Learning Decision Lists by Prepending Inferred Rules.

Geoffrey I Webb
School of Computing and Mathematics
Deakin University
Geelong 3217 Australia.
webb@deakin.edu.au
Tel: +61-52-272606
Fax: +61-52-272028

Abstract

This paper describes a new algorithm for learning decision lists that operates by prepending successive rules to front of the list under construction. This contrasts with the original decision list induction algorithm which operates by appending successive rules to end of the list under construction.. The new algorithm is demonstrated in the majority of cases to produce smaller classifiers that provide improved predictive accuracy than those produced by the original decision list induction algorithm.

Area: machine learning

Subarea: learning decision lists

Introduction

A decision list (Rivest, 1987) is an ordered list of classification rules. Each rule consists of a condition and a conclusion in the form of a classification statement. To apply a decision list to the classification of a case, the rules are examined in order. The conclusion of the first rule for which the condition is satisfied by the case is used to assign a class to that case. Decision lists can be thought of as a sequence of nested IF-THEN-ELSE statements, in that subsequent rules are only considered if previous rules do not succeed.

CN2 (Clark & Niblett, 1989) extends Rivest's (1987) original decision list induction algorithm by:

- seeking to develop rules in order from those with high evidence to those with low evidence, rather than in any order;
- allowing multiple disjoint classes, rather than the binary classification employed by Rivest; and
- enabling the induction of rules that are inconsistent with the training set, thereby allowing for the possibility of noise.

The top level of CN2 can be expressed as follows:

> let *rule_list* be the empty list.
> let *M* be the most common class in *E*.
> repeat until *best_rule* is nil or *E* is empty
>   let *best_rule* be the best rule as evaluated by the preference function with respect to *E*.
>   if *best_rule* is not nil
>   let *E'* be the examples covered by *best_rule*.
>   remove from *E* the examples in *E'*.
>   add *best_rule* to the end of *rule_list*.
>   end if
> end repeat
> add the default rule 'IF *TRUE* THEN the class is *M* ' to the end of *rule_list*.
> return *rule_list*.

Note that CN2 employs a specific heuristic search technique to approximate optimization of the preference function when selecting then 'best rule'. In recognition of this distinction, the above algorithm will be referred to as *append*.

Note also that the addition of the default rule to the end of the decision list is implicit in Clark & Niblett's (1989) description of CN2, being specified in the interpretation procedure, rather than as part of the induction algorithm. However, the default rule must be derived by the induction system as the rule interpreter has no access to the training set and thus cannot determine which class is most highly represented therein.

With typical preference functions, the consequent of the first rule developed by *append* is usually the most common class from the training set, as it is usually possible to obtain the most positive cases in support of such a rule, giving it the highest preference value. Similarly, because it is the most infrequent class that has the least positive cases available to support a rule, the last rule created within the repeat loop often takes the form, IF *TRUE* THEN conclude that the case belongs to the most infrequent class. This rule will precede the default rule which is always added to the end of the decision list and, as it will fire for all cases that are not covered by rules earlier in the list, the default rule will not apply. The default rule is not redundant in all cases, however, as the latter form of rule will not be developed for some data sets.

It is apparent that there is considerable inefficiency in the decision procedures created by this algorithm.

1. All cases belonging to the most common class could be handled by the default rule. Instead, most of these cases are typically handled by the first rule in the list.
2. It will often be the case that the default rule can never be reached due to prior rules necessarily firing first.

To summarize, *append* fails to take maximal advantage of the default rule.

In view of these deficiencies, an alternative induction algorithm is proposed that starts with the default rule, and then prepends to the list successive rules, each time prepending the rule that best improves the total performance of the decision list developed to date. Such an algorithm can be viewed as a successive refinement algorithm, at each step refining the list by prepending the rule that best improves global performance. By contrast, the *append* algorithm does not consider when developing rules the effect of the default rule that will be added to the end of the decision list.

The new algorithm, *prepend*, can be described as follows.

let *M* be the most common class in *E*.
let *rule_list* be a list containing the rule 'IF *TRUE* THEN the class is *M* '.
repeat until *best_rule* is nil or *best_rule* covers less positive than negative cases
  let *best_rule* be the best rule as evaluated by the preference function with respect to *E*.
  if *best_rule* is not nil and *best_rule* covers more positive than negative cases
    add *best_rule* to the start of *rule_list*.
  end if
end repeat
return *rule_list*.

