Reducing design complexity of multidisciplinary domain integrated products: a case study

Juan M. Jauregui-Becker, Wessel W. Wits and Fred J.A.M. van Houten
Laboratory of Design, Production and Management, Faculty of Engineering Technology,
University of Twente, The Netherlands.

Abstract
Multidisciplinary product development is well known for the complexity of its design process. It is commonly addressed by domain integration and a modular design approach. The former, often resulting in smaller products and integrated functions, is characterized by a complex non-linear design process. The latter, which may not result in such integrated functions, has a simpler –usually linear– design process, resulting in novel solutions. This paper presents a method for reducing design complexity of Multidisciplinary Domain Integrated Products by decomposing the problem into modular structures. Computational synthesis techniques are used to solve the resulting modules. Printed Circuit Board design is used as case study, as it is well known for its complexity and highly integrated product functionalities.

Keywords:
Domain integration, multidisciplinary design, complexity reduction, computational synthesis

1 INTRODUCTION

Advances in technology are driving modern products towards further miniaturization, better quality, more functionality, and yet cheaper prices. This trend is imposing a great challenge on product development, thus becoming extremely complex. To meet the challenging criteria, several disciplines are often integrated in one product and, as such, require a multidisciplinary product development. However, multidisciplinary product development is a very complex task, due to the amount of engineering disciplines that merge together into one design artifact.

Engineers often use two main approaches to tackle multidisciplinary product development, namely, modular design or domain integration. A modular approach isolates product functions and maps them onto independent modules. Domain integration incorporates different functions into a single module, resulting in Multidisciplinary Domain Integrated Products (MDIP). Designing for MDIP requires the knowledge of multiple engineering fields simultaneously, before arriving at any design solution. Decomposing the design into smaller chunks –a divide and conquer strategy- is often used to reduce complexity [1]. However, the resulting chunks are required to have clear boundaries, governing principles and requirements. According to Goel et al [2], design decomposition has to be based on problem structure rather than on designer's experience. This avoids unwanted side effects, such as multiple interdependencies among sub-problems and disjoints sub-problems, whose articulation constitutes a problem as such. However, given the integrative nature of MDIP, many interactions among decomposed sub-problems usually appear. This increases the complexity of the design significantly. Therefore, intelligent strategies to treat each sub-problem independently and modular while considering the interactions among each other are required.

This paper presents a method to reduce design complexity of MDIP. This method is demonstrated in the field of Printed Circuit Board (PCB) design, which will be introduced in Section 2. The problem is organized and structured as shown in Section 3. The results are then used in Section 4 to decompose the problem. Functional, topologic and physical characteristics are used for this purpose. As many interactions appear, an algorithm is presented to determine the optimal sequence for instantiating each sub-problem. Section 6 discusses how a Computational Synthesis System (CSS) can encapsulate the resulting problem chunks into separate modules, allowing designers to follow a modular design approach.

2 CASE STUDY: PRINTED CIRCUIT BOARD DESIGN

PCB technology is a well established mass-market production method, appreciated for its high degree of integration of mechanical and electronic functions. Polymeric layers, on which a conductive pattern is produced, are laminated together to form a PCB. On top and bottom of the PCB, electronic components (e.g. IC's, resistors, connectors) can be assembled, thus forming a circuit card assembly.

Traditionally, the design process of most electronic products has been dominated by electrical and mechanical requirements. As cooling requirements are (usually) not a primary function of electronic products, its design has been under-addressed. In fact, recent literature study indicates that a limit has been reached for cooling electronics in general [3]. An effective method to conquer this design challenge has been presented by Wits et al. [3]. Here, knowledge of heat transfer and production principles was combined through domain integration into the overall design process of electronic products. Their concept demonstrator, illustrated in Figure 1, incorporates heat pipes produced by the PCB structure itself to effectively transport heat away from the dissipating elements. Heat exchangers can be located independently and further away from their respective heat

source (i.e. the electronic components). For PCB technology in general, this concept leads to new manufacturing strategies where thermal management functions and electronic circuitry are fully integrated. It also results in a more compact electronic system, producible at a lower cost. However, inevitably, the design complexity of such a product also increases, thus, requiring new methods and strategies to cope with this family of highly integrated design challenges.

