EMERGENT DESIGNER[1]

An Integrated Research and Design Support Tool Based on Models of Complex Systems

RAFAL KICINGER, TOMASZ ARCISZEWSKI
CEIE Department, George Mason University, Fairfax, VA 22030 USA

and

KENNETH DE JONG
CS Department, George Mason University, Fairfax, VA 22030 USA

Abstract. The paper introduces an integrated research and design support tool, called *Emergent Designer*, developed at George Mason University. It is a tool that implements models of various complex systems, including cellular automata and evolutionary algorithms, to represent engineering systems and design processes. The system is intended for conducting design experiments in the area of structural design and for the analysis of their results. It implements state-of-the-art representations supporting generation of novel design concepts and efficient mechanisms for their subsequent optimization at the topology and sizing level. It also implements advanced methods, models, and tools from statistics and from the linear as well as nonlinear time series analysis to conduct the analysis of the design processes. Thus, it is a versatile tool that can be used both as a state-of-the-art design support tool and as an advanced research tool equipped with the methods and tools for the analysis of the design processes and of the obtained experimental results.

---

[1] *Citation:*
Kicinger, R., Arciszewski, T., and De Jong, K. A. "Emergent Designer: Generative design in structural engineering." *Proceedings of the Workshop on the Implementation Issues in Generative Design Systems at the First International Conference on Design Computing and Cognition (DCC'04), Massachusetts Institute of technology, Cambridge, MA, July 17-21, 2004*, L. G. Caldas and J. P. Duarte, eds., MIT, Cambridge, MA, 93-112.

1. Introduction

With the emergence of Information Technology, new design methods are being developed which are based on various computational models of design processes. However, up until very recently, computers in structural design were used mostly and merely for various analytical design activities conducted in the final part of the engineering design process, namely in the detailed design stage (Arciszewski and De Jong 2001). Today, we are finally witnessing the emergence of new design methods applicable both in the conceptual and detailed design stages. In order to fully benefit from this progress, these emerging design methods require new computer tools.

Another motivation comes from the fact that there is a growing trend to apply evolutionary design methods not only to strictly optimization tasks but to use them in finding creative/novel design concepts. Representations of engineering systems are one of the key issues to achieve this goal. When the focus is on finding an optimal design, the attention is usually restricted to a particular concept or at most several concepts of existing designs. In this case, design representations usually take a form of *parameterizations* of an engineering system, or its parts. Traditional representations frequently used in engineering optimizations problems, like binary representations, integer representations, real-valued representations can be included in this category.

Creative evolutionary design requires, however, more general and usually more complex representations. Representations that have been used in creative design are diverse but nevertheless share some similarities. Typically, phenotype representations are quite general and thus capable of representing large numbers of alternative shapes, forms, or morphologies (Bentley 1999). They range from direct representations, as in voxel-based representations (Baron et al. 1997) or array-based representations (Kane and Schoenauer 1995), to highly *indirect* representations, i.e. representations that do not encode solutions but rather rules on how to build these solutions. The most popular examples of indirect representations are grammars (Roston 1994), trees (Funes and Pollack 1999), shape grammars (Stiny 1980), cellular automata (Frazer 1995), L-systems (O'Reilly et al. 2000), and embryogenies (Bentley and Kumar 1999).

In this paper we introduce Emergent Designer, a computer tool developed at George Mason University, that addresses both important issues mentioned above. First, it implements a new design method called Emergent Engineering Design (Kicinger 2004) which is applicable both in the conceptual and detailed design stages. Second, it emphasizes both important objectives of the structural design process, i.e. development of novel/creative designs and their subsequent numerical optimization. The development of novel/creative designs is supported by indirect, or generative, representations based on various models of complex systems.

A general overview of the system describing its overall architecture and the flow of information is presented in the following section. Section 3 offers detailed description of its all major components. The actual implementation of Emergent Designer is presented in section 4. Finally, section 5 presents conclusions and recommendations for further work.

2. System Overview

Emergent Designer is intended for conducting design experiments in the area of structural design and for analysis of their results using methods, models, and tools from statistics and linear as well as nonlinear time series analysis. Thus, it can be used as a design support tool equipped with the state-of-the-art mechanisms for the generation of creative/novel design concepts and for conducting their optimization. Moreover, it is at the same time a versatile research tool that implements advanced methods and tools for the analysis of the design processes and of the obtained experimental results.

The following subsection provides an overview of the architecture of the system and briefly describes its major components/modules. Subsection 2.2 presents the information flow diagram and discusses integration issues as well as interactions among the components of the system.

2.1. ARCHITECTURE

Emergent Designer consists of several components. They can be divided into three major groups:

- **Design components**
  These components implement Emergent Engineering Design (Kicinger 2004), the design method which uses models of complex systems to represent structural engineering systems and design processes. They form the core of the system and implement the actual design processes.
- **Analysis components**
  These components implement tools and methods for the analysis of the experimental results and design processes. The components included in this group are aimed to provide quantitative information about the conducted design processes as well as statistical estimates of the performance of the design method. They are also intended to provide deeper understanding of the dynamics of design processes and the structure of the design spaces from global/holistic perspective.

- **Visualization components**
  These components implement various visualization methods and report generation mechanisms. They contain tools for visualization of the results of various analyses, e.g. statistical or time series, conducted by the system's components. Also, automated tools of generating experimental reports that include detailed descriptions of experimental parameter settings and obtained results are implemented.

