Utility Function Specification: A Grammatical Approach

Shadi Haj-Yahia*, Omar Mansour, Tomer Toledo

Faculty of Civil and Environmental Engineering,
Technion – Israel Institute of Technology, Haifa, Israel
Emails: shadi8@campus.technion.ac.il
omar@technion.ac.il
toledo@technion.ac.il

SHORT SUMMARY

This work addresses the challenges in specifying utility functions for discrete choice models (DCMs), which are essential in understanding and forecasting travel behavior. Traditionally, utility functions are manually specified by modelers through a subjective process, based on intuition and experience. While this approach benefits from maintaining an analytical form for easy interpretation, it suffers from inconsistency and potential inaccuracies due to the lack of a systematic framework. To overcome these limitations, this study proposes a method that combines machine learning's automation with the interpretability of analytical forms using grammar. The goal is to develop an automated approach for defining variables, transformations, and interactions in utility specification, while ensuring alignment with domain knowledge. This leads to analytically expressive and interpretable utility functions for DCMs. The proposed framework's potential is demonstrated through a case study.

Keywords: Discrete choice models, grammar, grammatical evolution, utility function specification, domain knowledge

1. INTRODUCTION

Traditional utility function specification in discrete choice models (DCMs) is performed through a subjective trial-and-error process, relying on the modeler's expertise. While this method keeps utility functions in an analytical form for easier parameter interpretation, it lacks consistency and does not systematically integrate domain knowledge. Incorrect specifications can lead to biased parameter estimates and predictions (Torres et al., 2011, Kling, 1989).

To address this, several methods have been proposed for utility specification assistance, like combinatorial optimization (Ortelli et al., 2021), bayesian framework for feature selection (Rodrigues et al., 2020), gradient boosting trees ensemble (Hillel et al., 2019), and simulated annealing for mixed logit models (Paz et al., 2019). While these studies assist in variable selection and transformation, they either do not consider possible high-order interactions, time-consuming, or lack domain knowledge integration.

This work aims to systematically assist modelers in DCM utility function specification by merging machine learning automation with interpretability and domain knowledge, using grammar-based methods. The objective is to develop a framework for automatic definition of variables, transformations, and interactions in utility specifications, while ensuring alignment with domain knowledge. The resulting utility functions maintain analytical clarity for straightforward interpretation.

2. METHODOLOGY

This section introduces the proposed methodology aimed at assisting modelers in the specification of utility functions in DCMs. Central to this methodology is the application of grammatical concepts, derived from computational linguistics, to systematically define utility function specifications. This approach provides the modelers with a structured, systematic framework that eases the specification process.

The section begins by outlining the specification requirements based on domain knowledge dictated by the modeler, followed by the presentation of the proposed grammar and its formulation. The overall framework is presented next, employing the grammatical evolution (GE) algorithm for optimal specification discovery. Next, the genetic operators of GE and the fitness evaluation process are briefly presented.

Domain knowledge and specification requirements

In this work, utility function specification considered requirements are categorized into two categories: grammar-encoded and post-estimation verified.

Grammar-encoded requirements:

1. **Model identification and conventions:** In choice models, only differences in utility matter (Train, 2009). With J alternatives, only $J-1$ alternative-specific constants (ASCs) can be included, normalizing one ASC to zero for identification. The non-zero ASCs indicate relative preference.

   Socio-demographic variables, being consistent across alternatives, enter the utility of only $J-1$ alternatives. To avoid the "dummy variable trap" in categorical variables, $K-1$ categories are included using dummy coding, with one category as the base.

   When including interactions between a categorical socio-demographic variable and an alternative attribute in utility functions, it is crucial to include both the attribute and interactions with $K-1$ categories. This approach captures the attribute's total effect on utility for all, not just a specific category. It prevents misinterpretation (like implying no main effect) and biases in other model parameters, allowing direct comparison of interaction effects to the base category.

2. **Level of interactions:** The choice of variable interaction levels balances capturing complex relationships and keeping the model interpretable. The maximum allowed level of interaction can therefore be predefined to reflect complex effect without losing interpretability.

3. **Adaptability in utility function specification:** Specification adaptability serves different data sets and research goals, depending on the modeler's needs and expertise. It allows specific function specification for each alternative, or a generic specification for various alternatives. This adaptability is key for a wide range of choice scenarios, from individualized choices like mode choice to scenarios with pronounced commonalities, such as route choice.

