A computational intelligence decision-support environment for architectural and building design

CIDEA

Chatzikonstantinou, Ioannis

DOI
10.1109/CEC.2016.7744282

Publication date
2016

Document Version
Accepted author manuscript

Published in
Proceedings 2016 IEEE Congress on Evolutionary Computation (CEC)

Accepted Author Manuscript. Link to published article (IEEE): http://dx.doi.org/10.1109/CEC.2016.7744282

A Computational Intelligence Decision-Support Environment for Architectural and Building Design: CIDEA

Ioannis Chatzikonstantinou
Faculty of Architecture, Yaşar University, Izmir, Turkey
Department of Building Technology, Faculty of Architecture, TU Delft, Delft, The Netherlands
i.chatzikonstantinou@yasar.edu.tr

***Abstract*— Environmentally friendly and comfortable buildings are a much sought after goal in today's architectural practice. In order to improve energy consumption of buildings without sacrificing indoor comfort, careful consideration of design decisions is needed. Simulation tools provide a solution to one aspect arising from this need, namely the requirement for accurate quantitative results. On the other hand, the complexity of the real-world design problems in question calls for decision support tools that integrate, in addition to simulation, optimization, analysis, and modeling. The aim of the paper is to present ongoing work on the development of such a tool. The focus of the tool is on abstraction of the technical complexity, while maintaining a sufficient level of flexibility. The tool is designed according to an integrated workflow beginning from sampling, data analysis, model creation and testing, up until the final analysis of the optimization results. We present the architecture of the platform, as well as its application in two case studies, one focusing on the design of an office tower, and one on the design of a sustainable facade. Results from qualitative usage cases indicate favorable performance in supporting decision-making.**

Keywords*— *Decision Support, Optimization, Surrogate Modeling, Data Analysis, Architectural Design

I. Introduction

It is well established that the built environment profoundly affects many aspects of our society, not the least of which are related to sustainability and human comfort. This calls for an informed decision making process, that is based on objective figures regarding the various performance aspects of the built environment. Moreover, this information needs to be made available as early as possible in the design decision-making process. Ideally, decision makers should be equipped with data on design performance and potential of design alternatives already from the conceptual design stage. Tools for this purpose exist, but they are either too technical in nature, or their capabilities are not in par with the state-of-the-art. The present research is inspired by the need for a software tool that may facilitate the application of computational decision support methods in the fields of sustainable building design and architecture.

The aim of this research, as such, is to produce a software tool that i. offers state-of-the-art computational decision support methods, focusing mainly on optimization, meta-modeling and cognition, and, ii. abstracts away as much as possible from the technical infrastructure from the end user, the decision maker, allowing for high-level operations. We term this tool, CIDEA.

The paper is structured as follows: In section 2, a brief background is presented, and popular tools are being outlined. In section 3, the proposed tool is being presented in detail, focusing on its software architecture, user interface, functionality and extensibility. Section 4 discusses two applications in the field of sustainable office design that have been carried out in order to evaluate the tools effectiveness. Section 5 provides a brief discussion on results from using the tool, and section 6 concludes the study.

II. Background

Simulation, while used extensively as a tool for supporting decisions in architectural design, is only one part of decision-making and can only provide information on solutions that have already been established in the design process. On the contrary, it is desired in many cases that computational decision support systems offer advance information in the form of alternative solutions and their performance. Going one step further, the full range of optimal and near-optimal solutions should be made available, and even further, decision maker preferences need to be considered.

Most decision support software tools use in the Architecture discipline focus mainly on the derivation and presentation of performance results on a single design solution, defined by the tool user. More advanced tools use techniques that allow the discovery of design alternatives, and performance evaluation thereof. However, these tools are generally either complicated for the end user, or offer limited functionality with respect to design discovery. Among varying reported tools, some are based on the use of what is mentioned as Parametric Analysis, which involves the perturbation of design parameters and subsequent comparison of results. This is the approach taken by the popular OpenStudio software and jePlus [1], among others.

Tools that employ more advanced methods, such as stochastic optimization, are often too specialized, or require advanced technical knowledge. Furthermore, many of the tools are software-specific. As such, collaboration between teams often breaks down in the orchestration of different software

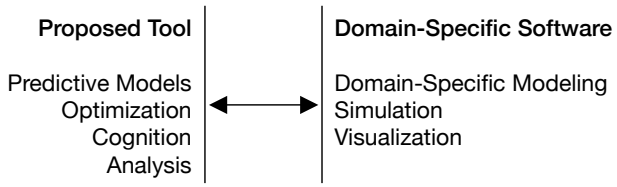

Fig. 1. Delineation of functionality between the proposed tool, CIDEA and domain-specific software packages.

packages. This turns out to be a significant obstacle during the design process, due to the fact that designing takes place in a time-constraint environment.

There have been several tools reported which make use of optimization, among which Paragen [2], GENE_ARCH [3], MOBO [4], ArDOT [5] as well as commercial packages such as TRNSYS and ModeFRONTIER, among others. For a comprehensive review on available tools, the reader is referred to [6].