The high-level overview of the architecture of *Emergent Designer* is presented in Figure 1. It shows the individual components of the system contained in each of the groups discussed above.

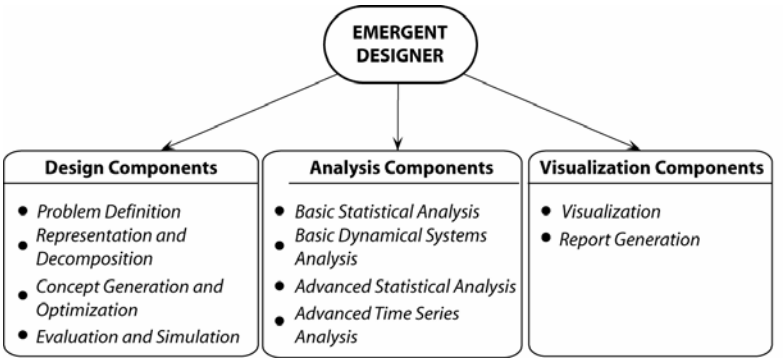

Figure 1. Architecture of *Emergent Designer*

2.2. INFORMATION FLOW

The flow of information in *Emergent Designer* is presented in Figure 2. It provides an overview of the relationships among the components discussed in the previous section and shows where user input/decisions are expected. It doesn't show, however, the information flow within the individual components. They are discussed individually in section 3.

Once *Emergent Designer* has been started, the user has the choice of conducting a new design experiment or using advanced statistical and time series analysis tools to analyze experimental data from the previous experiments. By default, a new design experiment is selected and the *Problem Definition Component* is called to define a design problem.

Problem Definition Component is intended to select a domain of interest, e.g. steel skeleton structures in tall buildings, and a specific design problem that will be solved, e.g. design of a wind bracing system. The component allows for the specification of values of the parameters defining the considered design problem, e.g. the number of stories in a tall building, or the height of a story. *Problem Definition Component* also implements

mechanisms for saving the system's parameters and their values to a file, and retrieving previously saved values from a file.

When the design problem is completely defined, the user has to decide whether, or not, to decompose the problem into several sub-problems. *Representation and Decomposition Component* is used for this purpose. If the design problem is decomposed, then one of the several decomposed representations can be selected. On the other hand, if the design problem is not divided into sub-problems, then the spectrum of possible representations includes the parameterized representations, the generative representations, and the self-adaptive generative representations (see section 3.1 for more details). *Representation and Decomposition Component* allows for the specification of values of representation specific parameters, e.g. shape of the local neighborhood in a cellular automaton used in the generative representations.

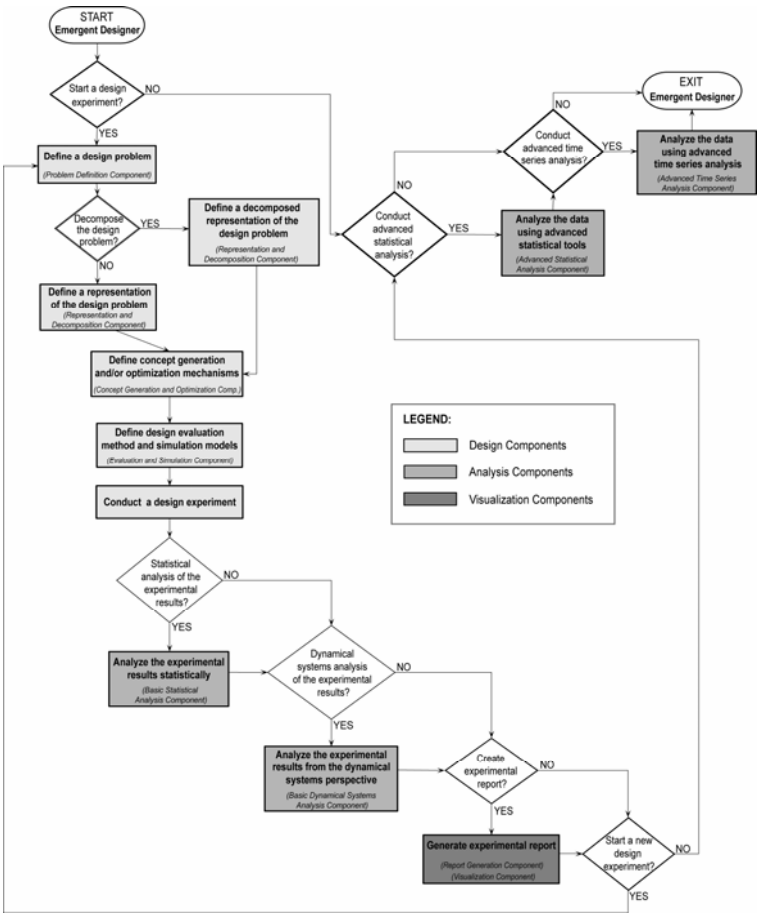

Figure 2. Information flow in *Emergent Designer*

When the design problem and its representation have been defined, the *Concept Generation and Optimization Component* is used to specify what type of concept generation mechanism will be used and whether, or not, topology optimization and/or sizing optimization mechanisms should be employed. If only pure concept generation mechanisms are selected, i.e. no optimization, then design concepts will be generated by the process of iteration conducted by cellular automata. On the other hand, if evolutionary algorithms are employed, then either only optimization of the design concepts is performed (when parameterized representations are used), or the generative representations are combined with optimization mechanisms to produce the design concepts.