4. **Variable transformations:** Dictates transformation types to align with empirical and theoretical understanding, like logarithmic for diminishing returns, power or linear.

Post-estimation verified requirements:

1. **Behavioral Constraints and Realism:** The specification should not only fit the data, but also align with economic theory and behavioral understanding. Modeled preferences and decisions in utility functions must reflect realistic mode shares and usage, consistent with theory and reality. For instance, attribute effects like travel time and cost should reduce utility and choice probability.

2. **Parsimony:** Parsimony in model specifications, aiming for simplicity without losing explanatory power, offers several benefits. It reduces overfitting and simplifies interpretation. Often, statistical significance is a key indicator – variables that add complexity without being significantly different from zero may be candidates for omission. However, variables that proxy unobserved effects can still provide useful explanatory value despite modest t-statistics. Model selection, using criteria like AIC or BIC, helps balance fit and complexity, ensuring theoretical consistency. The aim is a balanced, flexible model that accurately represents behaviors without unnecessary complexity.

Proposed grammar

Grammar is formulated in a Backus-Naur Form (BNF) format, defined as a tuple $G = (N, T, R, S)$. Terminal symbols (T) represent actual language elements like numbers or words, while non-terminal symbols (N) act as placeholders for rules, expanding into terminals, non-terminals, or their combinations. Production rules (R) dictate further expansions of non-terminal symbols, defined in the form $A \rightarrow a$, where the left-hand side is a non-terminal $A \in N$, while the right-hand side is a combination of non-terminals and terminals, $a \in (N \cup T)$. The grammar starts from a non-terminal symbol S and recursively applies these rules for expression derivation, ending with only terminal symbols.

The proposed grammar is shown in Figure 1. The starting symbol (S), decomposes into J distinct utility functions for J alternatives. Each utility function (U_j) expands to become either a single sub-expression (E_j) or include an additional sub-expression. Each sub-expression (E_j) expands to become either an interaction between the existing sub-expression and a transformed variable (W_j), or a transformed variable only. The interaction is determined by the operator (O), which can be either a multiplication or division. The transformed variable (W_j) is the variable (V_j) transformed by (F). Transformation (F) can be any allowed transformation specified by the modeler (e.g., linear, logarithmical, exponential, etc.). Variable (V_j) is determined by the set of variables available in the dataset. This design accommodates complex utility specifications and is flexible based on interaction levels and transformations guided by domain knowledge.

$N = \{U_j, E_j, W_j, V_j, O, F\} \quad j \in \{1,..,J\}$
$T = \{+, *, \text{available variables}\}$
$S = \{[<U_1>,\ldots,<U_j>,\ldots,<U_J>]\}$
$P =$

(1) $<S> \longrightarrow <U_1>,\ldots,<U_j>,\ldots,<U_J>$ (0)

(2) $<U_j> \longrightarrow <U_j>+<E_j>$ (0)
$|<E_j>$ (1)

(3) $<E_j> \longrightarrow <E_j><O><W_j>$ (0)
$|<W_j>$ (1)

(4) $<W_j> \longrightarrow <F>(V_j)$ (0)

(5) $<O> \longrightarrow *$ (0)
$|/$ (1)

(6) $<F> \longrightarrow$ log, exp, linear, etc.

(7) $<V_j> \longrightarrow$ set of available variables for alternative j

Figure 1. Proposed grammar

Grammatical evolution framework

Grammatical evolution (GE) combines grammar principles with evolutionary algorithms, offering structured and flexible solutions to complex problems (O'Neill & Ryan, 2001). GE, part of genetic programming, evolves language expressions and programs based on grammatical rules. It efficiently explores vast solution spaces through selection, crossover, and mutation processes. In GE, solutions are variable-length vectors or chromosomes of integer codons (Figure 2).

The proposed GE framework in Figure 3 starts with randomly generated initial population of feasible individuals. Each individual, decoded using grammar, forms J utility expressions for model specifications with non-estimated parameters. Fitness evaluation occurs in three stages: parameter estimation using training data, checking model's alignment with domain knowledge, and assessing performance on the validation set. The process ends if termination criteria (like fitness convergence or generation limit) are met, yielding the best solution. Otherwise, it continues, generating new candidates through genetic operations: selection, crossover, and mutation.