A positive case with respect to a rule is a case belonging to the class identified by the conclusion of the rule. A negative case with respect to a rule is any case that does not belong to the class identified by the conclusion of the rule. It is necessary to terminate the repeat loop when the best rule covers less positive than negative cases in order to guarantee termination of the algorithm. Otherwise the addition of a rule can increase the number of misclassified cases, which will lead to the need to develop further rules to allow for those cases. Such a process can continue in an infinite loop.

The same preference functions may be applied to selecting successive rules as for the *append* algorithm, with the difference that the positive cover is calculated only from cases not correctly covered by the existing decision list and the negative cover is calculated only from cases correctly covered by the existing decision list.

When investigating the value of a rule, with respect to negative cover, one should not consider cases that are already misclassified. Further misclassification of cases that are already misclassified will not affect the performance of the decision list. Similarly, one should not consider the correct classification of cases that are already correctly classified as this also will not affect the performance of the system.

It was hypothesized that *prepend* would develop smaller decision lists than *append* due to the manner in which it maximizes the effect obtained by the default rule. It was further hypothesized that this would result in a minimization of predictive classification error. This prediction was based on the assumption that each rule will on average introduce a small level of error and an increase in the number of rules will consequently result in an increase in the total level of error.

However, initial experimentation (Webb & Brkic, 1993) failed to support the former hypothesis. This paper presents further theoretical development of the initial hypotheses and resulting refinements of the *prepend* algorithm that do satisfy the initial expectations of the *prepend* algorithm.

Refined hypotheses

The failure of *prepend* to outperform the standard decision list algorithm, *append*, led to the development of a hypothesis that the predictive accuracy of *prepend* is, in general, decreased by the manner in which it elevates small disjuncts (rules covering few positive cases (Holte, Acker, & Porter, 1989)) to the front of the decision list. As small disjuncts are based on few positive cases, they are inferred last and hence placed by *prepend* at the front of the decision list but by *append* at the end of the list. However, due to the low level of evidence supporting small disjuncts, their expected error rate should be higher than that of large disjuncts. As *prepend* elevates them to the front of the decision list, their likelihood of firing is maximized, hence enhancing their likely contribution to the global error rate.

It was further hypothesized that this deficiency could be remedied by altering *prepend* to reduce the impact of small disjuncts. A number of techniques for achieving this end were investigated.

The first technique prevented the addition of small disjuncts to the decision list by halting induction when the preference value of the rules inferred fell to or below a pre-specified cut-off. Five cut-off values were investigated, each of the values of a rule covering one, two, three, five and ten positive along with no negative cases. Variations of this technique that pruned rules below pre-specified Laplace values, such as, 0.5 were also examined. This latter technique increases pruning as the number of classes increases, a strategy that proved counter-productive.

Another technique, *spec*, sought to minimize the impact of small disjuncts by specializing all rules added to the decision list. Every rule added to the decision list was replaced by the most specialized rule that covered all positive cases covered by the initial rule. This new rule was created by forming the least generalization (Plotkin, 1970) of the positive cases covered by the rule.

The third technique, *intern*, sought to minimize the impact of small disjuncts by placing each rule developed as deep within the decision list as possible without decreasing the overall classification accuracy of the list. As small disjuncts are developed last they tend to be placed toward the end of the list under this strategy. This strategy was combined with each of the preceding strategies when applied to *prepend*. This strategy is not applicable to *append* as *append* always places the small disjuncts at the end of a decision list.

Preliminary investigation (not reported here due to space constraints) indicated that pruning small disjuncts and the *intern* technique were both effective at improving the predictive accuracy of the *prepend* algorithm. The optimum definition of a small disjunct for pruning purposes varied substantially from data set to data set. There was some evidence that there was a relationship between the accuracy of the default rule and the optimum definition of small disjunct, such that rule with higher cover should be pruned when the accuracy of the default rule was higher. However, when restricted to the selection of a single definition of small disjunct, the best overall performance appeared to be achieved by pruning rules with a value less than or equal to that of a rule covering two positive and no negative cases (0.75).

The *spec* technique did not demonstrate consistent significant advantages over the plain *prepend* algorithm.