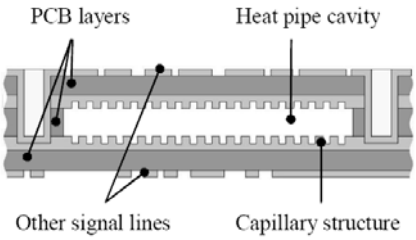

Figure 1: Heat pipes integrated into the PCB [3].

3 PROBLEM STRUCTURE

The formulation scheme of routine design problems presented by Jauregui-Becker et al [4] is used in this paper to structure the MDIP design, as it allows separating different types of knowledge present in design problems. In addition, it structures and models problems such that the computational processes that are required to automatically generate design solutions can easily be identified. The following sections present a short summary of what it embraces.

3.1 Definitions

Element: is a class description of a design artifact component.

Descriptions: characterize an element class by representing its attributes in the form of variables.

Embodiment: is considered as the subset of descriptions of an element upon which instances are created to generate design solutions.

Scenario: is considered as the subset of environment variables, attributed to elements in the natural world and considered in measuring a design artifact's ability to accomplish its function.

Performances: are those descriptions used to express and assess the artifacts behavior, and are calculated using analysis relations.

Analysis relations: use known theories, for instance the laws of physics or economics, to model the interaction of the design artifact with its environment and to predict its behavior.

Topology relations: define the configuration between embodiment and scenario elements.

Objective function: weighs and adds the performances to result in the overall performance of the design.

Confinement constraints: determine the range in which a description can be instantiated.

3.2 Design problem formulation

Embodiment elements and scenario elements, with their attributed descriptions, are used to describe the initial state of a design. Objective function, performance indicators and analysis relations express and assess the goal of the design artifact. Topological relations indicate the set of logical states between elements that have to hold for the design artifact to exist. The RCC-8 [5] formalism is used here to represent the topologic configuration of the design, as it is ideally suited for qualitative spatial representation and reasoning. It comprehends the base relations shown in Figure 2, and allows the formalization of concepts as convergence, connectedness and continuity.

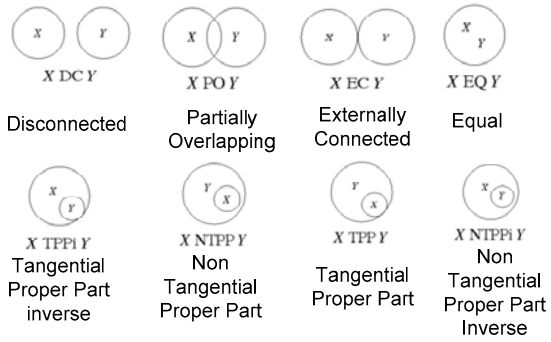

Figure 2: RCC-8 base relations.

In Figure 3, a formulation is presented of the PCB design of our case study, discussed in Section 2. For explanatory reasons, not all descriptions and relations have been included. Figure 3 shows that a PCB consists of laminates, heat pipes and wiring in one embodiment element. The properties of the electronic components –such as location, mass and generated heat– are regarded as scenario. Topology, analysis and objective function indicate how all elements in the formulation relate to each other. These relations also show the complexity of PCB design in general.

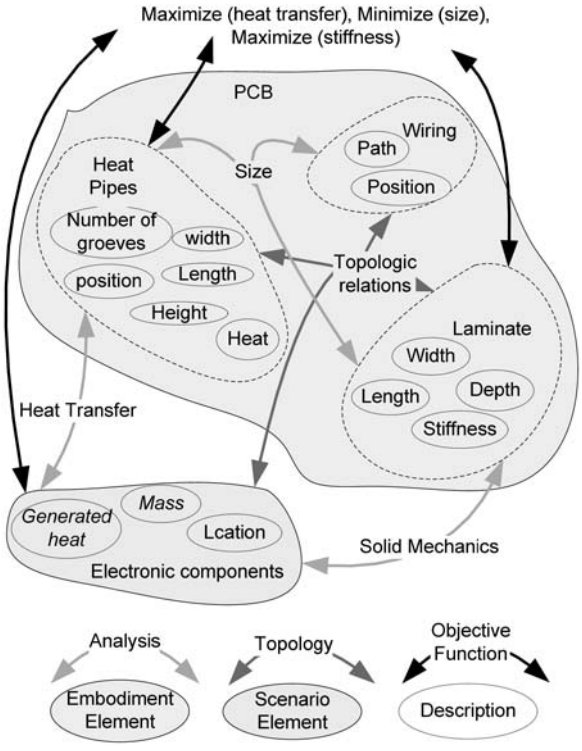

Figure 3: PCB problem formulation.