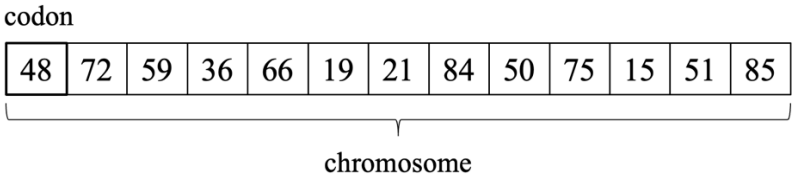

Figure 2. Example of an individual expressed by a chromosome

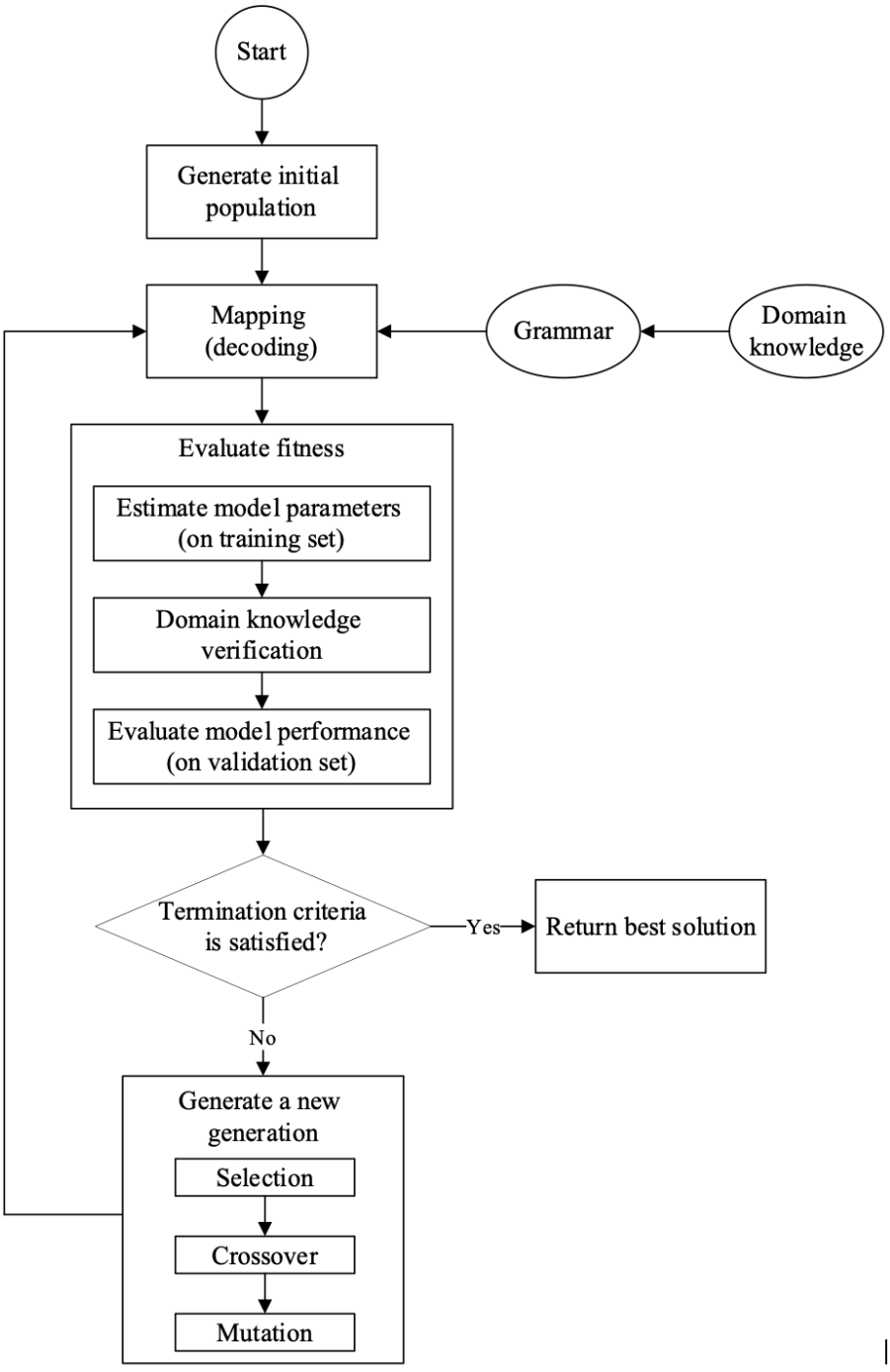

Figure 3. Overall framework

Genetic operators

GE employs three genetic operators – selection, crossover, and mutation – to generate new solutions:

Selection: Tournament selection is used where a subset of the population is randomly chosen, and the fittest individual is selected as a parent.