The overall best results for *prepend* were obtained by *prepend–ip2* (*prepend* with the intern technique coupled with pruning of rules with a Laplace value below that obtained with a positive cover of two and no negative cover).

These preliminary experiments suggested that *prepend–ip2* is a significant improvement over the *append* algorithm for developing decision lists. However, considerable caution must be employed in reaching such conclusions from post hoc analysis of multiple techniques. Greater confidence may be obtained if substantive predictions are made in advance of an experiment and those predictions are confirmed, rather than if multiple techniques are experimentally compared and that which happens to perform best is accepted as in general superior. To this end, it was predicted that *prepend–ip2* would significantly outperform *append*, with respect both to predictive accuracy and decision list size, on a wide variety of data sets.

Experimental evaluation

The following algorithms were evaluated:

- *Prepend* with *intern* and *prune 2*.
- *Append* with *prune 2*.
- *Append* .
- *C4.5rules* (Quinlan, 1993).
- *C4.5* (Quinlan, 1993).

Both *append* and *append–p2* were included in the study to control for the effect of the *prune2* technique. *C4.5rules* was included in the study to provide a comparison with a widely used machine learning system that learns decision lists by alternative means. (While the rules developed by *C4.5rules* are not explicitly ordered, the rule interpreter with which they are employed imposes an implicit order upon them, as a result of which, they are equivalent to decision lists.) *C4.5* was included in the study to provide a comparison with a widely used machine learning system that learns an alternative form of classifier (decision trees).

The *append*, *append-p2* and *prepend-i2p* algorithms were implemented in 'C' on a Solbourne 5/602 computer. The OPUS (Webb, 1993b) systematic search algorithm was used to find rules that maximized the preference function in place of the heuristic search employed within the *append* algorithm by CN2. The heuristic search employed within CN2 was not employed solely because code to implement it was not available. Identical search algorithms were employed within both *prepend*□and *append*. There is no reason to believe that the use of heuristic search would have significantly altered the relative performance of the two systems.

The antecedent of each rule took the form of a conjunction of attribute-value inequality tests, for example, *gender≠male* & *status≠married*. Such expressions have equivalent expressive power to internal disjunctive expressions (Webb, 1993a). The consequent of each rule was a simple classification statement. The current implementations are limited to nominal attributes.

A version of the Laplace preference function (Clark & Boswell, 1991) was used to select the 'best rule' at each iteration through the induction process. This preference function is defined as

$$value=\frac{pos\_cover+1}{pos\_cover+neg\_cover+2}.$$

The best rule was defined as the rule that maximized this preference function.

To evaluate the experimental hypothesis, in order to minimize the introduction of bias through the selection of data sets, all applicable experimental data sets were sought from the UCI machine learning repository (Murphy & Aha, 1993). The UCI repository contains twelve data sets that consist entirely of categorical attributes. The systematic search algorithm employed within *prepend* and *append* is currently limited to categorical data, and thus evaluation was restricted to these data sets. These data sets are described in Table 1. This table includes, in order by column: a brief description; the number of attributes describing each case; the number of attribute values, treating missing values as distinct values; the number of cases in the data set; the proportion of these cases for which there is a case with an identical description that belongs to another class; the percentage of cases belonging to the most common class (the class most highly represented in the data); and the number of classes.

Table 1: Summary of experimental data sets.

Domain	Description	Attribs	Values	Cases	Ident%	MCC%	Classes
Audiology	Medical diagnosis.	59	162	226	0	25.0	24
Breast Cancer	Medical prognosis.	9	57	286	5	70.3	2
House Votes 84	Predict political affiliations from US Senate voting record.	16	48	435	0	61.4	2
Lymphography	Medical diagnosis.	18	60	148	0	54.7	4
Monk 1	Artificial data.	6	17	556	0	50.0	2
Monk 2	Artificial data.	6	17	601	0	65.7	2
Monk 3	Artificial data.	6	17	554	0	52.0	2
Multiplexer	Artificial data.	11	22	500	0	50.8	2
Mushroom	Identify poisonous mushrooms.	22	126	8124	0	51.8	2
Primary Tumor	Medical diagnosis.	17	42	339	18	24.8	22
Soybean Large	Botanical diagnosis.	35	135	307	0	13.0	19
Tic Tac Toe	Identify won or lost positions.	9	27	958	0	65.3	2

Each data set was randomly divided into training (80% of data) and evaluation (remaining 20% of data) sets. Each algorithm was applied to each training set. Every classifier so developed was evaluated for predictive classification accuracy against the corresponding evaluation set. This process was repeated 100 times for each data set.