The proposed tool was built to address mainly two issues, as have been briefly outlined above: The first one is the relative lack of software tools that include advanced, state-of-the-art computational decision support implementations and are addressed to non-expert users in the field. The second one is the apparent fragmentation caused by platform-specific decision-support tools, which, as mentioned above, increases the technical complexity in the design process. CIDEA puts forward a clean delineation of responsibilities, namely: the concentration of problem-agnostic computational decision support functionality, focusing on optimization and cognition, and the exclusion of problem-specific functionality, which should be taking place in specialized design packages. A schematic demonstrating the framework of operation of CIDEA is available in Fig. 1.

III. Software Architecture and Functionality

A. Architecture

The software architecture of CIDEA is outlined in Fig. 2. A fundamental view towards the tool's architecture is that of a series of modules that reside in a common environment. The environment itself takes care of generic tasks, such as module management and interaction. Each of the modules, on the other hand, is a self-contained program, that carries out a specific task or set of tasks, and provides a Graphical User Interface (GUI) for the user to interact with, as well as exposes an Application Programming Interface (API) to allow interaction with other modules. The tasks of each module are performed on one or more background threads, with periodic GUI notifications, thus allowing the user to continue working while a task is running. CIDEA and it's modules follow the Model-View-Controller (MVC) software architecture.

The modules may provide references to other modules, and request data through the module API, optionally passing arbitrary parameters in the form of key-value pairs, along with the request. In essence, this allows the implicit creation of one or more directed graphs within the application environment,

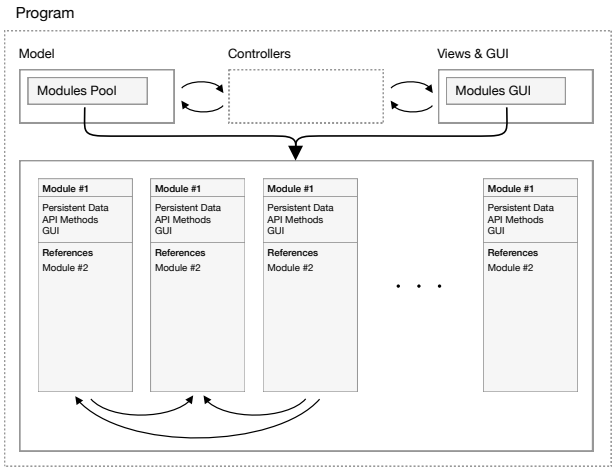

Fig. 2. The Software Architecture of CIDEA, with emphasis on the interlinkable modules that make up the bulk of the functionality of the tool.

through recursive referencing of nodes. The operation, in this case, is such that a node requests processed data from the referenced nodes upstream, which, in turn do the same for their own references, propagating the process until a node without upstream references is found.

While this scheme can easily provide a fully automated process, by relating all modules in a single undirected graph, in practice only small groups of modules are usually referenced. As will be seen later on, this provides a compromise between two extremes: One is that of the user fully defining a priori the

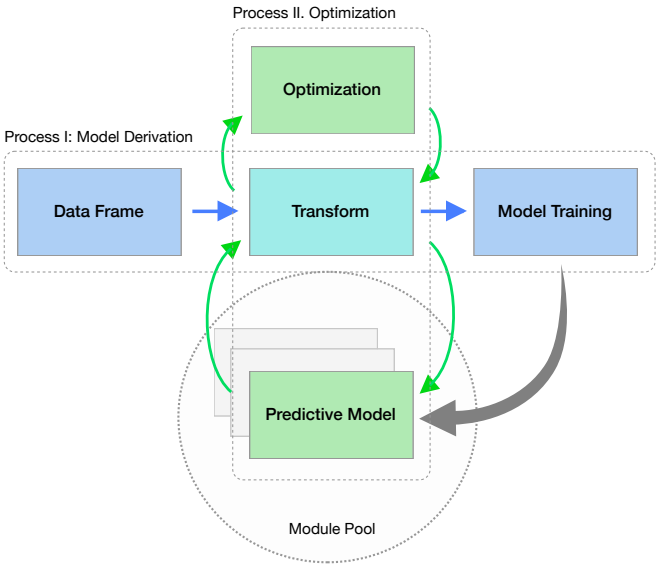

Fig. 3. Scenario involving generation of predictive model and use in optimization. In process I. a data sample is transformed by a user-defined function (Transform) and subsequently used in training (Model Training). The resulting model is placed in the Module Pool for reuse. In process II. an optimization process (Optimization) is communicating decision variable, objective function and constraint values to the recently trained predictive model (Predictive Model), through the same transformation used during training (Transform).

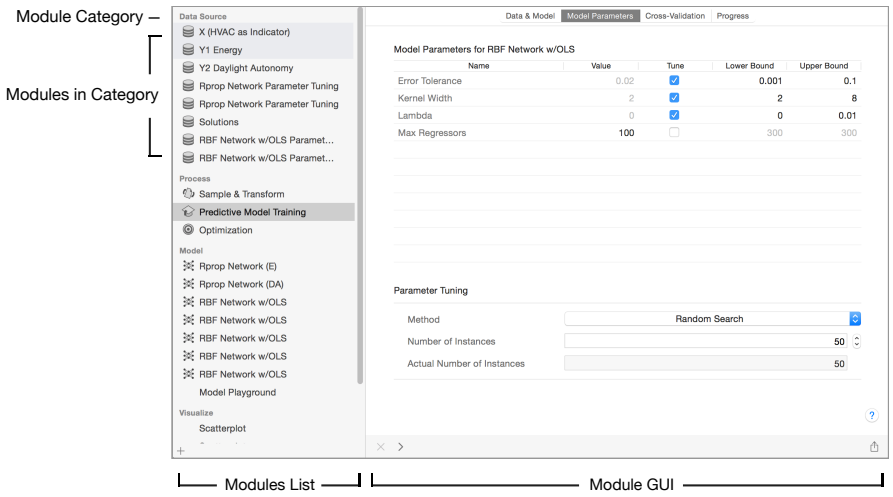

Fig. 4. The Graphical User Interface of the, CIDEA. On the left, a categorized list of active modules. On the right, the module-specific GUI.

process to be followed, and then letting it run and eventually altering it's parameters; the other is that of the user performing each step of the process manually.