Crossover: Combines two parent individuals to create offspring. The k-point-block crossover method merges genetic data from two parent chromosomes at designated block boundaries to create offspring. Each block represents a single utility function's chromosome. For instance, with three utility functions, each is a block, and crossover occurs at two points defining these blocks (Figure 4). This approach, using block grammar, prevents the mixing of unrelated blocks from different utility functions, avoiding the issues seen in traditional crossover methods.

Mutation: Uniform mutation is used, where each gene is altered based on a fixed mutation probability. It introduces diversity in the gene value spectrum, preventing early convergence and allowing for a wider exploration of the solution space.

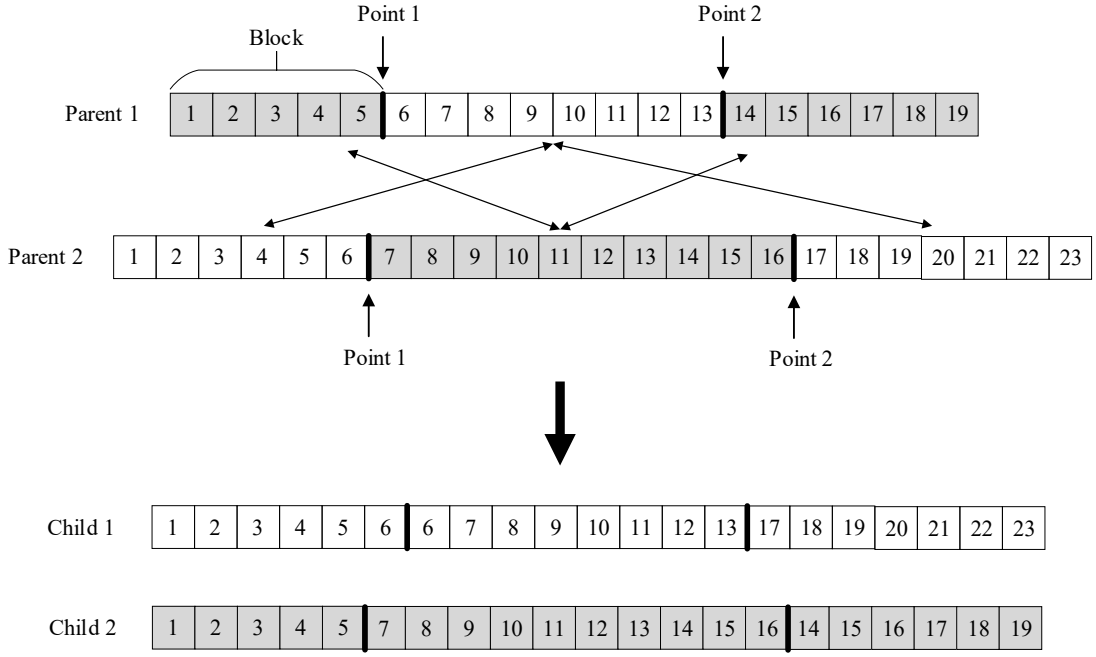

Figure 4. Two-point block crossover operator

Fitness evaluation

The fitness evaluation in the framework involves three steps:

Step 1 – Parameter estimation: The candidate's fitness is initially evaluated by estimating the parameters, using maximum likelihood estimation (MLE) on the training set.

Step 2 – Domain knowledge verification: Following parameters' estimation, the model is checked against predefined shape constraints reflecting prior knowledge, like the expected negative impact of travel time on utility. Violations lead to the worst fitness values, preventing advancement to the next generation. If constraints are violated, fitness is predetermined, and step 3 is skipped.

Step 3 – Model performance evaluation: Fitness is based on the model's performance on validation set to prevent overfitting. Fitness is also penalized by parameter count for parsimony, using the formula:

$$fitness = LL - \alpha \cdot k$$

where LL is log-likelihood, k the parameter count, and α a predefined penalty. The function aligns with AIC and BIC, which penalize log-likelihood relative to parameters. The modified function equates to AIC with $\alpha = 1$ and to BIC (for a dataset size of 3000) with $\alpha = 4$.

