Acquiring Process Knowledge in Hybrid Planning Domains using Machine Learning

Rubiya Reba[1] and **Rabia Jilani**[2] and **Alan Lindsay**[3] and **Thomas L. McCluskey**[4]
[1,2,4]School of Computing and Engineering, University of Huddersfield.
{rubiya.reba, r.jilani, t.l.mccluskey}@hud.ac.uk
[3]School of Mathematical and Computer Sciences, Heriot-Watt University.
Alan.Lindsay@hw.ac.uk

Abstract

Recent progress in Automated Planning not only involves improvements in the efficiency of planning engines but also in the granularity in which domains can be modelled. An important direction is in the move from discrete domain models to discrete and continuous (hybrid) models. While planning with hybrid models has been studied for decades, the engineering of those domain models remains a problem, particularly for complex domains. It is imperative to understand how to effectively and efficiently formulate planning problems to achieve maximum productivity with minimum wasted effort or cost. One of the main engineering challenges of hybrid domain models involves encoding the frequent fluctuation of the underlying processes with the continuous updates in the world state. The occurring numerical changes are complex enough that cannot be handled by human efforts every single time it happens.

To automatically learn accurate and run-time representative estimation of the continuous change, this paper proposes a general machine learning method of process learning to estimate the effects of numeric variables on continuous assignment expression using regression analysis. Our method exploits cross-sectional data to learn features of the regression model and enhance the already existing domain models. We empirically evaluated our approach on urban traffic management domain and coffee domain. We prove that the automatically learned values of continuous variables by our system are more rational than the formulation values of continuous variables declared statically in the initial states. We demonstrate in our evaluation that the learned knowledge provides more accurate simulation, which can lead to higher-quality plans.

Introduction

Planning with discrete and continuous (hybrid) domain models has been studied for decades, the engineering of these models remains a problem, particularly for complex domains and real-life applications (e.g. planning space missions, operation of underwater vehicles, urban traffic control etc). Automated Planning with Hybrid Domains (APHD) has become more acceptable due to the need in specific domains to express discrete-continuous changes. Although APHD has the potentiality to apply in real applications, the problems surrounding the modelling and maintenance of the time-dependent hybrid domain models has become more apparent than ever.

Critical to the success of any AI planning application is that the domain model adequately and accurately represents reality. For hybrid planning, although PDDL+ (Fox & Long 2002) has been adapted widely to model real-time autonomous systems due to granularity of its continuous representation, it is still a challenge for the knowledge engineers to handle its expressivity for process description. This concerns the learning and representation of hybrid operator schema, discrete or continuous resources, and processes and events involved e.g., domains like urban road traffic management, stock market, etc.

Knowledge Engineering (KE) of planning domain models using Machine Learning (ML) techniques is considered as a paramount for empowering autonomous learning systems with the capacity to fill implicit human knowledge gaps and errors, requiring least human intervention. The area of ML application to domain model learning systems has received active research attention in recent years but did not make as much stride as the learning of control knowledge. Not enough depth of research into the area has made it an adhoc process, where the skills of knowledge engineers significantly influence the quality of the resulting planning application and the accuracy of domain model still counts as a bottleneck for AI planning (McCluskey, Vaquero, & Vallati 2017).

To effectively learn the PDDL+ process models, the main focus of our research is the use of ML techniques with an emphasis on statistical modelling using regression analysis in hybrid hypothesis space. The learning system automatically acquires the model of inter-dependencies between continuous variables by analysing historical time series data set. It infers the relationships between the outcome continuous variable and predictor (one or more) features in the domain processes. To illustrate the feasibility of our method and to evaluate it on a real planning application, we use the PDDL+ encoding of Urban Traffic Control (UTC) domain and the Coffee domain. Both these domains use start-process-stop procedure to model continuous traffic flow turn-rate and yield of the coffee, respectively. We empirically evaluated our method using ENHSP (Scala *et al.* 2016) PDDL+ planner to generate simulation with the learnt process models. The evaluation demonstrates that this simulation is more rational and closer to the behaviour of the actual processes modelled without having to declare continuous change knowledge statically (Bryce, Benton, & Boldt 2016).

Description and Running Examples

A PDDL+ process simulates continuous changes in the numeric variables that are initiated by changes in the world. Learning dynamic values of numeric variables in PDDL+ specification is novel, and learning this automatically has hardly been done (Lindsay *et al.* 2020). Instead of real-time prediction and assignment of numeric variables values, these tend to be declared as static knowledge in the problem definition by knowledge engineers. This leads to reduced rationality in results and mismatch between the output from planner generated simulation and the real-time sensor output values.

In order to overcome this problem formulation drawback, our goal is to enhance the PDDL+ problem specification by learning the values of numeric variables in process effects automatically. We learn from real-time and historical data sources. Problem assumes that there is an existing hybrid domain model with a set of processes modelling the continuous change in application.

Our method is fundamentally based on the ML project workflow and is summarised in figure 1. Data sources used for each domain are discussed in the next running examples section. The initial stage of the process involves use-case driven data enrichment and preparation. It also includes extraction of representative sets of features to train the regression model. In terms of PDDL+ model specification, this hypothesize the independent numeric variables (primitive numeric expression) to include or exclude from a regression equation to attain the trade-off between being less biased and most precise model. In the UTC domain, for example, predicting turn-rate accurately across a junction may not only depend on the duration of the green signal (the longer the green time, usually the turn-rate increases - but not always!) but also on (potentially) the occupancy of nearby links, state of signal stages on nearby links, length of nearby links etc. This is done by employing various statistical tests and techniques discussed in the upcoming sections.