CIDEA takes a middle road, in an attempt to eliminate drawbacks of both extremes. Thus, a trivial data pre-processing task can be easily automated through referencing modules together, but at the same time, output of a critical process, such as a predictive model hyperparameter optimization, can be output as a separate module and examined by the user, before being used in another process, such as surrogate-based optimization. An example of such a process, involving derivation of predictive model based on transformed data and subsequent use in optimization, is presented graphically in Fig. 3.

CIDEA is based on the fundamental assumption of separation between algorithm and design problem. As such, the platform has been designed to be completely problem-agnostic in all respects, both in terms of algorithms in use, as well as its architecture and interface. This characteristic generates an interesting question as to how integration with domain- and problem-specific tools should occur. In this case, the decision was to make use of a client-server architecture through which the proposed tool may effectively communicate with software used by decision makers. In our client-server model the client is the tool itself, and the server is a small "adapter" program, that is written as a plugin to domain-specific software. Its task is to translate the standardized communication between CIDEA and the domain-specific software.

During an experiment of an optimization task, the adapters would be first set-up in one or more machines, running the domain-specific software. Subsequently, CIDEA would connect to the host and query required information. It should be noted that this scheme gives itself very easily to parallel processing, in contrast with configuration or file-based schemes, which require extensive preparation for each application, and demonstrate synchronization issues. The connection protocol is HTTP, and the data exchange scheme is JSON. So far, adapter server programs have been developed for the Grasshopper™ parametric design program, associated to the Rhino3d CAD software, and the Dynamo parametric design software, associated to the Autodesk Revit™ CAD program. A diagram of the inter-program communication is available in Fig. 5.

B. *Graphical User Interface (GUI)*

CIDEA implements a GUI for interacting with the various components, and visually setting up their references. The GUI consists of a single window, with a categorical list of modules that are included in a document, on the left of the window, and a module-specific detail view on the right. The latter is responsible for exposing configuration options, and visualizing the progress of ongoing tasks.

Referencing of modules is performed by a single drag-and drop operation: the user first selects the referencing module, so that it's configuration view shows up, and then drags the referenced module from the list on the left, to a corresponding placeholder on the module configuration screen. In this way, the act of module referencing is performed in an intuitive way, while reducing program complexity, as there are no individual module lists to manage for every reference. A number of snapshots of the GUI of CIDEA are available in Fig. 4.

Lastly, it should be mentioned that the proposed tool does away completely with the management of any sort of configuration files, which is a common shortfall of many of the existing tools.

C. *Functionality*

In this section, we briefly touch on the algorithms that are available as modules in CIDEA, and discuss the rationale behind their selection and implementation.

a) Data Processing and Generation

CIDEA offers algorithms for generating pseudorandom and quasi-random, low discrepancy, multi-dimensional sequences, as well as sampling from existing data.

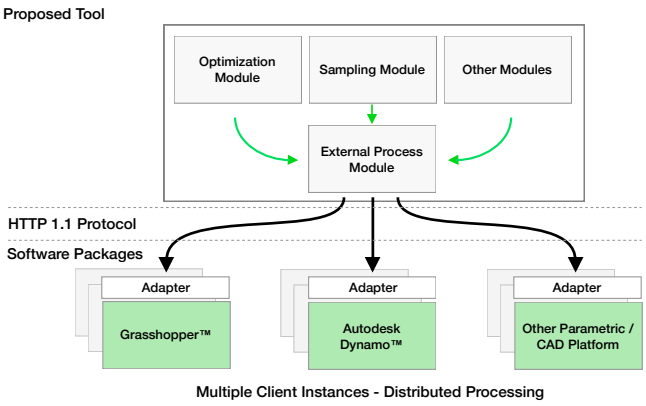

Fig. 5. Communication diagram between the prroposed tool and CAD and Parametric Design software packages. Communication happens in a Distributed manner, using the HTTP 1.1 protocol The External Process module can be referenced by other modules, and as such act as a delegate for processes depending on external programs.

Regarding basic pseudorandom generation, the tool supports generating numbers in user defined ranges, according to uniform and normal distributions (using the Box-Muller transform). In addition to pseudorandom sequences, two types of low-discrepancy sequences are supported. Low-discrepancy sequences, also known as "quasi- random" sequences, are numbers that are better equidistributed in a given volume than pseudo-random numbers [7], [8]. Low discrepancy sequences have been reported to offer advantages when used for meta-model and surrogate training [9], [10]. The sequences supported are Sobol sequence [8] and Halton sequence [11]. Finally, one may sample using random walks with uniform random starting points, user defined step size and optional restarts.