Efficiency considerations

In this framework, computational efficiency is achieved in three ways:

- In the first step of parameter estimation, to speed up computation, the variance-covariance of parameter estimates is avoided, as it is not required at this stage. It is calculated only for the final model after convergence.

- The second step of domain knowledge verification is a screening stage, determining which candidates advance to the model evaluation phase. Models failing this verification are not assessed on the validation set, leading to considerable time savings.

- A key benefit of GE is its suitability for parallel processing. While generation production is sequential, the evaluation of candidates within a generation is independent. This allows for the parallel fitness evaluation of individual candidates, significantly reducing time requirements.

3. RESULTS AND DISCUSSION

The proposed framework was applied to a mode choice dataset to examine the ability to discover complex utility specifications under different settings based on grammar and incorporated domain knowledge.

Dataset

The dataset utilized in the case study is the Swissmetro dataset (Bierlaire et al., 2001). It is a stated preference survey conducted in Switzerland in 1998, designed to capture the preferences for transportation modes among a new Swissmetro (SM), car, and train. The variables used from the

dataset consist of level of service variables (i.e., travel time, cost, headway, seats, etc.) and individuals' socio-demographics (i.e., age, gender, income, etc.). Categorical variables were one-hot encoded to dummy variables. Observations with unavailable alternatives, unknown features or outlier values were filtered, resulting in 7,778 samples. The dataset was then divided into training, validation, and testing sets in the ratio of 60:20:20.

Experimental Design

The aim of the case study experiments is to evaluate the versatility and effectiveness of the proposed framework for specifying utility functions for DCMs. The framework was applied for three utility function specification types:

1. Flexible specification: Each alternative has a unique utility specification, providing maximum flexibility.
2. Generic specification with alternative-specific coefficients: A shared structure across alternatives with distinct coefficients for each alternative.
3. Generic specification with shared coefficients: A common structure for all alternatives, and same coefficients across alternatives.

To explore the effect of model simplicity, varying levels of parsimony penalties α were applied on the flexible structure. Penalty values range from 0 to 10, where the model is "free" when $\alpha = 0$ and the number of estimated parameters does not affect its fitness, while $\alpha = 10$ represents the largest applied penalty.

All estimated models, regardless of specification type or parsimony, should align with behavioral realism. The incorporated behavioral constraints in this study dictate negative sensitivities of utilities to travel time and cost. These are hard constraints, and a violation of these constraints leads to the worst possible fitness.

Results

Model generation process

To examine the progression and efficiency of the model generation process, the performance of the best candidate model with flexible specification across the generations is plotted. Figure 5 presents the model performance on both training and validation sets. The performance is measured by the average log-likelihood (AvgLL). The training set curve shows the model's performance as its parameters are estimated, while the validation set curve assesses the model's generalizability and to prevent overfitting.

The performance on the training set is higher than on the validation set, which is expected since the model is optimized for the training data. There is a rapid improvement in both sets in the first 20 generations, indicating the algorithm's efficiency in propagating beneficial specifications. The improvement on validation set converges by the 60^{th} generation. However, the training set does not show continuous improvement. This indicates that in some cases, the model begins to overfit to the training set at the expense of its generalizability. As models are advanced based on validation set performance, training set improvements do not ensure better validation results. Thus, models less fitted to training set are chosen for better generalizability. This demonstrates the proposed process to effectively balance the trade-off between fitting the training set and maintaining the model's ability to generalize to unseen data.