Cross-validation has been used to estimate the skill of learnt models on test data. With iterative training and testing procedure, we tune the hyper-parameters of the model to attain better rationality in the simulation results. The original encoding of the process model is augmented with the generated hybrid automata of continuous behaviour using PDDL+ operational semantics of assignment propositions.

UTC Domain

As a traffic case study, we have exploited the UK research council funded SimplifAI project. In SimplifAI, a hybrid planning based traffic control system has been introduced to handle the vehicle flow in both regular and unexpected road situations (Vallati *et al.* 2016). (McCluskey & Vallati 2017) demonstrates the effectiveness of this hybrid planning-based approach with PDDL+ formulation of the UTC domain. We used PDDL+ model of traffic flow based on the same series of work in SimplifAI project. An industry-standard, proprietary simulator called AIMSUN (Barceló & Casas 2005) is used to provide process training data based on the microscopic model obtained from transport authorities.

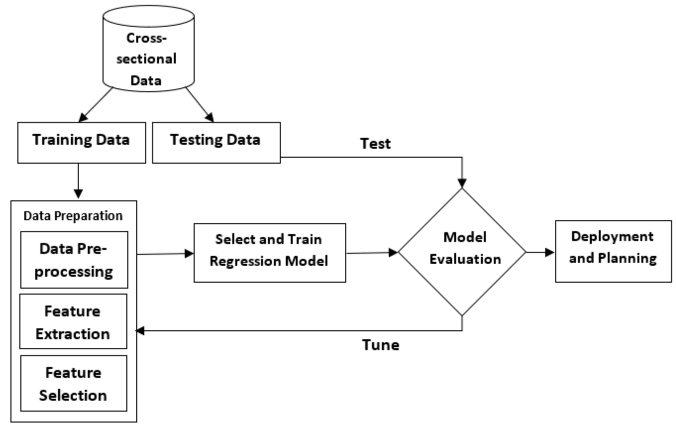

Figure 1: Abstract method architecture used to generate and employ regression models in hybrid planning domain.

In PDDL+ formulation of urban traffic control, hand-written continuous processes are used to model the flow of cars, to efficiently reduce congestion of specified roads (links) by controlling traffic light green phases (Figure 2). The link (uni-directional part of a road) roads leading to and leaving from a junction are referred to as road1 (r1) and road2 (r2), respectively. The process *flowrun_green* models the turn-rate from *r1* to *r2* through the phase *p* for each green time and is measured in standardized vehicles (PCU) per second. Complete details of the UTC domain along with other actions and events that control the *flowrun_green* process is available in (McCluskey & Vallati 2017). We refer this version of the domain as original domain model onward.

In the original formulation of the UTC domain, the process description for traffic flow between links across a junction is too coarse and assumes instantaneous traffic flows across each link. There is also static knowledge missing from the model (e.g. link lengths, or numbers of lanes in a link). At the same time, the principle continuous variable *turn-rate* is defined as a predefined static value in the problem definition.

To learn the current, accurate and representative value of the turn-rate, our objective is to determine the efficient linear approximation (rate of change) of vehicles turn-rate across each road junction. We analyse the real-time data taken from AIMSUN and attempt to refine PDDL+ process simulation by learning dynamic turn-rate of cars as a function of other numeric variables. We achieve this by estimating the causal relationship and multicollinearity between the controlled variable (turn-rate) and the predictor variables, i.e. the saturation of *r1* and *r2*, inter time and active green time of a phase *p*.

The hybrid automata of our target process representation for *flowrun_green* is presented in figure 3. It models the flow of traffic across a junction as a continuous process. The learnt values of β are declared as constants in the problem definition.

```
(:process flowrun_green : parameters (?p - stage ?r1 ?r2 - link)
    :precondition (and (active ?p)
                       (> (occupancy ?r1) 0.0)
                       (> (turnrate  ?p ?r1 ?r2 ) 0.0)
                       (< (occupancy ?r2) (capacity ?r2)))
    :effect (and (increase (occupancy ?r2) (* #t (turnrate  ?p ?r1 ?r2 )))
            (decrease (occupancy ?r1) (* #t (turnrate  ?p ?r1 ?r2)))))
```

Figure 2: flowrun_green process of Traffic domain.

```
(:process flowrun_green :parameters (?p - stage ?r1 ?r2 - link ?i - junction)
    :precondition (and (active ?p) (contains ?i ?p) (> (occupancy ?r1) 0.0)
                  (< (occupancy ?r2) (capacity ?r2)))
    :effect (and (assign (turnrate  ?p ?r1 ?r2 ) (+ (B0 ?p ?r1 ?r2)
                 (+ (* (B1 ?p ?r1 ?r2) (greentime ?i) )
                 (+ (* (B2 ?p ?r1 ?r2) (intertime ?i))
                 (+ (* (B3 ?p ?r1 ?r2) (/(occupancy ?r1) (capacity ?r1)))
                 (* (B4 ?p ?r1 ?r2) (/(occupancy ?r2) (capacity ?r2)))))))
        (increase (occupancy ?r2) (* #t (turnrate ?p ?r1 ?r2)))
        (decrease (occupancy ?r1) (* #t (turnrate ?p ?r1 ?r2)))))
```

Figure 3: Target flowrun_green process of Traffic domain. Coloured text identify the learnt regression model for the process

Coffee Domain

As a hybrid domain case study, we formulated the PDDL+ version of the Coffee domain partially inspired by its PDDL2.1 version and by Easthope's work in (Easthope 2015). The PDDL+ formulation of domain demonstrates the effectiveness of the hybrid planning-based approach in the coffee making. It complies with the latest coffee research by the National Coffee Association of U.S.A., Inc. (NCA 2020).