4 PROBLEM DECOMPOSITION

As a structured formulation of the problem is realized, decomposition is done according to:

1. A model of the elements and its topology is developed, as shown in Figure 4(a) for PCB design. The following convention is used to do so:
   - All elements in the topology must be connected by at least one base relation.
   - Relations have a direction indicating which the reference element of the relation is. For example, in Figure 4(a), the element '*Laminate'* is the reference element to which the element *'Electronic Components*' is related to.
   - The direction of the relation is given by interrogating which element has to exist in order to be able to accommodate the other.
2. The functions of the elements in the embodiment are listed. Functions that are not related to other elements in the topology are not taken into account. For PCB design, the following functions are listed:
   - Laminate: accommodate electronic components, heat pipes and wiring.
   - Wiring: Connect electronic components.
   - Heat pipes: Remove heat from electronic components.
3. Elements presenting several functions are split into new elements, one for each of the encountered functions. For integrated elements with several functions this results in independent elements, for each of the functions of the design artifact. Figure 4(b) illustrates the separation of elements for *'Laminates'*, *'Heat Pipes'* and *'Wiring'*.
4. As the topology has been split into new elements, relations emerging from the separation have to be re-formalized. This is done according to the same guidelines presented previously. The result is a new problem formulation, as shown in Figure 4(b). For instance, here *'Heat Pipes'* and *'Wiring'* are connected to *'Laminate'* by an inside relation (TPP).
5. Now, we categorize the analysis relations by their underlying theories. For this purpose, we make use of perspectives, which are regarded as abstraction levels founded by the underlying theories upon which the goal of the design is assessed. In our case heat transfer, solid mechanics and geometry. Perspectives are identified by assessing the performances. In the case of subject PCB design, three perspectives are defined, as shown in Figure 4(c):
   - Heat transport: Uses heat transfer as underlying theory to determine the amount of heat that the heat pipes are able to remove from the electronic components.
   - Stiffness of the laminates: Uses solid mechanics theory to determine the stiffness of the laminates under the presence of mass of the electronic components and the laminate itself.
   - Size of the PCB: Uses geometry to determine the total volume of the laminates.

   The direction of the relations is determined following the same convention as for the topologic relations.

The result of the applied method is a set of syntactic rules indicating how elements and functional element instantiations is limited by the previous instantiation of other elements. The final decomposition is shown in Figure 4(d). Here, labels are attached for usage in Section 5.