The produced design concepts are transferred to the *Evaluation and Simulation Component* which evaluates them and assigns fitness value(s) (multiple fitness measures are used in the multiobjective optimization). This component is used to select the evaluation model assumed in a given design experiment and the values of evaluation specific parameters, e.g. methods of determining wind loads acting on the steel structure, or magnitudes of dead and live loads, etc. Also, simulation parameters, including the number of runs, the number of fitness evaluations, have to be defined in order to run a design experiment.

The four components described earlier, i.e. *Problem Definition Component*, *Representation and Decomposition Component*, *Concept Generation and Optimization Component*, and *Evaluation and Simulation Component*, form a group of design components that implements Emergent Engineering Design (Kicinger 2004), the design method based on models of complex systems.

Once the values of all the parameters implemented in this group of components have been determined (default values are also used where possible), the actual design experiment is initiated. *Basic Statistical Analysis Component* and *Basic Dynamical Systems Analysis Component* allow the online monitoring of the design process by providing best-so-far fitness values and trajectories of the points (design concepts) in the design spaces. *Basic Statistical Analysis Component* also provides the mechanisms for the collection of relevant experimental data and saving them in files.

When the design experiment is finished, *Basic Statistical Analysis Component* can be used to calculate and display average best-so-far fitness values with corresponding 95% confidence intervals. At that point, the user can also generate a complete experimental report listing all the parameters and their values used in the design experiment as well as the experimental results. *Report Generation Component* and *Visualization Component* are employed during the process of the automatic generation of an experimental report. *Report Generation Component* gathers the names and values of the

parameters used in the design experiment and extracts relevant experimental results. It also collects important statistical data calculated by the *Basic Statistical Analysis Component*. *Visualization Component* can be used to produce a landscape visualization graph, if applicable, and charts representing progress of individual runs in the design experiments. When all the textual, numerical, and graphical data are available, *Report Generation Component* compiles them together into a single document that is subsequently displayed as an experimental report.

At this point, the user can choose to start a new design experiment, or to analyze the experimental data using advanced statistical and time series analysis tools, or simply exit the system. If a new design experiment is selected, *Problem Definition Component* is called again and the entire process described above is repeated. On the other hand, if advanced statistical analysis, or advanced time series analysis, is chosen then *Advanced Statistical Analysis Component* or *Advanced Time Series Analysis Component* is utilized, respectively.

3. System Components

This section describes in detail each group of the system's components that were briefly introduced in the previous sections. The actual implementation of the system's components is discussed in section 4.

3.1. DESIGN COMPONENTS

Design components implement Emergent Engineering Design (Kicinger 2004), the design method based on models of complex systems and inspired by the processes occurring in nature. Individual components included in this group correspond to the major phases of the design method which are shown schematically in Figure 3. The group of design components consists of four major components which are described below.

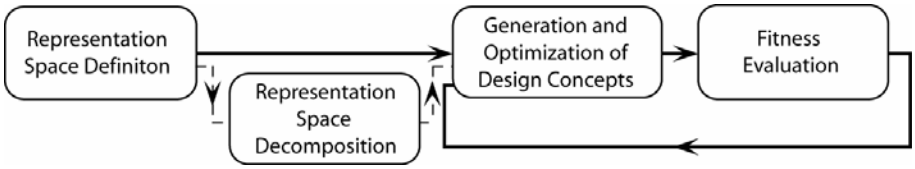

Figure 3. Phases of the design method implemented in *Emergent Designer*

3.1.1. Problem Definition Component

This component implements the preliminary phase of the design method in which a design problem is defined. The output of this component, i.e. a complete description of a design problem in terms of parameters and their

values, becomes the input to the *Representation and Decomposition Component.* This component provides necessary domain knowledge and specifies parameters of the considered design problem. It is used to perform the following tasks:

- Domain selection, e.g. steel skeleton structures in tall buildings.
- Problem selection, e.g. design of a wind bracing system, or design of the entire steel structural system.
- Specification of the problem parameters, e.g. the number of stories, or story height.

The external/user input to the component defines overall purpose (what to design), requirements, and constraints (feasibility criteria) the design should satisfy.

3.1.2. Representation and Decomposition Component

This component is used to conduct the first and second phases of the design method, i.e. *Representation Space Definition* and *Representation Space Decomposition*, in which representation of an engineering system and its decomposition, if any, are defined (see Figure 3). The input to this component consists of a complete definition of the design problem which is obtained from the *Problem Definition Component*. The output produced by the *Representation and Decomposition Component* defines the representation of the engineering system being designed. This component is used to conduct the following tasks:

- Selection of a representation for the design problem, e.g. parameterized representation, or generative representation based on one-dimensional or two-dimensional cellular automata.
- Selection of a decomposition of a given problem.
- Specification of parameters for a given type of representation (e.g., resolution for binary representations, or the neighborhood shape and the neighborhood radius for generative representations).

When all the decomposition and encoding parameters have been defined, the representation of the engineering system (artifact) is completely specified. The representation becomes the output of the *Representation and Decomposition Component* that is subsequently utilized by the *Concept Generation and Optimization Component.*

3.1.3. Concept Generation and Optimization Component

This component implements the third phase of the design method, namely *Generation and Optimization of Design Concepts* (see Figure 3). This component defines representations of the engineering design processes. The following tasks are handled using this component:

- Selection of the mechanisms of design concept generation, including various types of cellular automata (1D, totalistic 1D, 2D, totalistic 2D).
- Selection of the mechanisms of optimization of design concept, including various types of evolutionary algorithms, e.g. evolution strategies, genetic algorithms, etc.
- Specification of parameters for optimization algorithms, i.e. parent and offspring population sizes, types of genetic operators, etc.