In the PDDL 2.1 formulation, the only continuous change function is temperature, and its value is declared statically in the initial conditions to get a generic cup of coffee as a goal. Figure 4 presents the PDDL 2.1 formulation of durative action *makecoffee*.

We exploit observational data by (Easthope 2015), which illustrates that using different water temperatures for brewing coffee grounds has a significant effect on the taste (yield) and extraction of espresso. We extended Easthope's study in line with (Roman Corrochano 2017), which confirms that both brewing temperature and time have a vital influence on the outcome yield of the coffee. The PDDL+ formulation brews coffee according to the expected yield of the user with brew temperature and time as the continuous functions/features (NCA 2020). Figure 5 presents target representation of process *brewing* in PDDL+ formulation. We developed our ground truth model to check the learning results for accuracy compared to the real world.

The aim is to automate the estimation (rate of change) of yield, which is achieved by the process *brewing*. We

```
(: durative-action makecoffee : parameters (?c - coffee ?w - water)
    :duration (>= ?duration 1)
    :condition (and  (at start (boiled ?w))
        (over all (>= (temperature ?w) 60))
        (over all (<= (temperature ?w) 80))
        (at start (havecoffee ?c)))
    :effect (and (at end (madecoffee ?c ?w))))
```

Figure 4: Brewing process of Coffee making domain (PDDL 2.1).

```
(:process brewing :parameters (?c - coffee ?w - water)
    :precondition (and (brewing ?c ?w))
    :effect (and (assign (yield ?c ?w) (+ (B0 ?c ?w)
                 (+ (* (B1 ?c ?w) (temperature ?w))
                 (* (B2 ?c ?w) (brew-time ?c ?w)))))
        (increase (brew-time ?c ?w) (* #t 1))))
```

Figure 5: Target brewing process of Coffee domain (PDDL+). Coloured text identify the learnt regression model for the process.

do this by estimating the multiple regression model between the controlled variable (yield) and the predictor variables i.e. the variation of `(temperature ?w)` and `(brew-time ?c ?w)` which are highly linearly related.

Learning for Modelling Continuous Effects

The most used aspect of regression models is for predictive analytics and forecasting, time series modelling, and for finding the causal effect relationship between the dependent and independent variables. Out of many kinds of regression techniques available usually, they are chosen by the three primary metrics: the shape of the regression line, type of the dependent variable and the number of independent variables.

To learn the process model, the critical contribution of our work is to automate the estimation of the control variable after selection of the appropriate feature set automatically. The system explores the continuous features that best impact the value of the control variable in the amount of variability caused by multicollinearity in continuous features (that change in unison).

The regression model represents our process learning hypothesis. We exploit the regression techniques to conduct time series modelling and finding the causal effect of numerically changing variables on the continuous function. We do this by fitting a line to the data points, in such a manner that the differences between the distances of data points from the line are minimized. This is to understand the underlying relationships and structures that produce the observed data.

We regenerate the concise model of the continuous behaviour that complies with PDDL+ semantics. This demonstrates that the use of regression analysis can produce more accurate and automatically learnt model of numeric vari-

Input: Time Series Data, $TSD = \{y, X\}, i.e.$
Regressand, $y = \{y \mid y \in \mathbb{Q}\}$
Regressors, $X = \{x_1, x_2, \ldots, x_n \mid x \in \mathbb{Q}, n \in \mathbb{N}_1\}$
Output: Regression Model, $\hat{y} = \beta_0 + f(\beta_{x_i}, x_i)$
where, β_0 = Intercept,
β_{x_i} = Slope (Regression coefficient of y on x_i),
$x \in X$ and $i = \{1, 2, \ldots, n(X)\}$

```
1:  procedure Start(TSD)
2:      PCC ← PearsonCorrelation(TSD)
3:      if ∀|r_{y,X}| > 0.3 AND n(X) is 1 then
4:          p_f ← ANOVAtest(y, X)
5:          p_t ← T-test(y, X)
6:          if (p_f, p_t) < 0.05 then
7:              (β_0, β_1) ← LinearRegression(y, X)
8:          end if
9:      end if
10:     if ∃|r_{y,X}| > 0.3 AND n(X) > 1 then
11:         if ∀|r_{x_i,x_j}| < 0.3 then
12:             (β_0, β_{x_i}) ← StepwiseRegression(y, X)
13:         end if
14:         if ∃|r_{x_i,x_j}| > 0.3 then
15:             VIF_{x_i} ← MulticollinearityTest(X)
16:             if 1 < ∀(VIF_{x_i}) <= 5 then
17:                 (β_0, β_{x_i}) ← StepwiseRegression(y, X)
18:             end if
19:             if ∃(VIF_{x_i}) > 5 then
20:                 (β_0, β_{x_i}) ← RidgeRegression(y, X)
21:             end if
22:         end if
23:     end if
24: end procedure
```

Figure 6: Method for the process model learning.

ables in the process description. This section uses a step-by-step approach to describe the process.

Model specification and Feature Identification

In the PDDL+ model specification, we determine which independent variables to include and exclude from a regression equation to attain the trade-off between being less biased and most precise model. We use various metrics and algorithms for a model specification that also align with the domain-specific theoretical concerns.