Table 2 presents the mean and standard deviations of the predictive accuracy obtained along with the result of a two-tailed matched pairs t-test evaluating the difference in performance between *prepend–ip2* and each of the other algorithms. The *p* values for those differences that are significant at the 0.05 level are highlighted. Where *prepend-ip2* has higher predictive accuracy the *p* value is highlighted in bold type. Where the alternative algorithm has higher predictive accuracy the *p* value is italicized. Of the mean accuracies that are significantly different at the 0.05 level to that of *prepend–ip2*, *append* and *append–p2* have higher accuracy thrice and lower accuracy seven times; and *C4.5* and *C4.5rules* have higher accuracy four times and lower accuracy five times. This supports the hypothesis that *prepend–ip2* will in general produce decision lists with higher predictive accuracy than *append*. While *prepend–ip2* slightly outperforms *C4.5* and *C4.5rules*, this advantage is not sufficient to conclude a general advantage therefrom.

Table 2: Mean predictive accuracy

	prepend–ip2		*append-p2*			*append*			*C4.5rules*			*C4.5*		
	$\bar{x}$	s	$\bar{x}$	s	*p*	$\bar{x}$	s	*p*	$\bar{x}$	s	*p*	$\bar{x}$	s	*p*
Audiology	77.2	5.4	68.9	5.4	**0.00**	71.6	5.7	**0.00**	75.0	5.9	**0.00**	75.5	5.9	**0.01**
Breast Cancer	61.2	5.6	64.8	5.9	*0.00*	64.5	6.2	*0.00*	68.6	5.7	*0.00*	71.7	5.3	*0.00*
House Votes 84	94.9	2.5	94.6	2.6	**0.08**	94.5	2.6	**0.03**	95.5	1.8	*0.01*	95.4	1.9	*0.03*
Lymphography	84.3	7.4	81.4	7.5	**0.00**	81.2	7.7	**0.00**	79.2	6.9	**0.00**	78.3	6.9	**0.00**
Monk 1	100.0	0.0	100.0	0.0	1.00	100.0	0.0	1.00	100.0	0.0	1.00	97.4	3.6	**0.00**
Monk 2	99.1	1.5	98.1	2.0	**0.00**	98.2	1.9	**0.00**	72.4	4.4	**0.00**	62.7	4.8	**0.00**
Monk 3	97.9	1.2	97.5	1.4	**0.00**	97.3	1.3	**0.00**	98.7	0.9	*0.00*	98.8	0.9	*0.00*
Multiplexer	98.3	3.0	99.1	1.5	*0.01*	99.2	1.4	*0.00*	96.8	3.9	**0.00**	86.1	6.7	**0.00**
Mushroom	100.0	0.0	100.0	0.0	0.27	100.0	0.0	0.27	99.9	0.1	**0.00**	100.0	0.0	0.09
Primary Tumor	39.1	5.3	35.6	4.7	**0.00**	37.3	4.7	**0.00**	39.8	5.6	0.20	40.5	5.1	*0.01*
Soybean Large	83.6	5.3	78.9	6.0	**0.00**	79.8	5.9	**0.00**	84.4	4.6	0.11	83.3	4.9	0.49
Tic Tac Toe	96.9	1.0	98.7	2.0	*0.00*	98.7	1.9	*0.00*	97.7	2.1	*0.00*	84.4	3.3	**0.00**

It is interesting to note that two of the three data sets for which *append* enjoys a significant advantage over *prepend* are two of the data sets with the highest proportion of cases belonging to the most common class. For these data sets, both *append* and *prepend* will tend to first form a highly general rule for the most common class. This rule will be placed at the head of the decision list for *append* but toward the end of the decision list for *prepend*. Where this rule is overly general (for the available evidence), *prepend* will enjoy an advantage if the proportion of cases belonging to the most common class is low whereas *append* will enjoy an advantage if it is high. This provides a possible explanation for *append*'s advantage with respect to these data sets.