In addition, algorithms for sampling and partitioning datasets are present, to facilitate preprocessing and testing. Finally, corruption of existing dataset values according to a probability, is available. This is especially useful for generating data for training auto-associative models via methods similar to those used for Denoising Auto-encoders, such as those introduced in [12]. As will be seen later in the article, such models are useful to capture latent distributions in datasets of interest, and as such useful in building constraint-satisfaction models that impose constraints in preference vectors.

Outputs from all sampling methods are made available as datasets, to be used subsequently on other tasks, such as sampling of objective function space via simulation (e.g. for analysis or surrogate model training).

b) Predictive Modeling

CIDEA offers several algorithms to build predictive models, as well as tools for cross-validating models and parameter tuning. Currently, five training algorithms are implemented; details of the algorithms are not given in this study, for reasons of compactness, however the interested reader is invited to find detailed descriptions in the referenced studies. The implemented algorithms are as follows:

- Multilayer Perceptrons (MLP), trained using Back-Propagation [13], [14]
- MLPs trained using Resilient-Back-Prop [15]
- Radial Basis Function (RBF) Networks trained using the Orthogonal Least Squares (OLS) Algorithm [16]–[18]
- RBF Networks trained using the OLS-PRESS Algorithm [19]
- Support Vector Machine-based Regression, using the Sequential Minimal Optimization Algorithm [20]–[22]

For reasons of space efficiency we do not provide a description of each algorithm here; the interested reader is referred to the referred publications, which contain detailed treatments of each case. Among many different predictive algorithms, the above mentioned ones have been chosen because of their reported popularity in surrogate modelling [23]. All algorithms as they have been implemented, are capable of multi-output predictions. The algorithms are part of the YCML machine-learning framework[1], and make use of the fast matrix math implementations found in the BLAS and LAPACK libraries for improving the efficiency of the underlying matrix computations.

In training the algorithms, there are options for evaluating their performance using cross-validation. Cross-validation (CV) has been chosen over estimates of predictive performance despite it's increased computational cost, as it is a universal approach to providing less biased estimates of a predictive models performance. Two types of CV are available: k-Fold and Monte Carlo. In k-Fold CV, the training sample is partitioned in k folds; we produce k different models, trained on k-1 folds, and test them on the remaining fold each time. The prediction error is the mean of the errors on each individual fold. In Monte Carlo CV, we perform n different models, by splitting the training sample into two parts by a factor p. The model error is the mean of each individual trained model's error.

Finally, CIDEA offers several methods for optimizing the hyper-parameters of the predictive model. The user may choose which parameters are to be optimized, and the respective parameter domains. We implement several types of parameter search and optimization: Grid-based search, pseudorandom search, quasi-random search using two sequences, Sobol [8] and Halton [11]. The tool identifies the best performing model, as well as outputs a dataset with the configurations and corresponding performances of each instance that has been tested. Hyper-parameter search makes use of model performance either on the training set, or using the user-defined CV method, according to the results of which the model is ultimately chosen.

c) Optimization and Analysis

CIDEA implements optimization algorithms with a focus on multi-objectivity and stochastic optimization. Two

[1] The YCML machine learning framework is an Open Source (GPLv3) software library that is available online at the following address: https://github.com/yconst/YCML . The framework is currently under continued development, and will be subject to publication in the near future.

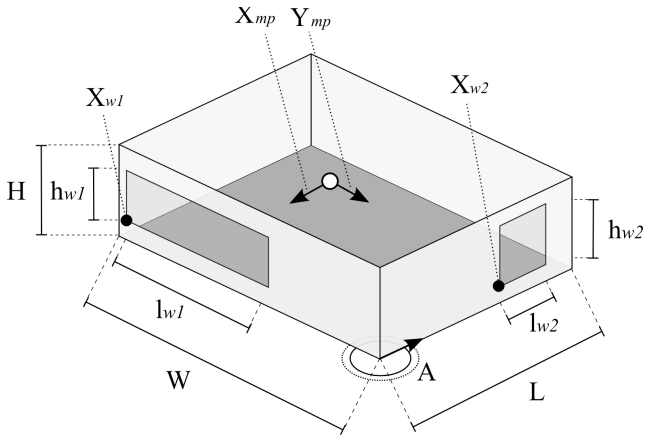

Fig. 6. The office space considered in the first application. It is a space of variable dimensions, and variable orientation to the North. It consists of two windows of variable dimensions, on each of two adjacent sides. Within the space, a single point with coordinate and orientation is used to measure illuminance and glare.

algorithms are currently implemented, namely the well-known NSGA-II [24], and the HYPE algorithm [25]. In addition to the optimization algorithms themselves, there are tools that aid in evaluating the algorithm performance. Currently, estimation of Hypervolume, using a stochastic sampling algorithm, as described in [25], as well as calculation of the Inverted Generational Distance Metric [26] is implemented.

The implementation of optimization algorithms in CIDEA follows a modular approach. To formulate and solve an optimization problem, one selects an algorithm and references one or more problem modules, which in turn define one or more objective functions or constraints. As problem module can be considered any module that accepts numerical data as an input, and provides numerical data as output. As such, it is possible to reference plain functions, predictive models, as well as a mixture of the above.