In the regression model, dependent variable (Y_i) is a function of independent (explanatory variable in terms of causal effect) variables X_i and β with ε representing an additive error term for statistical noise:

$$Y_i = f(X_i, \beta) + \varepsilon_i$$

Our target learning is to find $f(X_i, \beta)$ that most closely fits the data. Figure 6 shows the method for the process learning procedure and presents the basic approach adopted in this work. The steps taken in the method are explained in the relevant upcoming sections with the line numbers.

Pearson's Correlation Coefficient Pearson correlation coefficient (PCC) r_{xy} statistically infers how strong a relationship is between two variables based on the values of r_{xy}:

	Yield %	Brew Temp (°C)	Brew Time (sec)
Yield %	1		
Brew Temp (°C)	0.551416365	1	
Brew Time (sec)	0.529255767	-0.036273813	1

Figure 7: Correlation test (Pearson's r) for Coffee domain.

S0_link1	Turnrate	Greentime	Intertime	Road1_Sat
Greentime	-0.531443386			
Intertime	0.00353702	-0.187231537		
Road1_Sat	0.318631236	-0.76003397	-0.10083	
Road2_Sat	0.481740059	-0.405333726	-0.06429	0.093616259

Figure 8: Correlation test (Pearson's r) for UTC domain. The data is for the link (r1 - r2) of a particular phase of a particular junction.

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2}\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

where: n is sample size, x_i, y_i are the individual sample points indexed with i and $\bar{x}$ and $\bar{y}$ are the sample mean.

Varying in the range between -1 and +1, r_{xy} measures the magnitude of association, or correlation, as well as the direction of the identified features' relationship. Figure 7 and 8 illustrates the results of the correlation test for Coffee and UTC domain, respectively. The cutoff values for r signifies strong relationship for values between $\pm$ 0.5 and $\pm$ 1, the moderate relationship between $\pm$ 0.5 and $\pm$ 0.3 and a weak relationship between $\pm$ 0.3. It signifies no relationship (H_0) at r = 0 . The tables represent this by the use of green (stable) and red (weak) highlighting. Line 2 of method conducts the correlation test while line 3 and 10 later use these results to decide the particular regression type needed.

Regression Techniques based on Relationship Significance

In this observational study, we draw inference from a sample to a population where the independent variable is not under our control for logistical constraints.

From the values of above mentioned statistical test (PCC) and out of innumerable forms of regression, this section discusses the types of regression analysis that best suit our learning problem based on the type and number of independent variables. We consider the cases when:

1. y is related to a single x-variable: Linear regression
2. y is related to multiple x-variables: Stepwise regression
3. y is related to multiple x-variables where x variables have intercorrelation: Ridge regression

Type 1 and 2 analyse variance using the ANOVA test and type 3 multicollinearity test, i.e. VIF (Variance Inflation Factor) test.

1. **y is related to single x-variables:**
   This is the most common use case when the change in a single x variable impacts a continuous dependent variable. We exploit simple linear regression to understand

Coffee Domain	Yield, Btemp (yield, x1)	Yield, Btime (yield, x1)	Yield, Btemp, Btime (yield, x1, x2)
P-value	0.00522	0.00782	0.00043, 0.00062
Multiple R	0.55141	0.52925	0.77855
R Square	0.30406	0.28011	0.60614
Adjusted R Square	0.27242	0.24738	0.56863
Standard Error	0.33272	0.33839	0.25619

Figure 9: Anova test for Coffee domain (n = 24).

the mean change in a dependent variable given a one-unit change in each independent variable. We accomplish this by Least Square Method.

r_{xy} does not identify whether a change in one variable is directly caused by the other variable. To measure the significance of relationship by analysis of variance between variables, we conducted a one-way ANOVA test to observe the values of F and T-stat. For hypothesis testing, we reduced the model using the p-value. A small p-value (typically $\leq$ 0.05) indicates strong evidence against the null hypothesis. On the contrary, a significant p-value ($>$ 0.05) indicates weak evidence against the null hypothesis (H_0), in which case we discard our H_a. For example, for Coffee domain:

- H_0 (null hypothesis): change in brewing time does not affect yield

- H_a (alternative hypothesis): change in brewing time affects yield along with brewing temperature.

We evaluate the model performance using the metric R-square (R^2) to check how similar a regression line is to the data it is fitted in. R^2 value ranges from 0 to 1. For R^2 = 0.50, approximately half of the observed variation can be explained by the model.

The standard error (S) is a numeric assessment of how well the regression model fits the sample data. For the estimated R^2 and S values for the linear, we choose the models with greater R^2 and less S value.

The objective function of the Linear regression model is:

$$Y_i = \beta_0 + \beta_1 x_i + \varepsilon_i, \quad i = 1, \ldots, n.$$

where,

- β_0 = y-intercept (regression constant);
- β_1 = slope (regression coefficient for linear effect of x on y) and
- Y_i = estimated (or predicted) value

Line 3-9 of method compute these steps.

2. **y is related to multiple x-variables:**

The stepwise regression is a step-by-step method of selecting the most significant independent variables to handle higher dimensionality of data set using a set criterion such as highest absolute t-value.The most significant independent variables are retained in the regression model after selection using forward elimination.

Figure 9 shows the result of the hypothesis test for the Coffee domain, where independent variables include brewing temperature and time. The significance of influence is measured by F-test and T-test based on their p-values. If the p-value is less than 0.05, then the stepwise regression is applied to get the multiple regression model specified below.

$$Y_i = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_k x_k$$

where,

- β_i = the partial contributions of each of the x-variables;
- k = number of modelled factors;

Line 10-13 of method compute these steps.