The second hypothesis that this study seeks to evaluate is that *prepend* will produce shorter decision lists than *append*. Table 3 presents the mean and standard deviations of the number of rules developed along with the result of a two-tailed matched pairs t-test evaluating the difference in performance between *prepend–ip2* and each of the other decision list induction algorithms. The default rule is included in the tally of rules for *prepend-ip2*, but is not included in this tally for the other algorithms due it often being unreachable.

Of the differences in performance that are significant at the 0.05 level, *append* and *append–p2* develop more rules than *prepend–ip2* for eleven data sets and in no case develop less rules and *C4.5rules* develops more rules in ten cases and less in two. These results provide strong support for the hypothesis that *prepend–ip2* in general develops shorter decision lists than the *append* approach. *Prepend–ip2* is also demonstrated to produce in general shorter decision lists than *C4.5rules*.

Table 3: Number of rules

	prepend–ip2		*append-p2*			*append*			*C4.5rules*		
	$\bar{x}$	s	$\bar{x}$	s	*p*	$\bar{x}$	s	*p*	$\bar{x}$	s	*p*
Audiology	10.7	0.9	12.7	0.9	**0.00**	25.6	1.4	**0.00**	17.0	2.1	**0.00**
Breast Cancer	9.6	1.1	16.3	1.0	**0.00**	19.9	1.6	**0.00**	7.7	3.3	*0.00*
House Votes 84	5.0	0.5	8.3	1.1	**0.00**	10.7	1.3	**0.00**	6.8	1.2	**0.00**
Lymphography	5.1	0.6	7.3	0.8	**0.00**	9.1	0.7	**0.00**	9.7	1.5	**0.00**
Monk 1	4.7	0.5	6.0	0.4	**0.00**	6.0	0.4	**0.00**	21.7	1.2	**0.00**
Monk 2	15.8	0.4	25.1	1.5	**0.00**	25.6	1.7	**0.00**	26.5	5.4	**0.00**
Monk 3	4.9	0.7	10.3	1.2	**0.00**	12.8	1.6	**0.00**	12.0	0.1	**0.00**
Multiplexer	11.9	1.3	15.6	1.6	**0.00**	15.8	1.8	**0.00**	19.7	2.6	**0.00**
Mushroom	4.0	0.0	4.0	0.1	0.32	4.0	0.1	0.32	12.0	3.0	**0.00**
Primary Tumor	16.8	1.6	25.8	2.8	**0.00**	81.3	3.8	**0.00**	14.7	2.5	*0.00*
Soybean Large	20.7	0.8	24.0	1.1	**0.00**	27.3	1.3	**0.00**	26.5	2.0	**0.00**
Tic Tac Toe	12.0	0.8	16.3	2.0	**0.00**	16.6	2.4	**0.00**	21.2	4.3	**0.00**

Related research

A related decision list induction algorithm, *BBG* (Van Horn & Martinez, 1993) has been developed independently of the research reported herein. Like *prepend*, *BBG* infers a decision list by first inserting a default rule and then by inserting successive rules into positions prior to the default rule. Unlike *prepend* which only considers the effect of inserting a rule at the head of the decision list, *BBG* considers every insertion point before each insertion. Unlike the current research which employs systematic search, *BBG* employs heuristic search to select a rule for insertion. *BBG* is further distinguished from *prepend* by its use of a gain-cost ratio which trades the error rate of the decision list against the number of references to literals within the decision list. This contrasts with *prepend*'s use of a preference function that examines only the positive and negative cover of the rule to be added. The relative merits of each of these features is a subject for future research.

Prepend should also be distinguished from the incremental decision list induction algorithm *CDL* (Shen, 1992) which, while it may modify rules that appear in any position within a decision list, always appends new rules to the end of the list.

Summary and further research

This research has investigated an alternative approach to the induction of decision lists to that proposed by Rivest (1987) and refined by Clark & Niblett (1989). This new approach starts with a default rule and adds successive rules to the front of the list. It was hypothesized that this approach would produce shorter decision lists with greater predictive accuracy than the previous approach. However, initial research did not support the first of these hypotheses. It was hypothesized that this failure resulted from the manner in which the new algorithm promoted small disjuncts to the head of the decision list. This hypothesis was supported by experimental confirmation (not presented herein due to space constraints) that inserting small disjuncts as deeply as possible within the decision list decreased the error rate. Further decreases in the error rate occurred for some data sets when small disjuncts were not included in the decision lists. However, the optimal definition of a small disjunct varied substantially from data set to data set.