It is worth mentioning that in the case of referencing a predictive model as an objective function, we are practically turning the predictive model into a surrogate model. As such, the use of surrogate modeling in optimization is straightforward, and does not introduce any new machinery or user interaction.

D. Extensibility

CIDEA is build with extensibility in mind. An API is exposed which allows the creation of new modules, with their own logic, framework and library references, and user interface elements. Through the API, the functionality of each module is exposed to others, so that they may be combined together by referencing. The most essential API functions serve goals as outlined below:

- Being able to notify of what types of data the module is able to process, or output,
- Being able to accept arbitrary data of the correct type, and return the result,
- Being able to return persistent data (if any) of the requested type,
- Being able to report it's referenced modules.

The above functionality is essential for a module to be able to be incorporated in an automated or semi-automated process defined by the user.

IV. Applications

Two applications in the field of building performance analysis and optimization are reported. The purpose here is to illustrate the tool's functionality in a design environment, through demonstrating advanced use-cases on research-based design scenarios.

A. Surrogate Modeling for Visual Comfort Approximation

The first example aims to outline an application of CIDEA in the development of a surrogate model for daylighting and glare evaluation in an office space, and it's subsequent use in multi-objective optimization, in order to derive optimal solutions with respect to those objectives. This application is related to a previous publication on the same topic [27]. Specifically, the use of a problem definition interface, together with the surrogate models, for easily defining the optimization problem at hand, is outlined.

The problem at hand concerns the development of a predictive model that can approximate the adequacy and quality of daylight within an office space of variable dimensions and window configurations. The inputs to the predictive model are the dimensions, orientation and window configurations of the office, as well as the coordinates, within the office space, and direction of a sampling point corresponding to viewer position, and are available in Table I.

TABLE I. Independent variables for application A.

Name	Unit	Range
Room Width	m	[4, 8]
Room Length	m	[4, 8]
Orientation	rad	[0, 2π]
Window 1 Width	%	[0, 100]
Window 1 Height	m	[1.5, 2.5]
Window 1 Position	%	[0, 100]
Window 2 Width	%	[0, 100]
Window 2 Height	m	[1.5, 2.5]
Window 2 Position	%	[1.5, 2.5]
Sampling Point X	-	[0, 1]
Sampling Point Y	-	[0, 1]

An instance of the space in question is available in Fig. 6. The outputs are two: The Daylight Autonomy (DA) value [28], [29] for the particular point, and the Daylight Glare Probability (DGP) value [30], [31], for the point and direction in question. DA and DGP are popular metrics that can be used to define the visual comfort of a single point in an interior space. DA and UDI are defined as follows:

$$DA,UDI = \frac{1}{h}\sum_{i=1}^{h} b(I(p_i),B)$$

Where I(x) is a function giving the illuminance at point p, and b(x) a function that returns one if x is within bounds B, and

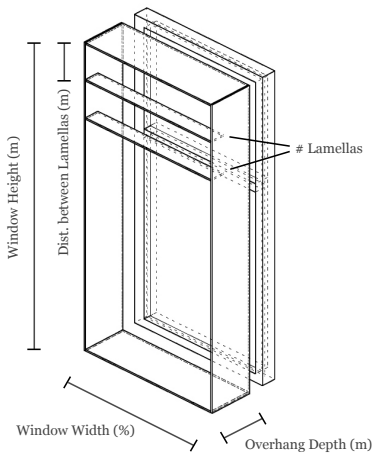

Fig. 7. The openings on the façade of the second application, together with the shading elements and corresponding decision variables.

0 otherwise. The main difference between DA and UDI is how B is defined. The reader is referred to the respective publications above for more information and derivations of the metrics in question. Calculation of DA was performed using the Radiance program [32], and of DGP using the evalglare program [33].

Subsequently, a multi-objective optimization scenario that includes improvement of daylight conditions, namely maximization of Daylight Autonomy and minimization of Glare is considered, and approached using two types of Evolutionary Algorithms: NSGA-II and HYPE. The objective function formulation is as follows:

$$\min(-DA, DGP)$$

Where DA and DGP correspond to the above defined Daylight Autonomy and Daylight Glare Probability metrics, and their values are obtained by means of the predictive models discussed.

B. *Cognitive Modeling for Sustainability in the context of Multi-Criterion Decision Making*

The second application study that we wish to present aims at highlighting how CIDEA can easily be used in the context of Multi-Criterion Decision Making (MCDM), and preference-based optimal design. The study concerns the identification of suitable designs for an office façade having shading elements, in order to maximize visual comfort and minimize energy consumption. These two quantities form objective functions, and are calculated by means of simulation. We make use of the Radiance [32] and EnergyPlus [34] programs. The problem formulation is as follows:

$$\min(UDI, -E)$$

There are four decision variables, namely Window Width (WW), Window Height (WH), Shader Count (SC) and Overhand Depth (OD). The correspondence of the decision variables to the façade design is outlined in Fig. 7.

It is well understood that the façade has a complex role in regulating heat exchange with the outside; through conductive, convective and radiative heat exchange. In this respect, the placement and size of exterior openings, as well as shading devices, is of great significance.