3. **y is related to multiple x-variables where x variables have inter-correlation:**

To cover one of the weaknesses of linear regression like sensitivity to both outliers and multicollinearity, we used more advanced variants of linear regression, i.e. ridge regression. This is the case when each of the multiple x-variables contains a unique piece of information about y, and the x-variables are inter-correlated (also known as data multicollinearity). This can be seen in figure 8 where on one hand dependent variable turnrate demonstrates relationship with independent variables, inter-correlation between the independent variables (Green time, r1 saturation and r2 saturation) can also be seen.

The potential complexity with multicollinearity is that it is difficult to calculate the coefficient estimate of one independent variable without considering the other independent variable(s) that changes in unison.

We conduct VIF (Variable Inflation Factor) test to identify the variables that are affected by multicollinearity, the strength of the correlation, and to estimate if multicollinearity needs fixing (only in critical intercorrelation).

$$\mathrm{VIF}_i = \frac{1}{1 - R_i^2}$$

A rule of thumb for interpreting the VIF is:

- VIF $\leq$ 1: not correlated
- 1 $<$ VIF $<$ 5: moderately correlated
- VIF $>$ 5: highly correlated

For the UTC domain, we calculated the VIF by considering each predictor and regressing it against every other predictor in the model to find the variance. We discuss the complete analysis of a roads junction in the upcoming analysis section. Line 14 onward in figure 6 measure multicollinearity and applies the appropriate regression technique.

We exploit stepwise regression if the x-variables are moderately correlated. Line 16-17 of method measure and applies stepwise regression. For high correlation, ridge regression is applied. Ridge regression is an alternative procedure to ordinary least squares (OLS) for analysing multiple regression with multicollinearity. It improves efficiency in parameter estimation problems in exchange for a tolerable degree of bias (penalty function, α) to reduce the standard error. Line 19-20 of method measure and applies ridge regression.

	Coefficients	Standard Error	t Stat	P-value	Lower 95%	Upper 95%
Intercept	3.95300285	2.738544286	1.443469	0.16364	-1.742112	9.648117
BTemp (°C)	0.09757356	0.023402588	4.169349	0.00043	0.048905	0.146242
BTime (sec)	0.2044137	0.050934066	4.0133	0.00063	0.098491	0.310337

Figure 10: Multiple regression model for coffee domain.

Empirical Analysis and Evaluation

As evidence of the concept of learning by the application of our system, this section includes empirical analysis and evaluation of the proposed method on the running examples.The focus of the analysis is to test the effectiveness of the learnt process model to enhance simulation accuracy. We check the simulation accuracy of the learned and original model and compare it against the AIMSUN output (considering AIMSUN as a gold standard for the UTC domain).

Using the same set of problems for the learned and original domain we exploited ENHSP (Scala *et al.* 2016) planner to generate simulations. Evaluation demonstrates that the learned domain model is more rational and pragmatic without having to declare continuously changing knowledge statically.

Coffee Domain

As explained in the running example section, we formulated the PDDL+ version of the Coffee domain. Bonafide coffee brewing requires using the right quantity of precisely ground coffee, controlled by the correct brewing time and temperature. We aim to gain precise control over the users' expected yield by calculating the accurate estimates of the variation in predictor variables, i.e. brew time (BTime) and temperature (BTemp). Figure 7 illustrates the PCC values for brew time and temperature that signifies their strong relationship with the yield of the coffee.

Figure 10 illustrates the final multiple regression results for the three-variable model (yield, BTemp, BTime). Very small p-values reject the null hypothesis and thus formulates the scaling factors by which each brew time and temperature affects the yield of the coffee. No multicollinearity has been detected between control and predictor variables, the regression model turns out to be straight forward:

$$Yield = \beta_0 + (\beta_1 * BTemp) + (\beta_2 * BTime)$$

To evaluate the skills of learnt process models, we employed k = 5 fold cross-validation model metric on a limited data set. Figure 11 illustrates the bar graph displaying coffee yield at different temperatures. Error (%) is the difference between observed and predicted yield with brew time and temperature as predictors.

UTC Domain

To empirically analyse the UTC domain we exemplify a set of active traffic junctions of a selected urban region as shown in figure 12. We systematically decided this region (and the number of active junctions shown as blue circles) by keeping in view the plan generation and simulation capability while testing the system. Each junction has various phases, and

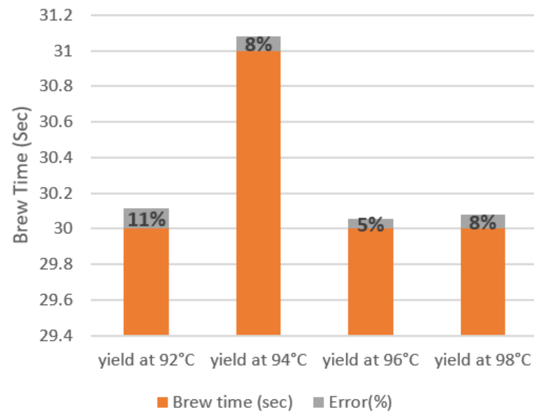

Figure 11: Bar graph displaying Brew Time (sec) with corresponding yield error (%) at various temperatures.