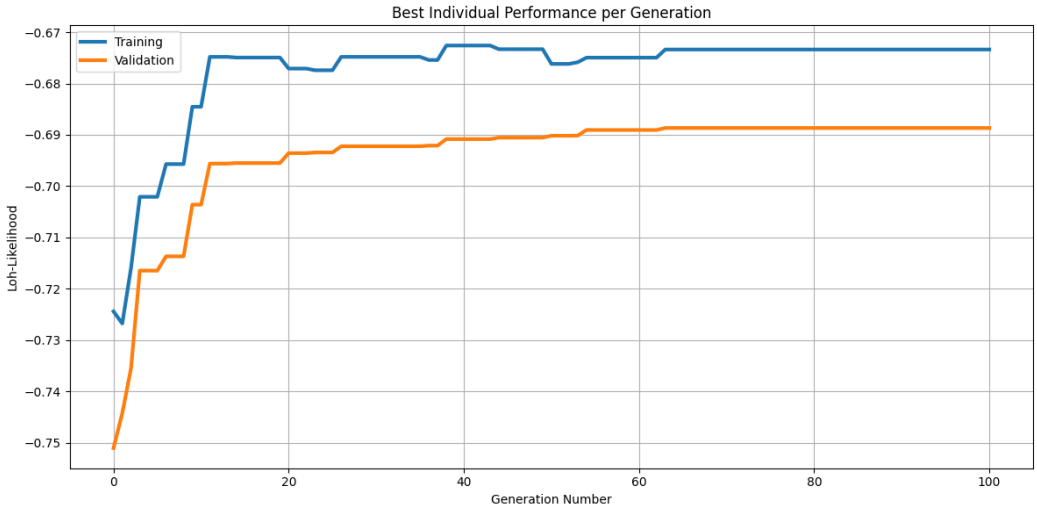

Figure 5. Best candidate performance on training and validation sets along generations.

Specification types

In addition to the flexible specification analyzed above, additional two specification types were examined. Table 1 presents the models' performances and number of estimated parameters. The flexible specification outperforms all models across all datasets, followed by the generic specification and the fully generic. However, the fully generic specification, while not leading in performance, demonstrates simplicity with its parsimonious parameter estimation, as the number of estimated parameters is significantly decreased.

Table 1. Performance (AvgLL) of different specification types

Model structure	Training	Validation	Testing	Estimated parameters
Flexible	-0.674	-0.689	-0.721	39
Generic specification with alternative specific parameters	-0.684	-0.702	-0.734	33
Fully generic	-0.693	-0.711	-0.738	18

Parsimony

Table 2 presents model performance across different α values on three sets: training, validation, and testing. The results demonstrate a clear trend of decrease in the number of estimated parameters as the penalty increases, as expected. A sharp drop occurs in the number of parameters as α increases from 0, which then levels as further increases in α result in additional marginal reductions in parameters. The gradual decline in AvgLL indicates a decrease in model fit with increased parsimony, a common trade-off in model selection.

The results show that there is a significant reduction in the number of parameters when penalties are applied within the range of less than 5. Within this range, the corresponding decrease in AvgLL is relatively limited, suggesting that the performance of the model is insignificantly affected despite the reduction in complexity. This indicates that penalties up to a threshold of 5 contribute to the development of more parsimonious models without substantially compromising their predictive accuracy. However, beyond this threshold, the penalties lead to a more

pronounced decline in model performance. This threshold marks the balance in the tradeoff between having model's simplicity and its performance.

Table 2. AvgLL as a function of α

α	0	1	2	3	4	5	6	7	8	9	10
Training	-0.674	-0.678	-0.681	-0.684	-0.689	-0.689	-0.712	-0.712	-0.712	-0.729	-0.729
Validation	-0.689	-0.690	-0.692	-0.698	-0.702	-0.702	-0.725	-0.725	-0.725	-0.744	-0.744
Testing	-0.721	-0.723	-0.725	-0.734	-0.734	-0.734	-0.755	-0.755	-0.755	-0.773	-0.773
Number of parameters	39	24	22	18	16	16	9	9	9	5	5

To further understand the included variables in the parsimonious models, the percentages of these models in which each variable appears were calculated and presented in Figure 6. The travel time and cost variables of each mode appear in all models, including the most parsimonious model with the fewest estimated parameters. This demonstrates that the model still includes travel time and cost variables of all alternatives even with the highest penalties. These variables were included in each utility function by an interaction to correspond with a single estimated parameter. This emphasizes the importance of these variables as the main factors affecting mode choice. The next prevailing factor is purpose of travel which appears in 9 out the 11 estimated models, dropped at $\alpha = 9$. This indicates the importance of including the purpose of travel in at least one utility function of the model to better represent the travelers' preferences and better predict their choices. The remaining factors appear to be less informative for the model when the objective is focused on parsimony. The male variable and seat configuration did not enter any of the models, including the non-penalized flexible model, apparently for their insignificant contribution to model fit while aiming to maintain generalizability.