Representation of the engineering system being designed is obtained from the *Representation and Decomposition Component* and forms the input to the *Concept Generation and Optimization Component*. The produced output consists of feasible design concepts with assigned fitness value(s).

The flow of information within the *Concept Generation and Optimization Component* is shown in Figure 4. It consists of three major subcomponents: *Concept Generation Component*, *Topology/Shape Optimization Component*, and *Sizing Optimization Component*. Depending on the type of representation of the engineering system provided as input and decisions made regarding the optimization mechanisms either only one subcomponent, or two, and even all three subcomponents can be utilized.

If only *Concept Generation Component* is used, then the design concepts are produced solely by the concept generation mechanisms, e.g. 1D or 2D cellular automata. In this case, no optimization mechanisms are employed during the design process. Generated design concepts are evaluated, given some evaluation criteria, and best designs are identified. Thus, in this case the focus of the design processes is shifted towards creativity/novelty.

On the other hand, if the engineering system is represented using parameterized encodings then no concept generation mechanism is necessary to produce design concepts from their representations. In this case, the design processes focus exclusively on optimality issues. Design optimization mechanisms can be applied at the topology/shape level (conceptual/embodiment design) using the *Topology/Shape Optimization Component* and/or member sizing level (detailed design) using *Sizing Optimization Component*.

It is possible to combine design concept generation and optimization mechanisms. In this case, creativity/novelty of generated design concepts and their optimality become equally important objectives. To achieve both objectives, concept generation mechanisms have to be defined using the *Concept Generation Component* and optimization mechanisms must be determined using the *Topology/Shape Optimization Component* and/or the *Sizing Optimization Component.*

Each produced design concept is tested for feasibility. When it satisfies all feasibility criteria defined by the *Problem Definition Component* then it is

passed to the *Evaluation and Simulation Component*, where it is evaluated and assigned fitness value(s). On the other hand, when a produced design concept is proved infeasible, then constraint-handling methods have to be employed. In the case when a repair algorithm is used, an attempt is made to repair the design concept and, if successful, the design concept is passed to *Evaluation and Simulation Component* and assigned fitness value(s). If the repair is unsuccessful, the design concept is determined infeasible and assigned worst possible fitness value(s).

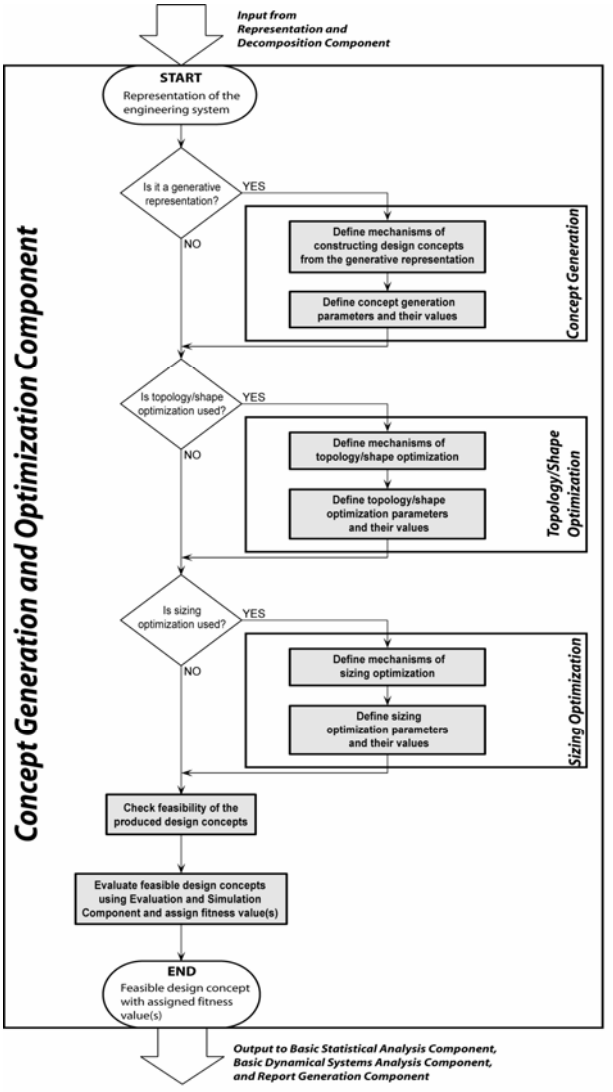

Figure 4. Information flow within the *Concept Generation and Optimization Component*

The output of Concept Generation and Optimization Component consists of feasible design concepts with assigned fitness value(s) which are subsequently passed to the *Basic Statistical Analysis Component*, *Basic Dynamical System Analysis Component*, and *Report Generation Component*.

3.1.4. Evaluation and Simulation Component

This component implements the last phase of the design method, namely *Fitness Evaluation*. It defines design evaluation models and general mechanisms of managing and monitoring simulations of design processes. The following tasks are conducted using this component:

- Specification of the load conditions considered during the evaluation process, e.g. wind loads acting on a steel structure in a tall building.
- Selection of the evaluation model and mechanisms used to measure goodness of the generated design concepts, e.g. a structural analysis package.
- Specification of the overall simulation parameters, e.g. number of runs, lengths of individual runs, etc., and monitoring of the simulation progress.