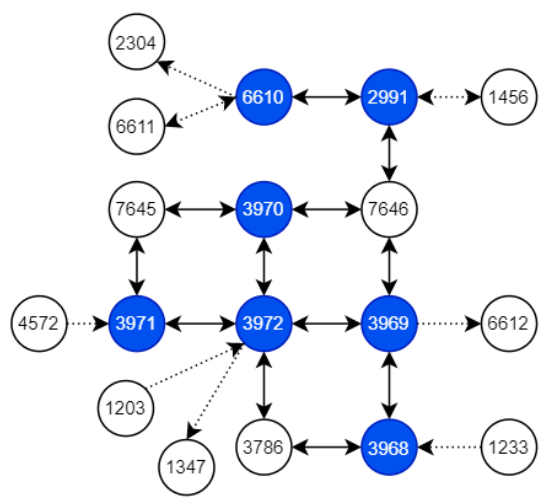

Figure 12: Abstract view of UTC region for evaluation. Blue circles shows active junctions. The arrows represent the direction of traffic flow

each phase has a variable number of links connecting the corresponding roads. We select junction 3969 (figure 13) to illustrate and report the empirical results. Junction 3969 has three phases: s0, s1 and s2, where each phase has various links. For the sake of brevity, we present full tests results for s0 and only final results for s1 and s2.

Eight cross-sectional data sets (including the contemporaneous values of road saturation, active/green phase time and inter-time) are generated for the considered network from AIMSUN in the form of variations in traffic flow over phases of the junctions. Each data set contains one hour (3600 seconds) long traffic observations recorded every 5 seconds. Each data set is generated using a unique set of initial states. Each plan provided around 10,000 data points. One simulation output (data) set has been exploited for training, while the remaining seven have been used for testing. Each set contains data about the topology of the road links, vehicle capacity of all links, minimum-maximum green time of sig-

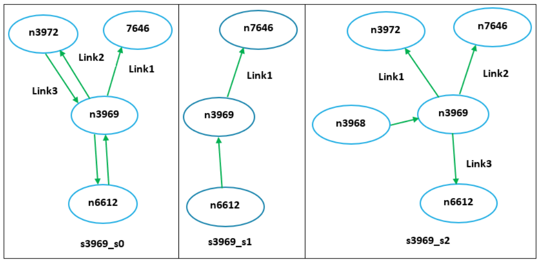

Figure 13: Junction n3969: Phase s0, s1 and s2.

s0_link2	Turnrate	Greentime	Intertime	Road1_Sat
Greentime	-0.0442			
Intertime	-0.0185	-0.187231537		
Road1_Sat	-0.07537	-0.76003397	-0.10083	
Road2_Sat	-0.25189	-0.062018813	-0.02625	0.161620153

Figure 14: Pearson r for Phase s0: link 2.

nal phase, traffic signal state (active or inter time) etc.

Intuitively, for predicting an accurate turn-rate of a phase, it may not only correlate with the active green time but also with the saturation of the connecting roads. Saturation of a road link in turn also depends on several co-factors, e.g. effects of signal phase on the nearby links, number of access points to the link, density of bus stops on the link (if any), speed-limit, commuter/non-commuter traffic and the varying structural dynamics and types of the roads leading to and leaving away from a junction, e.g. a varying number of lanes, length and occupancy of nearby links, low capacity roads with parking spaces, primitive roads, intersecting roads, fork in the road etc. This is also illustrated by the PCC values for the phase s0 links in the junction 3969 discussed below.

PCC Figure 8, 14 and 15 illustrates the PCC for link 1 and link2 and 3 of phase s0, respectively. For link 1 of s0, turn-rate moderately depends on three independent variables while there is a strong correlation between the green time - road1sat. and road1sat. - road2sat. pairs. For link 2, turn-rate apparently does not depend on any of the independent variables while there is a strong correlation of green time with the saturation road1. For link 3, turn-rate moderately depends on road1sat. and moderately on road2sat. while there is a moderate correlation between green time and road1sat. The PCC values indicate multicollinearity, while the varying degree of relationships' strength indicates that both r1 and r2 in each of the links are different types of roads. The turn-rate for each link proportionately depends on predictor variables for the difference in road topology.

VIF Figure 16 illustrates the VIF values for the links of phase s0. For the sake of brevity, X1 represents green time when regressed against inter time, road1 sat. and road2 sat. Similarly, X2 represents VIF for inter time, X3 for road1 sat. and X4 for road2 sat.

For phase s0, all X variables for link 1, 2 and 3 have 1 $<$ VIF $<$ 5. Therefore, all the variables are moderately in-

s0_link3	Turnrate	Greentime	Intertime	Road1_Sat
Greentime	-0.27526			
Intertime	-0.03644	-0.187231537		
Road1_Sat	0.619692	-0.367450258	-0.0507	
Road2_Sat	0.390038	0.06317072	-0.05207	0.158197155

Figure 15: Pearson r for Phase s0: link 3.

S0	Link1	Link2	Link3
X1 (greentime)	4.36891	2.86916	1.23525
X2 (Intertime)	1.37108	1.21335	1.05463
X3 (road1 saturation)	3.51405	2.85702	1.22118
X4 (road2 saturation)	1.55625	1.03716	1.04414

Figure 16: VIF values for phase s0 of Junction n3969.

tercorrelated. For the only link of phase s1, green time and road1 sat. values demonstrated strong intercorrelation (VIF $>$ 5) while all the links of phase s2 also showed moderate intercorrelation.

Regression Models

Based on the calculated VIF values in figure 16 this section reports the regression models for the junction 3969.

