utgoff, P. E. 1986. *Machine Learning of Inductive Bias*. Boston, MA: Kluwer Academic Publishers.

Washio, T., and Motoda, H. 1998. Discovering admissible simultaneous equations of large scale systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, 189–196. Madison, WI: AAAI Press.