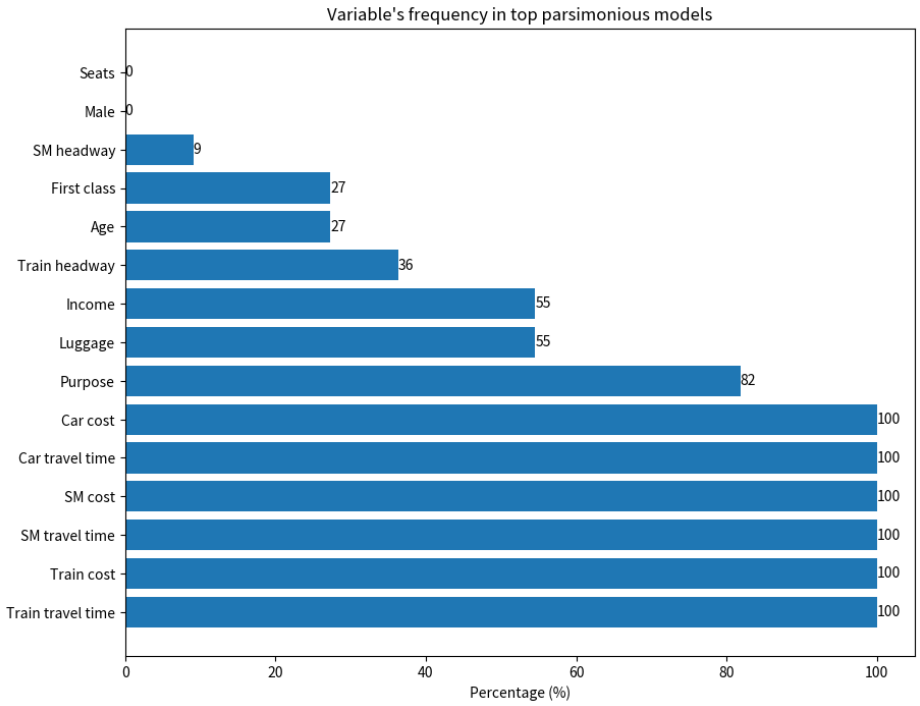

Figure 6. Variable frequency in top parsimonious models

4. CONCLUSIONS

This study presents a new method for specifying utility functions in DCMs using grammar, combining data-driven flexibility with interpretability. It employs domain-specific grammar for varied variable interactions and transformations, balancing model simplicity and performance through log-likelihood penalties for parsimony. The methodology includes ML dataset splitting for validation and leverages parallel processing in the GE algorithm for efficient exploration. The case study highlights the importance of dataset splitting and grammatical design's ability to support different specification types, with investigations into parsimony showing effective maintenance of key model attributes. This work, merging domain knowledge with algorithmic and data-driven approaches, provides a flexible tool for utility specification and future work aims to test its application to other logit structures, like nested and mixed logit, utilizing its adaptable nature.

REFERENCES

Bierlaire, M., Axhausen, K., & Abay, G. (2001). The acceptance of modal innovation: The case of Swissmetro. Swiss Transport Research Conference, CONF.

Hillel, T., Bierlaire, M., Elshafie, M., & Jin, Y. (2019). Weak teachers: Assisted specification of discrete choice models using ensemble learning. HEART 2019, 8th Symposium of the European Association for Research in Transportation, Ml.

Kling, C. L. (1989). The importance of functional form in the estimation of welfare. Western Journal of Agricultural Economics, 168–174.

O'Neill, M., & Ryan, C. (2001). Grammatical evolution. IEEE Transactions on Evolutionary Computation, 5(4), 349–358.

Ortelli, N., Hillel, T., Pereira, F. C., de Lapparent, M., & Bierlaire, M. (2021). Assisted specification of discrete choice models. Journal of Choice Modelling, 39, 100285.

Paz, A., Arteaga, C., & Cobos, C. (2019). Specification of mixed logit models assisted by an optimization framework. Journal of Choice Modelling, 30.

Rodrigues, F., Ortelli, N., Bierlaire, M., & Pereira, F. C. (2020). Bayesian automatic relevance determination for utility function specification in discrete choice models. IEEE Transactions on Intelligent Transportation Systems, 23(4), 3126–3136.

Torres, C., Hanley, N., & Riera, A. (2011). How wrong can you be? Implications of incorrect utility function specification for welfare measurement in choice experiments. Journal of Environmental Economics and Management.

Train, K. E. (2009). Discrete choice methods with simulation. Cambridge university press.