Figure 17 illustrates the values of intercept, estimated slops and errors for the regression models of three phases of junction 3969. Regression models for all the links of s0 and s2 are produced using multiple stepwise regression while s1 model is learnt using ridge regression.

As exploited in figure 3 for the PDDL+ process encoding in UTC domain, the generic form of the regression equation for all the links with stepwise model is following:

$$Turnrate = \beta_0 + (\beta_1 * Green_time) + (\beta_2 * intertime)$$

$$+ (\beta_3 * r1sat) + (\beta_4 * r2sat)$$

For ridge regression, we attain bias-variance trade-off using structural loss minimisation. It minimises both model loss and model complexity by adding a shrinkage penalty to the ordinary least square loss function to limit its squared L2 norm.

$$L(\overline{w}) = ||X\overline{w} - \overline{y}||_2^2 + \alpha||\overline{w}||_2^2$$

where $||X\overline{w} - \overline{y}||_2^2$ is a loss function and $\alpha||\overline{w}||_2^2$ is penalty term of the ridge regression. X is a sample matrix, w is a vector of coefficients, $X\overline{w}$ is an estimated value and $\overline{y}$ is the actual sample value. α is a penalty constant whose value can range from 0 to positive infinity. The most suitable value of α is where R^2 has the least value.

Figure 18 illustrates the turn-rate error (averaged for 7 test sets) comparison between learnt and original model with the turn-rate observed in AIMSUN after 200, 400, 600 and 800 seconds of simulation. Results stipulate that the learnt model improves simulation accuracy and produce less error across each of the network data sets as compared to the original domain model with static turn-rate across the network.

S0	Link1 SW	Link2 SW	Link3 SW
β_0 (intercept)	1.22277	0.08617	0.11398
β_1 (greentime)	-0.00689	-0.00034	-0.00143
β_2 (Intertime)	-0.36399	-0.04064	----------
β_3 (road1 saturation)	-0.29320	-0.06422	3.50341
β_4 (road2 saturation)	0.78620	-0.05864	1.74021
RMSE	0.35722	0.04200	0.35802
R-square	0.37373	0.09091	0.47975

S1	Link1 RR
β_0 (intercept)	0.06984
β_1 (greentime)	0.00005
β_2 (Intertime)	0.00256
β_3 (road1 saturation)	-0.05531
β_4 (road2 saturation)	-0.07224
RMSE	0.03957
R-square	0.13984

S2	Link1 SW	Link2 SW	Link3 SW
β_0 (intercept)	1.55359	0.41117	0.47791
β_1 (greentime)	----------	----------	-0.01131
β_2 (Intertime)	-0.08238	-0.05719	-0.04729
β_3 (road1 saturation)	-1.45545	-0.34248	-0.28681
β_4 (road2 saturation)	-0.81770	3.24449	----------
RMSE	0.27407	0.28077	0.23846
R-square	0.47688	0.27195	0.12591

Figure 17: Estimated coefficients of regression model for junction 3969. SW: Stepwise regression; RR: Ridge regression.

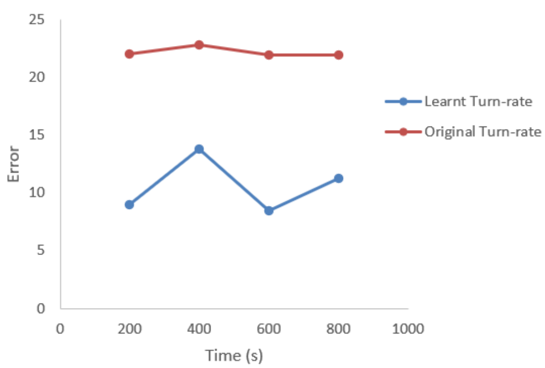

Figure 18: Error comparison between learnt and original domain models with the turn-rate observed in AIMSUN after 200, 400, 600 and 800 seconds of simulation.

Related Work

At state of the art, there exist several systems for automated acquisition of planning domain models. These include systems for synthesizing, refining or improving domain models at various stages of the planning process. (Jilani *et al.* 2014) critically compare several automated domain model acquisition (DMA) tools on a set of criteria consisting of input requirements, learned output domain component, learning efficiency, supported language, techniques used, ability to handle noisy plans, ability to refine existing models, user experience and availability.

In DMA, it has been common to assume accurate input data. This has allowed inductive and analytical learning approaches to be used most frequently by extrapolating from sample input plans as the evidence to make a probabilistic claim about all or most of the learned knowledge, e.g., (Cresswell & Gregory 2011). In recent work, researchers have examined noisy data, exploiting clustering (Lindsay *et al.* 2017), machine learning (Zhuo & Kambhampati 2013), and deep learning (Asai & Fukunaga 2018) as part of their processes. DMA has progressively considered richer target fragments of the PDDL language, from propositional (Wu, Yang, & Jiang 2007; Cresswell & Gregory 2011), including ADL (Zhuo *et al.* 2010); to learning action costs (Gregory & Lindsay 2016) and numeric constraints (Segura-Muros, Pérez, & Fernández-Olivares 2018).

The area of ML application to DMA systems has received active research attention in recent years but did not make as much stride as the learning of control knowledge. There is published work in learning discrete domain knowledge from execution traces as input training data since the early work in (Benson 1996) to the very recent work (Suárez-Hernández *et al.* 2020). Comparatively, there is very less work in learning or practical refinement of hybrid domain models. (Lindsay *et al.* 2020) in their recent work used ML to refine the already engineered hybrid domain models by identifying the varying situation and temporal features in the process effects. The system learns from the observation data of past executions and fits in an improved set of processes to refine the existing once. (Denenberg & Coles 2018) based their work on how better hybrid process modelling can improve hybrid planning performance by demonstrating three different methods of continuous effects cascading in hybrid domains.