The input to this component consists of a phenotypic representation of a design concept which is obtained from the *Concept Generation and Optimization Component*. The output produced by the *Evaluation and Simulation Component* consists of the same design concept, or design concepts, provided as input but this time with assigned fitness value, or fitness values in the case of multiobjective evaluation.

3.2. ANALYSIS COMPONENTS

Analysis components implement tools and methods for the analysis of the experimental results and design processes. The components included in this group are aimed to provide quantitative information about the conducted design processes as well as statistical estimates of the performance of the design method. They are also intended to provide deeper understanding of the dynamics of design processes and the structure of the design spaces from global/holistic perspective. A brief description of the four major components in this group is presented below.

3.2.1. Basic Statistical Analysis Component

This component implements basic statistical tools for the analysis of the results of design processes. The input to this component is obtained from *Concept Generation and Optimization Component* and consists of design concepts with their fitness values as well as their data regarding when they were generated during the simulation (their “birth dates”). The following tasks are performed using this component:

- Collection of the experimental data and calculation of the best-so-far fitness statistics for each design process.
- Calculation of various statistical estimates that quantitatively describe design processes, including average best-so-far fitness and confidence intervals around the mean.
- Comparison of statistical estimates (means and confidence intervals) calculated from the results obtained in different design experiments.

The first two tasks described above are performed online, i.e. during the actual design processes. When a new design concept has been generated and evaluated, its fitness value(s) and birth date are collected. This information is subsequently saved in the files storing the experimental results data. Next, the data are analyzed and best-so-far statistics are calculated. They are also saved in the files storing statistical analysis data. At the same time, best-so-far statistics are passed to the *Visualization Component*. When the simulation is over, the average best-so-far statistics for the entire experiment are calculated and saved in a file and they are also transferred to the *Visualization Component*. The output produced by the *Basic Statistical Analysis Component* consists of basic statistical analysis data which are subsequently passed to the *Visualization Component* and the *Report Generation Component*.

3.2.2. Basic Dynamical Systems Analysis Component

This component implements basic tools and methods for the analysis of the results of the design processes from the dynamical systems perspective. In this type of analysis, the design processes are considered as dynamical systems operating in the design spaces. The subjects of analyses are the properties of trajectories (coordinates of the generated points in the design space) of design processes and identification of attractors in the design spaces. The following tasks are conducted using this component:

- Collection of the trajectories data (coordinates of the generated points in the design space).
- Reconstruction of attractors in the design spaces from the experimental data.

These tasks are also performed online and show the actual dynamics of the design processes. The trajectory information is extracted from the collected data and passed to the *Visualization Component*. Further, the trajectory data are analyzed and methods and tools of reconstructing attractors are employed, e.g. delay coordinates. The results of these analyses are subsequently transferred to the *Visualization Component*. The output produced by the *Basic Dynamical Systems Analysis Component* consists of basic dynamical systems analysis data and is utilized by the *Visualization Component*.

3.2.3. Advanced Statistical Analysis Component

This component implements advanced statistical analysis methods, models, and tools for the analysis of the experimental results. The statistical analysis conducted by this component is performed offline when no design experiments are running. The input is obtained from the files storing the experimental results that were previously saved using the *Basic Statistical Analysis Component*. *Advanced Statistical Analysis Component* contains the tools for analyzing the sample distributions and making inferences about their means and medians. The following types of tasks can be conducted using this component:

- Reading the experimental data from file(s).
- Qualitative and quantitative analysis of the sample distributions, e.g. histograms, normal scores plots, skewness and kurtosis estimates, etc.
- Estimation of statistical quantities from the data, including means and medians, using various point estimates and interval estimates.
- Saving the results of the statistical analysis in a file.

The results of the advanced statistical analysis are transferred to the *Visualization Component* and subsequently displayed in the form of charts, graphs, and histograms and/or saved in the files.

3.2.4. Advanced Time Series Analysis Component

This component implements advanced tools and models from the linear and nonlinear time series analysis. The analysis, similar to the one performed by the *Advanced Statistical Analysis Component*, is conducted offline. Also, the input consists of the experimental results stored in the previously saved files. The following types of tasks can be conducted using this component:

- Reading the time series data from file(s).
- Qualitative and quantitative analysis of the time series data using various methods and tools, e.g. delay coordinates, power spectrum, autocorrelation, etc.
- Saving the results of the time series analysis in a file.

The output produced by the *Advanced Time Series Analysis Component* consists of the results of various analyses and is utilized by the *Visualization Component* and/or saved in the files.

3.3. VISUALIZATION COMPONENTS

Visualization components implement various visualization methods and report generation mechanisms. They contain tools for the visualization of the results of various analyses, e.g. statistical or time series, conducted by the system's components, and simple fitness landscapes. Also, automated tools of generating experimental reports that include detailed descriptions of

experimental parameters and their values and the obtained results are implemented. There are two major components in this group:

3.3.1. Visualization Component

This component implements various methods of data visualization. It is aimed to provide a qualitative analysis of the experimental results and the necessary functionality to save produced graphs and charts in files and experimental reports. The input to this component consists of the experimental data and is obtained from various components of the system. The following types of tasks can be conducted using this component:

- Display of generated design concepts.
- Interactive display of simple three-dimensional fitness landscapes.
- Display of statistical, dynamical, and time series analyses conducted using various components of the system.

