A Learning Classifier System Approach to Relational Reinforcement Learning

Drew Mellor

B. Comp. Sci. (Hons)

Submitted in partial fulfilment of the requirements for the degree of

Doctor of Philosophy (Computer Science)

School of Electrical Engineering and Computer Science

The University of Newcastle

Callaghan, 2308

Australia

February 2008

I hereby certify that the work embodied in this thesis is the result of original research and has not been submitted for a higher degree to any other University or Institution.

(Signed): .....

Acknowledgements

A PhD thesis is a sizeable undertaking and invariably depends on contributions from many people besides the author. I would like to extend a warm thank-you to the people in the following list, all of whom, although some of them may not realise it, contributed to this research. Mirka Miller encouraged me to do a PhD in the first place and was instrumental during the application process. Sašo Džeroski gave a keynote talk at ICML 2002 which inspired the thesis topic. My supervisors Stephan Chalup and Huilin Ye gave me their trust and a free hand to develop the topic. Frans Henskens went beyond his professional duty to help secure much needed travel funding for the presentation of my work. Stewart Wilson, Tim Kovacs and Martin Butz provided encouragement and validation at a crucial time. Sašo Džeroski, Kurt Driessens, Martijn van Otterlo and Federico Divina helpfully answered my queries about relational reinforcement learning and other topics. Robert King advised on appropriate statistical tests, while Aaron Scott, David Montgomery and Geoff Martin tirelessly provided valuable technical support without which this thesis would have come to nothing. During the writing phase many people proof read chapters: Alyssa Brugman, Dirk Brugman, Elena Prieto, Erol Engin, Linda Seymour, Michael Quinlan and of course my supervisors, Stephan and Huilin. Throughout the course of the candidature my parents Rob and Cherie Mellor always gave me their support, while Helen Giggins and the legged Robocup team, Michael, Craig, Naomi, Kenny, Steve and the others were at hand for much needed regular diversions. I would also like to thank anyone else that belongs on this list but whom I might have forgotten to add, for which I sincerely apologise. Last but not least, I would like to mention my long time pet cat and companion, "Crackles", who passed away during the candidature and to whom this thesis is dedicated.

Contents

List of Symbols

The lists below contain symbols and acronyms used commonly throughout the thesis. Where relevant, page numbers are given to where the symbol or abbreviation is introduced.

Logic

$\wedge$	conjunction
$\vee$	disjunction
$\neg$	negation
$\leftarrow$	implication
$\exists$	existential quantifier
$\forall$	universal quantifier
$\models$	entailment, 215
θ	a substitution
θ^{-1}	an inverse substitution
$const(\Phi)$	the set of constant symbols occurring in the logical sentence Φ
$vars(\Phi)$	the set of variables occurring in the logical sentence Φ

Reinforcement Learning

t	discrete time step, 18
s	a state, 18
a	an action, 18
r	a reward, 18
$\mathcal{S}$	state space, 18
$\mathcal{A}$	action space, 18
$T(s, a, s')$	the probability of transition from s to s' under a, 19
$R(s, a)$	expected reward from taking a in s, 19
$\mathcal{A}(s)$	set of actions possible in state s, 19
π	policy, 21
π^*	optimal policy, 22
$\pi(s)$	the action to take in state s under policy π, 21
$V^\pi(s)$	the value of state s under policy π, 21
$V^*(s)$	the value of state s under an optimal policy, 22
$Q^*(s, a)$	the value of state-action pair (s, a) under an optimal policy, 28
γ	discount factor, 21
α	learning rate, 27
ϵ	probability of selecting a random action under an ϵ-greedy policy, 29

Relational Reinforcement Learning

$\mathcal{L}$	an alphabet over first-order logic
$\mathcal{C}$	set of constant symbols
$\mathcal{F}$	set of function symbols
$\mathcal{P}$	set of predicate symbols
$\mathcal{P}_{\mathcal{A}}$	set of predicate symbols for representing actions
$\mathcal{P}_{\mathcal{S}}$	set of predicate symbols for representing states
$\mathcal{P}_{\mathcal{B}}$	set of predicate symbols for representing background knowledge