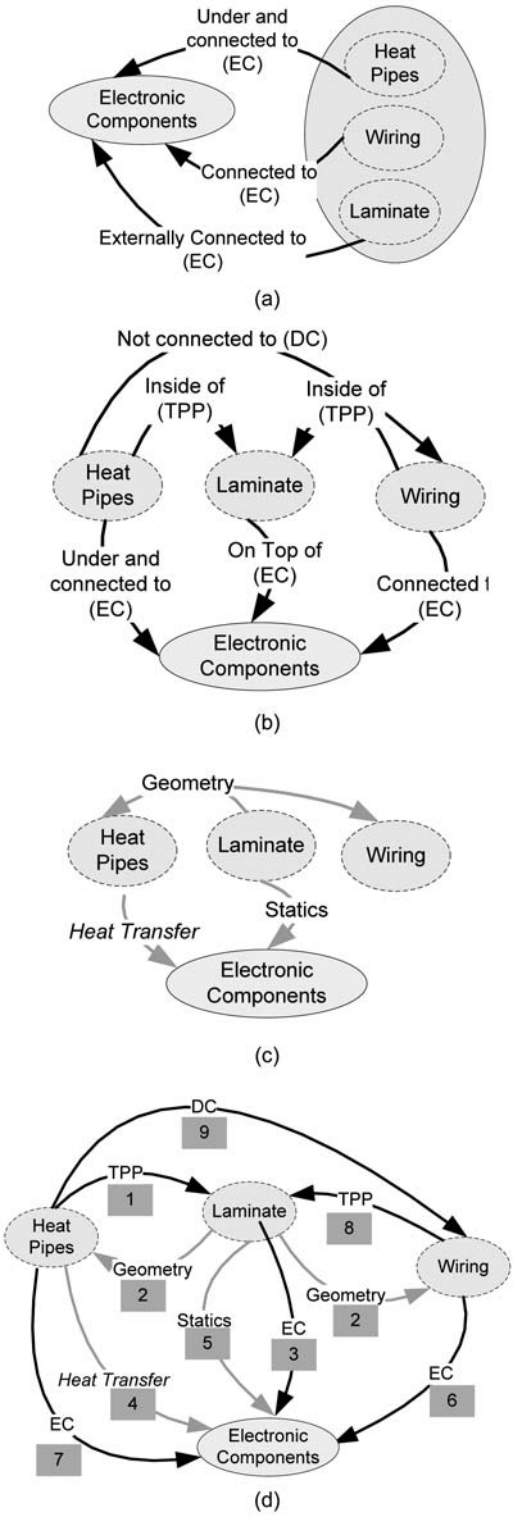

Figure 4: Several steps in PCB decomposition.

5 PROBLEM SOLVING STRATEGY

As shown in Figure 4(d), several additional interactions appear as consequence of the decomposition. Interactions do not necessarily represent an escalade of the design complexity. If managed properly, they determine how to proceed in solving the problem. For this purpose, a method was developed to determine strategies to analyse and optimize the decomposed problem. The basis of this method is found in Operations Research (OR). Here, mathematic algorithms are used to make optimal decisions in complex problems concerning optimization. The aim is to reduce NP-Hard complexities into NP ones. In this section, we will focus on the method itself and not in how it was implemented, as the object of this paper is rather instrumental and not analytical.

5.1 Design Structuring Matrix

The problem we want to solve is that of determining how to optimally proceed in the instantiation of the elements in the design structure, and which relations should be taken into account at each step in the strategy. To do so, we make use of a Design Structure Matrix (DSM). Figure 5 shows a DSM for PCB design strategy optimization. In this figure, E stands for *'Electronic Components'*, H for *'Heat Pipes'*, L for '*Laminate'* and W for '*Wiring'*. Each row and column in the matrix is labeled after one element. To fill the DSM, the following steps are taken:

1. Select one element of the model.
2. List the relations pointing towards that element.
3. Fill the row, attributed to that element, with the listed relations.

For example, the element *'Heat Pipes'* has one relation pointing inwards, connecting this element to the element *'Laminate'*. In the DSM this is represented by the label '2' at row H and column L. As shown in Figure 5, one row and one column are included for measuring the number of constraints to be satisfied in the instantiation of an element, and the number of constraints imposed by one element onto others, respectively. The former is regarded as *effort*, and its value equals the amount of relations present in a column. It represents the degree of complexity attributed to the instantiation of an element. The latter is regard as *potential*; its value equals the amount of relations present in a row. It represents a measure to reduce the complexity of following elements instantiations.

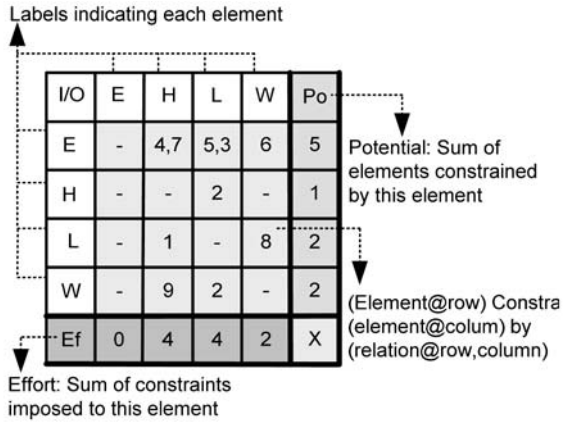

Figure 5: DSM for PCB design strategy.

5.2 Sequential strategy

After filling the DSM matrix, a number of steps follow to complete the design strategy. The design strategy describes the instantiation sequence, and the relations used at each instantiation step. The strategy is developed as follows:

1. Identify the element to be instantiated by:
   a. Finding the elements with the lowest effort.
   b. Selecting the element with the highest potential.
   c. Setting its effort to zero (0), as shown in Figure 6(b).

   The selected element thus becomes the next to be instantiated.
2. List the relations contained by the column of the selected element, and erase them from the matrix. This is shown in Figure 6(b), (c) and (d) by the zeros replacing the erased relations labels. These relations are to be taken into account at the instantiation of the selected element, as indicated in Table 1.
3. Calculate the potential with the remaining relations, as shown in Figure 6(b), (c) and (d) where, as the method progresses, new potentials are calculated with the remaining relations.
4. Calculate efforts considering elements with non-zero efforts.