The experimental results obtained as input to this component are collected and information relevant to display and visualization purposes is extracted from the data. Next, depending on data source, appropriate graphs and charts are produced including line charts, scatter plots, histograms, and renderings. The produced graphs are next displayed by Emergent Designer's graphical user interface (GUI). Each generated graph may also be saved in a file. This last option is implicitly used by the *Report Generation Component* which utilizes various graphs produced by the *Visualization Component* during the process of automatic generation of the experimental reports. The graphs included in the reports are first saved in files and subsequently read by the *Report Generation Component*.

3.3.2. Report Generation Component

This component implements mechanisms for the automatic generation of experimental reports. It is intended to provide complete information about the experimental parameters and their values as well as the obtained results. The input to this component is obtained from various components. The following types of tasks can be conducted using this component:

- Collection of the experimental parameters and their values used in the reported experiment.
- Collection of the numerical results of various runs in the reported experiment.
- Collection of statistical analysis data and various graphs illustrating progress of individual runs and average performance during the entire design experiment.
- Automatic generation of a full report containing all above mentioned elements.

The output produced by the *Report Generation Component* consists of a complete experimental report which is displayed in the system's GUI and/or saved in the file.

4. Implementation

Emergent Designer has been implemented with fully functional graphical user interface using Java. The decision to use this particular programming language was made due to the fact that several of the system's components were built upon existing packages written in Java. Moreover, *Emergent Designer* integrates several commercially available systems (e.g., *Mathematica*© (Wolfram 2003) and *OpenOffice.org*©) and communicates with them using available Java APIs.

Another important aspect that influenced the choice of the programming language was the fact that Java is portable and network-oriented. Portability offers the flexibility of running the system on various platforms. Built-in networking capabilities open the possibility of using distributed architectures. Both of these issues are particularly important in structural design where the process of evaluation of generated design concepts is usually computationally expensive and conducted using specialized structural analysis software.

4.1. DESIGN COMPONENTS

Components implementing the design method constitute the core of *Emergent Designe*r. The functionality described in sections 3.1.1 – 3.1.4 was either directly implemented or borrowed from several existing packages and commercial systems that were integrated with *Emergent Designer*.

Two domains have been implemented in the *Problem Definition Component*: the domain of steel structural systems in tall buildings and the domain of real-valued functions (added for testing purposes and analysis of the behavior of various components of the system). The domain of steel structural systems includes two major classes of design problems: design of a wind bracing system in a tall building and design of the entire steel structural system in a tall building.

Representation and Decomposition Component allows four types of representations: binary, real-valued, integer-valued and cellular automata. The first two types are used mostly for real-valued problems while the latter two are applied to encode the designs concepts of steel structural systems in tall buildings. Real-valued and binary representation implementations were inherited from several existing evolutionary computation packages, including ec1, ec2, and ec3 (De Jong to appear). On the other hand, integer-valued and cellular automata representations were directly implemented in the system.

Concept Generation and Optimization Component has been built upon four major existing packages and commercially available systems. Design concept generation utilizing various types of cellular automata is conducted by *Mathematica*© kernel which was integrated with *Emergent Designer* via JLink™. All major types of cellular automata (CA) are supported, including 1D CA, totalistic 1D CA, 2D CA, and totalistic 2D CA.

Topology/shape optimization using an evolutionary algorithm (EA) is performed by ec3 package (a Java-based EC toolkit (De Jong to appear)). Here, all canonical evolutionary can be utilized, including genetic algorithms, evolutionary programming, and evolution strategies. The system also offers an option of employing a unified EA (De Jong to appear) in which all major elements the EA, i.e. generational model, parent selection, offspring selection, population sizes, operators, etc., can be tuned to the particular problem being solved.

Sizing optimization, if applied, is conducted using an optimization algorithm based on traditional mathematical programming method and implemented in SODA. It is a commercially available structural analysis, design and optimization system developed by Waterloo Systems in Waterloo, Ontario, Canada which was integrated with *Emergent Designer* to perform evaluation of designs and their sizing optimization.

Evaluation and Simulation Component implements evaluation models used to determine fitness of generated solutions. Current status of the system allows only for a single objective evaluation of individual design concepts using one of the following evaluation criteria: the total weight (an estimate of the cost) or the maximal horizontal displacement (an estimate of the stiffness) of the steel structural system. The determination of a least-weight structure is performed by SODA and is conducted in conformance with the strength (stability) and stiffness (displacement) provisions of several commonly used steel codes, including AISC-ASD-89, AISC-LRFD-86, AISC-LRFD-93, CSA-Sl6.1-M89, or CSA-Sl6.1-94. Loading model required for evaluation of generated design concepts includes dead, live, and wind loads determined in conformance with the corresponding design codes. Wind forces are calculated for given design cases using a modified commercial system Wind Load© V2.2.S developed by Novel CyberSpace Tools.

Graphical user interface of *Emergent Designer* displaying *Representation and Decomposition Component* is presented in Figure 5.