Learning Classifier Systems

The following list gives the parameters associated with an individual rule j in XCS (the parameter's name is given in parentheses):

A_j	action advocated by j (action), 74
C_j	the set of states to which j applies (condition), 74
p_j	an estimate of the mean expected payoff for j (prediction), 75
ε_j	an estimate of the mean expected absolute difference between p_j and the actual payoff (error), 75
F_j	determines j's probability of selection for reproduction (fitness), 75
exp_j	the number of times j has been a member of an action set (experience), 75
ns_j	an estimate of the mean size of the action sets that j has been a member of (niche size), 76

ts_j the time step when the GA was most recently invoked on an action set that j belonged to (time step), 76

n_j the number of micro-rules represented by j (numerosity), 76

dv_j probability that j is deleted (deletion vote), 90

Other important symbols and parameters for XCS:

[P] the current set of rules contained within the system (population), 74

[M] the set of rules matching the current state (match set), 77

[A] the subset of [M] advocating the selected action (action set), 80

$[A]_{-1}$ [A] from the previous time step (action set), 80

N the maximum number of rules in terms of micro-rules that may exist in $[P]$ at any one time, 77

ρ the target value when updating p, 81

κ an inverse measure of ε used for calculating fitness (accuracy), 82

a, b, ϵ_0 parameters for calculating κ, 82

α learning rate for p, ε, F, and ns updates, 81

γ discount factor, 81

θ_{GA} a threshold used by the triggering mechanism for the GA, 89

θ_{del} a threshold used by the rule deletion mechanism, 90

δ the fraction of the mean fitness below which a rule's probability of deletion is increased, 90

θ_{sub} a threshold used by subsumption deletion, 91

The following symbols are specific to FOXCS:

Φ_j the logical part of j, replacing A_j and C_j, consisting of a definite clause over first-order logic, 115

μ_i determines the probability of selecting an evolutionary operation, $i \in \{\text{DEL,C2V,V2A,ADD,V2C,A2V,REP}\}$, 134

Acronyms

The following acronyms are used throughout this thesis:

GA Genetic algorithm, 104

ILP Inductive logic programming, 10, 222

LCS Learning classifier system, 10, 71

MDP Markov decision process, 18

RRL Relational reinforcement learning, 36, 47

Abstract

Machine learning methods usually represent knowledge and hypotheses using attribute-value languages, principally because of their simplicity and demonstrated utility over a broad variety of problems. However, attribute-value languages have limited expressive power and for some problems the target function can only be expressed as an exhaustive conjunction of specific cases. Such problems are handled better with inductive logic programming (ILP) or relational reinforcement learning (RRL), which employ more expressive languages, typically languages over first-order logic. Methods developed within these fields generally extend upon attribute-value algorithms; however, many attribute-value algorithms that are potentially viable for RRL, the younger of the two fields, remain to be extended.

This thesis investigates an approach to RRL derived from the learning classifier system XCS. In brief, the new system, FOXCS, generates, evaluates, and evolves a population of "condition-action" rules that are definite clauses over first-order logic. The rules are typically comprehensible enough to be understood by humans and can be inspected to determine the acquired principles. Key properties of FOXCS, which are inherited from XCS, are that it is general (applies to arbitrary Markov decision processes), model-free (rewards and state transitions are "black box" functions), and "tabula rasa" (the initial policy can be unspecified). Furthermore, in contrast to decision tree learning, its rule-based approach is ideal for *incrementally* learning expressions over first-order logic, a valuable characteristic for an RRL system.

Perhaps the most novel aspect of FOXCS is its inductive component, which synthesizes evolutionary computation and first-order logic refinement for incremental learning. New evolutionary operators were developed because previous combinations of evolutionary computation and first-order logic were non-incremental. The effectiveness of the inductive component was empirically demonstrated by benchmarking on ILP tasks, which found that FOXCS produced hypotheses of comparable accuracy to several well-known ILP algorithms. Further benchmarking on RRL tasks found that the optimality of the policies learnt were at least comparable to those of existing RRL systems. Finally, a significant advantage of its use of variables in rules was demonstrated: unlike RRL systems that did not use variables, FOXCS, with appropriate extensions, learnt scalable policies that were genuinely independent of the dimensionality of the task environment.