Conclusion and Future Work

For hybrid planner reasoning, adequate modelling of the continuous effects in the domain model is necessary to capture the critical features (Fox & Long 2006). Measurement of process variables are essential in control systems to control a process. This paper proposes a dynamic and versatile approach to automatically learn accurate and run-time representative estimation of the continuous change in PDDL+ process modelling. It assists knowledge engineering process by choosing effective process parameters and removing the irrelevant once. The learning system automatically acquires the model of inter-dependencies between continuous variables by analysing historical time series data set using regression analysis.

To illustrate the feasibility of our method and to evaluate it on a real planning application, we utilise the PDDL+ encoding of Urban Traffic Control (UTC) domain and the Coffee domain. The evaluation demonstrates that the approach leads to a more realistic and accurate simulation provided there is an adequate and representative data set available. We intend to enhance the PDDL+ problem specification by learning the run-time representative values of continuous variables automatically instead of defining them as static knowledge in the problem definition.

For future work, we aim to extend our approach with more functional forms to add flexibility and accuracy in higher dimensional curve fitting for process modelling, e.g. non-linear regression or support vector regression for continuous high dimensional data that changes over time.

References

Asai, M., and Fukunaga, A. 2018. Classical planning in deep latent space: Bridging the subsymbolic-symbolic boundary. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Barceló, J., and Casas, J. 2005. Dynamic network simulation with aimsun. In *Simulation approaches in transportation analysis*. Springer. 57–98.

Benson, S. S. 1996. *Learning action models for reactive autonomous agents*. Ph.D. Dissertation, stanford university PhD thesis.

Bryce, D.; Benton, J.; and Boldt, M. W. 2016. Maintaining evolving domain models. In *Proceedings of the twenty-fifth international joint conference on artificial intelligence*, 3053–3059.

Cresswell, S., and Gregory, P. 2011. Generalised domain model acquisition from action traces. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*.

Denenberg, E., and Coles, A. 2018. Modelling sequences of processes in pddl+ for efficient problem solving. *Proceedings of KEPS-2018* 70–77.

Easthope, A. 2015. Brew temperature and its effects on espresso. Accessed: 2019-09-25.

Fox, M., and Long, D. 2002. Pddl+: Modeling continuous time dependent effects. In *Proceedings of the 3rd International NASA Workshop on Planning and Scheduling for Space*, volume 4, 34.

Fox, M., and Long, D. 2006. Modelling mixed discrete-continuous domains for planning. *Journal of Artificial Intelligence Research* 27:235–297.

Gregory, P., and Lindsay, A. 2016. Domain Model Acquisition in Domains with Action Costs. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*.

Jilani, R.; Crampton, A.; Kitchin, D. E.; and Vallati, M. 2014. Automated knowledge engineering tools in planning: state-of-the-art and future challenges.

Lindsay, A.; Read, J.; Ferreira, J. F.; Hayton, T.; Porteous, J.; and Gregory, P. J. 2017. Framer: Planning models from natural language action descriptions. In *Proceedings of the International Conference on Automated Planning and Scheduling (ICAPS)*.

Lindsay, A.; Franco, S.; Reba, R.; and McCluskey, T. L. 2020. Refining process description from execution data in hybrid planning domain models.

McCluskey, T. L., and Vallati, M. 2017. Embedding automated planning within urban traffic management operations. In *Twenty-Seventh International Conference on Automated Planning and Scheduling*.

McCluskey, T. L.; Vaquero, T. S.; and Vallati, M. 2017. Engineering knowledge for automated planning: Towards a notion of quality. In *Proceedings of the Knowledge Capture Conference*, 1–8.

NCA. 2020. How to brew coffee-the nca guide to brewing essentials. Accessed: 2019-09-25.

Roman Corrochano, B. 2017. *Advancing the engineering understanding of coffee extraction*. Ph.D. Dissertation, University of Birmingham.

Scala, E.; Haslum, P.; Thiébaux, S.; et al. 2016. Heuristics for numeric planning via subgoaling.

Segura-Muros, J. Á.; Pérez, R.; and Fernández-Olivares, J. 2018. Learning numerical action models from noisy and partially observable states by means of inductive rule learning techniques. In *Proceedings of the ICAPS Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*.

Suárez-Hernández, A.; Segovia-Aguas, J.; Torras, C.; and Alenyà, G. 2020. Strips action discovery. *arXiv preprint arXiv:2001.11457*.

Vallati, M.; Magazzeni, D.; De Schutter, B.; Chrpa, L.; and McCluskey, T. L. 2016. Efficient macroscopic urban traffic models for reducing congestion: a pddl+ planning approach. In *Thirtieth AAAI Conference on Artificial Intelligence*.

Wu, K.; Yang, Q.; and Jiang, Y. 2007. ARMS: An automatic knowledge engineering tool for learning action models for AI planning. *The Knowledge Engineering Review* 22(2):135–152.

Zhuo, H. H., and Kambhampati, S. 2013. Action-model acquisition from noisy plan traces. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*.

Zhuo, H. H.; Yang, Q.; Hu, D. H.; and Li, L. 2010. Learning complex action models with quantifiers and logical implications. *Artificial Intelligence* 174(18):1540–1569.