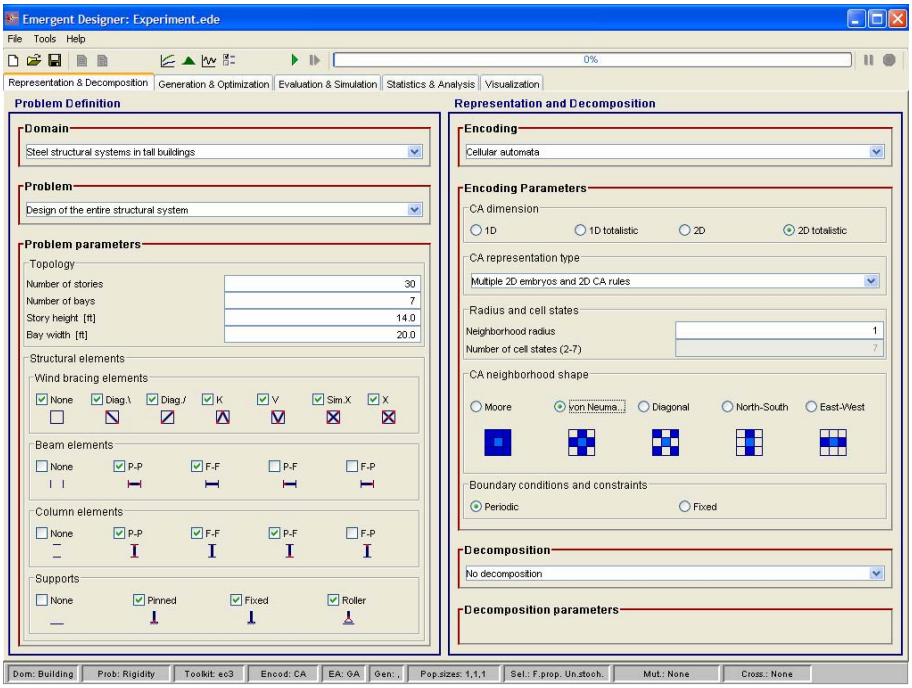

Figure 5. Graphical user interface of *Emergent Designer* showing the *Representation and Decomposition Component*

4.2. ANALYSIS COMPONENTS

Methods and models for basic statistical and dynamical systems analysis have been implemented directly in Java. The simple analysis is conducted online, i.e. during the actual design processes. Basic statistical analysis involves best-so-far fitness statistics calculated for individual runs and average best-so-far fitness statistics and 95% confidence intervals computed for the entire design experiment. This analysis is automatically saved in files.

Implemented methods of simple dynamical systems analysis include trajectory analysis which shows the dynamics of the processes in the design spaces as well as delay coordinates analysis. Delay coordinates are computed from the best-so-far fitness values with an arbitrarily assumed time lag.

Advanced statistical and time series analysis is performed offline, i.e. when the design processes are completed. Advanced statistical analysis includes estimation of sample distributions using histograms, normal scores plots, symmetry plots, and estimators of sample kurtosis and sample skewness. It also implements various estimators of means and medians of

the sample distributions and their corresponding confidence intervals. Implemented mean point estimators include: the sample mean and the trimmed mean. Confidence intervals around means can be computed using the following three methods: normal approximation, Student's t test, and Johnson's modified t test. Point estimates of medians may be calculated using the sample median and the trimmed mean while the confidence intervals around median are determined by the following two methods: sign test (Thompson-Savur formula) and normal approximation (using the conservative approach). Advanced statistical analysis tool and methods have been in part implemented directly and partially borrowed from *JMSL© Numerical Library* that was integrated with *Emergent Designer*.

Advanced time series analysis offers the following methods of analysis of the experimental data: visual analysis of the time series data, delay coordinates plots with adjustable parameters determining the embedding dimension and the time lag, power spectrum analysis, autocorrelation analysis with a flexible specification of autocorrelation lag and standard error bars according to either Barlett's or Moran's formula, and two types of recurrence plots: regular and thresholded with a flexible specification of the embedding dimension, time lag, and norm to calculate the distances between the points of a time series.

Also in this case, several tools and methods of advanced time series analysis were directly implemented in the system while several have been borrowed from *JMSL© Numerical Library*.

4.3. VISUALIZATION COMPONENTS

Experimental data are visualized in *Emergent Designer* in three major ways. First, line plots and scatter plots (or more generally signal plots) are used to visualize experimental data transferred from the *Basic Statistical Analysis Component* and *Basic Dynamical Systems Analysis Component*. These types of graphs include best-so-far fitness plots, average best-so-far fitness plots with error bars, trajectory plots, and delay coordinates plots. The plots are produced by a Java-based signal plotter called PtPlot developed at UC Berkeley. They are embedded in the Emergent Designer's graphical user interface and can be subsequently saved as bitmap files (in eps and png formats). Histograms are another type of graphs generated during the analysis conducted by the *Advanced Statistical Analysis Component*. These types of graphs are created using *JMSL© Graphical Library* integrated with *Emergent Designer*. They are also embedded in the system's GUI and provide functionality to save the produced graphs as bitmap files. Finally, interactive renderings of simple real-valued fitness landscapes can be produced. This type of visualization is produced using *Mathematica*'s

advanced graphical capabilities. Generated renderings are displayed in the system's GUI using *JLink*.

Automatic report generation capabilities have been achieved by the integration of *Emergent Designer* with *OpenOffice.org*© using its Java API. *Report Generation Component* collects and organizes the textual, numerical and graphical information that should be included in the experimental report and then creates an *OpenOffice.org* document which is subsequently displayed on the screen. The report can be later saved as a file in any of the format supported by the *OpenOffice.org* and thus provide complete summary of the experimental parameters and the obtained results.