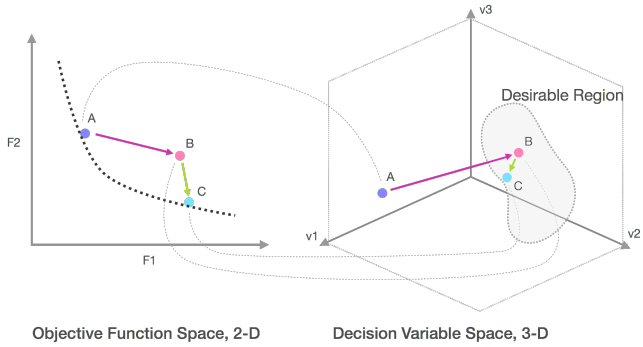

Fig. 8. Outline of the cognitive framework used in Application 2. One the left, Objective Function space of a 2-objective problem. On the right, decision variable space of the same problem, supposing 3 DVs for visual clarity. Solution A is a non-dominated solution with undesirable features. Solution B occurs through manipulation of features by the decision maker. It satisfies designer preferences but is not non-dominated. Solution C is a result of the cognitive system reaction to solution B, and is both preferable and near-optimal, to the degree allowed by the problem definition .

At the same time, these factors play an important role in the composition of the buildings image, as they belong to it's façade. As such, second-order criteria with respect to concrete object properties are at play.

We use CIDEA to train a cognitive auto-associative model, in accordance to the method proposed in [35], [36], to enable preference-based decision support for optimal façade configurations. The reader is referred to the above-mentioned references for an elaborate treatment of the said method. In a nutshell, we wish to produce a predictive, auto-associative model that accepts a preference vector as an input, comprising of values in the decision variable space, and outputs an adjusted vector, again in decision variable space, that corresponds to a near-optimal or optimal solution, or best-tradeoff in the Pareto sense. It is stressed here that the arbitrary preference vector introduced as an input to the model is highly unlikely to correspond to a near-optimal solution. The model is trained considering data identified from a Pareto front obtained using a stochastic optimization algorithm, in our case NSGA-II. A diagram of the action of the cognitive model is available in Fig. 8.

V. Discussion

Regarding the first application as described previously, it was first deemed necessary to identify a suitable set of hyper-parameters for our predictive models. This procedure was performed by comparing two different models: A Backpropagation-trained Feed-Forward Network, and a RBF Network. The Coefficient of Determination (R^2) of the best-performing models on the DA dataset is available in Table II. For each model, two sets of models were compared, according to different parameter search strategies: Grid-based search and Random search. For each case, a total of 50 variants were comapred. The results show a clear advantage of RBFN in prediction performance. Grid search was able to identify better performing FFN models.

TABLE II. R SQUARED OF DIFFERENT PREDICTIVE MODELS ON THE DA DATASET OF APPLICATION A. , USING DIFFERENT HYPER-PARAMETER SEARCH METHODS

Model / Search Method	Parameters	R^2
FFN / Grid	# Iterations, # Samples / Iteration	0.8097
FFN / Random		0.7428
RBF / Grid	Error Threshold, Kernel Bandwidth (β)	0.9164
RBF / Random		0.9294

Subsequently, the best model has been selected to model the objective functions for multi-objective optimization. An optimization run was performed using the NSGA-II algorithm. Finally, a sample of the solutions on the Pareto front was evaluated using the simulation model, instead of the metamodel.

As to the use of CIDEA, this problem was implementable in a straightforward manner; firstly, through the use of the "Sampling" module, a dataset of DV values was generated. OF values for each point were sampled using the "External Process" module. Subsequently, the generated datasets were used with the built-in parameter search option of the "Predictive Model Training" module. Resulting surrogate models were used with the "Optimization" to derive the non-dominated solutions. Simulation times for Radiance are 5.6 minutes, and for evalglare 10 seconds, on average. The produced surrogate model evaluates in the range of a few tens of milliseconds (approximate).

With respect to the second application, the HYPE algorithm was initially used in order to establish a Pareto front for the problem at hand. Function evaluation was performed using simulation interface DIVA for Rhino [37]. Data exchange between CIDEA and the Rhino 3D CAD program was established using a software adapter, as discussed in section 2. Subsequently, an RBF network was trained in an auto-associative manner, using a linear search to identify optimal value for the kernel bandwidth of the model. In this case, an RBF network that was trained with multi-output Orthogonal Least Squares was used. This is a similar approach as has been described by Çiftçioğlu and Bittermann [36]. For evaluating model performance, we did not use a least squares-based performance metric, as it is not suitable for this task; rather, we made use of the IGD metric [26], and make a comparison between IGD values output from a sample of uniform random values in the range of the problem decision variables, and the response of the trained model, when presented with the randomized sample as an input. Results for two different RBF networks are available in Table IV. In addition, Fig. 9 provides the reaction obtained by the trained algorithm for a number of preference vector inputs. This application study is part of an ongoing project that is subject to publication.

TABLE III. PERFORMANCE OF TWO RBF NETWORKS WITH DIFFERING KERNEL BANDWODTHS, ON THE COGNITIVE DESIGN PROBLEM OF APPLICATION B.

Bandwidth	Regressor Count	GD (random)	GD (response)
0.8	21	0.103	$2.66 * 10^{-3}$
3.0	9	0.092	$5.75 * 10^{-3}$

It should be mentioned that in both cases, CIDEA allowed carrying out tasks such as processing the input data, training and optimizing model performance, performing the optimization and analyzing and presenting the results, in a quick and efficient way. As such, it is the author's belief that CIDEA can provide a benefit in decision making during the design process.