For experimental evaluation, a compromise definition of small disjunct was adopted, under which all rules with a preference value less than or equal to the value of a rule covering two positive and no negative cases were pruned. *Prepend* with the pruning of small disjuncts under this definition and the insertion of rules as deeply as possible within the list was demonstrated to outperform the *append* induction algorithm in terms of both predictive accuracy and decision list size. While it achieved comparable (or possible slightly better) predictive accuracy than *C4.5* and *C4.5rules*, it developed shorter decision lists than the latter (the former does not develop decision lists).

While the induction of decision lists starting with a default rule and adding rules to prior positions has been investigated independently elsewhere (Van Horn & Martinez, 1993), this paper contributes experimental comparison with the *append* approach, identification of the problem that small disjuncts pose for the *prepend* approach and evaluation of solutions to that problem.

A number of outstanding issues are worthy of further research.

Is it an advantage to consider all insertion points, as per *BBG*, rather than considering only insertion of rules at the head of the decision list, as per *prepend*?

Is it possible to select an optimum definition of small disjunct for each data set? The relationship between the accuracy of the default rule and the optimal definition of small disjunct appears worthy of further investigation.

Does the advantage for *prepend* over *append* extend to domains that contain ordinal and continuous attributes? There is no reason to suppose that they should not. The current research was restricted to nominal attributes only by the limitations on the systematic search algorithm currently employed to identify the best rule for insertion.

Notwithstanding the work remaining to be done, in the majority of cases, the induction of decision lists by prepending rules and excluding small disjuncts appears to significantly outperform the induction of decision lists by appending rules in terms of both predictive accuracy and the complexity of the resulting descriptions.

Acknowledgments

This research has been supported by the Australian Research Council. I am grateful to Mike Cammeron-Jones for discussions that helped refine the ideas presented herein. The Breast Cancer, Lymphography and Primary Tumor data sets were compiled by M. Zwitter and M. Soklic at University Medical Centre, Institute of Oncology, Ljubljana, Yugoslavia. The Audiology data set was compiled by Professor Jergen at Baylor College of Medicine.

References

Clark, P., & Boswell, R. (1991). Rule induction with CN2: some recent improvements. In Proceedings of the Fifth European Working Session on Learning (pp. 151-163).

Clark, P., & Niblett, T. (1989). The CN2 induction algorithm. Machine Learning, 3, 261-284.

Holte, R. C., Acker, L. E., & Porter, B. W. (1989). Concept learning and the problem of small disjuncts. In Proceedings of the Eleventh International Joint Conference on Artificial Intelligence (pp. 813-818). Detroit: Morgan Kaufmann.

Murphy, P., & Aha, D. (1993). UCI Repository of Machine Learning Databases. Irvine, CA: University of California, Department of Information and Computer Science.

Plotkin, G. D. (1970). A note on inductive generalisation. In B. Meltzer & D. Mitchie (Eds.), Machine Intelligence 5 (pp. 153-163). Edinburgh: Edinburgh University Press.

Quinlan, J. R. (1993). C4.5: Programs For Machine Learning. Los Altos: Morgan Kaufmann.

Rivest, R. L. (1987). Learning decision lists. Machine Learning, 2, 229-246.

Shen, W.-M. (1992). Complementary discrimination learning with decision lists. In Proceedings Tenth National Conference on Artificial Intelligence (pp. 153-158). Menlo Park, Ca: AAAI Press.

Van Horn, K. S., & Martinez, T. R. (1993). The BBG rule induction algorithm. In C. Rowles, H. Liu, & N. Foo (Eds.), AI'93 (pp. 348-355). Singapore: World Scientific.

Webb, G. I. (1993a). OPUS: A systematic search algorithm and its application to categorical attribute-value data-driven machine learning (Technical Report No. C93/35). Deakin University School of Computing and Mathematics.

Webb, G. I. (1993b). Systematic search for categorical attribute-value data-driven machine learning. In C. Rowles, H. Liu, & N. Foo (Eds.), AI'93 (pp. 342-347). Singapore: World Scientific.

Webb, G. I., & Brkic, N. (1993). Learning decision lists by prepending inferred rules. In Proceedings of the AI'93 Workshop on Machine Learning and Hybrid Systems (pp. 6-10). Melbourne.