This process is repeated until all elements and relations have been addressed in the strategy. In the case of subject PCB design, this routine is applied four times. Figures 6(a-d) show how the matrix develops as the method is applied. Table 1 specifies the resulting instantiation order and weighs the reduced complexities by the updated efforts.

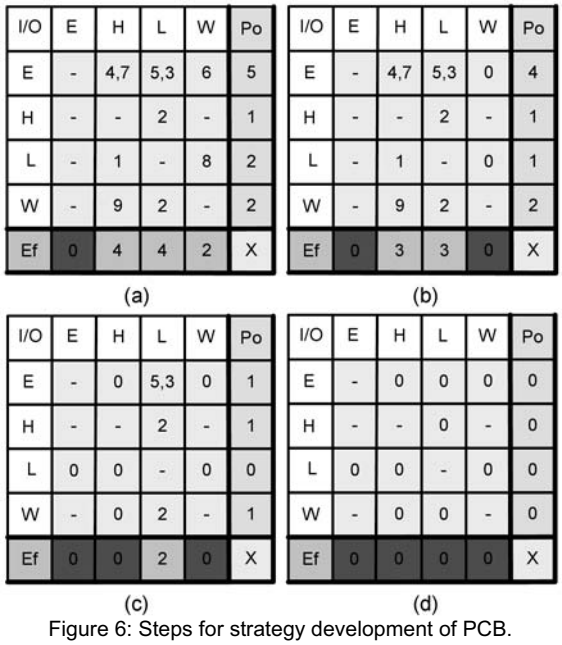

Figure 6: Steps for strategy development of PCB.

Table 1: Design strategy for PCB design.

Sequence	Element	Relation	Initial effort	Updated Effort
1	Wiring	6,8	2	2
2	Heat pipes	4,7,1,9	4	3
3	Laminate	5,3,2	4	2

6 SYNTHESIS TOOL DEVELOPMENT

The result of the decomposition is a linear design process, where each step is regarded as one sub-problem. Results of one step are regarded as scenario (input) for the following design steps. Several loops may be required before finding at least one valid solution. By applying CSS to each of these sub- problems, while designers control the flow of data between their interfaces, allows for further reduction of complexity. In this section, some generalities of CSS development are discussed. We limit the scope of CSS to routine design problems with parametric representations. A demonstration will be presented for heat pipe design in PCB.

6.1 General considerations for CSS development

A well accepted generic model of the design process is shown in Figure 7. According to this, embodiments are generated in a synthesis process. Then they are analyzed to calculate its performance and evaluated to assess whether the design is to be adjusted (path 1), rejected (path 2) or accepted (path 3). The four main processes to capture in a CSS are Synthesis, Analysis, Evaluation and Adjustment, of which Synthesis is the core one. Many techniques for its execution are found in literature, as for instance generation and test, backward reasoning, database lookup, search in a problem space, abduction, generative rules, case-based reasoning, grammar production, computational models, parametric search/generation, constraint-solving, and genetic algorithm.

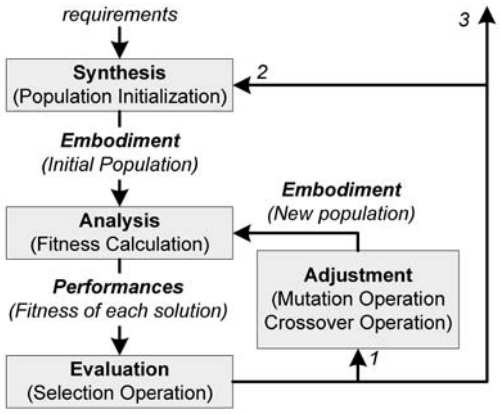

Figure 7: GA vs. generic design process.

Selecting a specific technique depends mainly on the type of model used in the representation. Jauregui-Becker et al [4] presented a classification for descriptions and relation models commonly used for design representations. Using this classification, four main problem distinctions were made: parametric, configuration, layout and shape. Parametric representations are commonly solved by generation and test, constraint-solving and genetic algorithm, whereas grammatical approaches are very common for the generation of configurations and shapes. Layout problems are usually solved by optimization algorithms, as simulated annealing or genetic algorithms.