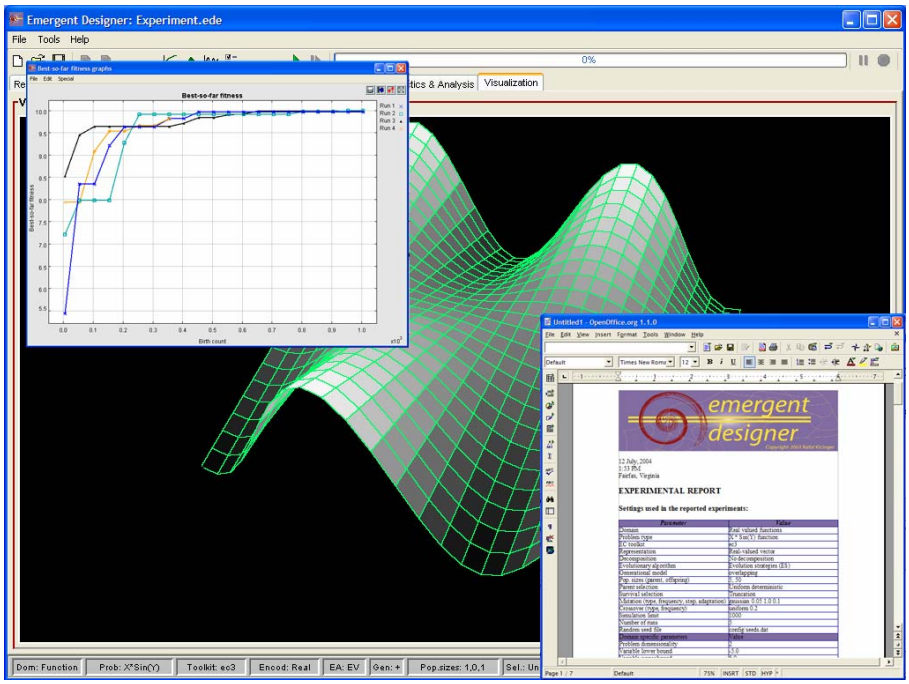

Figure 6. Visualization and report generation capabilities of *Emergent Designer*

5. Conclusions

Emergent Designer is a complex design support tool which belongs to a new generation of design tools being developed at George Mason University. They result from several years of intense research on evolutionary designing, on complex adaptive systems, and on cellular automata in the context of New Kind of Science as proposed by Wolfram (2002). The developed system is intended for research purposes, but in the future it will be adapted for the practical engineering design applications.

The initial experience with *Emergent Designer* has clearly demonstrated that the concept of the system was feasible. The system provides fascinating research opportunities related to evolutionary design and cellular automata. It is too early to provide a balanced evaluation of the system's advantages and disadvantages, but the authors believe that its use will soon lead to various discoveries related to design and engineering creativity, particularly related to emergence.

References

Arciszewski, T and De Jong, KA: 2001, Evolutionary computation in civil engineering: research frontiers, *in* BHV Topping (ed.) *Proceedings of the Eight International Conference on Civil and Structural Engineering Computing*, Saxe-Coburg Publications, Eisenstadt, Vienna, Austria.

Baron, P, Fisher, R, Mill, F, Sherlock, A and Tuson, A: 1997, A voxel-based representation for the evolutionary shape optimization of a simplified beam: a case-study of a problem-centered approach to genetic operator design, *in Proceedings of the 2nd On-line World Conference on Soft Computing in Engineering Design and Manufacturing (WSC2).*

Bentley, PJ: 1999, An introduction to evolutionary design by computers, *in* PJ Bentley (ed.) *Evolutionary Design by Computers*, Morgan Kaufmann Publishers, San Francisco, CA.

Bentley, PJ and Kumar, S: 1999, Three ways to grow designs: a comparison of embryogenies for an evolutionary design problem, *in* W Banzhaf, J Daida, AE Eiben, MH Garzon, V Honavar, MJ Jakiela and RE Smith (eds.), *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'99)*, Morgan Kaufmann Publishers, Orlando, Florida, USA, pp. 35-43.

De Jong, KA: to appear, Evolutionary computation: a unified approach, MIT Press, Cambridge, MA.

Frazer, J: 1995, An evolutionary architecture, Architectural Association Publications, London.

Funes, P and Pollack, JB: 1999, Computer evolution of buildable objects, *in* PJ Bentley (ed.) *Evolutionary design by computers*, Morgan Kaufmann Publishers, San Francisco, CA.

Kane, C and Schoenauer, M: 1995, Genetic operators for two-dimensional shape optimization, *in* J-M Alliot, E Lutton, E Ronald, M Schoenauer and D Snyers (eds.), *Artificial Evolution*, Springer Verlag.

Kicinger, R: 2004, Emergent Engineering Design: Design creativity and optimality inspired by nature, Ph.D. Dissertation, School of Information Technology and Engineering, George Mason University, Fairfax, VA, USA.

O'Reilly, U-M, Kangas, M and Testa, P: 2000, MoSS: Morphogenetic Surface Structure: a software tool for design exploration, *in Proceedings of the Greenwich 2000: Digital Creativity Symposium*, University of Greenwich, London, UK, pp. 71-80.

Roston, GP: 1994, A genetic methodology for configuration design, Ph.D. Dissertation, Department of Mechanical Engineering, Carnegie Mellon University, Pittsburgh, PA.

Stiny, G: 1980, Introduction to shape and shape grammars, *Environment and Planning B: Planning and Design* **7**(3): 343-351.

Wolfram, S: 2002, A new kind of science, Wolfram Media, Champaign, IL.

Wolfram, S: 2003, The Mathematica book, Wolfram Media, Champaign, IL.