VI. CONCLUSION

In this paper, a decision-support tool, named CIDEA, with application in design for the built environment was presented. The tool adopts a flexible approach of organizing together different modules, each with a specialized functionality, to generate a workflow by mixing automation and interaction. The following are presented as contributions of the presented work:

1. The tool includes instruments that help process data, create predictive models, make inference, perform optimization, and perform analysis on results, under one roof, allowing for immediate access and as such increased usability.
2. The systems functionality is based on connecting different modules together into one or more Directed Graphs (an action termed *referencing* in the current work). Such a mode of operation allows for complex orchestration and for performing more complex tasks than the individual modules can perform. As such, non-experts may realize research-intensive applications using the proposed tool, in an intuitive way.
3. The proposed tool relies on an Open-Source and high performance Machine Learning and Optimization library, and as such the algorithms in use can be easily controlled and validated.

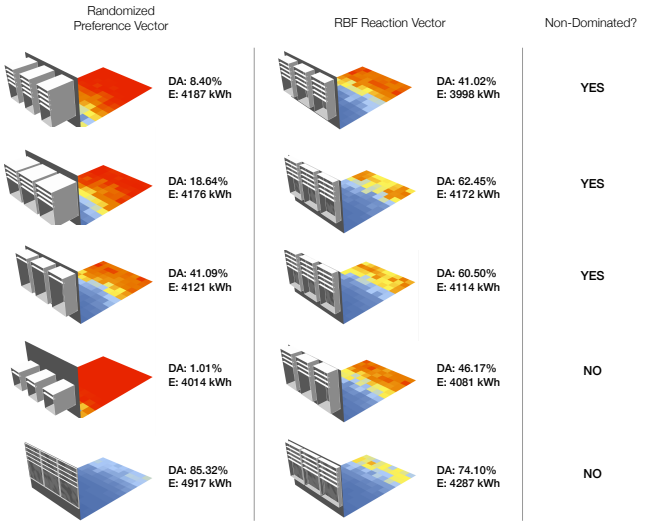

Fig. 9. Several solutions and corresponding performance according to Daylight Autonomy (DA) and Total Energy consumption, resulting from the response of the cognitive RBF model, for different preference vectors. The far right column indicates whether the cognitive response is non-dominated with respect to the input.

On the other hand, it is acknowledged as a limitation that a purely visual environment, such as in the presented work, may not offer the required flexibility for orchestrating and automating large scale, complex learning or optimization workflows. However, for such cases there exist specialized tools that are much more appropriate.

We have evaluated the applicability of CIDEA in supporting evaluation and optimization tasks in two different design scenarios, involving sustainability and visual comfort-related goals. It was deemed that, the proposed tool resulted in reduced time for performing tasks related with design decision support. It is thus the author's belief that the proposed tool may introduce a better adoption of state of the art computational decision support methods in the field of architecture.

ACKNOWLEDGMENT

I wish to thank my colleagues Dr. Onur Dursun and Berk Ekici for providing valuable feedback throughout this work.

REFERENCES

[1] Y. Zhang and I. Korolija, "Performing complex parametric simulations with jEPlus," *SET2010-9th Int. Conf. Sustain. Energ. Tech.,* 2010.

[2] M. Turrin, P. von Buelow, and R. Stouffs, "Design explorations of performance driven geometry in architectural design using parametric modeling and genetic algorithms," *Adv. Eng. Informatics*, vol. 25, no. 4, pp. 656–675, Oct. 2011.

[3] L. G. Caldas and L. K. Norford, "A design optimization tool based on a genetic algorithm," *Autom. Constr.*, vol. 11, no. 2, pp. 173–184, Feb. 2002.

[4] M. Palonen, M. Hamdy, and A. Hasan, "MOBO A New Software for Multi-Objective Building Performance Optimization," *13th Conf. Int. Build. Perform. Simul. Assoc.*, pp. 2567–2574, 2013.

[5] M. M. Mourshed, D. Kelliher, and M. Keane, "ArDOT: A Tool to Optimise Environmental Design of Buildings," *IBPSA 8º Int. Conf.*, vol. 50, no. 43, pp. 919–926, 2003.

[6] A.-T. Nguyen, S. Reiter, and P. Rigo, "A review on simulation-based optimization methods applied to building performance analysis," *Appl. Energy*, vol. 113, pp. 1043–1058, Jan. 2014.

[7] I. L. Dalal, D. Stefan, and J. Harwayne-Gidansky, "Low discrepancy sequences for monte carlo simulations on reconfigurable platforms," *Proc. Int. Conf. Appl. Syst. Archit. Process.*, pp. 108–113, 2008.

[8] I. M. Sobol', "On the distribution of points in a cube and the approximate evaluation of integrals," *USSR Comput. Math. Math. Phys.*, vol. 7, pp. 86–112, 1967.

[9] B. Iooss, L. Boussouf, V. Feuillard, and A. Marrel, "Numerical studies of the metamodel fitting and validation processes," *Int. J. Adv. Syst. Meas.*, vol. 3, pp. 11–21, 2010.

[10] J. Zhang, "Improving the Accuracy of Surrogate Models Using Inverse Transform Sampling," in *53rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Conference*, 2012.