6.2 Computational synthesis for heat pipe design

The physical model and implementation for a PCB with integrated heat pipe cooling is presented by Wits et al in [6]. This model is used here to derive the parameters and relations considered in the CSS. Figure 8 summarizes embodiment parameters in relation with performance and scenario parameters. For instance, a change in groove dimensions or internal height alters the heat pipe's performance, and thus its capacity to remove heat from the associated electronic components.

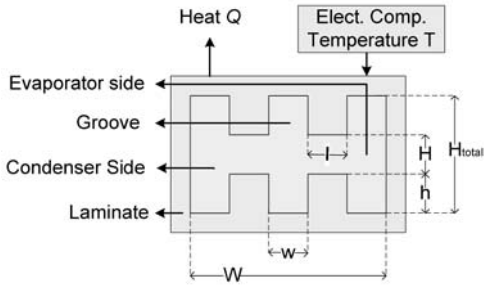

Figure 8: Embodiment and performance for PCB design.

As the model is fully parametric, we have chosen for Genetic Algorithms (GA) as solution generation technique, developed as indicated by Barrell in [7]. GA incorporates the process of Synthesis, Analysis, Evaluation and Adjustment required in a CSS, as shown in Figure 7 where the GA algorithm is presented together with the mentioned processes. This characteristic makes GA implementations straightforward for CSS applications for parametric routine design problems. The GA algorithm was implemented using C# as the programming language. The program consists of five classes: *Population*, *ListGenoma*, *Genom*, *Model and UserInterface*, as shown in Figure 9.

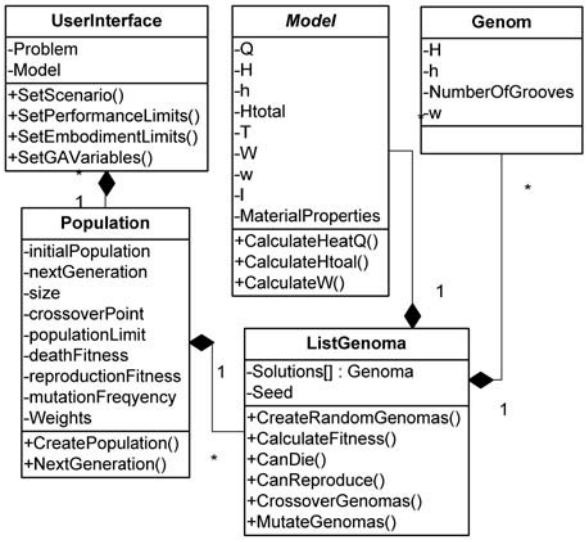

Figure 9: Class diagram of the CSS.

The class *Population* contains variables specifying the size of the initial population, maximum population, death fitness, reproduction fitness, mutation frequency, crossover point and weight of the performance parameters. It stores the generated populations in the class *ListGenoma*. The class *ListGenoma* contains a population of solutions, where each solution is an object of the class *Genom*. *ListGenoma* contain methods to compute the GA operators. These methods are called by the *Population class* using variables that characterize the solution generation and adjustment (e.g. size of population). *Genom* is the class that encapsulates the

embodiment parameters of one candidate solution. The *Model* class contains properties of the global physical model related to each Genom instantiation. This class responds to calls from the *ListGenoma* class to calculate the performance of each candidate solution. The class *UserInterface* allows the user to enter input requirements of a specific problem to be solved. The modularity of this architecture allows the re-use of program code by minor changed of the underlying classes.

Simulations were carried out with the developed software tool to determine how fast solutions could be generated. A first simulation is used to scan the solution space. The ranges of the desired requirements for this simulation are shown in Figure 10, where the user input interface of the developed tool is shown.

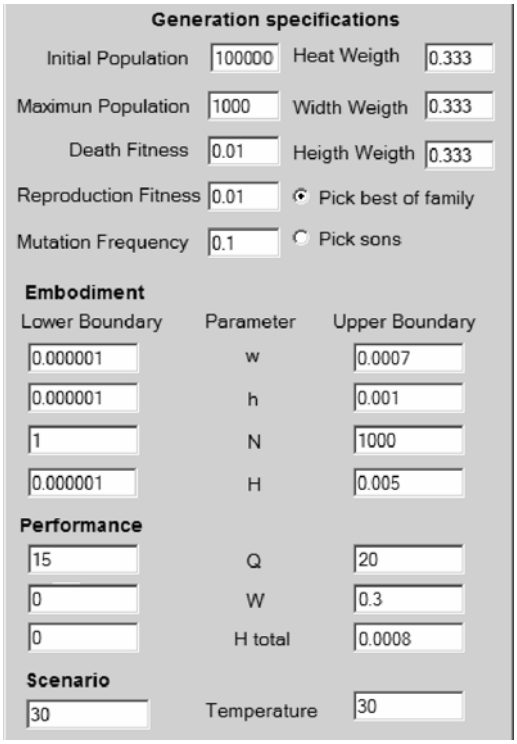

Figure 10: User input interface.

From the range of parameters, solutions are generated randomly and filtered for fitnesses higher than 0.001, in about 2*s*. The results are shown in Figure 11, where the performance parameters *'Heat'* and the *'Area'* of the designed heat pipes are shown. As the figure shows, 'generate and test' gives an overview of how the solutions are distributed in the solution space. The outcome of the initial simulation is used in a second iteration, for instance to zoom in to the selected region in Figure 11. This is done by updating the confinement constraints of the allowed values for the Heat (Q) and the area (WxH). This simulation took about 5*s* to find the local optima illustrated in Figure 11.

7 SUMMARY

This paper presented a method to reduce design complexity of Multidisciplinary Domain Integrated Products. This method allows decomposing of such MDIP design challenges into modular structures. The resulting structures can then be implemented into a Computational Synthesis System. This allows designers to assess the solution space in shorter times, while keeping an insight in the found solutions. The method was demonstrated by a case study to design a complex multilayer Printed Circuit Board with integrated heat pipes. A CSS was developed to optimize the dimensioning of the integrated heat pipes as a function of required cooling capacity and board layout. The results demonstrated the feasibility and applicability of the developed method for MDIP design.

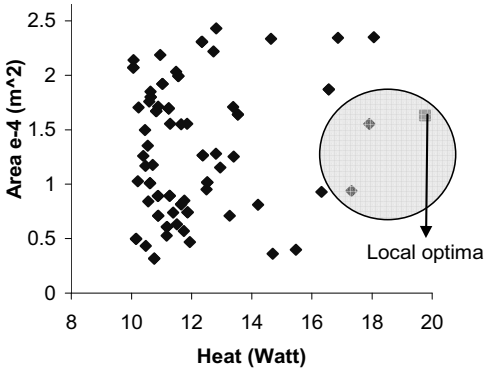

Figure 11: Performances for CSS generated designs.

LITERATURE

[1] Tomiyama, T., 2006, Dealing with Complexity in Design: A Knowledge Point of View, R. Design Methods for Practice, pp. 137-146.

[2] Goel, V., Pirolli, P., 1992, The structure of design problem spaces, Cognitive Science, 16(3), 395-429.

[3] Wits, Wessel W., Vaneker, Tom H.J., Mannak, Jan H., Houten, Fred J.A.M. van, 2007, Domain integration and cost reduction in electronic product design: a case study, Proceedings of 40th CIRP International Manufacturing Systems Seminar.

[4] Jauregui-Becker, JM., Tragter, H., Houten, FJAM. van, 2008, Structuring and modeling routine design problems for computational synthesis development, Proceedings of CIRP Design Seminar 2008. Abstract accepted.

[5] Renz J., Nebel B., 1998, Spatial Reasoning with Topological Information, Lecture Notes in Computer Science, 1404, 351.

[6] Wits, W., Legtenberg, R., Mannak, J. and Zalk, B. van, 2006, Thermal Management through In-Board Heat Pipes Manufactured using Printed Circuit Board Multilayer Technology, Proceedings International Electronic Manufacturing Technology (IEMT), pp. 55-61.

[7] Whitley, D., 1998, A Genetic Algorithm Tutorial Darrell, Statistics and Computing, 4:65-85.

ACKNOWLEDGMENTS

The authors gratefully acknowledge the support of the Dutch Innovation Oriented Research Program 'Integrated Product Creation and Realization (IOP-IPCR)'. This work is also part of the PACMAN project, SenterNovem project TSIT3049. Both activities are founded by the Netherlands Ministry of Economic Affairs. The authors especially want to thank Thales Nederland and in particular Jan Mannak for their effort and energy, contribution to this research project.