[11] J. H. Halton, "Algorithm 247: Radical-inverse quasi-random point sequence," *Commun. ACM*, vol. 7, no. June, pp. 701–702, 1964.

[12] P. Vincent, H. Larochelle, Y. Bengio, and P. Manzagol, "Extracting and Composing Robust Features with Denoising Autoencoders," pp. 1–16, 2008.

[13] D. Rumelhart, G. Hinton, and R. Williams, "Learning Internal Representations by Error Propagation," in *Parallel distributed processing: explorations in the microstructure of cognition, Vol. 1*, Cambridge, MA, USA: MIT Press, 1985, pp. 318–362.

[14] G. Hinton and G. Hinton, "A Practical Guide to Training Restricted Boltzmann Machines" 2010.

[15] M. Riedmiller and H. Braun, "A direct adaptive method for faster backpropagation learning: theRPROP algorithm," *IEEE Int. Conf. Neural Networks*, 1993.

[16] S. Chen, C. N. Cowan, and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks.," *IEEE Trans. Neural Netw.*, vol. 2, no. 2, pp. 302–9, Jan. 1991.

[17] S. Chen, P. Grant, and C. Cowan, "Orthogonal least-squares algorithm for training multioutput radial basis function networks," *Radar Signal Process. …*, vol. 139, no. December, 1992.

[18] S. Chen, E. Chng, and K. Alkadhimi, "Regularized orthogonal least squares algorithm for constructing radial basis function networks," *Int. J. Control*, no. 773565843, 1996.

[19] X. Hong, P. Sharkey, and K. Warwick, "Automatic nonlinear predictive model-construction algorithm using forward regression and the PRESS statistic," *IEEE Proc. - Control Theory Appl.*, vol. 150, no. 3, pp. 245–254, May 2003.

[20] J. C. Platt, "Fast Training of Support Vector Machines Using Sequential Minimal Optimization," *Adv. kernel methods*, pp. 185 – 208, 1998.

[21] G. W. Flake and S. Lawrence, "Efficient SVM regression training with SMO," *Mach. Learn.*, vol. 46, pp. 271–290, 2002.

[22] "CS 229, Autumn 2009: The Simplified SMO Algorithm," *Stanford Lecture Notes*. 2009.

[23] Y. Jin, "A comprehensive survey of fitness approximation in evolutionary computation," *Soft Comput.*, vol. 9, no. 1, pp. 3–12, Oct. 2003.

[24] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Trans. Evol. Comput.*, vol. 6, no. 2, pp. 182–197, Apr. 2002.

[25] J. Bader and E. Zitzler, "HypE: an algorithm for fast hypervolume-based many-objective optimization.," *Evol. Comput.*, vol. 19, no. 1, pp. 45–76, 2011.

[26] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, and W. Liu, "Multiobjective optimization Test Instances for the CEC 2009 Special Session and Competition," *Report*, pp. 1–30, 2009.

[27] I. Chatzikonstantinou and S. Sariyildiz, "Approximation of simulation-derived visual comfort indicators in office spaces: a comparative study in machine learning," *Archit. Sci. Rev.*, no. August 2015, pp. 1–16, Aug. 2015.

[28] C. F. Reinhart and O. Walkenhorst, "Validation of dynamic RADIANCE-based daylight simulations for a test office with external blinds," *Energy Build.*, vol. 33, no. 7, pp. 683–697, Sep. 2001.

[29] C. F. Reinhart, J. Mardaljevic, and Z. Rogers, "Dynamic Daylight Performance Metrics for Sustainable Building Design," *Leukos*, vol. 3, no. 1, pp. 7–31, 2006.

[30] J. Wienold and J. Christoffersen, "Towards a new daylight glare rating," *Lux Eur. Berlin*, pp. 1–8, 2005.

[31] S. Kleindienst and M. Andersen, "The Adaptation of Daylight Glare Probability to Dynamic Metrics in a Computational Setting," *Proc. Lux Europa 2009 – 11th Europ. Light. Conf.*, pp. 3–10, 2009.

[32] G. Ward, "The RADIANCE lighting simulation and rendering system," in *21st annual conference on Computer Graphics and Interactive Techniques*, 1994.

[33] J. Wienold, "DYNAMIC DAYLIGHT GLARE EVALUATION," *Proc. Build. Simul.*, pp. 944–951, 2009.

[34] B. D. B. Crawley and L. K. Lawrie, "EnergyPlus : Energy Simulation Program," *Ashrae*, vol. 42, no. 4, pp. 49–56, 2000.

[35] Ö. Ciftcioglu and M. S. Bittermann, "Generic Cognitive Computing for Cognition," in *Proceedings of the 2015 IEEE CEC*, 2015, pp. 574–581.

[36] Ö. Ciftcioglu and M. S. Bittermann, "Architectural Design by Cognitive Computing," in *Proceedings of the 2015 IEEE CEC*, 2015, pp. 2295–2302.

[37] Jakubiec JA, Reinhart CF. DIVA 2.0: Integrating Daylight and Thermal Simulations using Rhinoceros 3D, Daysim and Energyplus. Proc. Build. Simul. 2011 12th Conf. Int. Build. Perform. Simul. Assoc., Sydney: 2011, p. 2202–9..